BlackBerry WebWorks SDK for Tablets

Version: 2.1

Development Guide

BlackBerry

Published: 2011-09-12
SWD-1429486-0912020046-001

Contents

1 BlackBerry WebWorks applications 4
BlackBerry WebWorks SDK 4
BlackBerry WebWorks Packager 4

2 Creating BlackBerry WebWorks projects 5
Developing, running, and distributing a BlackBerry WebWorks application 5
Creating a BlackBerry WebWorks archive 5
Specifying the start file for a BlackBerry WebWorks application 5
Specifying the icons for a BlackBerry WebWorks application 6
Reserved file names and folder names 6
Creating a BlackBerry WebWorks configuration document 7
Code sample: Creating a BlackBerry WebWorks configuration document 7
BlackBerry WebWorks namespace 8
Widget element 8
Name element 9
Description element 9
Author element 10
Icon element 11
RIM orientation element 11
RIM loadingScreen element 12
Access element 14
Feature element 15
Content element 15
Permissions element 16
Category elements 17

3 Securing BlackBerry WebWorks applications 18
Allowing access to external resources and APIs 18
Allowing requests to any web site 18
Allowing requests to specific web sites 19

4 Compiling BlackBerry WebWorks applications 20
Compiling a BlackBerry WebWorks application 20
Compile a BlackBerry WebWorks application 20
Compilation parameters 20
Code sample: Compiling a BlackBerry WebWorks archive 21
Code sample: Compiling a signed BlackBerry WebWorks application 22
Enable the Web Inspector 22

Best practice: Compiling a BlackBerry WebWorks application 22

5 Signing your application 24
Command line parameters for the blackberry-keytool tool 25
Command line parameters for the blackberry-signer tool 26
Configure signing for tablet applications 27
Sign the .bar file for a BlackBerry WebWorks application 28
Running unsigned applications using a debug token 28
Configure application signing from the command line 28
Package your application from the command line 29
Set the debug token from the command line 30
Package and deploy your application from the command line 31
Retrieve the PIN of a BlackBerry tablet 31
Application Signing Errors 32
Attempt to rename [value1] to [value2] failed 32
Code signing request failed because Application-Development-Mode in Manifest is present and is not set to false 32
Code signing request failed because Common Name in developer certificate is not [value] 32
Code signing request failed because this file has been previously signed 33
Code signing request failed because Package-Author in Manifest is not set to [value] 33
Code signing request failed because value of Package-Name in Manifest is not allowed 33
-cskpass and -csjpin must be specified if -register is specified 34
cskpass required 34
Developer certificate and private key not found in keystore or store password not supplied 34
Incomplete certificate chain 35
Key associated with [value] not a private key 35
Keystore load: store password incorrect 35
keytool error: java.io.FileNotFoundException: certificate.p12 <Access is denied> 35
keytool error: java.lang.Exception: Key pair not generated, alias <author> already exists 36
Missing parameter for [option_name] option 36
No BAR file or CSJ file specified 36
No key name specified 36
No manifest 37
Only one of -setup, -register, or -verify can be specified 37
Server is not responding 37
Signature filename must consist of the following characters: A-Z, 0-9, _ or - 37
Unable to open BAR file 38
Certificate chain not found for: RDK. RDK must reference a valid KeyStore key entry containing a private key and corresponding public key certificate chain 38

keytool error: java.io.IOException: Incorrect AVA Format 38

6 Testing BlackBerry WebWorks applications 40
Run a BlackBerry WebWorks application on the BlackBerry Tablet Simulator 40
Package and deploy your application from the command line 40
Remove your application from the BlackBerry Tablet Simulator 41
Loading an application to the BlackBerry PlayBook tablet 41
Retrieving the IP address of the BlackBerry PlayBook tablet 41
Error Messages: Deploying applications to the BlackBerry Tablet Simulator 42
500 navigator interlock failed 42
Username or password is invalid or not specified or time on the device is out of sync 42
Connection to <IP address> Refused 43
Cannot connect: Connection to https://<IP address> refused. You may have to reboot the target. 43
Device is not in the development mode. Switch to Development from Security settings on the device. 43
Invalid Auth Credentials 44
500 development apps pool exhausted 44
BlackBerry Tablet Simulator doesn't start - black screen at startup 44

7 Using a proxy server from the command line 45

8 Glossary 46

9 Provide feedback 47

10 Legal notice 48

BlackBerry WebWorks applications

1

BlackBerry® WebWorks™ applications are standalone applications that consist of standard web components, including HTML, HTML5, XHTML, style sheets (CSS), JavaScript® code, image files, and other resources. BlackBerry WebWorks applications for the BlackBerry® PlayBook™ tablet follow the same security rules, configuration, and deployment model as other tablet applications.

You can extend the capabilities of applications by using the BlackBerry WebWorks APIs for the BlackBerry Tablet OS, which expose tablet capabilities.

BlackBerry WebWorks SDK

You can use the BlackBerry® WebWorks™ SDK to create BlackBerry WebWorks applications. The BlackBerry WebWorks SDK for BlackBerry® PlayBook™ includes the BlackBerry® WebWorks™ Packager, JRE™ 1.6, and documentation.

The application development process is tool independent. You can use any tool to create BlackBerry WebWorks applications and then use the BlackBerry WebWorks Packager to compile and sign the application for distribution. You can test the application by using the BlackBerry Tablet Simulator.

BlackBerry WebWorks Packager

The BlackBerry® WebWorks™ Packager is a command prompt tool that compiles your web code to create a BlackBerry WebWorks application. The BlackBerry WebWorks Packager creates the .bar file that you need to load your applications on the BlackBerry Tablet Simulator, or a BlackBerry® PlayBook™ tablet.

Creating BlackBerry WebWorks projects

2

Developing, running, and distributing a BlackBerry WebWorks application

When you develop a BlackBerry® WebWorks™ application, you perform the following actions:

1. Develop the HTML, style sheet, and JavaScript® code for your application.
2. Create a configuration document, which is the config.xml file that contains the details about your application. The config.xml file details are explained later in this document.
3. Create the application archive, which is a .zip file that consists of the application code, config.xml file, and resources.
4. Compile and package the application by using the BlackBerry® WebWorks™ Packager.
5. Install and test the application in the BlackBerry® Tablet Simulator.
6. Distribute the application to BlackBerry® PlayBook™ users.

Creating a BlackBerry WebWorks archive

A BlackBerry® WebWorks™ archive is a .zip file you can create with any zip archiving tool, which you then compile using the BlackBerry® WebWorks™ Packager command line tool (bbwp) to create an application.

The name of the .zip file *must not* be longer than 10 characters (not including the .zip extension). If you have previously submitted the application to BlackBerry App World™, you *can not* change the .zip file name for subsequent releases; it must remain the same. If the .zip file name has been changed, an attempt to upgrade the application will not succeed.

The application archive contains the following resources:

- configuration document
- start page
- application icons
- other resources and objects that are referenced in your web pages

You should not place any of your application's source files, or your application's archive file in the BlackBerry WebWorks Packager installation folder. When compiling an application using the bbwp command line tool, the default output folder created is "bin" which will attempt to overwrite the BlackBerry WebWorks SDK's "bin" directory which contains critical files.

Specifying the start file for a BlackBerry WebWorks application

The start file for a BlackBerry® WebWorks™ application specifies the screen that the application displays when it starts. You must specify a start file by using the **<content>** element in the configuration document. You can also specify a URL to use a remote file outside of the application archive.

Specifying the icons for a BlackBerry WebWorks application

You can specify the images to use as the application icons for your BlackBerry® WebWorks™ application. The icons are optional.

Note: for PlayBook tablet development there is a single <icon> only. The size of the icon is 86x86.

In the configuration document, you cannot specify an image that is outside of the BlackBerry WebWorks archive.

Application icons

The application icon is the image that appears on the BlackBerry® device Home screen.

The default file name for the application icon is icon.png, and the file must exist in the root folder of the application archive. When you compile your application, the BlackBerry® WebWorks™ Packager searches for this file if you do not specify an icon explicitly. The file name is not case sensitive.

You can specify the application icon by using the **<icon>** element in the configuration document for your application (for example, `<icon src="icons/appicon.png"/>`). The standard size for the icon for a tablet application is 86x86 pixels.

If you specify the application icon in the configuration document, the BlackBerry WebWorks Packager does not search the root folder for default icon files at compile time.

Reserved file names and folder names

The file names that are reserved by the BlackBerry® WebWorks™ Packager are case insensitive.

Local filename conventions: Supported characters are:

- `(.)` and `(_)` can be used anywhere in the filename.
- `(-)` can be used anywhere in the filename **except** the first charactor.

