THE DOCUMENTATION.

UTOR, SUPPLIER, INDEPENDENT CONTRACTOR OF RIM OR ANY AFFILIATES OF RIM HAVE ANY LIABILITY ARISING FROM OR RELATED TO

IN ADDITION TO THE LIMITATIONS AND EXCLUSIONS SET OUT ABOVE, IN NO EVENT SHALL ANY DIRECTOR, EMPLOYEE, AGENT, DISTRIB-

OTHER LEGAL THEORY AND SHALL SURVIVE A FUNDAMENTAL BREACH OR BREACHES OR THE FAILURE OF THE ESSENTIAL PURPOSE OF

DEMAND, OR ACTION BY YOU INCLUDING BUT NOT LIMITED TO BREACH OF CONTRACT, NEGLIGENCE, TORT, STRICT LIABILITY OR ANY

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW IN YOUR JURISDICTION, RIM SHALL HAVE NO OTHER OBLIGATION, DUTY,

SUCH DAMAGES.

WHETHER OR NOT SUCH DAMAGES WERE FORESEEN OR UNFORESEEN, AND EVEN IF RIM HAS BEEN ADVISED OF THE POSSIBILITY OF

DOWNTIME COSTS, LOSS OF THE USE OF RIM PRODUCTS OR SERVICES OR ANY PORTION THEREOF OR OF ANY AIRTIME SERVICES, COST

LOSS OF BUSINESS INFORMATION, LOSS OF BUSINESS OPPORTUNITY, OR CORRUPTION OR LOSS OF DATA, FAILURES TO TRANSMIT OR

THE FOLLOWING DAMAGES: DIRECT, CONSEQUENTIAL, EXEMPLARY, INCIDENTAL, INDIRECT, SPECIAL, PUNITIVE, OR AGGRAVATED

HARDWARE, SERVICE, OR ANY THIRD PARTY PRODUCTS AND SERVICES REFERENCED HEREIN INCLUDING WITHOUT LIMITATION ANY OF

OF DAMAGES RELATED TO THIS DOCUMENTATION OR ITS USE, OR PERFORMANCE OR NON-PERFORMANCE OF ANY SOFTWARE,

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW IN YOUR JURISDICTION, IN NO EVENT SHALL RIM BE LIABLE FOR ANY TYPE

OF DAMAGES RELATED TO THIS DOCUMENTATION OR ITS USE, OR PERFORMANCE OR NON-PERFORMANCE OF ANY SOFTWARE,

OF DAMAGES INCLUDING WITHOUT LIMITATION ANY OF THE FOLLOWING DAMAGES: DIRECT, CONSEQUENTIAL, EXEMPLARY, INCIDENTAL,

EXCEPT TO THE EXTENT SPECIFICALLY PROHIBITED BY APPLICABLE LAW IN YOUR JURISDICTION, ALL CONDITIONS, ENDORSEMENTS,

GUARANTEES, REPRESENTATIONS, OR WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY CON-

CONDITIONS, ENDORSEMENTS, GUARANTEES, REPRESENTATIONS OR WARRANTIES OF DURABILITY, FITNESS FOR A PARTICULAR PURPOSE

TO USE, MERCHANTABILITY, MERCHANTABLE QUALITY, NON-INFRINGEMENT, SATISFACTORY QUALITY, OR TITLE, OR ARISING FROM A

STATE OR CUSTOM OR A COURSE OF DEALING OR USAGE OF TRADE, OR RELATED TO THE DOCUMENTATION OR ITS USE, OR PERFORMANCE OR NON-PERFORMANCE OF ANY SOFTWARE, HARDWARE, SERVICE, OR ANY THIRD PARTY PRODUCTS AND SERVICES REFERENCED HEREIN, ARE HEREBY EXCLUDED. YOU MAY ALSO HAVE OTHER RIGHTS THAT VARY BY STATE OR PROVINCE. SOME JURIS-

DICTIONS MAY NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES AND CONDITIONS, TO THE EXTENT PERMITTED

BY LAW, ANY IMPLIED WARRANTIES OR CONDITIONS RELATING TO THE DOCUMENTATION TO THE EXTENT THEY CANNOT BE

EXCLUDED AS SET OUT ABOVE, BUT CAN BE LIMITED, ARE HEREBY LIMITED TO NINETY (90) DAYS FROM THE DATE YOU FIRST ACQUIRED

THE DOCUMENTATION OR THE ITEM THAT IS THE SUBJECT OF THE CLAIM.

THE LIMITATIONS, EXCLUSIONS, AND DISCLAIMERS HEREIN SHALL APPLY: (A) IRRESPECTIVE OF THE NATURE OF THE CAUSE OF ACTION,

DEMAND, OR ACTION BY YOU INCLUDING BUT NOT LIMITED TO BREACH OF CONTRACT, NEGLIGENCE, TORT, STRICT LIABILITY OR ANY

OTHER LEGAL THEORY AND SHALL SURVIVE A FUNDAMENTAL BREACH OR BREACHES OR THE FAILURE OF THE ESSENTIAL PURPOSE OF

THIS AGREEMENT OR OF ANY REMEDY CONTAINED HEREIN; AND (B) TO RIM AND ITS AFFILIATED COMPANIES, THEIR SUCCESSORS,

ASSIGNS, AGENTS, SUPPLIERS (INCLUDING AIRTIME SERVICE PROVIDERS), AUTHORIZED RIM DISTRIBUTORS (ALSO INCLUDING AIRTIME

SERVICE PROVIDERS) AND THEIR RESPECTIVE DIRECTORS, EMPLOYEES AND INDEPENDENT CONTRACTORS.

IN ADDITION TO THE LIMITATIONS AND EXCLUSIONS SET OUT ABOVE, IN NO EVENT SHALL ANY DIRECTOR, EMPLOYEE, AGENT, DISTRIBUT-

UTOR, SUPPLIER, INDEPENDENT CONTRACTOR OF RIM OR ANY AFFILIATES OF RIM HAVE ANY LIABILITY ARISING FROM OR RELATED TO


Prior to subscribing for, installing or using any Third Party Products and Services it is your responsibility to ensure that your airtime service provider has agreed to support all of their features. Some airtime service providers may not offer Internet browsing functionality with a subscription to BlackBerry® Internet Service. Check with your service provider for availability, roaming arrangements, service plans and features. Installation or use of Third Party Products and Services with RIM’s products and services may require one or more patent, trademark, copyright or other licenses in order to avoid infringement or violation of third party rights. You are solely responsible for determining whether to use. Third Party Products and Services and if any third party licenses are required to do so. If required you are responsible for acquiring them. You should not install or use Third Party Products and Services until all necessary licenses have been acquired. Any Third Party Products and Services that are provided with RIM’s products and services are provided as a convenience to you and are provided ‘AS IS’ with no express or implied conditions, endorsements, guarantees, representations or warranties of any kind by RIM and RIM assumes no liability whatsoever, in relation thereto. Your use of Third Party Products and Services shall be governed by and subject to you agreeing to the terms of separate licenses and other agreements applicable thereto with third parties, except to the extent expressly covered by a license or other agreement with RIM.

Certain features outlined in this documentation require a minimum version of BlackBerry® Enterprise Server, BlackBerry® Desktop Software, BlackBerry® Device Software and/or BlackBerry Handheld Software and may require additional development or Third Party Products and Services for access to corporate applications.

Certain products mentioned in this documentation include software developed by the Apache Software Foundation (http://www.apache.org/) and/or licensed pursuant to Apache License, Version 2.0 (http://www.apache.org/licenses/). For more information, see the NOTICE.txt file included with the software. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

The terms of use of any RIM product or service are set out in a separate license or other agreement with RIM applicable thereto. NOTHING IN THIS DOCUMENTATION IS INTENDED TO SUPERSEDE ANY EXPRESS WRITTEN AGREEMENTS OR WARRANTIES PROVIDED BY RIM FOR PORTIONS OF ANY RIM PRODUCT OR SERVICE OTHER THAN THIS DOCUMENTATION.
Chapter 5
Introduction to Browser Push

Objectives

- Define and explain Browser Push
- List the advantages and disadvantages of Browser Push
- Describe and explain the importance of the types of Browser Push, such as Browser Channel, Browser Message, Browser Cache, and Browser Channel-Delete
- Explain how to write a Browser Push Application
- Describe how mobile applications have benefited from Browser Push, such as push messaging from the BlackBerry® Enterprise Server
- Describe and use the transcoding feature of Browser Push

In this chapter, you learn about the benefits of using Browser Push, types of Browser Push and how to write a Browser Push Application. Pushing data to a BlackBerry smartphone is the most efficient way to get information to BlackBerry smartphone users. You also learn about the transcoding feature of Browser Push.
What is Browser Push?

