© 2009 Research In Motion Limited. All rights reserved. BlackBerry®, RIM®, Research In Motion®, SureType®, SurePress™ and related trademarks, names and logos are the property of Research In Motion Limited and are registered and/or used in the U.S. and countries around the world.

Adobe, Acrobat, and Illustrator are trademarks of Adobe Systems Incorporated; Bluetooth is a trademark of Bluetooth SIG; Google and Google Talk are trademarks of Google Inc.; GSM and Global System for Mobile communications are trademarks of the GSM MOU Association; IBM, Domino, Lotus, Lotus Notes, Lotus Organizer and Sametime are trademarks of International Business Machines Corporation; IEEE is a trademark of the Institute of Electrical and Electronics Engineers, Inc.; Java, J2ME, JavaScript, JDBC, and JRE are trademarks of Sun Microsystems, Inc.; Microsoft, Active Directory, Active X, Entourage, Excel, Internet Explorer, Outlook, Power Point, SQL Server, Visio, Visual Studio, Windows, Windows Media, Windows Mobile and Windows Server are trademarks of Microsoft Corporation; Novell and GroupWise are trademarks of Novell, Inc.; PGP is a trademark of PGP Corporation; W3C is a trademark of the Massachusetts Institute of Technology; Wi-Fi is a trademark of the Wi-Fi Alliance; All other trademarks are the property of their respective owners.

The BlackBerry smartphone and other devices and/or associated software are protected by copyright, international treaties, and various patents, including one or more of the following U.S. patents: 6,278,442; 6,271,605; 6,219,694; 6,075,476; 6,073,318; D445,428; D433,460; D416,256. Other patents are registered or pending in the U.S. and in various countries around the world. Visit www.rim.com/patents for a list of RIM (as hereinafter defined) patents.

This documentation including all documentation incorporated by reference herein such as those provided or made available by hyperlinks is provided or made accessible "AS IS" and "AS AVAILABLE", and without condition, endorsement, guarantee, representation or warranty of any kind by Research In Motion Limited and its affiliated companies ("RIM") and RIM assumes no responsibility for any typographical, technical, or other inaccuracies, errors or omissions in this documentation. In order to protect RIM proprietary and confidential information and/or trade secrets, this documentation may describe some aspects of RIM technology in generalized terms. RIM reserves the right to periodically change information that is contained in this documentation; however, RIM makes no commitment to provide any such changes, updates, enhancements, or other additions to this documentation to you in a timely manner or at all.

This documentation might contain references to third-party sources of information, hardware or software, products or services including components and content such as content protected by copyright and/or third-party web sites (collectively the "Third Party Products and Services"). RIM does not control, and is not responsible for, any Third Party Products and Services including, without limitation the content, accuracy, copyright compliance, compatibility, performance, trustworthiness, legality, decency, links, or any other aspect of Third Party Products and Services. The inclusion of a reference to Third Party Products and Services in this documentation does not imply endorsement by RIM of the Third Party Products and Services or the third party in any way.

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 CONDITIONS, ENDORSEMENTS, GUARANTEES, REPRESENTATIONS OR WARRANTIES OF DURABILITY, FITNESS FOR A PARTICULAR PURPOSE OR USE, MERCHANTABILITY, MERCHANTABILITY QUALITY, NON-INFRINGEMENT, SATISFACTORY QUALITY, OR TITLE, OR ARISING FROM A STATUTE 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 JURISDICTIONS 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 OR THE ITEM THAT IS THE SUBJECT OF THE CLAIM.

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, HARDWARE, SERVICE, OR ANY THIRD PARTY PRODUCTS AND SERVICES REFERENCED HEREIN INCLUDING WITHOUT LIMITATION ANY OF THE FOLLOWING DAMAGES: DIRECT, CONSEQUENTIAL, EXEMPLARY, INCIDENTAL, INDIRECT, SPECIAL, PUNITIVE, OR AGGRAVATED DAMAGES, DAMAGES FOR LOSS OF PROFITS OR REVENUES, FAILURE TO REALIZE ANY EXPECTED SAVINGS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, LOSS OF BUSINESS OPPORTUNITY, OR CORRUPTION OR LOSS OF DATA, FAILURES TO TRANSMIT OR RECEIVE ANY DATA, PROBLEMS ASSOCIATED WITH ANY APPLICATIONS USED IN CONJUNCTION WITH RIM PRODUCTS OR SERVICES, DOWNTIME COSTS, LOSS OF THE USE OF RIM PRODUCTS OR SERVICES OR ANY PORTION THEREOF OR OF ANY AIRTIME SERVICES, COST OF SUBSTITUTE GOODS, COSTS OF COVER, FACILITIES OR SERVICES, COST OF CAPITAL, OR OTHER SIMILAR PECUNIARY LOSSES, WHETHER OR NOT SUCH DAMAGES WERE FORESEEN OR UNFORESEEN, AND EVEN IF RIM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW IN YOUR JURISDICTION, RIM SHALL HAVE NO OTHER OBLIGATION, DUTY, OR LIABILITY WHATSOEVER IN CONTRACT, TORT, OR OTHERWISE TO YOU INCLUDING ANY LIABILITY FOR NEGLIGENCE OR STRICT LIABILITY.

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, DISTRIBUTOR, SUPPLIER, INDEPENDENT CONTRACTOR OF RIM OR ANY AFFILIATES OF RIM HAVE ANY LIABILITY ARISING FROM OR RELATED TO THE DOCUMENTATION.
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 6
BlackBerry Widgets

Objectives

- Describe widgets for mobile application development
- Describe the security model for widgets and for a mobile device such as a BlackBerry® smartphone
- Describe the benefits of a widget over regular web application development
- Describe the additional features and JavaScript® APIs available to widgets
- Explain how to create a widget for a mobile device application

This chapter outlines BlackBerry widgets. It discusses how widgets work on the BlackBerry smartphone platform, the security model used, and the benefits of developing a widget compared to developing web applications. This chapter also discusses the JavaScript APIs that are available to widgets, and explains how to create a widget for a BlackBerry smartphone.
About BlackBerry widgets

BlackBerry widgets are standalone BlackBerry applications that consist of standard web components, including HTML, XHTML, style sheets, JavaScript, SVG, image files, and other resources. BlackBerry widgets are based on the W3C® widget specification.