The bin and src folders in an application archive are not reserved, but you should be aware of the consequences of using them when compiling your application with the command line tool. If you build your application with the command line tool without specifying an output directory, the default output directory is a "bin" directory in the same folder as the zip archive passed into the command line tool. If your application's "bin" directory is at this location, it will be overwritten with the output of the command line tool. This same side effect will hold true if you have a "src" directory at the same location as your zip archive and you use the command line flags to generate the raw source code for your application. The simple way to avoid this issue is to specify an output directory for your application that is not the same as the directory that contains your "bin" and "src" directories.

Reserved name	Description
config.xml	This file name is reserved for the configuration document.
icon.png	This file name is reserved for the default icon image.

Creating a BlackBerry WebWorks configuration document

The BlackBerry® WebWorks™ configuration document is an .xml file that contains the elements to define the BlackBerry WebWorks application namespace, the name of the application, application permissions, the start page, and the icons to use for the application. It also contains the elements to define information such as the author, and email address. The configuration document contains the `widget` element at its root. The `widget` element provides a container for all other elements.

The configuration document must be located in the root folder of the BlackBerry WebWorks application archive.

Code sample: Creating a BlackBerry WebWorks configuration document

The valid file name for a configuration document for a BlackBerry® WebWorks™ application is config.xml, and is case insensitive. Within the application archive, the configuration document must use this file name. The config.xml file must exist in the root folder of the application archive.

```
<?xml version="1.0" encoding="utf-8"?>
<widget xmlns=" http://www.w3.org/ns/widgets"
        xmlns:rim="http://www.blackberry.com/ns/widgets"
        version="2.0.0.0">

  <name> The example application</name>

  <description>
    A sample application to demonstrate some of the possibilities.
  </description>
  <rim:orientation mode="auto"/>
  <rim:loadingScreen backgroundImage="background.png"
                     foregroundImage="foreground.gif"
                     onLocalPageLoad="true">
  <rim:transitionEffect type="zoomIn"/>
  </rim:loadingScreen>

  <rim:category name=”games”/>

  <rim:permissions>
    <rim:permit>access_shared</rim:permit>
    <rim:permit>read_geolocation</rim:permit>
    <rim:permit>use_camera</rim:permit>
  </rim:permissions>

  <icon src="icons/example.png"/>

  <content src="index.html"/>

  <feature id="blackberry.ui.dialog"/>
  <access uri="http://www.somedomain.com" subdomains="true">
    <feature id="blackberry.app.event"/>
    <feature id="blackberry.invoke"/>
  </access>

</widget>
```

BlackBerry WebWorks namespace

You must assign the namespace for the BlackBerry® WebWorks™ application to the `widget` element. If the namespace is missing, the application archive is not valid. The application namespace is `http://www.w3.org/ns/widgets`.

The namespace for BlackBerry WebWorks extensions is *not* optional. The namespace for BlackBerry specific application extensions is `xmlns:rim http://www.blackberry.com/ns/widgets`.

Widget element

The `widget` element provides a container for all other elements, and is the root element in the configuration document for a BlackBerry® WebWorks™ application.

For all other elements within the `widget` element, you use the following requirements:

- One `<name>` element
- Zero or one `<description>` elements
- Zero or more `<icon>` elements
- Zero or one `<author>` elements
- Zero or more `<access>` elements
- One `<content>` element
- Zero or more < `feature>` elements
- Zero or one `<rim:orientation>` elements
- Zero or one `<rim:loadingScreen>` elements

Attribute	Description
`version`	The `version` attribute specifies a valid version for the BlackBerry WebWorks application, in one of the following formats: • x.x.x • x.x.x.x If you specify a version number that is not valid, the application archive is not valid. This attribute is required.
`id`	The `id` attribute specifies a unique identifier for the application. This attribute is optional.
`xml:lang`	The `xml:lang` attribute specifies the language that is used in the element. For more information about this attribute, visit www.w3.org. This attribute is optional.

Name element

The `name` element specifies a human-readable name for a BlackBerry® WebWorks™ application that you can use, for example, in an application menu.

If you do not specify a `name` element, the application is not valid.

You can also specify a name by using the `blackberry.app.name` that is provided in the BlackBerry WebWorks API.

Attribute	Description
`xml:lang`	The `xml:lang` attribute specifies the language that is used in the element. For more information about this attribute, visit www.w3.org. This attribute is optional.
`its:dir`	The `its:dir` attribute specifies the directionality of the language. For example, `its:dir="rtl"` specifies a language that is written from right to left. For more information about this attribute, visit www.w3.org. You can use this attribute for localization.

Code sample: Specifying a name

```
<name>Hello World! application</name>
```

Description element

The `description` element specifies a human-readable description for a BlackBerry® WebWorks™ application.

You can access this element by using the `blackberry.app.description` that is provided in the BlackBerry WebWorks API.

Attribute	Description
`xml:lang`	The `xml:lang` attribute specifies the language that is used in the element. For more information about this attribute, visit www.w3.org. This attribute is optional.
`its:dir`	The `its:dir` attribute specifies the directionality of the language. For example, `its:dir="rtl"` specifies a language that is written from right to left. For more information about this attribute, visit www.w3.org. You can use this attribute for localization.

Code sample: Specifying a description

```
<description>
   This application displays "Hello World" on the screen.
</description>
```

Author element

The `author` element specifies information about the people or organizations that are involved with the creation of a BlackBerry® WebWorks™ application.

You can access this element by using the `blackberry.app.author` that is provided in the BlackBerry WebWorks API.

Attribute	Description
`href`	The `href` attribute specifies a web address that is associated with the author (for example, the web page of the author). You can access this attribute by using the `blackberry.app.authorURL` that is provided in the BlackBerry WebWorks API.
`rim:copyright`	The `rim:copyright` attribute specifies copyright information. You can access this attribute by using the `blackberry.app.copyright` that is provided in the BlackBerry WebWorks API. This attribute is optional.
`email`	The `email` attribute specifies the email address that is associated with the author. You can access this attribute by using the `blackberry.app.authorEmail` that is provided in the BlackBerry WebWorks API. This attribute is optional.
`xml:lang`	The `xml:lang` attribute specifies the language that is used in the element. For more information about this attribute, visit www.w3.org. This attribute is optional.
`its:dir`	The `its:dir` attribute specifies the directionality of the language. For example, `its:dir="rtl"` specifies a language that is written from right to left. For more information about this attribute, visit www.w3.org. You can use this attribute for localization.

Code sample: Specifying author information

```
<author href="http://www.example.com/" rim:copyright="Copyright 1998-2011 My Corp">My
Corp</author>
```

Icon element

The `icon` element specifies a custom icon for a BlackBerry® WebWorks™ application. The icon that you specify is the icon that appears for the application on the Home screen of the BlackBerry® PlayBook™ tablet. The icon must meet the requirements for custom or default icons. This element is optional.

If you do not specify the `icon` element, the BlackBerry® WebWorks™ Packager uses the default icon from the application archive. If the BlackBerry WebWorks Packager does not find the default icon in the archive, it uses the default tablet application icon.

Attribute	Description
`src`	The `src` attribute specifies the path for an image file in the application archive. This attribute is required.

Code sample: Specifying an application icon

```
<icon src="icons/appicon.png"/>
```

RIM orientation element

The `rim:orientation` element specifies the orientation for screens in your BlackBerry® WebWorks™ application. This element is optional.

Attribute	Description
`auto`	The `auto` attribute specifies that screen orientation changes according to the orientation of the BlackBerry® PlayBook™ tablet. By default, if you do not specify the `rim:orientation` element, do not specify it with any attributes, or specify it incorrectly, the orientation is set to auto. This attribute is optional.
`portrait`	The `portrait` attribute specifies that screen orientation remains in portrait mode, regardless of the orientation of the tablet. This attribute is optional.
`landscape`	The `landscape` attribute specifies that screen orientation remains in landscape mode, regardless of the orientation of the tablet. This attribute is optional.

Code sample: Specifying the screen orientation

```
<rim:orientation mode="portrait" />
```

RIM loadingScreen element

The `rim:loadingScreen` element specifies the properties of the screen that appears when the BlackBerry® WebWorks™ application is loading, before the start page appears and when you navigate between pages. You can determine when the loading screen displays by using the `onRemotePageLoad`, `onLocalPageLoad`, and `onFirstLaunch` attributes. This element is optional.