Browser Push is how your information, such as weather updates, is automatically sent to your BlackBerry® smartphone without you requesting it. Browser Push is designed to push information from a web application to your BlackBerry smartphone as the information becomes available. For example, you can use Browser Push to receive updates for weather reports or breaking news. After your BlackBerry smartphone receives an update, a new Browser Push message appears in a message list or on the Home screen. This list changes to indicate new information is available. The web application also pushes updated information to your browser cache without notifying you. If information is pushed to your browser cache, the information is available on your smartphone even when it is not connected to the wireless network.

To push data to a BlackBerry smartphone, two applications are required: a server-side push application, which submits the push request, and a client-side listener application, which listens for incoming push messages. Pushing content to the BlackBerry® Browser allows you to access the push technology that the BlackBerry® MDS Connection Service provides without having to create your own client-side application. The BlackBerry Browser is designed with a listening thread, which listens in the background for data to be delivered to the BlackBerry smartphone. You can push any content that is supported by the target version of the BlackBerry Browser.

A server-side application is created to send content to one or more BlackBerry smartphones through BlackBerry® Enterprise Server for MDS Applications. The BlackBerry MDS Integration Service includes a Browser Push engine that can poll specific web addresses for changes or updates and push the web content to BlackBerry smartphones at an interval that developers specify. For example, a developer can create a BlackBerry Browser Application using BlackBerry templates or standard web development tools that pushes a web page about the local weather to BlackBerry smartphones every morning.

The BlackBerry Enterprise Server provides a complete wireless platform to extend the benefits of messaging and collaboration environments and other tools to mobile professionals. It provides push-based access to email, calendar, contacts, tasks and notes, instant messaging, web-based applications, services and enterprise applications. BlackBerry Enterprise Server strengthens integration between BlackBerry smartphones and corporate enterprise systems, enabling mobile workers to be more productive.

Note:
The BlackBerry Browser is on all BlackBerry smartphones that have a BlackBerry MDS account on a BlackBerry Enterprise Server. The BlackBerry Browser uses the BlackBerry MDS services installed on the BlackBerry Enterprise Server to provide access to information and communications.

Components of a Push Application Service

A Push Application Service comprises three components:
• The server-side push service—Content providers develop a push service that offers users some usable data, for example, stock information, sports injury reports, news bulletins, and banking information. When new data is available, content providers can submit a push request, using the WAP PAP 2.2 standard, to have the data delivered to users. The server-side push application is designed to send HTTP POST requests to the BlackBerry MDS Connection Service on the web server listen port. The HTTP POST requests contain delivery parameters and the data to be pushed. The BlackBerry MDS Connection Service then delivers the data to a specified port number on the BlackBerry device within a specified time frame. Developers must create the server-side application. Push requests generated by this application must conform to either the WAP PAP 2.0 specification or to the RIM push format.

• A client-side application—Content providers develop a client-side Java application that makes use of the data offered by the associated push service. The client application incorporates a listener, which listens in the background for incoming data from the server. A client-side listener application listens for data to be delivered to its specific port number. You can create a custom BlackBerry® Java Application that is designed to listen for and handle pushed data, or you can push web content to the BlackBerry® Browser. The BlackBerry Browser has an integrated listener thread, which allows you to push web content to the BlackBerry smartphone without the need to create a custom client-side application.

• The BlackBerry® Infrastructure—The BlackBerry Infrastructure provides the middleware that receives requests to push data from the content provider, and then delivers the given data to the associated client application on the BlackBerry smartphone.

How the BlackBerry Push API works

1. The content provider sends a PAP push request to the BlackBerry Infrastructure. This request includes a list of the subscribed BlackBerry smartphones to which the data is delivered, as well as the data to be delivered.

2. The BlackBerry Infrastructure returns a response to the content provider, acknowledging receipt of the request, and queues the request for delivery.

3. The BlackBerry Infrastructure pushes the data to the specified BlackBerry smartphones.
4. Each BlackBerry smartphone returns a response to the BlackBerry Infrastructure, acknowledging receipt of the data.

5. The BlackBerry Infrastructure forwards the acknowledgement on to the content push initiator.

6. The Read notification is returned to the BlackBerry Infrastructure.

The figure below demonstrates the process flow of a push request.

1. Content provider sends a push request.

2. BlackBerry Infrastructure returns a response.
3. BlackBerry Infrastructure pushes the data to the BlackBerry smartphone.
4. BlackBerry smartphone returns a response to the BlackBerry Infrastructure.
5. BlackBerry Infrastructure forwards acknowledgement to content provider.
6. Read notification is returned to the BlackBerry Infrastructure.

**Advantages of Browser Push**

- Send information to the browser in a variety of ways (channel, cache, message), regardless of back-end server technology (Apache, IIS, WebSphere).
- Leverage industry standards such as PAP or use the simplified BlackBerry push technology.
- Greatly reduce the amount of data being sent/received while your users are waiting for updates.
- Applications can be written in most server-side languages.
- The push process is controlled from the server side.
- Push applications are relatively easy to write.
- Each push request is designed to refresh the currently cached copy when network coverage permits.

**Disadvantages of Browser Push**

- Data format is limited to what the BlackBerry Browser can handle or what a custom-developed application can decode.
- Development time for push technology applications is greater than creating static pages.
- More expensive additional software packages are needed and faster, more efficient, and more expensive computers are needed to run push software.
- Downloading of sounds, images, applets, and video in the background can eat up hard disk.
- Content delivery may cause bandwidth problems. Problems arise due to the bandwidth that push technologies can require when feeding data to thousands of end users.
- Security safeguards are needed.
1. Which of the following statements best defines Browser Push?
   A. The client requests information using a browser.
   B. The server requests information from the client.
   C. The server pushes information to the client based on the client’s request.
   D. The server initiates the request for a given transaction.

2. Which of the following steps is the first step in the push request flow?
   A. Content provider sends a push request.
   B. The BlackBerry Infrastructure pushes data to the BlackBerry smartphone.
   C. A Read notification is sent to the BlackBerry Infrastructure.
   D. BlackBerry Infrastructure forwards acknowledgement to the content provider.

3. Which of the following provides the middleware that receives requests to push data from the content provider, and then delivers the given data to the associated client application on the BlackBerry smartphone?
   A. The BlackBerry Enterprise Server
   B. The BlackBerry Infrastructure
   C. The Client-Side Application
   D. The Server-Side Push Service

4. Place the following steps in the order in which they occur in the push flow request by placing the numbers 1 to 6 in the spaces provided.
   _____ BlackBerry Infrastructure returns a response.
   _____ BlackBerry smartphone returns a response to the BlackBerry Infrastructure.
   _____ BlackBerry Infrastructure forwards acknowledgement to content provider.
   _____ Content provider sends a push request.
Read notification is returned to the BlackBerry Infrastructure.

BlackBerry Infrastructure pushes the data to the BlackBerry smartphone.

5. Which of the following is an advantage of using Browser Push?
   A. The BlackBerry smartphone supports server-side push communication.
   B. Information can be sent to the browser in a variety of ways.
   C. Browser Push leverages industry standards such as PAP.
   D. All of the above
   E. None of the above

6. Which of the following is a limitation of Browser Push?
   A. The push process being controlled from the server side
   B. The number of formats handled by BlackBerry
   C. An increase in data being sent while waiting for updates
   D. The difficulty of writing push applications

7. What two applications are required to push data to a BlackBerry smartphone?
Chapter 5

Answers

1. D
2. A
3. B
4. 2
5. 4
6. 5
7. 1
8. 6
9. 3
5. D
6. B
7. Server-side push application and client-side listener application
Types of Browser Push

The BlackBerry Browser supports several push options, allowing you to choose the approach that best suits your needs. You can specify which option to use by using an HTTP header. Additional HTTP headers that are specific to a browser push application can allow you to control how the content appears to the user. These browser-specific headers must be included in addition to the PAP control entity or RIM push headers that are part of every push request.