In addition to being created entirely with standard web technology, BlackBerry widgets provide the developer with the ability to do the following:

- leverage BlackBerry APIs in a secure and manageable container
- use the same distribution/management model as native applications and can be distributed through BlackBerry App World™ storefront
- author the user interface entirely in HTML/CSS
- write application logic in JavaScript
- access the SD Card, PIM, and native applications
- push and pull data

For example, application developers can create a BlackBerry widget that has sophisticated UIs for data entry and searching, and that supports multithreading, internationalization, network communication, and local data storage. A BlackBerry widget can communicate with networks using standard TCP and HTTP connections, regardless of the underlying wireless network.

Application developers can also create a BlackBerry widget that integrates tightly with core BlackBerry smartphone applications, such as the message list, organizer applications, phone, and browser, for an essentially seamless user experience.

After a developer creates the HTML, style sheet, and JavaScript code for a web application, those resources are packaged as a ZIP archive and are converted into a BlackBerry widget through the BlackBerry Widget Packager.

Figure 6-1 shows how a BlackBerry widget incorporates standard web components, and how it uses JavaScript extensions and BlackBerry APIs.
Tools and resources for widget development

In addition to standard web development resources, you need the BlackBerry® Widget SDK to develop a BlackBerry widget:

The BlackBerry Widget SDK enables you to create native BlackBerry applications using standard web resources such as HTML, style sheets, and JavaScript. The BlackBerry Widget SDK includes the BlackBerry Widget Packager, code samples, BlackBerry Smartphone Simulator, BlackBerry MDS Connection Service simulator, and documentation.

Both the BlackBerry® Web Plug-in for Eclipse and the BlackBerry® Web Plug-in for Microsoft® Visual Studio® include the BlackBerry Widget SDK as part of their installation process. You can use Eclipse or Microsoft Visual Studio to author a widget, or you can download a standalone SDK and create a widget using other tools. The BlackBerry widget development process is tool independent.

You can extend your widgets using BlackBerry widget APIs. The BlackBerry widget APIs are JavaScript APIs that expose BlackBerry smartphone capabilities that you can use to enhance the capabilities and
usefulness of your BlackBerry widget. Some BlackBerry widget APIs also enable you to automatically push application data to the BlackBerry widget running on a BlackBerry smartphone.

You can create BlackBerry widgets using familiar tools and then package them using the BlackBerry Widget Packager. The BlackBerry Widget Packager is a command line tool that compiles your web code to create the BlackBerry widget that you can distribute to BlackBerry smartphones as you would with any other native BlackBerry application. The BlackBerry Widget Packager creates the output files you need, including **COD**, **JAD**, and **ALX**, for your distribution strategy.

This illustration shows the model for creating the output files for a widget.

![Figure 8.7 The widget packager](image)

**Supported standards and content types**

You create the content and appearance for your BlackBerry widget using the BlackBerry® Browser. BlackBerry web development and the standards and content types it supports are continually changing, so as you design your content, you need to know what standards and content types are supported by a BlackBerry widget so that you can deliver device-appropriate content. For information about the standards and content supported by the BlackBerry Browser, visit the BlackBerry® Developer Zone website to see the following guides:

- BlackBerry Browser HTML Reference
- BlackBerry Browser CSS Reference
- BlackBerry Browser JavaScript Reference

**Communicating with the server: push and pull data**

BlackBerry widgets support push and pull communication.

With pull communication, you can use XMLHttpRequest objects as you would with AJAX development. This enables you to use either synchronous or asynchronous pull communication.

BlackBerry widgets support push data, that is, the ability for the backend server to push application data to the BlackBerry widget running on a BlackBerry smartphone. They use the BlackBerry widget API methods **blackberry.push** and **blackberry.push.Data** to enable BlackBerry widgets to use the same BlackBerry® Enterprise Server and public data push that other native BlackBerry applications use.
Example of a BlackBerry widget

Widgets for mobile devices are generally single-purpose applications that allow users to view reformatted and specialized web content. With widgets, users do not need to type a web address or configure settings: the widget delivers the services and content available on the Internet to the mobile device.

One example of a widget for a BlackBerry smartphone is BlackBerry® Push Weather Clients. This widget allows BlackBerry users to get weather information every day; users can choose to receive either AccuWeather or The Weather Network. The software lets users choose how to receive weather updates: they can receive a text message, or have a browser channel appear on their homepage.
1. What is a widget?

2. Name the specification that BlackBerry widgets are based on.

3. Which data components of a BlackBerry smartphone can a widget access?

4. Which of the following describes the BlackBerry Widget Packager? Choose two.
   A. A command line tool.
   B. A graphical user interface.
   C. A compiler.

5. List the types of output files created by the BlackBerry Widget Packager.

6. List the ways in which a widget can communicate with a server.

7. List the APIs that you can use to move application data between a server and a widget running on a BlackBerry smartphone.
Answers

1. Standalone applications, generally with a single-purpose, that consist of standard web components.

2. W3C widget specification.

3. SD card, PIM, and native BlackBerry applications.

4. A and C.

5. Output files are: COD, JAD, and ALX.

6. Push data; pull data.

7. The APIs are: `blackberry.push` and `blackberry.push.Data`. 
The security model

There are two main approaches to preventing malicious programs such as viruses, trojans, worms and spyware (collectively referred to as malware): detection and containment.

Detection is the process of determining whether a program is malicious (malware). Effectively detecting malware is very difficult. It requires a large, frequently updated, local database or a constant connection to an online database. While desktop computers can satisfy these requirements, mobile devices cannot. Mobile devices do not have enough storage space to hold a malware database and a constant connection to the Internet can not be guaranteed.