Attribute	Description
`backgroundColor`	The `backgroundColor` attribute specifies the hexadecimal color value to display as the background when the application starts and as the background displayed when you navigate between pages. By default, if you do not specify a color, the background is set to white. This attribute is optional.
`backgroundImage`	The `backgroundImage` attribute specifies the path for a local image to display as the background. You can specify .png or .jpg files. The image is scaled to fit the entire screen. This attribute is optional.
`foregroundImage`	The `foregroundImage` attribute specifies the path for a local image to display as the foreground. You can specify .bmp, .jpg, .gif, and .png files. The image is centered on the screen. This attribute is optional.
`onRemotePageLoad`	The `onRemotePageLoad` attribute specifies whether to display the loading screen when the application transitions to a web page from a remote location, including web pages with URLs that begin with `http` or `https`. The value for this attribute is either `true` or `false`. This attribute is optional.
`onLocalPageLoad`	The `onLocalPageLoad` attribute specifies whether to display the loading screen when the application transitions to a local page, including pages with URLs that begin with `local:///` The value for this attribute is either `true` or `false`. This attribute is optional.
`onFirstLaunch`	The `onFirstLaunch` attribute specifies whether to display the loading screen each time you start the application. The value for this attribute is either `true` or `false`. This attribute is optional.

Code sample: Specifying the background color for the loading screen

```
<rim:loadingScreen backgroundColor="#FFFFFF" />
```

Code sample: Displaying a background image when the application starts

```
<rim:loadingScreen backgroundImage="background.png"
            onFirstLaunch="true" />
```

Code sample: Displaying a background image on transitions from remote pages

You can display a background image when the application starts, and display a background image with a transition effect when the application transitions from a remote page.

```
<rim:loadingScreen backgroundColor="#FFFFFF"
             backgroundImage="background.png"
             onFirstLaunch="true"
             onRemotePageLaunch="true">
       <rim:transitionEffect type="fadeIn" direction="left" />
</rim:loadingScreen>
```

RIM transitionEffect element

The `rim:transitionEffect` element specifies the type and properties of a transition effect that appears when the BlackBerry® WebWorks™ application transitions between screens. The transition effect is not used for the initial loading screen, but you must define it within the `rim:loadingScreen` element. This element is optional.

Attribute	Description
`type`	The `type` attribute specifies the transition effect to use between screens in the application. You can specify the following values for this attribute: • `slideOver` • `fadeIn` • `fadeOut` • `wipeIn` • `wipeOut` • `zoomIn` Note that `slideOver`, `wipeIn`, and `wipeOut` behave the same. `fadeIn` and `fadeOut` also behave the same. This attribute is required.
`duration`	The `duration` attribute specifies the time it takes, in milliseconds, to complete the transition. The value for this attribute must be an integer between 250 and 1000. If you specify a value greater than 1000, the value is set to 1000. If you specify a value less than 250, the value is set to 250.

Attribute	Description
	If you do not specify a value, the value is set to 250. This attribute is optional.
`direction`	The `direction` attribute specifies the direction of the screen effect. Values for this attribute include `left`, `right`, `up`, and `down`. This attribute is valid for the following effects: • `slideOver` • `wipeIn` • `wipeOut` If you do not specify a direction, the transition moves toward the left side of the screen. This attribute is optional.

Code sample: Specifying a transition effect

```
<rim:loadingScreen backgroundColor="#AAA000"
                   onLocalPageLoad="true">
   <rim:transitionEffect type="zoomIn" />
</rim:loadingScreen>
```

Access element

The `access` element specifies that a BlackBerry® WebWorks™ application can access external network resources. By default, if you do not specify an `access` element, an application has access to all local resources. Local resources include any resources in the application archive file. If you specify more than one `access` element, the most specific definition is used. For example, if you use `http://somedomain.com` and `http://specific.somedomain.com`, the `access` element that uses the first definition (and any features defined under it) is ignored.

If your application makes calls to APIs on an external server, the external server must be declared in the access list.

Attribute	Description
`uri`	The `uri` attribute defines the web address for the access request. This attribute is required.
`subdomains`	The `subdomains` attribute is a Boolean value that specifies whether the host component in the access request applies to subdomains of the domain that is specified in the `URI` attribute. This attribute is optional. By default, if you do not specify the value of the `subdomains` attribute, the value is set to `False` and no access to subdomains is requested.

Code sample: Allowing access to external network resources and features

```
<access uri="http://www.mydomain.com" subdomains="true">
   <feature id="blackberry.app.event" required="true" version="1.0.0"/>
</access>
```

Feature element

The `feature` element specifies a BlackBerry® API that a BlackBerry® WebWorks™ application can use. You can use the `feature` element within the `access` element to access a feature under the external domain. You can also use the `feature` element at the root of the configuration document, before any `access` elements, for access to the local web page.

If you do not specify an `access` element above the `feature` element, the local domain is used. If you do not specify an API, the application cannot use the API for the specified domain.

Attribute	Description
`id`	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 wildcard (*) to use multiple namespaces at the same time. This attribute is required.
`required`	The `required` attribute is provided to conform to the W3C® specification. This attribute is optional.
`version`	The `version` attribute specifies the library version of the API. This attribute is optional.

Code sample: Specifying access to a feature

```
<feature id="blackberry.ui.dialog" required="true" version="1.0.0"/>
<access uri="http://www.somedomain.com" subdomains="true">
   <feature id="blackberry.app.event" required="true" version="1.0.0"/>
   <feature id="blackberry.invoke" required="true" version="1.0.0"/>
</access>
```

Content element

The `content` element specifies the start page that the BlackBerry® WebWorks™ application displays when it runs. The start page can contain the web address of a file that is located outside of the application archive.

Attribute	Description
`src`	The `src` attribute specifies the source HTML file in the application archive. This attribute is required.

Attribute	Description
`type`	The `type` attribute specifies the MIME type of the file that is specified in the `src` attribute. This attribute is optional.
`charset`	The `charset` attribute specifies the character set that is used by the file that is specified in the `src` attribute. This attribute is optional.

Code sample: Specifying a start page

```
<content src="startpage.html" />
```

Permissions element

The `<rim:permissions>` element specifies the permissions for various features in the BlackBerry® WebWorks™ application. You must enter the permission settings in your config.xml file.

Permission	Description	Default value
access_shared	The application can read and write files that are shared between all applications run by the current user.	`prompt`
record_audio	The application can access the audio stream from the microphone.	`prompt`
read_geolocation	The application can read the current GPS location of the device.	`prompt`
read_device_identifying_information	The application can discover unique identifiers such as the PIN number of the BlackBerry® PlayBook™ tablet.	`prompt`
use_camera	The application can access the data coming from one of the cameras.	`prompt`

Code sample: Specifying permissions

The following sample demonstrates how to set the permissions for reading and writing files, reading GPS location information, and accessing camera data.

```
<rim:permissions>
    <rim:permit>access_shared</rim:permit>
    <rim:permit>read_geolocation</rim:permit>
    <rim:permit>use_camera</rim:permit>
</rim:permissions>
```

Default permissions

The default permissions can be changed in the Security settings of the settings page in the notification area by the BlackBerry® tablet user.

Permission	Setting action
`allow`	The action is allowed without prompting the user, but the user may change the setting at any time in the Security settings page of notification area.
`prompt`	The user is prompted to allow or deny the action the first time the application opens.
`deny`	The action is denied without prompting the user, but the user may change the setting at any time in the Security settings page of notification area.

Permission behaviors

Permission settings can cause some unexpected behaviors in your application. For example, if you do not set the `use_camera` permission to `allow`, then each time the user attempts to open the camera the user is prompted to allow access. The following list of behaviors can occur when you set permissions in your application:

- If you do not specify the `record_audio` setting in the blackberry_tablet.xml file, the user is prompted when the app tries to use the microphone. After the user accepts the prompt, the app can use the microphone.
- For the`access_shared` and `use_camera` settings, if the permission is not specified, the user is not prompted and the feature doesn't work.

Category elements

The `<rim:category>` element specifies the category installation location of your BlackBerry® WebWorks™ application. For example, if your application is a game, you can specify that your application is installed in the Games folder on the BlackBerry® tablet.

```
<rim:category name=”games”/>
```

You can specify this category in the config.xml file. If the `rim:category` tag or attribute name and its value are not provided, or are misspelled, the default category is `All`. The values are case insensitive.

Attribute	Description
`<rim:category name="value"/>`	The value can be `games` to specify the Games folder, or `media` to specify the Media folder.

Securing BlackBerry WebWorks applications 3

The content of your BlackBerry® WebWorks™ application runs inside a standard BlackBerry® PlayBook™ tablet application, so the application can use the same policy and security configuration that other tablet applications use.

Inside a BlackBerry WebWorks application, you can specify that the application can access BlackBerry WebWorks APIs for a domain in an executable context. An executable context is a container in which you can run JavaScript® code within the context of a web page. An executable container can be a page, a frame, or an iframe. If a local page provides the BlackBerry WebWorks API with access to the menu and linked in JavaScript code, you can include the domain by using application permissions for the executable container without specifying any feature elements.