The BlackBerry smartphone supports different types of Browser Push messages. Service loading messages include updated information. Service indication messages include a link to updated information. Other Browser Push messages include messages that appear in your browser cache. The BlackBerry Enterprise Server supports the following types of BlackBerry Browser push applications:

Browser Channel Push

When the BlackBerry Browser receives content that has been pushed to a channel, the BlackBerry smartphone creates a new channel, or updates the channel if the specified channel ID already exists. When a channel is added or updated, the BlackBerry smartphone displays an unread icon on the Home screen to alert the BlackBerry smartphone user that content is available. For example, the icon for an order-tracking channel might change when product orders are entered. After the user clicks the icon and views the updated content, the BlackBerry device displays a read icon on the Home screen, which remains until new content is delivered for that channel.

A Browser Channel Push sends a web page and two icons to the BlackBerry wireless smartphone (one icon to indicate that new content is available on the BlackBerry smartphone (the unread icon) and one icon to indicate that the latest content has already been viewed (the read icon)). The web page is stored in the cache of the BlackBerry Browser, and an icon is shown on the Home screen of the smartphone. This provides the user with a visual indication as to when a page updates. It is useful for a news service so users are aware of when new data has arrived. The icons can change with every push, allowing an application to continually push different icons. This is useful for a weather application updating the current weather conditions. An icon displays on the Home screens of users’ BlackBerry smartphones to indicate whether users viewed the latest version of the web content that the Browser Push engine has pushed to their BlackBerry smartphones. After selected, the icon changes state (optional) and then calls the browser to load a cached page. Browser Channel Push is used where the server encodes the contact data in an HTML page and pushes the page to the BlackBerry Browser. When the BlackBerry Browser receives a Channel Push, it stores the pushed content in its cache and adds an icon to the main ribbon that acts as a bookmark to that page. In this form, the program on the BlackBerry smartphone is the BlackBerry Browser, so no ECL-specific code needs to be installed on the smartphone.

Your server-side application can push content to the BlackBerry smartphone to create or update a browser channel. A browser channel is an alternative entry point to the BlackBerry Browser application that displays pushed content instead of the user-defined home page that typically appears when the user opens the BlackBerry Browser. Browser channels appear to the user as a unique application, with an icon.
on the BlackBerry device Home screen. When the user clicks the icon, the BlackBerry Browser opens and pulls the content from the pushed content cache.

You must uniquely identify a browser channel by using a channel ID, specified by the X-Rim-Push-Channel-ID header. If a channel with the same channel ID already exists on the device, the associated content is replaced by the new content. The channel ID is also used to identify the channel in channel delete requests or delete notifications sent when the user deletes the channel.

You must post these icons on your web server and specify the URLs for each icon in the push request by using the X-Rim-Push-Unread-Icon and X-Rim-Push-Read-Icon headers. When the BlackBerry MDS Connection Service receives the browser channel push request, it retrieves the icons from the specified locations and delivers them to the recipient BlackBerry smartphones along with the pushed content. If you do not include these headers, the BlackBerry smartphone uses the icons that you specified in a previous push request associated with that channel ID, if they exist, or the system default icon.

To push to a browser channel, specify a value of browser-channel for the X-Rim-Push-Type header. You must include the also include the X-Rim-Push-Channel-ID header to specify a unique identifier for the channel. You can also choose to include one or more of the following optional HTTP headers:

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-Rim-Push-Title</td>
<td>Specifies a title used to identify the push application on the Home screen</td>
</tr>
<tr>
<td>X-Rim-Push-Unread-Icon-URL</td>
<td>Specifies the URL for the icon image that is displayed on the Home screen when new content has arrived</td>
</tr>
<tr>
<td>X-Rim-Push-Read-Icon-URL</td>
<td>Specifies the URL for the icon image that is displayed after the pushed content has been viewed</td>
</tr>
<tr>
<td>X-Rim-Push-Delete-URL</td>
<td>Specifies the URL to which notifications are sent if the user deletes the channel from the Home screen</td>
</tr>
</tbody>
</table>

**Browser Channel Delete Push**

A Browser Channel Delete works in a similar, but opposite fashion of the Browser Channel Push. It removes an icon on the BlackBerry Home screen that has been sent using a Browser Channel Push. Icons initiating from a Browser Channel Push can also be removed from the smartphone itself.

There are two ways that a browser channel can be deleted from the Home screen: The server-side push application can submit a browser channel delete request to delete the browser channel or the user can delete the browser channel manually. The icon that is associated with the browser channel is removed from the Home screen, and the associated content is cleared from the pushed content cache. If the server-side push application pushes content to the same channel ID, the channel reappears.
There might be instances when you want to remove a browser channel from a BlackBerry smartphone; for example, perhaps the push service is obsolete, or the data is time-sensitive and should only be available for a specific time period, or perhaps a specific user no longer needs to receive data pushed to that browser channel. The server-side push application can remove a browser channel from a BlackBerry device at any time by submitting a channel delete request. The channel delete request only deletes the instance of the specified browser channel that is currently stored on the BlackBerry smartphone. If the server pushes content to that browser channel again, it reappears on the smartphone Home screen.

To delete a browser channel, specify a value of browser-channel-delete for the X-Rim-Push-Type header. If you specify a value of browser-channel-delete, you must also include the X-Rim-Push-Channel-ID header to indicate which channel should be deleted.

A BlackBerry user can delete a browser channel at any time. However, when the BlackBerry smartphone user deletes a channel, they only delete the instance of the specified browser channel that is currently stored on the BlackBerry smartphone. If the server pushes content to that browser channel again, the channel is reopened, and the icon once again appears on the Home screen. In most cases, when a user deletes the icon from the Home screen, the push application should be designed to remove that user from the list of recipients for future push requests. You can construct your push request so that you receive a notification when the user has deleted the browser channel. Receiving a notification allows the push service administrator to either remove that user from the list of recipients, or whether the user should continue to receive content that is pushed to that browser channel.

You can include the X-Rim-Push-Delete-URL to specify a URL to which a notification is sent when the user deletes the channel. When the user deletes a browser channel, the browser sends a GET request to the Delete URL. The Delete URL is not opened for the user to view; instead, this URL is retrieved in the background. By retrieving the URL, the BlackBerry device notifies the push originator that the channel no longer exists on the smartphone.

When the browser sends the GET request to the URL, it does not supply any parameters describing the channel being deleted. You must add parameters to the URL so that you can uniquely identify the push when the URL is requested.

X-Rim-Push-Delete-URL: http://myserver/deleteNotify.php?pushID=45&user=54

An application that pushes data to the browser cache can include an expiry time that defines the length of time that the data remains in the cache before it is removed. The default length of time that push content is stored in cache memory varies with the BlackBerry Browser software version used. In BlackBerry Device Software 3.7 or earlier, the BlackBerry Browser clears the cache after 29 days. In BlackBerry Device Software 3.8 or later, the BlackBerry Browser clears the cache after 12 hours. In BlackBerry Device Software 4.2 or later, if the content URL has a query, the content is not cached. Expired content is cleared from the pushed content cache, even if the user has not viewed it.

To increase or decrease the length of time that the cache memory stores the content, specify a date and time in the Expires header in the push request’s HTTP header. The following example stores content in cache until September 30, 2009 at 8:00 am:
Browser Cache Push

The Browser Push engine pushes web content to the cache of the BlackBerry Browser on users’ BlackBerry smartphones. To view the web content, users browse to the appropriate web address using the BlackBerry Browser. A Browser Content Cache Push pushes a web page to the BlackBerry smartphone and stores it in the BlackBerry Browser’s cache. Browser Content Cache Push is the most discreet type of Browser Push. The user is not informed when this push takes place, or when the page has been updated.

Your server-side application can push content directly to the BlackBerry Browser cache. The user receives no notification indicating that new content is available, but the next time that the user visits the specified URL, the browser retrieves the content from the cache. Browser cache push requests can be used in conjunction with browser channel push requests or browser message push requests to preload the cache with external resources, such as images or style sheets, that might be referenced by other pushed content but are not included. If these resources are pushed to the cache, users can display the content quickly, even if they are not connected to a wireless or Wi-Fi® network. Otherwise, the BlackBerry Browser must make an HTTP request to retrieve referenced resources from the server, which reduces the benefit of a push application.

You can include the X-Rim-Push-Channel-ID header with a browser cache push to associate the cached content with an existing browser channel. For example, if a channel identifier is specified in the request, when the content is pushed to the cache, the URL of the content is added to the appropriate channel (if the channel is already active on the BlackBerry smartphone).

To push to the BlackBerry Browser cache, specify a value of browser-content for the X-Rim-Push-Type header. You can also choose to include the X-Rim-Push-channel-ID header, which specifies the channel ID of the browser channel with which to associate the content.