Containment is the process of preventing a malicious program from causing damage after it has appeared. Containment is relatively easy. It simply requires controlling access to the device software and other applications on the device. The BlackBerry solution focuses on containing malicious programs. The BlackBerry software and core applications are digitally signed to ensure authenticity and to control access to the APIs. Thus, the core BlackBerry functionality cannot be directly accessed by other applications.

BlackBerry widgets follow the same security rules, configuration, and deployment model as any other native BlackBerry application. BlackBerry widgets must be signed with a RIM® code signing key, and the same access control policies apply to widgets that apply to native applications.

Access to external content

By default, BlackBerry widgets cannot access data from external resources. For example, a widget cannot retrieve an HTML page or make an AJAX request to a web service, unless you configure the widget to be able to do so.

To enable access to external resources and APIs, you must define the resources and APIs you require in the widget configuration document. Essentially, you define the list of domains allowed access and the BlackBerry widget APIs allowed for each domain. This list is called a whitelist. Web resources outside of the widget can be pulled in from external sources as long as those sources match the whitelist provided with your widget.

Whenever you retrieve content from outside your widget, do the following to make the BlackBerry widget as secure as possible:

- Provide JavaScript access to sensitive APIs for trusted and secure web sites only.
- Protect your communication channel using HTTPS when you are exposing sensitive APIs to the domain.
- Use the same precautions you normally use for a hosted web site, to guard against cross-site scripting attacks.
Authentication

You can use device authentication, application authentication, and server-side authentication to ensure the security of your widget.

BlackBerry smartphone authentication and IT policy

BlackBerry smartphone users can set a password for their BlackBerry smartphone. When the device password is active, the BlackBerry smartphone users must provide the password to access the data and applications. Using device passwords is a good first step to limiting access to your BlackBerry widget on the BlackBerry smartphone.

Administrators can use the IT policies that are provided in the BlackBerry Enterprise Server to make sure that BlackBerry smartphone in the organization are password-protected. Administrators can also use IT policies to remotely lock a BlackBerry smartphone, change the password, or remove all of the data.

Application authentication

For a BlackBerry widget where security features are critical, you can provide a login screen that requires the BlackBerry smartphone user to log in to the application on the BlackBerry smartphone before using it. The UI classes provide simple password fields that hide the text entry with asterisk characters. Login screens can negatively impact the BlackBerry smartphone user experience, and if the BlackBerry smartphone user sets a password to protect the BlackBerry smartphone, your application may not require a login screen.

Server-side authentication

If your BlackBerry widget connects to an application on a server or to the Internet or an intranet, you can include additional authentication features when the BlackBerry smartphone users log in to the server. Most applications that require user authentication rely on HTTP Basic authentication, which uses a simple user name and password combination. You can use HTTP Basic authentication by adding the correct HTTP headers while opening the HTTP connection. You can also add more advanced authentication using certificates, although most applications do not require it.

Application control

The BlackBerry Enterprise Server application control policy rules are designed to allow or prevent the installation of specific applications on the BlackBerry smartphone and to limit the permissions of applications on the BlackBerry smartphone. For example, administrators can use the application control policy to make sure that a game application on the BlackBerry smartphone cannot access the phone application.
The administrator can apply application control policies only when the BlackBerry smartphone is associated with a BlackBerry Enterprise Server.

Data encryption

When you develop a BlackBerry widget, it is important to understand how data is encrypted in the network, on the BlackBerry smartphone, and on the microSD memory card.

Data encryption in transport

If you use the BlackBerry Enterprise Server as the network gateway for your BlackBerry widget, the BlackBerry Enterprise Server encrypts data using AES or TripleDES encryption at all points in the connection between the BlackBerry smartphone and the BlackBerry Enterprise Server behind the organization's firewall. If you require data to be encrypted further between the BlackBerry Enterprise Server and the destination server, you can use the HTTPS protocol and use SSL/TLS encryption.

If your BlackBerry widget uses the BlackBerry Internet Service or the Internet gateway of the wireless service provider, data traffic is not encrypted.

Data encryption on the BlackBerry smartphone

Administrators can set an IT policy to make sure that all BlackBerry smartphone user data stored in BlackBerry smartphone applications is encrypted locally in flash memory.

Encryption of data on a microSD media card

When a BlackBerry widget accesses a file on the microSD memory card, file decryption occurs and the file moves to main memory for an application to read. For a BlackBerry widget to access a file that is password protected, the BlackBerry smartphone must not be locked. Encrypted files have a REM extension and cannot be decrypted on platforms other than BlackBerry.

If the NVRAM is removed and the microSD media card is locked with a BlackBerry smartphone key, the data on the microSD media card is no longer accessible. To remove data that is not accessible, start the BlackBerry smartphone and remove all encrypted media files.

The BlackBerry smartphone uses a master key stored on the microSD media card to encrypt BlackBerry smartphone media files. The master key prevents the BlackBerry smartphone from having to decrypt or re-encrypt all media files when you disable encryption or change the password.
Using the microSD media card with more than one BlackBerry smartphone

If the BlackBerry smartphone user moves the microSD media card to a BlackBerry smartphone that does not use a BlackBerry smartphone password or uses a password that does not successfully decrypt the microSD media card master key, the BlackBerry smartphone prompts the BlackBerry smartphone user to enter the microSD media card password. If the BlackBerry smartphone has a password, the BlackBerry smartphone user can use the prompt to change the microSD media card password to the BlackBerry smartphone password.

IT policies and the microSD media card

You can apply the IT policy to encrypt data written to the microSD media card to any new or modified files that you store on the microSD media card. Only the files that you store on the microSD media card after an administrator sets the IT policy are encrypted.

Except for media files, all content is encrypted.

