CICS Transaction Server for z/OS
版本 4 发行版 1

因特网指南

S151-1156-00

CICS Transaction Server for z/OS
版本 4 发行版 1

因特网指南

S151-1156-00

注意！

在使用本资料及其支持的产品之前，请先阅读第 395 页的『声明』中的信息。

本版本适用于 CICS Transaction Server for z/OS V4R1（产品号 5655-S97）及所有后续发行版和修订版，直到在新版本中另行声明为止。

目录

前言

本书内容

本手册记录了供客户使用以编写程序来获取 V4R1 服务的编程接口。

本手册介绍了如何设置和管理 CICS® Web Support 以允许 CICS 区域充当 HTTP 服务器和 HTTP 客户机，还介绍了如何编写与 Web 客户机和服务器交互的 CICS 应用程序。

理解本书需要了解的内容

本书假定您担当熟悉 CICS 的系统管理员或系统或应用程序员。

CICS Transaction Server for z/OS V4R1 中的更改

有关本发行版中所做更改的信息，请参阅信息中心内的*新增功能*，或参阅以下出版物:

- *CICS Transaction Server for z/OS 新增功能*
- *CICS Transaction Server for z/OS Upgrading from CICS TS Version 3.2*
- *CICS Transaction Server for z/OS Upgrading from CICS TS Version 3.1*
- *CICS Transaction Server for z/OS Upgrading from CICS TS Version 2.3*

第 1 部分 概述: CICS 和 HTTP

因特网、HTTP、TCP/IP 和 CICS Web Support 概念的概述，以及有关 CICS Web Support 的结构和操作的信息。

第 1 章 将 CICS 连接到 Web

CICS 可以作为服务器与 Web 连接，从 Web 客户机接收请求；或者作为客户机对服务器发出请求。

由 CICS 应用程序提供服务的 Web 客户机请求

使用 CICS Web Support

CICS Web Support 允许 CICS 区域作为 HTTP 服务器。

- 使用 CICS 文档或静态文件，CICS Web Support 可以对 Web 客户机提供静态响应。
- 支持 Web 的用户应用程序可以接收和分析 HTTP 请求，并提供动态应用程序生成的响应。
- CICS Web Support 包括一系列 CICS 服务，这些服务支持 Web 客户机对不支持 Web 的应用程序的访问。Web 客户机可以发出请求以访问设计为与虚拟 3270 终端通信的 CICS 程序，以及设计为使用通信区域或通道从另一个 CICS 应用程序链接所至的 CICS 程序。
- CICS Web Support 还支持来自客户机的非 HTTP 请求。

使用 Web Service

Web Service 是一个软件系统，设计的目的是支持网络上协同工作的机器间的交互。它有一个用机器可处理的格式（明确地说是 Web Service 定义语言（WSDL））描述的接口。CICS Transaction Server V3 可以是 Web Service 的请求程序或提供程序。

CICS Web Services Guide 中描述了 Web Service。

使用 IBM® HTTP Server

IBM HTTP Server 通过外部 CICS 接口（EXCI）和 CICS 业务逻辑接口对 CICS 应用程序提供访问。

要获得更多信息，请参阅 第 391 页的附录 K, 『CICS Web 服务器插件』和 第 321 页的第 27 章, 『CICS 业务逻辑接口的介绍』。

使用 CICS Transaction Gateway

CICS Transaction Gateway 提供一组 Web 服务器设施，以由 Web 客户机访问 CICS 应用程序。它们包括用于编写特定于应用程序的服务器程序（servlet）和浏览器程序（applet）的 Java™ 类和 Java bean，以及 IBM 为公共功能提供的代码。有一些类用于访问传统和面向对象的 CICS 应用程序。Applet 和 servlet 使用 CICS 提供的类构造 ECI（外部调用接口）和 EPI（外部演示接口）请求。（请注意，CICS Transaction Gateway for z/OS® 支持 ECI，但是不支持 EPI。）要获取更多信息，请参阅 *CICS Transaction Gateway: z/OS Administration*。

要获取有关选择 Web 解决方案的指导，请参阅 IBM 红皮书出版物 *Revealed ! Architecting Web Access to CICS*, SG24-5466, 它可以从 http://www.redbooks.ibm.com/redbooks/pdfs/sg245466.pdf 获取。

访问 Web 的 CICS 应用程序

CICS Web Support 允许 CICS 区域作为 HTTP 客户机。CICS 中的用户应用程序可以启动对 HTTP 服务器的请求，并从 HTTP 服务器接收响应。CICS Web Support 处理响应用户应用程序中 EXEC CICS WEB 命令的消息。

第 2 章 因特网、HTTP 和 TCP/IP 概念

本部分说明超文本传输协议（HTTP）和传输控制协议/网际协议（TCP/IP）的关键元素。

TCP/IP 协议

TCP/IP 是用于在网络上连接计算机系统的一系列通信协议，它根据该系列中的两个协议命名：传输控制协议（TCP）和网际协议（IP）。超文本传输协议（HTTP）是 TCP/IP 系列的成员。

在许多情况下，TCP/IP 系列中的协议与开放式系统互连（OSI）模型的层相对应。表 1 按照 OSI 模型显示了 TCP/IP 系列的 HTTP 和底层的层。同时还显示系统网络体系结构（SNA）层，这些层与 OSI 层大致相同。

表 1. TCP/IP 协议系列层

层	OSI	SNA	TCP/IP
7	应用程序	应用程序	HTTP
6	演示	演示	（空）
5	会话	数据流	（空）
4	传输	传输	TCP
3	网络	路径控制	IP
2	数据链路	数据链路	子网
1	物理	物理	

网际协议（IP）

IP 是网络层协议，该协议提供 TCP 使用的无连接数据传输服务。一条链路接着一条链路地传输数据；呼叫期间从来不建立端到端连接。数据传输的单位是*数据报*。

传输控制协议（TCP）

TCP 是传输层协议，该协议提供可靠的、全双工的、面向连接的数据传输服务。大多数因特网应用程序使用 TCP。

超文本传输协议（HTTP）

HTTP 是应用层协议，该协议用于分布式、协作式以及超媒体信息系统。HTTP 是在 Web 客户机和 Web 服务器之间使用的协议。

许多 TCP/IP 实施提供 TCP 协议（也就是传输层）的应用程序编程接口。该接口通常称为*套接字接口*。用于 CICS 的 TCP/IP 套接字接口是 z/OS Communications Server IP CICS套接字接口。它随 z/OS Communications Server（而不是 CICS）一起提供，并且是 z/OS 的完整部分。它不是 CICS Web Support 的一部分，并且不使用 CICS SO 域。*z/OS Communications Server: IP CICS Sockets Guide*，SC31-8807，描述了 CICS 套接字接口。

IP 地址

TCP/IP 因特网上的每台服务器或客户机都由数字 IP（因特网协议）地址标识。IP 地址有两类: IPv4（IP V4）地址和 IPv6（IP V6）地址。

IP 地址由因特网编号分配机构（IANA）及其授权机构管理并分配给用户。因特网是一个网络集合；因此，因特网地址指定网络和单个主机。所指定的值因网络大小而异。

IPv6 地址

IPv6 地址是 128 位地址，通常以十六进制表示法表示:

```
十六进制表示法表示的 IP 地址: '00010022033344440000000abc0def0'x
Halfword 0:  0001 hexadecimal
Halfword 1:  0022 hexadecimal
Halfword 2:  0333 hexadecimal
Halfword 3:  4444 hexadecimal
Halfword 4:  0000 hexadecimal
Halfword 5:  0000 hexadecimal
Halfword 6:  abc0 hexadecimal
Halfword 7:  def0 hexadecimal
冒号十六进制表示法表示的 IP 地址: 1:22:333:4444::abc0:def0

十六进制表示法表示的 IP 地址: '00000000000000000000ffff01020304'x
Halfword 0:  0000 hexadecimal
Halfword 1:  0000 hexadecimal
Halfword 2:  0000 hexadecimal
Halfword 3:  0000 hexadecimal
Halfword 4:  0000 hexadecimal
Halfword 5:  ffff hexadecimal
Halfword 6:  0102 hexadecimal
Halfword 7:  0304 hexadecimal
冒号十六进制表示法表示的 IP 地址: ::ffff:1.2.3.4 或 ::ffff:0102:0304
```

该地址由 8 个半字字段组成。在地址输出中，用以下方式处理 0:

- 如果某字段包含前导 0，那么这些 0 将被忽略；例如 0001 将表示为 1
- 如果地址中一个或多个连续字段包含值 0000，那么将使用表示法 :: 表示这些字段。

  例如，0000000000000ffff 将表示为 ::ffff

  一个地址中只使用一次 :: 替换，以避免在计算替换了多少个字段时引起混乱。

IPv4 地址

IPv4 地址是 32 位地址，通常以点分十进制表示法表示:

```
IP address in hexadecimal notation :   '817EB263'x
                                    Byte 0:  81 hexadecimal = 129 decimal
                                    Byte 1:  7E hexadecimal = 126 decimal
                                    Byte 2:  B2 hexadecimal = 178 decimal
                                    Byte 3:  63 hexadecimal =  99 decimal
IP address in dotted decimal notation: 129.126.178.99
```

在本示例中，129.126 指定网络，178.99 指定该网络中的主机。

IP 地址格式

CICS 接受特殊格式的 IPv4 和 IPv6 地址，并进行处理。

IPv6 地址格式

CICS 仅接受下列格式的 IPv6 地址:

- 不含方括号或 /nn 表示法的本机 IPv6 冒号十六进制地址; 例如, ::a:b:c:d

位于下列位置的“IP Version 6 Addressing Architecture”（RFC 4291）更加详细地描述了 IPv6 地址的语法: http://www.ietf.org/rfc/rfc4291.txt。

IPv4 地址格式

CICS 接受下列格式的 IPv4 地址:

- 不含 /nn 表示法的本机 IPv4 点分十进制地址; 例如, 1.2.3.4
- 已迁移至 IPv6 格式的 IPv4 地址（从 IPv4 映射而来的 IPv6 地址）; 例如, ::ffff:1.2.3.4
  - CICS 在内部将地址转换为 0:0:0:0:0:ffff:0102:0304 的二进制等价值
- 与 IPv6 兼容的地址（与 IPv4 兼容的 IPv6 地址）; 例如, ::1.2.3.4
  - CICS 在内部将地址转换为 0:0:0:0:0:0:0102:0304 的二进制等价值

  其例外情况有:
  - CICS 不允许下列项:
    - 0.0.0.0
    - ::0.0.0.0
    - ::0

不论您为 IPv4 地址指定了何种格式，CICS 都会以本机 IPv4 点分十进制地址格式显示所有 IPv4 地址; 例如, 1.2.3.4

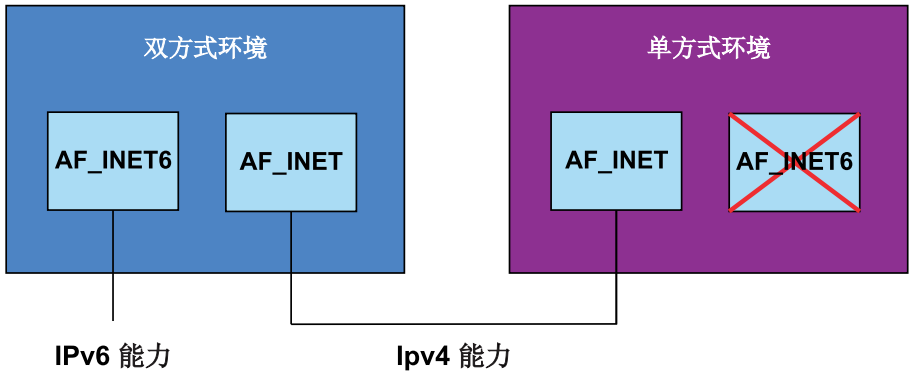
了解 IPv6 和 CICS

IPv6 协议专为替代 IPv4 而设计。要使用 IPv6 寻址，发送和接收环境必须支持双方式寻址（IPv4 和 IPv6），并且您的 CICS 区域必须运行在正确级别的 CICS 上。

IPv6 的基础结构需求

要支持 IPv4 和 IPv6 寻址，需要实施双方式 TCP/IP。单方式（IPv4）环境在 AF_INET 套接字和其他区域中的另一个 AF_INET 套接字之间建立连接时，会使用 AF_INET 地址系列。在 AF_INET 套接字上不支持 IPv6 地址; 这些地址需要发送和接收区域中的 AF_INET6 地址系列和 AF_INET6 套接字才能建立连接。双方式环境提供 AF_INET 和 AF_INET6 套接字。有关 AF_INET 和 AF_INET6 的更多信息，请参阅 *z/OS Communications Server IPv6 Network and Application Design Guide*。

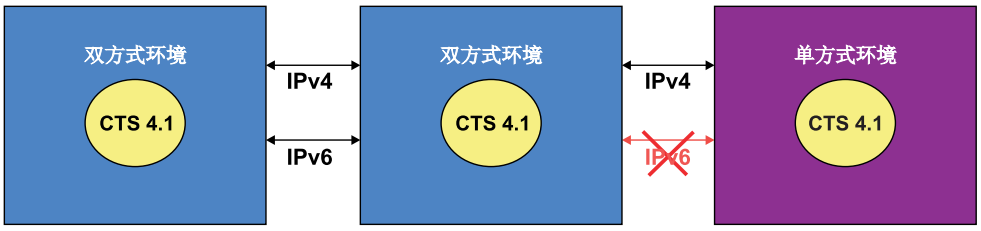
该图显示了单方式环境不具备 IPv6 能力，因为它没有 AF_INET6 套接字。



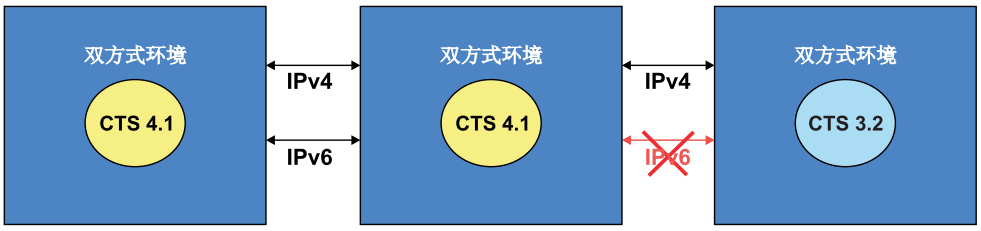
CICS 的 IPv6 需求

使用 IPv6 进行通信时，最少需要 CICS TS 4.1 级别。CICS TS 4.1 区域必须运行在双方式（IPv4 和 IPv6）环境下，而正与 CICS 通信的客户机或服务器也必须运行在双方式环境下。

该图显示了“CICS 到 CICS”通信，在这种通信方式中，两个双方式 CICS TS 4.1 环境可以使用 IPv4 或 IPv6 寻址来通信。也可以连接单方式 CICS TS 4.1 环境，但是该环境只可以使用 IPv4 进行通信。



该图显示了“CICS 到 CICS”通信，在这种通信方式中，两个双方式 CICS TS 4.1 环境可以使用 IPv4 或 IPv6 寻址来通信。也可以连接双方式 CICS TS 3.2 环境，但是该环境只可以使用 IPv4 进行通信。



主机名

因特网上的主机或 Web 站点由主机名识别，例如，`www.example.com`。主机名有时称为域名。主机名映射到 IP 地址，但是主机名和 IP 地址之间没有一对一关系。

当 Web 客户机发出到主机的 HTTP 请求时，使用主机名。发出请求的用户可能会指定服务器的 IP 地址，而不是主机名，但现在这在因特网上不常见。对于用户来说，主机名比数字 IP 地址更方便。公司、组织和个人常常选择其 Web 站点的主机名，用户能很容易地记住这些主机名。

现代 HTTP 实施中更重要的是，在 HTTP 请求中使用主机名意味着:

- 一个主机名中的服务可以由许多服务器提供，它们有不同的 IP 地址。
- 具有一个 IP 地址的一台服务器可以提供许多主机名中的服务。这称为**虚拟主管**。『虚拟主管』介绍了此过程。

主机名由称为 **DNS 服务器**或**域名服务器**的服务器映射到 IP 地址。DNS 代表域名服务。在大型网络中，许多 DNS 服务器可以相互协作，以提供主机名和 IP 地址之间的映射。

虚拟主管

HTTP 包括虚拟主管（virtual hosting）的概念，即单个 HTTP 服务器可代表具有同一 IP 地址的多台主机。

DNS 服务器可以向同一 IP 地址分配多个不同的主机名。当 HTTP 客户机向特定主机发出请求时，它使用 DNS 服务器查找与该主机名对应的 IP 地址，并向该 IP 地址发送请求。

在 HTTP/1.0 中，HTTP 消息中未出现主机名；这是因为解析 IP 地址后主机名已丢失。这意味着如果该 IP 地址所代表的服务器拥有多组资源，那么服务器很难区分哪些资源属于哪台主机。

然而，HTTP/1.1 请求中会提供主机名（通常在 Host 头中提供）。消息中存在主机名使 HTTP 服务器能将包含不同主机名的请求导向到每台主机的相应资源。HTTP 的该功能称为虚拟主管。CICS Web Support 通过使用 URIMAP 定义提供对虚拟主管的支持。

端口号

在一台服务器中，多个用户进程同时使用 TCP 是有可能的。为了识别与每个进程有关的数据，将使用端口号。端口号是 16 位的，虽然实际上通常只使用端口号的一小部分子集，但允许的最大端口号是 65535。

当客户机进程第一次与服务器进程联系时，它可能使用*熟知端口号*来启动通信。熟知端口号由 IANA（因特网编号分配机构）通过因特网分配给特殊服务。熟知端口号的范围是 0 到 1023。表 2 中显示了某些示例:

表 2. 服务及其熟知端口号

服务	熟知端口号
文件传输协议（FTP）	21
Telnet	23

表 2. 服务及其熟知端口号 (续)

服务	熟知端口号
超文本传输协议（HTTP）	80
具有安全套接字层（SSL）的 HTTP	443
CORBA 因特网 ORB 间协议（IIOP）	683
具有 SSL 的 CORBA IIOP	684

CICS 外部调用接口（ECI）有注册的端口号 1435。

仅使用熟知端口来建立客户机进程和服务器进程之间的通信。完成此操作后，服务器将分配*临时端口号*以供随后使用。临时端口号是进程开始通信时动态分配的唯一端口号。当通信结束时，释放这些端口号。

IANA 介质类型和字符集

因特网编号分配机构（IANA）是负责为因特网上使用的协议指定名称的国际性机构。

- IANA 媒体类型是在因特网上普遍发送的数据类型的名称。在该地址有关于它们的描述: http://www.iana.org/assignments/media-types/

  文本介质类型（如以 `text/`，开始的类型或包含 `+xml` 的类型）由 RFC 3023 标识，可以在 http://www.ietf.org/rfc/rfc3023.txt 上获得它们。
- IANA 字符集是字符集注册表的名称。在该地址有关于它们的描述: http://www.iana.org/assignments/character-sets

  CICS 不支持用于代码页转换的所有 IANA 字符集。CICS 支持的字符集在 第 333 页的附录 A, 『HTML 编码字符集』中描述。

URL 的组成部分

URL（统一资源定位符）是 URI（通用资源标识）的特定类型。URL 通常在因特网上查找现有资源。当 Web 客户机向服务器发出对资源的请求时，使用 URL。

URI 和 URL 的概念由因特网协会和 IETF（因特网工程任务组织）请求评论文档 RFC 2396 *统一资源标识（URI）: 一般语法*定义（http://www.ietf.org/rfc/rfc2396.txt）。简要地说，URI 是定义为识别资源的任何一个字符串。URL 定义为按资源的位置或用户访问它的方式，而不是按资源的名称或其他属性来识别资源的那些 URI。

IRI（国际资源标识）是一种较新的资源标识格式，允许使用适用于英语以外的其他本地语言的字符和格式。可使用 IRI 替换 URI 或 URL，请求和响应中所包含的应用程序为 IRI 提供支持。有关 IRI 的更多信息，请参阅第 221 页的『国际资源标识（IRI）』。

HTTP（HTTPS）的 URL 通常由三或四个组成部分组成:

1. **方案。**方案识别用于访问因特网上的资源的协议。它可以是 HTTP（不带 SSL）或 HTTPS（带 SSL）。
2. **主机。**主机名识别拥有资源的主机。例如，`www.example.com`。服务器在主机的名称中提供服务，但主机和服务器之间没有一对一映射。第 9 页的『主机名』说明了关于主机名的更多信息。

主机名也可以后跟**端口号**。第 9 页的『端口号』说明了关于这方面的更多内容。通常从 URL 省略服务的常用端口号。因为多数服务器将熟知端口号用于 HTTP 和 HTTPS，所以多数 HTTP URL 省略端口号。

3. **路径。**路径识别主机中 Web 客户机要访问的特定资源。例如，`/software/htp/cics/index.html`。
4. **查询字符串。**如果使用查询字符串，那么它跟随路径部分，并且提供一串字符串，资源使用这些字符串可以完成某些操作（例如，作为用于搜索的参数或用于处理的数据）。查询字符串通常是一串名称和值对，例如，`term=bluebird`。名称和值对之间用“与”符号（&）分开，例如，`term=bluebird&source=browser-search`。

URL 的方案和主机部分不定义为区分大小写，但是路径和查询字符串是区分大小写的。通常，整个 URL 指定为小写字母。

URL 的组成部分如下所示进行组合和定界:

```
scheme://host:port/path?query
```

- 方案后跟冒号和两个正斜杠。
- 如果指定端口号，那么主机名后面是号码，并用冒号分隔。
- 路径名以单正斜杠开始。
- 如果指定查询字符串，那么在它的前面加个问号。

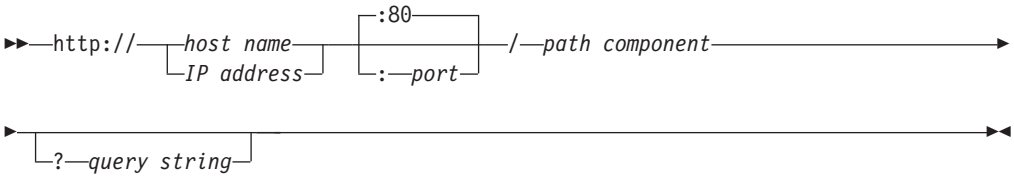


图 1. HTTP URL 语法

这是 HTTP URL 的示例:

```
http://www.example.com/software/index.html
```

如果指定了端口号，那么 URL 为:

```
http://www.example.com:1030/software/index.html
```

URL 的后面可以跟片段标识。URL 与片段标识之间使用的分隔符是字符 #。片段标识用于使 Web 浏览器指向它刚检索的项中的引用或函数。例如，如果 URL 标识 HTML 页面，那么可使用片段标识，以子节的标识来指示页面中的子节。对于这种情况，Web 浏览器通常向用户显示页面，以使用户可以看到子节。根据项的介质类型以及为该介质类型的片段标识所定义含义的不同，Web 浏览器为片段标识所采取的操作也会不同。

其他协议（如文件传输协议（FTP）或 Gopher）也使用 URL。这些协议使用的 URL 可能与 HTTP 使用的 URL 的语法不同。

HTTP 协议

HTTP 请求和响应的正确格式取决于由客户机和服务器使用的 HTTP 协议（或 HTTP 规范）版本。

通常在因特网上使用的 HTTP 协议版本（或“HTTP 版本”）是 HTTP/1.0（这是包含较少功能的早期协议）和 HTTP/1.1（这是包含较多功能的后期协议）。客户机和服务器可能使用不同版本的 HTTP 协议。客户机和服务器都必须在其消息的第一行声明其请求或响应的 HTTP 版本。

因特网协会和 IETF（因特网工程任务组织）请求评论文档（称为 RFC）为 HTTP 协议提供正式定义。这些文档为:

HTTP/1.0

RFC 1945, *超文本传输协议 - HTTP/1.0*, 可从 http://www.ietf.org/rfc/rfc1945.txt 获取

HTTP/1.1

RFC 2616, *超文本传输协议 - HTTP/1.1*, 可从 http://www.ietf.org/rfc/rfc2616.txt 获取

RFC 声明客户机和服务器为了以下目的应该执行的操作: 以适当的方式为每个版本的 HTTP 协议交换请求和响应。这些操作描述为“需求”。满足其 HTTP 协议版本需求的客户机或服务器被说成是“遵从”HTTP 规范。

HTTP 请求

HTTP 请求是客户机对位于服务器上的命名主机发出的。请求的目的是访问服务器上的资源。

要发出请求，客户机将使用 URL（统一资源定位符，包含访问资源所需的信息）的组成部分。第 10 页的『URL 的组成部分』介绍了 URL。

正确组成的 HTTP 请求包含以下元素:

1. 一个请求行。
2. 一系列 HTTP 头或头字段。
3. 消息体（需要的话）。

每个 HTTP 头后跟回车符换行（CRLF）。在 HTTP 头的最后，使用另一个 CRLF（空一行），然后开始是消息体。

请求行

请求行是请求消息中的第一行。它至少由三个项组成:

1. 一个**方法**。该方法是一个词的命令，告诉服务器它应该对资源采取什么操作。例如，可以要求服务器将资源发送到客户机。
2. 请求 URL 的路径部分。该路径识别服务器上的资源。
3. HTTP 版本号，显示客户机已尝试使消息遵从的 HTTP 规范。

请求行的一个示例为:

```
GET /software/htp/cics/index.html HTTP/1.1
```

在本示例中:

- 方法是 GET
- 路径是 /software/htp/cics/index.html
- HTTP 版本是 HTTP/1.1

请求行可能包含某些其他项:

- 一个查询字符串。它提供了资源可以为了某些目的而使用的一串信息。它跟随路径，前面有一个问号。
- URL 的方案和主机部分（除了路径）。资源位置以此方式指定时，它称为**绝对 URI** 格式。对于 HTTP/1.1，当请求将通过代理服务器时，使用此格式。同样对于 HTTP/1.1，如果 URL 的主机部分未包含在请求行中，那么它必须包含在 Host 头的消息中。

HTTP 头

在消息上写 HTTP 头，以对接收方提供有关消息、发送方以及发送方想要与接收方通信的方式的信息。每个 HTTP 头由名称和值组成。HTTP 协议规范定义 HTTP 头的标准集合，并描述如何正确使用它们。HTTP 消息还可以包含扩展头，它们不是 HTTP/1.1 或 HTTP/1.0 规范的一部分。

客户机请求的 HTTP 头包含一些信息，服务器可以使用这些信息决定如何响应请求。例如，以下系列头可用于指定最终用户只想读取法语或德语的已请求文档，且仅当自从客户机上一次获取该文档的日期和时间以来它已更改的情况下，才发送它:

```
Accept-Language: fr, de
If-Modified-Since: Fri, 10 Dec 2004 11:22:13 GMT
```

请求消息中的 HTTP 头系列后放了一个空行（即，单个 CRLF），以将头与消息体分隔开。

消息体

任何 HTTP 消息的主体内容可以称为消息体或**实体主体**。从技术上说，实体主体是消息的实际内容。消息体包含实体主体，它可以处于其原始状态，或可以用某种方式编码以进行传输，例如，通过分成块（分块的传输编码）。为了方便起见，请求的消息体可以称为请求主体。

消息体适合某些请求方法而不适合其他请求方法。例如，具有 POST 方法的请求（它将输入数据发送到服务器）具有包含数据的消息体。具有 GET 方法的请求（它要求服务器发送资源）没有消息体。

HTTP 响应

服务器向客户机发出 HTTP 响应。响应的目的是对客户机提供它请求的资源，或通知客户机已执行它请求的操作；否则将通知客户机处理其请求时出现错误。

HTTP 响应包含:

1. 一个状态行。
2. 一系列 HTTP 头或头字段。
3. 一个消息体（通常需要）。

在一个请求消息中，每个 HTTP 头后跟回车符换行（CRLF）。在 HTTP 头的最后，使用另一个 CRLF（空一行），然后开始是消息体。

状态行

状态行是响应消息中的第一行。它由三个项组成:

1. HTTP 版本号，显示服务器已尝试使消息遵从的 HTTP 规范。
2. **状态码**，它是表明请求结果的三位数的号码。
3. **原因短语**，也称为状态文本，它是人类可以阅读的文本，总结了状态码的含义。

响应行的一个示例为:

```
HTTP/1.1 200 OK
```

在本示例中:

- HTTP 版本是 `HTTP/1.1`
- 状态码是 `200`
- 原因短语是 `OK`

第 15 页的『状态码和原因短语』说明有关这些状态行元素的更多内容。

HTTP 头

服务器的响应的 HTTP 头包含一些信息，客户机可以使用这些信息找到有关响应和发送响应的服务器的更多内容。该信息可以帮助客户机显示对用户的响应，存储（或高速缓存）响应以供将来使用，以及帮助客户机现在或将来对服务器发出更多请求。例如，以下系列头告诉客户机发送响应的时间，该响应由 CICS 发送，并且它是 JPEG 图像:

```
Date: Thu, 09 Dec 2004 12:07:48 GMT
Server: IBM_CICS_Transaction_Server/3.1.0(zOS)
Content-type: image/jpg
```

在不成功请求的情况下，头可以用于告诉客户机它必须执行什么操作以成功完成其请求。

响应消息中的 HTTP 头系列后放了一个空行（即，单个 CRLF），以将头与消息体分隔开。

消息体

为了方便起见，响应的消息体可以称为响应主体。

消息体用于大多数响应。异常发生在服务器对使用 HEAD 方法（它请求头，但不请求响应主体）的客户机请求做出响应的情况下，以及服务器正在使用特定状态码的情况下。

对于对成功请求的响应，消息体包含客户机请求的资源，或有关客户机请求的操作状态的某些信息。对于对不成功请求的响应，消息体可能提供有关错误原因的进一步信息，或有关客户机为了成功完成请求需要执行的某些操作的进一步信息。

状态码和原因短语

在发送到客户机的 HTTP 响应中，三位数的状态码伴有总结代码含义的原因短语（也称为状态文本）在一起。这些项与响应的 HTTP 版本一起放在响应的第一行中，因此这一行称为状态行。

状态码按数字范围分类，这些代码的每个类都有相同的基本含义。

- 范围 100-199 分类为“信息”。
- 200-299 为“成功”。
- 300-399 为“重定向”。
- 400-499 为“客户机错误”。
- 500-599 为“服务器错误”。

将范围描述为整体时，它可能命名为“1xx”、“2xx”等。HTTP 协议规范不定义任何 600 或更大的状态码。

每个范围中只有少数状态码由 HTTP/1.0 和 HTTP/1.1 规范定义。HTTP/1.1 规范包含的状态码比 HTTP/1.0 规范的状态码多。

建议使用 HTTP 规范中定义的原因短语（例如，“未找到”或“错误请求”），但这些短语是可选的。HTTP/1.1 规范声明：每个状态码的原因短语可以由本地同等短语替换。

200（确定）状态码用于常规响应，该响应提供 Web 客户机请求的完整资源。大多数其他状态码在以下情形下使用：存在阻止请求执行的错误，或者客户机需要执行某些其他操作以成功完成其请求（例如，跟随重定向 URL，或修改请求以使服务器可以接受它）。

响应的 HTTP 头和/或响应主体可以为客户机提供进一步指示信息和信息。HTTP 规范包括响应内容的需求和建议以及每个状态码。这些需求指定：

- 必须或可以在响应上使用的任何 HTTP 头。例如，如果使用状态码 405（不允许方法），那么必须使用 Allow 头以声明*允许*哪些方法。
- 是否应该使用响应主体。例如，不允许消息体具有状态码 204、205 和 304。
- 如果使用了响应主体，它可以提供什么信息。例如，重定向的消息体可以提供重定向 URL 的超链接。

要获取有关状态码的含义和正确使用的完整信息，应该查询您正在遵从的 HTTP 规范。请参阅 第 12 页的『HTTP 协议』以获取有关 HTTP 规范的更多信息。

转义和未转义数据

为了有助于传输的正确和 HTTP 请求的解释，对于 URL 中使用某些字符存在一些限制。传输请求后，这些字符必须转换为安全格式。

在 URI 或 URL 中，一个或多个 URI 或 URL 组成部分上下文中有特殊用途的字符称为**保留字符**。例如，字符 /、?、& 和 : 用作各种组件的定界符。除了保留字符的特殊用途外，如果还出于任何其他原因使用这些字符，那么机器解释器可能误释 URI 或 URL。

而且，不允许或**排除**在 URI 或 URL 中的任何地方使用特定字符，这是因为它们是混淆机器或人类用户的潜在原因，或是因为已知它们会导致某些机器解释器的问题。例如，URL 中不允许空格字符。

因特网协会和 IETF（因特网工程任务组织）请求评论文档 RFC 2396 *统一资源标识（URI）：一般语法*列出了 URI 和 URL 中保留或排除的字符。RFC 2396 可以从 http://www.ietf.org/rfc/rfc2396.txt 获取。

如果除了其特殊用途外，还出于任何其他原因在 URL 中需要保留字符，或者如果 URL 中需要排除的字符，那么当包含 URL 组成部分的请求发送到服务器时，必须对这些字符进行**转义**。这包括以查询字符串发送的数据中的字符。

通过用格式为 %xx 的三个字符的字符串替换字符，可以对它们进行转义，其中 xx 是保留字符的 ASCII 十六进制表示法。例如，%2F 替代 / 字符。作为特例，空格字符可以由 + 替代。由于这种格式，转义也称为**百分比编码**。

请求到达服务器时，服务器可以**取消转义**转义字符，即，将它们从转义序列转换回原始字符。为了避免语法分析应用程序误释保留或排除字符的风险，取消转义应该仅当已语法分析 URL 和查询字符串中的信息后发生。

请求中的表单数据（不管它存在于 URL 中还是消息体中）通常与转义的特殊字符一起发送，这是因为表单的缺省编码（**application/x-www-form-urlencoded**）转义保留或禁止使用的字符。『HTML 表单』提供了更多信息。

HTML 表单

在 HTML 中，表单是由 <form> 标记定界的区域，包含文本输入框、按钮、复选框和图形用户界面的其他功能部件。Web 应用程序使用表单以允许最终用户提供要发送到服务器的数据。

在表单中，用户可与之交互作用以提供数据的元素称为**表单字段**。HTML 中对每个表单字段赋予一个名称，这会将它识别到服务器应用程序，但是对于用户不可视。

虽然一个表单的各种元素以不同样式显示给用户，但是它们都以相同的方式将信息传输给服务器应用程序：作为一系列由 & 字符分开的名称和值对。每个名称是表单字段的名称，而值是用户的操作产生的数据。例如，如果表单包含两个文本输入框以供用户输入他们的姓和名，那么数据可能看上去与以下内容类似：

```
firstname=Maria&lastname=Smith
```

根据在 <form> 标记中指定哪种方法（GET 或 POST），表单数据以两种方式之一传输到服务器：

- 当方法是 GET 时，表单数据在 URL 的查询字符串中传输。
- 当方法是 POST 时，在消息体中传输表单数据。

编码表单数据所需的字符集由 CHARACTERSET 选项指定，它应该与相应的 HTML 表单确定的表单编码匹配（有关更多信息，请参阅第 17 页的『如何确定客户机编码』）。

表单数据通常用转义的特殊字符传输。第 15 页的『转义和未转义数据』说明了转义的用途。

如果用 GET 方法定义表单，那么因为数据作为 URL 中的查询字符串发送，所以必须总是转义保留或排除的字符。

如果用 POST 方法定义表单，那么数据在消息体中发送。然而，如 HTML 2.0 规范中定义，所有表单的缺省编码类型是 **application/x-www-form-urlencoded**。（请参阅 http://www.w3.org/MarkUp/html-spec/html-spec_8.html#SEC8.2.1）当此编码用于具有 POST 方法的表单时，虽然数据在消息体中发送，但是也将转义保留或排除的字符，就如它们在 URL 中的情形一样。

如果为表单指定了备用编码类型 **multipart/form-data**（通过在 HTML `<form>` 标记上使用 ENCTYPE 属性来完成），那么应该转义字段名中的非 ASCII 字符，但是字段值中的非 ASCII 字符不需要转义。该数据也出现在消息体的一系列独立部分中。较旧的应用程序可能不支持此编码。CICS 支持它。**multipart/form-data** 编码在因特网协会和 IETF 请求评论文档 RFC 1867 *HTML 中基于表单的文件上载*（http://www.ietf.org/rfc/rfc1867.txt）中进行了描述。

如何确定客户机编码

HTTP 客户机用于表单数据（用于 GET 和 POST 方法）的字符编码（**charset** 参数）由 HTML 表单中的信息决定。

HTTP 客户机通常使用 HTML 表单所用的字符编码来提交表单数据，该字符编码可以由 Content-Type 头上的 **charset** 参数确定，也可以由嵌入在 HTML 中的等效 `META` 标记指定，例如:

```
<META http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

HTML FORM 元素上的 accept-charset 属性还可以用来指定其他可接受的字符编码。如果未指定代码页，那么 CICS 会从 **charset** 参数获取该信息。HTML 表单字符编码通常是 ISO-8859-1（CCSID 819）或 UTF-8（CCSID 1208），但不限于这些值。

字符编码信息通常不出现在已提交的表单请求中，因此，如果没有使用针对因特网的缺省字符集（ISO-8859-1），那么读取表单的应用程序必须使用 CHARACTERSET 关键字指定编码。如果忽略 CHARACTERSET，但 HTTP 客户机在 Content-Type 头中提供了一个字符集值（这不是 HTML 表单提交的标准做法），那么使用该字符集值，否则 CICS 认为使用 ISO-8859-1。

分块的传输编码

分块的传输编码（也称为分块）将消息的主体作为一系列块传送，每个块都带有各自的块大小头。消息的结束由长度为零且带一空行的块来表明。

这个已定义过程意味着: 可以在合适的段中发送应用程序生成的实体主体或大的实体主体。当收到长度为零的块时，客户机或服务器便知道分块的消息已完成。

分块消息的主体后可以跟可选尾部，该尾部包含称为尾部头的补充 HTTP 头。接受尾部不需要客户机和服务器，因此补充 HTTP 头只应该提供非基本的信息，除非服务器知道客户机将接受尾部。

要使用分块的传输编码，客户机和服务器都必须使用 HTTP/1.1。分块的消息无法发送到 HTTP/1.0 客户机。HTTP/1.1 规范（RFC 2616）中定义了应用到分块的传输编码的需求和对尾部的使用。

管道传送

管道传送是指客户机将多个 HTTP 请求发送到服务器而不等待响应。随后，响应必须以接收请求的相同顺序从服务器返回。

请求方应确保请求是幂等的。幂等性是指: 当重复所有或部分请求系列时，总是获得相同的结果。如果与服务器的连接出错，这将确保客户机可以重试请求系列，即使它并不知道服务器是否已实施所有或部分请求，或根本未实施请求。

HTTP/1.1 规范（RFC 2616）定义有关 HTTP 请求的幂等性的规则。请参阅 第 12 页的『HTTP 协议』以获取有关 HTTP 规范的更多信息。简单地说，大多数请求方法在单独使用时都是幂等的，这是因为每次使用方法时都会获得相同结果。（POST 方法除外，因为它更改服务器上的资源。）然而，在管道传送期间发送一系列请求时，该序列可能不是幂等的，特别是在资源有所更改的情况下。

如果计划对请求进行管道传送，请检查是否可以随时终止请求序列，然后重新从头开始，而不会导致逻辑错误。如果不是这种情况，那么分别发出请求并在每次请求后等待确认。

相关任务

第 138 页的『样本程序: 将请求通过管道传送给 HTTP 服务器』
样本程序 DFH$WBPA（汇编程序）、DFH$WBPC（C）和 DFH0WBPO （COBOL）演示了 CICS 如何通过管道将客户机请求传送到 HTTP 服务器。

持续连接

持续连接是 Web 客户机和服务器之间可重复用于交换多个请求和响应的连接。

在 HTTP/1.0 中，服务器的缺省操作是当从 Web 客户机收到请求并发送响应后，即关闭连接。如果 Web 客户机希望服务器保持连接打开，那么必须在请求中发送 Connection: Keep-Alive 头。

对于 HTTP/1.1，持续连接是缺省值。在 Web 客户机和服务器之间建立连接后，缺省情况下，服务器应该保持连接打开。仅当 Web 客户机通过发送 Connection: close 头请求了关闭，或达到服务器的超时设置，或服务器遇到错误时，才应该关闭连接。

因为不必为每个请求建立新的连接，所以持续连接提高了网络性能。与使用现有连接来发出请求相比，建立新的连接会消耗大量网络资源。

HTTP 基本认证

HTTP 基本认证是简单的提问和响应机制，服务器可以通过它从客户机请求认证信息（用户标识和密码）。客户机在 Authorization 头中向服务器传递认证信息。认证信息是基本 64 位编码的。

对于客户机发出的请求，如果服务器需要认证信息，那么服务器将发送具有以下各项的 HTTP 响应：401 状态码、表明认证错误的原因短语和 WWW-Authenticate 头。大多数 Web 客户机通过从最终用户请求用户标识和密码来处理该响应。

HTTP 基本认证的 WWW-Authenticate 认证头的格式是：

```
WWW-Authenticate: Basic realm="Our Site"
```

WWW-Authenticate 头包含域属性，该属性标识所请求的认证信息（即用户标识和密码）针对的资源集。当 Web 客户机请求用户标识和密码时，它们会向最终用户显示该字符串。每个域可能需要不同的认证信息。Web 客户机可能会为每个域存储认证信息，以便最终用户不需要为每个请求重新输入信息。

当 Web 客户机获得了用户标识和密码时，它会重新发送具有 Authorization 头的原始请求。或者，客户机在发出其原始请求时可以发送 Authorization 头，服务器可以接受该头，以避免提示和响应过程。

Authorization 头的格式是：

```
Authorization: Basic userid:password
```

用户标识和密码是使用基本 64 位编码方案进行编码的。

RFC 2617，*HTTP Authentication: Basic and Digest Access Authentication*（可从 http://www.ietf.org/rfc/rfc2617.txt 获得）具有更多有关基本认证的详细信息。

注：仅当 Web 客户机和服务器之间的连接安全时，HTTP 基本认证方案才能被认为是安全的认证方法。如果该连接不安全，那么该方案不提供足够的安全性以阻止未授权的用户发现和使用服务器的认证信息。如果存在密码被拦截的可能性，那么基本认证应该与 SSL 结合使用，以便 SSL 加密用于保护用户标识和密码信息。

第 3 章 CICS Web Support 概念和结构

CICS Web Support 是允许 CICS 区域作为 HTTP 服务器和作为 HTTP 客户机的 CICS 服务集合。

作为 HTTP 服务器的 CICS

当 CICS 是 HTTP 服务器时，Web 客户机可以将 HTTP 请求发送到 CICS 并接收响应。该响应可以是 CICS 从文档模块或静态文件创建的静态响应，或者是由用户应用程序动态创建的应用程序生成的响应。

作为 HTTP 服务器的 CICS 的操作由以下各项控制:

1. 包括 TCPIPSERVICE 定义和 URIMAP 定义的系统初始化参数和资源定义，它们用于配置 CICS Web Support 并指示 CICS 如何处理请求和响应。
2. 可用于分析和处理 HTTP 请求和响应的 CICS 实用程序。
3. 用于接收 HTTP 请求并提供 HTTP 响应的资料的用户编写的应用程序。它们可以是为了与 CICS Web Support 协同使用而设计的支持 Web 的应用程序，或者是原先设计时未考虑到要与 CICS Web Support 协同使用而不支持 Web 的 CICS 应用程序。

作为 HTTP 服务器的 CICS Web Support 的行为有条件地遵从 HTTP/1.1 规范，如 RFC 2616 中所描述。请参阅 第 12 页的『HTTP 协议』以获取有关 HTTP 规范的更多信息。

作为 HTTP 客户机的 CICS

当 CICS 是 HTTP 客户机时，CICS 中的用户应用程序可以启动对 HTTP 服务器的请求，并从 HTTP 服务器接收响应。

作为 HTTP 客户机的 CICS 的操作由用户编写的应用程序控制。EXEC CICS WEB 应用程序编程接口包括一些命令，应用程序可以使用这些命令构造和启动来自 CICS 的 HTTP 请求，并接收服务器发送的响应。URIMAP 资源定义可用于提供诸如 URL 或客户机证书标签的信息。

CICS Web Support 和非 HTTP 消息

CICS Web Support 还支持来自客户机的非 HTTP 请求。您可以使用 CICS Web Support 的多种组件（包括 TCPIPSERVICE 定义、CICS 实用程序和用户编写的程序），为您所定义的任何请求格式提供请求处理。 CICS Web Support 处理的非 HTTP 消息使用 TCPIPSERVICE 资源定义中的特殊协议（USER 协议），以便不对它们执行 CICS 为 HTTP 消息执行的检查。

在 CICS Transaction Server for z/OS V4R1 中，该设施主要用于为用户编写的、使用非标准请求格式的客户机发出的请求提供支持。对请求执行的处理由用户定义。该设施不会为任何正式定义的用于客户机/服务器通信的协议提供特定支持。

CICS Web Support 为非 HTTP 消息提供的支持不同于 CICS 的 TCP/IP 套接字接口。与 z/OS Communications Server 一起提供的 IP CICS 套接字接口有一个应用程序编程接口，该接口允许客户机经由 TCP/IP 直接与 CICS 应用程序通信。CICS Web Sup-

port 不涉及该过程。*z/OS Communications Server: IP CICS Sockets Guide*，SC31-8807，描述了 CICS 套接字接口。

CICS Web Support 的组件

CICS Web Support 包括用于所有 CICS Web Support 任务的某些基本组件，以及您为个别 CICS Web Support 任务选择并配置的某些特定于任务的组件。

基本组件

- **TCP/IP 支持**在 CICS 中由 CICS SO（套接字）域提供，并具有 z/OS 提供的网络服务（z/OS Communications Server 以及对 DNS 服务器的访问）。
- **z/OS UNIX® 系统服务**用作 TCP/IP 支持的一部分，并且 CICS 区域需要访问这些服务。
- **安全套接字层（SSL）支持**用于为 CICS Web Support 实施提供安全性。CICS 支持安全套接字层（SSL）3.0 协议，以及传输层安全性（TLS）1.0 协议。（不支持 SSL 2.0）。请注意，CICS 文档中使用术语 SSL 时，它通常指 SSL 和 TLS。*CICS RACF Security Guide* 提供了更多关于 SSL 和 TLS 的信息。
- **DOCCODEPAGE 系统初始化参数**指定 CICS 文档模板支持使用的缺省主机代码页。
- **LOCALCCSID 系统初始化参数**指定本地 CICS 区域的编码字符集标识（它是被 CICS 当作应用程序缺省值的代码页）。
- **TCPIP 系统初始化参数**在启动时激活 CICS TCP/IP 服务。
- **WEBDELAY 系统初始化参数**仅在涉及 Web 3270 网桥设施的情况下才为不活动的 CICS Web 任务定义超时周期。其他 CICS Web 任务的超时由相关事务的 RTIMOUT 值处理，或（对于作为 HTTP 服务器的 CICS）由 TCPIPSERVICE 定义的 SOCKETCLOSE 属性处理。
- **套接字侦听器任务（CSOL）**通过连接 Web 连接任务检测入站 TCP/IP 连接请求，并调用 CICS Web Support。
- **Web 连接任务（CWXN、CWXU 或别名）**从 Web 客户机接收数据并处理请求的初始处理，这包括 URIMAP 匹配、HTTP 头的代码页转换、请求的分析和消息体的代码页转换。任务还预处理从 Web 客户机接收的分块的和管道化消息。如果交付了（使用 URIMAP 定义）静态响应，那么 Web 连接任务也处理该处理。

资源定义

- **TCPIPSERVICE 资源定义**用于定义为作为 HTTP 服务器的 CICS 使用的每个端口，包括该端口上连接的安全选项以及入站请求的超时和最大大小限制。它们不用于作为 HTTP 客户机的 CICS。

  注: TCPIPSERVICE 资源定义仅供 CICS 提供的 TCP/IP 服务使用，而与 z/OS 通信服务器 IP CICS 套接字接口无关。CICS 的 TCP/IP 套接字接口随 z/OS Communications Server 一起提供，它是 z/OS 的主要部分，且不使用 CICS SO 域。
- **URIMAP 资源定义**匹配来自 Web 客户机的请求或到 HTTP 服务器的请求的 URL，并对 CICS 提供有关如何处理请求的信息。URIMAP 定义合并并且可以替换 CICS Web Support 处理函数，这些函数已由与 TCPIPSERVICE 定义关联的分析器程序在 CICS Transaction Server for z/OS V3R1 之前提供。URIMAP 定义也可用于传递对 Web 客户机请求的静态响应，而不涉及应用程序。

- **TRANSACTION 资源定义**用于为 HTTP 请求处理定义别名事务。CICS 为缺省别名事务 CWBA 提供资源定义。当 Web 连接任务已完成请求的初始处理时，如果要生成应用程序生成的响应，那么别名事务将处理剩余处理阶段。这些阶段包括接收请求、执行应用程序的业务逻辑、HTTP 响应的构造和 HTTP 响应的代码页转换。

用户应用程序

- **支持 Web 的应用程序**可以为 CICS Web Support 设计，它使用 EXEC CICS WEB 和 EXEC CICS DOCUMENT 应用程序编程接口。对于作为 HTTP 服务器的 CICS，这些程序可以接收和分析 HTTP 请求，并向 Web 客户机提供应用程序生成的响应。对于作为 HTTP 客户机的 CICS，CICS 中的用户应用程序可以启动对服务器的 HTTP 请求，并从服务器接收响应。
- **通信区域应用程序**设计为使用通信区域接口从另一个程序链接的程序，可以将 CICS Web Support 与转换器程序协同使用来访问，以将它们的输出转换成 HTML 供传输到 Web 客户机。或者，您可以写链接到通信区域应用程序的支持 Web 的应用程序，并使用它的输出来提供 HTTP 响应。
- **3270 显示应用程序**，为了与 3270 终端通信而设计的程序，可以使用 Web 终端转换应用程序来访问。Web 终端转换应用程序创建的 HTML 输出可显示在 Web 浏览器中。

编程接口

- **EXEC CICS WEB** 应用程序编程接口解释和构造 HTTP 请求和 HTTP 响应。某些命令用于作为 HTTP 服务器的 CICS，某些命令用于作为 HTTP 客户机的 CICS，而某些命令对于两种 CICS Web Support 格式都可用。
- **EXEC CICS DOCUMENT** 应用程序编程接口构造 CICS 文档以提供从 CICS 发出的响应或请求的主体。

CICS Web Support 实用程序

- **分析器程序**与 TCPIPSERVICE 定义关联。它们用于在这些情况下解释 HTTP 请求：如果 URIMAP 定义指定使用分析器程序，或如果不存在 URIMAP 定义。CICS 提供缺省分析器程序 DFHWBAAX（该程序提供基本错误处理），以及样本分析器程序 DFHWBADX（该程序支持使用 URL 格式的请求，CICS TS 3.1 前版本的 CICS Web Support 使用该格式。）这两种分析器都可以用作您自己的分析器程序的基础。
- **转换器程序**可用于译码 HTTP 请求并将输入构造到用户应用程序。支持 Web 的应用程序通常不需要转换器程序，但那些不提供 CICS Web Support 的不支持 Web 的应用程序可能需要它们。CICS 不提供转换器程序。您可以编写许多转换器程序，并选择您的 CICS 区域中的任何转换器程序来处理请求。
- **Web 出错程序**在 CICS Web Support 过程中发生请求错误或异常终止时，对 Web 客户机提供出错响应。CICS 提供 Web 出错程序 DFHWBEP（该程序在大多数错误情形中使用），以及 Web 出错应用程序 DFHWBERX（当 URIMAP 匹配失败时，该程序与缺省分析器 DFHWBAAX 协同使用（且可以为其他情形指定））。Web 出错程序是用户可替换的程序，可以修改它们以定制或更改在每种错误情形下发送给 Web 客户机的出错响应。
- **Web 终端转换应用程序 DFHWBTTA**（以及它用于备用处理的别名 DFHWBTTB 和 DFHWBTTC）可用于从为了与 3270 终端通信而设计的程序创建 HTML 输出。该程序使用 CICS 3270 网桥机制。使用 BMS 和应用程序和不使用 BMS 的应用程序都是受支持的。使用该功能不需要更改应用程序。

- **密码到期管理程序 DFHWBPW** 为连接指定了基本认证且用户密码到期后使用。该程序指导用户完成设置新密码的整个过程。您可以通过 DFHWBPW 定制或替换向用户呈现的 Web 页面。

文档构造设施

- **z/OS UNIX 系统服务文件**可以作为对来自 Web 客户机的 HTTP 请求的响应主体。
- **文档模板支持**允许消息体从 HTML 片段构建，这些片段已在脱机状态下准备。
- **BMS 宏**从 BMS 映射集构造 HTML 文档模板。

代码页转换

CICS 提供一些设施，以将 HTTP 消息转换为适合用户应用程序的代码页或适合在因特网上使用的代码页。CICS 使用 z/OS 转换服务处理代码页转换。

在先前 CICS 发行版中必需的代码页转换表（DFHCNV）通常对 CICS Transaction Server for z/OS V4R1 中的 CICS Web Support 不是必需的。异常情况是如果您要在先前 CICS 发行版中使用您编码的分析器程序引用 DFHCNV。在这种情况下，您必须继续提供代码页转换表或对分析器程序进行更新。第 51 页的『升级代码页转换表（DFHCNV）中的条目』具有有关该内容的更多信息。

CICS Web Support 的任务结构

当 CICS Web Support 在 CICS 区域中活动时，对于作为 HTTP 服务器的 CICS，使用不同的任务来侦听入站连接请求；接收来自套接字的数据并执行初始处理；以及包括由应用程序执行的、有关请求的工作。对于作为 HTTP 客户机的 CICS，只适用一个任务，即应用程序发出 HTTP 请求的任务。

套接字侦听器任务（CSOL）

这是一个长期运行的 CICS 任务。CICS 系统中有一个套接字侦听器任务的实例。

该任务检测所有 CICS 定义的端口上的入站 TCP/IP 连接请求，并且调用与端口相关联的 CICS 服务。当该端口打算用于 CICS Web Support（即，指定 HTTP 或 USER 作为协议）时，Web 连接任务会定义为该端口的 TCPIPSERVICE 资源定义中的事务，因此侦听器会连接该任务。

Web 连接任务（CWXN、CWXU 或别名）

当端口的 TCPIPSERVICE 定义具有协议 HTTP 时，Web 连接任务的缺省事务标识是 CWXN。当协议为 USER 时，缺省值为 CWXU。可改为使用别名，但事务总是执行程序 DFHWBXN。

当 Web 连接任务由套接字侦听器任务调用时，它要做的第一件事是发出 SOCKET RECEIVE 请求以从 Web 客户机接收数据。当已接收到某些数据时，Web 连接任务来应对 Web 客户机请求的初始处理。

- 对于 HTTP 请求（在 HTTP 协议上），初始处理包括 URIMAP 匹配、HTTP 头的代码页转换、请求的分析和消息体的代码页转换。该任务还预处理从 Web 客户机接收的分块的消息和管道消息。如果使用分析器程序，那么该事务包含它。
- 对于非 HTTP 请求（在 USER 协议上），不发生初始处理。

如果静态响应交付到 HTTP 请求（使用 URIMAP 定义），那么 Web 连接任务也处理该处理。如果需要应用程序生成的响应，那么 Web 连接任务连接一个别名事务。

来自 Web 客户机的每个 HTTP 请求都有一个 Web 连接任务的实例，它处于处理的初始阶段。CICS Transaction Server for z/OS V3R1 之前，如果 Web 客户机和 CICS 有持续连接，那么 CWXN 事务在持续连接的持续时间内会保留在系统中。现在，来自 Web 客户机的请求已传递到别名事务或已传递静态响应后，CWXN 事务会终止。套接字侦听器任务监控套接字，并在持续连接上为每个请求启动 CWXN 的新实例。当保留在系统中的 CWXN 事务无法连接别名事务来处理更多请求时，该行为（称为异步接收）避免在已达到最大任务指定数（MXT）的情况下可能出现死锁。

应用程序生成的响应的别名事务

当 Web 连接任务已完成请求的初始处理时，如果要生成应用程序生成的响应，Web 连接任务会连接为该请求的剩余处理阶段指定的别名事务。CICS 为缺省别名事务 CWBA 提供资源定义。在提供静态响应的情况下不使用别名事务。

别名事务处理应用程序生成的响应的处理阶段，包括接收请求、执行应用程序的业务逻辑、构造 HTTP 响应和 HTTP 响应的代码页转换。如果转换器程序用于处理请求，那么别名事务也处理它。每个处于这些处理阶段中的 HTTP 请求都有一个别名事务的实例。

作为 HTTP 客户机的 CICS

对于作为 HTTP 客户机的 CICS，发出 HTTP 客户机请求的应用程序引起的所有活动都由单个任务所包含。这包括应用程序的操作、CICS 发送请求和接收请求的操作和套接字活动。如果应用程序使用 EXEC CICS LINK 命令链接到其他程序，那么这些程序也包含在该任务中。该任务具有触发应用程序的事务标识。

该任务从应用程序的活动的开始到结束都保留在系统中。该任务可以涉及多个请求和响应，并且应用程序可以打开和保持到服务器的多个连接。当该任务结束时，会自动关闭所有打开的连接。

作为 HTTP 服务器的 CICS 的 HTTP 请求和响应处理

对作为 HTTP 服务器的 CICS 的 HTTP 请求由对 CICS 发出请求的 Web 客户机启动。CICS 对 Web 客户机发出的请求提供响应。响应可以从 URIMAP 资源定义指定的静态文档创建，或者由用户应用程序动态创建。

第 26 页的图 2 显示了 CICS Web Support 从 Web 客户机接收请求并提供响应时所执行的处理。

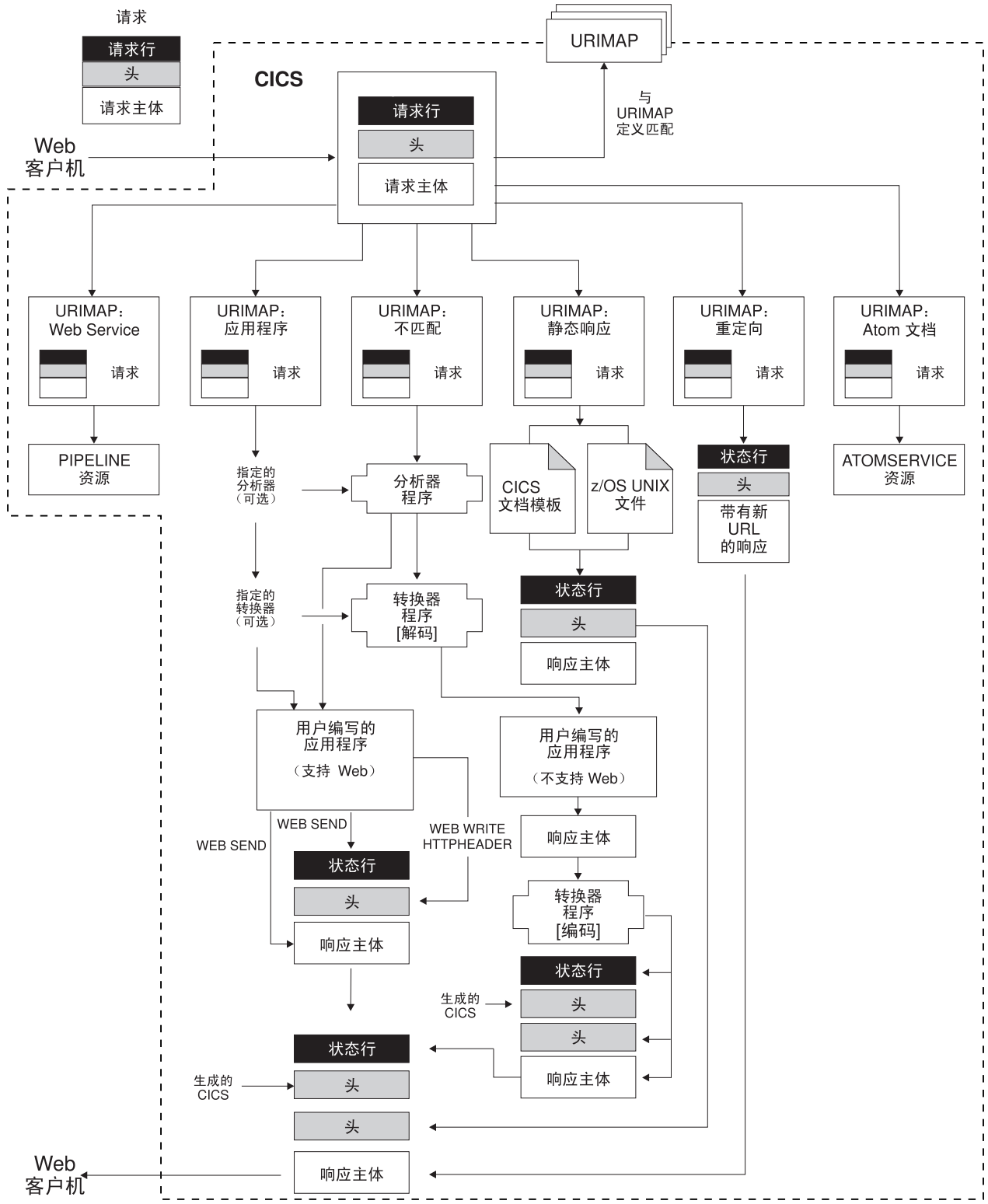


图 2. 作为 HTTP 服务器的 CICS 的处理

下面描述执行适用于作为 HTTP 服务器的 CICS 的处理:

1. **CICS 接收 TCP/IP 连接请求。**CICS 套接字域使用端口的 TCPIPSERVICE 资源定义来确定 CICS Web Support 应处理的请求。TCPIPSERVICE 定义指定应用于请求的安全属性，指定接收请求消息的超时设置，并且限制可为单个请求接收的最大数据量。
2. **CICS 将请求的 URL 与 URIMAP 定义匹配（如果可用的话）。**CICS 尝试将 HTTP 请求中指定的 URL 和与 TCPIPSERVICE 定义相关，并且适用于作为 HTTP 服务器的 CICS 的任何 URIMAP 资源定义进行匹配。如果成功进行了匹配，那么 URIMAP 定义将告诉 CICS 如何处理请求。如果未找到匹配，CICS 将按缺省过程继续执行，该过程从处理阶段 5 的分析器程序开始。
3. **如果 URIMAP 定义指定重定向，那么 CICS 将 Web 客户机重定向到指定的 URL。**CICS 组成重定向消息并将它传输到 Web 客户机。对该 HTTP 请求的处理即告完成。
4. **如果 URIMAP 定义与 Atom 文档有关，那么 CICS 将找到指定的 ATOMSERVICE 资源来处理该请求。** 第 215 页的第 20 章，『CICS 中 Atom 订阅源如何工作』 中描述了针对 Atom 文档的处理过程。
5. **如果 URIMAP 定义与 Web Service 有关，那么 CICS 将找到指定的 PIPELINE资源来处理该请求。** *CICS Web Services Guide* 中描述了针对 Web Service 的处理过程。
6. **如果 URIMAP 定义指定静态响应，那么 CICS 形成并提供该响应。**CICS 使用文档模板或 z/OS UNIX System Services 文件，并加上合适的 HTTP 头，以构成 HTTP 响应。在响应经过适当的代码页转换之后，CICS 将响应发送给 Web 客户机。对该 HTTP 请求的处理即告完成。
7. **如果 URIMAP 定义指定了分析器程序的使用，或者没有找到匹配的 URIMAP 定义，那么会运行分析器程序。**分析器程序可以动态解释请求，或者它可用于监控或审计目的。

   如果还没有为请求设置 URIMAP 定义，那么必须在请求处理路径中使用 TCPIPSERVICE 定义的分析器程序。如果您将具有特殊需求的、不是支持 Web 的应用程序用于代码页转换或 CICS TS V3 之前的兼容性处理，那么可能也需要该分析器程序。（第 111 页的第 10 章，『分析器程序』说明了这些情况。）另外，分析器程序是可选的，但要注意的是，如果未找到 URIMAP 定义，那么调用分析器程序来处理该请求。

   如果正在使用分析器程序，那么将 HTTP 请求和 HTTP 头传递给分析器程序。分析器程序可以解释请求以确定:

   - 要使用哪些 CICS 资源对请求提供服务。
   - 哪个用户标识将与请求关联。
   - 需要执行哪些剩余的处理阶段。
8. **转换器程序可以用于对请求进行解码，并为应用程序构造输入。**支持 Web 的应用程序应该接受 HTTP 请求而不进行任何解码。然而，如果要使用需要通信区域输入的不支持 Web 的应用程序来对 HTTP 请求提供服务，那么可以使用转换器程序来对请求解码，并构造符合您的应用程序需求的输入。可以使用 URIMAP 定义来指定转换器程序，或者可以由分析器程序来选择它。
9. **执行应用程序以对请求提供服务。**您可以使用 URIMAP 定义或使用分析器程序来指定应用程序。使用 EXEC CICS WEB 和 EXEC CICS DOCUMENT 应用程序编程接口的支持 Web 的应用程序可用于处理请求和构造。可以使用转换器程序（它

将 Web 客户机的请求转换为可接受的输入并且根据程序的输出构成 HTTP 响应），或者使用调用不支持 Web 的程序并使用其输出的支持 Web 的应用程序，为 Web 启用不支持 Web 的应用程序。

在别名事务下运行的应用程序。

应用程序可以执行以下任务:

- 如果应用程序支持 Web，那么它可以检查请求的 HTTP 头，从请求行抽取信息（例如，查询字符串），将请求主体接收到缓冲区中以供处理，为响应的状态行选择状态码和文本，并写响应的 HTTP 头。EXEC CICS WEB API 命令（如 WEB SEND 和 WEB WRITE HTTPHEADER）用于构造响应。
- 不管应用程序是否支持 Web，它都可以生成构成响应的主体的输出。支持 Web 的应用程序可以从 CICS 文档模板或从数据缓冲区构成实体主体。不支持 Web 的应用程序可以生成输出，该输出可以由支持 Web 的应用程序或转换器程序转换为实体主体。

10. **转换器程序可以用于编码来自应用程序的输出并构造 HTTP 响应。**如果应用程序不支持 Web，且它的输出不是发送到 Web 客户机的正确格式，那么可以使用转换器程序以生成包括状态行和 HTTP 头的适当 HTTP 响应。转换器程序还可以在需要时对输出执行其他类型的处理。

    转换器程序可以指定重复处理阶段 6（使用转换器程序解码或进行其他处理）、7（应用程序）和 8（使用转换器程序编码或进行其他处理）。因为转换器程序可以更改应用程序的名称，所以您可以使用该设施以允许多个应用程序依次处理同一请求，并提供单个响应。

11. **如果 CICS Web Support 进程中发生请求错误或者异常终止，那么会发送一个错误响应给 Web 客户机，该响应可以使用用户可替换的 Web 出错程序进行定制。**DFHWBEP 或 DFHWBERX 接收有关错误情况的信息，以及 CICS 计划发送给 Web 客户机的缺省 HTTP 响应（包含状态码和状态文本）。用户可替换的程序可以定制响应或构建新的响应，并且将它返回给 CICS 以进行发送。

    不是在所有的错误情况中都会使用 Web 出错程序。当请求的初始处理中出现问题，且后继处理中发生异常终止或故障时会使用它们。在处理（例如用户编写的应用程序执行的处理）正确地完成以及发生预期的错误或重定向响应的情况下，不使用 Web 出错程序。

12. **CICS 生成某些必需的 HTTP 头并将它们添加到消息。**根据响应的 HTTP 版本生成合适的头。如果响应是 HTTP/1.1 响应，那么 CICS 添加 HTTP/1.1 消息需要的头。如果响应是 HTTP/1.0 响应，CICS 会在客户机请求持续连接后添加 Connection: Keep-Alive 头，并添加少数其他头。部分这些头的值由 CICS 直接生成（例如，Date 头），而其他头的值基于支持 Web 的应用程序（使用 WEB SEND 命令）或 URIMAP 定义提供的信息。这些头可以添加到支持 Web 的应用程序的输出，也可以添加到转换器程序的输出。

13. **CICS 将完整的 HTTP 响应传输到 Web 客户机。**如果 Web 客户机支持持续连接，那么 CICS 使连接保持打开状态以接收更多可能的 HTTP 请求，直到用户应用程序或 Web 客户机请求关闭或达到超时值。

在该过程中，当消息进入和离开 CICS 环境时通常需要代码页转换，以便 CICS Web Support 处理和用户编写的应用程序（通常使用 EBCDIC 编码）可以与 Web 客户机（通

常使用 ASCII 编码）通信。第 37 页的『CICS Web Support 的代码页转换』说明了这种情况发生的时间和方式。可以在 WEB SEND 命令或 WEB RECEIVE 命令中使用一些选项来指定需要的代码页转换的类型。

作为 HTTP 客户机的 CICS 的 HTTP 请求和响应处理

对于作为 HTTP 客户机的 CICS，CICS 是 Web 客户机，并且它与 HTTP 服务器通信。用户编写的应用程序通过 CICS 将请求发送到 HTTP 服务器，并从 CICS 接收响应。CICS 维护与服务器的持续连接。在该应用程序发出的命令中使用会话令牌来识别连接。

发出 HTTP 请求并接收响应的应用程序必须使用 EXEC CICS WEB API 命令以明确地指示与服务器进行交互操作。可以使用支持 Web 的应用程序发出 HTTP 请求，然后处理结果以将信息提供给不支持 Web 的应用程序。

应该将发出 HTTP 请求的应用程序设计为处理 CICS 从服务器接收的任何内容以响应该请求，这些内容可能包含错误响应、重定向到另一个 URL、嵌入式超文本链接、HTML 表单、图像源或者请求应用程序的操作的其他项。如果需要，CICS 可以为请求和响应执行代码页转换。

第 30 页的图 3 显示本主题中描述的过程。

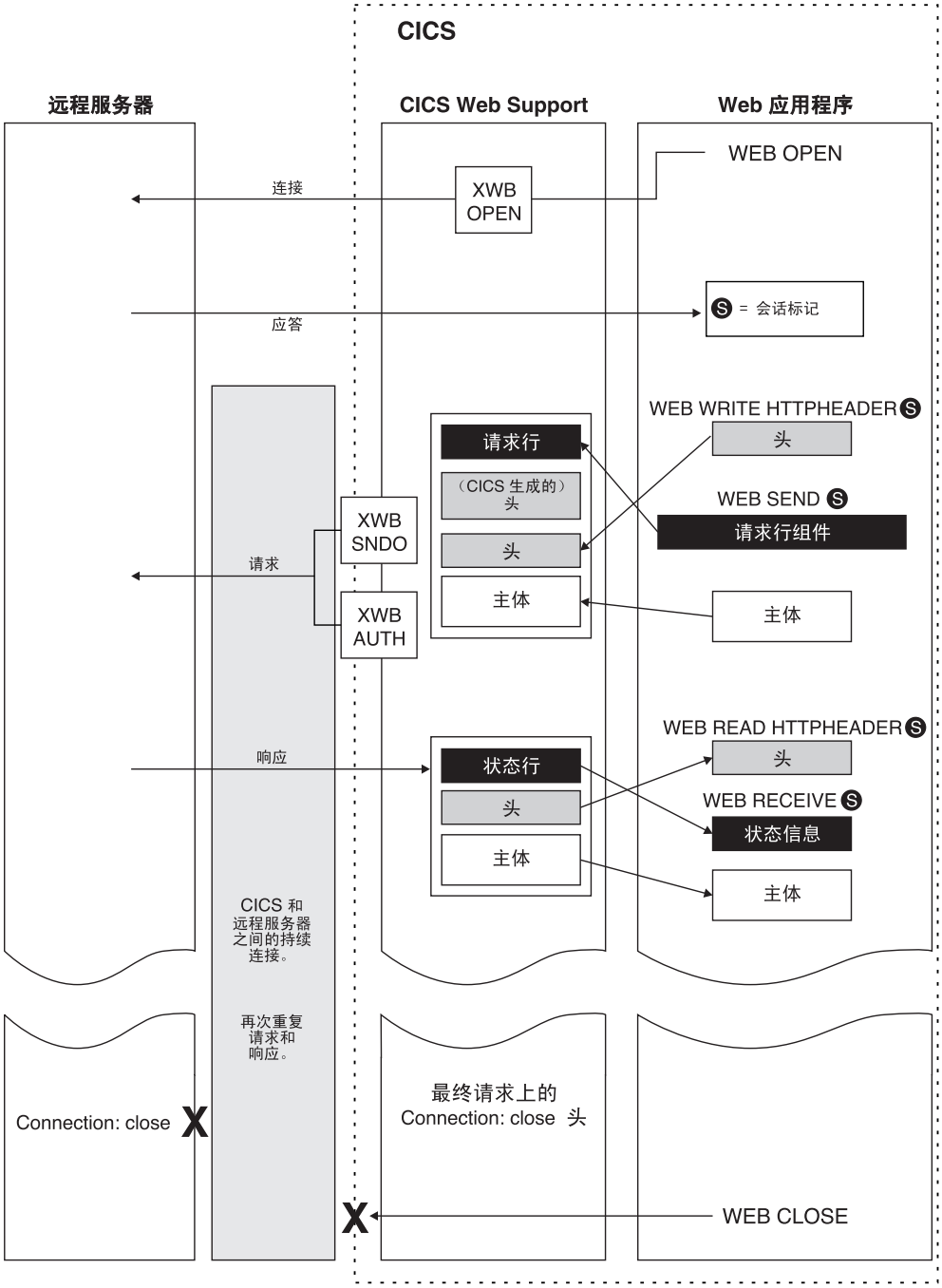


图 3. 作为 HTTP 客户机的 CICS 的处理

作为 HTTP 客户机的 CICS 的处理过程如下所示:

1. **应用程序通过 CICS 启动与 HTTP 服务器的连接。**应用程序通过发出 EXEC CICS WEB OPEN 命令来执行该操作。可以引用您创建的 URIMAP 资源定义为连接指定方案和主机名，也可以由应用程序提供该信息。（*CICS Resource Definition Guide* 提供了关于 URIMAP 定义的更多信息。） 应用程序一次可以打开多个连接。
2. **CICS 建立与服务器之间的连接。**使用应用程序提供的信息，CICS 在套接字上打开 TCP/IP 连接并联系服务器。在该过程中，如果需要，可使用 XWBOPEN 用户出口（如果已使用 ENABLE PROGRAM 命令激活它）通过代理服务器重定向应用程序

的请求，并将安全策略应用到主机的连接。建立 TCP/IP 连接之后，CICS 将会话令牌返回给应用程序以唯一地标识该连接。该会话令牌将用于由应用程序发出的、与该连接有关的所有剩余命令。『会话令牌』介绍了有关会话令牌的更多信息。

3. **应用程序可为它的请求编写 HTTP 头。**可使用 WEB WRITE HTTPHEADER 命令构建用户编写的 HTTP 头并存储以备发送。
4. **应用程序指定请求行的组成部分。**使用 WEB SEND 或 WEB CONVERSE 命令指定请求方法、路径信息以及查询字符串。请求的 HTTP 版本由 CICS 提供。
5. **应用程序可为它的请求生成实体主体。**在 WEB SEND 命令或 WEB CONVERSE 命令上指定请求的内容。它可以从 CICS 文档（使用 DOCUMENT 接口）形成或从缓冲区的内容形成。如果服务器是 HTTP/1.1，那么可将分块的传输编码用于从缓冲区的内容形成的请求主体（但不用于 CICS 文档）。
6. **应用程序开始传输请求。**应用程序发出 WEB SEND 或 WEB CONVERSE 命令时，请求传递给 CICS 以通过会话标识指定的连接进行发送。
7. **CICS 生成一些必需的 HTTP 头并将它们添加到请求，然后将请求发送到服务器。**其中部分头（如 Date 头）的值由 CICS 直接生成，而其他头的值则基于应用程序（使用 WEB SEND 或 WEB CONVERSE 命令）或 URIMAP 定义提供的信息。在发送请求期间，如果需要，可以调用两个用户出口。调用 XWBSNDO 以对个别请求应用安全策略，而 XWBAUTH 指定了基本认证必需的用户名和密码详细信息。
8. **服务器接收并处理请求，并提供响应。**CICS 将响应传递给应用程序。
9. **应用程序会检查响应。**可使用 WEB READ HTTPHEADER 命令或 HTTP 头浏览命令检查响应的头。WEB RECEIVE 或 WEB CONVERSE 命令接收可由应用程序处理的响应的主体（如果有），以及响应的状态码和状态文本。
10. **应用程序会启动更多的请求和响应。**如果服务器支持持续连接，那么通过会话令牌标识的连接会保持打开状态，以接收更多请求。
11. **应用程序开始关闭与服务器连接。**当完成所有请求和响应时，应用程序发出 WEB CLOSE 命令，CICS 会关闭本端的 TCP/IP 连接。如果应用程序不发出 WEB CLOSE 命令，那么在任务结束时关闭连接。

在该过程中，当消息进入和离开 CICS 环境时通常需要代码页转换，以便 CICS Web Support 处理和用户编写的应用程序（通常使用 EBCDIC 编码）可以与 HTTP 服务器（通常使用 ASCII 编码）通信。第 37 页的『CICS Web Support 的代码页转换』说明了这种情况发生的时间和方式。可使用 WEB SEND、WEB RECEIVE 或 WEB CONVERSE 命令中的选项指定所需的代码页转换类型。对于作为 HTTP 客户机的 CICS，缺省情况是当发送和接收消息时进行代码页转换。

会话令牌

会话令牌是一个 8 字节的二进制值，它唯一地识别作为 HTTP 客户机的 CICS 和作为 HTTP 服务器的 CICS 之间的连接。对每个连接使用会话令牌意味着 CICS Web Support 可以通过不同的任务管理与服务器的多个连接，并还意味着应用程序可以控制多个连接。

在响应用户应用程序发出的 WEB OPEN 命令时开始建立连接。成功完成 WEB OPEN 命令后返回会话令牌，并且在与该连接相关的应用程序发出的所有 EXEC CICS 命令中使用该会话令牌。

使用该连接，用户应用程序可向服务器发出 HTTP 客户机请求并接收来自它的响应。该连接可在一个请求和一个响应的多次交换期间持久保存，直到应用程序或服务器选择终止该连接。第 36 页的『CICS Web Support 如何处理持续连接』提供了有关 CICS Web Support 如何处理和终止持续连接的更多详细信息。

如果服务器终止连接，那么应用程序无法使用该连接发送任何进一步的连接，但它可以读服务器在终止连接之前发送的响应。应用程序发出 WEB CLOSE 命令之前，会话令牌都保持有效，以便可以用于访问该数据的命令。发出 WEB CLOSE 命令之后，应用到连接的会话令牌不再有效。如果应用程序不发出 WEB CLOSE 命令，那么在任务结束时关闭连接。

最大打开客户机连接的数（每一个由一个会话令牌表示），可以在一个 CICS 区域中同时出现的是 32768 个。

CICS Web Support 的 URL

在 CICS Web Support 提供的资源的请求 URL 中，URL 的路径部分由您确定。在 CICS Web Support 中，URIMAP 定义或分析器程序在 CICS 提供的请求 URL 和资源之间创建链接，因此 URL 不需要与 CICS 资源有任何直接关系。然而，您可以设计 URL 以提供用于处理或管理目的的信息。

第 10 页的『URL 的组成部分』说明了请求 URL 及其角色的不同组成。

IRI（国际资源标识）是一种较新的资源标识格式，允许使用适用于英语以外的其他本地语言的字符和格式。可使用 IRI 替换 URI 或 URL，请求和响应中所包含的应用程序为 IRI 提供支持。 CICS 支持在 URIMAP 资源定义中使用 IRI。有关 IRI 的更多信息，请参阅第 221 页的『国际资源标识（IRI）』。

应用程序生成的响应的 URL

请求 URL 中的信息可由分析器程序和用户编写的应用程序使用。

如果在处理请求时使用分析器程序，您设计的 URL 应该向分析器程序指明将要处理哪些程序和事务。CICS 提供的分析器程序 DFHWBADX 分析路径部分格式为 `/converter/alias/program/other path information` 的 URL，其中 `converter` 命名转换器程序（如果存在），`alias` 命名别名事务，`program` 命名用户应用程序，而 `other path information` 提供分析器未使用的其他信息。

用于提供响应的支持 Web 的应用程序还可以使用 URL 的路径部分的信息。路径部分可由应用程序使用 WEB EXTRACT 命令抽取，经过分析后便可用于确定合适的操作。例如，路径部分可用于指定应用程序提供的特殊函数。或者，如果支持 Web 的应用程序为其他多个应用程序提供前端，那么 URL 的路径部分可以识别请求所针对的应用程序。

对于使用 URIMAP 定义管理的那些应用程序生成的响应，在设计 URL 的路径部分时，可以使其将多个请求 URL 映射到同一个应用程序。为此，可以使所有 URL 的路径部分均以同一方式开头，并创建带通配符的单个 URIMAP 定义，从而将所有请求 URL 映射到单个资源。例如，通过创建一个 URIMAP 定义并在其中指定路径 `/staffapps/ordering/*` 和相关的应用程序，所有以 `/staffapps/ordering/` 作为路径开头的请求便可映射到特定的 CICS 应用程序。然后，该应用程序就可以抽取并分析 URL 其余部分的信息，以确定各个请求的适宜操作。

静态响应的 URL

在 CICS Web Support 中，URL 不需要与 CICS 资源有任何直接关系。对于静态响应，这意味着 URL 不必包含提供响应的文件的完整路径。URIMAP 定义会将请求 URL 与相应的文件相匹配。

但是，如果 z/OS UNIX 文件用作静态响应，那么当您设计请求 URL 的路径部分时，应使其与 z/OS UNIX 上使用的目录相匹配。如果由 CICS Web Support 提供的所有 z/OS UNIX 文件都位于同一目录的子目录下，比如均位于 CICS 区域用户标识的主目录下，那么您可以省略这一目录，并使请求 URL 与文件路径的其余部分相匹配。例如，如果主目录是 /u/cts/CICSHome，而且您想要提供以下 z/OS UNIX 文件作为静态响应:

```
/u/cts/CICSHome/FAQs/ordering.html
/u/cts/CICSHome/help/directory/viewing.html
```

您可以使用请求 URL，如:

```
http://www.example.com/faqs/ordering.html
http://www.example.com/help/directory/viewing.html
```

记住，URL 的路径部分是区分大小写的，z/OS UNIX 名称也一样。通常将 URL 指定为小写。指定 URIMAP 定义中的各项时要注意使用正确的大小写，尤其是在文件名为大小写混用而 URL 均为小写的时候。

您可能希望使请求 URL 与文件目录结构相匹配，以便:

- 简化资源管理。
- 对 Web 服务器执行标准操作。
- 减少要创建的 URIMAP 定义数。

您可以创建带通配符的单个 URIMAP 定义，以利用路径匹配机制传递多个静态响应。您可以使所有静态响应的 URL 路径部分都以相同的方式开头，并将静态响应的文件都存储在同一个 z/OS UNIX 文件目录中。通配符可用在 URL 路径部分结尾处，也可用在 z/OS UNIX 文件的文件路径结尾处。在上例中，存储在 FAQ 目录中的所有 HTML 文档都由一个 URIMAP 定义提供，该定义指定了路径 /faqs/*，并将 HFSFILE 属性指定为 /u/cts/CICSHome/FAQs/*。对于名称开头方式相同的 CICS 文档模板，也可采用类似的方法。请注意，静态响应的 URIMAP 定义会指定媒介类型（如 text/html），因此如果您需要以该方法提供不同的文件类型，请确保将其分别存储在不同的目录中。

查询字符串

请求 URL 中的查询字符串可用于选择替换的 URIMAP 定义。要在 URIMAP 匹配中使用查询字符串，必须在 URIMAP 定义的路径属性中指定完整和确切的查询字符串以及路径本身。

对于应用程序生成的响应，应用程序可以通过使用 WEB EXTRACT 命令或 WEB READ FORMFIELD 命令，抽取并分析查询字符串中的信息。无论是否将查询字符串用于 URIMAP 匹配，都可以做到这一点。

如果通过文档模板提供静态响应，那么 CICS 会自动将查询字符串的内容传递到指定的 CICS 文档模板（相当于符号列表）。如果希望将查询字符串的内容用于文档模板，那么可在您的文档模板中加入将用查询字符串的内容替代的相应变量。只有当查询字符串尚未用于 URIMAP 匹配时，才可以做到这一点。

URL 长度: CICS Web Support

CICS Web Support 对 URL 长度有以下限制:

- 对于入站 HTTP 请求（由作为 HTTP 服务器的 CICS 接收）的 URL，CICS 接受的长度可达 32K。该长度至少是某些常用 Web 浏览器客户机支持的长度的八倍。如果 CICS 接收的 URL 的长度超过它所能处理的长度，那么它将返回 414（请求 URI 太长）状态码。
- 对于出站 HTTP 请求（由作为 HTTP 客户机的 CICS 发出）的 URL，CICS 支持 URIMAP 资源定义中的路径部分最多采用 255 个字符。发出请求的用户应用程序可能覆盖 URIMAP 定义（或根本就一个也不使用），并提供更长的路径部分。检查可以由服务器处理的 URL 长度。

URL 长度: URIMAP 定义

在为使用 URIMAP 定义提供的资源选择 URL 时，请注意对 URL 长度的以下限制:

- CICS 支持 URIMAP 资源定义中多达 255 个字符的路径部分。不要尝试使用比这个长度长的路径部分。HTTP/1.1 规范声明: 对于总长度（由方案、主机和路径部分以及定界符组成）超过 255 个字符的 URL，服务器应该谨慎，这是因为较旧的 Web 客户机和代理可能不会正确支持这些 URL。如果正在使用的 IRI 包含百分比编码的 Unicode 字符，那么请注意该上下文中的一个字符表示一个 ASCII 字符，而不是原始的 Unicode 字符。例如，根据该 255 个字符限制，百分比编码表示形式为 %D0%B4 的西里尔字母字符表示 6 个字符。
- 如果需要使用更长的路径部分，那么通常可以这么做，因为您不必在 URIMAP 资源定义中指定完整路径。星号（*）可用作路径末尾的通配符。如果满足以下条件，那么 URIMAP 定义的行为将是正确的:
  - URL 的指定部分对于该资源来说是唯一的。
  - URL 的指定部分对于该资源来说不是唯一的，但您提供的是静态响应，并且使用路径匹配机制来完成 URL。
- 如果您为了 URIMAP 匹配目的而使用查询字符串，并在 URIMAP 定义的路径属性中指定它，那么总长度必须仍然限制为 255 个字符。（如果该操作仍然正确，那么路径部分可由星号替换，但是星号不能用在查询字符串中。）如果没有为此目的使用查询字符串，那么可以接受任何长度的查询字符串，最长为 CICS Web Support 的 URL 长度的 32K 限制。
- 对于重定向（使用 URIMAP 定义中的 LOCATION 和 REDIRECTTYPE 属性），CICS 支持多达 255 个字符的重定向 URL。该 URL 必须是**完整的** URL，包括方案、主机和路径部分以及合适的定界符。如果计划将资源用作已重定向客户机的目标，那么确保其完整 URL 符合该 255 个字符的限制。

CICS Web Support 如何处理分块的传输编码

CICS 可以发送和接收使用分块的传输编码的消息。

作为 HTTP 服务器的 CICS 可以将分块的消息作为请求接收，或将分块的消息作为响应发送。作为 HTTP 客户机的 CICS 可以将分块的消息作为请求发送，或将分块的消息作为响应接收。CICS Web Support 处理这些不同情况，具体如下:

- 当作为 HTTP 服务器的 CICS 将分块的消息作为 HTTP 请求接收时，CICS Web Support 会识别分块的编码。它将等待接收完所有块（由接收到零长度的块来标示），并组装这些块以形成完整的消息。用户应用程序可以使用 WEB RECEIVE 命令接收组织的消息体。
  - 在与请求到达的端口相关的 TCPIPSERVICE 资源定义中使用 MAXDATALEN 选项，您可以限制 CICS 接受单个分块消息的总数据量。
  - 当 CICS 是 HTTP 服务器时，接收分块消息的超时值由 TCPIPSERVICE 定义的 SOCKETCLOSE 属性设置。
  - 可以使用 HTTP 头命令读取分块消息的尾部头。尾部头识别以尾部头出现的头的名称。如果在处理请求时使用分析器程序，请注意，尾部头不与主请求头一起传递到分析器程序。
- 当作为 HTTP 客户机的 CICS 将分块的消息作为对应用程序请求的响应接收时，在将这些块作为实体主体传递到应用程序之前，会先组装它们，而任何尾部头可以使用 HTTP 头命令读取。要指定应用程序将等待多久以接收响应，可以针对与该应用程序相关的事务标识，使用其事务概要文件定义中的 RTIMOUT 属性。
- CICS 作为 HTTP 服务器（响应）或作为 HTTP 客户机（请求）发送分块消息时，通过 WEB SEND 命令中使用的 CHUNKING(CHUNKYES) 选项，应用程序可以为每个消息块指定分块的传输编码。可以用最便于应用程序采用的方式来分割消息。CICS 发送每个消息块，并且添加合适的 HTTP 头向接收方表明正在使用分块的传输编码。应用程序发出带 CHUNKING(CHUNKEND) 的 WEB SEND 以表明消息结束。然后，CICS 会发送一个空块（包含空白行）以及需要的任何尾部头以结束分块的消息。

第 78 页的『使用分块的传输编码发送 HTTP 请求或响应』说明了从 CICS 发送 HTTP 消息时，采用分块的传输编码的过程。为了使分块的消息可供接收方接受，应按该过程进行操作。

CICS Web Support 如何处理管道传送

管道化的请求序列可以由 CICS 发送和接收。作为 HTTP 服务器的 CICS 可以接收 Web 客户机通过管道发出的请求序列，而作为 HTTP 客户机的 CICS 可以通过管道将请求序列发送至服务器。

CICS Web Support 处理管道化的请求序列以及对它们的响应，具体如下:

- 当作为 HTTP 服务器的 CICS 接收管道化的 HTTP 请求序列时，将按顺序处理这些请求。这是为了确保返回响应的顺序与发送请求的顺序相同。CICS 将管道化序列中的每条消息视作独立的事务，要么提供在 URIMAP 定义中指定的静态响应，要么将消息传递到应用程序并等待该应用程序生成响应。每个事务处理单个请求并提供响应。管道传送的消息序列中的剩余请求由 CICS 保留，直到发送了对先前请求的响应，然后才启动一个新的事务以处理每个进一步请求。
- 当作为 HTTP 客户机的 CICS 发送管道化的请求序列时，将自动启用管道传送。每个 HTTP 请求被立即发送，因此应用程序可以在接收任何响应前发送多个 HTTP 请求。当管道化序列中的最后一条消息发出后，应用程序可以开始接收响应。
- 当作为 HTTP 客户机的 CICS 收到管道化的请求序列中的 HTTP 响应时，响应返回到应用程序的顺序将是 CICS 从服务器接它们的顺序。支持管道传送的服务器按接收请求的顺序来提供响应。应用程序在发送完所有 HTTP 请求后开始接收响应。

对于作为 HTTP 客户机的 CICS，将由应用程序确保请求的任何管道化序列是幂等的。第 18 页的『管道传送』介绍了幂等性。为了不对应用程序的逻辑以及服务器造成影响，如果您不确定请求序列是不是幂等的，建议您单独发出请求，等收到对请求的响应后再发出下一个请求。

相关任务

第 138 页的『样本程序: 将请求通过管道传送给 HTTP 服务器』
样本程序 DFH$WBPA（汇编程序）、DFH$WBPC（C）和 DFH0WBPO （COBOL）演示了 CICS 如何通过管道将客户机请求传送到 HTTP 服务器。

CICS Web Support 如何处理持续连接

持续连接是 CICS Web Support 的缺省行为。

在 CICS TS 3.1 之前，连接行为是: 从 Web 客户机接收数据后，如果 Web 客户机没有发送 Connection: Keep-Alive 头，CICS 通常会关闭连接。

现在，当在 Web 客户机和作为 HTTP 服务器的 CICS 之间建立连接，或者在作为 HTTP 客户机和服务器的两个 CICS 之间建立连接时，缺省情况下，CICS 会尝试保持连接畅通。

当 CICS 是 HTTP 服务器时，在以下情况下，将关闭持续连接:

- 处理请求的用户编写的应用程序关闭连接（通过在 WEB SEND 命令中指定 CLOSESTATUS(CLOSE) 选项）。
- Web 客户机关闭连接（由 Connection: close 头通知）。
- Web 客户机是不发送 Connection: Keep-Alive 头的 HTTP/1.0 客户机。
- 达到超时期限（表明连接已失败，或 Web 客户机主动退出连接）。

否则，CICS 将使持续连接保持畅通，以供 Web 客户机发送更多请求。如果存在与客户机的持续连接，那么通过 Web 出错程序发送错误响应后，CICS 保持连接打开。特例是 CICS 为响应选择 501（方法未实施）状态码，在这种情况下，CICS 关闭该连接。

使用 CICS Web Support 的 TCPIPSERVICE 资源定义，TCPIPSERVICE 定义的 SOCKETCLOSE 属性不应该指定为零。SOCKETCLOSE 设置为零意味着作为 HTTP 服务器的 CICS 从 Web 客户机接收数据后立即关闭连接，除非有更多数据正在等待。这意味着无法维持持续连接。

当 CICS 是 HTTP 客户机时，在以下情况下，将关闭持续连接:

- 服务器关闭连接（由发送 Connection: close 头的 HTTP/1.1 服务器通知或由无法发送 Connection: Keep-Alive 头的 HTTP/1.0 服务器通知）。
- 用户应用程序关闭连接（通过在 WEB SEND 或 WEB CONVERSE 命令中指定 CLOSESTATUS(CLOSE) 选项，或通过发出 WEB CLOSE 命令）。
- 任务已结束而连接尚未关闭。

如果应用程序需要测试服务器是否已请求终止连接，那么可以使用 READ HTTPHEADER 命令在服务器返回的最后一条消息中查找 Connection: close 头。如果服务器请求关闭连接，但应用程序尚未发出 WEB CLOSE 命令，那么 CICS 关闭连接，但是仍保留与连接有关的数据（包括从服务器收到的最后一个响应及其 HTTP 头）。应用程序可以继续使用该数据，直到它发出 WEB CLOSE 命令或任务结束为止。

对作为 HTTP 客户机的 CICS 发出 WEB CLOSE 命令，不会使 CICS 通知服务器应终止持续连接。它只是让 CICS 关闭连接。好的做法是在发出给服务器的最后一个请求上包含 Connection: close 头。使用该头意味着服务器可以在发送最后一个响应后立即关闭它的持续连接，而不是等着看 CICS 是否还要发送请求直到超时为止。要包含该头，请在 WEB SEND 或 WEB CONVERSE 命令中指定 CLOSESTATUS(CLOSE) 选项。

如果作为 HTTP 客户机的 CICS 正在与 HTTP/1.0 服务器通信，那么 CICS 自动在 HTTP 消息上发送 Connection: Keep-Alive 头。请求连接的应用程序不需要提供这些头。Keep-Alive 通知服务器需要持续连接。

CICS Web Support 的代码页转换

当 CICS 与 Web 客户机或服务器交换消息时，在进入和离开 CICS 环境时通常需要对这些消息中的字符数据进行代码页转换。

出于两个原因需要对这些消息中的文本进行代码页转换:

- CICS 和用于 CICS 的用户编写的应用程序通常使用 EBCDIC 编码，但是 Web 客户机和服务器通常使用 ASCII 编码。
- 在每个编码中，使用大量不同的代码页来支持本地语言。

消息的非文本内容（如图像或应用程序数据）不需要进行代码页转换。

在 CICS Transaction Server for z/OS V3R1 之前的 CICS 发行版中，CICS Web Support 的代码页转换是使用代码页转换表（DFHCNV）处理的。在 CICS Transaction Server for z/OS V4R1 中，除了用于升级目的的极少数情况外，其他情况下 CICS Web Support 均不再需要代码页转换表。CICS Web Support 使用 z/OS 转换服务处理代码页转换。

在 CICS Web Support 中，文本代码页转换的缺省值如下:

- 缺省字符集是 ASCII Latin-1 字符集 ISO-8859-1。在 HTTP 消息中，请求或状态行和 HTTP 头通常都是 US-ASCII 字符集，它是 ISO-8859-1 较早的子集。包含文本的消息体通常是 ISO-8859-1。
- CICS 环境中数据的缺省 EBCDIC 代码页由 CICS 区域的 **LOCALCCSID** 系统初始化参数指定。**LOCALCCSID** 的缺省值是 EBCDIC 拉丁字符集（代码页 037）。

有时可以标识更合适的备用代码页:

- Web 客户机或服务器可以在 Content-Type 头中为请求或响应指定字符集，它是已用于消息体的字符集。
- Web 客户机可以在请求上发送 Accept-Charset 头，以表明对于响应哪些字符集是可接受的。
- 对于非 HTTP 请求和某些较早的 HTTP 实施，传输消息时使用的字符集可能未在消息头中标识，并且您可能需要从您自己对消息源的了解来标识它。
- 如果缺省值不合适，那么应用程序员需要标识他们的应用程序可用于接收消息数据的合适代码页。

CICS 不支持 IANA 命名的所有字符集。CICS 为进行代码页转换支持的 IANA 字符集在 第 333 页的附录 A, 『HTML 编码字符集』中列出。

在大多数情况下，消息的介质类型可以确定是否发生代码页转换。具有非文本介质类型的请求或响应主体通常不经历代码页转换。与支持 Web 的应用程序的兼容性的异常在先前发行版中编码；如果命令中使用的选项表明应用程序在 CICS Transaction Server for z/OS V3R1 前编码，那么介质类型不影响代码页转换。

根据消息类型和处理路径的不同，代码页转换信息可由 CICS 自动识别，可在 URIMAP 定义中指定，可由分析器程序指定，也可在支持 Web 的应用程序发出的命令中指定。『作为 HTTP 服务器的 CICS 的代码页转换』针对作为 HTTP 服务器的 CICS 介绍了该过程，而第 39 页的『作为 HTTP 客户机的 CICS 的代码页转换』针对作为 HTTP 客户机的 CICS 介绍了该过程。

作为 HTTP 服务器的 CICS 的代码页转换

当作为 HTTP 服务器的 CICS 与 Web 客户机交换消息时，通常需要为消息体进行代码页转换。指定代码页转换的方法取决于您作出应用程序生成的响应还是静态响应，以及您使用支持 Web 的应用程序还是不支持 Web 的应用程序。

请求行和 HTTP 头

请求行或状态行的代码页转换、以及 HTTP 头的代码页转换按如下方式处理:

- 在收到请求后，CICS 立即将请求行（包括所有查询字符串）和 HTTP 头从原有字符集转换成 LOCALCCSID 系统初始化参数指定的 EBCDIC 代码页（它应用于整个本地 CICS 区域，且缺省值为 037）。要成功进行转换，您应当将 LOCALCCSID 系统初始化参数设置为某一 EBCDIC 代码页 – ASCII Latin-1 字符集 ISO-8859-1（代码页 819）可以转换成该代码页。如果 LOCALCCSID 被设置成不适当的代码页，那么 CICS 会使用缺省的 EBCDIC 代码页 037。
- 在应用程序使用 WEB EXTRACT、WEB READ HTTPHEADER 或 WEB READ FORMFIELD 命令来抽取请求行（包括所有查询字符串）和 HTTP 头中的信息时，信息会以转换后的格式显示，即采用 LOCALCCSID 系统初始化参数指定的 EBCDIC 代码页（或缺省值 037）显示。
- 当 CICS 准备发出响应时，状态行和 HTTP 头可以由 CICS 生成，或由应用程序使用 WEB WRITE HTTPHEADER 命令指定。在发送前，所有头和状态行从指定的 EBCDIC 代码页转换为 US-ASCII 字符集。

消息体: 应用程序生成的响应

如果请求要从用户编写的应用程序获取动态响应，那么消息体的代码页转换按如下方式处理:

- 如果支持 Web 的应用程序收到请求，当在 WEB RECEIVE 命令中使用任何代码页转换选项指定转换时，CICS 会执行该代码页转换。如果这些选项未出现，那么不执行代码页转换。您可以提供字符集或允许 CICS 识别字符集，并在缺省代码页不适合的情况下请求代码页。
- 如果在处理请求时使用分析器程序，那么分析器程序可以为传递到存储器块中后继处理阶段的请求副本指定或禁止代码页转换。您提供所使用的字符集和应用程序代码。CICS 仍然保留请求主体的原始版本。使用 EXEC CICS WEB API 命令的应用程序或转换器程序访问的是原始主体，而不是存储器块，而且它们可以通过 EXEC CICS WEB API 命令指定代码页转换。
- 在转换器程序传递存储器块中的请求时，如果处理时没有使用分析器程序，那么 CICS 转换该存储器块中的请求主体，识别字符集并转换为缺省代码页。

- 为了识别 Web 客户机用于请求主体的字符集，CICS 会检查请求头。如果请求头不提供该信息或指定的字符集不受支持，那么 CICS 会假设字符集为缺省字符集，即消息体为 ISO-8859-1 字符集。如果消息体不是该字符集，且头中没有信息，那么需要识别正确的字符集。
- 缺省情况下，CICS 将请求主体转换为 LOCALCCSID 系统初始化参数指定的 EBCDIC 代码页（它应用到整个本地 CICS 区域，且具有缺省值 037）。如果您的应用程序需要另一个代码页（可以是 EBCDIC 或 ASCII），那么您可以指定该代码页。
- 如果应用程序或转换器程序使用 EXEC CICS WEB API 命令发送响应，当在 WEB SEND 命令上指定任何代码页转换选项时，CICS 会执行该代码页转换。如果这些选项未出现，那么不执行代码页转换。
- 如果转换器程序在存储器块中生成响应并将其传递给 CICS 以便发送，那么 CICS 会执行为请求所执行的相同代码页转换。使用分析器程序的字符集和主机代码页设置，如果没有分析器程序，那么使用缺省设置。如果分析器程序禁止对请求执行代码页转换，那么不会对响应主体执行代码页转换。

消息体: 静态响应

如果请求要获取由 URIMAP 定义确定的静态响应，那么消息体的代码页转换按如下方式处理:

- 对于静态响应，CICS 不检查出现在 Web 客户机的请求中的任何消息体，因此不需要代码页转换。
- 您在生成静态响应的 URIMAP 定义中为响应主体指定代码页转换。如果响应包含文本，那么 URIMAP 定义需要指定所有以下几项:
  - 文本介质类型，使用 MEDIATYPE 属性。该属性没有缺省值。
  - Web 客户机的字符集，使用 CHARACTERSET 属性。
  - 使用 HOSTCODEPAGE 属性为响应编写的 CICS 文档模板或 z/OS UNIX 文件所用的代码页。

CICS 检索 z/OS UNIX 文件或 CICS 文档模板并创建文档，然后执行适当的代码页转换。

作为 HTTP 客户机的 CICS 的代码页转换

当作为 HTTP 客户机的 CICS 与服务器交换消息时，通常需要为消息体进行代码页转换。打开连接时指定应用程序代码。通常字符集可以由 CICS 识别或允许使用缺省值。

请求行和 HTTP 头

请求行或状态行的代码页转换、以及 HTTP 头的代码页转换按如下方式处理:

- 当 CICS 准备发出请求时，请求行和 HTTP 头可以由 CICS 生成，或由应用程序使用 WEB WRITE HTTPHEADER 命令指定。发送前，所有头从指定它们的 EBCDIC 代码页转换为 US-ASCII 字符集。
- 接收响应后不久，CICS 将状态行和 HTTP 头从 US-ASCII 字符集转换成 EBCDIC 代码页 037。应用程序接收状态行和其他信息，并检查 EBCDIC 代码页 037（HTTP 头的转换格式）中的 HTTP 头。

消息体

消息体的代码页转换按如下方式处理:

- 由应用程序使用的 EBCDIC 代码页是在启动与服务器通信的 WEB OPEN 命令上指定的。缺省值是由 LOCALCCSID 系统初始化参数（它应用到整个本地 CICS 区域，且具有缺省值 037）指定的 EBCDIC 代码页。CICS 使用该信息在该连接上转换请求和响应的消息体。
- 对于应用程序发出的每个请求，WEB SEND 或 WEB CONVERSE 命令中的 CLIENTCONV 选项指定 CICS 是否为请求主体执行代码页转换。缺省情况是执行代码页转换。如果正在使用 WEB CONVERSE 命令，那么可以为请求主体和/或响应主体指定代码页转换，或者都不指定。
- 如果您已为请求指定转换，那么缺省情况是 CICS 将请求主体转换成 ISO-8859-1 字符集。如果您知道服务器更希望使用另一个代码页，那么可以使用 WEB SEND 或 WEB CONVERSE 命令的 CHARACTERSET 选项来选择另一个代码页。
- 对于应用程序接收的每个响应，WEB RECEIVE 或 WEB CONVERSE 命令的 CLIENTCONV 选项指定 CICS 是否为响应主体执行代码页转换，以将它转换为打开连接时指定的 EBCDIC 代码页。缺省情况是执行代码页转换。CICS 检查响应头以识别服务器用于响应主体的字符集。如果响应头不提供该信息或指定的字符集不受支持，那么 CICS 会假设字符集为缺省字符集，即消息体为 ISO-8859-1 字符集。

作为 HTTP 服务器的 CICS 的 HTTP/1.1 一致性

CICS Web Support 现在支持 HTTP/1.1。

CICS Transaction Server for z/OS V3R1 之前的 CICS 发行版支持 HTTP/1.0。CICS Web Support 现已得到增强，可以提供 HTTP/1.1 规范中的功能，包括分块传送编码、管道传送和持续连接。

正如因特网协会和 IETF（因特网工程任务组织）请求评论 RFC 2616 *Hypertext Transfer Protocol - HTTP/1.1*（http://www.ietf.org/rfc/rfc2616.txt）中所描述，CICS Web Support 有条件地遵从 HTTP/1.1 规范。

有条件地遵从 HTTP/1.1 规范意味着 CICS 满足所有“MUST”级别需求，但并不满足所有的“SHOULD”级别需求。HTTP/1.1 规范中详细描述了这些需求，这些需求与 CICS 本身提供的功能有关。满足其协议的所有 MUST 或 REQUIRED 级别和所有 SHOULD 级别需求的实施称为无条件地遵从。如果实施无法满足它实施的协议的一个或多个 MUST 或 REQUIRED 级别需求，那么该实施是不遵从的。

遵从 HTTP/1.1 规范的 CICS Web Support 有三个方面。

- CICS Web Support 执行 HTTP 服务器需要的操作。例如，CICS Web Support 接收入站请求、维护持续连接、写特定 HTTP 头并传输响应。CICS Web Support 在这些操作中的行为将有条件地遵从 HTTP/1.1 规范。必要时，可能要改变 CICS 以前发行版的行为。第 41 页的『遵从 HTTP/1.1 的 CICS Web Support 行为』描述 CICS Web Support 的行为如何遵从 HTTP/1.1，并说明了它在哪些方面与 CICS Transaction Server for z/OS V3R1 之前的 CICS 发行版不同。
- HTTP/1.1 规范的某些部分与 CICS Web Support 无关。例如，CICS Web Support 不充当代理服务器，也不提供高速缓存设施。第 42 页的『CICS Web Support 不支持的 HTTP 功能』记录了这些区域，因此您可以知道在哪些区域中 HTTP/1.1 一致性对于 CICS Web Support 和用户应用程序无关紧要。
- CICS 中的支持 Web 的用户应用程序可用于创建应用程序生成的 HTTP 响应，并指示 CICS Web Support 如何对响应提供服务。如果您希望 CICS Web Support 实施

遵从 HTTP 协议规范（特别是 HTTP/1.1），那么您的用户应用程序共同承担职责，以保证它们执行的操作的一致性。虽然本文档中提供了一些基本指导信息，但是您可以查看您所遵从的 HTTP 规范以了解更多详细信息。第 67 页的第 6 章，『为作为 HTTP 服务器的 CICS 编写支持 Web 的应用程序』描述了针对 CICS Web Support 编写用户应用程序的过程。

在实践方面，HTTP/1.1 规范中的不同需求级别（MUST、SHOULD 或 MAY）应该由您的应用程序处理，具体如下:

- 必须实施 MUST 级别需求以维护一致性。CICS Web Support 设计为处理或协助您遵从所有 MUST 级别需求，这些需求应用到直接的活动。如果选择满足 CICS Web Support 尚未处理的可选需求，那么可能应用某些其他 MUST 级别需求。而且，某些 MUST 级别需求与您的应用程序在特定情况下必须不执行的操作有关。
- SHOULD 级别需求对于有条件地遵从 HTTP 规范不是必需的。CICS 不遵从 HTTP/1.1 规范中的所有 SHOULD 级别需求。然而，如果您的应用程序可以遵从 SHOULD 级别需求而没有什么不方便，那么建议您这么做。
- MAY 操作对于任何 HTTP 实施是可选的（不管它的一致性级别如何）。应该将它们作为建议或推荐来对待。

遵从 HTTP/1.1 的 CICS Web Support 行为

为了您的利益，CICS Web Support 遵从 HTTP/1.1 规范中的许多要求。

不管您正在使用 URIMAP 定义还是分析器程序处理作为 HTTP 服务器的 CICS 的 HTTP 请求，都将应用此处描述的大多数操作，但有一些例外。

- **CICS 检查入站消息是否遵从 HTTP/1.1，并处理或拒绝不遵从该规范的消息。**在涉及 URIMAP 定义或分析器程序前，接收请求时立即进行检查。会进行各种基本验收检查，如果消息不可接受且由 CICS 自己处理问题也不合适，那么可能的话会将错误响应返回给 Web 客户机。不会为 HTTP/1.0 消息执行这些基本验收检查，如果在 TCPIPSERVICE 定义中指定了 USER 协议（而不是 HTTP 协议），也不会执行基本验收检查。

  注: CICS 要求在具有消息体的所有入站 HTTP/1.1 消息中都有 Content-Length 头。如果有消息体但未提供头，或头的值不正确，那么错误消息或后继消息的套接字接收会产生不可预测的结果。对于具有消息体的 HTTP/1.0 消息，Content-Length 头是可选的。
- **CICS 按照 HTTP/1.1 规则来比较 URL。**方案名和主机名比较不区分大小写，但是路径比较区分大小写。比较之前 URL 的所有组成部分都未转义。CICS 还处理请求（其中主机名在请求行中指定）中的绝对 URI 格式。注意，当使用分析器程序而不是 URIMAP 定义处理入站 HTTP 请求时，如果您需要达到这方面的一致性，那么必须对分析器程序进行编码以按照 HTTP/1.1 规范中声明的规则来执行 URL 比较。（请参阅 第 12 页的『HTTP 协议』以获取有关 HTTP 规范的更多信息。）
- **CICS 在出站消息的头一行提供适合的 HTTP 版本号。**除非 CICS 知道 Web 客户机或服务器为 HTTP/1.0 级别，否则它通常将消息识别为 HTTP/1.1。在这种情况下，CICS 将消息识别为 HTTP/1.0。CICS 不支持 Web 客户机从一个 HTTP 版本升级到另一个 HTTP 版本或不同安全协议的请求。
- **在出站 HTTP/1.1 消息中，CICS 提供通常应该出现的 HTTP 头，以使消息遵从 HTTP/1.1。**CICS 还生成一些头，实现一致性并不需要它们，但是它们支持应用程序已请求的操作（例如，Expect 头）。第 335 页的附录 B，『CICS Web Support 的 HTTP

头参考』描述 CICS 编写的头以及创建这些头的环境。这些一致性头提供给由支持 web 的应用程序和不支持 web 的应用程序发送的消息。如果为 TCPIPSERVICE 定义指定 USER 协议（而不是 HTTP 协议），那么不提供它们。对于 HTTP/1.0 消息，只提供 Connection: Keep-Alive、Content-Length、Date 和 Server 头。

- **CICS 对入站请求和出站请求的 Expect 头执行操作。**当作为 HTTP 服务器的 CICS 接收具有 Expect 头的请求时，它将 100-Continue 响应发送给 Web 客户机，并等待剩余请求。对于作为 HTTP 客户机的 CICS，可以使用 WEB SEND 命令中的 EXPECT 选项使 CICS 将 Expect 头发送到服务器，并在发送请求前等待 100-Continue 响应。如果服务器返回另一个响应，那么 CICS 通知应用程序并取消发送。
- **CICS 处理来自 Web 客户机的 OPTIONS 请求，并做出合适的响应。**OPTIONS *（整个服务器，而不是特定资源上的查询）是唯一接受的格式。响应包含有关作为 HTTP 服务器的 CICS 的基本信息（HTTP 版本和服务器软件描述）。未涉及用户应用程序。
- **CICS 处理来自 Web 客户机的 TRACE 请求，并做出合适的响应。**当 CICS Web Support 接收具有 TRACE 方法的正确格式的请求时，它返回包含请求及其原始头以及它获取的任何头（例如，via 头字段）的响应。未涉及用户应用程序。
- **CICS 接受具有分块的传输编码的入站消息，并进行汇编，且支持您使用分块的传输编码以发送出站消息。**可通过 HTTP 头读、写和浏览命令来操纵分块消息的尾部头。这意味着应用程序可将分块的消息当作正常消息来处理。CICS 还支持从用户应用程序发送分块消息，但您必须确保您按正确的过程进行操作以使分块消息可被接收方接受。第 34 页的『CICS Web Support 如何处理分块的传输编码』说明了这方面的 CICS Web Support 行为。
- **CICS 支持入站和出站消息的管道传送。**CICS 允许您从 Web 客户机接收管道化的请求。CICS 依次将每个请求传递到应用程序，通过按接收请求的顺序来响应它们，以确保应用程序遵从 HTTP/1.1。CICS 还允许您将管道化的请求发送到服务器，但是您必须确保按正确的过程进行操作以使管道化的请求序列遵从 HTTP/1.1。第 35 页的『CICS Web Support 如何处理管道传送』说明这方面的 CICS Web Support 行为。
- **CICS 支持虚托管（同一 IP 地址的多个主机名）。**对虚拟主机的支持基于您的 URIMAP 定义。第 9 页的『虚拟主管』说明所提供的支持。
- **缺省情况下，连接是持续的。**对于作为 HTTP 服务器的 CICS 和作为 HTTP 客户机的 CICS 都是这种情况。如果 CICS 正在与 HTTP/1.1 级别的 Web 客户机或服务器通信，那么它保持连接打开，除非 CICS 中的 Web 客户机、服务器或用户应用程序特别请求关闭或任务结束。如果 CICS 正在与 HTTP/1.0 级别的 Web 客户机或服务器通信，那么当支持持续连接时它会发送 Connection: Keep-Alive 头。第 36 页的『CICS Web Support 如何处理持续连接』说明了这方面的 CICS Web Support 行为。

CICS Web Support 不支持的 HTTP 功能

HTTP/1.1 规范为使用 HTTP 协议的各方定义各种角色。CICS Web Support 执行适合于原始服务器、客户机和用户代理（尽管用户可能不涉及每一个 HTTP 客户机请求）的许多功能。HTTP/1.1 规范还包括与代理、网关、隧道和高速缓存相关的需求，而这些需求与 CICS Web Support 不相关，可以忽略它们。

- **CICS 不担当代理。**您可以忽略与代理行为有关的 HTTP/1.1 规范中的所有需求。
- **CICS 不担当网关（另一台服务器的调解者）或隧道（HTTP 连接间的中继设备）。**您可以忽略与网关和隧道行为有关的 HTTP/1.1 规范中的所有需求。

- **CICS 不提供高速缓存设施，也不提供对用户编写的高速缓存设施的支持。**您可以忽略与高速缓存行为有关的 HTTP/1.1 规范中的所有需求。虽然您可以存储从服务器接收的任何信息，但是应该小心不要将存储的信息交付到正在发出请求的用户，该请求期望从服务器接收当前信息。
- **CICS 不旨在用作 Web 浏览器。**通过作为 HTTP 客户机的 CICS，用户应用程序可以为服务器提供的个别、已知资源发出请求，但是浏览因特网时通常不需要它们。CICS 不提供历史记录列表（列出 Web 浏览器的收藏夹或其他功能），因此可以忽略有关这些项的任何需求。

请参阅 第 12 页的『HTTP 协议』以获取有关 HTTP 规范的更多信息。

Atom 格式和发布协议一致性

CICS 支持 RFC 4287 和 RFC 5023 中描述的 Atom 格式和 Atom 发布协议。

以下因特网社区和 IETF（因特网工程任务组织）的“请求评论”（RFC）文档中描述了“Atom 联合格式”（Atom Syndication Format）以及“Atom 发布协议”（Atom Publishing Protocol）：

The Atom Syndication Format（**RFC 4287**）
: Atom 订阅源和条目文档的结构规范。可以从 http://www.ietf.org/rfc/rfc4287.txt 获取该规范。

The Atom Publishing Protocol（**RFC 5023**）
: 用于通过 HTTP 以 Atom 文档格式发布和编辑资源的协议，包括 Atom 服务和类别文档的结构描述。可以从 http://www.ietf.org/rfc/rfc5023.txt 获取该规范。

遵循 Atom 格式和协议的 CICS 行为

CICS 处理遵循 Atom 格式和协议所需的众多任务

- CICS 会针对 Atom 订阅源文档、Atom 条目文档及集合验证 Atom 配置文件的格式，确保其符合 RFC 4287 中的需求。只要可以执行此操作，CICS 都会进行检查以确保您已提供所需的元素，并且会在需要时应用缺省值。用于设置 Atom 订阅源的 CICS 文档介绍了 CICS 无法验证数据而您应负责提供正确数据的一些情况。
- CICS 会处理您所创建的 Atom 服务文档和 Atom 类别文档，并向客户机交付这些文档。然而，CICS 不会验证这些文档的格式，不论它们是位于 Atom 配置文件中，还是作为静态文档进行交付。您应确保符合 RFC 5023 中对这些 Atom 文档类型格式的详细要求。
- CICS 提供了 EXEC CICS API 命令，该命令可以使用 Atom 文档中正确的时间构造格式创建时间戳记。如果您不选择使用 EXEC CICS API 命令来产生时间戳记，那么应确保符合 RFC 4287 中对时间结构内容的详细要求。
- CICS 会处理针对 IRI 的映射和比较任务，这些会在 <atom:id> 元素和 <atom:link> 元素中使用。
- CICS 为 Atom 订阅源文档提供 <atom:generator> 元素，该元素提供执行生成操作的代理程序的名称，包含 URI 和版本号。
- 当 CICS 为 Atom 订阅源提供服务时，它可以采用满足 RFC 4287 中规范的格式，为每个条目生成一个 Atom 标识。按照第 226 页的『Atom 条目的 Atom 标识』中描述的条件，该 Atom 标识是唯一的，并且它的生存期与每个 Atom 条目的生存期相同。

- CICS 会根据 RFC 5023 中指定的要求，执行服务器处理客户机请求所需的操作，以处理集合中的普通 Atom 条目，但 『CICS 不支持的 Atom 功能』中特别说明的除外。值得注意的是，CICS 不支持集合中的媒体资源和媒体链接条目。
- CICS 通过 CICS 资源安全性和 CICS Web Support 安全性，使您能够实施 Atom 订阅源及包含其数据的资源的安全性。您应选择并设置适当的安全功能来保护 Atom 订阅源。RFC 5023 建议您使用认证机制，它还讨论了众多威胁案例。

CICS 不支持的 Atom 功能

对于 Atom 格式和协议的部分要求与 CICS 无关或不受 CICS 支持。

- RFC 4287 和 RFC 5023 中对于“Atom 处理器”的要求与 CICS 无关。由于 CICS 不会接收 Atom 订阅源文档进行处理，所以它不能充当 Atom 处理器。
- CICS 不支持对 Atom 文档的数字签名和加密，但是按照 RFC 4287 的要求，CICS 不会拒绝包含签名的 Atom 文档。
- CICS 不支持文本结构的 HTML 或 XHTML 内容（如 <atom:title> 元素）。您必须提供不含子元素的纯文本内容。
- CICS 允许 HTML、XHTML 和其他文本介质类型（如 XML）作为 <atom:content> 元素的内容。但是，您必须自己验证该内容。CICS 不会尝试对该内容进行语法分析来检查其标记是否有效，而是将您提供的内容原封不动地提供给客户机。
- 如果内容采用有效的 Base64 编码格式，那么您可以使用 <atom:content> 元素来提供非文本介质类型的内容。CICS 不是不接受这种内容，只是不支持非文本内容并且不检查是否存在 <atom:summary> 元素，而这个元素是 Base64 编码的内容所必需的。
- CICS 不支持集合中的介质资源和介质链接条目。Atom 发布协议（RFC 5023）指定的介质资源，主要用作组织集合中非文本内容的一种方法。如果客户机请求尝试在集合中创建的条目不是 Atom 条目（介质类型为 application/atom+xml，带/不带 type=entry 参数），那么 CICS 将通过 415 状态码来拒绝这类客户机请求。请勿在 CICS 服务文档的 <app:accept> 元素中指定任何其他介质类型。
- CICS 中的 Atom 订阅源支持旨在为客户机提供含数据的订阅源。CICS 不会提供不含数据的 Atom 条目，例如，使用“src”属性引用远程内容的条目。
- CICS 不支持条目的可选 <atom:source> 元素。在将 Atom 条目从一个订阅源复制到另一个订阅源时，可以使用 <atom:source> 元素来保存元数据。
- CICS 仅支持链接元素 <atom:link rel=″self″> 和 <atom:link rel=″edit″>。不支持其他链接元素，如 related、enclosure、via 和 alternate 链接元素。 CICS 不支持链接元素的 title 或 length 属性。
- CICS 不会拒绝集合中按类别划分的 Atom 条目。您可以在服务文档或类别文档中使用 <app:categories> 元素来指定集合中条目可接受的类别，但是 CICS 不检查客户机是否采用这些类别。
- CICS 会忽略 Slug 头，并为 Atom 条目提供其自带的 URI。
- 出于性能考虑，CICS 不会自动按集合中 Atom 条目最近一次被编辑的顺序（如条目中的 <app:edited> 元素所示）来返回这些 Atom 条目。该功能是 RFC 5023 中针对完整条目列表的一个“SHOULD”需求，但它却是针对部分条目列表的“MUST”需求。CICS 为了保持较短的响应时间而忽略了此“SHOULD”需求，但它仍然为部分列表提供了有用的功能。在使用服务例程为 CICS 提供条目时，如果希望集合符合此“SHOULD”需求并且资源可存储上一次编辑条目的日期和时间戳记，那么您可以按这个日期和时间戳记的顺序来提供条目。

- CICS 不支持 Atom 条目中的 <app:control> 和 <app:draft> 元素，并会忽略这些元素。
- 为 CICS 提供 Atom 条目数据的某些资源无法遵循 RFC 4287 中有关将 Atom 标识与 Atom 条目一起存储的建议。但是，在第 226 页的『Atom 条目的 Atom 标识』中描述的条件下，CICS 有能力在每次为一个 Atom 条目提供服务时，为该条目生成相同的 Atom 标识。
- RFC 4287 要求：当 Atom 条目在其他位置复用或移到其他位置时，Atom 标识必须与 Atom 条目存储在一起。如果将 Atom 标识与 Atom 条目存储在一起，那么可以将 Atom 条目移到其他位置，并仍然符合该要求。如果未将 Atom 标识与 Atom 条目存储在一起，请勿将 Atom 条目移到其他位置。

第 2 部分 CICS Web Support

有关规划、配置和管理 CICS Web Support 体系结构，以及编写 CICS Web Support 的应用程序和实用程序的信息。

第 4 章 配置 CICS Web Support 基本组件

所有 CICS Web Support 任务都需要 CICS Web Support 的基本组件。开始使用 CICS Web Support 前，您需要配置这些组件。

关于此任务

第 22 页的『CICS Web Support 的组件』列出了所有组件。您需要设置的基本组件是:

- TCP/IP 支持
- 对 z/OS UNIX 系统服务的访问
- SSL 支持
- 系统初始化参数

如果要在先前的 CICS 发行版中使用编码的分析器程序来引用代码页转换表 DFHCNV，那么可能需要设置某些 DFHCNV 条目。对于新的 CICS Web Support 开发，不需要代码页转换表条目。

已设置这些基本组件后，您可以使用提供的样本程序验证 CICS Web Support 的操作。

1. 按照《*CICS Transaction Server for z/OS 安装指南*》中的指示信息，为 CICS 区域启用 TCP/IP 支持。 该过程包括设置 Communications Server 和建立通过 z/OS 对域名服务器（DNS）的访问。
2. 通过将 OMVS 段包含在 CICS 区域用户标识的用户概要文件中，使 CICS 能够访问 z/OS UNIX System Services，按照授予 CICS 区域对 z/OS UNIX System Services 的访问权《*CICS Transaction Server for z/OS 安装指南*》中的指示信息操作。
3. 设置 SSL 支持，按 *CICS RACF Security Guide* 中的指示信息操作。 *CICS RACF Security Guide* 还阐述了 SSL 提供的设施。
4. 按照『为 CICS Web Support 指定系统初始化参数』中的步骤，指定合适的系统初始化参数。 某些系统初始化参数在该阶段是可选的。
5. 尽可能根据需要为 CICS Web Support 保留属于 z/OS Communications Server 的端口。 第 50 页的『保留 CICS Web Support 的端口』具有有关该内容的更多信息。
6. 可选: 如果您在 CICS Web Support 中有现有请求处理结构，而这些结构包括引用转换表（DFHCNV）中条目的分析器程序，并且您希望继续使用这些未更改的请求处理结构，那么第 51 页的『升级代码页转换表（DFHCNV）中的条目』会介绍如何操作。 对于任何其他 CICS Web Support 任务，不需要代码页转换表。
7. 使用提供的样本验证 CICS Web Support 的操作。第 51 页的『验证 CICS Web Support 的操作』介绍了如何执行该操作。

为 CICS Web Support 指定系统初始化参数

您需要指定某些系统初始化参数来启用 CICS Web Support。

关于此任务

1. 必需: 指定 `TCPIP=YES` 以激活 CICS TCP/IP 服务。 缺省设置为 NO。必须指定 YES 才能启用 CICS Web Support。

2. 使用 **LOCALCCSID** 系统初始化参数为本地 CICS 区域指定编码字符集标识。 这是 CICS 认为是应用程序的缺省值的代码页。缺省值是 EBCDIC 代码页 037。如果未选择备用代码页转换选项，那么 CICS 将入站 HTTP 请求的数据内容传递到应用程序之前将它转换为该代码页，并将假定应用程序已用该代码页提供 HTTP 响应。
3. 如果您正在规划使用 CICS 文档模板支持，那么对 HTTP 请求提供静态响应，或作为应用程序生成的响应的一部分，使用 **DOCCODEPAGE** 系统初始化参数为文档域指定缺省主机代码页。缺省值是 EBCDIC 代码页 037。
4. 如果您正在规划对 3270 显示应用程序提供 Web 客户机访问，那么使用 **WEBDELAY** 系统初始化参数指定合适的超时周期。 两个超时周期控制:
   - 以分钟为单位的时间长度，如果 Web 任务及其相关数据在这段时间内没有发生活动，那么将它们标记为删除。缺省值是 5 分钟。
   - 以分钟为单位的频率，以该频率运行无用信息收集事务 CWBG 以删除标记的任务及其数据。缺省值为 60 分钟。

   WEBDELAY 不应用到任何其他不涉及 3270 显示应用程序的 CICS Web Support 任务。
5. 如果您想要对 CICS Web Support 使用安全性，那么还需要设置下列额外的系统初始化参数的值:
   - **CRLSERVER**
   - **ENCRYPTION**
   - **KEYRING**
   - **MAXSSLTCBS**
   - **SSLCACHE**

   CICS RACF Security Guide 说明如何使 SSL 与 CICS 一起运作，包括指定这些系统初始化参数。

保留 CICS Web Support 的端口

建议您根据需要为 CICS Web Support 保留尽可能多的属于 z/OS Communications Server 的端口，并确保 CICS Web Support 需要时可以独占这些端口。

- 对于 HTTP，熟知（或缺省）端口号是 80，而对于 HTTPS，熟知端口号是 443。 您应该在可能使用熟知端口的同一 IP 地址上小心地解决与任何其他服务器的冲突。
- 对于非标准的服务器，应用程序员可以使用从 1024 到 32767 的端口号。 1024 以下的端口是由 IANA 为了特殊函数构建的熟知端口号，因此需要 HTTP 端口 80 和 HTTPS 端口 443，它们不应该用于 CICS Web Support。 SSL 和非 SSL 请求必须使用不同的端口。
- 要保留 CICS Web Support 侦听入站客户机请求的端口，您可以在 PROFILE.TCPIP 数据集中指定 PORT 语句或 CICS 作业名，如 *z/OS Communications Server: IP Configuration Reference*，SC31-8776 中所描述。
- 程序正在侦听的 TCP/IP 端口的任何请求队列的最大长度是由 PROFILE.TCPIP 数据集中的 SOMAXCONN 参数控制的。CICS 侦听 TCP/IP 端口，因此您必须协调该参数的值与为 TCPIPSERVICE 定义中 BACKLOG 参数选择的值。

升级代码页转换表（DFHCNV）中的条目

在 CICS Transaction Server for z/OS V4R1 中，对于 CICS Web Support，通常不需要代码页转换表。然而，如果您有现有请求处理结构，而这些结构包括引用转换表中的条目的分析器程序，并且您不希望更改分析器程序，那么可继续提供 DFHCNV 条目。

关于此任务

在 CICS Transaction Server for z/OS V3R1 之前的 CICS 发行版中，代码页转换表（DFHCNV）用于定义 CICS 中使用的代码页和 Web 客户机使用的 ASCII 代码页之间的代码页转换。对于 CICS Transaction Server for z/OS V4R1 中的 CICS Web Support，您不需要在代码页转换表中创建任何新条目。CICS Web Support 使用 z/OS 转换服务处理代码页转换。

然而，如果您要继续使用先前 CICS 发行版中编码的分析器程序来引用 DFHCNV，那么必须继续提供代码页转换表中的条目，或更改分析器程序。更改分析器程序涉及对两个指定客户机和服务器代码页的新输出参数进行编码，以替换指定 DFHCNV 条目名称的输出参数。如果这样做，那么不需要升级 DFHCNV 条目。第 113 页的『编写分析器程序』介绍了如何用这种方式对输出参数进行编码。

注：所提供的 CICS 提供的样本分析器 DFHWBADX 指定样本代码页转换表 DFHCNVW$ 中定义的条目。无需进行任何配置就可以使用样本转换表，但是您可能宁愿修改 DFHWBADX 以使用新的输出参数，从而提供更好的控制并避免使用样本转换表。

如果您愿意继续使用 DFHCNV，那么遵循这些步骤操作:

1. 在先前 CICS 发行版中查找您要用于定义转换表的 DFHCNV 资源定义宏的源。 这些宏的顺序应包括每个代码页对的 DFHCNV TYPE=ENTRY 宏。
2. 按照 *CICS Intercommunication Guide* 中描述的过程，使用这些宏来设置 DFHCNV 转换表。 您需要定义、汇编和链接编辑该表。

验证 CICS Web Support 的操作

样本程序 DFH$WB1A（汇编程序）和 DFH$WB1C（C）可帮助您测试 CICS Web Support 是否运行正常。样本程序使用 EXEC CICS WEB 和 DOCUMENT 命令来接收您的请求，并且构建和发送一个简单的响应。

关于此任务

可以使用 CICS 提供的样本分析器程序 DFHWBADX 来访问 DFH$WB1A。可以使用提供的样本 URIMAP 定义 DFH$URI1 或样本分析器程序来访问 DFH$WB1C。如果计划使用 CICS 作为 HTTP 客户机，请注意用于传送客户机请求的 CICS 提供的样本程序可用于安装了 DFH$WB1C 和 DFH$URI1 的 CICS 区域，因此您需要选择该选项。

样本程序按如下方式构造 HTTP 响应:

```
DFH$WB1A on system applid successfully invoked through CICS Web Support
```

其中 *applid* 是正在运行 CICS Web Support 的 CICS 系统的应用程序标识。

要运行样本程序:

1. 根据需要进行修改，然后安装组 DFH$SOT 中提供的样本 TCPIPSERVICE 定义 HTTPNSSL。在 TCPIPSERVICE 定义中指定 CICS 提供的样本分析器程序 DFHWBADX。您可能需要更改下列选项:
   a. **PORTNUMBER**: HTTPNSSL 使用端口 80 – 最常用的 HTTP 端口号。如果未保留端口 80 供 CICS 使用，请在 z/OS Communications Server 所含的其他端口中指定一个 CICS 专用端口。
   b. **HOST** 或 **IPADDRESS**: 由于 HTTPNSSL 不指定 IP 地址，所以该选项的缺省值通常为缺省 z/OS Communications Server TCP/IP 协议集对应的 IP 地址。这种设置最为常见。如果 z/OS 映像中有多个 TCP/IP 协议集，而您想要使用某个非缺省的协议集，请指定该协议集所对应的 IP 地址。
2. 如果要使用样本程序 DFH$WB1C，请安装 DFH$WEB 资源定义组中提供的该程序的 PROGRAM 资源定义。 DFH$WB1A 的 PROGRAM 资源定义位于 DFHWEB 资源定义组中，已将其作为 DFHLIST 的一部分来安装。
3. 如果您正在使用样本程序 DFH$WB1C，并且希望使用 URIMAP 定义，请安装 DFH$WEB 资源定义组中提供的样本 URIMAP 定义 DFH$URI1。
4. 在 Web 浏览器上，使用以下 URL 组成部分输入连接到 CICS Web Support 的 URL:

   Scheme

   HTTP

   Host 指定到 z/OS 映像的主机名。如果您不知道该主机名，那么可以使用 HTTPNSSL TCPIPSERVICE 定义中的点分十进制 IP 地址。如果您未明确指定 IP 地址，它将由 CICS 填写，您可以在已安装的 TCPIPSERVICE 定义中查看该地址。

   Port number

   在 TCPIPSERVICE 定义中指定的端口号。如果为 80，那么您无需指定。

   Path

   - 要访问 DFH$WB1A，请使用路径 `/CICS/CWBA/DFH$WB1A`
   - 要访问 DFH$WB1C，如果您已安装了样本 URIMAP 定义，请使用路径 `/sample_web_app`；如果您希望使用样本分析器程序，那么使用路径 `/CICS/CWBA/DFH$WB1C`。
5. 如果您现在不打算进一步执行测试，请卸载样本 TCPIPSERVICE 定义 HTTPNSSL，并禁用 URIMAP 定义 DFH$URI1。 以后，您可以使用自己正确构建的 TCPIPSERVICE 定义来替换 HTTPNSSL。

相关任务

第 138 页的『样本程序: 将请求通过管道传送给 HTTP 服务器』
样本程序 DFH$WBPA（汇编程序）、DFH$WBPC（C）和 DFH0WBPO （COBOL）演示了 CICS 如何通过管道将客户机请求传送到 HTTP 服务器。

配置 HTTP TRACE 方法

缺省情况下，所有 HTTP TRACE 请求接收 HTTP 200（确定）响应。您可以覆盖该设置以禁用跟踪。

关于此任务

您将更改 HTTP TRACE 请求设置，使得 HTTP TRACE 请求接收 HTTP 501（未实施）响应。

创建一个只有汇编程序数据的模块 DFHWBMTH，该模块包含一个半字后跟 7 个字节的字符“NOTRACE”字段。

示例

以下是帮助您创建模块的示例:

```
//DFHWBMTH  JOB   'accounting info',name,MSGCLASS=A
      //ASM      EXEC PGM=ASMA90,REGION=2048K,
      //            PARM=(DECK,NOOBJECT,ALIGN)
      //SYSPRINT DD SYSOUT=*
      //SYSLIB   DD DSN=SYS1.MACLIB,DISP=SHR
      //SYSUT1   DD SPACE=(CYL,(3,2))
      //SYSUT2   DD SPACE=(CYL,(1,1))
      //SYSUT3   DD SPACE=(CYL,(1,1))
      //SYSPUNCH DD DSN=&&OBJMOD,DISP=(,PASS),
//             SPACE=(CYL,(1,1)),
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=400)
//SYSIN    DD DATA,DLM='@@'
DFHWBMTH CSECT
DFHWBMTH AMODE 31
DFHWBMTH RMODE ANY
LENGTH   DC AL2(ENDDATA-*)
OPTIONS  DC CL7'NOTRACE'
ENDDATA  EQU *
         END     DFHWBMTH
@@
//LKED     EXEC PGM=IEWL,REGION=2048K,
//            PARM=(LIST,XREF),
//            COND=(7,LT)
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD SPACE=(CYL,(1,1))
//SYSLIN   DD DSN=&&OBJMOD,DISP=(OLD,DELETE)
//         DD DDNAME=SYSIN
//SYSLMOD  DD DSN=CICS.DFHRPL,DISP=SHR
//SYSIN    DD DATA,DLM='@@'
    MODE AMODE(31),RMODE(ANY)
    NAME DFHWBMTH(R)
@@
```

第 5 章 规划作为 HTTP 服务器的 CICS 的 CICS Web Support 体系结构

根据您希望作为 HTTP 服务器的 CICS 执行的任务不同，它的 CICS Web Support 体系结构也不同。CICS Web Support 的某些可配置组件对于所有任务来说都是必需的，例如，对接收入站请求的端口的 TCPIPSERVICE 定义。其他可配置组件仅对特定任务来说是必需的。

开始之前

开始规划作为 HTTP 服务器的 CICS 的 CICS Web Support 体系结构前，请阅读第 25 页的『作为 HTTP 服务器的 CICS 的 HTTP 请求和响应处理』主题，以了解可能涉及的处理阶段。

- 使用特殊设计的支持 Web 的 CICS 应用程序，您可以对 HTTP 请求提供动态、应用程序生成的响应。 『用支持 Web 的应用程序提供动态 HTTP 响应』介绍了如何执行该操作。
- 使用可用于 CICS 的文档或文件，您可以对 HTTP 请求提供静态响应。 第 59 页的『用 CICS 文档模板或 z/OS UNIX 文件提供静态 HTTP 响应』介绍了如何执行该操作。
- 您可以允许 Web 客户机使用 HTTP 访问 CICS 中的现有通信区域应用程序。 第 63 页的『提供对通信区域应用程序的 Web 客户机访问』介绍了如何执行该操作。
- 您可以允许 Web 客户机使用 HTTP 访问 CICS 中的现有 3270 显示应用程序。 第 169 页的第 15 章，『CICS Web Support 和 3270 显示应用程序』介绍了如何执行该操作。
- 您可以从客户机接收非 HTTP 请求，并提供应用程序生成的响应。 第 163 页的第 14 章，『CICS Web Support 和非 HTTP 请求』介绍了如何执行该操作。

用支持 Web 的应用程序提供动态 HTTP 响应

您可以使用支持 Web 的应用程序对来自 Web 客户机的 HTTP 请求提供应用程序生成的响应。

开始之前

必须在开始前配置 CICS Web Support 的基本组件，如第 49 页的第 4 章，『配置 CICS Web Support 基本组件』中所描述。

关于此任务

以下 CICS Web Support 的特定于任务的组件用于该任务:

- TCPIPSERVICE 资源定义
- URIMAP 资源定义
- 支持 web 的应用程序，使用 EXEC CICS WEB 编程接口
- 应用程序的别名事务

- 分析器程序
- 安全设施
- Web 出错程序

第 57 页的图 4 介绍了该 CICS Web Support 任务的体系结构元素。第 25 页的『作为 HTTP 服务器的 CICS 的 HTTP 请求和响应处理』说明了这些过程元素如何在一起工作。

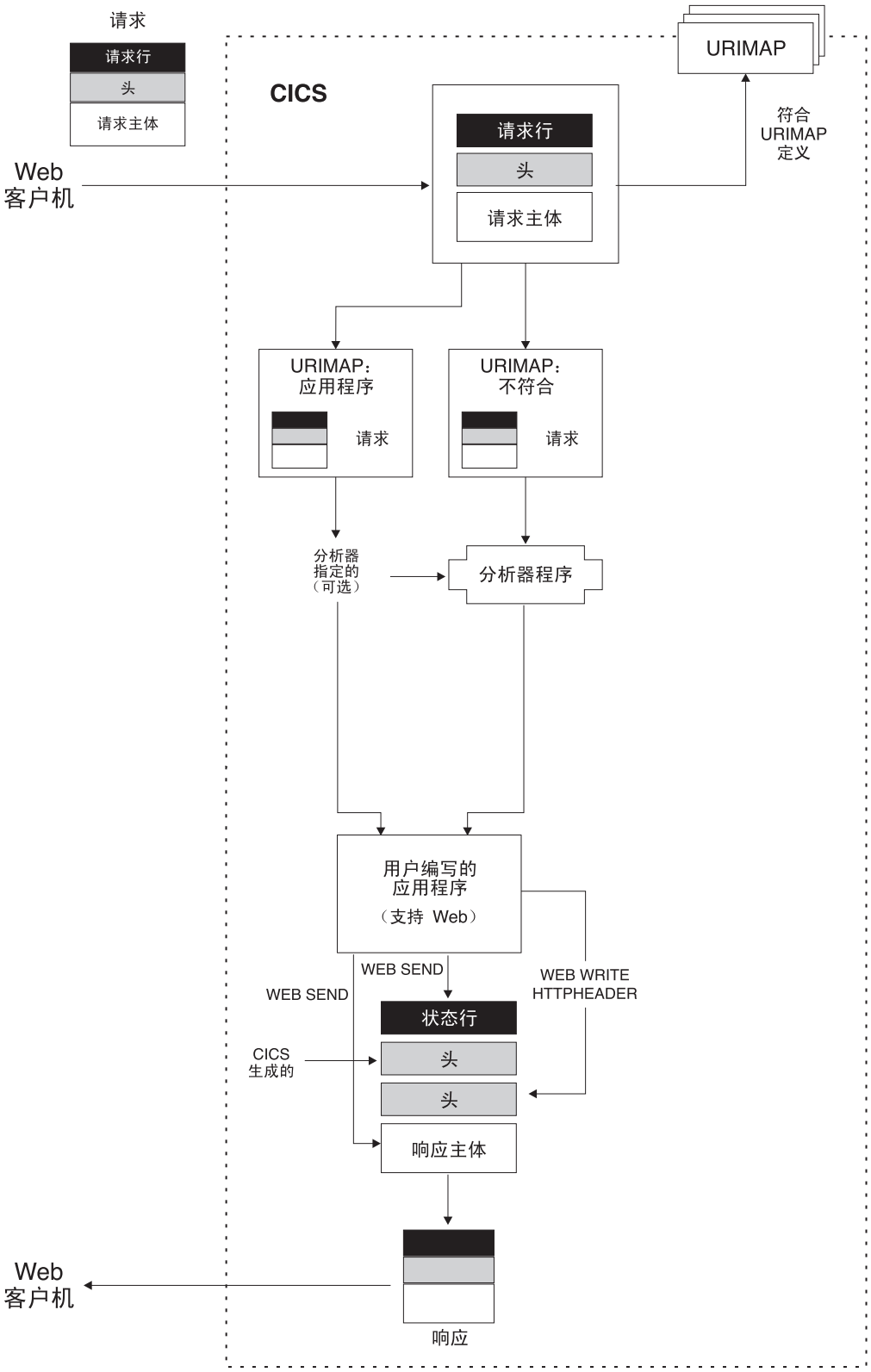


图 4. 具有支持 Web 的应用程序的动态 HTTP 响应

1. 使用，使用第 67 页的第 6 章，『为作为 HTTP 服务器的 CICS 编写支持 Web 的应用程序』中的信息，设计和编码一个或多个支持 Web 的应用程序以对 Web 客户机的每个请求提供响应。 支持 Web 的应用程序可以使用 EXEC CICS WEB 和

EXEC CICS DOCUMENT 命令以接收和分析 HTTP 请求，并写响应以及对请求发送响应。每个程序处理单个请求和响应。

注: 使用 EXEC CICS WEB 命令的支持 Web 的应用程序必须在接收 Web 客户机请求的 CICS 区域中运行。然而，它们可以链接到其他 CICS 区域中的应用程序（例如，执行业务逻辑）。

2. 为该 CICS Web Support 任务考虑安全问题。 CICS 可以为连接实施 HTTP 基本认证，其中用户必须提供标识和密码。对于应用程序生成的响应或静态响应所用的各项资源，您可以使用该用户标识来控制对这些资源的访问。 如果您需要保护在因特网上传递的信息（包括用于基本认证的用户标识和密码），那么考虑使用安全套接字层（SSL）。要获取更多信息，请参阅 第 151 页的第 13 章，『CICS Web Support 的安全性』。
3. 决定 Web 客户机将用于每个请求的 URL，包括方案、主机和路径部分以及任何查询字符串。 第 10 页的『URL 的组成部分』说明了这些组成部分以及如何对它们定界。第 32 页的『CICS Web Support 的 URL』 说明了为 CICS Web Support 选择 URL 要考虑的因素和限制。
4. 决定将用于请求的端口。 第 50 页的『保留 CICS Web Support 的端口』有关可由 CICS Web Support 使用的端口的更多信息。 对于 HTTP，缺省端口号是 80，而对于 HTTPS（具有 SSL），缺省端口号是 443。需要在请求的 URL 中明确指定不是方案的缺省值的端口号。 如果您愿意，可以允许请求使用与 CICS Web Support 关联的任何端口。
5. 如果接收请求的端口尚未有 TCPIPSERVICE 定义，那么按第 83 页的『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』中的过程操作以设置一个 TCPIPSERVICE 定义。 使用该定义来为端口指定安全性措施（如，使用 SSL 和基本认证）。 相关 TCPIPSERVICE 定义的名称在请求的 URIMAP 定义中指定。不指定 TCPIPSERVICE 定义意味着 URIMAP 定义匹配的请求可以使用存在 TCPIPSERVICE 定义的任何端口。
6. 为涉及本任务的用户应用程序选择别名事务标识。缺省别名事务是 CWBA。按第 86 页的『为 CICS Web Support 创建 TRANSACTION 资源定义』中的指示信息进行操作，您可以创建您自己的别名事务。 您可以使用 URIMAP 定义或分析器程序为每个 HTTP 请求指定别名事务。 如果正在使用 Web 客户机的用户标识执行资源级安全性，那么用户标识用于该事务，并且需要有权访问该事务使用的受保护 CICS 资源和命令。
7. 确定应如何使用与 TCPIPSERVICE 定义关联的分析器程序，并选择合适的程序。第 111 页的第 10 章，『分析器程序』提供了关于使用分析器程序可执行的操作的更多信息。 对于支持 web 的应用程序，您可从以下策略中选择:
   a. 使用 CICS 提供的缺省分析器程序 DFHWBAAX 提供错误句柄。DFHWBAAX 适合于这种情况: 使用该端口的所有请求是使用 URIMAP 定义处理的。 如果找到匹配 URIMAP 定义，那么 DFHWBAAX 不执行操作。如果找不到匹配，那么它控制用户可替换 Web 出错程序 DFHWBERX 以生成错误响应。
   b. 对于使用 URIMAP 定义的请求和按照 CICS TS 3.1 之前的 CICS Web Support 使用的同一过程的请求，使用 CICS 提供的样本分析器程序 DFHWBADX 为它们提供基本支持。 如果找到匹配 URIMAP 定义，那么 DFHWBADX 不执行操作。如果找不到匹配，那么它用 CICS TS 3.1 之前要求的格式分析 URL。如果分析失败，那么 DFHWBADX 控制用户可替换 Web 出错程序

DFHWBEP 来生成错误响应。(如果您的 URIMAP 定义中指定的 URL 不适合 CICS TS 3.1 之前要求的格式，那么您应定制 DFHWBADX 或 DFHWBEP 以便提供更有意义的响应。)

c. 使用您自己的分析器程序提供定制支持。 这可能包括:

- 对使用 URIMAP 定义的请求的处理进行动态更改。
- 请求处理期间提供监控或审计操作。
- 支持按照 CICS TS 3.1 之前 CICS Web Support 使用的同一过程的请求。
- 使用用户可替换的 Web 出错程序 DFHWBEP 和/或 DFHWBERX 提供错误响应。

可以使用 URIMAP 定义中的 ANALYZER(YES) 属性指定定制分析器程序，并在处理请求时使用该程序。如前所述，如果从 URIMAP 定义调用 DFHWBAAX 和 DFHWBADX，那么它们不执行操作。

8. 对于 HTTP 请求以及用户应用程序对它们提供的响应，决定希望如何执行代码页转换。 代码页转换通常由以下操作组成: 将使用 ASCII 字符集发出的 Web 客户机请求转换为 EBCDIC 代码页，以供应用程序使用; 然后颠倒该过程以将应用程序的输出返回给 Web 客户机。第 37 页的『CICS Web Support 的代码页转换』说明了代码页转换的过程。 您可以在支持 Web 的应用程序发出的 EXEC CICS WEB API 命令中指定代码页转换设置。
9. 设置 URIMAP 定义以处理每个请求。按第 87 页的『为作为 HTTP 服务器的 CICS 的任何请求启动 URIMAP 资源定义』和第 89 页的『为作为 HTTP 服务器的 CICS 的 HTTP 请求的应用程序响应完成 URIMAP 定义』中的过程进行操作。 按照 CICS TS 3.1 之前的 CICS Web Support 使用的同一过程，可以将 HTTP 请求直接传递到分析器程序而无需使用 URIMAP 定义。然而，使用 URIMAP 定义可以更容易地管理 HTTP 请求。没有 URIMAP 定义的情况下，如果要更改 CICS 对特殊 HTTP 请求响应的方式，那么需要更改分析器程序中的逻辑。使用 URIMAP 定义，您可以将这些更改作为系统管理任务动态执行。
10. 确保您的体系结构中包括的用户可替换 Web 出错程序向 Web 客户机提供适当的响应。 如果 CICS Web Support 进程中发生初始处理错误、异常终止或故障情况，就需要使用 Web 出错程序。并不是在所有错误情况下都可以使用它们。第 103 页的第 9 章，『Web 出错程序』说明了使用 DFHWBEP 和 DFHWBERX 的情形，并介绍如何定制它们提供的响应。

用 CICS 文档模板或 z/OS UNIX 文件提供静态 HTTP 响应

您可以使用 CICS 文档模板或 z/OS UNIX 系统服务文件为来自 Web 客户机的 HTTP 请求提供静态响应。

开始之前

必须在开始前配置 CICS Web Support 的基本组件，如第 49 页的第 4 章，『配置 CICS Web Support 基本组件』中所述。

关于此任务

以下 CICS Web Support 的特定于任务的组件用于该任务:

- TCPIPSERVICE 资源定义
- URIMAP 资源定义

- z/OS UNIX 文件
- CICS 文档模板支持
- 安全设施
- Web 出错程序

图 5 说明该 CICS Web Support 任务的体系结构元素。第 25 页的『作为 HTTP 服务器的 CICS 的 HTTP 请求和响应处理』说明了这些过程元素如何在一起工作。

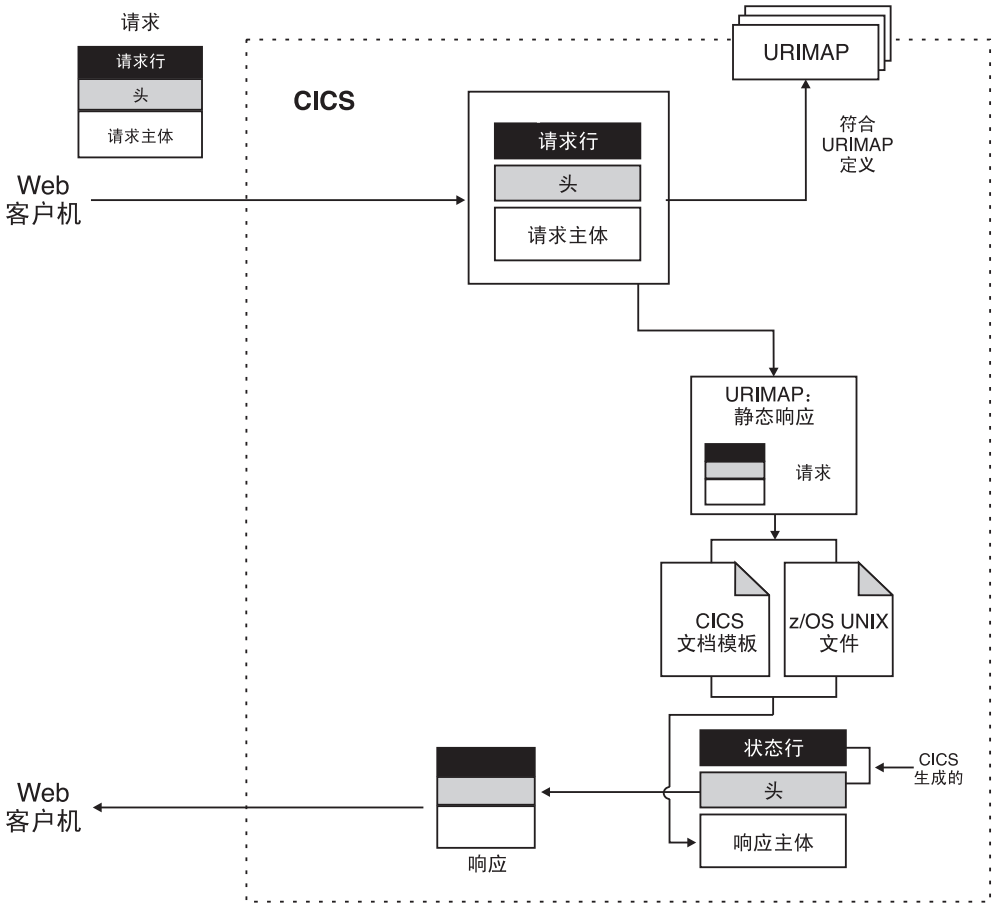


图 5. 静态 HTTP 响应

1. 为该 CICS Web Support 任务考虑安全问题。 CICS 可以为连接实施 HTTP 基本认证，其中用户必须提供标识和密码。对于应用程序生成的响应或静态响应所用的各项资源，您可以使用该用户标识来控制对这些资源的访问。 如果您需要保护在因特网上传递的信息（包括用于基本认证的用户标识和密码），那么考虑使用安全套接字层（SSL）。要获取更多信息，请参阅 第 151 页的第 13 章，『CICS Web Support 的安全性』。
2. 如果要使用 z/OS UNIX 系统服务文件来提供响应，请创建文件并将它放在您在 z/OS UNIX 文件系统中选择的适当位置。 标识该响应时，CICS 检索该文件并执行适当的代码页转换。
   a. 不要在 z/OS UNIX 文件中包含任何 HTTP 头或状态行信息。 发送响应后，CICS 生成必需的信息。z/OS UNIX 文件只提供响应主体。

b. 如果要使用路径匹配，那么文件位置很重要，如本主题中稍后所述。如果不想使用路径匹配，那么文件位置不需要与请求的 URL 有任何关系。

c. CICS 区域必须有访问 z/OS UNIX 的许可权，而且它必须具有访问包含文件的目录以及文件本身的许可权。 *Java Applications in CICS* 阐述如何授予这些许可权。

d. 如果正在使用 Web 客户机的用户标识执行访问控制，这些用户标识还必须有权访问包含文件的目录以及文件本身。 第 155 页的『CICS Web Support 的 CICS 系统和资源安全性』提供了更多信息。

3. 如果希望使用 CICS 文档模板来提供响应，请按照 *CICS Application Programming Guide* 中的指示信息来创建文档模板。 文档模板是使用 DOCTEMPLATE 资源定义来定义的。文档模板可以保存在以下各项中：分区数据集、CICS 程序、文件、临时存储队列、瞬时数据队列、出口程序或 z/OS UNIX 系统服务文件。 标识该响应时，CICS 使用模板创建文档、检索该文档，并执行适当的代码页转换。

   a. 不要在文档模板中包含任何 HTTP 头或状态行信息。 发送响应后，CICS 生成必需的信息。文档模板只提供响应主体。

   b. 组成名称值对的查询字符串可用作符号列表并替换成文档模板。 （如果查询字符串已用于 URIMAP 匹配，作为 URIMAP 定义的 PATH 属性的一部分，那么不能这样使用它。） 使客户机发送 URL 中期望格式的查询字符串的建议方式是：发送具有 GET 方法的 HTML 表单以供用户填写。 查询字符串中的任何名称都可在文档模板中作为符号编码，并且使用模板后，CICS 用每个符号替代在查询字符串中指定的值。例如，如果已获取包括名称和值对 `username=Peter` 的查询字符串，那么可以通过将 `username` 编码为符号以在文档模板中使用该查询字符串：

   ```
   欢迎使用财务系统，&username;。
   ```

   传递到用户的结果静态响应将读到

   ```
   欢迎使用财务系统，Peter。
   ```

   注： 文档模板中的符号是区分大小写的。将名称的大小写指定为与原始查询字符串中出现的大小写一样。
   与文档模板中的符号不符合的任何名称和值对都会被忽略。

   c. 如果正在使用 Web 客户机的用户标识执行资源级安全性，那么用户标识必须有权访问文档模板。 第 155 页的『CICS Web Support 的 CICS 系统和资源安全性』说明了如何授权。请注意，如果文档模板是 z/OS UNIX 系统服务文件，那么 Web 客户机不需要获得对该文件的许可权，而只需获得对 DOCTEMPLATE 资源定义的许可权。

4. 识别 z/OS UNIX 文件或 CICS 文档模板提供的介质类型（数据内容的类型）。 请参阅 第 10 页的『IANA 介质类型和字符集』，以获取有关介质类型的更多信息。请注意，使用 URIMAP 定义发送静态响应时，不支持使用质量指标（“q”参数）。质量指标可用于在客户机的可接受介质类型或字符集列表中选择，如 Accept 头中所指定。如果要执行这种类型的分析，那么可以代之以使用应用程序生成的响应。

5. 识别 CICS 做出静态响应的代码页转换所需要的信息。 只有在指定文本介质类型时才执行代码页转换。第 37 页的『CICS Web Support 的代码页转换』说明了代码页转换的过程。

a. 将静态响应发送到 Web 客户机前，识别 CICS 应该将该静态响应转换到的字符集。 CICS 为进行代码页转换支持的 IANA 字符集在 第 333 页的附录 A, 『HTML 编码字符集』中列出。

b. 识别对提供响应主体的文档模板或 z/OS UNIX 文件进行编码时采用的 IBM 代码页（EBCDIC）。

对于静态响应，该信息在请求的 URIMAP 定义中指定。

6. 决定 Web 客户机将用于每个请求的 URL，包括方案、主机和路径部分以及任何查询字符串。 第 10 页的『URL 的组成部分』说明了这些组成部分以及如何对它们定界。第 32 页的『CICS Web Support 的 URL』 说明了为 CICS Web Support 选择 URL 要考虑的因素和限制。

7. 决定是否要在 URIMAP 定义中使用路径匹配。如果是，那么安排好您的请求 URL，并调整 CICS 文档模板名称或 z/OS UNIX 文件的位置，以支持该操作。 在路径匹配中，通配符用在 URIMAP 定义的路径部分，也用在 URIMAP 定义指定的 CICS 文档模板或 z/OS UNIX 文件的名称中。通配符所代表的部分路径用于选择文档模板或 z/OS UNIX 文件以提供响应。

   a. 对于 CICS 文档模板，由通配符覆盖的路径部分替换为模板名的最后一部分。您可以创建名称以相同方式开头的文档模板集合，并通过单个 URIMAP 定义，使用路径以相同方式开头的请求 URL 来访问它们。

   b. 对于 z/OS UNIX 文件，由通配符代表的路径部分将替换为文件名的最后一部分。您可以将其中一些文件存储在同一目录中，并通过单个 URIMAP 定义，使用路径开始部分相同的请求 URL 来访问它们。 请记住，因为 URIMAP 定义必须指定一种数据内容类型（MEDIATYPE 属性），因此单个 URIMAP 定义只能处理生成同一类型数据内容的一组 z/OS UNIX 文件。

8. 决定将用于请求的端口。 第 50 页的『保留 CICS Web Support 的端口』有关可由 CICS Web Support 使用的端口的更多信息。 对于 HTTP，缺省端口号是 80，而对于 HTTPS（具有 SSL），缺省端口号是 443。需要在请求的 URL 中明确指定不是方案的缺省值的端口号。 如果您愿意，可以允许请求使用与 CICS Web Support 关联的任何端口。

9. 如果接收请求的端口尚未有 TCPIPSERVICE 定义，那么按第 83 页的『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』中的过程操作以设置一个 TCPIPSERVICE 定义。 使用该定义来为端口指定安全性措施（如，使用 SSL 和基本认证）。 相关 TCPIPSERVICE 定义的名称在请求的 URIMAP 定义中指定。不指定 TCPIPSERVICE 定义意味着 URIMAP 定义匹配的请求可以使用存在 TCPIPSERVICE 定义的任何端口。

10. 设置 URIMAP 定义以处理每个请求。按第 87 页的『为作为 HTTP 服务器的 CICS 的任何请求启动 URIMAP 资源定义』和第 90 页的『为作为 HTTP 服务器的 CICS 的 HTTP 请求的静态响应完成 URIMAP 定义』中的过程进行操作。 URIMAP 定义可以识别 z/OS UNIX 文件或文档模板。

11. 为该 CICS Web Support 任务检查错误句柄过程。

    a. 检查与 TCPIPSERVICE 定义关联的分析器程序行为，以了解在哪个端口上接收请求。如果为请求的 URIMAP 匹配失败，那么请求被传递到分析器程序。 如果该端口只用于静态响应，那么 CICS 提供的缺省分析器程序 DFHWBAAX 将提供合适的错误处理。另外，分析器程序的选择可能取决于用户应用程序的需求，并且您可能需要定制它以为静态响应提供合适的错误处理。第 111 页的第 10 章，『分析器程序』提供了关于使用分析器程序可执行的操作的更多信息。

b. 确保您的体系结构中包括的用户可替换 Web 出错程序向 Web 客户机提供适当的响应。 第 103 页的第 9 章，『Web 出错程序』说明了应用 DFHWBEP 和 DFHWBERX 的情形，并介绍如何定制它们提供的响应。

相关信息

授予 CICS 区域访问 HFS 目录和文件的许可权

z/OS UNIX 的 CICS Web Support 资源

当使用 z/OS UNIX 文件对 Web 客户机发出的请求提供静态响应时，接收这些请求并提供响应的 CICS 区域需要具有这些文件及其目录的**读**访问权。

如果将每个 CICS 区域的所有相关文件都存储在该 CICS 区域的主目录下的目录结构中，那么可以使该 CICS 区域成为各个目录和文件的所有者（具有合适的所有者权限）。如果某些 z/OS UNIX 文件被多个 CICS 区域使用，那么需要使用其他可能的解决方案，例如，组许可权或访问控制表（ACL）。使用『其他』许可权会对每个 z/OS UNIX 用户授予访问权，因此，建议不要用于生产环境中的 CICS Web Support。

相关信息

授予 CICS 区域访问 HFS 目录和文件的许可权

提供对通信区域应用程序的 Web 客户机访问

您可以使用 CICS Web Support 以允许 Web 客户机与 CICS 应用程序交互，这些应用程序使用通信区域接口与其他程序通信。您可以写支持 Web 的应用程序，它会链接到应用程序并使用其输出提供HTTP 响应。或者，您可以使用转换器程序将 Web 客户机的输入转换为合适的 COMMAREA，并将应用程序的输出转换为 HTTP 响应。

开始之前

必须在开始前配置 CICS Web Support 的基本组件，如第 49 页的第 4 章，『配置 CICS Web Support 基本组件』中所述。

关于此任务

以下 CICS Web Support 的特定于任务的组件用于该任务:

- TCPIPSERVICE 资源定义
- URIMAP 资源定义
- 通信区域应用程序
- 要么是支持 Web 的应用程序，它们使用 EXEC CICS WEB 编程接口链接到通信区域应用程序并使用这些应用程序的输出
- 要么是转换器程序，它们可以提供合适的 COMMAREA 输入，并将应用程序的输出转换为 HTTP 响应
- 包括此进程中涉及的用户应用程序的别名事务
- 分析器程序
- 安全设施
- Web 出错程序

图 6 显示该 CICS Web Support 任务的体系结构元素。第 25 页的『作为 HTTP 服务器的 CICS 的 HTTP 请求和响应处理』说明了这些过程元素如何在一起工作。

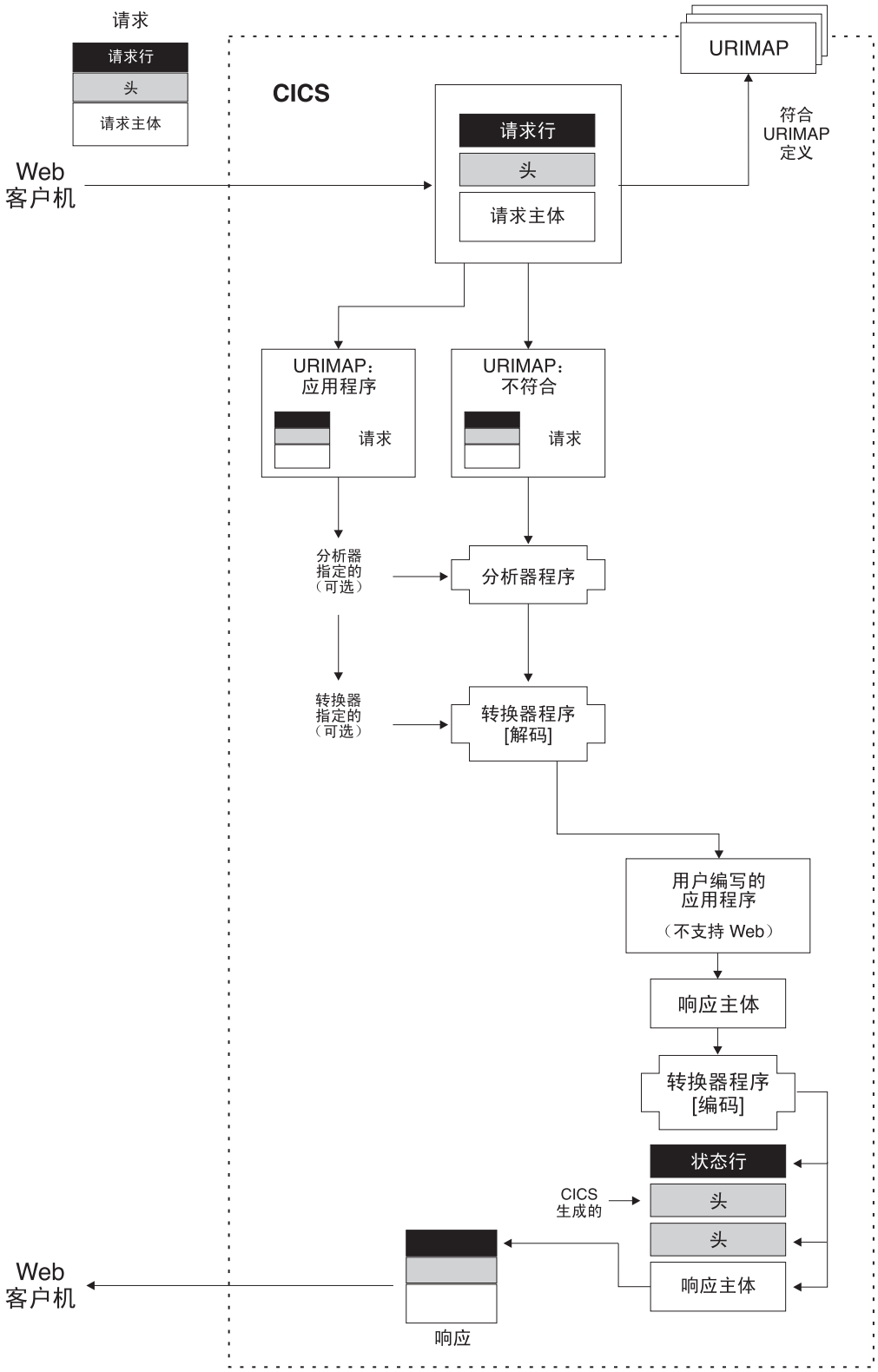


图 6. 来自通信区域应用程序的 HTTP 响应

1. 决定是写支持 Web 的应用程序，该应用程序处理 HTTP 请求并链接到 COMMAREA 应用程序；还是写转换器程序以将 Web 客户机的输入转换为合适的 COMMAREA，并将应用程序的输出转换为 HTTP 响应。转换器程序可以使用 EXEC CICS WEB API 命令来读取 Web 客户机的请求并生成响应，或使用存储器块中的请求和响应。

   a. 如果要使用支持 Web 的应用程序，那么按第 55 页的『用支持 Web 的应用程序提供动态 HTTP 响应』中的步骤进行操作。编写支持 Web 的应用程序的代码以链接到通信区域应用程序并使用其输出。 唯一不能由支持 Web 的应用程序执行，但能由转换器程序执行的任务是：接收分析器程序已创建的、要传递到下一个处理阶段（在用户令牌或共享工作区中）的信息。开发新的 CICS Web Support 应用程序时，可能不需要考虑这一点。

   b. 如果要使用转换器程序，那么按本主题中的步骤进行操作。

2. 为该 CICS Web Support 任务考虑安全问题。 CICS 可以为连接实施 HTTP 基本认证，其中用户必须提供标识和密码。对于应用程序生成的响应或静态响应所用的各项资源，您可以使用该用户标识来控制对这些资源的访问。 如果您需要保护在因特网上传递的信息（包括用于基本认证的用户标识和密码），那么考虑使用安全套接字层（SSL）。要获取更多信息，请参阅 第 151 页的第 13 章，『CICS Web Support 的安全性』。

3. 确定应如何使用与 TCPIPSERVICE 定义关联的分析器程序，并选择合适的程序。第 111 页的第 10 章，『分析器程序』提供了有关使用分析器程序可执行的操作的更多信息。 URIMAP 定义或分析器程序可用于将请求从 Web 客户机映射到适当的转换器程序和用户编写的应用程序。 对于不支持 Web 的应用程序，即使您具有 URIMAP 定义，在以下这些情况中，您也需要在处理请求时使用定制分析器程序:

   a. 如果转换器程序使用存储器块中的请求和响应，那么需要进行非标准的代码页转换。 转换器程序没有为包含 HTTP 请求和响应的存储器块指定代码页转换的机制。当缺少分析器程序时，CICS 使用第 124 页的『编写转换器程序』中描述的标准设置，来对存储器块中输入和输出提供的消息体进行转换。如果此操作不适合，那么需要在处理路径中使用分析器程序来指定备用设置，或使用 EXEC CICS WEB API 命令来代替存储器块的使用。第 37 页的『CICS Web Support 的代码页转换』提供了有关代码页转换过程的更多信息。

   b. 如果您需要将除标准输入之外的任何信息与转换器程序进行通信。 将提供一个用户令牌，分析器和转换器程序可以用它来交换少量信息或交换包含更多信息的共享工作区的地址。

   c. 如果您需要可以由分析器程序执行的监控或审计操作。

   d. 如果您需要对过程元素（如，使用的转换器程序、处理请求的应用程序，或者用于请求的别名事务和用户标识）进行动态更改。

   如果您不需要这些功能中的任何一个功能，那么可使用 CICS 提供的缺省分析器程序 DFHWBAAX 或 CICS 提供的样本分析器程序 DFHWBADX 来提供基本错误句柄。DFHWBAAX 适合于这种情况：使用该端口的所有请求是使用 URIMAP 定义处理的。对于使用 URIMAP 定义的请求和按照 CICS TS 3.1 之前的 CICS Web Support 使用的同一过程的请求，DFHWBADX 提供基本支持。

4. 使用第 123 页的第 11 章，『转换器程序』中的信息以创建合适的转换器程序。 转换器程序被调用了两次，一次用于**解码**函数，该函数检查 Web 客户机的请求和 URIMAP 定义或分析器程序提供的任何其他信息，并创建适当的 COMMAREA 以传递到应用程序。另一次则是为**编码**函数调用转换器程序，该函数接收应用程序

的输出，并创建 HTTP 响应。如果有多个应用程序提供数据，那么转换器程序可以重复调用解码函数。第 128 页的『从转换器程序调用多个应用程序』 说明了如何实现这一点。

5. 决定 Web 客户机将用于每个请求的 URL，包括方案、主机和路径部分以及任何查询字符串。 第 10 页的『URL 的组成部分』说明了这些组成部分以及如何对它们定界。第 32 页的『CICS Web Support 的 URL』 说明了为 CICS Web Support 选择 URL 要考虑的因素和限制。
6. 决定将用于请求的端口。 第 50 页的『保留 CICS Web Support 的端口』有关可由 CICS Web Support 使用的端口的更多信息。 对于 HTTP，缺省端口号是 80，而对于 HTTPS（具有 SSL），缺省端口号是 443。需要在请求的 URL 中明确指定不是方案的缺省值的端口号。 如果您愿意，可以允许请求使用与 CICS Web Support 关联的任何端口。
7. 为涉及本任务的用户应用程序选择别名事务标识。缺省别名事务是 CWBA。按第 86 页的『为 CICS Web Support 创建 TRANSACTION 资源定义』中的指示信息进行操作，您可以创建您自己的别名事务。 您可以使用 URIMAP 定义或分析器程序为每个 HTTP 请求指定别名事务。 如果正在使用 Web 客户机的用户标识执行资源级安全性，那么用户标识用于该事务，并且需要有权访问该事务使用的受保护 CICS 资源和命令。
8. 设置 URIMAP 定义以处理每个请求。按第 87 页的『为作为 HTTP 服务器的 CICS 的任何请求启动 URIMAP 资源定义』和第 89 页的『为作为 HTTP 服务器的 CICS 的 HTTP 请求的应用程序响应完成 URIMAP 定义』中的过程进行操作。 按照 CICS TS 3.1 之前的 CICS Web Support 使用的同一过程，可以将 HTTP 请求直接传递到分析器程序而无需使用 URIMAP 定义。然而，使用 URIMAP 定义可以更容易地管理 HTTP 请求。没有 URIMAP 定义的情况下，如果要更改 CICS 对特殊 HTTP 请求响应的方式，那么需要更改分析器程序中的逻辑。使用 URIMAP 定义，您可以将这些更改作为系统管理任务动态执行。
9. 如果接收请求的端口尚未有 TCPIPSERVICE 定义，那么按第 83 页的『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』中的过程操作以设置一个 TCPIPSERVICE 定义。 使用该定义来为端口指定安全性措施（如，使用 SSL 和基本认证）。 相关 TCPIPSERVICE 定义的名称在请求的 URIMAP 定义中指定。不指定 TCPIPSERVICE 定义意味着 URIMAP 定义匹配的请求可以使用存在 TCPIPSERVICE 定义的任何端口。
10. 为该 CICS Web Support 任务检查错误句柄过程。
    a. 检查与 TCPIPSERVICE 定义关联的分析器程序行为，以了解在哪个端口上接收请求。如果为请求的 URIMAP 匹配失败，那么请求被传递到分析器程序。 第 111 页的第 10 章，『分析器程序』提供关于使用分析器程序可执行的操作的更多信息。
    b. 确保您的体系结构中包括的用户可替换 Web 出错程序向 Web 客户机提供适当的响应。 第 103 页的第 9 章，『Web 出错程序』说明应用 DFHWBEP 和 DFHWBERX 的情形，并介绍了如何定制它们提供的响应。

第 6 章 为作为 HTTP 服务器的 CICS 编写支持 Web 的应用程序

在 CICS 中，编写支持 Web 的应用程序是使用 EXEC CICS WEB 命令通过 CICS 与 Web 客户机或服务器交互的程序。对于作为 HTTP 服务器的 CICS，这些程序可以接收和分析 HTTP 请求，并向 Web 客户机提供应用程序生成的响应。

开始之前

开始为作为 HTTP 服务器的 CICS 编写支持 Web 的应用程序代码前，请阅读第 25 页的『作为 HTTP 服务器的 CICS 的 HTTP 请求和响应处理』主题，以了解会涉及到的处理阶段。

如果希望使提供给 Web 客户机的服务遵从 HTTP 协议规范，特别是 HTTP/1.1，请阅读主题第 40 页的『作为 HTTP 服务器的 CICS 的 HTTP/1.1 一致性』以了解 CICS 和您的应用程序为此所应采取的操作。

关于此任务

对于需要应用程序生成的响应的每个 HTTP 请求，CICS 调用请求的 URIMAP 定义中指定，或者，如果使用分析器，由分析器程序指定的支持 Web 的应用程序。如果使用 URIMAP 定义指定应用程序，您可选择单个应用程序，以使用特殊 URL 为所有请求提供服务。如果您使用分析器程序代替（或除了）URIMAP 定义，它可在请求上执行分析，并确定备用应用程序。

切记: 使用 EXEC CICS WEB 命令的支持 Web 的应用程序必须在接收 Web 客户机请求的 CICS 区域中运行。然而，它们可以链接到其他 CICS 区域中的应用程序（例如，执行业务逻辑）。

对于作为 HTTP 服务器的 CICS，当应用程序已对请求发送响应且将控制返回给 CICS 后，它不等待来自 Web 客户机的更多请求。即使请求形成了逻辑序列，或正在使用持续连接发出请求，或者是管道化的请求，都处于这种情况。如果需要在不同的程序间（或同一程序的新实例间）共享信息，那么可以使用 CICS 管理的资源或使用 Web 客户机发送的请求的元素来执行此操作。

您可以编码每个支持 Web 的应用程序以执行处理 HTTP 请求的以下部分或所有操作:

1. 使用 WEB EXTRACT 命令，从请求行（包括请求 URL）检索您的应用程序所需的任何信息。第 68 页的『检查 HTTP 请求的请求行』介绍了如何执行该操作。 请求行包括 HTTP 方法，它表明应用程序应该执行的操作。您还可以设计请求 URL 的路径部分以对应用程序提供处理信息。如果请求 URL 中有查询字符串，那么应用程序可以将它作为整体来抽取以便进行处理。
2. 使用 HTTP 头命令读取或浏览请求的 HTTP 头。第 69 页的『检查消息的 HTTP 头』介绍了如何执行该操作。 HTTP 头中的信息可能对应用程序处理和响应请求是有用的。
3. 获取有关您的应用程序需要使用的请求的任何技术信息。您可使用 EXEC CICS 命令访问有关 TCP/IP 环境和安全选项的信息。 第 70 页的『检索有关 HTTP 请求的技术和安全信息』介绍了如何执行该操作。

4. 如果请求包含您要抽取的表单数据，那么可以使用表单字段命令来读取或浏览该数据。第 71 页的『检查 HTTP 请求中的表单数据』介绍了如何执行该操作。 该数据可以在请求主体中或者作为 URL 中的查询字符串，且 CICS 可以从这些位置中的任一位置抽取该数据。
5. 如果请求包含您需要使用的消息体，那么可以使用 WEB RECEIVE 命令将它接收到缓冲区中。第 72 页的『接收 HTTP 请求的实体主体』介绍了如何执行该操作。 如果有消息体，那么 CICS 不要求您再接收一个，而某些请求没有消息体。
6. 使用您已收集的信息为请求处理执行业务逻辑。 您可能要包含其他应用程序以执行处理。 根据从不支持 Web 的程序接收的信息，支持 Web 的应用程序可以产生到 HTTP 请求的响应。 建议您将业务逻辑与表示逻辑分开。在支持 Web 的应用程序中，表示逻辑控制与 Web 客户机的交互。要获得如何分开业务逻辑和表示逻辑的建议，请参阅 *CICS Application Programming Guide*。
7. 使用 WEB WRITE HTTPHEADER 命令编写响应的 HTTP 头。第 74 页的『为响应写 HTTP 头』介绍了如何执行该操作。 CICS 自动提供某些必需的头（如，Date 头）。您可以出于其他目的提供其他头。
8. 生成作为 HTTP 响应内容的实体主体或消息体。第 76 页的『为 HTTP 消息产生实体主体』介绍了如何执行该操作。 实体主体可以由 CICS 文档（使用 EXEC CICS DOCUMENT 应用程序编程接口创建的）形成，或由应用程序提供的数据缓冲区形成。
9. 使用 WEB SEND 命令将响应发送到 Web 客户机。第 76 页的『从作为 HTTP 服务器的 CICS 发送 HTTP 响应』介绍了如何执行该操作。 您需要选择合适的状态码和原因短语，并指定实体主体。CICS 使用这些项和 HTTP 头汇编响应。 如果要使用分块的传输编码，还需要按第 78 页的『使用分块的传输编码发送 HTTP 请求或响应』中的特殊指示信息进行操作。
10. 如果您希望与该 Web 客户机交换更多的请求和响应，且需要跨请求序列共享数据，那么使用第 80 页的『跨 HTTP 请求序列管理应用程序状态』中的建议来实现这一目的。

下一步做什么

当 EXEC CICS WEB 命令用于作为 HTTP 服务器的 CICS，**不要**使用 SESSTOKEN 选项。SESSTOKEN 选项表明命令用于作为 HTTP 客户机的 CICS。

检查 HTTP 请求的请求行

CICS 存储用于每个 HTTP 请求的请求行，以供应用程序在需要时可以访问。应用程序可以使用 WEB EXTRACT 命令来抽取请求 URL 的部分（包括路径、主机名、端口号和查询字符串）、用于请求的方法或请求的 HTTP 版本。非 HTTP 请求也可以用这种方式识别。

关于此任务

第 12 页的『HTTP 请求』说明请求行中的项。请求 URL 是请求行的主要元素；第 10 页的『URL 的组成部分』说明 URL 的不同部分。为了处理请求并提供适当的响应，您的应用程序可能需要检查请求行中的任何项。从请求行抽取信息的某些常见原因如下:

- 因为调用同一应用程序来处理很多不同的请求，也许作为逻辑请求序列的一部分请求，或作为与同一资源相关的不同请求。

- 查看 HTTP 方法从应用程序请求的操作。第 349 页的附录 D, 『CICS Web Support 的 HTTP 方法参考』说明了 Web 客户机可能对请求使用的各种方法，并建议了适合在每种情况下采取的操作。
- 使用 URL 的路径部分。这将识别应用请求的资源。除了用于将请求映射到处理应用程序外，URL 的路径部分还可以设计为对应用程序提供处理信息。例如，路径部分可用于指定应用程序提供的特殊函数。或者，如果支持 Web 的应用程序为其他多个应用程序提供前端服务，那么 URL 的路径部分可以识别请求所针对的应用程序。第 32 页的『CICS Web Support 的 URL』说明了如何实现这一点。
- 获取供应用程序处理的查询字符串。
- 识别 Web 客户机的 HTTP 版本，以便应用程序可以提供适当的响应。Web 客户机使用的 HTTP 版本会影响响应的 HTTP 头、状态码和消息内容。HTTP/1.0 客户机可能不了解 HTTP/1.1 规范中描述的更多高级功能。
- 识别非 HTTP 请求。第 163 页的第 14 章, 『CICS Web Support 和非 HTTP 请求』提供了有关处理非 HTTP 请求的更多信息。

CICS Application Programming Reference 有 WEB EXTRACT 命令中可用的选项的完整参考信息和描述。WEB EXTRACT 命令可以让您获取以下项:

- 使用 HOST 选项以获取请求 URL 的主机部分，如请求的 Host 头字段或请求行（如果绝对 URI 格式用于请求）中所指定。
- 使用 HTTPMETHOD 选项以获取请求的 HTTP 方法（例如，GET 或 PUT）。
- 使用 HTTPVERSION 选项以识别 HTTP 版本（HTTP/1.1 或 HTTP/1.0）。
- 使用 PATH 选项以获取 URL 的路径部分。
- 使用 PORTNUMBER 选项以获取应用到 URL 的端口号。 通常从 URL 省略服务的常用端口号。如果 URL 中不存在该端口号，那么 WEB EXTRACT 命令将根据方案识别并将它返回。对于 HTTP，熟知端口号是 80，而对于 HTTPS，熟知端口号是 443。
- 使用 QUERYSTRING 选项以获取整个查询字符串。 查询字符串以其转义形式返回，具有 %xx 序列表示特定字符，这些字符会妨碍正确的语法分析。请参阅第 15 页的『转义和未转义数据』以获取有关此内容的说明。 另外，如果查询字符串包括作为名称和值对（例如，`account=40138025`）的表单数据，那么可以使用 WEB READ FORMFIELD 命令以非转义形式获取该数据。 第 71 页的『检查 HTTP 请求中的表单数据』介绍了如何执行该操作。
- 使用 REQUESTYPE 选项以识别非 HTTP 请求。

检查消息的 HTTP 头

请求或响应消息的每个 HTTP 头由头名称和头值组成。如果需要，CICS 存储此信息以供应用程序访问。应用程序可以接收指定头的值，或浏览请求或响应的所有头的名称和值。您还可以将从头中获取的结构化日期和时间戳记字符串转换为 ABSTIME 格式。

关于此任务

为了处理请求或响应并构造后续消息，您的应用程序可能需要检查头中的信息。例如:

- TE 头告诉应用程序: 分块的响应消息中是否允许尾部头。
- 条件头可以对应用程序提供指示信息，例如，仅当响应文档更改后才应答。

请记住，除非您知道 HTTP 请求或响应的确切格式，否则您的应用程序不应该依赖于任何特殊头的存在，这是因为 Web 客户机和服务器发送的头可能不一致。

某些 HTTP 头包含日期和时间戳记。CICS 提供 CONVERTTIME 命令，将结构化日期和时间戳记字符串的常见格式转换为 ABSTIME 格式，以供应用程序使用。

标准 HTTP 头在 HTTP/1.1 规范（RFC 2616）和 HTTP/1.0 规范（RFC 1945）中描述。第 335 页的附录 B，『CICS Web Support 的 HTTP 头参考』说明 CICS Web Support 中 HTTP 头的一般用法，以及 CICS Web Support 针对消息中接收的特定头采取的操作。某些 HTTP 头会被 CICS 忽略，并且由用户应用程序来决定执行适当的操作以响应这些头。要获取有关每个 HTTP 头的含义和正确使用的详细指导和需求，请检查您正在遵从的 HTTP 规范。

如果消息包含任何尾部头，那么您可使用 EXEC CICS WEB 命令，通过用于标准头的相同方法读取这些尾部头。消息上的尾部头指定作为尾部头发送的所有 HTTP 头的名称。

- 要检查特殊 HTTP 头的内容，请使用 WEB READ HTTPHEADER 命令。 您的应用程序需要提供接收头内容的缓冲区。如果请求中不存在头，那么 CICS 返回 NOTFND 条件。
- 要浏览请求或响应中的所有头:
  1. 使用 WEB STARTBROWSE HTTPHEADER 命令以开始浏览头行。
  2. 使用 WEB READNEXT HTTPHEADER 命令以检索每一行的头名称和头值。 您的应用程序需要提供两个缓冲区: 一个接收头的名称，而另一个接收其内容。读取所有头后，CICS 将返回 ENDFILE 条件。
  3. 当您的程序已检索所有相关头信息后，使用 WEB ENDBROWSE HTTPHEADER 命令结束浏览。
- 要转换 HTTP 头中提供的结构化日期和时间戳记字符串，请使用 WEB READ HTTPHEADER 命令将它接收到缓冲区中，然后使用 CONVERTTIME 命令处理它。您不需要识别日期和时间戳记的格式；CONVERTTIME 命令会识别在因特网上常用的三种不同日期和时间戳记格式并进行转换。它们是 RFC 1123 格式（Web 标准）、RFC 850 格式（较早的格式）和 ASCtime 格式（C 函数的输出）。 使用 FORMATTIME 命令，应用程序可以将 ABSTIME 转换为其他格式。

检索有关 HTTP 请求的技术和安全信息

应用程序可以获取有关 HTTP 请求的 TCP/IP 环境的信息（包括正在使用的安全选项），以及有关 Web 客户机已提供的客户机证书的信息。

关于此任务

CICS 管理 Web 客户机和服务器之间的 TCP/IP 连接，应用适当的安全措施，并管理 Web 客户机身份的认证过程。 CICS 针对每个连接采取的操作，由您在接收 Web 客户机请求的端口的 TCPIPSERVICE 定义中设置的选项决定。用户编写的应用程序可以检查此过程获取的信息是否可用于确定如何处理该请求。例如，您可以获取发送 HTTP 请求的 Web 客户机的主机名和 IP 地址，或检查连接的安全和加密级别。

EXTRACT TCPIP 命令提供有关 TCP/IP 连接，以及 TCPIPSERVICE 定义中指定的安全选项的信息。EXTRACT CERTIFICATE 命令提供从任何 X.509 客户机证书获取的信息，该证书是在安全套接字层（SSL）握手期间从 Web 客户机接收的。*CICS Applica-*

tion Programming Reference 提供有关这些命令可用选项的完整参考信息和描述。

- 要获取发送 HTTP 请求的 Web 客户机的主机名和 IP 地址，请使用具有 CLIENTNAME 和 CLIENTADDR 选项的 EXTRACT TCPIP 命令。 此 IP 地址以二进制数字形式提供，或以包含其冒号十六进制或点分十进制表示法的字符串形式提供。
- 要获取运行应用程序的主机系统（即，CICS 本身）的主机名和 IP 地址，可使用带有 SERVERNAME 和 SERVERADDR 选项的 EXTRACT TCPIP 命令。 同样，此 IP 地址也以二进制数字形式提供，或以包含其冒号十六进制或点分十进制表示法的字符串形式提供。
- 要获取接收请求的端口号，可以使用具有 PORTNUMBER 选项的 EXTRACT TCPIP 命令。 端口号作为二进制数字或字符串提供。也可以使用具有 PORTNUMBER 选项的 WEB EXTRACT 命令。
- 要获取与请求关联的 TCPIPSERVICE 资源定义的名称，使用具有 TCPIPSERVICE 选项的 EXTRACT TCPIP 命令。
- 要识别 TCPIPSERVICE 定义中指定的认证类型（基本认证、客户机证书认证或没有认证），使用具有 AUTHENTICATE 选项的 EXTRACT TCPIP 命令。 第 151 页的『作为 HTTP 服务器的 CICS: 认证和标识』说明有关不同类型的认证的更多信息。
- 要识别是否在 TCPIPSERVICE 定义中指定了安全套接字层（SSL）支持和使用的 SSL 加密级别，那么使用具有 SSLTYPE 和 PRIVACY 选项的 EXTRACT TCPIP 命令。第 160 页的『具有 CICS Web Support 的 SSL』说明有关 SSL 的更多内容。
- 要从 X.509 证书（该证书是在 SSL 握手期间从 Web 客户机接收的）检索信息，请使用 EXTRACT CERTIFICATE 命令。 CICS 已验证了所提供的证书，方法是对照安全管理器数据库和可设置的证书撤销列表来检查该证书。 证书包含用于标识证书主体（有时称为所有者或用户）的字段，以及用于标识发布证书的认证中心（发布者）的字段。可以通过指定 OWNER 或 ISSUER 选项来选择您需要的信息。您还可以使用 SERIALNUM 和 USERID 选项来获取证书号以及与证书关联的 RACF® 用户标识。 *CICS RACF Security Guide* 提供了有关证书内容以及如何使用证书的更多信息。

检查 HTTP 请求中的表单数据

表单数据是用户通过与 HTML 表单中的元素（例如，文本输入框、按钮或复选框）交互作用而提供的信息。该信息作为一系列名称和值对发送。CICS 可以搜索 HTTP 请求以检取表单字段，因此应用程序可使用 CICS 命令获取数据而无需接收和分析整个请求主体。

关于此任务

第 16 页的『HTML 表单』说明有关表单和表单字段的更多内容。

应用程序可以接收指定表单字段的值，或浏览请求中包含的所有表单字段的名称和值。如果需要将数据转换到不同的代码页中以供应用程序使用，那么可以指定代码页转换选项。

使用 GET 方法时，Web 客户机在查询字符串中发送表单数据，而使用 POST 方法时，它在消息体中发送表单数据。CICS 可以从这两个位置中的任一位置抽取数据，所以您不

需要指定使用哪个方法。作为备用方法，如果在查询字符串中发送表单数据，那么可以使用 WEB EXTRACT 命令检索整个查询字符串。第 68 页的『检查 HTTP 请求的请求行』介绍了如何执行该操作。

仅当 CICS 是 HTTP 服务器时，才会读取表单数据。当 CICS 是 HTTP 客户机时，该设施不可用。

- 要获取 HTML 表单的特殊字段的值，请使用 WEB READ FORMFIELD 命令。 您的应用程序可以提供将接收值的缓冲区，或者可以提供 CICS 设置为值地址的指针。如果表单数据不包含具有指定名称的字段，那么 CICS 返回 NOTFND 条件。 表单数据返回前CICS 不对它进行转义，%xx 序列将转换回原始字符。请参阅第 15 页的『转义和未转义数据』以获取有关此内容的说明。
- 要浏览表单数据中的所有字段:
  1. 使用 WEB STARTBROWSE FORMFIELD 命令开始浏览这些字段。
  2. 使用 WEB READNEXT FORMFIELD 命令依次检索每个字段的名称和值。 您的应用程序需要提供两个缓冲区: 一个接收字段名，而另一个接收其内容。读取所有字段后，CICS 将返回 ENDFILE 条件。
  3. 当您的程序已检索所有相关字段后，使用 WEB ENDBROWSE FORMFIELD 命令以结束浏览。
- CICS 对您接收的数据进行代码页转换。 您可以使用 WEB STARTBROWSE FORMFIELD 和 WEB READ FORMFIELD 命令上的 CHARACTERSET 和 HOSTCODEPAGE 选项，以指定 Web 客户机和应用程序使用的代码页。
  1. 客户机应用程序对 GET 和 POST 方法使用的字符编码（charset 参数）由 HTML 表单中的信息决定。但是，该信息通常不包含在已提交的表单请求中，因此由使用 CHARACTERSET 选项的应用程序提供。该信息应该符合相应 HTML 表单所确定的表单编码（请参阅第 17 页的『如何确定客户机编码』以获取更多信息）。
  2. HOSTCODEPAGE 指定了应用程序使用的 CICS（主机）代码页。该代码页通常是 EBCDIC 代码页。如果未指定代码页，那么以 LOCALCCSID 系统初始化参数指定的 EBCDIC 代码页返回数据，前提是 CICS Web 接口支持该代码页。否则，CICS 以缺省的 EBCDIC 代码页 037 返回数据。

  有关 CHARACTERSET 和 HOSTCODEPAGE 选项的更多信息，请参阅 WEB READ FORMFIELD 和 WEB STARTBROWSE FORMFIELD 命令。

接收 HTTP 请求的实体主体

应用程序可以发出 WEB RECEIVE 命令以接收 HTTP 请求的实体主体。您只可以接收实体主体的第一部分，或使用一系列 WEB RECEIVE 命令来以较小的多个部分接收整个主体。

关于此任务

WEB RECEIVE 命令不设置超时值。只有已从 Web 客户机成功接收完整请求后才调用用户应用程序，且该应用程序由 CICS 保存。对于作为 HTTP 服务器的 CICS，端口的 TCPIPSERVICE 定义中的 SOCKETCLOSE 属性确定 Web 客户机必须完成其请求发送所需的时间。这段时间到期时，CICS 将 408（请求超时）响应返回给 Web 客户机。

如果使用分块的传输编码发送请求消息，那么 CICS 在将块传递到应用程序前，会把块汇编到单个消息中。如果发送了一系列管道化的请求，那么 CICS 将每个请求作为独立的事务对待，而且在使下一次请求可供下一个用户应用程序处理前，需要来自用户应用程序的响应。

CICS Application Programming Reference 提供有关 WEB RECEIVE 命令中可用选项的完整参考信息和描述。发出 WEB RECEIVE 命令时:

1. 标识您是否需要为此请求接收实体主体。
   a. 对于某些请求方法（如 GET 方法），实体主体不适用，而且允许您的应用程序忽略存在的任何实体主体。 第 349 页的附录 D, 『CICS Web Support 的 HTTP 方法参考』表明适用实体主体的方法。如果存在不适用的实体主体，那么如果您需要，您可仍然接收它。 第 68 页的『检查 HTTP 请求的请求行』介绍了如何标识请求方法。
   b. 对于 HTTP/1.1 请求，实体主体的存在由请求上非零的 Content-Length 头（或者如果消息已分块，由 Transfer-Encoding 头）表明。 如果 Content-Length 头的值为零，或者如何 Transfer-Encoding 头和 Content-Length 头都没有提供，那么不存在实体主体。 第 69 页的『检查消息的 HTTP 头』介绍了如何读取消息的 HTTP 头。
   c. 没有规定 HTTP/1.0 请求必须指定 Content-Length 头，但是它们可能会进行指定。如果您在请求上找到非零的 Content-Length 头，这表明存在实体主体。 如果不存在 Content-Length 头，但是请求方法（特别是 POST 方法）表明实体主体适用，那么可能存在实体主体。
2. 通过指定 INTO 选项（对于数据缓冲区）或 SET 选项（对于指针引用）和 LENGTH 选项来接收实体主体。 返回时，LENGTH 选项设置为所接收的数据的长度。
3. 如果要限制从实体主体接收的数据量，那么指定 MAXLENGTH 选项。
   a. 如果要只接收第一部分实体主体，并丢弃超出此长度的任何数据，那么省略 NOTRUNCATE 选项。 这是缺省值。
   b. 如果要保留（而不是丢弃）超出此长度的任何数据，那么指定 NOTRUNCATE 选项。 可以使用更多 WEB RECEIVE 命令来获取任何剩余数据。

   如果已使用分块的传输编码发送数据，CICS 会将块组装到单条消息中，然后再将该消息传递到应用程序。因此 MAXLENGTH 选项将应用于已分块消息的整个实体主体长度，而不是应用到各个块。对于一条消息，CICS 接受的数据总量由 TCPIPSERVICE 定义的 MAXDATALEN 属性限制。
4. 指定您要在此处设置的任何选项以执行代码页转换。
   a. SERVERCONV 选项提供代码页转换的全部控制。使用它指定是否进行代码页转换。 对于作为 HTTP 服务器的 CICS，为了兼容在早期发行版中编码的支持 Web 的应用程序，如果未指定 SERVERCONV 但指定了另一个代码页转换选项，那么会进行代码页转换。如果您要阻止代码页转换，那么指定 SERVERCONV (NOSRVCONVERT) 或省略所有代码页转换选项。

      注: 如果您接收已打包或已压缩的实体主体，如消息的 Content-Encoding 头表明的，那么确保禁止了代码页转换。CICS 不对这些消息类型进行解码，并且如果应用代码页转换，结果将不可预测。如果您无法对已打包或已压缩的实体主体进行解码，那么可以通过返回 415 状态码告知 Web 客户机。
   b. 如果您希望进行代码页转换，但 CICS 无法确定 Web 客户机的字符集，那么使用 CHARACTERSET 选项指定它。 对于较早的 Web 客户机，请求头可能不提

供该信息。在这种情况下，CICS 采用 ISO-8859-1 字符集，因此，您只需在所采用的字符集不正确的情况下指定字符集。

c. 如果您希望进行代码页转换，但本地 CICS 区域的缺省代码页（由 LOCALCCSID 系统初始化参数指定）不适合于您的应用程序，那么使用 HOSTCODEPAGE 选项指定备用主机代码页。

对于指定非文本介质类型的消息，不进行代码页转换（除非您不指定 SERVERCONV，在这种情况下，出于兼容目的，会不考虑介质类型）。注意，出于兼容目的，如果入站消息不指定介质类型，CICS 会偏离 HTTP/1.1 要求而缺省为 `application/octet-stream`。CICS 改为使用 `text/plain` 作为缺省值，以便可以为消息执行代码页转换。

5. 如果指定了 MAXLENGTH 和 NOTRUNCATE，且有更多数据要接收，那么发出更多 WEB RECEIVE 命令。 使用 SET 选项但不带 MAXLENGTH 选项的一条 RECEIVE 命令接收所有剩余的数据，而不管它的长度有多长。或者，您也可以使用一连串 RECEIVE 命令（带 NOTRUNCATE 选项）接收相应块中的剩余数据。 持续发出 RECEIVE 命令，直到不再获取 LENGERR 响应。 请记住，如果接收的长度比 MAXLENGTH 选项上请求的长度短，这并不必然地表明数据的结束；如果 CICS 需要避免返回数据末尾的部分字符，那么可能发生这种情况。

为响应写 HTTP 头

对于应用程序创建的动态响应，根据用于消息的 HTTP 协议版本，CICS 自动提供基本消息所需的 HTTP 头。您可能需要将更多 HTTP 头添加到您的响应。

关于此任务

如果消息需要某些 HTTP 头，那么 CICS 将自动创建它们。您的应用程序不需要写这些头。CICS 创建的完整头列表如下:

- ARM 相关因子
- Connection
- Content-Type（由 CICS 编写，但在需要复杂头时，可由客户机应用程序提供）
- Content-Type
- Date
- Expect
- Host
- Server
- TE（由 CICS 编写，但是可能添加了更多实例）
- Transfer-Encoding
- User-Agent
- WWW-Authenticate

请注意，只有 CICS 是 HTTP 客户机时，这些头中的一部分才是适用的且会被创建。创建这些头的情况在 第 335 页的附录 B, 『CICS Web Support 的 HTTP 头参考』中描述。如果您在响应中写这些头，那么 CICS 不覆盖它们，但会使用您应用程序提供的版本。

CICS 在发送响应时提供的头是通常应该为基本消息所写的、以使该消息遵从相应 HTTP 协议规范的头。您可能为了诸如以下目的要将更多 HTTP 头添加到响应:

- 高速缓存和文档到期的控制（例如，Cache-Control、Expires 和 Last-Modified）
- 内容协商（例如，Accept-Ranges 和 Vary）
- Web 客户机的信息（例如，Title、Warning 和更多 Content 头）

如果您的应用程序正在执行复杂操作，或者如果为您的响应选择某些状态码，那么您所遵从的 HTTP 规范可能要求将特定的 HTTP 头用于您的消息。当将任何 HTTP 头添加到响应时，检查一些重要需求所遵从的 HTTP 规范是否应用于那些头。请参阅 第 12 页的『HTTP 协议』以获取有关 HTTP 规范的更多信息。

在您发出 WEB SEND 命令以发生消息**前**，为消息编写其他 HTTP 头。该规则的特例是您编写的头要在已分块消息上作为尾部头发送，在该情况下，应用下面提到的特殊进程。为响应写 HTTP 头时:

- 对您要添加到消息的每个头使用 WEB WRITE HTTPHEADER 命令。 确保以您正在遵从的 HTTP 规范所描述的格式为每个头指定名称和值。（CICS 不验证 HTTP 头的内容，这是因为它可能要使用新的或用户定义的头。） 该命令添加单个头，您可以重复该命令以添加更多头。如果您编写已经写好的头，那么除了现有的头之外，CICS 还会将新头添加到请求或响应。确保只在 HTTP 规范声明可以重复头的地方执行该操作。 CICS 发送请求时，会存储准备添加到该请求的头。
- 如果您使用的任何 HTTP 头可能不适合低于 HTTP/1.1 级别的 Web 客户机，那么写那些头前，请检查 Web 客户机已提供给您的 HTTP 版本信息。 使用 WEB EXTRACT 命令来获取此信息。 为了允许您使用用户定义的（非标准）头，CICS 不会除去用户所写的不适用的头。低于 HTTP/1.1 级别的服务器无法识别某些 HTTP 头，这可能导致处理您的请求时出错。

  注: CICS 对低于 HTTP/1.0 级别的服务器或 Web 客户机不提供任何特殊支持。CICS 运作时假定它们都处于 HTTP/1.0 级别，并将 HTTP/1.0 作为 HTTP 版本返回。
- 如果要产生要在某个 HTTP 头（例如，Last-Modified 头）中使用的日期和时间戳记，那么可以使用 FORMATTIME 命令。 命令中的 STRINGFORMAT 选项用于将当前日期和时间（ABSTIME 格式）或应用程序生成的日期和时间转换为适合在 Web 上使用的日期和时间戳记格式。其他日期和时间戳记格式可能不被与 CICS 通信的某些 Web 客户机或服务器所接受。
- 如果希望生成在 ETag HTTP 头中使用的强实体标记，那么可以使用由 BIF DIGEST 命令生成的 SHA-1 摘要。 通过强实体标记，客户机能够针对在 If-Match、If-None-Match 或 If-Range 头中使用实体标记的资源发出条件请求，与 Last-Modified 日期和时间字符串相比，这是一种更为准确的资源状态检查方法。如果希望允许条件请求，那么应用程序必须为这些请求提供支持; CICS 自身没有为针对 HTTP GET 请求的 If-Match、If-None-Match 或 If-Range 函数提供支持。
- 如果正在使用分块的传输编码发送 HTTP 请求或响应，并且要将尾部头包括在分块消息的末尾，那么按第 78 页的『使用分块的传输编码发送 HTTP 请求或响应』中的特殊指示信息进行操作。 在发送消息的第一块前，需要写尾部头。继 WEB SEND 命令后为第一块写的所有 HTTP 头作为尾部头来对待。
- 确保您的应用程序执行用户所写的头暗示的任何操作。 例如，如果已写内容协商头，那么应用程序需要提供不同版本的资源。

为 HTTP 消息产生实体主体

支持 web 的应用程序可以产生从 CICS 文档或数据缓冲区形成的实体主体。

开始之前

关于此任务

CICS 文档可以用作 HTTP 消息的实体主体。它们是使用 EXEC CICS DOCUMENT 命令创建的。它们可以由应用程序和文档模板（定义为 CICS 资源或由另一个 CICS 程序创建的文档的一部分）直接指定的数据植入。文档和文档模板可以存储以供再次使用。

您还可以指定应用程序创建的一缓冲区的数据。您可能发现此选项对于简短或简单的实体主体来说更方便，而且它是允许您使用消息的已分块转换编码的唯一选项。然而，以此方式创建的数据无法那么容易地存储以供再次使用。

1. 要创建 CICS 文档，请按照 *CICS Application Programming Guide* 中的指示信息操作。 该文档是使用 EXEC CICS DOCUMENT 应用程序编程接口（EXEC CICS DOCUMENT CREATE, INSERT 和 SET 命令）创建的。WEB SEND 命令中的 DOCTOKEN 选项用作完成的文档指定文档令牌。CICS 根据您在 WEB SEND 命令上指定的选项，检索文档并执行适当的代码页转换。 无法从 CICS 文档形成分块的消息体。
2. 或者，组装您的应用程序中的消息体。 WEB SEND 命令中的 FROM 选项用于指定数据的缓冲区。 对于数据缓冲区的大小未设置最大值限制，但是您需要考虑实际上可能限制其大小的以下因素:
   - CICS 区域的 EDSA 限制。
   - 在 CICS 区域中可以同时汇编的其他消息体数。调度约束可能由应用到 CICS Web Support 事务的任何事务类定义的 MAXACTIVE 设置实施。
   - 用于消息体的代码页转换类型。在从 EBCDIC 代码页 037 转换到 ASCII 代码页 ISO-8859-1 时，CICS 会覆盖同一个数据缓冲区，因此无需使用额外存储。对于任何其他类型的代码页转换，CICS 需要额外的存储器以包含转换的消息体。根据使用的字符集，此额外存储器区域的大小范围可以是: 从与原始消息体相同的大小，到理论上最大为原始消息体大小的四倍（这不太可能）。例如，使用 FROM 选项发送的 2MB 缓冲区的数据将需要总数至少为 4MB 的存储器。双字节字符集（DBCS）或多字节字符集可能需要此范围中较大的存储器区域。

从作为 HTTP 服务器的 CICS 发送 HTTP 响应

使用 WEB SEND 命令使 CICS 汇编 HTTP 响应的 HTTP 头、实体主体、状态码和原因短语，执行代码页转换，并将响应传输到 Web 客户机。

开始之前

发出 WEB SEND 命令前，使用 WEB WRITE HTTPHEADER 命令写响应的任何其他 HTTP 头，如第 74 页的『为响应写 HTTP 头』中所描述。还将生成消息所需的任何实体主体，如『为 HTTP 消息产生实体主体』中所描述。

您需要在 WEB SEND 命令中指定状态码和原因短语。第 15 页的『状态码和原因短语』介绍了该状态码和原因短语。第 341 页的附录 C, 『CICS Web Support 的 HTTP 状态

码参考』概括了应用程序可能使用的状态码。要规划使用状态码并查找有关它们的更多信息，应该查询您正在遵从的 HTTP 规范。请参阅 第 12 页的『HTTP 协议』以获取有关 HTTP 规范的更多信息。

关于此任务

需要的话，响应可以分块（分块的传输编码）发送。您不能将管道化的响应发送回 Web 客户机（必须对 Web 客户机发送的每个请求发送单个响应）。

CICS Application Programming Reference 提供有关这些 WEB SEND 命令中可用选项的完整参考信息和描述。发出该命令时:

1. 根据情形，指定 STATUSCODE 选项为响应选择适当的状态码，并指定 STATUSTEXT 和 STATUSLEN 选项以提供原因短语。 CICS 不验证您选择的状态码，而确保该值有效且符合 HTTP 状态码的规则是用户应用程序的职责。 根据您选择的状态码，可能需要在发出 WEB SEND 命令前完成以下某些或所有步骤:
   a. 检查 Web 客户机请求的 HTTP 版本，以确保可理解状态码。 HTTP/1.1 规范包含的状态码比 HTTP/1.0 规范的状态码多。
   b. 如果 HTTP 规范声明状态码应该伴有特定 HTTP 头，那么使用 WRITE HTTPHEADER 命令来创建那些头。
   c. 如果 HTTP 规范声明状态码应该伴有提供特殊信息的消息体，那么创建适当的实体主体。 通常如果状态码表明错误或从客户机请求进一步操作时，会出现这种情况。 不允许消息体具有状态码 204、205 和 304。如果您已选择不允许消息体的状态码，并尝试使用消息体，那么 CICS 向 WEB SEND 命令给出错误响应。
2. 通过执行以下操作识别响应的任何实体主体的源: 为已创建的 CICS 文档指定 DOCTOKEN 选项，或为已汇编的数据的主体指定 FROM 选项。如果使用 FROM 选项，那么指定 FROMLENGTH 选项给出实体主体或块（如果使用分块的传输编码）的长度。 对于分块的传输编码，不能使用 DOCTOKEN 选项。
3. 使用 MEDIATYPE 选项指定响应主体的介质类型。 CICS 不会针对数据内容检查规范的有效性。 没有缺省值。如果您未指定此选项，那么 CICS 不为响应构建 Content-Type 头。
4. 如果要立即发送消息，而不是在任务结束时发送（这是缺省值），那么为 ACTION 选项指定 IMMEDIATE。 如果正在使用分块的传输编码，那么 IMMEDIATE 是缺省值，因此不需要进行此选择。

   注: 一个任务中只能发送一个响应。这可以是使用一个 WEB SEND 命令的标准响应，也可以是使用一系列 WEB SEND 命令的分块响应。
5. 如果要在发送响应后关闭连接，那么为 CONNECTION 选项指定 CLOSE。 CICS 在响应上编写 Connection: close 头，这将通知 Web 客户机: 连接已关闭，且不应该再发送请求。（对于 HTTP/1.0 级别的 Web 客户机，CICS 通过省略 Connection: Keep-Alive 头来达到相同的效果。）
6. 为消息体的代码页转换指定适当的设置。
   a. SERVERCONV 选项提供代码页转换的全部控制。使用它指定是否进行代码页转换。 对于作为 HTTP 服务器的 CICS，为了兼容在早期发行版中编码的支持 Web 的应用程序，如果未指定 SERVERCONV 但指定了另一个代码页转换选项，那么会进行代码页转换。如果您要阻止代码页转换，那么指定 SERVERCONV (NOSRVCONVERT) 或省略所有代码页转换选项。

b. 如果您希望进行代码页转换，但 CICS 选择的字符集不合适，那么使用 CHARACTERSET 选项指定一个替换字符集。 缺省情况下，CICS 使用 Web 客户机的原始请求的 Content-Type 头中指定的字符集。如果该字符集不受支持或未声明，那么 CICS 改为使用 ISO-8859-1 字符集。

   Web 客户机可以在 Accept-Charset 头中指定替换的可接受字符集。如果您要指定这些字符集中的一个字符集，那么由您的应用程序来决定分析头（可能包括质量值以表明 Web 客户机的首选项）和选择适当的支持字符集。 CICS 不支持 IANA 命名的所有字符集。第 333 页的附录 A，『HTML 编码字符集』列出了 CICS 支持的、可用于代码页转换的 IANA 字符集。

c. 如果您希望进行代码页转换，并要使用 FROM 选项指定消息体，那么在您应用程序的代码页**不是**本地 CICS 区域的缺省代码页（如 LOCALCCSID 系统初始化参数中指定的）的情况下，需要使用 HOSTCODEPAGE 选项识别您应用程序的代码页。 如果您使用 CICS 文档（DOCTOKEN 选项），那么 CICS 从 CICS 文档域的文档主机代码页记录中识别主机代码页。

对于指定非文本介质类型的消息，不进行代码页转换（除非您不指定 SERVERCONV，在这种情况下，出于兼容目的，会不考虑介质类型）。 HTTP 头和状态行总是被 CICS 转换成 ISO-8859-1 字符集。

7. 如果您要使用分块的传输编码（或分块），那么除了本主题中的基本指示信息，您还需要遵循『使用分块的传输编码发送 HTTP 请求或响应』中的某些特殊指示信息。您需要确保正确按照该主题中描述的过程进行操作，从而接收方可以接受分块的消息。分块的消息是使用具有特殊选项的 WEB SEND 命令的多个实例发送的。

使用分块的传输编码发送 HTTP 请求或响应

本主题说明如何对作为 HTTP 客户机的 CICS 发出的 HTTP 请求，或对作为 HTTP 服务器的 CICS 提供的 HTTP 响应，设置分块的传输编码。

开始之前

设置分块的传输编码前，需要规划您要发送的项的以下属性:

1. 应该在消息开头使用的 HTTP 头。CICS 提供其常见消息头，这在 第 335 页的附录 B，『CICS Web Support 的 HTTP 头参考』中列出。对于分块的消息，CICS 为分块的传输编码提供合适的头，包括 Transfer-Encoding: chunked 头。如果消息开头需要任何其他头，那么应用程序可以在进行第一个 WEB SEND 命令前编写它们。
2. 消息末尾的应该在尾部发送的任何头。这些头称为尾部头。注意，HTTP/1.1 规范设置使用尾部头的要求，即使接收方忽略它们也应该没有关系。
3. 应该如何分割消息。可以用对应用程序来说最方便的任何方式执行此操作。例如，来自许多其他应用程序的输出可以用它所产生的方式发送，或者，可以单独读取和发送表的每一行的数据。
4. 将发送的每个数据块的长度。不要包含任何尾部头的长度。

关于此任务

本主题中描述的过程允许您创建正确构造的分块消息，如 HTTP/1.1 规范中定义的。请参阅第 12 页的『HTTP 协议』以获取有关 HTTP/1.1 规范的更多信息。如果未正确构造已分块消息，那么接收方可能丢弃它。

第 76 页的『从作为 HTTP 服务器的 CICS 发送 HTTP 响应』是编写用于发送服务器响应的应用程序的主要指示信息集合。第 129 页的『通过作为 HTTP 客户机的 CICS 发出 HTTP 请求』是编写用于发出客户机请求的应用程序的主要指示信息集合。您可以将当前主题中的指示信息与这两个指示信息集合中的任一个结合使用。

分块消息的主体无法直接从 CICS 文档形成（因此不能使用 DOCTOKEN 选项）。FROM 选项必须用于指定数据以形成分块的消息体。

已开始发送部分分块的消息后，直到发送最后一个空块并完成分块的消息后，您才能发送其他消息或接收项。

1. 开始分块的消息前，请验证 Web 客户机或服务器是否为 HTTP/1.1 版本。 所有 HTTP/1.1 应用程序对于理解分块的传输编码都是必需的。分块的消息不能被发送到 HTTP/1.0 接收方。
   a. 对于作为 HTTP 服务器的 CICS 发送的响应，使用 WEB EXTRACT 命令以检查为 Web 客户机的请求指定的 HTTP 版本。
   b. 对于作为 HTTP 客户机的 CICS 发送的请求，如果在 WEB OPEN 命令上指定 HTTPVNUM 和 HTTPRNUM 选项，该命令将返回所连接的服务器的 HTTP 版本。如果没有这样做，那么使用 WEB EXTRACT 命令来检查服务器的 HTTP 版本。
   c. 或者，您可以在发出 WEB SEND 命令来发送消息的第一个块时，省略这项检查，让 CICS 检查 Web 客户机或服务器的版本。 如果接收方采用 HTTP/1.0，那么 CICS 不会执行发送，而是返回一个错误响应。
2. 必要的话，多次使用 WRITE HTTPHEADER 命令以写任何 HTTP 头，这些 HTTP 头应该在消息体之*前*发送。 不要为分块的传输编码写头；CICS 会使用应用程序提供的块长度信息自己写这些头。
3. 如果要将尾部头（在消息体之*后*发送的头）与分块的消息包含在一起，那么使用 WRITE HTTPHEADER 命令编写 Trailer 头：将您打算在尾部中发送的所有 HTTP 头的名称指定为尾部头的值。 除了 Transfer-Encoding、Trailer 和 Content-Length，您可以发送任何头作为尾部头。
   a. 对于作为 HTTP 服务器的 CICS 发送的响应，您需要确保 Web 客户机在它的请求中已发送 TE: 尾部头。 该头表示客户机了解尾部头。 如果您尝试在客户机未发送 TE: 尾部时写尾部头，那么 CICS 向 WRITE HTTPHEADER 命令返回 RESP2 值为 6 的 INVREQ 响应。或者，您可使用 READ HTTPHEADER 命令检查是否存在 TE: 尾部头。
   b. 对于作为 HTTP 客户机的 CICS 发送的请求，可以包含尾部头而无需引用 TE 头。

   在分块的发送过程期间，写尾部头本身。
4. 使用 WEB SEND 命令发送消息的第一个块。
   a. 指定 CHUNKING(CHUNKYES) 以告诉 CICS 这是一个消息块。
   b. 使用 FROM 选项指定来自消息体的第一块数据。
   c. 使用 FROMLENGTH 选项指定块的长度。
   d. 对于作为 HTTP 客户机的 CICS 发出的请求，必须在 METHOD 选项中指定合适的方法。分块的传输编码与没有消息体的请求无关，因此它与 GET、HEAD、DELETE、OPTIONS 和 TRACE 方法无关，但是它可以用于 POST 和 PUT 方法。

e. 指定应用到分块和非分块消息的任何其他选项，如您的主要指示信息集合中所提供。 例如，如果此分块的消息是您要发送到此服务器或 Web 客户机的最后一条消息，那么指定 CLOSESTATUS(CLOSE) 选项。

5. 必要的话，多次使用 WEB SEND 命令以发送每个剩余消息块。在每个 WEB SEND 命令中，只指定以下项:
   a. CHUNKING(CHUNKYES)。
   b. FROM 选项，它给定数据块。
   c. FROMLENGTH 选项，它给定块的长度。

   不要为该命令指定任何其他选项。当您发出该命令时，CICS 发送每个块。
6. 可选: 为第一个块发出 WEB SEND 命令之后，但为最后一个空块发出 WEB SEND 命令之前的任何时候（请参阅下一步），使用 WRITE HTTPHEADER 命令创建应该作为尾部头发送的更多 HTTP 头。 假如尾部头已写在第一块消息上，那么分块的发送过程期间写的 HTTP 头被 CICS 当作尾部头看待，且它们与最后一个空块一起发出。（如果未写尾部头，那么 CICS 不允许写任何尾部头。） 注意，CICS 不检查您的尾部头是否与您在第一块消息的初始 Trailer 头中指定的名称匹配。
7. 当已发送最后一个数据块时，再指定一个带 CHUNKING(CHUNKEND) 但不带 FROM 或 FROMLENGTH 选项的 WEB SEND 命令。 CICS 然后生成空块并将它发送给接收方以完成分块的消息。空块与包含您所写的任何尾部头的尾部一起发送。
8. 对于作为 HTTP 服务器的 CICS，如下所示处理错误:
   a. 如果序列中的某个 WEB SEND 命令失败，那么返回错误响应并且以后的发送也将失败。应用程序应对这种情况进行相应的处理。
   b. 如果成功发送所有块，但应用程序未发出带 CHUNKING(CHUNKEND) 的最终 WEB SEND 命令，那么异常终止事务，且带有异常终止代码 AWBP。 这是必然出现的情况，因为 CICS 无法保证分块的消息是完整的并且是正确的，因而无法代表应用程序发出最终的空块。

   接收方应忽略不完整的分块消息并丢弃它。Web 客户机会决定是否重试请求。
9. 对于作为 HTTP 客户机的 CICS，如下所示处理错误:
   a. 如果您的应用程序在已分块 transfer-coding 过程的任何时候获知到错误，那么使用 WEB CLOSE 命令停止该过程并关闭连接。 服务器将不接收最终的空块，因此应忽略和丢弃迄今为止您已发送的数据。您可以决定是否重试请求。
   b. 如果您不发送最终空块或发出 WEB CLOSE 命令，那么在终止对 CWBO（CICS Web Support 消息的瞬时数据队列）的任务时写警告消息。 服务器应使接收超时并忽略和丢弃您已发送的数据。

跨 HTTP 请求序列管理应用程序状态

CICS 为 Web 客户机发出的每个请求启动新的别名事务和新的程序。这是用于以下各项的情况: 管道化的请求、使用持续连接发出的请求、形成逻辑序列的请求和个别的单机请求。您需要考虑如何在请求之间管理应用程序的状态。如果需要跨请求序列，在不同的程序间或同一程序的不同实例间共享数据，那么可以使用 CICS 管理的资源或使用 Web 客户机发送的请求的元素执行此操作。

关于此任务

为了成功完成任务需要 Web 客户机和 CICS 之间的多个请求和响应交换时，序列中的每个新步骤都由 Web 客户机启动。您可以设计 CICS 发送的响应以指导 Web 客户机和 Web 客户机的任何个人用户到下一步。例如，实体主体可以包含最终用户可用于组成下一个请求的控件（例如，链接或按钮）。然而，您不能简单地实现请求的正确序列。特殊情况下，如果发生以下情况，那么规划的序列会被破坏:

- 客户机是 Web 浏览器，且最终用户输入已知 URL 以启动特殊请求，而不是选择先前响应提供的 HTML 页面中的控件。
- 通过关闭 Web 客户机或通过更改为 Web 客户机的某个替换活动，最终用户将完全放弃活动。

最终用户也可能延迟序列中任何请求的启动。

应该设计您的应用程序，使它们可以应付请求序列中的延迟或破坏。例如，如果正在跨请求序列共享数据，那么如果该请求序列未完成或过分延迟，那么应该确保清除了数据。如果您的应用程序更新受保护的资源，那么应该确保必须一起提交或复原的更新要在同一事务中进行。（这意味着应该设计来自 Web 客户机的单个请求以完成更新。）

应用程序的理想情形是：请求和响应的每次交换是自包含的，且完成任务的独立元素。然而，此设计并不总是可行，特别是当任务复杂时或当 Web 客户机已发送管道化的请求序列时。可能需要伪会话模型，其中应用程序的状态必须在请求间进行管理。这可以使用以下技术来安排:

- 您可以设计 Web 客户机发送的请求，以便应用程序状态或共享数据合并在请求中，例如，作为 Web 客户机提交 HTML 表单时使用的请求 URL 的一部分。 下一个程序可以检查请求 URL 以获取共享数据。
- 您可以使用隐藏字段在作为响应返回给 Web 客户机的 HTML 表单中存储少量的应用程序状态。 当用户执行计划序列中的下一个操作时，他们发送到 CICS 的请求可以包括这些隐藏字段，下一个应用程序会找到它们并读取。
- 对于数量较大的状态，以及具有扩展生存期的状态，您可创建 CICS 管理的资源来维持应用程序的状态并传递表示该资源的令牌。 CICS 提供样本状态管理程序（DFH$WBST 和 DFH$WBSR），它们在主存储器或临时存储器队列中存储应用程序状态，并提供应用程序可用于访问该信息的令牌。 令牌在伪会话中可作为 HTML 表单中的隐藏字段在程序间转换，或作为 URL 中的查询字符串在交互间转换。 该技术可用于保存整个伪会话中的信息，还可用于保存最终用户和各种 CICS 应用程序间的整个扩展交互中的信息，这也许要跨几个伪会话。

第 7 章 作为 HTTP 服务器的 CICS 的资源定义

关于此任务

CICS 提供的资源定义组 DFHWEB 包含下列 CICS Web Support 资源:

- CICS Web Support 任务（例如，CWBA 和 CWXN）的事务定义
- CICS Web Support 实用程序，包括:
  - 缺省分析器程序 DFHWBAAX 和样本分析器程序 DFHWBADX
  - Web 出错程序 DFHWBEP 和 DFHWBERX
- 临时存储模型 DFHWEB

CICS Web Support 消息的瞬时数据队列 CWBO（对于大多数消息）和 CWBW（Warning 头消息的独立队列）在组 DFHDCTG 中。

资源定义组 DFH$WEB 包含样本 CICS Web Support 应用程序的大多数 PROGRAM 资源定义和 URIMAP 定义。

您需要为每个要执行的 CICS Web Support 任务创建一些其他资源定义。第 55 页的第 5 章，『规划作为 HTTP 服务器的 CICS 的 CICS Web Support 体系结构』提供了规划指导，其指定执行每个任务所需的资源定义。您可能需要设置的资源定义如下:

- 创建 TCPIPSERVICE 资源定义以定义每个端口，您使用这些端口接收用于 CICS Web Support 的入站 HTTP 请求。您可以在该资源定义中指定要应用于每个端口的安全措施，以及端口操作的技术信息。『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』介绍了如何执行该操作。
- 可选: 为要用于入站 HTTP 请求处理的任何别名事务创建 TRANSACTION 资源定义。第 86 页的『为 CICS Web Support 创建 TRANSACTION 资源定义』介绍了如何执行该操作。
- 创建 URIMAP 资源定义，以为 CICS（作为 HTTP 服务器）的每个 HTTP 请求提供处理信息。
  1. 对于作为 HTTP 服务器的 CICS 的所有 HTTP 请求，按第 87 页的『为作为 HTTP 服务器的 CICS 的任何请求启动 URIMAP 资源定义』中的步骤执行，以开始定义。
  2. 对于提供了应用程序生成的响应的 HTTP 请求，按第 89 页的『为作为 HTTP 服务器的 CICS 的 HTTP 请求的应用程序响应完成 URIMAP 定义』中的步骤执行。
  3. 对于提供了静态响应的 HTTP 请求，按第 90 页的『为作为 HTTP 服务器的 CICS 的 HTTP 请求的静态响应完成 URIMAP 定义』中的步骤执行。

为 CICS Web Support 创建 TCPIPSERVICE 资源定义

TCPIPSERVICE 资源定义用于定义端口和 CICS 服务（包括 CICS Web Support）之间的关联。为您用于 CICS Web Support 的每个端口定义和安装 TCPIPSERVICE 资源定义。

关于此任务

CICS 系统中活动的每个 TCPIPSERVICE 定义必须指定唯一的端口号。在端口接收到入站 TCP/IP 连接请求时，CICS 将 TCPIPSERVICE 定义用于该端口，以确定要调用哪个 CICS 服务。使用 PROTOCOL 属性确定该服务。为标准 CICS Web Support 指定 HTTP，而为使用 CICS Web Support 处理的非 HTTP 请求指定 USER。

对于 CICS Web Support，为用于因特网服务的缺省或熟知端口号创建 TCPIPSERVICE 定义。对于 HTTP，缺省端口号为 80，而对于 HTTPS，缺省端口号为 443。您还可以使用非标准的端口号。

每个 TCPIPSERVICE 定义只能为 Web 连接任务指定一个分析器程序和一个事务定义。如果您需要使用这些项中的多个项，那么必须使用不同的 TCPIPSERVICE 定义并因而使用不同的端口。

CICS 为组 DFH$SOT 中的 CICS Web Support 提供样本 TCPIPSERVICE 定义:

HTTPNSSL
不具有 SSL 支持的 CICS Web TCPIPSERVICE

HTTPSSL
具有 SSL 支持的 CICS Web TCPIPSERVICE

要点: 使用 TCPIPSERVICE 资源定义指定应用于每个端口的安全措施。您可以选择是否使用 SSL；如果您使用 SSL，请选择要应用的确切安全措施；例如，认证方法、通过客户机和服务器发送证书以及消息的加密。有关可用于保证 CICS Web Support 设施安全的安全措施的更多信息，请参阅 第 151 页的第 13 章，『CICS Web Support 的安全性』。

CICS Resource Definition Guide 介绍了资源定义的各种方法，并提供了有关将要在该过程中使用的所有 TCPIPSERVICE 资源定义属性的参考信息。

1. 确定要用于 CICS Web Support 的 TCP/IP 端口。 建议您保留该端口号以供 CICS Web Support 使用。有关端口用途的信息，请参阅第 50 页的『保留 CICS Web Support 的端口』。
2. 使用 *CICS Resource Definition Guide* 中列出的某种方法，使 TCPIPSERVICE 定义以您选择的名称和组开头。 当您在该端口为入站 HTTP 请求设置 URIMAP 定义时，请指定 TCPIPSERVICE 定义的名称。
3. 使用 STATUS 属性指定 CICS 是否在安装该定义后立即开始侦听该服务。 如果您指定 CLOSED，那么必须在可以使用该服务之前将它设置为打开。您可以使用 CEMT 事务或 SET TCPIPSERVICE 系统编程命令将服务设置为打开或关闭。
4. 指定 PORTNUMBER 属性作为该定义适用的 TCP/IP 端口的号码。
5. 使用 HOST 属性指定 TCPIPSERVICE 将侦听入站连接的点分十进制或冒号十六进制 IP 地址。您还可以使用 IPADDRESS 属性为现有程序指定点分十进制 IP 地址。另外，为了配置多个 IP 协议集，您可以指定 INADDR_ANY 以使 CICS 尝试绑定到定义它的每个协议集上的端口。如果您具有一个多协议集的 CINET 环境，并且只希望向缺省的 TCP/IP 协议集分配亲缘关系，那么您可以指定 DEFAULT 执行该操作。 有关该 TCPIPSERVICE 资源定义属性的参考信息详细列举了其他一些注意事项，如果您希望多个 CICS 区域共享该 TCPIPSERVICE 定义，或希望多个 CICS 区域绑定到它指定的端口号，那么应特别留意这些注意事项。

6. 使用 DNSGROUP 和 GRPCRITICAL 属性指定服务在综合系统域中使用的 DNS 组名，以及该服务的关键状态。 该信息允许 CICS 注册到工作负载管理器，以优化 DNS 连接。*Java Applications in CICS* 具有更多关于这方面的信息。
7. 使用 PROTOCOL 属性指定 CICS Web Support 在该端口上处理请求。
   a. 对于常规 HTTP 请求，指定 HTTP。 如果您指定端口 80 或 443，那么 CICS 会强制使用 HTTP。该选项适用带 SSL 的 HTTP 和不带 SSL 的 HTTP。SSL 选项指定是否包含 SSL。
   b. 为使用 CICS Web Support 处理的非 HTTP 请求指定 USER。 指定 USER 后，CICS Web Support 用于处理请求，但是对于使用该协议发送和接收的消息，不执行验收检查。请求标记为非 HTTP 并直接传递到分析器程序。 URIMAP 定义不用于这些请求。
8. 将 TRANSACTION 属性指定为 Web 连接任务的 4 字符标识；对于 HTTP 请求，它通常是 CWXN，而对于非 HTTP（USER 协议）请求，它通常是 CWXU。 该任务处理请求的初始处理。如果您指定端口 80 或 443，那么 CICS 提供 CWXN 作为缺省值。为了满足记帐或监控目的，您可以为必须运行程序 DFHWBXN 的 CWXN 或 CWXU 指定别名。
9. 指定 URM 属性作为与该 TCPIPSERVICE 定义关联的分析器程序的名称。 对于非 HTTP（USER 协议）请求，总是使用分析器程序。 对于 HTTP 请求，如果 URIMAP 定义指定使用分析器程序，或者如果不存在 URIMAP 定义，那么分析器程序用于解释该请求。您必须指定分析器程序。只能为每个 TCPIPSERVICE 定义选择一个分析器程序，但是您可以对该分析器程序进行编码以处理任何请求。 如果您打算使用 URIMAP 定义处理您所有的 HTTP 请求，第 111 页的第 10 章，『分析器程序』会介绍关于您的分析器程序必须提供的基本支持的信息。第 55 页的第 5 章，『规划作为 HTTP 服务器的 CICS 的 CICS Web Support 体系结构』中的体系结构指导可帮助您确定是否将分析器程序用于任何特定的 HTTP 请求。
10. 使用 SOCKETCLOSE 属性指定 CICS 在发出接收套接字上的入站数据的请求后并在关闭该套接字前要等待多久。 NO 意味着套接字保持打开状态直到接收到数据为止或直到 Web 客户机关闭它为止。为防止速度较慢的 Web 客户机或连接中断的 Web 客户机阻塞套接字，请指定超时值而不是指定 NO。当 Web 连接任务在建立连接后发出第一条接收命令时，将忽略该超时值，并且该任务将等待由 CICS 确定的一段时间（对于 HTTP 为 30 秒）以接收来自 Web 客户机的数据。这种延迟将防止套接字连接一启动就关闭（即使没有立即可用的数据），从而防止 Web 客户机上出现连接复位错误。

    注：对于 CICS Web Support，请勿为 SOCKETCLOSE 指定零设置。SOCKETCLOSE(0) 意味着即使 Web 客户机请求持续连接，也无法维持它。
11. 使用 BACKLOG 属性指定 TCP/IP 开始拒绝来自 Web 客户机的入站请求之前可排队的连接数。 缺省值为 1。
12. 使用 MAXDATALEN 属性指定该连接上可以接收的最长数据长度。 缺省值为 32 KB，最大值为 524 288 KB。该选项有助于防止在涉及大量数据传输时受到拒绝服务的攻击。
13. 使用 SSL 属性指定是否将安全套接字层（SSL）用于该端口。 YES 意味着使用 SSL，且 CICS 将服务器证书发送到 Web 客户机。CLIENTAUTH 意味着使用 SSL，以及除了 CICS 将服务器证书发送到 Web 客户机，还请求 Web 客户机将客户机证书发送到 CICS。如果您指定端口号 443，那么 CICS 提供 YES 作为缺

省值，如果您指定端口号 80，那么强制指定为 NO。 第 151 页的第 13 章，『CICS Web Support 的安全性』说明了如果您要使用 SSL，需执行的操作。

14. 如果您已指定 SSL（YES）或 SSL（CLIENTAUTH），那么使用 CERTIFICATE 属性指定 X.509 证书的标号，CICS 在 SSL 握手期间将该证书用作服务器证书。 如果省略该属性，那么使用 CICS 区域用户标识的密钥环中定义的缺省证书。该证书必须存储在外部安全性管理器数据库中的密钥环中。第 151 页的第 13 章，『CICS Web Support 的安全性』具有关于使用这些证书的更多信息。
15. 使用 AUTHENTICATE 属性指定用于在该端口发出请求的 Web 客户机的认证级别。 第 151 页的第 13 章，『CICS Web Support 的安全性』说明了有关认证和身份识别的信息。
    a. 如果发送认证或身份识别信息不需要 Web 客户机，那么指定 NO。 如果客户机发送已向安全管理器注册的有效证书，那么 CICS 可以使用它。
    b. 指定 BASIC 以使 CICS 尝试 HTTP 基本认证，在这种情况下，CICS 从 Web 客户机请求用户标识和密码。 第 18 页的『HTTP 基本认证』更详细地说明了基本认证。
    c. 指定 CERTIFICATE 以使用 SSL 客户机证书认证。 Web 客户机必须发送已向安全管理器注册并与用户标识关联的有效证书。如果未接收到有效证书，或者证书与用户标识无关，那么拒绝连接。 如果使用该选项，那么必须指定 SSL(CLIENTAUTH)。
    d. 指定 AUTOREGISTER 以使用向安全管理器自动注册的 SSL 客户机证书认证。Web 客户机必须发送有效证书。如果 CICS 发现该证书尚未注册到安全管理器，那么使用 HTTP 基本认证来请求用户标识和密码，CICS 还会使用该信息来注册该证书。 如果使用该选项，那么必须指定 SSL(CLIENTAUTH)。
    e. 指定 AUTOMATIC 以将 SSL 客户机证书认证用于安全管理器的自动注册（关于 AUTOREGISTER 选项），或如果未发送证书，那么使用 HTTP 基本认证（关于 BASIC 选项）。
16. 使用 REALM 属性来指定 HTTP 基本认证所用的域。 在基本认证过程中，最终用户会看见该域。它标识了所请求的认证信息（即用户标识和密码）将应用的资源集。
    a. 如果使用不同 TCPIPSERVICE 定义提供的资源需要不同的认证信息，请指定不同的域以便最终用户了解该需求。
    b. 如果最终用户在您的各项资源中使用相同的认证信息，那么您可以在多个 TCPIPSERVICE 定义中指定相同的域。
    c. 如果您没有指定 REALM 属性，那么使用缺省域。 缺省域为:

       ```
       realm="CICS application aaaaaaaa"
       ```

       其中 *aaaaaaaa* 是 CICS 区域的应用程序标识。

为 CICS Web Support 创建 TRANSACTION 资源定义

TRANSACTION 资源定义用于为 CICS Web Support 定义别名事务。别名事务处理 HTTP 请求的稍后处理阶段，包括接收请求、执行应用程序的业务逻辑、HTTP 响应的构造和 HTTP 响应的代码页转换。别名事务也可用于处理非 HTTP 请求。

关于此任务

CICS 为缺省别名事务 CWBA 提供资源定义。您可以为了以下目的想使用备用别名事务名:

- 审计、监控或记帐
- 对安全性的资源和命令检查
- 分配启动优先级
- 分配 DB2® 资源
- 将不同的失控值指定给不同的 CICS 应用程序
- 事务类限制

您可以根据需要设置足够数量的别名事务定义。您可以使用 URIMAP 定义或分析器程序指定特殊请求需要的别名事务。

要点: 确保用于应用程序生成的响应（如 CWBA）的别名事务的优先级等于或高于与 Web 连接任务（如 CWXN 或 CWXU）关联的事务的优先级。*CICS Performance Guide* 说明了这一点的重要性。

CICS Resource Definition Guide 提供了有关该类型的资源定义的完整指示信息。当您正在按这些指示信息进行操作时，请记住:

- 在 CWBA 定义的基础上建立您的别名事务定义，进行您需要的任何更改，例如对优先级的更改。 CWBA 的定义如下:

```
DEFINE TRANSACTION(CWBA)   GROUP(DFHWEB)
       PROGRAM(DFHWBA)     TWASIZE(0)
       PROFILE(DFHCICST)   STATUS(ENABLED)
       TASKDATALOC(BELOW)  TASKDATAKEY(USER)
       RUNAWAY(SYSTEM)     SHUTDOWN(ENABLED)
       PRIORITY(1)         TRANCLASS(DFHTCL00)
       DTIMOUT(NO)         INDOUBT(BACKOUT)
       SPURGE(YES)         TPURGE(NO)
       RESSEC(NO)          CMDSEC(NO)
```

- 您的别名事务定义必须使用 CICS 提供的别名程序 DFHWBA。 别名程序调用您已为处理请求而指定的用户应用程序。
- 您的别名事务定义必须是本地事务。

为作为 HTTP 服务器的 CICS 的任何请求启动 URIMAP 资源定义

URIMAP 资源定义用于定义如何处理 HTTP 请求。对于作为 HTTP 服务器的 CICS 的任何 HTTP 请求，通过为 Web 客户机的请求指定期望的 URL 的组成部分（方案、主机和路径）和其他基本信息来启动 URIMAP 定义。

开始之前

如果您还未计划如何向作为 HTTP 服务器的 CICS 的 HTTP 请求提供响应，第 55 页的『用支持 Web 的应用程序提供动态 HTTP 响应』和第 59 页的『用 CICS 文档模板或 z/OS UNIX 文件提供静态 HTTP 响应』会介绍如何操作。

关于此任务

CICS Resource Definition Guide 介绍了资源定义的各种方法，并提供了关于所有将在该过程中使用的 URIMAP 资源定义属性的完整参考信息。

1. 识别您计划作为来自 Web 客户机的 HTTP 请求而接收的 URL。 该 URL 表示您计划通过 CICS 可用于 Web 客户机的资源。
2. 将 HTTP 请求的 URL 划分成它的方案、主机和路径部分。 第 10 页的『URL 的组成部分』说明了这些每一个组成部分以及如何对它们定界。 例如，在 URL `http://www.example.com/software/index.html` 中:
   - 方案部分是 `http`
   - 主机部分是 `www.example.com`
   - 路径部分是 `/software/index.html`

   如果您希望 URIMAP 定义与多个路径匹配，那么可以在路径结束处使用星号作为通配符。例如，指定路径 `/software/*` 会使 URIMAP 定义与路径以字符串 `/software/` 开头的所有请求匹配。如果多个带通配符的 URIMAP 定义与 HTTP 请求匹配，那么采用最特定的匹配。
3. 如果查询部分出现在 URL 中，并且您希望 URIMAP 定义只与该特定查询匹配，那么可以包含该特定查询作为 PATH 规范的一部分。 可以在包含星号作为通配符的路径后面使用查询字符串，但查询字符串本身不能包含星号作为通配符。必须指定完整和精确的查询字符串。 对于带有 CICS 文档模板的静态响应，可使用查询字符串选择 URIMAP 定义或可将它替换到文档模板中，但是两者无法同时发生。如果在 URIMAP 定义中不包含查询字符串，那么只进行路径匹配，且为了匹配会自动忽略请求中出现的任何查询字符串。
4. 使 URIMAP 定义以您选择的名称和组开头，如 *CICS Resource Definition Guide* 中所述。
5. 使用 STATUS 属性指定在启用状态还是在禁用状态安装 URIMAP 定义。
6. 指定 SERVER（作为 HTTP 服务器的 CICS）的 USAGE 属性。
7. 指定 SCHEME 属性作为 HTTP 请求的 URL 的方案部分。可使用 HTTP（不带 SSL）或 HTTPS（带 SSL）。 不要包含跟随方案部分的定界符 `://`。 指定 HTTP 方案的 URIMAP 接受使用 HTTP 方案或更安全的 HTTPS 方案发出的 Web 客户机请求。指定 HTTPS 方案的 URIMAP 仅接受使用 HTTPS 方案发出的 Web 客户机请求。
8. 可选: 指定 TCPIPSERVICE 属性作为 TCPIPSERVICE 定义的名称，它定义与该 URIMAP 定义相关的入站端口。 如果不指定该属性，那么 URIMAP 定义将应用于任何入站端口上的匹配 HTTP 请求。 当将 HTTPS（带 SSL 的 HTTP）作为方案的 URIMAP 定义与 Web 客户机发出的请求进行匹配时，CICS 检查该请求使用的入站端口是否使用 SSL。如果没有为该端口指定 SSL，那么将拒绝该请求，状态码为 403（禁止）。当 URIMAP 定义应用到所有入站端口时，该检查会确保 Web 客户机无法使用不安全的端口来访问安全的资源。由于不会对采用 HTTP 方案的 URIMAP 定义进行任何检查，所以 Web 客户机可以使用非安全或安全的（SSL）端口访问这些资源。
9. 如果您需要区别包含不同主机名的 URL，那么指定 HOST 属性作为 HTTP 请求的 URL 的主机部分。 不要包含端口号。 IPv4 或 IPv6 地址可用作主机名。 如果您指定单个星号作为 HOST 属性，那么 URIMAP 定义与入站 URL 中的任何主机名匹配。如果您不使用多个主机名或不打算区别它们，请使用该选项。
10. 指定 PATH 属性作为 HTTP 请求的 URL 的路径部分，如果需要，包含星号作为通配符。 您可以在路径部分开始包含或省略定界符 /（正斜杠）；如果您省略它，

CICS 会自动提供它。 如果查询部分出现在 URL 中，并且您希望只将 URIMAP 定义应用于该特定查询，那么包含该特定查询作为路径部分的一部分（在字符串的开始包含问号）。

注: 除非也安装了更多特定的 URIMAP 定义（在这种情况下，会进行最特定的匹配），否则将 PATH 属性指定为 /* 会使 URIMAP 定义与导向到主机（以 HOST 属性命名）的所有请求相匹配。

11. 现在，来完成您的定义:
    a. 对于应用程序生成的响应，按照『为作为 HTTP 服务器的 CICS 的 HTTP 请求的应用程序响应完成 URIMAP 定义』中的指示信息操作。
    b. 对于静态响应，按照第 90 页的『为作为 HTTP 服务器的 CICS 的 HTTP 请求的静态响应完成 URIMAP 定义』中的指示信息操作。

为作为 HTTP 服务器的 CICS 的 HTTP 请求的应用程序响应完成 URIMAP 定义

在您已通过指定期望的 URL 的组成部分（方案、主机和路径）和其他基本信息启动 URIMAP 定义时，如果您要向 HTTP 请求提供应用程序生成的响应，通过提供有关应处理请求和提供 HTTP 请求的一个或多个应用程序的信息来完成该定义。

开始之前

如果您还未计划如何向作为 HTTP 服务器的 CICS 的 HTTP 请求提供响应，第 55 页的『用支持 Web 的应用程序提供动态 HTTP 响应』会介绍如何操作。然后您需要如第 87 页的『为作为 HTTP 服务器的 CICS 的任何请求启动 URIMAP 资源定义』中描述的启动您的 URIMAP 定义。

关于此任务

当您已规划应用程序生成的响应并开始 URIMAP 定义后，按本主题中的指示信息完成该定义。*CICS Resource Definition Guide* 介绍了资源定义的各种方法，并提供了关于所有将在该过程中使用的 URIMAP 资源定义属性的完整参考信息。

1. 在该请求的处理路径中，如果涉及与该 URIMAP 定义相关的 TCPIPSERVICE 定义（或多个定义）关联的分析器程序，那么为 ANALYZER 属性指定 YES 以激活它。如果使用分析器程序，您仍然可以指定 CONVERTER、TRANSACTION、USERID 和 PROGRAM 属性。为这些属性指定的值将用作分析器程序的输入，但分析器程序可以覆盖这些值。也可以将这些属性保留为空白，由分析器程序来指定。
2. 如果您要使用转换器程序，那么指定 CONVERTER 属性作为该程序的名称。 该程序可以是 CICS 中可用的任何转换器程序；转换器程序和 TCPIPSERVICE 定义之间没有关联，这是因为分析器程序与该定义之间有关联。如果使用转换器程序，那么您仍然可以指定 PROGRAM 属性。您为该属性指定的值用作转换器程序的输入。转换器程序可以更改 PROGRAM 属性以指定不同的应用程序来处理请求。
3. 指定 TRANSACTION 属性作为 CICS 中可用的别名事务的名称，CICS 可使用该别名事务运行提供响应的应用程序。 缺省别名事务是 CWBA。如果指定 ANALYZER (YES)，那么事务名称还可以由分析器程序更改或提供。
4. 指定 USERID 属性作为附加别名事务所凭借的用户标识。 您在 URIMAP 定义中指定的用户标识会被直接从 Web 客户机获取（由连接的 TCPIPSERVICE 定义的 AUTHENTICATE 属性指定）的任何用户标识覆盖。如果指定 ANALYZER

(YES)，那么分析器程序可以更改这些用户标识中的任意一个，或提供一个。如果未通过这些方式中的任何一种指定用户标识，那么缺省用户标识为 CICS 缺省用户。

5. 指定 PROGRAM 属性作为向请求提供响应的应用程序的名称。 如果 URIMAP 定义中未指定分析器程序或转换器程序，那么 HTTP 请求直接被传递到该应用程序。如果指定分析器程序或转换器程序，那么您可以保留该属性为空白并让分析器程序或转换器程序指定它。

为作为 HTTP 服务器的 CICS 的 HTTP 请求的静态响应完成 URIMAP 定义

对于静态响应，当通过指定期望的 URL 的组成部分（方案、主机和路径）和其他基本信息来启动 URIMAP 定义时，使用文档模板或 z/OS UNIX 文件，通过提供 CICS 构造请求的静态响应时所需的信息来完成该定义。您可将通配符用于路径匹配，在这种情况中，CICS 将通配符所涵盖的每个 HTTP 请求的路径部分替换为模板名称或 z/OS UNIX 文件路径的最后一部分。

开始之前

如果您还未计划如何向 HTTP 请求提供静态响应，第 59 页的『用 CICS 文档模板或 z/OS UNIX 文件提供静态 HTTP 响应』会介绍如何这样操作。然后您需要如第 87 页的『为作为 HTTP 服务器的 CICS 的任何请求启动 URIMAP 资源定义』中描述的启动您的 URIMAP 定义。

关于此任务

当您已规划静态响应并开始 URIMAP 定义后，按本主题中的指示信息完成该定义。*CICS Resource Definition Guide* 介绍了资源定义的各种方法，并提供了关于所有将在该过程中使用的 URIMAP 资源定义属性的完整参考信息。

注: URIMAP 定义不用于控制作为静态响应交付的 CICS 文档模板和 z/OS UNIX 文件的安全性。如果要使用基本认证和资源级的安全性来保护这些项，请参阅 第 151 页的第 13 章，『CICS Web Support 的安全性』，其中说明了如何进行设置。

1. 指定 MEDIATYPE 属性作为 CICS 提供的静态响应的数据内容。 例如，`text/html` 或 `text/xml` 分别是 HTML 和 XML 数据内容的名称。 （请参阅 第 10 页的『IANA 介质类型和字符集』以获取有关介质类型的更多信息。） 由于该属性没有缺省值，因此必须为它指定值。 CICS 使用该信息创建响应的 Content-Type 头。
2. 如果静态响应是从文本文档（文档模板或 z/OS UNIX 文件）形成的，那么指定代码页转换所需的属性。在 MEDIATYPE 属性指定数据内容的文本类型的情况下，才进行代码页转换。
   a. 指定 CHARACTERSET 属性作为将静态响应发送到 Web 客户机之前 CICS 将它转换所至的字符集。 CICS 不支持 IANA 命名的所有字符集。第 333 页的附录 A，『HTML 编码字符集』列出了 CICS 支持的 IANA 字符集。 该信息包含在响应的 Content-Type 头中。
   b. 将 HOSTCODEPAGE 属性指定为编码静态文档的 IBM 代码页（EBCDIC）。
3. 如果要使用 CICS 文档模板形成静态响应，那么指定 TEMPLATENAME 属性作为该文档模板的名称。 必须使用 DOCTEMPLATE 资源定义来定义文档模板。 如果您要使用路径匹配，那么在 CICS 文档模板的名称结束处，以及 PATH 属性指定的

路径结束处包含星号作为通配符。CICS 采用该通配符涵盖的每个 HTTP 请求的路径部分，并将它替换为模板名称的最后一部分。 *CICS Resource Definition Guide* URIMAP 定义属性具有这方面的示例。

当指定 TEMPLATENAME 属性时，如果查询字符串出现在 URL 中，那么 CICS 将查询字符串的内容作为符号列表传递到指定的 CICS 文档模板。只有 URIMAP 定义的 PATH 属性中还未使用查询字符串时才会发生这种情况。

4. 如果正在使用 z/OS UNIX 文件形成静态响应，那么指定 HFSFILE 属性作为该文件的标准（绝对）名称或相对名称。 z/OS UNIX 文件可指定为以斜杠开始的绝对或标准路径，或指定为不以斜杠开始的相对路径。相对路径相对于 CICS 区域用户标识的 HOME 目录。 CICS 区域必须具有 z/OS UNIX 的访问许可权，并且必须具有包含文件的 z/OS UNIX 目录及文件的访问许可权。*Java Applications in CICS* 说明了如何授予这些许可权。 如果您要使用路径匹配，那么在 z/OS UNIX 文件的路径的结束处包括作为通配符的星号，并还在 PATH 属性指定的路径的结束处包括星号。CICS 获取每个 HTTP 请求路径中以通配符涵盖的部分，并将其替换为 z/OS UNIX 文件路径的最后一部分。 *CICS Resource Definition Guide* 具有这方面的示例。

   注: 不能在 HFSFILE 规范中单独使用星号。必须指定至少一个目录结构级别。不能将查询字符串替换成 z/OS UNIX 文件。

第 8 章 管理作为 HTTP 服务器的 CICS

已配置 CICS 以执行各种 CICS Web Support 任务，且已开始响应来自 Web 客户机的请求后，您可能需要执行某些管理活动以管理您的 CICS Web Support 结构，并在资源不可用时，对请求提供适当的处理。

关于此任务

通过用 URIMAP 定义来管理您的 HTTP 请求，可以轻松地管理作为 HTTP 服务器的 CICS。URIMAP 定义允许您执行以下操作:

- 如果特定的 HTTP 请求所需的资源（例如，CICS 程序）不可用，那么在运行中的 CICS 系统中动态重定向或拒绝这些请求。
- 让 CICS 创建虚拟主机，可以使用 CICS 命令来管理它们。

如果没有 URIMAP 定义，您可以在 TCPIPSERVICE 资源定义级别管理 CICS Web Support，这将管理特殊端口上的所有请求，但是如果在 URIMAP 资源定义级别管理的话，控制程度和详细程度就更高。

- 您可以使用 CICS 系统编程接口和 CICS 提供的事务创建、安装、更新和删除 CICS Web Support 资源，包括 TCPIPSERVICE、URIMAP 和 TRANSACTION 资源定义。『管理 CICS Web Support 资源』说明您可用于管理这些资源的命令。
- 您可使用 INQUIRE HOST 命令和虚拟主机浏览命令查看 CICS 从您的 URIMAP 定义创建的虚拟主机，并使用 SET HOST 命令更改它们的状态。 第 95 页的『管理虚拟主管』介绍了如何执行该操作。
- 如果 CICS 系统中的应用程序或资源暂时不可用，并且您要通过重定向提供另一种方式，那么可以重定向由 URIMAP 定义处理的 HTTP 请求，并在资源再次可用时除去重定向。如果永久更改了资源的位置或请求 URL 格式，那么可以设置永久重定向。第 96 页的『将 HTTP 请求重定向到另一个 URL』介绍了如何执行该操作。
- 如果 CICS 系统中的应用程序或资源暂时不可用，并且您无法通过重定向提供另一种方式，或者如果已永久除去应用程序或资源，那么您可以通过禁用资源定义来拒绝几个不同级别（单独请求 URL、虚拟主机、端口或所有端口）的 HTTP 请求。 您可使用这些方法终止全部或部分 CICS Web Support 服务。 第 97 页的『拒绝 HTTP 请求』介绍了如何执行该操作。
- 您可能希望为每个主机提供 favicon 或 robots.txt 文件。 许多 Web 浏览器会自动请求这些项。 第 98 页的『提供 favicon』 和第 99 页的『提供 robots.txt 文件』介绍了如何提供这些项。
- CICS 记录从入站消息中的警告标题到瞬时数据队列 CWBW 的信息。 通常该信息供用户阅读。第 101 页的『Warning 头』介绍了该信息。

管理 CICS Web Support 资源

您可以使用 CICS 系统编程接口和 CICS 提供的事务 CEMT 和 CEDA 来创建、安装、更新和删除 CICS Web Support 资源。

关于此任务

CICS 提供的事务 CEDA 可用于创建和安装 TCPIPSERVICE、URIMAP、TRANSACTION 和 DOCTEMPLATE 资源定义。第 83 页的第 7 章，『作为 HTTP 服务器的 CICS 的资源定义』提供了有关为 CICS Web Support 设置资源定义的更多信息。

CICS 系统编程接口包括用于 CICS Web Support 管理的以下命令:

INQUIRE TCPIP

查询 CICS 系统中的 TCP/IP 支持状态。该命令返回 TCP/IP 的打开状态和 CICS 区域中的当前最大 IP 套接字数。

SET TCPIP

您可以使用此命令打开或关闭 TCP/IP 支持，无论是正常情况（允许活动任务完成）还是立即（异常终止活动任务）。关闭 TCP/IP 支持意味着拒绝所有入站和出站请求，且 CICS Web Support 完全停止。您还可以使用此命令增加或减少 CICS 区域中的最大 IP 套接字数。如果您没有超级用户权限，那么您可以设置的限制较低，且 CICS 将通知您它是否已利用此较低限制。

CREATE TCPIPSERVICE

为端口创建 TCPIPSERVICE 定义。

DISCARD TCPIPSERVICE

删除端口的 TCPIPSERVICE 定义。可以丢弃 TCPIPSERVICE 定义前，必须关闭它（使用 SET TCPIPSERVICE 命令）。

INQUIRE TCPIPSERVICE

查询 TCPIPSERVICE 定义。象显示定义的属性一样，该查询显示定义的当前 DNS 注册状态和打开状态。还可以浏览 TCPIPSERVICE 定义。

SET TCPIPSERVICE

您可以使用此命令以更改 TCPIPSERVICE 定义的请求队列限制、数据接收限制、DNS 注册状态或分析器程序。您还可以使用此命令关闭 TCPIPSERVICE 定义。您可以选择停止侦听端口的方式: 正常 - 允许活动的任务完成; 立即 - 异常终止活动的任务。

CREATE URIMAP

为请求创建 URIMAP 定义。

DISCARD URIMAP

删除 URIMAP 定义。由已删除定义处理的请求可能与不太特定的 URIMAP 定义匹配，该 URIMAP 定义的路径中有通配符。否则，它们将传递到 TCPIPSERVICE 定义的分析器程序。如果要拒绝请求而不进行此可选处理，那么禁用 URIMAP 定义，而不是丢弃它。

INQUIRE URIMAP

查询 URIMAP 定义。象显示定义的属性一样，该查询说明: 是否已在个别基础上禁用了 URIMAP，或者它是否因为虚拟主机（它是该虚拟主机的一部分）已被禁用而不可用。还可以浏览 URIMAP 定义。

SET URIMAP

您可以使用此命令以启用或禁用 URIMAP 定义。禁用 URIMAP 定义后，CICS

通过 Web 出错程序将 HTTP 503 响应（服务不可用）发送到 Web 客户机。您还可以使用此命令来设置 LOCATION 和 REDIRECTTYPE 属性，以指定重定向或结束重定向。

INQUIRE HOST
查询虚拟主机。还可以浏览虚拟主机。『管理虚拟主管』说明了如何管理虚拟主机。

SET HOST
启用或禁用虚拟主机。『管理虚拟主管』说明了如何管理虚拟主机。

您用于 CICS Web Support 的 TRANSACTION、DOCTEMPLATE 和 PROGRAM 资源也可以使用 SPI 命令来管理。*CICS System Programming Reference* 提供了有关所有这些命令的完整信息。

CICS 提供的事务 CEMT 包括用于 CICS Web Support 管理的以下命令:

- INQUIRE TCPIP
- SET TCPIP
- INQUIRE TCPIPSERVICE
- SET TCPIPSERVICE
- INQUIRE URIMAP
- SET URIMAP
- INQUIRE HOST
- SET HOST

您可以使用 CEMT 来管理用于 CICS Web Support 的 TRANSACTION 和 PROGRAM 资源，并可以对 DOCTEMPLATE 资源进行查询。*CICS 提供的事务*提供了有关所有这些命令的完整信息，并说明了如何使用 CEMT。

CICSPlex® SM 还可以用于管理此处列出的资源。

管理虚拟主管

CICS 通过 URIMAP 资源定义对象支持虚拟主管。

您为作为 HTTP 服务器的 CICS（URIMAP 定义中具有 USAGE(SERVER)）设置的每个 URIMAP 定义都包含 Web 客户机期望在它的请求中提供的主机名。通过将 CICS 区域中指定同一主机名和同一 TCPIPSERVICE 定义的所有 URIMAP 定义一起分组成单个数据结构，CICS 自动为您创建虚拟主机。未指定 TCPIPSERVICE 定义并因而会应用该定义的所有 URIMAP 定义都会添加到指定匹配主机名的所有数据结构中，因此这些 URIMAP 定义可能属于多个数据结构。然后这些 URIMAP 定义组中的每一组都形成可作为一个单元管理的虚拟主机。

您可使用以下 CICS 命令管理 CICS 已从您的 URIMAP 定义创建的虚拟主机:

- INQUIRE HOST 命令用于查询虚拟主机的状态。该命令介绍虚拟主机的主机名、它关联的 TCPIPSERVICE 定义（或如果它与 CICS 区域中的每个 TCPIPSERVICE 定义关联）以及启用它还是禁用它。

- SET HOST 命令用于将虚拟主机的状态设置为启用或禁用。禁用虚拟主机意味着应用程序不能访问构成虚拟主机的所有 URIMAP 定义。（但要注意的是，不能丢弃已用这种方式禁用的 URIMAP 定义。）当禁用虚拟主机时，CICS 向 Web 客户机返回 HTTP 503 响应（服务不可用）。
- 虚拟主机浏览命令用于浏览 CICS 系统中的虚拟主机。

统计程序 DFH0STAT 包含一个报告，它显示 CICS 已创建的虚拟主机。

如果已删除构成虚拟主机的所有 URIMAP 定义，那么 CICS 自动删除虚拟主机。如果您不希望管理 CICS 已为您创建的虚拟主机，那么可以忽略它们，并在您的 URIMAP 定义的级别进行管理。

您还可以使用分析器程序处理虚拟主机。将 HTTP 请求的主机名传递给分析器程序，并且您可以对程序进行编码，以向请求提供与主机相关的响应。然而，无法使用 INQUIRE HOST、SET HOST 和虚拟主机浏览命令管理以这种方式设置的虚拟主机。

将 HTTP 请求重定向到另一个 URL

可以使用 URIMAP 定义将作为 HTTP 服务器的 CICS 的 HTTP 请求重定向到另一个 URL。

关于此任务

您可能希望应该总是通过将 Web 客户机重定向到另一个 URL 来提供资源。而当需要的资源不可用（例如，页面告诉请求方它们需要的应用程序已脱机）时，可能要使用重定向以提供对请求的临时响应。在任一情况下，都可以使用与请求匹配的 URIMAP 定义重定向请求，具体如下:

1. 找到要重定向的 URL 的 URIMAP 定义。
2. 使用 URIMAP 定义的 LOCATION 属性指定多达 255 个字符的 URL，匹配的 HTTP 请求被重定向到该 URL。 该 URL 必须是完整的 URL，包括方案、主机和路径部分。包括所有定界符。CICS 会检查 URL 是否完整以及是否正确定界，但 CICS 不检查目标是否有效。
   a. 可选: 您可以使用 LOCATION 属性中的片段标识（前面带 # 字符），以将 Web 浏览器指向 URL 标示项中的引用或功能。例如，片段标识可以是文档中子节的标识。请参考所提供内容类型（例如，HTML）的技术规范，以查看是否应使用以及如何使用片段标识。
3. 使用 URIMAP 定义的 REDIRECTTYPE 属性指定临时或永久重定向。 当在临时基础上重定向请求时，用于响应的 HTTP 状态码是 302（已找到）。当永久重定向请求时，用于响应的 HTTP 状态码是 301（永久移动）。 CICS 编写重定向响应且无法定制重定向响应。
4. 安装更改的 URIMAP 定义。 当指定 REDIRECTTYPE(TEMPORARY) 或 REDIRECTTYPE(PERMANENT) 时，URIMAP 定义中的 LOCATION 属性覆盖 URIMAP 定义中的其他任何属性，并重定向 HTTP 请求。 安装 URIMAP 定义后，您可使用 SET URIMAP LOCATION 命令更改 LOCATION 属性。
5. 如果资源再次变得可用时，使用命令 SET URIMAP REDIRECTTYPE(NONE) 关闭重定向并重新安装已更改的定义。 会保留 LOCATION 属性中指定的 URL，但除非您重新激活重定向，否则不使用该 URL。

拒绝 HTTP 请求

如果 CICS 系统中的应用程序或资源不可用，有时您可能需要拒绝 Web 客户机向作为 HTTP 服务器的 CICS 发出的请求。

关于此任务

您可以在几个不同的级别拒绝 HTTP 请求:

1. 在特定请求 URL 级别。要达到该详细程度级别，URIMAP 定义应包含请求 URL。如果您没有 URIMAP 定义，那么可通过更改处理 HTTP 请求的分析程序来修改这些请求的处理，但这不太方便。
2. 在虚拟主机级别（包含特定主机名的所有请求）。对于被合并到虚拟主机中的请求，它必须包含在 URIMAP 定义中。
3. 在端口级别。端口映射到 TCPIPSERVICE 定义。例如，对于缺省 HTTP 端口 80，禁用 TCPIPSERVICE 定义使 CICS 除了接收使用 SSL 或使用非标准端口的 HTTP 请求外，不会接收任何其他 HTTP 请求。
4. 完全在所有端口级别。在 CEMT 事务或 CPSM 中，您可关闭 CICS 内部 TCP/IP 套接字支持，并因而完全关闭 CICS Web Support。

一般地，如果您在越详细的级别拒绝 HTTP 请求，CICS 就会向 Web 客户机给出越合适和详尽的错误响应。例如，如果您通过禁用 URIMAP 定义或虚拟主机来拒绝 HTTP 请求，那么 CICS 通过 Web 出错程序将 HTTP 503 响应（服务不可用）返回到 Web 客户机。您可以定制 Web 出错程序来修改该响应。然而，如果您通过禁用 TCPIPSERVICE 定义来拒绝 HTTP 请求，那么 Web 客户机将只收到表明服务器错误的一般错误响应。

- 要拒绝到特定请求 URL 的请求:
  1. 如果您有 URL 的 URIMAP 定义，那么使用第 93 页的『管理 CICS Web Support 资源』中描述的某种方法来禁用 URIMAP 定义。 检查请求 URL 是否与路径中有通配符的不特定 URIMAP 定义不匹配。 CICS 通过 Web 出错程序将 HTTP 503 响应（服务不可用）返回给 Web 客户机。您可以通过更改 Web 出错程序来定制该响应。
  2. 如果您没有 URL 的 URIMAP 定义，那么可拒绝请求，方法是更改与发出请求的端口的 TCPIPSERVICE 定义关联的分析器程序。 您可能希望对分析器程序进行编码以为每个 URL 提供各自的拒绝消息，或可能更喜欢提供涵盖任何不可用的 URL 的一条消息。 第 111 页的第 10 章，『分析器程序』介绍了哪些操作适用于处理所拒绝的请求。
- 要拒绝到虚拟主机的请求（即，到某个主机名的所有请求），如第 95 页的『管理虚拟主管』中描述的，使用 SET HOST 命令禁用虚拟主机。 CICS 通过 Web 出错程序将 HTTP 503 响应（服务不可用）返回给 Web 客户机。您可通过更改该 Web 出错程序来定制该响应。
- 要拒绝特定端口上的所有请求，使用第 93 页的『管理 CICS Web Support 资源』中描述的某种方法来禁用 TCPIPSERVICE 定义。 您可以选择停止侦听端口的方式: 正常 – 允许活动的任务完成; 立即 – 异常终止活动的任务。
- 要拒绝所有入站请求和出站请求并完全停止 CICS Web Support，如第 93 页的『管理 CICS Web Support 资源』中描述的，使用 CEMT 事务或 CPSM 关闭 TCP/IP。 您可以选择正常关闭 – 允许活动任务完成，或立即关闭 – 异常终止活动任务。

提供 favicon

许多 Web 浏览器在用户访问或收藏 Web 页面时，会自动请求 favicon（favorites icon，网站图标）。您可以通过使用 URIMAP 定义，提供一个 favicon 作为静态响应。

开始之前

关于此任务

Web 浏览器使用 URL