Clearing a browser cache

1. On the Home screen, click the Browser icon.
2. Press the Menu key.
3. Click **Options**.

4. Click **Cache Operations**.

5. Beside a cache type, click **Clear**.

6. Press the **Escape** key.

### Browser Message Push

Instead of creating an icon on the Home screen of the BlackBerry smartphones, a Browser Message Push arrives in the Inbox on the smartphone to provide a link to new or updated web pages. When the user opens the message, the default browser on the BlackBerry smartphone is launched and then opens the page. Unlike the Browser Channel Push or Browser Content Push, the web page is stored as a browser message instead of in the BlackBerry Browser's cache. This means that the page is available until the user deletes the message. It is not removed from the BlackBerry Browser cache when other events (browsing, cache clear) take place. The message created from a Browser Message Push is not sent to, and does not appear in, a user's Inbox on their desktop. The message is sent directly to the smartphone. A message appears in the message list on users' BlackBerry smartphone to provide a link to new or updated web content. A link to the web address is stored as an email message.

Your server-side application can push content pages to the messages application on the BlackBerry smartphone. A browser message push request can include a descriptive title, which appears as the subject of the message in the messages application. Otherwise, the browser message displays the URL of the associated content as the subject of the message. Users can click on the message in the messages application to open the BlackBerry Browser and view the content.

When you push content to the messages application, the pushed content is stored within the browser message item, not in the BlackBerry Browser push content cache. When the user deletes the message, the pushed content is deleted with it. When a browser message arrives on the BlackBerry smartphone, a notification icon appears in the BlackBerry smartphone banner. If the user has configured it to do so, the BlackBerry smartphone issues an alert. By default, the BlackBerry smartphone provides no notification when a browser message arrives.

To push content to the messages application, specify a value of browser-message for the **X-Rim-Push-Type** header. You can also choose to include the **X-Rim-Push-Title** header, which specifies a title used as the subject of the message when it appears in the messages application.

### Data Push

Pushed data are transmitted to the BlackBerry smartphone with the expectation that a customer listener, as opposed to the browser, has been set up to accept the pushed data.
Browser Push options

Process MDS—Set whether your BlackBerry smartphone accepts Browser Push messages from the BlackBerry Enterprise Server or BlackBerry® Internet Service.

MDS Hosts—Set whether your BlackBerry smartphone accepts push messages from all BlackBerry Enterprise Servers or BlackBerry Internet Services, or no BlackBerry Enterprise Servers or BlackBerry Internet Services.

Process SMSC—Set whether your BlackBerry smartphone accepts SMS-based Browser Push messages.

SMSC Hosts—Set which SMSC hosts your BlackBerry smartphone accepts Browser Push messages from. To accept Browser Push messages from a specific SMSC host, type the phone number for the SMSC host.

Process IP—Set whether your BlackBerry smartphone accepts Browser Push messages that web applications send through a WAP gateway.

IP Hosts—Set which WAP gateways your BlackBerry smartphone accepts Browser Push messages from. To accept Browser Push messages from a specific IP host, type the IP address for the WAP gateway.
1. An icon displaying on the Home screen of a user’s BlackBerry smartphone to indicate whether the user viewed the latest version of pushed web content is which type of Browser Push?
   A. Browser Channel Push
   B. Browser Cache Push
   C. Browser Message Push
   D. Data Push

2. When a link to the web address is stored as an email, which type of Browser Push is being used?
   A. Browser Channel Push
   B. Browser Cache Push
   C. Browser Message Push
   D. Data Push

3. New web content stored in the cache of the BlackBerry Browser on the BlackBerry smartphone is which type of Browser Push?
   A. Browser Channel Push
   B. Browser Cache Push
   C. Browser Message Push
   D. Data Push

4. When a custom listener, as opposed to the browser, has been set up to accept the pushed information, which type of push is being implemented?
   A. Browser Channel Push
   B. Browser Cache Push
   C. Browser Message Push
D. Data Push

5. Which of the following does not have a Browser Push process or host option?
   A. MDS
   B. SMSC
   C. WAP
   D. IP

6. It is necessary to create your own client-side listener application with Browser push. True or false?
   A. True
   B. False

7. The user receives notification indicating that new content is available in the BlackBerry Browser cache. True or false?
   A. True
   B. False

8. In what two ways can a browser channel be deleted from the Home screen?

9. When the user deletes a Browser push message, the pushed content is deleted with it. True or false?
   A. True
   B. False

10. A browser channel is an alternative entry point to the BlackBerry Browser application that displays pushed content instead of the user-defined home page that typically appears when the user opens the BlackBerry Browser. True or false?
    A. True
    B. False
Introduction to Browser Push

**Answers**

1. A
2. C
3. B
4. D
5. C
6. B
7. B
8. The server-side push application can submit a browser channel delete request to delete the browser channel or the user can delete the browser channel manually.
9. A
10. A
Writing a Browser Push application

You can use three methods to create applications for BlackBerry smartphones. You can create a web site to be accessed using the BlackBerry Browser or you can develop a server-based application that pushes content such as a web page and images, or data in general, to the the BlackBerry smartphone. You can also write a software application that operates on the BlackBerry smartphone and on a remote server.

Design considerations

When you begin to develop push applications, consider the type of information that users in your organization need pushed to them, and how often you must push that information. For example, some users require only event-based information that is a response to a business event, such as changes to a contacts list or a notification when inventory levels drop below a specified level. Other users might require recurring reports pushed to them. The use drives the type of push mechanism that you choose.

Consider having users subscribe to the push information they require. The users can manage their own subscriptions or an administrator can manage the subscriptions centrally. Subscriptions are not specific to an application.

Design overview

The development of most push applications requires the following tasks:

- Create a web address for the location where you post data.
- Fetch or create the content for the push application.
- Identify the email addresses or BlackBerry smartphone personal identification numbers (PINs) of the content recipients.

Push Application process flow

1. The push application sends an HTTP POST request to the BlackBerry MDS using the following format:

   post/push?DESTINATION=<destination>&PORT=7874&REQUEST-URI=<request_URI> <header><content>

2. The BlackBerry MDS receives the HTTP request. It stores the destination and port information, and then formats the request as follows:

   post <request_URI><header><content>
3. If applicable, the BlackBerry MDS applies a transcoder according to its transcoder rules.

4. The BlackBerry MDS sends the content to the BlackBerry smartphone.

**Example of a RIM Push Application in Java**

This section demonstrates how to write a Channel Push application that uses the RIM push service implementation to create an icon on the Home screen of the BlackBerry smartphone.

This example provides sample code in Java®, but you can use the programming language of your choice, including C#, C++, perl, or Microsoft.NET. Visit www.blackberry.com/developers/journal/ for sample code in other languages.

**Write the Push Method**

1. Define a method that accepts the parameters for the channel push submission.

   ```java
   public static void pushPage(String mdsHostName, int mdsPort,
   String email,
   String pushUrlString, String pushType, String pushTitle,
   String unreadIconUrl, String readIconUrl,
   String pushReliability, String notifyUrl,
   String PushId) {
   ```

2. Define variables for connections to two web addresses: one to the server with the content and one to the BlackBerry MDS Services.

   ```java
   HttpURLConnection pushConn;
   HttpURLConnection mdsConn;
   URL pushUrl;URL
   mdsUrl;
   ```

3. Create the connection to the BlackBerry MDS Services by invoking the openConnection method on the web address. The push listener thread on the BlackBerry smartphone listens on port 7874.