Access to memory

The BlackBerry widget is designed to avoid causing problems accidentally or maliciously in another application or BlackBerry widget on the BlackBerry smartphone. A BlackBerry widget can write only to the BlackBerry smartphone memory that the BlackBerry® Java® Virtual Machine uses. The BlackBerry Widget cannot access the virtual memory or the persistent storage of other applications (unless granted access to do so). A BlackBerry widget can access persistent storage or user data, or communicate with other applications, only through specific BlackBerry widget APIs.
Quiz

1. Name two approaches for preventing malicious programs from causing damage to a device.

2. Which one of these approaches is the focus of the BlackBerry security model? Why?

3. List two ways that digital signatures ensure security.

4. How can you allow BlackBerry widgets to access external resources and APIs?
Answers

1. Detection; containment.

2. Containment.

3. Ensure authenticity; control access to APIs.

4. By defining a whitelist.
Widgets and web applications: what is the difference?

On mobile devices, where screen space is limited, it can be difficult for users to run and view multiple applications, and to switch between applications. You can address these limitations through the creation of widgets, which reformat and deliver specific web content.

Widget development for BlackBerry smartphones uses the same standard technologies as web application development: HTML, CSS, JavaScript, and AJAX. Widgets, however, can provide a more personalized experience, and provide greater ease-of-use than web applications do.

Flexible development using standard technologies

Widgets are a method of deploying an application to a mobile device without learning the languages that are supported by the mobile device. For example, you can create widgets for the BlackBerry smartphone without first learning Java® for BlackBerry. This approach provides you with the ability to use highly familiar technologies to build the same types of applications normally created in Java. You can use Java to extend your BlackBerry widgets as well; the BlackBerry widget development model means that you can choose the technology that best suits your needs.

In addition to leveraging standard web technologies, you can also leverage both client-side and server-side resources.

Server-side resources

When you develop a widget, you can leverage the server-side assets that you have already created and extend them to the BlackBerry smartphone. You can write a BlackBerry widget that pulls in all your resources from your web server. In the BlackBerry widget, you can specify your own custom header that is passed down on every web request so that you know the request for the content is coming from your widget. You can then have your server-based web content react to this header and add functionality to the markup specific to the BlackBerry smartphone.

For example, instead of writing a search in your client application that would return XML, and then provide the user with a results interface that is based on the XML, you can instead pass the search to the server, and have the server return the rendered HTML back to the widget user interface with specific JavaScript markup. The markup can provide interaction with local resources, such as PIM.

Client-side resources

You can also leverage client-side resources. For example, your application can reside locally on the BlackBerry smartphone, but link to your CSS from remote sites. This approach means that you can still change your UI dynamically from a server, while maintaining the application logic on the device.
Enhanced user experience

Widgets for mobile devices provide an enhanced experience for the end user. These single-purpose applications provide Internet content and services in a format suitable for mobile devices, where screen size is an important consideration. In addition, widgets can deliver web content that is specific to a user, and personalized to meet his or her requirements. For example, the location of a mobile device is important information that you can use to enhance the experience of the user. You can use this information to create widgets that offer location-based services, such as local maps, local offers, currency converters, and weather information. Users can combine a variety of widgets to meet their needs.
1. Name two ways in which widgets are different than web applications?

2. List three reasons why widgets are an efficient way for you to create web applications for mobile devices?

3. What is the purpose of specifying a custom header in your widget? Provide two examples.

4. Name a client-side resource on the BlackBerry smartphone that your widget can access.
Answers

1. Reformat content for mobile devices; deliver specific or personalized content.

2. Use standard web components for authoring; use existing client-side resources; use existing server-side resources.

3. Verify that content requests received by your web server have originated from your widget; add functionality.

4. PIM.
JavaScript APIs and widgets

You can extend the capabilities of BlackBerry widgets using BlackBerry widget APIs, which add specific functionality to your BlackBerry widget.

The BlackBerry widget APIs are JavaScript APIs that expose BlackBerry smartphone capabilities that you can use to enhance the capabilities and usefulness of your BlackBerry widget. Some BlackBerry widget APIs also enable you to automatically push application data to the BlackBerry widget running on a BlackBerry smartphone.

You can use the BlackBerry APIs to create a custom application with specific features for a BlackBerry smartphone. These features include GPS and location-based services, multimedia, calendar and email message access, a customizable UI, and interaction with a trackwheel, trackball, or touch screen. You can find information about the available APIs on the BlackBerry Developer Zone website.

The illustration shows how JavaScript extensions are included in the BlackBerry widget.

![JavaScript extensions extend the functionality of widgets](image)

**Figure 8.8** JavaScript extensions extend the functionality of widgets

**Note:**
JavaScript extensions are not accessible in the BlackBerry Browser; they are available only in widgets.
Code sample: using a JavaScript extension

The following example shows the JavaScript code for creating an appointment.

```javascript
// Create our Appointment
var newAppt = new blackberry.pim.Appointment();
newAppt.location = "Your office";
newAppt.summary = "Talk about new project";
newAppt.freeBusy = blackberry.pim.Appointment.FREE;
// Create our hour time slot
var start = new Date();
newAppt.start = start;
var end = start.setHours(start.getHours() + 1);
newAppt.end = end;
// Create Attendee
var attendees = new Array();
var onlyAttendee = new blackberry.pim.Attendee();
onlyAttendee.address = "john@foo.com";
onlyAttendee.type = blackberry.pim.Attendee.INVITED;
attendees.push(onlyAttendee);
// Save Appointment
newAppt.attendees = attendees;
newAppt.save();
```


1. Are JavaScript extensions available in the BlackBerry Browser?

2. List three benefits of using APIs to extend a widget.
Answers

1. Blackberry GPS
   - Blackberry Maps
   - Blackberry Address Book
   - Blackberry Calendar
   - Blackberry Email

2. No.

3. Push application data to the widget.
   - Add capabilities or functions to the widget.
   - Integrate with specific features or functions of the BlackBerry smartphone.
Creating a BlackBerry widget

The BlackBerry widget development process is tool independent. You can create BlackBerry widgets using the tool of your choice and then package them using the BlackBerry Widget Packager. The BlackBerry Widget Packager creates the output files you need, including COD, JAD, and ALX, for your distribution strategy.