Allowing access to external resources and APIs

By default, BlackBerry® WebWorks™ applications cannot access data from external resources. For example, a BlackBerry WebWorks application cannot retrieve an HTML web page or make an AJAX request to a web service, unless you configure the application to allow access.

To allow access to external resources and BlackBerry WebWorks APIs, you must specify in the configuration document for the application the resources and the APIs that you require. You can define the list of domains that your application is allowed to access and the BlackBerry WebWorks APIs that are allowed for each domain. You can define this list using application permissions.

The BlackBerry® WebWorks™ Packager follows the same origin policy for the resources that the application requests by matching the resources to entries in the permissions list.

Whenever you retrieve content from external resources, consider the following best practices to help make the application as secure as possible:

- Provide JavaScript® access to sensitive APIs only to trusted and secure web sites.
- Protect your communication channel by using HTTPS when you expose sensitive APIs to the domain.
- Use the same precautions that you would use for a hosted web site, to protect against users with malicious intent.

Allowing requests to any web site

If your BlackBerry® WebWorks™ application is designed to access data from an unknown domain or a changing domain, you can use the `access` element with the wildcard character to make sure that your requests are not blocked.

When you use the wildcard character, web pages that your application accesses cannot access any of the application APIs.

In the following code sample, all requests that do not require access to application APIs are allowed.

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

Allowing requests to specific web sites

If your BlackBerry® WebWorks™ application is designed to access API functionality on a domain, you must use the `access` and `feature` elements to specify the domain to make sure that your requests are not blocked.

For example, if you want to update or change menu items from a domain, you must specify the domain and the APIs that you require.

In this example, the APIs that you specify under the domain `mydomain` are allowed. The ellipses in the example represent specific APIs that your application uses.

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

Compiling BlackBerry WebWorks applications 4

When you compile a BlackBerry® WebWorks™ application, all of the resources in the archive, including a start page, a configuration document, icons, and other resources, are used to create a .bar file. You can install the resulting .bar file on the BlackBerry® Tablet Simulator or on the BlackBerry® PlayBook™ tablet.

Compiling a BlackBerry WebWorks application

When you compile a BlackBerry® WebWorks™ application, the BlackBerry® WebWorks™ Packager performs the following actions:

1. Validates the contents of the BlackBerry WebWorks archive file.
2. Creates the output target folder and cleans up any old files (if necessary).
3. Creates the source target folder (if specified) and cleans up any old files (if necessary).
4. Creates the .bar file.

Compile a BlackBerry WebWorks application

To package and compile your application, you must run the BlackBerry® WebWorks™ Packager.

Before you begin:

Create your application .zip archive with all the required elements and objects.

1. At a command prompt, navigate to the installation folder for the BlackBerry WebWorks Packager. The file path may vary based on where you installed the BlackBerry WebWorks Packager.

   ```
   cd C:\Program Files\Research In Motion\BlackBerry WebWorks SDK for TabletOS<x.x.x.x>\bbwp
   ```

2. Compile the application by using the following syntax:

   ```
   bbwp "C:\myapp\myarchive.zip" -o "C:\myapp\output"
   ```

   The BlackBerry WebWorks Packager displays multiple messages, and indicates a successful compile if the application archive contains no errors.

3. If successful, this process creates a .bar file in the folder specified by the `-o` parameter. If no `-o` parameter is specified, the .bar file is created in a subfolder named "bin" that's located in the same folder location as the archive file.

Compilation parameters

Parameter	Description
`archive`	This parameter specifies the name of the BlackBerry® WebWorks™ archive to compile. The archive is a .zip file.

Parameter	Description
`-o`	This parameter saves the output files to the specified folder. If you do not use the `-o` parameter with a folder, the output is saved to the bin folder within same path as the application archive.
`-s`	This parameter saves the source files. By default, if you specify `-s` without a folder, the source files are saved to the src file within same path as the application archive. If you specify a folder, the source files are saved in the src folder within the folder that you specify.
`-gcsk` `-gp12`	These parameters specify passwords for the long term key (csk) and developer key (p12) for a signed application. If you do not specify either of these parameters, the .bar file will be unsigned. Otherwise, you must specify both parameters.
`-buildId`	For signed applications, this parameter specifies the build number. Typically, this number should be incremented from the previous signed application. If you specified a version number in the config.xml file, the build number you specify becomes the fourth digit of the version number. For example, either 1.2.0 or 1.2.0.1 (in config.xml) become 1.2.0.7 if you specify a build number of 7. If you do not specify a build number, the version number stays the same (if it is four digits long) or a fourth digit of 0 is added (if it is three digits long). For example, 1.2.0.1 remains the same, and 1.2.0 becomes 1.2.0.0.
`-v`	This parameter displays verbose messages during compilation, and is optional.
`-h`	This parameter displays command line help, and is optional.
`-d`	This parameter sets the developer mode on which affects Web Inspector feature and the debug token. -d will always enable web inspector which allows you to open your app in the Google Chrome™ browser, and use Web Inspector to debug your code. -d option will use debug token information only if both of the following are true: (1) There is a <debug_token> element in bbwp.properties and it contains the full pathname of a valid debug token file (2) Signing is NOT specified (i.e., no -gcsk and no -gp12 specified) when signing your app.

Code sample: Compiling a BlackBerry WebWorks archive

The BlackBerry® WebWorks™ Packager uses the BlackBerry® WebWorks™ archive test.zip file to produce an unsigned version of the compiled application.

```
bbwp test.zip
```

Code sample: Compiling a signed BlackBerry WebWorks application

The BlackBerry® WebWorks™ Packager uses the BlackBerry® WebWorks™ archive test.zip to produce the following output:

- a signed version of the compiled application, in the .\compiled\test.bar file
- templates that generate the .bar file, in the .\archive_source\src\blackberry\ folder
- extracted files from test.zip, in the .\archive_source\src\ folder

```
bbwp c:\test.zip -gcsk cskpassword -gp12 p12password -buildId build_number -s
archive_source -o compiled
```

Enable the Web Inspector

You can use the `-d` option flag when you package your application file. This allows you to use a WebKit desktop browser such as Google Chrome™ and the Safari browser to navigate to the IP address of the tablet simulator where you can view the currently loaded page source in your browser. You can also view the IP address in the tablet simulator when you run your application. This is a good way to test and debug your code.

Note: Your BlackBerry® tablet must be connected through Wi-Fi® to access the IP address.
Example:

```
bbwp "C:\myapp\myarchive.zip" -d -o "C:\myapp\output"
```

With your application running in the simulator, in a desktop browser, type **http://*<IP address of simulator>*:*<port number>***.

Only use the `-d` option when you build development versions of your application. Do not use it when building release versions (involving code signing). Leaving the debugging port open may allow unintentional access to your application.

Best practice: Compiling a BlackBerry WebWorks application

Best practice	Description
Name the archive file and resources consistently and correctly.	When you compile a BlackBerry® WebWorks™ application, the BlackBerry® WebWorks™ Packager uses various resources. The BlackBerry WebWorks Packager does not verify the file names, so you must make sure that all the file names and file paths match the settings in the application configuration document. If the archive or files within the archive are named improperly, the application will not compile. The BlackBerry WebWorks Packager checks references to the application permissions, the start page, and the application icons.

Best practice	Description
	Consider the following when naming the archive file and its contents: • For the archive file name, use only alphanumeric characters, and begin file names with a letter. • For files within the archive, use only alphanumeric characters, the underscore, or periods. File names must begin with a letter. • The name of the .bar file for the BlackBerry® PlayBook™ tablet will be the same as the archive file name. • Resource names within the .bar file are case sensitive.
Verify that all the required elements are in the configuration document.	At a minimum, you must define the namespace and specify the application, version, and name elements in the configuration document. You might require the author element if your application is distributed by another service or application. Check for proper syntax in the configuration document and that it is a valid .xml file.
Make sure that the elements your application requires are defined in the configuration document.	Some elements are not required by the BlackBerry WebWorks Packager but might be required for your application to run properly. Make sure that the permissions are defined properly if the application must be able to make requests to external domains, and that feature elements are defined for any BlackBerry WebWorks APIs you use.

Signing your application

5

Before you can distribute your application for use on a BlackBerry® tablet, you must sign your application. The signing tool included with the BlackBerry® Tablet OS SDK adds cryptographic hash values to your application package (BAR file) during the signing process. The hash files help verify the authorship of your application to users and the BlackBerry Tablet OS.

If you want to test your application on a device without signing it, you can create and install a debug token on the device. For more information about debug tokens, see "Running unsigned applications using a debug token".