   ```java
   mdsUrl = new URL("http", mdsHostName, mdsPort, "push?DESTINATION=" + email +
   ```
"&PORT=7874&REQUESTURI="/";
mdsConn = (HttpURLConnection)mdsUrl.openConnection();

4. Set additional header properties for the Channel Push.

mdsConn.setRequestProperty("Content-Location", pushUrlString);
mdsConn.setRequestProperty("X-Rim-Push-Title", pushTitle);
mdsConn.setRequestProperty("X-Rim-Push-Type", pushType);
if (pushType.equals(CHANNEL) || pushType.equals(CHANNEL_DELETE)) {
    mdsConn.setRequestProperty("X-Rim-Push-Channel-ID", pushUrlString);
    if (pushType.equals(CHANNEL)) {
        mdsConn.setRequestProperty("X-Rim-Push-UnRead-Icon-URL",
                                   unreadIconUrl);
    }
    mdsConn.setRequestProperty("X-Rim-Push-Read-Icon-URL",
                               readIconUrl);
}
mdsConn.setRequestProperty("X-Rim-Push-Reliability", pushReliability);
mdsConn.setRequestProperty("X-Rim-Push-NotifyURL", notifyUrl);
mdsConn.setRequestProperty("X-Rim-Push-ID", pushId);
try {
    mdsConn.setRequestMethod("POST");
} catch (ProtocolException e) {
    throw new RuntimeException("problems setting request method: " + e.getMessage());
}
mdsConn.setAllowUserInteraction(false);
mdsConn.setDoInput(true);
mdsConn.setDoOutput(true);

5. Invoke the openConnection method on the web address to connect to the content server.

pushUrl = new URL(pushUrlString);
pushConn = (HttpURLConnection)pushUrl.openConnection();

6. Set properties for the HTTP GET request to the content server.

   pushConn.setAllowUserInteraction(false);
   pushConn.setDoInput(true);
   pushConn.setDoOutput(false);
   pushConn.setRequestMethod("GET");

7. Open the connection to the content server.

   pushConn.connect();

8. Set properties for the HTTP POST request to the BlackBerry MDS Services. Copy the header properties from the GET request and then set additional properties for the push submission.

   String name;
   String value;
   for (int i = 0; true; i++) {
       name = pushConn.getHeaderFieldKey(i);
       value = pushConn.getHeaderField(i);
       if (((name ==null) && (value == null)) { break; }
       if (((name ==null) || (value == null)) { continue; }
       if (name.equals("X-Rim-Push-Type")) { continue; }
       if (name.equals("Transfer-Encoding")) { continue; }
       System.out.println("setting header property " + name + "=" + value);
       mdsConn.setRequestProperty(name, value);
   }
9. Read content from the content server connection and write it to the BlackBerry MDS Services connection.

```java
InputStream ins = pushConn.getInputStream();
OutputStream outs = mdsConn.getOutputStream();
copyStreams(ins, outs);
```

**Note:**
This step is optional for applications that use the Channel Push method.

10. Open the connection to the BlackBerry MDS Services.

```java
mdsConn.connect();
...
```

11. Retrieve the response code that is returned from the BlackBerry MDS Services, and display a status message.

```java
int rescode = mdsConn.getResponseCode();
if (rescode != HttpURLConnection.HTTP_OK) {
throw new RuntimeException("Cannot push data, received bad response code from Mobile Data Service: " + rescode);
}
System.out.println("Pushed page to the handheld");
```

**Test the example**

The BlackBerry Developer site, www.blackberry.com/developers, provides downloads for the BlackBerry Web Plug-in for Eclipse. The BlackBerry Web Plug-in for Eclipse (package) contains a BlackBerry MDS simulator and different BlackBerry smartphone simulators. You can use the simulators to test applications on multiple BlackBerry smartphones in a controlled nonproduction environment.

You can use the BlackBerry MDS simulator in the BlackBerry Web Plug-in for Eclipse for both pull and push testing. The BlackBerry MDS simulator works from a desktop computer and uses both the desktop connection and the enterprise environment.
1. On a web application server, deploy the web content and push icons that you want to push to BlackBerry smartphone.

2. Place the `BrowserChannelPush.java` and `rim_browserpush.properties` files into a `com\rim\docs\samples\browserchannelpush` folder that is in your CLASSPATH.

3. Edit the `rim_browserpush.properties` file to specify the location and names of the files to push to the BlackBerry devices.

4. Start the BlackBerry MDS Services and BlackBerry smartphone simulators.

5. Turn on the BlackBerry MDS Services push functionality in the BlackBerry smartphone simulator.

6. Compile and run the Java file.

7. In the BlackBerry smartphone simulator, the icon that is specified in the `unreadIconUrl` property displays on the device Home screen. Click the icon to view the web page that is specified in the `pushUrlString` property.

**Troubleshooting**

Each mobility environment is unique, with specific BlackBerry MDS configuration settings and different security policies implemented. Issues can arise across a range of users when an application is mobilized.

**General Push submission issues**

- Check that each BlackBerry smartphone is enabled for the BlackBerry MDS and that the BlackBerry Browser is installed on the BlackBerry smartphone. If a smartphone is not enabled, content sent to the BlackBerry MDS is not pushed to the user’s BlackBerry smartphone.

- Make sure that push messages reach the BlackBerry MDS. If you have multiple BlackBerry Enterprise Servers in a single environment, you can set up one as a central push server. Applications can push new or updated applications to the central push server. The central push server then manages the delivery of messages to the other BlackBerry Enterprise Servers with the BlackBerry MDS installed.

- Confirm that push submissions are directed to the correct BlackBerry Enterprise Server. The server must be able to recognize the recipient.

- Make sure that specified listening ports are correct.

- Verify network connectivity to the BlackBerry MDS, and make sure there is not a firewall or other security device blocking port 8080 or port 8300.

- Ensure that the BlackBerry smartphone can access the web addresses of the content that is pushed by the BlackBerry Browser.
• Ensure that the BlackBerry MDS is accepting the push request and acknowledging it with a 200 OK response. (See Response codes on the next page.)
• If a BlackBerry smartphone is not in a coverage area and a push request expires in the MDS Services on the BlackBerry Enterprise Server, the push submission is not delivered to the BlackBerry smartphone. When this occurs, pushed content can be sent multiple times.

### Response codes

<table>
<thead>
<tr>
<th>Response code</th>
<th>What it means</th>
<th>Possible Cause</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK. The push was accepted successfully by the BlackBerry MDS.</td>
<td>A valid push has occurred.</td>
</tr>
<tr>
<td>400</td>
<td>General Error</td>
<td>Invalid push format—In the BlackBerry MDS version 3.7 and preceding versions, the email address or BlackBerry smartphone PIN used in the push submission was not recognized as belonging to a BlackBerry Enterprise Server user who is enabled on the BlackBerry MDS. The BlackBerry MDS versions 4.0 and later use response code 403 instead of code 400.</td>
</tr>
<tr>
<td>403</td>
<td>Access control error or unknown email address or BlackBerry smartphone PIN specified</td>
<td>The email address or BlackBerry smartphone PIN used in the push submission was not recognized as belonging to a BlackBerry Enterprise Server user who is enabled on the BlackBerry MDS. Push access control has been enabled on the BlackBerry Enterprise Server, and the user or computer that is submitting the push has not been granted permission to perform a push submission.</td>
</tr>
<tr>
<td>404</td>
<td>Page not found</td>
<td>A possible cause is that the push request was not received by the BlackBerry MDS. Verify that the correct web address and port are used in the push submission to the BlackBerry MDS.</td>
</tr>
<tr>
<td>Response code</td>
<td>What it means</td>
<td>Possible Cause</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>503</td>
<td>Server is busy</td>
<td>The BlackBerry MDS Connection Service is busy or is unable to manage the push request due to temporary overloading or server maintenance.</td>
</tr>
</tbody>
</table>

**Note:**
This response code was added in the BlackBerry MDS version 4.0. Previous versions used response code 400 to respond to this error.
1. The development of most push applications requires which of the following tasks?
   A. Create a web address for the location where you post data.
   B. Fetch or create the content for the push application.
   C. Identify the email addresses or BlackBerry smartphone PIN of the content recipients.
   D. All of the above

2. When writing a Browser Push application, which of the following considerations is not specific to the application?
   A. How to manage the subscription
   B. How often to push the information
   C. Types of information being pushed
   D. Type of push mechanism being used

3. What is the final step when testing the application in the smartphone simulator?
   A. Deploy the web content.
   B. Click the icon to view the web page.
   C. Compile and run the Java file.
   D. Place the necessary files in your CLASSPATH.

4. What does the response code 200 mean?
   A. Page not found
   B. General error
   C. OK. The push was accepted successfully by the BlackBerry MDS.
   D. Access control error
5. Only the server can remove the pushed content from the BlackBerry smartphone. *True or false?*
   A. True
   B. False

6. An application that pushes data to the browser cache can include an expiry time that defines the length of time that the data remains in the cache before it is removed. *True or false?*
   A. True
   B. False
Answers

1. D
2. A
3. B
4. C
5. C
6. A
How Browser Push benefits mobile applications

BlackBerry Push APIs offer a number of concrete benefits for BlackBerry users and application developers. Browser Push offers the following benefits:

Provides information immediately—Pushing data to the BlackBerry smartphone is the most time-efficient way to get information to BlackBerry users. With BlackBerry Push APIs, developers can deliver data to their Java applications on the BlackBerry smartphone as it becomes available, when it matters. In addition, developing applications using BlackBerry Push APIs considerably reduces the impact of network latency. Because wireless networks have considerably less bandwidth than wired networks, data transfer rates are much slower over wireless connections. However, because BlackBerry Push APIs deliver data to BlackBerry smartphones in the background without user involvement, from the user’s perspective, there is no wait time. The most recent data is synchronized and available on their BlackBerry smartphones the moment they open the application.