To create a BlackBerry widget, develop the HTML, style sheet, and JavaScript code for your web application. After you have created these components, follow the process outlined in Figure 6-4. Detailed information about each of the steps in the process is included in the following sections.
Figure 8.9 Process for creating a BlackBerry widget
Installing the BlackBerry Widget Packager

Before you begin, download the BlackBerry Widget Packager from www.blackberry.com/developers.

1. Double-click the installer file.
2. On the Welcome page, click Next.
3. On the License Agreement page, click I accept the terms in the license agreement.
4. Click Next.
5. Enter customer information and click Next.
6. To change the installation folder, click Change and navigate to a new location.
7. Click Next.
8. Click Install.
9. When the install completes, click Finish.

Creating a BlackBerry widget configuration document

The BlackBerry widget configuration document is an XML file that contains information required for various parts of the widget, including elements to define the namespace, the name of the widget, whitelist information for security, and elements for setting the start page and icons used. The widget configuration document uses the widget element at its root. The widget element serves as a container for all other elements.

The configuration document must be in the root directory of the BlackBerry widget archive.

Code sample: BlackBerry widget configuration document

The valid BlackBerry widget configuration document file name is config.xml, and is case insensitive. Within the widget archive, the configuration document must use this file name. The config.xml file must exist in the root directory of the widget archive.

```xml
<?xml version="1.0" encoding="utf-8"?>
<widget xmlns="http://www.w3.org/ns/widgets"

    xmlns:rim="http://www.blackberry.com/ns/widgets"
    version="2.0"
    rim:header="RIM-Widget: rim/widget">
```
<name> The example widget</name>
<description>

A sample widget to demonstrate some of the possibilities.
</description>
<icon src="icons/example.png"/>
<icon src="icons/boo.png" rim:hover="true"/>
<content src="index.html"/>
<feature id="blackberry.ui.dialog" required="true" version="1.0.0"/>
<access uri="http://www.somedomain.com" subdomains="true">
  <feature id="blackberry.pim.memo" required="true" version="1.0.0"/>
  <feature id="blackberry.invoke.MemoArguments" required="true" version="1.0.0"/>
</access>
<rim:connection timeout="25">
  <id>TCP_WIFI</id>
  <id>MDS</id>
  <id>BIS-B</id>
  <id>TCP_CELLULAR</id>
  <id>WAP2</id>
  <id>WAP</id>
</rim:connection>

<license>

Example license (based on MIT License)
Copyright (c) 2008 The Foo Bar Corp.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, INSULT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
THE SOFTWARE.

</license>
</widget>
Assigning the widget namespace to the widget element

You must assign the widget namespace to the widget element. If the namespace is missing, the widget archive is invalid. The namespace for BlackBerry widget extensions is optional.


Widget element

The widget element provides a container for all other elements, and is the root element in the configuration document.

You use the following order for all other elements within the widget element:

- one name element
- zero or one description elements
- zero or more icon elements
- zero or more access elements
- zero or one content elements
- zero or more feature elements
- zero or one rim:loadingScreen elements
- zero or one rim:connection elements

The following table lists the attributes for the widget element.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
</table>
| version    | The version attribute specifies a valid version, in one of the following formats:  
  - a.b  
  - a.b.c  
  - a.b.c.d  
  
  For version numbers that contain fewer than four digits, additional zeroes are added to the number. For example, if you specify 1.0 for the version number, the resulting version number for the widget is 1.0.0.0.  
  
  If you specify an invalid version number, the widget archive is invalid. This attribute is required. |
### Name element

The `blackberry.app.name` element specifies a human-readable name for a BlackBerry widget that you can use, for example, in an application menu.

If a name attribute is not supplied, the widget is considered invalid. You can also specify an icon using `blackberry.app.name` in the API. The following table lists the attributes for the name element.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>rim:header</code></td>
<td>The <code>rim:header</code> attribute specifies a header value that is added to every request for data from the widget. This enables you to distinguish widget requests from BlackBerry Browser requests. This attribute is optional.</td>
</tr>
<tr>
<td><code>id</code></td>
<td>The <code>id</code> attribute specifies a unique identifier for the widget. This attribute is optional.</td>
</tr>
<tr>
<td><code>xml:lang</code></td>
<td>The <code>xml:lang</code> attribute defines the language used in the element, for example, <code>en</code> for English. This attribute is optional.</td>
</tr>
</tbody>
</table>

### Description element

The `blackberry.app.description` element provides a human-readable description of the widget. You can access this element using `blackberry.app.description` in the API. The following table lists the attributes for the description element.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>xml:lang</code></td>
<td>The <code>xml:lang</code> attribute defines the language used in the element. This attribute is optional.</td>
</tr>
<tr>
<td><code>its:dir</code></td>
<td>The <code>its:dir</code> attribute specifies the directionality of the language used in the element, for example, <code>its:dir=&quot;rtl&quot;</code> specifies a right-to-left language. This attribute is used for localization.</td>
</tr>
</tbody>
</table>
The icon element specifies a custom icon for a BlackBerry widget. You can set one or more icon elements. The first icon found that does not have `rim:hover` set to `True` is used as the main image on the BlackBerry smartphone. If you do not specify the icon element, the BlackBerry Widget Packager uses a default icon from the widget archive. If it does not find an icon in the archive, it uses the default BlackBerry application icon. The icon must meet the requirements for custom or default icons. This element is optional.