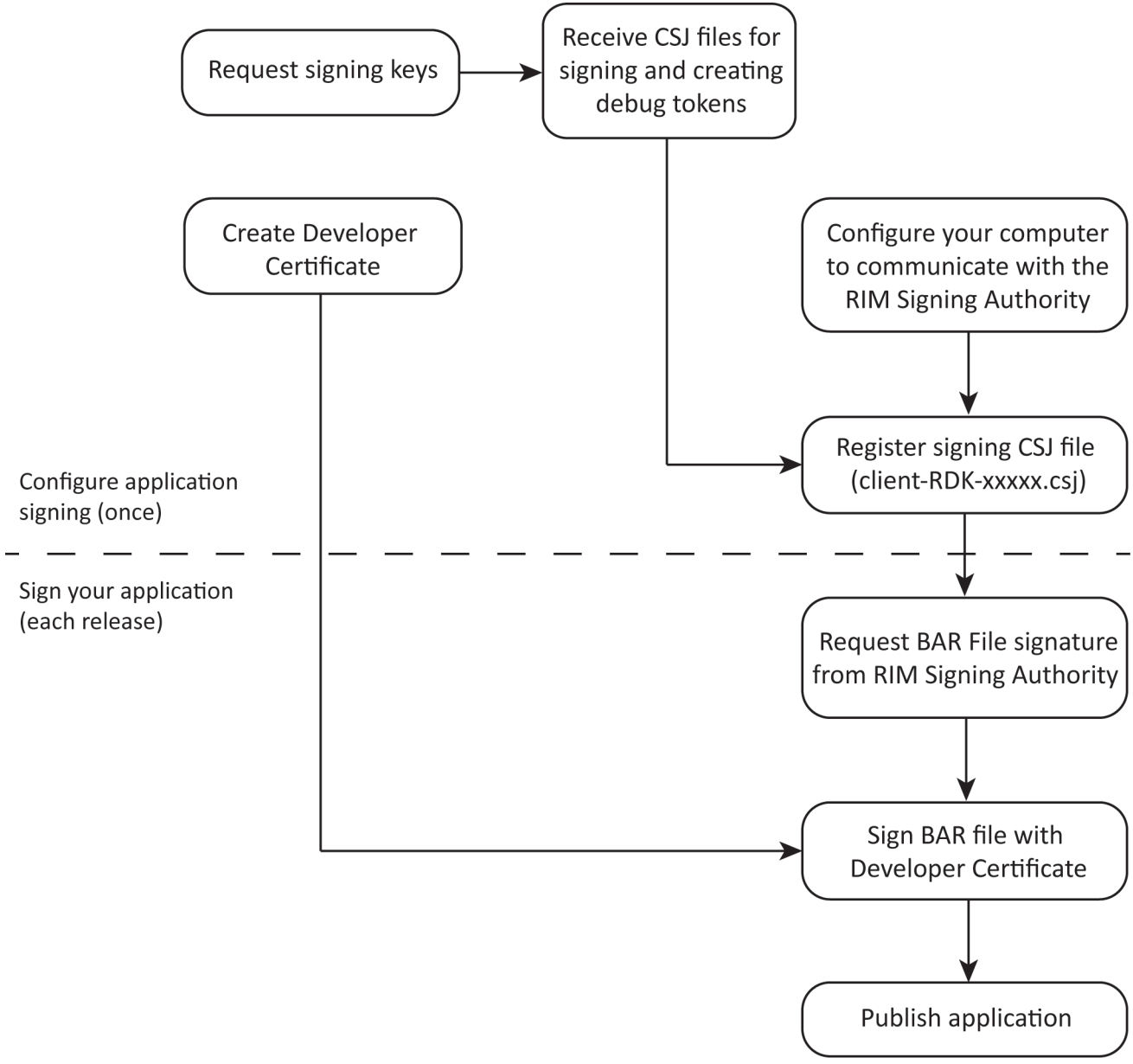
The following steps describe the application signing process. For more information about how to sign an application using a particular tool, follow the links to the tool-specific topics below. There are two phases in the signing process. In the first phase, you configure your computer to sign applications. You should only need to configure your computer once. To configure your computer:

1. Request permission to sign BlackBerry Tablet OS applications by completing the web form at https://www.blackberry.com/SignedKeys. After your application is accepted, you receive two CSJ registration files by email. One file allows you to configure your keystore to sign applications; the other file allows you to create debug tokens. Each file arrives in a separate email with information about the purpose of the file attached.
2. Create a Developer Certificate. Ensure that the common name (company name) parameter matches the name submitted when you requested the CSJ registration files. For more information about the blackberry-tablet.xml file, see Configuring your application.
3. Configure your computer to communicate with the RIM® Signing Authority
4. Register with the RIM Signing Authority using your CSJ registration file for application signing.

In the second phase of the signing process, you sign your application. You must sign your application each time you want to publish it. To sign your application:

1. Request a BAR file signature for your application from the RIM Signing Authority.
2. Apply your signature to the BAR file using your Developer Certificate.

After you sign your application, you can publish it to the BlackBerry App World™ storefront.



Command line parameters for the blackberry-keytool tool

You can use the `blackberry-keytool` tool to generate a code-signing certificate.

Parameter	Description
`genkeypair`	This parameter instructs the tool to generate a pair of mathematically related values used to encrypt and decrypt data. Data encrypted with one value must be decrypted by the other value. Anyone can verify that an application encrypted with one value, which is kept private (signed using the private key), by decrypting it with the other value (the public key).
`keystore`	This parameter specifies the location of the file that contains the public and private key pair.
`storepass`	This parameter specifies the password that is used to control access to the key store.
`dname`	This parameter specifies the name to associate with the certificate that the tool creates. The value for this parameter must match exactly: • the name you entered in the **Company** field on the web form to request permission to sign applications • the **publisher** element in your blackberry-tablet.xml file
`alias`	This parameter refers to an entry in the key store database. The value for this parameter must be the literal value `author`.

Command line parameters for the blackberry-signer tool

You can use the `blackberry-signer` tool to communicate with the RIM® Signing Authority, and sign your application.

Parameter	Description
`csksetup`	This parameter instructs the tool to generate a pair of mathematically related values used to encrypt data exchanged during communication with the RIM Signing Authority.
`cskpass`	This parameter specifies the password that is used to control access to the key pair created by using the `csksetup` parameter
`register`	This parameter instructs the tool to request that the RIM Signing Authority associate an CSJ registration file with the public key that is created by using the `csksetup` parameter
`csjpin`	This parameter specifies the PIN for the CSJ registration file that is submitted during a registration request.
CSJ file	This parameter specifies the location of the Developer Certificate from the RIM Signing Authority.

Configure signing for tablet applications

Before you begin: Register with RIM to receive a .csj file, the .csj PIN, and a developer company name. Visit https://www.blackberry.com/SignedKeys and complete the registration form.

After you complete the form, Research In Motion sends an email message containing a .csj file. The .csj file contains a list of signatures and your registration information.

Upgrade Instructions: The following steps are for a first-time setup. However, if you are performing a BlackBerry® Tablet OS SDK upgrade then you need to only perform step 1 -generate sigtool.p12 using the same password as you used before. Alternatively you can make a backup copy of the sigtool.p12 file and restore it to the location of the upgraded SDK after the upgrade. **Note** that an upgrade will **delete** the sigtool.p12 file.

1. From the "\bbwp\blackberry-tablet-sdk\bin" folder located in the BlackBerry WebWorks SDK installation folder, run the RIM key tool to generate a developer certificate:

   ```
   blackberry-keytool -genkeypair -keystore sigtool.p12 -storepass <p12 password> -
   dname "cn=<company name>" -alias author
   ```

   Note that `author` is a literal keyword. You must type the word "author" as it appears here, and not substitute any other value in the command.

   This produces the sigtool.p12 file in the current directory where <p12 password> is a password you choose and <company name> is the company name sent to you during the signing registration process. Copy or move this file into the bbwp/bin folder of the BlackBerry WebWorks SDK.

2. From the same "\bbwp\blackberry-tablet-sdk\bin" folder located in the BlackBerry Tablet OS SDK installation folder, run the .bar signing tool to set up long term keys for communication with the signing authority server:

   ```
   blackberry-signer -csksetup -cskpass <csk password>
   ```

   where *<csk password>* is a password you choose.

   This produces the barsigner.csk in the C:\Document and Settings\<name>\Local Settings\Application Data\Research In Motion folder (in Windows®) or the ~\Library\Research In Motion folder (in Mac OS).

3. From the same "\bbwp\blackberry-tablet-sdk\bin" folder located in the BlackBerry Tablet OS SDK installation folder, run the .bar signing tool to enroll with the signing authority server:

   ```
   blackberry-signer -register -csjpin <csj pin> -cskpass <csk password>
   <filename.csj>
   ```

   where *<csj pin>* and *<filename.csj>* are the values sent by email during the signing process.