```
http://www.example.com/favicon.ico
```

来请求缺省的 favicon，其中 www.example.com 是站点的主机名。根据情况，也可能使用 HTTPS。您可以选择:

- CICS 区域使用的所有主机名都返回一个缺省的 favicon。
- CICS 区域使用的每个主机名的不同缺省 favicon。

如果 Web 浏览器请求 favicon，而您没有提供，那么 CICS 会将错误响应发送至浏览器，如下所示:

- 如果正在使用 CICS 提供的缺省分析器 DFHWBAAX，那么返回 404（未找到）响应。在这种情况下，不会发出任何 CICS 消息。
- 如果正在使用样本分析器 DFHWBADX 或一个类似的分析器（只能识别 CICS TS V3 之前版本所需的 URL 格式），那么该分析器可能会将路径 favicon.ico 误解为一个错误指定的转换器程序名。在这种情况下，会发出消息 DFHWB0723，并且将 400® （错误请求）响应返回到浏览器。要避免这种情况，您可以修改分析器程序，以识别 favicon 请求并提供更恰当的出错响应；您也可以使用 URIMAP 定义来提供 favicon（这意味着使样本分析器程序忽视这些请求）。

要使用 URIMAP 定义为全部或部分主机名提供 favicon:

1. 创建 favicon，并将其存储在 z/OS UNIX 文件系统上的适当位置中。
   a. 您可以使用图标编辑器创建 favicon，或使用图标转换器程序来转换用其他格式创建的图像。
   b. favicon 必须是 16 x 16 像素。 浏览器可能会忽略不符合该大小的 favicon。
   c. 必须将 favicon 保存为 Windows® 图标格式（.ico 文件扩展名），并命名为 favicon.ico。

   多数 Web 服务器需要将 favicon 存储在主机名的根目录下。对于 CICS，URIMAP 定义可以提供存储在 z/OS UNIX 上任意位置的 favicon。 CICS 区域必须具有 z/OS UNIX 的访问许可权，并且必须具有包含文件的 z/OS UNIX 目录及文件的访问许可权。*Java Applications in CICS* 说明了如何授予这些许可权。
2. 创建 URIMAP 定义以将 favicon 作为静态响应提供。 第 87 页的『为作为 HTTP 服务器的 CICS 的任何请求启动 URIMAP 资源定义』和第 90 页的『为作为 HTTP 服务器的 CICS 的 HTTP 请求的静态响应完成 URIMAP 定义』将指导您完成 URIMAP 定义的创建。 可以指定下列样本 URIMAP 定义属性，以便为 CICS 区域使用的所有主机名提供 favicon:

```
Urimap       ==> favicon        - URIMAP name
Group        ==> MYGROUP        - Any suitable
Description  ==> Favicon
STatus       ==> Enabled
USAge        ==> Server         - For CICS as HTTP server
```

```
UNIVERSAL RESOURCE IDENTIFIER
 SCheme       ==> HTTP           - Will also match HTTPS requests
 HOST         ==> *              - * matches any host name.
                                        Specify host name if you
                                        provide different favicons
 PAth         ==> /favicon.ico   - Browsers use this path to
                                        request favicons
ASSOCIATED CICS RESOURCES
 TCpipservice ==>                - Blank matches any port
STATIC DOCUMENT PROPERTIES
 Mediatype    ==> image/x-icon   - This media type is suitable
 HFsfile      ==> /u/cts/CICSHome/favicon.ico
                                        - Location of favicon in HFS
```

注: favicon 不需要代码页转换，因此无需指定 CHARACTERSET 或 HOSTCODEPAGE 选项。

提供 robots.txt 文件

Web 搜索机器人是对服务器自动发出请求的程序。例如，搜索引擎使用搜索机器人（有时也称为 Web 搜寻器）来检索要包含在其搜索数据库中的页面。您可以提供 robots.txt 文件来指示不允许搜索机器人访问的 URL。

关于此任务

在访问 Web 站点时，robot 应该会使用以下 URL 请求文档 robots.txt:

`http://www.example.com/robots.txt`

来请求缺省的 favicon，其中 `www.example.com` 是站点的主机名。如果可以使用多个端口号访问主机名，那么搜索机器人应该会请求每个主机名和端口号组合的 robots.txt 文件。文件中列出的策略可应用于所有搜索机器人，或针对特定的搜索机器人。使用禁止语句来指定搜索机器人不应访问的 URL。请注意，即使在您提供了 robots.txt 文件后，那些不符合排除标准的搜索机器人仍可以访问您的 Web 页面并建立索引。

如果 Web 浏览器请求 robots.txt 文件，而您没有提供，那么 CICS 会将错误响应发送到浏览器，如下所示:

- 如果正在使用 CICS 提供的缺省分析器 DFHWBAAX，那么返回 404（未找到）响应。在这种情况下，不会发出任何 CICS 消息。
- 如果正在使用样本分析器 DFHWBADX 或一个类似的分析器（该分析器只能识别 CICS TS V3 之前版本必需的 URL 格式），那么该分析器可能会将路径 `robots.txt` 误认为是一个错误指定的转换器程序名。在这种情况下，会发出消息 DFHWB0723，并且将 400（错误请求）响应返回到浏览器。要避免这种情况，您可以修改分析器程序以识别 robots.txt 请求并提供合适的错误响应，或者使用 URIMAP 定义来提供 robots.txt 文件（这意味着样本分析器程序将忽略这些请求）。

要为全部或某些主机名提供 robots.txt 文件:

1. 创建 robots.txt 文件的文本内容。 有关创建 robots.txt 文件的信息以及详细示例可从多个 Web 站点中获得。搜索『robots.txt』或『robots exclusion standard』，然后选择一个合适的站点。
2. 决定如何存储和提供 robots.txt 文件。您可以通过仅使用 URIMAP 定义或使用应用程序来提供该文件。

a. 您可以将 robots.txt 文件存储在 z/OS UNIX System Services 上，并使用 URIMAP 定义将该文件作为静态响应提供。 多数 Web 服务器将 robots.txt 文件存储在主机名的根目录中。对于 CICS，URIMAP 定义可以提供存储在 z/OS UNIX 的任意位置的文件，而且同一个文件可用于多个主机名。

   如果使用存储在 z/OS UNIX 中的文件，那么 CICS 区域必须具有对 z/OS UNIX 的访问权，并且还必须拥有包含文件的 z/OS UNIX 目录以及文件的访问许可权。 *Java Applications in CICS* 说明了如何授予这些许可权。

b. 您可以使用 robots.txt 文件制作 CICS文档，并使用 URIMAP 定义将其作为静态响应或作为应用程序生成的响应提供。*CICS Application Programming Guide* 说明了如何创建 CICS 文档模板。 文档模板是使用 DOCTEMPLATE 资源定义定义的，它可以保存在分区数据集、CICS 程序、文件、临时存储器队列、瞬时数据队列、出口程序或 z/OS UNIX System Services 文件中。

c. 如果要使用应用程序提供 robots.txt 文件的内容，请创建合适的支持 Web 的应用程序。 第 67 页的第 6 章，『为作为 HTTP 服务器的 CICS 编写支持 Web 的应用程序』向您介绍如何编写使用 EXEC CICS WEB API 命令的应用程序。 例如，您可以使用带 FROM 选项的 EXEC CICS WEB SEND 命令来指定包含 robots.txt 信息的数据缓冲区。另外，您可以使用应用程序来传递模板中的 CICS 文档。指定 `text/plain` 介质类型。

   您可能想要使用应用程序来处理搜索机器人发出的请求，以便跟踪正在访问 Web 页面的搜索机器人。搜索机器人请求中的 User-Agent 头应给出搜索机器人的名称，而 From 头应该包含搜索机器人所有者的联系信息。应用程序可以读取并记录这些 HTTP 头。

3. 开始 URIMAP 定义，该定义与 Web 搜索机器人对 robots.txt 文件发出的请求相匹配。 第 87 页的『为作为 HTTP 服务器的 CICS 的任何请求启动 URIMAP 资源定义』列出了创建与请求相匹配的 URIMAP 资源定义的步骤。 可以指定以下样本 URIMAP 定义属性以匹配任意主机名的 robots.txt 文件的请求:

```
 Urimap       ==> robots         - URIMAP name
 Group        ==> MYGROUP        - Any suitable
 Description  ==> Robots.txt
 STatus       ==> Enabled
 USAge        ==> Server         - For CICS as HTTP server
UNIVERSAL RESOURCE IDENTIFIER
 SCheme       ==> HTTP           - Will also match HTTPS requests
 HOST         ==> *              - * matches any host name.
                                        Specify host name if you
                                        provide separate robots.txt files
 PAth         ==> /robots.txt    - Robots use this path to
                                        request robots.txt
ASSOCIATED CICS RESOURCES
 TCpipservice ==>                - Blank matches any port. Specify
                                        TCPIPSERVICE definition name if
                                        you provide different robots.txt
                                        files depending on the port
```

   请记住，URL 的路径部分是区分大小写的。必须以小写字母指定路径 `/robots.txt`。

4. 如果提供 robots.txt 文件作为静态响应，请完成 URIMAP 定义来指定文件位置和 CICS Web Support 构造响应所需的其他信息。 第 90 页的『为作为 HTTP 服务器的 CICS 的 HTTP 请求的静态响应完成 URIMAP 定义』指导您完成这一过程。 例如，可以指定以下 URIMAP 定义属性以提供 robots.txt 文件，该文件是使用 EBCDIC 代码页 037 创建的，并存储在 `/u/cts/CICSHome` 目录中:

```
STATIC DOCUMENT PROPERTIES
 Mediatype    ==> text/plain
 CHaracterset ==> iso-8859-1
 HOSTCodepage ==> 037
 HFsfile      ==> /u/cts/CICSHome/robots.txt
```

HFsfile 名称区分大小写。

5. 如果您使用应用程序提供 robots.txt 文件的内容，请完成 URIMAP 定义以指定将由程序处理请求。 第 89 页的『为作为 HTTP 服务器的 CICS 的 HTTP 请求的应用程序响应完成 URIMAP 定义』指导您完成这一过程。 例如，以下 URIMAP 定义属性可用来让支持 Web 的应用程序 ROBOTS 来处理请求，且不会调用任何分析器或转换器程序:

```
ASSOCIATED CICS RESOURCES
 Analyzer     ==>No             - Analyzer not used for request
 COnverter    ==>              - Blank means no converter program
 TRansaction  ==>              - Blank defaults to CWBA
 PRogram      ==> ROBOTS
```

Warning 头

如果 Warning 头出现在 HTTP 消息中，那么它通常包含打算供用户阅读的信息。如果 CICS Web Support 接收到带 Warning 头的消息，那么将与该标题关联的文本写到 CWBW 瞬时数据队列。

对于作为 HTTP 服务器的 CICS，用于记录请求中 Warning 头的消息号是 DFHWB0750，而对于作为 HTTP 客户机的 CICS，用于记录响应的 Warning 头的消息号是 DFHWB0752。每个 Warning 头的消息包含:

- 与 Warning 头关联的文本。
- 服务器和客户机的 IP 地址。

消息被写到 CICS 提供的瞬时数据队列 CWBW，它对于 CSSL 是间接的。在组 DFHDCTG 中，有一个队列的样本定义。

CWBO 是通常用于 CICS Web Support 的队列，而 CWBW 是用于单独保存警告消息的队列。如果您接收太多的 Warning 头或不再有用的 Warning 头（如服务器总是发送一个警告以响应您重复发出的客户机请求），那么可以除去 CWBW 瞬时数据队列以禁止这些记录。

第 9 章 Web 出错程序

当 CICS Web Support 过程中发生请求错误或异常终止时，用户可替换的 Web 出错程序将出错响应提供给 Web 客户机。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

在以下情况下使用 Web 出错程序:

- 当 CICS Web Support 在初始处理来自 Web 客户机的请求时检测到问题；例如，如果请求中缺少必需的信息，或者如果信息发送太慢，而且达到接收超时。
- 已安装的 URIMAP 定义与请求匹配，但 URIMAP 定义或虚拟主机被禁用，或者无法访问静态响应的资源。
- URIMAP 匹配失败，为 TCPIPSERVICE 定义指定的分析器不能处理请求，并将控制权传递给 Web 出错程序。
- URIMAP 定义和分析器程序以及转换器程序都无法确定应执行什么应用程序来服务请求。
- 分析器程序、转换器程序或用户编写的应用程序中发生异常终止。这会确保即使处理已失败也会将响应返回给 Web 客户机。

在以下情形下**不**使用 Web 出错程序:

- 发生套接字发送错误或接收错误。在这种情况下，关闭套接字并不向 Web 客户机发送响应。
- 当 URIMAP 指定重定向响应时。这些响应由 CICS 组成，并且不能定制。
- 用户编写的应用程序已成功完成处理并要返回表明错误的响应；例如，如果客户机指定了资源不支持的方法。这些响应由应用程序编写和发送。
- 对于涉及作为 HTTP 客户机的 CICS 的处理。向服务器发送出错响应，Web 客户机不是必需的。从服务器接收的响应由客户机应用程序处理。

如果存在与客户机的持续连接，那么通过 Web 出错程序发送错误响应后，CICS 保持连接打开。特例是 CICS 为响应选择 501（方法未实施）状态码，在这种情况下，CICS 关闭该连接。

CICS 中提供了两个用户可替换 Web 出错程序:

DFHWBERX（Web 出错应用程序）

DFHWBERX 可以由分析器程序或在 URIMAP 定义中指定为应用程序以处理请求。当 CICS 提供的缺省分析器 DFHWBAAX 作为 TCPIPSERVICE 定义中的分析器程序指定，且没有找到请求的匹配 URIMAP 定义时，使用它。DFHWBERX 使用 EXEC CICS WEB 和 DOCUMENT API 命令来获取有关 Web 客户机请求的信息，并创建和发送错误响应。

DFHWBEP（Web 出错程序）

当 CICS 在请求处理中检测到错误时，DFHWBEP 在需要 Web 出错程序的所有其他情形中使用。CICS 向 DFHWBEP 提供说明错误状况的参数列表，并提供位于存储器块中的缺省出错响应。DFHWBEP 可以使用 EXEC CICS WEB 和

DOCUMENTAPI 命令来创建自己的出错响应，并将它发送至 Web 客户机；它也可以修正或接受 CICS 提供的缺省出错响应。

要了解关于编写用户可替换程序的更多信息，请参阅 *CICS Customization Guide* 中的利用用户可替换程序进行定制。

DFHWBERX，Web 出错应用程序

DFHWBERX 使用 EXEC CICS WEB 和 DOCUMENT API 命令来获取有关 Web 客户机请求的信息，并创建和发送错误响应。它称为应用程序。如果请求总是需要错误响应，那么 DFHWBERX 可以由分析器程序指定，或作为 URIMAP 定义中的 PROGRAM 属性指定。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

当 CICS 提供的缺省分析器 DFHWBAAX 作为 TCPIPSERVICE 定义中的分析器程序指定，且没有找到请求匹配的 URIMAP 定义时，使用 DFHWBERX。使用 `wbra_server_program` 输出参数，DFHWBAAX 将 DFHWBERX 设置为应用程序以处理请求。

DFHWBERX 是用户可替换的。CICS 仅为汇编程序中的 DFHWBERX 提供源代码。

DFHWBERX 不会从 CICS 接收参数列表或缺省的 HTTP 响应。它使用 EXEC CICS 命令来获取有关 Web 客户机请求的信息，并创建和发送错误响应。

DFHWBERX 提供如下错误响应:

- 如果 Web 客户机的请求是介质类型为 `text/xml` 的 POST 请求，那么假定它为 SOAP 1.1 请求，且返回 SOAP 1.1 故障响应。
- 如果请求是介质类型为 `application/soap+xml` 的 POST 请求，那么假定它是 SOAP 1.2 请求，且返回 SOAP 1.2 故障响应。
- 所有其他请求假定为标准 HTTP 请求，因此用 404（未找到）状态码组成并返回合适的 HTTP 响应。

在 DFHWBERX 中:

- EXEC CICS WEB EXTRACT 命令用于获取需要错误响应的 Web 客户机请求的 URL。
- EXEC CICS DOCUMENT 命令用于构造消息体。
- 对于 SOAP 故障响应，EXEC CICS WEB WRITE HTTPHEADER 命令用于编写合适的 SOAP 操作头。
- EXEC CICS WEB SEND 命令用于指定合适的状态码，并将响应发送给 Web 客户机。为响应主体的代码页转换指定了 UTF-8 字符集。

DFHWBERX 不会使用 EXEC CICS WEB RECEIVE 命令来接收 Web 客户机请求的内容。如果用您自己的应用程序替换 DFHWBERX，请注意不应在 Web 出错程序中使用该命令。如果正在使用 CICS 提供的缺省分析器 DFHWBAAX，那么 DFHWBERX 用于将错误响应发送到任何与 URIMAP 定义不匹配的请求。无法了解这些请求的内容，而且其目的也可能是恶意的，因此建议不要尝试接收该请求。

DFHWBEP，Web 出错程序

DFHWBEP 从 CICS 接收说明错误状况的参数列表，并接收一个存储器块，其中包含 CICS 将发送至 Web 客户机的缺省 HTTP 响应（包括状态码和状态文本）。该程序可以使用或修改缺省响应，或使用 EXEC CICS WEB 和 DOCUMENT API 命令创建并发送自己的响应。DFHWBEP 是用户可替换的。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

评估错误状况

由 CICS 传递到 DFHWBEP 的参数列表包含 CICS 已在缺省响应中使用的三位数 HTTP 状态码。参数列表还提供说明错误状况的信息，如错误代码、异常终止代码、CICS 消息号、响应码和原因码，以及发生了错误的程序的名称。

如果定制 DFHWBEP，那么确保您正在使用合适范围的输入参数来识别应用定制响应的情形，而不只依赖状态码。CICS Web Support 可以将每个状态码用于各种用途。任何具有您的程序不了解的状态码的 HTTP 响应都应未经更改地传递。

除了检查 CICS 提供的参数列表，您可能还需要使用 EXEC CICS WEB EXTRACT 命令和 EXEC CICS EXTRACT TCPIP 命令，来检查请求行，并详细了解需要出错响应的 Web 客户机请求。也可以使用 WEB READ HTTPHEADER 命令或 HTTPHEADER 浏览命令来读取请求的 HTTP 头。当然，您应当意识到当请求处于无效状态或已超时时，可能无法使用这些命令。

然而，请注意，EXEC CICS WEB RECEIVE 命令（接收 Web 客户机请求的内容）**不**应该在 Web 出错程序中使用。DFHWBEP 处理的 Web 客户机请求出错状况包括：超时、缺少必需的信息、可能带有异常的有害内容、已由另一个应用程序接收。如果接收处于以上状态之一的请求，将导致问题或不可预测的结果，因此建议不要在您的 Web 出错程序中接收 Web 客户机请求。

创建并发送出错响应

CICS 提供的参数列表包含指向存储器块（包含对于检测到的错误的缺省 HTTP 响应）的指针，并且还包含指示响应长度的参数。该存储器块包含完整的 HTTP 响应：状态行、HTTP 头和消息主体，其中的长度是完整的响应消息的长度。

Web 出错程序会选择下列操作之一：

1. 不更改缺省响应，允许 CICS 将其发送给 Web 客户机。当程序无法识别 HTTP 响应的状态码时，或者当您已评估了该状况并认为缺省响应较为适合时，请采取该操作。
2. 使用 EXEC CICS WEB 和 DOCUMENT API 命令创建一个新的响应，并将其发送至 Web 客户机。如果要更改该响应，建议采取该操作，因为这将使 CICS 能为检测消息提供更多帮助。使用该方式创建的响应会替换缺省的响应，并将其丢弃。WEB WRITE HTTPHEADER 命令可用于为响应编写 HTTP 头，而 WEB SEND 命令可用于组织并发送响应。必须在您的命令中指定 ACTION(IMMEDIATE)，这是因为 DFHWBEP 不允许采用 ACTION(EVENTUAL) 的缺省值。第 74 页的『为响应写 HTTP 头』、第 76 页的『为 HTTP 消息产生实体主体』和第 76 页的『从作为 HTTP 服务器的 CICS 发送 HTTP 响应』说明了如何使用 API 命令来创建和发送响应。

3. 手动修改存储器块中的缺省响应，对长度参数进行相应更新，并允许 CICS 将其发送给 Web 客户机。如果只希望对缺省响应作细微更改（例如，用一段简要文本替换缺省消息体），那么可以考虑采取该操作，但是必须注意确保使 HTTP 响应仍然有效，并指定正确的长度。
4. 在新的存储器块中手动构建新的 HTTP 响应，回传新存储器块的地址以及新响应的长度，并允许 CICS 将其发送给 Web 客户机。新响应会替换缺省响应，并将其丢弃。由于 CICS 无法在检查以该方式构建的消息时提供全面帮助，所以不再推荐使用该操作。如果具有在 CICS Transaction Server for z/OS V3R2 之前定制的并采取了该操作的 DFHWBEP 版本，那么您应该考虑使用 HTTP 响应来替换该操作，该 HTTP 响应是使用 EXEC CICS WEB 和 DOCUMENT API 命令来构建和发送的。

更正出错响应的内容

无论您使用 EXEC CICS WEB 和 DOCUMENT API 命令来创建新响应，还是在存储器块中手动修改缺省响应，或在新的存储器块中手动构建新的 HTTP 响应，都可能需要修改出错响应中的所有项。但是，您必须确保使 HTTP 响应仍然有效，并确保在手动处理存储器块中的响应时，指定正确的长度。

响应必须包含 HTTP 版本、状态码、状态文本、所需的任何 HTTP 头和消息体。响应的格式应该符合您正在遵从的 HTTP 协议规范（HTTP/1.0 或 HTTP/1.1）。如果使用 API 命令，那么 CICS 会为所有这些元素提供帮助。

请注意以下针对出错响应中各个项的指南:

HTTP 版本（HTTP/1.1 或 HTTP/1.0）

在缺省响应中，这由 CICS 根据 Web 客户机的 HTTP 版本来确定。如果您正在处理存储器块中的缺省响应，那么不要修改这个响应元素。如果您正在使用 API 命令创建新的响应，请使用 WEB EXTRACT 命令来识别 Web 客户机的 HTTP 版本，并相应地调整您的响应。Web 客户机使用的 HTTP 版本会影响您为响应选择的 HTTP 头、状态码和消息内容。HTTP/1.0 客户机可能不了解 HTTP/1.1 规范中描述的更多高级功能。

数字状态码（例如，404 或 500）

CICS 为缺省响应选择了一个状态码。修改缺省响应的该元素或为新响应选择状态码时，需要特别小心。第 341 页的附录 C，『CICS Web Support 的 HTTP 状态码参考』列出了 CICS Web Support 使用的状态码以及使用它们的原因。HTTP/1.1 规范具有有关所有状态码和正确使用它们的需求的更多信息。如果您确定有另一种状态码比 CICS Web Support 选择的状态码更适合，那么确保您的用法遵循 HTTP/1.1 规范中的需求。特别要检查状态码是否适合 Web 客户机的 HTTP 版本。对于非 HTTP 错误，CICS 总是使用 400 状态码。

原因短语或状态文本（例如，未找到）

您可以修改这个缺省响应的元素，或为新响应提供自己的原因短语。推荐使用 HTTP/1.1 规范建议的原因短语（例如，“未找到”或“错误请求”），但这些短语是可选的。HTTP/1.1 规范声明：每个状态码的原因短语可以由本地同等短语替换。

HTTP 头

缺省响应包含 CICS 为响应编写的头（如 Date 和 Server 头）。如果您正在使用 API 命令创建新响应，那么当您发送该响应时，CICS 会自动添加这些头。第 335 页的附录 B，『CICS Web Support 的 HTTP 头参考』列出 CICS 可以编写的头。CICS 编写的头适合于消息的 HTTP 版本，如果您正在处理缺省响应，那么不应将其除

去，因为遵循 HTTP 规范可能需要这些头。如果合适，可以为响应添加更多 HTTP 头。检查 HTTP/1.1 规范是否允许在该上下文中使用头。如果已选择不同的状态码，那么 HTTP/1.1 规范可能需要某些头。

消息体

缺省响应的消息体会重复发行版中给出的状态码和原因短语。您可以修改这个缺省响应的元素，或为新响应提供自己的消息体。对于许多状态码，消息体可用于向用户提供更多信息。某些状态码不能与消息体一起出现。

代码页转换

存储器块中的缺省 HTTP 响应会以 EBCDIC 代码页 037 传递到 DFHWBEP。

当您在 Web 出错程序中使用 EXEC CICS WEB API 命令生成一个新的出错响应，并将其发送到 Web 客户机时，将按您在命令中所指定的那样执行代码页转换，转换方式与其他任何使用 EXEC CICS WEB API 命令的程序所使用的方式完全相同。

DFHWBEP 无法为在存储器块中生成的响应指定代码页转换设置。如果您修改了存储器块中的缺省出错响应，或在新的存储器块中提供了一个新的出错响应，并将它返回到 CICS 以便发送，那么 CICS 会假定存储器块中提供的这个响应也采用 EBCDIC 代码页 037。CICS 在将响应发送到客户机之前，会对响应执行代码页转换，以将其转换成合适的 ASCII 字符集。如果分析器程序参与路径的处理，并且为代码页转换设置参数（作为独立服务器和客户机代码页参数，或作为 DFHCNV 键），那么 CICS 将这些选项用于代码页转换。如果未使用分析器程序，或者在发生错误之前未调用分析器，那么响应使用 ISO-8859-1 字符集。 如果该输出不合适，请使用 EXEC CICS WEB API 命令来生成响应。

Web 出错程序 DFHWBEP 的输入参数

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

CICS 将多个参数传递给 DFHWBEP，包括错误代码、异常终止代码和错误消息。

要获取 CICS 传递给 DFHWBEP 的列表及其内所有参数的技术描述，请参阅第 379 页的附录 H, 『Web 出错程序 DFHWBEP 的参考信息』。

以下是 DFHWBEP 的输入参数:

- 标识原始错误原因的错误代码（wbep_error_code）。DFHWBUCO 副本列出了这些代码。
- 发生错误时进行的处理类型（服务器或管道）。
- 发生错误的程序名称。
- 与错误关联的 CICS 异常终止代码（wbep_abend_code）。
- 与错误和指向消息文本的指针关联的 CICS 消息号（wbep_message_number）。
- 分析器程序或转换器程序（如果使用）返回的响应码和原因码。
- 接收到请求的端口的 TCPIPSERVICE 资源定义名。（如果使用了分析器程序，那么该名称标识该分析器程序的名称。）
- 所使用的转换器程序的名称。
- 目标程序（即，旨在为请求提供响应的应用程序）的名称。

- 缺省 HTTP 响应中的 3 位 HTTP 状态码（wbep_http_response_code）。要获取 CICS Web Support 使用的状态码列表以及使用它们的原因，请参阅第 341 页的附录 C, 『CICS Web Support 的 HTTP 状态码参考』。
- 指向包含缺省 HTTP 响应的存储器块的指针。该响应是一个完整的 HTTP 响应，包括状态行、HTTP 头和消息体。
- 缺省 HTTP 响应的长度。响应缓冲区的最大长度是 32KB。
- 以点分十进制或冒号十六进制表示法表示的该请求的服务器（作为 HTTP 服务器的 CICS）IP 地址。
- 以点分十进制或冒号十六进制表示法表示的 Web 客户机的 IP 地址。

有关 IP 地址表示法的更多信息，请参阅 第 6 页的『IP 地址格式』。

Web 出错程序 DFHWBEP 的输出参数

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

DFHWBEP 的输出参数是 wbep_response_ptr 和 wbep_response_len。

第 379 页的附录 H, 『Web 出错程序 DFHWBEP 的参考信息』完整列举了 CICS 传递到 DFHWBEP 的参数列表中的所有参数，并给出了技术描述。

DFHWBEP 的输出参数是:

- 包含 HTTP 响应的存储器块的地址（wbep_response_ptr）。
- 存储器块中 HTTP 响应的长度（wbep_response_len）。响应的最长长度为 32K。指定的长度是整个缓冲区的长度；CICS 计算消息体的长度并提供合适的 Content-Length 头。

缺省情况下，与存储器块相关的这些输出参数包含由 CICS 生成的缺省 HTTP 响应。

- 如果您在 DFHWBEP 中已使用 EXEC CICS WEB 和 DOCUMENT API 命令来创建新的响应，并将其发送至 Web 客户机，CICS 会忽略并丢弃存储器块中的 HTTP 响应，因此可以不更改这些输出参数。
- 如果您已经修改了存储器块中的缺省响应，那么需要更新 wbep_response_len 中的长度，给出整个缓冲区的新长度。您不必计算消息体长度，或更改响应中 Content-Length 头。CICS 检查您已提供的消息体的长度，并相应地更正 Content-Length 头。
- 如果您在新的存储器块中手动创建了一个新的 HTTP 响应，那么需要在 wbep_response_ptr 中回传新存储器块的地址，并在 wbep_response_len 回传新响应的长度。

CICS Web Support 缺省状态码和错误响应

由分析器或转换器程序设置的响应代码和原因码映射到缺省状态码和关联响应。如果在使用 URIMAP 定义产生静态响应时发生错误，CICS 也选择缺省状态码和相关响应。状态码和响应可由用户可替换的 Web 出错程序 DFHWBEP 进行修改。

关于此任务

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

HTTP 协议规范定义了一些状态码，当请求无法成功完成时，服务器可以在 HTTP 响应中返回这些状态码。第 341 页的附录 C, 『CICS Web Support 的 HTTP 状态码参考』提供了有关这些状态代码的信息。

要获取有关 HTTP 响应结构的更多信息，请参阅 第 13 页的『HTTP 响应』。

当 CICS Web Support 处理期间出错时，会将信息传递到参数列表中的 Web 错误程序 DFHWBEP，以帮助确定相应的错误响应:

- 如果分析器或转换器程序处理期间出错，那么可以使用来自参数列表中的程序的响应和原因码来识别错误。
- 如果在使用 URIMAP 定义生成静态响应时出错，那么可以使用参数列表中关联的 CICS 消息号和文本来识别错误。

对于所有类型的错误，会将完整的缺省错误响应（包括状态码）传递到 Web 出错程序，以供接受、修改或替换。伴随着错误响应的是一条 CICS 消息和异常跟踪项。

分析器程序使用的响应代码的缺省状态码如下所示:

表 3. 分析器程序处理错误的缺省状态码

wbra_response	**缺省状态码**
除 URP_OK 以外的任何值	400 错误请求

转换器程序的缺省状态码如下所示:

表 4. 转换器的解码函数的缺省状态码

decode_response	**decode_reason**	**缺省状态码**
URP_EXCEPTION	URP_CORRUPT_CLIENT_DATA	400 错误请求
URP_EXCEPTION	URP_SECURITY_FAILURE	403 禁止
URP_EXCEPTION	任何其他值	501 未实施
URP_INVALID	任何值	501 未实施
URP_DISASTER	任何值	501 未实施
任何其他值	任何值	500 内部服务器错误

表 5. 转换器的编码函数的缺省状态码

encode_response	**编码原因**	**缺省状态码**
任何值，除了 URP_OK URP_OK_LOOP	任何	501 未实施

使用 URIMAP 定义产生静态响应时所发生错误的缺省状态码如下所示:

表 6. 静态响应处理错误（使用 URIMAP 定义）的缺省状态码

CICS 消息号	**错误**	**缺省状态码**
0758	用户没有对生成静态响应所需资源的 READ 访问权（CICS 文档模板或 z/OS UNIX 文件）。	403 禁止

表 6. 静态响应处理错误（使用 URIMAP 定义）的缺省状态码 (续)

CICS 消息号	错误	缺省状态码
0759	无法找到生成静态响应所需的资源（CICS 文档模板或 z/OS UNIX 文件）。	404 未找到
0760	无法读取生成静态响应所需的 z/OS UNIX 文件。	500 内部服务器错误
0761	任何其他错误。	500 内部服务器错误

第 10 章 分析器程序

分析器程序与 TCPIPSERVICE 定义关联。它们的主要作用是在以下情况下解释 HTTP 请求：如果 URIMAP 定义指定使用分析器程序，或者如果不存在 URIMAP 定义。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

当 CICS 作为 HTTP 客户机，或者用于 Web 服务处理时，不能调用分析器程序；仅当 CICS 作为 HTTP 服务器时才能调用它们。第 25 页的『作为 HTTP 服务器的 CICS 的 HTTP 请求和响应处理』中描述了在 CICS 作为 HTTP 服务器时 CICS Web Support 进程中分析器程序的角色。第 55 页的第 5 章，『规划作为 HTTP 服务器的 CICS 的 CICS Web Support 体系结构』提供信息，帮助您规划将 CICS 作为 HTTP 服务器的体系结构。

分析器程序和 URIMAP 定义之间的关系

CICS Transaction Server for z/OS V3R1 前，作为 HTTP 服务器的 CICS 的所有 HTTP 请求由分析器程序解释。在 CICS TS V3 中，URIMAP 定义是用于控制 HTTP 请求处理的关键性设施。它们取代了分析器程序的主要功能，将请求的 URL 与处理请求的应用程序匹配，以及指定使用转换器程序和别名事务。

然而，URIMAP 定义可以为选择的 HTTP 请求调用分析器程序，以接管某些处理阶段并执行其他操作（例如，监控或审计操作）。命名为 CONVERTER（转换器程序名）、TRANSACTION（别名事务）、USERID（别名事务的用户标识）和 PROGRAM（处理请求的应用程序名）的 URIMAP 定义的属性（这些属性重新产生分析器函数）可以传递到分析器程序中，且分析器程序可以选择覆盖它们。

按照 CICS Transaction Server for z/OS V3R1 前 CICS Web Support 使用的同一过程操作，您可以选择不使用 URIMAP 定义就直接将 HTTP 请求传递到分析器程序。然而，没有 URIMAP 定义的情况下，如果要更改 CICS 对特殊 HTTP 请求响应的方式，那么需要更改分析器程序中的逻辑。使用 URIMAP 定义，您可以将这些更改作为系统管理任务动态执行。也请注意，如果继续使用分析器程序而不是 URIMAP 定义来处理请求，且如果需要在这方面遵从 HTTP/1.1，那么根据 HTTP/1.1 规范（RFC 2616）中声明的规则，必须编码分析器程序以执行 URL 比较。

注: 已发现请求的匹配 URIMAP 定义后，所提供的 CICS 提供的样本分析器程序 DFHWBADX 和 CICS 提供的缺省分析器程序 DFHWBAAX 不执行该请求的任何分析，即使 URIMAP 指定 ANALYZER(YES) 时也是如此。

使用分析器程序进行错误处理

虽然现在并不是每个 HTTP 请求的处理路径中都需要分析器程序，但是仍必须为用于 CICS Web Support 的每个 TCPIPSERVICE 资源定义指定一个分析器程序。

分析器程序的名称在资源定义的 URM 属性中指定。您可以在每个 TCPIPSERVICE 定义中指定不同的分析器，或在多个 TCPIPSERVICE 定义中指定相同的分析器。如果正在从 URIMAP 定义调用分析器程序，那么无法在不同的分析器程序间进行选择；您只能选择是否使用为 TCPIPSERVICE 定义指定的分析器程序。

如果 CICS 未找到匹配的 URIMAP 定义用于此请求，那么调用为 TCPIPSERVICE 定义指定的分析器程序，以处理 HTTP 请求。这可能由用户输入请求 URL 时出错而引起，或者是因为没有安装正确的 URIMAP 定义。（如果 URIMAP 定义存在，但是已禁用，那么请求由 Web 出错程序而不是分析器程序处理。）

因为这个原因，最低限度，为每个 TCPIPSERVICE 定义指定的分析器程序应该包含一个过程，以处理它不识别的任何 HTTP 请求，并提供合适的错误响应。您还可以识别本应该由 URIMAP 定义处理的特定请求，并提供更相关的错误响应。错误情况下分析器程序的输出将传递给可用于修改 HTTP 响应的 Web 出错程序。第 103 页的第 9 章，『Web 出错程序』说明了如何定制这些内容。

当 TCPIPSERVICE 定义指定 PROTOCOL(HTTP) 时，CICS 提供的缺省分析器程序 DFHWBAAX 是缺省值。当端口上所有请求应该由 URIMAP 定义处理时，DFHWBAAX 提供基本错误处理。它不提供对使用 URL 格式的请求的支持，此格式是在 CICS TS 3.1 之前，CICS Web Support 使用的格式。如果您需要在分析器程序中为 URIMAP 定义不处理的请求提供处理，TCPIPSERVICE 定义中指定的分析器程序应该是 CICS 提供的样本分析器程序 DFHWBADX 或自己定制的分析器程序。

将分析器程序用于某些不支持 Web 的应用程序和非 HTTP 消息

从 URIMAP 定义直接调用不支持 Web 的应用程序时，这些应用程序可能正确运行。然而，某些应用程序可能取决于只能由分析器程序为它们提供的设施。在以下情况下，可能需要在 HTTP 请求的处理路径中使用分析器程序:

- 您正在使用不支持 Web 的应用程序和转换器程序产生响应，而它需要标记为 CICS TS V3 前的兼容性处理，这是因为 Web 客户机要求响应与它可能已接收的 CICS TS V3 前的响应一样。（例如，用户编写的客户机在使用新错误响应或其他 HTTP 头时可能遇到问题。）仅当转换器程序在存储器块中手工生成响应时，该标志才起作用。如果转换器程序使用 EXEC CICS WEB API 命令来发送响应，那么该标志不起作用。
- 您正在使用不支持 Web 的应用程序和转换器程序生成响应，而传递到存储器块中转换器程序的 Web 客户机请求副本或通过转换器程序在存储器块中手工生成的 HTTP 响应都需要非标准的代码页转换。转换器程序无法为传入存储器块的 HTTP 请求或响应指定代码页转换设置。第 124 页的『编写转换器程序』中描述了当处理路径中未提供分析器程序时 CICS 用于代码页转换的标准设置。如果这些标准设置不合适，或者如果不希望进行代码页转换，那么可以在处理路径中使用分析器程序来指定备用代码页转换设置。另一种使用分析器程序的方法是，在转换器程序中使用 EXEC CICS WEB API 命令来检查 Web 客户机的请求或生成响应，而不是使用存储器块。对于这种情况，可以象往常一样在 EXEC CICS WEB API 命令中指定代码页转换。

如果需要分析器程序来处理这些情况之一，可以为请求设置 URIMAP 定义，但是它必须指定分析器程序。

对于使用 TCPIPSERVICE 定义中用户定义的（USER）协议的非 HTTP 请求，始终需要分析器程序来处理这些请求，且不能使用 URIMAP 定义。第 163 页的第 14 章，『CICS Web Support 和非 HTTP 请求』说明了如何处理非 HTTP 请求。

将分析器程序用于其他处理

在可以选择是否在处理路径中使用分析器程序的情况下，您可能为了以下原因选择使用分析器程序:

- 根据请求内容，您可能要对处理路径的元素进行动态更改。HTTP 请求的每个 URL 由单个 URIMAP 定义匹配，它定义了单个处理路径。分析器程序可以解释请求内容并更改某些元素，例如，处理请求的应用程序、转换器程序的涉及，或用于请求的别名事务和用户标识。
- 您要将监控或审计操作引入进程。分析器程序是执行此操作的理想位置。
- 您正在从 CICS TS V2 升级现有 CICS Web Support 体系结构，且您的现有分析器程序提供您要在请求处理期间维护的其他功能，例如，将信息传递到转换器程序。

『编写分析器程序』说明了分析器程序可以执行的完整范围函数。

用 URIMAP 定义替换分析器程序

通常，您应当使用 URIMAP 资源定义来替换分析器程序的请求处理功能，这些 URIMAP 资源定义可通过 CICS 系统编程命令来更改和控制。

URIMAP 定义可用于匹配请求的 URL 并将它们映射到应用程序中，还可用于指定转换器程序、别名事务以及用户标识。如果您的分析器程序提供了其他功能，那么您可以继续使用它（而不用替换为 URIMAP 定义），也可以将它与 URIMAP 定义结合使用。

迁移为使用 URIMAP 后:

- 可以每次只针对少量请求，逐步引入 URIMAP 资源定义。根据分析器程序采用的处理类型以及处理请求的应用程序类型不同，您可以选择是否在每个请求的处理路径中继续使用分析器程序。
- 对于 URIMAP 资源定义处理的请求，您可能更偏向于选择和发布新的 URL，而不是继续使用现有的 URL。当您准备停用请求的旧处理路径时，您可以设置 URIMAP 定义，以便永久性地将请求从旧的 URL 重定向到新的 URL 上。
- 确保您的分析器程序仍然包含针对未识别请求的基本处理过程，即使所有请求的处理路径中已不再涉及该处理过程。TCPIPSERVICE 定义中仍然需要分析器程序，因为可以在诸如最终用户误输入了 URL 的情况下使用分析器程序来接收请求。

编写分析器程序

您可以用汇编程序、C、COBOL 或 PL/I 编写分析器程序。

关于此任务

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

分析器程序的输入和输出参数在通信区域中传递。第 353 页的附录 E, 『分析器程序的参考信息』中描述了语言相关的头文件、包含文件和映射通信区域的副本。

分析器程序可以执行的完整范围的功能如下:

- 确定是否应该为请求继续处理，或者 CICS 是否应该将错误响应返回到 Web 客户机。
- 分析请求内容和已从 URIMAP 定义传递到转换器程序的任何参数，以确定需要哪些后续处理阶段，以及需要哪些 CICS 资源以执行每个阶段。（该分析期间可以使用 EXEC CICS WEB API 命令。）
- 在请求传递到应用程序之前，指定要处理请求的转换器程序名。转换器程序通常与不支持 Web 的应用程序协同使用。需要的话，提供用户令牌以使分析器程序与转换

器程序通信。Web 客户机请求被传递到转换器程序，该程序位于参数列表中的指针所指向的一个 32K 存储器块中。第 123 页的第 11 章，『转换器程序』说明了转换器程序的功能。

- 指定要处理请求并提供响应的用户编写的应用程序名。
- 指定处理剩余处理阶段的别名事务的事务标识。
- 指定要与别名事务关联的用户标识。
- 指定或禁止传递到存储器块中转换器程序的请求的代码页转换，以及转换器程序在存储器中手工构建的任何响应的代码页转换。这不影响转换器程序或用户编写的应用程序，它们使用 EXEC CICS WEB API 命令来查看 HTTP 请求并生成响应；且它们直接从 CICS 请求代码页转换。第 37 页的『CICS Web Support 的代码页转换』解释了代码页转换的过程。
- 指定用于升级的标志，该标志表明不支持 Web 的应用程序需要在哪里进行 CICS TS V3 前版本的兼容性处理。这不影响转换器程序或用户编写的应用程序，它们使用 EXEC CICS WEB API 命令来查看 HTTP 请求并生成响应。
- 修改请求主体。所做的任何更改在传递到存储器块中转换器程序的数据中都是可视的，但是在 EXEC CICS WEB API 命令的数据中不可视。

CICS 提供缺省分析器程序 DFHWBAAX（它在 第 119 页的『CICS 提供的缺省分析器程序 DFHWBAAX』 中描述）和样本分析器程序 DFHWBADX（它在 第 119 页的『CICS 提供的样本分析器程序 DFHWBADX』 中描述）。如果这些分析器不符合您的需求，那么需要编写您自己的分析器。您可能可以将 DFHWBADX 作为示例使用。

用户可替换的程序对于操作 CICS Web Support 的系统来说都必须是本地的。如果您不使用程序的自动安装，那么必须为 CICS Web Support 使用的所有用户可替换程序（包括分析器和转换器程序）定义和安装程序定义。如果使用程序的自动安装，那么必须确保用正确的属性安装了用户可替换程序。请注意，必须使用 EXECKEY（CICS）来定义分析器程序。

要了解关于编写用户可替换程序的更多信息，请参阅 *CICS Customization Guide* 中的利用用户可替换程序进行定制。

分析器程序的输入

输入参数传递到通信区域中的分析器程序，提供有关请求特性和内容的信息，以及 URIMAP 定义提供的任何输入。分析器程序可选择接受这些值并将它们作为输出参数传递，或它可根据它对请求内容的分析来动态地重设它们。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

第 354 页的『分析器程序的参数』具有通信区域中所有参数的列表和技术描述。

输入参数包含以下各项，或指向它们的指针:

- 分析器参数列表的识别标识。
- 客户机和服务器（作为 HTTP 服务器的 CICS）的冒号十六进制或点分十进制 IP 地址。
- 请求是否为 HTTP 请求的指示符。
- 是否找到请求的匹配 URIMAP 定义的指示符。如果该指示符为正，那么 URIMAP 定义可能已将其他输入参数传递到分析器程序。

- HTTP 版本。
- 请求方法。
- 为请求指定的主机名，从 Host 头或（对于绝对 URI）从请求 URL 获取。对于 HTTP/1.1 请求，必需主机名，因此该参数总是传递到分析器。对于 HTTP/1.0 请求，可能不提供主机名。
- URL 的路径部分。
- 为请求指定的任何查询字符串。
- 请求的 HTTP 头。

  如果已使用分块的传输编码发送请求，那么尾部头不会与主请求头一起被传递到分析器程序。
- 请求主体（或请求主体的绝大部分）将存储在一个 32 KB 的存储器块中。该请求主体是指向包含请求的独立存储器块的指针。

对于在使用 SSL 客户机认证的连接上接收到的 HTTP 请求，还传递以下参数:

- 从客户机证书获取的用户标识。

如果找到请求的匹配 URIMAP 定义并且已调用分析器程序，那么在以下参数已存在于 URIMAP 定义的情况下，将它们从 URIMAP 定义传递到分析器程序:

- 请求被传递到应用程序之前处理它的建议转换器程序的名称（URIMAP 定义中的 CONVERTER 属性）。
- 处理请求并提供响应的用户编写的建议应用程序的名称（URIMAP 定义中的 PROGRAM 属性）。
- 覆盖剩余处理阶段的建议别名事务的事务标识（URIMAP 定义中的 TRANSACTION 属性）。
- 要与别名事务关联的建议用户标识（URIMAP 定义中的 USERID 属性）。如果客户机提供了用户标识，那么可以覆盖该用户标识。

wbra_urimap 输入参数可用于测试是否在请求的处理路径中使用 URIMAP 定义。

如果您正在使用分析器程序（而不是 URIMAP 定义）来处理请求，并且您需要在这方面遵循 HTTP/1.1，那么必须按照 HTTP/1.1 规范中声明的规则编码您的分析器程序以执行 URL 比较。按照这些规则，比较方案名和主机名不区分大小写，但比较路径区分大小写。比较之前 URL 的所有组成部分都未转义。当 CICS 将 URL 与 URIMAP 定义比较时，它将遵循这些规则。

如果愿意，您还可以使用 EXEC CICS WEB API 命令检查 HTTP 请求。使用 EXEC CICS WEB 命令可以增加您的请求分析的准确性和完整性，特别当检查内容和用法容易发生较大变化的 HTTP 头时。EXEC CICS WEB 命令还简化了从请求查找和抽取查询字符串或格式字段信息的过程，这可能是后续处理的决定因素。

您可以使用 EXTRACT TCPIP 命令获取有关正在处理的客户机请求的以下信息:

- Web 客户机的 IP 地址
- Web 客户机的主机名（已告知给 DNS 服务器）
- Web 客户机发送它的连接请求的端口号
- 服务器（即作为 HTTP 服务器的 CICS）的 IP 地址
- 使用的认证类型

- 使用中的 SSL 支持级别
- 与请求关联的 TCPIPSERVICE 资源定义

分析器程序的输出

分析器程序在通信区域中提供输出。该输出包括响应代码，以及可用于指定更多处理阶段并用于与转换器程序共享信息的可选输出参数范围。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

第 354 页的『分析器程序的参数』具有通信区域中所有参数的列表和技术描述。

分析器程序必须在它的通信区域中提供以下输出:

- 响应码。
  - 如果分析器程序返回响应码 URP_OK，处理会继续下一步。
  - 如果分析器程序返回任何其他值，那么 CICS 将错误响应返回到 Web 客户机。可通过用户可替换的 Web 出错程序来修改该响应。第 108 页的『CICS Web Support 缺省状态码和错误响应』介绍了如何将来自分析器的返回码映射到 CICS 返回给 Web 客户机的状态码。

分析器程序也可能提供以下输出:

- 在请求传递到用户编写的应用程序之前，用于处理它的转换器程序的名称。
  - 如果已从 URIMAP 定义输入转换器程序名，那么您可接受或重设它。
  - 如果分析器表明不需要转换器程序，请求的第一个 32K 字节传递到一块存储器中用户编写的应用程序。支持 Web 的应用程序可以忽略该内容并使用 EXEC CICS WEB API 命令读取请求。
- 处理请求并提供响应的应用程序的名称。
  - 如果已从 URIMAP 定义输入程序名，那么您可接受或重设它。
  - 如果您要使用转换器程序，那么转换器程序可指定或重设程序名。可以用这种方式使用转换器以在处理请求时涉及到多个程序。
- 涵盖剩余处理阶段的别名事务的事务标识。如果已从 URIMAP 定义输入事务标识，那么您可接受或重设它。
- 与别名事务关联的用户标识。如果已从 URIMAP 定义输入用户标识，那么您可接受或重设它。如果您未指定用户标识，那么 CICS 将这样确定:
  - 如果已从 URIMAP 定义输入用户标识，那么使用它。
  - 如果 HTTP 请求在客户机认证下使用 SSL，就从客户机证书中获得用户标识。
  - 在其他情况下，使用 CICS 缺省用户标识。
- 如果转换器程序在存储器块中手工进行转换，那么是与包含请求的 32K 存储器块的代码页转换相关的参数，以及与响应主体的代码页转换相关的参数。

  注: 这不影响转换器程序或用户编写的应用程序，它们使用 EXEC CICS WEB API 命令来查看 HTTP 请求并生成响应；且它们直接从 CICS 请求代码页转换。
  您可以用两种方式之一为包含请求的一块存储器转换指定参数:
  - 作为一对参数，它指定 Web 客户机使用的字符集（wbra_characterset）和适用于应用程序的主机代码页（wbra_hostcodepage）。用这种方式指定参数意味着代码页转换表（DFHCNV）中的条目不是必需的。

– 作为 DFHCNV 代码页转换表中条目的键（wbra_dfhcnv_key）。建议仅将其用于升级目的。

如果未指定任何这些参数，那么 CICS 的缺省行为是使用第 111 页的第 10 章，『分析器程序』中描述的标准设置转换文本消息。如果您要禁止对存储器块中的请求和响应执行代码页转换，那么将 wbra_dfhcnv_key 设置为空。

- 该标志表明，使用转换器程序的非 Web Support 的应用程序需要 CICS TS V3 兼容性预处理（wbra_commarea）。提供该标志的目的在于升级。它只在特殊情况下供不使用 EXEC CICS WEB API 命令的应用程序使用（即，它们在存储器中手工生成响应），Web 客户机需要与它从 CICS TS V3 之前版本接受到的响应相同的响应。设置该标志意味着:
  – CICS 不添加通常为 HTTP/1.1 消息插入的任何响应头。仅使用 CICS Transaction Server for z/OS V3R1 前发送到客户机的头。
  – 如果需要错误处理，CICS 发送适合于，并标记为 HTTP/1.0 响应的出错响应，无论 Web 客户机的 HTTP 版本是什么。CICS 将通常使用 HTTP/1.1 出错响应来应答 HTTP/1.1 客户机，但是这可能会使客户机误认为该应用程序通常将发送 HTTP/1.1 响应。
- 八字节的用户令牌，用于在分析器程序和转换器程序之间共享信息。『共享分析器和转换器程序之间的数据』介绍了该工作方式。
- 已修改的请求主体长度的值

分析器可以修改请求的内容:

- 修改过的数据长度可以短于原始数据，或者和原始数据相同。不能延长请求主体。
- 所做的任何更改在传递到转换器程序的数据中可视，但是对于 EXEC CICS WEB API 命令**不**可视。

共享分析器和转换器程序之间的数据

CICS 在分析器程序和转换器程序之间传递三个参数，使数据可由这些处理阶段共享。

user_data 指针

该参数包含阶段间传递的 32K 存储器块的地址。在分析器程序的入口处，指针指向包含 HTTP 请求的存储器块。当转换器程序编码函数完成时，CICS Web Support 使用它来查找包含 HTTP 响应的存储器块，除非 EXEC CICS WEB API 命令已经用于产生响应。

尽管您可以修改该指针所指存储器块地址的内容，但禁止更改分析器程序中的指针值。

在转换器程序和用户编写的应用程序之间，您可以将指针不更改地从一个阶段传递到另一个阶段，或者您可以在程序中发出 GETMAIN 命令并传递新获取的存储器地址（以指针的形式）。

user_data 长度

该参数是 user_data 指针定位的存储器的长度。

用户令牌

用户令牌是一个由分析器程序和转换器程序共享的 8 字节字段。它可以包含任何您想要的信息:

- 您可以在用户令牌里直接传递少量的共享信息。

- 要传递更多的信息，您可以在某个程序中发出 GETMAIN 命令来获取共享工作区域的存储器。然后使用用户令牌传递共享存储器的地址。

  您可以在每个程序中更改用户令牌的内容：例如，用户令牌在从分析器程序传递到转换器程序的解码函数时可能有某种含义，而传递到编码函数时会有不同含义。

分析器程序可以修改参数列表中被传递到转换器程序的任何参数。不能更改指针，但可以更改指针所指向的数据。不应更改每个字段的长度。

注: 分析器和转换器程序在不同的 CICS 任务下执行。因此，如果您在分析器程序中发出 GETMAIN 命令，而存储器要在转换器程序中可见的话，您必须对 SHARED 选项进行编码。通常，CICS 不会自动释放用 SHARED 选项获取的存储器，因此当您的程序不再需要该存储器时，必须发出 FREEMAIN 命令。然而，在 HTTP 响应已发送到 Web 客户机后，CICS 将释放 user_data 指针寻址的存储器。

从分析器程序选择转义数据或未转义数据

传递给分析器程序以进行解析的 HTTP 请求使用它的转义形式。URL 中或消息体中表单数据中的保留字符或排除字符作为 %xx 序列出现，其中 xx 是保留字符的 ASCII 十六进制表示法。分析器可以用以下两种形式将一块 32K 存储器中的请求传递到后续处理阶段：转义序列仍然出现的转义形式，或转义序列转换回原始字符的未转义形式。使用 EXEC CICS WEB API 命令的支持 Web 的应用程序不使用该机制接收响应，且它们直接从 CICS 请求未转义。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

第 15 页的『转义和未转义数据』说明了转义及其目的。转义和未转义仅应用到 HTTP 请求的以下元素:

- 请求行的 URL 部分，包括任何查询字符串。查询字符串可能是来自具有 GET 方法的表单的数据。
- 从具有 POST 方法的表单返回的表单数据以及缺省编码 **application/x-www-form-urlencoded**。该数据出现在消息体中。第 16 页的『HTML 表单』提供了关于表单数据的更多信息。

如果一块 32K 存储器中的请求以未转义形式传递，那么分析器可以将数据从转义形式转换成未转义形式，或由 CICS 执行该转换。

- 要以转义形式传递请求，请将分析器中的 WBRA_UNESCAPE 设置为 WBRA_UNESCAPE_NOT_REQUIRED。WBRA_UNESCAPE_NOT_REQUIRED 是缺省值。
- 要以未转义形式传递请求并由 CICS 执行转换，请将分析器中的 WBRA_UNESCAPE 设置为 WBRA_UNESCAPE_REQUIRED。
- 要在分析器执行转换以后，以未转义形式传递请求，请将 WBRA_UNESCAPE 设置为 WBRA_UNESCAPE_NOT_REQUIRED。

使用 EXEC CICS WEB API 命令的支持 Web 的应用程序不使用通信区域机制接收和发送响应，且它们直接从 CICS 请求未转义。对于使用 EXEC CICS WEB API 命令的支持 Web 的应用程序，当使用 WEB READ FORMFIELD 命令或表单字段浏览命令从

请求抽取表单数据时，CICS 执行取消转义操作，将数据以其未转义的形式返回。当使用 WEB EXTRACT 命令从请求抽取查询字符串时，数据以它的转义形式返回。

如果您正用可以通过 CICS Web Support 或通过 CICS 业务逻辑接口运行的通信区域接口编写应用程序，那么确保 WBRA_UNESCAPE 设置为 WBRA_UNESCAPE_NOT_REQUIRED，且将任何未转义委托给应用程序。如果不执行该操作，那么应用程序通过 CICS 业务逻辑接口传递未转义的数据，而通过 CICS Web Support 传递转义数据，这可能会导致不可预测的结果。

CICS 提供的缺省分析器程序 DFHWBAAX

CICS 提供缺省分析器程序 DFHWBAAX。DFHWBAAX 为用于 CICS Web Support 的 TCPIPSERVICE 资源定义提供错误处理函数。当使用一个端口的所有请求都是使用 URIMAP 定义处理时，它适合使用。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

CICS 仅为汇编程序中的 DFHWBAAX 提供源代码。

DFHWBAAX 是指定 PROTOCOL(HTTP) 的 TCPIPSERVICE 定义的缺省分析器程序。

在通信区域中，DFHWBAAX 将相同的输入和输出参数作为标准分析器程序接收。如前所述，它不使用大多数这些参数，并且它不对使用 URL 格式的请求提供支持，该格式是在 CICS TS 3.1 之前的 CICS Web Support 使用的格式。它而是采用如下的简单操作:

- 已发现请求的匹配 URIMAP 定义后，DFHWBAAX 不执行进一步处理，即使 URIMAP 指定 ANALYZER(YES) 时也是如该。它使用 `wbra_urimap` 输入参数测试 URIMAP 定义是否存在，以及结果是否为正，不对请求 URL 执行任何分析就返回。这意味着，会自动接受在 URIMAP 定义中为别名事务指定的设置、转换器程序（如果使用的话）和应用程序，并将它们用于确定后续处理阶段。
- 如果未找到匹配的 URIMAP 定义，那么 DFHWBAAX 对用户可替换 Web 出错事务程序 DFHWBERX 提供控制以生成错误响应。使用 `wbra_server_program` 输出参数，通过将 DFHWBERX 设置为应用程序以处理请求可以实现此目的。DFHWBAAX 不对通信区域进行任何其他更改。根据 Web 客户机发出的请求，接收控制时，DFHWBERX 提供 404（未找到）状态码的 HTTP 响应，或 SOAP 故障请求。

DFHWBAAX 使用标准范围的响应: **URP_OK**、**URP_EXCEPTION** 和 **URP_INVALID**。如前所述，没有为 DFHWBAAX 构建的原因值。请注意，如果响应不同于 **URP_OK**，那么它表明处理中的错误，且控制传递到用户可替换的 Web 出错程序 DFHWBEP，而不是 Web 出错应用程序 DFHWBERX。

CICS 提供的样本分析器程序 DFHWBADX

CICS 提供有效的样本分析器程序 DFHWBADX。如果您需要通过分析器程序提供请求处理程序，又或者替代通过 URIMAP 定义，您可以将 DFHWBADX 用作编写您自己的分析器程序的起点。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

CICS 提供多种语言的源代码:

- DFHWBADX （汇编语言）
- DFHWBAHX （C）
- DFHWBALX （PL/I）
- DFHWBAOX （COBOL）

已发现匹配请求的 URIMAP 定义后，即使 URIMAP 指定 ANALYZER(YES)，所提供的 DFHWBADX 也不执行请求的任何分析。这意味着为别名事务在 URIMAP 定义中指定设置，自动接受转换器程序和应用程序，并将它们用于确定后续处理阶段。

DFHWBADX 使用 wbra_urimap 输入参数测试 URIMAP 定义是否存在，如果结果是存在，那么不对请求 URL 执行任何分析就返回。如果编写您自己的分析器程序并希望它与 URIMAP 定义交互，那么不要复制 DFHWBADX 处理的这一方面。为了以其他方式修改分析器程序的处理，您可能要测试 wbra_urimap 输入参数。例如，您可以测试该参数以决定是根据来自 URIMAP 定义的输入参数执行分析，还是直接对请求 URL 执行分析。

DFHWBADX 如何解释请求 URL

DFHWBADX 解释 HTTP 请求，这些请求中的 URL 路径部分有以下语法:

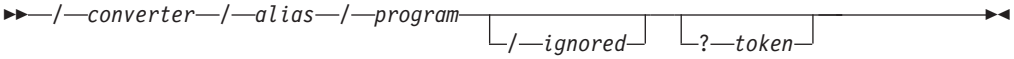


图 7. DFHWBADX 解释的路径部分语法

分析器程序处理的所有字段都转换为大写。转换之后:

converter
指定用于请求的转换器程序的名称。它最多可以有 8 个字符。

作为特例，值 'CICS'（由四个字符组成）表示不使用转换器程序。请参阅第 123 页的第 11 章，『转换器程序』，了解有关如何将转换器程序与 URIMAP 定义结合使用的信息。

alias
指定用于后续请求处理的别名事务的事务标识。它最多可以有 4 个字符。

program
指定服务于请求的 CICS 应用程序的名称。它最多可以有 8 个字符。

ignored
DFHWBADX 忽略这部分路径（但是转换器程序或应用程序可能使用它）。

token
前 8 个字节指定要传递给转换器程序的用户令牌。DFHWBADX 忽略令牌的前八个字节后的数据（但转换器程序或应用程序可能使用它）。

在示例路径 /cics/cwba/dfh$wb1a 中:

- 未使用转换器程序。
- 别名事务是 CWBA。
- CICS 应用程序是 DFH$WB1A。

除了原始 HTTP 请求派生的输出之外，DFHWBADX 还设置下列输出:

- 代码页转换模板为 DFHWBUD。这个模板在样本转换表 DFHCNVW$ 中定义，它在 ASCII Latin-1 字符集（代码页 ISO 8859–1）和 EBCDIC Latin 字符集（代码页 037）之间转换数据。无需进行任何配置就可以使用样本转换表，但是请注意，可以使用输出参数 `wbra_characterset` 和 `wbra_hostcodepage` 来代替 `wbra_dfhcnv_key` 输出参数以提供更好的控制，并避免使用转换表。
- DFHWBADX 以转义形式传递请求，并设置 WBRA_UNESCAPE_NOT_REQUIRED。

来自 DFHWBADX 的响应

DFHWBADX 产生的响应含义如下:

URP_OK

分析器发现请求符合缺省 HTTP 请求格式，并生成适合该别名的输出。

URP_EXCEPTION

分析器发现请求不符合缺省格式。提供的原因码如下:

1 资源长度小于 6。（使用 DFHWBADX 识别的 URL 格式，可能的最短资源的规范是 `/A/B/C` ，要求程序 C 在事务 B 下与转换器 A 一同运行。）响应和原因是当进入请求不是 HTTP 请求时的响应和原因。

2 资源规范不以『/』开始。

3 资源规范包含一个『/』，但少于 3 个。

4 资源规范中转换器名称的长度为 0 或大于 8。

5 资源规范中事务名称的长度为 0 或大于 4。

6 资源规范中 CICS 应用程序名称的长度为 0 或大于 8。

响应和原因代码在消息 DFHWB0723 中显示。错误响应 400（错误请求）状态码返回到 Web 客户机。这可以用用户可替换的 Web 出错程序 DFHWBEP 来修改。

URP_INVALID

eye-catcher 是无效的。这表示一个内部错误。

第 11 章 转换器程序

转换器程序主要与应用程序协同使用，原先没有为了与 Web 协同使用而编码这些应用程序。它们还可用于将多个应用程序的输出组合到单个 HTTP 消息中。

关于此任务

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

当 CICS 是 HTTP 客户机时，或对于 Web Service 处理，不使用转换器程序；仅当 CICS 是 HTTP 服务器时才可以调用它们。转换器程序在作为 HTTP 服务器的 CICS 的 CICS Web Support 进程中的角色如第 25 页的『作为 HTTP 服务器的 CICS 的 HTTP 请求和响应处理』中所述。第 55 页的第 5 章，『规划作为 HTTP 服务器的 CICS 的 CICS Web Support 体系结构』提供信息，帮助您规划将 CICS 作为 HTTP 服务器的体系结构。

URIMAP 定义可以调用转换器程序以执行 HTTP 请求的相关处理。如果 CICS Web Support 处理中使用了分析器程序，那么分析器程序也可以调用转换器程序。转换器程序在以下情况下可以有用:

- 最初不是编码与 Web 协同使用的应用程序需要接收通信区域格式的输入，或者需要将它们的输出转换为 HTTP 响应。使用 EXEC CICS WEB 和 EXEC CICS DOCUMENT 应用程序编程接口编码的支持 Web 的应用程序不需要进行此转换。您可以使用转换器程序以执行此转换或请求内容上的其他处理。
- 当您要使多个应用程序对序列中的同一请求数据起作用，并将单个 HTTP 响应返回到 Web 客户机时。

如果从 URIMAP 定义直接调用转换器程序，那么 URIMAP 定义的 PROGRAM 属性（它指定要处理请求的应用程序名）可以传递到转换器程序，且转换器程序可以选择覆盖它。

转换器程序将 Web 客户机请求以及可提供更多请求信息的参数列表接收到存储器块中。转换器程序将请求内容处理成适合于将为响应提供数据的应用程序的格式，然后将它传递到 COMMAREA 中的应用程序。这个序列称为转换器程序的**解码**函数。如果转换器程序没有使用**解码**函数为应用程序创建 COMMAREA，那么用于接收 HTTP 请求的 32 767 字节缓冲区将被传递给该应用程序。

应用程序将其结果返回给转换器程序。如果需要多个应用程序为响应产生数据，那么转换器程序可以调用更多的应用程序或程序。当转换器程序具有应用程序中的所有必需数据时，它会产生 HTTP 响应，以发送到 Web 客户机。这个序列称为转换器程序的**编码**函数。

转换器程序不会与分析器程序以同样的方式与 TCPIPSERVICE 定义关联。您可以使用任何转换器程序来处理任何 HTTP 请求，但 HTTP 对于接收请求的 CICS 系统来说必须是本地转换器程序。对于给定的请求，为解码和编码函数调用同一个转换器程序。

用户可替换的程序对于操作 CICS Web Support 的系统来说都必须是本地的。如果您不使用程序的自动安装，那么必须为 CICS Web Support 使用的所有用户可替换程序（包

括分析器和转换器程序）定义和安装程序定义。如果使用程序的自动安装，那么必须确保用正确的属性安装了用户可替换程序。

CICS 业务逻辑接口也使用转换器程序。CICS 业务逻辑接口中的转换器程序角色在第 322 页的『使用 CICS 业务逻辑接口调用程序』中描述。CICS 业务逻辑接口的调用者确定是否需要转换器程序，以及应该调用哪个转换器程序。

编写转换器程序

要编写转换器程序，需要构造解码和编码函数，并考虑代码页转换问题。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

您可以用汇编程序、C、COBOL 或 PL/I 来编写转换器程序。在第 361 页的附录 F，『转换器程序的参考信息』中，描述了和语言相关的头文件、包含文件以及副本。

解码函数: 查看并处理 HTTP 请求

转换器程序的解码函数接收来自 Web 客户机的 HTTP 请求，以及提供有关该请求的更多信息的参数列表。HTTP 请求会传递到转换器程序的一块 32K 存储器中，该存储器区域由参数列表中的指针表示。请求已被分为多个单独的元素，例如，方法、请求头和请求主体。（请注意，如果请求太长而不适合存储器块，那么其余数据将不传递到转换器程序。）如果处理路径中使用分析器程序，那么分析器程序可能修改了请求的内容。

在 CICS Web Support 的转换器程序中，如果您希望，可以使用 EXEC CICS WEB API 命令来检查 HTTP 请求。WEB EXTRACT 命令检索有关请求的信息（如方法和版本）。WEB READ HTTPHEADER 命令或 HTTPHEADER 浏览命令可用于读取 HTTP 头。WEB RECEIVE 命令可用于接收请求的主体。如果使用任意一个 EXEC CICS WEB API 命令，那么请注意，这些命令将返回 Web 客户机请求中的原始信息，而且您不能使用它们来查看分析器程序所做的任何修改。分析器程序所做的更改只会在直接传递给转换器程序的参数列表和存储器块中反映出来。

参数列表中提供了用户编写的应用程序的名称，这些程序用于为响应提供数据，而名称来自请求的 URIMAP 定义或由分析器程序设置。如果使用了分析器程序，那么它可以通过用户令牌向转换器程序提供更多的信息。

通过使用您所获得的关于 Web 客户机请求的信息，转换器程序的解码函数需要:

- 确定是否应该为请求继续处理，或者 CICS 是否应该将错误响应返回到 Web 客户机。
- 指定要处理请求并提供响应的用户编写的应用程序名。如果名称已从 URIMAP 定义或由分析器程序输入，那么转换器程序可以接受或更改此内容。
- 构造传递到用户编写的应用程序的 COMMAREA。通信区域包括来自 Web 客户机的请求的数据，该请求已转换为应用程序可接受的输入格式。可以复用包含 HTTP 请求的存储器块，或者可以指定新的 COMMAREA。

编码函数: 生成响应

当用户编写的应用程序已通过使用由转换器程序提供的输入执行处理时，转换器程序的编码函数会从应用程序接收输出 COMMAREA。使用该数据，转换器程序的编码函数需要:

- 如果需要多个应用程序提供数据，那么调用更多的应用程序。为了做到这一点，编码函数会设置循环响应来再次调用解码函数。解码函数更改应用程序的名称，并在 COMMAREA 中提供合适的输入。输出将再次返回到编码函数。第 128 页的『从转换器程序调用多个应用程序』具有有关此内容的更多信息。
- 构造要发送到 Web 客户机的 HTTP 响应。

在 CICS Web Support 的转换器程序中，可以使用 EXEC CICS WEB API 命令来生成响应并将它发送到 Web 客户机。WEB WRITE HTTPHEADER 命令可用于编写响应的 HTTP 头。WEB SEND 命令可用于组装并发送响应。

或者，可使用转换器程序在存储缓冲区中手工构造 HTTP 响应，然后将它返回给 CICS，以便发送到 Web 客户机。响应必须包含 HTTP 版本、状态码、状态文本、所需的任何 HTTP 头和消息体。响应的格式应该符合您正在遵从的 HTTP 协议规范（HTTP/1.0 或 HTTP/1.1）。要获取 HTTP 响应的存储缓冲区，您可以:

- 发出 GETMAIN 命令以获取存储器。
- 使用在处理的早期阶段（例如分析器程序）中获得的存储器。
- 构造用户编写的应用程序返回的 COMMAREA 中的响应。

用于响应的区域的第一个字必须包含区域长度（也就是说，HTTP 响应长度加 4）。在转换器程序的编码函数的出口，参数列表中的数据指针必须指向该存储器块。（如果使用 EXEC CICS WEB API 命令来发送响应，那么 CICS 会忽略并丢弃该指针所表示的任何存储器块。）

不论使用哪种方法构造 HTTP 响应，CICS 通常会插入适合 HTTP/1.0 或 HTTP/1.1 响应的某些 HTTP 头，这些头在第 335 页的附录 B，『CICS Web Support 的 HTTP 头参考』中列出。如果转换器程序生成的响应已经包含这些头，那么 CICS 不会替换它们。如果分析器程序已经将响应标记为与 CICS TS V3 之前的版本的处理兼容，这是因为 Web 客户机需要的响应与它在 CICS TS V3 之前的版本接收到的响应是相同的，那么仅使用发送到 CICS Transaction Server for z/OS V3R1 之前的客户机的头。仅当使用转换器程序在存储器块中手工生成响应时，该标志才起作用。如果转换器程序使用 EXEC CICS WEB API 命令来发送响应，那么该标志不起作用。

代码页转换

当在转换器程序中使用 EXEC CICS WEB API 命令来查看 HTTP 请求并生成响应时，会象您在命令中指定的那样进行代码页转换，其方法与使用 EXEC CICS WEB API 命令的任何其他程序一样。

转换器程序无法为传入 32K 存储器块的 HTTP 请求指定代码页转换设置。如果直接从 URIMAP 定义调用转换器程序，并且 Web 客户机请求的头表明该消息体是文本，那么 CICS 使用以下标准设置转换存储器块中提供的消息体:

- 对于字符集，如果 Web 客户机的请求具有一个 Content-Type 头，其中指定了 CICS 所支持的字符集，那么使用该字符集。 如果 Web 客户机的请求没有 Content-Type 头或指定的字符集不受支持，那么使用 ISO-8859-1 字符集。
- 对于主机代码页，CICS 将缺省代码页用于本地 CICS 区域，如 LOCALCCSID 系统初始化参数中所指定。

如果这些标准设置不合适，或者如果不希望进行代码页转换，那么在处理路径中使用分析器程序来指定备用代码页转换设置，或使用 EXEC CICS WEB API 命令来处理请求。

如果您使用转换器程序在存储器缓冲区中手工构造 HTTP 响应，那么 CICS 会反映出已经为传入 32K 存储器块的请求执行的代码页转换。通过使用分析器程序指定的字符集和主机代码页设置，或缺少分析器程序时使用本主题中描述的标准设置，将响应发送到 Web 客户机。如果分析器程序禁止对请求执行代码页转换，那么不会对响应主体执行代码页转换。 如果该输出不合适，请使用 EXEC CICS WEB API 命令来生成响应。

CICS 业务逻辑接口的转换器程序

使用具有 CICS 业务逻辑接口的转换器程序时，某些限制可能会影响您如何构造传递给用户编写的应用程序的通信区域，以及包含响应的存储器缓冲区。要获取更多信息，请参阅第 328 页的『偏移方式和指针方式』。

不要在为 CICS 业务逻辑接口编写的转换器程序中使用 EXEC CICS WEB API 命令。

转换器程序解码函数的输入参数

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

输入参数会传递到参数列表中的解码函数。

这些参数包含:

- Web 客户机的 IP 地址。
- 指向 Web 客户机请求的 HTTP 版本的指针。
- 指向请求方法的指针。
- 指向 URL 的路径部分的指针。
- 指向请求的 HTTP 头的指针。
- 指向请求消息的实体主体的指针。
- 为请求提供数据的 CICS 应用程序的名称（由分析器程序设置，或在 URIMAP 定义中指定）。
- 八字节的用户令牌，用于在分析器程序和转换器程序之间共享信息。请参阅第 117 页的『共享分析器和转换器程序之间的数据』。
- 用于记录为每个 HTTP 请求调用解码函数次数的迭代计数器。计数器在第一次调用解码函数之前设置成 1，然后在以后的每次调用之前加一。
- 实体主体的地址是否可以成为 FREEMAIN 命令的目标的指示。

在将参数列表传递到转换器程序之前，分析器程序可以更改其中任何参数的值。如果要查看 Web 客户机中的原始请求，请在转换器程序中使用 EXEC CICS WEB API 命令。

转换器程序解码函数的输出参数

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

解码函数必须在 COMMAREA 中提供以下输出: 响应代码, 以及 COMMAREA 的地址和长度。

- 响应码（可选地由原因码限定）。

  如果解码函数返回 URP_OK 响应代码, 处理将继续到下一步。

  如果解码函数返回其他任何值, 那么会用错误响应表示拒绝 HTTP 请求。有关这种情况下 CICS 所作响应的详细信息, 请参阅 第 108 页的『CICS Web Support 缺省状态码和错误响应』。
- 传递到用户编写的应用程序的 COMMAREA 的地址和长度。如果未调用应用程序, 那么将通信区域不更改地传递给编码函数。

解码函数也可以提供以下输出:

- 为请求提供数据的用户编写的应用程序的名称。如果分析器程序提供了名称, 转换器程序可以更改它, 或者指定不应调用应用程序。
- 八字节的用户令牌, 用于在分析器程序和转换器程序之间共享信息。请参阅第 117 页的『共享分析器和转换器程序之间的数据』。

转换器程序编码函数的输入参数

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

输入参数被传递到 COMMAREA 中的编码函数。

这些参数包含:

- 由用户编写的应用程序返回的 COMMAREA 的地址和长度。如果未调用应用程序, 那么从解码函数不更改地传递通信区域。
- 八字节的用户令牌, 用于在分析器程序和转换器程序之间共享信息。请参阅第 117 页的『共享分析器和转换器程序之间的数据』。
- 用于记录为每个 HTTP 请求进入的编码函数次数的迭代计数器。计数器在第一次调用编码函数之前设置成 1, 然后在以后的每次调用之前加一。

转换器程序编码函数的输出参数

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

编码函数可以提供以下输出: 响应代码、存储器缓冲区的地址、HTTP 响应的长度以及 8 字节的用户令牌。

- 响应代码（可选地由原因限定）:
  - 如果编码函数返回 URP_OK 响应码, 那么 CICS 会将提供的 HTTP 响应发送给 Web 客户机, 除非您已使用 EXEC CICS WEB API 命令来完成这一操作。
  - 如果编码函数返回 URP_OK_LOOP 响应码, 处理将继续到解码函数。要获取更多信息, 请参阅第 128 页的『从转换器程序调用多个应用程序』。
  - 如果编码函数返回其他任何值, 那么会用错误响应表示拒绝 HTTP 请求。有关这种情况下 CICS 所作响应的详细信息, 请参阅 第 108 页的『CICS Web Support 缺省状态码和错误响应』。

- 如果您在存储器缓冲区中手工构造了 HTTP 响应，存储器缓冲区的地址和 HTTP 响应的长度。缓冲区的第一个字必须包含数据长度（也就是说，HTTP 响应长度加 4）。如果使用 EXEC CICS WEB API 命令来发送响应，那么 CICS 会忽略并丢弃该指针指示的任何存储器块。
- 八字节的用户令牌，用于在分析器程序和转换器程序之间共享信息。请参阅第 117 页的『共享分析器和转换器程序之间的数据』。

从转换器程序调用多个应用程序

有时，构造对 HTTP 请求的响应所需的数据来自于用户编写的多个应用程序。

关于此任务

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

如果是这种情况，那么可以根据需要重复以下序列:

- 转换器的解码函数。
- 应用程序。
- 转换器的编码函数。

通过在编码函数中将响应设置成 URP_OK_LOOP 以完成这一操作。当 HTTP 响应完成时，将响应设置成 URP_OK。

当解码函数在第二个和随后场合被调用时，以下输入参数不可用:

- HTTP 版本。
- 方法。
- URL 的路径部分。
- 请求头。
- 实体主体。

然而，可以使用 WEB EXTRACT 命令来检索相同的信息。

使用参数列表中的数据指针和用户令牌，以共享解码和编码函数之间的数据。最后一次调用编码函数时，如果正在存储缓冲区中手工构造 HTTP 响应，那么请确保数据指针（**encode_data_ptr**）指向一个有效的 HTTP 响应。如果正在使用 EXEC CICS WEB API 命令来生成并发送响应，那么在该阶段执行该操作；在这种情况下，CICS 会忽略并丢弃该指针所指向的任何存储器块。

第 12 章 来自 CICS 应用程序的 HTTP 客户机请求

CICS 可以担当 HTTP 客户机，并与因特网上的 HTTP 服务器通信。用户编写的应用程序通过 CICS 将请求发送到 HTTP 服务器，并从 CICS 接收响应。

在 CICS Transaction Server for z/OS V3R1 中，CICS 担当 HTTP 客户机的能力完全集成到 CICS Web Support 中。为了达到以下目的，您可能要使用该设施:

- 使用 HTTP 协议控制硬件或软件（例如，通常可以用该方式控制打印机）。
- 访问提供信息项（例如，股价）的 HTTP 应用程序，并检索该信息以在应用程序中使用。

注意，CICS Web Support 的 HTTP 客户机设施不旨在用作浏览器。用户应用程序可以对服务器提供的个别、已知资源发出请求，但是浏览因特网时通常不需要它们。可能从服务器接收的响应范围和处理这些响应所需执行的操作应该只与以下内容有关: 您预选的资源，以及可能与那些资源和您正在发出的请求类型关联的错误响应。

第 29 页的『作为 HTTP 客户机的 CICS 的 HTTP 请求和响应处理』说明了作为 HTTP 客户机的 CICS 的处理结构。编写发出 HTTP 客户机请求的应用程序前，确保您了解这些请求的处理阶段，因为大多数阶段由应用程序本身启动。

当 CICS 是 HTTP 客户机时，使用多个 CICS Web Support 设施:

- 您可以在 CICS 中使用 EXEC CICS WEB API 命令打开到服务器的 HTTP 连接、发出请求，并接收供应用程序处理的响应。『通过作为 HTTP 客户机的 CICS 发出 HTTP 请求』说明了如何执行该操作。
- 对于 CICS 发出的请求和接收的响应，需要执行代码页转换。第 39 页的『作为 HTTP 客户机的 CICS 的代码页转换』介绍了该过程。
- URIMAP 定义可用于避免在您的应用程序中指定 URL 或证书标签之类的信息。第 142 页的『通过作为 HTTP 客户机的 CICS 创建 HTTP 请求的 URIMAP 定义』介绍了如何创建这些定义。
- 全局用户出口 XWBAUTH、XWBOPEN 和 XWBSNDO 可用于为 HTTP 客户机请求提供基本认证凭证、指定代理服务器的使用和应用安全策略。请参阅 第 143 页的『HTTP 客户机发送出口 XWBAUTH』、第 147 页的『HTTP 客户机开放出口 XWBOPEN』 和 第 148 页的『HTTP 客户机发送出口 XWBSNDO』，以了解更多有关这些出口的信息。

用于作为 HTTP 服务器的 CICS 的 TCPIPSERVICE 资源定义不应用到作为 HTTP 客户机的 CICS，并且您不需要创建它们以发出 HTTP 客户机请求。

通过作为 HTTP 客户机的 CICS 发出 HTTP 请求

因特网上从 CICS 到服务器进行的 HTTP 客户机请求由用户编写的应用程序启动。本主题将介绍如何编写发出 HTTP 客户机请求的应用程序。

开始之前

由于大多数阶段由应用程序本身启动，因此编写发出 HTTP 客户机请求的应用程序前，请先阅读有关这些请求的处理阶段的信息。第 29 页的『作为 HTTP 客户机的 CICS 的 HTTP 请求和响应处理』说明了应用程序需要执行哪些操作，以及该过程期间 CICS 会执行哪些操作。

关于此任务

对于作为 HTTP 客户机的 CICS，应用程序向服务器发出请求，并等待响应。使用一个会话令牌来区别多个连接，应用程序就可以控制它们。

要发出 HTTP 请求和接收响应，请编写您的应用程序以按以下过程进行操作:

1. 使用 WEB OPEN 命令启动到服务器的连接。 第 131 页的『打开到 HTTP 服务器的连接』介绍了如何执行该操作。 该步骤确保服务器可用，并生成可用于管理连接的会话令牌。
2. 使用 WEB WRITE HTTPHEADER 命令编写请求的 HTTP 头。第 132 页的『为请求写 HTTP 头』介绍了如何执行该操作。 CICS 自动提供 HTTP/1.1 消息需要的头。您可以出于其他目的提供其他头。
3. 需要的话，生成实体主体或消息体，它是 HTTP 请求的内容。该过程与 CICS 作为 HTTP 服务器时是一样的，如第 76 页的『为 HTTP 消息产生实体主体』中描述的。对于许多请求方法，不使用实体主体，但是对于 POST 和 PUT 方法（它们用于对服务器提供数据），它是必需的。 实体主体可以由 CICS 文档（使用 EXEC CICS DOCUMENT 应用程序编程接口创建的）形成，或由应用程序提供的数据缓冲区形成。
4. 使用 WEB SEND 或 WEB CONVERSE 命令将请求发送到 Web 客户机。第 133 页的『写 HTTP 请求』介绍了如何执行该操作。 您需要选择合适的方法，并指定实体主体。CICS 使用这些项和 HTTP 头汇编请求。 如果要将实体主体作为一系列较小的部分（或块）进行发送，那么您还需要遵循第 78 页的『使用分块的传输编码发送 HTTP 请求或响应』中的特殊指示信息。
5. 如果要将更多请求作为管道化的序列发送，那么使用第 135 页的『发送管道化序列的请求』中的指导来执行该操作。
6. 如果服务器要求用户名和密码，请根据第 135 页的『提供基本认证的凭证』中的指示来提供这些详细信息。
7. 等待并接收请求，按第 136 页的『接收 HTTP 响应』中的过程进行操作:
   a. 使用 WEB RECEIVE 命令（或您先前发出的 WEB CONVERSE 命令）接收响应的消息体。
   b. 使用 WEB READ HTTPHEADER 命令或 HTTP 头浏览命令读取响应的头。
   c. 根据状态码，处理服务器的响应，并执行应用程序的业务逻辑。
   d. 如果发送了管道化序列的请求，那么使用更多 WEB RECEIVE 命令从服务器接收其余响应。
8. 如果需要更多请求和响应，请重复该过程。 如果服务器支持持续连接，且不关闭连接，那么不需要打开一个新连接。您可以继续使用同一会话令牌。 服务器关闭连接后，如果要发出更多请求，那么需要再次发出 WEB OPEN 命令。
9. 完成与服务器的合作后，关闭连接。 第 138 页的『关闭到 HTTP 服务器的连接』说明了如何执行该操作。

打开到 HTTP 服务器的连接

在 CICS Web Support 中发出 HTTP 客户机请求时，必须在发送第一次请求前打开到服务器的连接。CICS 返回表示连接的会话令牌。

开始之前

关于此任务

通过发出 WEB OPEN 命令以启动与服务器的连接，具体如下:

1. 指定服务器的主机名、主机名长度和要使用的方案（HTTP 或 HTTPS）。 如果主机的端口号不是已指定方案的缺省值，那么也要指定它。 您还可以在 WEB OPEN 命令中指定 URIMAP 选项，以直接从现有 URIMAP 定义使用该信息。 也可以使用 SCHEME、HOST、HOSTLENGTH 和 PORTNUMBER 选项提供信息。要抽取这些详细信息，可以使用 WEB PARSE URL 命令从已知 URL 抽取，或使用 WEB EXTRACT URIMAP 命令从现有 URIMAP 定义抽取。
2. 需要的话，指定 CODEPAGE 选项将该连接的 EBCDIC 代码页更改为与本地 CICS 区域的缺省代码页（由 LOCALCCSID 系统初始化参数设置）不同的代码页。这可能是另一种本地语言的 EBCDIC 代码页。 服务器返回其响应时，如果指定了转换，那么 CICS 将响应主体传递到应用程序之前将它转换到该代码页中。
3. 如果您使用 HTTPS 方案，那么指定合适的安全选项:
   a. 如果您需要提供 SSL 客户机证书，那么指定 CERTIFICATE 选项来执行该操作。 如果您在 WEB OPEN 命令中指定 URIMAP 选项，那么可以直接使用现有 URIMAP 定义中的信息。
   b. 使用 CIPHERS 和 NUMCIPHERS 选项指定用于连接的密码套件代码列表。 如果您在 WEB OPEN 命令中指定 URIMAP 选项，那么可以接受 URIMAP 定义的设置，或指定您自己的密码套件代码列表以覆盖 URIMAP 规范。
4. 如果第一个规划的请求所含操作不受部分版本的 HTTP 协议支持，而且您要检查服务器的 HTTP 版本以确认该操作能否成功，那么指定 HTTPVNUM 和/或 HTTPRNUM 选项以返回该信息。 如果您还不知道服务器的 HTTP 版本，而且要执行受 HTTP 协议影响的操作（如下所示），那么可能需要采用上述方法:
   - 写 HTTP 头，这些头用于请求低于 HTTP/1.1 级别的服务器可能无法正确执行的操作。
   - 使用 HTTP 方法，这些方法可能不适用于低于 HTTP/1.1 级别的服务器。
   - 使用分块的传输编码。
   - 发送管道化请求序列。

   记住: 在执行受版本影响的操作之前，无需一直检查服务器的 HTTP 版本。请参考所采用的 HTTP 级别的规范，看看是否可以用版本不匹配的服务器来执行操作。例如，接收方可能会忽略一些不适合的 HTTP 头。您可以在无需检查的情况下尝试请求，并处理来自服务器的错误响应。 如果不需要该信息，那么不要指定 HTTPVNUM 和 HTTPRNUM 选项，这样可以提高性能。
5. WEB OPEN 命令驱使 XWBOPEN 用户退出。需要的话，您可以创建一个用户出口程序以使到服务器的连接通过代理服务器，或者将安全策略应用到主机名。 第 147 页的『HTTP 客户机开放出口 XWBOPEN』 具有帮助您执行该操作的信息。

结果

CICS 打开与服务器的连接，并将会话令牌返回给应用程序。

下一步做什么

保存会话令牌并将它用于与该连接有关的所有后续命令。

为请求写 HTTP 头

对于客户机 HTTP 请求，根据用于消息的 HTTP 协议版本，CICS 自动提供基本消息必需的 HTTP 头。您可能需要将更多 HTTP 头添加到您的请求。

关于此任务

如果消息需要某些 HTTP 头，那么 CICS 将自动创建它们。CICS 创建的完整头列表如下:

- ARM 相关因子
- Connection
- Content-Type（由 CICS 编写，但在需要复杂头时，可由客户机应用程序提供）
- Content-Type
- Date
- Expect
- Host
- Server
- TE（由 CICS 编写，但是可能添加了更多实例）
- Transfer-Encoding
- User-Agent
- WWW-Authenticate

这些头中的一部分只适合于作为 HTTP 服务器的 CICS。创建这些头的情况在 第 335 页的附录 B，『CICS Web Support 的 HTTP 头参考』中描述。除了 Content-Type 和 TE 头外，CICS 不允许您编写自己版本的 CICS 提供的请求头。

CICS 为请求提供的头是通常应该为基本 HTTP/1.1 消息所写的、以使该消息遵从 HTTP/1.1 规范的头。（CICS 以提供的 HTTP/1.1 为 HTTP 版本发送您的请求。）您可能为了诸如以下目的要添加更多 HTTP 头:

- 声明服务器的首选项（例如，Accept-Encoding、Accept-Language）
- 发出条件请求（例如，If-Match、If-Modified-Since）
- 对服务器或代理提供基本认证信息（Authorization、Proxy-Authorization）

要获取与您决定用于消息的任何其他 HTTP 头相关的需求，请检查您正在遵从的 HTTP 规范。请参阅 第 12 页的『HTTP 协议』以获取有关 HTTP 规范的更多信息。

在您发出 WEB SEND 命令以发生消息**前**，为消息编写其他 HTTP 头。该规则的特例是您编写的头要在已分块消息上作为尾部头发送，在该情况下，应用下面提到的特殊进程。为请求写 HTTP 头时:

- 对于所有命令，使用 SESSTOKEN 选项为该连接指定会话令牌。

- 对您要添加到消息的每个头，使用 WEB WRITE HTTPHEADER 命令。 确保以您正在遵从的 HTTP 规范所描述的格式为每个头指定名称和值。 该命令添加单个头，您可以重复该命令以添加更多头。如果您编写已经为请求写好的头，那么 CICS 除了现有头外，还将新的头添加到请求。确保只在 HTTP 规范声明可以重复头的地方执行该操作。 CICS 发送请求时，会存储准备添加到该请求的头。
- 如果您不知道服务器的 HTTP 版本，并且想使用头来请求一个操作（低于 HTTP/1.1 级别的服务器可能无法正确执行该操作），那么使用 WEB EXTRACT 命令来检查服务器的 HTTP 版本。 记住: 在执行受版本影响的操作之前，无需一直检查服务器的 HTTP 版本。请参考所采用的 HTTP 级别的规范，看看是否可以用版本不匹配的服务器来执行操作。例如，接收方可能会忽略一些不适合的 HTTP 头。您可以在无需检查的情况下尝试请求，并处理来自服务器的错误响应。
- 如果希望生成要在某个 HTTP 头（例如，If-Modified-Since 头）中使用的日期和时间戳记，那么可以使用 FORMATTIME 命令。 命令中的 STRINGFORMAT 选项用于将当前日期和时间（ABSTIME 格式）或应用程序生成的日期和时间转换为适合在 Web 上使用的日期和时间戳记格式。其他日期和时间戳记格式可能不被与 CICS 通信的某些 Web 客户机或服务器所接受。
- 如果正在使用分块的传输编码发送 HTTP 请求，并且要将尾部头包括在分块消息的末尾，那么按第 78 页的『使用分块的传输编码发送 HTTP 请求或响应』中的特殊指示信息进行操作。 在发送消息的第一块前，需要写尾部头。继 WEB SEND 命令后为第一块写的所有 HTTP 头作为尾部头来对待。
- 确保您的应用程序执行用户所写的头暗示的任何操作。

写 HTTP 请求

对于作为 HTTP 客户机的 CICS，WEB SEND 命令或 WEB CONVERSE 命令可用于发出请求。WEB CONVERSE 命令组合了 WEB SEND 命令和 WEB RECEIVE 命令中可用的选项，因此您可以使用单条命令以发出请求和接收响应。

开始之前

发出请求时，您需要指定 HTTP 方法。该方法告诉服务器对您的请求执行何种操作。第 349 页的附录 D,『CICS Web Support 的 HTTP 方法参考』提供了有关可与 CICS Web Support 一起使用的各种方法的基本指导。要获取更详细的指导（包括应用到方法使用的任何需求），应该查询您正在遵从的 HTTP 规范。请参阅 第 12 页的『HTTP 协议』以获取有关 HTTP 规范的更多信息。CICS 以提供的 HTTP/1.1 为 HTTP 版本发送您的请求。

发出请求前，使用 WEB WRITE HTTPHEADER 命令写请求的任何额外 HTTP 头，如第 132 页的『为请求写 HTTP 头』中所描述。

关于此任务

需要的话，请求可以按块（分块的传输编码）发送，或者可以发送管道化请求序列。

CICS Application Programming Reference 提供有关 WEB SEND 和 WEB CONVERSE 命令中可用选项的完整参考信息和描述。发出所选的命令时:

1. 使用 SESSTOKEN 选项为该连接指定会话令牌。

2. 为请求指定 HTTP 方法（OPTIONS、GET、HEAD、POST、PUT、DELETE 或 TRACE）。第 349 页的附录 D, 『CICS Web Support 的 HTTP 方法参考』提供了正确使用所有这些方法的指导。
3. 为应用程序需要访问的服务器上的资源指定路径信息。 缺省值是任何 URIMAP 定义中提供的路径，您在该连接的 WEB OPEN 命令中引用了该定义。通过使用 URIMAP 选项指定可以从中获取路径的另一个 URIMAP 定义，可以指定备用路径。（新的 URIMAP 定义必须为当前连接指定正确的主机名。）也可以使用 PATH 和 PATHLENGTH 选项来提供路径信息。
4. 使用 QUERYSTRING 和 QUERYSTRLEN 选项为您的请求指定任何查询字符串。
5. 为 HTTP 请求指定任何实体主体及其长度。 第 349 页的附录 D, 『CICS Web Support 的 HTTP 方法参考』说明了请求主体适用以及不适用的情形。
   - 对于 GET、HEAD、DELETE 和 TRACE 方法，请求主体不适合。
   - 对于 OPTIONS 方法，请求主体是允许的，但是目前 HTTP/1.1规范不会为该主体定义任何目的。
   - 对于 POST 和 PUT 方法，必须使用请求主体。

   如果需要请求主体，那么主体内容可以来自 CICS 文档（通过 CICS DOCUMENT 接口并指定 DOCTOKEN 选项来确定该文档）或者来自缓冲区内容（指定 FROM 选项）。第 76 页的『为 HTTP 消息产生实体主体』说明了如何生成该内容，并介绍了有关主体长度的大小限制的信息。
6. 使用 MEDIATYPE 选项为您正在提供的任何实体主体指定介质类型。 对于具有 POST 和 PUT 方法的请求（它需要主体），需要指定 MEDIATYPE 选项。对于具有其他方法的请求（其中未提供主体内容），不需要 MEDIATYPE 选项。
7. 如果请求主体**不**需要代码页转换，那么指定合适的转换选项（根据您正在使用的是 WEB SEND 命令还是 WEB CONVERSE 命令），因此 CICS 不转换请求主体。 对于作为 HTTP 客户机的 CICS，缺省设置是转换请求主体， 除非它具有非文本介质类型。
8. 如果需要代码页转换，并且缺省 ISO-8859-1 字符集不适合，那么指定适合于服务器的字符集。 大多数服务器都应接受 ISO-8859-1，除非您区别于服务器选择备用项的较早连接。
9. 如果要使用 Expect 头来测试服务器是否接受请求，那么为 ACTION 选项指定 EXPECT。 该设置使 CICS 将 Expect 头与请求行和请求头一起发送，并在将消息体发送到服务器前等待 100-Continue 响应。如果接收到的不是 100-Continue 响应，那么 CICS 会通知应用程序并且取消发送。如果等待一段时间后未接收到响应，那么 CICS 无论如何也发送消息体。 低于 HTTP/1.1 的服务器不支持 Expect 头。如果 CICS 还不知道服务器的 HTTP 版本，那么 CICS 会在发出您的请求之前发出另一个请求，以便确定服务器的 HTTP 版本。如果 Expect 头不适用，那么 CICS 会发送不带该头的请求。
10. 如果这是您要对该服务器发出的最后一次请求，那么为 CLOSESTATUS 选项指定 CLOSE。 CICS 写请求的 Connection: close 头，或者对于 HTTP/1.0 级别的服务器，省略 Connection: Keep-Alive 头。在发出最后一次请求时指定该选项是好行为，因为头中的信息意味着服务器在发送最后一条响应后可以立即关闭它与您的连接，而不是等着看您是否在超时前发送更多请求。仍将接收来自服务器的响应，并且该响应对应用程序可用。然而，您将无法使用该连接将任何进一步请求发送到服务器。

11. 如果要使用分块的传输编码将请求主体作为一系列块发送，那么按第 78 页的『使用分块的传输编码发送 HTTP 请求或响应』中的其他指示信息进行操作。

    注: 以下各项不支持分块的传输编码:

    - 低于 HTTP/1.1 级别的服务器。
    - WEB CONVERSE 命令。
    - CICS 文档（DOCTOKEN 选项）。

12. 如果要发送管道化序列的请求，那么按『发送管道化序列的请求』中的指示信息进行操作。

结果

CICS 汇编请求行、HTTP 头和请求主体，并将请求发送到服务器。

发送管道化序列的请求

您可以发送更多请求而无需等待来自服务器的响应。这称为管道传送。使用 WEB SEND 命令来发送管道化的请求，而不能使用 WEB CONVERSE 命令（因为该命令包括等待响应）。

开始之前

关于此任务

第 35 页的『CICS Web Support 如何处理管道传送』具有有关管道传送的更多信息。如果您要管道化请求:

1. 确保与服务器有持续连接。 HTTP/1.1 规范允许您尝试一次以发送管道化的序列，而无需检查连接是否持续。如果该次尝试失败，那么必须在重试请求前进行检查。要确定连接的特性:
   a. 如果在对连接执行的 WEB OPEN 命令上指定了 HTTPVNUM 和 HTTPRNUM 选项，请检查返回的信息，以确定服务器的 HTTP 版本。
   b. 如果没有在 WEB OPEN 命令上指定这些选项，那么使用 WEB EXTRACT 命令来确定服务器的 HTTP 版本。
   c. 如果已从服务器接收到先前的响应，那么使用 WEB READ HTTPHEADER 命令以检查服务器是否发送 Connection:close 头或 Connection: Keep-Alive 头。

   处于 HTTP/1.1 级别且**不**发送 Connection: close 头的服务器，以及处于 HTTP/1.0 级别且**发送** Connection:Keep-Alive 头的服务器支持持续连接。 CICS 不会为您执行检查，这是因为管道化请求没有特殊的头以示区分，因此 CICS 不能确定客户机应用程序是否发送了管道化请求序列。
2. HTTP/1.1 规范声明: 您的请求序列应该是幂等的; 即，如果重复所有或部分序列，那么应该获得同样的结果。 第 18 页的『管道传送』具有有关幂等性的更多信息。
3. 除了管道化序列中的*最后*一个请求外，不要对任何其他请求指定 CLOSESTATUS(CLOSE)。

提供基本认证的凭证

当接收到 HTTP 401 WWW-Authenticate 消息时，应用程序必须提供服务器进行基本认证所需的用户名和用户名（凭证）。应用程序也可以提供这些凭证，而无需等待 401 消息。

1. 使用 WEB OPEN 命令以及 SESSTOKEN 选项，打开服务器的 Web 会话。 会话成功打开后会返回 SESSTOKEN，而且会话令牌必须用在与该连接相关的所有 CICS WEB 命令中。
2. 发出 WEB SEND 命令，指定该连接的 SESSTOKEN。 该 WEB SEND 命令从服务器中检索域。
3. 发出 WEB RECEIVE 命令。服务器返回一个状态码。在 WEB RECEIVE 命令中使用 STATUSCODE 选项，以检查 401 响应。
4. 如果状态码是 401（服务器需要认证详细信息），那么重复发出第一个 WEB SEND 请求，但该次添加 AUTHENTICATE(BASICAUTH) 选项。XWBAUTH 全局用户出口由客户机应用程序调用。 第二个 WEB SEND 命令使用从第一个 WEB SEND 命令接收的域和 XWBAUTH 出口，来确定必需的用户名和密码。
5. 您可以在初始 WEB SEND 命令中指定 AUTHENTICATE(BASICAUTH)，而不是等待 401 响应。您可以:
   a. 使用 AUTHENTICATE(BASICAUTH) 选项在 WEB SEND 命令中提供用户名和密码。
   b. 通过指定 AUTHENTICATE(BASICAUTH) 选项来调用 XWBAUTH 全局用户出口，但省略凭证。该用户出口将被调用，但传递给该出口的域将为空，因为尚未从服务器收到该域。用户出口必须从其他参数（如 HOST 和 PATH）中得出必需凭证。
6. 如果应用程序需要了解 401 响应中发送的域，请使用 WEB EXTRACT 命令。

结果

CICS 将用户名和密码凭证通过认证头传递到服务器。

接收 HTTP 响应

WEB RECEIVE 命令或 WEB CONVERSE 命令用于从服务器接收响应。（WEB CONVERSE 命令结合了 WEB SEND 和 WEB RECEIVE 命令的功能。）WEB READ HTTPHEADER 命令或 HTTP 头浏览命令用于检查头。

开始之前

应用程序为了接收响应准备等待的时间由 RTIMOUT 值表明，该值在别名事务的事务概要文件定义中指定。超时限制不应用到对响应头的读取。

当 RTIMOUT 指定的时间段到期后，CICS 向应用程序返回 TIMEDOUT 响应。RTIMOUT 值为零意味着应用程序准备无限地等待下去。事务概要文件定义中 RTIMOUT 的缺省设置为零，因此检查并更改该设置很重要。

关于此任务

使用 WEB RECEIVE 或 WEB CONVERSE 命令:

1. 使用 SESSTOKEN 选项为该连接指定会话令牌。
2. 指定用于接收服务器发送的 HTTP 状态码的数据区，以及为描述状态码而由服务器返回的任何文本。 请注意，数据以其未转义的格式返回。
3. 指定数据区以接收响应主体的介质类型。

4. 通过指定 INTO 选项（对于数据缓冲区）或 SET 选项（对于指针引用）和 LENGTH 选项来接收响应主体。 该数据以其转义格式返回，并转换到适合应用程序的代码页中，除非您另外请求。
5. 如果要限制从响应主体接收的数据量，那么指定 MAXLENGTH 选项。如果要保留（而不是丢弃）超出该长度的任何数据，那么也指定 NOTRUNCATE 选项。 可以使用更多 WEB RECEIVE 命令来获取任何剩余数据。 如果已使用分块的传输编码发送数据，CICS 会将块组装到单条消息中，然后再将该消息传递到应用程序。因此 MAXLENGTH 选项将应用于已分块消息的整个实体主体长度，而不是应用到各个块。
6. 如果响应主体**不**需要代码页转换，那么指定合适的转换选项（根据您正在使用的是 WEB RECEIVE 命令还是 WEB CONVERSE 命令），因此 CICS 不转换响应主体。缺省情况是应该发生转换，且在该情况下 CICS 将服务器的响应主体转换为本地 CICS 区域的缺省代码页，或者在 WEB OPEN 命令中指定的任何备用 EBCDIC 代码页。

   注: 如果您接收已打包或已压缩的实体主体，如消息的 Content-Encoding 头表明的，那么确保禁止了代码页转换。CICS 不对这些消息类型进行解码，并且如果应用代码页转换，结果将不可预测。如果您不希望接收已打包或已压缩的实体主体，那么可以在对服务器的请求中使用 Accept-Encoding 头来指定。

   发出 WEB RECEIVE 或 WEB CONVERSE 命令时，CICS 从状态行返回响应主体和信息。
7. 使用合适的 CICS 命令检查服务器的响应的 HTTP 头:
   - 如果要读取特定 HTTP 头（您知道服务器提供了该头），那么使用 WEB READ HTTPHEADER 命令检查该头的内容。您的应用程序必须提供将接收头内容的缓冲区。如果请求中不存在头，那么 CICS 返回 NOTFND 条件。
   - 如果要浏览响应中的所有 HTTP 头，那么使用 WEB STARTBROWSE HTTPHEADER 命令开始浏览头行。使用 WEB READNEXT HTTPHEADER 命令以获取每一行的头名称和头值。您的应用程序必须提供两个缓冲区: 一个将接收头名，而另一个将接收其内容。读取所有头后，CICS 将返回 ENDFILE 条件。当您的程序已获取所有相关头信息后，使用 WEB ENDBROWSE HTTPHEADER 命令。

   记得在每条 HTTP 头命令中包括会话令牌。
8. 处理服务器的响应并执行应用程序的业务逻辑。 如果响应具有“常规”或参考状态码（例如，200（确定）），那么可以如常处理响应。（发出 WEB RECEIVE 命令时，接收到状态码。）如果响应中包含表明错误或请求更多操作的状态码，那么应该执行备用处理以解决该问题。第 341 页的附录 C, 『CICS Web Support 的 HTTP 状态码参考』提供了有关对状态码做出响应的基本指导。
9. 如果发送了管道化序列的请求，那么使用更多 WEB RECEIVE 命令从服务器接收其余响应。 CICS 保留响应并按 CICS 从服务器接收它们的顺序将它们返回到应用程序。处理管道化的请求的服务器将以接收请求相同顺序提供响应。

   提示: 正在接收对管道化的请求的响应时，如果您正在使用多个 WEB RECEIVE 命令接收过长的消息体，请仔细地记录已发出多少 WEB RECEIVE 命令。您可能发现在单条 WEB RECEIVE 命令中接收这些响应中每一个的整个主体会更方便。

关闭到 HTTP 服务器的连接

当 CICS 是 HTTP 客户机时，CICS 和服务器之间的连接可以由服务器终止，或由 CICS 按应用程序发出的命令进行操作来终止，还可以在任务结束时终止。

开始之前

关于此任务

为了有效管理连接关闭，建议进行以下行为:

1. 在要对服务器发出的最后一次请求上，为 WEB SEND 或 WEB CONVERSE 命令上的 CLOSESTATUS 选项指定 CLOSE。 CICS 写请求的 Connection: close 头，或者对于 HTTP/1.0 级别的服务器，省略 Connection: Keep-Alive 头。在发出最后一次请求时指定该选项是好行为，因为头中的信息意味着服务器在发送最后一条响应后可以立即关闭它与您的连接，而不是等着看您是否在超时前发送更多请求。仍将接收来自服务器的响应，并且该响应对应用程序可用。然而，您将无法使用该连接将任何进一步请求发送到服务器。指定 CLOSESTATUS 选项与发出 WEB CLOSE 命令的效果范围不同。
2. 如果需要测试服务器是否已请求连接终止，那么使用 WEB READ HTTPHEADER 命令在来自服务器的最后一条消息中查找 Connection: close 头。 如果服务器终止连接，那么应用程序无法使用该连接发送更多请求，但是它可以接收服务器在它终止连接前发送的响应。
3. 完成所有 HTTP 请求和响应后，发出 WEB CLOSE 命令，指定会话令牌。 连接关闭时，连接应用的会话令牌不再能使用。需要会话令牌以从服务器接收响应并读取响应的 HTTP 头，因此直到已完成与服务器和它发送的最后一条响应的所有交互后，才应该发出 WEB CLOSE 命令。 如果应用程序不发出 WEB CLOSE 命令，那么在任务结束时关闭连接。

样本程序: 将请求通过管道传送给 HTTP 服务器

样本程序 DFH$WBPA（汇编程序）、DFH$WBPC（C）和 DFH0WBPO （COBOL）演示了 CICS 如何通过管道将客户机请求传送到 HTTP 服务器。

开始之前

样本程序将请求发送到运行 CICS Web Support 的 CICS 区域。管道传送的请求由 CICS 提供的样本程序 DFH$WB1C 处理。在使用样本程序之前，请按照第 49 页的第 4 章，『配置 CICS Web Support 基本组件』中描述的过程，将 CICS 区域设置为 HTTP 服务器。按第 51 页的『验证 CICS Web Support 的操作』所述，通过设置样本程序 DFH$WB1C 和样本 URIMAP 定义 DFH$URI1 来完成该过程。请注意，如果 CICS 区域已设置完毕且正作为 HTTP 服务器运行，并且您拥有自己的正确构建的 TCPIPSERVICE 定义，请勿再次安装样本 TCPIPSERVICE 定义 HTTPNSSL；只需设置 DFH$WB1C 和 DFH$URI1 即可。

关于此任务

将 CICS 区域设置为 HTTP 服务器时，请完成下列步骤以使用管道传送样本程序:

1. 确定将作为 HTTP 客户机的 CICS 区域。 要尝试使用样本程序，您有三个选项:
   - 您可以将同一个 CICS 区域同时用作服务器和客户机；当该区域发出和接收请求时就像是有两个单独的 CICS 区域在处理请求一样，而且得到的结果也是一样

的。在这种情况下，无需进一步的 CICS Web Support 设置，因为作为 HTTP 服务器运行的 CICS 区域也可以作为 HTTP 客户机运行。

- 可以将已经为 CICS Web Support 设置的其他 CICS 区域用作客户机。再次重申，对于这种情况，无需进一步的 CICS Web Support 设置。
- 可以将还没有设置 CICS Web Support 的其他 CICS 区域用作客户机。在这种情况下，您必须执行一些基本的 CICS Web Support 设置，如第 2 步中所述。

2. 可选: 如果使用其他 CICS 区域作为 HTTP 客户机，并且还没有为 CICS Web Support 设置该区域，请执行基本设置，如下所示:
   a. 按照《*CICS Transaction Server for z/OS 安装指南*》中的指示信息，为 CICS 区域启用 TCP/IP 支持。 该过程包括设置 Communications Server 和建立通过 z/OS 对域名服务器（DNS）的访问。
   b. 为区域指定系统初始化参数 **TCPIP=YES**，以激活 CICS TCP/IP 服务。

   该设置使 CICS 区域作为 HTTP 客户机运行。
3. 在设置为 HTTP 服务器的 CICS 区域中，为客户机区域发出其请求时所用的端口确定 TCPIPSERVICE 定义。 选择只使用 HTTP 协议而不使用 SSL（即，通过已指定 PROTOCOL(HTTP) 和 SSL(NO) 的 TCPIPSERVICE 定义）定义的任何端口。您可以选择任何适合的端口，因为服务器中用于访问样本程序 DFH$WB1C 的样本 URIMAP 定义 DFH$URI1 与任何主机名和端口号匹配。
4. 在 HTTP 客户机区域中，修改提供的样本 URIMAP 定义 DFH$URI2，这是在 DFH$WEB 资源定义组中提供的。 由于 DFH$WEB 是受保护的组，所以请将该定义复制到另一个组中以进行编辑。 DFH$URI2 是一个使用属性为 CLIENT 的 URIMAP 定义。它指定 URL 的某些组成部分，样本程序使用这些部分向 HTTP 服务器区域发出请求。要修改该样本，请执行下列步骤:
   a. 在 DFH$URI2 中指定的方案（SCHEME 属性）是 HTTP。请勿更改该方案。
   b. DFH$URI2 中指定的主机（HOST 属性）是虚主机名。修改该名称以插入实际的主机名，如下所示:
      - 为 HTTP 服务器区域指定分配给 z/OS 映像的主机名。如果您不知道主机名，那么可以使用在第 3 步中选择的 TCPIPSERVICE 定义中的冒号十六进制或点分十进制 IP 地址。
      - 如果您选择的 TCPIPSERVICE 定义用于 80（这是 HTTP 的常用端口号）以外的其他端口号，请在 TCPIPSERVICE 定义中在主机名之后指定端口号，并用冒号将主机名和端口号隔开。
   c. DFH$URI2 中指定的路径（PATH 属性）是 `/sample_web_app`，这与 DFH$URI1 中的路径匹配。请勿更改该路径。
5. 在 HTTP 客户机区域中，安装您所修改的 URIMAP 定义 DFH$URI2。
6. 在 HTTP 客户机区域中，安装 PROFILE 定义 DFH$WBPF，这是在 DFH$WEB 资源定义组中提供的。
7. 用您所需的语言翻译并编译下列某个样本程序。 在接收用管道传送的样本程序时，不会对其进行编译。SDFHSAMP 库提供了样本程序。样本程序及其相应事务的名称如下所示:

语言	程序	事务
汇编程序	DFH$WBPA	WBPA
C	DFH$WBPC	WBPC

语言	程序	事务
COBOL	DFH0WBPO	WBPO

8. 在 HTTP 客户机区域中，为所选的样本程序安装 PROGRAM 资源定义及对应的 TRANSACTION 资源定义。 资源定义是在 DFH$WEB 资源定义组中提供的。
9. 在 HTTP 客户机区域中，为您所选的样本程序运行事务。 当样本程序成功发送所有这三个 HTTP 请求并成功接收所有这三个 HTTP 响应之后，它会将参考消息发送到终端。如果任何消息发送或接收失败，那么会提供错误消息。HTTP 请求和响应的内容也不会显示出来。
10. 如果已使用完样本程序，出于安全性考虑，请禁用 URIMAP 定义 DFH$URI1 和 DFH$URI2，卸载样本 TCPIPSERVICE 定义 HTTPNSSL（如果正在使用的话）。

相关概念

第 18 页的『管道传送』
管道传送是指客户机将多个 HTTP 请求发送到服务器而不等待响应。随后，响应必须以接收请求的相同顺序从服务器返回。

第 35 页的『CICS Web Support 如何处理管道传送』
管道化的请求序列可以由 CICS 发送和接收。作为 HTTP 服务器的 CICS 可以接收 Web 客户机通过管道发出的请求序列，而作为 HTTP 客户机的 CICS 可以通过管道将请求序列发送至服务器。

相关任务

第 51 页的『验证 CICS Web Support 的操作』
样本程序 DFH$WB1A（汇编程序）和 DFH$WB1C（C）可帮助您测试 CICS Web Support 是否运行正常。样本程序使用 EXEC CICS WEB 和 DOCUMENT 命令来接收您的请求，并且构建和发送一个简单的响应。

样本程序: 以分块方式发送和接收 HTTP 请求

样本程序 DFH$WBCA（汇编程序）、DFH$WBCC（C）和 DFH0WBCO（COBOL）演示了作为 HTTP 客户机的 CICS 如何以分段或分块的方式将请求发送到 HTTP 服务器，并接收分块形式的响应消息。样本程序 DFH$WBHA（汇编程序）、DFH$WBHC（C）和 DFH0WBHO（COBOL）演示了作为 HTTP 服务器的 CICS 如何以分块方式从 HTTP 客户机接收请求并发送分块形式的响应。

开始之前

样本程序在运行 CICS Web Support 的 CICS 区域之间发送和接收请求。客户机分块样本 DFH$WBCA、DFH$WBCC 和 DFH0WBCO 安装在 HTTP 客户机区域中，而服务器分块样本 DFH$WBHA、DFH$WBHC 和 DFH0WBHO 安装在 HTTP 服务器区域中。例如，客户机样本 DFH$WBCA 会打开与其对应的服务器样本 DFH$WBHA 的会话。DFH$WBHA 从 DFH$WBCA 接收分块请求，然后发送分块响应。客户机样本 DFH$WBCA 将响应作为分块消息接收。

在使用样本程序之前，必须按照第 49 页的第 4 章，『配置 CICS Web Support 基本组件』中描述的过程，将 CICS 区域设置为 HTTP 服务器。如果 CICS 区域已设置完毕且正作为 HTTP 服务器运行，并且您拥有自己的正确构建的 TCPIPSERVICE 定义，请不要再次安装样本 TCPIPSERVICE 定义 HTTPNSSL。

关于此任务

将 CICS 区域设置为 HTTP 服务器时，请完成下列步骤以使用分块样本程序:

1. 确定将作为 HTTP 客户机的 CICS 区域。 要尝试使用样本程序，您有三个选项:
   - 您可以将同一个 CICS 区域同时用作服务器和客户机；就象是从两个单独的 CICS 区域发出和接收请求，而结果也是一样。在这种情况下，无需进一步的 CICS Web Support 设置，因为作为 HTTP 服务器运行的 CICS 区域也可以作为 HTTP 客户机运行。
   - 可以将已经为 CICS Web Support 设置的其他 CICS 区域用作客户机。再次重申，对于这种情况，无需进一步的 CICS Web Support 设置。
   - 可以将还没有设置 CICS Web Support 的其他 CICS 区域用作客户机。在这种情况下，您需要执行一些基本的 CICS Web Support 设置，如第 2 步中所述。
2. 可选: 如果使用其他 CICS 区域作为 HTTP 客户机，并且还没有为 CICS Web Support 设置该区域，请如下执行基本设置:
   a. 按照《*CICS Transaction Server for z/OS 安装指南*》中的指示信息，为CICS 区域启用 TCP/IP 支持。 该过程包括设置 Communications Server 和建立通过 z/OS 对域名服务器（DNS）的访问。
   b. 为区域指定系统初始化参数 `TCPIP=YES`，以激活 CICS TCP/IP 服务。

   该设置使 CICS 区域作为 HTTP 客户机运行。
3. 在设置为 HTTP 服务器的 CICS 区域中，为客户机区域用来生成其请求的端口确定 TCPIPSERVICE 定义。 选择用 HTTP 协议（但不使用 SSL）定义的任何端口，因此将 TCPIPSERVICE 定义指定为 PROTOCOL(HTTP)，将 SSL 指定为 NO。您可以选择任何适合的端口，因为服务器中用于访问服务器分块样本程序的样本 URIMAP 定义 DFH$URI4 与任何主机名和端口号匹配。
4. 在 HTTP 客户机区域中，修改提供的样本 URIMAP 定义 DFH$URI3，这是在 DFH$WEB 资源定义组中提供的。因为 DFH$WEB 是受保护的组，所以您需要将定义复制到另一个组以启用编辑。 DFH$URI3 是一个使用属性为 CLIENT 的 URIMAP 定义。它指定 URL 的某些组成部分，样本程序使用这些部分向 HTTP 服务器区域发出请求。
   a. 在 DFH$URI3 中指定的方案（SCHEME 属性）是 HTTP。您不必对其进行更改。
   b. DFH$URI3 指定虚主机名（HOST 属性）。可修改该属性以插入实际的主机名，如下所示:
      - 为 HTTP 服务器区域指定分配给 z/OS 映像的主机名。如果您不知道主机名，那么可以使用在第 3 步中选择的 TCPIPSERVICE 定义中的 IP 地址。
      - 如果您选择的 TCPIPSERVICE 定义用于 80（这是 HTTP 的常用端口号）以外的端口号，那么需要在主机名之后指定 TCPIPSERVICE 定义的端口号，并用冒号分开主机名和端口号。
   c. DFH$URI3 中指定的路径（PATH 属性）是 `/chunking_sample_application`，这与 DFH$URI4 中的路径匹配。您不必对其进行更改。
5. 在 HTTP 客户机区域中，安装您所修改的 URIMAP 定义 DFH$URI3。
6. 在 HTTP 客户机区域中，安装 PROFILE 定义 DFH$WBPF，这是在 DFH$WEB 资源定义组中提供的。

7. 用您希望使用的语言翻译并编译客户机和服务器样本程序。 在接收分块样本程序时，不会对其进行编译。SDFHSAMP 库提供了样本程序。样本程序及其对应事务的名称是:

样本类型	语言	程序	事务
客户机分块样本	汇编程序	DFH$WBCA	WBCA
客户机分块样本	C	DFH$WBCC	WBCC
客户机分块样本	COBOL	DFH0WBCO	WBCO
服务器分块样本	汇编程序	DFH$WBHA	-
服务器分块样本	C	DFH$WBHC	-
服务器分块样本	COBOL	DFH0WBHO	-

8. 在 HTTP 服务器区域中，为您选择的服务器分块样本程序安装 PROGRAM 资源定义，并安装提供的样本 URIMAP 定义 DFH$URI4。 资源定义是在 DFH$WEB 资源定义组中提供的。
   a. 如果选择 C 或 COBOL 样本而不是汇编程序样本，那么在安装所提供的样本 URIMAP 定义 DFH$URI4 之前先对它进行修改。 因为 DFH$WEB 是受保护的组，所以您需要将定义复制到另一个组以启用编辑。
   b. 将 DFH$URI4 指定的程序（PROGRAM 属性）从 DFH$WBHA（汇编程序分块样本程序）更改成您首选的服务器分块样本程序。
   c. 安装您所修改的 URIMAP 定义 DFH$URI4。
9. 在 HTTP 客户机区域中，为所选的客户机分块样本程序安装 PROGRAM 资源定义及对应的 TRANSACTION 资源定义。 资源定义是在 DFH$WEB 资源定义组中提供的。
10. 在 HTTP 客户机区域中，为您所选的客户机分块样本运行事务。 当样本程序成功地将所有这四块消息和两个头的尾部发送到 HTTP 服务器时，它就会将参考消息输出到终端。同时还会显示一条消息，确认已接收消息。如果任何一个块发送失败，那么会提供错误消息。实际 HTTP 请求和响应的内容则不会显示出来。
11. 在 HTTP 服务器区域中，您选择的服务器分块样本将由对应的客户机分块样本调用。 当样本程序成功地将所有这四块消息和两个头的尾部发送到正在等待的 HTTP 客户机时，它会将参考消息输出到终端。如果任何一个块发送失败，那么会提供错误消息。实际 HTTP 请求和响应的内容则不会显示出来。
12. 当您完成样本程序的使用后，出于安全性考虑，应该禁用 URIMAP 定义 DFH$URI3 和 DFH$URI4，卸载样本 TCPIPSERVICE 定义 HTTPNSSL（如果正在使用的话）。

通过作为 HTTP 客户机的 CICS 创建 HTTP 请求的 URIMAP 定义

您可以创建 URIMAP 定义，它指定 HTTP 客户机请求的 URI 的组成部分（方案、主机和路径）以及用于请求的 SSL 客户机证书（如果需要）。可以在 WEB OPEN 命令中指定 URIMAP 定义，来为连接提供方案和主机名以及缺省路径。还可以在 WEB SEND 命令中命名它，来为相关请求提供路径。或者，您可使用 WEB EXTRACT URIMAP 命令从 URIMAP 定义抽取信息并直接在发出 HTTP 客户机请求的应用程序中使用。

1. 识别您计划用于 HTTP 客户机请求的 URL。 该 URL 表示您计划在服务器上访问的资源。

2. 识别客户机证书对于请求是否是必需的，并获取合适的证书标签。 如果用于请求的方案是 HTTPS（带 SSL 的 HTTP），那么服务器可能请求 SSL 客户机证书。如果发生这种情况，那么 CICS 提供 URIMAP 定义中指定的证书标签。
3. 将请求的 URL 划分成它的方案、主机和路径部分。 第 10 页的『URL 的组成部分』说明了这些每一个组成部分以及如何对它们定界。如果 URL 中已显式地指定端口号，那么还应使用端口号。 例如，在 URL `http://www.example.com:1030/software/index.html` 中:
   - 方案部分是 `http`
   - 主机部分是 `www.example.com`
   - 端口号是 `1030`
   - 路径部分是 `/software/index.html`

   跟随方案部分的定界符 `://`（冒号和两个正斜杠）不用于 URIMAP 定义。路径部分开头的定界符 `/`（正斜杠）可以包含在 PATH 规范中，也可省略；如果您省略它，那么 CICS 会自动提供该符号。如果您要在请求的 URL 中提供查询字符串，那么可以使用 QUERY 选项在 WEB SEND 命令中指定它。
4. 使 URIMAP 定义以您选择的名称和组开头。
5. 使用 STATUS 属性指定应在启用状态还是在禁用状态安装 URIMAP 定义。
6. 指定 CLIENT（作为 HTTP 客户机的 CICS）的 USAGE 属性。
7. 指定 SCHEME 属性作为请求的 URL 的方案部分。可使用 HTTP（不带 SSL）或 HTTPS（带 SSL）。 不要包含跟随方案部分的定界符 `://`。
8. 指定 HOST 属性作为请求的 URL 的主机部分。主机部分可以是显式的 IPv4 或 IPv6 地址或字符主机名。
9. 指定 PATH 属性作为请求的 URL 的路径部分。 **不能**在作为 HTTP 客户机的 CICS 的 URIMAP 定义中使用通配符（星号）。 您可以在路径部分开始包含或省略定界符 `/`（正斜杠）；如果您省略它，CICS 会自动提供它。 如果在 WEB OPEN 命令中引用 URIMAP 定义，那么该路径成为该连接的 WEB SEND 命令的缺省路径。如果在 WEB SEND 命令中引用了 URIMAP 定义，那么该路径将用于该 WEB SEND 命令，但是请注意，该 URIMAP 定义的主机属性必须与 WEB OPEN 命令中为连接指定的主机相匹配。 不要在路径部分包含查询字符串；您可以使用 QUERY 选项在 WEB SEND 命令中指定它。
10. 可选: 如果使用 SSL，那么指定 CERTIFICATE 属性作为要用作该请求的 SSL 客户机证书的证书标签。
11. 可选: 如果使用 SSL 或 TLS，那么指定 CIPHERS 属性作为要用于该请求的密码。

HTTP 客户机发送出口 XWBAUTH

使用 XWBAUTH，您可以为目标服务器或服务供应商指定基本认证凭证（用户名和密码）。XWBAUTH 会根据请求将这些凭证传递给 CICS，以创建使用 HTTP 转发的 Authorization 头。在处理 EXEC CICS WEB SEND（客户机）或 EXEC CICS WEB CONVERSE 命令期间会调用 XWBAUTH。可将主机名和路径信息传递到具有可选限定域的用户出口。

当在 EXEC CICS WEB SEND（客户机）或 WEB CONVERSE 命令中指定 AUTHENTICATE(BASICAUTH) 时，应用程序可提供用户名和密码。如果未提供这些信息，那么将调用 XWBAUTH，这样会以另一种方法来指定这些凭证。

通常，用户名和密码特定于远程服务器环境，其长度可能超过 RACF 系统使用的 8 个字符的标准长度。用户名和密码字段最多可以有 256 个字符。不会验证这些字段的语法。

主机作为 UEPHOST 参数传递到用户出口程序，而路径作为 UEPPATH 参数传递。可以选择将域作为 UEPREALM 参数传递。在响应中，用户出口程序将用户名和密码作为 UEPUSNM 和 UEPPSWD 参数返回。返回码 UERCNORM 表示成功返回的用户名和密码。返回码 UERCBYP 表示无法识别用户名和密码，因此不会将 Authorization 头添加到请求中。返回码 UERCERR 表示出口无法提供凭证，而且正在请求的 CICS 命令会失败。

CICS 样本库 SDFHSAMP 中提供了下列样本出口程序:

- DFH$WBPI
- DFH$WBEX
- DFH$WBX1
- DFH$WBX2
- DFH$WBGA，映射 DFH$WBPI、DFH$WBX1、DFH$WBX2 和 DFH$WBEX 样本使用的全局工作区的副本。

有关客户机样本出口程序的更多信息，请参阅 *CICS Customization Guide*。有关设置 LDAP 概要文件的更多信息，请参阅 *CICS RACF Security Guide*。

出口 XWBAUTH

调用时机

当 EXEC CICS WEB SEND 或 WEB CONVERSE 命令指定 AUTHENTICATE (BASICAUTH)，但未指定 USERNAME 和 PASSWORD 时。

特定于出口的参数

UEPHOST（CICS 提供的输入）
字段的地址，该字段包含在用于连接的 WEB OPEN 命令的 HOST 选项中指定的主机名地址、IPv4 或 IPv6 地址。当主机名保存在该字段中时，它将转换为小写字符。在匹配主机名时，您的用户出口程序必须执行该转换。

UEPHOSTL（CICS 提供的输入）
字段的地址，该字段包含主机名的半字长度。

UEPPATH（CICS 提供的输入）
字段的地址，该字段包含 WEB SEND 或 WEB CONVERSE 命令的 PATH 选项中指定的路径的地址。该路径允许采用大小写混合形式，即保留输入时的大小写。

UEPPATHL（CICS 提供的输入）
字段的地址，包含路径的半字长度。

UEPREALM（CICS 提供的输入）
字段的地址，该字段包含与目标关联的域名的地址（如果服务器在上一个 HTTP 401 响应中返回了域名）。

UEPREALML（CICS 提供的输入）
字段的地址，该字段包含域名的半字长度。

UEPAUTHT（CICS 提供的输入）
表示认证类型的 1 字节代码的地址。这是二进制的 01，表示基本认证。

UEPUSNM（用户出口提供的输出）
全字字段的地址，包含访问 HTTP 服务器所需的用户名的地址。CICS 创建预定义的地址和 64 字节区域，用于存储用户名。您可以将用户名放在该 64 字节区域中，而不更改 UEPUSNM 中的地址。或者，您可以将用户名放在自己的区域中，用您的用户名地址替换 UEPUSNM 中的地址。如果您创建自己的用户名区域，那么字段长度最大为 256 字节。

UEPUSNML（CICS 提供的输入以及用户出口提供的输出）
半字字段的地址，它最初包含 UEPUSNM 中提供的缓冲区地址的长度。用户出口程序必须将该缓冲区的长度设置为 UEPUSNM 中提供的实际用户名长度。

UEPPSWD（用户出口提供的输出）
全字字段的地址，包含访问 HTTP 服务器所需的密码的地址。CICS 创建预定义的地址和 64 字节区域，用于存储密码。您可以将密码放在该 64 字节区域中，而不更改 UEPPSWD 中的地址。或者，您可以将密码放在自己的区域中，用您的密码地址替换 UEPPSWD 中的地址。如果您创建自己的密码区域，那么字段长度最大为 256 字节。

UEPPSWDL（CICS 提供的输入以及用户出口提供的输出）
半字字段的地址，它最初包含 UEPPSWD 中提供的缓冲区地址的长度。用户出口程序必须将该缓冲区的长度设置为 UEPPSWD 中提供的实际用户名长度。

UEPHOSTT（CICS 提供的输入）
表示 UEPHOST 参数中包含的主机类型的 1 字节代码的地址。

二进制 01 表示主机名，二进制 02 表示 IPv4 地址，而二进制 03 表示 IPv6 地址。

返回码

UERCNORM
出口已经成功地返回了用户名和密码。

UERCBYP
出口无法识别用户名和密码。未发送 Authorization 头。

UERCERR
出口无法识别用户名和密码。必须终止 WEB SEND（客户机）或 WEB CONVERSE 命令。

XPI 调用
可以使用所有 XPI 调用。

API 和 SPI 命令

可使用除 EXEC CICS SHUTDOWN 和 EXEC CICS XCTL 以外的 API 和 SPI 命令。

XWBAUTH 使用 LDAP XPI 函数的典型情况

期望使用的 DFHDDAPX 函数（与 XWBAUTH 全局用户出口关联）包括打开和关闭 LDAP 会话、浏览凭证结果、扫描并定位结果、关闭浏览，返回正确值以及关闭搜索。

BIND_LDAP

建立与 LDAP 服务器的会话。只在首次调用全局用户出口 XWBAUTH 时使用一次。LDAP 会话令牌存储在 XWBAUTH 的全局工作区（如果提供的话）中，供随后调用 LDAP_SEARCH 时使用。

UNBIND_LDAP

释放与 LDAP 服务器的连接。仅在 CICS 关闭处理时，才需要使用该函数。可以在执行 XSTERM（系统终止）全局用户出口时使用该函数。

SEARCH_LDAP

通过指定用于标识所需用户信息的 URL 和域的 LDAP 专有名称，来搜索凭证。用下列格式指定专有名称:

```
racfcid=uuuuuuuu, ibm-httprealm=rrrrrrrr, labeledURI=xxxxxxxx, cn=BasicAuth
```

其中:

- uuuuuuuu 是当前的用户标识，这是从 XWBAUTH 参数 UEPUSER 获得的。
- rrrrrrrr 是 HTTP 401 域，这是从 XWBAUTH 参数 UEPREALM（如果存在的话）获得的。
- xxxxxxxx 是目标 URL，这是通过将“http://”、XWBAUTH 参数中的主机名“UEPHOST”以及 XWBAUTH 参数中的路径“UEPPATH”连接起来获得的。
- cn=BasicAuth 是任意后缀，该后缀在 LDAP 服务器中配置，用于存储基本的认证凭证。

START_BROWSE_RESULTS

开始扫描由 SEARCH_LDAP 返回的结果。

GET_NEXT_ENTRY

定位 SEARCH_LDAP 返回的一系列项中的下一个结果项。通常使用 SEARCH_LDAP 中指定的 URL 定位唯一项，而不使用 GET_NEXT_ENTRY 函数。

GET_NEXT_ATTRIBUTE

定位当前结果项中的下一个属性。通常，将选择特定属性，而不使用 GET_NEXT_ATTRIBUTE 函数。

END_BROWSE_RESULTS

终止 SEARCH_LDAP 启动的浏览会话。

GET_ATTRIBUTE_VALUE

返回目标专有名称的各种属性值。对于 XWBAUTH，这些属性值是存储在属性 uid 和 userpassword 中的用户名和密码。XWBAUTH 将这些属性值作为凭证返回。

FREE_SEARCH_RESULTS

关闭 SEARCH_LDAP 启动的搜索，释放关联的存储器。

HTTP 客户机开放出口 XWBOPEN

XWBOPEN 使系统管理员能指定代理服务器，而这些代理服务器应该用于作为 HTTP 客户机的 CICS 的 HTTP 请求，并使系统管理员能将安全策略应用到为那些请求指定的主机名。

处理 EXEC CICS WEB OPEN 命令期间会调用 XWBOPEN，应用程序使用该命令打开与服务器的连接。处理 EXEC CICS INVOKE SERVICE 命令期间也会调用 XWBOPEN。

CICS 本身没有任何有关为作为 HTTP 客户机的 CICS 发出的 HTTP 请求使用（或不使用）代理服务器的要求，并且 CICS 也不会为那些请求应用任何安全策略。如果您的系统或组织需要这些设施，那么由您负责进行设置。

EXEC CICS WEB OPEN 命令指示 CICS Web 域打开与服务器的连接。在打开连接之前调用 XWBOPEN。EXEC CICS WEB OPEN 命令的 HOST 选项指定的连接的主机名（例如，`www.example.com`），作为 UEPHOST 参数传递给用户出口程序以供检查。此时，用户出口程序可用于两个目的:

- **确定 HTTP 请求是否需要使用代理服务器，并返回所需的任何代理服务器的名称。**如果需要代理服务器，那么使用返回码 UERCPROX，并将代理服务器的名称返回到 CICS Web 域（在 UEPPROXY 标识的缓冲区中）并使用它连接到服务器。如果不需要代理服务器，那么使用返回码 UERCNORM。
- **对主机名应用安全策略。**返回码 UERCBARR 表明不允许对主机的访问。如果不允许访问主机，那么一个 NOTAUTH 响应将返回给 WEB OPEN 命令，而应用程序员应放弃尝试打开该连接。如果您要为个别资源以及主机资源（或不是主机资源）应用安全策略，那么可以在 EXEC CICS WEB SEND 和 EXEC CICS WEB CONVERSE 命令中使用 XWBSNDO 用户出口将安全策略应用到 URL 的路径部分。

XWBOPEN 用户出口不支持使用 EXEC CICS 命令。

样本程序 DFH$WBPI 和 DFH$WBEX 以及关联的副本 DFH$WBGA 向您显示如何在全局工作区中设置代理服务器信息或安全策略。例如，如果从您的 CICS 系统发出的所有请求都应该使用单个代理服务器，那么可以将代理服务器名称指定为初始化参数。如果您使用很多代理服务器或者要为不同的主机名应用安全策略，那么可以装入或者构建一个将主机名与相应的代理服务器相匹配的，或者将它们标记为禁止的表，然后，在处理 EXEC CICS WEB OPEN 命令期间该表可用作查找表。可以在初始化后的程序列表（PLTPI）处理期间或期望使用 EXEC CICS WEB OPEN 命令之前的任何时候运行样本程序。

出口 XWBOPEN

调用时机

处理 EXEC CICS WEB OPEN 或 EXEC CICS INVOKE SERVICE 命令期间。

特定于出口的参数

UEPHOST（CICS 提供的输入）

字段的地址，该字段包含在 WEB OPEN 命令的 HOST 选项中指定的主机名、IPv4 或 IPv6 地址。

注: 当主机名保存在该字段中时，它将转换为小写字母。在匹配主机名时，必须为您的用户出口程序考虑到这一点。

UEPHOSTL（CICS 提供的输入）
字段的地址，该字段包含主机名的半字长度。

UEPPROXY（用户出口提供的输出）
字段的地址，该字段包含指向代理服务器名称的地址。代理服务器的名称必须采用 URL 格式。 在用户出口程序的输入中，该参数设置为字段的地址，该字段包含 2046 个字节区域的地址。您可以在该区域中放置代理服务器名，并且不更改 UEPPROXY 中的地址。或者，您可以在自己的区域中放置代理服务器名，并用包含您自己区域的地址的字段地址替换 UEPPROXY 中的地址。

UEPPROXYL（用户出口提供的输出）
字段的地址，该字段包含代理服务器名的半字长度。

UEPHOSTT（CICS 提供的输入）
表示 UEPHOST 参数中包含的主机类型的 1 字节代码的地址。

注: 二进制 01 表示主机名，二进制 02 表示 IPv4 地址，而二进制 03 表示 IPv6 地址。

返回码

UERCNORM
对于该 HTTP 请求，不需要代理服务器，但不禁止主机名。

UERCPROX
对于该 HTTP 请求，需要代理服务器。已将 UEPPROXY 设置为所需代理服务器的名称，并且已将 UEPPROXYL 设置为代理服务器名的长度。

UERCBARR
服务器的主机名被禁止。

UERCERR
处理出口时发生错误。

XPI 调用
可以使用所有 XPI 调用。

API 和 SPI 命令
不能使用 EXEC CICS 命令。

HTTP 客户机发送出口 XWBSNDO

XWBSNDO 使系统管理员能为作为 HTTP 客户机的 CICS 发出的 HTTP 请求指定安全策略。处理 EXEC CICS WEB SEND 或 EXEC CICS WEB CONVERSE 命令期间调用 XWBSNDO。主机名和路径信息将传递到出口，并且可将安全策略应用于这两个组件之一或全部。

CICS 自己不会为作为 HTTP 客户机的 CICS 发出的 HTTP 请求应用任何安全策略；因此如果您的系统或组织需要它，您就要负责设置该设施。

WEB OPEN 命令中的 XWBOPEN 出口可用于阻止对整个主机的访问，可将 XWBSNDO 出口用于执行同样的的操作或阻止对主机中特定路径的访问。如果您希望阻止对整个

主机的访问，使用 XWBOPEN 出口会节省时间，这是因为应用程序无法打开连接，因此在创建应发送的请求时不会浪费时间。为 XWBSNDO 出口提供的主机名主要是为了帮助您区分不同主机使用的相同路径。

如果针对 HTTP 请求使用分块的传输编码，那么 XWBSNDO 仅在分块的消息的第一个 WEB SEND 命令中调用。

XWBSNDO 用户出口不支持使用 EXEC CICS 命令。

主机作为 UEPHOST 参数传递到用户出口程序，而路径作为 UEPPATH 参数传递。返回码 UERCNORM 表明路径是允许的，而返回码 UERCBARR 表明该路径是不允许的。如果不允许该路径，那么一个 NOTAUTH 响应将返回给 WEB SEND 或 WEB CONVERSE 命令，而应用程序员应通过 WEB CLOSE 命令来关闭连接以处理这种情况。

出口 XWBSNDO

调用时机

为作为 HTTP 客户机的 CICS 的 HTTP 请求处理 EXEC CICS WEB SEND 或 EXEC CICS WEB CONVERSE 命令期间。在 WEB SEND 命令中使用 SESSTOKEN 参数来表示客户机请求。

特定于出口的参数

UEPHOST
字段的地址，该字段包含在用于连接的 WEB OPEN 命令的 HOST 选项中指定的主机名、IPv4 或 IPv6 地址。

注: 当主机名保存在该字段中时，它将转换为小写字母。在匹配主机名时，必须为您的用户出口程序考虑到这一点。

UEPHOSTL
字段的地址，该字段包含主机名的半字长度。

UEPPATH
字段的地址，该字段包含 WEB SEND 命令的 PATH 选项中指定的路径。该路径是大小写混合的，如同指定它的方式一样。

UEPPATHL
字段的地址，包含路径的半字长度。

UEPHOSTT
表示 UEPHOST 参数中包含的主机类型的 1 字节代码的地址。

注: 二进制 01 表示主机名，二进制 02 表示 IPv4 地址，而二进制 03 表示 IPv6 地址。

返回码

UERCNORM
允许路径。

UERCBARR
不允许路径。

XPI 调用

可以使用所有 XPI 调用。

API 和 SPI 命令

不能使用 EXEC CICS 命令。

第 13 章 CICS Web Support 的安全性

当 CICS 连接到因特网后，为了阻止对 CICS 应用程序和数据的未授权访问，也为了阻止获取通过因特网发送的私有信息的第三方，安全措施是必需的。

您应该在 CICS Web Support 体系结构的整个开发过程中考虑安全性，它应该作为 CICS Web Support 应用程序和实用程序设计的一部分，为相关 CICS 设施创建资源定义时也是如此。本部分总结了可用于增强 CICS Web Support 实施的安全性的措施。

HTTP 客户机的认证和身份识别

客户机的认证和身份识别使服务器能保护它的资源不让未授权的用户访问。

作为 HTTP 服务器的 CICS: 认证和标识

对于作为 HTTP 服务器的 CICS，认证方案由 TCPIPSERVICE 定义的 AUTHENTICATE 属性指定。标识可以从认证过程中获取，或在不需要认证的情况下由 CICS 提供。

从 Web 客户机获取认证和标识是保护您的 CICS 系统不被未授权的用户访问的一个关键步骤。

TCPIPSERVICE 资源定义是您为充当 HTTP 服务器的 CICS 指定适用安全措施的主要位置。您需要为用于 CICS Web Support 的每个端口指定 TCPIPSERVICE 资源定义。TCPIPSERVICE 资源定义规定:

- SSL 是否用于端口。
- 用于端口的认证方案。
- 基本认证域。

认证

CICS 支持两种认证方案，以与 HTTP 协议协同使用:

- **基本认证**是一种 HTTP 设施，它允许客户机通过提供用户标识和密码将其本身认证和标识到服务器。此信息是使用基本 64 位编码来编码的，解码也比较简单。因为这个原因，所以仅当不可能拦截密码时，才适合使用基本认证作为唯一的认证方式。在大多数环境中，基本认证应该与 SSL 结合使用，以便 SSL 加密用于保护用户标识和密码信息。第 18 页的『HTTP 基本认证』更详细地说明了基本认证。
- 使用可信第三方（或认证中心）发出的客户机证书，并使用 SSL 加密发送，**SSL 客户机证书认证**是认证客户机的更安全方法。*CICS RACF Security Guide* 说明了它如何工作。客户机证书不含可用作 CICS 标识的用户标识。要获得标识，客户机证书可以在使用前与 RACF 或同等安全管理器中的用户标识关联，或者当客户机发出请求时自动与其关联（使用基本认证）。每次使用证书时，RACF 用户标识就会成为客户机的用户标识。*CICS RACF Security Guide* 阐述了如何设置它。

第 83 页的『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』介绍如何为 CICS Web Support 设置 TCPIPSERVICE 定义，该定义指定这些认证方案中的某一个方案。

使用基本认证或客户机证书认证时，CICS 处理这一过程：用户请求认证、对认证信息进行解码（必要的话）、依照安全管理器的数据库检查所提供的认证，以及在认证不可接受的情况下拒绝请求。只有在验证和接受了认证后才会调用分析器程序或用户编写的应用程序。

Web 客户机使用的所有用户标识都必须在 RACF 或等效的外部安全性管理器中具有用户概要文件。*CICS RACF Security Guide* 具有更多信息。

对于基本认证，如果发现用户提供的密码已到期，那么 CICS 会提示用户提供新密码，并帮助他们重新提交其请求。CICS 提供的实用程序 DFHWBPW 用于执行此操作。您可以定制 CICS 在此过程期间显示给用户的 Web 页面上的文本。第 153 页的『HTTP 基本认证的密码到期管理』介绍了执行此操作所需的信息。

对于客户机证书认证，CICS 通过依照安全管理器的数据库并依照您已设置的任何证书撤回列表（可选）检查所提供的证书来验证它。用户编写的应用程序可以检查此过程获取的信息，如果这可用于确定如何处理请求。使用 EXTRACT CERTIFICATE 命令以检索：

- 发布者或主题的专有名称的组成部分。*CICS RACF Security Guide* 说明了专有名称。
- 与证书关联的 RACF 用户标识。

标识

标识发生在获取 Web 客户机的用户标识时。在以下情形下，标识可以从 Web 客户机获取：

- 在基本认证期间。
- 通过用户标识与客户机证书的关联。

仅对于应用程序生成的响应，可以由 CICS 代表 Web 客户机提供用户标识：

- 在分析器程序中，该程序在处理应用程序生成的响应时使用。（这可以覆盖为 Web 客户机获取的用户标识。）
- 在请求的 URIMAP 定义中。（这无法覆盖为 Web 客户机获取的用户标识。）
- 作为 CICS 缺省用户标识（如果无法确定其他用户标识）。

要注意如果您代表 Web 客户机提供了用户标识，那么不存在客户机标识的认证。仅当与您自己的客户机系统通信时才应执行该操作，前提是该系统已经认证了其用户，并在安全的环境中与服务器通信。*CICS RACF Security Guide* 更为详细地阐述如何根据 TCPIPSERVICE 定义的设置来确定用户标识。

标识了客户机后，可以使用 RACF 或等效的外部安全性管理器，授予客户机的用户标识对 CICS 资源的访问权（与其他任何用户标识一样）。您可以选择将资源级安全性应用于 Web 客户机可以在 CICS 中访问的任何或所有单独资源，例如作为 CICS 文档模板或 z/OS UNIX 文件存储的 Web 页面，或者提供响应的应用程序所用的 CICS 命令。第 155 页的『CICS Web Support 的 CICS 系统和资源安全性』说明了如何保护这些资源，以及如何在不需要资源级安全性时将其除去。

作为 HTTP 客户机的 CICS: 认证和标识

通过 CICS 发出 HTTP 客户机请求时，服务器或代理可能要求您执行基本认证、代理认证或 SSL 客户机证书认证。可通过使用 WEB SEND 或 WEB CONVERSE 命令的 AUTHENTICATE 选项来进行基本认证。代理认证由用户应用程序执行。使用 URIMAP 定义可以提供客户机证书。

可能要求您的客户机应用程序按以下方式认证它本身:

- **基本认证**允许您提供用户名和密码以访问特定信息。对服务器发出请求时，服务器可能向您发送具有 401 状态码和 WWW-Authenticate 头的响应。头命名需要基本认证的域。要接收您请求的信息，需要提供用户名和密码，然后 CICS 重新发送带有 Authorization 头的请求，指定访问该域所需的凭证（用户名和密码）。CICS 也可以将 Authorization 头直接发送到有需要的服务器，这会排除对 401 响应的需求。CICS 将用户名和密码转换为 ASCII，并应用 base-64 编码，这是基本认证协议所必需的。这允许您通过 WEB SEND 或 WEB CONVERSE 命令，或通过 XWBAUTH 用户出口，以常规字符提供凭证。第 135 页的『提供基本认证的凭证』介绍了在应用程序中设置基本认证所需的步骤。第 18 页的『HTTP 基本认证』介绍了有关基本认证的更多信息。
- **代理认证**由代理服务器启动。对于代理认证，响应的状态码是 407，来自代理服务器的质询头是 Proxy-Authenticate，而响应头必须是 Proxy-Authorization。CICS 不支持该协议。
- **SSL 客户机证书认证**使用可信第三方（或认证中心）发出的客户机证书。当您发出 HTTPS 请求时，服务器可能要求，也可能不要求您提供此认证。*CICS RACF Security Guide* 说明如何获取证书并将其存储在 RACF 数据库的密钥环中或同等的外部安全性管理器中。如果服务器请求客户机证书，那么 CICS 提供在 URIMAP 定义中指定的证书标签，在用于连接的 WEB OPEN 命令中使用了该定义。或者，也可以在 WEB OPEN 命令的选项中直接指定证书标签。（如果使用 URIMAP 定义却未指定证书标签，那么使用在密钥环中为 CICS 区域用户标识定义的缺省证书。）

某些服务器可能要求您提供其他类型的认证或标识。如果无法将可接受的认证或标识提供给服务器，您的请求将被拒绝。对于基本认证或代理认证，服务器拒绝您的请求时使用的状态码与提示的状态码（对于服务器是 401，或对于代理是 407）相同。如果您响应提示，但是随后接收到进一步响应（它具有这些状态码中的某一个状态码），那么您使用的授权信息无效。

HTTP 基本认证的密码到期管理

当基本认证用于 HTTP 连接时，CICS Web Support 检查外部安全性管理器中的用户标识和密码。如果密码已到期，那么 CICS 提供的实用程序 DFHWBPW 用于提示用户选择新的密码。您可通过 DFHWBPW 定制或替换向用户显示的页面。

当应用到请求的 TCPIPSERVICE 定义是使用 AUTHENTICATE 属性的 BASIC、AUTOREGISTER 或 AUTOMATIC 选项定义的，那么 DFHWBPW 仅用于密码到期管理。尽管 DFHWBPW 具有类似于转换器程序的结构，但它不是正常 CICS Web Support 处理路径的一部分，因此您不需要为其他任何目的向它添加代码。当用户已选择他们的新密码时，DFHWBPW 通过将客户机重定向到原始请求的 URL 来重新启动请求提交，以使请求的完整处理路径与正常的路径一样出现。

DFHWBPW 向用户显示两个 Web 页面:

1. 密码提示页面。该页面包含两个元素:
   a. 有关密码有效性的消息。显示给用户的初始消息说明密码已到期。如果用户尝试更改密码时有问题（例如，所提供的新密码的两个副本不匹配），那么显示更多消息来说明该问题。
   b. 用户用于更改他们的密码的 HTML 表单。
2. 确认和请求刷新页面。该页面确认已成功替换到期的密码，并提供刷新标记和 URL 链接以便可自动或手工重新发出请求。

DFHWBPW 使用三个 CICS 文档模板（DFHWBPW1、DFHWBPW2 和 DFHWBPW3）构建这些 Web 页面。这些模板的 CICS 提供的定义将它们定义为可装入的程序: 即，它们的类型是 PROGRAM(DFHWBPW1) 等等。这些定义在 CICS 提供的 RDO 组 DFHWEB 中。您可以更改这些定义，方法是通过将它们复制到另一个组并使用 RDO ALTER 命令更改它们，以使模板来自不同的源。或者，您可以不更改 RDO 定义而是修改装入的程序。三个程序 DFHWBPW1、DFHWBPW2 和 DFHWBPW3 是仅汇编语言数据模块，并且它们的源在 CICS 样本库 SDFHSAMP 的相应成员中提供给您。您可以修改这些样本，并将它们重新汇编并链接编辑到您连接到 RPL 数据定义语句中的一个正常 CICS 程序库。

提示: 当您在汇编语言中对“与”符号（&）进行编码时，您必须将它们输入为双“与”符号（&&）。

每个 DFHWBPW 模板的内容和功能如下所示:

DFHWBPW1

密码提示页面部分。提供页面的 HTML 页面标题，并为可能的密码有效性消息设置符号（使用用于设置符号的服务器端包含技术）。这些消息向用户传达以下信息:

消息 1
密码已到期。

消息 2
输入的用户标识无效。

消息 3
所申请的新密码的两个副本不匹配。

消息 4
输入的前一个密码（刚到期的密码）不正确。

消息 5
由于密码质量规则，外部安全性管理器不接受所申请的新密码。

消息 6
目前已撤销用户标识。

DFHWBPW 程序选择合适的符号以插入到密码提示页面的文档中。您可以定制 DFHWBPW1 以更改页眉和标题，或改变 `body` 标记以更改页面颜色或背景。您还可以更改消息符号的内容。

DFHWBPW2

密码提示页面部分。构建 HTML 表单，用户可在其中输入用户标识、旧（已过期）密码和申请的新密码的两个相同副本。您可以定制 DFHWBPW2 以更改用

于提示用户的文本，或更改页面的布局。但您不能修改 form 标记的内容或任何一个 input 标记。如果您修改，DFHWBPW 可能不会按计划中的工作。

DFHWBPW3

确认和请求刷新页面。该文本通知用户已成功替换已到期的密码，并说明客户机会立刻提示用户再次输入密码，然后应再次输入新密码。您可以定制文本和页面的布局。

DFHWBPW3 是为重新启动请求过程设计的。它包含 meta http-equiv="Refresh" 标记，当检测到已到期的密码，该标记会在十秒后使自动重定向到用户原来请求过的页面。如果您不希望用户被自动重定向，您可以更改该标记的时间限制或完全除去它。然而，所修改的页面应总是包含指向原来请求的页面的链接。该页面的 URL 在符号 &dfhwbpw_target_url; 中。重新启动请求过程意味着如果 Web 客户机已将旧密码置于高速缓存，那么会立即用新密码替换它，并且还意味着 CICS Web Support 处理过程不受影响。

CICS Web Support 的 CICS 系统和资源安全性

当 CICS 是 HTTP 服务器时，必须保护 CICS 系统以不让未授权的用户访问。如果未正确地保护系统，那么用户也许能访问机密数据，或阻塞系统致使拒绝对其他用户的服务。

通常，要保护对 CICS Web Support 的访问，您应要求发出 HTTP 客户机请求的每个用户提供身份，并认证用户声明的身份。入站端口的 TCPIPSERVICE 定义用于指定这些需求。第 151 页的『作为 HTTP 服务器的 CICS: 认证和标识』说明了如何对作为 HTTP 服务器的 CICS 实现这一点。

Web 客户机使用的所有用户标识都必须在 RACF 或等效的外部安全性管理器中具有用户概要文件。RACF 用户概要文件提供了这方面的更多信息。

获取 Web 客户机的已认证用户标识之后，您可以使用该标识来对用来提供响应的 CICS 区域中的资源实施资源级安全性。过程因您提供的响应类型而异:

- 应用程序生成的响应。
- 静态响应，使用提供 CICS 文档模板作为响应的 URIMAP 定义。
- 静态响应，使用提供 z/OS UNIX Systems Services 文件作为响应的 URIMAP 定义。

对于应用程序生成的响应，CICS 系统缺省值指定不执行任何资源安全性检查，但执行事务安全性检查（特别是别名事务的事务连接安全性）。如果事务安全性在 CICS 区域中是活动的，那么您需要执行与应用程序生成的响应的安全性有关的一些操作，即便您不准备使用 Web 客户机的已认证用户标识进行安全性检查。

对于静态响应，事物连接安全性不会应用于 Web 客户机的用户标识。但是，如果用户标识可用于 Web 客户机，那么 CICS 系统缺省值指定**执行**资源级安全性检查。如果正从 Web 客户机获取已认证的用户标识，那么您需要设置这些用户标识的资源许可权，或采取操作以禁用资源层次安全性检查。

无论您是否选择对 CICS Web Support 提供的每个响应使用 Web 客户机的用户标识来实施资源层次安全性，都需要确保:

- 采取措施来保护入站端口，以防未经授权的或恶意的访问。

- 保护 CICS 系统组件，以防未经授权的用户进行修改，并确保已授权的用户能够正确访问它们。

相关信息

使用 HFS 文件的静态响应的资源层次安全性

实施 HFS 文件的安全性

入站端口的安全性

用于 CICS Web Support 的每个端口由 TCPIPSERVICE 资源定义来定义。TCPIPSERVICE 定义指定端口的安全性选项，包括是否使用 SSL 以及从客户机请求的认证的级别。需要尽可能严密地监视端口，以防未经授权的或恶意的访问。

第 83 页的『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』介绍了如何为用于 CICS Web Support 的端口创建 TCPIPSERVICE 定义。

为保持端口的安全，记住这些要点:

- 在每个 TCPIPSERVICE 定义中指定 MAXDATALEN 属性。该选项限制 CICS 为单个请求将要接受的最大数据量，并且它有助于保护 CICS 在进行大量数据传输时免受拒绝服务的攻击。
- 无论在何种情况下，如果您希望确保您与 Web 客户机的交互保持机密并无法被第三方拦截，都可使用安全套接字层（SSL）。在传输机密数据或将授权信息（如用户标识和密码）传递到服务器的情况下，使用 SSL 显得尤为重要。第 160 页的『具有 CICS Web Support 的 SSL』说明了如何将 SSL 用于 CICS Web Support。

如果您在一个或多个 CICS Web Support 端口上执行不寻常的操作，那么可使用 CICS 系统命令在不同级别（单个请求、虚拟主机、端口或整个 CICSWeb Support）关闭 CICS Web Support，而无需关闭 CICS 系统。第 97 页的『拒绝 HTTP 请求』说明了如何执行该操作。

URIMAP 资源定义指定 HTTP 或 HTTPS（带 SSL 的 HTTP）作为请求的方案。指定 HTTP 方案的 URIMAP 接受使用 HTTP 方案或更安全的 HTTPS 方案发出的 Web 客户机请求。指定 HTTPS 方案的 URIMAP 仅接受使用 HTTPS 方案发出的 Web 客户机请求。

当将 HTTPS 作为方案的 URIMAP 定义与 Web 客户机发出的请求匹配时，CICS 检查该请求使用的入站端口是否使用 SSL。如果没有为该端口指定 SSL，那么将拒绝该请求，状态码为 403（禁止）。当 URIMAP 定义应用到所有入站端口时，该检查会确保 Web 客户机无法使用不安全的端口来访问安全的资源。由于不会对采用 HTTP 方案的 URIMAP 定义进行任何检查，所以 Web 客户机可以使用非安全或安全的（SSL）端口访问这些资源。

CICS 系统组件的安全性

在正常的请求过程中，除非可发出 CICS 系统命令的 CICS 提供的事务是 Web Support 的，否则不能直接由 Web 客户机访问 CICS 系统组件。然而，与任何其他 CICS 资源一样，不让未授权的用户修改这些组件是很重要的。您还需要确保授权用户（尤其是 CICS 区域）具有使用这些组件的必需权限。

许多组件（如应用程序和资源定义）用于控制 CICS Web Support。这些组件在第 22 页的『CICS Web Support 的组件』中列出。如果您没有对这些组件提供保护以防止未经

授权的访问，那么 CICS Web Support 体系结构的安全性可能会遭到破坏。例如，对端口的 TCPIPSERVICE 定义具有访问权的用户会除去 Web 客户机使用 SSL 或提供身份的要求。*CICS RACF Security Guide* 说明了如何保护 CICS 事务、资源和命令，以防止未经授权的使用。

对于某些 CICS 系统组件，您可能需要设置其他权限以允许已授权的用户进行访问:

- 对于 URIMAP 定义，可能需要其他权限以设置 Web 客户机的用户标识。如果 CICS 区域中启用代理用户检查（将 XUSER=YES 指定为系统初始化参数），那么 CICS 检查用于安装 URIMAP 定义的用户标识是否被授权为 USERID 属性指定的用户标识的代理。
- 文档模板可用于作为 HTTP 服务器的 CICS 生成的响应的主体，或作为 HTTP 客户机的 CICS 生成的请求的主体。它们由 DOCTEMPLATE 资源定义来定义。如果文档模板存储在已分区的数据集，那么 CICS 区域用户标识必须具有该数据集的读权限。
- z/OS UNIX Systems Services 文件可用于从作为 HTTP 服务器的 CICS 生成静态响应的主体。它们可以用它们自己的名称指定或由 DOCTEMPLATE 资源定义来定义。如果使用 z/OS UNIX 文件，那么 CICS 区域必须具有 z/OS UNIX 的访问许可权，并且还必须具有包含文件的 z/OS UNIX 目录以及文件的访问许可权。*Java Applications in CICS* 说明了如何授予这些许可权。

应用程序生成的响应的资源和事务安全性

如果已获取了 Web 客户机（它在安全管理器中有一个概要文件）的已认证用户标识，那么该用户标识应用于别名事务，该别名事务负责处理应用程序生成的响应。您需要为 Web 客户机的用户标识授予合适的权限，或者提供自己的标准用户标识进行覆盖。无论是否使用 Web 客户机的用户标识进行资源安全性检查，都需要确保别名事务的用户标识具有合适的权限。

开始之前

关于此任务

别名事务由 TRANSACTION 资源定义指定。每个应用程序生成的响应的别名事务都由请求的 URIMAP 定义指定，或由分析器程序指定。缺省值为 CICS 提供的别名事务 CWBA。当使用支持 Web 的应用程序或 COMMAREA 应用程序来提供响应时，即适用以上情况。

别名事务运行所用的用户标识必须具有以下权限:

- 连接别名事务，如果对 CICS 区域指定了事务连接安全性的话。事务连接安全性由系统初始化参数 XTRAN 控制。缺省值为“YES”（事务连接安全性是活动的）。
- 访问别名事务使用的任何 CICS 资源，如果对别名事务指定了资源安全性的话。资源安全性由别名事务的 TRANSACTION 资源定义中的 RESSEC 属性控制。缺省值为“NO”（没有资源安全性），而且“NO”也是为 CWBA 提供的设置。
- 访问别名事务使用的任何 CICS 系统编程命令，如果对别名事务指定了命令安全性的话。这些系统编程命令会在生成响应的、用户编写的应用程序中使用。命令安全性由别名事务的 TRANSACTION 资源定义中的 CMDSEC 属性控制。缺省值为“NO”（没有命令安全性），而且“NO”也是为 CWBA 提供的设置。

当 Web 客户机对 CICS Web Support 发出请求，并且应用程序提供响应之后，CICS 会按以下优先级顺序为别名事务选择用户标识:

1. 使用分析器程序设置的用户标识。该用户标识可以覆盖从 Web 客户机中获取的用户标识或 URIMAP 定义提供的用户标识。
2. 使用基本认证从 Web 客户机获取的用户标识，或与 Web 客户机发送的客户机证书关联的用户标识。
3. 在请求的 URIMAP 定义中指定的用户标识。
4. CICS 缺省用户标识（如果不能确定其他用户标识）。

根据 CICS Web Support 体系结构，您可以将其中一个或多个用户标识类型用于不同的请求。如果获取了 Web 客户机的已认证用户标识，那么该标识会用于别名事务，除非您覆盖了该标识。

您需要对应用程序生成的响应执行以下安全性操作:

1. 如果获取了 Web 客户机的已认证用户标识，但**不**希望使用这些标识对应用程序生成的响应进行安全性检查，那么需要使用分析器程序，用相关别名事务的标准用户标识来覆盖 Web 客户机的用户标识。（您可以使用 CICS 缺省用户标识。）必须在处理请求时，将分析器程序用于进行覆盖的位置。 第 111 页的第 10 章，『分析器程序』介绍了如何编写分析器程序，以及如何将其与 URIMAP 定义结合使用。 请确保该用户标识在安全管理器中具有用户概要文件。 设置标准用户标识之后，您可以按照该过程的其余步骤所述，为标准用户标识分配必需的权限。
2. 如果没有获得 Web 客户机的已认证用户标识，那么选择合适的用户标识作为别名事务的标准用户标识。如果不希望使用 CICS 缺省用户标识，请在 URIMAP 定义中指定请求的用户标识，或设置分析器程序以指定这些标识。 确保标准用户标识在安全管理器中有已定义的用户概要文件。
3. 假定已为 CICS 区域指定了事务连接安全性，那么您需要确保别名事务所有可能采用的用户标识都有权连接事务。 这可以包括 Web 客户机用户标识（如果您获取了这些标识而且没有覆盖它们的话）或在 URIMAP 定义或分析器程序中指定的标准用户标识，也可以只包含 CICS 缺省用户标识。*CICS RACF Security Guide* 说明了如何向用户授予事务连接许可权。
4. 可选: 如果要对别名事务使用的资源应用资源级安全性检查:
   a. 识别别名事务使用的所有 CICS 资源，并确定要在 CICS 区域中对其中哪些资源进行资源安全性检查。 CICS Web Support 的应用程序可以使用的资源，以及控制对其进行资源安全性检查的系统初始化参数，包括:
      - CICS 文档模板（XDOC 系统初始化参数）。
      - 由主要应用程序调用以执行业务逻辑的其他应用程序（XPPT 系统初始化参数）。
      - 临时存储器队列，用于在 HTTP 请求序列上共享应用程序状态（XTST 系统初始化参数）。
      - CICS 文件控制管理的文件（XFCT 系统初始化参数）。

      注: 当应用程序使用 HFS 文件时，不会对 HFS 文件进行资源安全性检查（XHFS 系统初始化参数）。这是因为只有当文件被定义为 CICS 文档模板时，才能由应用程序处理这些文件，而且此时将由 CICS 文档模板安全性控制对这些文件的访问。

CICS RACF Security Guide 说明了如何为这些资源设置资源安全性检查（如果尚未执行该操作的话）。

如果正在使用分析器程序，这将是别名事务的主程序，因此无需进行资源安全性检查（只要进行事务连接安全性检查）。但是，请注意，用户编写的 Web 应用程序本身以及您使用的任何转换器程序，都将分别进行资源安全性检查。同样，如果使用转换器程序而非分析器程序，那么该转换器程序就是别名事务的主程序，但转换器程序调用的应用程序要分别进行资源安全性检查。

b. 对于所有有权连接别名事务的用户标识，是指使用别名事务所用安全资源的许可权。

c. 在事务的 TRANSACTION 资源定义中指定 RESSEC(YES)。

5. 可选: 如果要将命令安全性检查应用于别名事务使用的任何 CICS 系统编程命令:

a. 确认命令安全性在 CICS 区域中是活动的。命令安全性已被 XCMD 系统初始化参数激活。

b. 识别与事务关联的应用程序或程序、分析器程序（如果使用的话）以及分析器程序（如果使用的话）使用的 CICS 系统编程命令。 *CICS RACF Security Guide* 提供了命令核对表。

c. 对于所有有权连接别名事务的用户标识，是指使用别名事务所用命令的许可权。

d. 在别名事务的 TRANSACTION 资源定义中指定 CMDSEC(YES)。

下一步做什么

对于在 CICS 区域中执行的任何安全性检查，都必须设置系统初始化参数 SEC=YES。

使用文档模板的静态响应的资源层次安全性

对于 CICS Web Support 通过 URIMAP 定义中指定的 CICS 文档模板提供的静态响应，缺省情况下，已启用资源安全性检查。如果已经实施基本认证或客户机证书认证，并且还希望控制用户对特定 Web 页面的访问，那么您可以使用 Web 客户机的已认证的用户标识来控制对各个 CICS 文档模板的访问，这些文档模板用于提供静态响应。

开始之前

关于此任务

CICS 文档模板的资源安全性由 **XRES** 系统初始化参数控制。该参数的缺省值为“YES”，表示资源安全性是活动的。如果**不**希望对 CICS 区域中用于任何目的的 CICS 文档模板进行资源安全性检查，那么可以通过将该系统初始化参数设置为 NO 来停用它。

所有静态响应的事务是缺省的 Web 侦听器事务 CWXN，或备用事务，即指定通过使用 TCPIPSERVICE 定义中的 TRANSACTION 属性来替代 CWXN 的备用事务。对于 CICS 文档模板，资源安全性检查还可由事务的 TRANSACTION 资源定义中的 RESSEC 属性来控制。对于由 CICS 提供的 CWXN，指定 RESSEC(YES)，这意味着资源安全性是活动的。如果**不**希望对静态响应进行资源安全性检查，那么停用它的最好方法是用指定程序 DFHWBXN 且具有 RESSEC(NO) 的备用事务，来替代 TCPIPSERVICE 定义中的 CWXN。这只会停用 CICS 文档模板作为静态响应时执行的资源安全性检查。（请注意，对于由 HFSFILE 属性指定的 z/OS UNIX 文件，RESSEC 属性无法控制其安全性检查。）

注: 文档模板可以从各种源中检索，包括分区数据集、CICS 程序、CICS 文件、z/OS UNIX System Services 文件、临时存储器队列、瞬时数据队列以及出口程序。当对文档模板执行资源安全性检查时，CICS **不会**在提供文档模板的资源上执行其他任何安全性检查，即使已在 CICS 区域中对这种类型的资源指定了资源安全性。

要使用 CICS 文档模板设置静态响应的资源层次安全性:

1. 识别 Web 客户机使用的已认证的用户标识。提供这些标识是进行资源安全性检查的前提。（您无法使用分析器程序提供覆盖，但使用应用程序生成的响应就可以。） 已认证的用户标识具有已在安全性管理器中定义的用户概要文件。
2. 识别用来提供静态响应的所有 CICS 文档模板。
3. 请按照 CICS 文档模板的安全性中的指示信息，对 CICS 区域中的 CICS 文档模板实施安全性。 对于用来提供静态响应的各 CICS 文档模板，您需要将概要文件定义到安全管理器，并且为每个已认证的用户标识授权访问相应的 CICS 文档模板。
4. 确保在 CWXN 的 TRANSACTION 资源定义中指定 RESSEC(YES)，或在指定备用事务替换 CWXN。 RESSEC(YES) 已在 CICS 提供的 CWXN 中指定，但对于一般 TRANSACTION 资源定义，缺省值是 RESSEC(NO)。 该步骤激活了对静态响应的资源安全性检查，因此请确保在 Web 客户机提供用户标识时，设置相应的权限。

下一步做什么

对于要在 CICS 区域中执行的任何安全性检查，必须将系统初始化参数 SEC 设置为 YES。

具有 CICS Web Support 的 SSL

安全套接字层（SSL）可与 HTTP 协同使用以启用加密、消息认证，以及使用证书的客户机和服务器认证。HTTPS 方案是具有 SSL 的 HTTP。将 CICS 配置为使用 SSL 后，作为 HTTP 服务的 CICS 和作为 HTTP 客户机的 CICS 都可以使用其设施。

CICS RACF Security Guide 阐述 SSL 提供的设施， 告知您如何将 SSL 和 CICS 一起使用。

当 CICS 是 HTTP 服务器时，可以使用 SSL 保护与 Web 客户机的交互。为了执行该操作，在 TCPIPSERVICE 定义中为 CICS 接收客户机请求的端口指定合适的安全选项。

与指定对 SSL 的使用一样，您可以要求基本认证或要求客户机证书。要对 Web 客户机提供更多帮助，可以允许客户机提供客户机证书，然们将它们自己注册到安全管理器，为 CICS 环境提供标识。您还可以允许客户机根据需要使用自注册或基本认证以提供标识。所有这些活动都由 CICS 自己处理，因此如果您正在提供应用程序生成的响应，那么您的应用程序不需要处理这些活动。第 83 页的『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』说明了如何创建包含这些安全选项的 TCPIPSERVICE 定义。

当 CICS 是 HTTP 客户机时，服务器可能需要对某些连接使用 SSL。如果是这种情况，那么需要执行以下某些或所有操作:

- 将 HTTPS 用作连接方案。
- 提供您希望用于连接的密码套件列表。您可以在 URIMAP 定义中指定这些内容，您在用于连接的 WEB OPEN 命令中使用了该定义。
- 提供客户机证书。并非所有 SSL 事务都需要客户机证书，但服务器可能需要用于特殊事务的客户机证书。如果服务器请求客户机证书，您可以在 URIMAP 定义中指定

一个适当证书的标签，以用于 WEB OPEN 命令中的连接或 WEB OPEN 本身。客户机证书必须存储在安全管理器的密钥环中。（如果使用 URIMAP 定义却未指定证书标签，那么使用在密钥环中为 CICS 区域用户标识定义的缺省证书。）

第 14 章 CICS Web Support 和非 HTTP 请求

您可以使用 CICS Web Support 来处理非 HTTP 格式的入站 TCP/IP 客户机请求。在 CICS Transaction Server for z/OS V4R1 中，该设施主要用于为用户编写的、使用非标准请求格式的客户机发出的请求提供支持。针对请求所执行的处理及提供的响应均由用户定义。对于任何正式定义的用于客户机/服务器通信的协议，没有提供特定支持。

当 CICS 是服务器时，CICS Web Support 只处理非 HTTP 消息。作为客户机的 CICS 无法发出非 HTTP 请求。通过 CICS Web Support 发出的客户机请求使用 HTTP 协议。

当 CICS Web Support 设施用于处理非 HTTP 请求时:

- 您可以使用 TCPIPSERVICE 资源定义来控制接收请求的端口。
- 您可以使用分析器程序来组装请求并其进行语法分析、指定代码页转换，并确定后续的请求处理。您可以对分析器程序进行编码，以按照您所定义的任何请求格式对请求进行语法分析，但请注意，该 CICS 设施没有为任何已正式定义的特定协议提供特定支持。
- 您可以将支持 Web 的应用程序或不支持 Web 的应用程序用于转换器程序，以提供针对请求的响应。可以使用 EXEC CICS WEB 编程接口的某些元素来处理请求和响应，或在 COMMAREA 中的 CICS 应用程序之间传递请求和响应。
- 如果在分析器程序、转换器程序或用户编写的应用程序中发生异常终止，并且如果分析器程序和转换器程序无法确定应该执行什么应用程序来服务请求，那么 Web 出错程序 DFHWBEP 会提供错误响应。缺省情况下，使用标准的 HTTP 错误消息，但您可以根据需要定制这些消息。

某些 CICS Web Support 设施不可用于非 HTTP 请求:

- 帮助您解释 HTTP 请求并构造响应的某些设施不可用。例如，不能单独访问消息头。
- CICS TS V3 中引入的增强（包括分块的传输编码）通常不可用于非 HTTP 请求。
- 持续连接不受支持。
- URIMAP 定义不能用于非 HTTP 请求。

CICS Web Support 为非 HTTP 消息提供的支持不同于 CICS 的 TCP/IP 套接字接口。z/OS Communications Server IP CICS 套接字接口提供了应用程序编程接口，以允许客户机经由 TCP/IP 直接与 CICS 应用程序通信。CICS Web Support 不涉及该过程。

CICS 套接字接口是与 z/OS Communications Server 一起提供的，而不是与 CICS 一起提供。*z/OS Communications Server: IP CICS Sockets Guide*（SC31-8807）描述了 CICS 套接字接口。

处理非 HTTP 请求

要使用 CICS Web Support 设施处理非 HTTP 请求，您需要编写一个分析器程序以确定如何处理请求，并编写为请求提供响应的应用程序。您还需要创建一些资源定义。

开始之前

必须在开始前配置 CICS Web Support 的基本组件，如第 49 页的第 4 章，『配置 CICS Web Support 基本组件』中所述。

关于此任务

以下 CICS Web Support 组件用于处理非 HTTP 请求:

- TCPIPSERVICE 资源定义。
- 分析器程序。
- 转换器程序（如果需要）。
- 用户编写的应用程序。
- 应用程序的别名事务。
- Web 出错程序 DFHWBEP。

URIMAP 定义不与非 HTTP 请求协同使用。

HTTP 请求的处理和非 HTTP 请求的处理是分开进行的。非 HTTP 请求是通过 TCPIPSERVICE 定义中指定的 USER 协议来接收的。这确保 CICS 可以对 HTTP 请求和响应执行基本验收检查，并确保非 HTTP 请求不执行这些检查。验收检查将对非 HTTP 请求生成错误响应，而且不处理请求。

要使用 CICS Web Support 来处理非 HTTP 请求:

1. 决定将用于请求的端口。 请注意，因为每个端口只能存在一个活动的 TCPIPSERVICE 定义，所以非 HTTP 请求不能与 HTTP 请求使用同一端口。常用端口号 80（用于 HTTP）和 443（用于 HTTPS）必须用于支持 HTTP 协议，因此它们不能接受非 HTTP 请求。发出非 HTTP 请求的 Web 客户机必须在 URL 中为其请求明确指定端口号。
2. 使用第 165 页的『非 HTTP 请求的资源定义』中的信息，为请求设置资源定义。 非 HTTP 请求的 TCPIPSERVICE 定义必须指定 USER 协议。您还可以创建别名事务以覆盖请求处理。
3. 使用第 165 页的『分析器程序和非 HTTP 请求』中的信息，对分析器程序编码以处理每个请求。 需要分析器程序以确定请求的处理。它指定应用程序和（如果使用）转换器程序以处理每个请求。它还可以指定代码页转换参数。
4. 使用第 166 页的『用于非 HTTP 请求的应用程序编程』中的信息，设计和编码一个或多个应用程序以对每个请求提供一个响应。 这些程序可以使用 EXEC CICS WEB 编程接口的特定元素。它们还可以是不支持 Web 的应用程序，并生成由转换器程序编码的输出。
5. 确保 Web 出错程序 DFHWBEP 对于错误条件提供合适的响应。 对于非 HTTP 请求，如果分析器程序、转换器程序或用户编写的应用程序发生异常终止，或者如果分析器程序和转换器程序无法确定应当执行什么应用程序来响应请求，那么使用 DFHWBEP。 缺省情况下，DFHWBEP 输出标准 HTTP 消息，将其作为相同情况下 HTTP 请求的错误响应进行发送，但是您可以根据需要定制这些消息。 第 103 页的第 9 章，『Web 出错程序』说明了使用 DFHWBEP 的情况，以及如何定制提供的消息。

非 HTTP 请求的资源定义

非 HTTP 请求需要 TCPIPSERVICE 和 TRANSACTION 资源定义。非 HTTP 请求的 TCPIPSERVICE 资源定义必须指定 USER（用户定义的）协议，该协议与 CICS 提供的事务 CWXU 关联。当通过 USER 协议接收请求时，不使用 URIMAP 资源定义。

开始之前

关于此任务

1. 为您用于非 HTTP 请求的每个端口创建带 USER 协议的 TCPIPSERVICE 资源定义。可与 USER 协议一起使用的属性与可与 HTTP 协议一起使用的属性相同。第 83 页的『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』介绍了如何执行该操作。
2. 对于每个 TCPIPSERVICE 资源定义，决定是否使用 CICS 提供的事务 CWXU、CICS Web 用户定义的协议连接事务或备用事务。 DFHCURDI 样本包括 CWXU 的样本定义。CWXU 执行 CICS 程序 DFHWBXN。可能使用执行 DFHWBXN 的备用事务，但是不使用为 TCPIPSERVICE 资源定义中的协议指定的其他缺省事务。
3. 可选: 为要用于请求处理的任何别名事务创建 TRANSACTION 资源定义。第 86 页的『为 CICS Web Support 创建 TRANSACTION 资源定义』介绍了如何执行该操作。

分析器程序和非 HTTP 请求

需要分析器程序以处理非 HTTP 请求。它可以构造为了通过网络传输而分割的请求，指定请求的代码页转换，并执行确定后续请求处理所需要的任何语法分析。

重建非 HTTP 请求

入站请求可以分割成多部分，以通过网络传输。对于非 HTTP 请求，CICS 不会在调用分析器程序前重建请求，而且您应该相应地编写您的分析器代码。

在分析器的入口处，user_data 指针对包含第一部分入站请求的通信区域寻址。要接收请求的下一部分，将返回码设置为 URP_EXCEPTION，并将原因码设置为 URP_RECEIVE_OUTSTANDING。CICS Web Support 再次调用分析器，而 user_data 指针对消息的下一部分寻址。可以按需要多次重复这个过程直到整个请求被接收，直到最大支持长度 32767 字节。

此过程的结果对于 CICS WEB API 命令不可视。然而，重建的消息可以传递到转换器程序。

为非 HTTP 请求指定代码页转换

对于非 HTTP 请求，CICS Web Support 在调用分析器程序前不对请求执行任何代码页转换。

通过使用代码页转换表（DFHCNV）键或客户机和服务器代码页输出参数，分析器可以指定非 HTTP 请求的代码页转换，就像为 HTTP 请求执行此操作那样。第 113 页的『编写分析器程序』说明了如何执行该操作。

另外，支持 Web 的应用程序可以在 WEB RECEIVE 命令上指定入站非 HTTP 请求的代码页转换。

请注意，不将非 HTTP 请求解析为请求行、头和主体元素。必须为整个请求执行已执行的任何代码页转换。

确定非 HTTP 请求处理

与 HTTP 请求相关的以下输入字段在非 HTTP 请求的分析器程序中未定义:

- HTTP 版本
- 方法
- 请求的路径部分
- 请求头

因此，必须通过检查请求的内容来确定后续处理阶段。

分析器程序可以指定由转换器程序或支持 Web 的应用程序执行的后续请求处理。第 113 页的『编写分析器程序』说明了分析器程序的输入和输出，以及如何使用它们来决定请求处理。

用于非 HTTP 请求的应用程序编程

非 HTTP 请求的应用程序可以使用 EXEC CICS WEB 编程接口的特定元素。它们还可以是不支持 Web 的应用程序，并生成由转换器程序编码的输出。

伪会话编程模型不适合非 HTTP 请求。应用程序应该设计为接收单个请求并提供单个响应。

支持 Web 的应用程序

如果要使用支持 Web 的应用程序以响应非 HTTP 请求，那么可以使用以下 CICS API 命令:

- WEB RECEIVE 命令可用于接收非 HTTP 请求。如果该应用程序用于响应 HTTP 和非 HTTP 请求，那么可以在 WEB RECEIVE 命令上使用 TYPE 选项，以区分这两种请求。请注意，CICS 不会为非 HTTP 消息执行任何语法分析，而且不自动组装为了通过网络传输而分成多块的请求。如果分析器程序组合请求，那么结果对于 CICS WEB API 命令不可视。
- EXEC CICS DOCUMENT 命令可用于组成 CICS 文档，从而形成响应主体。
- WEB SEND 命令可用于将响应发送到非 HTTP 客户机。然而，与 HTTP 特定操作相关的以下选项不合适:
  - STATUSCODE 和 STATUSTEXT。如果指定，那么忽略这些内容。
  - CLOSESTATUS。如果指定，那么忽略此内容。
  - CHUNKING。如果指定，这会导致命令出错。
- WEB RETRIEVE 命令可用于检索早期 EXEC CICS WEB SEND 命令中发送的 CICS 文档。

其他 EXEC CICS WEB 命令仅与 HTTP 请求相关，并且如果与非 HTTP 请求协同使用会导致 INVREQ 情况。

应用程序可以使用 WEB RECEIVE 命令指定非 HTTP 请求的代码页转换。

具有转换器程序的不支持 Web 的应用程序

通过不支持 Web 的应用程序，您可以使用转换器程序将 Web 客户机的输入转换为适合应用程序的通信区域，并将应用程序的输出转换为 HTML 以提供响应。如果为了通过网络传输而分割请求后，分析器程序已重构该请求，那么此操作的结果会传递到转换器程序。

与 HTTP 请求相关的以下输入字段在非 HTTP 请求的转换器程序中未定义:

- HTTP 版本
- 方法
- URL 的路径部分
- 请求头

第 123 页的第 11 章，『转换器程序』具有有关编写转换器程序的更多信息。

第 15 章 CICS Web Support 和 3270 显示应用程序

当 Web 客户机访问 3270 事务时，CICS 可以将输出显示为 HTML 表单。使用各种 Web 终端转换应用程序（DFHWBTTA、DFHWBTTB 或 DFHWBTTC）提供 Web 客户机对应用程序的访问，这些应用程序原先是为使用 3270 显示系统而设计的。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

可用两种方式之一从 3270 事务的输出创建 HTML 表单:

- 对于使用 BMS 的应用程序，将从 BMS 映射中生成 HTML 模板，并将其存储到模板库。您可以定制模板的生成。然而，如果您需要对生成的 HTML 进行的唯一更改可以包含在标题或页脚部分，那么不需要从 BMS 映射生成模板，因为映射可以在执行时处理以生成 HTML 表单。
- 对于不使用 BMS 的应用程序，在执行生成 HTML 表单时处理 3270 出站数据流。

Web 终端转换应用程序可用于对 Web 浏览器显示 HTML 表单。

注: Web 终端转换应用程序在 HTTP/1.0 级别操作。它不完全使用 CICS Web Support 中提供的设施（例如，EXEC CICS WEB API），因此不提供与 HTTP/1.1 规范的一致性。这意味着:

- 不根据 HTTP 协议规范检查来自 Web 客户机的请求和来自应用程序的响应。
- 通常或错误情形下，CICS 不提供 HTTP/1.1 响应，即使客户机为 HTTP/1.1 级别时也是如此。

所有三种不同的 Web 终端转换应用程序都支持非会话式、会话式和伪会话事务。

- DFHWBTTA 和 DFHWBTTB 执行 3270 数据流和 HTML 之间的转换，以及 BMS 映射和 HTML 生成的模板之间的转换。如果 HTML 模板为 32767 字节（32K）数据或更小，那么使用 DFHWBTTA，如果 HTML 模板大于 32K，那么使用 DFHWBTTB。（将 DFHWBTTB 用于较小的 HTML 模板会导致不必要的性能开销。）
- 当没有模板生成时，DFHWBTTC 执行 BMS 映射和 HTML 之间的转换。以此方法使用的 BMS 映射必须指定 TERM=3270，或者忽略 TERM 参数。DFHWBTTC 支持任何长度的 HTML 输出。如果您无须生成 HTML 模板，那么使用 DFHWBTTC。

DFHWBTTB 和 DFHWBTTC 是 DFHWBTTA 的别名；在每种情况下调用同一程序（DFHWBTTA）。CICS 使用调用此程序的名称以确定需要哪个处理。

DFHWBTTA、DFHWBTTB 和 DFHWBTTC 生成符合 HTML 3.2 规范的 HTML。如果您使用不支持 HTML 3.2 的 Web 浏览器，一些功能可能不能正确工作。

为终端生成的 HTML 的页面大小会导致字段位置大于 4095（x'FFF'），这种 HTML 不起作用，尤其在使用 DFHWBTTC 时。使用旧的样式模板是一种例外情况。（旧样式模板是由 DFHWBTLG 从 PTF UQ53534 之前的 CICS TS 1.2 或 CICS TS 1.3 生成的）。已经提供了代码，以便在使用 DFHWBTTA 或 DFHWBTTB（而不是 DFHWBTTC）时允许 BMS 发送这类模板。

您可以将指定 DFHWBTTA、DFHWBTTB 或 DFHWBTTC 的 URIMAP 定义创建为程序，将为了处理请求而调用该程序（PROGRAM 属性）。Web 客户机用于访问程序的方法是类似，但是使用 URIMAP 定义为您提供了可用于阻止或重定向请求的在线管理设施。当使用 URIMAP 定义时，可以选择是否使用分析器程序。第 171 页的『3270 显示应用程序的 URL 路径部分』说明了使用 URIMAP 定义时如何正确指定 URL 路径。

CICS Web Support 不对以下各项提供支持：分区、逻辑设备代码、磁槽式阅读器、外部格式化或其他硬件功能部件。您可以使用具有光笔支持的可检测字段。

3270 应用程序的 CICS Web Support 处理

警告： 本主题包含产品敏感的编程接口和相关的指导信息。

图 8 说明 CICS Web Support 如何处理面向终端的事务。

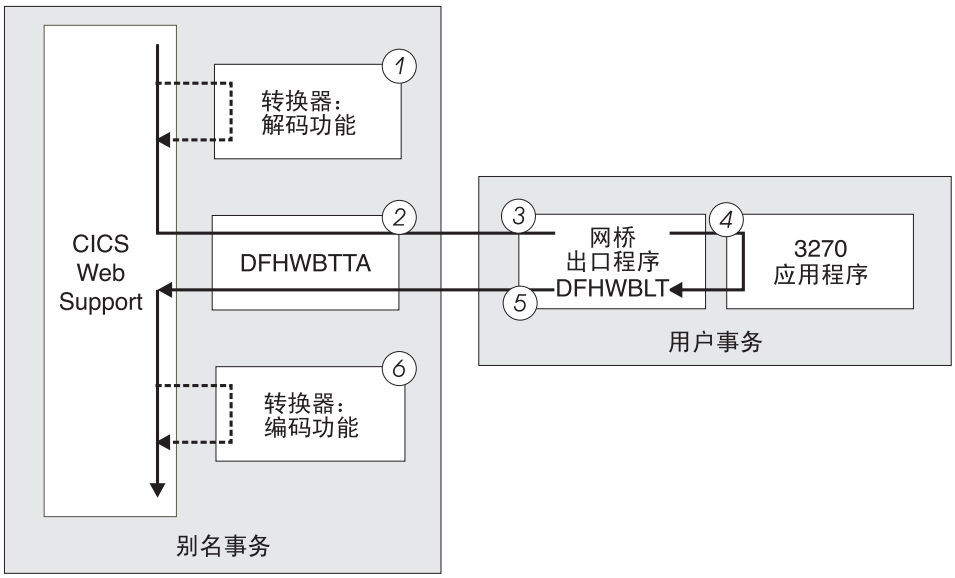


图 8. CICS Web Support 如何与 3270 应用程序交互作用

图中显示的步骤如下：

1. 可选地，转换器程序构造传递到程序 DFHWBTTA 的输入。
2. DFHWBTTA 与用户事务连接，指定 DFHWBLT 作为网桥出口程序，并等待来自 DFHWBLT 的响应。用户事务在 3270 网桥环境中执行。
3. 网桥出口为用户应用程序设置 3270 环境。
4. 应用程序处理输入，并构造 3270 输出。
5. 网桥出口解释 3270 输出，并把 HTTP 响应传递到 DFHWBTTA。
6. 可选地，转换器程序修改传递到 Web 客户机的输出。

注： 当您将 CICS Web Support 与 3270 应用程序协同使用时，应用程序在它自己的事务下执行，而不是在别名事务下执行。

要获取关于 3270 网桥的更多信息，请参阅 *CICS External Interfaces Guide* 中的至 3270 事务的网桥。

3270 显示应用程序的 URL 路径部分

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

为了从 Web 浏览器调用 CICS 3270 应用程序，您必须输入具有路径部分的 URL（该路径部分通过调用应用程序名 DFHWBTTA、DFHWBTTB 或 DFHWBTTC 启动）以及合适的别名事务和转换器程序（如果需要）。请注意，该别名事务不应用到 3270 应用程序本身，它在它自己的事务下运行。

使用分析器程序

如果您正在使用分析器程序（如 CICS 提供的样本分析器 DFHWBADX）处理请求，那么 URL 的路径部分包括应用程序（DFHWBTTA、DFHWBTTB 或 DFHWBTTC）的名称。它还包括您正在使用的任何转换器程序的名称，以及用于请求处理的别名事务（例如，缺省 CICS 提供的别名事务 CWBA）的名称。如 第 119 页的『CICS 提供的样本分析器程序 DFHWBADX』 中所说明，路径的这些元素由分析器程序抽取，并用于调用子序列处理阶段。

使用 URIMAP 定义

如果您正在使用 URIMAP 定义处理请求，那么 URL 的路径部分在 PATH 属性中指定。使用 URIMAP 定义，URL 的路径部分不需要包含有关应用程序、转换器程序和别名事务的明确信息（虽然它仍可以包含这些内容）。使用 PROGRAM、CONVERTER 和 TRANSACTION 属性，所有这些元素都可以在 URIMAP 定义中指定。然后，这部分路径可以由您选择的任何路径所替换。为了满足 DFHWBTTA 本身的需求，请在您要在 URIMAP 定义中指定的路径末尾使用星号作为通配符。这将允许剩余路径部分更改为控制 DFHWBTTA。

使用 URIMAP 定义和分析器程序

通过在 URIMAP 定义中指定 ANALYZER(YES) 选项，您可以在请求的处理路径中使用分析器程序。分析器程序可以动态修改转换器程序、URIMAP 定义指定的别名事务标识和程序名，而 DFHWBTTA 可以看到这些更改。

提供调用应用程序所需的信息后，URL 的下一部分路径用于对 DFHWBTTA 提供控制信息。该信息包括:

- 指定是否应该使用未格式化方式的关键字。
- 您要使用的 3270 应用程序的事务标识。
- 指定事务的输入参数，它使用加号（+）作为定界符。

图 9 显示 DFHWBTTA 解释的路径部分语法。

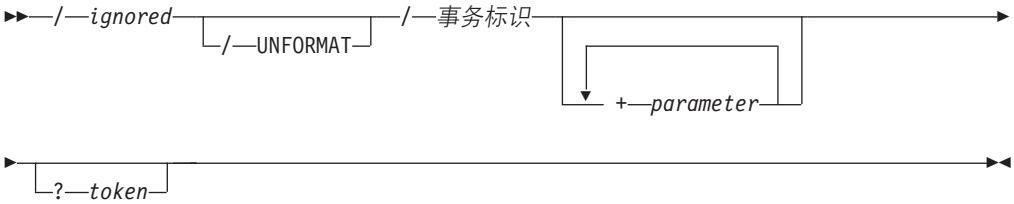


图 9. DFHWBTTA 解释的路径语法

DFHWBTTA 将 URL 的路径部分解释如下:

ignored
DFHWBTTA 将忽略路径的第一部分。这是由分析器解释或与 URIMAP 定义匹配的部分，用于提供调用应用程序所需的信息。

UNFORMAT
3270 显示可以在两种方式下操作，格式化方式和未格式化方式。如果出现这个关键字，DFHWBTTA 模拟以未格式化方式操作的 3270 显示。如果这个关键字被省略，DFHWBTTA 模拟以格式化方式操作的 3270 显示。

有关 3270 显示如何以未格式化方式操作的更多信息，请参阅 *CICS Application Programming Guide* 中的未格式化方式。

事务标识
在初始请求中，该信息指定要运行的 CICS 事务。在连续请求中，路径的这个元素将被忽略。

parameter
指定事务的输入参数。使用加号（+）（而不是空格）作为定界符以分隔事务标识和该数据，并分隔该数据的元素。

token
DFHWBTTA 将忽略它。它可以由分析器程序使用。

URL 必须总是以该格式编码。

例如，如果您正在使用 CICS 提供的分析器程序 DFHWBADX，那么您可以使用以下 URL 路径发出 CEMT INQ TAS 命令:

```
/cics/cwba/dfhwbtta/CEMT+INQ+TAS
```

在本示例中:

- `cics` 用于表明不需要转换器程序。
- `cwba` 是用于请求处理的别名事务的名称。
- `dfhwbtta` 是应用程序名。
- `CEMT+INQ+TAS` 告诉 DFHWBTTA 访问 CEMT 事务并发出 INQ TAS 命令。

另外，您可以设置包含以下属性的 URIMAP 定义:

```
Path:        /terminal/*
Transaction:  CWBA
Program:      DFHWBTTA
```

启用该 URIMAP 定义后，您可以使用以下 URL 路径发出 CEMT INQ TAS 命令:

```
/terminal/CEMT+INQ+TAS
```

在本示例中:

- `terminal` 与 URIMAP 定义匹配，它指定别名事务和应用程序的名称。
- `CEMT+INQ+TAS` 被 URIMAP 定义忽略，但是它告诉 DFHWBTTA 访问 CEMT 事务并发出 INQ TAS 命令。

初始和连续请求

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

DFHWBTTA 通过下列 HTTP 请求在事务中的上下文来区分其类型: 初始请求和连续请求。

初始请求

初始请求启动 CICS 事务。用以下某种方法发送初始请求:

- 明确输入 URL。
- 选择 HTML 页面中的链接。
- 选择 HTML 表单中的按钮。在表单中输入的任何数据将被忽略。

连续请求

连续请求继续现有的 CICS 事务。用以下方式发送连续请求:

- 选择作为先前请求的响应而显示的 HTML 表单中的按钮。

连续请求使用 HTML POST 方法; 表单数据在 HTML 请求的实体主体中传输。

在会话式或伪会话事务中，Web 客户机和 CICS 之间存在多个交互，其中有一个初始请求，后跟一个或多个连续请求。在更简单的事务中，仅有一个交互，其中有一个初始请求，无连续请求。

由初始请求显示并由后续请求返回的 HTML 表单中的隐藏元素（DFH_STATE_TOKEN）用于辨别初始请求和连续请求，并用于将连续请求与正确的事务关联起来。

连续请求的事务标识

在连续请求中，URL 是以先前请求显示的格式进行编码的。然而，在连续请求中，将忽略在 URL 中编码的事务标识。

反之，可以用下列方式确定事务:

- 当连续请求是对话式事务的一部分时，相同的事务继续执行。
- 当连续请求是伪对话式事务的一部分时，那么:
  - 如果先前事务是由具有 TRANSID 选项的 EXEC CICS RETURN 命令终止的，那么指定的事务标识将被使用。
  - 如果先前的事务没有在它的 EXEC CICS RETURN 命令中指定事务标识，但是 AID 与事务标识关联，那么使用该事务标识。
  - 如果没有在 EXEC CICS RETURN 命令中指定事务标识，并且没有与 AID 关联的事务标识，那么 CICS 从 HTML 表单获取事务标识。

HTML 表单中的事务标识

从 3270 显示附加一个事务时，CICS 希望在 3270 数据流的第一个已修改字段中找到事务标识。

Web 客户机发送表单数据的顺序并不总是可预料的，因此 CICS 使用表单字段名称和 3270 屏幕上相应位置之间的映射:

- 对于不使用 BMS 映射的事务，映射直接使用字段名，因为该名称反映了字段在 3270 屏幕的位置。

- 对于使用 BMS 映射的事务，字段名并不总是反映字段在 3270 屏幕的位置，所以使用间接映射。映射使用隐藏变量 DFH_NEXTTRANSID.*n*。当从 BMS 映射中创建一个 HTML 模板时，最多将创建五个变量。每个变量的值是一个输入字段的名称，按 3270 缓冲区位置的顺序排列。

  当 CICS 接收 HTTP 请求时，它依次检查每个 DFH_NEXTTRANSID 字段，以确定它引用的输入字段名称，以及 HTTP 请求是否包含该字段的值。如果它包含，那是因为最终用户修改了它，并因此假设其中包含下一个事务的事务标识。

  当通过合并几个 BMS 以及非 BMS SEND 命令的输出来构造屏幕时，会出现输入字段被抑制的情况（要获取更多信息，请参阅第 189 页的『如何选择页脚部分』）。因此 CICS 可以正确地识别 3270 数据流中的事务标识，您必须确保合并的 HTML 页面中没有禁止可能包含事务标识的输入字段。它引用的输入字段名称，以及 HTTP 请求是否包含该字段的值。

3270 应用程序的 CICS Web Support 中的终端控制命令

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

3270 应用程序的 CICS Web Support 支持以下终端控制命令: SEND、CONVERSE 和 RECEIVE

关于此任务

它还支持最小函数 BMS 和 SEND TEXT 命令。

忽略 SEND 和 CONVERSE 命令中的 DEFRESP 选项。这可能影响应用程序恢复。

从 BMS 映射生成的 HTML 模板

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

3270 显示系统的功能与 HTML 表单有许多相似之处:

- 在两种情况下，显示区域都可以包含固定文本和用户能够输入数据的区域。
- 3270 键盘上的 AID 键与 HTML 表单上显示的按钮具有相似的功能。
- 在两种情况下，都可以检测最终用户是否修改了数据输入字段的内容。

从 BMS 映射生成的模板包含许多表示 3270 显示功能的元素:

- 映射中的保护字段作为普通 HTML 文本显示。
- 映射中的未保护字段作为文本输入元素显示。 CICS 为每个元素提供一个两部分的名称，该名称最多可以长达 32 个字符:
  - 名称的第一部分长度为 11 个字符，并且具有以下格式:

    F*rrcclllll*_

    其中
    - *rr* 是一个二位数字，表示在 3270 屏幕上显示的字段所在的行。
    - *ccc* 是一个三位数字，表示在 3270 屏幕上显示的字段所在的列。
    - *llll* 是一个四位数字，表示字段的长度。

– 对于在映射中命名的 BMS 字段，第二部分包含映射中使用的名称，如果必要，可以将它截断到 21 个字符的长度。
– 对于未命名的 BMS 字段，第二部分格式为 DFH_*nnnn*，其中 *nnnn* 是一个 4 位数。按照字段在 BMS 映射中出现的顺序为它们依次编号。

例如，假设第三个未命名的未保护字段被定位到屏幕的第 2 行、第 11 列，并且长度为 16 个字符。生成的两部分的名称为

```
F020110016_DFH0003
```

现在假设在 BMS 映射中有名称为 TOTAL_MONTHLY_PURCHASES 的相同字段。CICS 为 HTML 元素生成的名称是:

```
F020110016_TOTAL_MONTHLY_PURCHAS
```

注: 字段在 3270 屏幕上显示的顺序和它们在 BMS 映射定义中编码的顺序可能不同。当相应的模板在 Web 客户机上显示时，字段是按编码顺序显示的。

- 3270 显示支持的每个辅助操作请求键都是模拟提交按钮的。按钮命名为:
  – DFH_PF01 到 DFH_PF24
  – DFH_PA1 到 DFH_PA3
  – DFH_ENTER, DFH_CLEAR

  当最终用户选择其中一个按钮时，相应的变量在 HTTP 请求中被发送。CICS 使用这个变量来确定在 3270 应用程序中模拟哪个 AID。

  一个名为 DFH_PEN 的附加提交按钮与可检测字段一起使用。
- 可检测字段是用来模拟有前导复选框的文本元素的。请参阅第 181 页的『使用可检测字段』以获取更多信息。
- 隐藏元素（DFH_STATE_TOKEN）用来维护应用程序在许多与 Web 客户机的交互上所看到的显示状态。
- 隐藏元素（DFH_CURSOR）和 JavaScript™ 函数（dfhinqcursor()）通过协作来向应用程序返回光标位置。
- 一系列隐藏元素（从 DFH_NEXTTRANSID.1 到 DFH_NEXTTRANSID.*n*）用于捕捉输入到 Web 客户机字段的事务标识符。

从 3270 数据流生成的 HTML 页面

对于不使用 BMS 的应用程序，CICS Web Support 生成由以下三个部分组成的 HTML 页面: 标题部分、屏幕图像部分和页脚部分。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

标题部分

CICS Web Support 生成以下标题部分:

```
<!doctype html public "-//W3C//DTD HTML 3.2//EN">
<html>
<STYLE TYPE="text/css">
<!--
        TABLE, TR, TD
        { padding: 0mm    }
        TABLE
        { width: 60%    }
```

```
-->
.BRIGHT
{font-weight: bold}
{font-family: courier}
.INPUT
{font-family: courier}
</STYLE>
<head>
<title>CICS Web Support screen emulation - tranid</title>
<meta name="generator" content="CICS Transaction Server/2.2.0">
<script language="JavaScript">
<!--
  function dfhsetcursor(n)
    {for (var i=0;i<document.form3270.elements.length;i++)
          {if (document.form3270.elements[i].name == n)
                  {document.form3270.elements[i].focus();
                    document.form3270.DFH_CURSOR.value=n;
                    break}}}
  function dfhinqcursor(n)
      {document.form3270.DFH_CURSOR.value=n}
 // -->
</script>
</head>
<body onLoad="dfhsetcursor('&DFH_CURSORPOSN;')">
```

您可以通过提供自己的标题部分来修页面外观。要获取更多信息，请参阅第 178 页的『修改 DFHWBTTA 的输出』。

屏幕图像部分

HTML 页面的这部分是从 3270 屏幕图像的内部表示法直接生成的，其大小由与您的事务关联的 FACILITYLIKE 终端定义中的 DEFSCREEN 和 ALTSCREEN 定义决定。它包含以下元素:

普通 HTML 文本
模拟保护字段

文本输入元素
模拟未保护字段。为每个元素提供一个 11 个字符长的名称，并具有以下格式:

*Frrcccllll*_

其中

- *rr* 是一个二位数字，表示在 3270 屏幕上显示的字段所在的行。
- *ccc* 是一个三位数字，表示在 3270 屏幕上显示的字段所在的列。
- *llll* 是一个四位数字，表示字段的长度。

例如:

- 一个在 3270 显示的第 1 行、第 1 列，并且长度为 78 字节的字段将被命名为

  F010010078_

具有复选框的文本元素
模拟可检测字段。请参阅第 181 页的『使用可检测字段』以获取更多信息。

隐藏元素
名为 DFH_STATE_TOKEN 的隐藏元素用来维护应用程序在许多与 Web 客户机的交互上所看到的显示状态。

隐藏元素（DFH_CURSOR）和 JavaScript 函数（dfhinqcursor()）通过协作来向应用程序返回光标位置。CICS 使用 JavaScript focus() 方法来定位由 DFH_CURSOR 指定的输入框或字段中光标的位置。注: focus() 无法确定输入框或字段中特定字符上光标的位置，而只能定位在第一个字符位置上。

```
<!doctype html public "-//W3C//DTD HTML 3.2//EN">
<html>
<head>
<title>CICS Web Support screen emulation - tranid</title>
<meta name="generator" content="CICS Transaction Server/2.1.0">
<script language="JavaScript">
<!--
  function dfhsetcursor(n)
    {for (var i=0;i<document.form3270.elements.length;i++)
          {if (document.form3270.elements[i].name == n)
                  {document.form3270.elements[i].focus();
                    document.form3270.DFH_CURSOR.value=n;
                    break}}}
  function dfhinqcursor(n)
      {document.form3270.DFH_CURSOR.value=n}
 // -->
</script>
</head>
<body onLoad="dfhsetcursor('&DFH_CURSORPOSN;')">
```

从 3270 屏幕图像中生成的 HTML 和从用于 BMS 映射的模板中生成的 HTML 很相似。通过使用 HTML 表来实现页面上信息的水平和垂直对齐:

- HTML 表对于 3270 屏幕中包含字段开始的每个不同的列都包含一列。例如，如果 3270 屏幕包含的字段开始于第 2、11、21 和 55 列，那么 HTML 表将包含四列。因此，所有在 3270 屏幕上其起始位置垂直对齐的字段，将在 HTML 页面上垂直对齐。
- HTML 表对于 3270 屏幕上包含字段开始的每一行都包含一行。因此，所有起始位置在 3270 屏幕上水平对齐的字段将在 HTML 页面上水平对齐。3270 屏幕上不包含字段的行在 HTML 表上将不表示出来。
- 在表中，文本以比例字体显示。

考虑包含以下字段的 3270 屏幕:

字段	行	起始列
Field_1	2	2
Field_2	3	2
Field_3	3	35
Field_4	4	2
Field_5	4	35
Field_6	9	2
Field_7	9	18
Field_8	9	35

所有字段都从 3270 屏幕的第 2、18 或 35 列开始。所以结果 HTML 表将有三列。同样，所有字段都位于 3270 屏幕的第 2、3、4 或 9 行，所以 HTML 表将有四行。

您可以使用转换器程序的编码函数来修改屏幕图像部分。要获取更多信息，请参阅第 180 页的『将转换器程序与 DFHWBTTA 协同使用』。

页脚部分

CICS Web Support 生成以下页脚部分。3270 显示支持的每个辅助操作请求键都是模拟提交按钮的。当最终用户选择其中一个按钮时，相应的变量在 HTTP 请求中被发送。CICS 使用这个变量来确定在 3270 应用程序中模拟哪个 AID。一个名为 DFH_PEN 的附加提交按钮与可检测字段一起使用。

```
<input type="submit" name="DFH_PF1"  value="PF1">
<input type="submit" name="DFH_PF2"  value="PF2">
<input type="submit" name="DFH_PF3"  value="PF3">
<input type="submit" name="DFH_PF4"  value="PF4">
<input type="submit" name="DFH_PF5"  value="PF5">
<input type="submit" name="DFH_PF6"  value="PF6">
<input type="submit" name="DFH_PF7"  value="PF7">
<input type="submit" name="DFH_PF8"  value="PF8">
<input type="submit" name="DFH_PF9"  value="PF9">
<input type="submit" name="DFH_PF10"  value="PF10">
<input type="submit" name="DFH_PF11"  value="PF11">
<input type="submit" name="DFH_PF12"  value="PF12">
<br>
<input type="submit" name="DFH_PF13"  value="PF13">
<input type="submit" name="DFH_PF14"  value="PF14">
<input type="submit" name="DFH_PF15"  value="PF15">
<input type="submit" name="DFH_PF16"  value="PF16">
<input type="submit" name="DFH_PF17"  value="PF17">
<input type="submit" name="DFH_PF18"  value="PF18">
<input type="submit" name="DFH_PF19"  value="PF19">
<input type="submit" name="DFH_PF20"  value="PF20">
<input type="submit" name="DFH_PF21"  value="PF21">
<input type="submit" name="DFH_PF22"  value="PF22">
<input type="submit" name="DFH_PF23"  value="PF23">
<input type="submit" name="DFH_PF24"  value="PF24">
<br>
<input type="submit" name="DFH_PA1"   value="PA1">
<input type="submit" name="DFH_PA2"   value="PA2">
<input type="submit" name="DFH_PA3"   value="PA3">
<input type="submit" name="DFH_CLEAR" value="Clear">
<input type="submit" name="DFH_ENTER" value="Enter">
<input type="submit" name="DFH_PEN"   value="Pen">
<input type="reset"  value="Reset">
</form>
</body>
</html>
```

您可以通过提供自己的页脚部分来修页面外观。要获取更多信息，请参阅『修改 DFHWBTTA 的输出』。

修改 DFHWBTTA 的输出

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

您可以通过定制 HTML 或通过提供您自己的标题和页脚部分来修改 DFHWBTTA 的输出。

关于此任务

对于使用 BMS 的应用程序，您可以定制 BMS 映射所创建的 HTML 模板。有关定制 HTML 模板的更多消息，请参阅第 185 页的『生成定制的 HTML 模板』。

对于非 BMS 应用程序和使用 DFHWBTTC 调用的 BMS 应用程序，可以通过提供您自己的标题和页脚部分来修改页面的外观。您不能直接更改屏幕图像部分，但是您在标题部分插入的标记将影响以后各部分的外观。

要提供您自己的标题和页脚部分，定义和安装下列模板中的一个或多个，其名称在 DOCTEMPLATE 定义的 TEMPLATENAME 字段中定义:

*tran***HEAD**
: 这是在事务 *tran* 输出的 HTML 页面标题中插入的模板（如果安装）。

CICSHEAD
: 这是要在作为没有安装相应 *tran*HEAD 模板的事务输出的 HTML 页面标题处插入的模板。

*tran***FOOT**
: 这是在作为事务 *tran* 输出的 HTML 页面的页脚处插入的模板（如果安装）。如果没有安装这个模板，将使用 CICSFOOT。

CICSFOOT
: 这是要在作为没有安装相应 *tran*FOOT 模板的事务输出的 HTML 页面页脚处插入的模板。

有关创建文档模板的更多信息，请参阅 *CICS Application Programming Guide* 中的使用文档和文档模板编程。

由 CICS Web Support 生成的标题部分（包括 DFHWBTTC）使用 EBCDIC Latin 字符集（代码页 037）。如果在 CICS 系统中使用不同的代码页，那么必须使用您自己的代码页创建类似的标题部分:

1. 创建包含标题部分，称为 CICSHEAD 的文档模板。
2. 定义和安装模板的 DOCTEMPLATE 定义。

. 用于 CICS 生成的标题部分中的以下字符在不同于 037 的代码页中有不同的表示法:

```
! [ ] { }
```

提供您自己的标题模板

如果您提供了自己的标题模板，那么还必须提供 HTML 页面的一些必需元素。

关于此任务

标题模板应该包含以下 HTML 元素:

- doctype 标记，例如:

  ```
  <!doctype html public "-//W3C//DTD HTML 3.2//EN>
  ```

- <html> 标记
- <head> 标记
- <STYLE> 标记，它必须包含 BRIGHT 和 INPUT 类的样式表规则。例如:

  ```
  <STYLE TYPE="text/css">
  <!--
          TABLE, TR, TD
          { padding: 0mm    }
          TABLE
          { width: 60%    }
  -->
  .BRIGHT
  ```

```
{font-weight: bold}
{font-family: courier}
.INPUT
{font-family: courier}
</STYLE>
```

您可使用 TABLE 元素的 width 属性微调屏幕图像部分的外观。

- </head> 标记
- <body> 标记。您可以用这个标记指定文本颜色，或者指定用于页面背景的图像。例如:

```
<body background="/dfhwbimg/background2.gif" bgcolor=#FFFFFF"
 text="#000000" link="#00FFFF" vlink="#800080" alink="#FF0000>
onLoad="dfhsetcursor('&DFH_CURSORPOSN;')"
```

注: 此示例使用第 182 页的『使用 DFHWBIMG 显示图形』中描述的 DFHWBIMG。

- 定制页面所需的任何其他 HTML 元素，可选。

提供您自己的页脚模板

如果您提供了自己的页脚模板，那么还必须提供 HTML 页面的一些必需元素。

关于此任务

页脚模板应该包含以下 HTML 元素:

- 代表任何已编程功能键或 ENTER 键的输入按钮。例如:

```
<input type="submit" name="DFH_PF1"  value="Help">
<input type="submit" name="DFH_PF3"  value="Quit">
<input type="submit" name="DFH_ENTER"  value="Continue">
```

HTML 表单的这些表单部分由 CICS 开始。用户选择该按钮时，产生第 175 页的『从 3270 数据流生成的 HTML 页面』中所讨论的 AID 指示符，所以应该具有在此描述的名称。*value* 参数指定出现在生成的按钮上的图注。DFHWBTTA 不使用它。

- </form> 标记
- 定制页面所需的任何其他 HTML 元素，可选。
- 用于关闭页面的 </body> 标记
- </html> 标记

将转换器程序与 DFHWBTTA 协同使用

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

您可以使用转换器程序的解码函数修改传递到 DFHWBTTA 的请求。

关于此任务

- 当客户机使用一个表示辅助操作请求键的按钮来提交 HTML 表单时，请求包含指示哪个按钮被选中的字段。可以通过修改请求来模拟另一个辅助操作请求键的影响。将字段值改为所需的辅助操作请求键，或者在通过 Web 客户机发送的字段后面插入一个新字段。
- 当客户机提交 HTML 表单时，DFH_CURSOR 字段包含含有光标的字段名称。可以通过修改请求来模拟另一个光标位置的影响。更改 DFH_CURSOR 字段的值以包含另一个字段名，或者在通过 Web 客户机发送的字段后面插入一个新的 DFH_CURSOR 字段。

- 可以在连续请求中更改 DFH_NEXTTRANSID.*n* 变量以选择下一个事务标识。可以插入或删除一个变量，或修改其值。有关如何使用这些字段来确定下一个事务标识的详细信息，请参阅 第 173 页的『HTML 表单中的事务标识』。

不要修改 DFH_STATE_TOKEN 的值。

您可以使用转换器程序的编码函数修改 DFHWBTTA 的输出:

- 响应存在于一个缓冲区中，该缓冲区以一个用于指定缓冲区长度的 32 位无符号数开始。缓冲区的其余部分是 HTTP 响应。响应中的 HTML 对应于来自事务程序的输出 BMS 映像或 3270 数据流。
- HTTP 响应中的 HTTP 头是由 DFHWBTTA 自动生成的。由 DFHWBTTA 生成的头是:
  - Content-type: text/html
  - Content-length: <length of the entity body>
  - Pragma: no-cache
  - Connection: Keep-Alive（如果是 HTTP 1.0 持续连接的话）

  如果需要任何附加的头，应该使用转换器的编码函数来将它们添加到 HTTP 响应中。

启用可检测字段

要启用 CICS Web Support 3270 网桥上的可检测字段处理，必须定义允许光笔支持的网桥设施。

关于此任务

要这样做，按照这些步骤操作:

1. 将下列定义复制到新组。除非运行在 CICS 系统上的所有应用程序需要光笔支持，否则还要重命名这两个定义。
   - 组 DFHTERM 中，由 CICS 提供的网桥设施 CBRF。
   - 组 DFHTYPE 中，它的缺省值 TYPETERM: DFHLU2。
2. 在 TYPETERM 定义中，将“DEVICE PROPERTIES”下的 LIGHTPEN 选项更改为 YES。
3. 在 TERMINAL 定义中，将 TYPETERM 参数更改成指向新的 TYPETERM。
4. 将定义安装在 CICS 区域中。
5. 如果您已创建新的网桥设施定义，那么更新将与 CICS Web Support 一起运行的 3270 事务的 PROFILE 定义，以便在新的 TERMINAL/TYPETERM 定义中建模网桥设施:
   a. 通过使用 CEDA 来标识事务使用的 PROFILE，以查看 TRANSACTION 定义的 PROFILE 参数。
   b. 如果概要是由 CICS 提供的，那么将它复制到您自己的组中并重命名。
   c. 改变新的 PROFILE 并在 FACILITYLIKE 参数中输入新网桥设施的名称。
   d. 改变 TRANSACTION 定义以使用新的 PROFILE 定义。

使用可检测字段

当 CICS 从 3270 数据流生成 HTML 页面时，它用复选框后面的文本输入字段模拟可检测字段。

开始之前

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

要使用可检测字段，必须配置与事务关联的网桥设施。第 181 页的『启用可检测字段』介绍了如何执行该操作。

关于此任务

可检测字段存在于:

- 将字段标识为可检测的或增强的字段属性字节。
- *并且* 3270 字段的第一个字符包含有效的指示字符。这可以是“与”符号（&）、右尖括号（>）、问号（?）、空白或空。

要获取有关可检测字段的更多信息，请参阅 *CICS Application Programming Guide* 中的字段选择功能部件。

当复选框和文本输入字段显示在 Web 客户机上时:

- 3270 字段中的指示字符将不被显示。因此，在 Web 客户机中的字段长度比在 3270 数据流中的字段长度少一个字符。
- 如果指示字符是右尖括号（>），那么复选框包含一个选中符号（✔)。否则，复选框为空。

要在 Web 客户机上使用可检测字段:

- 选中复选框以模拟设置 3270 数据流的修改数据标记（MDT）位。清除该框将关闭修改数据标记。 在 HTML 页面的文本字段中输入数据不会更改修改数据标记。
- 要将数据传输到 CICS 应用程序，那么选取复选框，并选择名为 DFH_PEN 的按钮。
  - 如果只选取了一个注意字段，那么 CICS 应用程序只接收该字段的内容。EIBAID 字段被设置为 DFHPEN。
  - 如果选取了多个注意字段，那么 CICS 应用程序接收最接近 3270 屏幕第 1 行和第 1 列的字段内容。EIBAID 字段被设置为 DFHPEN。
  - 如果未选取注意字段，那么 CICS 接收所有字段的内容。EIBAID 字段被设置为 DFHENTER。

使用 DFHWBIMG 显示图形

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

CICS 提供了以下可用于 Web 应用程序中的图形。

关于此任务

CICS.GIF
CICS 徽标

MASTHEAD.GIF
具有文本“CICS Web Interface”的 CICS 徽标

BACKGROUND1.GIF
包含“CICS”字符的背景

BACKGROUND2.GIF
包含“CWI”字符的背景

TEXTURE1.JPEG
纹理背景

TEXTURE2.JPEG
纹理背景

TEXTURE3.JPEG
纹理背景

TEXTURE4.JPEG
纹理背景

TEXTURE5.JPEG
纹理背景

TEXTURE6.JPEG
纹理背景

要在 Web 浏览器中显示图形，输入的 URL（在转换到大写后）的路径应该是以下格式:

```
/DFHWBIMG/filename
```

其中 **filename** 列出的一个图形的名称。例如:

```
 /DFHWBIMG/Texture1.jpeg
```

要在您的输出中合并任何图形，将路径包含在相应的 HTML 标记中。例如，您可以用下面的标记包含一幅纹理背景:

```
<body background="/DFHWBIMG/background1.gif" ... >
```

CICS 将路径以“/DFHWBIMG”开头的 HTTP 请求作为特殊情况处理；不调用分析器，而 DFHWBIMG 作为转换器程序运行。

CICS 在用于 CICS 提供的事务的模板中使用这些图形的一部分。

CICS 提供的图形作为 DFHWBIMG 的一部分硬编码，且不可以作为独立的文件提供；DFHWBIMG 不支持图形显示（除了指定的那些以外）。

第 16 章 从 BMS 定义创建 HTML 模板

如果希望从您不具有其源代码的现有 BMS 映射集创建 HTML 模板，可以从相应的装入模块中重新构造源代码。

关于此任务

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

使用 BMS 宏生成实用程序（DFHBMSUP），在 *CICS Operations and Utilities Guide* 中的 BMS 宏生成实用程序中描述了该实用程序。

CICS 为安装从 BMS 映射集中创建的 HTML 模板提供了编目的过程 DFHMAPT。请参阅 *CICS Application Programming Guide* 中的安装映射集和分区集，以获取详细信息。

有关 BMS 生成的模板

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

通过 BMS 映射生成的模板包含: 常量和输入字段、按钮、隐藏变量、JavaScript 函数和 JavaScript 异常处理程序。

- 映射中的常量和输入字段
- 表示以下内容的按钮:
  - ENTER 和 CLEAR 键
  - PA1、PA2 和 PA3 键
  - 程序功能键 PF1 到 PF24
  - HTML 复位
- 最多 5 个隐藏的变量，`DFH_NEXTTRANSID.1` 到 `DFH_NEXTTRANSID.5`，它们的值是映射中前 5 个字段的名称。这些变量的使用在 第 169 页的第 15 章，『CICS Web Support 和 3270 显示应用程序』中说明。
- 隐藏的变量 `DFH_CURSOR`，它的值是在映射中设置了光标的字段的名称。如果光标位于一个未命名的字段上，那么 `DFH_CURSOR` 为零。
- JavaScript 函数 `dfhsetcursor()`。当 `DFH_CURSOR` 包含字段名时，函数将光标位置设置到该字段。
- 用于 `onFocus` 异常的 JavaScript 异常处理程序。这个函数调用 `dfhsetcursor`，并跟踪光标的移动。

生成定制的 HTML 模板

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

可以使用以下三种方法定制从 BMS 映射生成的 HTML 模板。

关于此任务

- 可以通过对自己的 DFHMSX 宏版本进行编码来修改从 BMS 映射生成 HTML 模板的方式。

- 可以通过在 BMS 映射定义中使用 DFHWBOUT 宏向生成的映射添加 HTML 文本
- 可以手工编辑生成的 HTML。在以下情况下这会有用:
  - 您要覆盖当程序发出 MAP SEND 命令时发生的属性动态更改。
  - 您要在 Web 3270 环境之外使用 HTML 模板。

  在这两种情况中，您都将需要更改 *Frrcccllll* 变量，而这些变量是通过模板生成过程来添加的。

强烈建议不要编辑 CICS 生成的 HTML 模板，除非所有 SEND MAP 命令都使用 ERASE 选项。在 CICS 运行时，不带 ERASE 的 SEND MAP 命令会导致 HTML 合并。运行时逻辑期望遇到 CICS 模板生成器生成的 HTML。尤其应该避免更改 `<tr>` 标记。

有关定制的模板示例，请参阅第 197 页的『定制示例』。

CICS 为以下 CICS 提供的事务（使用 BMS）提供 HTML 模板:
CETR

提供的模板使用 EBCDIC Latin 字符集（代码页 037）。如果您在 CICS 系统中使用不同的代码页，那么必须生成这些模板的您自己的版本: CICS 生成的头部分中使用的以下字符在不同于 037 的代码页中有不同的表示法:

`! [ ] { }`

使用 DFHMDX 宏上的 **CODEPAGE** 参数指定代码页。

使用 DFHMSX 宏来定制

可以通过对自己的 DFHMSX 宏版本进行编码来修改从 BMS 映射生成 HTML 模板的方式。

关于此任务

您可以指定:

- 由按钮表示的 3270 键
- 显示在每个按钮上的文本或图像
- HTML 页面的标题
- 要在 HTML 页面顶部显示的报头图形
- 作为图形文件或颜色的页面背景
- 常规文本、未访问的链接、已访问的链接和活动链接的颜色
- 页面是否应该包含 HTML 复位按钮，以及显示在上面的文本
- BMS 映射中使用的颜色和用于 HTML 模板中相应文本的颜色之间的映射
- HTML 页面中应该禁止哪些 BMS 字段
- JavaScript `onLoad()` 和 `onUnload()` 异常处理程序
- 模板中的文本是以比例字体显示还是以非比例字体显示
- 生成模板时要使用的代码页，以及用于特殊字符 `[]{}` 和 `!` 的代码点
- 受保护字段应该在 HTML 页面中右对齐

注:

1. BMS 字段的 ATTRB=BRT 选项对于未命名、未受保护的（输入）字段不起作用。
2. 如果逻辑映射中命名的字段被清空（例如，使用 DEL 键）或该字段在以前的 SEND 命令上已经为空（空或空格），并关闭了字段的“修改数据标记（MDT）”，那么不设置 DFHBMEOF（字段的属性字节的 3270 属性位）。

对自己的 DFHMSX 宏版本进行编码时，可以指定将编码的选项应用于:

- 所有映射集中的所有映射
- 特定映射集中的所有映射
- 个别映射

安装 HTML 模板

关于此任务

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

过程如下:

1. 复查 CICS 应用程序及其对 BMS 的使用，了解定制是否必要。
2. 对于需要定制的 HTML 页面的应用程序，创建一个定制宏定义，并将它存储在与汇编程序 SYSLIB DD 语句中指定的宏库并置的库中。编写合适的 DFHWBOUT 宏调用，然后将它们放在映射定义中合适的地方。
3. 在 DFHMSD 宏上使用 TYPE=TEMPLATE，或者在传递给汇编程序的参数中使用 SYSPARM=TEMPLATE 汇编现有的映射定义。请注意，DFHMSD 宏上的标签用于命名为处理的映射集中每个映射生成的 HTML 模板。HTML 模板名由 DFHMSD 宏的标签组成，后跟用字符 A-Z 和 0–9 生成的一到两个字符的后缀。当映射集中有多于 36 个映射时使用两个字符的后缀，并且在这种情况下，映射集名必须是六个字符或更少。 在程序发出 BMS SEND 或 RECEIVE 时，为了使网桥出口与带 BMS 映射的 HTML 模板相匹配，HTML 模板成员必须与在 SEND 和 RECEIVE 语句上使用的映射集值的名称匹配。如果使用的是定制宏，必须向 TYPE 添加定制宏的名称。汇编程序生成 IEBUPDTE 源代码语句，为映射集中的每个映射建立一个模板。
4. 使用 IEBUPDTE 将模板存储在模板库中。如果模板库的记录格式不是固定的块格式，需要将它们存储在另一个分区的数据集中，然后使用诸如 ISPF COPY 这样的命令将它们转换成模板库的记录格式。
5. 如果您希望将模板放在不是 DFHHTML DDname 中指定的一个已分区的数据集中，那么必须为您的模板定义 DOCTEMPLATE 定义，并指定备用 DDname。备用 DDname 还必须在 CICS JCL 中指定。

   要将包含模板的分区数据集分配给特定的 DDname 以从中安装模板，可以使用 ADYN 样本事务。首先安装包含 ADYN 及其相关程序的 DFH$UTIL 组，然后运行 ADYN:

   ```
   ADYN
   ALLOC DDNAME(ddname) DATASET('template-pds') STATUS(SHR)
   ```

   其中 *ddname* 是在 DOCTEMPLATE 定义中指定的 DDname，*template-pds* 是包含要安装的模板的分区数据集的名称。要获取有关安装和使用 ADYN 的更多信息，请参阅 *CICS Customization Guide*。

结果

HTML 模板的大小限制

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

不存在对使用 3270 网桥运行的事务所使用模板大小的限制。然而，存储超过 32K 的模板与较小的模板的处理方式不同。

关于此任务

要处理大于 32K 的模板，您必须指定映射到 HTTP 请求中 DFHWBTTB 的程序名的路径。要获取更多信息，请参阅 第 171 页的『3270 显示应用程序的 URL 路径部分』。

注意，如果符号替换显著增加了数据量，那么所需存储小于 32K 的模板可将存储扩展到大于 32K。

生成模板时，DFHWBTLG 发出的消息包含要从 DFHHTML 数据集读取的每个模板所需的存储量。使用这些消息确定您是否应该使用 DFHWBTTA 或 DFHWBTTB 的程序名。

编写定制宏定义

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

必须提供由 CICS 提供的汇编程序宏调用的完整汇编程序宏定义。必须根据汇编程序宏定义的规则编写定制宏的定义。定义中的宏调用也必须遵循汇编程序语言宏语句的规则。

关于此任务

定制宏定义包含以下元素:

1. 一个开始定义的 MACRO 语句。
2. 宏的名称。
3. 任意数量的 DFHMDX 宏的调用。

   在 第 191 页的『DFHMDX 宏』中描述了 DFHMDX 的语法，在第 197 页的『定制示例』中描述了它的使用。DFHMDX 是从 DFHMSX 中调用的。
4. 一个结束定义的 MEND 语句。

处理空格

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

定制宏定义时，必须考虑对于空白的 HTML 规范。对于 3270 终端，空格（EBCDIC X'40'）和空（EBCDIC X'00'）可用于格式化屏幕数据位置。将这样的数据流转换成 HTML 时，客户机对它的解释所生成的输出与在 3270 终端上找到的不一样。

关于此任务

如果空格字符串后紧跟着一个开始标记，客户机就忽略它，并将所有后续的连续空格序列解释成一个空格。要强制呈现所有空格，您可使用 <pre> 和 </pre> 标记。

没有指定空字符的处理，客户机处理它们的方式也不一致。它们可能显示，也可能不显示。

组合 BMS 和非 BMS 输出

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

事务可以发出一系列 BMS 和非 BMS 命令以构建 3270 显示屏幕的内容。

本主题解释了如何组合所有命令的输出以构造显示在 Web 浏览器上的 HTML 页面。这些步骤为:

1. 当发出 BMS 或非 BMS SEND 命令时，生成一个 HTML 页面（包含一个标题部分、一个屏幕图像部分和一个页脚部分），但不发送给 Web 客户机。
2. 当事务发出 RECEIVE 命令或终止时:
   - 从以前生成的标题部分中选择一个。
   - 通过将所有以前生成的屏幕图像部分合并起来，创建一个新的屏幕图像部分。
   - 从以前生成的页脚部分中选择一个。
3. 结果 HTML 页面发送给 Web 客户机。

如何选择标题部分

根据 HTML 页面的起始位置和创建顺序，从它们中选择标题部分。

1. 选择起始位置最接近屏幕第一行的页面。
2. 如果在选择过程中留有多个页面，那么再次比较其余页面的起始位置。这次，选择起始位置最接近屏幕第一列的页面。
3. 最后，如果还留有多个页面，那么选择最早生成的页面。

剩下的那个选择页面中的标题部分用在发送给 Web 客户机的 HTML 页面中。

注: 在该描述中:

- 从 BMS 映射中生成的 HTML 页面的起始位置是映射左上角的行和列。
- 从非 BMS 命令中生成的 HTML 页面的起始位置是屏幕的左上角（第 1 行，第 1 列）。

如何选择页脚部分

根据 HTML 页面的结束位置和创建顺序，从它们中选择页脚部分。

1. 选择结束位置最接近屏幕最后一行的页面。
2. 如果在选择过程中留有多个页面，那么再次比较其余页面的结束位置。这次，选择结束位置最接近屏幕最后一列的页面。
3. 最后，如果还留有多个页面，那么选择最后生成的页面。

剩下的那个选择页面的页脚部分用在发送给 Web 客户机的 HTML 页面中。

注: 在该描述中:

- 从 BMS 映射中生成的 HTML 页面的结束位置是映射右下角的行和列。
- 从非 BMS 命令中生成的 HTML 页面的结束位置是屏幕的右下角。

如何合并屏幕图像部分

合并了通过一系列 BMS 和非 BMS SEND 命令创建的屏幕图像部分后，就会创建一个新的屏幕图像部分；它尽可能地包含用于构造它的所有屏幕图像部分的所有字段。

然而，如果来自两个或多个构成屏幕图像的字段被完全或部分重叠的话，就不可能发生这种情况，可能修改或完全禁止某些重叠字段:

- 字段重叠时，与早期 BMS 或非 BMS SEND 命令相关的字段会被修改或禁止而保留以后的命令。
- 如果输入字段被部分或完全重叠，就丢弃整个输入字段，并且不会在最终的 HTML 中出现。
- 如果输入字段与一些常规文本部分重叠，那么无论在 3270 设备上输入字段结尾之后是否还有更多文本可见，到输入字段开始为止的任何可见文本在最终 HTML 中都是可见的，并丢弃其余数据。
- 如果表单元格包含水平规则标记（<hr>），那么重叠单元格内容将产生不可预测的结果。

表 7 中总结了这些规则。

表 7. 合并的屏幕图像部分中的重叠字段

早期 SEND 生成的字段	稍后 SEND 生成的字段	结果
输入（未保护）	输入（未保护）或文本（受保护）	完全禁止早期字段
文本（受保护）	输入（未保护）	基于 3270 屏幕中的字符位置: • 保留输入字段左侧的受保护字符。 • 禁止由输入字段覆盖的受保护字符。 • 禁止输入字段右侧的受保护字符。
文本（受保护）	文本（受保护）	根据 3270 屏幕中的字符位置，稍后 SEND 命令生成的字符将覆盖早期 SEND 命令生成的字符。

在应用程序中使用从 BMS 映射中创建的 HTML 模板之前可以编辑它们。然而，屏幕图像部分合并所依据的算法需要部分中的 HTML 具有特定的结构。因此，当编辑模板中的屏幕图像部分，并且该部分将与其他部分合并时，您必须遵循一些准则:

- 每个 HTML 页面都包含两个分别带有字符串 DFHROW 和 DFHCOL 的注释。跟随在这些字符串后的值在合并过程中非常重要，因为它们用于计算 3270 屏幕上每个字段的位置。如果这些注释被修改或删除，将不合并屏幕图像部分，但出现在最终 HTML 页面上，一个附加在另一个的下面。

- 表单元格的结束标记（</td>）表行的结束标记（</tr>）都是可选的。
- 表单元格必须包含带有或不带有额外属性标记的一段常规文本，或者必须包含一个输入字段。另外，如果文本和输入字段一个接一个，并且它们之间没有额外的标记的话，它们可以在同一个表单元格中包含混合的文本和输入字段。
- 空的表单元格在开始标记和结束标记之间可能不包含空值（X'00'）或空格。换句话说，必须将空单元编码为 <td></td>。
- 可以使用以下一对或多对标记将一段文本或一个输入字段部分括起:

  强调 ...

  加强 ...

  字体 ...

  下划线 <u> ... </u>

  闪烁 <blink> ... </blink>

  如果使用这些标记，必须确保每个标记都有相应的结束标记。还必须确保正确嵌套了开始和结束标记。例如，<u> ... </u> 嵌套是正确的，但 <u> ... </u> 嵌套得不正确。
- 如果将其他标记插入表单元格，那么在同一个表单元格中，它们必须在文本或输入字段之前或之后出现，不能同时出现在文本或输入字段之前和之后。
- 允许在表单元格中出现 HTML 注释，它可以出现在一段常规文本或输入字段之前、之后或两侧。
- 如果单元格包含注释，它还必须包含一段常规文本或一个输入字段。

DFHMDX 宏

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

DFHMDX 宏是从 DFHMSX 中调用的。

第 192 页的图 10 中显示了它的语法。

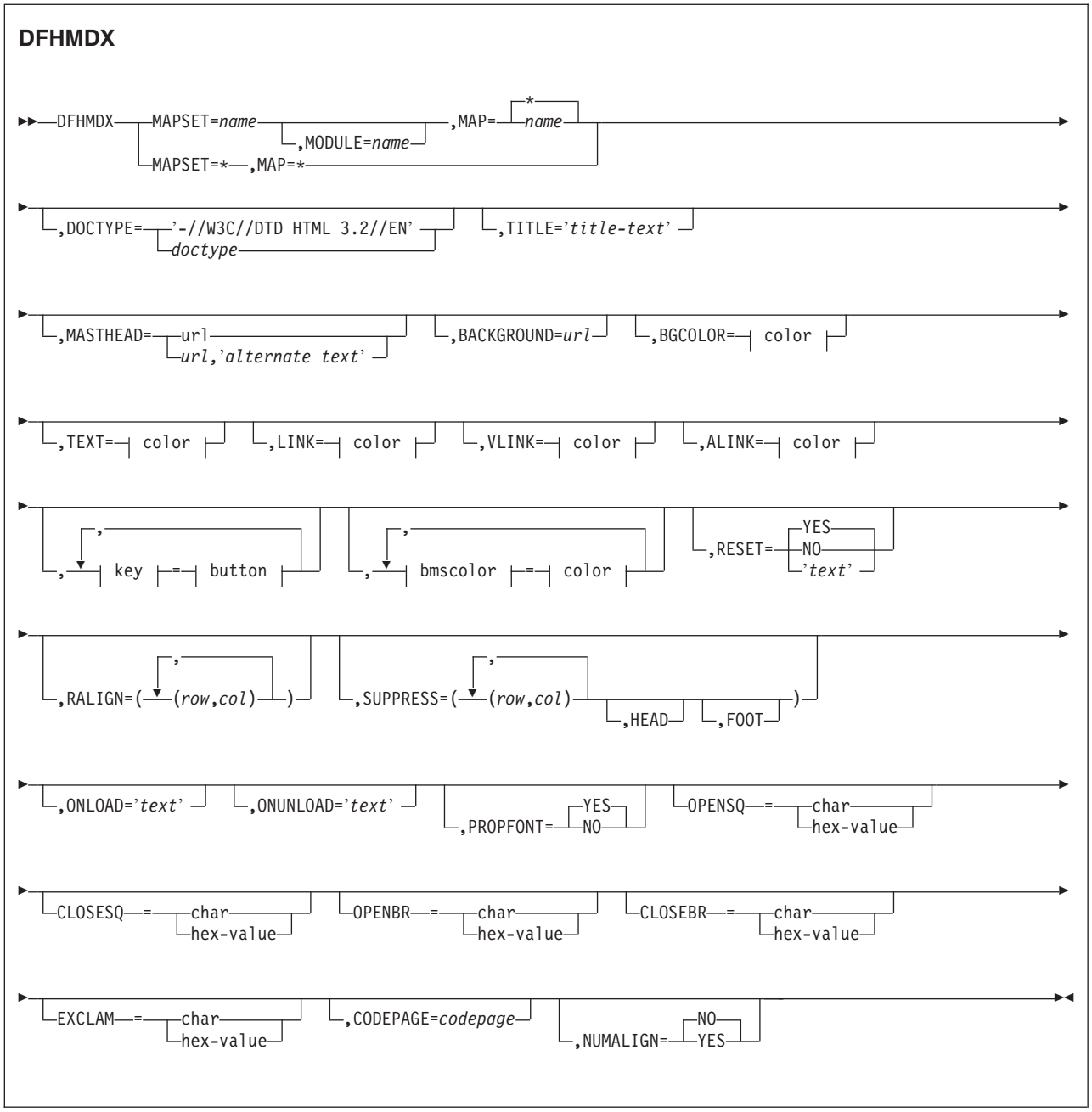


图 10. DFHMDX 的语法

这个宏的关键字参数可以以任何顺序出现。

MAPSET

指定包含其他选项所引用的映射的映射集的名称。如果指定星号，选项对于所有以后的映射集来说就成为缺省值。

MODULE

指定将映射集链接编辑到的装入模块的名称。如果未指定 MAPSET=*，那么只能使用这个参数。通过添加单一字符后缀，指定的名称（只能是七个字符）用于构造模板的名称。缺省值是映射集的名称。

MAP 指定选项引用的映射的名称，该映射位于 MAPSET 中指定的映射集里。如果指定星号，选项对于所有以后的映射来说就成为缺省值。

DOCTYPE
指定希望出现在 HTML 模板中的 <!doctype> 标记的 DTD 公共标识部分。缺省值是 -//W3C//DTD HTML 3.2//EN，它指定 HTML 3.2。在某些 HTML 标记中，颜色支持需要级别 3.2。

TITLE 指定要用作 HTML 标题以及第一个 <h1> 标记内容的标题。

MASTHEAD
指定在页标题上出现的报头图形的 URL，它在第一个 <h1> 标记之前。如果提供*备用*文本，那么如果它无法装入指定的图形，客户机将使用文本。

BACKGROUND
指定用于页面背景的图形文件的 URL。

BGCOLOR
指定页面背景的颜色。

TEXT 指定常规文本的颜色。

LINK 指定页面上未访问超文本链接的颜色。

VLINK 指定页面上已访问超文本链接的颜色。

ALINK
指定页面上已激活的超文本链接的颜色。

PF1-PF24
指定为相应 3270 程序功能键分配给仿真按钮的名称或图像。

PA1-PA3
指定为相应 3270 程序辅助操作请求键分配给仿真按钮的名称或图像。

CLEAR
指定为 3270 Clear 键分配给仿真按钮的名称或图像。

ENTER
指定为 3270 Enter 键分配给仿真按钮的名称或图像。

PEN 指定为笔型选择分配给仿真按钮的名称或图像。

BLUE 指定要出现在已在 BMS 映射中指定了蓝色的 HTML 页面上的颜色。缺省值是 #0000FF。

限制：DFHMDX 将只覆盖未命名字段的颜色；它保留命名字段未更改。

GREEN
指定要出现在已在 BMS 映射中指定了绿色的 HTML 页面上的颜色。缺省值是 #008000。

限制：DFHMDX 将只覆盖未命名字段的颜色；它保留命名字段未更改。

NEUTRAL
指定要出现在已在 BMS 映射中指定了中间色的 HTML 页面上的颜色。缺省值是 #000000。

限制：DFHMDX 将只覆盖未命名字段的颜色；它保留命名字段未更改。

PINK 指定要出现在已在 BMS 映射中指定了粉色的 HTML 页面上的颜色。缺省值是 #FF00FF。

限制: DFHMDX 将只覆盖未命名字段的颜色；它保留命名字段未更改。

RED 指定要出现在已在 BMS 映射中指定了红色的 HTML 页面上的颜色。缺省值是 #FF0000。

限制: DFHMDX 将只覆盖未命名字段的颜色；它保留命名字段未更改。

TURQUOISE

指定要出现在已在 BMP 映射中指定了青绿色的 HTML 页面上的颜色。缺省值是 #00FFFF。

限制: DFHMDX 将只覆盖未命名字段的颜色；它保留命名字段未更改。

YELLOW

指定要出现在已在 BMS 映射中指定了黄色的 HTML 页面上的颜色。缺省值是 #FFFF00。

限制: DFHMDX 将只覆盖未命名字段的颜色；它保留命名字段未更改。

RESET

指定是否支持 HTML 复位功能。指定 YES 可以获得一个带有缺省图注 Reset 的缺省复位按钮。指定 NO 不会获得复位按钮。使用自己的图注为复位按钮指定自己的文本。

RALIGN

指定 BMS 映射字段中的数据将在 HTML 页面中向右对齐。指定的值 *rr* 和 *cc* 必须与 DFHMDF 宏的 POS=(*rr*,*cc*) 规范对应，以使字段向右对齐。每一对都必须用圆括号括起，各对的完整列表也必须用圆括号括起。如果希望以特定列结束的每个限定字段都向右对齐，指定结束列号，然后为行规范输入星号。通过将字段的开始列号添加到 LENGTH 中来计算字段的结束列号，如 DFHMDF 宏中定义的那样。只有字段是在受保护、未命名并在 DFHMDF 宏中使用 INITIAL、XINIT 或 GINIT 值初始化的情况下才能右对齐。如果使用 MAP=* 或 MAPSET=* 指定 RALIGN 参数，那么忽略它。

如果希望在宏定义中指定一个列表，而此列表的长度超过汇编程序对于字符串长度的限制（256 个字符），那么使用相同的 MAPSET 和 MAP 值对额外的 DFHMDX 宏进行编码，并将更多的值放入 RALIGN 参数中。

SUPPRESS

指定不出现在 HTML 页面中的 BMS 映射字段。为要禁止的字段的起始位置指定任何号码的行列对。指定的值 *rr* 和 *cc* 必须与 DFHMDF 宏的 POS=(*rr*,*cc*) 规范对应，以禁止字段。每一对都必须用圆括号括起，各对的完整列表也必须用圆括号括起。如果希望禁止一行中的所有字段，指定行号，然后为列规范输入星号。如果使用 MAP=* 或 MAPSET=* 指定 SUPPRESS 参数，那么忽略它。

使用关键字 HEAD 禁止模板中的标题部分。使用关键字 FOOT 禁止模板中的页脚部分。

如果希望在宏定义中指定一个列表，而此列表的长度超过汇编程序对于字符串长度的限制（256 个字符），请使用相同的 MAPSET 和 MAP 值对额外的 DFHMDX 宏进行编码，并将更多值放入 SUPPRESS 参数中。

ONLOAD

指定要用于替换 HTML 页面的标准 `onLoad` 异常处理程序的 JavaScript 文本。

文本不能包含双引号（"），必须按照常规的汇编语言约定将单引号（'）加倍（''）。如果使用这个参数，那么将光标的设置禁止到由 DFH_CURSOR 标出的字段，DFH_CURSOR 是由标准 onLoad 异常处理程序提供的。可以使用函数 dfhsetcursor 来设置光标位置。

ONUNLOAD

指定要用作 HTML 页面的 onUnload 异常处理程序的 JavaScript 文本。文本不能包含双引号（"），必须按照常规的汇编语言约定将单引号（'）加倍（''）。

PROPFONT

指定字体。如果指定 YES，模板指定文本将以比例字体出现，并将连续空格减少到一个空格。如果指定 NO，模板指定文本以固定间距字体指定，并保留连续空格。

OPENSQ

用于显示左方括号的十六进制值或字符。缺省值是 X'BA'（代码页 37）。

CLOSESQ

用于显示右方括号的十六进制值或字符。缺省值是 X'BB'（代码页 37）。

OPENBR

用于显示左大括号的十六进制值或字符。缺省值是 X'C0'（代码页 37）。

CLOSEBR

用于显示右大括号的十六进制值或字符。缺省值是 X'D0'（代码页 37）。

EXCLAM

用于显示感叹号的十六进制值或字符。缺省值是 X'5A'（代码页 37）。

CODEPAGE

指定用于编码模板生成进程所生成的所有文本的 IBM 代码页号。这个代码页必须与 CICS 使用模板时使用的代码页匹配，使用的代码页在 EXEC CICS DOCUMENT 命令的 HOSTCODEPAGE 选项或分析器程序选择的 DFHCNV 宏的 SRVERCP 选项中。

主机代码页名称的标准 CICS 格式由使用 3 到 5 个十进制数字编写而成的代码页编号（或更为普通的 CCSID）组成，并根据需要，尾部可以填充空格至 8 个字符。对于少于 3 位数字的代码页 37，标准格式是 037。CICS 接受范围从 1 到 65535 的任何十进制数字（最多 8 位，尾部用空格填充）作为代码页名称，即使其格式不是标准的，也是如该。

由于用于符号和符号列表处理的定界符被假定为 EBCDIC 格式，因此如果需要处理任何符号，那么 CODEPAGE 参数必须指定基于 EBCDIC 的代码页。

缺省代码页是 037。

NUMALIGN

指定在 HTML 模板中的表单元中如何对齐 DFHMDF 宏中明确定义为数字的字段:

NO 指定在表单元中不右对齐的数字字段。这是缺省值。

YES 指定在表单元中右对齐的数字字段:

- 对于保护字段，生成的 HTML 文本在单元中右对齐。如果文本包含多个尾部空格，那么可能不保存它们: 某些客户机将使用单个空格替换它们。

注: RALIGN 参数保存尾部多个空格；NUMALIGN 参数不保存。如果两个参数都应用于一个字段（即，如果数字字段由 RALIGN 参数标识，而且指定了 NUMALIGN=YES），那么不保存尾部的多个空格。

- 对于未保护字段，HTML 文本输入元素（但不是元素中的文本）在单元中右对齐。

color 可以是显式规范 *#rrggbb*，其中 *rr*、*gg* 和 *bb* 是 2 位十六进制数字，它们提供了在所请求颜色的红色、绿色和蓝色的强度，它也可以是以下颜色名称中的任何一个：AQUA、BLACK、BLUE、FUCHSIA、GRAY、GREEN、LIME、MAROON、NAVY、OLIVE、PURPLE、RED、SILVER、TEAL、WHITE 和 YELLOW。

key 可以是 PF1 到 PF24、PA1 到 PA3、CLEAR、ENTER 和 PEN 中的任何一个。

button 可以是 (IMAGE,*url*)，其中 *url* 指定要用于按钮的图象的 URL，或者是"*text*"，其中 *text* 是要放在按钮上的文本，如果按钮不出现，那么为 NO。

bmscolor 可以是 BLUE、GREEN、NEUTRAL、PINK、RED、TURQUOISE 和 YELLOW 中的任何一个。

使用 DFHWBOUT 宏定制模板

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

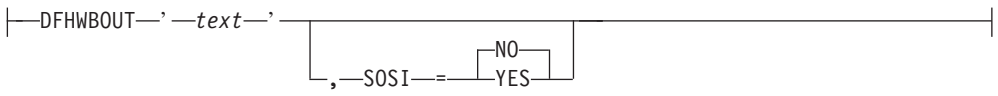
DFHWBOUT 宏用于向从 BMS 映射生成的 HTML 页面添加文本。文本仅作为 HTML 页面的一部分出现。如果在映射中，宏在 DFHMDF 第一次出现之前使用，那么将文本放在 HTML 页面的 <head> 部分。如果宏用在映射中的其他地方，那么文本紧跟着前面的 DFHMDF 宏生成的文本放置。

关于此任务

当应用程序使用多个 BMS 映射构建屏幕显示时，不要使用 DFHWBOUT 宏。

DFHWBOUT

DFHWBOUT 宏



这个宏的参数如下：

text 要插入 HTML 页面中的文本。

SOSI 文本是否包含由 shift-out（X'0E'）和 shift-in（X'0F'）定界的 DBCS 字符。缺省值是 SOSI=NO。

在使用 DFHWBOUT 宏时，要知道插入的 HTML 文本可能影响从 BMS 映射字段中生成的页面布局。可能需要调整插入的文本来确保正确的页面布局。

定制示例

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

下面的样本显示了定制宏定义。通过为映射集名和映射名指定 *, DFHMDX 的第一个调用为应用到 DFHMDX 后面的调用设置缺省值。稍后的调用对映射集中特定的映射重设或添加参数。接续字符位于第 72 列，接下去的文本在第 16 列继续。

关于此任务