Optimizes network efficiency—For retail consumers, most of whom pay for a wireless data plan, network efficiency is paramount. Each network transaction has a financial implication, because exceeding their data allotment incurs charges from their wireless service provider. As a result, frivolous or excessive network usage has a tangible financial cost to the user. Applications that are developed using BlackBerry Push APIs are very network-efficient. Firstly, applications no longer need to repeatedly poll servers for new data. Although these polling requests and responses are small individually, they can add up quickly with frequent polling intervals across multiple applications. In polling scenarios, the user receives limited return on their data investment. Secondly, BlackBerry Push APIs are designed to push consumable amounts of data. Content providers can push up to 8KB of data to a BlackBerry smartphone with a push request. If more data is available, the user can decide whether to retrieve it or not, providing them a measure of control over how much data they transfer, and by extension, how much they pay.

Preserves the battery life of the BlackBerry smartphone—Preserving battery life is an important consideration for developers of wireless smartphone applications. After all, applications can only function if the host device has enough battery power to operate. To retrieve needed data from a server, some applications repeatedly poll the servers to determine whether new data is available. These applications can quickly drain the device battery, reducing the effectiveness of that application, and worse, compromising the convenience of the BlackBerry smartphone itself. BlackBerry Push APIs minimize the impact on battery life. Instead of actively checking for new data, the application simply listens in the background for the server to push data to it. Once delivered, the application can process it as required.

Reduces complexity for developers—BlackBerry Push APIs build upon the established WAP PAP 2.2 standard, which uses XML to define the push parameters for the BlackBerry Infrastructure. Because the BlackBerry Infrastructure handles the process of managing and delivering push requests, much of the complexity of a pushed-based data delivery service is hidden from application developers. If developers are already familiar with the PAP standards, then they already know all that is needed to access BlackBerry Push technology and begin pushing data to users.
Improves developer margins—BlackBerry Push APIs are offered as an addition to BlackBerry development partners. By gaining access to these APIs, development partners receive an even greater return on their investment. Because developers can now create applications which can exploit the unique capabilities that only the BlackBerry Infrastructure provides, they can create applications for BlackBerry users that are much more compelling, that can address a wider variety of use cases, and that result in a true BlackBerry experience for the user. BlackBerry Independent Software Vendor (ISV) Alliance Program members who enhance their applications with BlackBerry Push technology can benefit from this user engagement to drive more traffic back to revenue generators such as transaction, subscription or advertising.

**Delivering pushed content**

You can notify users that new content is available or, depending on the network gateway, you can push new content directly to the BlackBerry smartphone. The level of support for pushing content depends on the network gateway.

**BlackBerry MDS Connection Service**—Among the three network gateways, the BlackBerry MDS Connection Service offers the most support for pushed content delivery. Content is pushed directly to the BlackBerry smartphone and is stored in a dedicated cache for pushed content. You can develop the push service to place an icon on the Home screen (the icon changes to notify the user of new content), to send the content to the message list (that appears like all other messages), or to deliver the content directly to the cache without user notification. When the user next views the content, it is available on the BlackBerry smartphone. The user does not need to download it.

**BlackBerry Internet Service Browsing**—The BlackBerry Internet Service Browsing network gateway supports pushed content delivery. Content is pushed to the BlackBerry smartphone through web signals using a push/pull architecture. You can develop a web signal to place an icon on the Home screen. The icon changes to notify the user that new content is available, prompting the user to download the content.

**WAP network gateways**—WAP network gateways support pushed content delivery, typically of SMS or UDP messages, depending on the network. To support pushed content delivery through WAP network gateways, the WAP Push service record must be provisioned on the BlackBerry smartphone. WAP Push service records are typically provisioned when a BlackBerry smartphone is registered with the wireless service provider.

**Pushed content supported by the BlackBerry MDS Connection Service**

The BlackBerry MDS Connection Service is designed to provide extensive support for pushing content or data to BlackBerry smartphones. To push content to the BlackBerry smartphone, you can develop server-side applications that make HTTP POST requests to the BlackBerry MDS Connection Service. Requests include a list of the destination BlackBerry smartphones that are targeted to receive the pushed content.
content. The BlackBerry MDS Connection Service sends the content to the appropriate BlackBerry smartphones using the users’ email addresses. The BlackBerry MDS Connection Service manages the connection to the wireless network and verifies that content is delivered as soon as a user is in a wireless coverage area. On the BlackBerry smartphone, a separate browser listener thread listens on port 7874 for incoming messages and processes incoming messages. The BlackBerry MDS Connection Service supports the following content push protocols:

- **RIM® Push**—The RIM Push protocol sends the content as a byte stream to the destination BlackBerry smartphone to the port that is specified in the web address of the pushed message. Pushed data can be stored in RAM on the BlackBerry MDS Connection Service server or in the BlackBerry Enterprise Server database.

- **WAP PAP Push**—The PAP sends an HTTP POST request containing a PAP message. The message is a MIME multipart message that includes the control entity and the pushed content. The control entity is an XML document that specifies information about the destination BlackBerry smartphone address, message ID, and delivery time stamps.

**BlackBerry Internet Browser**

The BlackBerry Internet Browser was developed to use the BlackBerry Internet Service as a gateway to the Internet. This component is hosted by the BlackBerry Infrastructure and is a service offered by certain carriers. The features provided are similar to those provided through MDS. As an example, both are designed to optimize web content for wireless browsing, and both transcode content types into appropriate formats for display on the BlackBerry smartphone. The key differences between the BlackBerry Internet Service and MDS are as follows:

- BlackBerry Internet Service does not require BlackBerry Enterprise Server.
- BlackBerry Internet Service is not used through a corporate firewall.
- BlackBerry Internet Service is made available through selected BlackBerry service providers.
- BlackBerry Internet Service does not support Triple DES encryption, and secure sites (HTTPS) are not accessible.

BlackBerry Internet Service communicates with the BlackBerry Infrastructure using HTTP over the RIM IP Proxy Protocol (IPPP). Delivery of HTML is both faster and more efficient than HTTP over WAP in most current implementations.
1. How have mobile applications benefited from Browser Push?
   A. Increased network latency
   B. Increased complexity for developers
   C. Immediate access to information
   D. Polling servers repeatedly for new data

2. Which of the following network gateways offers the most support for pushed content delivery?
   A. BlackBerry MDS Connection Service
   B. BlackBerry Internet Service Browsing
   C. WAP Network Gateway
   D. Windows CE Network Gateway

3. For which of the following gateways is content pushed to the BlackBerry smartphone through web signals using a push/pull architecture?
   A. BlackBerry MDS Connection Service
   B. WiFi Network Gateway
   C. WAP Network Gateway
   D. BlackBerry Internet Service Browsing

4. Client-side applications make HTTP POST requests to the BlackBerry MDS Connection Service. True or false?
   A. True
   B. False
Answers

1. C
2. A
3. D
4. B
Chapter 5

Browser Push Transcoding

If it has been configured to do so, the BlackBerry MDS Connection Service automatically transcodes pushed content before it sends it to the BlackBerry Browser. When the BlackBerry MDS Connection Service transcodes content, it changes the pushed resources into formats that are more suitable for sending over the wireless network and for rendering in the BlackBerry Browser.

Push requests can override the transcoding rules configured for the BlackBerry MDS Connection Service using the X-Rim-Transcode-Content header. This header can be used to specify a list of MIME types that the BlackBerry MDS Connection Service should transcode, should it encounter them. For example, if the HTTP header X-Rim-Transcode-Content: application/xhtml+xml is set, then the BlackBerry MDS Connection Service transcodes XHTML content before pushing it to the BlackBerry smartphones.

You can also specify a value of none to prevent the BlackBerry MDS Connection Service from transcoding any content. For best results, you should typically allow the BlackBerry MDS Connection Service to transcode all your content. Although most often used with browser content, the X-Rim-Transcode-Content header can also be used in conjunction with a BlackBerry Java Application, if a transcoder exists on the BlackBerry MDS Connection Service for the data format you are sending.