Sign the .bar file for a BlackBerry WebWorks application

You can sign your BlackBerry® WebWorks™ application when you compile it, by using the password that you defined when you requested and installed the registry key.

Use the BlackBerry® WebWorks™ Packager and specify the `-gcsk`, `-gp12`, and `-buildId` options with their respective passwords:

```
bbwp "C:\myarchive.zip" -gcsk mycskpassword -gp12 myp12password -buildId 10 -o "C:
\myoutputdir"
```

Running unsigned applications using a debug token

You can run unsigned applications on a BlackBerry® tablet by using a debug token. Debug tokens allow an organization to separate the process of application creation and publication. A developer can create and test an application using a debug token, then deliver the application to their supervisor or their client for signing and publication.

When you run an unsigned application using a debug token, you can avoid:

- changing the version number of your application
- accessing the internet
- exporting a release build of your application

A debug token must be created by an individual or organization that has permission to sign BlackBerry® Tablet OS applications. To request permission to sign applications, complete the web form at https://www.blackberry.com/SignedKeys. After your request is approved, you receive two CSJ registration files by email. One file allows you to configure your computer to sign applications; the other file allows you to create debug tokens. Each file arrives in a separate email with information about the purpose of the file attached.

After you receive your CSJ registration files, you can configure your computer to create debug tokens. When you create a debug token, you specify the PIN for each tablet on which the token can be used. You can distribute the debug tokens you create to developers to install on those tablets, or install them yourself. You are limited to a total of 100 tablet PINs across all of your debug tokens that are currently active. If you create debug tokens that address 100 PINs, you must wait for some of your debug tokens to expire before you create more.

Debug tokens are valid for 30 days. When a debug token expires, the BlackBerry Tablet OS no longer allows unsigned apps that rely on that token to run.

Configure application signing from the command line

The following steps help you configure your computer to sign applications. You should only perform these steps once.

You can find `blackberry-signer` and `blackberry-keytool` in the bin subfolder where you installed the BlackBerry® Tablet OS SDK.

Before you begin: Request permission to sign BlackBerry Tablet OS applications by completing the web form at https://www.blackberry.com/SignedKeys. After your application is accepted, you receive a CSJ registration file by email.

1. Configure your keystore to communicate with the RIM® Signing Authority by using the `blackberry-signer` tool.

   The password you specify in the `-cskpass` parameter allows you to access the RIM Signing Authority using your identity. You should protect the password.

   ```
   blackberry-signer -csksetup -cskpass <create_a_password>
   ```

   If you already configured your keystore and registered your CSJ file, this command returns the following message:

   ```
   CSK file already exists.  Use -cskdelete to delete first.
   ```

   Do not use the `cskdelete` command line option unless you forgot your CSK password. If you forgot your CSK password and delete the CSK file, you need to request a new CSJ registration file. To request a new CSJ file, visit https://www.blackberry.com/SignedKeys.
2. Register with the RIM Signing Authority using your CSJ registration file for appilcation signing by using the `blackberry-signer` tool. In this step, you need to specify the registration PIN you entered in the **PIN** field on the web form when you requested permission to sign applications.

   If you connect to the Internet through a proxy server, you must specify additional command line options to contact the RIM Signing Authority. For more information about using a proxy server from the command line, see Using a proxy server from the command line.

   ```
   blackberry-signer -register -csjpin <PIN>
       -cskpass <csk password from step 1> <CSJ_file>
   ```
3. Create a Developer Certificate by using the `blackberry-keytool` tool. Ensure that the common name (company name) parameter matches the name submitted when you requested the CSJ registration files. For more information about the blackberry-tablet.xml file, see Configuring your application.

   The following command creates a Developer Certificate in the file named *output_file*.p12. The password you specify in the `-storepass` parameter allows you to use the .p12 file to sign BAR files. You should protect the .p12 file and its password.

   ```
   blackberry-keytool -genkeypair -keystore <output_file.p12>
        -storepass <create_a_password> -dname "cn=<company_name>" -alias author
   ```

For more information about the command line tools used in this task, please see Command line parameters for the blackberry-keytool tool, and Command line parameters for the blackberry-signer tool.

Package your application from the command line

Before you begin:

Add the path to the bin subfolder where you installed the BlackBerry® Tablet OS SDK to the PATH environment variable for your operating system. This path should be C:\Program Files\Research In Motion\BlackBerry WebWorks SDK for TabletOS\bbwp.

1. Open a command prompt.
2. Navigate to the folder where your application files are stored.
3. Use the `bbwp` command to package your application. For example:

```
bbwp "myarchive.zip" -o "c:\myoutputdir"
```

To enable your unsigned application to run using a debug token, or to enable using your application with Web Inspector, include the `-d` option on the command line.

Set the debug token from the command line

The BlackBerry® Tablet OS allows you to run unsigned applications on a BlackBerry® tablet that has a debug token installed. After installing the debug token on the device, all that the developer needs to do is to edit the `bbwp.properties` file to tell it where to find the debug token. This is useful in the testing phase of your application development process.

Set the debug token

Before you begin:

Developers must be registered with the signing authority server by running the **DebugTokenRequest** (similar to BarSigner) tool with a Windows command line like this: **blackberry-debugtokenrequest -register -cskpass <csk_password> -csjpin <csj_pin> <csj_file>.**

Mac OS Note: Replace back slashes with front slashes.

The device must be set in Development mode.

1. A registered developer requests a debug token by running the following Windows command line with a list of device Ids specified in the command line: You can find the the blackberry-debugtokenrequest in the folder in yourBlackBerry® WebWorks™ SDK installation.

   ```
   blackberry-debugtokenrequest  -cskpass <csk_password>  -keystore <p12_file>
   -storepass <p12_password> -deviceid <device_id_1> [... -deviceid <device_id_n>]
    <debug_token_filename>
   ```

   Device ID is the hex value of the PIN (prefixed with '0x'). For example, 8-digit hex value: 0x12345678
2. Install the debug token on device using the **blackberry-deploy -installDebugToken** command.

   ```
   blackberry-deploy -installDebugToken <path to token> -device <device> -password
   <password>
   ```

   - where

     <token> - is a path to debug token file.
     -device <device> - is hostname or the IP address of the target device or simulator.

     -password <password> - device's password.
3. Edit the `bbwp.properties` file and insert the path information in the debug token in <debug_token> tags as shown in the example below.

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <wcp>
   <additional>-quiet</additional>
   <java>C:\Program Files\Research In Motion\BlackBerry WebWorks SDK for TabletOS
   2.1.0\jre</java>
   ```