```
         MACRO
         DFHMSX
         DFHMDX MAPSET=*,MAP=*,                                        *
       PF1='Help',PF3='Exit',PF4='Save',PF9='Messages'
         DFHMDX MAPSET=DFHWB0,MAP=*,                                   *
                     TITLE='CICS Web Interface',                       *
                     PF3='Messages'
         DFHMDX MAPSET=DFHWB0,MAP=DFHWB02,                             *
                     TITLE='CICS Web Interface Enable',                *
                     PF3='Save'
               MEND
```

当 CICS 为每个 BMS 映射定义创建模板时，它调用在 DFHMAPT 过程中的 SYSPARM 参数上指定的定制宏。如果 SYSPARM 参数未指定定制宏名称，那么将使用 DFHMSX。按顺序处理每个宏，并存储参数值（如果存在）。为特定映射或映射集指定重复参数的情况下，新值只替换该映射或映射集的先前值。

- 本示例中的第一个 DFHMDX 宏指定 `MAPSET=*,MAP=*` 和 `PF3='Exit'`。PF3 的这个值应用于每个映射集和在后继 DFHMDX 宏中未指定不同值的映射。
- 第二个 DFHMDX 宏指定 `MAPSET=DFHWB0,MAP=*` 和 `PF3='Messages'`。PF3 的这个值应用于在后继 DFHMDX 宏中未指定不同值的映射集 DFHWBO 中的每个映射。
- 第三个 DFHMDX 宏指定 `MAPSET=DFHWB0,MAP=DFHWB02` 和 `PF3='Save'`。该值仅应用于映射集 DFHWBO 中的映射 DFHWB02。

从 BMS 映射中生成的缺省模板包含表示以下所有键的按钮:

- ENTER 和 CLEAR 键
- PA1、PA2 和 PA3 键
- 程序功能键 PF1 到 PF24
- HTML 复位

然而，如果使用 DFHMDX 宏指定模板中希望存在的按钮，那么只有指定的按钮才包含在模板中。例如，如果您指定

```
DFHMDX MAPSET=*,MAP=*,PF3='Exit',ENTER='Continue'
```

则模板将只包含 PF3 和 Enter 键的按钮。

这里有进一步的示例显示可以如何定制从 BMS 映射生成的 HTML 模板。

- **支持应用程序使用不是标准输出中的键。**

  可以按如下方式向映射 AD001 添加按钮:

  ```
  DFHMDX MAP=AD001,PF1xx='Resubmit'
  ```

  其中 PFxx 是您希望指定的新 PF 键号码。Web 客户机显示带有图注“Resubmit”的按钮。如果用户单击该按钮，它就会作为 PFxx 报告给应用程序。

- **禁止 HTML Reset 功能。**

可以按如下方式禁止映射 AD001 的 Reset 功能:

```
DFHMDX MAP=AD001,RESET=NO
```

Web 客户机显示不包含 Reset 按钮的页面。

- **更改按钮的外观或与它们关联的文本。**

  可以按如下方式示更改 PF1 按钮上的图注:

  ```
  DFHMDX PF1='Help'
  ```

  Web 客户机显示具有图注“Help”的按钮。如果用户单击该按钮，它就作为 PF1 提供给应用程序。

- **为 HTML 页面提供 HTML 标题。**

  可以按如下方式向显示的映射添加标题:

  ```
  DFHMDX MAP=DFHWB01,TITLE='CICS Web Interface'
  ```

  Web 客户机显示“CICS Web 接口”作为页面标题。

- **为 HTML 页面提供报头图形。**

  为将具有头的映射编写 DFHMDX 宏。例如:

  ```
  DFHMDX MASTHEAD=(/dfhwbimg/masthead.gif,'CWI')
  ```

  Web 客户机使用指定的头，如果找不到图形文件，那么显示“CWI”作为头。

- **更改背景的颜色，或指定特殊背景。**

  为将具有特殊背景的映射编写 DFHMDX 宏。例如:

  ```
  DFHMDX MAP=AD001,BACKGROUND=/dfhwbimg/texture4.jpeg
  ```

  Web 客户机使用指定的文件作为页面背景。

  要更改背景的颜色，使用 BGCOLOR 参数。

- **修改 BMS 颜色。**

  要修改 BMS 颜色，编写类似于以下内容的 DFHMDX 宏:

  ```
  DFHMDX MAP=AD001,BLUE=AQUA,YELLOW=#FF8000
  ```

  Web 客户机以 HTML 浅绿色（与 BMS 青绿色相同）显示 BMS 蓝色文本，以亮橙色显示 BMS 黄色文本。

- **禁止 BMS 映射的某些部分。**

  可以按如下方式禁止映射中的字段:

  ```
  DFHMDX MAP=AD001,SUPPRESS=((5,2),(6,2),(7,*))
  ```

  显示的页面不包含映射中第 5 行第 2 列的字段，也不包含第 6 行第 2 列的字段，也不包含第 7 行中的任何字段。

- **添加 Web 客户机控制函数。**

  如果希望在装入页面时调用 JavaScript 函数，在定制宏中使用 DFHMDX 宏的 ONLOAD 参数。例如，如果对以下内容进行编码:

  ```
  DFHMDX MAP=AD001,ONLOAD='jset(''CWI is wonderful'',''Hello there!'')'
  ```

则在装入页面时使用给定参数调用 JavaScript 函数 jset()。

要完成这种定制，必须使用 DFHWBOUT 宏将 jset 函数的定义添加到 HTML 页面的头中。必须将宏调用放在 BMS 映射定义中第一个 DFHMDF 宏之前。下面是一个样本:

```
DFHWBOUT '<script language="JavaScript">'
DFHWBOUT 'function jset(msg,wng)'
DFHWBOUT '         {window.status = msg; alert(wng)}'
DFHWBOUT '</script>'
```

装入页面时，窗口底部的状态区包含消息“CWI is wonderful”，并且打开一个包含消息“Hello there!”的警告窗口。

- **添加只出现在 HTML 页面上但不是 BMS 映射一部分的文本。**

  将 DFHWBOUT 宏放在 BMS 映射定义中希望文本出现的地方。例如:

```
DFHWBOUT '<p>This text illustrates the use of the DFHWBOUT macro,'
DFHWBOUT 'which can be used to output text that should only appear'
DFHWBOUT 'in HTML templates, and will never appear in the'
DFHWBOUT 'corresponding BMS map.'
```

  将在 HTML 模板中生成以下几行:

```
<p>This text illustrates the use of the DFHWBOUT macro,
which can be used to output text that should only appear
in HTML templates, and will never appear in the
corresponding BMS map.
```

- **向 HTML 页面添加 HTML 头信息。**

  将 DFHWBOUT 宏放在 BMS 映射定义中出现的第一个 DFHMDF 之前。例如:

```
DFHWBOUT '<meta name="author" content="E Phillips Oppenheim">'
 DFHWBOUT '<meta name="owner"  content="epoppenh@xxxxxxx.yyy.co*
       m">'
 DFHWBOUT '<meta name="review" content="19980101">'
 DFHWBOUT '<meta http-equiv="Last-Modified" content="&WBDATE&W*
       BTIME GMT">'
```

  将在 HTML 模板头部分中生成以下几行:

```
<meta name="author" content="E Phillips Oppenheim">
<meta name="owner"  content="epoppenh@xxxxxxx.yyy.com">
<meta name="review" content="19980101">
<meta http-equiv="Last-Modified" content="23-Dec-1997 12:06:46 GMT">
```

  DFHMSD 将 &WBDATE 和 &WBTIME 的值设置成汇编宏时的时间和日期。

- **在 JavaScript 和 HTML 中使用特定于国家或地区的字符。**

  用于生成模板的缺省美式英语代码页 37 可以修改为其他代码页。例如:

```
DFHMDX OPENSQ=[,CLOSESQ=],OPENBR={,CLOSEBR=},EXCLAM=!
```

  这指定需要的替换。字符必须在代码页与 DFHCNV 调用上的 SERVERCP 相应的终端上输入。

- **使字段在 HTML 页面中向右对齐。**

  可以按如下方式将字段中的数据向右对齐:

```
DFHMDX MAPSET=MAPSETA,MAP=AD001,RALIGN=((3,5),(*,15),(*,3),(6,7),(*,83))
```

在本示例中，下面所有字段中的数据都向右对齐

```
DFHMDF POS=(3,5),LENGTH=4,INITIAL='TEXT',ATTRB=PROT
DFHMDF POS=(5,80),LENGTH=3,INITIAL='123',ATTRB=PROT
DFHMDF POS=(2,10),LENGTH=5,INITIAL='  EXT',ATTRB=ASKIP
DFHMDF POS=(4,8),LENGTH=7,INITIAL='INITEX ',ATTRB=PROT
DFHMDF POS=(1,1),LENGTH=2,XINIT='C1C2',ATTRB=ASKIP
DFHMDF POS=(6,7),LENGTH=4,XINIT='0E44850F',ATTRB=PROT,SOSI=YES
DFHMDF POS=(2,9),LENGTH=6,XINIT='0E448544830F',SOSI=YES,ATTRB=PROT
DFHMDF POS=(2,9),LENGTH=6,XINIT='448544834040',PS=8,ATTRB=PROT
```

- **使数字字段右对齐**

  您可以使带有 NUMERIC 属性的所有字段在其 html 表单元中右对齐，如下所示:

  ```
  DFHMDX MAPSET=MAPSETA,MAP=AD001,NUMALIGN=YES
  ```

第 17 章 CICSPlex 中的 CICS Web Support

您可以单独或组合使用以下方法，在 CICSPlex 中分布使用 CICS Web Support 的应用程序:

- 您可使用网络负载均衡将来自 Web 客户机的请求分布到多个 CICS 区域。
- CICS Web Support 和业务应用程序可以在同一 CICS 区域中执行。
- CICS Web Support 可以在路由器区域中执行，而业务应用程序可以在一个或多个应用程序所属区域（AOR）中执行。然而，您无法在 AOR 中使用 EXEC CICS WEB API，因此支持 Web 的应用程序不能在 AOR 中执行。您可以在 AOR 中使用 EXEC CICS DOCUMENT API，但是您必须提供您自己的机制，将 HTML 输出传输回路由器区域。要获取更多信息，请参阅第 202 页的『将 Web 客户机的请求路由到 AOR』。

图 11 描述这些配置。

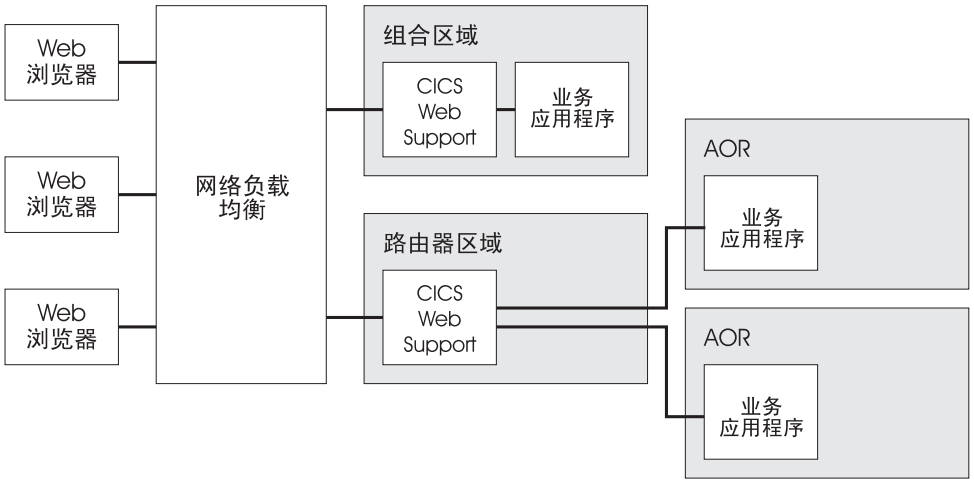


图 11. CICSPlex 中的 CICS Web Support 配置

您可以用相同的方式分布使用 CICS 业务逻辑接口的请求。这在第 202 页的图 12 中有说明。

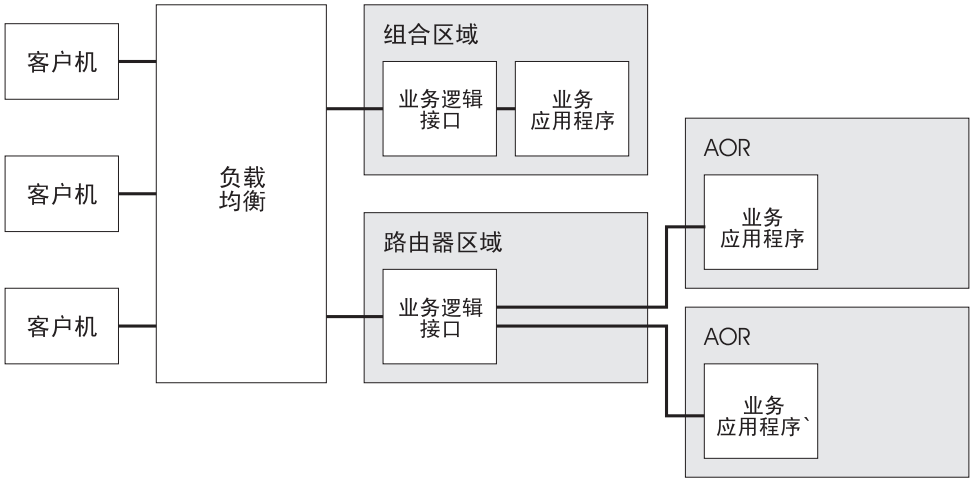


图 12. CICSPlex 中的 CICS 业务逻辑接口配置

当您计划以该方法分布应用程序时，您应该考虑应用程序各部分之间存在的任何亲缘关系。要获取有关亲缘关系的更多信息，请参阅 *CICS Interdependency Analyzer for z/OS User's Guide and Reference*。

您应考虑如何在请求之间管理应用程序的状态。第 80 页的『跨 HTTP 请求序列管理应用程序状态』讨论有关使用伪会话模型的任何 CICS Web Support 应用程序的注意事项。在以下情况下，可能存在其他注意事项:

- 动态路由选择用于选择执行业务应用程序的 AOR。
- 工作负载和连接平衡用于选择路由器区域（并间接地选择 AOR）。

CICS 提供您可用于管理应用程序状态的样本状态管理程序（DFH$WBSR）。DFH$WBSR 使通过资源共享应用程序状态变得容易，这些资源可以由多个 CICS 区域共享。它在第 389 页的附录 J，『状态管理样本 DFH$WBST 和 DFH$WBSR 的参考信息』中描述。（另一个样本 DFH$WBST 创建亲缘关系，因此它不适合在 CICSPlex 中使用。）

要获取有关在 CICSPlex 中配置 CICS Web Support 和 CICS 业务逻辑接口的指导，请参阅 *Workload Management for Web Access to CICS*。

将 Web 客户机的请求路由到 AOR

不能在 AOR 中使用 EXEC CICS WEB API，只能在路由器区域中使用它。

关于此任务

如果要使用支持 Web 的应用程序响应请求，那么一个解决方案是在支持 Web 的应用程序（它在路由器区域中执行）编码您的表示逻辑，并将您的业务逻辑（它在 AOR 中执行）编码为完全独立于演示。指定支持 Web 的应用程序作为处理请求的程序，并且它必须管理它自己与执行业务逻辑的应用程序的通信。第 25 页的『作为 HTTP 服务器的 CICS 的 HTTP 请求和响应处理』说明了支持 Web 的应用程序的处理阶段。

对于不支持 Web 的应用程序，路由器区域中的转换器程序可用于从 AOR 中的应用程序提供的信息生成 HTTP 响应。第 203 页的图 13 显示在应用程序所属区域中执行不支持 Web 的应用程序时，与来自 Web 客户机的请求关联的处理阶段和任务结构。

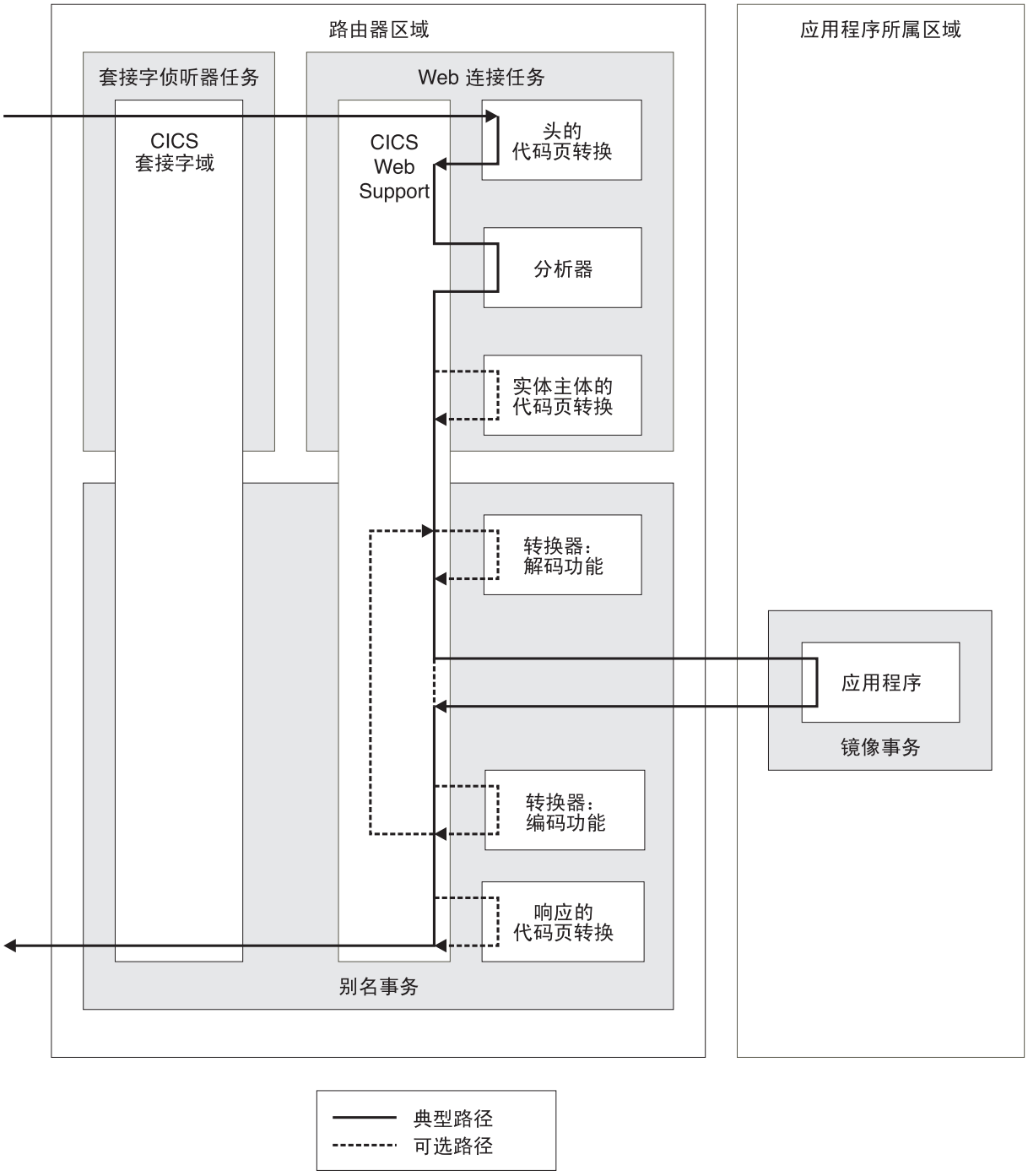


图 13. CICS Web Support 如何将不支持 Web 的应用程序请求路由到 AOR

CICS 业务逻辑接口的相应阶段在第 204 页的图 14 中显示。

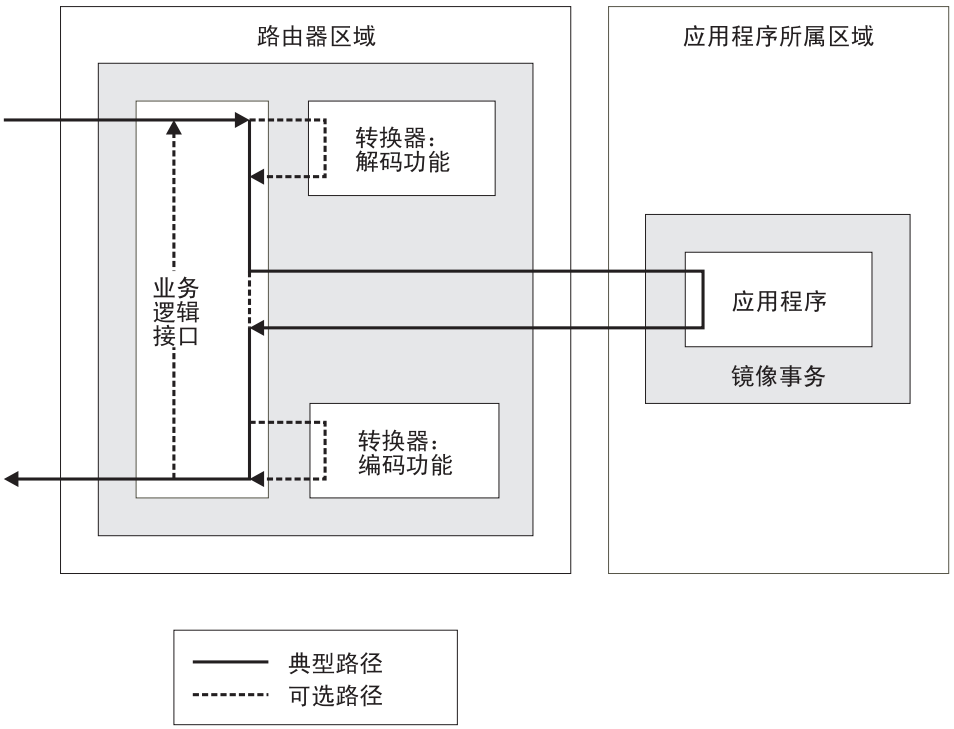


图 14. CICS 业务逻辑接口如何将应用程序请求路由到 AOR

CICS 使用分布式程序链接（DPL）调用 AOR 中的应用程序；应用程序在镜像任务下执行。要获取有关 DPL 的信息，请参阅 *CICS Intercommunication Guide*。

要在 AOR 中执行业务应用程序:

- 您必须指定 REMOTESYSTEM 属性，或在应用程序的 PROGRAM 定义中指定 DYNAMIC(YES)。如果您指定 DYNAMIC(YES)，动态路由选择程序确定执行应用程序的位置。
- 其他资源定义（用于分析器程序、支持 Web 的应用程序或转换器程序以及别名事务）必须指定它们在路由器区域中执行。
- 您必须定义路由器区域和 AOR 之间的 MRO 或 APPC 连接。

如果 AOR 中执行的应用程序设计为完全独立于演示，那么它将输出返回到支持 Web 的应用程序或转换器程序，然后它构造 HTML 输出。另外，如果您正在使用转换器程序，那么可能要在 AOR 中使用 EXEC CICS DOCUMENT API 以构造 HTML 输出。转换器程序可以使用该输出来生成一个完整的 HTTP 响应。

您必须提供您自己的机制，以将应用程序的输出传送回路由器区域。输出可以在通信区域中传输。另外，您还可以使用某些其他机制（例如，临时存储队列），并传输以该机制表示数据的令牌。路由器区域中的程序可以使用令牌获取输出，然后处理它并将它传递到 Web 客户机。CICS 提供您可用于执行该操作的样本状态管理程序（DFH$WBSR）。它在第 389 页的附录 J, 『状态管理样本 DFH$WBST 和 DFH$WBSR 的参考信息』中描述（其他样本 DFH$WBST 创建亲缘关系，所以不适合于在 CICSPlex 中使用）。

网络负载均衡

要避免依赖单个 CICS 路由器区域，请考虑使用多个路由器区域来分担网络中的工作负载。

关于此任务

有几项您可用于均衡路由器区域之间工作负载的技术:

综合系统分发器

Sysplex Distributor 是 z/OS Communications Server 的一种功能部件，它使用集群 IP 地址，这是一种特定于应用程序的动态虚拟 IP 地址（VIPA），用于表示综合系统（sysplex）中 CICS 服务器实例的集群。要获取有关综合系统分发器的更多信息，请参阅 *z/OS Communications Server: IP Configuration Guide*，SC31-8775。

特定于应用程序的虚拟 IP 地址（VIPA）

特定于应用程序的 VIPA 是 z/OS Communications Server 的一种功能部件，它提供激活绑定的虚拟 IP 地址。当 CICS 区域启动时，VIPA 在 TCP/IP 协议集上变成活动状态，并且 VIPA 可以通过特定的 CICS 区域在综合系统（sysplex）上到处移动。有关 VIPA 的更多信息，请参阅 *z/OS Communications Server: IP Configuration Guide*，SC31-8775。

DNS 方法

DNS 连接优化平衡 z/OS 综合系统 IP 域中的 IP 连接，这基于来自 MVS™ WLM 有关已注册应用程序运行状况的反馈。为了供 CICS 使用，它仍受支持。有关 DNS/WLM 支持的信息，请参阅 *z/OS Communications Server: IP Configuration Guide*，SC31-8775。

端口共享

TCP/IP 端口共享提供一种简单的方法，它可以将 HTTP 请求遍布在一组在同一 z/OS 映像中运行的 CICS 路由器区域上。不同区域中的 CICS TCPIPSERVICE 配置为在同一端口上侦听，而 TCP/IP 由 SHAREPORT 选项配置。然后，TCP/IP 协议集平衡跨侦听器的连接请求。要获取有关 TCP/IP 端口共享的更多信息，请参阅 *z/OS Communications Server: IP Configuration Reference*，SG24-5466。

第 3 部分 来自 CICS 的 Atom 订阅源 - 从此开始

CICS 可以向 Web 客户机提供 Atom 订阅源。Atom 订阅源由 CICS 资源或应用程序提供的数据组成。当您将 CICS 资源或应用程序作为 Atom 订阅源或集合列出时，用户便可以通过外部客户机应用程序（如订阅源阅读器或 Web mashup 应用程序）发出 HTTP 请求来阅读和更新数据。

CICS 提供了一些样本，可用于帮助您使用自己的数据设置 Atom 订阅源或集合。要了解有关 CICS 如何支持 Atom 订阅源以及 Web 客户机如何与这些订阅源交互的更多信息，请按照 CICS TS for z/OS V4.1 信息中心（`https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp`）内的 Web 2.0 案例“创建 Atom 订阅源以使用员工信息”中的指示信息，设置并使用该样本 Atom 集合。

第 18 章 Atom 订阅源

Web 订阅源（有时简称为“订阅源”）是内容提供方在因特网上发布的一系列相关项。Atom 订阅源是使用 Atom 联合格式和 Atom 发布协议的 Web 订阅源。

Atom 由一种用于描述 Atom 订阅源及其信息项的基于 XML 的格式和一个用于发布和编辑 Atom 订阅源的协议组成。因特网协会和 IETF（因特网工程任务组织）这两篇 RFC（请求评论）文档中描述了这种格式和协议:

“The Atom Syndication Format”（RFC 4287）的网址为: http://www.ietf.org/rfc/rfc4287.txt

“The Atom Publishing Protocol”（RFC 5023）的网址为: http://www.ietf.org/rfc/rfc5023.txt

内容提供方通常以称为 RSS（Really Simple Syndication）的较早格式来交付 Web 订阅源。CICS 支持 Atom，但不支持 RSS。

构成 Atom 订阅源的信息项被称为 *Atom 条目*。内容提供方可以发布或“联合”Atom 订阅源，方法是通过在因特网上以 URL 形式提供该 Atom 订阅源并用新项对其进行更新。Web 用户可以通过订阅源阅读器（可嵌入在 Web 浏览器或作为单独应用程序运行）获取订阅源中的项。Web 页面也可以显示 Atom 订阅源中的项。Atom 订阅源可以用作 *mashup* 的一部分，mashup 是一种用于合并来自大量数据源的内容以便让用户以新的方式使用和理解数据的 Web 应用程序。在 mashup 中，Atom 订阅源中的数据可以由*窗口小部件*（是运行在 Web 页面中的脚本应用程序）来处理。

Atom 发布协议指定了一种方法，可供用户通过向存储条目的服务器发出 HTTP 请求来在 Atom 订阅源中添加、删除、编辑或查看各 Atom 条目。GET 请求用于检索要查看的条目，POST 请求用于添加一个完整的新条目，PUT 请求用于编辑现有的条目，而 DELETE 请求用于删除条目。服务器以合适的方式处理用户客户机请求的更改，并通过更改确认来响应用户客户机。

Atom 条目文档

Atom 条目文档是包含 Atom 订阅源的单个信息项（称为条目）的 XML 文档。

Atom 条目文档由包含大量子元素的 <atom:entry> 元素组成。这些子元素提供条目内容以及有关条目的元数据，例如，条目标题或第一次发布条目的时间。

Atom 条目的内容可以是纯文本、HTML、XHTML 或其他 IANA（因特网编号分配机构）介质类型。http://www.iana.org/assignments/media-types/ 中列出了 IANA 介质类型。Atom 条目还可将指向介质资源（例如，电影或图像）的链接作为其内容，在这种情况下它被称为介质链接条目。

Atom 条目文档的介质类型是 application/atom+xml。

Atom 条目文档的格式由位于下列位置的 *The Atom Syndication Format*（RFC 4287）定义: http://www.ietf.org/rfc/rfc4287.txt。

Atom 订阅源文档

Atom 订阅源文档是一个 XML 文档，用于提供有关 Atom 订阅源和一个或多个订阅源条目的元数据。当客户机请求订阅源中的信息时，服务器为满足该请求会生成包含适当数量的 Atom 条目的订阅源文档。

Atom 订阅源文档由包含大量子元素的 <atom:feed> 元素组成。<atom:entry> 元素是该元素最重要的子元素，但是通常订阅源的条目作为单独的 XML 文档存在，并且服务器在提供订阅源文档时就会添加这种子元素。Atom 订阅源文档在不包含任何 <atom:entry> 元素时，仍是一个可接受的文档。

其他子元素包含有关订阅源的元数据，例如，其标题和子标题，或主要作者。Atom 订阅源文档中的某些元数据项（例如，作者的详细信息和有关知识产权的信息）可应用于订阅源中的所有条目，除非条目含有其自己的元数据项。

Atom 订阅源文档的介质类型是 application/atom+xml。

Atom 订阅源文档的格式由位于下列位置的 *The Atom Syndication Format*（RFC 4287）定义：http://www.ietf.org/rfc/rfc4287.txt。

Atom 集合

Atom 集合是特定类型的 Atom 订阅源文档，它列出了可编辑的 Atom 条目的 URL。其格式与附带有某些专用元素的普通 Atom 订阅源文档的格式类似。它列示在 Atom 服务文档中，故称之为集合。

Atom 集合包含某些专用的 <atom:link> 元素。如果集合足够大，需要多个订阅源文档来返回所有条目，那么元素 <atom:link rel="first">、<atom:link rel="last">、<atom:link rel="next"> 和 <atom:link rel="previous"> 提供订阅源文档之间的导航。条目还有一个 <atom:link rel="edit"> 链接，编辑请求可被定向到该链接。<app:edited> 元素已添加至集合中的条目，以指定每个条目的上一次编辑时间。

当 Atom 条目可用作集合时，客户机可以编辑或删除现有条目，并为该集合创建新条目。客户机通过将 HTTP 请求发生至服务器来处理条目，如下所示:

GET 检索单个 Atom 条目或 Atom 条目列表。如 Atom 服务文档所述，将针对 Atom 条目列表的 GET 请求发送至集合的 URL。如条目的 <atom:link rel="edit"> 链接中所述，将针对单个 Atom 条目的 GET 请求发送至集合中单个 Atom 条目的 URL。

POST 创建新的 Atom 条目。POST 请求被发送至集合的 URL。

PUT 编辑客户机使用 GET 请求获取的现有 Atom 条目。PUT 请求被发送至集合中单个 Atom 条目的 URL。

DELETE
删除现有 Atom 条目。DELETE 请求被发送至集合中单个 Atom 条目的 URL。

在所有情况下，服务器都会将一个适当的 HTTP 响应发送至客户机。服务器可以更改客户机为条目提供的元数据，因此，当客户机成功发出 POST 或 PUT 请求时，服务器还会将新条目的副本作为响应主体返回。

除了包含标准 Atom 条目以外，集合还可包含介质资源，例如，电影或图像。如果服务器支持介质资源，那么它将在集合中创建被称为介质链接条目的专用 Atom 条目，以便提供至这些资源的链接。CICS 不提供介质资源支持。

Atom 集合的格式由位于下列位置的 *The Atom Publishing Protocol*（RFC 5023）定义: http://www.ietf.org/rfc/rfc5023.txt。

相关概念

第 217 页的『来自 CICS 的 Atom 订阅源的 URL』
订阅源或集合中的 Atom 订阅源文档、集合和 Atom 条目文档包含 Web 客户机可用于和文档进行交互的 URL (统一资源定位符)。每个 URL 都在 Atom 文档的 <atom:link> 元素中提供。Atom 文档可以有多个 <atom:link> 元素，该元素的 rel 属性（称为链接关系）指定不同 URL 的用途。

第 221 页的『国际资源标识（IRI）』
国际资源标识（IRI）是因特网的一个资源标识格式，它允许使用适用于本地语言（而不仅是英语）的字符和格式。IRI 可代替 URI 或 URL 用于受请求和响应支持的应用程序。

Atom 服务文档

Atom 服务文档是列举服务器中可用集合的 XML 文档。

Atom 服务文档具有根元素 <app:service>。(app: 前缀是 Atom 发布协议名称空间的前缀。) 其具有一个或多个 <app:workspace> 元素，这些元素用于定义包含大量 <app:collection> 元素的工作空间。工作空间仅用于对集合进行分组；您不能对工作空间执行任何操作。

<app:collection> 元素列出每个集合的 URL 和标题，还可以表示集合接受的输入类型以及可用于条目的类别。

Atom 服务文档的介质类型是 application/atomsvc+xml。

Atom 服务文档的格式由位于以下位置的 *The Atom Publishing Protocol*（RFC 5023）定义: http://www.ietf.org/rfc/rfc5023.txt。

Atom 类别文档

类别文档包含集合中条目的类别列表。同时还可以在服务文档中指定类别。如果要使用相同的类别来定义多个 Atom 订阅源，那么单独的类别文档将非常有用。

Atom 类别文档具有根元素 <app:categories>。(app: 前缀是 Atom 发布协议名称空间的前缀。) <app:categories> 元素包含集合中的条目所允许的 <atom:category> 元素列表。类别列表可以是固定的，在这种情况下，服务器可以拒绝具有其他类别的条目，也可以是开放式的，这样就可以使用其他类别。

如果在 Atom 服务文档中使用单独的 Atom 类别文档，而不是使用类别列表，那么会在服务文档中通过该类别文档的 URL 来进行引用。

Atom 集合的介质类型是 application/atomcat+xml。

RFC 5023（*The Atom Publishing Protocol*）中定义了 Atom 类别文档的格式，请参阅: http://www.ietf.org/rfc/rfc5023.txt。

第 19 章 CICS 如何支持 Atom 订阅源

CICS 使用 CICS Web Support 的 HTTP 服务器功能来支持 Atom 订阅源，并使用其他一些功能来执行支持 Atom 格式和协议的服务器所需的操作。您必须选择或设置为 Atom 订阅源提供数据的资源，并且定义针对 CICS 的订阅源。

通过 CICS 为 Atom 订阅源提供服务之前，您必须配置 CICS Web Support 的基本组件以将 CICS 设置为 HTTP 服务器。

您可以通过保存在现有资源中或由现有资源生成的数据创建 Atom 订阅源，这些资源包括：临时存储器队列、文件、数据库应用程序中的记录、Web service 或现有应用程序生成的输出。资源中的每条记录保存一个 Atom 条目的数据。或者，您可以设置新资源以包含 Atom 条目。

如果资源是为 CICS 定义的文件或临时存储器队列，并且描述资源中记录的语言结构采用 COBOL、C、C++ 或 PL/I 编写，那么 CICS 可以直接从资源抽取数据以生成 Atom 订阅源。您可以使用语言结构作为 CICS XML 助手的输入来生成用于定义资源结构的 XML 绑定，以使 CICS 可以将数据映射到 Atom 文档中的正确元素。

您还可以通过编写程序（即服务例程），从资源中的每条记录抽取数据来构成 Atom 条目，并通过一组容器向 CICS 提供数据，从而将任何资源作为 Atom 订阅源。如果能够针对您的资源生成 XML 绑定，那么服务例程可以使用 XML 绑定中的信息，但服务例程无需 XML 绑定。

标识或创建资源并生成 XML 绑定或编写服务例程之后，您可以通过创建以下项以在 CICS 中定义 Atom 订阅源：

- ATOMSERVICE 资源定义，指定 CICS 从何处获取数据来生成 Atom 文档以响应 Web 客户机请求。
- URIMAP 资源定义，指定 CICS 如何处理 Web 客户机要求获得 Atom 订阅源的 HTTP 请求。URIMAP 资源引用 ATOMSERVICE 资源定义。为支持您的 URIMAP 资源定义，必须具有一个 TCPIPSERVICE 定义，以用于为 CICS Web 支持定义入站端口，以使 CICS 可以通过该端口接收 HTTP 请求。
- Atom 配置文件，包含 Atom 订阅源文档的 XML 语法，以及某些特定于 CICS 的元素，例如，标识包含订阅源数据的资源的元素。CICS 使用 Atom 配置文件中的信息来构建 Atom 订阅源文档，该文档包含大量 Atom 条目，这些条目由 CICS 使用资源中的数据生成的。

如果希望启用 Web 客户机来管理和编辑订阅源中的 Atom 条目，那么可以执行进一步的步骤，将 Atom 订阅源设置为集合。要设置集合，应创建新的 URIMAP 定义，以将集合与订阅源分开提供。还可通过从相同数据为 Atom 订阅源复制等效文件，然后重新定义这些文件以声明它们用于集合并稍做更改，来创建新的 ATOMSERVICE 定义和 Atom 配置文件。然后，您可以创建 Atom 服务文档，并可以选择创建 Atom 类别文档以定义集合，并且通过 CICS 提供这些文档。如果正在使用服务例程，必须对其编码以便能处理 Web 客户机请求，从而添加、编辑和删除集合中的 Atom 条目。

与 Atom 订阅源交互

在设置 Atom 订阅源后，Web 客户机可以对其进行访问以获取 Atom 条目列表。CICS 以及使用的服务例程充当服务器，接收 Web 客户机的 HTTP 请求，并返回包含众多 Atom 条目的 Atom 订阅源文档。许多免费或商用的 Web 客户机应用程序都可以请求、接收和显示 Atom 订阅源，这些应用程序包括大多数现代的 Web 浏览器、专用的读者订阅源和提供更多功能的应用程序，例如，用于创建 mashup 的应用程序。检查该应用程序是否描述为支持 Atom 格式。您还可以编写自己的 Web 客户机应用程序来发出针对 Atom 订阅源数据的 GET 请求。

如果还将 Atom 订阅源设置为集合，那么您或其他用户可以依据“Atom 发布协议”中所述，通过支持针对 Atom 订阅源的 HTTP POST、PUT 和 DELETE 请求的 Web 客户机来管理和编辑订阅源中的条目。如果没有具备此功能的 Web 客户机，那么可以使用使您能够组成并发送自己的 HTTP 请求以及查看响应的 Web 客户机应用程序。您也可以编写自己的 Web 客户机应用程序，以向 Atom 集合发出 POST、PUT 和 DELETE 请求。如果 CICS 直接管理资源，那么它会对集合中已经可用的数据应用 Web 客户机的编辑请求，并返回适当的响应。如果您正在使用服务例程来提供数据，CICS 会使用容器接口将 Web 客户机的请求传递给该服务例程，您可以对服务例程进行编码，修改资源以响应这些请求。

要了解有关 CICS 如何支持 Atom 订阅源以及 Web 客户机如何与这些订阅源交互的更多信息，请按照 CICS TS for z/OS V4.1 信息中心（`https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp`）内的 Web 2.0 案例“创建 Atom 订阅源以使用员工信息”中的指示信息，设置并使用该样本 Atom 集合。

来自 CA8K SupportPac 的 Atom 订阅源

如果您在 CICS TS for z/OS V3.1 或 CICS TS for z/OS V3.2 中使用 CA8K SupportPac 来设置 Atom 订阅源，并且希望升级以在 CICS TS for z/OS V4.1 中使用针对 Atom 订阅源的支持，那么可以继续使用自己的服务例程。然而，您必须使用 ATOMSERVICE 资源定义、Atom 配置文件和 XML 绑定，而不是使用 PIPELINE 资源定义、管道配置文件和资源布局映射结构。您还必须更改服务例程代码，以重命名容器以及解释其中一个容器中的新参数，然后重新编译这些模块。

第 20 章 CICS 中 Atom 订阅源如何工作

为了提供 Atom 文档，CICS 必须包含适当的 URL，必须能标识并获取 Atom 条目的数据，并能确定 Atom 条目在 Atom 文档中的排列方式。在设置 Atom 订阅源或集合时，必须在 Atom 配置文件或服务例程中提供信息，以帮助 CICS 完成这些任务。

相关概念

第 209 页的第 18 章，『Atom 订阅源』
Web 订阅源（有时简称为“订阅源”）是内容提供方在因特网上发布的一系列相关项。Atom 订阅源是使用 Atom 联合格式和 Atom 发布协议的 Web 订阅源。

第 213 页的第 19 章，『CICS 如何支持 Atom 订阅源』
CICS 使用 CICS Web Support 的 HTTP 服务器功能来支持 Atom 订阅源，并使用其他一些功能来执行支持 Atom 格式和协议的服务器所需的操作。您必须选择或设置为 Atom 订阅源提供数据的资源，并且定义针对 CICS 的订阅源。

通过 CICS 为 Atom 订阅源执行数据处理

为了生成包含 Atom 条目的 Atom 订阅源或集合文档，CICS 直接从文件或临时存储器队列或者通过从其他资源抽取数据的服务例程获取 Atom 条目的数据。

图 15 显示了 CICS 如何使用 Atom 配置文件识别和抽取文件记录中的相关数据:

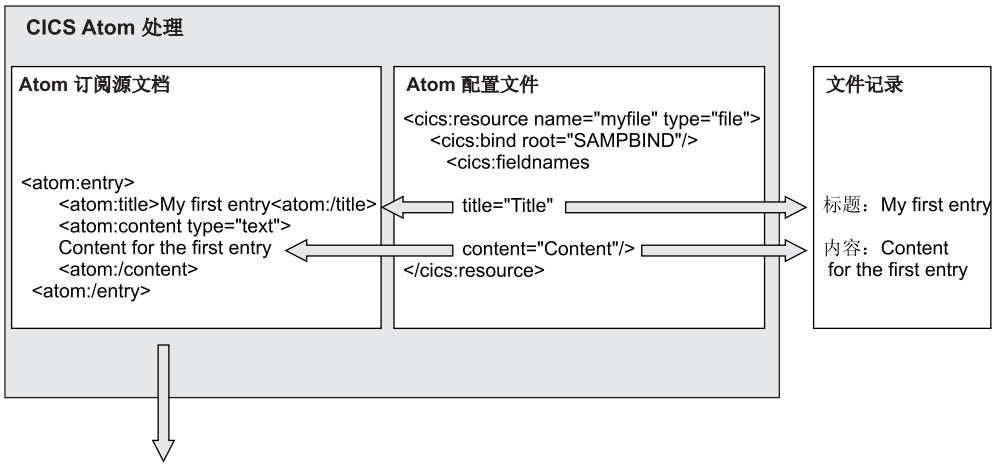


图 15. 直接从文件中抽取 Atom 条目数据

- 文件中的记录包含称为“标题”和“内容”的字段，这些字段保存了 Atom 条目的数据。
- Atom 配置文件包含一个用于识别该文件的 <cics:resource> 元素、一个引用该文件的 XML 绑定文件的 <cics:bind> 元素和一个 <cics:fieldnames> 元素。 <cics:fieldnames> 元素的标题属性用于识别文件记录中的“标题”字段，该字段保存了 Atom 条目的标题数据。内容属性用于识别文件记录中的“内容”字段，该字段保存了 Atom 条目的内容数据。

- CICS 使用 Atom 配置文件和 XML 绑定文件中的信息，在文件记录中的“标题”和“内容”字段查找和抽取数据，然后使用这些数据填充 Atom 订阅源文档中 Atom 条目的 <atom:title> 和 <atom:content> 元素。
- 当 CICS 执行完针对文件中一组记录的处理过程并生成所需的 Atom 条目数，会将包含 Atom 条目的 Atom 订阅源文档发送给 Web 客户机。

图 16 显示了 CICS 如何通过用户撰写的服务例程程序从数据库记录中获取数据:

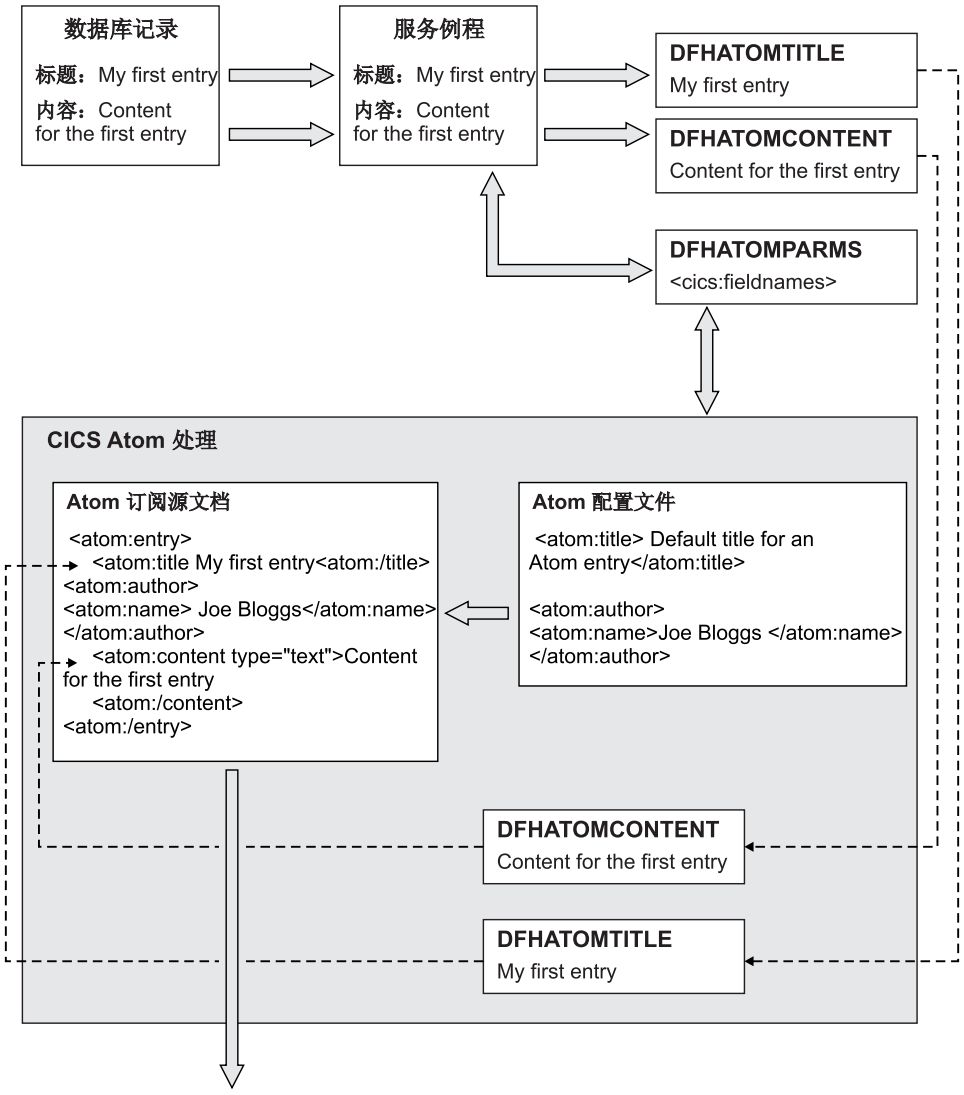


图 16. 使用服务例程提供 Atom 条目数据

- 该数据库中的记录包含称为“标题”和“内容”的字段，这些字段保存了 Atom 条目的数据。
- 服务例程从数据库中的“标题”和“内容”字段抽取数据。如果服务例程使用的资源具有 XML 绑定文件，那么它可以从 Atom 配置文件的 <cics:fieldnames> 元素的信息中获取相关字段的名称，在称为 DFHATOMPARMS 的容器中，这些名称将作为参数通过 CICS 传递至服务例程。DFHATOMPARMS 还包含有关 Web 客户机请求的其他信息。

- 服务例程创建称为 DFHATOMTITLE 和 DFHATOMCONTENT 的容器，并将“标题”和“内容”字段中的数据写入这两个容器中。然后，将这些容器返回至 CICS Atom 处理过程。
- Atom 配置文件包含为 Atom 条目提供缺省标题的 <atom:title> 元素，以及包含 <atom:name> 元素（提供作者姓名Joe Bloggs）的 <atom:author> 元素。
- CICS 通过使用服务例程在 DFHATOMTITLE 和 DFHATOMCONTENT 容器中提供的标题和内容组成 Atom 条目。由于该服务例程未提供作者姓名，因此 CICS 将使用 Atom 配置文件中的作者姓名。由于服务例程已提供标题数据，因此 CICS 不需要使用 Atom 配置文件中的缺省标题。
- 当 CICS 已多次调用服务例程以提供不同数据库记录中的数据来生成所需的 Atom 条目数时，会将包含 Atom 条目的 Atom 订阅源文档发送给 Web 客户机。

来自 CICS 的 Atom 订阅源的 URL

订阅源或集合中的 Atom 订阅源文档、集合和 Atom 条目文档包含 Web 客户机可用于和文档进行交互的 URL (统一资源定位符)。每个 URL 都在 Atom 文档的 <atom:link> 元素中提供。Atom 文档可以有多个 <atom:link> 元素，该元素的 rel 属性（称为链接关系）指定不同 URL 的用途。

由 CICS 提供的 Atom 订阅源文档或集合最多可包含以下四类 URL:

- 查找整个 Atom 订阅源或集合的 URL。该订阅源 URL 是在 <atom:link rel=″self″ > 元素中提供的，该元素是 <atom:feed> 元素的子元素。Web 客户机可以使用该 URL 从 Atom 订阅源或集合中获取包含多个条目的 Atom 订阅源文档。
- 在订阅源或集合中查找各个 Atom 条目的个别 URL。这些条目 URL 是在 <atom:link rel=″self″ > 元素中提供的，该元素是 <atom:entry> 元素的子元素。Web 客户机可以使用这些 URL 从订阅源或集合中检索单个的 Atom 条目。
- 编辑 URL，Web 客户机可使用这些 URL 来发出编辑集合的请求。这些 URL 是在 <atom:link rel=″edit″ > 元素中提供的。CICS 为整个集合提供一个编辑 URL，作为集合文档中 <atom:feed> 元素的子元素，为集合中每个 Atom 条目提供个别编辑 URL，作为 <atom:entry> 元素的子元素。CICS 还为集合和集合中的 Atom 条目提供 <atom:link rel=″self″ > URL。
- 导航 URL，Web 客户机可使用这些 URL 来检索 Atom 订阅源或集合中 Atom 条目的部分列表。这些 URL 是在 rel 属性为“first”、“previous”、“next”和“last”的 <atom:link> 元素中提供的。这些 URL 使 Web 客户机能够浏览整个 Atom 订阅源或集合，而无需一次检索所有 Atom 条目。CICS 为 Atom 订阅源文档中的 <atom:link rel=″next″> 元素提供 Web 客户机可用于从订阅源中检索 Atom 条目下一个窗口的 URL。在包含集合中部分条目列表的 Atom 文档中，CICS 添加 rel 属性为“first”、“previous”、“next”和“last”的 <atom:link> 元素，以提供到集合中其他部分 Atom 条目列表的导航。

对于 Atom 订阅源，针对整个订阅源的 URL 通常是在因特网或公司的内部网中公布的。当 Web 客户机通过使用订阅源 URL 获取 Atom 订阅源文档时，Atom 订阅源文档中的 Atom 条目将包含其自己的的个别 URL，并且 Web 客户机可以使用这些 URL 来检索单个的 Atom 条目。

对于包含可编辑 Atom 条目的集合，服务器可用的服务文档提供服务器中每个集合的编辑 URL。Web 客户机可以使用这些 URL 之一查看集合中的 Atom 条目，并可以发出向集合添加更多条目的请求。Web 客户机可针对单个 Atom 条目使用编辑 URL，以发出更新或删除该条目的请求。

Atom 联合格式允许使用国际资源标识（IRI），IRI 允许使用适用于英语以外的其他本地语言的 Unicode 字符和格式。您可以使用包含 Unicode 字符（作为 CICS 的 Atom 订阅源的资源定位符）的 IRI 来替换普通 URL。在 Atom 联合格式和 Atom 发布协议的 RFC 中，针对 Atom 订阅源和 Atom 条目的资源定位符被称为 IRI。第 221 页的『国际资源标识（IRI）』介绍了 IRI 以及如何对 Atom 订阅源使用它们。

如何指定和解析 Atom URL

在 CICS 中，使用 Atom 配置文件中 <atom:feed> 和 <atom:entry> 元素的子元素 <atom:link> 来指定针对整个 Atom 订阅源或集合的 URL 以及针对个别 Atom 条目的标准 URL。在 Atom 配置文件中，始终为这些子元素指定 <atom:link rel=″self″>，当 CICS 发出 Atom 文档时，CICS 会将 <atom:link rel=″edit″ > 元素中的同一链接添加到集合及集合中的 Atom 条目。您可以省略 Atom 配置文件中的 URL 的方案和主机部分，而仅指定路径部分。CICS 会在向客户机返回订阅源或条目文档时为 URL 添加方案和主机部分。

在 Atom 配置文件中，您不需要为导航 URL 指定 rel 属性为“first”、“previous”、“next”和“last”的任何 <atom:link> 元素。 CICS 会为您创建这些链接。

您为整个订阅源指定的 URL 和为个别条目指定的标准 URL 必须包含以相同方式开始的路径部分。您可以在 CICS 用于处理 Atom 订阅源的 Web 客户机请求的 URIMAP 资源定义中指定路径部分的公共部分，并使用星号表示要用于路径匹配的剩余路径。路径部分的公共部分是 CICS 用于标识 Atom 订阅源或集合的部分，因此该部分对于该 Atom 订阅源或集合（是通过指定主机名提供服务的所有 Atom 订阅源和集合的一部分）必须是唯一的。

当 Web 客户机使用包含路径部分的此公共部分的 URL 发出请求时，CICS 将查找匹配的 URIMAP 资源定义，并使用一些其他资源将该请求 URL 映射为 Atom 订阅源的数据。第 219 页的图 17 显示订阅源 URL 的此过程:

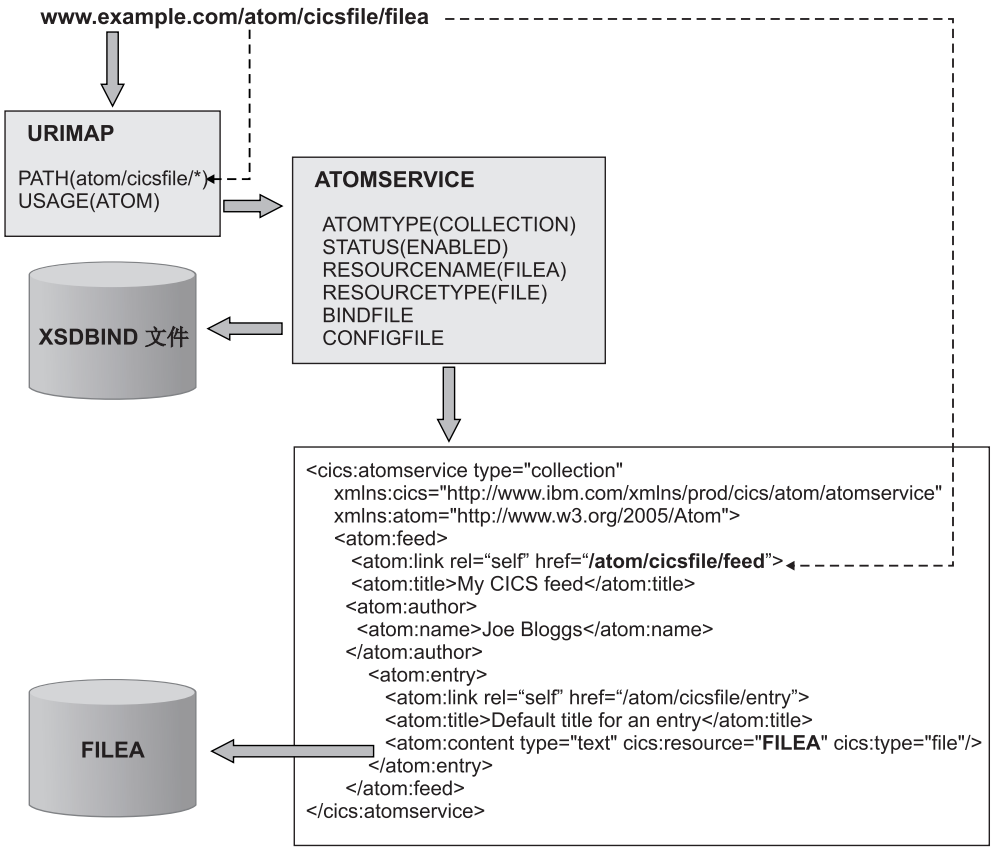


图 17. Atom 订阅源的请求 URL

- 要处理来自 Web 客户机的传入请求，您需要创建一个 URIMAP 资源定义，用于指定路径部分中对订阅源和条目 URL 为公共的部分。在本示例中，路径部分的公共部分是 `atom/cicsfile/`。当 Web 客户机使用为 Atom 订阅源或集合或 Atom 条目定义的 URL 发出请求时，CICS 将查找与路径部分的公共部分匹配的 URIMAP 资源。在本示例中，Web 客户机使用订阅源 URL `www.example.com/atom/cicsfile/filea` 请求 Atom 订阅源。
- URIMAP 资源指定一个 ATOMSERVICE 资源，该 ATOMSERVICE 资源指定提供 Atom 订阅源的 Atom 配置文件、XML 绑定（XSDBIND 文件）和 CICS 资源。例如，ATOMSERVICE 资源指定 FILEA 文件作为保存 Atom 条目数据的资源。
- CICS 使用该 ATOMSERVICE 资源查找 Atom 配置文件，并将由 Web 客户机使用的入站 URL 的路径部分与 Atom 配置文件的所有 <atom:link> 元素中指定的 URL 进行比较。当 CICS 在某个 <atom:link> 元素中找到一个包含匹配路径部分的 URL 时，它将采取适当的操作，返回 Atom 订阅源或条目文档，或实施编辑请求。在本示例中，由 Web 客户机使用的请求 URL 与为 Atom 配置文件的 Atom 订阅源指定的 URL 匹配，因此 CICS 必须返回一个 Atom 订阅源文档。
- Atom 配置文件（如 ATOMSERVICE 资源）将指定 FILEA 文件作为保存 Atom 条目数据的资源。正如第 215 页的『通过 CICS 为 Atom 订阅源执行数据处理』中的说明，CICS 可以直接从包含 Atom 条目数据的文件或临时存储器队列中抽取数据，或向服务例程传递相关请求。

在图 17 中，整个 Atom 订阅源的 URL 的路径为 `/atom/cicsfile/filea`，如在 Atom 配置文件的 <atom:feed> 元素的子元素 <atom:link> 中所指定。 Atom 配置文件的

<atom:entry> 元素也有一个 <atom:link> 子元素，该元素包含路径 /atom/cicsfile/entry。这是 Atom 条目的标准路径。针对 Atom 条目的此标准路径以路径部分的公共部分 atom/cicsfile/ 开始。Atom 条目的标准路径的其余部分不得与在 <atom:feed> 元素的子元素 <atom:link> 中指定的 Atom 订阅源的路径相同。 CICS 使用该部分路径在 Atom 配置文件中进行路径匹配，以确定是否需要 Atom 订阅源文档或 Atom 条目文档。

图 18 显示了 CICS 如何处理针对单个 Atom 条目的来自 Web 客户机的请求，以及如何识别正确的 Atom 条目:

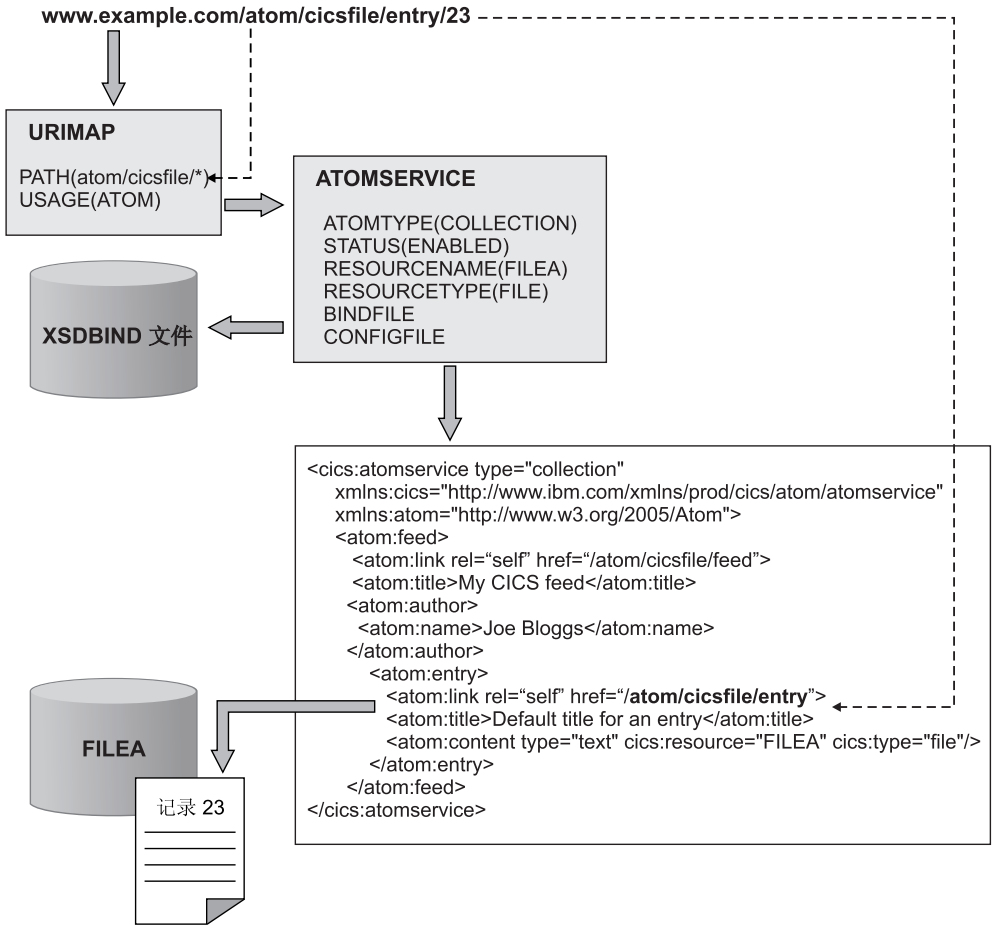


图 18. 针对 Atom 条目的请求 URL

- Web 客户机使用 URL www.example.com/atom/cicsfile/entry/23 请求单个 Atom 条目。 Web 客户机从 Atom 条目的 <atom:link> 子元素获取该 URL，而 Web 客户机最初收到该 URL 是作为 Atom 订阅源文档的一部分。
- Atom 条目 URL 包含 Atom 订阅源的路径部分的公共部分 atom/cicsfile/，因此它可以由与 订阅源 URL 相同的 URIMAP 和 ATOMSERVICE 资源进行处理。在本示例中，由 Web 客户机使用的请求 URL 与为 Atom 配置文件中的 Atom 条目指定的标准路径匹配，因此 CICS 必须返回一个 Atom 条目文档。
- CICS 通过检查跟在标准路径后的请求 URL 的其余部分，识别要返回给 Web 客户机的 Atom 条目。在本示例中，请求 URL 包含数字“23”。这是条目的选择器值。选择器值是一个标识，通常是一个数值，用于确定文件、临时存储器队列或其他包含

Atom 条目数据的资源中的记录。在本示例中，为 Atom 条目选择的选择器值是记录号。当选择了选择器值时，必须确保在附加了选择器值之后，针对整个 Atom 订阅源的 URL 和针对 Atom 条目的标准路径仍然不相同。第 223 页的『Atom 条目的选择器值』介绍了有关如何选择和使用选择器值的更多详细信息。

CICS 还会使用该选择器值，在 rel 属性为“first”、“previous”、“next”和“last”的 <atom:link> 元素中，构建到来自 Atom 订阅源或集合的条目部分列表的导航链接。CICS 通过获取为整个 Atom 订阅源或集合指定的路径，并附加 Atom 条目显示在部分列表开头的选择器值，来构建这些导航链接。CICS 将该信息与为 Atom 订阅源或集合指定的窗口设置一起使用，以向 Web 客户机返回部分列表。在此处所使用的 Atom 订阅源示例中，对于以选择器值为“9”的 Atom 条目开始的条目部分列表，CICS 将创建链接 `www.example.com/atom/cicsfile/filea/9`。

这些示例显示以缺省格式（称为“segment”格式）附加到 URL 的选择器值，在此格式中，选择器值将用作路径部分的最后一段。或者，您可以选择一种与使用 CA8K SupportPac 开发的应用程序兼容的 URL 格式（针对 Atom 条目的选择器值位于查询字符串中）。您可以使用 Atom 配置文件中的 <cics:selector> 元素指定该备用“query”格式。如果为此处使用的 Atom 订阅源示例指定 <cics:selector style=″query″/>，那么 CICS 将为个别 Atom 条目创建以下格式的链接：`www.example.com/atom/cicsfile/entry?s=23`。相同的格式也会用于导航链接。

国际资源标识（IRI）

国际资源标识（IRI）是因特网的一个资源标识格式，它允许使用适用于本地语言（而不仅是英语）的字符和格式。IRI 可代替 URI 或 URL 用于受请求和响应支持的应用程序。

国际资源标识（IRI）（RFC 3987）中描述了 IRI，可从 http://www.ietf.org/rfc/rfc3987.txt 中获取。 CICS 支持在 ATOM 订阅源文档以及针对作为 HTTP 服务器的 CICS 的入站 Web 客户机请求的 URIMAP 资源中使用 IRI。

主机名

为了满足域名服务器的需求，Web 客户机将 IRI 格式的主机名转换为称为 Punycode 格式的主机名。*Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)*（RFC 3492）中描述了 Punycode，可从 http://www.ietf.org/rfc/rfc3492.txt 中获取。该算法将主机名编码为仅由字母数字、连字符和句点组成的字符串。

如果您希望在 URIMAP 资源定义（定义 Web 客户机向 CICS 发出的请求）中，使用 IRI 作为 CICS 提供的 Web 资源或 Atom 订阅源的链接，那么必须指定 Punycode 格式的主机名。CICS 没有提供执行该转换的工具，但是在因特网上提供了免费应用程序可支持从 Unicode 到 Punycode 的转换。如果您使用单个星号代替主机名，以使 URIMAP 资源可以与任何主机名匹配，那么无需使用 Punycode。

路径部分

Web 客户机不会将 IRI 的路径部分转换为 Punycode，但它们会对路径中的 Unicode 字符执行转义或进行百分比编码。

如果您在 URIMAP 资源定义中，对 CICS 提供的 Web 资源使用 IRI，那么必须对指定路径中的所有 Unicode 字符进行百分比编码。如果您没有可将 Unicode 字符转换为百分比编码表示法的应用程序，那么可以从因特网上获取免费应用程序来执行此任务。注：第 32 页的『CICS Web Support 的 URL』中列出的 URL 长度限制同样适用于 Atom 订阅源的 URL，这表示在 URIMAP 资源定义中指定的 URL 的路径部分必须小于或等于 255 个字符。该上下文中的字符表示单个 ASCII 字符，而不是原始的 Unicode 字符。例如，根据该 255 个字符限制，百分比编码表示形式为 %D0%B4 的西里尔字母字符表示 6 个字符。

当 CICS 安装 URIMAP 资源定义时，CICS 以建议用于 URL 的规范格式存储路径，并取消转义某些字符，但是查看 URIMAP 资源时显示的路径与输入的路径保持一致。

当使用 IRI 作为 Atom 订阅源或条目文档的链接时，请在 Atom 配置文件和 URIMAP 资源定义中指定 IRI。您必须对 Atom 配置文件中使用 IRI 的任何 Unicode 字符进行百分比编码。

当 CICS 发送包含 IRI 的 Atom 文档时，会将百分比编码的字符转换为 XML 字符引用，以使 XML 有效。要使用 Web 客户机请求中的结果链接，您必须将 XML 字符引用转回百分比编码的字符。

该 URIMAP 资源示例包含一个路径，该路径使用 Unicode 字符来指定 Atom 订阅源的 IRI 开头部分，而其末尾带有一个星号表示路径匹配部分用于 IRI 的其余部分:

```
  Urimap         : ALEXANDR
  Group          : IRIMAPS
  DEScription    :
  STatus         : Enabled            Enabled | Disabled
  USAge          : Atom               Server | Client | Pipeline | Atom
 UNIVERSAL RESOURCE IDENTIFIER
  SCheme         : HTTP               HTTP | HTTPS
  POrt           : No                 No | 1-65535
  HOST           : *
  (Mixed Case)   :
  PAth           : %D0%90%D0%BB%D0%B5%D0%BA%D1%81%D0%B0%D0%BD%D0%B4%D1%80%D0%
  (Mixed Case)   : A1%D0%BE%D0%BB%D0%B6%D0%B5%D0%BD%D0%B8%D1%86%D1%8B%D0%BD*
```

该 Atom 条目示例包含了一个IRI，该 IRI 将同等 XML 字符引用用于 URIMAP 资源示例中表示的 Unicode 字符:

```
<entry>
<link rel="self" href="http://example.com:5050/&#x0410;&#x043B;&#x0435;
&#x043A;&#x0441;&#x0430;&#x043D;&#x0434;&#x0440;&#x0421;&#x043E;&#x043B;&#x0436;
&#x0435;&#x043D;&#x0438;&#x0446;&#x044B;&#x043D;/000100"/>
<id>tag:example.com,2009-02-13:file:FILEA:000100</id>
<title>FILEA item 000100</title>
<rights>Copyright (c) 2009, Joe Bloggs</rights>
<published>2008-11-06T12:35:00.000Z</published>
<author>
 <name>Joe Bloggs</name>
 <email>JBloggs@example.com</email>
</author>
<app:edited>2009-03-11T14:42:38+00:00</app:edited>
<updated>2009-03-11T14:42:38+00:00</updated>

<content type="text/xml">
  <DFH0CFIL xmlns="http://www.ibm.com/xmlns/prod/cics/atom/filea">
  <filerec>
  <numb>000100</numb><name>S. D. BORMAN</name><amount>$0100.11</amount>
  </filerec>
```

```
  </DFH0CFIL>
</content>

</entry>
```

Atom 条目的选择器值

Atom 条目的选择器值是 CICS 或服务例程可用于在包含 Atom 条目数据的文件、临时存储器队列或其他资源中查找记录的任何标识。合适的选择器值是始终适用于保存 Atom 条目数据的资源中的给定记录的任何唯一标识，例如，项编号或唯一键。

当 CICS 为响应 Web 客户机而发送 Atom 文档时，会使用个别 Atom 条目的选择器值来构造直接指向这些 Atom 条目的链接，并作为这些条目的已生成 Atom 标识的一部分。Web 客户机可使用这些链接来为单个 Atom 条目发出请求。每个链接的选择器值可在包含 Atom 条目数据的资源中识别出正确的记录。第 217 页的『来自 CICS 的 Atom 订阅源的 URL』 说明了如何向链接附加选择器值。

当 CICS 直接从文件或临时存储器队列交付 Atom 条目时，会根据资源类型来识别合适的选择器值。对于临时存储器队列，选择器值是用于识别临时存储器队列中记录的一个数字，CICS 假定该数字为十进制数。对于文件，选择器值是该文件的唯一键。CICS 如下假定每种文件类型的选择器值的格式:

- 对于 RRDS 和 VRRDS 文件，格式为十进制数。
- 对于 ESDS 和扩展的 ESDS 文件，格式为二进制数。
- 对于其他类型的 VSAM 文件，格式为字符串。

如果文件的键不是 CICS 假定的格式，那么您可以在 Atom 配置文件的 <cics:selector> 元素中指定正确的格式。

当服务例程为 Atom 条目返回数据时，您可以选择相应的选择器值。选择器值可以是供程序在资源中查找正确记录以便为 Atom 条目提供数据的任何值。例如，如果该资源是一个数据库，那么可以使用为这些记录提供键的唯一标识。如果该键不是一个字符串，那么您必须在 Atom 配置文件的 <cics:selector> 元素中指定您要使用的十六进制的选择器值。

当服务例程从订阅源或集合返回 Atom 条目时，您必须使用 DFHATOMPARMS 容器中的 **ATMP_NEXTSEL** 参数来为订阅源中可以使用的下一个 Atom 条目返回选择器值。如果 Web 客户机已请求大量条目，那么 CICS 会使用该选择器值再次链接到程序，这样，您的程序就可以识别并返回在资源中保存为记录的下一个 Atom 条目。该过程会一直持续，直到 CICS 具有足够的订阅源条目，或程序返回空值来表示资源中没有其他可用的 Atom 条目。

当服务例程从集合返回 Atom 条目时，您必须使用 DFHATOMPARMS 容器中的 **ATMP_PREVSEL**、**ATMP_FIRSTSEL** 和 **ATMP_LASTSEL** 参数来为集合中的上一个、第一个和最后一个 Atom 条目返回选择器值。CICS 使用这些值构造 <atom:link> 元素，该元素包含指向集合中其他部分的条目列表的链接。如果您认为这些值可供 Web 客户机从订阅源检索其他窗口的 Atom 条目，那么可以从订阅源返回 Atom 条目的这些值，但是对于订阅源，它们不是必需的。使用 **ATMP_PREVSEL** 生成链接的处理过程会增加响应时间，因此如果 Web 客户机已设置为使用这种形式的导航，那么仅为订阅源指定该值。

订阅源或集合中的第一个、上一个、下一个和最后一个 Atom 条目的身份取决于您选择返回 Atom 条目的顺序。『Atom 条目的顺序』说明了 CICS 如何确定返回 Atom 条目的顺序，并对服务例程返回 Atom 条目的顺序提出建议。

Atom 条目的顺序

CICS 或您的服务例程必须确定多个 Atom 条目在 Atom 订阅源文档中的排列顺序。

RFC 5023: *The Atom Publishing Protocol*（http://www.ietf.org/rfc/rfc5023.txt）规定了 Atom 集合中的条目应按照编辑顺序（由条目中的 <app:edited> 元素指定）返回至 Web 客户机。最近编辑的 Atom 条目应首先返回，因此它是出现在 Atom 订阅源文档中的第一个 Atom 条目。接下来应该返回排在最近编辑第二位的 Atom 条目，以此类推，最后返回最早编辑的条目。在 RFC 5023 中，该功能对于完整 Atom 条目列表是“应当执行”的需求，在此情况下，整个集合是在单个订阅源文档中返回的；但对于部分 Atom 条目列表，该功能是“必须执行”的需求。RFC 5023 和 RFC 4287 对未定义为集合的 Atom 订阅源中 Atom 条目的排序未作任何要求，因此对于 Atom 订阅源中的条目，服务器可以选择任何一致且合乎逻辑的顺序方法。

出于性能考虑，CICS 不会自动按最近编辑的顺序返回集合中的 Atom 条目。CICS 为了保持较短的响应时间而忽略了此“SHOULD”需求，但它仍然为部分列表提供了有用的功能。对于 Atom 订阅源和集合，当 CICS 直接从资源中抽取数据以生成 Atom 文档时，只要 CICS 可以确定，便会按照 Atom 条目作为记录写入资源中的时间顺序返回这些条目。最近写入的 Atom 条目首先返回，排在最近写入第二位的 Atom 条目第二个返回，以此类推。CICS 按照以下方式确定写入的顺序:

- 对于临时存储器队列，具有最高记录号的 Atom 条目首先返回。
- 对于 ESDS 和扩展的 ESDS 文件，具有最高 RBA（相对字节地址）或 XRBA（扩展的相对字节地址）的 Atom 条目首先返回。
- 对于 RRDS 和 VRRDS 文件，具有最高 RRN（相对记录号）的 Atom 条目首先返回。
- 对于 KSDS 和 AIX 文件（它们没有写入顺序的概念），Atom 条目按照记录键顺序返回，具有**最低**记录键的 Atom 条目首先返回。

如果您使用服务例程为 Atom 条目提供数据，那么可以选择返回 Atom 条目的顺序。如果您希望按照编辑时间的顺序返回集合中的 Atom 条目（遵循 RFC 5023），服务例程可以针对存储在文件中的 Atom 条目执行此操作。要按照编辑顺序返回 Atom 条目，请执行以下操作:

1. 在您的文件中，在包含 ABSTIME 格式的时间戳记的记录中添加一个字段，该时间戳记显示上次编辑该条目的时间。您可将 Atom 条目中的此信息作为 <app:edited> 元素输出。
2. 将包含该时间戳记的字段定义为该文件的备用索引。
3. 在您的服务例程中，使用备用索引来查找包含 Atom 条目数据的记录，然后按照最近编辑（由最近的时间戳记表示）的条目首先返回的顺序，返回这些条目。

如果您希望按照订阅源中 Atom 条目的更新时间（而不是条目首次编写的时间）返回这些条目，您也可以使用此方法。如果您无法在保存 Atom 条目数据的文件中存储适当的时间戳记，或者您发现使用该信息对条目排序时响应时间不可接受，请按照一致且符合逻辑的任何顺序（诸如 CICS 直接从资源中抽取数据时所使用的顺序）返回 Atom 条目。

服务例程为 DFHATOMPARMS 容器中的 **ATMP_PREVSEL**、**ATMP_NEXTSEL**、**ATMP_FIRSTSEL** 和 **ATMP_LASTSEL** 参数提供的值取决于您所选择的返回 Atom 条目的顺序。如果按照由时间戳记指明的编辑或更新时间返回 Atom 条目，那么这些参数的值如下所示:

- 上一个 Atom 条目是在编辑当前条目之后编辑的 Atom 条目。
- 下一个 Atom 条目是在编辑当前条目之前编辑的 Atom 条目。
- 第一个 Atom 条目是最近编辑的 Atom 条目。
- 最后一个 Atom 条目是最早编辑的 Atom 条目。

如果按照首次写操作的时间的顺序来返回 Atom 条目，那么这些参数的值如下所示:

- 上一个 Atom 条目是在编写当前条目之后编写的 Atom 条目。
- 下一个 Atom 条目是在编写当前条目之前编写的 Atom 条目。
- 第一个 Atom 条目是最近编写的 Atom 条目。
- 最后一个 Atom 条目是最早编写的 Atom 条目。

Atom 条目的日期和时间戳记

Atom 条目的元数据可以包含日期和时间戳记，以显示首次发布该 Atom 条目的时间以及上一次更新或上一次编辑该 Atom 条目的时间。

“Atom 联合格式”和“Atom 发布协议”定义如下日期和时间戳记:

<atom:published>
首次创建 Atom 条目或 Atom 条目首次可用的日期和时间。例如，如果您的 Atom 订阅源使用数据库中的记录为 Atom 条目提供数据，那么该日期和时间将是包含该数据的记录被添加到数据库的时间点。

<atom:updated>
上一次对 Atom 条目进行重大更改的日期和时间。例如，如果数据库记录中字段的值被更改，您可能会记录该日期和时间戳记。

<app:edited>
上一次编辑 Atom 条目的日期和时间。该日期和时间戳记仅适用于集合中包含的 Atom 条目，在这种情况下该日期和时间戳记是必需的（作为一个 SHOULD 需求）。

如果您正在设置用于包含 Atom 条目数据的新资源，那么您可以在资源的记录中包含用于保存日期和时间戳记的字段。服务例程可以通过覆盖 DFHATOMPARMS 容器中的 **ATMP_PUBLISHED**、**ATMP_UPDATED** 和 **ATMP_EDITED** 参数来返回该数据。如果您正在使用 DFHATOMPARMS 容器中的资源处理参数，那么 **ATMP_PUBLISHED_FLD**、**ATMP_UPDATED_FLD** 和 **ATMP_EDITED_FLD** 参数包含资源的记录中相关字段的名称和长度。

如果资源中的记录不包含任何保存元数据的字段，那么 CICS 将提供当前日期和时间作为所有元素的缺省时间戳记。在这种情况下，服务例程会对 DHFATOMPARMS 容器中的相关参数返回空格。对于 <atom:published> 元素，您可以在 Atom 配置文件的原型 Atom 条目中指定一个备用缺省时间戳记。您不能为 Atom 配置文件的原型 Atom 条目中的 <atom:updated> 和 <app:edited> 元素指定备用缺省值。

用于这些元素的日期和时间戳记必须采用 RFC 3339 格式（又称为 XML dateTime 数据类型）。*Date and Time on the Internet: Timestamps* （RFC 3339）是使用 UTC（全球标准时间）的日期和时间戳记的格式规范，摘自 ISO 8601 标准。可以从 http://www.ietf.org/rfc/rfc3339.txt 获取该规范。

使用 EXEC CICS ASKTIME 命令，后跟 EXEC CICS FORMATTIME 命令，来生成 RFC 3339 格式的日期和时间戳记。或者，如果您的服务例程可以使用 TRANSFORM DATATOXML 命令，那么您可以将保存在资源记录中的 CICS ABSTIME 值转换为该格式的日期和时间戳记。

如果您正在将 Web 客户机（POST 请求）提供的新 Atom 条目的数据填充到资源的记录中，或正在按照 Web 客户机请求（PUT 请求）编辑资源的记录中的字段，那么 Web 客户机可能会在 <atom:updated>、<atom:published> 或 <app:edited> 元素中提供日期和时间戳记。至于 <atom:updated> 和 <app:edited> 元素，建议您忽略这些，而是生成新的日期和时间戳记来确保精确性和有效性。尤其对于 PUT 请求，日期和时间戳记可能正是从资源中现有记录原值返回的日期和时间戳记。

相关参考

第 240 页的『DFHATOMPARMS 容器』
DFHATOMPARMS 是一个指定了 DATATYPE（CHAR）的容器，它包含 CICS 在与服务例程（为 Atom 订阅源提供数据）进行通信时所用到的参数。

Atom 条目的 Atom 标识

每个 Atom 条目都有唯一的 Atom 标识，在 Atom 条目的生存期内，其 Atom 标识必须保持不变。

Atom 条目的 Atom 标识在 <atom:id> 元素中指定。它必须采用有效的“国际化资源标识”（IRI）的形式，但不需要与实际的资源位置关联。

标记 URI

当 CICS 为 Atom 订阅源提供服务时，可以通过使用您在 Atom 配置文件的 <cics:authority> 元素中指定的信息，为每个 Atom 条目生成标记 URI 格式的唯一 Atom 标识。RFC 4151: *The 'tag' URI Scheme* 中描述了标记 URI 方案。

要为 Atom 条目的 Atom 标识生成标记 URI，CICS 必须按顺序使用以下项:

1. “标记”的方案
2. 您在 Atom 配置文件的 <cics:authority> 元素中指定的权限名称和日期
3. 您在 Atom 配置文件的 <cics:resource> 元素中指定的资源类型和资源名称组成的特定内容，以及个别 Atom 条目的选择器值

权限名称和日期以逗号分隔，而其他元素则以冒号分隔。CICS 生成的标记 URI 的示例如下所示:

```
tag:example.com,2009-01-08:tsqueue:WB20TSQ:23
```

标记 URI 中的权限名称是针对您或贵公司注册的域名或电子邮件地址，日期是您或贵公司拥有该权限名称的日期。第 270 页的『<cics:authority> 元素』提供了权限名称和日期的详细需求。

对于 CICS 直接从文件或临时存储器队列获取数据的 Atom 订阅源，资源类型和资源名称是文件或临时存储器队列的资源类型和资源名称。对于由用户编写的服务例程提供数据的 Atom 订阅源，资源类型和资源名称是该服务例程的资源类型和资源名称。

CICS 生成的作为 Atom 标识的标记 URI 具有以下特征:

- 只要您未更改文件、临时存储器队列或服务例程的名称，未更改 Atom 配置文件中的相关信息，也未将 Atom 条目移到其他资源中，那么在每个 Atom 条目的生存期内，Atom 标识都保持不变。
- 如果从不同的 CICS 区域以相同的方式提供相同的资源，那么 Atom 标识也保持不变。
- 如果对文件、临时存储器队列或服务例程重命名，Atom 标识会发生变化。为了与 Atom 格式保持一致，当您重命名某个资源后，就不能再将其作为具有相同 URL 的 Atom 订阅源，因为该资源的 Atom 标识已发生变化。
- Atom 标识在一个 CICS 区域中是唯一的，但并不保证它在多个不同的 CICS 区域中也是唯一的。如果您希望根据同名同类型但位于不同 CICS 区域中的资源设置 Atom 订阅源，那么可以在 Atom 配置文件的 <cics:authority> 元素中为每个订阅源指定不同的权限名称或不同的日期。即使所有其他信息都是相同的，具有不同日期的标记 URI 也互不相同。
- 如果用户编写的服务例程只处理一个 Atom 订阅源，那么 Atom 标识对于该例程提供的 Atom 条目是唯一的，但如果用户编写的服务例程提供多个订阅源，Atom 标识就不是唯一的。如果用户编写的服务例程提供了多个订阅源，那么可以为 Atom 标识选择备用格式，或在 Atom 配置文件的 <cics:authority> 元素中为每个订阅源使用一个不同的权限名称或日期。

Atom 标识的备用格式

如果不使用由 CICS 生成的标记 URI 格式，您可以使用 Atom 配置文件中针对原型 Atom 条目的 <atom:id> 元素，为 Atom 标识指定备用格式。CICS 会将选择器值附加到备用格式，从而为每个 Atom 条目生成唯一的 Atom 标识。

如果您使用备用 Atom 标识格式，请确保生成的 Atom 标识是唯一的，并且符合 RFC 4287 中有关 Atom 格式规范的要求。

为确保格式化正确无误，CICS 会忽略由 Web 客户机提供的任何 Atom 标识，而使用在 Atom 配置文件中为该订阅源指定的格式。

存储 Atom 标识

由于 CICS 每次为 Atom 条目提供服务时，都会为该 Atom 条目生成相同的 Atom 标识，因此不需要将 Atom 条目和 Atom 标识一起存储。该功能支持您从不包含用于存储元数据的字段的资源提供 Atom 条目数据，前提是使 Atom 标识保持不变，不更改配置文件中的 Atom 标识，不将 Atom 标识移到其他资源，（对标记 URI）也不更改资源或服务例程的名称。

但 RFC 4287 建议将 Atom 标识与 Atom 条目一起存储。如果您可以将 Atom 标识存储在保存了 Atom 条目数据的资源中，那么可以采用该建议。如果您将 Atom 条目存储在文件中，那么该字段可成为记录的唯一键。CICS 或服务例程将 Atom 条目的完整 Atom 标识存储在字段中，与 Atom 条目一起保存的 Atom 标识可以不同于 CICS 为该 Atom 条目生成的 Atom 标识，并可以覆盖后者。

对于服务例程，CICS 通过使用在 Atom 配置文件的 <cics:authority> 元素或 <atom:id> 元素中指定的信息，利用 **ATMP_ATOMID** 参数发送 Atom 条目的原型 Atom 标识。要生成完整的 Atom 标识，服务例程可以通过附加选择器值来完成原型 Atom 标识，也可忽略该标识并用自己的有效 Atom 标识替代。例如，您可以使用十六进制“通用唯一标识”（UUID），以 urn:uuid 方案生成 URI，UUID 在 RFC 4122: *A Universally Unique IDentifier (UUID) URN Namespace* 中进行了描述。服务例程可以使用在 **ATMP_ID_FLD** 参数中指定的字段，将 Atom 标识存储在资源记录中，然后使用 **ATMP_ATOMID** 参数返回该标识。

为确保准确性，CICS 忽略由 Web 客户机提供的 Atom 标识，并且不将它们存储在文件或临时存储器队列中的记录内，也不会将它们传递给服务例程。

RFC 4287 要求: 当 Atom 条目在其他位置复用或移到其他位置时，Atom 标识必须与 Atom 条目存储在一起。如果将 Atom 标识与 Atom 条目存储在一起，那么可以将 Atom 条目移到其他位置，并仍然符合该要求。如果未将 Atom 标识与 Atom 条目存储在一起，请勿将 Atom 条目移到其他位置。

Atom 订阅源的 Atom 标识

Atom 订阅源也具有唯一标识。如果使用 Atom 配置文件中的 <cics:authority> 元素使 CICS 生成标记 URI 以作为 Atom 标识，那么 CICS 将为 Atom 订阅源生成 Atom 标识（其格式与 Atom 条目的 Atom 标识的格式相同），但是该 Atom 标识不含为 Atom 条目附加的选择器值或唯一标识。例如:

```
tag:example.com,2009-01-08:tsqueue:WB20TSQ
```

如果您喜欢使用备用 Atom 标识格式，那么可以使用 Atom 订阅源的 <atom:id> 元素，以便为该订阅源指定完整的 Atom 标识。请确保该 Atom 标识是唯一的且符合 RFC 4287 中 Atom 格式规范的要求。

相关参考

第 270 页的『<cics:authority> 元素』
Atom 配置文件中的 <cics:authority> 元素提供了权限名称及相关日期，供 CICS 创建标记 URI 以用作个别 Atom 条目的 Atom 标识时使用。

第 21 章 Atom 订阅源的 CICS 样本

CICS 提供样本 URIMAP 和 ATOMSERVICE 资源定义、Atom 配置文件、XML 绑定和服务例程。

Atom 订阅源的样本位于以下两个位置:

- CICS 资源组 DFH$WEB2。
- z/OS UNIX 中 CICS 文件的根目录（由 CICS 系统初始化参数 USSHOME 指定）的 `/samples/web2.0/` 子目录。 USSHOME 的缺省值是 `/usr/lpp/cicsts/cicsts41`。

资源组 DFH$WEB2 中的 CICS 资源使用具有 USSHOME 的缺省值的路径 `/usr/lpp/cicsts/cicsts41/samples/web2.0/` 引用 `/samples/web2.0/` 子目录中的文件。如果您 CICS 区域的 USSHOME 系统初始化参数为 z/OS UNIX 上的 CICS 文件指定了非缺省值的根目录，那么要使用样本，您必须完成以下步骤:

1. 将资源组 DFH$WEB2 复制到新的资源组。
2. 在 DFH$WEB2 组的副本中修改 URIMAP 资源定义，将 HFSFILE 属性中出现的缺省目录 `/usr/lpp/cicsts/cicsts41` 更改为您的 CICS 区域用于 z/OS UNIX 中 CICS 文件的根目录名称。
3. 在 DFH$WEB2 组的副本中修改 ATOMSERVICE 资源定义，将 CONFIGFILE 和 BINDFILE 属性中出现的缺省目录 `/usr/lpp/cicsts/cicsts41` 更改为您的 CICS 区域用于 z/OS UNIX 中的 CICS 文件的根目录名称。

FILEA 中的样本 Atom 订阅源

CICS 提供了一组资源，以将 CICS 样本文件 FILEA 作为直接来自 CICS 的 Atom 订阅源。

下表中显示的组件用于为该 Atom 订阅源提供服务。

表 8. 用于运行 DFH0W2F1 的组件

组件	用途	位置
URIMAP 资源 DFH$W2F1	处理针对使用 FILEA 的 Atom 订阅源的 HTTP 请求	CICS 组 DFH$WEB2 或您的副本
ATOMSERVICE 资源 DFH$W2F1	用于生成 Atom 订阅源的名称资源	CICS 组 DFH$WEB2 或您的副本
Atom 配置文件 `filea.xml`	为 Atom 订阅源文档和 Atom 条目提供元数据和结构	*USSHOME*`/samples/web2.0/atom`
XML 绑定 `filea.xsdbind` 和 XML 模式 `filea.xsd`	告知 CICS 有关 FILEA 中记录的结构	*USSHOME*`/samples/web2.0/atom`

DFH0W2F1 COBOL 样本服务例程

样本服务例程 DFH0W2F1 是一个 COBOL 程序，用于演示用户编写的服务例程如何处理针对 Atom 条目的 GET、POST、PUT 和 DELETE 请求，这些条目使用来自 CICS 样本文件 FILEA 的数据。要了解有关该服务例程的信息，请参阅 第 313 页的『Atom 订阅源的 DFH0W2F1 COBOL 样本服务例程』。

下表中显示的组件用于运行 DFH0W2F1。

表 9. 用于运行 DFH0W2F1 的组件

组件	用途	位置
URIMAP 资源 DFH$W2P1	处理针对使用 FILEA 的 Atom 订阅源的 HTTP 请求，该资源将 DFH0W2F1 作为服务例程。	CICS 组 DFH$WEB2 或您的副本
ATOMSERVICE 资源 DFH$W2P1	用于生成 Atom 订阅源的名称资源	CICS 组 DFH$WEB2 或您的副本
Atom 配置文件 dfh0w2f1.xml	为 Atom 订阅源文档和 Atom 条目提供元数据和结构	*USSHOME*/samples/web2.0/atom

ATOMSERVICE 资源 DFH$W2P1 不为 FILEA 文件指定 XML 绑定，由于 FILEA 不包含可作为 Atom 条目元数据的任何字段，因此 DFH0W2F1 不会使用 Atom 配置文件中的 <cics:fieldnames> 元素。

样本 Atom 集合

样本 Atom 集合演示如何创建 Atom 集合以包含有关员工的数据，包括他们所处的地理位置。有关设置和使用样本 Atom 集合的指示信息位于 CICS TS for z/OS V4.1 信息中心（https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp）内的 Web 2.0 案例“创建 Atom 订阅源以处理员工信息”。

下表中显示的组件用于生成样本 Atom 集合。

表 10. 样本 Atom 集合的组件

组件	用途	位置
URIMAP 资源 DFH$W2Q1	处理针对样本 Atom 集合以及个别 Atom 条目的 HTTP 请求	CICS 组 DFH$WEB2 或您的副本
ATOMSERVICE 资源 DFH$W2Q1	用于生成 Atom 集合的名称资源	CICS 组 DFH$WEB2 或您的副本
Atom 配置文件 dfh0w2q1.xml	为 Atom 订阅源文档和 Atom 条目提供元数据和结构	*USSHOME*/samples/web2.0/atom
XML 绑定 dfh0w2q1.xsdbind 和 XML 模式 dfh0w2q1.xsd	告知 CICS 有关临时存储器队列中记录的结构	*USSHOME*/samples/web2.0/atom
URIMAP 资源 DFH$W2AC	处理要求在 Web 浏览器中查看 Atom 配置文件或 XML 绑定的 HTTP 请求	CICS 组 DFH$WEB2 或您的副本
COBOL 语言结构（副本）DFH0W2Q1	描述临时存储器队列中记录的结构；以前用于生成 XML 绑定	SDFHSAMP 样本库

表 10. 样本 Atom 集合的组件 (续)

组件	用途	位置
CICS 图标 cics-icon.gif 和 CICS 徽标 cics-logo.gif	样本 Atom 集合的图标和徽标图像	*USSHOME*/samples/web2.0/image
URIMAP 资源 DFH$W2GI	处理针对图标和徽标图像的 HTTP 请求	CICS 组 DFH$WEB2 或您的副本

下表中显示的组件用于生成支持您与样本 Atom 集合交互的内容聚合 Web 页面。

表 11. 内容聚合 Web 页面的组件

组件	用途	位置
内容聚合 Web 页面 dfh$w2q1.html	允许您与样本 Atom 集合交互的内容聚合	*USSHOME*/samples/web2.0/html
URIMAP 资源 DFH$W2HT	处理针对内容聚合 Web 页面的 HTTP 请求	CICS 组 DFH$WEB2 或您的副本
样式表 dfh$w2ss.css	控制内容聚合 Web 页面的外观	*USSHOME*/samples/web2.0/style
URIMAP 资源 DFH$W2SS	处理针对样式表的 HTTP 请求	CICS 组 DFH$WEB2 或您的副本
脚本 dfh$w2w2.js	一个用在内容聚合 Web 页面中的窗口小部件，用于显示和接收来自 Atom 条目的数据	*USSHOME*/samples/web2.0/script
URIMAP 资源 DFH$W2JS	处理针对脚本 dfh$w2w2.js 的 HTTP 请求	CICS 组 DFH$WEB2 或您的副本

第 22 章 设置资源以提供 Atom 条目数据

Atom 订阅源或集合由一组 Atom 条目组成，这些条目包括数据项和合适的元数据项。对于 CICS 提供的 Atom 订阅源，ATOM 条目数据来自资源中的记录，该资源可能是一个文件、一个临时存储器队列或其他资源（例如，数据库表）。单个记录提供单个 Atom 条目。

关于此任务

资源中的记录可能保存了 Atom 条目的元数据项以及 Atom 条目的内容，或者可能仅保存了 Atom 条目的内容。当设置 Atom 订阅源时，您可以让 CICS 提供资源记录中没有的任何所需的元数据项。

您可以使用以下某个资源来为 Atom 订阅源中的 Atom 条目提供数据:

- 您创建的用于包含 Atom 条目的新 VSAM 文件或临时存储器队列。
- 在 CICS 中定义的现有 VSAM 文件或临时存储器队列，CICS 可以直接从中抽取数据来生成 Atom 订阅源。除了已使用 NONUNIQUEKEY 属性定义的备用索引文件外，CICS 可以从任何类型的 VSAM 文件为 Atom 订阅源抽取数据。该文件必须具有文件记录的唯一键。CICS 无法直接从 BDAM 文件为 Atom 订阅源抽取数据。
- 您可以从 CICS 应用程序访问的任何其他资源。您可以使用 CICS 应用程序（称为服务例程）来交付 CICS 或非 CICS 资源，该程序会从资源中抽取 Atom 条目的数据并将其提供给容器中的 CICS。

对于 ESDS 文件，

- 要创建新的 VSAM 文件或临时存储器队列以包含 Atom 条目，请遵循第 234 页的『创建可存储 Atom 条目的 CICS 资源』中的指示信息。
- 要公开现有 VSAM 文件或临时存储器队列作为 Atom 订阅源:
  1. 查找或撰写描述资源中记录结构的语言结构。
     - 您可以使用 COBOL、C、C++ 或 PL/I 等高级语言结构或副本。高级语言结构必须位于分区数据集中。对于 CICS 应用程序所使用的文件或临时存储器队列，必须已经存在语言结构。如果不存在，那么您可以针对这些记录撰写一个语言结构。
     - 或者，您可以使用描述资源中记录结构的 XML 模式或 WSDL 文档。
  2. 按照 *CICS Application Programming Guide* 中的“通过语言结构生成映射”所描述的信息，使用语言结构生成资源的 XML 绑定文件。 如果您选择使用 XML 模式或 WSDL 文档，那么请遵循 *CICS Application Programming Guide* 的“通过 XML 模式生成映射”中的备用指示信息。
- 要交付任何其他 CICS 或非 CICS 资源，请撰写服务例程，以便从资源记录中为每个 Atom 条目抽取数据，并通过一组容器向 CICS提供这些数据。 有关撰写服务例程的指示信息，请参阅第 237 页的『编写提供 Atom 条目数据的程序』。

下一步做什么

在您选择保存 Atom 条目数据的资源并创建 XML 绑定文件或服务例程以支持数据交付之后，请遵循第 261 页的第 23 章，『为 Atom 订阅源设置 CICS 定义』中的指示信息来设置 Atom 订阅源。

具有 Atom 订阅源的 ESDS 文件

您可以使用 ESDS（输入顺序数据集）文件来保存 Atom 订阅源的 Atom 条目数据，但对于删除 Atom 条目有一定的限制，如果您将订阅源设置为可编辑的集合，这些限制就适用。

Web 客户机可以通过发出具有 DELETE 方法的 HTTP 请求来删除集合中的 Atom 条目。对于 ESDS 文件，仅当 ESDS 未定义备用索引时，才支持具有 DELETE 方法的 HTTP 请求。

作为对 DELETE 请求的响应，CICS 通过改写相关记录，将 'FF'x 作为第一字节以表示逻辑删除，从 ESDS 中删除该记录。如果 Web 客户机发出具有 GET 方法的后续 HTTP 请求，以检索位于已删除记录中的 Atom 条目，那么 CICS 将针对这些 GET 请求返回"未找到"响应。

如果将 ESDS 文件定义为 Atom 集合，您必须使用以下方法之一来确保使用该 ESDS 文件的其他应用程序可以正确地处理已删除的记录:

- 在 ESDS 的 FILE 资源定义中，将 DELETE 设置为 NO。
- 或者，对应用程序进行编码，将以 'FF'x 开头的记录作为逻辑删除处理。

如果您正在设置新的资源以存储集合的 Atom 条目数据，为避免这些限制，请选择 ESDS 以外的其他 VSAM 文件类型。

如果 ESDS 文件仅用于未定义为集合的 Atom 订阅源，那么 Web 客户机将无法发出具有 DELETE 方法的请求，这些限制就不适用。但是，如果您正在设置新的资源以存储 Atom 订阅源的 Atom 条目数据，请避免使用 ESDS 文件，以防将来决定将 Atom 订阅源设置为集合。

创建可存储 Atom 条目的 CICS 资源

要将数据存储为 Atom 条目，请在 CICS 中创建一个文件或临时存储器队列，并用 COBOL、C、C++ 或 PL/I 编写一个语言结构来解释其结构。

关于此任务

在新文件或临时存储器队列中，每个记录都代表一个单独的 Atom 条目。记录中的每个字段都包含 Atom 条目中一个元素的数据，此数据可以是元素的内容或诸如标题等元数据项。设置 Atom 订阅源时，将使用 Atom 配置文件中的 <cics:fieldnames> 元素向 CICS 指定这些字段的名称，CICS 将从每个记录中抽取数据来组成 Atom 条目。

Atom 条目中可能出现的元素的完整列表和描述可以在位于以下位置的 *The Atom Syndication Format*（RFC 4287）中找到: http://www.ietf.org/rfc/rfc4287.txt。CICS 不支持所有这些元素，并且 CICS 支持的元素中有一部分在您的文件或临时存储器队列中不受支持，仅可以在 Atom 配置文件中指定。要获取在文件和临时存储器队列中 CICS 支持

的元素的列表和描述，请参阅 第 272 页的『<cics:fieldnames> 元素』。要获取在 Atom 订阅源及 Atom 条目中 CICS 支持和不支持的元素的完整列表，请参阅 第 281 页的『CICS 的 Atom 元素引用』。

1. 决定使用临时存储器队列还是文件作为存储 Atom 条目数据的资源。
   - 如果您正在 CICS 中试用 Atom 订阅源，那么临时存储器队列是合适的选择，这是因为在开始使用临时存储器队列之前无需向 CICS 进行定义，不过在您想要应用安全措施时仍然需要设置 CICS 资源定义。它还适用于不会长期使用 Atom 条目的 Atom 订阅源；例如，为应用程序中的事件发出警报的情况。
   - 设置文件的时间要比设置临时存储器队列的时间长，这是因为在使用文件之前必须向 CICS 定义该文件，并且文件通常需要一个物理数据集。但是，文件为所有 Atom 订阅源（包括您可能想要设置为可编辑集合的订阅源）提供合适的长期存储。保存 Atom 条目的文件必须具有记录的唯一键，并且您不能使用通过 NONUNIQUEKEY 属性定义的备用索引文件。您可以使用任何类型的 VSAM 文件来保存 Atom 条目，但请注意，由于第 234 页的『具有 Atom 订阅源的 ESDS 文件』中提及的原因，ESDS（输入顺序数据集）文件对于可能希望设置为可编辑集合的订阅源来说并不是好的选择。您不能使用 BDAM 文件。
2. 规划文件或临时存储器队列中记录的内容。 由于可以在 Atom 配置文件中指定所有的元数据，所以 CICS 只需在记录中指定 Atom 条目的内容。但是，当您设置包含 Atom 条目的专用文件或临时存储器队列时，可以添加包含记录中元数据的字段，这样您可以用这些字段提供特定于每个 Atom 条目的元数据。 以下列表概括了您可以将其作为记录中的字段进行包含的数据项，并指出了它们是可选还是必填:

   Atom 标识

   Atom 条目的唯一标识。要了解有关 Atom 标识格式的更多信息，请参阅 第 226 页的『Atom 条目的 Atom 标识』。

   Atom 条目必须具有唯一的 Atom 标识。当 CICS 为 Atom 订阅源提供服务时，它可以使用在 Atom 配置文件中指定的信息为每个条目生成一个唯一 Atom 标识。只要您未更改文件或临时存储器队列的名称，未更改 Atom 配置文件中的相关信息，也未将 Atom 条目移到其他资源中，那么在 Atom 条目的生存期内，由 CICS 创建的 Atom 标识将保持不变。因此，记录中不必包含用于存储 Atom 标识的字段。

   但是，为了完全符合 Atom 格式的要求，在条目被复用或移到其他位置时 Atom 标识必须与条目存储在一起。如果您认为在该文件或临时存储器队列以外的其他地方您可能会用到 Atom 条目，或者只是希望遵循 RFC 4287 中 Atom 标识应与条目存储在一起的建议，请在记录中包含用于保存 Atom 标识的字段。如果您将 Atom 条目存储在文件中，那么该字段可成为记录的唯一键。

   作者的详细信息

   Atom 条目主要作者的个人姓名、电子邮件地址和 Web 站点，位于三个独立的字段。您必须为 Atom 订阅源或所有 Atom 条目提供作者姓名，但是其他字段是可选的。如果您的 Atom 条目具有多个作者，请在记录中包含一个姓名字段和其他详细信息字段（如果需要）。如果作者的姓名和其他详细信息对于所有 Atom 条目都相同，请改在 Atom 配置文件中指定该姓名和详细信息。

   类别 用于对条目进行分类的类别术语。该字段是可选的。如果您计划将该 Atom

订阅源设置为可编辑集合，并且使用类别来描述集合，请包括该字段。如果您不打算将该订阅源设置为集合，但如果该字段可帮助使用者使用该订阅源，您仍然可以包括该字段。

内容 Atom 条目中要发布的全部内容。CICS 需要每个 Atom 条目的内容。您的内容可以是纯文本、HTML、XHTML、XML 或其他文本介质类型。 CICS 不支持非文本的内容，也不支持没有内容的 Atom 条目。如果您包含了用于元数据的任何字段，那么必须在记录中添加一个字段或嵌套字段的子结构来保存内容。如果您未添加包括元数据的任何字段，那么 CICS 会将来自文件或临时存储器队列的记录整体作为条目的内容进行发布。

内容类型

Atom 条目内容的介质类型，如文本或 XML。该字段是可选的。如果所有的 Atom 条目都具有相同类型的内容，那么可改在 Atom 配置文件中指定介质类型。

上一次编辑的日期

指示上一次编辑记录的时间戳记。您可以使用 RFC 3339 中描述的 XML dateTime 格式的时间戳记，或 CICS ABSTIME 值。要了解有关日期和时间戳记的更多信息，请参阅 第 225 页的『Atom 条目的日期和时间戳记』。如果您计划将该 Atom 订阅源设置为可编辑的集合，那么包含该字段使您能够按照上次编辑 Atom 条目的时间顺序返回这些条目，对于集合，这是“Atom 发布协议”推荐的返回顺序。如果您不打算将该订阅源设置为集合，请不要包括该字段。

首次发布的日期

指示首次将记录创建或发布为 Atom 条目的时间戳记或 ABSTIME 值。该字段是可选的。如果您认为该字段有助于使用者使用您的订阅源，请包括该字段。

标题 Atom 条目的标题。CICS 只支持纯文本的标题。每个 Atom 条目都需要一个标题，所以您通常需要包括该字段。如果所有的 Atom 条目都具有相同的标题，那么可改在 Atom 配置文件中指定该标题。

摘要 Atom 条目的摘要。CICS 只支持纯文本的摘要。该字段是可选的。只有当条目的内容是非文本介质类型时才需要摘要。CICS 不支持 Atom 条目中的非文本内容。

上一次更新的日期

指示上一次更新记录的时间戳记或 ABSTIME 值。每个 Atom 条目都需要更新时间戳记，所以您通常需要包括该字段。如果您无法在文件或临时存储器队列中包括该时间戳记，那么您可以省略该字段，并且 CICS 可以提供在 Atom 订阅源文档中发布条目的日期和时间或您在 Atom 配置文件中指定的合适备选缺省值。

3. 使用 COBOL、C、C++ 或 PL/I 为您的文件或临时存储器队列编写语言结构或副本。语言结构描述文件或临时存储器队列的记录中的字段，按照每个字段的出现顺序描述其名称、内容类型和长度。 如果打算通过应用程序使用 COBOL、C、C++ 或 PL/I 在文件或临时存储器队列中创建记录，那么该应用程序将使用该语言结构对文件或临时存储器队列执行写操作。 如果也需要该语言结构来生成文件或临时存储器队列的 XML 绑定，所以即使是在使用其他语言编写该应用程序或您目前没有使用该应

用程序时，该副本也是必需的。例如，当您正在 CICS 中试用 Atom 订阅源并且您的记录结构十分简单以致于您可以使用 CECI 事务对文件或临时存储器队列执行写操作时。

注：确保用于提供 Atom 条目元数据的字段未嵌套到您的语言结构中。必须使用同一级别的语言结构列出记录中的所有元数据字段。您可以在提供 Atom 条目内容的字段中使用嵌套字段结构。

将语言结构存储在固定记录长度为 80 个字节的分区数据集。 本示例 COBOL 语言结构声明了若干包含每个元素数据的适当长度的字母数字字段:

```
******************************************************************
* Name: SAMPBIND.cob                                             *
*                                                                *
*                                                                *
* This is a COBOL copy book to describe the data record.         *
* You can generate a binding file from this.                     *
*                                                                *
******************************************************************