The Transcoder API provides the classes and interfaces for an encoding scheme for data sent to and from BlackBerry smartphones and the BlackBerry Enterprise Server located in your organization. You can use the encoding scheme to encode and decode all Gateway Messaging Envelope (GME) packets that the BlackBerry Enterprise Server sends to the BlackBerry smartphone and all GME packets that the BlackBerry device sends to the BlackBerry Enterprise Server.

If the BlackBerry Enterprise Server administrator allows third-party applications to use the Transcoder API on the BlackBerry smartphone, those applications, if not functioning correctly, might affect the security, usability, and performance of the BlackBerry Enterprise Solution, and might cause loss of BlackBerry smartphone data.

RIM provides a Java interface and a C++ header file. These files define the methods required to encode and decode messages. The BlackBerry smartphone uses the Java interface; the BlackBerry Enterprise Server uses the C++ header file. The BlackBerry device and BlackBerry Enterprise Server pass all incoming and outgoing data messages to the Transcoder implementation. Because two threads can call the encoding or decoding method at the same time, the implementation must be synchronized on every method call.

Device-Side interface

The Transcoder implementation on the BlackBerry smartphone filters which messages to encode or decode using the willTranscode() method and ensures that a corresponding decoding method exists on the BlackBerry Enterprise Server before it encodes the message.

The Transcoder implementation is responsible for the key management that the encoding scheme requires. The BlackBerry device calculates a CRC of the message before the Transcoder implementation
encodes the message. The BlackBerry device uses the CRC to check that the corresponding decoding process is accurate and the transmission is error free. The CRC is not used to prevent malicious Transcoder implementations.

If encoding succeeds, the code provided in Transcoder API prepends the identifier of the Transcoder to the front of the encoded data. The standard BlackBerry decryption process checks this identifier before it passes the data to the decoder.

The implementation is responsible for providing useful exception handling and logging. If the Transcoder implementation throws an exception, the message is dropped and the failure is logged. The logs display a unique GUID to distinguish them from other log entries. You must implement the following methods for the device-side Transcoder implementation:

- Transcoder()—This constructor creates an instance of a Transcoder.
- getID()—This method returns a byte that uniquely identifies this instance of the Transcoder.
- willTranscode()—This method queries the Transcoder with a context to see if the Transcoder will perform an operation in the supplied context. This method filters calls to the encode and decode methods.
- encode()—This method encodes the data in an input stream and outputs the encoded data into an output stream.
- decode()—This method decodes the data in an input stream and outputs the decoded data into an output stream.

**Server-Side interface**

The Transcoder implementation on the BlackBerry Enterprise Server filters the messages to encode and decode, throws exceptions, logs any encoding and decoding errors, and provides the key management required as part of the encoding scheme. To set up the implementation file, complete the following:

1. Download the BEStranscoderAPI.h file. To download the BESTranscoderAPI.h file, visit the BlackBerry Developer’s web site and search for transcoderapiheaderfile.

2. In the main file of the server-side Transcoder implementation, include the following line: 
   ```
   #include “BESTranscoderAPI.h”.
   ```

3. To use the helper macro provided by the Transcoder API, include the following line:
   ```
   DEFINE_BES_TRANSCODER_DLL. This macro ensures that the required function definitions are present.
   ```

You must implement the following methods for the server-side Transcoder implementation:

- LoadDLL()—This function is called once when the BlackBerry Enterprise Server loads the DLL.
- FreeDLL()—This function is called once when the BlackBerry Enterprise Server unloads the DLL.
- GetID()—This function returns a byte that uniquely identifies this instance of the Transcoder.
• WillTranscode()—This function queries the Transcoder with a context to see if the Transcoder will perform an operation in the context.
• Encode()—This function encodes the data in an input stream and outputs the encoded data into an output stream.
• Decode()—This function decodes the data in an input stream and outputs the decoded data into an output stream.

Transcoding data on the BlackBerry smartphone

Before the Transcoder implementation can encode or decode messages, it must register with the TranscoderManager on the Blackberry smartphone by calling the TranscoderManager register() method. The TranscoderManager is available to signed clients. The TranscoderManager can only recognize one Transcoder implementation. If you attempt to register two Transcoder implementations, the BlackBerry device registers only the first implementation. If a Transcoder implementation fails to register with the TranscoderManager, the TranscoderManager throws a TranscoderRegistrationException.

The process flow for encoding data on the BlackBerry smartphone entails the following steps:

1. The BlackBerry smartphone prepares a message to send.
2. If a registered Transcoder implementation exists on the BlackBerry smartphone and the message is targeted for the BlackBerry Enterprise Server bound through IT Policy, the BlackBerry smartphone calculates a CRC of the datagram and uses the Transcoder implementation to encode both the CRC and the datagram.
3. The Transcoder implementation prepends the Transcoder ID to the encoded data.
4. The BlackBerry smartphone processes the encoded data using the standard BlackBerry encryption and compression process.

The process flow for decoding data on the BlackBerry smartphone entails the following steps:

1. The BlackBerry smartphone receives the datagram and decrypts and decompresses the data using the standard BlackBerry decryption and decompression process.
2. If the Transcoder ID prepended to the data matches the Transcoder implementation registered on the BlackBerry smartphone, the Transcoder implementation decodes the data.
3. The BlackBerry smartphone calculates the CRC of the decoded datagram. If the CRC matches the CRC appended to the decoded data, the transcoding succeeds.

Transcoding data on a BlackBerry Enterprise Server

To register the Transcoder implementation on the BlackBerry Enterprise Server requires the following tasks. On the BlackBerry Enterprise Server, the Transcoder implementation is a DLL. The location of the
Transcoder DLL is specified by the following registry key: HKEY_LOCAL_MACHINE\SOFTWARE\Research In Motion\BlackBerry Enterprise Server\Dispatcher\Transcoder. A string value named Transcoder holds the full path and file name of the DLL.

On the BlackBerry Enterprise Server, the BlackBerry Dispatcher service is designed to handle the standard BlackBerry encryption process. On startup of the BlackBerry Enterprise Server, the BlackBerry Dispatcher service loads the Transcoder DLL specified by the registry key. If a Transcoder implementation is specified, the BlackBerry Dispatcher sets a flag, which alerts the BlackBerry encryption process to send data to the Transcoder implementation. The BlackBerry Dispatcher must be restarted to load the Transcoder implementation; it does not support dynamic loading and unloading of Transcoder implementations.

The process flow for encoding data on the BlackBerry Enterprise Server entails the following steps:

1. The BlackBerry Enterprise Server prepares a datagram to send.
2. If a registered Transcoder implementation exists on the BlackBerry Enterprise Server, the BlackBerry Enterprise Server calculates a CRC of the datagram and uses the Transcoder implementation to encode both the CRC and the datagram.
3. The Transcoder implementation prepends the Transcoder ID to the encoded data.
4. The BlackBerry Enterprise Server processes the encoded data using the standard BlackBerry encryption and compression process.

The process flow for decoding data on the BlackBerry Enterprise Server entails the following:

1. The BlackBerry Enterprise Server receives a datagram and decrypts and decompresses the data using the standard BlackBerry decryption and decompression process.
2. If the Transcoder ID prepended to the data matches the Transcoder implementation registered on the BlackBerry Enterprise Server, the Transcoder decodes the data.
3. The BlackBerry Enterprise Server calculates the CRC of the decoded datagram. If the CRC matches the CRC appended to the decoded datagram, the transcoding succeeds.

**Security on the BlackBerry Enterprise Server**

The BlackBerry Enterprise Solution permits only Transcoder implementations that the RIM® signing authority system has digitally signed to access the Transcoder API. The RIM signing authority system uses the RIM Cryptographic API key to authorize and authenticate the Transcoder implementation code. All Transcoder implementations must have their .cod files signed by the BlackBerry® Signing Authority Tool.

A Transcoder implementation can register with the TranscoderManager on the BlackBerry device if the hash of the .cod file is specified in the IT policy on the BlackBerry device. The BlackBerry Enterprise Server administrator must use the Security Transcoder Cod File Hashes IT policy rule to specify the hash.
of the .cod file for the third-party encryption scheme that the BlackBerry smartphone permits to register as a Transcoder implementation. The BlackBerry smartphone compares the hash of the third-party encoding scheme .cod file with the hashes specified in the Security Transcoder Cod File Hashes IT policy rule. This measure is designed to prevent unauthorized third-party encoding schemes from registering with the TranscoderManager on the BlackBerry smartphone. For more information about using the Security Transcoder Cod File Hashes IT policy rule, see the BlackBerry Enterprise Server Policy Reference Guide.