You can also specify an icon using `blackberry.app.setHomeScreenIcon(src, rim:hover)` in the API. The following table lists the attributes for the icon element:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>src</code></td>
<td>The <code>src</code> attribute specifies the path for an image file within the widget archive. This attribute is required.</td>
</tr>
<tr>
<td><code>rim:hover</code></td>
<td>The <code>rim:hover</code> attribute is a boolean value that specifies the icon to be used as the hover icon on the BlackBerry home screen. The first hover icon in the configuration document is used as the hover icon for the widget. This attribute is optional. If you do not specify the <code>rim:hover</code> attribute, the value is set to <code>False</code> by default.</td>
</tr>
</tbody>
</table>

The access element enables permission to access external network resources. If you do not specify an access element, a widget has access to all local resources by default. Local resources include all resources within the archive file. The following table lists the attributes for the access element:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>URI</code></td>
<td>The <code>URI</code> attribute defines the address for the access request. This attribute is required.</td>
</tr>
</tbody>
</table>
Feature element

The feature element specifies an API that is used by the BlackBerry widget. You can use this element within the access element to enable the feature under the external domain. If you do not specify an access element above the feature element, the local domain is used. If you do not declare an API as a feature, the API cannot be used by the widget for the relevant domain. The following table lists the attributes for the feature element.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The id attribute specifies the name of the BlackBerry API. The value must match the name of the API, but is not case sensitive. You cannot use a wild card (*) to use multiple namespaces at a time. This attribute is required.</td>
</tr>
<tr>
<td>required</td>
<td>This attribute is provided for conformance with the W3C specification, and is optional.</td>
</tr>
<tr>
<td>version</td>
<td>This attribute references the library version of the web APIs, and is optional.</td>
</tr>
</tbody>
</table>

Content element

The content element declares the start file used by the BlackBerry widget when it runs. If you do not specify a content element, a default start file from within the widget archive is used. The following table lists the attributes for the content element.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>src</td>
<td>The src attribute specifies the source HTML file within the widget archive. This attribute is required.</td>
</tr>
<tr>
<td>type</td>
<td>The type attribute specifies the MIME type for the file referenced in the src attribute. This attribute is optional.</td>
</tr>
</tbody>
</table>
RIM loadingScreen element

The RIM loadingScreen element specifies the properties of the loading screen that appears when the BlackBerry widget starts. This element is optional. The following table lists the attributes for the RIM loadingScreen element.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bgColor</td>
<td>The bgColor attribute defines the HEX color value to display as the background before the initial start page appears. This attribute is optional. If you do not specify a color, the background is set to white.</td>
</tr>
</tbody>
</table>

RIM connection element

The RIM connection element specifies connection preferences for the BlackBerry device. The following table lists the attributes for the RIM connection element.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>The timeout attribute overrides the default timeout when the device attempts to connect to a particular transport. If you do not specify this attribute, the device uses the default timeout for native BlackBerry applications. Specify the default value in milliseconds; if you do not specify a valid numeric value, the timeout specified is ignored.</td>
</tr>
</tbody>
</table>

Id element

The id element specifies the type of communication transport you want to use for the BlackBerry widget. This element is optional.

If you do not specify an id element, the default order for the connection manager is as follows:

- TCP_WIFI
- MDS
- BIS-B
- TCP_CELLULAR
• WAP2
• WAP

If you specify more than one transport, the connection manager attempts only the transports you specify, in order.

Creating a BlackBerry widget archive

A BlackBerry widget archive is a standard ZIP file that you compile using the BlackBerry Widget Packager to create a BlackBerry widget.

The widget archive contains the following:
• configuration document
• start page
• application icons
• other resources and objects referenced in your web pages

When you create BlackBerry widgets, you must store them outside of the BlackBerry Widget Packager installation directory.

Specifying the BlackBerry widget start file

The start file for a BlackBerry widget is the file the widget uses when it launches. If you do not explicitly declare a start file using the content element, or if there is an error in the definition, the widget uses a default start file name. The default start file names, in search order, are index.htm and Index.html. The default MIME type is text/html. Default start file names are case insensitive.

Specifying icons for a BlackBerry widget

You can specify icon images for your BlackBerry widget that are used as the application icon on the BlackBerry smartphone. These icons are optional, and you can include them to represent one of two states for the application icon image on the BlackBerry smartphone.

Application icons

On a BlackBerry smartphone, the application icon is the image that appears for your BlackBerry widget in its regular state, or when it is not selected. The default application icon file name is either image.gif or image.png, and must exist in the root folder of the BlackBerry widget archive. When you compile your widget, the BlackBerry Widget Packager searches first for icon.png, and then for icon.gif. The file name is not case sensitive.
You can also set the application icon using the icon element in the configuration document for your BlackBerry widget, for example, `<icon src="icons/appicon.gif"/>`.

If you set the application icon in the configuration document, the BlackBerry Widget Packager does not search the root directory for default icon files when you compile your widget.

**Hover icons**

On a BlackBerry smartphone, the hover icon is the image that appears for your BlackBerry widget when it is in focus, or selected.

You can set the hover icon using the icon element in the configuration document for your BlackBerry widget, for example, `<icon src="icons/appfocus.gif" rim:hover="true"/>`.

If you do not set a hover icon in the configuration document, the regular application icon appears regardless of its state.

**Identifying reserved files and folders**

The following table contains a list of file and folder names reserved by the BlackBerry Widget Packager. You must consider reserved file names to be case insensitive.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config.xml</td>
<td>This file name is used for the widget configuration document.</td>
</tr>
<tr>
<td>signature[0..9].xml</td>
<td>These file names are used for Signature Tool configuration.</td>
</tr>
<tr>
<td>icon.gif, icon.png</td>
<td>These file names are reserved for the default icon.</td>
</tr>
<tr>
<td>Thumbnail.gif, Thumbnail.png</td>
<td>These file names are reserved for thumbnail graphics.</td>
</tr>
</tbody>
</table>

The **bin** and **src** directories are not reserved, but you must not use them for project resources. The **bin** directory and its contents are overwritten at compile time with compiled widget output. The **src** directory is overwritten if you specify the option to save source at compile time.

**Securing a BlackBerry widget**

You can secure device data using a whitelist.

All requests to external domains from your widget must match web addresses that you provide in a corresponding whitelist. A whitelist is a security mechanism that you use to protect the BlackBerry widget and the consumer of the widget from malicious intent.
**Example 1: code that allows access to any site**

If your BlackBerry widget accesses data from an unknown or changing domain, you can use the access element with a wildcard to ensure your requests are not blocked.