      03 TITLE-FIELD                      PIC X(50).
      03 SUMMARY-FIELD                    PIC X(500).
      03 ATOMID-FIELD                     PIC X(20).
      03 CONTENT-FIELD                    PIC X(500).
      03 AUTHOR-NAME-FIELD                PIC X(30).
      03 AUTHOR-EMAIL-FIELD               PIC X(256).
      03 AUTHOR-URI-FIELD                 PIC X(256).
      03 EDITED-FIELD                     PIC X(25).
      03 UPDATED-FIELD                    PIC X(25).
      03 PUBLISHED-FIELD                  PIC X(25).
      03 CATEGORY-FIELD                   PIC X(20).
```

4. 按照 *CICS Application Programming Guide* 中的步骤，将您的语言结构用作 CICS XML 助手的输入来创建 XML 绑定。
5. 如果您已决定使用文件来存储 Atom 条目:
   a. 按照 *CICS System Definition Guide* 中的过程，设置合适的 VSAM 数据集。
   b. 使用 *CICS Resource Definition Guide* 中的信息，通过创建和安装 FILE 资源定义来向 CICS 定义文件。
6. 如果您已经决定使用临时存储器队列来存储您的 Atom 条目，并且希望为临时存储器队列指定安全性和恢复设置，请使用 *CICS Resource Definition Guide* 中的信息定义临时存储器模型（TSMODEL）。

下一步做什么

如果您已拥有可以处理文件或临时存储器队列中记录的应用程序，请使用您的应用程序或其他合适的方法测试设置，通过 WRITEQ TS 命令（针对临时存储器队列）或 WRITE 命令（针对文件）将至少一条记录写入临时存储器队列或文件。

编写提供 Atom 条目数据的程序

您可以编写一个服务例程，以根据可由 CICS 程序访问的任何数据（例如，DB2 数据库中的记录、文件中的记录或 COMMAREA）提供 Atom 订阅源。下列指示信息将介绍如何编写用于响应针对 Atom 订阅源的 HTTP GET 请求的程序。

关于此任务

Web 客户机可以请求订阅源中的大量 Atom 条目，也可以请求某个特定的条目。CICS 接收来自 Web 客户机的请求，并根据每个客户机请求的相关信息链接到该程序。对于客户机请求的每个 Atom 条目，CICS 都会链接到该程序一次，并且该程序每次都会返回一个条目。

该程序使用从资源（如数据库或文件）的记录中抽取的数据提供 Atom 条目，这些资源会保存针对该订阅源 Atom 条目的数据。有关该过程的概述，请参阅第 215 页的『通过 CICS 为 Atom 订阅源执行数据处理』。

CICS 使用容器接口与服务例程进行通信。使用 EXEC CICS GET CONTAINER 和 EXEC CICS PUT CONTAINER 命令与容器进行交互。样本服务例程 DFH$W2S1 显示如何使用容器。

由于 Web 客户机请求是 HTTP 请求，因此您还可以使用 CICS Web API 命令（例如 WEB READ HTTPHEADER 和 WEB READ QUERYPARM 命令）与其进行交互。如果您知道如何使用这些命令，那么可以在服务例程中使用它们，以从 Web 客户机请求中直接获取信息，包括 CICS 在 DFHATOMPARMS 容器中未提供的任何信息。

如果您可以为包含 Atom 条目数据的资源创建一个 XML 绑定，那么就可以将信息（有关包含 Atom 条目数据的资源记录中的字段名称和长度）从 CICS 传递到 DFHATOMPARMS 容器中的程序。您的程序可以使用该信息查找资源记录中的元数据字段。通过使用这些资源处理参数，您可以创建可处理多个资源的类属服务例程。但是，您不一定得使用资源处理参数；如果您希望使用它们，那么可以直接将有关资源结构的信息以代码形式编写到程序中。

要响应 GET 请求，您的服务例程必须执行下列这些任务:

1. 使用 EXEC CICS GET CONTAINER 命令检索 DFHATOMPARMS 容器中的数据。CICS 使用该容器向服务例程提供有关请求的信息。 样本服务例程 DFH$W2S1 显示如何读取 DFHATOMPARMS 中的参数。 第 240 页的『DFHATOMPARMS 容器』完整记录了 CICS 在该容器中传递的参数。
2. 检查 **ATMP_HTTPMETH** 参数的值，以验证请求方法是否为 GET。 CICS 返回错误或对除 GET、POST、PUT 和 DELETE 以外的方法做出适当的响应。此处未提及有关用于响应 Atom 集合的 HTTP PUT、POST 或 DELETE 请求的指示信息，可参阅第 307 页的『如何在服务例程中处理 Atom 集合编辑请求』，获取这些指示信息
3. 使用 DFHATOMPARMS 容器中的 **ATMP_ATOMTYPE** 和 **ATMP_SELECTOR** 参数的值来确定资源中包含程序必须返回给 CICS 的 Atom 条目数据的记录。**ATMP_SELECTOR** 参数可能包含标识特殊 Atom 条目的选择器值。第 223 页的『Atom 条目的选择器值』介绍了选择器值可以是什么样的值，以及 CICS 和服务例程如何对其进行使用。
   a. 如果 **ATMP_SELECTOR** 为 NULL 且 **ATMP_ATOMTYPE** 包含值“feed”，那么客户机不会指定特殊的 Atom 条目，而是在保存最近添加到订阅源的 Atom 条目的资源中查找记录。 例如，如果 Atom 条目数据保存在数据库中，那么会使用添加到该数据库的最新记录。
   b. 如果 **ATMP_SELECTOR** 包含选择器值且 **ATMP_ATOMTYPE** 包含值“feed”，那么将在由该选择器值标识的资源中查找记录。 这个值组合可能表明 CICS 需要来自订阅源的第二个或后续 Atom 条目才能完成客户机请求，并且 CICS 正

使用服务例程在上一个迭代中提供的选择器值来请求这些 Atom 条目中的一个。当客户机请求包含特定范围的 Atom 条目（如部分列表）的订阅源文档时，该值组合还用于请求中的第一个 Atom 条目。

c. 如果 **ATMP_SELECTOR** 包含选择器值且 **ATMP_ATOMTYPE** 包含值“entry”，那么将在由该选择器值标识的资源中查找记录。 这个值组合指示客户机正请求订阅源中某个已知的 Atom 条目。

4. 如果您具有包含 Atom 条目数据的资源的 XML 绑定，并且希望使用资源处理参数传递有关资源中字段的信息，那么请对服务例程进行编码，以使用 **ATMP_TITLE_FLD** 参数和其他以 **_FLD** 结尾的参数的值，来确定包含 Atom 条目元素数据的各个字段的名称和长度。 在您为使用资源中数据的 Atom 订阅源设置 Atom 配置文件时，您将需要在 Atom 配置文件的 <cics:fieldnames> 元素中指定这些字段的名称，CICS 会使用资源处理参数将这些名称传递给服务例程。 第 240 页的『DFHATOMPARMS 容器』记录了资源处理参数。

5. 使用 PUT CONTAINER 命令，按照您从资源确定的记录中的描述，创建一个具有 DATATYPE(CHAR) 属性、包含 Atom 条目内容、名为 DFHATOMCONTENT 的容器。 该容器是必需的。样本服务例程 DFH$W2S1 显示如何更新容器，第 250 页的『DFHATOMCONTENT 容器』介绍了在容器中置入什么内容。

6. 如果您从资源确定的记录包含为 Atom 条目提供元数据的任何字段（例如标题），那么会按照第 251 页的『返回容器中的 Atom 条目元数据』中的步骤，使用可选容器将该元数据返回到 CICS。

7. 如果您从资源确定的记录包含提供有关创建或更新数据时的日期和时间戳记的任何字段，那么会将这些日期和时间戳记作为 DFHATOMPARMS 容器中 **ATMP_PUBLISHED** 和 **ATMP_UPDATED** 参数的新值返回。 样本服务例程 DFH$W2S1 显示如何返回这些参数的新值。 有关这些日期和时间戳记的格式信息，请参阅第 225 页的『Atom 条目的日期和时间戳记』。

8. 如果 DFHATOMPARMS 容器中的 **ATMP_SELECTOR** 参数在服务例程输入中为 NULL（说明 Web 客户机未请求特定的 Atom 条目），那么请使用要返回的当前条目的合适的选择器值替换 NULL 值。 样本服务例程 DFH$W2S1 显示如果 **ATMP_SELECTOR** 参数为 NULL，那么将如何返回选择器值。 第 223 页的『Atom 条目的选择器值』说明了如何选择选择器值。 如果 **ATMP_SELECTOR** 参数在服务例程的输入中包含选择器值，请勿对其进行更改。

9. 如果 DFHATOMPARMS 容器中的 **ATMP_ATOMTYPE** 参数具有值“feed”，表示客户机想要多个条目，请检查资源是否包含可用于提供更多 Atom 条目的任何其他旧数据。

   a. 如果存在旧数据，请查找提供 Atom 条目的下一个数据项，并返回该数据项的要用于 **ATMP_NEXTSEL** 参数的合适的选择器值。 第 224 页的『Atom 条目的顺序』说明了您应当以何种顺序返回 Atom 条目。

   b. 如果没有其他数据可用，请将 **ATMP_NEXTSEL** 参数中数据的当前长度设置为零，以返回空值。

10. 请参阅 DFHATOMPARMS 容器中的 **ATMP_ATOMID** 参数，以了解条目的原型 Atom 标识。原型 Atom 标识必须通过附加 **ATMP_SELECTOR** 参数中指定的 Atom 条目选择器值来补全。如果您愿意，服务例程也可以忽略原型 Atom 标识，并用自己有效的 Atom 标识为 Atom 条目替换原型 Atom 标识。 要了解有关 Atom 标识要求的更多信息，请参阅第 226 页的『Atom 条目的 Atom 标识』。

a. 如果您已在 Atom 条目的资源记录中存储了完成的 Atom 标识，请在 **ATMP_ATOMID** 参数中返回该 Atom 标识，后跟该标识的长度。 如果正在使用 DFHATOMPARMS 容器中的资源处理参数，那么 **ATMP_ID_FLD** 参数包含资源中相关字段的名称和长度。

b. 如果资源未存储 Atom 标识，请将 **ATMP_ATOMID** 参数所表示数据的当前长度设置为零。 CICS 将附加选择器值以生成完整的 Atom 标识。

11. 返回合适的响应代码，以在 DFHATOMPARMS 容器的 **ATMP_RESPONSE** 参数中使用。 样本服务例程 DFH$W2S1 显示了如何执行该操作。 代码初始化为零，表示成功补全。 如果返回错误响应，那么 CICS 会生成适当的缺省 HTTP 错误响应以发送至 Web 客户机。第 249 页的『DFHATOMPARMS 容器中的 ATMP_RESPONSE 参数』列出了可用的响应代码以及各种情况下 CICS 发送的 HTTP 错误响应。 样本服务例程 DFH$W2S1 在设置响应代码之后会向 CICS 交还控制权。

下一步做什么

如果您已编写了服务例程，请在 CICS 中创建并安装合适的 PROGRAM 资源定义以描述该服务例程。在您的 PROGRAM 资源定义中，请使用 EXECKEY(USER) 属性。您需要在 ATOMSERVICE 资源定义中为 Atom 订阅源指定该 PROGRAM 资源。

如果您已设置使用服务例程为 Atom 订阅源提供数据的 CICS 定义，那么可以使用 CEDX 事务在服务例程响应 HTTP 请求时对其进行监控和调试。 CW2A 是 Atom 订阅源的缺省别名事务，服务例程将在该事务下运行，除非您设置了备用别名事务。CEDX 监控您所指定事务的下一个实例，因此，如果其他用户在该 CICS 区域中使用相同的别名事务处理 Atom 订阅源，请设置一个备用别名事务，以在您调试自己的服务例程期间使用。

DFHATOMPARMS 容器

DFHATOMPARMS 是一个指定了 DATATYPE（CHAR）的容器，它包含 CICS 在与服务例程（为 Atom 订阅源提供数据）进行通信时所用到的参数。

DFHW2AP 系列副本将 DFHATOMPARMS 容器中传递的参数映射到服务例程。定义了以下副本:

- DFHW2APD（针对汇编程序）
- DFHW2APH（针对 C）
- DFHW2APL（针对 PL/I）
- DFHW2APO（针对 Cobol）

DFHW2CN 系列副本包含 DFHW2AP 系列副本引用的常量值。定义了以下副本:

- DFHW2CND（针对汇编程序）
- DFHW2CNO（针对 Cobol）
- DFHW2CNH（针对 C）
- DFHW2CNL（针对 PL/I）

DFHATOMPARMS 容器中仅针对输入的参数

CICS 使用这些参数为服务例程提供有关 Web 客户机请求的信息。这些参数中包括资源处理参数，如 **ATMP_TITLE_FLD**。

DFHATOMPARMS 容器中每个仅针对输入的参数都是一个双字地址，其中包含指向某个区域的指针以及该区域中数据的当前长度。您的服务例程不能更改这些指针、长度或存储器。

以 **FLD** 结尾的参数用于处理资源。CICS 使用这些资源处理参数为 CICS 提供的正在处理 CICS 资源（如临时存储器队列）的服务例程提供有关资源记录中字段的信息。CICS 从您在 Atom 配置文件中为 <atom:content> 元素和 <cics:fieldnames> 元素指定的属性以及从该资源的 XML 绑定获取字段的名称。如果您希望编写一个用于从 Atom 配置文件中获取有关资源结构的信息的服务例程，而不是将该信息直接编码到服务例程中，那么您可以使用这些参数。如果要使用这些参数，那么必须为包含数据的资源创建一个 XML 绑定。

ATMP_RESNAME
为 Atom 订阅源提供数据的 CICS 资源的名称。对于您的服务例程，该值始终是服务例程的名称。CICS 需要为 CICS 提供的直接处理各种资源的服务例程提供该参数。CICS 从 <atom:content> 元素的 cics:resource 属性获取该信息。

ATMP_RESTYPE
CICS 资源的类型（大写）。资源类型可以是 PROGRAM、TSQUEUE 或 FILE。对于您的服务例程，资源类型始终为 PROGRAM。CICS 需要为 CICS 提供的直接处理各种资源的服务例程提供该参数。CICS 从 <atom:content> 元素的 cics:type 属性获取该信息。

ATMP_ATOMTYPE
正在处理的 Atom 文档的类型（小写）。类型字符串的值为“feed”、“collection”或“entry”。“feed”表明客户机已请求了 Atom 订阅源中的许多条目。“collection”表明客户机已请求了集合中条目的列表。“entry”表明客户机已请求了订阅源或集合中的某个指定的 Atom 条目。

ATMP_HTTPMETH
客户机请求的 HTTP 方法（已补足长度）。HTTP 方法可以是 GET、POST、PUT 或 DELETE。

ATMP_TAG_AUTHORITY
在 Atom 配置文件中 <cics:authority> 元素的 name 属性中指定的权限名称。该权限名称是可用于构造标记 URI 的标准域名或电子邮件地址。如果您选择了这种格式，那么该权限名称将成为原型 Atom 标识的一部分。

ATMP_TAG_DATE
在 Atom 配置文件中 <cics:authority> 元素的 date 属性中指定的日期。该日期与权限名称一起用于构造标记 URI。如果您选择了这种格式，那么该日期将成为原型 Atom 标识的一部分。

ATMP_XMLTRANSFORM
XMLTRANSFORM 资源的名称。当您为 CICS 资源生成 XML 绑定并安装指定该绑定的 ATOMSERVICE 资源时，会创建 XMLTRANSFORM 资源。XMLTRANSFORM 资源将资源中的记录布局描述为 XML 结构。如果该名称长度为零，那么不会为资源创建任何 XML 绑定，并且服务例程必须在资源记录和 Atom 条目元素之间执行自己的映射。

ATMP_ROOT_ELEMENT
由 XMLTRANSFORM 资源映射的 XML 结构的 root 元素的名称。

ATMP_MTYPEIN

Web 客户机 HTTP 请求主体的介质类型。只有当 **ATMP_HTTPMETH** 参数指定的 HTTP 方法为 POST 或 PUT 时，才会出现请求主体。介质类型始终为“application/atom+xml”，这表示 Atom 条目。CICS 会将请求主体传递给 DFHREQUEST 容器中的服务例程。GET 和 DELETE 请求没有请求主体，因此对于这些 HTTP 方法，指针和长度均为零。

ATMP_MTYPEOUT

Atom 条目的期望内容的介质类型，这在针对 Atom 订阅源的 Atom 配置文件的 <atom:content> 元素的 type 属性中指定。在 RFC 4287 中，使用介质类型“text”替代 IANA 介质类型“text/plain”来表示纯文本，使用“html”替代“text/html”，使用“xhtml”替代“application/xhtml+xml”。如果 Atom 配置文件不包含该信息，那么 CICS 会将缺省的介质类型“application/xml”传递给服务例程。服务例程可以使用该介质类型来确定适合于它在 DFHATOMCONTENT 容器中所返回数据的标记。如果您正在使用资源处理参数，并且资源记录中有一个字段用于存储个别 Atom 条目的介质类型，那么 **ATMP_CONTENT_TYPE_FLD** 参数将包含该字段的名称。

ATMP_WINSIZE

订阅源窗口大小。该值是一个数字串，包含要在每个订阅源中返回的缺省条目数或 Web 客户机已请求的备用条目数。该参数仅供参考，因为 CICS 会针对各个条目向服务例程发出一系列请求。

ATMP_ID_FLD

资源记录中某个字段的名称，该字段包含 Atom 条目的 Atom 标识。CICS 从针对 Atom 订阅源的 Atom 配置文件中 <cics:fieldnames> 元素的 atomid 属性获取该字段的名称。如果 CICS 将该信息传递给服务例程，那么服务例程可以使用这个指定字段存储或定位条目的 Atom 标识，并在 **ATMP_ATOMID** 参数中返回该标识。该数据用于条目的 <atom:id> 元素。

ATMP_PUBLISHED_FLD

资源记录中某个字段的名称，该字段包含上次发布资源的时间。CICS 从针对 Atom 订阅源的 Atom 配置文件中 <cics:fieldnames> 元素的 published 属性获取该字段的名称。如果 CICS 将该信息传递给服务例程，那么服务例程可以使用这个指定的字段来查找时间戳记值或 ABSTIME 值，这些值可用于构造在 **ATMP_PUBLISHED** 参数中返回的值。该数据用于条目的 <atom:published> 元素。

ATMP_UPDATED_FLD

资源记录中某个字段的名称，该字段包含上次更新资源的时间。CICS 从针对 Atom 订阅源的 Atom 配置文件中 <cics:fieldnames> 元素的 updated 属性获取该字段的名称。如果 CICS 将该信息传递给服务例程，那么服务例程可以使用这个指定的字段来查找时间戳记值或 ABSTIME 值，这些值可用于构造在 **ATMP_UPDATED** 参数中返回的值。该数据用于条目的 <atom:updated> 元素。

ATMP_EDITED_FLD

资源记录中某个字段的名称，该字段包含上次编辑资源的时间。CICS 从 <cics:fieldnames> 元素的 edited 属性中获取该字段的名称。如果 CICS 将该信息传递给服务例程，那么服务例程可以使用这个指定的字段来查找时间戳记值或 ABSTIME 值，这些值可用于构造在 **ATMP_EDITED** 参数中返回的值。该数据用于条目的 <app:edited> 元素。

ATMP_TITLE_FLD

资源记录中某个字段的名称，该字段包含请求的 Atom 条目的标题。CICS 从 <cics:fieldnames> 元素的 title 属性中获取该字段的名称。如果 CICS 将该信息传递

给服务例程，那么服务例程可以使用这个指定字段定位条目的标题，并在 DFHATOMTITLE 容器中返回该标题。来自 DFHATOMTITLE 容器的数据用于该条目的 <atom:title> 元素。

ATMP_SUMMARY_FLD
资源记录中某个字段的名称，该字段包含请求的 Atom 条目的摘要。CICS 从 <cics:fieldnames> 元素的 summary 属性中获取该字段的名称。如果 CICS 将该信息传递给服务例程，那么服务例程可以使用这个指定字段定位条目的摘要，并在 DFHATOMSUMMARY 容器中返回该摘要。来自 DFHATOMSUMMARY 容器的数据用于该条目的 <atom:summary> 元素。

ATMP_CONTENT_FLD
资源记录中某个字段的名称，该字段包含请求的 Atom 条目的所有内容。CICS 从 <cics:fieldnames> 元素的 content 属性中获取该字段的名称。如果 CICS 将该信息传递给服务例程，那么服务例程可以使用这个指定字段定位条目的内容，并在 DFHATOMCONTENT 容器中返回该内容。来自 DFHATOMCONTENT 容器的数据用于该条目的 <atom:content> 元素。

ATMP_CONTENT_TYPE_FLD
资源记录中某个字段的名称，该字段包含 Atom 条目内容的介质类型，例如 application/xml 或 text。对于 **ATMP_MTYPEOUT** 参数，使用介质类型“text”、“html”和“xhtml”来替换全部 IANA 介质类型。Atom 条目的 <atom:content> 元素的 type 属性中指定了介质类型。CICS 从 <cics:fieldnames> 元素的 content_type 属性中获取该字段的名称。如果 CICS 将该信息传递给服务例程，那么服务例程可以使用这个指定的字段找到内容的介质类型，并确定适合于 DFHATOMCONTENT 容器中内容的标记。如果资源记录中没有字段可用于存储 Atom 条目内容的介质类型，那么会应用在 Atom 配置文件的 <atom:content> 元素的 type 属性中指定的介质类型。CICS 会将该介质类型传递到 **ATMP_MTYPEOUT** 参数中的服务例程。

ATMP_CATEGORY_FLD
资源记录中某个字段的名称，该字段包含应用于请求的 Atom 条目的类别术语。CICS 从 <cics:fieldnames> 元素的 category 属性中获取该字段的名称。如果 CICS 将该信息传递给服务例程，那么服务例程可以使用这个指定字段定位类别，并在 DFHATOMCATEGORY 容器中返回该类别。DFHATOMCATEGORY 容器中的数据用于该条目的 <atom:category> 元素。

ATMP_AUTHOR_FLD
资源记录中某个字段的名称，该字段包含 Atom 条目主要作者的姓名。CICS 从 <cics:fieldnames> 元素的 author 属性中获取该字段的名称。如果 CICS 将该信息传递给服务例程，那么服务例程可以使用这个指定字段定位作者的姓名，并在 DFHATOMAUTHOR 容器中返回该姓名。DFHATOMAUTHOR 容器中的数据用于该条目的 <atom:name> 元素。

ATMP_AUTHORURI_FLD
资源记录中某个字段的名称，该字段包含与 Atom 条目主要作者关联的 Web 站点的 URI。CICS 从 <cics:fieldnames> 元素的 authoruri 属性中获取该字段的名称。如果 CICS 将该信息传递给服务例程，那么服务例程可以使用这个指定字段定位 URI，并在 DFHATOMAUTHORURI 容器中返回该 URI。 DFHATOMAUTHORURI 容器中的数据用于该条目的 <atom:uri> 元素。

ATMP_EMAIL_FLD
资源记录中某个字段的名称，该字段包含 Atom 条目主要作者的电子邮件地址。CICS 从 <cics:fieldnames> 元素的 email 属性中获取该字段的名称。如果 CICS 将

该信息传递给服务例程，那么服务例程可以使用这个指定的字段找到电子邮件地址，并在 DFHATOMEMAIL 容器中返回该地址。DFHATOMEMAIL 容器中的数据用于该条目的 <atom:email> 元素。

DFHATOMPARMS 容器中的输入/输出参数

服务例程使用这些参数为 CICS 提供有关所返回的 Atom 条目的信息。

DFHATOMPARMS 容器中的每个输入/输出参数都是一个三字地址，其中包含指向某个区域的指针、该区域中数据的当前长度以及该区域的最大长度。

要使用参数为 CICS 提供信息，服务例程可以执行以下操作之一:

- 将某些数据复制到指针所指定的区域中，然后将该区域的当前长度设置为数据的长度。用于存储 DFHATOMPARMS 容器中输入/输出参数值的存储器位于用户键中，因此，如果服务例程是用 EXECKEY(USER) 定义的，您就可以访问该存储器。
- 将指针设置为指向服务例程自有存储器中的某些数据，该存储器（例如 TWA 存储器）的生存期必须长于程序的生存期；并将区域的当前长度设置为数据的长度。如果您所拥有值的长度超过所提供区域的最大长度，那么可能需要执行该操作。

如果服务例程没有关于特定参数的信息，而 CICS 必须使用该例程为此参数提供的缺省值，那么该服务例程必须通过将数据的当前长度设置为零以向 CICS 说明这一点。

ATMP_ATOMID

条目的 Atom 标识。Atom 标识是 Atom 条目的唯一标识。要了解有关 Atom 标识格式的更多信息，请参阅 第 226 页的『Atom 条目的 Atom 标识』。

在输入时，CICS 使用该区域向服务例程发送该条目的原型 Atom 标识。根据希望使用标记 URI 格式还是备用格式来生成唯一标识，您可以通过在 Atom 配置文件中包含 <cics:authority> 元素或 <atom:id> 元素来确定原型 Atom 标识的格式。CICS 会忽略 Web 客户机提供的 Atom 标识，并且不会向服务例程传递这些标识。

RFC 4287 中的 Atom 格式规范建议您将 Atom 标识存储在 Atom 条目的资源记录中。对于 POST 请求，如果您的资源可以存储 Atom 标识，那么服务例程必须通过为 Atom 条目附加 **ATMP_SELECTOR** 参数指定的选择器值来完成原型 Atom 标识，然后将完整的 Atom 标识存储在资源记录中由 **ATMP_ID_FLD** 参数指定的相应字段内。如果您愿意，服务例程也可以忽略原型 Atom 标识，并用自己有效的 Atom 标识为 Atom 条目替换原型 Atom 标识。服务例程可以使用 **ATMP_TAG_AUTHORITY** 和 **ATMP_TAG_DATE** 参数的值作为输入来构造 Atom 标识。

请注意，当您使用标记 URI 格式时，如果用户编写的服务例程处理单一 Atom 订阅源，那么生成的 Atom 标识对于该例程提供的 Atom 条目是唯一的，但如果用户编写的服务例程提供多个订阅源，那么该标识就不是唯一的。如果用户编写的服务例程提供了多个订阅源，那么可以为 Atom 标识选择备用格式，或在 Atom 配置文件的 <cics:authority> 元素中为每个订阅源使用一个不同的权限名称或日期。

在输出时，服务例程必须使用 **ATMP_ATOMID** 参数，如下所示:

- 如果您已将完整的 Atom 标识存储在 Atom 条目的资源记录中，那么服务例程必须返回来自资源记录中由 **ATMP_ID_FLD** 参数指定的字段的完整 Atom 标识，后跟该标识的长度。
- 如果您的资源未存储 Atom 标识，那么服务例程必须将 **ATMP_ATOMID** 参数所表示数据的当前长度设置为零。在这种情况中，CICS 会将选择器值附加到原型 Atom 标识之后，以生成完整的 Atom 标识。

ATMP_ETAGVAL

所选资源记录的实体标记（即 Etag）值。要生成实体标记，服务例程可以使用 EXEC CICS BIF DIGEST 命令来计算记录的 SHA-1 摘要，或使用其他合适的方法来生成符合 HTTP/1.1 协议要求的实体标记。

在输入时，CICS 使用 **ATMP_ETAGVAL** 参数为服务例程提供 Atom 条目的任何实体标记。当 Web 客户机发出 PUT 或 DELETE 请求来编辑 Atom 条目时，CICS 要求客户机在包含实体标记的请求中提供 If-Match HTTP 头。如果 CICS 使用该参数提供实体标记，服务例程必须针对现有记录计算该实体标记，并将其与 Web 客户机的实体标记进行比较。如果这两个标记不匹配，表明该条目已由其他代理程序更改，服务例程必须用响应代码 `atmp_resp_etag_no_match` 拒绝该请求。 Web 客户机可能会使用一个星号代替实体标记，以表明即使条目已被其他代理程序更改，也应该编辑或删除该条目，并且服务例程的优先级低于该请求。

在输出时，服务例程必须使用 **ATMP_ETAGVAL** 参数，如下所示:

- 实体标记不用于 Atom 订阅源中的条目。如果当前 Atom 条目是 Atom 订阅源的一部分，那么服务例程必须将数据的当前长度设置为零。
- CICS 需要针对集合中条目的实体标记。如果当前 Atom 条目是集合的一部分，那么服务例程必须计算并返回实体标记。请勿在资源记录中存储实体标记，而是在需要时通过计算得到实体标记。

ATMP_PUBLISHED

服务例程可以使用该参数返回首次发布返回的 Atom 条目的日期和时间。“Published”表示首次创建该数据或该数据首次可用的时间点。如果您的资源未存储该数据，那么服务例程必须通过将该数据的当前长度设置为零以说明这一点，在这种情况中，CICS 将提供当前时间作为缺省值。如果服务例程返回日期和时间戳记，那么该戳记必须采用 RFC 3339 格式（也称为 XML dateTime 数据类型）。您可以使用 EXEC CICS FORMATTIME 命令来提供这种格式的日期和时间，或者如果您的服务例程可以使用 TRANSFORM DATATOXML 命令，那么可以将 CICS ABSTIME 值转换为这种格式的日期和时间戳记。

ATMP_UPDATED

服务例程可以使用该参数返回上次更新返回的 Atom 条目的日期和时间。“Updated”表示对该数据进行重大更改的时间点。如果您的资源未存储该数据，那么服务例程必须通过将该数据的当前长度设置为零以说明这一点，在这种情况中，CICS 将提供当前时间作为缺省值。如果服务例程返回日期和时间戳记，那么该戳记必须采用 RFC 3339 格式。

ATMP_EDITED

服务例程可以使用该参数返回上次编辑返回的 Atom 条目的日期和时间。如果您的资源未存储该数据，那么服务例程必须通过将该数据的当前长度设置为零以说明这一点，在这种情况中，CICS 将提供当前时间作为缺省值。如果服务例程返回日期和时间戳记，那么该戳记必须采用 RFC 3339 格式。

ATMP_SELECTOR

服务例程必须提供的 Atom 条目的选择器值。第 223 页的『Atom 条目的选择器值』说明了什么是选择器值。

- 如果客户机针对订阅源发出了常规请求，那么在输入时，CICS 会针对 **ATMP_SELECTOR** 参数发送一个空值，并且输入参数 **ATMP_ATOMTYPE** 的值为“feed”。接收组合值时，服务例程必须执行以下操作:
  - 返回最近添加到订阅源的 Atom 条目的数据。

– 使用 **ATMP_SELECTOR** 参数返回用于标识该条目的选择器值。如果您的资源未保存条目的 Atom 标识，那么 CICS 将在为条目生成的 Atom 标识中使用该选择器值。

– 使用 **ATMP_NEXTSEL** 参数返回用于标识订阅源中下一个条目的选择器值。

- 如果 CICS 需要来自订阅源的第二个或后续 Atom 条目以完成客户机请求，或客户机请求了包含特定范围的 Atom 条目的订阅源文档，那么在输入时，CICS 会使用 **ATMP_SELECTOR** 参数发送针对订阅源文档中某个 Atom 条目的选择器值，并且输入参数 **ATMP_ATOMTYPE** 的值为“feed”。接收组合值时，服务例程必须执行以下操作:
  – 返回由选择器值表示的 Atom 条目的数据。
  – 不要更改 CICS 为 **ATMP_SELECTOR** 参数提供的数据或长度。
  – 使用 **ATMP_NEXTSEL** 参数返回用于标识订阅源中下一个条目的选择器值。
- 当客户机请求订阅源中的特定条目时，CICS 使用 **ATMP_SELECTOR** 参数发送从该条目的 URL 中抽取的选择器值，并且输入参数 **ATMP_ATOMTYPE** 的值为“entry”。接收组合值时，服务例程必须执行以下操作:
  – 返回由选择器值表示的 Atom 条目的数据。
  – 不要更改 CICS 为 **ATMP_SELECTOR** 参数提供的数据或长度。
  – 对于 **ATMP_NEXTSEL** 参数，通过将数据的当前长度设置为零以返回空值。

注: 对于集合，如果将值“feed”用于 Atom 订阅源，那么 CICS 会将值“collection”用于 **ATMP_ATOMTYPE** 参数。值“entry”与针对集合或 Atom 订阅源中条目的值相同。

ATMP_NEXTSEL
服务例程必须使用该参数返回下一个可用的 Atom 条目（如果有的话）的选择器值。第 224 页的『Atom 条目的顺序』说明了返回 Atom 条目的顺序。

无论服务例程处理订阅源还是集合，都必须提供该值。如果客户机请求单个特定条目（**ATMP_ATOMTYPE** 的值为“entry”），或没有其他数据可用于提供 Atom 条目，就不需要该值。如果不需要该值，那么服务例程必须通过将数据的当前长度设置为零，为该参数返回空值。

CICS 使用服务例程提供的值请求来自该服务例程的更多 Atom 条目，以完成 Atom 文档。如果 Atom 文档已完成，那么 CICS 会使用该值在 Atom 文档中生成 <atom:link rel=″next″> 链接，Web 客户机可用该链接从订阅源检索 Atom 条目的下一个窗口，或者从集合检索 Atom 条目的下一个部分列表。

ATMP_PREVSEL
处理集合的服务例程必须使用该参数返回集合中上一个 Atom 条目（如果有的话）的选择器值。第 224 页的『Atom 条目的顺序』说明了返回 Atom 条目的顺序。

如果 **ATMP_ATOMTYPE** 参数的值为“collection”，那么必须提供该值。如果客户机请求单个特定条目（**ATMP_ATOMTYPE** 的值为“entry”），或没有上一个 Atom 条目，就不需要该值。如果不需要该值，那么服务例程必须通过将数据的当前长度设置为零，为该参数返回空值。

当 Atom 文档完成时，CICS 会使用该值执行一系列针对服务例程的请求，以便在 Atom 文档中生成 <atom:link rel=″previous″> 链接，Web 客户机可使用该链接从集合检索 Atom 条目的上一个部分列表。

对于正在处理普通 Atom 订阅源的服务例程，该值不是必需的。如果 <atom:link rel="previous"> 链接可帮助 Web 客户机从订阅源检索 Atom 条目的上一个窗口，那么可以指定该值。然而，生成该链接的处理的响应时间会增加，因此，仅当将 Web 客户机设置为使用这种格式的导航时，才会为订阅源指定该值。

ATMP_FIRSTSEL

处理集合的服务例程必须使用该参数返回集合中第一个 Atom 条目的选择器值。第 223 页的『Atom 条目的选择器值』说明了返回 Atom 条目的顺序。

如果 **ATMP_ATOMTYPE** 参数的值为“collection”，那么必须提供该值。在与同一 Web 客户机请求相关的后续调用中，CICS 使用 **ATMP_FIRSTSEL** 参数为服务例程提供该选择器值，因此服务例程不需要再次提供该值。

如果客户机请求单个特定条目（**ATMP_ATOMTYPE** 的值为“entry”），就不需要该值。如果不需要该值，那么服务例程必须通过将数据的当前长度设置为零，为该参数返回空值。

当 Atom 文档完成时，CICS 会使用该值在 Atom 文档中生成 <atom:link rel="first"> 链接，Web 客户机可使用该链接从集合检索 Atom 条目的第一个（最新的）部分列表。

对于正在处理普通 Atom 订阅源的服务例程，该值不是必需的。如果 <atom:link rel="first"> 链接可帮助 Web 客户机从订阅源检索 Atom 条目的第一个（最新的）窗口，那么可以指定该值。CICS 不会执行任何其他处理来生成该链接。

ATMP_LASTSEL

处理集合的服务例程必须使用该参数返回集合中最后一个 Atom 条目的选择器值。第 224 页的『Atom 条目的顺序』说明了返回 Atom 条目的顺序。

如果 **ATMP_ATOMTYPE** 参数的值为“collection”，那么必须提供该值。在与同一 Web 客户机请求相关的后续调用中，CICS 使用 **ATMP_LASTSEL** 参数为服务例程提供该选择器值，因此服务例程不需要再次提供该值。

如果客户机请求单个特定条目（**ATMP_ATOMTYPE** 的值为“entry”），就不需要该值。如果不需要该值，那么服务例程必须通过将数据的当前长度设置为零，为该参数返回空值。

当 Atom 文档完成时，CICS 会使用该值在 Atom 文档中生成 <atom:link rel="last"> 链接，Web 客户机可使用该链接从集合检索 Atom 条目的最后一个（最早的）部分列表。CICS 发出仅包含单个条目的最后一个部分列表，即，订阅源中的最后一个条目。Web 客户机可以使用 <atom:link rel="previous"> 链接来检索先前所有的部分列表。

对于正在处理普通 Atom 订阅源的服务例程，该值不是必需的。如果 <atom:link rel="last"> 链接可帮助 Web 客户机从订阅源检索 Atom 条目的最后一个（最早的）窗口，那么可以指定该值。CICS 不会执行任何其他处理来生成该链接。

DFHATOMPARMS 容器中的 ATMP_OPTIONS 参数

ATMP_OPTIONS 参数是包含 64 个选项位的双字地址，这些选项位用于表明您的服务例程正在为可选容器提供诸如 Atom 条目标题之类的数据。该选项字符串通过 **ATMP_OPTIONS_BITS** DSECT 映射。

ATMP_OPTIONS 指向的选项位图有两种映射方式：ATMP_OPTIONS_BITS 和 ATMP_OPTIONS_WORDS。ATMP_OPTIONS_BITS 是语言中使用的一系列用于标识位值的字节和位定义。ATMP_OPTIONS_WORDS 是一对全字值，用于 COBOL，其中位值较难编码。

ATMP_OPTIONS_BITS

在 ATMP_OPTIONS_BITS 中，有意义的位值在字节 ATMP_OUTOPT_BYTE1 中，它们是:

OPTTITLE
服务例程使用 DFHATOMTITLE 容器返回用作条目标题的字符串。

OPTSUMMA
服务例程使用 DFHATOMSUMMARY 容器返回用作条目摘要的字符串。

OPTAUTHOR
服务例程使用 DFHATOMAUTHOR 容器返回用作条目作者姓名的字符串。

OPTAUTHEML
服务例程使用 DFHATOMEMAIL 容器返回用作条目作者电子邮件地址的字符串。

OPTAUTHURI
服务例程使用 DFHATOMAUTHORURI 容器返回用作与条目作者关联的 Web 站点 URI 的字符串。

OPTCATEG
服务例程使用 DFHATOMCATEGORY 容器返回用作条目类别术语的字符串。

ATMP_OPTIONS_WORDS

ATMP_OPTIONS_WORDS 包含以下两个全字值:

ATMP_OPTIONS_IN
存储输入选项值的全字，目前未使用该全字。

ATMP_OPTIONS_OUT
存储输出选项值的全字。与 ATMP_OUTOPT_BYTE1 中的位值相等的全字值是在副本 DFH0W2CO 中指定。可以根据需要同时添加这些值，以生成合适的位图值。

副本 DFH0W2CO 包含用于 ATMP_OPTIONS_OUT 中、表示 ATMP_OUTOPT_BYTE1 中位值的二进制整数，如下所示:

OPTTITLE_NUM
与 OPTTITLE 等效

OPTSUMMA_NUM
与 OPTSUMMA 等效

OPTAUTHOR_NUM
与 OPTAUTHOR 等效

OPTAUTHEML_NUM
与 OPTAUTHEML 等效

OPTCATEG_NUM
与 OPTCATEG 等效

OPTAUTHURI_NUM
与 OPTAUTHURI 等效

DFHATOMPARMS 容器中的 ATMP_RESPONSE 参数

ATMP_RESPONSE 参数是双字地址，用于返回向 CICS 表明成功或错误的响应代码。如果服务例程发送了一个表明错误的响应代码，那么 CICS 将生成一个合适的缺省 HTTP 错误响应并将其发送至 Web 客户机。服务例程可以使用 DFHHTTPSTATUS 容器返回备用的状态代码和文本以覆盖缺省的错误响应。

请勿更改双字地址。第一个全字 **ATMP_RESPONSE_CODE** 包含响应代码。其初始值为 0，表明成功完成。第二个全字 **ATMP_REASON_CODE** 的初始值也是 0，服务例程不能更改该全字；它为将来使用而保留。

ATMP_RESPONSE_CODE 的符号值是在 DFHW2CN 系列副本中定义的。这些值如下:

```
atmp_resp_normal         constant(0);  ! Normal success response
atmp_resp_not_found      constant(4);  ! Resource not found
atmp_resp_not_auth       constant(8);  ! Resource not authorized
atmp_resp_disabled       constant(12); ! Resource is disabled
atmp_resp_already_exists constant(16); ! Resource already exists
atmp_resp_etag_no_match     constant(20); ! If-Match compare failed
atmp_resp_invalid_request   constant(24); ! Request not valid
atmp_resp_access_error      constant(32); ! Other resource error
atmp_resp_conversion_failed constant(36); ! XML Conversion error
```

如果该参数按原样返回，CICS 将发送表明请求成功完成的 HTTP 响应。如果服务例程发送了一个表明错误的响应代码，那么 CICS 将生成一个合适的缺省 HTTP 错误响应并将其发送至 Web 客户机。缺省 HTTP 错误响应如下:

表 12. 服务例程中针对 Atom 订阅源的缺省 HTTP 错误响应

ATMP_RESPONSE_CODE 值	HTTP 状态码	HTTP 状态文本
atmp_resp_normal	200（对于 POST 请求，该值为 201）	OK（对于 POST 请求，该值为“Created”）
atmp_resp_not_found	404	未找到
atmp_resp_not_auth	403	已禁止
atmp_resp_disabled	503	服务不可用
atmp_resp_already_exists	409	重复的资源
atmp_resp_etag_no_match	412	前置条件不满足
atmp_resp_invalid_request	400	无效的请求
atmp_resp_access_error	500	资源错误
atmp_resp_conversion_failed	500	资源错误

当服务例程返回一个错误时，它可以使用 DFHHTTPSTATUS 容器返回备用状态码和文本以替换缺省 HTTP 错误响应。有关可能在错误响应中使用的状态码列表，请参阅第 341 页的附录 C, 『CICS Web Support 的 HTTP 状态码参考』。您无法覆盖成功请求的缺省 HTTP 响应（响应代码为 0）。

DFHATOMCONTENT 容器

DFHATOMCONTENT 是一种 DATATYPE(CHAR) 容器，可用于服务例程，以提供 Atom 条目的内容。

该容器是必需的。CICS 将容器中的数据作为 Atom 条目的 <atom:content> 元素返回。

如果正在使用 DFHATOMPARMS 容器中的资源处理参数，那么 **ATMP_CONTENT_FLD** 参数将具有保存该容器数据的资源记录中字段的名称和长度。您可以使用来自单个字段的数据，也可以使用来自嵌套字段结构的数据。如果未使用资源处理参数，那么可以对服务例程进行编码，以将整个资源记录作为条目的内容返回，或选择资源记录中相应字段组成内容。

您可以提供没有子元素的纯文本内容，也可以使用 XML 或其他类型的标记（例如，HTML 或 XHTML）对数据进行格式化。DFHATOMPARMS 容器中的 **ATMP_MTYPEOUT** 参数包含期望的 Atom 条目内容的介质类型，这在 Atom 订阅源的 Atom 配置文件的 <atom:content> 元素中指定。在 RFC 4287 中，使用介质类型“text”替代 IANA 介质类型“text/plain”来表示纯文本，使用“html”替代“text/html”，使用“xhtml”替代“application/xhtml+xml”。如果在 Atom 配置文件中没有指定介质类型，那么 CICS 会在该参数中提供缺省的介质类型“application/xml”。

您也可以将个别 Atom 条目的介质类型存储在资源记录中。如果正在使用资源处理参数，那么 **ATMP_CONTENT_TYPE_FLD** 参数包含以下字段的名称和长度：该字段包含 Atom 条目内容的介质类型。

Atom 条目内容的 <content> 标记

如果您的内容不采用 CICS 在 DFHATOMPARMS 容器的 **ATMP_MTYPEOUT** 参数中为服务例程提供的介质类型，那么必须在容器中内容起始位置包含开始标记 <content>，并在结束位置包含结束标记 </content>。如果内容采用非纯文本格式，那么还必须向 <atom:content> 标记添加 type 属性以指定内容的介质类型。以下是一些可能的 type 属性：

- <content type=″html″> 指定 HTML。
- <content type=″xhtml″> 指定 XHTML。
- <content type=″text/xml″> 是通常用于人类可读 XML 文档的介质类型。

当提供纯文本时，您可以指定 <content type=″text″>，但如果未指定任何 type 属性，Atom 文档的接收方会假定采用该介质类型。如果内容采用任何其他格式，请指定在因特网上通常用于这种格式的 IANA 介质类型。http://www.iana.org/assignments/media-types/ 中列出了可用的介质类型。请注意，CICS 不支持非文本介质类型。

对于 CICS 在 DFHATOMPARMS 容器的 **ATMP_MTYPEOUT** 参数中为服务例程提供的采用期望介质类型的内容，您可以省略 <content> 和 </content> 标记。在这种情况中，CICS 提供开始和结束标记，并将 type 属性指定为 **ATMP_MTYPEOUT** 参数中提供的介质类型。

Atom 条目内容的标记

如果内容使用的格式不是纯文本，请阅读 RFC 4287: *The Atom Syndication Format*（http://www.ietf.org/rfc/rfc4287.txt）的 4.1.3.3 节中的处理信息。这些规则阐述必须如何排列标记，以及 Atom 文档的接收方（称为“Atom 处理器”）必须如何根据使用的标记

类型来解释和呈现内容。尤其需要注意，HTML 标记必须转义，例如，标记“
”应该编写为“
”。CICS 不会验证您所使用的标记。

如果希望从资源记录中的字段生成 XML 内容，并且资源包含具有关联的 XMLTRANSFORM 资源的 XML 绑定，那么可以使用 CICS 函数将应用程序数据转换为 XML。如果您具有一个语言结构（即副本，用于描述资源 DFHLS2SC 过程所支持的任何高级语言中的数据结构，这些语言包括 COBOL、C、C++ 或 PL/I），那么可以创建一个 XML 绑定。从语言结构生成映射介绍了如何执行该操作。当您安装用于命名 XML 绑定的 ATOMSERVICE 资源定义时，CICS 会动态创建 XMLTRANSFORM 资源。

如果 XMLTRANSFORM 资源可用，那么 CICS 会在 DFHATOMPARMS 容器的 **ATMP_XMLTRANSFORM** 参数中提供该资源的名称。要了解有关 TRANSFORM DATATOXML 命令的更多信息以及使用数据映射功能的指示信息，请参阅 *CICS Application Programming Guide*。

返回容器中的 Atom 条目元数据

如果包含 Atom 条目数据的资源在其记录中具有向各个 Atom 条目提供元数据的字段（例如，标题或摘要），那么请将可选元数据容器（如，DFHATOMTITLE）用于服务例程，以便向 CICS 提供这些数据。

关于此任务

保存 Atom 条目内容的 DFHATOMCONTENT 容器即是服务例程返回给 CICS 所需的容器。大量其他可选容器可用于返回服务例程从资源（包含 Atom 条目的数据）记录中抽取的任何元数据。资源中的记录可能不包含保存元数据的任何字段，例如，当您正在从现有文件（最初没有为用作 Atom 订阅源而进行设置）创建 Atom 订阅源时。如果记录中没有任何元数据，那么无需返回这些容器，但是在某些情况下，如果设置了 Atom 订阅源的 Atom 配置文件，那么需要提供缺省元数据。

样本服务例程 DFH$W2S1 显示了如何使用 PUT CONTAINER 命令创建可选容器，以及如何使用从资源记录中抽取的数据来填充这些容器。

1. 如果资源中的记录包含 Atom 条目的单独标题，请使用 PUT CONTAINER 命令创建具有 DATATYPE(CHAR) 属性、包含该 Atom 条目标题、名为 DFHATOMTITLE 的容器。 该容器是可选的，但是如果您不提供服务例程中的这些数据，那么必须在 Atom 配置文件中指定缺省标题。第 252 页的『DFHATOMTITLE 容器』说明了向容器中放入哪些数据。
2. 如果资源中的条目具有单独的摘要，请使用 PUT CONTAINER 命令创建具有 DATATYPE(CHAR) 属性、包含 Atom 条目摘要、名为 DFHATOMSUMMARY 的容器。 该容器是可选的，但是如果条目的内容不是文本或 XML，那么 Atom 规范需要一个摘要。第 252 页的『DFHATOMSUMMARY 容器』说明了向容器中放入哪些数据。
3. 如果您想要使用服务例程来提供条目作者的姓名，请使用 PUT CONTAINER 命令创建具有 DATATYPE(CHAR) 属性、包含条目作者姓名、名为 DFHATOMAUTHOR 的容器。 该容器是可选的，但是如果您不提供服务例程中的这些数据，那么必须在 Atom 配置文件中指定缺省姓名或接受 CICS 缺省值。第 253 页的『DFHATOMAUTHOR 容器』说明了向容器中放入哪些数据。

4. 如果您想要使用服务例程来提供条目作者的电子邮件地址和 URI（Web 站点地址），请使用 PUT CONTAINER 命令创建具有 DATATYPE(CHAR) 属性、包含这些项的数据、名为 DFHATOMEMAIL 和 DFHATOMAUTHORURI 的容器。 这些容器是可选的，可以全部提供、全部不提供或提供其中一个。第 253 页的『DFHATOMEMAIL 容器』和 第 253 页的『DFHATOMURI 容器』说明了向容器中放入哪些数据。
5. 如果您想要使用服务例程来提供条目的类别，请使用 PUT CONTAINER 命令创建具有 DATATYPE(CHAR) 属性、包含条目类别术语、名为 DFHATOMCATEGORY 的容器。 该容器是可选的。第 254 页的『DFHATOMCATEGORY 容器』说明了向容器中放入哪些数据。
6. 在 DFHATOMPARMS 容器中，通过 **ATMP_OPTIONS_OUT** 参数为每个返回给 CICS 的可选容器设置合适的选项位。 第 240 页的『DFHATOMPARMS 容器』记录了该参数。样本服务例程 DFH$W2S1 显示了如何设置这些选项位。

DFHATOMTITLE 容器

DFHATOMTITLE 是一种 DATATYPE(CHAR) 容器，可用于服务例程，以提供 Atom 条目的标题。

该容器是可选的。CICS 将容器中的数据作为 Atom 条目的 <atom:title> 元素返回。

如果资源记录不包含 Atom 条目的个别标题，那么可以使用 Atom 配置文件为订阅源中的每个条目指定相同的标题。由于 Atom 条目必须具有标题，所以 CICS 需要在 Atom 配置文件中指定一个缺省标题。如果没有适当的缺省标题，您可以使用空白缺省标题，不过在这种情况下您必须使用 DFHATOMTITLE 容器来为每个条目提供标题，从而与 Atom 格式规范保持一致。

如果正在使用 DFHATOMPARMS 容器中的资源处理参数，那么 **ATMP_TITLE_FLD** 参数将具有保存该容器数据的资源记录中字段的名称和长度。

请勿为容器中的数据添加标记，请勿使用任何标记对容器进行格式化。CICS 只支持纯文本的标题。

在为 CICS 提供该容器时，请在 DFHATOMPARMS 容器 **ATMP_OPTIONS** 参数中设置 **OPTTITLE** 位值。

DFHATOMSUMMARY 容器

DFHATOMSUMMARY 是一种 DATATYPE(CHAR) 容器，可用于服务例程，以提供 Atom 条目的摘要。

该容器是可选的。CICS 将容器中的数据作为 Atom 条目的 <atom:summary> 元素返回。

如果资源记录不包含 Atom 条目的个别摘要，那么可以使用 Atom 配置文件为订阅源中的每个条目指定相同的摘要。如果条目的内容不是文本或 XML，那么 Atom 规范需要摘要；因此，在您想要提供不属于这些类别的内容时，必须提供 DFHATOMSUMMARY 容器或使用 Atom 配置文件来提供该数据。CICS 不检查您是否为非文本和非 XML 条目提供摘要，所以您必须自己确保遵循了这方面的规范。

如果正在使用 DFHATOMPARMS 容器中的资源处理参数，那么 **ATMP_SUMMARY_FLD** 参数将具有保存该容器数据的资源记录中字段的名称和长度。

请勿为容器中的数据添加标记，请勿使用任何标记对容器进行格式化。

在为 CICS 提供该容器时，请在 DFHATOMPARMS 容器 **ATMP_OPTIONS** 参数中设置 **OPTSUMMA** 位值。

DFHATOMAUTHOR 容器

DFHATOMAUTHOR 是一种 DATATYPE(CHAR) 容器，可用于服务例程，以提供 Atom 条目作者的姓名。

该容器是可选的。CICS 将容器中的数据作为 Atom 条目的 <atom:author> 元素的 <atom:name> 子元素返回。

Atom 规范要求每个 Atom 条目都具有作者姓名。如果资源中的记录具有指定了个别作者的字段，请提供该数据；否则，服务例程可以为订阅源中的所有条目提供同一作者的姓名。或者，您可以使用订阅源的 Atom 配置文件，为订阅源中的每个条目提供单一作者的姓名。如果未以任何形式提供作者姓名，那么 CICS 在响应 Web 客户机时会使用缺省作者姓名“CICS Transaction Server”。

如果正在使用 DFHATOMPARMS 容器中的资源处理参数，那么 **ATMP_AUTHOR_FLD** 参数将具有保存该容器数据的资源记录中字段的名称和长度。

请勿为容器中的数据添加标记，请勿使用任何标记对容器进行格式化。

在为 CICS 提供该容器时，请在 DFHATOMPARMS 容器 **ATMP_OPTIONS** 参数中设置 **OPTAUTHOR** 位值。

DFHATOMEMAIL 容器

DFHATOMEMAIL 是一种 DATATYPE(CHAR) 容器，可用于服务例程，以提供 Atom 条目作者的电子邮件地址。

该容器是可选的。CICS 将容器中的数据作为 Atom 条目的 <atom:author> 元素的 <atom:email> 子元素返回。

Atom 规范不需要该数据，因此，如果您没有该数据或不希望分发该数据，那么可以将其省略。如果资源中的记录包含个别作者的电子邮件地址，那么可以提供该数据。如果您正在为订阅源中的每个条目提供单个作者的名称，那么可以通过服务例程或在订阅源的 Atom 配置文件中，对电子邮件地址执行相同的操作。

如果正在使用 DFHATOMPARMS 容器中的资源处理参数，那么 **ATMP_EMAIL_FLD** 参数将具有保存该容器数据的资源记录中字段的名称和长度。

请勿为容器中的数据添加标记，请勿使用任何标记对容器进行格式化。

在为 CICS 提供该容器时，请在 DFHATOMPARMS 容器 **ATMP_OPTIONS** 参数中设置 **OPTEMAIL** 位值。

DFHATOMURI 容器

DFHATOMURI 是一种 DATATYPE(CHAR) 容器，可用于服务例程，以提供与 Atom 条目的作者相关的 URI（Web 站点地址）。

该容器是可选的。CICS 将容器中的数据作为 Atom 条目的 <atom:author> 元素的 <atom:uri> 子元素返回。

对于作者的电子邮件地址，ATOM 规范不需要该数据，因此，如果您没有该数据或不希望分发该数据，那么可以将其省略。如果资源中的记录包含个别作者的 Web 站点，那么可以提供该数据。如果所有作者都来自于您的公司，那么可以提供公司主页的 URI。如果您正在为订阅源中的每个条目提供单个作者的名称，那么可以通过服务例程或在订阅源的 Atom 配置文件中，对 URI 执行相同的操作。

如果正在使用 DFHATOMPARMS 容器中的资源处理参数，那么 **ATMP_AUTHORURI_FLD** 参数将具有保存该容器数据的资源记录中字段的名称和长度。

请勿为容器中的数据添加标记，请勿使用任何标记对容器进行格式化。

在为 CICS 提供该容器时，请在 DFHATOMPARMS 容器 **ATMP_OPTIONS** 参数中设置 **OPTAUTHURI** 位值。

DFHATOMCATEGORY 容器

DFHATOMCATEGORY 是一种 DATATYPE(CHAR) 容器，可用于服务例程，以提供 Atom 条目的类别。

该容器是可选的。CICS 将容器中的数据作为 Atom 条目的 <atom:category> 元素的“term”属性返回。CICS 不支持 <atom:category> 元素的可选方案和标签属性。

如果资源记录不包含 Atom 条目的类别，那么您可以使用 Atom 配置文件为订阅源中的每个条目指定相同的类别，前提是您知道这样做对自己有帮助的原因。Atom 规范不需要类别。

如果正在使用 DFHATOMPARMS 容器中的资源处理参数，那么 **ATMP_CATEGORY_FLD** 参数将具有保存该容器数据的资源记录中字段的名称和长度。

请勿为容器中的数据添加标记，请勿使用任何标记对容器进行格式化。

在为 CICS 提供该容器时，请在 DFHATOMPARMS 容器 **ATMP_OPTIONS** 参数中设置 **OPTCATEG** 位值。

Atom 订阅源的 DFH$W2S1 C 样本服务例程

样本服务例程 DFH$W2S1 是用 C 语言编写的框架程序，该程序显示了如何读取 DFHATOMPARMS 容器中的参数、更新元数据和内容容器（例如，DFHATOMTITLE 和 DFHATOMCONTENT）以及更新并返回 DFHATOMPARMS 容器。

如果您为 DFH$W2S1 创建并安装了适当的 RDO 定义（必须指定 EXECKEY(CICS)），那么可以运行样本服务例程来为测试提供某些数据，以响应针对 Atom 订阅源的 GET 请求。正如已经发布的，该程序仅会返回使用其代码设置的缺省数据，而不会处理 POST、PUT 或 DELETE 请求。从保存 Atom 条目数据的资源识别所需的记录、以及从该记录抽取适当字段、或更新该记录以对 POST、PUT 或 DELETE 请求做出响应的过程是专门针对您选择的资源以及该资源中的记录结构的。要获取服务例程如何与资源进行交互的示例，请参阅第 313 页的『Atom 订阅源的 DFH0W2F1 COBOL 样本服务例程』中 DFH0W2F1 的描述。DFH0W2F1 与 CICS 样本文件 FILEA 进行交互。

DFH$W2S1 样本服务例程执行以下任务:

- 包含具有 DFHATOMPARMS 参数列表的副本 DFHW2APH 以及具有响应代码常量的副本 DFHW2CNH 的 C 头文件。
- 定义将放入临时工作区（TWA）的称为“outdata”的结构。该程序会使用此结构存储 DFHATOMPARMS 参数的新数据，并会返回指向该新数据的指针，以便 CICS 替换该参数的现有值。存储 DFHATOMPARMS 参数新数据的存储器必须位于 TWA 中，以便 CICS Atom 处理过程可以读取这些数据。

```
typedef struct
{
    char atomid??(50??);
    char published??(30??);
    ...
    char selector??(20??);
} outdata;
```

- 定义称为“indata”的结构，该结构是一个占位符，显示如何从 DFHATOMPARMS 参数抽取所有值。实际上，服务例程可以抽取这些值，因为它需要这些值来执行每个处理步骤。

```
typedef struct
{
    char* resname;
    char* restype;
    char* atomtype;
    ...
    char* emailfld;
} indata;
```

- 提供一个帮助程序方法 getParameter，用于从 DFHATOMPARMS 的参数中获取指针和长度信息、读取内存中参数的值、添加 NULL 终止符（零字节）以便用作 C 字符串以及返回指向新内存位置的指针。

```
char* getParameter(atmp_parameter* ParamPtr)
{
    char* DataPtr;
    int DataLen;
    DataLen  = ParamPtr->atmp_parameter_len;

    EXEC CICS GETMAIN SET(DataPtr) FLENGTH(DataLen+1)
                      INITIMG(0x00);
    if (DataLen != 0) {
        memcpy(DataPtr, ParamPtr->atmp_parameter_ptr, DataLen);
    }

    return DataPtr;
}
```

- 提供一个帮助程序方法 updatedAtomParam，您可以通过使用该方法，利用指向包含新数据的新字符串值的指针和该字符串的长度来更新 CICS 所发送的 DFHATOMPARMS 参数的指针和长度。

```
void updatedAtomParam(char *value, atmp_parameter *ParamPtr)
{
    ParamPtr->atmp_parameter_ptr = (unsigned long*)value;
    ParamPtr->atmp_parameter_len = strlen(value);
}
```

- 提供一个帮助程序方法 updateAtomContainer，此方法使用 EXEC CICS PUT CONTAINER 命令并用 Atom 条目数据（这些数据来自保存 Atom 条目数据的资源记录中的某个字段）来更新某个元数据和内容容器（例如，DFHATOMTITLE 和 DFHATOMCONTENT），并且还会对 DFHATOMPARMS 提供指针的相应选项位进行设置。

```
void updateAtomContainer(char *cName, char *cChannel, char *value,
                         atmp_options_bits *optPtr)
{
    int len = strlen(value);

    EXEC CICS PUT CONTAINER(cName)
                  CHANNEL(cChannel)
                  FROM(value)
                  FLENGTH(len)
                  FROMCODEPAGE("IBM037");

    /* work out which option to set */
    if (strcmp(cName, "DFHATOMTITLE    ") == 0) {
        (*optPtr).atmp_options_outbit.atmp_outopt_byte1.opttitle = 1;
    }
    else if (strcmp(cName, "DFHATOMSUMMARY  ") == 0) {
        (*optPtr).atmp_options_outbit.atmp_outopt_byte1.optsumma = 1;
    }
    else if (strcmp(cName, "DFHATOMCATEGORY  ") == 0) {
        (*optPtr).atmp_options_outbit.atmp_outopt_byte1.optcateg = 1;
    }
    else if (strcmp(cName, "DFHATOMAUTHOR    ") == 0) {
        (*optPtr).atmp_options_outbit.atmp_outopt_byte1.optauthor = 1;
    }
    else if (strcmp(cName, "DFHATOMAUTHORURI ") == 0) {
        (*optPtr).atmp_options_outbit.atmp_outopt_byte1.optautheml = 1;
    }
    else if (strcmp(cName, "DFHATOMEMAIL     ") == 0) {
        (*optPtr).atmp_options_outbit.atmp_outopt_byte1.optauthuri = 1;
    }

}
```

- 在主程序中，设置变量（例如，存储将放入元数据和内容容器（如 DFHATOMTITLE 和 DFHATOMCONTENT ）的数据的存储器），以及“outdata”和“indata”结构。
- 发送 EXEC CICS ADDRESS TWA 命令，以设置 TWA 中存储“outdata”结构的存储器，并用缺省值填充该结构。在您运行所提供的样本服务例程时，会返回这些缺省值。通过将 **ATMP_NEXTSEL** 参数的值设置为 0，样本服务例程会使 CICS 不断请求相同的 Atom 条目（由缺省数据组成），直到获得 Atom 订阅源文档的窗口指定的条目数。当您对提供实际数据的服务例程进行编码时，请勿使用缺省值，因为根据请求方法的不同，您可能希望大部分数据项保持不变。

```
EXEC CICS ADDRESS TWA(outData);

    strcpy(outData->atomid,
           "cics-atomservice-sample:2009-02-02T00:00:00Z");
    strcpy(outData->published, "2009-02-02T00:00:00Z");
    strcpy(outData->updated, "2009-02-20T14:54:34Z");
    strcpy(outData->edited, "2009-02-20T14:54:34Z");
    strcpy(outData->etagval, "");
    strcpy(outData->selector, "");
    strcpy(outData->nextsel, "0");
    strcpy(outData->prevsel, "");
    strcpy(outData->firstsel, "");
    strcpy(outData->lastsel, "");
```

- 发送 EXEC CICS GETMAIN 命令，以获取包含元数据和内容容器（例如，DFHATOMTITLE 和 DFHATOMCONTENT）值的存储器，并用缺省数据填充这些容器。这些容器中的数据不需要位于 TWA 中。DFHATOMCONTENT 容器中 Atom 条目的内容必须附有 <atom:content> 打开和关闭标记，它可能指定确定内容类型的类型属性。您可以参照样本服务例程将 <atom:content> 缩写为 <content>。

```
EXEC CICS GETMAIN SET(AtomContent)   FLENGTH(512) INITIMG(0x00);
EXEC CICS GETMAIN SET(AtomTitle)     FLENGTH(50) INITIMG(0x00);
EXEC CICS GETMAIN SET(AtomSummary)   FLENGTH(100) INITIMG(0x00);
EXEC CICS GETMAIN SET(AtomCategory)  FLENGTH(30) INITIMG(0x00);
EXEC CICS GETMAIN SET(AtomAuthor)    FLENGTH(40) INITIMG(0x00);
EXEC CICS GETMAIN SET(AtomAuthorUri) FLENGTH(256) INITIMG(0x00);
EXEC CICS GETMAIN SET(AtomEmail)     FLENGTH(256) INITIMG(0x00);

strcpy(AtomContent,
       "<content type='text/xml'>Hello world</content>");
strcpy(AtomTitle,     "Sample Service Routine Entry");
strcpy(AtomSummary,
       "This is an entry from the sample service routine");
strcpy(AtomCategory,  "sample-entry");
strcpy(AtomAuthor,    "CICS Sample service routine");
strcpy(AtomAuthorUri, "");
strcpy(AtomEmail,     "");
```

- 发送 EXEC CICS ASSIGN CHANNEL 命令，以获取当前通道的名称，供后续 GET 和 PUT CONTAINER 命令使用。

```
EXEC CICS ASSIGN CHANNEL(cChannel);
```

- 检查 DFHREQUEST 容器是否存在，CICS 使用该容器将 Web 客户机的 POST 或 PUT 请求的主体传递给服务例程。第一个 EXEC CICS GET CONTAINER 命令请求容器的长度，如果返回非零响应代码（表示容器不存在或者存在某些问题），将中断请求。如果请求主体已通过，那么样本服务例程将读取其存储器中的内容。

```
EXEC CICS GET CONTAINER("DFHREQUEST      ")
             CHANNEL(cChannel)
             NODATA
             FLENGTH(DataLen)
             RESP(Resp) RESP2(Resp2);

if(Resp != 0 || Resp2 != 0)
{
    DataLen = -1;
}

/* If we have a request body then get it */
if(DataLen != -1)
{
    DataLen++;
    EXEC CICS GETMAIN SET(RequestBody) FLENGTH(DataLen)
                      INITIMG(0x00);

    EXEC CICS GET CONTAINER("DFHREQUEST      ")
                      INTO(RequestBody)
                      FLENGTH(DataLen);
}
```

- 发送 EXEC CICS GET CONTAINER 命令，以将指针 ParamList 设置到 DFHATOMPARMS 参数的数据起始点。

```
EXEC CICS GET CONTAINER("DFHATOMPARMS    ")
             SET(ParamListData)
             FLENGTH(DataLen);

ParamList = (atmp_parameter_list*)ParamListData;
```

- 浏览 DFHATOMPARMS 参数的数据，使用帮助程序方法 getParameter 来获取每个参数的值，并在“inData”结构中为参数设置指针。

```
optPtr = (atmp_options_bits*)ParamList->atmp_options;

ParamPtr = (atmp_parameter*)ParamList->atmp_resname;
inData.resname = getParameter(ParamPtr);
```

```
ParamPtr = (atmp_parameter*)ParamList->atmp_restype;
inData.restype = getParameter(ParamPtr);

...

ParamPtr = (atmp_parameter*)ParamList->atmp_email_fld;
inData.emailfld = getParameter(ParamPtr);
```

- 根据 DFHATOMPARMS 参数和 DFHREQUEST 容器中的信息，指明您必须提供与保存 Atom 条目数据的资源进行交互的代码的位置。服务例程需要在资源中找到所需的记录，并从该记录抽取中适当的字段以响应 GET 请求，或者更新记录以响应 POST、PUT 或 DELETE 请求。**ATMP_HTTPMETH** 参数的值向服务例程告知所用的请求方法。
- 显示如何将从资源记录中获取的数据放到 outData 结构中指针指向的内存位置中以及元数据和内容容器（例如，DFHATOMTITLE 和 DFHATOMCONTENT）的存储器中。该示例显示了当服务例程的输入中 **ATMP_SELECTOR** 参数为 null时，如何为当前记录添加选择器值。

```
 *  strcpy(outData->atomid,    ".....");
 *  strcpy(outData->published, ".....");
 *  strcpy(outData->updated,   ".....");
 *  strcpy(outData->edited,    ".....");
 *  strcpy(outData->etagval,   ".....");
 *  strcpy(outData->nextsel,   ".....");
 *  strcpy(outData->prevsel,   ".....");
 *  strcpy(outData->firstsel,  ".....");
 *  strcpy(outData->lastsel,   ".....");
 *
 *  if (strlen(inData->selector) == 0) {
 *      strcpy(outData->selector, ".....");
 *  }
 *
 *  strcpy(AtomContent,   "....");
 *  strcpy(AtomTitle,     "....");
 *  strcpy(AtomSummary,   "....");
 *  strcpy(AtomCategory,  "....");
 *  strcpy(AtomAuthor,    "....");
 *  strcpy(AtomAuthorUri, "....");
 *  strcpy(AtomEmail,     "....");
```

- 使用 updatedAtomParam 帮助程序方法并用新数据项的指针和长度更新每个 DFHATOMPARMS 参数的存储器。

```
updatedAtomParam(outData->atomid,
                 (atmp_parameter*)ParamList->atmp_atomid);
updatedAtomParam(outData->published,
                 (atmp_parameter*)ParamList->atmp_published);
updatedAtomParam(outData->updated,
                 (atmp_parameter*)ParamList->atmp_updated);
updatedAtomParam(outData->edited,
                 (atmp_parameter*)ParamList->atmp_edited);
updatedAtomParam(outData->etagval,
                 (atmp_parameter*)ParamList->atmp_etagval);
updatedAtomParam(outData->nextsel,
                 (atmp_parameter*)ParamList->atmp_nextsel);
updatedAtomParam(outData->prevsel,
                 (atmp_parameter*)ParamList->atmp_prevsel);
updatedAtomParam(outData->firstsel,
                 (atmp_parameter*)ParamList->atmp_firstsel);
updatedAtomParam(outData->lastsel,
                 (atmp_parameter*)ParamList->atmp_lastsel);
```

```
if (strlen(outData->selector) != 0) {
    updatedAtomParam(outData->selector,
                     (atmp_parameter*)ParamList->atmp_selector);
}
```

- 使用 updateAtomContainer 帮助程序方法，将新数据添加到元数据和内容容器中，并设置选项位以表明每个容器是否存在。

```
updateAtomContainer("DFHATOMTITLE    ",cChannel,AtomTitle,optPtr);
updateAtomContainer("DFHATOMSUMMARY  ",cChannel,AtomSummary,optPtr);
updateAtomContainer("DFHATOMCATEGORY ",cChannel,AtomCategory,optPtr);
updateAtomContainer("DFHATOMAUTHOR   ",cChannel,AtomAuthor,optPtr);
updateAtomContainer("DFHATOMAUTHORURI",cChannel,AtomAuthorUri,optPtr);
updateAtomContainer("DFHATOMEMAIL    ",cChannel,AtomEmail,optPtr);
updateAtomContainer("DFHATOMCONTENT  ",cChannel,AtomContent,optPtr);
```

- 从 DFHW2CNH 副本定义的选择中提供响应代码。由于当前原因码被 CICS 忽略并留做以后使用，因此其内容是随意的。

```
ResponsePtr = (atmp_responses*)ParamList->atmp_response;
ResponsePtr->atmp_response_code = ATMP_RESP_NORMAL;
/*
 * ResponsePtr->atmp_response_code = ATMP_RESP_NOT_FOUND;
 * ResponsePtr->atmp_response_code = ATMP_RESP_NOT_AUTH;
 * ResponsePtr->atmp_response_code = ATMP_RESP_DISABLED;
 * ResponsePtr->atmp_response_code = ATMP_RESP_ALREADY_EXISTS;
 * ResponsePtr->atmp_response_code = ATMP_RESP_ACCESS_ERROR;
 */
ResponsePtr->atmp_reason_code = 0;
```

- 当主程序结束时，会自动将控制权返回给 CICS。 CICS Atom 处理过程使用元数据和内容容器中的数据以及 TWA 中 DFHATOMPARMS 参数（服务例程为其提供了新指针和长度）的数据来构造 Atom 条目。然后，CICS 会以同样的方式请求服务例程中的其他 Atom 条目，直到构造 Atom 订阅源文档的条目窗口完成为止。

第 23 章 为 Atom 订阅源设置 CICS 定义

要通过 CICS 为 Atom 订阅源提供服务，请设置 URIMAP 和 ATOMSERVICE 资源定义以及 Atom 配置文件。您还可以设置备用别名事务。

开始之前

在通过 CICS 为 Atom 订阅源提供服务之前，您必须配置 CICS Web Support 的基本组件（如果您尚未对 CICS 区域进行此配置），以将 CICS 设置为 HTTP 服务器。第 49 页的第 4 章，『配置 CICS Web Support 基本组件』说明了如何执行此操作。

针对您希望 Web 客户机用于发出 Atom 订阅源 HTTP 请求的端口，您的 CICS 区域必须具有 TCPIPSERVICE 资源定义。如果您已对该 CICS 区域进行了设置并将其用作 HTTP 服务器，那么它可能已具有合适的 TCPIPSERVICE 资源，如 HTTP（端口 80）公认端口号的定义。如果先前未将该 CICS 区域用作 HTTP 服务器并且您没有任何 TCPIPSERVICE 资源，或者如果您希望针对尚未在 CICS 区域中定义的 Atom 订阅源使用非标准端口号，那么就需要定义新的 TCPIPSERVICE 资源。第 83 页的『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』说明了如何定义 TCPIPSERVICE 资源。

您还必须选择为 Atom 条目提供数据的资源，并创建 XML 绑定或服务例程，以支持该数据的交付。有关资源选项的更多信息以及创建 XML 绑定或服务例程的指示信息，请参阅第 233 页的第 22 章，『设置资源以提供 Atom 条目数据』。

关于此任务

完成本部分列出的任务，以设置 CICS 定义，进而提供由所选资源中数据组成的 Atom 订阅源，然后使用您创建的 XML 绑定或服务例程交付该 Atom 订阅源。

完成这些任务后，您就拥有了一个 Atom 订阅源，而 Web 客户机可以通过访问此订阅源来获取 Atom 格式的条目列表。Web 客户机必须自行处理和显示数据。许多免费或收费的 Web 客户机应用程序都可以请求、接收和显示 Atom 订阅源，这些应用程序包括专用的订阅源阅读器以及提供更多功能的类似应用程序，例如，用于创建 mashup 的应用程序。检查该应用程序是否描述为支持 Atom 格式。您还可以编写自己的 Web 客户机应用程序来发出针对 Atom 订阅源数据的 GET 请求。

相关任务

第 298 页的『向 Atom 订阅源或集合发出 GET 请求』
Web 客户机可以通过向 Atom 订阅源或集合的 URL 发出 HTTP GET 请求来获取该订阅源或集合中现有 Atom 条目的列表，也可以向 Atom 条目的 URL 发出 HTTP GET 请求以便从 Atom 订阅源或集合中获取个别 Atom 条目。

为 Atom 订阅源创建别名事务

别名事务完成 Atom 订阅源的后期处理阶段。CICS 为缺省 Atom 订阅源别名事务 CW2A 提供资源定义。如果您希望定义备用别名事务，请设置 TRANSACTION 资源定义。

关于此任务

对于由 CICS Web Support 处理的非 Atom HTTP 请求，您只可在用户编写的应用程序处理这些请求时使用别名事务。但是，对于 Atom 订阅源，无论是否涉及用户编写的服务例程，别名事务都将用于处理所有请求。

出于以下目的，您可能需要使用备用别名事务的名称:

- 审计、监控或记帐
- 修改资源和命令安全性设置
- 分配启动优先级
- 分配 DB2 资源
- 将不同的失控值指定给不同的 CICS 应用程序
- 事务类限制

您可以根据需要设置足够数量的别名事务定义。您可以使用 URIMAP 定义来指定特殊请求所需的别名事务。

CW2A 指定 RESSEC(YES) 和 CMDSEC(YES)，这表示如果对 CICS 区域启用了资源和命令安全性，那么它将应用于该事务。如果您为别名事务指定资源和命令安全性，那么您将需要向 Web 客户机提供适当的许可权以访问该事务使用的资源和命令。有关 Atom 订阅源和集合的安全性的更多信息，请参阅第 317 页的第 26 章，『Atom 订阅源的安全性』。

请按照 *CICS Resource Definition Guide* 中的指示信息来创建事务资源定义。如果您正在遵循这些指示信息进行操作，请注意以下几点:

- 在 CW2A 定义的基础上建立您的别名事务定义，并进行所需的任何更改。 以下是 CW2A 的定义:

```
DEFINE TRANSACTION(CW2A)   GROUP(DFHWEB2)
       PROGRAM(DFHW2A)     TWASIZE(512)
       PROFILE(DFHCICST)   STATUS(ENABLED)
       TASKDATALOC(ANY)    TASKDATAKEY(CICS)
       RUNAWAY(SYSTEM)     SHUTDOWN(DISABLED)
       PRIORITY(1)         TRANCLASS(DFHTCL00)
       DTIMOUT(NO)         TPURGE(NO)        SPURGE(YES)
       RESSEC(YES)         CMDSEC(YES)
       DESCRIPTION(CICS Web2.0 Atomservice alias transaction)
```

- 您的别名事务定义必须使用 CICS 提供的别名程序 DFHW2A。 别名程序访问用户编写的服务例程或在 AtomSERVICE 定义中指定的 CICS 资源。
- 您的别名事务定义必须是本地事务。
- 确保别名事务的优先级等于或高于与 Web 连接任务（例如 CWXN 或 CWXU）关联的事务的优先级。 *CICS Performance Guide* 说明了这一点的重要性。

为 Atom 文档创建 URIMAP 资源定义

为 Atom 文档创建 URIMAP 资源以处理来自 Web 客户机的传入请求，并为该文档指定 ATOMSERVICE 资源。

开始之前

为支持 URIMAP 资源定义，您必须拥有为 CICS Web Support 定义入站端口（CICS 在其上接收 HTTP 请求）的 TCPIPSERVICE 定义。您可以在 URIMAP 资源定义中指定特定的 TCPIPSERVICE 定义，以为 Atom 订阅源请求指定单一端口。您也可以省略 TCPIPSERVICE 定义名称以使 URIMAP 资源定义适用于所有入站端口，但在这种情况下，您的 CICS 区域必须至少安装了一个合适的 TCPIPSERVICE 定义，该 TCPIPSERVICE 定义用于为 HTTP 请求定义入站端口。可以在 TCPIPSERVICE 资源定义中指定要应用到端口的安全措施（例如 SSL），并且可以使用这些安全措施防止对 Atom 订阅源和集合进行未授权访问。有关设置 TCPIPSERVICE 定义的指示信息以及如果您的 CICS 区域尚不具有一个合适的 TCPIPSERVICE 定义，请参阅第 83 页的『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』。

关于此任务

CICS 资源组 DFH$WEB2 包含两个用于 Atom 集合的样本 URIMAP 资源定义: DFH$W2F1 和 DFH$W2Q1。 DFH$W2Q1 用于交付 CICS TS for z/OS V4.1 信息中心（可在 `https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp` 站点找到）的 Web 2.0 方案“创建 Atom 订阅源来使用员工信息”中描述的样本 Atom 集合。

CICS Resource Definition Guide 介绍了资源定义的各种方法，并提供了关于所有将在该过程中使用的 URIMAP 资源定义属性的完整参考信息。

1. 确定您计划用于 Web 客户机来获取 Atom 文档的 URL。 第 217 页的『来自 CICS 的 Atom 订阅源的 URL』 说明了如何为 Atom 订阅源构建 URL。 该 URL 必须包含以下项:
   - `http` 或 `https` 方案组成部分。
   - CICS Web Support 实施的合适主机名，例如 `www.example.com`。如果您指定单个星号作为 HOST 属性，那么 URIMAP 定义与入站 URL 中的任何主机名匹配。
   - 对于 Atom 订阅源文档或集合，路径部分的开始部分，此部分可扩展为提供到整个文档或单个条目的链接，其格式可以是 Atom 格式或备选格式。可以在 URIMAP 资源定义中使用星号作为通配符来指定路径部分的此公共部分（例如 `/myatomfeed/*`），并在 Atom 配置文件的 <atom:link> 元素中指定订阅源的完整路径。CICS 使用该完整路径来确定和返回 Atom 订阅源中的相应项。

     注: 对于该 Atom 订阅源或集合（是通过指定主机名提供服务的所有 Atom 订阅源和集合的一部分），路径部分的公共部分必须是唯一的。如果通过主机名提供任何其他 Web 资源，那么路径部分的公共部分不能与任何这些资源的路径开始部分相同。
   - 对于 Atom 服务或类别文档，针对服务或类别文档的适当 URL 的完整路径部分，例如 `/servicedocument`。

     注: Atom 服务或类别文档的路径不能以通过指定主机名提供服务的任何 Atom 订阅源和集合的路径部分的公共部分开始。
2. 使 URIMAP 定义以您选择的名称和组开头，如 *CICS Resource Definition Guide* 中所述。
3. 使用 STATUS 属性指定在启用状态还是在禁用状态安装 URIMAP 定义。
4. 指定 Atom 的 USAGE 属性。

5. 指定 SCHEME 属性作为 HTTP 请求的 URL 的方案部分。可使用 HTTP（不带 SSL）或 HTTPS（带 SSL）。 不要包含跟随方案部分的定界符 **://**。 指定 HTTP 方案的 URIMAP 接受使用 HTTP 方案或更安全的 HTTPS 方案发出的 Web 客户机请求。指定 HTTPS 方案的 URIMAP 仅接受使用 HTTPS 方案发出的 Web 客户机请求。
6. 如果您需要区别包含不同主机名的 URL，那么指定 HOST 属性作为 HTTP 请求的 URL 的主机部分。 不要包含端口号。 IPv4 或 IPv6 地址可用作主机名。 如果您指定单个星号作为 HOST 属性，那么 URIMAP 定义与入站 URL 中的任何主机名匹配。如果您不使用多个主机名或不打算区别它们，请使用该选项。
7. 对于 Atom 订阅源文档或集合，指定 PATH 属性作为 Atom 订阅源请求的 URL 路径部分的公共部分，其后跟一个作为通配符的星号。 对于 Atom 服务或类别文档，指定 PATH 属性作为文档的完整路径部分。 可以在路径的开始处包含或省略定界符 **/**（正斜杠）；如果省略，CICS 会自动提供该符号。
8. 可选: 指定 TCPIPSERVICE 属性作为 TCPIPSERVICE 定义的名称，它定义与该 URIMAP 定义相关的入站端口。 如果不指定该属性，那么 URIMAP 定义将应用于任何入站端口上的匹配 HTTP 请求。 当将 HTTPS（带 SSL 的 HTTP）作为方案的 URIMAP 定义与 Web 客户机发出的请求进行匹配时，CICS 检查该请求使用的入站端口是否使用 SSL。如果没有为该端口指定 SSL，那么将拒绝该请求，状态码为 403（禁止）。当 URIMAP 定义应用到所有入站端口时，该检查会确保 Web 客户机无法使用不安全的端口来访问安全的资源。由于不会对采用 HTTP 方案的 URIMAP 定义进行任何检查，所以 Web 客户机可以使用非安全或安全的（SSL）端口访问这些资源。
9. 可选: 如果您按照第 261 页的『为 Atom 订阅源创建别名事务』中的过程设置了备用别名事务，请指定 TRANSACTION 属性作为该别名事务的名称。 Atom 订阅源的缺省别名事务是 CW2A，位于 DFHWEB2 组中。
10. 指定 ATOMSERVICE 属性作为 Atom 文档的 ATOMSERVICE 资源定义的名称。ATOMSERVICE 资源确定为 Atom 文档提供数据的资源。
11. 可选: 指定 USERID 属性作为用于连接别名事务的缺省用户标识。 您在 URIMAP 定义中指定的用户标识将由从 Web 客户机获得的任何认证的用户标识（使用由连接的 TCPIPSERVICE 定义的 AUTHENTICATE 属性指定的认证方法）覆盖。如果未使用这些方法之一指定用户标识，那么缺省用户标识为 CICS 缺省用户。 当 CICS 区域的资源和命令安全性处于活动状态时，CICS 在安全性检查中会使用该用户标识。有关 Atom 订阅源安全性的更多信息，请参阅第 317 页的第 26 章，『Atom 订阅源的安全性』。有关如何在安全性检查中使用 Web 客户机的用户标识的说明，请参阅第 157 页的『应用程序生成的响应的资源和事务安全性』。

为 Atom 订阅源创建 Atom 配置文件

创建 Atom 配置文件，以为 Atom 订阅源提供元数据和指示 CICS 资源提供的数据。您的 Atom 配置文件可基于样本 Atom 配置文件 `filea.xml`。

关于此任务

要创建 Atom 配置文件，请复制 `filea.xml` 样本 Atom 配置文件并进行更改，以便为您的 Atom 订阅源指定信息。安装 CICS Transaction Server 时，样本 Atom 配置文件

将安装在 z/OS UNIX 中 CICS 文件的根目录（由 CICS 系统初始化参数 USSHOME 指定）的 /samples/web2.0/atom 子目录中。 USSHOME 的缺省值是 /usr/lpp/cicsts/cicsts41。

使用扩展名 .xml 重命名 filea.xml 样本的副本，并将其存储到所选的 z/OS UNIX 系统服务目录中。您可以使用 XML 编辑器或文本编辑器编辑 Atom 配置文件。

请在 Atom 配置文件中执行以下更改:

1. 在根元素 <cics:atomservice> 中，将属性 type="collection" 更改为 type="feed"。 在 filea.xml 中指定根元素，如下所示:

   ```
   <cics:atomservice type="collection"
        xmlns:cics="http://www.ibm.com/xmlns/prod/cics/atom/atomservice"
        xmlns:atom="http://www.w3.org/2005/Atom">
   ```

   第 268 页的『<cics:atomservice> 元素』提供了有关该元素的参考信息。

2. 如果希望更改每个 Atom 订阅源文档中 CICS 应返回的 Atom 条目数，那么请在 <cics:feed> 元素中更改属性 window="6"。 filea.xml 中指定的元素如下:

   ```
   <cics:feed window="6">
   ```

   如果除去 window 属性，那么 CICS 将使用 window 缺省值（8 个条目）。 第 269 页的『<cics:feed> 元素』提供了有关该元素的参考信息。

3. 在 <cics:resource> 元素中，使用名称和类型属性为向 Atom 订阅源提供数据的 CICS 资源指定名称（1 至 16 个字符）和类型。 filea.xml 包含指定 FILEA 样本的 <cics:resource> 元素，如下所示:

   ```
   <cics:resource name="FILEA" type="file">
   </cics:resource>
   ```

   以大写形式指定该资源名称。资源类型可为“file”、“tsqueue”或“program”。 如果您正在使用服务例程为 Atom 订阅源提供数据，请指定服务例程的名称并将资源类型指定为“program”，而不指定服务例程抽取数据的资源的名称。 第 269 页的『<cics:resource> 元素』提供了有关该元素的参考信息。

4. 在 <cics:bind> 元素中，指定为包含 Atom 条目数据的资源设置的 XML 绑定文件的顶级数据结构（根元素）的名称。

   - 如果 CICS 直接从文件或临时存储器队列获取数据，那么请为该文件或临时存储器队列指定来自 XML 绑定文件的根元素的名称。
   - 如果您正在使用服务例程为 Atom 订阅源提供数据，并且可以为服务例程抽取数据的资源设置 XML 绑定文件，那么请指定来自 XML 绑定文件的根元素的名称。
   - 如果您正在使用服务例程，但没有保存数据的资源的 XML 绑定文件，那么将省略 <cics:bind> 元素。

   filea.xml 指定 FILEA 样本的 XML 绑定文件，如下所示:

   ```
   <cics:bind root="DFH0CFIL"/>
   ```

   第 269 页的『<cics:resource> 元素』提供了有关 <cics:bind> 元素的参考信息。

5. 在 <cics:authority> 元素中，指定 CICS 可用于为 Atom 订阅源和 Atom 条目生成 Atom 唯一标识的适当的权限名称和日期。 filea.xml 包含 <cics:authority> 元素的示例，如下所示:

   ```
   <cics:authority name="example.com" date="2009-02-14"/>
   ```

第 270 页的『<cics:authority> 元素』说明了如何选择权限名称和日期，以及希望使用 Atom 标识的备用格式时应执行的操作。

6. 如果需要与使用 CA8K SupportPac 开发的应用程序兼容或者 Atom 条目的选择器值无法以字符串形式表示，那么请更改 <cics:selector> 元素。 样本 Atom 配置文件中指定的选择器样式是缺省“segment”样式，它将标准 URL 样式用于各个 Atom 条目：

   ```
   <cics:selector style="segment"/>
   ```

   第 271 页的『<cics:selector> 元素』提供了有关该元素的参考信息。

7. 如果包含 Atom 条目数据的资源在其记录中具有用于保存 Atom 条目元数据的字段（例如，标题或时间戳记），那么请添加 <cics:fieldnames> 元素作为 <cics:resource> 元素的子元素，并使用该元素的属性来为包含元数据的字段指定名称。 第 272 页的『<cics:fieldnames> 元素』提供了有关该元素的信息。 由于为样本 Atom 订阅源提供数据的 FILEA 文件不包含任何保存 Atom 条目元数据的字段，因此样本 Atom 配置文件不包含该元素。如果包含 Atom 条目的资源也不具有任何元数据，那么请勿添加 <cics:fieldnames> 元素。

8. 在 <atom:feed> 元素的子元素 <atom:link rel=″self″ href=″ ″> 中，指定 Web 客户机可用于检索 Atom 订阅源文档的 URL。 路径的开始部分必须与您在 URIMAP 资源定义中为订阅源指定的局部路径相匹配。例如，如果指定了 /myatomfeed/* 作为 URIMAP 资源定义中的路径部分，那么您可以在 Atom 配置文件中指定 <atom:link rel=″self″ href=″/myatomfeed/feed″>。 filea.xml 显示了以下示例链接：

   ```
   <atom:link rel="self" href="/atom/f/filea/feed"/>
   ```

   第 274 页的『<atom:feed> 元素』提供了有关 <atom:feed> 元素的所有子元素的参考信息。

9. 更改 <atom:feed> 元素的其他子元素，以指定 Atom 订阅源的元数据（例如，标题和主要作者的姓名）。 filea.xml 包含合适的子元素选择，如下所示：

   ```
   <atom:title>Sample CICS file FILEA</atom:title>
   <atom:subtitle>A RESTFUL service for FILEA</atom:subtitle>
   <atom:icon>../../../web2.0/image/cics-icon.gif</atom:icon>
   <atom:logo>../../../web2.0/image/cics-logo.gif</atom:logo>
   <atom:rights>Copyright (c) Example Corp 2009</atom:rights>
   <atom:category term="Demonstration"/>
   <atom:author>
       <atom:name>CICS Development</atom:name>
       <atom:email>cics@example.com</atom:email>
       <atom:uri>http://www.example.com/cics</atom:uri>
   </atom:author>
   ```

   第 274 页的『<atom:feed> 元素』列出了所有可能的子元素，并说明了 Atom 规范是否需要这些子元素。

10. 在 <atom:entry> 元素的子元素 <atom:link rel=″self″ href=″ ″> 中，指定 CICS 可用于为各个 Atom 条目创建 URL 的标准路径。 CICS 通过将 Atom 条目的选择器值附加到该标准路径来创建 URL。 路径的开始部分必须与您在 URIMAP 资源定义中为订阅源指定的局部路径相匹配。标准路径的其余部分不得与订阅源的路径相同。例如，如果将 /myatomfeed/* 指定为 URIMAP 资源定义中的路径部分，并指定 <atom:link rel=″self″ href=″/myatomfeed/feed.atom″> 作为 Atom 配置文件中整个订阅源的链接，那么可以将 <atom:link rel=″self″ href=″/myatomfeed/entries/″> 指定为条目的标准路径。对于具有选择器值 23 的 Atom 条目，如果您

在 <cics:selector> 元素中指定了段样式，那么 CICS 将创建链接 <atom:link rel="self" href="/myatomfeed/entries/23">。 filea.xml 显示了以下示例链接:

```
<atom:link rel="self" href="/atom/f/filea"/>
```

第 277 页的『<atom:entry> 元素』提供了有关 <atom:entry> 元素的所有子元素的参考信息。

11. 更改 <atom:entry> 元素的元数据子元素，以便为 Atom 条目指定适当的缺省元数据（例如，缺省标题）。 filea.xml 包含合适的子元素选择，如下所示:

```
<atom:title>FILEA item</atom:title>
<atom:author>
   <atom:name>CICS Development</atom:name>
   <atom:uri>http://www.example.com/cics</atom:uri>
</atom:author>
<atom:rights>Copyright (c) Example Corp 2009</atom:rights>
<atom:published>2009-02-28T12:00:00Z</atom:published>
```

第 277 页的『<atom:entry> 元素』列出了所有可能的子元素，并说明了 Atom 规范是否需要这些子元素。

12. 在 <atom:entry> 元素的 <atom:content> 子元素中:

    a. 使用 cics:resource 和 cics:type 属性，指定为订阅源中 Atom 条目提供内容的 CICS 资源或服务例程的名称（1 至 16 个字符）和类型，这与您在 <cics:resource> 元素中指定的内容完全相同。 filea.xml 包含指定 FILEA 样本的 <atom:content> 元素，如下所示:

    ```
    <atom:content cics:resource="FILEA" cics:type="file"/>
    ```

    b. 添加 type 属性，并使用该属性为订阅源中 Atom 条目的内容指定介质类型。 以下是一些可能的 type 属性:

    - type="text" 指定纯文本。
    - type="html" 指定 HTML。
    - type="xhtml" 指定 XHTML。
    - type="text/xml" 是通常用于人类可读 XML 文档的介质类型。

    如果您省略了该属性，CICS 会使用缺省的“application/xml”介质类型。 发出 Atom 文档时，CICS 会使用介质类型或缺省值标注内容。如果您正在使用服务例程，那么可以覆盖该介质类型或缺省值并指定备用介质类型。

第 277 页的『<atom:entry> 元素』提供了有关 <atom:content> 子元素的参考信息。

13.

示例

此处显示的 filea.xml 内容（不包括该样本中的注释）用于演示 XML 元素的嵌套:

```
<?xml version="1.0"?>
<cics:atomservice type="collection"
     xmlns:cics="http://www.ibm.com/xmlns/prod/cics/atom/atomservice"
     xmlns:atom="http://www.w3.org/2005/Atom">
   <cics:feed window="6">
     <cics:resource name="FILEA" type="file">
         <cics:bind root="DFH0CFIL"/>
     </cics:resource>
     <cics:authority name="example.com" date="2009-02-14"/>
     <cics:selector style="segment"/>
   </cics:feed>
   <atom:feed>
```

```
      <atom:link rel="self" href="/atom/f/filea/feed"/>
      <atom:title>Sample CICS file FILEA</atom:title>
      <atom:subtitle>A RESTFUL service for FILEA</atom:subtitle>
      <atom:icon>../../../web2.0/image/cics-icon.gif</atom:icon>
      <atom:logo>../../../web2.0/image/cics-logo.gif</atom:logo>
      <atom:rights>Copyright (c) Example Corp 2009</atom:rights>
      <atom:category term="Demonstration"/>
      <atom:author>
          <atom:name>CICS Development</atom:name>
          <atom:email>cics@example.com</atom:email>
          <atom:uri>http://www.example.com/cics</atom:uri>
      </atom:author>
      <atom:entry>
         <atom:link rel="self" href="/atom/f/filea"/>
         <atom:title>FILEA item</atom:title>
         <atom:author>
            <atom:name>CICS Development</atom:name>
            <atom:uri>http://www.example.com/cics</atom:uri>
         </atom:author>
         <atom:rights>Copyright (c) Example Corp 2009</atom:rights>
         <atom:published>2009-02-28T12:00:00Z</atom:published>
         <atom:content cics:resource="FILEA" cics:type="file"/>
      </atom:entry>
   </atom:feed>
</cics:atomservice>
```

下一步做什么

在设置 Atom 配置文件后，请遵循第 283 页的『为 Atom 订阅源创建 ATOMSERVICE 定义』中的指示信息来创建引用该文件的 ATOMSERVICE 资源定义。

<cics:atomservice> 元素

<cics:atomservice> 元素是 Atom 配置文件的根元素。

属性

type=*typevalue*

正在配置的 Atom 文档的类型。该属性是必需的。*typevalue* 必须与 ATOMSERVICE 资源定义（该定义引用此 Atom 配置文件）的 ATOMTYPE 属性中所指定的文档类型相匹配。

type="feed"

Atom 订阅源文档。

type="service"

Atom 服务文档。

type="collection"

Atom 集合文档。

type="category"

Atom 类别文档。

xmnls:cics="http://www.ibm.com/xmlns/prod/cics/atom/atomservice"

用于将该配置文件与 CICS Atom XML 名称空间绑定在一起的名称空间声明。请勿更改该名称空间声明。

xmnls:atom="http://www.w3.org/2005/Atom"

用于将该配置文件与 Atom XML 名称空间绑定在一起的名称空间声明。请勿更改该名称空间声明。

名称空间声明 **xmnls:app="http://www.w3.org/2007/app"** 用于将 Atom 文档绑定到“Atom 发布协议”的名称空间。但是，该名称空间声明不会在 Atom 订阅源或集合的 Atom 配置文件中出现，因为这些 Atom 配置文件不包含具有 app: 前缀的任何元素。当 CICS 在为集合提供的 Atom 文档中使用 <app:edited> 元素时，它会自动添加该名称空间声明。在 Atom 服务或类别文档的 Atom 配置文件中该名称空间声明是必需的。

包含:

『<cics:feed> 元素』

第 274 页的『<atom:feed> 元素』

示例

```
<cics:atomservice type="feed"
      xmlns:cics="http://www.ibm.com/xmlns/prod/cics/atom/atomservice"
      xmlns:atom="http://www.w3.org/2005/Atom">
</cics:atomservice>
```

<cics:feed> 元素

Atom 配置文件中的 <cics:feed> 元素包含其他元素，用于描述将作为订阅源发布的 CICS 资源。它还指定了窗口的条目数，这是 CICS 将在每个订阅源文档中返回的条目数，并且包含了一些元素，用于指定请求 URL 的类型和条目的 Atom 标识。

包含于:

第 268 页的『<cics:atomservice> 元素』

属性

window="*number***"|"8"**

CICS 将在 Atom 订阅源文档中所返回条目的缺省数目。该属性是可选的，缺省窗口为 8 个条目。客户机可以指定不同的窗口大小。仅当客户机使用针对整个 Atom 订阅源的 URL 发出请求时，窗口大小才适用；否则，将使用针对 Atom 条目部分列表的导航 URL。当客户机使用个别 Atom 条目的 URL 发出请求时，CICS 仅返回单个请求的 Atom 条目。

包含:

第 271 页的『<cics:selector> 元素』

第 270 页的『<cics:authority> 元素』

『<cics:resource> 元素』

示例

```
<cics:feed window="10">
</cics:feed>
```

<cics:resource> 元素

Atom 配置文件中的 <cics:resource> 元素用于指定将作为订阅源发布的 CICS 资源的名称和类型。

包含于:

『<cics:feed> 元素』

属性

name=*"cics-resource-name"*

为订阅源提供数据的 CICS 资源的名称。该属性是必需的。该名称是区分大小写的，因此大写通常是正确的。资源名称必须与引用该 Atom 配置文件的 ATOMSERVICE 资源定义的 RESOURCENAME 属性值相匹配。

type=*"cics-resource-type"*

CICS 资源的类型。该属性是必需的。以小写形式指定该资源类型。资源类型必须与引用该 Atom 配置文件的 ATOMSERVICE 资源定义的 RESOURCETYPE 属性值相匹配。

type="tsqueue"

一个临时存储队列。

type="file"

一个文件。

type="program"

一个程序（服务例程）。

包含:

<cics:bind> 元素

<cics:bind> 元素具有一个 **root**=*"root-element-name"* 属性，用于指定 XML 绑定中顶级数据结构的名称。

第 272 页的『<cics:fieldnames> 元素』

示例

```
<cics:resource name="feedq" type="tsqueue">
     <cics:bind root="SAMPBIND"/>
</cics:resource>
```

<cics:authority> 元素

Atom 配置文件中的 <cics:authority> 元素提供了权限名称及相关日期，供 CICS 创建标记 URI 以用作个别 Atom 条目的 Atom 标识时使用。

标记 URI 的其他元素包括一个“tag”前缀、由您在 Atom 配置文件的 <cics:resource> 元素中指定的资源类型和资源名称组成的特定内容，以及个别 Atom 条目的选择器值。第 226 页的『Atom 条目的 Atom 标识』说明了标记 URI 格式和使用 Atom 标识的需求。

如果您宁愿使用备选的 Atom 标识格式，那么在 Atom 条目原型中使用 <atom:id> 元素来指定 Atom 标识原型，并忽略 <cics:authority> 元素。如果您使用备用 Atom 标识格式，请确保 Atom 标识是唯一的且符合 RFC 4287 中 Atom 格式规范的要求。

标记 URI 在一个 CICS 区域中是唯一的，但不能保证它们在不同的 CICS 区域中也是唯一的。如果您希望根据同名同类型但位于不同 CICS 区域中的资源设置 Atom 订阅源，那么可以在 Atom 配置文件的 <cics:authority> 元素中为每个订阅源指定不同的权限名称或不同的日期。即使所有其他信息都是相同的，具有不同日期的标记 URI 也互不相同。

如果用户编写的服务例程只处理一个 Atom 订阅源，那么标记 URI 对于该例程提供的 Atom 条目是唯一的，但如果用户编写的服务例程提供多个订阅源，标记 URI 就不是唯

一的，因为服务例程的名称在标记 URI 中用作资源名称。如果用户编写的服务例程提供了多个订阅源，那么可以为 Atom 标识选择备用格式，或在 Atom 配置文件的 <cics:authority> 元素中为每个订阅源使用一个不同的权限名称或日期。

包含于:

第 269 页的『<cics:feed> 元素』

属性

name=*"authority-name"*

权限名称。要遵循 RFC 4151，权限名称必须是标准域名或者您或贵公司注册的电子邮件地址。您不得使用其他人拥有的域名或电子邮件地址。RFC 4151 建议您以小写字母指定权限名称，以不同大小写字母形式指定相同权限名称的标记 URI 各不相同。您应确保已征得公司同意使用某域名，公司内的其他人可能也会使用该域名来生成标记 URI。

date=*"YYYY-MM-DD"*

您或贵公司拥有该权限名称的日期。您可以使用从首次拥有该权限名称的时间一直到至今为止这一期间内的任何日期，但是，RFC 4151 建议不要使用首次拥有之日或随后的几天。您不得使用未来的日期。CICS 会进行检查，确保该日期是有效的过去日期或当前日期。

示例

```
<cics:authority name="example.com" date="2009-01-08"/>
```

<cics:selector> 元素

Atom 配置文件中的 <cics:selector> 元素标识某个 Atom 条目的选择器值的 URL 中的特征和位置。CICS 使用该信息在 HTTP 请求中找到并抽取特定 Atom 条目的选择器值，并在将某个 Atom 条目作为 Atom 订阅源文档或集合的一部分发出时生成正确格式的 URL。

包含于:

第 269 页的『<cics:feed> 元素』

属性

style=*"style-type"*

样式如下所示:

*"***segment***"*

<cics:selector style="segment"/> 表示个别 Atom 条目的标准 URL 样式，其中 Atom 条目的选取器值位于 URL 路径部分的最后一段。在该 URL 中，选择器值是“25”:

```
http://www.example.com/web20/sample_atom_feed/entry/25
```

"segment" 是缺省样式，因此，如果您需要该样式并且选取器值采用 CICS 假定该资源类型所用的格式，那么可以在 Atom 配置文件中省略 <cics:selector> 元素。

″**query**″

<cics:selector style=″query″/> 表示兼容使用 CA8K SupportPac 开发的应用程序的 URL 样式，其中 Atom 条目的选择器值位于查询字符串中。在该 URL 中，选择器值是“25”:

```
http://www.example.com/web20/sample_atom_feed/entry?s=25
```

name=*″query-parm-name″*

如果您选择 style=″query″，那么可以使用该属性来指定用于标识选择器值的查询字符串关键字的名称。如果忽略该名称属性，那么使用缺省关键字“s”。当您选择 style=″segment″ 时，请勿使用该属性。

format=*″format-name″*

该属性标识选择器值的格式。如果您的选择器值采用的格式与 CICS 假定该包含 Atom 条目的资源的类型所用格式相同，请勿指定该属性。仅当资源包含非标准的选择器值时才指定该属性。格式如下所示:

″**decimal**″

选择器值是十进制数。CICS 假定该格式用于临时存储器队列以及 RRDS 和 VRRDS 文件。如果您正在使用其他任何类型的资源，并且该资源将十进制数作为选择器值，请指定该设置。

″**hexadecimal**″

选择器值是二进制数。CICS 假定该格式用于 ESDS 和扩展的 ESDS 文件。如果您正在使用其他任何类型的资源，并且该资源将二进制数作为选择器值，请指定该设置。

对于其他任何类型的 VSAM 文件，CICS 假定选择器值的格式为字符串。您不能使用该属性显式地指定字符串格式。

ccsid=*″nnnn″*

该属性指定选择器值在传递到服务例程之前必须转换成的编码字符集标识（CCSID，即代码页）。如果指定 format=″decimal″ 或 format=″hexadecimal″，或 CICS 假定您的资源使用这两种格式之一，请勿指定该属性。仅当选择器值为字符串格式，并且该选择器值可能包含在请求 URL 中用百分号编码的非字母数字字符时才指定该属性。*″nnnn″* 是用百分号编码的 UTF-8 字符将转换成的 EBCDIC CCSID 的编号。如果 CICS 在选择器值中遇到用百分号编码的字符，但省略了 ccsid 属性，CICS 将使用在 LOCALCCSID 系统初始化参数中指定的值。 *CICS System Definition Guide* 中的 LOCALCCSID 系统初始化参数的描述介绍了 CICS 支持的 CCSID。

示例

```
<cics:selector style="query" name="sel" format="hexadecimal"/>
```

<cics:fieldnames> 元素

Atom 配置文件中的 <cics:fieldnames> 元素用于标识 CICS 资源的记录中的字段名称，它们为 Atom 条目文档提供元素据项或内容。元素的这些属性用于指定重要的字段名称。

包含于:

第 269 页的『<cics:resource> 元素』

属性

为支持使用 <cics:fieldnames> 元素，您必须为包含 Atom 条目数据的资源创建一个 XML 绑定。*CICS Application Programming Guide* 介绍了如何创建该绑定。在 <cics:fieldnames> 元素中，使用在创建 XML 绑定时由 CICS XML 助手生成的 XML 名称来指定字段名称，而不是在高级语言结构或描述资源结构的副本中声明的初始字段名称。

对于文件或临时存储器队列的 CICS 资源类型（CICS 直接从资源中将 Atom 条目的数据抽取到这些文件或队列），如果文件或临时存储器队列中的记录包含任何具有 Atom 元数据的字段（例如，标题或时间戳记），那么必须使用 <cics:fieldnames> 元素。如果记录包含任何元数据，那么请添加 <cics:fieldnames> 元素，并使用元素属性在提供 Atom 条目元数据的 CICS 资源的记录中指定字段。如果 CICS 资源中的记录仅包含条目内容而没有任何元数据，请省略 <cics:fieldnames> 元素。

在您使用称为服务例程的程序来提供 Atom 条目时，如果您使用 DFHATOMPARMS 容器中的资源处理参数将字段名称传递给该程序，请添加 <cics:fieldnames> 元素。使用元素的属性来命名资源中记录内的字段，程序为获取 Atom 条目的元数据和内容，需要访问该资源。您必须具有资源的 XML 绑定才能执行此操作。如果程序自身包含关于资源结构的信息，并且未使用资源处理参数，请省略 <cics:fieldnames> 元素。

<cics:fieldnames> 元素的所有属性都是可选的。如果您不指定特定属性（例如作者名称），那么 CICS 会通过配置文件中 <atom:entry> 元素中的相应元素提供元素据项，或者将其忽略。如果您忽略所有这些属性，那么还可以忽略 <cics:fieldnames> 元素。

atomid=*"fieldname"*
: 资源记录中字段的名称，包含 Atom 条目的唯一标识。

author=*"fieldname"*
: 资源记录中字段的名称，包含 Atom 条目主要作者的个人名称。您不能通过 CICS 资源记录中的这些字段提供其他作者或贡献者的详细信息。您可以在配置文件中的 <atom:feed> 元素中提供这些信息，但是它们会应用于所有条目。如果您使用 <cics:fieldnames> 元素提供来自 CICS 资源的作者详细信息，并且还在配置文件中指定了一个或多个 <atom:author> 实例，那么会先应用来自 CICS 资源的作者详细信息，然后再应用来自配置文件的作者详细信息。

authoruri=*"fieldname"*
: 资源记录中字段的名称，包含与 Atom 条目的主要作者关联的 URL，例如，博客站点或公司 Web 站点。

category=*"fieldname"*
: 资源记录中字段的名称，包含用于对条目进行分类的类别。

content=*"fieldname"*
: 资源记录中字段的名称，包含将要在 Atom 条目中发布的完整内容。该名称可以是记录中子结构的名称，在这种情况下，根据需要，会将整个子结构传播至发布的内容中，作为具有子元素的 XML 元素。如果忽略该属性，那么 CICS 会发布完整的资源记录作为条目的内容。

content_type=*"fieldname"*
: 资源记录中某个字段的名称，该字段包含 Atom 条目内容的介质类型。

edited=*"fieldname"*
: 资源记录中字段的名称，包含用于表明上一次编辑记录时间的时间戳记。从 ATOMSERVICE 资源定义的 BINDFILE 属性所指定 XML 绑定中的描述获取指定

字段的数据类型。如果已命名字段的字符串长度至少为 20，那么 CICS 会假设其包含 XML dateTime 格式的时间戳记，正如 RFC 3339 中所述；例如，“2008-05-20T11:39:50.325Z”。如果指定字段是长度为 8 的压缩十进制字段，并且在准备 XML 绑定时使用可选的 DATETIME=PACKED15 参数，那么 CICS 假定该字段包含 CICS ABSTIME 值。

email=*"fieldname"*
资源记录中字段的名称，包含 Atom 条目主要作者的电子邮件地址。

published=*"fieldname"*
资源记录中字段的名称，包含表明第一次发布记录的时间戳记或 ABSTIME 值。CICS 标识日期格式的方式与针对已编辑属性的方式相同。

title=*"fieldname"*
资源记录中字段的名称，包含 Atom 条目的标题。

summary=*"fieldname"*
资源记录中字段的名称，包含 Atom 条目的摘要。

updated=*"fieldname"*
资源记录中字段的名称，包含表明上次更新记录的时间戳记或 ABSTIME 值。CICS 标识日期格式的方式与针对已编辑属性的方式相同。

示例

以下字段名称是第 234 页的『创建可存储 Atom 条目的 CICS 资源』中显示的示例 COBOL 语言结构中字段名称的 XML 版本。

```
<cics:fieldnames title="title_field"
                 summary="summary_field"
                 atomid="atomid_field"
                 content="content_field"
                 author="author_name_field"
                 email="author_email_field"
                 authoruri="author_uri_field"
                 edited="edited_field"
                 updated="updated_field"
                 published="published_field"
                 category="category_field"/>
```

相关参考

第 240 页的『DFHATOMPARMS 容器』
DFHATOMPARMS 是一个指定了 DATATYPE（CHAR）的容器，它包含 CICS 在与服务例程（为 Atom 订阅源提供数据）进行通信时所用到的参数。

<atom:feed> 元素

Atom 配置文件中的 <atom:feed> 元素是 CICS 返回的 Atom 订阅源文档的原型。它为 Atom 订阅源提供元数据，并包含单个原型 <atom:entry> 元素。

包含于:

第 268 页的『<cics:atomservice> 元素』

子元素

<atom:feed> 元素的子元素是按照 RFC 4287 中的 Atom 格式规范进行定义。其中有些子元素是必需的，有些是可选的。

对于定义为“文本结构”的子元素（例如 <atom:title> 元素），RFC 4287 允许其具有纯文本、HTML 或 XHTML 内容。CICS 仅支持这些元素的纯文本内容，因此您不能使用“type”属性来指定备用内容类型。您必须提供不含子元素的纯文本内容。CICS 允许将 <atom:content> 元素中的 HTML、XHTML 和其他文本介质类型（例如，XML）作为 Atom 条目的内容，您在为 Atom 条目提供数据的 CICS 资源中指定该元素。

<atom:feed> 元素的子元素如下所示:

<atom:author>

Atom 订阅源的主要作者（可能是个人或组织）的个人详细信息。配置文件中可以拥有多个该元素。该数据由以下子元素提供:

<atom:name>

人员名称。如果您指定 <atom:author> 元素，那么该子元素是必需的，CICS 会检查您是否包含了该元素。

<atom:uri>

与人员关联的 URL，例如博客站点或公司的 Web 站点。该子元素是可选的。CICS 将确保您没有在 <atom:author> 元素中指定多个 <atom:uri> 元素。 CICS 不会尝试验证 URL 是否有效，因此您必须确保 URL 正确无误。

<atom:email>

人员的电子邮件地址。该子元素是可选的。CICS 将确保您没有在 <atom:author> 元素中指定多个 <atom:email> 元素。

除非 Atom 订阅源中的所有 Atom 条目都具有 <atom:author> 元素，否则 <atom:author> 元素是必需的。如果 Atom 条目的数据由临时存储器队列或文件提供，那么 CICS 将检查您是将 <atom:author> 元素指定为 <atom:feed> 元素或原型 <atom:entry> 元素的子元素，还是指定了 <cics:fieldnames> 元素的 author 属性。如果 Atom 条目的数据由程序提供，那么 CICS 将不执行该检查。根据 RFC 4287 的要求，如果您选择不在配置文件的任何位置指定 <atom:author> 元素，那么您必须确保资源中的所有 Atom 条目都包含该数据。

<atom:category term="">

对 Atom 订阅源进行分类的类别的名称。该元素是可选的。CICS 仅支持该元素的单个实例。 term 属性指定类别的名称。

<atom:contributor>

Atom 订阅源的辅助作者的个人详细信息。该元素是可选的。配置文件中可以拥有多个该元素。该数据由以下子元素提供:

<atom:name>

人员名称。当使用 <atom:contributor> 元素时，该子元素是必需的。

<atom:uri>

与人员关联的 URL，例如博客站点或公司的 Web 站点。该子元素是可选的。

<atom:email>

人员的电子邮件地址。该子元素是可选的。

<atom:entry>

在 Atom 配置文件中，需要有一个原型 <atom:entry> 元素。第 277 页的『<atom:entry> 元素』中描述了该元素。

<atom:generator>

生成 Atom 订阅源的代理程序的名称。请勿在 Atom 配置文件中指定该元素；在生成 Atom 订阅源文档时，CICS 会提供该元素。CICS 提供一个自身标识，作为 Atom 订阅源的生成器。

<atom:icon>

指向表示 Atom 订阅源的小图标的 URL。该元素是可选的。CICS 会进行检查，以确保您没有为 Atom 订阅源指定多个 <atom:icon> 元素，但不会检查该图像的宽高比，此比例应该是 1（宽）比 1（高）。

<atom:id>

Atom 订阅源的唯一标识。Atom 格式规范要求 Atom 订阅源要有一个 <atom:id> 元素。如果您在 Atom 配置文件中指定了 <cics:authority> 元素，以使 CICS 生成标记 URI 以作为 Atom 标识，那么可以省略 Atom 订阅源的 <atom:id> 元素，CICS 将为 Atom 订阅源生成与 Atom 条目的 Atom 标识格式相同的 Atom 标识，但是此 Atom 标识不含为 Atom 条目附加的选择器值或唯一标识。例如:

```
tag:example.com,2009-01-08:tsqueue:WB20TSQ
```

如果您喜欢使用备用 Atom 标识格式，或者您在原型 Atom 条目中使用 <atom:id> 元素来指定 Atom 标识格式并且希望复制该格式，那么可以为该 Atom 订阅源包含 <atom:id> 元素并为该 Atom 订阅源指定完整的 Atom 标识。请确保该 Atom 标识是唯一的且符合 RFC 4287 中 Atom 格式规范的要求。

<atom:link>

标识 Atom 订阅源文档并使 Web 客户机能够检索该文档的 URL。第 217 页的『来自 CICS 的 Atom 订阅源的 URL』说明了如何构造该 URL。

Atom 订阅源文档必须具有一个 <atom:link rel="self"> 元素以作为 <atom:feed> 元素的子元素。href 属性包含可供 Web 客户机用来检索 Atom 订阅源文档的 URL。CICS 不支持在 Atom 订阅源文档中使用其他类型的 <atom:link> 元素。

在 Atom 配置文件中，<atom:link rel="self"> 元素必须声明可供 Web 客户机用来检索 Atom 订阅源文档的完整路径，该路径的开始部分必须与您在 URIMAP 资源定义中为 Atom 订阅源或集合声明的部分路径匹配。例如，如果将 `/myatomfeed/*` 指定为 URIMAP 资源定义中的路径部分，那么可以在 Atom 配置文件中指定 <atom:link rel="self" href="/myatomfeed/feed.atom">。第 32 页的『CICS Web Support 的 URL』中列出的 URL 长度限制同样适用于 Atom 订阅源的 URL。

在 Atom 配置文件中，您可以省略 URL 的方案和主机部分，而仅指定路径部分。CICS 在向客户机返回 Atom 订阅源或 Atom 条目文档时，会对 URL 添加方案和主机部分，以符合 Atom 格式规范。

<atom:logo>

指向表示 Atom 订阅源的大徽标的 URL。该元素是可选的。CICS 会进行检查，以确保您没有为 Atom 订阅源指定多个 <atom:logo> 元素，但不会检查该图像的宽高比，此比例应该是 2（宽）比 1（高）。

<atom:rights>

包含已声明的知识产权（例如，版权）的文本字符串。对于该元素，CICS 只支持纯文本。该元素是可选的。

<atom:subtitle>

Atom 订阅源的子标题。CICS 只支持纯文本的子标题。该元素是可选的。CICS 将进行检查，以确保您没有为 Atom 订阅源指定多个 <atom:subtitle> 元素。

<atom:title>

Atom 订阅源的标题。CICS 只支持纯文本的标题。<atom:title> 元素是必需的。CICS 将进行检查，以确保您为 Atom 订阅源指定了一个 <atom:title> 元素。

<atom:updated>

上次更新 Atom 订阅源的时间。请勿在 Atom 配置文件中指定该元素；在生成 Atom 订阅源文档时，CICS 会提供该元素。CICS 提供该时间戳记以作为构成 Atom 订阅源的内部 <atom:entry> 元素的所有包含的 <atom:updated> 元素的最新时间。如果包含 Atom 条目的 CICS 资源未提供该数据，那么 CICS 将当前日期和时间作为缺省值。

包含:

『<atom:entry> 元素』

示例

```
<atom:feed>
    <atom:title>CICS Atom feed</atom:title>
    <atom:subtitle>My first Atom feed from CICS</atom:subtitle>
    <atom:link rel="self" href="/web20/sample_atom_feed" />
    <atom:rights>Copyright (c) 2009, Joe Bloggs</atom:rights>
    <atom:author>
         <atom:name>Joe Bloggs</atom:name>
         <atom:uri>http://www.ibm.com/JBloggs/</atom:uri>
         <atom:email>JBloggs@uk.ibm.com</atom:email>
     </atom:author>
     <atom:contributor>
         <atom:name>John Doe</atom:name>
     </atom:contributor>
</atom:feed>
```

<atom:entry> 元素

Atom 配置文件中的 <atom:feed> 元素包含单个原型 <atom:entry> 元素。CICS 通过用 <cics:resource> 元素中所描述 CICS 资源提供的数据来填充该元素的子元素，从而生成一个 Atom 条目文档，并针对客户机请求的所有 Atom 条目重复此过程。

包含于:

第 274 页的『<atom:feed> 元素』

子元素

<atom:entry> 元素的子元素按照 RFC 4287 中的 Atom 格式规范进行定义，但 CICS 不支持可选的 <atom:source> 元素。其中有些子元素是 Atom 条目所必需的，有些则是可选的。

<atom:entry> 元素的大多数子元素都包含缺省元数据，用于为 CICS 资源或服务例程未提供的个别 Atom 条目提供任何数据项。

- 如果您的 CICS 资源或服务例程始终为这些子元素之一提供数据，那么通常可以在配置文件的 <atom:entry> 元素中省略该元素。CICS 所需的 <atom:title> 元素是例外，尽管当 Atom 条目都具有一个标题并且没有适合的缺省标题时，您可以使用空元素。

- 如果 Atom 条目保存在某个资源中，而这些资源的记录缺少用于保存部分或全部可能元数据项的字段，那么您可以通过在 Atom 配置文件中指定来提供这些元数据项。在 <atom:entry> 元素中提供元数据时，只要 CICS 资源或服务例程未提供对应的数据项，那么 CICS 就将使用该元数据。
- 如果不希望为这些项提供缺省值，那么可以省略任何可选的元数据项。

<atom:entry> 元素的某些子元素对应于 <cics:fieldnames> 元素的属性。如果子元素在 <cics:fieldnames> 元素中有对应的属性，那么 CICS 可以从资源提供数据项，或者由服务例程提供数据项（如果存在）。如果子元素在 <cics:fieldnames> 元素中无对应的属性，那么只能在 Atom 配置文件中指定。

对于定义为“文本结构”的子元素（例如 <atom:title> 元素），RFC 4287 允许其具有纯文本、HTML 或 XHTML 内容。CICS 仅支持这些元素的纯文本内容，因此您不能使用“type”属性来指定备用内容类型。您必须提供不含子元素的纯文本内容。CICS 允许将 <atom:content> 元素中的 HTML、XHTML 和其他文本介质类型（例如，XML）作为 Atom 条目的内容，您在为 Atom 条目提供数据的 CICS 资源中指定该元素。

<atom:entry> 元素的子元素如下所示:

<app:edited>

上次编辑 Atom 条目的时间。该元素仅适用于集合中包含的 Atom 条目，在这种情况中，该元素是必需的（作为一个“应当执行”的需求）。您不能在配置文件的原型 Atom 条目中指定 <app:edited> 元素。该元素对应于 <cics:fieldnames> 元素的 edited 属性。如果 CICS 资源或服务例程未提供该数据，那么 CICS 会提供当前日期和时间作为缺省值。虽然 CICS 支持使用 <app:edited> 元素，但出于性能考虑，CICS 在以列表的形式向客户机返回集合中的 Atom 条目时，不会按该元素自动对这些条目进行排序。要了解有关对集合中 Atom 条目排序的更多信息，请参阅第 224 页的『Atom 条目的顺序』。

<atom:author>

Atom 条目的主要作者（可能是个人或组织）的个人详细信息。配置文件中可以拥有多个该元素。该数据由 <atom:feed> 元素的 <atom:name>、<atom:uri> 和 <atom:email> 子元素提供。如果 Atom 订阅源拥有一个 <atom:author> 元素，那么该元素是可选的，因为 Atom 订阅源的作者适用于不包含作者的 Atom 条目。该元素对应于 <cics:fieldnames> 元素的 author、email 和 authoruri 属性，因此如果您的 CICS 资源始终提供您所需的数据，那么您可以省略该元素。如果您使用 <cics:fieldnames> 元素提供来自 CICS 资源的作者详细信息，并且还在配置文件中指定了一个或多个 <atom:author> 实例，那么会先应用来自 CICS 资源的作者详细信息，然后再应用来自配置文件的作者详细信息。

<atom:category term="">

对 Atom 条目进行分类的类别的名称。该元素是可选的。CICS 仅支持该元素的单个实例。 term 属性指定类别的名称。该元素对应于 <cics:fieldnames> 元素的 category 属性，因此如果您的 CICS 资源始终提供合适的数据，那么可以省略该元素。

<atom:content type="" cics:resource="" cics:type=""/>

Atom 条目的内容。该元素是必需的。

在配置文件中指定的 <atom:content> 元素不包含任何数据，因为 CICS 在发布订阅源文档时通过该资源来提供数据。您必须确保您的资源为每个 Atom 条目提供内容。CICS 不支持提供不含数据的 Atom 条目，例如，使用“src”属性引用远程内容的 Atom 条目。

type=″ ″

type 属性指定 CICS 期望用于 Atom 条目的内容类型，这可以是“text”、“html”、“xhtml”或 IANA 介质类型。如果该属性不存在，CICS 会使用缺省的“application/xml”介质类型。在 RFC 4287 中，使用介质类型“text”替代 IANA 介质类型“text/plain”来表示纯文本，使用“html”替代“text/html”，使用“xhtml”替代“application/xhtml+xml”。如果内容采用任何其他格式，请指定在因特网上通常用于这种格式的 IANA 介质类型。http://www.iana.org/assignments/media-types/ 中列出了可用的介质类型。请注意，CICS 不支持非文本介质类型。

如果 CICS 直接从资源提供 Atom 条目，介质类型或缺省类型必须适合于该资源中的数据，因为在发布 Atom 文档时，CICS 会用该介质类型标注内容。如果您使用服务例程为 Atom 条目提供内容，那么会将该介质类型或缺省类型提供给服务例程。服务例程可以覆盖该类型并指定备用介质类型，或允许 CICS 用 Atom 配置文件中的介质类型标注内容。

“cics:resource=”和“cics:type=″ ″”

属性 cics:resource 和 cics:type 声明为 Atom 订阅源提供数据的 CICS 资源的名称和类型。这些属性的值必须与为 <cics:resource> 元素指定的 name 和 type 属性的值相匹配。有关这些属性值的描述，请参阅第 269 页的『<cics:resource> 元素』。

<atom:content> 元素对应于 <cics:fieldnames> 元素的 content 和 content_type 属性。

<atom:contributor>

Atom 条目的辅助作者的个人详细信息。该元素是可选的。该数据由 <atom:feed> 元素的 <atom:name>、<atom:uri> 和 <atom:email> 子元素提供。您不能在个别资源中为 <atom:contributor> 元素指定数据，因此 <cics:fieldnames> 元素没有对应的属性，您提供的数据将应用于所有 Atom 条目。由于该数据应用于所有 Atom 条目，您可能希望在 <atom:feed> 元素中指定内容提供者。

<atom:id>

Atom 条目的唯一标识。Atom 格式规范要求每个 Atom 条目都要有一个 <atom:id> 元素。第 226 页的『Atom 条目的 Atom 标识』说明了使用 Atom 标识的需求。

CICS 可以生成标记 URI，将其用作 Atom 条目的唯一 Atom 标识。标记 URI 包含您在 Atom 配置文件的 <cics:authority> 元素中指定的属性、在 <cics:resource> 元素中指定的资源类型和资源名称，以及单个 Atom 条目的选择器值。如果这种格式满足您的需要，请在原型 Atom 条目中省略 <atom:id> 元素。

您可以使用原型 Atom 条目的 <atom:id> 元素来代替由 CICS 生成的 URI 格式，以指定备用格式的原型 Atom 标识。CICS 将附加选择器值或合适的唯一标识来生成唯一 Atom 标识。如果您使用备用 Atom 标识格式，请确保 Atom 标识是唯一的且符合 RFC 4287 中 Atom 格式规范的要求。

<atom:link>

标识 Atom 条目并支持 Web 客户机检索该条目的标准 URL。第 217 页的『来自 CICS 的 Atom 订阅源的 URL』说明了如何构造该 URL。

对于 CICS，您必须具有一个 <atom:link rel="self"> 元素以作为 <atom:entry> 元素的子元素。对于集合中的 Atom 条目，Atom 格式规范要求使用“edit”链接关系而非“self”，CICS 在发送集合中的 Atom 条目时会自动提供该链接关系。您必须在 Atom 配置文件中指定 <atom:link rel="self">，而不论 Atom 条目位于 Atom 订阅源还是在集合中。

在 Atom 配置文件的 <atom:link rel="self"> 元素中将一个 URL 指定为标准路径，您可以扩展该 URL 以应用于任何 Atom 条目文档。该路径的开始部分必须与您在 URIMAP 资源定义中为 Atom 订阅源或集合的声明的部分路径匹配。该标准路径的其余部分必须与您在 Atom 订阅源的 <aatom:link rel="self"> 元素中指定的完整路径不同。例如，如果您将 `/myatomfeed/*` 指定为 URIMAP 资源定义中的路径部分，并将 <atom:link rel="self" href="/myatomfeed/feed.atom"> 指定为 Atom 配置文件中整个 Atom 订阅源的链接，那么可以将 <atom:link rel="self" href="/myatomfeed/entries/"> 指定为 Atom 条目的标准路径。第 32 页的『CICS Web Support 的 URL』中列出的 URL 长度限制同样适用于 Atom 订阅源的 URL。

当 CICS 向客户机返回 Atom 条目文档时，它会将 Atom 条目的选择器值附加到该路径，以创建完整的链接。第 223 页的『Atom 条目的选择器值』说明了什么是选择器值。您可以在 Atom 配置文件中使用 <cics:selector> 元素指定将选择器值附加到该路径的方式。在选择缺省“segment”样式或省略元素时，CICS 会创建诸如 <atom:link rel="self" href="/myatomfeed/entries/23"> 的链接。另一种“query”样式将生成与针对 CA8K SupportPac 开发的应用程序兼容的格式。如果需要以十六进制指定选择器值，那么还可以使用 <cics:selector> 元素。

在 Atom 配置文件中，您可以省略 URL 的方案和主机部分，而仅指定路径部分。CICS 在向客户机返回 Atom 订阅源或 Atom 条目文档时，会对 URL 添加方案和主机部分，以符合 Atom 格式规范。

<atom:published>

首次创建或发布 Atom 条目的时间。该元素是可选的。它对应于 <cics:fieldnames> 元素的 published 属性。如果 CICS 资源不提供该数据，那么 CICS 将提供当前的日期和时间作为缺省值。您可以按 RFC 3339 所述，利用 XML dateTime 格式的备用缺省时间戳记在 Atom 配置文件中指定 <atom:published> 元素。

<atom:rights>

包含已声明的知识产权（例如，版权）的文本字符串。对于该元素，CICS 只支持纯文本。该元素是可选的，如果未提供该元素，那么将应用 Atom 订阅源的 <atom:rights> 元素。<cics:fieldnames> 元素没有对应的属性，因此您提供的数据将应用于所有 Atom 条目。

<atom:summary>

Atom 条目内容的简短描述。CICS 只支持纯文本的摘要。该元素对应于 <cics:fieldnames> 元素的 summary 属性。如果 Atom 条目的内容不是文本、HTML、XHTML 或 XML，那么该元素是必需的，因此如果您计划提供这些类别以外的任何内容，并且 CICS 资源并不始终提供摘要，应当使用该元素提供缺省摘要。如果 CICS 资源不提供摘要，并且您希望针对所有 Atom 条目显示

相同的摘要，那么也可以使用该元素。否则，您可以省略该元素。CICS 会进行检查，以确保您没有为 Atom 条目指定多个 <atom:summary> 元素。

<atom:title>

Atom 条目的标题。<atom:title> 元素是必需的。 CICS 只支持纯文本的标题。该元素对应于 <cics:fieldnames> 元素的 title 属性。您必须在原型 Atom 条目中包含 <atom:title> 元素，即使 CICS 资源始终为 Atom 条目提供标题，也是如此。使用可应用于任何 Atom 条目的合适的缺省标题，或者，如果 Atom 条目都具有一个标题并且没有适合的缺省标题时，您可以使用空元素。

<atom:updated>

上次对 Atom 条目进行重大更新的时间。该元素是 Atom 规范所必需的，但不能在 Atom 配置文件的原型 Atom 条目中指定该元素。该元素对应于 <cics:fieldnames> 元素的 updated 属性。如果 CICS 资源不提供该数据，那么 CICS 将提供当前的日期和时间作为缺省值。

示例