```
<wcp_template>AirAppTemplates</wcp_template>
<air_template>AirAppTemplates</air_template>
<tablet_sdk>C:\Program Files\Research In Motion\BlackBerry WebWorks SDK for
TabletOS 2.1.0\bbwp\blackberry-tablet-sdk</tablet_sdk>
<extension_repository>ext</extension_repository>
<debug_token>C:\my_debug_token.bar</debug_token>
</wcp>
```

You can now load your debug-enabled application on a tablet device without signing it.

Setting debug token with -d option behaviors

The -d option setting is a development mode setting that also enables the Web Inspector feature by default.

-d will always enable Web Inspector. It will set the debug token only if both of the following are true:

- There is a <debug_token> element in bbwp.properties and it contains the full pathname of a valid debug token file.
- Signing is NOT specified (i.e. no -gcsk and no -gp12 specified when you sign your application).

Package and deploy your application from the command line

1. Open a command prompt.
2. Navigate to the folder where your application and XML configuration files are stored.
3. Type the following command to package, deploy, and launch your application from the command line.

   ```
   blackberry-deploy -installApp -device <deviceIP> -package <BAR file path>
    -password <device password>
   ```

   For information about device password and IP address refer Configure a virtual machine for the BlackBerry Tablet Simulator.
4. Actual example.

   ```
   C:\Program Files\Research In Motion\WebWorks Packager
   for PlayBook\bbwp\blackberry-tablet-sdk\bin>blackberry-deploy -installApp
   -device 192.168.198.134 -package "c:\sandbox\web apps\helloWorld\bin
   \helloWorld.bar"
    -password xxxxx
   ```

   Note: You can use a batch file to make testing easier. Examples can be found here. ANT script examples can be found here.

Retrieve the PIN of a BlackBerry tablet

1. In the upper-right corner of the Home screen on your BlackBerrytablet, press the gear icon to open the BlackBerry tablet options.

2. In the **View information about your tablet** field, select **Hardware**.
3. Record the value from the **PIN** field.

Application Signing Errors

Attempt to rename [value1] to [value2] failed

Description

During the signing process, the signing tool makes a copy of the input .bar file. After the tool signs the copy of the .bar file successfully, the copy is renamed to the original input file name and the original file is deleted. This error indicates that one of these rename operations failed.

Possible solution

Make sure that you have permissions to modify files in the folder where your .bar file is stored.

Code signing request failed because Application-Development-Mode in Manifest is present and is not set to false

Description

This message appears when you try to sign a .bar file that was packaged in Development mode.

Possible solution

Repackage the .bar file. If you use `blackberry-packager`, or another command line tool, do not include the `debug` option in your command line.

Code signing request failed because Common Name in developer certificate is not [value]

Description

The Common Name (CN) property in your Developer Certificate does not match the value you provided in the Company field when you requested permission to sign applications by using the RIM® Signing Authority.

Possible solution

Obtain a Developer Certificate that matches the name you provided when you requested permission to sign applications.

Possible solution

Create a self-signed Developer Certificate using the `blackberry-keytool` tool, that has a CN property that matches the name you provided when you requested permission to sign applications.

Possible solution

Reapply for permission to sign applications with the RIM Signing Authority at http://www.blackberry.com/SignedKeys. On the web form, in the **Company** field, provide the name that appears in your Developer Certificate.

Code signing request failed because this file has been previously signed

Description

You can sign an application only once per version number.

Possible solution

1. In the *your_project_name*-app.xml file, increment the value in the `versionNumber` element.
2. Repackage your application.

Code signing request failed because Package-Author in Manifest is not set to [value]

Description

The publisher element in your blackberry-tablet.xml file does not match the value you provided in the Company field when you requested permission to sign applications by using the RIM® Signing Authority.

Possible solution

Change the value of the publisher element in your blackberry-tablet.xml file.\

Possible solution

Reapply for permission to sign applications with the RIM Signing Authority at http://www.blackberry.com/SignedKeys. On the web form, in the **Company** field, provide the name that appears in your publisher element.

Code signing request failed because value of Package-Name in Manifest is not allowed

Description

Your application has a name that is restricted.

Possible solution

1. Change the name of your project.
2. Recompile your project.
3. Repackage your project.

-cskpass and -csjpin must be specified if -register is specified

Description

This message appears when you try to run the `blackberry-signer` tool with the `register` option, but you did not include both the `cskpass` and `csjpin` options.

The `blackberry-signer` tool tries to encrypt communication between your computer and the RIM® Signing Authority. The tool stores the encryption keys an encrypted file that is designed to be unlocked by using the password your provide in the cskpass option.

The CSJ Pin is a 6 to 10 digit number you selected when you requested permission to sign applications. The number is designed to verify that the CSJ registraion file is only used by the person who requested it.

Possible solution

Add the `cskpass` and `csjpin` options to your command line. Make sure you provide appropriate values for each option.

cskpass required

Description

You requested a signature from the RIM® Signing Authority, but did not include the `cskpass` option. To contact the RIM Signing Authority, the `blackberry-signer` tool must decrypt the barsigner.csk file using the password you provide using the `cskpass` option.

Possible solution

Include the `cskpass` parameter and the password in your command line.

Developer certificate and private key not found in keystore or store password not supplied

Description

You attempted to sign an application using the `blackberry-signer` tool, but you did not provide a value for the storepass option, or your Developer Certificate has no private key named author.

Possible solution

Verify that your command line includes the storepass option, and specifies the correct password.

Possible solution

Recreate your Developer Certificate. You can use the `blackberry-keytool` tool. Make sure that you provide the literal value "author" (not your name) for the alias option.

Incomplete certificate chain

Description

The Developer Certificate that you provided to the `blackberry-signer` tool was issued by a Certificate Authority. However, the series of signatures (from a root authority to your certificate) that establish the authenticity of your certificate was not found in your keystore along with your Certificate.

Possible solution

Add the certificate chain to the keystore.

Key associated with [value] not a private key

Description

The `blackberry-signer` tool found a application signing key using the keyname parameter you specified on the command line. However, the key found is inappropriate for application signing.

Possible solution

- To request a signature from the RIM® Signing Authority, make sure you specify **RDK** for the keyname parameter.
- To sign your BAR file using your Developer Certificate, make sure you specify **author** for the keyname parameter.

Keystore load: store password incorrect

Description

The value you specified for the `storepass` option is incorrect. The `blackberry-signer` tool cannot decrypt the keystore for your Developer Certificate.

Possible solution

Provide the correct password for your keystore.

keytool error: java.io.FileNotFoundException: certificate.p12 <Access is denied>

Description

You tried to use the `blackberry-keytool` tool to create a self-signed Developer Certificate. The tool does not have permission to write to the location you provided for the *`output_file.p12`* parameter.

Possible solution

Acquire read-write permissions for the folder to which you want to write the P12 file.

Possible solution

Choose another location for your P12 file.

Possible solution

Run `blackberry-keytool` with superuser permissions.

keytool error: java.lang.Exception: Key pair not generated, alias <author> already exists

Description

You tried to use the `blackberry-keytool` tool to create a self-signed Developer Certificate. The file name you provided for the *output_file.p12* parameter already exists.

Possible solution

Change the file name so that `blackberry-keytool` creates the output file using a different name or location.

Possible solution

Delete your existing P12 file. Then run `blackberry-keytool` again.

Missing parameter for [option_name] option

Description

You specified a command line option, `option_name`, that takes a parameter. However, you did not supply a value for the parameter.

Possible solution

Provide a value for the parameter.

No BAR file or CSJ file specified

Description

This error appears when you try to use the `blackberry-signer` tool to sign or verify a .bar file, but no .bar file was specified on the command line.

This error can also appear when you try to use the `blackberry-signer` tool to register with the RIM® Signing Authority, but no CSJ file was specified on the command line.

Possible solution

Specify the appropriate file.

No key name specified

Description

You cannot sign your .bar file because you did not specify a signing key.

The final parameter of the `blackberry-signer` tool specifies the key with which to sign your .bar file. Your .bar file must be signed twice: first by the RIM® Signing Authority, then by your Developer Certificate.

Possible solution

To request a signature from the RIM Signing Authority, type **RDK** to the end of your command line.

Possible solution

To sign your BAR file using your Developer Certificate, type **author** to the end of your command line.

No manifest

Description

The BAR file is corrupt.

Possible solution

Repackage your project using the `blackberry-packager` tool.

Only one of -setup, -register, or -verify can be specified

Description

You cannot execute any of the following functions concurrently: setup, register, and verify. They are mutually exclusive operations.

Possible solution

Perform these functions separately.

Server is not responding

Description

The signing tool cannot contact the RIM® Signing Authority.

A problem with the network connection prevents the `blackberry-signer` tool from contacting the RIM Signing Authority. The signing tool cannot sign your BAR file until it contacts RIM Signing Authority.

Possible solution

Check your computer's network connection.

Signature filename must consist of the following characters: A-Z, 0-9, _ or -

Description

The value for the keyname option on the `blackberry-signer` command line contains unexpected characters.

Possible solution

The keyname must be RDK, or author.

Unable to open BAR file

Description

The `blackberry-signer` tool cannot read the BAR file you specified on the command line. You may not have permission to access this file, or the file may be missing or corrupt.

Possible solution

Verify that you have permission to access the file and that it exists at the location you provided on the command line. If the file exists and you have access to it, then try repackaging your project using the `blackberry-packager` tool.

Certificate chain not found for: RDK. RDK must reference a valid KeyStore key entry containing a private key and corresponding public key certificate chain.

Description

This message appears when you request a signature from the RIM® Signing Authority, but you did not successfully register your computer before running this command.

Possible solution

Make sure that you properly configured your computer for signing.

keytool error: java.io.IOException: Incorrect AVA Format

Description

You tried to use the `blackberry-keytool` tool to create a self-signed Developer Certificate. The `cn` value for the `dname` parameter contains a special character. The special character set includes:

- Commas (,)
- Additon symbol (+)
- Quotation mark (")
- Back-slash (\)
- Angled brackets(< and >)
- Semi-colon (;)

Possible solution

Add a back-slash in front of the special character in your command line. The following shows an excerpt from an incorrect command line:

```
... -dname "cn="One, two, three" ...
```

The following corrects the excerpt above:

```
... -dname "cn=One\, two\, three" ...
```

Testing BlackBerry WebWorks applications 6

Run a BlackBerry WebWorks application on the BlackBerry Tablet Simulator

Before you distribute your BlackBerry® WebWorks™ application to users, you can test it using the BlackBerry® Tablet Simulator. The BlackBerry Tablet Simulator enables you to run and view your applications exactly as you would on a real BlackBerry® PlayBook™ tablet.

Before you begin:

Successfully compile a BlackBerry WebWorks application.

1. Start the BlackBerry Tablet Simulator.
2. In the simulator, click the first icon to the right of the clock and ensure that development mode is enabled. Record the IP address of the tablet.
3. From a command prompt, navigate to bbwp/blackberry-tablet-sdk/bin under the installation folder.
4. Type the following command to load and run your application on the simulator:

```
blackberry-deploy -installApp -password <simulator password> -device <simulator IP address> -package <BAR file path>
```

Package and deploy your application from the command line

1. Open a command prompt.
2. Navigate to the folder where your application and XML configuration files are stored.
3. Type the following command to package, deploy, and launch your application from the command line.