In this sample, all requests are allowed, as long as they do not require access to BlackBerry widget features.

*Figure 8.10 Sample code for access to all sites*

```
<access uri="*"/>
```

**Example 2: code that allows access to specific sites**

If your BlackBerry widget accesses functionality on a domain, you must specify the domain using the access element to ensure your requests are not blocked. For example, if you update or change menu items from a domain, you need to specify the domain and the API functionality you are using.

In this sample, features specified under the domain mydomain are allowed. The ellipses in the sample represent specific APIs used by your BlackBerry widget.

*Figure 8.11 Sample code for access to specific sites*

```
<access uri="mydomain" subdomains="true">
  <feature id=" . . . />
  <feature id=" . . . />
</access>
```

**Compiling a BlackBerry widget**

When you compile a BlackBerry widget, all of the widget resources in the archive, including a start page, a configuration document, icons, and other resources, are used to create one or more COD files. You can then install the resulting COD files on the BlackBerry Smartphone Simulator or on a BlackBerry smartphone, just as you do for a regular BlackBerry application.

When you compile a BlackBerry widget, the following actions are performed:

- Validate the contents of the BlackBerry widget archive.
• Create the output target directory and clean up old files if necessary.
• Create the source target directory, if specified, and clean up old files if necessary.
• Create the following output:
  • COD
  • ALX
  • JAD
  • CSO
  • CSL
• Update the COD file for a signed widget with a signed version.

Compile a BlackBerry widget

Before you compile a BlackBerry widget, you must create the widget archive with all required elements and objects. Ensure that the \texttt{wcpc.properties} file exists and that file paths are correct.

You must run the BlackBerry Widget Packager from its installation directory.

1. At a command prompt, change to the installation directory for the BlackBerry Widget Packager.
2. Use the following syntax to compile the widget:

   \texttt{wcpc [drive:] [path] archive [/s [dir]] [/[-]g [password]] [/o dir]}

The compile process displays multiple messages, indicating a successful compile if the widget archive contains no errors. The following table lists the compilation parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>archive</td>
<td>The archive parameter specifies the name of the widget archive to compile. The widget archive is a ZIP file.</td>
</tr>
<tr>
<td>/g</td>
<td>This parameter signs the resulting .cod file using a password. The default setting is /-g for an unsigned widget. The widget is only signed if you specify /g and a password.</td>
</tr>
<tr>
<td>/o</td>
<td>This parameter saves the output files to the specified directory. If you do not use the /o parameter with a directory, the output is saved to the same path as the widget archive.</td>
</tr>
<tr>
<td>/s</td>
<td>This parameter saves the source files. By default, if you specify /s without a directory, source files are saved to the same path as the widget archive. If you specify a directory, source files are saved in the src directory within the directory you specify.</td>
</tr>
</tbody>
</table>
Code sample: compiling a basic BlackBerry widget archive

In the code sample, the BlackBerry Widget Packager uses the BlackBerry widget archive test.zip to produce an unsigned version of the compiled application.

wcpc test.zip

Code sample: compiling a signed BlackBerry widget

In the code sample, the BlackBerry Widget Packager uses the BlackBerry widget archive test.zip to produce the following:

- signed version of the compiled application, in the file \compiled\test.cod
- templates used to generate the .cod file, in \archive_source\src\net\n- extracted files from test.zip, in \archive_source\src\n
wcpc test.zip /g /s archive_source /o compiled

Best practices

The best practices that you must implement when you compile a BlackBerry widget are as follows:

- Name resources consistently and correctly.
  - When you compile a BlackBerry widget, the BlackBerry Widget Packager uses various resources in the process. The BlackBerry Widget Packager does not verify file names, so ensure that all file names and file paths match the settings in the widget configuration document.
  - The BlackBerry Widget Packager also checks references to whitelist entries, the startup page, and application icons.
  - Use alphanumeric file names only and ensure that file names and references to them are case sensitive.

- Verify all required elements in the widget configuration document.
  - Ensure that at a minimum, the namespace and the widget, version, and name elements are defined in the widget configuration document. You may also require the author element if your application is deployed using another service or application. Check for proper syntax in the configuration document and that it is a valid XML file.

- Ensure that elements required by your widget are defined in the widget configuration.
  - Some elements are not required by the BlackBerry Widget Packager but may be required for your widget to run properly. Ensure that a whitelist is defined properly if the widget must be able to make requests to external domains, and that feature elements are defined for all BlackBerry widget APIs you use.
Signing the BlackBerry widget application COD file

The BlackBerry Widget Packager includes the BlackBerry® Signature Tool, so you can install a registration key and sign your BlackBerry widget application COD file.

Applications that use controlled BlackBerry widget APIs require signatures from the BlackBerry Signing Authority tool. You must use the BlackBerry Signature Tool to request and receive signatures, so you can use the code.

If you have already installed a registration key before you installed the BlackBerry Widget Packager, you must create an unsigned widget application COD file, and then sign the .cod file using the method you used to request the registration key.

Install the new registry key with the BlackBerry Signature Tool

1. In the BlackBerry Developer Zone at https://www.blackberry.com/SignedKeys, complete the registration form.
   After you complete the form, RIM sends you an email message containing a CSI file. The CSI file contains a list of signatures, along with your registration information.

2. At the command prompt, navigate to the bin folder within the BlackBerry Widget Packager installation directory.

3. Type the following command, including the full path of the .csi file: java -jar Signature-Tool.jar <.csi file path>

4. If a dialog box indicates that a private key cannot be found, perform the following actions:
   A. Click Yes.
   B. Type a password for the private key.
   C. Type the password to confirm it.
   D. Click OK.

5. Move the mouse to generate data for the new private key.

6. In the Registration PIN field, type the PIN that RIM provided.

7. In the Private Key Password field, type a password of at least eight characters. This password is the private key password.