```
<atom:entry>
    <atom:title>An entry from my feed</atom:title>
    <atom:summary>DEFAULT --- This is the default summary</atom:summary>
    <atom:link rel="self" href="/web20/sample_atom_feed/entry" />
    <atom:author>
          <atom:name>Joe Bloggs</atom:name>
          <atom:uri>http://www.hursley.ibm.com/JBloggs/</atom:uri>
          <atom:email>JBloggs@uk.ibm.com</atom:email>
    </atom:author>
    <atom:contributor>
          <atom:name>John Doe</atom:name>
    </atom:contributor>
    <atom:category term="Comments" />
    <atom:published>2008-12-02T15:41:00</atom:published>
    <atom:content type="text" cics:resource="WB20TSQ" cics:type="tsqueue" />
 </atom:entry>
```

CICS 的 Atom 元素引用

这些表提供了有关 Atom 订阅源文档中使用的元素、ATOM 条目元素中使用的元素、<cics:fieldnames> 元素的属性以及 CICS 可传递至服务例程以进行资源处理的参数之间关系的参考信息。

在 CICS 中创建 Atom 订阅源文档时，您可以指定按 RFC 4287 中 Atom 格式规范定义的元素。其中部分元素用作 <atom:feed> 元素的子元素，为整个 Atom 订阅源提供元数据，例如，订阅源的标题。部分元素用作 <atom:entry> 元素的子元素，为单个条目提供元数据或内容。大部分元素是同时针对订阅源和单个条目指定的；例如，每个条目都有一个由 <atom:id> 元素指定的唯一标识，并且订阅源也有一个唯一标识。有关每个元素的完整描述，请阅读 RFC 4287。

在 CICS 的 Atom 订阅源文档中，CICS 使用单个原型 <atom:entry> 元素来生成单个条目。如果在该元素中指定子元素，那么缺省情况下，此处的元数据将应用于所有条目。但是，如果您正用于为 Atom 订阅源提供内容的 CICS 资源包含合适的元数据，那么可以使用 <cics:fieldnames> 元素的属性来通知 CICS 这些子元素的数据是否在资源记录中出现以及出现的位置。例如，您可以在资源记录中指定一个字段，用于为包含在该记录中的条目提供标题。如果资源记录不包含某些元数据（例如，作者姓名），那么您可以省略 <cics:fieldnames> 元素的该属性。CICS 要么在配置文件的原型

<atom:entry> 元素中的相应元素中提供该元数据项，要么将其省略。如果资源记录不包含任何合适的元数据，那么可以完全省略 <cics:fieldnames> 元素，并且 CICS 会将整个资源记录作为条目内容发布。

CICS 传递到服务例程的部分参数对应于 <cics:fieldnames> 元素的属性。如果您希望编写一个用于从 Atom 配置文件中获取有关资源结构的信息的服务例程，而不是将该信息直接编码到服务例程中，那么可以使用这些资源处理参数。通过该方法，您可以创建能够处理多个资源的类属服务例程。

表 13. <atom:feed> 和 <atom:entry> 元素的子元素

元素	含义	对于订阅源	对于条目	**<cics:fieldnames> 属性（对于条目）**	**服务例程参数（对于条目）**
<app:edited>	上一次编辑条目的时间	未使用	必需（如果在集合中），由 CICS 生成	edited	ATMP_ EDITED
<atom:author>	主要作者的详细信息	必需（只要不是所有条目都包含该元素）	可选（如果订阅源包含该元素）	不适用（数据位于子元素中）	不适用（数据位于子元素中）
<atom:category>	对订阅源或条目进行分类的类别	可选	可选	category	ATMP_ CATEGORY_ FLD
<atom:content type=" ">	条目的内容	未使用	必需	content, content_type	ATMP_ CONTENT_ FLD 和 ATMP_ CONTENT_ TYPE_FLD
<atom:contributor>	辅助作者的详细信息	可选	可选	不适用（数据位于子元素中）	不适用（数据位于子元素中）
<atom:email>	作者或参与者的电子邮件地址	可选	可选	email（仅限作者，不支持参与者）	ATMP_ EMAIL_ FLD
<atom:generator>	生成订阅源的代理程序	由 CICS 生成	未使用	不适用（不用于条目）	不适用（不用于条目）
<atom:icon>	表示订阅源的图标	可选	未使用	不适用（不用于条目）	不适用（不用于条目）
<atom:id>	订阅源或条目的唯一标识	必需，指定 <cics: authority> 元素时，由 CICS 生成	必需，指定 <cics: authority> 元素时，由 CICS 生成	atomid	ATMP_ ID_ FLD
<atom:link rel="self">	用于检索订阅源或条目文档的 URL	必需	必需（对于 CICS）	不适用（未存储在资源中）	不适用（未存储在资源中）
<atom:link rel="edit">	用于编辑集合中条目的 URL（成员 URI）	CICS 为集合生成	由 CICS 生成（条目在集合中时）	不适用（未存储在资源中）	不适用（未存储在资源中）
<atom:logo>	订阅源的徽标	可选	未使用	不适用（不用于条目）	不适用（不用于条目）

表 13. <atom:feed> 和 <atom:entry> 元素的子元素 (续)

元素	含义	对于订阅源	对于条目	<cics:fieldnames> 属性（对于条目）	服务例程参数（对于条目）
<atom:name>	作者或参与者的姓名	必需（在作者或参与者元素中）	必需（在作者或参与者元素中）	author（仅限作者，不支持参与者）	ATMP_ AUTHOR_ FLD
<atom:published>	第一次创建或发布条目的时间	未使用	可选	published	ATMP_ PUBLISHED_ FLD
<atom:rights>	订阅源的知识产权	可选	可选	在资源中不受支持	在资源中不受支持
<atom:source>	通过使用其他订阅源中的条目而获得的元数据	未使用	可选，但 CICS 不支持	不支持	不支持
<atom:subtitle>	订阅源的子标题	可选	未使用	不适用（不用于条目）	不适用（不用于条目）
<atom:summary>	条目内容的简要描述	未使用	必需（如果内容为非文本或 XML）	summary	ATMP_ SUMMARY_ FLD
<atom:title>	订阅源的标题	必需	必需	title	ATMP_ TITLE_ FLD
<atom:updated>	上一次更新订阅源的时间	必需，由 CICS 生成	必需，由 CICS 生成	updated	ATMP_ UPDATED_ FLD
<atom:uri>	作者或参与者 Web 站点的 URL	可选	可选	authoruri（仅限作者，不支持参与者）	ATMP_ AUTHORURI_ FLD

为 Atom 订阅源创建 ATOMSERVICE 定义

创建 ATOMSERVICE 定义以标识定义订阅源文档的 Atom 配置文件、为订阅源提供数据的 CICS 资源以及描述资源布局的 XML 绑定。

关于此任务

CICS 资源组 DFH$WEB2 包含一些针对 Atom 集合的样本 ATOMSERVICE 资源定义: DFH$W2F1、DFH$W2P1 和 DFH$W2Q1。 DFH$W2Q1 用于交付 CICS TS for z/OS V4.1 信息中心（可在 `https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp` 站点找到）的 Web 2.0 方案“创建 Atom 订阅源来使用员工信息”中描述的样本 Atom 集合。 DFH$W2P1 用于运行针对 Atom 订阅源的 DFH0W2F1 COBOL 样本服务例程。DFH$W2F1 用于提供 FILEA 样本，以作为直接来自 CICS 的 Atom 订阅源。

CICS Resource Definition Guide 提供有关不同资源定义方法的信息以及 ATOMSERVICE 资源定义属性的完整参考信息。

1. 使用 *CICS Resource Definition Guide* 中列出的一种方法，使 ATOMSERVICE 资源定义以在 URIMAP 资源定义中为 Atom 文档指定的名称及选择的组开头。
2. 使用 STATUS 属性来指定应在启用还是禁用状态下安装 ATOMSERVICE 资源定义。

3. 指定 FEED 的 ATOMTYPE 属性。
4. 指定 CONFIGFILE 属性作为针对 Atom 文档而创建的 Atom 配置文件的名称和路径。
5. 使用 RESOURCETYPE 属性，指定为 Atom 条目提供数据的 CICS 资源是服务例程（PROGRAM）、CICS 直接处理的文件（FILE），还是 CICS 直接处理的临时存储器队列（TSQUEUE）。
6. 将 RESOURCENAME 属性指定为 CICS 资源的名称（1 到 16 个字符）。
7. 将 BINDFILE 属性指定为使用 CICS XML 助手创建的 XML 绑定的名称和路径。XML 绑定文件是 FILE 或 TSQUEUE 资源所必需的。 如果正在使用服务例程（PROGRAM）并且针对保存 Atom 条目数据的资源创建了一个 XML 绑定，那么请为该资源指定 XML 绑定文件。如果正在使用包含关于其自身的资源结构信息的服务例程，并且没有 XML 绑定，那么请勿指定该属性。
8. 安装 ATOMSERVICE 资源定义和在第 262 页的『为 Atom 文档创建 URIMAP 资源定义』中创建的相应 URIMAP 资源定义。 如果为第 261 页的『为 Atom 订阅源创建别名事务』中的别名事务创建了 TRANSACTION 资源定义，那么也要安装该资源。

结果

在安装了 ATOMSERVICE 资源定义及其支持资源后，您就拥有一个 Atom 订阅源，Web 客户机可访问该资源以获取 Atom 格式的条目列表。Web 客户机必须自行处理和显示数据。

许多免费或收费的 Web 客户机应用程序都可以请求、接收和显示 Atom 订阅源，这些应用程序包括专用的订阅源阅读器以及提供更多功能的类似应用程序，例如，用于创建 mashup 的应用程序。检查该应用程序是否描述为支持 Atom 格式。您还可以编写自己的 Web 客户机应用程序来发出针对 Atom 订阅源数据的 GET 请求。要了解如何编写针对 Atom 订阅源的 HTTP GET 请求的指示信息，请参阅第 298 页的『向 Atom 订阅源或集合发出 GET 请求』。

下一步做什么

您可以采取进一步措施以将 Atom 订阅源设置为集合，这样您或其他人就可以通过 Web 客户机（支持针对 Atom 订阅源的 HTTP POST、PUT 和 DELETE 请求，如 Atom 发布协议中所述）来管理和编辑订阅源中的条目。

第 24 章 将 Atom 订阅源加入集合

要通过现有 Atom 订阅源创建可编辑的集合，请设置新的 URIMAP 和 ATOMSERVICE 资源定义，以及新的 Atom 配置文件。新定义使用了初始 Atom 订阅源的大部分设置，只进行了少量更改。

开始之前

如果 Atom 订阅源包含从资源抽取数据并提供给 CICS 的服务例程，那么在将数据用作集合之前，请先阅读第 295 页的第 25 章，『如何编辑 Atom 集合』，并修改服务例程，以便按照第 307 页的『如何在服务例程中处理 Atom 集合编辑请求』中的指示信息对集合中条目的 POST、PUT、DELETE 和 GET 请求采取相应操作。

关于此任务

通过获取 Atom 条目列表并显示这些列表，客户机可以像使用普通 Atom 订阅源一样使用集合。因此，为了满足集合的要求，您可以更改 Atom 订阅源的 ATOMSERVICE 定义和配置文件，并让 CICS 将该集合（代替初始 Atom 订阅源）交付给所有用户。但是，建议您最好创建单独的 CICS 资源定义并使用单独的 URL 使 Atom 条目既可用作集合，又可继续单独用作订阅源。将相同的数据同时设置为订阅源和集合，确实会涉及额外的工作，但是这样做具有一些重要的优点:

- 您可以将适当的安全措施应用于可编辑的集合，并确保可自由地使用只读订阅源。
- 通过向大多数用户交付更高性能的订阅源文档，可以实现更短的响应时间。交付集合中的条目比交付订阅源中的条目需要更多的处理时间，因为 CICS 需要为集合提供额外的导航。

要从 Atom 订阅源创建可编辑的集合:

1. 为集合规划适当的安全措施，以便您可以仅允许获得认证的 Web 客户机来编辑集合中的条目。 有关安全性的更多信息，请参阅第 317 页的第 26 章，『Atom 订阅源的安全性』。
2. 设置 TCPIPSERVICE 资源定义，以指定供 Web 客户机用于请求集合的端口所应用的安全措施。 第 83 页的『为 CICS Web Support 创建 TCPIPSERVICE 资源定义』说明了如何执行该操作。
3. 遵循第 262 页的『为 Atom 文档创建 URIMAP 资源定义』中的过程，为集合选择一个适合的 URL（不同于 Atom 订阅源所用的 URL），并为集合的 URL 创建新的 URIMAP 定义。 除了与 Atom 订阅源的 URL 不同以外，您选择的 URL 还必须与使用相同主机名提供的其他 Atom 订阅源和集合的 URL 不同。
4. 完成第 286 页的『为集合创建 ATOMSERVICE 定义和 Atom 配置文件』中的步骤，以根据 Atom 订阅源的现有文件来设置新的 ATOMSERVICE 定义和 Atom 配置文件。 如果您正在使用资源和命令安全性来保护集合，请确保 Web 客户机的用户标识有权访问 ATOMSERVICE 定义及其引用的资源，包括服务例程使用的任何 CICS 资源和命令。
5. 按照第 287 页的『创建 Atom 服务文档』中的指示信息，创建包含集合的 Atom 服务文档。 按照第 290 页的『创建 Atom 类别文档』中的信息，您还可以创建 Atom 类别文档来指定集合的类别。

6. 按照第 292 页的『交付 Atom 服务或类别文档作为 Atom 配置文件』或第 293 页的『交付 Atom 服务或类别文档作为静态响应』中的指示信息，设置 CICS 资源定义来交付 Atom 服务和类别文档。

结果

在完成这些任务后，Web 客户机可以对集合执行添加、更新和删除条目操作。Web 客户机可以通过获取服务文档来找到集合的 URL。

下一步做什么

第 295 页的第 25 章，『如何编辑 Atom 集合』说明了如何通过 Web 客户机发出 HTTP GET、POST、PUT 和 DELETE 请求来编辑集合，以及服务例程需要如何处理编辑请求。

为集合创建 ATOMSERVICE 定义和 Atom 配置文件

通过针对来自相同数据的 Atom 订阅源复制同等文件并稍作更改，为集合创建 ATOMSERVICE 定义和 Atom 配置文件。

关于此任务

CICS Resource Definition Guide 提供有关资源定义的不同方法的信息以及 ATOMSERVICE 资源定义属性的完整参考信息。

1. 复制用于 Atom 订阅源的 Atom 配置文件，并且使用所选的任何适当名称重新命名该文件。 对该文件进行以下更改:
   a. 将根元素从 `<cics:atomservice type="feed">` 更改为 `<cics:atomservice type="collection">`。
   b. 将 <atom:feed> 元素的子元素 <atom:title> 更改为适当的集合标题。 不必更改 <atom:feed> 元素的子元素 <atom:title>。
   c. 将 <atom:feed> 元素的子元素 <atom:id> 更改为适当的唯一集合标识。
   d. 更改 <atom:feed> 元素的子元素 <atom:link rel="self" href=" "> 的 href 属性，以指定 Web 客户机可用于检索集合的完整路径。路径的开头必须与集合的 URIMAP 资源定义中规定的部分路径相匹配。 例如，如果在 URIMAP 资源定义中指定 `/myatomcoll/*` 为路径部分，那么可以在 Atom 配置文件中指定 <atom:link rel="self" href="/myatomcoll/collection.atom">。 您可以省略 URL 的方案和主机部分，而仅指定路径部分。
   e. 在原型 <atom:entry> 元素的 <atom:link rel="self" href=" "> 子元素中，更改 href 属性，从而使这个标准路径的开始部分与在集合的 URIMAP 资源定义中声明的部分路径匹配。 同时更改路径的其余部分对用户而言也非常有帮助（但不是必需的），这样该路径就与来自相应数据的普通 Atom 订阅源的路径有所区别。整个路径必须和在 <atom:link rel="self" href=" "> 元素中为集合指定的完整路径有所区别。 CICS 会在路径末尾附加由服务例程提供的选择器值或 Atom 条目适当的唯一标识，从而为每个 Atom 条目创建完整的链接。

      例如，如果在 URIMAP 资源定义中指定 `/myatomcoll/*` 为路径部分，且在 Atom 配置文件中指定 <atom:link rel="self" href="/myatomcoll/collection.atom"> 为整个集合的链接，那么可以指定 <atom:link rel="self" href="/myatomcoll/edit/"> 为集合中条目的标准路径。 当 CICS 发出 Atom 条目文档时，CICS 会根据“Atom 发

布协议”的要求将 ″self″ 属性更改为 ″edit″。请勿自行在 Atom 配置文件中更改该属性。在该示例中，CICS 发布链接，如 <atom:link rel=″edit″ href=″/myatomcoll/edit/23″>。

2. 复制该 Atom 订阅源的 ATOMSERVICE 资源定义，并使用在集合的 URIMAP 资源定义中所指定的名称重新命名该定义。 对该定义进行以下更改:
   a. 将 ATOMTYPE 属性从 FEED 更改为 COLLECTION。
   b. 将 CONFIGFILE 属性更改为刚为集合创建的 Atom 配置文件的名称。
3. 安装为该集合创建的 ATOMSERVICE 资源定义和相应的 URIMAP 资源定义。 您还可以按照第 261 页的『为 Atom 订阅源创建别名事务』中的指示信息为集合的别名事务创建 TRANSACTION 资源定义，以及安装该资源定义。

创建 Atom 服务文档

创建 Atom 服务文档，以向客户机通知服务器中可用的集合。Atom 服务文档仅列出要作为可编辑集合提供的 Atom 订阅源；其不包括不可编辑的普通 Atom 订阅源。

关于此任务

通常，针对一个 CICS 区域中可用的集合，只会创建一个 Atom 服务文档。Atom 服务文档存储在 z/OS UNIX 系统服务中。Atom 服务文档是 XML 文档，文件的扩展名为 .xml。您可以使用任何 XML 编辑器或文本编辑器创建该文件。

您可以选择将 Atom 服务文档创建为 Atom 配置文件（CICS 通过 ATOMSERVICE 资源定义来管理该配置文件），或将其创建为由 CICS Web Support 静态内容交付提供的文件。将 Atom 服务文档定义为 Atom 配置文件时需要一个额外的资源定义，但是这意味着该文档与 Atom 订阅源的 CICS 支持进行了集成。

如果您选择将 Atom 服务文档定义为 Atom 配置文件，那么应以根元素 <cics:atomservice type=″service″> 作为该文档的开头。除该元素外，CICS 集合的 Atom 服务文档不包含任何特定于 CICS 环境的元素。其使用由 RFC 5023 中的 Atom 发布协议规范定义的标准元素。因此 CICS 文档不包含有关这些元素的其他参考信息。如果要获取有关 Atom 服务文档中这些元素的其他参考信息，请参阅 RFC 5023: *The Atom Publishing Protocol*。

CICS 不验证 Atom 服务文档的内容。如果正确执行此处列出的步骤，那么您可以生成一个符合 Atom 发布协议规范的有效 Atom 服务文档。如果发现客户机在读取您的服务文档时遇到任何问题，或者没有以您期望的方式解释该服务文档，请重新按照这些指示信息和 RFC 5023 中的 Atom 发布协议规范来检查该服务文档。

1. 如果您想将 Atom 服务文档定义为 Atom 配置文件，请将根元素 <cics:atomservice type=″service″>，作为该文档的开头，并添加 <app:service> 元素作为子元素。

```
<?xml version="1.0"?>
<cics:atomservice type="service">
 <app:service>
 </app:service>
</cics:atomservice>
```

   如果不想将 Atom 服务文档定义为 Atom 配置文件，请使用 <app:service> 元素作为根元素，如下所示:

```
<?xml version="1.0"?>
<app:service>
</app:service>
```

2. 在 Atom 服务文档的根元素中包括 Atom XML 名称空间和 Atom 发布协议名称空间的名称空间声明，即， <cics:atomservice> 元素或 <app:service> 元素。 如果您已使用 <cics:atomservice> 元素作为根元素，那么还应包含针对 CICS Atom XML 名称空间的名称空间声明。 例如:

```
<?xml version="1.0"?>
<cics:atomservice type="service"
      xmlns:cics="http://www.ibm.com/xmlns/prod/cics/atom/atomservice"
      xmlns:atom="http://www.w3.org/2005/Atom"
      xmlns:app="http://www.w3.org/2007/app">
 <app:service>
 </app:service>
</cics:atomservice>
```

将 <app:service> 元素用作根元素时，这些声明如下所示:

```
<?xml version="1.0"?>
<app:service
      xmlns:atom="http://www.w3.org/2005/Atom"
      xmlns:app="http://www.w3.org/2007/app">
</app:service>
```

3. 至少添加一个 <app:workspace> 元素作为 <app:service> 元素的子元素，并向每个工作空间添加一个具有合适标题的 <atom:title> 元素。 <app:workspace> 元素只是在服务文档中将集合分组在一起，并不会要求服务器和客户机执行任何与这些集合有关的操作。但是，Atom 发布协议要求您至少使用一个工作空间，而且每个工作空间都要有一个易读的标题。 该示例显示可以包含所有集合的单个工作空间。

```
<?xml version="1.0"?>
<cics:atomservice type="service"
      xmlns:cics="http://www.ibm.com/xmlns/prod/cics/atom/atomservice"
      xmlns:atom="http://www.w3.org/2005/Atom"
      xmlns:app="http://www.w3.org/2007/app">
 <app:service>
  <app:workspace>
     <atom:title>CICS Atom collections</atom:title>
  </app:workspace>
 </app:service>
</cics:atomservice>
```

4. 对于每个集合，添加 <app:collection> 元素作为 <app:workspace> 元素（如果您已创建多个 <app:workspace> 元素，那么选择合适的那个）的子元素。 在每个 <app:collection> 元素中指定以下必需项:

   a. <app:collection> 元素的 href 属性，用于指定集合的完整 URL，具有在 Atom 配置文件中为该集合指定的完整路径。 Web 客户机在 GET 请求中使用该 URL 来检索集合中 Atom 条目的列表。还可以在 POST 请求中使用该 URL 来向集合提交新的 Atom 条目。 您必须确保该链接有效，并且已经设置支持集合的所有项，包括 URIMAP 资源定义和 ATOMSERVICE 资源定义。

   b. <atom:title> 元素作为子元素，用于指定集合的标题。

   例如:

```
<app:workspace>
    <atom:title>CICS Atom collections</atom:title>
  <app:collection
      href="http://www.example.com/customers/customercol.atom">
         <atom:title>Customer collection</atom:title>
  </app:collection>
</app:workspace>
```

5. 可选: 如果您不希望客户机在您的一个或多个集合中创建新 Atom 条目，请包括空 <app:accept> 元素作为 <app:collection> 元素的子元素。 例如:

```
<app:collection
    href="http://www.example.com/customers/customercol.atom">
       <atom:title>Customer collection</atom:title>
       <app:accept/>
</app:collection>
```

   您必须实施其他的安全措施以防止客户机编辑集合。客户机会将空 <app:accept> 元素解释为其不能创建条目，但是空 <app:accept> 元素并不能让 CICS 阻止客户机尝试创建条目。

   如果您希望客户机创建 Atom 条目，请省略 <app:accept> 元素。由于 CICS 不支持介质资源，所以请勿使用 <app:accept> 元素来为条目指定任何其他介质类型。如果客户机请求要创建的条目不是 Atom 条目文档（即介质类型为 application/atom+xml、带/不带 type=entry 属性的文档），那么 CICS 将拒绝这类客户机请求。由于该介质类型是缺省类型，所以在省略 <app:accept> 元素后，客户机应了解其只能在该集合中创建 Atom 条目文档。

6. 可选: 如果想要在一个或多个集合中为 Atom 条目指定类别（可选），请决定是在服务文档本身中提供类别列表还是提供对某个单独类别文档的引用。 如果类别列表过长或想要针对多个集合复用同一类别列表，那么使用单独的类别文档就很有用。Atom 发布协议允许在服务文档中使用多个 <app:categories> 元素，这样您就可以选择同时提供对共享 Atom 类别文档的引用和特定于集合的类别列表。 做出决定后，请继续执行以下操作:

   a. 要在服务文档中提供类别，请添加 <app:categories> 元素作为 <app:collection> 元素的子元素，然后添加一个或多个 <atom:category> 元素作为 <app:categories> 元素的子元素。向每个 <atom:category> 元素添加一个 term 属性，用于指定类别的名称。 CICS 不支持可选方案和标签属性，因此请勿使用这些属性。 例如:

```
<app:collection
    href="http://www.example.com/customers/customercol.atom">
       <atom:title>Customer collection</atom:title>
    <app:categories>
       <atom:category term="Customers" />
       <atom:category term="Actions" />
    </app:categories>
</app:collection>
```

      您指定的类别不会影响 CICS 的行为方式。如果客户机请求指定您列表以外的类别，那么 CICS 将接受该客户机请求。正因为如此，对于 CICS 直接处理资源的请求，请勿在 <app:categories> 元素中使用 fixed=″yes″ 属性，这会表明服务器不允许任何其他类别。当您忽略 fixed 属性时，会假设其值为“no”。如果您正在使用服务例程来更改资源，那么可以对服务例程进行编码以根据类别拒绝客户机请求，并在 <app:categories> 元素中予以表明。

   b. 要在单独类别文档中提供类别，请按照第 290 页的『创建 Atom 类别文档』中的指示信息来创建类别文档及其需要的资源定义。然后添加 <app:categories> 元素作为 <app:collection> 元素的子元素，同时向其添加一个 href 属性，用于指定可供客户机检索类别文档的 URL。 例如:

```
<app:collection
    href="http://www.example.com/customers/customercol.atom">
       <atom:title>Customer collection</atom:title>
    <app:categories href="http://www.example.com/cat/customercol"/>
</app:collection>
```

当指定对类别文档的引用时，请勿在 <app:categories> 元素中使用任何属性或子元素。

具有两个集合的示例 Atom 服务文档

该 Atom 服务文档被定义为 Atom 配置文件，具有根元素 <cics:atomservice type="service">。其包含一个具有两个集合的工作空间。其中一个集合在服务文档中有一个类别列表。另一个集合具有对单独类别文档的引用。

```
<?xml version="1.0"?>
<cics:atomservice type="service"
      xmlns:cics="http://www.ibm.com/xmlns/prod/cics/atom/atomservice"
      xmlns:atom="http://www.w3.org/2005/Atom"
      xmlns:app="http://www.w3.org/2007/app">
 <app:service>
  <app:workspace>
    <atom:title>CICS Atom collections</atom:title>
   <app:collection
      href="http://www.example.com/customers/customercol.atom">
         <atom:title>Customer collection</atom:title>
      <app:categories>
         <atom:category term="Customers" />
         <atom:category term="Actions" />
      </app:categories>
   </app:collection>
   <app:collection
      href="http://www.example.com/sysadmin/messagecol.atom">
      <atom:title>Messages from the systems administrator</atom:title>
      <app:categories href="http://www.example.com/cat/messagecol"/>
   </app:collection>
  </app:workspace>
 </app:service>
</cics:atomservice>
```

下一步做什么

Atom 发布协议允许您在 Atom 订阅源文档中指定 <app:collection> 元素，作为 <atom:feed> 元素的子元素。在将集合链接到该数据的普通 Atom 订阅源时，该协议很有用。理解该标记的客户机可以看到其请求的 Atom 订阅源作为集合提供，并且可使用 URL 来创建新条目或浏览条目以进行编辑，而无需查看服务文档。

如果希望指定 Atom 订阅源文档中的 <app:collection> 元素，请将 <app:collection> 元素及其所有子元素从 Atom 服务文档复制到该 Atom 订阅源的 Atom 配置文件，作为 <atom:feed> 元素的子元素。请确保其就是该元素的子元素，而不是 <cics:feed> 元素或原型 <atom:entry> 元素的子元素。您还必须在 Atom 服务文档中保持 <app:collection> 元素的规范性。如果将来要在 Atom 服务文档中更改 <app:collection> 元素的规范，请对 Atom 配置文件中的副本进行相应的更改。

然后，遵循第 292 页的『交付 Atom 服务或类别文档作为 Atom 配置文件』或第 293 页的『交付 Atom 服务或类别文档作为静态响应』中的指示信息来设置资源定义，以将 Atom 服务文档交付给 Web 客户机。

创建 Atom 类别文档

如果希望为集合提供类别的长列表，或者针对不同的集合复用相同的类别列表，那么可以创建 Atom 类别文档。简单的类别列表或唯一的类别列表对于定义个别类别文档几乎没有意义；因此请在 Atom 服务文档中指定类别。

关于此任务

Atom 类别文档存储在 z/OS UNIX 系统服务中。Atom 类别文档是 XML 文档，文件的扩展名为 .xml。您可以使用任何 XML 编辑器或文本编辑器创建该文件。

至于 Atom 服务文档，您可以选择是创建 Atom 类别文档作为 Atom 配置文件，还是将这些文档作为由 CICS Web Support 静态内容交付传递的文件。所选方法应与在第 287 页的『创建 Atom 服务文档』中为 Atom 服务文档选择的方法相同。CICS 不会验证 Atom 类别文档的内容。

但 <cics:atomservice type=″category″> 根元素除外，如果您使用该元素，那么 CICS 中的 Atom 类别文档仅使用 RFC 5023（*The Atom Publishing Protocol*）中定义的标准元素。如果您需要进一步的参考信息，请查阅该文档。

在类别文档中指定的类别不会影响 CICS 的行为方式。对于指定的类别并未包含在文档中的客户机请求，CICS 会接受这些请求。正因为如此，对于 CICS 直接处理资源的请求，请勿在 <app:categories> 元素中使用 fixed=″yes″ 属性，这会表明服务器不允许任何其他类别。如果您正在使用服务例程来更改资源，那么可以对服务例程进行编码以根据类别拒绝客户机请求，并在 <app:categories> 元素中予以表明。

请注意，对于资源中保存的数据，CICS 针对每个 Atom 条目仅支持一个类别。

1. 如果要定义 Atom 类别文档作为 Atom 配置文件，那么以根元素 <cics:atomservice type=″category″> 开始该文档，并添加 <app:categories> 元素作为子元素。

   ```
   <?xml version="1.0"?>
   <cics:atomservice type="category">
    <app:categories>
    </app:categories>
   </cics:atomservice>
   ```

   如果不想将 Atom 类别文档定义为 Atom 配置文件，那么使用 <app:categories> 元素作为根元素，如下所示:

   ```
   <?xml version="1.0"?>
   <app:categories>
   </app:categories>
   ```

   因为 CICS 不支持可选的 scheme 属性，所以请勿使用该属性。对于 CICS 直接处理资源的请求，因为 CICS 不会针对不接受的类别提供任何支持，所以请勿使用 fixed=″yes″ 属性。当您忽略 fixed 属性时，会假设其值为“no”。对于您正在使用服务例程来处理资源的请求，如果服务例程根据类别拒绝了请求，那么可以使用 fixed=″yes″ 属性。

2. 在 Atom 类别文档的根元素（即 <cics:atomservice> 元素或 <app:categories> 元素）中，包含针对 Atom XML 名称空间的名称空间声明以及针对“Atom 发布协议”的名称空间。 如果您已使用 <cics:atomservice> 元素作为根元素，那么还应包含针对 CICS Atom XML 名称空间的名称空间声明。 例如:

   ```
   <?xml version="1.0"?>
   <cics:atomservice type="category"
         xmlns:cics="http://www.ibm.com/xmlns/prod/cics/atom/atomservice"
         xmlns:atom="http://www.w3.org/2005/Atom"
         xmlns:app="http://www.w3.org/2007/app">
    <app:categories>
    </app:categories>
   </cics:atomservice>
   ```

如果使用 <app:categories> 元素作为根元素，那么声明类似如下:

```
<?xml version="1.0"?>
<app:categories
      xmlns:atom="http://www.w3.org/2005/Atom"
      xmlns:app="http://www.w3.org/2007/app">
</app:categories>
```

3. 添加一或多个 <atom:category> 元素作为 <app:categories> 元素的子元素，并且给出每个 <atom:category> 元素的术语属性，用于指定类别的名称。 例如:

```
 <app:categories>
    <atom:category term="Events" />
    <atom:category term="Comments" />
 </app:categories>
```

CICS 不支持可选方案和标签属性，因此请勿使用这些属性。

4. 通过在每个适用的集合中指定具有 href 属性的 <app:categories> 元素，在 Atom 服务文档中包含针对类别文档的 URL。 以下示例显示如何在 Atom 服务文档中将 <app:categories> 元素指定为 <app:collection> 元素的子元素:

```
  <app:collection
      href="http://www.example.com/events/eventcol.atom">
         <atom:title>Events collection</atom:title>
      <app:categories href="http://www.example.com/cat/eventcol"/>
  </app:collection>
```

示例

将 Atom 类别文档定义为 Atom 配置文件，且根元素为 <cics:atomservice type=″category″>。

```
 <?xml version="1.0"?>
<cics:atomservice type="category"
      xmlns:cics="http://www.ibm.com/xmlns/prod/cics/atom/atomservice"
      xmlns:atom="http://www.w3.org/2005/Atom"
      xmlns:app="http://www.w3.org/2007/app">
 <app:categories>
    <atom:category term="Events" />
    <atom:category term="Comments" />
 </app:categories>
</cics:atomservice>
```

下一步做什么

遵循『交付 Atom 服务或类别文档作为 Atom 配置文件』或第 293 页的『交付 Atom 服务或类别文档作为静态响应』中的指示信息来设置资源定义，以便向 Web 客户机交付 Atom 类别文档。

交付 Atom 服务或类别文档作为 Atom 配置文件

如果创建了 Atom 服务或类别文档作为含有根元素 <cics:atomservice> 的 Atom 配置文件，那么可以设置 URIMAP 和 ATOMSERVICE 资源定义来交付该文档。

1. 按照第 262 页的『为 Atom 文档创建 URIMAP 资源定义』中的指示信息，为 Atom 服务或类别文档设置 URIMAP 定义。 您为 Atom 服务文档选择的 URL 的路径部分**不能**以 URIMAP 资源定义中为（通过使用相同主机名提供服务的）Atom 订阅源或集合指定的路径部分的任何公共部分开始。
2. 使用 *CICS Resource Definition Guide* 中列出的一种方法，使 ATOMSERVICE 资源定义以在 URIMAP 资源定义中为 Atom 文档指定的名称及选择的组开头。

3. 使用 STATUS 属性来指定应在启用还是禁用状态下安装 ATOMSERVICE 资源定义。
4. 为 Atom 服务文档指定 SERVICE 的 ATOMTYPE 属性，或为 Atom 类别文档指定 CATEGORY。
5. 指定 CONFIGFILE 属性作为针对 Atom 文档而创建的 Atom 配置文件的名称和路径。
6. 安装 ATOMSERVICE 资源定义和在第 262 页的『为 Atom 文档创建 URIMAP 资源定义』中创建的相应 URIMAP 资源定义。 如果为第 261 页的『为 Atom 订阅源创建别名事务』中的别名事务创建了 TRANSACTION 资源定义，那么也要安装该资源。

交付 Atom 服务或类别文档作为静态响应

如果创建了含有根元素 <app:service> 或 <app:categories> 的普通 Atom 服务或类别文档，那么可以设置 URIMAP 资源定义以通过 CICS Web Support 将文档作为静态响应进行交付。

1. 按照第 59 页的『用 CICS 文档模板或 z/OS UNIX 文件提供静态 HTTP 响应』中的描述，完成规划步骤，将 z/OS UNIX 文件作为静态响应进行交付。
2. 按照第 87 页的『为作为 HTTP 服务器的 CICS 的任何请求启动 URIMAP 资源定义』和第 90 页的『为作为 HTTP 服务器的 CICS 的 HTTP 请求的静态响应完成 URIMAP 定义』中的描述，完成相应的步骤，为作为静态响应的 z/OS UNIX 文件设置 URIMAP 定义。 在执行以下这些步骤时，请在 URIMAP 定义中对您的 Atom 服务或类别文档做如下选择:
   a. 对于服务或类别文档，将 PATH 属性指定为适当的 URL，例如 `/servicedocument`。 Atom 服务或类别文档的路径不能以通过指定主机名提供服务的任何 Atom 订阅源和集合的路径部分的公共部分开始。
   b. 切记指定 SERVER（作为 HTTP 服务器的 CICS）的 USAGE 属性，而不是 Atom 的该属性。
   c. 对于 Atom 服务文档，将 MEDIATYPE 属性指定为 `application/atomsvc+xml`，或对于 Atom 类别文档，将该属性指定为 `application/atomcat+xml`。
   d. 将 HFSFILE 属性指定为包含服务或类别文档的 z/OS UNIX 文件的名称。

第 25 章 如何编辑 Atom 集合

当 Atom 条目可用作集合时，客户机可以编辑或删除现有条目，并为该集合创建新条目。

为了发现服务器上可用的集合，客户机需要请求来自服务器的服务文档。服务文档列出可用于客户机的集合的 URL。然后，客户机可以通过向服务器发出 HTTP 请求，与 Atom 条目进行交互，如下所示:

GET 检索单个 Atom 条目或 Atom 条目列表。如 Atom 服务文档所述，将针对 Atom 条目列表的 GET 请求发送至集合的 URL。如条目的 <atom:link rel="edit"> 链接中所述，将针对单个 Atom 条目的 GET 请求发送至集合中单个 Atom 条目的 URL。

POST 创建新的 Atom 条目。POST 请求被发送至集合的 URL。

PUT 编辑客户机使用 GET 请求获取的现有 Atom 条目。PUT 请求被发送至集合中单个 Atom 条目的 URL。

DELETE
删除现有 Atom 条目。DELETE 请求被发送至集合中单个 Atom 条目的 URL。

对于 POST 和 PUT 请求，客户机发送的请求主体包含完整的 Atom 条目文档。（CICS 不支持 Atom 集合中的其他介质类型。）尽管 PUT 请求通常只更新现有条目的一部分，但是客户机应该发送整个条目，包括现有条目中未编辑的元素，以避免服务器可能造成的错误解释。服务器会添加新条目或更新现有条目，然后向客户机发送具有适当 HTTP 状态码的 HTTP 响应。服务器可以更改、添加或除去客户机为条目提供的元数据项（例如，Atom 标识或日期和时间戳记）。因此，当客户机成功发出 POST 或 PUT 请求后，服务器还会将新条目的副本作为响应主体返回。对于集合中的 Atom 条目，CICS 需要 PUT 请求的实体标记（ETags），这使服务器能够确认客户机的编辑请求基于最新的 Atom 条目副本。

RFC 5023（*The Atom Publishing Protocol*）可从 http://www.ietf.org/rfc/rfc5023.txt 中获得，它完整陈述了用于与集合中的 Atom 条目进行交互的协议，以及所使用 HTTP 请求和响应的一些示例。

对于 CICS 直接为其从文件或临时存储队列中抽取数据而不涉及服务例程的集合，CICS 会执行服务器操作，以处理客户机请求、更新资源以及向客户机返回响应。请注意，这些操作将永久修改文件或临时存储器队列的内容。CICS 添加或编辑记录以响应 POST 和 PUT 请求。对于和文件相关的 DELETE 请求，CICS 会删除记录，但 ESDS 除外，因为其中的记录无法删除。对于和临时存储器队列相关的 DELETE 请求，CICS 通过将第一个字节设置为 'FF'x 来除去记录。如果这些操作不适合您的环境，请改用服务例程来处理客户机请求。

CICS 不支持“Atom 发布协议”（RFC 5023）中针对集合描述的一些可能操作，具体如下:

- CICS 不支持集合中的介质资源和介质链接条目。Atom 发布协议（RFC 5023）指定的介质资源，主要用作组织集合中非文本内容的一种方法。如果客户机请求尝试在集合中创建的条目不是 Atom 条目（介质类型为 application/atom+xml，带/不带

type=entry 参数），那么 CICS 将通过 415 状态码来拒绝这类客户机请求。请勿在 CICS 服务文档的 <app:accept> 元素中指定任何其他介质类型。

- CICS 不会拒绝集合中按类别划分的 Atom 条目。您可以在服务文档或类别文档中使用 <app:categories> 元素来指定集合中条目可接受的类别，但是 CICS 不检查客户机是否采用这些类别。
- 出于性能考虑，CICS 不会自动按集合中 Atom 条目最近一次被编辑的顺序（如条目中的 <app:edited> 元素所示）来返回这些 Atom 条目。该功能是 RFC 5023 中针对完整条目列表的一个“SHOULD”需求，但它却是针对部分条目列表的“MUST”需求。CICS 为了保持较短的响应时间而忽略了此“SHOULD”需求，但它仍然为部分列表提供了有用的功能。在使用服务例程为 CICS 提供条目时，如果希望集合符合此“SHOULD”需求并且资源可存储上一次编辑条目的日期和时间戳记，那么您可以按这个日期和时间戳记的顺序来提供条目。

第 44 页的『CICS 不支持的 Atom 功能』列出了其他一些次要的或与集合并不明确相关的不受支持的项。

对于由 CICS 提供并包含服务例程从资源中抽取并提供给 CICS 的数据的集合，CICS 会通过一组容器将客户机的请求传递至服务例程。必须对服务例程进行编码，以对资源中的数据应用该请求，然后向 CICS 返回一个响应以便发送给客户机。在这种情况下，就某些方面而言，您与 CICS 同样有责任确保符合“Atom 发布协议”（RFC 5023）。第 307 页的『如何在服务例程中处理 Atom 集合编辑请求』解释了服务例程如何处理针对集合的 GET、POST、PUT 和 DELETE 请求。

在完成设置 Atom 集合和服务例程（根据需要）后，可以使用用于处理 HTTP 协议的任何适当的客户机来编辑条目。发出请求和查看响应的确切过程会因所选客户机而异。要了解有关如何使用 GET、POST、PUT 和 DELETE 请求与集合中的条目进行交互的更多信息，请参阅『使用 Web 客户机编辑 Atom 集合』。

第 285 页的第 24 章，『将 Atom 订阅源加入集合』
要通过现有 Atom 订阅源创建可编辑的集合，请设置新的 URIMAP 和 ATOMSERVICE 资源定义，以及新的 Atom 配置文件。新定义使用了初始 Atom 订阅源的大部分设置，只进行了少量更改。

使用 Web 客户机编辑 Atom 集合

使用 Web 客户机发出 HTTP GET、POST、PUT 和 DELETE 请求，以读取、创建、编辑和删除 Atom 集合中的条目。

开始之前

CICS 提供一个由内容聚合 Web 页面支持的样本 Atom 集合，该页面允许您针对集合发出 Web 客户机请求以及查看请求与响应。要了解有关 Web 客户机可以如何在 CICS 中编辑 Atom 集合，请按照 CICS TS for z/OS V4.1 信息中心（`https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp`）内的 Web 2.0 案例“创建 Atom 订阅源以使用员工信息” 中的指示信息，设置并使用样本 Atom 集合。

关于此任务

众多免费或商用 Web 客户机应用程序都可以发出 HTTP GET 请求，以获取并显示 Atom 订阅源或集合。然而，并不是所有这些 Web 客户机都可以发出 HTTP POST、PUT 和 DELETE 请求来编辑 Atom 条目。请进行检查，确保该应用程序被明

确描述为支持“Atom 发布协议”，而不只是 Atom 格式。如果您的 Web 客户机具有该功能，那么请参阅客户机文档，了解发出请求和查看响应的过程。Web 客户机在利用任何支持“Atom 发布协议”的服务器时，都能够与 CICS 进行交互，但第 295 页的第 25 章，『如何编辑 Atom 集合』中列出的某些功能除外。

如果不具备明确支持向 Atom 条目发出 POST、PUT 和 DELETE 请求的 Web 客户机应用程序，那么可以使用使您能够组成并发送自己的 HTTP 请求以及查看响应的 Web 客户机应用程序。您也可以编写自己的 Web 客户机应用程序，以向 Atom 集合发出 POST、PUT 和 DELETE 请求。要了解有关通过 CICS 应用程序发出 Web 客户机请求的指示信息，请参阅第 129 页的『通过作为 HTTP 客户机的 CICS 发出 HTTP 请求』。

如果正在编写自己的 Web 客户机应用程序以编辑 Atom 集合，或者正在适当的 Web 客户机中生成自己的 HTTP 请求，请按照此处所述的步骤发出请求。如果需要关于任何步骤的进一步信息，请参阅 http://www.ietf.org/rfc/rfc5023.txt 中的 RFC 5023（*The Atom Publishing Protocol*），以获取用于和集合中的 Atom 条目进行交互的协议的详细信息。RFC 2616（*Hypertext Transfer Protocol -- HTTP/1.1*）可从 http://www.ietf.org/rfc/rfc2616.txt 获取，它描述了用于发出 HTTP 请求的协议。

您的集合应实施适当的安全措施以控制 Web 客户机访问。这些指示信息假定您正在使用的 Web 客户机具有完全权限，可以读取和修改 Atom 条目。对于 CICS 提供的 Atom 订阅源和集合，您可以允许某些 Web 客户机对一些项具有只读访问权，这样就只能发出 HTTP GET 请求，但是，其他 Web 客户机可以具有 UPDATE 访问权，这样它们还可以发出 HTTP POST、PUT 和 DELETE 请求。如果 Web 客户机不具有适当的权限而无法对集合执行操作，那么 CICS 会返回 HTTP 错误响应，状态码为 403（已禁止）。有关 Atom 订阅源和集合的安全性的更多信息，请参阅第 317 页的第 26 章，『Atom 订阅源的安全性』。

1. 如果您不知道想要编辑的集合的 URL 或者正在编写的可处理多个集合的应用程序，那么首先应针对第 287 页的『创建 Atom 服务文档』中设置的 Atom 服务文档发出 HTTP GET 请求。 Atom 服务文档列出在服务器上可用的集合的 URL。 您还可以检查集合中条目的可能类别，这些类别将在服务文档或单独的类别文档中列出。
2. 要获取集合中现有 Atom 条目的列表，请按照第 298 页的『向 Atom 订阅源或集合发出 GET 请求』中的指示信息向集合的 URL 发出 HTTP GET 请求。 这些指示信息也适用于针对未定义为集合的 Atom 订阅源的 GET 请求。
3. 要从集合获取个别 Atom 条目，请按照第 298 页的『向 Atom 订阅源或集合发出 GET 请求』中的指示信息，向 Atom 条目的 URL 发出 HTTP GET 请求。 这些指示信息也适用于针对未定义为集合的 Atom 订阅源的 GET 请求。
4. 要在集合中创建新的 Atom 条目，请按照第 302 页的『向 Atom 集合发出 POST 请求』中的指示信息向该集合的 URL 发出 HTTP POST 请求。
5. 要编辑集合中的现有 Atom 条目，请按照第 305 页的『向 Atom 集合发出 PUT 请求』中的指示信息向 Atom 条目的 URL 发出 HTTP PUT 请求。
6. 要删除集合中的现有 Atom 条目，请按照第 306 页的『向 Atom 集合发出 DELETE 请求』中的指示信息向 Atom 条目的 URL 发出 HTTP DELETE 请求

向 Atom 订阅源或集合发出 GET 请求

Web 客户机可以通过向 Atom 订阅源或集合的 URL 发出 HTTP GET 请求来获取该订阅源或集合中现有 Atom 条目的列表，也可以向 Atom 条目的 URL 发出 HTTP GET 请求以便从 Atom 订阅源或集合中获取个别 Atom 条目。

开始之前

如果您希望编辑一个集合，但又不知道其 URL；或者正在编写可处理多个集合的应用程序，那么首先应针对在第 287 页的『创建 Atom 服务文档』中设置的 Atom 服务文档发出 HTTP GET 请求。Atom 服务文档列出在服务器上可用的集合的 URL。 Atom 订阅源的 URL 无法通过 Atom 服务文档获得，但是，这些 URL 通常在相关位置发布，例如，作为 Web 站点上针对预订订阅源的邀请。

1. 要获得 Atom 订阅源或集合中现有 Atom 条目的列表，请针对该订阅源或集合的 URL 发出 HTTP GET 请求。请按照以下方式组成请求:
   a. 以包含 GET 方法的请求行开始，接着是 Atom 订阅源或集合 URL 的路径部分，最后是 `HTTP/1.1`，即该请求的 HTTP 版本。 您还可以在 URL 中包含方案（HTTP 或 HTTPS）和主机名。 有关请求行的解释，请参阅 第 12 页的『HTTP 请求』。
   b. 如果您希望获得的 Atom 条目数不等于 CICS 为响应针对该 Atom 订阅源或集合的单个请求而发出的缺省数目，请在 URL 的末尾指定查询字符串，该字符串包含名称 `w`（表示“窗口”）以及表示希望在该 Atom 文档中收到的 Atom 条目数的值。 该查询字符串请求 6 个 Atom 条目:

      ```
      ?w=6
      ```

   c. 请如下所示为请求编写 HTTP 头，每个另起一行:
      - Host 头，用于通过 Atom 订阅源或集合的 URL 提供主机名 – 如果尚未在请求行中包含主机名。
      - Authorization 头，包含访问 Atom 订阅源或集合所需的任何安全信息，例如，用于基本认证的用户标识和密码。

      在上一个 HTTP 头之后再输入一个回车换行符（CRLF），以给出一个空行。

   请勿包含请求主体。 向服务器发送请求。 服务器返回 Atom 订阅源文档，这是单个 HTTP 响应，其中包含 Atom 订阅源或集合中全部或部分 Atom 条目的列表。

   注: 由 CICS 提供服务的 Atom 文档中的元素通常不包含 atom: 名称空间前缀。由于已在 Atom 文档起始处的元素中将 Atom 名称空间定义为缺省名称空间，因此子元素不需要 atom: 前缀。但是，在引用 CICS 文档中的元素时，为清晰起见，请在元素名称中使用 atom: 前缀。
2. 如果收到 Atom 订阅源或集合中 Atom 条目的部分列表，并希望获得更多条目，请针对指定更多部分列表的 Atom 订阅源文档中的链接发出更多 HTTP GET 请求，如下所示:
   a. 使用在 <atom:link rel="next"> 元素中声明的 URL 获取条目的下一个窗口或部分列表。 CICS 为 Atom 订阅源和集合都提供该链接。 如上所述，CICS 针对子元素未包含 atom: 名称空间前缀，因此该元素在 Atom 订阅源文档中显示为 <link rel="next">。
   b. 使用在 <atom:link rel="previous"> 元素中声明的 URL 获取条目的上一个部分列表。 CICS 为集合提供该链接。

c. 使用在 <atom:link rel="first"> 元素中指定的 URL 获取条目的第一个部分列表。CICS 为集合提供该链接。

d. 使用在 <atom:link rel="last"> 元素中指定的 URL 获取条目的最后一个部分列表。CICS 为集合提供该链接。 对于 CICS 提供的 Atom 文档，为提高性能，这个部分列表应当只包含订阅源中的最后一个 Atom 条目。您可以使用 <atom:link rel="previous"> 链接来检索先前所有的部分列表。

请注意，针对更多部分列表的链接表示在 CICS 发出使 Atom 文档包含 Atom 订阅源或集合的 Atom 命令时该订阅源或集合的快照。在向集合添加或从中删除 Atom 条目时，这些链接可能变为无效。因此，链接适合于短期目的，用于浏览不会快速变化的集合。

3. 要从 Atom 订阅源或集合获取个别 Atom 条目，请针对 Atom 条目的 URL 发出 HTTP GET 请求，该请求在条目的 <atom:link rel="self"> 或 <atom:link rel="edit"> 元素中声明。 <atom:link rel="self"> 提供订阅源中 Atom 条目的链接，<atom:link rel="edit"> 提供集合中 Atom 条目的链接。集合中的 Atom 条目还可能具有 <atom:link rel="self"> 链接。 请按照以下方式组成请求:

   a. 以包含 GET 方法的请求行开始，接着是 Atom 条目 URL 的路径部分，最后是 HTTP/1.1，即该请求的 HTTP 版本。 您还可以在 URL 中包含方案（HTTP 或 HTTPS）和主机名。 有关请求行的解释，请参阅 第 12 页的『HTTP 请求』。

   b. 请如下所示为请求编写 HTTP 头，每个另起一行:

      - Host 头，用于通过 Atom 订阅源或集合的 URL 提供主机名 – 如果尚未在请求行中包含主机名。
      - Authorization 头，包含访问 Atom 订阅源或集合所需的任何安全信息，例如，用于基本认证的用户标识和密码。

      在上一个 HTTP 头之后再输入一个回车换行符（CRLF），以给出一个空行。

   请勿包含请求主体。 向服务器发送请求。 该服务器会返回一个 HTTP 响应，其中包含各个 Atom 条目的副本。

示例

以下 HTTP 请求针对具有 URL http://www.mycics.com:80/web20/myfeed 的 Atom 集合:

```
GET /web20/myfeed HTTP/1.1
Host: www.mycics.com:80
```

以下 HTTP 请求针对具有 URL http://www.mycics.com:80/web20/entry/7 的 Atom 条目:

```
GET /web20/entry/7 HTTP/1.1
Host: www.mycics.com:80
```

以下 HTTP 响应示例显示 CICS 为响应针对单个 Atom 条目的请求而发送的 Atom 文档。该示例仅显示与 Atom 订阅源相关的 HTTP 头; 响应中可能包含其他 HTTP 头。HTTP 响应的 ETag 头为 Atom 条目提供实体标记，如果您发出 PUT 请求以编辑条目，那么必须在 If-Match 头中使用该标记。

```
HTTP/1.1 200 OKContent-Type: application/atom+xml
Content-Length: 1005
ETag: c4826af12991fb102ef13099c927c2ac24e4caa2

<?xml version="1.0" encoding="utf-8"?>
```

```
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:app="http://www.w3.org/2007/app">
    <generator uri="http://www.ibm.com/cics/" version="6.6.0">
    CICS Transaction Server Version 4.1.0
    </generator>
    <link rel="self" href="http://www.mycics.com:80/web20/entry/7"/>
    <link rel="edit" href="http://www.mycics.com:80/web20/entry/7"/>
    <id>tag:http://www.mycics.com/web20/myfeed,2009-01-20:tsqueue:WB20TSQ:7</id>
    <title>This is entry 7</title>
    <summary>
    Entry 7 is about something to do with feeds...
    </summary>
    <category term="Test Feeds"/>
    <rights>Copyright (c) 2009, Joe Bloggs</rights>
    <published>2008-12-02T15:41:00</published>
    <author>
     <name>Joe Bloggs</name>
     <email>JBloggs@example.com</email>
     <uri>http://www.example.com/JBloggs/</uri>
    </author>
    <contributor>
     <name>John Doe</name>
    </contributor>
    <app:edited>2009-02-02T16:29:36+00:00</app:edited>
    <updated>2009-02-02T16:29:36+00:00</updated>
    <content type="text/xml">
        <SAMPBIND xmlns="http://www.ibm.com/xmlns/prod/cics/atom/bindfile/sampbind">
        <data_field>
        Here is some content for entry 7
        </data_field>
        </SAMPBIND >
    </content>
</entry>
```

以下 HTTP 响应示例显示 CICS 为响应针对 Atom 集合中 Atom 条目列表的请求而发送的 Atom 文档。同样，该示例仅显示与 Atom 订阅源相关的 HTTP 头；响应中可能包含其他 HTTP 头。

```
HTTP/1.1 200 OKContent-Type: application/atom+xml
Content-Length: 8661

<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:app="http://www.w3.org/2007/app">
    <generator uri="http://www.ibm.com/cics/" version="6.6.0">
    CICS Transaction Server Version 4.1.0
    </generator>
    <link rel="self" href="http://www.mycics.com:80/web20/myfeed"/>
    <link rel="edit" href="http://www.mycics.com:80/web20/myfeed"/>
    <id>tag:http://www.mycics.com/web20/myfeed,2009-01-20:tsqueue:WB20TSQ</id>
    <title type="text">CICS ATOM FEED TITLE</title>
    <subtitle>CICS ATOM FEED SUBTITLE</subtitle>
    <rights>Copyright (c) 2009, Joe Bloggs</rights>
    <category term="my-first-feed"/>
    <author>
     <name>Joe Bloggs</name>
     <email>JBloggs@example.com</email>
     <uri>http://www.example.com/JBloggs/</uri>
    </author>
    <contributor>
     <name>John Doe</name>
    </contributor>
    <!--------------------------------------------------------------------->
    <entry>
        <link rel="self" href="http://www.mycics.com:80/web20/entry/9"/>
        <link rel="edit" href="http://www.mycics.com:80/web20/entry/9"/>
        <id>tag:http://www.mycics.com/web20/myfeed,2009-01-20:tsqueue:WB20TSQ:9</id>
        <title>This is entry 9</title>
```

```
|          <summary>
|          Entry 9 is about something to do with feeds...
|          </summary>
|          <category term="Test Feeds"/>
|          <rights>Copyright (c) 2009, Joe Bloggs</rights>
|          <published>2008-12-02T15:41:00</published>
|          <author>
|           <name>Joe Bloggs</name>
|           <email>JBloggs@example.com</email>
|           <uri>http://www.example.com/JBloggs/</uri>
|          </author>
|          <contributor>
|           <name>John Doe</name>
|          </contributor>
|          <app:edited>2009-02-02T16:29:36+00:00</app:edited>
|          <updated>2009-02-02T16:29:36+00:00</updated>
|          <content type="text/xml">
|           <SAMPBIND xmlns="http://www.ibm.com/xmlns/prod/cics/atom/bindfile/sampbind">
|           <data_field>
|           Here is some content for entry 9
|           </data_field>
|           </SAMPBIND >
|          </content>
|      </entry>
|      <!--------------------------------------------------------------------->
|      <entry>
|          <link rel="self" href="http://www.mycics.com:80/web20/entry/8"/>
|          <link rel="edit" href="http://www.mycics.com:80/web20/entry/8"/>
|          <id>tag:http://www.mycics.com/web20/myfeed,2009-01-20:tsqueue:WB20TSQ:8</id>
|          <title>This is entry 8</title>
|          <summary>
|          Entry 8 is about something to do with feeds...
|          </summary>
|          <category term="Test Feeds"/>
|          <rights>Copyright (c) 2009, Joe Bloggs</rights>
|          <published>2008-12-02T15:41:00</published>
|          <author>
|           <name>Joe Bloggs</name>
|           <email>JBloggs@example.com</email>
|           <uri>http://www.example.com/JBloggs/</uri>
|          </author>
|          <contributor>
|           <name>John Doe</name>
|          </contributor>
|          <app:edited>2009-02-02T16:29:36+00:00</app:edited>
|          <updated>2009-02-02T16:29:36+00:00</updated>
|          <content type="text/xml">
|           <SAMPBIND xmlns="http://www.ibm.com/xmlns/prod/cics/atom/bindfile/sampbind">
|           <data_field>
|           Here is some content for entry 8
|           </data_field>
|           </SAMPBIND >
|          </content>
|      </entry>
|      <!--------------------------------------------------------------------->
|      <entry>
|          <link rel="self" href="http://www.mycics.com:80/web20/entry/7"/>
|          <link rel="edit" href="http://www.mycics.com:80/web20/entry/7"/>
|          <id>tag:http://www.mycics.com/web20/myfeed,2009-01-20:tsqueue:WB20TSQ:7</id>
|          <title>This is entry 7</title>
|          <summary>
|          Entry 7 is about something to do with feeds...
|          </summary>
|          <category term="Test Feeds"/>
|          <rights>Copyright (c) 2009, Joe Bloggs</rights>
|          <published>2008-12-02T15:41:00</published>
|          <author>
```

```
          <name>Joe Bloggs</name>
          <email>JBloggs@example.com</email>
          <uri>http://www.example.com/JBloggs/</uri>
         </author>
         <contributor>
          <name>John Doe</name>
         </contributor>
         <app:edited>2009-02-02T16:29:36+00:00</app:edited>
         <updated>2009-02-02T16:29:36+00:00</updated>
         <content type="text/xml">
          <SAMPBIND xmlns="http://www.ibm.com/xmlns/prod/cics/atom/bindfile/sampbind">
          <data_field>
          Here is some content for entry 7
          </data_field>
          </SAMPBIND >
         </content>
    </entry>
    <!--------------------------------------------------------------------->
    <entry>
         <link rel="self" href="http://www.mycics.com:80/web20/entry/6"/>
         <link rel="edit" href="http://www.mycics.com:80/web20/entry/6"/>
         <id>tag:http://www.mycics.com/web20/myfeed,2009-01-20:tsqueue:WB20TSQ:6</id>
         <title>This is entry 6</title>
         <summary>
         Entry 6 is about something to do with feeds...
         </summary>
         <category term="Test Feeds"/>
         <rights>Copyright (c) 2009, Joe Bloggs</rights>
         <published>2008-12-02T15:41:00</published>
         <author>
          <name>Joe Bloggs</name>
          <email>JBloggs@example.com</email>
          <uri>http://www.example.com/JBloggs/</uri>
         </author>
         <contributor>
          <name>John Doe</name>
         </contributor>
         <app:edited>2009-02-02T16:29:36+00:00</app:edited>
         <updated>2009-02-02T16:29:36+00:00</updated>
         <content type="text/xml">
          <SAMPBIND xmlns="http://www.ibm.com/xmlns/prod/cics/atom/bindfile/sampbind">
          <data_field>
          Here is some content for entry 6
          </data_field>
          </SAMPBIND >
         </content>
    </entry>
</feed>
```

向 Atom 集合发出 POST 请求

Web 客户机可以通过针对集合的 URL 发出 HTTP POST 请求，在集合中创建新的 Atom 条目。

开始之前

如果您不知道想要编辑的集合的 URL 或者正在编写的可处理多个集合的应用程序，那么首先应针对第 287 页的『创建 Atom 服务文档』中设置的 Atom 服务文档发出 HTTP GET 请求。Atom 服务文档列出在服务器上可用的集合的 URL。

1. HTTP POST 请求以包含 POST 方法的请求行开始，接着是集合 URL 的路径部分，最后是 HTTP/1.1，即该请求的 HTTP 版本。 您还可以在 URL 中包含方案（HTTP 或 HTTPS）和主机名。 有关请求行的解释，请参阅 第 12 页的『HTTP 请求』。

2. 请如下所示为请求编写 HTTP 头，每个另起一行:
   - Host 头，用于通过集合的 URL 提供主机名 – 如果尚未在请求行中包含主机名。
   - Authorization 头，包含访问集合所需的任何安全信息，例如，用于基本认证的用户标识和密码。
   - Content-Type 头，具有值 `application/atom+xml;type=entry`。
   - Content-Length 头，声明消息体的长度（字节，即八位元）。在编写完消息体后，填充该头的值。

   在上一个 HTTP 头之后再输入一个回车换行符（CRLF），以给出一个空行。
3. 设置消息体，该消息体包含要发布的 Atom 条目的 XML 标记。
   a. 如果集合中已有 Atom 条目，请发出 GET 请求以获取该集合的订阅源或条目文档，并将集合中的现有 Atom 条目复制到您的消息体中。
   b. 如果集合中尚无任何 Atom 条目，请根据这些主题中的示例为第一个 Atom 条目编写 XML 标记。
   c. 检查 Atom 条目起始处的 <entry> 标记是否包含名称空间声明 xmlns="http://www.w3.org/2005/Atom"，如果未包含，请添加该声明。 在由 CICS 发送的 Atom 文档中，这些元素通常不包含 atom: 名称空间前缀。由于已在 Atom 文档起始处的元素中将 Atom 名称空间定义为缺省名称空间，因此不需要对子元素使用 atom: 前缀。在 CICS 文档中，为了清晰起见，会在元素名称中使用该前缀。如果您希望在请求内的元素中使用 atom: 名称空间前缀，请将名称空间声明更改为 xmlns:atom="http://www.w3.org/2005/Atom"。
   d. 在 Atom 条目之前添加元素 <?xml version="1.0" ?>。
4. 在 Atom 条目的相应元素中填入您需要的数据，删除不使用的任何元素。 您不必要为 CICS 发出的 Atom 条目文档中存在的所有元素都提供数据。将某个条目发布到由 CICS 管理的集合时，您可以省略未存储在 CICS 资源中的任何元素，以及虽然存储在 CICS 资源中但由 CICS 或服务例程生成的任何元素。通常，您至少可以省略以下元素:
   - 包含时间戳记的元素，例如 <atom:updated> 元素（除非您的服务例程接受针对这些元素的用户输入）
   - <atom:id> 元素
   - <atom:contributor> 元素
   - <atom:link> 元素
   - <atom:rights> 元素

   如果不确定，请包含该元素，CICS 或服务例程将会忽略不想要的元素。 RFC 5023 提供 Atom 条目中可能元素的完整列表。有关 Atom 条目中必需元素、允许元素或禁止使用元素的概述，请参阅 第 281 页的『CICS 的 Atom 元素引用』。有关每个元素内容的描述，请参阅 第 277 页的『<atom:entry> 元素』。
5. 向服务器发送请求。

结果

该服务器会发送 HTTP 响应，状态码为 200，表明请求已成功完成或适当的错误响应。如果请求已成功完成，那么响应会包含新的 Atom 条目的副本。根据最初提交的请求检查来自服务器的响应中的 Atom 条目，以确保对新条目满意。

如果收到错误响应，请阅读该响应的消息体，然后参阅第 341 页的附录 C, 『CICS Web Support 的 HTTP 状态码参考』中的 CICS 为 Web 客户机提供的状态码的列表。如果错误响应具有 HTTP 状态码 400（“错误的请求”或“无效的请求”），请检查 CICS 区域是否发出了消息 DFHML0100。CICS 使用 z/OS XML System Services 解析器对 Atom 条目的 XML 标记进行语法分析。如果该解析器发现标记存在问题，那么 CICS 会发出消息 DFHML0100，其中包含来自该解析器的返回码和原因码。要获取返回码和原因码的说明，请参阅 *z/OS XML System Services User's Guide and Reference*（http://www.ibm.com/servers/eserver/zseries/zos/xml/）。

示例

以下请求要求在集合中新建具有 URL `http://www.mycics.com:80/web20/myfeed` 的 Atom 条目:

```
POST /web20/myfeed HTTP/1.1
Host: www.mycics.com:80
Content-Type: application/atom+xml;type=entry
Content-Length: 763

<?xml version="1.0" encoding="utf-8"?>
<entry xmlns="http://www.w3.org/2005/Atom">
    <title>This is my posted entry</title>
    <summary>
    This is my new posted entry
    </summary>
    <category term="Test Feeds"/>
    <author>
     <name>Joe Bloggs</name>
     <email>JBloggs@example.com</email>
     <uri>http://www.example.com/JBloggs/</uri>
    </author>
    <content type="text/xml">
     <SAMPBIND xmlns="http://www.ibm.com/xmlns/prod/cics/atom/bindfile/sampbind">
      <data_field>
       Here is content for my posted entry
     </data_field>
     </SAMPBIND >
    </content>
</entry>
```

以下是 CICS 发送的响应:

```
HTTP/1.1 201 Created
Content-Type: application/atom+xml;type=entry
Content-Length: 1029

<?xml version="1.0" encoding="utf-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:app="http://www.w3.org/2007/app">
    <generator uri="http://www.ibm.com/cics/" version="6.6.0">
    CICS Transaction Server Version 4.1.0
    </generator>
    <link rel="self" href="http://www.mycics.com:80/web20/entry/10"/>
    <link rel="edit" href="http://www.mycics.com:80/web20/entry/10"/>
    <id>tag:http://www.mycics.com/web20/myfeed,2009-01-20:tsqueue:WB20TSQ:10</id>
    <title>This is my posted entry</title>
    <summary>
    This is my new posted entry
    </summary>
    <category term="Test Feeds"/>
    <rights>Copyright (c) 2009, Joe Bloggs</rights>
    <published>2009-04-23T15:00:44+00:00</published>
    <author>
     <name>Joe Bloggs</name>
```

```
      <email>JBloggs@example.com</email>
      <uri>http://www.example.com/JBloggs/</uri>
     </author>
     <app:edited>2009-04-23T15:00:44+00:00</app:edited>
     <updated>2009-04-23T15:00:44+00:00</updated>
     <content type="text/xml">
      <SAMPBIND xmlns="http://www.ibm.com/xmlns/prod/cics/atom/bindfile/sampbind">
       <data_field>
        Here is content for my posted entry
       </data_field>
      </SAMPBIND >
     </content>
</entry>
```

向 Atom 集合发出 PUT 请求

Web 客户机可以通过向 Atom 条目的 URL 发出 HTTP PUT 请求（在该条目的 <atom:link rel="edit"> 元素中指定），编辑集合中的现有 Atom 条目。

关于此任务

使用个别 Atom 条目的 URL，您一次只能编辑一个 Atom 条目。您无法在单个请求中编辑多个 Atom 条目。CICS 会拒绝针对集合的 URL 发出的 PUT 请求。

1. 对 Atom 条目的 URL 发出 HTTP GET 请求（在该条目的 <atom:link rel="edit"> 元素中的声明），以检索该条目的当前副本并从响应的 ETag HTTP 头获取其实体标记。 第 298 页的『向 Atom 订阅源或集合发出 GET 请求』说明了如何执行此操作。
2. HTTP PUT 请求以包含 PUT 方法的请求行开始，接着是Atom 条目的 URL 的路径部分，最后是 `HTTP/1.1`，即该请求的 HTTP 版本。 您还可以在 URL 中包含方案（HTTP 或 HTTPS）和主机名。 有关请求行的解释，请参阅 第 12 页的『HTTP 请求』。
3. 请如下所示为请求编写 HTTP 头，每个另起一行:
   - Host 头，用于通过 Atom 条目的 URL 提供主机名 – 如果尚未在请求行中包含主机名。
   - Authorization 头，包含访问集合所需的任何安全信息，例如，用于基本认证的用户标识和密码。
   - If-Match 头，包含为现有 Atom 条目获取的实体标记。如果必须覆盖该检查，那么可以使用星号替换实体标记，这使服务器可以将编辑应用于 Atom 条目，即使自从您获取了该条目之后已经由其他代理程序对它进行了更改。请慎重使用该选项。
   - Content-Type 头，具有值 `application/atom+xml;type=entry`。
   - Content-Length 头，声明消息体的长度（字节，即八位元）。在编写完消息体后，填充该头的值。

   在上一个 HTTP 头之后再输入一个回车换行符（CRLF），以给出一个空行。
4. 将 Atom 条目复制到消息体中，然后在前面添加元素 <?xml version="1.0" ?>。 检查 <entry> 标记是否包含名称空间声明 xmlns="http://www.w3.org/2005/Atom"。
5. 编辑要更改的 Atom 条目中的元素内容。 CICS 提供了正确的时间戳记，服务例程也必须如此，因此请勿更新条目中的时间戳记。 有关每个元素内容的描述，请参阅第 277 页的『<atom:entry> 元素』。
6. 向服务器发送请求。

结果

当接受您的请求时，该服务器会发送 HTTP 响应，状态码为 200，表明请求已成功完成或适当的错误响应。CICS 将已编辑的 Atom 条目的副本作为响应主体返回，因此您可以根据需要验证自己的更改。

对于 POST 请求，如果收到错误响应，请阅读响应的消息体，然后参阅第 341 页的附录 C，『CICS Web Support 的 HTTP 状态码参考』中 CICS 为 Web 客户机提供的状态码的列表。对于 HTTP 状态码为 400 的错误响应，请查看来自 CICS 区域的消息 DFHML0100，并参阅 *z/OS XML System Services User's Guide and Reference*，以了解返回码和原因码的说明。

示例

以下是要求编辑具有 URL `http://www.mycics.com:80/web20/entry/10` 的 Atom 条目的 HTTP PUT 请求:

```
PUT /web20/entry/10 HTTP/1.1
Host: www.mycics.com:80
Content-Type: application/atom+xml;type=entry
Content-Length: 1034
If-Match: c4826af12991fb102ef13099c927c2ac24e4caa2

<?xml version="1.0" encoding="utf-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:app="http://www.w3.org/2007/app">
    <generator uri="http://www.ibm.com/cics/" version="6.6.0">
    CICS Transaction Server Version 4.1.0
    </generator>
    <link rel="self" href="http://www.mycics.com:80/web20/entry/10"/>
    <link rel="edit" href="http://www.mycics.com:80/web20/entry/10"/>
    <id>tag:http://www.mycics.com/web20/myfeed,2009-01-20:tsqueue:WB20TSQ:10</id>
    <title>This is my updated entry</title>
    <summary>
    This is my new updated entry
    </summary>
    <category term="Test Feeds"/>
    <rights>Copyright (c) 2009, Joe Bloggs</rights>
    <published>2009-04-23T15:00:44+00:00</published>
    <author>
     <name>Joe Bloggs</name>
     <email>JBloggs@example.com</email>
     <uri>http://www.example.com/JBloggs/</uri>
    </author>
    <app:edited>2009-04-23T15:00:44+00:00</app:edited>
    <updated>2009-04-23T15:00:44+00:00</updated>
    <content type="text/xml">
     <SAMPBIND xmlns="http://www.ibm.com/xmlns/prod/cics/atom/bindfile/sampbind">
      <data_field>
       Here is new content for my updated entry
      </data_field>
     </SAMPBIND >
    </content>
</entry>
```

向 Atom 集合发出 DELETE 请求

Web 客户机可以通过向 Atom 条目的 URL 发出 HTTP DELETE 请求（在该条目的 <atom:link rel=″edit″> 元素中指定），从集合中删除现有的 Atom 条目。

关于此任务

使用针对个别 Atom 条目的 URL，您一次只能从集合中删除一个 Atom 条目。您无法一次删除整个集合。CICS 会拒绝针对集合的 URL 发出的 DELETE 请求。

1. HTTP DELETE 请求以包含 DELETE 方法的请求行开始，接着是 Atom 条目的 URL 的路径部分，最后是 `HTTP/1.1`，即该请求的 HTTP 版本。 您还可以在 URL 中包含方案（HTTP 或 HTTPS）和主机名。 有关请求行的解释，请参阅 第 12 页的『HTTP 请求』。
2. 请如下所示为请求编写 HTTP 头，每个另起一行:
   - Host 头，用于通过 Atom 条目的 URL 提供主机名 – 如果尚未在请求行中包含主机名。
   - Authorization 头，包含访问集合所需的任何安全信息，例如，用于基本认证的用户标识和密码。

   在上一个 HTTP 头之后再输入一个回车换行符（CRLF），以给出一个空行。 请勿包含任何消息体。
3. 向服务器发送请求。

结果

当接受您的请求时，该服务器会发送 HTTP 响应，状态码为 200，表明请求已成功完成或适当的错误响应。该响应不包含已删除条目的副本。

示例

以下是要求删除具有 URL `http://www.mycics.com:80/web20/entry/10` 的 Atom 条目的请求:

```
DELETE /web20/entry/10 HTTP/1.1
Host: www.mycics.com:80
```

如何在服务例程中处理 Atom 集合编辑请求

在根据服务例程提供的现有 Atom 订阅源创建集合时，请更新程序以针对集合中条目的 POST、PUT、DELETE 和 GET 请求执行适当的操作。

CICS 与服务例程之间的接口处理集合的方式基本上与处理普通 Atom 订阅源的方式相同。CICS 为 DFHATOMPARMS 容器中的程序提供关于客户机请求的信息。您的程序必须分析该请求，然后在需要时将 DFHATOMPARMS 容器和其他字符容器返回至 CICS 以进行响应。样本服务例程 DFH$W2S1 演示了操作方法。

程序可能需要通过修改用于保存该订阅源的 Atom 条目数据的资源中（数据库、文件或其他资源）的记录，对客户机请求执行操作，这是针对集合所涉及的额外任务。程序可能需要按如下方式响应，而不是简单返回从资源获取的单个 Atom 条目的数据:

- 对于 POST 请求，添加新记录并在记录中填入针对客户机所提供新 Atom 条目的数据。
- 对于 PUT 请求，根据客户机请求的更改编辑现有记录。
- 对于 GET 请求，删除记录。

根据请求方法不同，程序的响应可能只是一个用于确认所请求更改的 HTTP 状态行，而不含其他数据。服务例程所做的更改将永久修改资源内容，因此，必须确保实施适当的安全措施。

对于使用 POST 和 PUT 方法的客户机请求，CICS 在名为 DFHREQUEST 的容器中提供请求主体。客户机应该发送包含完整 Atom 条目文档的请求主体。您的程序必须对 Atom 条目文档的标记进行语法分析，将 Atom 条目中的元素与资源中记录内的字段匹配，并使用元素的内容来创建新的资源记录或更新现有资源记录。该程序还必须提供某些元素的内容，例如，日期和时间戳记。

Atom 发布协议（RFC 5023）允许服务器可以足够自由地修改客户机提交的条目的元数据或内容。如果生成的条目满足 Atom 的格式要求，并且服务例程忽略或覆盖客户机请求中的元素内容对于资源记录而言是最佳方法，那么服务例程通常会执行这样的操作。如果希望对服务例程编码以忽略或覆盖特定元素，那么对该元素执行查询之前，请参阅 RFC 4287 和 RFC 5023 以检查提议的操作是否可接受，然后继续。

当服务例程收到与集合相关的客户机请求时，它必须执行本部分所描述的任务。这些指示信息假定您已编写了一个服务例程，用于响应 Web 客户机对 Atom 订阅源的 HTTP GET 请求，如第 237 页的『编写提供 Atom 条目数据的程序』中所述，样本服务例程 DFH$W2S1 提供了演示。

在对服务例程进行编码以执行这些任务时，如果在 CICS 区域中启用了资源和命令安全性，并且集合中正在使用这些安全性，请确保 Web 客户机的用户标识具备正确的权限来访问服务例程使用的 CICS 资源和命令。有关保护 Atom 集合的安全措施的更多信息，请参阅第 317 页的第 26 章，『Atom 订阅源的安全性』。

处理针对 Atom 集合的 GET 请求

当 Web 客户机发出针对 Atom 集合的 GET 请求时，服务例程必须用集合中的一个条目或一组条目来进行响应。

1. 使用 EXEC CICS GET CONTAINER 命令检索 DFHATOMPARMS 容器中包含请求相关信息的数据。 样本服务例程 DFH$W2S1 显示了如何执行该操作。 第 240 页的『DFHATOMPARMS 容器』描述 CICS 在该容器中传递的所有参数。
2. 检查 **ATMP_HTTPMETH** 参数的值以确定请求方法。 CICS 返回错误或对除 GET、POST、PUT 和 DELETE 以外的方法做出适当的响应。
3. 使用 DFHATOMPARMS 容器中的 **ATMP_ATOMTYPE** 和 **ATMP_SELECTOR** 参数的值来识别在该实例中程序必须返回至 CICS 的 Atom 条目。
   a. 如果 **ATMP_SELECTOR** 为 NULL 且 **ATMP_ATOMTYPE** 包含值“collection”，那么表示客户机未指定特定的 Atom 条目。如果可以，请返回集合中最近被编辑的 Atom 条目。 如果您的资源不能存储该数据，那么将返回最近添加到集合中的 Atom 条目。
   b. 如果 **ATMP_SELECTOR** 包含选择器值且 **ATMP_ATOMTYPE** 包含值“collection”，那么返回选择器值确定的条目。 这些值的组合表明客户机正在从集合请求第二个或下一个条目。
   c. 如果 **ATMP_SELECTOR** 包含选择器值且 **ATMP_ATOMTYPE** 包含值“entry”，那么返回选择器值确定的条目。 这些值的组合表明客户机正在从集合请求单个已知的 Atom 条目。

4. 在您确定了 GET 请求所需的 Atom 条目后，请像对待普通 Atom 订阅源一样返回集合中的该条目，并执行以下附加步骤:

   a. 使用 DFHATOMPARMS 容器中的 **ATMP_EDITED** 参数来返回上一次编辑该 Atom 条目的日期和时间。 如果您正在使用 DFHATOMPARMS 容器中的资源处理参数，那么 **ATMP_EDITED_FLD** 参数将包含资源中相关字段的名称和长度。如果您的资源没有存储该数据，那么将返回空格，并且 CICS 会采用当前日期和时间。 第 240 页的『DFHATOMPARMS 容器』记录了该字段所需的格式。

   b. 使用 DFHATOMPARMS 容器中的 **ATMP_ETAGVAL** 参数来返回 Atom 条目的实体标记，后跟该标记的长度。 要创建实体标记，您可以使用 EXEC CICS BIF DIGEST 命令来计算资源记录的 SHA-1 摘要，或使用其他适合的方法生成符合 HTTP/1.1 协议要求的实体标记。

   c. 如果 **ATMP_ATOMTYPE** 包含值“collection”（这表示客户机需要多个条目），并且您的资源存储了上一次编辑这些条目的时间数据，那么请返回 **ATMP_NEXTSEL** 参数作为集合中下一个 Atom 条目的选择器值。 第 224 页的『Atom 条目的顺序』说明了返回 Atom 条目的顺序。为了符合 Atom 发布协议的要求，请按编辑顺序返回这些条目。

   d. 如果 **ATMP_ATOMTYPE** 包含值“collection”，那么返回 **ATMP_PREVSEL**、**ATMP_FIRSTSEL** 和 **ATMP_LASTSEL** 参数，以作为集合中上一个、第一个和最后一个 Atom 条目的选择器值。 CICS 使用这些值构造 <atom:link> 元素，该元素包含指向集合中其他部分的条目列表的链接。 有关这些参数的更多信息，请参阅第 240 页的『DFHATOMPARMS 容器』。

   第 237 页的『编写提供 Atom 条目数据的程序』中列出了返回普通 Atom 订阅源条目的步骤。

处理针对 Atom 集合的 POST 请求

当 Web 客户机发出针对 Atom 集合的 POST 请求时，服务例程必须在集合中创建新的条目并将该条目的副本返回给客户机。

关于此任务

Web 客户机在 HTTP POST 请求的主体中提供完整的 Atom 条目文档，然后 CICS 会将该请求主体传递给 DFHREQUEST 容器中的服务例程。

如果保存 Atom 条目数据的资源具有关联的 XMLTRANSFORM 资源的 XML 绑定文件，那么您可以使用 CICS 功能将 XML 转换为应用程序数据，以对 Web 客户机提供的 Atom 条目文档的标记进行语法分析。如果 XMLTRANSFORM 资源可用，那么 CICS 会在 DFHATOMPARMS 容器的 **ATMP_XMLTRANSFORM** 参数中提供该资源的名称。有关 TRANSFORM XMLTODATA 命令的更多信息以及使用数据映射功能的指示信息，请参阅 *CICS Application Programming Guide*。

如果保存 Atom 条目数据的资源不具有 XML 绑定文件，那么可根据您服务例程的语言，使用其他设施:

- 如果该程序是采用 C 语言或汇编语言编写的，那么可以使用 z/OS XML System Services 解析器来对条目标记进行语法分析。
- 如果该程序是采用 Enterprise COBOL 语言编写的，那么可以像 DFH0W2F1 COBOL 样本服务例程一样，使用 XML PARSE 动词。
- 如果该程序是采用 Enterprise PL/I 编写的，那么请使用 PLISAXA 库函数。

1. 使用 EXEC CICS GET CONTAINER 命令检索 DFHATOMPARMS 容器中包含请求相关信息的数据。 样本服务例程 DFH$W2S1 显示了如何执行该操作。 第 240 页的『DFHATOMPARMS 容器』描述 CICS 在该容器中传递的所有参数。
2. 检查 **ATMP_HTTPMETH** 参数的值以确定请求方法。 CICS 返回错误或对除 GET、POST、PUT 和 DELETE 以外的方法做出适当的响应。
3. 使用 EXEC CICS GET CONTAINER 命令来检索 DFHREQUEST 容器（其中包含新的 Atom 条目）中的数据。 样本服务例程 DFH$W2S1 显示了如何执行该操作。 CICS 不支持 Atom 订阅源中的其他介质类型，并以 415 状态码拒绝符合以下条件的任何客户机请求: 未描述为 Atom 条目、介质类型为 application/atom+xml、带或不带 type=entry 参数的请求。
4. 使用将 XML 转换为应用程序数据的 CICS 功能或其他合适的设施来对 Atom 条目的 XML 标记进行语法分析，以标识 CICS 提供支持的 Atom 条目的所有元素以及保存 Atom 条目数据的资源（具有用于存储数据的合适字段）。 有关受 CICS 支持的 Atom 条目的所有元素的列表和描述，请参阅第 277 页的『<atom:entry> 元素』。忽略 CICS 不支持、您的服务例程无法识别或您无法在资源（保存 Atom 条目数据）记录的字段中存储的任何元素。您的 Atom 配置文件应该已经进行如下设置: 对于您的资源无法存储其数据的任何必需元素，为其提供合适的缺省值。

   提示: 如果您发现 Web 客户机提交了包含无效 XML 标记或数据的请求，请用响应代码 `atmp_resp_invalid_request` 拒绝该请求。
5. 在保存 Atom 条目数据的资源中创建新记录，并用已识别的元素的内容填充该记录的字段。 根据客户机的特性和保存 Atom 条目数据的资源的灵敏度，服务例程可以包含您认为合适的任何级别的错误检查。 如果必须覆盖客户机提供的任何数据，那么在执行此操作后，您需要根据 RFC 4287 和 RFC 5023 来检查此操作，以确保覆盖有效。 CICS 会忽略由 Web 客户机提供的 Atom 标识，您的服务例程也应忽略该标识。替换 Atom 标识，该标识是通过完成由 CICS 在 DFHATOMPARMS 容器的 **ATMP_ATOMID** 参数中传递的原型 Atom 标识而生成的；或使用另一种有效的格式。要了解有关 Atom 标识格式的更多信息，请参阅 第 226 页的『Atom 条目的 Atom 标识』
6. 如果您的资源存储了上一次编辑条目的时间（<app:edited> 元素）、上一次更新条目的时间（<atom:updated> 元素）或第一次发布条目的时间（<atom:published> 元素），那么使用 EXEC CICS ASKTIME 和 EXEC CICS FORMATTIME 命令可以针对当前日期和时间生成 RFC 3339 格式的日期和时间戳记，并使用该时间戳记来填充这些字段。 如果客户机提供 <atom:updated>、<atom:published> 或 <app:edited> 元素中的日期和时间戳记，那么您可能希望生成新的日期和时间戳记来确保精确性和有效性。 有关处理日期和时间戳记的更多信息，请参阅第 225 页的『Atom 条目的日期和时间戳记』。
7. 如果客户机请求成功，请像对待普通 Atom 订阅源一样返回新条目，这会提供刚刚添加到新资源记录中的数据，但是在以下方面有别于正常过程:
   a. 请勿返回 **ATMP_NEXTSEL** 参数。
   b. 使用 EXEC CICS BIF DIGEST 命令来计算新资源记录的 SHA-1 摘要，或使用其他适合的方法生成符合 HTTP/1.1 协议要求的实体标记。使用 DFHATOMPARMS 容器中的 **ATMP_ETAGVAL** 参数将结果作为 Atom 条目的实体标记返回，后跟该标记的长度。

c. 对于 **ATMP_EDITED** 参数，您可以返回存储在资源中的日期和时间戳记，或者返回空格以允许 CICS 提供当前日期和时间（前提是您没有在 Atom 配置文件中指定备用缺省值）。

样本服务例程 DFH$W2S1 显示了如何将 Atom 条目返回给 CICS。 CICS 创建 HTTP 状态码为 201 的响应，并提供 Location 头来向客户机提供新 Atom 条目的 URI，以及匹配的 Content-Location 头，以便客户机了解该响应是一个完整的 Atom 条目表示。客户机可以检查消息主体中的条目，以查看对其请求中提供的数据所进行的任何修改。

8. 如果客户机请求不成功，那么将使用 DFHATOMPARMS 容器中的 **ATMP_RESPONSE** 参数来返回适当的响应代码。 在返回错误响应时，CICS 将生成适当的缺省 HTTP 错误响应以发送至 Web 客户机。第 240 页的『DFHATOMPARMS 容器』列出可以使用的响应代码以及各种情况下 CICS 发送的 HTTP 错误响应。

处理针对 Atom 集合的 PUT 请求

当 Web 客户机发出针对 Atom 集合的 PUT 请求时，服务例程必须更新该条目并指明请求是否成功。

关于此任务

与 POST 请求一样，Web 客户机在 HTTP PUT 请求的主体中提供完整的 Atom 条目文档，然后 CICS 会将该请求主体传递给 DFHREQUEST 容器中的服务例程。如果保存 Atom 条目数据的资源具有关联的 XMLTRANSFORM 资源的 XML 绑定文件，那么您可以使用 CICS 功能将 XML 转换为应用程序数据，以对 Web 客户机提供的 Atom 条目文档的标记进行语法分析。如果保存 Atom 条目数据的资源不具有 XML 绑定文件，那么您可以使用第 309 页的『处理针对 Atom 集合的 POST 请求』中列出的某个其他设施来对 Atom 条目文档的标记进行语法分析。

1. 使用 EXEC CICS GET CONTAINER 命令检索 DFHATOMPARMS 容器中包含请求相关信息的数据。 样本服务例程 DFH$W2S1 显示了如何执行该操作。 第 240 页的『DFHATOMPARMS 容器』描述 CICS 在该容器中传递的所有参数。
2. 检查 **ATMP_HTTPMETH** 参数的值以确定请求方法。 CICS 返回错误或对除 GET、POST、PUT 和 DELETE 以外的方法做出适当的响应。
3. 使用 EXEC CICS GET CONTAINER 命令来检索 DFHREQUEST 容器中的数据，其中包含已更新的 Atom 条目。 样本服务例程 DFH$W2S1 显示了如何执行该操作。
4. 使用 **ATMP_SELECTOR** 参数在保存 Atom 条目数据的资源中选择 Atom 条目（客户机希望更新的）数据的记录。
5. 使用 EXEC CICS BIF DIGEST 命令来计算包含 Atom 条目数据的资源中当前记录的 SHA-1 摘要，或使用您的服务例程备用方法来计算实体标记，然后将其与 **ATMP_ETAGVAL** 参数中提供的实体标记进行比较。 如果这两个标记不匹配，表明自 Web 客户机获取该 Atom 条目的副本后，该条目的数据已由其他代理程序更改，请用响应代码 `atmp_resp_etag_no_match` 拒绝该请求。 如果实体标记为星号，那么表示 Web 客户机已选择要覆盖此过程，并且您应该接受这个请求。
6. 使用将 XML 转换为应用程序数据的 CICS 功能或其他合适的设施来对 Atom 条目的 XML 标记进行语法分析，以标识 CICS 提供支持的 Atom 条目的所有元素以及保存 Atom 条目数据的资源（具有用于存储数据的合适字段）。 有关受 CICS

支持的 Atom 条目的所有元素的列表和描述，请参阅第 277 页的『<atom:entry> 元素』。 忽略 CICS 不支持、您的服务例程无法识别或您无法在资源（保存 Atom 条目数据）记录的字段中存储的任何元素。您的 Atom 配置文件应该已经进行如下设置: 对于您的资源无法存储其数据的任何必需元素，为其提供合适的缺省值。

提示: 如果您发现 Web 客户机提交了包含无效 XML 标记或数据的请求，请用响应代码 `atmp_resp_invalid_request` 拒绝该请求。

7. 使用已识别的元素内容对记录中的字段进行修改，以更新包含 Atom 条目数据的资源中的记录。 在 PUT 请求的主体中，客户机需要提供完整的 Atom 条目，包括已更改和未更改的元素，这样，在理论上您可以更新资源记录中的所有字段，而无需比较它们来查看更改了哪些元素。但是，根据客户机的特性和资源的灵敏度，您可能希望对具有特定内容格式要求的任何字段进行错误检查，并检查客户机是否未更改任何逻辑上不能更改的字段的内容，例如，ATOM 标识或第一次发布的时间。 如果必须覆盖客户机提供的任何数据，那么在执行此操作后，您需要根据 RFC 4287 和 RFC 5023 来检查此操作，以确保覆盖有效。
8. 如果您的资源存储了上一次编辑条目的时间（<app:edited> 元素）和上一次更新条目的时间（<atom:updated> 元素），那么使用 EXEC CICS ASKTIME 和 EXEC CICS FORMATTIME 命令可以针对当前日期和时间生成 RFC 3339 格式的日期和时间戳记，并使用该时间戳记来填充这些字段。 如果客户机提供 <atom:updated> 或 <app:edited> 元素中的日期和时间戳记，请忽略这些项，因为它们可能只是前一个返回的日期和时间戳记，且未经过更改。 有关处理日期和时间戳记的更多信息，请参阅第 225 页的『Atom 条目的日期和时间戳记』。
9. 如果客户机请求成功，请像处理普通 Atom 订阅源一样返回更新的条目，从而提供已更新资源记录中的数据，但是以下方面有别于正常过程:
   a. 请勿返回 **ATMP_NEXTSEL** 参数。
   b. 使用 EXEC CICS BIF DIGEST 命令来计算更新的资源记录的 SHA-1 摘要，或使用其他适合的方法生成符合 HTTP/1.1 协议要求的实体标记。使用 DFHATOMPARMS 容器中的 **ATMP_ETAGVAL** 参数将结果作为 Atom 条目的新实体标记返回，后跟该标记的长度。
   c. 对于 **ATMP_EDITED** 参数，您可以返回存储在资源中的日期和时间戳记，或者返回空格以允许 CICS 提供当前日期和时间（前提是您没有在 Atom 配置文件中指定备用缺省值）。

   样本服务例程 DFH$W2S1 显示了如何将 Atom 条目返回给 CICS。 CICS 创建了 HTTP 状态码为 200 的响应，表明请求成功完成。客户机可以检查消息主体中的条目，以查看对其请求中提供的数据所进行的任何修改。
10. 如果客户机请求不成功，那么将使用 DFHATOMPARMS 容器中的 **ATMP_RESPONSE** 参数来返回适当的响应代码。 在返回错误响应时，CICS 将生成适当的缺省 HTTP 错误响应以发送至 Web 客户机。第 240 页的『DFHATOMPARMS 容器』列出可以使用的响应代码以及各种情况下 CICS 发送的 HTTP 错误响应。

处理针对 Atom 集合的 DELETE 请求

当 Web 客户机发出针对 Atom 集合的 DELETE 请求时，服务例程必须删除该条目并指明请求是否成功。

1. 使用 EXEC CICS GET CONTAINER 命令检索 DFHATOMPARMS 容器中包含请求相关信息的数据。 样本服务例程 DFH$W2S1 显示了如何执行该操作。 第 240 页的『DFHATOMPARMS 容器』描述 CICS 在该容器中传递的所有参数。
2. 检查 **ATMP_HTTPMETH** 参数的值以确定请求方法。 CICS 返回错误或对除 GET、POST、PUT 和 DELETE 以外的方法做出适当的响应。
3. 使用 **ATMP_SELECTOR** 参数在保存 Atom 条目数据的资源中选择包含 Atom 条目（客户机希望删除的）数据的记录。
4. 删除资源中与 **ATMP_SELECTOR** 参数的选择器值对应的记录，并使用 DFHATOMPARMS 容器中的 **ATMP_RESPONSE** 参数来返回响应代码 0。 CICS 将忽略 DFHATOMPARMS 容器中的其他参数，因此您无需对它们进行任何更改。 CICS 将状态码为 200 的响应发送至客户机，表明请求成功完成。
5. 如果不执行该请求，请返回 **ATMP_RESPONSE** 参数中的备用响应代码。 第 240 页的『DFHATOMPARMS 容器』列出可以使用的响应代码以及各种情况下 CICS 发送的 HTTP 错误响应。

Atom 订阅源的 DFH0W2F1 COBOL 样本服务例程

样本服务例程 DFH0W2F1 是一个 COBOL 程序，用于处理针对使用 CICS 样本文件 FILEA 数据的 Atom 条目的 GET、POST、PUT 和 DELETE 请求。您可以将这些交互作为一个模型，以便在您自己的服务例程中处理资源。

您可以运行 DFH0W2F1 来处理 Web 客户机请求，并提供 FILEA 样本文件中的数据，以用于测试或演示。在运行 DFH0W2F1 之前，必须完成下列任务:

1. 为必须指定 EXECKEY(CICS) 的 DFH0W2F1 创建并安装适当的 RD0 定义。
2. 创建 FILEA VSAM 文件，并为其安装适当的 RDO 定义。在 CICS 提供的 RDO 样本组 DFH$FILA 中提供了适当的 RDO 定义。
3. 确保已启用并打开 FILEA 文件。

CICS 在可用于运行 DFH0W2F1 的 DFH$WEB2 组中提供样本 URIMAP 和 ATOMSERVICE 资源。这两个资源都命名为 DFH$W2P1。您可以按照 CICS TS for z/OS V4.1 信息中心（`https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp`）内的 Web 2.0 案例“创建人员位置的 Atom 订阅源”中的指示信息，以设置样本 Atom 集合的相同方式设置这些资源。要求通过 DFH0W2F1 访问作为 Atom 集合的 FILEA 样本的 Web 客户机请求的 URL 为:

`http://`*`host`*`:`*`port`*`/atom/p/filea/`

，其中:

- *host* 是在您用于该集合的 TCPIPSERVICE 资源的 HOST 属性中定义的字符主机名、IPv4 地址或 IPv6 地址。
- *port* 是在您用于该集合的 TCPIPSERVICE 资源的 PORTNUMBER 属性中定义的端口号。

ATOMSERVICE 资源 DFH$W2P1 指定 Atom 配置文件 dfh0w2f1.xml，该文件提供 Atom 订阅源文档及 Atom 条目的元数据和结构。dfh0w2f1.xml 位于 z/OS UNIX 中的 CICS 文件根目录（由 CICS 系统初始化参数 USSHOME 指定）的 /samples/web2.0/ 子目录中。 USSHOME 的缺省值是 /usr/lpp/cicsts/cicsts41。ATOMSERVICE 资源

DFH$W2P1 不为 FILEA 文件指定 XML 绑定，由于 FILEA 不包含可作为 Atom 条目元数据的任何字段，因此 DFH0W2F1 不会使用 Atom 配置文件中的 <cics:fieldnames> 元素。

DFH0W2F1 样本程序执行以下任务:

工作存储节

- 复制具有响应代码常量的副本 DFHW2CNO。
- 设置工作存储中的局部变量。
- 包含 FILEA 记录布局的定义，该定义在 CICS 随附的副本 DFH0CFIL 中描述。
- 设置包含 XML 标记和字段的原型，以保存 Atom 条目的内容。每个 XML 元素之间都使用回车符换行（X'0D25'）。通过读取 FILEA 文件中的记录、将该记录中字段的数据插入该原型并在 DFHATOMCONTENT 容器中返回，该服务例程将生成 Atom 条目的内容。
- 设置将用于 DFHATOMTITLE 和 DFHATOMSUMMARY 容器的原型 Atom 条目标题和概要。尽管 FILEA 文件中的记录不包含标题和摘要，但该服务例程将使用文件记录中的信息将它们组合起来。该服务例程提供的标题和摘要将替换 Atom 配置文件中指定的缺省标题和摘要。
- 设置为 Atom 条目返回的内容，以便在无法读取 FILEA 时指示错误。该内容供演示使用，不适用于产品的 Atom 订阅源。
- 设置在对 Atom 条目进行语法分析期间使用的 XML 堆栈。
- 设置包含 Atom 条目实体标记（ETag）的存储器。

链接节

- 复制描述 DFHATOMPARMS 容器布局的副本 DFHW2APO。
- 定义可用于 DFHATOMPARMS 容器中任何参数的可变长度字符串。
- 描述包含来自 Web 客户机的请求主体（在 DFHREQUEST 容器中传递）的存储器。
- 定义存储器以包含文件中第一条、最后一条、下一条和上一条记录的选择器值，CICS 使用这些选择器值为集合创建其他导航链接。

主程序

- 使用 EXEC CICS GET CONTAINER 命令获取 DFHATOMPARMS 容器中的参数。
- 获取 ATMP_SELECTOR 参数的选择器值。该选择器值是 FILEA 中记录的键，该记录包含请求的 Atom 条目的数据。在 FILEA 中，记录的键是唯一客户编号。FILEA 的键必须正好为 6 位，并且 DFH0W2F1 不会将键值填充至正确长度。如果 CICS 未提供选择器值，那么服务例程将读取文件中的第一条记录，并将该记录的键复制到 ATMP_SELECTOR 参数存储器中。该服务例程将按照键的升序返回这些记录（与 CICS 处理 KSDS 文件的方式相同）。
- 如果提供了实体标记值，那么将通过 If-Match 头保存实体标记（Etag）值。
- 获取请求的 HTTP 方法。该服务例程使用 HTTP 方法来确定后续操作:
  - 对于 GET 请求，该服务例程将读取键值与选择器值相等的记录，并使用 RETURN-FILE-CONTENT 过程将其内容作为 Atom 条目返回。
  - 对于 PUT 请求，该服务例程将通过使用更新功能的读操作，读取键值与选择器值相等的记录。它会计算刚读取的记录的 ETag 值，然后将该值与从 If-Match 头收到的值进行比较。如果这两个值不相等，那么该文件将从读操作中解锁以进行更

新，并且该服务会返回一个错误响应。如果 ETag 匹配，那么服务例程会使用 PARSE-REQUEST-BODY 过程来对 Web 客户机提供的已更新 Atom 条目进行语法分析，并使用这些数据更新 FILEA 中的记录。该文件记录使用 REWRITE 函数重新编写。然后，该服务例程会使用 RETURN-FILE-CONTENT 过程将已更新的 Atom 条目的副本返回给 Web 客户机。

- 对于 POST 请求，该服务例程会使用 PARSE-REQUEST-BODY 过程对 Web 客户机提供的已更新 Atom 条目进行语法分析，并使用这些数据撰写新的文件记录。Web 客户机在 POST 请求中提供的客户编号将成为新文件记录的键值，并且如果已存在带有该编号的记录，那么该服务例程会返回一个错误。然后，该服务例程会使用 RETURN-FILE-CONTENT 过程将新的 Atom 条目的副本返回给 Web 客户机。
- 对于 DELETE 请求，将删除键值与选择器值相等的记录。

PARSE-REQUEST-BODY 和 XML-HANDLER 过程

在 PARSE-REQUEST-BODY 过程中，该服务例程使用 EXEC CICS GET CONTAINER 命令来获取 DFHREQUEST 容器的内容，该内容是 Web 客户机在 POST 或 PUT 请求的主体中提供的 Atom 条目。然后，该命令会使用 XML PARSE 函数来对此 Atom 条目进行语法分析。

XML-HANDLER 过程将对所提供的 Atom 条目的元素进行分析，使用 XML 堆栈来跟踪是否每个 XML 元素都在提供的 Atom 条目的 <atom:content> 元素中（“IN-CONTENT”标志），并记录每个元素的嵌套级。当发现元素状态为“IN-CONTENT”并具有可识别的名称时，UPDATE-RECORD-FIELD 过程会将其数据保存到 FILEA 记录中。该服务例程会忽略 Atom 条目中 <atom:content> 元素以外的元数据元素，以及 <atom:content> 元素中名称无法识别的任何元素（因为该服务例程无法将这些元素中的数据存储到 FILEA 记录中）。

COBOL XML 解析器不会试图严格分析 XML 名称空间前缀。如果您正在使用该解析器，那么在提供的 Atom 条目中的 <atom:content> 元素必须包含前缀“atom”，而该内容中的任何 XML 元素不得包含名称空间前缀。

RETURN-FILE-CONTENT 过程

对于 GET、POST 和 PUT 请求，RETURN-FILE-CONTENT 过程会从 FILEA 文件的记录中返回数据，以作为 Atom 条目的内容，并创建和返回 Atom 条目的标题和摘要。该过程执行以下步骤:

- 将 FILEA 记录的字段中的数据放在工作存储节中设置的原型 Atom 条目内容中。通过使用 EXEC CICS PUT CONTAINER 命令，将结果 XML 结构放入 DFHATOMCONTENT 容器中。
- 完成原型 Atom 条目标题，并将它保存在 DFHATOMTITLE 容器中。将为 ATMP-OPTIONS-OUT 参数设置用于指示这一点的标志。
- 为 Atom 条目创建摘要，并将它保存在 DFHATOMSUMMARY 容器中。将为 ATMP-OPTIONS-OUT 参数设置用于指示这一点的标志。
- 使用 EXEC CICS BIF DIGEST 命令为 FILEA 记录生成 ETag，并将该值返回至 DFHATOMPARMS 容器的 ATMP-ETAGVAL 参数中。
- 调用

ADD-NAVIGATION-LINKS 过程

ADD-NAVIGATION-LINKS 过程返回 FILEA 文件中第一条、最后一条、下一条和上一条记录的选择器值（即，键值）。这些选择器值保存在临时工作区（TWA）中。CICS 将使用这些选择器值为 Atom 集合文档创建其他导航。

第 26 章 Atom 订阅源的安全性

CICS Web Support 提供适当的安全性协议和认证方法来控制 Web 客户机对 Atom 集合或 Atom 订阅源（需要时）的访问。您可以使用 CICS 资源和命令安全性来保护用于交付 Atom 订阅源或集合的资源。

RFC 5023 建议您使用认证来保护 Atom 集合。在您将 Atom 订阅源数据用作可编辑集合时，Web 客户机可以插入新条目、修改现有条目或删除条目。因此，您必须确保已验证 Web 客户机的身份并且仅允许可信的客户机访问集合，特别是在集合中包含了业务数据时。尽管在 Atom 订阅源包含机密业务数据或仅供某些用户使用时，您可能需要限制对它们的访问，但是，Web 客户机无法编辑的普通 Atom 订阅源通常会无任何安全限制地供所有订户使用。

RFC 4287 和 5023 探讨了如何针对 Atom 文档使用数字签名和加密。CICS 不支持对 Atom 文档的数字签名和加密，但是按照 RFC 4287 的要求，CICS 不会拒绝包含签名的 Atom 文档。

CICS Web Support 具有以下安全性功能，可用于防止对 Atom 订阅源或集合进行未经授权的访问或更新:

SSL 或 TLS 安全性协议
RFC 5023 建议使用传输层安全性（TLS）1.0 作为集合的最低级别安全性协议。CICS 支持安全套接字层（SSL）3.0 协议和传输层安全性（TLS）1.0 协议。*CICS RACF Security Guide* 说明了 SSL 和 TLS 提供的设施。

HTTP 基本认证
RFC 5023 建议使用 HTTP 基本认证作为集合的最低级别认证。第 18 页的『HTTP 基本认证』中介绍了该机制。

客户机证书认证
客户机证书认证是更加安全的客户机认证方法，它使用由可信第三方（或认证中心）发布的客户机证书，并使用 SSL 加密进行发送。*CICS RACF Security Guide* 介绍了其工作原理。

当您设置 CICS Web Support 中的这些功能时，通过使用端口（CICS 通过该端口接收针对 Atom 订阅源或集合的 Web 客户机请求）的 TCPIPSERVICE 定义的属性，可将这些功能应用于 Atom 订阅源或集合。有关设置 CICS Web Support 的 SSL 支持的信息，请参阅*CICS RACF Security Guide*。有关设置 CICS Web Support 的基本认证或客户机证书认证的信息，请参阅第 151 页的『作为 HTTP 服务器的 CICS: 认证和标识』。

Atom 订阅源的资源和命令安全性

CICS 资源安全性和命令安全性防止对为 Atom 订阅源提供数据的 CICS 资源（包括文件、瞬时数据队列和 ATOMSERVICE 资源定义）进行未经授权的访问；同时还防止通过用于交付 Atom 订阅源的服务例程执行未经授权的系统命令。您可以允许某些客户机对订阅源或集合进行只读访问（仅限 HTTP GET 请求），还可以授予客户机用于更新资源的其他许可权，以便它们可以发出所有类型的 HTTP 请求。

对于资源安全性，在 RCICSRES 类或 WCICSRES 分组类，或在 XRES 系统初始化参数中指定的同等类中，将 ATOMSERVICE 资源定义指定 RACF。ATOMSERVICE 资源的命令安全性使用 CCICSCMD 类或 VCICSCMD 分组类中的 ATOMSERVICE 资源。使用针对 *CICS RACF Security Guide* 中的资源列出的类，可以将资源安全性用于交付 Atom 订阅源或集合的其他类型的资源和命令。

如果在 CICS 区域中启用了资源和命令安全性，那么要将这些功能用于 Atom 订阅源或集合，您需要使用 RESSEC(YES) 和 CMDSEC(YES) 定义其别名事务。在 Atom 订阅源或集合的 URIMAP 定义的 TRANSACTION 属性中为该订阅源或集合的别名事务命名。Atom 订阅源的缺省别名事务是 CW2A，它已使用 RESSEC(YES) 和 CMDSEC (YES) 进行定义。有关 Atom 订阅源的别名事务的更多信息，请参阅第 261 页的『为 Atom 订阅源创建别名事务』。

Atom 订阅源的资源和命令安全性与应用程序通过 CICS Web Support 生成的响应的资源和命令安全性工作方式相同。在为别名事务指定资源和命令安全性时，必须为别名事务的所有可能用户标识提供相应许可权，以便使用该事务访问的资源以及使用的命令，其中包括 ATOMSERVICE 资源定义。获得 READ 访问权的用户标识可以使用 HTTP GET 请求从订阅源或集合获取条目，但是它们无权发出其他类型的请求。获得 UPDATE 访问权的用户标识可以执行 POST、PUT 和 DELETE 请求来修正资源中的数据。如果某个 Web 客户机获得了已认证的用户标识，那么该用户标识将应用于别名事务，或者您可以提供自己的标准用户标识来覆盖该标识。有关如何在安全性检查中使用 Web 客户机用户标识的说明以及为 Web 别名事务设置资源和命令安全性的指示信息，请参阅第 157 页的『应用程序生成的响应的资源和事务安全性』。

如果未授权 Web 客户机访问别名事务使用的资源，那么 CICS 会发出安全违例消息 DFHXS1111，并向客户机返回 HTTP 错误响应和状态码 403（已禁止）。

第 4 部分 CICS 业务逻辑接口

有关 CICS 业务逻辑接口的信息。

第 27 章 CICS 业务逻辑接口的介绍

CICS 业务逻辑接口使链接到支持 Web 的业务应用程序成为可能，而不是通过 CICS HTTP 侦听器调用它。

例如，z/OS 上运行的 Web 服务器可以使用外部 CICS 接口（EXCI）链接到使用 CICS 业务逻辑接口的应用程序。在该方式中，Web 客户机可以通过介体 Web 服务器与 CICS 应用程序通信，而不是直接连接到 CICS。

第 371 页的附录 G, 『DFHWBBLI CICS 业务逻辑接口的参考信息』具有该接口的参考信息。

如何使用 CICS 业务逻辑接口

您可以在可链接到 CICS 应用程序的任何环境中调用 CICS 业务逻辑接口。

例如:

- 您可以从 CICS 应用程序发出 EXEC CICS LINK 命令。
- 您可以使用外部 CICS 接口（EXCI）。
- 可以从客户机上使用外部调用接口（ECI）。
- 您可以从 ONC RPC 客户机使用 CICS ONC RPC 支持。

CICS 业务逻辑接口由以下项使用:

- CICS Web 服务器插件。该插件使用外部 CICS 接口调用 CICS 业务逻辑接口。

处理示例

CICS 业务逻辑接口如何处理来自使用 EXCI 或 ECI 的 MVS 应用程序的请求的示例。

图 19 显示了 CICS 业务逻辑接口如何处理来自使用 EXCI 的 MVS 应用程序的请求。

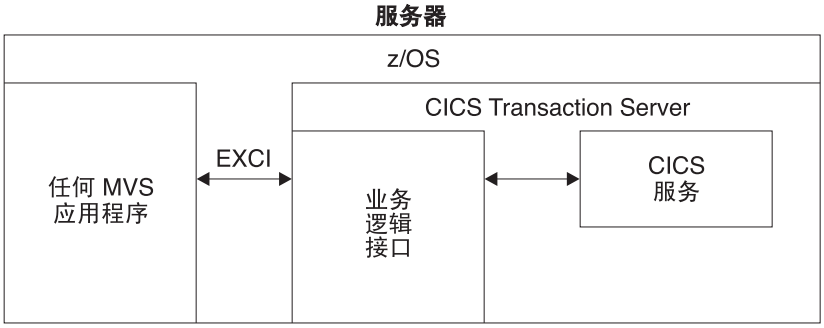


图 19. 处理 EXCI 的请求

1. MVS 应用程序构造包含 CICS 业务逻辑接口的参数的通信区域。
2. MVS 应用程序使用 EXCI 调用 CICS 业务逻辑接口。
3. CICS 业务逻辑接口调用请求的服务，并在通信区域中返回任何输出。

图 20 显示了 CICS 业务逻辑接口如何处理来自使用 ECI 的 CICS 客户机的请求。

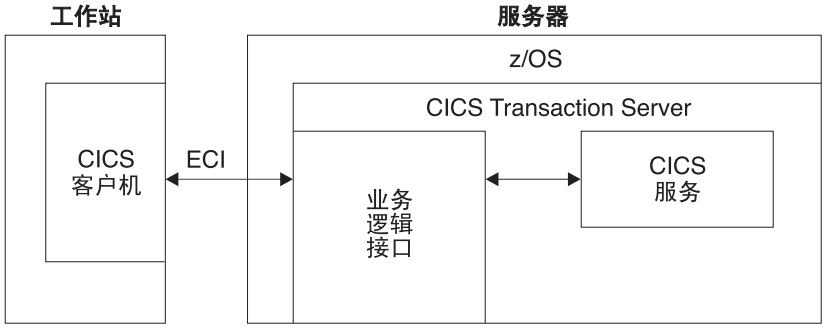


图 20. 处理 ECI 请求

1. 在工作站环境中运行的客户机构造包含 CICS 业务逻辑接口的参数的通信区域。
2. 客户机使用 ECI 调用 CICS 业务逻辑接口。
3. CICS 业务逻辑接口调用请求的服务，并在通信区域中返回任何输出。

ECI 与 SNA 协议或 TCP62 协同运行，以允许通过 TCP/IP 建立 SNA 连接（要获取更多信息，请参阅 *CICS Family: Client/Server Programming*）。

请求处理中的控制流

要决定有关您将使用的设施以及如何定制它们，需要理解 CICS 业务逻辑接口的组件是如何交互作用的。

使用 CICS 业务逻辑接口调用程序

图 21 显示了通过 CICS 业务逻辑接口到达程序的控制流。通过到 PROGRAM DFHWBBLI 的 LINK 命令来访问 CICS 业务逻辑接口。

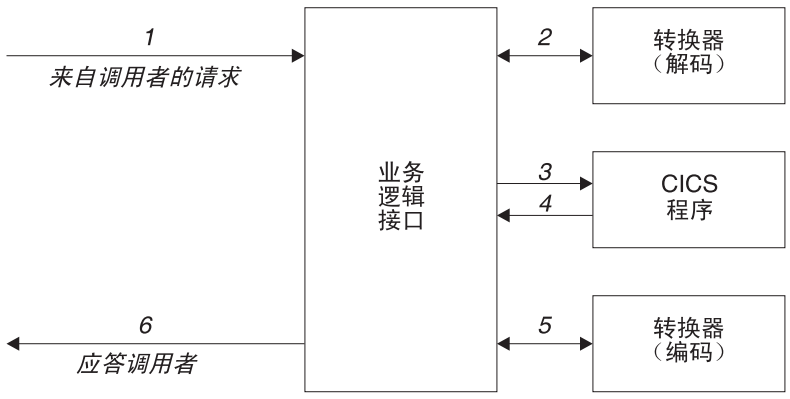


图 21. 用 CICS 业务逻辑接口调用程序 - 控制流

1. 到达 CICS 业务逻辑接口的请求。
2. 如果调用者请求转换器，那么 CICS 业务逻辑接口调用它，并请求**解码**函数。**译码**为 CICS 应用程序设置通信区域。

3. CICS 业务逻辑接口调用调用者指定的 CICS 应用程序。传递到应用程序的通信区域是由译码设置的一个区域。如果 CICS 业务逻辑接口的调用者表明不需要转换器，请求的第一个 32K 字节传递到它的通信区域中的 CICS 应用程序。
4. CICS 应用程序处理请求，并在通信区域中返回输出。
5. 如果调用者请求转换器，那么 CICS 业务逻辑接口调用转换器的**编码**函数，该函数使用通信区域来准备响应。如果没有调用转换器程序，那么 CICS 业务逻辑接口假定 CICS 应用程序已将需要的响应放在通信区域中。
6. CICS 业务逻辑接口将应答发送回调用者。

使用 CICS 业务逻辑接口运行面向终端的事务

图 22 显示了控制流，该控制流通过 CICS 业务逻辑接口到达面向终端的事务的请求。请注意，业务逻辑接口在 CICS 镜像事务下运行，而不是在 Web CICS 事务下运行。处理的第一部分与调用程序相同，但如果您要运行事务，那么必须在 **wbbl_server_program_name** 中将 DFHWBTTA 指定为要调用的 CICS 应用程序。

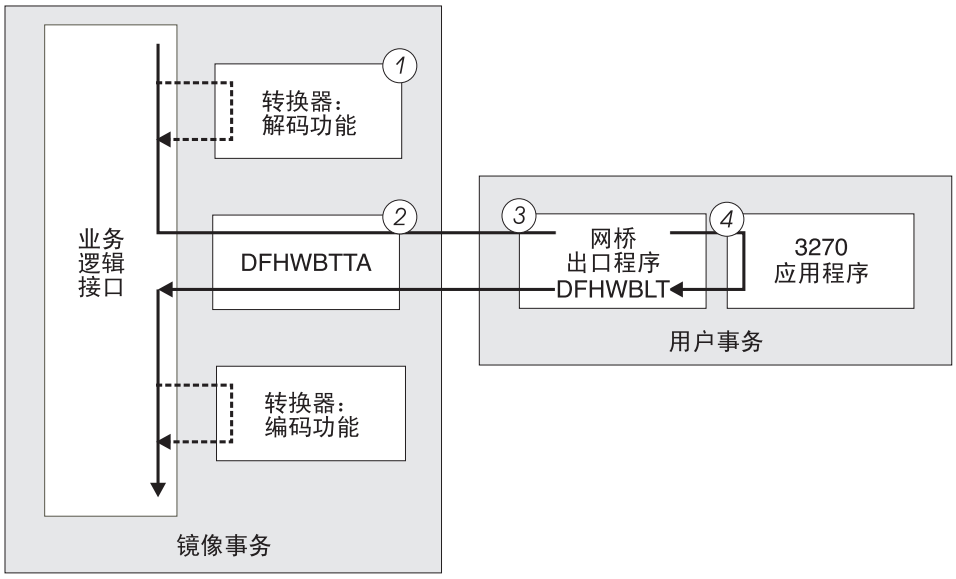


图 22. 用 CICS 业务逻辑接口运行事务 - 控制流

1. 如果调用者请求转换器，那么 CICS 业务逻辑接口调用它，并请求**解码**函数。**译码**为 DFHWBTTA 设置通信区域。
2. CICS 业务逻辑接口调用 DFHWBTTA。传递到 DFHWBTTA 的通信区域是由**译码**设置的一个区域。如果没有调用转换器程序，那么通信区域包含整个请求。
3. DFHWBTTA 从 HTTP 请求抽取面向终端的事务的事务标识，并启动运行 CICS Web 网桥出口的事务。
4. 当程序尝试写到它的主要功能时，CICS Web 网桥出口拦截这些数据。该出口构造返回到 CICS 业务逻辑接口的 HTML 响应。如果调用者请求转换器，那么 CICS 业务逻辑接口调用转换器的**编码**函数，该函数使用通信区域来准备响应。如果没有调用转换器程序，那么 CICS 业务逻辑接口假定通信区域包含需要的响应。

请求处理中的数据流

要决定有关您将使用的设施以及如何定制它们，需要理解数据是如何在 CICS 业务逻辑接口中传递的。

转换器程序和 CICS 业务逻辑接口

CICS 系统中可以存在多个转换器程序，以支持 CICS 业务逻辑接口的操作。

第 322 页的图 21 和第 323 页的图 22 中说明了转换器在 CICS 业务逻辑接口中的位置。每个转换器必须提供两个函数:

- **解码**在调用 CICS 应用程序前使用。它可以:
  - 以应用程序需要的格式，使用入站请求中的数据来构建通信区域。
  - 提供应用程序通信区域中输入和输出数据的长度。
  - 执行有关请求的管理任务。
- **编码**在已调用 CICS 应用程序后使用。它可以:
  - 使用应用程序中的数据来构建响应。
  - 执行有关响应的管理任务。

注:

- 如果改变了 DECODE_DATA_PTR 或 ENCODE_DATA_PTR 来寻址另一个存储位置，那么转换器程序负责将原来的存储器清空。
- CICS 业务逻辑接口的调用者负责释放 ENCODE_DATA_PTR 寻址的缓冲区（即，字段 WBBL_OUTDATA_PTR 中返回的地址减 4）。
- 如果转换器异常终止，那么 CICS 将尝试释放 DECODE_DATA_PTR 和 ENCODE_DATA_PTR 寻址的存储器。因此，您应该确保这些指针从不包含已释放的存储地址。

使用 CICS 业务逻辑接口调用程序

数据通过 CICS 业务逻辑接口流到程序，然后返回请求程序。

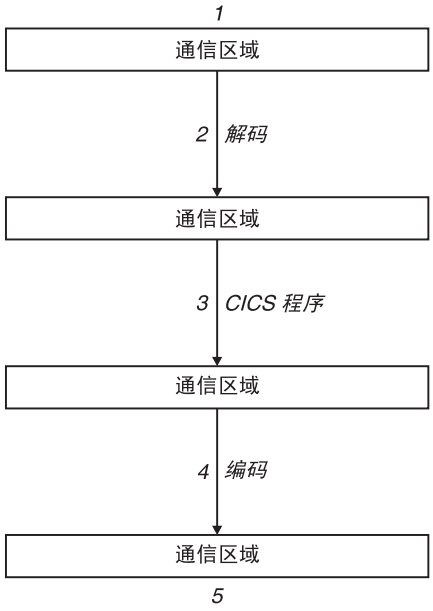


图 23. 用 CICS 业务逻辑接口调用程序 - 数据流

1. CICS 业务逻辑接口的调用者提供包含要处理的请求的通信区域。该通信区域的内容必须是后续进程可以接受的代码页。通常这意味着它们必须是 EBCDIC。
2. 如果调用者请求转换器，那么转换器的**译码**函数为 CICS 应用程序构造通信区域。
3. CICS 应用程序更新通信区域。
4. 如果调用者请求转换器，那么转换器的**编码**函数构造要返回给调用者的通信区域。
5. CICS 业务逻辑接口返回到它的调用者，现在该调用者就可以使用通信区域的内容了。

面向终端的事务的请求

第 326 页的图 24 显示了启动一个面向终端事务的请求的数据流。

图 24. 启动面向终端的事务 – 数据流

该图显示了经过 CICS 业务逻辑接口的 3270 BMS 应用程序的数据流。

1. CICS 业务逻辑接口的调用者提供包含要处理的请求的通信区域。该通信区域的内容必须是后续进程可接受的代码页，并且 DFHWBTTA 需要 EBCDIC。
2. 如果需要，可以使用转换器的**解码**函数来修改请求。
3. 由于这是对话或者伪会话的第一个事务，因此请求包括了事务标识，还可能包括事务程序可用的数据。DFHWBTTA 抽取数据，以使之可通过 RECEIVE 命令用于事务程序。
4. 事务程序使用 RECEIVE 命令接收数据。然后它构造输出映射，并使用 SEND MAP 命令将其发送给请求程序。
5. 映射及其数据内容转换为 HTML 格式。这个转换使用 DOCTEMPLATE 定义中定义的模板。
6. 如果需要，可以使用转换器的**编码**函数来修改响应。
7. CICS 业务逻辑接口返回到它的调用者，现在该调用者就可以使用通信区域的内容了。

图 25 显示了继续面向终端事务请求的数据流。

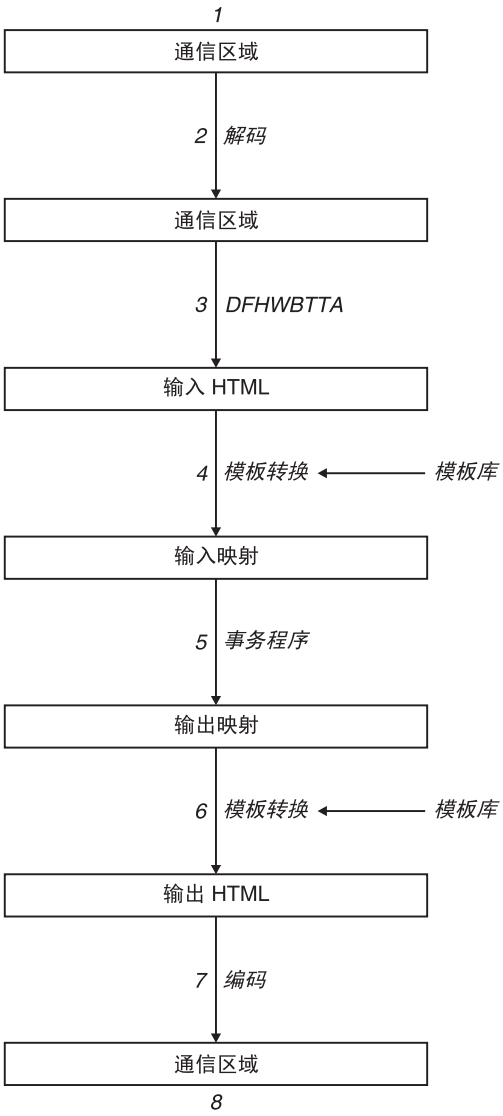


图 25. 继续面向终端的事务 - 数据流

该图显示了当 CICS 业务逻辑接口处理请求时的数据流。

1. CICS 业务逻辑接口的调用者提供包含要处理的请求的通信区域。该通信区域的内容必须是后续进程可以接受的代码页。通常这意味着它们必须是 EBCDIC。
2. 转换器的**译码**函数为 DFHWBTTA 构造通信区域。
3. 由于这不是对话或者伪会话的第一个事务，因此请求包括了 HTML 格式，它与事务程序希望接收的映射相对应。DFHWBTTA 抽取表单数据，以使之可通过 RECEIVE MAP 命令用于事务程序。
4. 进入表单的输入数据转换成一个 BMS 映射。这个转换使用 DOCTEMPLATE 定义中的模板。
5. 事务程序使用 RECEIVE MAP 命令接收数据。然后它构造输出映射，并使用 SEND MAP 命令将它发送给请求程序。

6. 映射及其数据内容转换为 HTML 格式。这个转换使用 DOCTEMPLATE 定义中的模板。
7. 转换器的**编码**函数使用转换器进程的 HTML 输出来构造要返回给调用者的通信区域。
8. CICS 业务逻辑接口返回到它的调用者，现在该调用者就可以使用通信区域的内容了。

偏移方式和指针方式

可以用两种方式调用 CICS 业务逻辑接口:

偏移方式

在偏移方式中，有单个存储器区域（在图 26 中为存储区 1），它包含 DFHWBBLI 通信区域和 CICS 应用程序区域。DFHWBBLI 通信区域中的字段 *wbbl_indata_offset* 包含了以存储器区域开始为起点的应用程序通信区域的偏移量。这个存储器区域可以存储的最大数据量为 32K 字节。

在偏移方式中，转换器程序不能更改 DECODE_DATA_PTR 或 ENCODE_DATA_PTR 的值。

指针方式

在指针方式中，有两个独立的存储器区域: 一个区域（图 26 中的存储器区域 1）包含 DFHWBBLI 的通信区域，而另一个区域（存储器区域 2）包含 CICS 应用程序的区域。DFHWBBLI 通信区域中的字段 *wbbl_indata_ptr* 包含了应用程序通信区域的地址。

在指针方式中，转换器程序可以更改 DECODE_DATA_PTR 或 ENCODE_DATA_PTR 的值。

图 26 中说明了这两种方式。

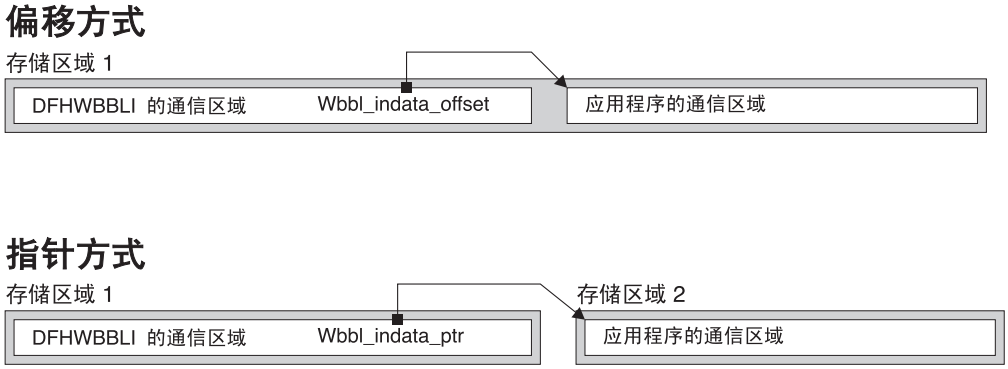


图 26. CICS 业务逻辑接口中的偏移方式和指针方式

调用 CICS 业务逻辑接口时，必须指定方式:

- wbbl_mode 设置为“O”时，表示偏移方式
- wbbl_mode 设置为“P”时，表示指针方式

在转换器程序中，可以测试 `decode_volatile` 或 `encode_volatile` 来确定表示的方式:

- 0 表示偏移方式
- 1 表示指针方式

当调用 CICS 业务逻辑接口时，来自以下任何源的所有请求都使用偏移方式:

- 使用 IBM HTTP Server 的 Web 客户机。
- 使用本地网关功能的 Java 应用程序。
- DCE RPC 客户机。
- 使用 CICS Transaction Gateway 的 Web 客户机。

代码页转换和 CICS 业务逻辑接口

CICS 业务逻辑接口不执行代码页转换；您传递到业务应用程序的数据以及返回的数据都在应用程序接口使用的代码页中。

然而，EXEC CICS WEB 应用程序编程命令允许您指定客户机代码页，数据转换将在应用程序编程接口本身进行。因此，当您使用这些命令时，代码页转换是在应用程序和 CICS 业务逻辑接口之间执行的。通过接口传递的数据采用 EXEC CICS WEB 命令的 CHARACTERSET 选项（或其同义词 CLNTCODEPAGE）中指定的代码页。

配置 CICS 业务逻辑接口

关于此任务

1. 必须设置 **WEBDELAY** 系统初始化参数，如第 49 页的『为 CICS Web Support 指定系统初始化参数』中所述。
2. 如果没有使用程序的自动安装，必须定义 CICS 业务逻辑接口的调用者使用的所有用户可替换程序（转换器）。 如果使用了程序的自动安装，就不需要定义转换器。所有的转换器对于正在操作 CICS 业务逻辑接口的系统必须是本地的。

第 5 部分 附录

附录 A. HTML 编码字符集

本参考列出了支持的 IANA 注册字符集名称（在 HTTP 头中指定为 charset=），以及 IBM CCSID 等价值。

所有这些值对于以下命令中的代码页转换选项都是有效的:

- WEB RECEIVE（客户机）
- WEB RECEIVE（服务器）
- WEB SEND（客户机）
- WEB SEND（服务器）
- WEB CONVERSE
- DOCUMENT RETRIEVE
- WEB READ FORMFIELD
- WEB STARTBROWSE FORMFIELD

表 14. 编码字符集

语言	编码字符集	IANA 字符集	IBM CCSID
阿尔巴尼亚语	ISO/IEC 8859-1	iso-8859-1	819
阿拉伯语	ISO/IEC 8859-6	iso-8859-6	1089
保加利亚语	Windows 1251	windows-1251	1251
白俄罗斯语	Windows 1251	windows-1251	1251
加泰罗尼亚语	ISO/IEC 8859-1	iso-8859-1	819
简体中文	GB	gb2312	1381 或 5477
繁体中文	Big 5	big5	950
克罗地亚语	ISO/IEC 8859-2	iso-8859-2	912
捷克语	ISO/IEC 8859-2	iso-8859-2	912
丹麦语	ISO/IEC 8859-1	iso-8859-1	819
荷兰语	ISO/IEC 8859-1	iso-8859-1	819
英语	ISO/IEC 8859-1	iso-8859-1	819
爱沙尼亚语	ISO/IEC 8859-1	iso-8859-1	819
芬兰语	ISO/IEC 8859-1	iso-8859-1	819
法语	ISO/IEC 8859-1	iso-8859-1	819
德语	ISO/IEC 8859-1	iso-8859-1	819
希腊语	ISO/IEC 8859-7	iso-8859-7	813
希伯来语	ISO/IEC 8859-8	iso-8859-8	916
匈牙利语	ISO/IEC 8859-2	iso-8859-2	912
意大利语	ISO/IEC 8859-1	iso-8859-1	819
日语	Shift JIS EUC Japanese	x-sjis 或 shift-jis euc-jp	943（943 的子集 932 也是有效的） 5050（EUC）

表 14. 编码字符集 (续)

语言	编码字符集	IANA 字符集	IBM CCSID
韩国语	EUC Korean	euc-kr	970（用于 AIX® 或 Unix）
拉托维亚语	Windows 1257	windows-1257	1257
立陶宛语	Windows 1257	windows-1257	1257
马其顿语	Windows 1257	windows-1257	1251
挪威语	ISO/IEC 8859-1	iso-8859-1	819
波兰语	ISO/IEC 8859-2	iso-8859-2	912
葡萄牙语	ISO/IEC 8859-1	iso-8859-1	819
罗马尼亚语	ISO/IEC 8859-2	iso-8859-2	912
俄语	Windows 1251	windows-1251	1251
塞尔维亚语（西里尔语）	Windows 1251	windows-1251	1251
塞尔维亚语（拉丁 2）	Windows 1250	windows-1250	1250
斯洛伐克语	ISO/IEC 8859-2	iso-8859-2	912
斯洛文尼亚语	ISO/IEC 8859-2	iso-8859-2	912
西班牙语	ISO/IEC 8859-1	iso-8859-1	819
西班牙语	ISO/IEC 8859-15	iso-8859-15	923
瑞典语	ISO/IEC 8859-1	iso-8859-1	819
土耳其语	ISO/IEC 8859-9	iso-8859-9	920
乌克兰语	Windows 1251	windows-1251	1251
Unicode	UCS-2	iso-10646-ucs-2	1200（不断增加的）或 13488（不变的）
Unicode	UTF-16	utf-16	1200
Unicode	UTF-16 大尾数法	utf-16be	1201
Unicode	UTF-16 小尾数法	utf-16le	1202
Unicode	UTF-8	utf-8	1208

附录 B. CICS Web Support 的 HTTP 头参考

在 CICS Web Support 中，CICS 会自动在出站消息上提供某些 HTTP 头，您也可以添加自己的 HTTP 头。消息发送到 CICS 后，CICS 将对某些 HTTP 头执行响应操作，而用户应用程序可以对其他头执行响应操作。本参考信息描述 CICS Web Support 如何处理 HTTP 头。

标准 HTTP 头在 HTTP/1.1 规范（RFC 2616）和 HTTP/1.0 规范（RFC 1945）中描述。存在很多可能的 HTTP 头，包括扩展头，它们不是 HTTP 协议规范的一部分。要获取更完整的列表，请参阅您所遵从的 HTTP 规范。第 12 页的『HTTP 协议』提供了关于 HTTP 规范的更多信息。

本主题说明 CICS Web Support 中 HTTP 头的一般用法，以及 CICS Web Support 为特定头执行的操作。要获取有关您应该如何使用 HTTP 头的详细指导和需求（例如，头值的正确格式和应该使用每个头的上下文），请检查您正在遵从的 HTTP 规范。

CICS 收到的消息上的 HTTP 头

- CICS 接收 HTTP 请求或响应时，其中一些 HTTP 头用于确定 CICS Web Support 执行的操作。第 336 页的表 15 说明了 CICS 针对 HTTP 请求上的头执行的操作。第 337 页的表 16 说明了 CICS 针对 HTTP 响应上的头执行的操作。CICS 不使用其他头，而对它们执行何种合适的响应操作则由用户应用程序决定。
- 使用 WEB READ HTTPHEADER 命令和 HTTP 头浏览命令，为消息接收的所有头（不管它们是否已由 CICS 使用）供用户应用程序进行检查。CICS 不提醒用户应用程序消息上存在任何特殊头。忽略应用程序不需要或不理解的任何头。
- CICS 已经处理 HTTP/1.1 规范中的 MUST 级别需求，这些需求与服务器或客户机在接收消息时必须执行的操作有关。因为这个原因，所以你可以接收并使用请求或响应，而无需检查头。然而，您将可能需要检查头以获取一些信息，这些信息与您将来与 Web 客户机或服务器通信所执行的操作有关。
- HTTP 头包括头名称和头值，二者用冒号隔开。HTTP/1.1 规范规定：冒号和头值之间应留有一个空格。您应使头遵循这一通用格式。在 HTTP/1.0 规范中，规定应采用一个空格。但 HTTP/1.1 规范允许应用程序使用多个空格或不使用空格。为保持向后兼容性，CICS 要求在某些头（例如，Content-Length 头）中采用单个空格的通用格式，在消息处理期间 CICS 将对这些头执行操作。如果您正在设计向 CICS 发送 HTTP 请求的应用程序，请确保对于所有 HTTP 头都遵循 HTTP/1.1 规范中推荐的格式。EXEC CICS WEB WRITE HTTPHEADER 命令会生成具有适当格式的头，而且 CICS 自动编写的所有头都将采用合适的格式。

从 CICS 发出的消息上的 HTTP 头

- 在从 CICS 发出的 HTTP/1.1 版本的 HTTP 请求或响应上，CICS 自动提供通常应该为基本消息所写的、以使该消息遵从 HTTP/1.1 规范的密钥头。在版本为 HTTP/1.0 的 HTTP 响应上，CICS 自动提供较少量的头。其中一些头由 CICS 为每条消息生成，而一些头是因为您在用户应用程序中的 WEB SEND 命令中指定的选项而生成的。第 338 页的表 17 和第 339 页的表 18 列出为每个 HTTP 版本编写的头，以及头的源代码。

  如果用户应用程序写 CICS 也会生成的头，那么 CICS 将根据以下情形处理它:

- 对于作为 HTTP 服务器的 CICS，如果该头适合于响应，那么 CICS 不覆盖它，但允许使用应用程序的版本。
- 对于作为 HTTP 客户机的 CICS，如果该头适合于请求，那么 CICS 不允许应用程序写它，并将错误响应返回给 WEB WRITE HTTPHEADER 命令。TE 头和 Content-Type 头例外。应用程序可以添加 TE 头的更多实例。如果所需的头需要包含空格或超过 56 个字符，而因此无法在 WEB SEND 命令的 MEDIATYPE 选项上指定，那么这些应用程序也可以提供 Content-Type 头。
- 如果该头通常不适合于消息类型（请求或响应），那么 CICS 允许它，如所有用户定义的头一样。如果您的消息遵从您正在遵从的 HTTP 规范，那么不应该发生该情形。

- 用户应用程序可以使用 WEB WRITE HTTPHEADER 命令将更多 HTTP 头添加到请求或响应中。CICS 允许并传递任何其他 HTTP 头。注意，对于作为 HTTP 服务器的 CICS，如果要提供具有 CICS 文档或 HFS 文件的静态响应，那么头只能添加到 CICS 自动提供的响应。
- CICS 不检查用户所写的头的名称或值。您应该确保您的应用程序正在以符合您正在遵从的 HTTP 规范的方式提供内容正确且格式正确的信息。如果您的应用程序正在执行复杂的操作，请特别小心地检查 HTTP 规范以获取可应用的需求。可能存在一些重要的（MUST 或 SHOULD 级别）需求以提供描述这些操作的特定头。例如，如果您正在执行以下操作，那么需要特殊 HTTP 头:
  - 正在使用文档或实体标记的修改日期响应或发出条件请求。
  - 根据 Web 客户机的客户机功能或本地语言需求变化响应的内容。
  - 提供响应或发出请求，它涉及某范围的文档，而不是完整的文档。
  - 为响应提供高速缓存控制信息。

  在您的响应上使用特定状态码可能还需要特殊 HTTP 头。例如，如果使用状态码 405（不允许方法），那么必须使用 Allow 头来声明允许的方法。第 341 页的附录 C, 『CICS Web Support 的 HTTP 状态码参考』提供了关于使用状态码的更多信息。

Upgrade 头

- 注意，作为 CICS Web Support 中的特例，不支持协议升级。这意味着:
  - 对于作为 HTTP 服务器的 CICS，应用程序不可能执行任何操作以响应 Web 客户机发送的 Upgrade 头。
  - 对于作为 HTTP 客户机的 CICS，不能在请求上写 Upgrade 头。

  连接期间，CICS 不支持 HTTP 版本中的交换，且安全层的升级也不受支持。

作为 HTTP 服务器的 CICS: CICS 接收 HTTP 请求时执行操作的头

表 15 说明 CICS 为从 Web 客户机接收的请求上的特定头而执行的操作。

表 15. 作为 HTTP 服务器的 CICS: 对 HTTP 请求头的 CICS 操作。

从 Web 客户机接收的头	CICS 执行的操作，其中的响应将由用户应用程序处理	CICS 执行的操作，其中的响应将由静态文档提供
Authorization	将提供的用户标识和密码传递到 RACF 进行验证，并在这些内容无效的情况下拒绝请求。	关于应用程序生成的响应。
Connection	执行 Web 客户机的请求以使连接在发送响应后关闭。	关于应用程序生成的响应。

表 15. 作为 HTTP 服务器的 CICS: 对 HTTP 请求头的 CICS 操作。 (续)

从 Web 客户机接收的头	CICS 执行的操作，其中的响应将由用户应用程序处理	CICS 执行的操作，其中的响应将由静态文档提供
Content-Length	CICS 要求在具有消息体的所有入站 HTTP/1.1 消息中都有 Content-Length 头。如果有消息体但未提供头，或头的值不正确，那么错误消息或后继消息的套接字接收会产生不可预测的结果。对于具有消息体的 HTTP/1.0 消息，Content-Length 头是可选的。	虽然消息体不用于静态响应的处理，但必须仍然从套接字接收静态响应，因此相同的要求也应用于应用程序生成的响应。
Content-Type	对头进行语法分析，为代码页转换识别介质类型和字符集。	对头进行语法分析，为响应的代码页转换识别字符集。
Expect	将 100-Continue 响应发送到 Web 客户机并等待剩余请求。	关于应用程序生成的响应。
Host	如果该头不存在且客户机版本是 HTTP/1.1，那么将 400（错误请求）响应发送到 Web 客户机。	关于应用程序生成的响应。
If-Modified-Since	CICS 不采取任何操作。用户应用程序可以根据需要检查该头及响应是否存在，也可以忽略头并假定应用程序生成的响应已被修改。	文档模板: 假定响应已修改并发送了请求的项。HFS 文件: 检查修改日期，并根据检查结果作出响应。如果尚未修改项，那么发送 304 响应。
If-Unmodified-Since	如果头存在，那么总是将 412（预备条件失败）响应发送到 Web 客户机，表明自指定的时间以来已修改响应。（这意味着用户应用程序不必检查该头是否存在。）	文档模板: 对于应用程序生成的响应，假定响应已修改，并发送 412 响应。HFS 文件: 检查修改日期，并根据检查结果作出响应。
Trailer	通过 WEB READ HTTPHEADER 命令使独立的尾部头对应用程序可用。	分块的消息不适合静态响应。
Transfer-Encoding	对于“分块”，接收所有块并组装到单条消息中以传递到应用程序。对于除“分块”外的任何内容，将 501（未实施）响应发送到 Web 客户机。Transfer-Encoding 保留在消息上，但是它仅供参考。	分块的消息不适合静态响应。
Warning	将警告文本写入 TS 队列 CWBW。如果使用的字符多于 128 个，那么会截断警告文本。	关于应用程序生成的响应。

作为 HTTP 客户机的 CICS: CICS 接收 HTTP 响应时执行操作的头

表 16 说明 CICS 为从服务器接收的响应上的特定头而执行的操作。

表 16. 作为 HTTP 客户机的 CICS: 对 HTTP 响应头的 CICS 操作。

从服务器接收的头	CICS执行的操作
Connection	执行服务器的请求以使连接在接收响应后关闭。

表 16. 作为 HTTP 客户机的 CICS: 对 HTTP 响应头的 CICS 操作。 (续)

从服务器接收的头	CICS执行的操作
Content-Length	CICS 要求在具有消息体的所有入站 HTTP/1.1 消息中都有 Content-Length 头。如果有消息体但未提供头，或头的值不正确，那么错误消息或后继消息的套接字接收会产生不可预测的结果。对于具有消息体的 HTTP/1.0 消息，Content-Length 头是可选的。
Content-Type	对头进行语法分析，为代码页转换识别介质类型和字符集。
Trailer	通过 WEB READ HTTPHEADER 命令使尾部头对应用程序可用。
Transfer-Encoding	对于“分块”，接收所有块并组装到单条消息中以传递到应用程序。对于除“分块”外的任何内容，将 501（未实施）响应发送到 Web 客户机。Transfer-Encoding 保留在消息上，但是它仅供参考。
Warning	将警告文本写入 TS 队列 CWBW。如果使用的字符多于 128 个，那么会截断警告文本。

作为 HTTP 服务器的 CICS: CICS 为 HTTP 响应写的头

表 17 显示了响应来自 Web 客户机的请求时 CICS 所写的头、这些头使用的 HTTP 版本以及 CICS 在头中提供的信息源。

表 17. 作为 HTTP 服务器的 CICS: CICS 为 HTTP 响应写的头

CICS 写的头	HTTP 版本	用户应用程序处理响应的源代码	静态文档提供响应的源代码
Connection	1.0 和 1.1	WEB SEND 命令中的 CLOSESTATUS 选项。如果未指定 close，且客户机版本是 HTTP/1.0，那么发送 Keep-Alive。如果指定了 close，那么发送 Connection: close，或如果是 HTTP/1.0 客户机，那么省略 Keep-Alive。	在静态响应中发送 Keep-Alive。
Content-Length（除非使用了分块的传输编码）	1.0 和 1.1	其中响应主体是一缓冲区数据，长度从 WEB SEND 命令的 FROMLENGTH 选项获取。（CICS 检查您指定的长度。如果长度不正确，那么 CICS 发送响应但省略 Connection: Keep-Alive 头。）其中响应主体是 CICS 文档，长度由 CICS 计算。	由 CICS 计算。
Content-Type	1.0 和 1.1	WEB SEND 命令中的 MEDIATYPE 选项和响应主体的字符集。（只有在指定 MEDIATYPE 选项时才创建头。）	请求的 URIMAP 资源定义的 MEDIATYPE 属性和响应主体的字符集。
Date	1.0 和 1.1	CICS 生成的当前日期和时间。	CICS 生成的当前日期和时间。
Last-Modified（仅用于静态 HFS 文件）	1.0 和 1.1	**不为动态响应提供。**应用程序应该在可行之处生成该内容。	对于 HFS 文件: 文件的修改日期。对于文档模板: 不提供。
Server	1.0 和 1.1	预先设置为“IBM_CICS_Transaction_Server/ 4.1.0 (zOS)”。	预先设置为“IBM_CICS_Transaction_Server/ 4.1.0 (zOS)”。
Transfer-Encoding	仅 1.1	WEB SEND 命令上的 CHUNKING 选项。	未使用。
WWW-Authenticate	1.0 和 1.1	TCPIPSERVICE 资源定义的 AUTHENTICATE 属性。	TCPIPSERVICE 资源定义的 AUTHENTICATE 属性。

作为 HTTP 客户机的 CICS: CICS 为 HTTP 请求写的头

表 18 显示了应用程序将客户机请求发送到服务器时 CICS 所写的头、这些头使用的 HTTP 版本以及 CICS 在头中提供的信息源。

表 18. 作为 HTTP 客户机的 CICS: CICS 为 HTTP 请求写的头

CICS 写的头	HTTP 版本	源
Connection	1.0 和 1.1	WEB SEND 命令中的 CLOSESTATUS 选项。根据服务器的 HTTP 版本选择头值。
Content-Length（除非使用了分块的传输编码）	1.0 和 1.1	WEB SEND 命令中的 FROMLENGTH 选项。（CICS 检查您指定的长度。如果长度不正确，那么 CICS 发送响应但省略 Connection: Keep-Alive 头。）
Content-Type	1.0 和 1.1	WEB SEND 命令中的 MEDIATYPE 选项和响应主体的字符集。（只有在指定 MEDIATYPE 选项时才创建头。）如果所需的头需要包含空格或超过 56 个字符，而因此无法在 MEDIATYPE 选项上被指定，那么应用程序（而不是 CICS）可以提供 Content-Type 头。
Date	1.0 和 1.1	CICS 生成的当前日期和时间，格式为带 GMT 时间的 RFC 1123。
Expect	仅 1.1	WEB SEND 命令中的 ACTION(EXPECT) 选项。仅当您的请求有消息体时才能使用该选项。CICS 不将头发送到 HTTP/1.0 服务器。如果 CICS 还未识别出服务器版本，那么指定 ACTION (EXPECT) 选项将触发使用 OPTIONS 方法的其他请求。
Host	1.0 和 1.1	WEB OPEN 命令中的 HOST 选项。
TE	仅 1.1	发送到 HTTP/1.1 服务器时总是由 CICS 添加，以声明接受分块的消息和尾部。（HTTP/1.0 服务器不发送分块的消息。）应用程序可以添加更多 TE 头。
Transfer-Encoding	仅 1.1	序列中发送分块消息的第一个 WEB SEND 命令（该命令中的 CHUNKING 选项表明已分块的 transfer-coding）。Transfer-Encoding 头仅写在第一块消息上。
User-Agent	1.0 和 1.1	预先设置为“IBM_CICS_ Transaction_Server/ 4.1.0 (zOS)”。

附录 C. CICS Web Support 的 HTTP 状态码参考

HTTP 状态码由服务器提供给客户机以说明客户机请求的结果。当 CICS 是 HTTP 服务器时，根据不同的情况，CICS Web Support 或用户应用程序为每个响应选择合适的状态码。当 CICS 是 HTTP 客户机时，从服务器接收的大多数状态码传递到用户应用程序以供处理。

第 15 页的『状态码和原因短语』说明了状态码如何用于 HTTP 响应。

要获取有关状态码的含义和正确使用的完整信息，应该查询您正在遵从的 HTTP 规范。第 12 页的『HTTP 协议』提供了关于 HTTP 规范的更多信息。

本主题提供与 CICS Web Support 相关的 HTTP/1.1 状态码的简短摘要。当选择通过 Web 出错程序或直接从用户应用程序发送状态码时，检查您所遵从的 HTTP 规范是很重要的。HTTP 规范提供有关应如何使用状态码的详细指导和要求（如，响应主体的内容应该是什么以及应包括什么 HTTP 头）。

CICS 发送的响应的状态码（当 CICS 是 HTTP 服务器时）

- CICS Web Support 在以下情况下生成对 Web 客户机的响应:
  - 当 CICS Web Support 在初始处理来自 Web 客户机的请求时检测到问题；例如，如果请求中缺少必需的信息，或者如果信息发送太慢，而且达到接收超时。
  - 当已安装的 URIMAP 定义与请求匹配，但禁用 URIMAP 定义或虚拟主机，或者，无法读取静态响应的资源时。
  - 当匹配的 URIMAP 定义将请求提交给 ATOMSERVICE 资源定义，但 ATOMSERVICE 定义处于禁用状态，或者无法读取 Atom 订阅源的 CICS 资源时。
  - URIMAP 匹配失败，为 TCPIPSERVICE 定义指定的分析器不能处理请求，并将控制权传递给 Web 出错程序。
  - URIMAP 定义和分析器程序以及转换器程序都无法确定应执行什么应用程序来服务请求。
  - 分析器程序、转换器程序或用户编写的应用程序中发生异常终止。这会确保即使处理已失败也会将响应返回给 Web 客户机。
  - 当 URIMAP 指定重定向响应时。
  - 当 Web 客户机无权访问提供响应时所需的资源时。

  在这些情况中，CICS 选择合适的状态码并创建缺省响应。第 342 页的表 19 描述了 CICS 针对这些目的所使用的状态码。注意，在以下这种情况下，CICS 不生成响应: 用户编写的应用程序已成功完成处理并要返回表明错误的响应；例如，当客户机指定了资源不支持的方法时，就会发生这种情况。在这种情况下，用户编写的应用程序创建响应。
- 对于状态码为 4xx 和 5xx 的 CICS 所生成的响应，通过编辑用户可替换的 Web 出错程序 DFHWBEP 和 DFHWBERX，可以修改发送给 Web 客户机的响应。如果 CICS 生成的响应涉及 1xx、2xx 和 3xx 状态码，那么不能修改。Web 出错程序可以更改响应的状态码、原因短语、HTTP 头和消息体。当修改 Web 出错程序时，确

保按照您所遵从的 HTTP 规范中的要求选择状态码和响应内容。第 103 页的第 9 章, 『Web 出错程序』说明了如何定制 Web 出错程序。

- 对客户机的请求作出响应的用户应用程序需要为响应选择合适的状态码。状态码可将以下消息传送到 Web 客户机:
  - 请求已按照预期的完成。
  - 发生阻碍完成请求的错误。
  - 客户机需要执行某些操作以成功完成它的请求。这包括遵循重定向 URL，或修正请求以使它可被服务器接受。

  状态码影响到响应的其他内容，即消息体和 HTTP 头。第 76 页的『从作为 HTTP 服务器的 CICS 发送 HTTP 响应』介绍了如何汇编和发送响应，包括状态码和原因短语。

CICS 接收的响应的状态码（当 CICS 是 HTTP 客户机时）

- 当 CICS 是 HTTP 客户机时，CICS Web Support 将带有大部分状态码的响应直接传递到用户应用程序以进行处理。少量的状态码由 CICS 处理并且不返回到应用程序。如果状态码传递到应用程序，那么表明 CICS 未采取任何操作来响应该代码，并且由应用程序负责检查该代码并采取相应的操作。
- 您应适当地设计您的用户应用程序，当它接收带有表明错误的状态码的消息时，能作出合适的反应。特别地，在以下情况下您应总是检查状态码:
  - 如果您打算现在或者在将来连接时向服务器发送相同的请求。
  - 如果您打算使用该连接向服务器发送更多请求。
  - 如果您的应用程序正在执行任何进一步的处理，而该处理取决于响应中接收的信息。

  检查您所遵从的 HTTP 规范以获取有关什么操作较合适的指导。HTTP/1.1 规范不包含 MUST 级别要求，它们要求在接收状态码时，应用程序需进一步操作，但包含某些 SHOULD 要求（如要求遵循重定向）。

作为 HTTP 服务器的 CICS: CICS 向 Web 客户机提供的状态码。

表 19 显示 CICS 向 Web 客户机的请求提供响应时使用的状态码。可以通过修改 Web 出错程序来定制这些响应中的一部分。用户应用程序还可使用此处列出的许多状态码。

某些状态码仅适用于 HTTP/1.1 客户机。CICS 不会将这些状态码返回给 HTTP/1.0 客户机。

表 19. 作为 HTTP 服务器的 CICS: 发送给 Web 客户机、由 CICS 生成的响应的状态码

所提供的状态码和原因短语	发送到 HTTP/1.0 客户机了吗?	提供该响应的情况	可以在 Web 出错程序中修改吗?
100 继续	否	Web 客户机已发送 Expect 头。	否
200 正常	是	正常响应的传递	否
201 已创建	是	新对象已创建。	否
301 已永久移开	是	URIMAP 定义指定重定向，具有属性 REDIRECTTYPE (PERMANENT)。	否

表 19. 作为 HTTP 服务器的 CICS: 发送给 Web 客户机、由 CICS 生成的响应的状态码 (续)

所提供的状态码和原因短语	发送到 HTTP/1.0 客户机了吗?	提供该响应的情况	可以在 Web 出错程序中修改吗?
302 已找到	是	URIMAP 定义指定重定向，具有属性 REDIRECTTYPE (TEMPORARY)。	否
304 未修改	是	对请求使用了 If-Modified-Since 头，并且 CICS 能够验证静态响应是否未经过修改。	是
400 错误请求（某些情况下: 无效请求）	是	请求中的语法错误（如: 错误地指定请求行，请求未完成或 Atom POST 或 PUT 请求的 Atom 条目有问题）。或者，未提供 Host 头（仅 HTTP/1.1）。或者，收到不带 If-Match 头的 PUT 请求。对于想要更新对象而又不知道当前实体标记的客户机，应该指定“If-Match: *”。	是
401 基本认证错误	是	基本认证需要用户标识和密码。这由端口的 TCPIPSERVICE 定义的安全设置确定。	是
403 已禁止（某些情况: 客户机认证错误）	是	基本认证未成功。或者，客户机证书有问题。或者，用户无权访问提供响应时所需的资源，例如， ATOMSERVICE 资源定义、别名事务、程序使用的 CICS 命令，或包含响应数据的 CICS 资源。	是
404 未找到（某些情况: 未找到程序，未找到文件）	是	未找到指定对请求作出响应的程序。或者，未找到提供响应时所需的资源。或者，在用于为 Atom 订阅源提供数据的 CICS 资源中未找到记录。或者，未找到图像文件。	是
408 请求超时	否	已超过请求的接收超时。这由端口的 TCPIPSERVICE 定义中的 SOCKETCLOSE 属性确定。	是
409 冲突（某些情况下: 资源重复）	是	某个现有对象已在使用该指定的 URL，所以不会创建新对象。	否
412 预备条件失败	是	请求中已使用 If-Unmodified-Since 头。或者，If-Match 头上的实体标记值与正在更新的对象的实体标志不匹配。	是
417 未达到期望的	否	接收到没有值“100-continue”的 Expect 头。	否
500 内部服务器错误（某些情况下: 资源错误）	是	涉及处理请求和提供响应的某个程序中的异常终止。或在读取针对静态响应的 z/OS UNIX 文件时出错。或者，在包含 Atom 订阅源的资源时出错，如在从用作 Atom 条目内容的资源记录生成 XML 标记时出错。	是
501 方法未实施	是	对于该 HTTP 版本，方法不受 CICS 的支持。（包括受支持但不是关于客户机请求方面的方法，如引用特定资源的 OPTIONS 请求。）或者，请求的介质类型是“multipart/byteranges”，它不受支持。或者，请求的转换编码不同于“分块的”。（注: 连接由 CICS 关闭。）	是
503 服务不可用	是	存在匹配 URIMAP 定义，但禁用它，或禁用它为其中一部分的虚拟主机。或者，匹配的 URIMAP 定义引用了被禁用的 ATOMSERVICE 资源定义。或者，禁用 URIMAP 定义或 ATOMSERVICE 定义中为提供响应数据而指定的资源。	是
505 不受支持的版本	否	HTTP 版本比 1.1 新，并且方法没有被 CICS 支持的最新版本识别。	是

作为 HTTP 服务器的 CICS: 用户应用程序中的状态码

表 20 显示每个状态码，描述它与用户应用程序的关联，并根据 HTTP/1.1 规范中的推荐建议合适的操作。

请记住，CICS 不采取可能由这些状态码暗示的任何特定操作，并且 CICS 通常不会根据消息的内容检查它们的有效性。您应该确保状态码正确并且您已采取任何必要的操作。确保检查所遵从的 HTTP 规范以获取更多信息和适用于每个状态码的要求。

表 20. 作为 HTTP 服务器的 CICS: 发送给 Web 客户机的、用户编写的响应的状态码

状态码和通常的原因短语	适用于 HTTP/1.0 客户机吗?	您可能提供该响应的情况	对消息体和 HTTP 头的影响（状态码适用于用户应用程序的情况）。请参阅 HTTP 规范以获取更多信息。
100 继续	否	不要使用。CICS 处理需要的请求并自己发送 100-Continue 响应。	
101 转换协议	否	不要使用。CICS 不支持 HTTP 版本或安全协议的升级。	
200 正常	是	您已完成请求。正常的响应。	提供正常响应体。
201 已创建	是	您已创建新的资源。（如果还未创建资源则使用 202 已接受）。	需要消息体内容和一个或多个头。
202 已接受	是	您已接受请求但还未处理它，并且不保证要处理它。	需要消息体内容。
203 非权威信息	否	不要使用。您提供的头将给出权威信息。	
204 无内容	是	您未发送消息体，也许因为您只需发送已更新的头。	不允许消息体。
205 重置内容	否	您希望客户机清除发出请求的表单。	不允许消息体。
206 部分内容	否	您支持字节范围请求，并且该响应满足了请求。	正常响应体。需要一个或多个头。
300 多个选项	是	您能提供资源的多个版本（例如，不同语言的文档）。	需要消息体内容和一个或多个头。
301 已永久移开	是	建议不要由用户应用程序发出。可使用 URIMAP 定义中的 LOCATION 和 REDIRECTTYPE 属性管理重定向，以使 CICS 生成正确的响应而无需调用应用程序。REDIRECTTYPE (PERMANENT) 选择该状态码。	
302 已找到	是	建议不要由用户应用程序发出。当将 URIMAP 定义用于重定向时，REDIRECTTYPE (TEMPORARY) 选择该状态码。	
303 参阅其他内容	否	您希望客户机为另一个给出响应（特别是有关 POST 请求的结果的响应）的资源发出 GET 请求。	需要消息体内容和一个或多个头。
304 未修改	是	客户机发出有条件的请求并且您提供的资源未更改。注意，由应用程序动态构建的响应可能要在每个请求上修改。对于不更改的资源，考虑使用 URIMAP 定义传递静态响应。	不允许消息体。（您可以使用 DOCTOKEN 选项指定不含内容的文档。）需要一个或多个头。

表 20. 作为 HTTP 服务器的 CICS: 发送给 Web 客户机的、用户编写的响应的状态码 (续)

状态码和通常的原因短语	适用于 HTTP/1.0 客户机吗?	您可能提供该响应的情况	对消息体和 HTTP 头的影响（状态码适用于用户应用程序的情况）。请参阅 HTTP 规范以获取更多信息。
305 使用代理	否	您希望客户机通过命名代理来发出它的请求。	需要一个或多个头。
307 临时重定向	否	建议不要由用户应用程序发出。对于 URIMAP 重定向，CICS 使用 302 状态码而不是该状态码。	
400 错误请求	是	客户机的请求包含语法错误或类似的问题，并且您无法处理它。	需要消息体内容。
401 未授权	是	不要使用。当在 TCPIPSERVICE 定义的安全设置中指定时，由 CICS 处理基本认证过程。	
403 禁止	是	您拒绝客户机的请求。	需要消息体内容。
404 未找到	是	您没有响应该请求的资源; 或者，您要拒绝请求而没有说明；或者，没有其他相关的状态码。	需要消息体内容。
405 不允许的方法	否	客户机使用了该资源不支持的方法。	需要消息体内容和一个或多个头。
406 不可接受	否	客户机使用 Accept 头发出了有条件的请求，但您没有满足它们的标准的资源版本。注意，作为使用该状态码的替换方法，您可发送不满足条件的响应。	需要消息体内容。
407 需要代理认证	否	不要使用。CICS不担当代理服务器。	
408 请求超时	否	建议不要由用户应用程序发出。应使用 TCPIPSERVICE 定义中的 SOCKETCLOSE 属性指定超时以由 CICS Web Support 处理。	
409 冲突	否	已更改资源并且无法将客户机的请求应用到资源，因为它现在处于停止状态。	需要消息体内容。
410 不可用	否	资源永久地不可用。	需要消息体内容。
411 需要长度	否	不要使用。CICS 需要 HTTP/1.1 请求为成功的套接字接收指定 Content-Length 头。	
412 预备条件失败	否	客户机发出了有条件的请求且条件未满足。	需要消息体内容。
413 请求实体太大	否	建议不要由用户应用程序发出。应使用 TCPIPSERVICE 定义中的 MAXDATALEN 属性指定请求大小限制以由 CICS Web Support 处理。	
414 请求 URI 太长	否	客户机的请求 URL 太长而使应用程序无法处理。	需要消息体内容。
415 不支持的介质类型	否	客户机发送的消息体具有您不支持的介质类型或内容编码。	需要消息体内容。

表 20. 作为 HTTP 服务器的 CICS: 发送给 Web 客户机的、用户编写的响应的状态码 (续)

状态码和通常的原因短语	适用于 HTTP/1.0 客户机吗?	您可能提供该响应的情况	对消息体和 HTTP 头的影响（状态码适用于用户应用程序的情况）。请参阅 HTTP 规范以获取更多信息。
416 无法满足请求的范围	否	客户机使用 Range 头字段（而不是 If-Range 头字段）发出请求，并且尽管您支持字节范围，该范围也不出现在资源中。	需要消息体内容和一个或多个头。
417 未达到期望的	否	不要使用。CICS 处理期望的请求。	
500 内部服务器错误	是	由于应用程序或系统错误，您无法处理请求。	需要消息体内容。
501 未实施	是	不支持客户机的请求的方法。在客户机为 HTTP/1.0 或您在使用 USER 协议的情况下才应发出该状态码。对于 HTTP 协议，初始处理期间，CICS 拒绝任何具有无法识别的方法的请求。如果方法被识别但不适用于这些资源，那么“405 不允许的方法”应该用于 HTTP/1.1 客户机。	需要消息体内容。
502 错误网关	是	不要使用。CICS 不担当代理或网关。	
503 服务不可用	是	除非用户应用程序需要访问另一个临时不可用的应用程序或系统，否则它可能与使用该状态码的情况无关。	需要消息体内容和一个或多个头。
504 网关超时	否	不要使用。CICS 不担当代理或网关。	
505 不支持的 HTTP 版本	否	不要使用。CICS 将响应的 HTTP 版本与客户机的请求的 HTTP 版本相匹配。	

作为 HTTP 客户机的 CICS: 处理有关服务器的响应的状态码

表 21 显示您可能在来自服务器的响应上接收的状态码，并根据 HTTP/1.1 规范中的推荐建议合适的操作。WEB RECEIVE 命令返回状态码和状态文本。记住，服务器可能已更改 HTTP 规范中建议的原因短语的文本。

确保检查所遵从的 HTTP 规范以获取更多信息和适用于每个状态码的要求。

表 21. 作为 HTTP 客户机的 CICS: 处理有关响应的状态码

状态码和可能的原因短语	服务器为何要发送该状态码?	建议的用户应用程序操作
100 继续	在 WEB SEND 命令中使用了 ACTION (EXPECT) 选项，并且服务器接受全部消息发送。	CICS 通过发送消息体处理该响应。用户应用程序将不接收该状态码。
101 转换协议	不应使用。CICS Web Support 不支持协议升级。	用户应用程序不应接收该状态码。
200 正常	请求成功。正常的响应。	按计划继续处理响应。
201 已创建	您已请求创建资源并且已完成创建。	按计划继续处理响应。
202 已接受	服务器接受您的请求但还未执行处理。	按计划继续处理响应，但注意，不一定已提交您所作的任何更改，并且可能从不提交。
203 非权威信息	与消息体相关的头与服务器上的头不完全匹配。	按计划继续处理响应。

表 21. 作为 HTTP 客户机的 CICS: 处理有关响应的状态码 (续)

状态码和可能的原因短语	服务器为何要发送该状态码?	建议的用户应用程序操作
204 无内容	没有响应的消息体。	按计划继续处理响应，但注意，没有要接收的主体。
205 重置内容	服务器希望您清除导致发送请求的表单。	清除任何您用于发出请求的表单字段。
206 部分内容	您使用 Range 头字段发出请求并且它已成功。	按计划继续处理响应。
300 多个选项	资源的不同版本可用。	从提供的信息中选择您首选的版本，并发出新的请求。可能有 Location 头，它包含服务器的首选项的 URL。
301 已永久移开	已将资源永久移动到新的位置。	对服务器提供的 URL 发出新请求（可能在 Location 头中），并将该 URL 用于将来所有的请求。
302 已找到	已将资源临时移动到新的位置。	对服务器提供的 URL 发出新请求（可能在 Location 头中），但不将该 URL 用于将来的请求。
303 参阅其他内容	服务器希望您为另一个给出响应（特别是有关 POST 请求的结果的响应）的资源发出 GET 请求。	使用 GET 方法对服务器提供的 URL（可能在 Location 头中）发出新请求。
304 未修改	您发出了有条件的请求并且资源未更改。	参阅响应的现有存储版本以获取信息，但不要将它作为当前信息显示给用户，这是因为 CICS 不支持高速缓存。
305 使用代理	服务器希望您将指定的代理用于您的请求。	使用服务器提供的 URL（在 Location 头）发出新请求。
307 临时重定向	关于“302 已找到”。	关于“302 已找到”。
400 错误请求	您的请求的语法不正确。	检查请求，进行更改并再试一次。
401 未授权	服务器需要权限；或者，已拒绝您提供的权限。	请参阅第 153 页的『作为 HTTP 客户机的 CICS: 认证和标识』。
403 禁止	服务器拒绝您的请求。	不要重复请求。消息体可能包含有关为何拒绝请求的信息。
404 未找到	服务器未找到请求的 URL。	检查是否如您希望的指定请求。因为这种情况可能是暂时的，所以考虑稍后再试一次。
405 不允许的方法	您指定了该资源不支持的方法。	读响应中的 Allow 头以获取受支持的方法的列表，并在需要时使用其中某种方法发出新请求。
406 不可接受	您使用 Accept 头发出了请求，但服务器没有满足您的标准的资源版本。	检查消息体以获取有关服务器具有的资源的信息，并在需要时对其中某个资源发出新请求。
407 需要代理认证	代理服务器需要权限；或者，已拒绝您提供的权限。	请参阅第 153 页的『作为 HTTP 客户机的 CICS: 认证和标识』。
408 请求超时	服务器将不再等待您来完成您的请求。	需要时重复请求。检查您的应用程序是否未花较长的时间来汇编和发送消息。
409 冲突	已更改资源并且无法将您的请求应用到资源，因为它现在处于停止状态。	检查消息体以获取有关冲突原因的信息，并在需要时根据该信息发出新请求。
410 不可用	资源永久地不可用。	将来不要重复请求。
411 需要长度	服务器要求您提供 Content-Length 头。	除非您使用 TCPIPSERVICE 定义中的 USER 协议，否则 CICS 通常提供该头。如果是这种情况，那么自己写头并发出新请求。
412 预备条件失败	您发出了有条件的请求且条件未满足。	按计划继续处理，注意，未应用您的请求中指定的任何操作。

表 21. 作为 HTTP 客户机的 CICS: 处理有关响应的状态码 (续)

状态码和可能的原因短语	服务器为何要发送该状态码?	建议的用户应用程序操作
413 请求实体太大	您的消息体太大使得服务器无法处理。	读 Retry-After 头以查看这种情况是否是暂时的。您可以等待，或缩短消息体的长度并再试一次。您可能需要打开新连接。
414 请求 URI 太长	您的请求 URL 太长使得服务器无法处理。	检查请求并再试一次，或放弃请求。
415 不支持的介质类型	您发送了一个消息体，而它具有服务器（对于该资源）不支持的介质类型或内容编码。	检查您已指定的介质类型，并且如果出错，那么改正并重复请求。
416 无法满足请求的范围	您使用 Range 头字段发出请求，但该范围未出现在资源中。	读 Content-Range 头以查看资源的实际长度，并在需要时用合适的字节范围重复请求。
417 未达到期望的	在 WEB SEND 命令中使用了 ACTION (EXPECT) 选项，但服务器不接受全部消息发送。	您可以在不带 ACTION (EXPECT) 选项的情况下重复同一请求，但这可能会再次失败。检查是否正确指定请求，并且如果出错，那么改正并重复请求。
500 内部服务器错误	由于意外的错误，服务器无法处理请求。	因为这种情况可能是暂时的，所以考虑稍后再尝试请求一次。
501 未实施	服务器不支持该请求方法。	不要重复请求。
502 错误网关	您的请求已通过代理或网关，而它已接收到来自另一个服务器的无效响应。	因为这种情况可能是暂时的，所以考虑稍后再尝试请求一次，或许可能的话，避开该代理或网关。
503 服务不可用	服务器暂时无法处理请求。	读 Retry-After 头以查看这种情况是否是暂时的，如果是，那么过一会儿再试一次。
504 网关超时	您的请求已通过代理或网关，而它未接收到来自另一个服务器的及时响应。	需要时重复请求，或许可能的话，避开该代理或网关。
505 不支持的 HTTP 版本	不应使用。CICS Web Support 发送版本为 HTTP/1.1 的客户机请求。	用户应用程序不应接收该状态码。

附录 D. CICS Web Support 的 HTTP 方法参考

HTTP 请求包含方法，它是个关键字，说明了客户机希望服务器对请求中包含的资料执行的操作。CICS Web Support 实施 HTTP/1.1 规范定义的所有标准请求方法和早期 CICS 发行版中接受的某些其他方法。

要获取对方法的正确使用和对它们的正确响应操作的详细指导，并且要获取有关可应用需求的信息，请总是查询您正在遵从的 HTTP 规范。

- 对于从 HTTP/1.1 Web 客户机（当 CICS 是 HTTP 服务器时）接收的请求，会接受 HTTP/1.1 规范定义的标准方法。这些方法是 GET、HEAD、POST、PUT、TRACE、OPTIONS 和 DELETE。
- 对于从 HTTP/1.0 以及更低版本的 Web 客户机（当 CICS 是 HTTP 服务器时）接收的请求，会接受 HTTP/1.0 规范定义的方法和某些其他方法:
  - 由 HTTP/1.0 规范定义的方法是 GET、HEAD 和 POST。
  - 入站到 CICS 的 HTTP/1.0 请求上接受的其他方法是 PUT、DELETE、LINK、UNLINK 和 REQUEUE。
- 对于作为 HTTP 客户机的 CICS 发出的请求:
  - 可以使用 HTTP/1.1 规范定义的标准方法。这些方法是 GET、HEAD、POST、PUT、TRACE、OPTIONS 和 DELETE。
  - LINK、UNLINK 和 REQUEUE 方法不支持此目的。
  - 提供的请求版本总是 HTTP/1.1。
  - 某些 HTTP/1.0 服务器可以接受不在 HTTP/1.0 规范中定义的方法。HTTP/1.0 服务器应该对它无法接受的方法返回状态码“501 未实施”。
- 消息体适合某些请求方法而不适合其他请求方法。
  - 对于作为 HTTP 服务器的 CICS，应该知道某些客户机（特别是用户所写的客户机）可能对不适合消息体的方法发送消息体，您可以选择处理或忽略这一情况。
  - 对于作为 HTTP 客户机的 CICS，CICS 禁止对不适合消息体的方法发送消息体，而要求对适合的方法发送消息体。
- 当 CICS 是 HTTP 服务器时，对于从 Web 客户机接收的请求，CICS Web Support 根据方法和客户机的 HTTP 版本，采取一些操作对方法做出响应。
  - 具有最多方法的请求直接传递到应用程序以供处理。
  - CICS 为 OPTIONS 和 TRACE 方法自动返回合适的响应，而不调用用户应用程序。
  - 如果方法未在请求的 HTTP 版本实施，那么 CICS 对 Web 客户机返回一个错误响应，而不调用用户应用程序。
- 除了 HTTP 规范中定义的标准请求方法外，某些服务器可能还实施非标准请求方法（称为扩展方法）。
  - 对于作为 HTTP 服务器的 CICS，CICS Web Support 不接受 HTTP 协议上有非标准方法的请求。（在 CICS Transaction Server for z/OS V4R1 前，这些请求作为非 HTTP 接受和处理。）如果需要以非标准方法接收请求，可以通过用户定义的协议（TCPIPSERVICE 定义中的 USER 选项）来执行该操作，此时将不执行 HTTP 验收检查。

– 对于作为 HTTP 客户机的 CICS，不能在 EXEC CICS WEB API 命令中使用非标准方法。

本参考中的表列出了可能使用每种方法的情况。要获取有关本参考中提及的方法的更详细指导，请查询您正在遵从的 HTTP 规范。

作为 HTTP 服务器的 CICS: 从 Web 客户机接收的处理请求方法

表 22 显示 CICS 对请求方法执行的操作，以及对用户应用程序建议的操作。要获取详细指导和任何相关需求，检查您正在遵从的 HTTP 规范很重要。

表 22. 作为 HTTP 服务器的 CICS: 从 Web 客户机接收的请求方法

方法	HTTP/1.0 客户机的 CICS 操作	HTTP/1.1 客户机的 CICS 操作	消息体是否适合请求?	用户应用程序的适合操作
GET（资源的请求）	接受。请求传递到应用程序。	接受。请求传递到应用程序。	否	将资源发送到 Web 客户机，或发送说明为何无法执行该操作的错误响应。
HEAD（响应头的请求）	接受。请求传递到应用程序。	接受。请求传递到应用程序。	否	将资源发送到 Web 客户机，就好象对同一资源的 GET 请求做出响应一样。CICS 除去响应主体，只保留头。
POST（发送输入数据）	接受。请求传递到应用程序。	接受。请求传递到应用程序。	是	对方法的支持是可选的。抽出数据（可能是表单字段），处理它并将响应发送到 Web 客户机。也可用于更改或创建资源，在这种情况下，如同对 PUT 请求进行处理一样。
PUT（发送新的项）	接受。请求传递到应用程序。	接受。请求传递到应用程序。	是	对方法的支持是可选的。如果请求有效，那么按需使用消息内容创建具有指定 URL 的资源，或用消息内容替换现有资源。将应答发送到 Web 客户机。HTTP/1.1 规范对正确的操作有详细需求。 **提示:** 该请求类型可能不适合于您的 CICS Web Support 实施。需要的话，它可以通过为指定的 URL 创建 URIMAP 定义并存储要作为静态响应提供的资源来实现。
TRACE（请参阅请求的路径和最终状态）	拒绝，具有状态码 “501 未实施”。未调用用户应用程序。	接受。CICS 响应。未调用用户应用程序。	否	未传递到用户应用程序。CICS 返回的响应包含具有原始头和它需要的任何头（例如，via 头字段）的请求。
OPTIONS（对于服务器相关信息的请求）	拒绝具有状态码 400 的错误请求。未调用用户应用程序。	CICS 只支持不带路径的 OPTIONS（带有路径的 OPTIONS 将由于 405 而被拒绝）。注：接受 OPTIONS *。CICS 响应。未调用用户应用程序。	未定义	未传递到用户应用程序。CICS 返回具有基本信息（HTTP 版本和服务器软件描述）的响应。
DELETE（删除资源）	接受。请求传递到应用程序。	接受。请求传递到应用程序。	否	对方法的支持是可选的。如果请求有效，请删除现有资源，并将应答发送到 Web 客户机。

表 22. 作为 HTTP 服务器的 CICS: 从 Web 客户机接收的请求方法 (续)

方法	HTTP/1.0 客户机的 CICS 操作	HTTP/1.1 客户机的 CICS 操作	消息体是否适合请求?	用户应用程序的适合操作
LINK、UNLINK、REQUEUE	接受。请求传递到应用程序。	拒绝，具有状态码“501 未实施”。未调用用户应用程序。	未定义	不推荐使用，因为没有在 HTTP/1.1 规范中描述。为获取兼容性，仍将 HTTP/1.0 请求传递给应用程序。

作为 HTTP 客户机的 CICS: 使用对服务器请求的方法。

表 23 列出了 CICS API 对于 HTTP 客户机请求支持的请求方法，并总结了这些方法的正确使用。要获取正确使用每种方法的指导，以及应用到使用该方法的 HTTP 客户机的任何需求，请检查您正在遵从的 HTTP 规范。

表 23. 作为 HTTP 客户机的 CICS: 发送到服务器的请求方法。

方法	是否发送到 HTTP/1.0 服务器?	是否发送到 HTTP/1.1 服务器?	请求中是否有消息体?	用途
GET（资源的请求）	是	是	否	从服务器获取资源。
HEAD（响应头的请求）	是	是	否	从服务器获取资源的头。允许您检查资源的特征、状态或大小，而不必获取整个主体。
POST（发送输入数据）	是	是	是	将数据发送到服务器。例如，表单数据可能以这种方式发送。支持该方法不需要服务器。
PUT（发送新的项）	可能不受服务器支持。	是	是	创建或修改服务器上的资源。您的请求的 URL 是资源在服务器上具有的 URL。请求可用于更新现有项或新建一个项。支持该方法不需要服务器。
TRACE（请参阅请求的路径和最终状态）	可能不受服务器支持。	是	否	获取显示请求最终状态的响应及其带到服务器的路径（在 via 头字段中显示）。您可以查看正在使用什么代理服务器以处理您的请求。支持该方法不需要服务器。
OPTIONS（对于服务器相关信息的请求）	可能不受服务器支持。	是	允许，但是当前不打算为它定义。	获取有关服务器的信息。通过将 *（星号）指定为请求路径，将请求应用到整个服务器，或指定完整请求路径以获取有关该资源的信息。支持该方法不需要服务器。
DELETE（删除资源）	可能不受服务器支持。	是	否	删除服务器上的资源。请求 URL 是要删除的项的 URL。支持该方法不需要服务器。
通常是 LINK、UNLINK、REQUEUE 和扩展方法	不允许。返回的 INVREQ 响应和未发送的请求。	不允许。返回的 INVREQ 响应和未发送的请求。	未定义	对于作为 HTTP 客户机的 CICS，在 WEB API 上不可用。

附录 E. 分析器程序的参考信息

本部分提供分析器程序的参考信息，包括输入和输出参数，以及响应和原因码。

分析器程序参数摘要

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

分析器程序的参数和常量名（已转换成支持的不同编程语言的适当格式）在作为 CICS 的一部分提供的文件中定义。

语言	参数文件	常量文件
汇编程序	DFHWBTDD	DFHWBUCD
C	DFHWBTDH	DFHWBUCH
COBOL	DFHWBTDO	DFHWBUCO
PL/I	DFHWBTDL	DFHWBUCL

这些文件给出了有关通信区域中字段数据类型的特定于语言的信息。如果您使用这些文件，那么必须在“转换程序”步骤指定 XOPTS(NOLINKAGE)；否则会导致编译失败。

在下表中，参数的名称以缩写形式给出；表中每个参数名都必须使用 **wbra_** 作为前缀。

表 24. 分析器程序的参数

输入 wbra_	输入输出 wbra_	输出 wbra_
client_ip_address	**alias_tranid**	**application_style**
client_ipv6_address	**converter_program**	**alias_termid**
content_length	**server_program**	**characterset**
eyecatcher	**user_data_length**	**commarea**
function	**userid**	**dfhcnv_key**
hostname_length		**hostcodepage**
hostname_ptr		**reason**
http_version_length		**response**
http_version_ptr		**unescape**
method_ptr		**user_token**
method_length		
querystring_length		
querystring_ptr		
request_header_length		
request_header_ptr		
request_type		
resource_escaped_ptr		
resource_length		
resource_ptr		
server_ip_address		
server_ipv6_address		
urimap		
user_data_ptr		
version		

分析器程序的参数

分析器程序的参数名都附带了简短的说明，其中可以表明参数是仅输入参数、仅输出参数还是既可输入又可输出的参数。

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

wbra_alias_tranid
（输入和输出）

长度为 4 的字符串。用于覆盖该请求的剩余处理的别名事务的事务标识。如果涉及 URIMAP 定义，那么该字符串包含 TRANSACTION 属性的值。如果不设置该字段，或将其设置为空白，那么使用 CWBA。

wbra_alias_termid
（仅输出）

长度为 4 的字符串。用于别名事务的 START 请求的终端标识，该别名事务将覆盖该请求的剩余处理。

wbra_characterset
（仅输出）

客户机用于请求的实体主体的 IANA 字符集的名称。该信息用于请求和响应的实体主体的代码页转换。如果请求不是 HTTP 请求，那么该字符集用于转换整个请求和响应。还必须提供 **wbra_hostcodepage**。

wbra_client_ip_address

（仅输入）

如果未指定 **wbra_client_ipv6_address**，那么这是指定客户机的二进制 IPv4 地址的全字 32 位字段。**wbra_client_address** 不支持 IPv6 地址。

如果 **wbra_client_address** 中的值为非零，那么会使用该值，并会忽略 **wbra_client_ipv6_address** 中的任何值。 因此，如果您正在使用 IPv6 寻址，那么必须清除 **wbra_client_address** 中的内容，以便使用 **wbra_client_ipv6_address** 中的值。

wbra_client_ipv6_address

（仅输入）

正在使用 IPv6 寻址时，或正在使用 IPv4 寻址且未指定 **wbra_client_address** 时，必须设置的 16 字节字段。该字段支持 IPv4 和 IPv6 地址，并可设置为客户机的二进制 IPv6 地址或者客户机的 IPv6 格式的 IPv4 地址。有关 IP 地址格式的更多信息，请参阅 第 6 页的『IP 地址』。

wbra_commarea

（仅输出）

用于表明响应需要 CICS TS V3 之前版本的兼容性处理的标志，该响应使用不支持 Web 的应用程序和转换器程序。该标志意味着 Web 客户机接收的响应与它在 CICS TSV3 之前版本接收的响应相同。

wbra_content_length

（仅输入）

如所接收的数据中 Content-Length HTTP 头指定的，实体主体长度的 32 位二进制表示。

wbra_converter_program

（输入和输出）

长度为 8 的字符串。用于处理请求的转换器程序的名称。如果涉及 URIMAP 定义，那么该字符串包含 CONVERTER 属性的值。如果未在输出上设置该字段，那么不调用转换器程序。

wbra_dfhcnv_key

（仅输出）

长度为 8 的字符串。DFHCNV 表中转换模板的名称，该模板用于请求和响应的实体主体的代码页转换。如果请求不是 HTTP 请求，那么该模板用于转换整个请求和响应。

CICS 将该字段初始化为较大的值。如果使用该字段指定转换模板，那么必须在 DFHCNV 表中定义所选的名称，如第 51 页的『升级代码页转换表（DFHCNV）中的条目』中所描述。另外，您可以设置 **wbra_hostcodepage** 和 **wbra_characterset** 字段以指定用于代码页转换的代码页对。如果将 **wbra_dfhcnv_key** 设置为空或空的，且不设置 **wbra_hostcodepage** 和 **wbra_characterset**，那么禁止代码页转换。

wbra_eyecatcher

（仅输入）

长度为 8 的字符串。它的值为“>analyze”。

wbra_function

（仅输入）

表明正在调用分析器程序的代码。其值为 1。

wbra_hostcodepage

（仅输出）

适合正在处理请求的应用程序的主机代码页（IBM EBCDIC 代码页）的名称。该信息用于请求和响应的实体主体的代码页转换。如果请求不是 HTTP 请求，那么该代码页用于转换整个请求和响应。还必须提供 wbra_characterset。

wbra_hostname_length

（仅输入）

HTTP 请求中指定的主机名的长度（以字节为单位）。如果未指定主机名，那么表示未定义该值。

wbra_hostname_ptr

（仅输入）

指向主机名的指针，该主机名在客户机发送的 HTTP 请求中指定。如果将绝对 URI 用于请求，那么主机名从该 URI 获取。否则主机名在请求的 Host 头中指定。对于 HTTP/1.1 请求，必需主机名，因此该参数总是传递到分析器。对于 HTTP/1.0 请求，可能不提供主机名，在这种情况下不定义值。

wbra_http_version_length

（仅输入）

对于 HTTP 请求，标识客户机请求的 HTTP 版本的字符串长度（以字节为单位）。如果请求不是 HTTP 请求，其值为 0。

wbra_http_version_ptr

（仅输入）

对于 HTTP 请求，指向标识客户机请求的 HTTP 版本的字符串的指针。如果请求不是 HTTP 请求，那么不定义该值。

wbra_method_length

（仅输入）

对于 HTTP 请求，标识 HTTP 请求中所指定的方法的字符串的字节长度。如果请求不是 HTTP 请求，其值为 0。

wbra_method_ptr

（仅输入）

对于 HTTP 请求，指向 HTTP 请求中所指定的方法的指针。如果请求不是 HTTP 请求，那么不定义该值。

wbra_querystring_length

（仅输入）

HTTP 请求上指定的查询字符串的长度（以字节为单位）。如果未发送查询字符串，那么表示没有定义该值。

wbra_querystring_ptr

（仅输入）

指向查询字符串的指针，该查询字符串在客户机发送的 HTTP 请求中指定。如果未发送查询字符串，那么表示没有定义该值。

wbra_reason

（仅输出）

分析器程序返回的原因码。请参阅第 359 页的『响应和原因码』。

wbra_request_header_length

（仅输入）

对于 HTTP 请求，HTTP 请求中第一个 HTTP 头的长度。如果请求不是 HTTP 请求，其值为 0。

wbra_request_header_ptr

（仅输入）

对于 HTTP 请求，指向 HTTP 请求中第一个 HTTP 头的指针。在请求缓冲区域中，其他 HTTP 头紧跟该头。如果请求不是 HTTP 请求，那么不定义该值。

wbra_request_type

（仅输入）

如果该请求是 HTTP 请求，那么该值为 WBRA_REQUEST_HTTP。如果它不是 HTTP 请求，那么该值为 WBRA_REQUEST_NON_HTTP。

wbra_resource_escaped_ptr

（仅输入）

对于 HTTP 请求，指向尚未取消转义（即，仍采用其转义形式）的请求的 HTTP 头副本的指针。

wbra_resource_length

（仅输入）

对于 HTTP 请求，URL 的路径部分的长度（以字节为单位）。如果请求不是 HTTP 请求，其值为 0。

wbra_resource_ptr

（仅输入）

对于 HTTP 请求，指向 URL 的路径部分的指针。如果涉及 URIMAP 定义，那么该指针包含 PATH 属性的值。如果请求不是 HTTP 请求，那么不定义该值。

wbra_response

（仅输出）

分析器程序生成的响应值。请参阅第 359 页的『响应和原因码』。

wbra_server_ip_address

（仅输入）

如果未指定 **wbra_server_ipv6_address**，那么这是指定 HTTP 服务器的二进制 IPv4 地址的全字 32 位字段。**wbra_server_address** 不支持 IPv6 地址。

如果 **wbra_server_address** 中的值为非零，那么会使用该值，并会忽略 **wbra_server_ipv6_address** 中的任何值。 因此，如果您正在使用 IPv6 寻址，

那么必须清除 **wbra_server_address** 中的内容，以便使用 **wbra_server_ipv6_address** 中的值。

wbra_server_ipv6_address

（仅输入）

正在使用 IPv6 寻址时，或正在使用 IPv4 寻址且未指定 **wbra_client_address** 时，必须设置的 16 字节字段。该字段支持 IPv4 和 IPv6 地址，并可设置为服务器的二进制 IPv6 地址或者服务器的 IPv6 格式的 IPv4 地址。有关 IP 地址格式的更多信息，请参阅 第 6 页的『IP 地址』。

wbra_server_program

（输入和输出）

长度为 8 的字符串。用于处理请求的 CICS 应用程序的名称。如果涉及 URIMAP 定义，那么该字符串包含 PROGRAM 属性的值。程序名传递到 **wbra_converter_program** 中指定的任何转换器程序。如果没有设置该字段，那么所传的值为 null。必须在此处或由转换器程序来设置程序名；否则将不会调用任何 CICS 应用程序。

wbra_unescape

（仅输出）

- 要指定将未转义形式的数据传递到 CICS 应用程序，请将该参数设置为 WBRA_UNESCAPE_REQUIRED。
- 要指定以转义形式将数据传递给 CICS 应用程序，请将该参数设置为 WBRA_UNESCAPE_NOT_REQUIRED。该值为缺省值。

  如果分析器已经将数据转换为转义形式，那么也会将该参数设置为 WBRA_UNESCAPE_NOT_REQUIRED。

wbra_urimap

（仅输入）

请求的处理路径中涉及的任何匹配 URIMAP 定义的名称。如果该字段非空，那么 CICS 提供的缺省分析器 DFHWBADX 将返回未经处理的 URL 路径部分。

wbra_user_data_length

（输入和输出）

15 位整数，表示 HTTP 请求中实体主体的长度。如果请求是非 HTTP 请求，其值为请求的长度。传递给分析器的长度包括任何可能定界实体主体结尾的尾部回车和换行（CRLF）符。如果分析器减少了实体主体的长度，那么空字符 X'00' 将替换新的冗余字节。修改的值传递到字段 **wbbl_user_data_length** 中的 CICS 业务逻辑接口，并传递到字段 **decode_user_data_length** 中的转换器程序。

wbra_user_data_ptr

（仅输入）

对于 HTTP 请求，指向 HTTP 请求中实体主体的指针。如果请求不是 HTTP 请求，那么该指针指向该请求。

wbra_user_token

（仅输出）

作为 **decode_user_token** 传递到转换器程序的 64 位令牌。如果没有设置该字段，所传的值为空。如果没有该请求的转换器程序，那么忽略该值。

wbra_userid

（输入和输出）

长度为 8 的字符串。在输入时，该字符串包含由客户机提供的用户标识（使用基本认证或客户机证书认证），或者如果涉及 URIMAP 定义，那么该字符串包含 USERID 属性的值（如果指定）。在输出时，它包含用于别名事务的用户标识，这可以是提供的用户标识或分析器程序选择的用户标识。如果输出时该字段是空的或为空，那么使用 CICS 缺省用户标识。

wbra_version

（仅输入）

表明当前使用的参数列表的版本的半字二进制数。使用常量值 **wbra_current_version** 来设置该值。

响应和原因码

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

分析器程序必须在 **wbra_response** 中返回表中显示的一个值。

符号值	数值	说明
URP_OK	0	启动别名事务。
URP_EXCEPTION	4	未启动别名事务。Web 附加处理写异常跟踪项（跟踪点 4510），并发出消息 （DFHWB0523）。 如果请求是 HTTP 请求，那么将错误响应发送到 Web 客户机。缺省状态码是 400（错误请求），可以使用用户可替换 Web 出错程序 DFHWBEP 来配置它。 如果请求不是 HTTP 请求，那么不发送响应，且关闭套接字。
URP_INVALID	8	未启动别名事务。服务器控制器写异常跟踪项（跟踪点 4510），并发出消息（DFHWB0523）。 如果请求是 HTTP 请求，那么将错误响应发送到 Web 客户机。缺省状态码是 400（错误请求），可以使用用户可替换 Web 出错程序 DFHWBEP 来配置它。 如果请求不是 HTTP 请求，那么不发送响应，且关闭套接字。

符号值	数值	说明
URP_DISASTER	12	未启动别名事务。CICS 写异常跟踪项（跟踪点 4510），并发出消息（DFHWB0523）。 如果请求是 HTTP 请求，那么将错误响应发送到 Web 客户机。缺省状态码是 400（错误请求），可以使用用户可替换 Web 出错程序 DFHWBEP 来配置它。 如果请求不是 HTTP 请求，那么不发送响应，且关闭套接字。

如果在 **wbra_response** 中返回任何其他值，那么服务器控制器写异常跟踪项（跟踪点 4510），并且发出消息（DFHWB0523）。如果请求是 HTTP 请求，那么将具有状态码 400（错误请求）的消息发送给 Web 客户机。如果请求不是 HTTP 请求，那么不发送响应，且关闭套接字。

在出错情况下，可以在 **wbra_reason** 中提供 32 位的原因码来提供进一步的信息。CICS Web Support 不对分析器程序返回的原因码执行任何操作，但是用户可替换 Web 出错程序 DFHWBEP 可使用它决定如何修改缺省响应。原因码在任何调用分析器程序所导致的跟踪项中输出，并在消息 DFHWB0523 中。

附录 F. 转换器程序的参考信息

本部分提供: 转换器程序的**解码**函数的参考信息，以及转换器程序的**编码**函数的参考信息。

传递到转换器程序的通信区域中的参数和常量名，转换为不同编程语言支持的适当格式，并在作为 CICS Web Support 的一部分提供的文件中定义。下表列出了各种语言的文件。

语言	参数文件	常量文件
汇编程序	DFHWBCDD	DFHWBUCD
C	DFHWBCDH	DFHWBUCH
COBOL	DFHWBCDO	DFHWBUCO
PL/I	DFHWBCDL	DFHWBUCL

这些文件给出了有关通信区域中字段数据类型的特定于语言的信息。如果使用这些文件，那么必须在“转换程序”步骤中指定 XOPTS(NOLINKAGE)；否则会导致编译失败。

转换器程序解码函数的参数列表

如果分析器程序 URIMAP 定义或 CICS 业务逻辑接口的调用者为请求指定了转换器程序名称，那么在调用为请求提供数据的用户编写应用程序之前先调用**解码**。

参数摘要

在下表中，参数的名称以缩写形式给出；表中每个参数名都必须使用 **decode_** 作为前缀。

表 25. 解码的参数

输入 decode_	输入输出 decode_	输出 decode_
client_address	data_ptr	output_data_len
client_ipv6_address	input_data_len	reason
client_address_string	server_program	response
client_ipv6_address_string	user_token	
eyecatcher		
entry_count		
function		
http_version_length		
http_version_ptr		
method_length		
method_ptr		
request_header_length		
request_header_ptr		
resource_length		
resource_ptr		
user_data_length		
user_data_ptr		
version		
volatile		

参数

decode_client_address

（仅输入）

如果未指定 **decode_client_ipv6_address**，必须设置为客户机的二进制 IPv4 地址的全字 32 位字段。**decode_client_address** 不支持 IPv6 地址。

如果 **decode_client_address** 中的值为非零，那么会使用该值，并会忽略 **decode_client_ipv6_address** 中的任何值。 因此，如果您正在使用 IPv6 寻址，那么必须清除 **decode_client_address** 中的内容，以便使用 **decode_client_ipv6_address** 中的值。

decode_client_ipv6_address

（仅输入）

正在使用 IPv6 寻址时，或正在使用 IPv4 寻址且未指定 **decode_client_address** 时必须设置的 16 字节字段。该字段支持 IPv4 和 IPv6 地址，并可设置为客户机的二进制 IPv6 地址或者客户机的 IPv6 格式的 IPv4 地址。有关 IP 地址格式的更多信息，请参阅 第 6 页的『IP 地址』。

decode_client_address_string

（仅输入）

以点十进制格式表示的客户机的 IPv4 地址。

decode_client_ipv6_address_string

（仅输入）

客户机的 IP 地址，如果是 IPv4 地址，那么使用点分十进制格式表示；如果是 IPv6 地址，那么使用逗号十六进制格式表示。该字段的最大长度可为 39 个字节。

decode_data_ptr
（输入和输出）

在输入时，它是指向客户机请求（可能已经由分析器程序修改）的指针，或是指向 **encode_data_ptr** 响应数据的指针（如果该调用是**编码**转换器函数返回的循环）。

对于输出，它是指向要传递到用户编写的应用程序的 COMMAREA 的指针。当 **decode_volatile** 的值为 **0** 时，请勿修改这个参数。

decode_entry_count
（仅输入）

一个计数，用于记录为当前 Web 请求输入**解码**转换器的次数。

decode_eyecatcher
（仅输入）

一个长度为 8 的字符串。其**解码**值为“>decode”。

decode_function
（仅输入）

一个半字长代码，被设置为常量值 **URP_DECODE**，表示正在调用**解码**。

decode_http_version_length
（仅输入）

标识客户机支持的 HTTP 版本的字符串的长度（以字节为单位）。如果请求不是 HTTP 请求，或者 **decode_entry_count** 大于 1，那么值为 0。

decode_http_version_ptr
（仅输入）

一个指针，指向标识客户机支持的 HTTP 版本的字符串。如果分析器修改了请求的这一部分，那么更改在该可见。如果 **decode_http_version_length** 为 0，那么表示没有定义该值。

decode_input_data_len
（输入和输出）

在输入时，这是由 decode_data_ptr 所指向的请求数据的长度（以字节为单位）。

decode_method_length
（仅输入）

HTTP 请求中指定的方法的长度（以字节为单位）。如果请求不是 HTTP 请求，或者 **decode_entry_count** 大于 1，那么值为 0。

decode_method_ptr
（仅输入）

指向 HTTP 请求中指定的方法的指针。如果分析器修改了请求的这一部分，那么更改在该可见。如果 **decode_method_length** 为 0，那么表示没有定义该值。

decode_output_data_len
（仅输出）

传递到用户编写的应用程序的 COMMAREA 的长度（以字节为单位），如指针 **decode_data_ptr** 所指示。如果没有设置该输出，那么缺省值为 32 KB。

decode_reason
（仅输出）

原因码；请参阅第 365 页的『响应和原因码』。

decode_request_header_length
（仅输入）

HTTP 请求中第一个 HTTP 头的长度。如果请求不是 HTTP 请求，或者 **decode_entry_count** 大于 1，那么值为 0。

decode_request_header_ptr
（仅输入）

指向 HTTP 请求中第一个 HTTP 头的指针。如果分析器程序修改了请求的这一部分，那么更改会在此处反映出来。如果 **decode_request_header_length** 为 0，那么表示没有定义该值。

decode_resource_length
（仅输入）

HTTP 请求中 URL 的路径部分长度（以字节为单位）。如果请求不是 HTTP 请求，或者 **decode_entry_count** 大于 1，那么值为 0。

decode_resource_ptr
（仅输入）

指向 HTTP 请求中 URL 的路径部分的指针。如果分析器程序修改了请求的这一部分，那么更改会在此处反映出来。如果 **decode_resource_length** 为 0，那么表示没有定义该值。

decode_response
（仅输出）

响应；请参阅第 365 页的『响应和原因码』。

decode_server_program
（输入和输出）

长度为 8 的字符串。输入时，它是 **wbra_server_program** 中的分析器提供的值，或者是由 CICS 业务逻辑接口的调用者提供的值。对于输出，它是为请求提供服务的用户编写的应用程序的名称。应用程序的名称必须在此处或在分析器程序中设置；如果未设置，将不会调用任何应用程序。

decode_user_data_length
（仅输入）

HTTP 请求的实体主体的长度（以字节为单位）。如果分析器程序修改了该值，那么已修改的值会在此处反映出来。如果请求中没有实体主体，那么长度为 0。如果请求不是 HTTP 请求，那么该值表示请求的长度。如果 **decode_entry_count** 大于 1，那么该值为零。

decode_user_data_ptr
（仅输入）

指向 HTTP 请求的任意一个实体主体的指针。如果分析器修改了请求的这一部分，那么更改在该可见。如果请求不具有实体主体，那么指针为 0。如果请求不是 HTTP 请求，那么该指针与 **decode_data_ptr** 值相同。如果 **decode_entry_count** 大于 1，那么表示没有定义该值。

decode_user_token

（输入和输出）

一个 64 位的令牌。输入时，它是由作为 **wbra_user_token** 的分析器提供的用户令牌，或者是 CICS 业务逻辑接口的调用者提供的用户令牌。输出时，令牌作为 **encode_user_token** 传到**解码**。

decode_version

（输入）

表明当前使用的参数列表的版本的半字二进制数。使用常量值 **decode_current_version** 来设置该值。

decode_volatile

（输入）

单字符代码，表示 **decode_data_ptr** 所指向的数据区域能否被替换。可能的值包括:

0 该区域是另一个通信区域的一部分，且不能被替换。

1 **decode_data_ptr** 指向的存储器可以被释放并被不同大小的工作区替换。

响应和原因码

您必须在 **decode_response** 返回下列值之一:

符号值	数值	说明
URP_OK	0	处理继续进行。如果请求了 CICS 应用程序，请运行该应用程序。否则，继续处理转换器程序的**编码**函数。

符号值	数值	说明
URP_EXCEPTION	4	采取的操作取决于原因码: • **CICS Web Support** – 1 (URP_SECURITY_FAILURE)。CICS 写异常跟踪条目（跟踪点 455A），并发出消息（DFHWB0121）。如果请求是 HTTP 请求，那么状态码 403 发送到 Web 客户机。如果请求不是 HTTP 请求，那么不发送响应，并且会关闭 TCP/IP 套接字。 – 2 (URP_CORRUPT_CLIENT_DATA)。CICS 写异常跟踪条目（跟踪点 4559），并发出消息（DFHWB0121）。如果请求是 HTTP 请求，那么将状态码 400 发送到 Web 客户机。如果请求不是 HTTP 请求，那么不发送响应，并且会关闭 TCP/IP 套接字。 – 任何其他值。CICS 写异常跟踪条目（跟踪点 455B），并发出消息（DFHWB0121）。如果请求是 HTTP 请求，那么将状态码 501 发送到 Web 客户机。如果请求不是 HTTP 请求，那么不发送响应，并且会关闭 TCP/IP 套接字。 • **CICS 业务逻辑接口** – 2 (URP_CORRUPT_CLIENT_DATA)。CICS 业务逻辑接口写异常跟踪条目（跟踪点 4556），发出消息（DFHWB0120），并将响应 400 返回给它的调用者。 – 任何其他值。CICS 写异常跟踪条目（跟踪点 455B），发出消息（DFHWB0121），并将响应 501 返回给它的调用者。 CICS 应用程序和转换器程序的**编码**函数都未在运行。
URP_INVALID	8	CICS 应用程序和转换器程序的**编码**函数都未在运行。 • **CICS Web Support** – CICS 写异常跟踪条目（跟踪点 455C），并发出消息（DFHWB0121）。如果请求是 HTTP 请求，那么将状态码 501 发送到 Web 客户机。如果请求不是 HTTP 请求，那么不发送响应，并且会关闭 TCP/IP 套接字。 • **CICS 业务逻辑接口** – CICS 写异常跟踪条目（跟踪点 455C），发出消息（DFHWB0121），并将响应 501 返回给它的调用者。

符号值	数值	说明
URP_DISASTER	12	CICS 应用程序和解码器的**编码**函数都未在运行。 • **CICS Web Support** – CICS 写异常跟踪条目（跟踪点 455D），并发出消息（DFHWB0121）。如果请求是 HTTP 请求，那么将状态码 501 发送到 Web 客户机。如果请求不是 HTTP 请求，那么不发送响应，并且会关闭 TCP/IP 套接字。 • **CICS 业务逻辑接口** – CICS 写异常跟踪条目（跟踪点 455D），发出消息（DFHWB0121），并将响应 501 返回给它的调用者。
任何其他值		CICS 应用程序和解码器的**编码**函数都未在运行。 • **CICS Web Support** – CICS 写异常跟踪条目（跟踪点 455E），并发出消息（DFHWB0121）。如果请求是 HTTP 请求，那么将状态码 500 发送到 Web 客户机。如果请求不是 HTTP 请求，那么不发送响应，并且会关闭 TCP/IP 套接字。 • • **CICS 业务逻辑接口** – CICS 写异常跟踪条目（跟踪点 455E），发出消息（DFHWB0121），并将响应 501 返回给它的调用者。

您可以在 **decode_reason** 中提供一个 32 位的原因码以便在出错情况下提供更多信息。CICS Web Support 和 CICS 业务逻辑接口都不对**解码**返回的原因码执行任何操作，以上在 URP_EXCEPTION 中提到的原因码除外。该原因码包含在**解码**调用生成的任何跟踪条目中。

转换器程序编码函数的参数列表

参数摘要

下表中，参数的名称以缩写的格式给出：表中每个名称都必须以 **encode_** 作为前缀来给出参数名。

表 26. 编码的参数

输入 encode_	输入输出 encode_	输出 encode_
eyecatcher **entry_count** **function** **input_data_len** **user_token**	**data_ptr**	**reason** **response**

函数

如果分析器程序或 CICS 业务逻辑接口的调用者为请求指定了转换器程序名，那么**编码**在用户编写的应用程序结束后调用。它使用应用程序返回的通信区域中的数据构造响应。

参数

encode_data_ptr

（输入和输出）

输入时，这是指向由 CICS 应用程序返回的通信区域的指针。如果没有调用应用程序，那么这是指向转换器程序的**解码**函数创建的通信区域的指针。

对于输出，如果使用转换器程序在 CICS 存储器缓冲区中手工构造了 HTTP 响应，然后将它发送给 Web 客户机，那么这是指向包含响应的缓冲区的指针。必须确保指针指向有效位置，否则结果将不可预测。缓冲区域必须是双字对齐。头 4 个字节必须是 32 位的无符号数，用来指定缓冲区域的长度。（在 COBOL 中，将其指定为 PIC 9(8) COMP。）缓冲区域的余下部分则为响应。

如果转换器程序使用了 EXEC CICS WEB API 命令来发送响应，那么 CICS 忽略并丢弃该指针指示的任何存储器块。在这种情况下，指针可以保留为指向 CICS 应用程序返回的 COMMAREA 的指针；其设置无关紧要。

从 CICS 业务逻辑接口（它以偏移量方式调用）调用转换器时，请不要将该字段用作输出。

encode_entry_count

（仅输入）

一个计数，表示为当前 Web 请求调用转换器程序**编码**函数的次数。

encode_eyecatcher

（仅输入）

一个长度为 8 的字符串。它的**编码**的值为“>encode ”。

encode_function

（仅输入）

一个半字长代码，被设置为常量值 **URP_ENCODE**，表示正在调用**编码**。

encode_input_data_len

（仅输入）

通信区域的长度由**译码**在 **decode_output_data_len** 中指定。

编码原因

（仅输出）

原因码（请参阅第 369 页的『响应和原因码』）。

encode_response

（仅输出）

响应（请参阅第 369 页的『响应和原因码』）。

encode_user_token

（仅输入）

解码函数以 **decode_user_token** 形式输出的 64 位令牌。

encode_version

（输入）

一个单字符的参数列表版本标识，随参数列表布局的改变而改变。它的值可以是表明 CICS TS 1.3 之前版本的参数列表的二进制零（X'00'），或表明 CICS TS 1.3 或更高版本参数列表的字符零（X'F0'）。

encode_volatile

（输入）

单字符代码，表示 **encode_data_ptr** 所指向的数据区域能否被替换。可能的值为:

0 该区域是另一个通信区域的一部分，且不能被替换。

1 **encode_data_ptr** 指向的存储器可以被释放并被不同大小的工作区域替换。

响应和原因码

必须在 **encode_response** 中返回下列值之一:

符号值	数值	说明
URP_OK	0	**encode_data_ptr** 所指向的缓冲区中的响应将发送到客户机，除非 EXEC CICS WEB API 命令已经用于发送响应。
URP_DISASTER	12	**CICS Web Support** • CICS 写异常跟踪条目（跟踪点 455D），并发出消息（DFHWB0122）。如果请求是 HTTP 请求，那么状态码 501 发送到 Web 客户机。如果请求不是 HTTP 请求，那么不发送响应，并且会关闭 TCP/IP 套接字。 **CICS 业务逻辑接口** • CICS 写异常跟踪条目（跟踪点 455D），发出消息（DFHWB0122），并将响应 501 返回给它的调用者。
URP_OK_LOOP	16	CICS 返回到**解码**函数的开头。存储在 **encode_user_token** 中的值被复制到 **decode_user_token** 为**解码**转换器函数所用。
任何其他值		**CICS Web Support** • CICS 写异常跟踪条目（跟踪点 455E），并发出消息（DFHWB0122）。如果请求是 HTTP 请求，那么状态码 501 发送到 Web 客户机。如果请求不是 HTTP 请求，那么不发送响应，并且会关闭 TCP/IP 套接字。 **CICS 业务逻辑接口** • CICS 写异常跟踪条目（跟踪点 455E），发出消息（DFHWB0122），并将响应 501 返回给它的调用者。

您需要在 **encode_reason** 中提供一个 32 位的原因码以便在出错情况下提供更多信息。CICS Web Support 和 CICS 业务逻辑接口都不对**编码**函数返回的原因码执行任何操作。原因码在**编码**函数调用生成的所有跟踪项中输出。

附录 G. DFHWBBLI CICS 业务逻辑接口的参考信息

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

业务逻辑接口允许调用者指定在 CICS 应用程序之前和之后要执行的显示逻辑。它有两种操作方式:

- 指针方式: **解码**的输入数据位于同业务逻辑接口通信区域分开分配的存储器中。通信区域包含指向**解码**的输入数据的指针（**wbbl_data_ptr**）。当对业务逻辑接口的调用结束时，来自**编码**的输出位于与业务逻辑接口的通信区域分开分配的存储器中，并且通信区域包含指向来自**编码**的输出的指针（**wbbl_outdata_ptr**）。
- 位移方式: **解码**的输入数据是业务逻辑接口通信区域的一部分。通信区域包含了**解码**的输入数据的偏移量（**wbbl_data_offset**）。当调用业务逻辑接口结束时，**编码**的输出是业务逻辑接口通信区域的一部分，并且通信区域包含来自**解码**的输出的偏移量（**wbbl_outdata_offset**）。

业务逻辑接口的调用者使用 **wbbl_mode** 来指示要使用的操作方式。

有关为业务逻辑接口编写转换器的信息，请参阅第 124 页的『编写转换器程序』。

参数摘要

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

在 CICS Web Support 随附的文件中定义了参数和常量的名称，这些名称已转换为各种受支持的编程语言所需的对应格式。这些文件给出了有关通信区域中字段数据类型的特定于语言的信息。

各种语言的文件如下所示:

语言	文件
汇编程序	DFHWBBLD
C	DFHWBBLH
COBOL	DFHWBBLO
PL/I	DFHWBBLL

表 27. 业务逻辑接口参数

输入	输出
wbbl_client_address	**wbbl_outdata_length**
wbbl_client_ipv6_address	**wbbl_outdata_offset**
wbbl_client_address_length	**wbbl_outdata_ptr**
wbbl_client_ipv6_address_length	
wbbl_client_address_string	
wbbl_client_ipv6_address_string	
wbbl_converter_program_name	
wbbl_eyecatcher	
wbbl_header_length	
wbbl_header_offset	
wbbl_http_version_length	
wbbl_http_version_offset	
wbbl_indata_length	
wbbl_indata_offset	
wbbl_indata_ptr	
wbbl_length	
wbbl_method_length	
wbbl_method_offset	
wbbl_mode	
wbbl_prolog_size	
wbbl_resource_length	
wbbl_resource_offset	
wbbl_server_address	
wbbl_server_ipv6_address	
wbbl_server_program_name	
wbbl_ssl_keysize	
wbbl_status_size	
wbbl_user_token	
wbbl_user_data_length	
wbbl_vector_size	
wbbl_version	

使用早期版本的 CICS 业务逻辑接口（DFHWBA1）编写的程序，由调用 DHWBBLI 的兼容性接口支持。

用于业务逻辑接口的参数，DFHWBBLI

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

下面列出了用于业务逻辑接口的、前缀为 **wbbl_** 的输入和输出参数的名称与描述。

在将输入插入通信区域中前，必须将它清为二进制零。

wbbl_eyecatcher
（仅输入）

必须设置为字符串 **>DFHWBBLIPARMS** 的 14 字符字段。

wbbl_client_address
（仅输入）

如果未指定 **wbbl_client_ipv6_address**，必须设置为客户机的二进制 IPv4 地址的全字 32 位字段。**wbbl_client_address** 不支持 IPv6 地址。

如果 **wbbl_client_address** 中的值为非零，那么会使用该值，并会忽略 **wbbl_client_ipv6_address** 中的任何值。 因此，如果您正在使用 IPv6 寻址，那么必须清除 **wbbl_client_address** 中的内容，以便使用 **wbbl_client_ipv6_address** 中的值。

wbbl_client_ipv6_address
（仅输入）

正在使用 IPv6 寻址时，或正在使用 IPv4 寻址且未指定 **wbbl_client_address** 时，必须设置的 16 字节字段。该字段支持 IPv4 和 IPv6 地址，并可设置为客户机的二进制 IPv6 地址或者客户机的 IPv6 格式的 IPv4 地址。有关 IP 地址格式的更多信息，请参阅 第 6 页的『IP 地址』。

wbbl_client_address_length
（仅输入）

必须设置为 **wbbl_client_address_string** 的长度的 1 字节二进制字段。

wbbl_client_ipv6_address_length
（仅输入）

必须设置为 **wbbl_client_ipv6_address_string** 的长度的 1 字节二进制字段。

wbbl_client_address_string
（仅输入）

一个最多由 15 个字符组成的字符串，该字符串是 **wbbl_client_address** 的点分十进制表示，并在右端填充二进制零。将 **wbbl_client_ipv6_address_string**（而不是 **wbbl_client_address_string**）用于所有的新程序。

wbbl_client_ipv6_address_string
（仅输入）

一个最多由 39 个字符组成的字符串，该字符串是 **wbbl_client_ipv6_address** 的冒号十六进制或点分十进制表示，并在右端填充二进制零。

wbbl_converter_program_name
（仅输入）

要用于转换器 DECODE 和 ENCODE 函数的程序的 8 字符名称。

wbbl_header_length
（仅输入）

必须包含与该请求相关联的 HTTP 头长度的全字二进制数。

wbbl_header_offset
（仅输入）

必须包含与该请求相关联的 HTTP 头的偏移量（从请求数据的开始处起算）的全字二进制数。

wbbl_http_version_length
（仅输入）

必须包含用于处理请求的 HTTP 协议的版本长度的全字二进制数。

wbbl_http_version_offset
（仅输入）

必须包含用于处理请求的 HTTP 协议的版本的偏移量的全字二进制数。

wbbl_indata_length
（仅输入）

必须设置为由 **wbbl_indata_ptr** 或 **wbbl_indata_offset** 定位的数据的长度的全字二进制数。如果分析器修改了数据长度值，那么修改部分会在此处反映出来。如果请求不是 HTTP 请求，请不要设置该字段。

wbbl_indata_offset
（仅输入）

如果 **wbbl_mode** 是“O”，那么该字段是传递给应用程序的 HTTP 请求数据的偏移量（从参数列表的开始处起算）。

wbbl_indata_ptr
（仅输入）

如果 **wbbl_mode** 是“P”，那么该字段就是传递给应用程序的 HTTP 请求数据的地址。

wbbl_length
（仅输入）

必须设置成为 BLI 参数列表的总长度的半字二进制数。

wbbl_method_length
（仅输入）

必须包含用于处理请求的 HTTP 方法的长度的全字二进制数。该方法可以是: GET、POST、HEAD、PUT、DELETE、LINK、UNLINK 或 REQUEUE。

wbbl_method_offset
（仅输入）

必须包含用于处理请求的 HTTP 方法的偏移量（从请求数据的开始处起算）的全字二进制数。该方法可以是:
GET、POST、HEAD、PUT、DELETE、LINK、UNLINK 或 REQUEUE。

wbbl_mode
（仅输入）

表明 **wbbl_indata** 和 **wbbl_outdata** 寻址方式的单个字符。必须将其设置为“P”以表明这些值都是指针，或设置为“O”以表明这些值是从参数列表的开始处起算的偏移量。

wbbl_outdata_length
（仅输入）

全字二进制字段，DFHWBBLI 在该字段中返回由 **wbbl_outdata_ptr** 或 **wbbl_outdata_offset** 定位的响应数据的长度。

wbbl_outdata_offset
（仅输入）

如果 **wbbl_mode** 是“O”，那么该字段是全字，DFHWBBLI 在其中返回来自应用程序的响应数据的偏移量（从参数列表的开始处起算）。该地址不一定要和 **wbbl_indata_offset** 相同。

wbbl_outdata_ptr
（仅输入）

如果 **wbbl_mode** 是“P”，那么该字段是全字地址，DFHWBBLI 在其中返回来自应用程序的响应数据的地址。该地址不一定要和 **wbbl_indata_ptr** 相同。

wbbl_prolog_size
（仅输入）

必须设置为 56 的半字二进制数；即，**wbbl_prolog** 子结构的长度。

wbbl_resource_length
（仅输入）

一个全字二进制数，它必须包含被请求的 URI 资源的长度；即，URL 中第一个“/”字符开始的 URL 非网络部分。

wbbl_resource_offset
（仅输入）

一个全字二进制数，它必须包含正请求的 URI 资源的偏移量（从请求数据的开始处起算）；即，URL 中第一个“/”字符开始的 URL 非网络部分。

wbbl_response
（仅输入）

DFHWBBLI 在其中返回响应代码的全字二进制字段。

wbbl_server_address
（仅输入）

如果未指定 **wbbl_server_ipv6_address**，必须设置为服务器的二进制 IPv4 地址的全字 32 位字段。**wbbl_server_address** 不支持 IPv6 地址。

如果 **wbbl_server_address** 中的值为非零，那么会使用该值，并会忽略 **wbbl_server_ipv6_address** 中的任何值。 因此，如果您正在使用 IPv6 寻址，那么必须清除 **wbbl_server_address** 中的内容，以便使用 **wbbl_server_ipv6_address** 中的值。

wbbl_server_ipv6_address
（仅输入）

正在使用 IPv6 寻址时，或正在使用 IPv4 寻址且未指定 **wbbl_client_address** 时，必须设置的 16 字节字段。该字段支持 IPv4 和 IPv6 地址，并可设置为服务器的二进制 IPv6 地址或者服务器的 IPv6 格式的 IPv4 地址。有关 IP 地址格式的更多信息，请参阅 第 6 页的『IP 地址』。

wbbl_server_program_name
（仅输入）

用于处理请求并生成响应的 CICS 应用程序的名称（8 个字符）。

wbbl_ssl_keysize
（仅输入）

加密密钥的大小，如果使用安全套接字层，那么该大小在 SSL 握手期间协商确定。如果不使用 SSL，那么其值为零。

wbbl_status_size
（仅输入）

必须设置为 **wbbl_status** 子结构的长度的 1 字节二进制字段。

wbbl_user_data_length
（仅输入）

必须设置为实体主体长度的全字二进制数。如果分析器修改了长度值，那么修改部分会在此处反映出来。如果请求不是 HTTP 请求，请不要设置该字段。

wbbl_user_token
（仅输入）

一个 8 字符字段，DFHWBBLI 的调用者使用该字段可以传输标识客户机当前对话状态的数据。通常将它设置为 URL 的 **query-string** 部分的前 8 个字符；即，问号（？）以后的所有数据。

wbbl_vector_size
（仅输入）

必须设置为 64（即，**wbbl_vector** 子结构的长度）的半字二进制数。

wbbl_version
（仅输入）

表明当前使用的 BLI 参数列表的版本的半字二进制数。使用常量值 **wbbl_current_version** 来设置该值。

业务逻辑接口响应

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

wbbl_response 中返回下列值之一。这些值同将要发送给 HTTP 客户机的 HTTP 响应相一致。

400 某个转换器函数返回了带有 URP_CORRUPT_CLIENT_DATA 原因码的 URP_EXCEPTION 响应。业务逻辑接口写异常跟踪项（跟踪点 4556），并发出消息（DFHWB0120）。

403 对 **wbbl_server_program_name** 中指定的程序的 LINK 命令接收到 NOTAUTH 响应。业务逻辑接口写异常跟踪项（跟踪点 4556），并发出消息（DFHWB0120）。

404 对 **wbbl_server_program_name** 中指定的程序的 LINK 命令接收到 PGMIDERR 响应。业务逻辑接口写异常跟踪项（跟踪点 4556），并发出消息（DFHWB0120）。

500 发生以下某种情况:

- 业务逻辑接口检测到异常终止。根据异常终止的程序发出相应的消息。对于在 **wbbl_server_program_name** 中指定的程序，消息为 DFHWB0125。对于转换器的**编码**函数，消息为 DFHWB0126。对于转换器的**解码**函数，消息为 DFHWB0127。对于任何其他程序，消息为 DFHWB0128。任何情况下，都会写入异常跟踪项（跟踪点 4557）。

- 对 **wbbl_server_program_name** 中指定的程序的 LINK 命令接收到 INVREQ 或 LENGERR 或意外响应。业务逻辑接口写异常跟踪项（跟踪点 4556），并发出消息（DFHWB0120）。

501 发生以下某种情况:

- **解码**返回了 URP_EXCEPTION 响应，并带有未定义的原因码。业务逻辑接口写异常跟踪项（跟踪点 455B），并发出消息（DFHWB0121）。
- **解码**返回了 URP_INVALID 响应。业务逻辑接口写异常跟踪项（跟踪点 455C），并发出消息（DFHWB0121）。
- **解码**返回了 URP_DISASTER 响应。业务逻辑接口写异常跟踪项（跟踪点 455D），并发出消息（DFHWB0121）。
- **解码**返回了未定义响应。业务逻辑接口写异常跟踪项（跟踪点 455E），并发出消息（DFHWB0121）。
- **编码**返回了 URP_EXCEPTION 响应，并带有未定义的原因码业务逻辑接口写异常跟踪项（跟踪点 455B），并发出消息（DFHWB0122）。
- **编码**返回了 URP_INVALID 响应。业务逻辑接口写异常跟踪项（跟踪点 455C），并发出消息（DFHWB0122）。
- **编码**返回了 URP_DISASTER 响应。业务逻辑接口写异常跟踪项（跟踪点 455D），并发出消息（DFHWB0122）。
- **编码**返回了未定义响应。业务逻辑接口写异常跟踪项（跟踪点 455E），并发出消息（DFHWB0122）。

503 发生以下某种情况:

- 对 **wbbl_server_program_name** 中指定的程序的 LINK 命令接收到 TERMERR 响应。业务逻辑接口写异常跟踪项（跟踪点 4555），并发出消息（DFHWB0120）。
- 对 **wbbl_server_program_name** 中指定的程序的 LINK 命令接收到 SYSIDERR 或 ROLLEDBACK 响应。业务逻辑接口写异常跟踪项（跟踪点 4556），并发出消息（DFHWB0120）。

附录 H. Web 出错程序 DFHWBEP 的参考信息

警告: 本主题包含产品敏感的编程接口和相关的指导信息。

下表列出了参数列表中的参数和常量名称，这些参数和常量传递到 Web 出错程序 DFHWBEP，并转换成所支持的编程语言的对应格式。

语言	参数文件
汇编程序	DFHWBEPD
C	DFHWBEPH
COBOL	DFHWBEPO
PL/I	DFHWBEPL

参数

wbep_response_ptr 和 **wbep_response_len** 既可输入又可输出，而 **wbep_suppress_abend** 和 **wbep_close_conn** 只可输出，除了它们以外的所有 DFHWBEP 参数都只可输入。

wbep_abend_code
（仅输入）

与该异常关联的 8 字符异常终止代码。

wbep_activity
（仅输入）

发生错误时进行的处理类型。0 表明服务器处理，而 2 表明管道处理。

wbep_analyzer_reason
（仅输入）

分析器程序返回的原因码 – 如果调用了分析器程序。

wbep_analyzer_response
（仅输入）

分析器程序返回的响应码 – 如果调用了分析器程序。

wbep_client_address
（仅输入）

如果未指定 **wbep_client_ipv6_address**，必须设置为客户机的二进制 IPv4 地址的 15 字符字段。**wbep_client_address** 不支持 IPv6 地址。

如果 **wbep_client_address** 中的值为非零，那么会使用该值，并会忽略 **wbep_client_ipv6_address** 中的任何值。 因此，如果您正在使用 IPv6 寻址，那么必须清除 **wbep_client_address** 中的内容，以便使用 **wbep_client_ipv6_address** 中的值。

wbep_client_ipv6_address
（仅输入）

客户机的冒号十六进制 IPv6 地址或点分十进制 IPv4 地址。该字段的最大长度可为 39 个字符。

正在使用 IPv6 寻址时，或正在使用 IPv4 寻址且未指定 **wbep_client_address** 时，必须设置该字段。该字段支持 IPv4 和 IPv6 地址，并可设置为客户机的二进制 IPv6 地址或者客户机的 IPv6 格式的 IPv4 地址。有关 IP 地址格式的更多信息，请参阅 第 6 页的『IP 地址』。

wbep_client_address_len
（仅输入）

wbep_client_address 中包含的点分十进制 IP 地址的长度。如果该地址为 IPv6 格式，那么该字段将包含零。

wbep_client_ipv6_address_len
（仅输入）

wbep_client_ipv6_address 或 **wbep_client_address** 中包含的 IP 地址的长度。

wbep_close_conn
（仅输出）

单字符字段（Y 或 N），表明在将响应发送给客户机之后，连接是否关闭。缺省值 N 表明连接未关闭。

wbep_converter_program
（仅输入）

用于失败请求的转换器程序名称 - 如果使用了转换器程序。

wbep_converter_reason
（仅输入）

转换器程序返回的原因码（如果调用了转换器程序）。

wbep_converter_response
（仅输入）

转换器程序返回的响应码（如果调用了转换器程序）。

wbep_error_code
（仅输入）

错误代码，标识检测到的错误。

wbep_eyecatcher
（仅输入）

包含有助于诊断的 eyecatcher 的单字符字段。DFHWBA 在调用 Web 出错程序以前将它设置成 `>wbepca`。

wbep_failing_program
（仅输入）

包含发生错误的程序名称的 8 字符字段。

wbep_http_response_code
（仅输入）

CICS 为该错误返回的缺省 HTTP 状态码。

wbep_length
（仅输入）

DFHWBEPC 副本的长度。

wbep_message_len
（仅输入）

wbep_message_ptr 确定的 CICS 消息文本长度。

wbep_message_number
（仅输入）

与该错误关联的 CICS WB 域消息的全字数。

wbep_message_ptr
（仅输入）

指向与该错误关联的 CICS 消息文本的指针。

wbep_response_len
（输入和输出）

在输入时，该字段是该错误的缺省 HTTP 响应的全字长度。CICS 仅为 HTTP 请求提供缺省响应；对于非 HTTP 请求，该字段为零。在输出时，该字段包含缺省 HTTP 响应的长度、同一存储器块中已修改响应的长度，或新存储器块中替换响应的长度。

wbep_response_ptr
（输入和输出）

在输入时，这是指向包含该错误的缺省 HTTP 响应的存储器块的指针。CICS 仅为 HTTP 请求提供缺省响应。缺省响应是一个完整的 HTTP 响应，包括状态行、HTTP 头和消息体。在输出时，这是指向包含原始或已修改的缺省响应的同一存储器块的指针，或者是指向包含替换响应的新存储器块的指针。如果 DFHWBEP 成功地使用了 EXEC CICS WEB SEND 命令来创建新的响应并将其发送到 Web 客户机，那么 CICS 会忽略并丢弃存储器块中的 HTTP 响应。否则，会将存储器块中的响应发送到 Web 客户机。

wbep_server_address
（仅输入）

如果未指定 **wbep_server_ipv6_address**，那么必须设置为服务器（作为 HTTP 服务器的 CICS）的 IPv4 地址（由 15 个字符组成）。**wbep_server_address** 不支持 IPv6 地址。

如果 **wbep_server_address** 中的值为非零，那么会使用该值，并会忽略 **wbep_server_ipv6_address** 中的任何值。 因此，如果您正在使用 IPv6 寻址，那么必须清除 **wbep_server_address** 中的内容，以便使用 **wbep_server_ipv6_address** 中的值。

wbep_server_ipv6_address
（仅输入）

正在使用 IPv6 寻址时，或正在使用 IPv4 寻址且未指定 **wbep_server_address** 时必须设置的 16 字节字段。该字段支持 IPv4 和 IPv6 地址，并且设置为服务

器（作为 HTTP 服务器的 CICS）的二进制 IPv6 地址或者服务器的 IPv6 格式的 IPv4 地址。有关 IP 地址格式的更多信息，请参阅 第 6 页的『IP 地址』。

wbep_server_address_len
（仅输入）

wbep_server_address 中包含的点分十进制 IPv4 地址的长度。如果该地址为 IPv6 格式，那么该字段将包含零。

wbep_server_ipv6_address_len
（仅输入）

wbep_server_ipv6_address 或 **wbep_server_address** 中包含的 IP 地址的长度。

wbep_target_program
（仅输入）

指定为处理 Web 客户机请求的目标用户编写应用程序。

wbep_tcpipservice_name
（仅输入）

接收请求的端口的 TCPIPSERVICE 定义名。

wbep_version
（仅输入）

表明当前使用的参数列表的版本的半字二进制数。使用常量值 **wbep_current_version** 来设置该值。

wbep_suppress_abend
（仅输出）

1 位标志，当设置为 on 时表示禁止异常终止 AWBM。

附录 I. DFHWBCLI Web 客户机接口

DFHWBCLI 是 CICS 提供的实用程序，您可以通过 EXEC CICS LINK 调用它，以提供 Web 客户机服务或出站 HTTP。在 CICS Transaction Server for z/OS V4R1 中，它仅用于升级目的。

出于兼容性原因保留了 DFHWBCLI Web 客户机接口的函数。为了获得增强的功能，您可以升级使用 DFHWBCLI 接口的 HTTP 客户机应用程序，以针对客户机请求使用 EXEC CICS WEB API 命令（具有 SESSTOKEN 选项）。一个值得注意的重要区别是，在 EXEC CICS WEB API 中，代理服务器的使用由 WEB OPEN 命令上的用户出口（XWBOPEN）指定，而代理服务器的 URL 由该用户出口提供。第 129 页的『通过作为 HTTP 客户机的 CICS 发出 HTTP 请求』描述了现在如何发出 HTTP 客户机请求。

如果要使用 DFHWBCLI，那么必须设置 CICS 以使用名称服务器。请参阅第 49 页的第 4 章，『配置 CICS Web Support 基本组件』。

要使用 DFHWBCLI，您必须使用包含下列副本映射其内容的参数列表的通信区域链接到它:

- DFHWBCLD for Assembler
- DFHWBCLO for Cobol
- DFHWBCLL for PL/I
- DFHWBCLH for C

这些参数具有以下含义:

WBCLI_VERSION_NO
一字节二进制数，其指定该参数列表的版本号。它应该设置为符号常量 WBCLI_VERSION_CURRENT 指定的值。

WBCLI_FUNCTION
一字节二进制数，其指定您要 DFHWBCLI 执行的函数。它应该设置为下列值之一:

0 (WBCLI_FUNCTION_CONVERSE)
发送 HTTP 请求到目标服务器，并接收相应的响应

1 (WBCLI_FUNCTION_SEND)
发送 HTTP 请求到目标服务器，并返回控制，而无须等待响应

2 (WBCLI_FUNCTION_RECEIVE)
等待并接收对前一个 SEND 函数所发送 HTTP 请求的响应

3 (WBCLI_FUNCTION_INQUIRE_PROXY)
请求 `INITPARM=(DFHWBCLI,'PROXY=http://....')` 系统初始化参数中指定的代理服务器的名称

4 (WBCLI_FUNCTION_CLOSE)
关闭先前由 SEND 函数建立的连接，但是无须等待 HTTP 响应

WBCLI_METHOD
一字节二进制数，其指定要在 HTTP 请求中指定的 HTTP 方法。它应该设置为下列值之一:

1 (WBCLI_METHOD_GET)

2 (WBCLI_METHOD_POST)

WBCLI_FLAGS

一字节二进制位串，其可用于指定与 HTTP 请求及其所期待响应关联的选项。二进制位串中的二进制位必须设置为下列值:

1... .... (WBCLI_OFFSET_MODE)
参数列表中的指针值指定为从参数列表开始处的位移值。这暗示这种指针的所有目标都包含在通信区域中。

.1.. .... (WBCLI_DOCUMENT)
HTTP 请求主体是由 DOCUMENT CREATE 命令创建的 CICS 文档，并且由 WBCLI_REQUEST_DOCTOKEN 中的文档令牌指定

..1. .... (WBCLI_USE_PROXY)
HTTP 请求通过在 WBCLI_PROXY_URL_PTR 中指定其 URL 的代理服务器发送

...1 .... (WBCLI_SET_RESP_BUFFER)
CICS 要获取合适大小的缓冲区，以包含 HTTP 响应主体，并在 WBCLI_REQUEST_BODY_PTR 中返回其地址

注: 该地址不构成位移，无论 WBCLI_OFFSET_MODE 的设置是怎样的

.... ..1. (WBCLI_NATIVE_REQUEST_BODY)
应用程序将以其本机格式提供 HTTP 请求主体，而 CICS 不需要将它从 EBCDIC 转换为 ASCII

....1 (WBCLI_NATIVE_RESPONSE_BODY)
应用程序将以其本机格式处理 HTTP 响应主体，而 CICS 不需要将它从 ASCII 转换为 EBCDIC

WBCLI_RESPONSE

半字二进制数，其设置为下列值之一，以表明函数的结果:

0 (WBCLI_RESPONSE_OK)

4 (WBCLI_RESPONSE_EXCEPTION)

8 (WBCLI_RESPONSE_DISASTER)

WBCLI_REASON

半字二进制数，其设置为下列值之一，以限定响应代码:

1 (WBCLI_REASON_INVALID_URL)
WBCLI_URL_PTR 找到的 URL 的格式无效，或者名称服务器无法解析主机位置

2 (WBCLI_REASON_INVALID_HEADER)
WBCLI_HEADER_PTR 所找到列表中的一个 HTTP 头的格式不正确

3 (WBCLI_REASON_INVALID_DOCUMENT)
WBCLI_REQUEST_DOCTOKEN 中指定的文档令牌未找到有效 CICS 文档

4 (WBCLI_REASON_GETMAIN_ERROR)
当 DFHWBCLI 尝试为其内部工作区之一获取存储器时发生错误

5 (WBCLI_REASON_PROXY_ERROR)
WBCLI_PROXY_URL_PTR 找到的代理服务器无法找到，或返回出错响应

6 (WBCLI_REASON_SOCKET_ERROR)
执行套接字操作时返回意外响应

7 (WBCLI_REASON_HTTP_ERROR)
服务器返回意外 HTTP 响应

8 (WBCLI_REASON_TRANSLATE_ERROR)
当 CICS 在主机代码页和服务器代码页之间转换数据时，返回一条错误。这可能是因为 CICS 不支持必需的转换

9 (WBCLI_REASON_TRUNCATED)
WBCLI_RESPONSE_BODY_LEN 中所指定用户提供响应缓冲区的长度不足，无法包含服务器返回的响应。超出该长度的数据已丢弃。

WBCLI_SESSION_TOKEN
不透明的八字节二进制标记，其表示所建立与 HTTP 服务器的连接。它由 SEND 函数设置，而且对于 RECEIVE 和CLOSE 函数是必需的。其他函数不使用它。

WBCLI_URL_PTR
EBCDIC 字符串的地址，其包含目标 HTTP 服务器的 URL（统一资源定位符）。URL 必须是标准的：即，它必须以'`http://`'或'`https://`'开头。

WBCLI_URL_LEN
全字二进制数，其包含 WBCLI_URL_PTR 所找到 URL 的长度。

WBCLI_PROXY_URL_PTR
EBCDIC 字符串的地址，其包含访问防火墙外远程站点所需的代理服务器的 URL（统一资源定位符）。URL 必须是标准的：即，它必须以'`http://`'开头。要使用代理，您还必须设置 WBCLI_USE_PROXY 标志。

WBCLI_PROXY_URL_LEN
全字二进制数，其包含 WBCLI_PROXY_URL_PTR 所找到 URL 的长度。

WBCLI_HEADER_PTR
要与 HTTP 请求一起发送的 HTTP 头列表的地址。头必须在 EBCDIC 中已编码，以下列格式表示：

headername`:` *headervalue*`§`*headername*`:` *headervalue*`§ ...`

其中
headername
是头的名称
headervalue
是头的值

分隔这两个的冒号（`:`）和空格应该如所示出现；此处显示的"§"应该由下列一个或多个定界符替代：
回车符（X'0D'）
换行（X'25'）
新行（X'15'）
字段分隔符（X'1E'）

注: 发送头时不使用这些定界符: CICS 使用结构正确的 HTTP 定界符。
您可以按需要在列表中编码足够的头。然而，您不能包含以下头，因为 CICS 将提供它们:

Host
User-Agent
Content-Length
Content-Type

在列表最后一个头后，您无须提供定界符。

WBCLI_HEADER_LEN
全字二进制数，其包含 WBCLI_HEARER_PTR 所找到头列表的长度。

WBCLI_REQUEST_DOCTOKEN
一个 16 字节二进制文档令牌，由 DOCUMENT CREATE 命令创建，它表示用作 HTTP 请求主体的 CICS 文档。您必须通过设置 WBCLI_DOCUMENT 标志表明您正在使用该标记。

WBCLI_REQUEST_BODY_PTR
EBCDIC 字符串的地址，其包含 HTTP 请求主体的整个内容。在未设置 WBCLI_DOCUMENT 标志时使用该参数。

WBCLI_REQUEST_BODY_LEN
全字二进制数，其包含 WBCLI_REQUEST_BODY_PTR 所找到请求主体的长度。

WBCLI_RESPONSE_BODY_PTR
缓冲区的地址，DFHWBCLI 在其中返回来自服务器的 HTTP 响应主体。

- 如果未设置标志 WBCLI_SET_RESP_BUFFER，该地址和 WBCLI_RESPONSE_BODY_LEN 必须由调用者设置。如果该缓冲区不够大，以致无法包含响应主体，那么主体被截断。
- 如果设置了标志 WBCLI_SET_RESP_BUFFER，该地址和 WBCLI_RESPONSE_BODY_LEN 被忽略。CICS 获取大小足够包含整个响应的新缓冲区，并且它的地址也在该字段中返回。该地址不会转换到位移中，无论 WBCLI_OFFSET_MODE 标志的值是什么。

通常，当调用 DFHWBCLI 的任务结束时，CICS 释放该地址的存储器。或者，您可以通过在您的应用程序中发出 EXEC CICS FREEMAIN 命令来较早地释放存储器。在较长时间运行的任务中重复调用 DFHWBCLI 时，建议您这样做，以防止 CICS 变得存储器不够。

WBCLI_RESPONSE_BODY_LEN
全字二进制数，它包含 WBCLI_RESPONSE_BODY_PTR 所定位的响应缓冲区的长度。

- 在输入时，如果未设置 WBCLI_SET_RESP_BUFFER，使用该参数指定用户所提供缓冲区的长度。
- 在输出时，它包含所返回响应的实际长度。

WBCLI_MEDIATYPE
40 字节 EBCDIC 空填充字符串，其包含 HTTP 主体的 IANA 媒体类型（也称为 MIME 类型）。

- 在输入时，使用该参数指定 HTTP 请求主体的媒体类型。该媒体类型将在 HTTP Content-Type 头中发送

- 在输出时，它将包含 HTTP 响应主体的媒体类型，同在 HTTP Content-Type 头中接收到的一样。

对于使用 POST 方法的 SEND 请求（其中对 WBCLI_FUNCTION_SEND 和 WBCLI_METHOD_POST 都进行了设置），必须指定该媒体类型。

WBCLI_CHARSET

40 字节 EBCDIC 字符串，其包含 HTTP 主体的 IANA 字符集。

- 在输入时，如果未设置 WBCLI_NATIVE_REQUEST_BODY，那么使用该参数指定您要 CICS 将 HTTP 请求主体转换到的字符集名称。您指定的字符集用于限定 HTTP Content-Type 头中的媒体类型。如果您未指定值，那么仅当 WBCLI_MEDIATYPE 包含值 TEXT 时，才会使用缺省值 `iso-8859-1`。
- 在输出时，它将包含 HTTP 响应主体的字符集，同在 HTTP Content-Type 头中接收到的一样。该字符集用于转换 HTTP 响应主体（除非设置了 WBCLI_NATIVE_RESPONSE_BODY）。

WBCLI_HOST_CODEPAGE

10 字符 EBCDIC 空填充字符串，其包含您的应用程序所使用 EBCDIC 代码页的名称。它与 WBCLI_CHARSET 组合在一起使用，以确定要在 HTTP 文档主体上执行哪种转换（除非 WBCLI_NATIVE_REQUEST_BODY 或 WBCLI_NATIVE_RESPONSE_BODY 标志阻止转换）。如果省略它，那么 CICS 使用代码页 037。

WBCLI_HTTP_STATUS_CODE

三位数字 EBCDIC 字符串，在其中返回 HTTP 状态代码。它表明 HTTP 请求是否成功。200 状态码用于常规响应，2xx 范围中的其他状态码也表示成功。其他状态码表明有错误阻止执行请求，或者客户机需要执行其他操作以成功完成请求，如跟随重定向 URL。

附录 J. 状态管理样本 DFH$WBST 和 DFH$WBSR 的参考信息

两个状态管理样本程序 DFH$WBST 和 DFH$WBSR 与 CICS Web Support 一起提供。它们允许一个事务保存数据，以便将来通过这个事务本身或其他事务来检索。所保存的数据由第一个事务的状态管理程序所创建的令牌来访问。第一个事务必须把令牌传递给要检索数据的事务。DFH$WBST 使用 GETMAIN 命令为保存的数据分配存储器。DFH$WBSR 将数据保存到临时存储器队列，每个队列对应一个令牌，所以，使用合适的临时存储器表定义，就可以从多个 CICS 系统访问数据。本节的剩余部分均衡地应用于两个程序。

状态管理程序提供下列操作:

- 创建新令牌。
- 存储信息并将它与先前创建的令牌关联。
- 检索先前与令牌关联的信息。
- 销毁与令牌关联的信息并使令牌无效。

DFH$WBST 还除去到期的信息和令牌。您可以周期性地运行这个程序以便清除期满的状态数据:

- 将程序作为 CWBT 事务运行，以便清除所有一小时内未更新的状态数据。
- 将程序作为 CWBP 事务运行，以便清除所有状态数据。

268 字节的通信区域布局如下表所示。在为您所需要的函数设置输入之前，必须将通信区域清空为二进制零。

表 28. 用于状态管理程序的参数

偏移量	长度	类型	值	注
0	4	C		Eyecatcher
4	1	C	’C’ ’R’ ’S’ ’D’	Create Retrieve Store Destroy 这是函数码。它是每次调用所需的输入。
5	1	X		返回码。这是来自每次调用的输出。
6	2	X		保留。
8	4	F		令牌。这是来自 Create 调用的输出，以及到每个其他调用的输入。
12	256	C		用户数据。这是到 Create 和 Store 调用的输入，以及来自 Retrieve 调用的输出。它不用于其他调用。

返回码如下:

0 执行了所请求的函数。

- 如果是 Create 函数，新令牌在偏移量为 8 时可用。

- 如果是 Retrieve 函数，与输入令牌关联偏移量为 8 的实体主体现在处于偏移量为 12 的实体主体区域。
- 如果是 Store 函数，偏移量为 12 的输入实体主体现在与偏移量为 8 的输入令牌相关联的。先前与该令牌相关联的任何实体主体将被覆盖。
- 如果是 Destroy 函数，与输入令牌关联偏移量为 8 的数据将被丢弃，并且令牌将不再有效。

2 偏移量为 4 的函数码是无效的。校正设置通信区域的程序。

3 函数是 Create，但是 GETMAIN 命令给出错误响应。

4 函数是 Retrieve、Store 或者 Destroy，但是偏移量为 8 的输入令牌找不到。原因是输入令牌不是 Create 返回的令牌，或者它已经期满。

5 将内部数据写到临时存储器队列时，WRITEQ TS 命令给出错误响应。

7 ASKTIME 命令给出错误响应。

8 从临时存储器队列读取内部数据时，READQ TS 给出错误响应。

9 超时处理期间，ASKTIME 命令给出错误响应。

11 函数是 Create，但是 WRITEQ TS 命令给出错误响应。这个返回码只能由 DFH$WBSR 产生。

12 函数是 Retrieve，但是 READQ TS 命令给出错误响应。这个返回码只能由 DFH$WBSR 产生。

13 函数是 Store，但是 WRITEQ TS 命令给出错误响应。这个返回码只能由 DFH$WBSR 产生。

14 函数是 Destroy，但是 DELETEQ TS 命令给出错误响应。这个返回码只能由 DFH$WBSR 产生。

附录 K. CICS Web 服务器插件

出于兼容性考虑，保留了 CICS Web 服务器插件的功能。建议您迁移至使用 CICS Web Service、CICS Web Support 或 CICS Transaction Gateway 的解决方案。

提供的这个插件利用 CICS 业务逻辑接口，实现了从 IBM HTTP Server 经外部 CICS 接口（EXCI）到 CICS Web Support 的传递机制。该接口上可传递的最大数据量为 32KB。

配置 IBM HTTP Server

出于兼容性考虑，保留了 CICS Web 服务器插件的功能。建议您在新的应用程序中使用 CICS Transaction Gateway。

关于此任务

如果 IBM HTTP Server 要使用 CICS 业务逻辑接口来提供其服务的话，那么必须更改其中的配置信息。*z/OS HTTP Server Planning, Installing, and Using*，SC34-4826 提供有关配置语句的详细信息。

您可使用以下过程:

1. 必须如下设置 CICS:
   a. 用 ISC=YES 初始化 CICS 区域。
   b. 安装 RDO 组 DFHWEB
   c. 为 EXCI 定义一个常规连接；例如，通过安装样本组 DFH$EXCI。
   d. 确保 IRC 是开的。
2. 向 RACF 程序控制定义 CICSTS41.CICS.SDFHDLL1 装入库和 CICSTS41.CICS.SDFHEXCI。 RACF 程序控制会记录包含该库的卷的卷序列号，且不允许使用其他卷。如果您稍后将装入库或 CICSTS41.CICS.SDFHEXCI 库移动到另一个卷，那么必须将它重新定义到 RACF 程序控制。
3. 将 CICSTS41.CICS.SDFHDLL1 数据集和 CICSTS41.CICS.SDFHEXCI 库添加到 IBM HTTP Server 的 JCL 中的 STEPLIB 并置。 SDFHEXCI 和 SDFHDLL1 向下兼容所有受支持的 CICS 发行版。
4. 在包含 IBM HTTP Server 的 httpd.conf 文件的目录中使用以下命令:

   ```
   ln -e DFHWBAPI dfhwbapi.so
   ```

   在 STEPLIB 并置中使用该命令时，它建立从 IBM HTTP Server 的主目录到 CICSTS41.CICS.SDFHDLL1 库中成员 DFHWBAPI 中的 DLL dfhwbapi.so 的链接。
5. 在 httpd.conf 文件中添加一个或多个服务伪指令。 服务伪指令将最终用户输入的 URL 映射到将满足请求的 CICS 资源。DFHWBAPI 的服务伪指令具有以下格式:

   ```
   Service  /sourceurl/*  /home/dfhwbapi.so:DFHService/targeturl/*
   ```

   其中的值为:

   home 是包含 IBM HTTP Server 的 httpd.conf 文件的目录。

sourceurl

是一个字符串，它选择 DFHWBAPI 要处理的入站 URL。紧跟其后的星号是一个通配符字符串，表示进入的 URL 的其余字符。*sourceurl* 可以是任何格式，所以象 *APPLID* 和 *transaction* 这样的细节可以对最终用户隐藏。

targeturl

targeturl 是一串字符串，DFHWBAPI 将使用它确定哪些 CICS 资源会满足用户请求。在替代了通配符后，*targeturl* 的格式必须为:

```
/applid/converter/tran/program/filename
```

其中的值为:

applid 目标 CICS 区域的应用程序标识

converter

在 CICS 区域中使用的转换器程序的名称，或者如果没有要使用的转换器，那么名称为 CICS。

tran 要在 CICS 区域中执行的事务。因为该事务是 EXCI 请求的目标事务，因此它不应该是 Web 别名事务 CWBA，而应该是镜像事务，如 CSM3。该事务接收 *targeturl/** 而不是 *sourceurl/**，以该作为入站 URL。

program

要在 CICS 区域中执行的程序的名称。

filename

是由*程序检测的更多信息*。

如果 DFHWBAPI 用于访问 3270 应用程序，那么 CICS 生成在 Web 客户机上显示的 HTML 表单。CICS 插入到 HTML 表单中的 URL 与先前请求中使用的 *targeturl* 匹配。要处理这种情况，除了上述几点外，还必须提供以下格式的服务伪指令:

```
Service  /targeturl/*  /home/dfhwbapi.so:DFHService
```

这里，*targeturl* 在未经更改下传递给 DFHWBAPI。

6. 如果想要显示 CICS 提供的某些模板定义所引用的图形文件，请按如下方式包括伪指令:

```
Service  /dfhwbimg/*  /home/dfhwbapi.so:DFHService/applid/DFHWBIMG/CSM3/*
```

其中 *applid* 指定将提供图形文件的 CICS 区域（这个区域可能与执行网桥工作的 CICS 区域不同）。 DFHWBIMG 是 CICS 提供的、供 CICS Web 网桥专用的转换器程序。

7. 如果正在使用 CICS Web Support 和 CICS 业务逻辑接口访问 CICS Web 应用程序，那么必须为这两者指定相同的主机代码页。 对于 CICS，缺省主机代码页是 IBM-037，但是对于 IBM HTTP Server，它是 IBM-1047。
   - 您可以使用 DefaultFsCp 配置伪指令来更改 IBM HTTP Server 的缺省代码页。例如:

     ```
     DefaultFsCp IBM-1047
     ```

   - 要更改 CICS 使用的缺省代码页，请在 DOCCODEPAGE 系统初始化参数中指定它；例如，DOCCODEPAGE=1047。

使用这个缺省值所引用的文档和文档碎片必须以指定的代码页进行编码。特别是，如果使用 BMS 映射定义生成的文档模板，必须使用模板定制宏来更改用于生成模板的代码页。使用 DFHMDX 宏的 **CODEPAGE** 参数来指定它；例如:

```
DFHMDX MAPSET=*,MAP=*,CODEPAGE=1047
```

要获得有关定制 BMS 映射定义生成的模板的更多信息，请参阅第 185 页的第 16 章，『从 BMS 定义创建 HTML 模板』。

下一步做什么

转义数据和 IBM HTTP Server

如果使用 IBM HTTP Server 和 CICS 业务逻辑接口访问同一个 CICS 应用程序，那么必须确保在这两种情况下对转义数据进行一致的处理。

IBM HTTP Server 将数据以其未转义形式传递到 CICS 应用程序；因此，必须确保 CICS Web Support 执行相同的操作。

要获取更多信息，请参阅第 118 页的『从分析器程序选择转义数据或未转义数据』

IBM HTTP Server 的处理示例

图 27 显示了 CICS Web Support 如何处理来自连接到 IBM HTTP Server 的 Web 客户机的请求。

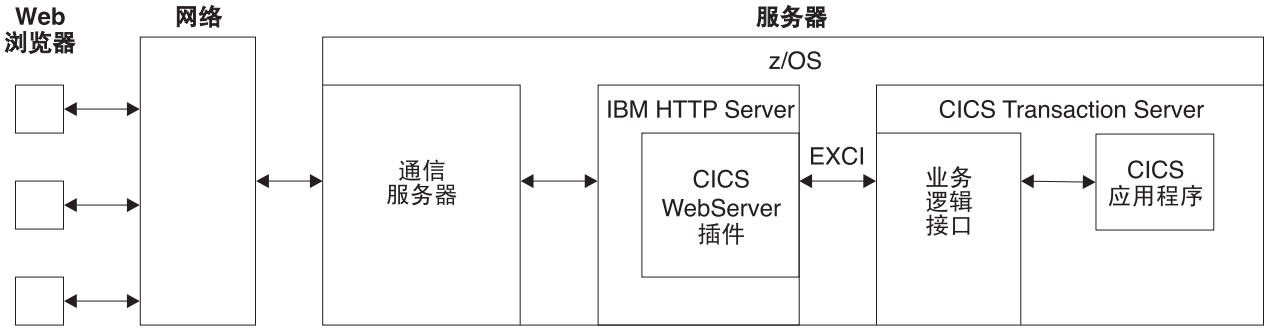


图 27. 处理来自 IBM HTTP Server 的请求

1. Web 客户机构造通过网络传递到 Communications Server 的 HTTP 请求。
2. Communications Server 将请求中继至 IBM HTTP Server。
3. IBM HTTP Server 调用 CICS Web 服务器插件。
4. CICS Web 服务器插件为 CICS 业务逻辑接口构造一个请求，并通过使用外部 CICS 接口（EXCI）将该请求传递到 CICS。
5. CICS 业务逻辑接口调用请求的 CICS 应用程序，并在 COMMAREA 中返回所有输出。

声明

本信息是为在美国提供的产品和服务编写的。IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能特性。要获取有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代理咨询。 任何对 IBM 的产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，那么由用户自行负责。

IBM 可能已拥有或正在申请与本文档内容有关的各项专利权。提供本文档并未授予用户使用这些专利的任何许可证。您可以用书面方式将许可证查询寄往:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

有关双字节（DBCS）信息的许可证查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区:

International Business Machines Corporation“按现状”提供本出版物，不附有任何种类的（无论是明示的还是暗含的）保证，包括（但不限于）暗含的有关非侵权、适销和适用于某特定用途的保证。某些国家或地区在某些交易中不允许免除明示或暗含的保证。因此本条款可能不适用于您。

本出版物中可能包含技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本出版物的新版本中。 IBM 可以随时对本资料中描述的产品和/或程序进行改进和/或更改，而不另行通知。

本程序的被许可方如果要了解有关程序的信息以达到如下目的: （i）允许在独立创建的程序和其他的程序（包括本程序）之间进行信息交换，以及（ii）允许对已经交换的信息进行相互使用，请与下列地址联系: IBM United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN。只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际程序许可证协议或任何同等协议中的条款提供。

商标

IBM、IBM 徽标和 ibm.com 是 International Business Machine Corp., 在全球许多司法区域注册的商标或注册商标。其他产品和服务名称可能是 IBM 或其他公司的商标。Web 上的版权和商标信息（www.ibm.com/legal/copytrade.shtml）提供了 IBM 商标的最新列表。

Java 和所有基于 Java 的商标和徽标是 Sun Microsystems, Inc. 在美国和/或其他国家或地区的商标。

Microsoft 和 Windows 是 Microsoft Corporation 在美国和/或其他国家或地区的商标。

UNIX 是 The Open Group 在美国和其他国家或地区的注册商标。

其他公司、产品或服务名称可能是其他公司的商标或服务标记。

参考书目

有关 CICS Transaction Server for z/OS 的 CICS 书籍

常规

CICS Transaction Server for z/OS Program Directory, GI13-0536
CICS Transaction Server for z/OS 新增功能, G151-1222
CICS Transaction Server for z/OS Upgrading from CICS TS Version 2.3, GC34-6996
CICS Transaction Server for z/OS Upgrading from CICS TS Version 3.1, GC34-6997
CICS Transaction Server for z/OS Upgrading from CICS TS Version 3.2, GC34-6998
CICS Transaction Server for z/OS 安装指南, G151-1155

访问 CICS

CICS 因特网指南, S151-1156
CICS Web Services Guide, SC34-7020

管理

CICS System Definition Guide, SC34-6999
CICS Customization Guide, SC34-7001
CICS Resource Definition Guide, SC34-7000
CICS Operations and Utilities Guide, SC34-7002
CICS RACF Security Guide, SC34-7003
CICS Supplied Transactions, SC34-7004

编程

CICS Application Programming Guide, SC34-7022
CICS Application Programming Reference, SC34-7023
CICS System Programming Reference, SC34-7024
CICS Front End Programming Interface User's Guide, SC34-7027
CICS C++ OO Class Libraries, SC34-7026
CICS Distributed Transaction Programming Guide, SC34-7028
CICS Business Transaction Services, SC34-7029
Java Applications in CICS, SC34-7025

诊断

CICS Problem Determination Guide, SC34-7034
CICS Performance Guide, SC34-7033
CICS Messages and Codes, SC34-7035
CICS Diagnosis Reference, GC34-7038
CICS Recovery and Restart Guide, SC34-7012
CICS Data Areas, GC34-7014
CICS Trace Entries, SC34-7013
CICS Supplementary Data Areas, GC34-7015

CICS Debugging Tools Interfaces Reference, GC34-7039

通信

CICS Intercommunication Guide, SC34-7018
CICS External Interfaces Guide, SC34-7019

数据库

CICS DB2 Guide, SC34-7011

CICS IMS Database Control Guide, SC34-7016

CICS Shared Data Tables Guide, SC34-7017

有关 CICS Transaction Server for z/OS 的 CICSPlex SM 书籍

常规

CICSPlex SM Concepts and Planning, SC34-7044
CICSPlex SM Web User Interface Guide, SC34-7045

监管

CICSPlex SM Administration, SC34-7005
CICSPlex SM Operations Views Reference, SC34-7006
CICSPlex SM Monitor Views Reference, SC34-7007
CICSPlex SM Managing Workloads, SC34-7008
CICSPlex SM Managing Resource Usage, SC34-7009
CICSPlex SM Managing Business Applications, SC34-7010

编程

CICSPlex SM Application Programming Guide, SC34-7030
CICSPlex SM Application Programming Reference, SC34-7031

诊断

CICSPlex SM Resource Tables Reference, SC34-7032
CICSPlex SM Messages and Codes, GC34-7035
CICSPlex SM Problem Determination, GC34-7037

其他 CICS 出版物

以下出版物包含有关 CICS 的进一步信息，但它们未在 CICS Transaction Server for z/OS V4R1 中提供。

Designing and Programming CICS Applications, SR23-9692

CICS Application Migration Aid Guide, SC33-0768

CICS Family: API Structure, SC33-1007

CICS Family: Client/Server Programming, SC33-1435

CICS Family: Interproduct Communication, SC34-6853

CICS Family: Communicating from CICS on System/390, SC34-6854

CICS Transaction Gateway for z/OS Administration, SC34-5528

CICS Family: General Information, GC33-0155

CICS 4.1 Sample Applications Guide, SC33-1173

CICS/ESA 3.3 XRF Guide, SC33-0661

其他 IBM 出版物

下列出版物包含与 IBM 产品相关的信息

UNIX 系统服务

z/OS UNIX System Services User's Guide, SA22-7801
z/OS UNIX System Services Command Reference, SA22-7802
z/OS UNIX System Services Messages and Codes, SA22-7807
z/OS UNIX System Services Programming Tools, SA22-7805
z/OS UNIX System Services Programming: Assembler Callable, SA22-7803
z/OS UNIX System Services File System Interface Reference, SA22-7808
z/OS Using REXX and z/OS UNIX System Services, SA22-7806
z/OS V1R1 UNIX System Services Parallel Environment: MPI Programming and Subroutine Reference, SA22-7812

z/OS Communications Server

z/OS Communications Server: IP Configuration Reference, SC31-8776
z/OS Communications Server: IP Configuration Guide, SC31-8775
z/OS Communications Server: IP Migration, GC31-8773
z/OS Communications Server: IP CICS Sockets Guide, SC31-8807
z/OS Communications Server: IP Application Programming Interface Guide, SC31-8788
z/OS Communications Server: IP Programmer's Reference, SC31-8787
z/OS Communications Server: IP User's Guide and Commands, SC31-8780
z/OS Communications Server: Quick Reference, SX75-0124
z/OS Communications Server: IP Diagnosis, GC31-8782

IBM Redbooks

IBM Redbooks 由 IBM 国际技术支持组织开发和发布。它们研究现实客户方案的集成、实现和操作。

下列 IBM Redbooks 包含与该出版物中的材料相关的信息:
Accessing CICS Business Applications from the World Wide Web, SG24-4547
How to Secure the Internet Connection Server for MVS/ESA, SG324-4803
Revealed! Architecting Web Access to CICS, SG24-5466
Workload Management for Web Access to CICS, SG24-6118

Web 上的信息

本书中提供的 URL 并不保证能正常使用。

RFC（请求评论）

RFC 1945, *超文本传输协议 - HTTP/1.0* （HTTP/1.0 规范）: http://www.ietf.org/rfc/rfc1945.txt
RFC 1867, *HTML 中基于表单的文件上载*: http://www.ietf.org/rfc/rfc1867.txt
RFC 2616, 超文本传输协议 - HTTP/1.1（HTTP/1.1 规范）: http://www.ietf.org/rfc/rfc2616.txt

RFC 2617，HTTP 认证：基本和摘要访问认证：http://www.ietf.org/rfc/rfc2617.txt

RFC 2396，统一资源标识（URI）：一般语法：http://www.ietf.org/rfc/rfc2396.txt

RFC 3023，XML 介质类型：http://www.ietf.org/rfc/rfc3023.txt

RFC 文档由因特网协会和因特网工程任务组织（IETF）发布。

IANA（因特网编号分配机构）

IANA 注册的介质类型：http://www.iana.org/assignments/media-types/

IANA 注册的字符集名称：http://www.iana.org/assignments/character-sets

IANA 注册的端口号：http://www.iana.org/assignments/port-numbers

万维网联盟（W3C）

W3C 主页：http://www.w3.org/

W3C HTTP 概述：http://www.w3.org/Protocols/Overview.html

W3C HTML 主页：http://www.w3.org/MarkUp/

IBM IBM developerWorks：http://www.ibm.com/developerworks/

辅助功能选项

辅助功能帮助身体有残障（如行动不便或视力受限）的用户顺利使用软件产品。

您可以通过以下方法之一执行设置、运行和维护 CICS 系统所需的大多数任务:

- 使用 3270 仿真器登录到 CICS
- 使用 3270 仿真器登录到 TSO
- 使用 3270 仿真器作为 MVS 系统控制台

“IBM 个人通信”为残障用户提供了具有辅助功能的 3270 仿真。该产品可以为用户提供使用 CICS 系统时所需的辅助功能。

如果使用“Web 终端转换应用程序（DFHWBTTA）”来访问 CICS 提供的事务，那么辅助功能选项功能部件是由 Web 浏览器提供的。

索引

[G]

[H]

[J]

[K]

[L]

[M]

[P]

[Q]

[R]

[S]

[T]

[W]

[数字]

A

B

C

D

E

F

G

H

I

L

M

P

R

S

T

U

W

X

Z

S151-1156-00