If the IT policy is not specified, a Transcoder implementation cannot register with the TranscoderManager and the TranscoderManager throws a TranscoderRegistrationException. You must have BlackBerry Enterprise Server administrator permissions to set the IT policy.

To set the IT policy on the BlackBerry smartphone:

1. **Retrieve the .cod file hash or hashes with the following command:**

   ```
   javaloader siblinginfo TranscoderImpl.cod
   ```

   **The output of this command is:**

   RIM Wireless Handheld Java Loader
   Copyright 2001-2007 Research In Motion Limited
   Name Sibling? Num Siblings
   -------------------------- -------- ------------
   TranscoderImpl.cod Yes 2
   TranscoderImpl.cod No
   TranscoderImpl.cod-1 No
   -------------------------- -------- ------------
   Name Hash Offset Length
   -------------------------- -------------------------------------
   TranscoderImpl.cod
   TranscoderImpl.cod b02845188bd3e2b7d60307ba721505c9260e7a5 52 60044
   TranscoderImpl.cod-1 6b856777430b09f906fecad4d156da52ec2b6033 60146 15212
   -------------------------- -------------------------------------
If you are not sure which sibling .cod file contains the Transcoder implementation, enter the following as the value of the Security Transcoder Cod File Hashes IT policy rule in the IT policy for the BlackBerry smartphone:

b02845188bd3e2b7d60307ba721505c92600e7a5;6b856777430b09f906fecad4d156da52ec2b603

If you know that the Transcoder implementation is in the TranscoderImpl.cod-1 sibling file, enter the following as the value of the Security Transcoder Cod File Hashes IT policy rule in the IT policy for the BlackBerry smartphone:

6b856777430b09f906fecad4d156da52ec2b6033
1. What is the function of transcoding?
   A. Classes and interfaces for encoding data
   B. Encoding and decoding data sent to and from the BlackBerry smartphone
   C. A and B
   D. None of the above

2. The BlackBerry Enterprise Solution permits only Transcoder implementations that the RIM signing authority system has digitally signed to access the Transcoder API. True or false?
   A. True
   B. False

3. Place the numbers 1 to 4 in the spaces provided below to indicate the order in which the steps occur when encoding data on the BlackBerry Enterprise Server.
   A. ____ The BlackBerry Enterprise Server prepares a datagram to send.
   B. ____ The Transcoder implementation prepends the Transcoder ID to the encoded data.
   C. ____ The BlackBerry Enterprise Server processes the encoded data using the standard BlackBerry encryption and compression process.
   D. ____ If a registered Transcoder implementation exists on the BlackBerry Enterprise Server, the BlackBerry Enterprise Server calculates a CRC of the datagram and uses the Transcoder implementation to encode both the CRC and the datagram.

4. Which of the following functions returns a byte that uniquely identifies an instance of the Transcoder?
   A. Decode()
   B. LoadDLL()
   C. WillTranscode()
   D. GetID()
5. You can specify which of the following values to prevent the BlackBerry MDS Connection Service from transcoding any content.

A. X-Rim-Transcode-Content=<1

B. none

C. X-Rim-Transcode-Content=0

D. zero
Answers

1. C

2. A

3. 1, 3, 4, 2

4. D

5. B
Introduction to Browser Push

Browser Push is designed to push information from a web application to your BlackBerry smartphone as the information becomes available. For example, you can use Browser Push to receive updates for weather reports, the stockmarket, or breaking news. When your smartphone receives an update, a new Browser Push message appears in a message list or on the Home screen of your BlackBerry smartphone.

Browser Push works as follows. The content provider sends a push request and the BlackBerry Infrastructure returns a response. BlackBerry Infrastructure pushes the data to the BlackBerry smartphone and the BlackBerry smartphone returns a response to the BlackBerry Infrastructure. BlackBerry Infrastructure forwards acknowledgement to content provider. A read notification is returned to the BlackBerry Infrastructure.

Browser Push allows you to take advantage of true server-side push communication, which is supported by every BlackBerry smartphone. Browser Push sends information to the browser in a variety of ways (channel, cache, message), regardless of back-end server technology (Apache, IIS, WebSphere) and leverages industry standards such as PAP.

Types of Browser Push include Browser Channel, Browser Message, Browser Cache, and Browser Channel-Delete. The BlackBerry Enterprise Server supports these types of BlackBerry Browser Push applications.

You can use three methods to create applications for BlackBerry smartphones. You can create a web site to be accessed using the BlackBerry Browser or you can develop a server-based application that pushes content such as a web page and images, or data in general, to the BlackBerry smartphone. You can also write a software application that operates on the BlackBerry smartphone and on a remote server.

The BlackBerry Enterprise Server provides push-based access to email, calendar, contacts, tasks and notes, instant messaging, web-based applications, services, and enterprise applications. BlackBerry Enterprise Server strengthens integration between BlackBerry smartphones and corporate enterprise systems, enabling mobile workers to be more productive.

The Transcoder API provides the classes and interfaces for an encoding scheme for data sent to and from BlackBerry smartphones and the BlackBerry Enterprise Server located in your organization. You can use the encoding scheme to encode and decode all Gateway Messaging Envelope (GME) packets that the BlackBerry Enterprise Server sends to the BlackBerry smartphone and all GME packets that the BlackBerry smartphone sends to the BlackBerry Enterprise Server.

An application that pushes data to the browser cache can include an expiry time that defines the length of time that the data remains in the cache before it is removed. The default length of time that push content is stored in cache memory varies with the BlackBerry Browser software version used.

Summary

Browser Push is designed to push information from a web application to your BlackBerry smartphone as the information becomes available. For example, you can use Browser Push to receive updates for weather reports, the stockmarket, or breaking news. When your smartphone receives an update, a new Browser Push message appears in a message list or on the Home screen of your BlackBerry smartphone.

Browser Push works as follows. The content provider sends a push request and the BlackBerry Infrastructure returns a response. BlackBerry Infrastructure pushes the data to the BlackBerry smartphone and the BlackBerry smartphone returns a response to the BlackBerry Infrastructure. BlackBerry Infrastructure forwards acknowledgement to content provider. A read notification is returned to the BlackBerry Infrastructure.

Browser Push allows you to take advantage of true server-side push communication, which is supported by every BlackBerry smartphone. Browser Push sends information to the browser in a variety of ways (channel, cache, message), regardless of back-end server technology (Apache, IIS, WebSphere) and leverages industry standards such as PAP.

Types of Browser Push include Browser Channel, Browser Message, Browser Cache, and Browser Channel-Delete. The BlackBerry Enterprise Server supports these types of BlackBerry Browser Push applications.

You can use three methods to create applications for BlackBerry smartphones. You can create a web site to be accessed using the BlackBerry Browser or you can develop a server-based application that pushes content such as a web page and images, or data in general, to the the BlackBerry smartphone. You can also write a software application that operates on the BlackBerry smartphone and on a remote server.

The BlackBerry Enterprise Server provides push-based access to email, calendar, contacts, tasks and notes, instant messaging, web-based applications, services, and enterprise applications. BlackBerry Enterprise Server strengthens integration between BlackBerry smartphones and corporate enterprise systems, enabling mobile workers to be more productive.

The Transcoder API provides the classes and interfaces for an encoding scheme for data sent to and from BlackBerry smartphones and the BlackBerry Enterprise Server located in your organization. You can use the encoding scheme to encode and decode all Gateway Messaging Envelope (GME) packets that the BlackBerry Enterprise Server sends to the BlackBerry smartphone and all GME packets that the BlackBerry smartphone sends to the BlackBerry Enterprise Server.

An application that pushes data to the browser cache can include an expiry time that defines the length of time that the data remains in the cache before it is removed. The default length of time that push content is stored in cache memory varies with the BlackBerry Browser software version used.
1. In your own words, define Browser Push.

2. Provide a brief description of each of the following types of Browser Push:
   A. Browser Channel
   B. Browser Message
   C. Browser Cache
   D. Browser Channel-Delete

3. List two items to consider when writing Browser Push Applications.

4. Briefly describe how mobile applications have benefited from BlackBerry Enterprise Server Push messaging.

5. Describe how data is transcoded on the BlackBerry smartphone.

6. How do you delete a channel from a BlackBerry smartphone?