```
blackberry-deploy -installApp -device <deviceIP> -package <BAR file path>
 -password <device password>
```

   For information about device password and IP address refer Configure a virtual machine for the BlackBerry Tablet Simulator.

4. Actual example.

```
C:\Program Files\Research In Motion\BlackBerry WebWorks Packager
for PlayBook\bbwp\blackberry-tablet-sdk\bin>blackberry-deploy -installApp
-device 192.168.198.134 -package "c:\sandbox\web apps\helloWorld\bin
\helloWorld.bar"
 -password xxxxx
```

   Note: You can use a batch file to make testing easier. Examples can be found here and also ANT script examples here.

Remove your application from the BlackBerry Tablet Simulator

1. Open a command prompt window.
2. From the installation folder, navigate to bbwp/blackberry-tablet-sdk/bin.
3. Type the following command to remove your application from the BlackBerry® Tablet Simulator.

```
blackberry-deploy -uninstallApp -device <IP_address> -package <bar file name> -
password <device password>
```

Loading an application to the BlackBerry PlayBook tablet

Before you begin: To deploy your application to the BlackBerry® PlayBook™ tablet you first must build an application that you wish to deploy, and complete the application signing process, and obatin a debug token.

As part of your development process, you might wish to an online simulation tool called Ripple before you test your application on the tablet, or you can use the simulator that is provided in the BlackBerry® WebWorks™ SDK.

Make sure that your BlackBerry PlayBook tablet is on the same network as your computer. You might want to disconnect from VPN connections to avoid connection problems.

1. Turn on **Development Mode** on your BlackBerry PlayBook tablet.
2. Obtain the IP address of your BlackBerry PlayBook tablet.
3. At the command prompt, navigate to the bin folder of the BlackBerry® WebWorks™ SDK for BlackBerry® Tablet OS installation folder. For example:

```
cd WebWorks SDK Install Dir\bbwp\blackberry-tablet-sdk\bin"
```

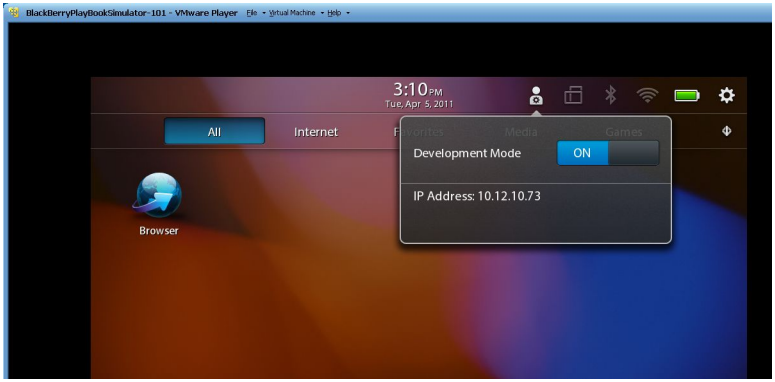
4. At the command prompt, run the `blackberry-deploy` command.

```
blackberry-deploy -installApp -password playbook -device 192.100.34.10 -package
myApp.bar
```

Retrieving the IP address of the BlackBerry PlayBook tablet

You must provide the IP address of the BlackBerry® PlayBook™ tablet to the deployment tool of your choice to load your application on the tablet. To display the IP address, the tablet must be in development mode. Development mode is enabled by default. To learn how to enable development mode manually, see Enable development mode.

You can view the tablet's IP address by tapping the icon to the right of the clock on the status bar.



Error Messages: Deploying applications to the BlackBerry Tablet Simulator

To deploy your application to the BlackBerry® Tablet Simulator you must perform the following actions:

- Enable development mode on the target BlackBerry Tablet Simulator
- Use the `blackberry-deploy` utility service from the command line prompt to deploy your application file to the simulator.

A `result:success` message appears if your application is deployed successfully. An error message appears if there is a problem with the deployment.

500 navigator interlock failed

Description

This message appears when a version of the application is already running on the simulator.

Possible solution

- Wait for the simulator to release the application lock. If it fails to release the application lock then restart the simulator.
- Use the *blackberry-deploy -uninstallApp* utility from the command line prompt to remove the application from the simulator.

Username or password is invalid or not specified or time on the device is out of sync

Description

This message appears in the following cases:

- If the VMware® Player was suspended, rather than the BlackBerry® Tablet OS being shut down, the timestamp may be out of sync with that of the host machine.
- The simulator password that was used during development is incorrect or expired.

Possible solution

- Update the simulator system time to match that of your development machine.
- Verify that the password is correct and not expired.
- Verify that the version of the BlackBerry® Tablet OS SDK is the same as the version of the BlackBerry Tablet OS.

Connection to <IP address> Refused

Description

This message appears when a VPN connection is blocking deployment.

Possible solution

1. Test the IP address by typing a `ping ip_address` at the command prompt, using the IP address of your simulator.
2. If testing the IP address with the `ping` command is unsucsessful, then the VPN connection is interfering with the connection to the simulator. Disable the VPN connection before you deploy your application.

Cannot connect: Connection to https://<IP address> refused. You may have to reboot the target.

Description

This message appears when the simulator is not responding.

Possible solution

- Restart the simulator.
- If the password is no longer accepted by the `blackberry-deploy` utility service, clear the password and set it again.

Device is not in the development mode. Switch to Development from Security settings on the device

Description

This message appears when you do not have development mode enabled on your simulator.

Possible solution

1. On the simulator, click the **System** icon on the taskbar.
2. Select the **Security** tab.
3. Switch the **development mode** option to **On**.

Invalid Auth Credentials

Description

This message appears after you upgrade your simulator and you are running an older version of the BlackBerry® Tablet OS SDK.

Possible solution

Upgrade the BlackBerry® Tablet OS SDK to the same version of the simulator that you are using.

500 development apps pool exhausted

Description

This message appears if you have more than ten applications installed on the simulator while in development mode.

Possible solution

Remove any unused applications from the simulator before you deploy your new application to the simulator. Refer to Remove your application from the BlackBerry Tablet Simulator

for instructions.

BlackBerry Tablet Simulator doesn't start - black screen at startup

Description

When you start the BlackBerry® Tablet OS simulator, it might launch a black screen on startup or it might freeze at the "BlackBerry PlayBook" splash screen.

Possible solution

- Shut down the simulator. Ensure the 3D accelerator checkbox is checked on the VMware® Player properties screen.
- Reset the simulator from the VMware Player player menu. Click on **Virtual Machine**, >**Power**, >**Reset.**

Using a proxy server from the command line 7

Your computer connects to the RIM® Signing Authority during the signing process. If your computer connects to the Internet through a proxy server, you should append the following options to your blackberry-signer command line to connect to the RIM® Signing Authority through your proxy server.

Option	Description
`-proxyhost` *<host>*	This option specifies the network host that provides the proxy service. The host parameter can be an IP address or a fully qualified domain name.
`-proxyport` *<port>*	This option specifies the port number on your proxy server through which blackberry-signer should communicate with the RIM Signing Authority.
-proxyusername *<user name>*	This option allows you to provide your user name, if your proxy server requires authentication.
-proxypassword *<password>*	This options allows you to provide your password, if your proxy server requires authentication.

Example: blackberry-signer command line

```
blackberry-signer -proxyhost 192.168.1.1 -proxyport 80 -register
                -csjpin <PIN> -cskpass <csk password> <CSJ_file>
```

Glossary

8

AJAX
Asynchronous JavaScript® and XML

API
application programming interface

HTML
Hypertext Markup Language

HTTP
Hypertext Transfer Protocol

HTTPS
Hypertext Transfer Protocol over Secure Sockets Layer

MIME
Multipurpose Internet Mail Extensions

URI
Uniform Resource Identifier

XHTML
Extensible Hypertext Markup Language

XML
Extensible Markup Language

Provide feedback

9

To provide feedback on this deliverable, visit www.blackberry.com/docsfeedback.

Legal notice

10

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 might not offer Internet browsing functionality with a subscription to the 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, and/or BlackBerry® Device Software.

Research In Motion Limited
295 Phillip Street
Waterloo, ON N2L 3W8
Canada

Research In Motion UK Limited
Centrum House
36 Station Road
Egham, Surrey TW20 9LF
United Kingdom

Published in Canada