8. Click Register.

9. Click Exit.
Signing the BlackBerry widget application COD file

Sign your BlackBerry widget when you compile it, using the password you defined when you requested and installed the registry key. Use the BlackBerry Widget Packager, specifying the /g option with a password: WCPC [drive:] [path] archive [/s [dir]] [/g [password]] [/o dir]

Testing BlackBerry widgets in the BlackBerry Smartphone Simulator

Before you distribute your BlackBerry widget to users, you can test it using the BlackBerry® Smartphone Simulator. The BlackBerry Smartphone Simulator enables you to run and view your widgets exactly as you would on a real BlackBerry smartphone.

You can download additional BlackBerry Smartphone Simulators at http://na.blackberry.com/eng/developers/resources/simulators.jsp.

Run a BlackBerry widget on the BlackBerry Smartphone Simulator

Before you begin, ensure that you have successfully compiled a BlackBerry widget.

1. On the Start menu, click Programs > Research In Motion > BlackBerry Widget Packager > BlackBerry Smartphone Simulator.
2. On the File menu, click Load Java Program.
3. Navigate to the output directory for your BlackBerry widget and select the widget COD file that you want to run.
4. Click Open.
5. On the Applications screen, open the Downloads folder.
6. In the Downloads folder, click the BlackBerry widget you created. Run a BlackBerry widget on the BlackBerry Smartphone Simulator.

Make BlackBerry widgets available to users

You can use any of the following distribution methods to distribute your BlackBerry widgets to users:

- **BlackBerry App World storefront**
  - This method installs your BlackBerry widget through the BlackBerry App World storefront on a BlackBerry smartphone. You use the ALX and COD files produced by the BlackBerry widget Packager for this method.
• *BlackBerry Desktop Manager*
  
  This method installs your BlackBerry widget directly to a connected BlackBerry smartphone using the BlackBerry Desktop Manager. You use the ALX files produced by the BlackBerry Widget Packager for this method.

• *Wireless network using the BlackBerry Browser*
  
  This method installs your BlackBerry widget directly to a BlackBerry smartphone over the wireless network. Users visit a web page using the BlackBerry Browser on the device and download the application using a link on the web page. The web server hosts the JAD and COD files produced by the BlackBerry Widget Packager.

• *BlackBerry Application Web Loader*
  
  This method installs your BlackBerry widget to a BlackBerry smartphone from a web page. Users connect the BlackBerry smartphone to their computer and visit the page containing a link to the BlackBerry widget. A Microsoft® ActiveX® control enables users to install the BlackBerry widget directly from the web page. The web server hosts the .jad and .cod files produced by the BlackBerry Widget Packager.

• *Wireless push from the BlackBerry Enterprise Server*
  
  This method installs your BlackBerry widget to BlackBerry smartphones wirelessly, using a push from the BlackBerry Enterprise Server to multiple devices on the network. The server hosts the .cod and .alx files produced by the BlackBerry Widget Packager.
1. What file name must you use for the BlackBerry widget configuration document, and where must you locate this file?

2. List the eight types of elements that can be contained within the widget element.

3. What is the function of the name element?

4. If you do not specify an access element, which of the following occurs? Choose all that apply.
   A. The widget can access all local resources.
   B. The widget can access no local resources.
   C. The widget can access limited local resources.
   D. The widget can access all external resources.
   E. The widget can access no external resources.
   F. The widget can access limited external resources.

5. What is the purpose of the feature element, and where do you use it?

6. What file formats can you use for application icons?

7. List the file names that are reserved by the BlackBerry Widget Packager.

8. What information do you need before you can digitally sign your application COD file?
Answers

1. config.xml
   In the root directory of the widget archive.

2. name element
description element
icon element
access element
content element
feature element
rim:loadingScreen
rim:connection

3. Specifies a human-readable name for the widget.

4. A and E.

5. Specifies an API used by the widget; used within the access element.

6. GIF and PNG.

7. config.xml
   signature[0..9].xml
   icon.gif, icon.png
   thumbnail.gif, thumbnail.png

8. CSI file and registration PIN from RIM.
On mobile devices, where screen space is limited, it can be difficult for users to run and view multiple applications, and to switch between applications. Developers can address these limitations through the creation of widgets, which reformat and deliver specific web content.

BlackBerry widgets are standalone BlackBerry applications that consist of standard web components, including HTML, XHTML, style sheets, JavaScript, SVG, image files, and other resources. You can extend widgets using BlackBerry widget APIs.

BlackBerry widget APIs are JavaScript extensions that enable you to automatically push application data to the BlackBerry widget running on a BlackBerry smartphone. These BlackBerry widget APIs expose BlackBerry smartphone capabilities that you can use to enhance the capabilities and usefulness of your BlackBerry widget.

BlackBerry widgets support both push and pull communication. With pull communication, you can use XMLHttpRequest objects as you would with AJAX development. This enables you to use either synchronous or asynchronous pull communication. With push data, the backend server can push application data to the BlackBerry widget running on a BlackBerry smartphone.

BlackBerry widgets follow the same security rules, configuration, and deployment model as any other native BlackBerry application. BlackBerry widgets must be signed with a RIM code signing key, and the same access control policies apply to widgets that apply to native applications.

By default, BlackBerry widgets cannot access data from external resources. To enable access to external resources and APIs, you must define the resources and APIs you require in the widget configuration document. This list is called a whitelist. Web resources outside of the widget can be pulled in from external sources as long as those sources match the white list provided with your widget.

After you have developed the HTML, style sheet, and JavaScript code for a web application, those resources are packaged as a ZIP archive along with the configuration file and are converted into a BlackBerry widget through the BlackBerry Widget Packager.
1. List and provide a brief description of the technologies you can use to develop a BlackBerry widget.

2. List three ways that widgets can create a positive experience for the BlackBerry user.

3. Describe the methods you can use to make a widget as secure as possible.

4. Describe how a widget can use server-side resources.

5. Describe how a widget can use client-side resources.

6. List and provide a brief description of the distribution strategies available for widgets.

7. List the best practices that you must follow when you compile a widget.

8. List the files and resources that you must include in the widget archive.