BlackBerry Browser
Version: 4.6.0

Content Design Guidelines

::: BlackBerry®

SWD-574511-1024045803-001

Contents

Optimizing web sites for the BlackBerry Browser 1

Design principles for BlackBerry Browser web content

When you create or adapt web content for the BlackBerry® Browser, keep in mind the following design principles:

- Design to meet the user's needs.
- Design based on supported standards and supported content types.
- Design for the characteristics and features of the BlackBerry device.
- Design for wireless networks.

Design content for your users

Whether you are designing content to meet some business function or for entertainment, when you create content for the BlackBerry® Browser, your design choices should be governed by the needs of the intended users.

- Focus on the user experience. Where possible, attempt to understand who will use your web site, what their needs and expectations are, and how your web site will meet those needs and expectations.
- Balance the user's need for web pages that load quickly with their desire for feature-rich content. A clear understanding of your target users can inform your decisions about how to present content or develop script behavior, and where to make design compromises.
- Create an appropriate site hierarchy. Structure your web site based on its purpose, and organize the site to minimize the time that it takes users to find information or perform tasks.
- Avoid design practices that distance your content from users. Reduce the number of links a user must follow and strive to reduce the number of steps the user must take to view and work with your content.

  If a web page is designed correctly, users should not have to switch to either Page View or Column View, they should not have to scroll horizontally to view page content, and they should not have to rescale the content to read it.

  For best results, you should:

  - Design page layouts that make effective use of small screens.
  - Use `<meta>` tags to control the initial scale of your content.

Use supported standards and content types

The BlackBerry® Browser and the standards and content types it supports are continually improving and changing. Not all users will have the latest version of the BlackBerry® Device Software, however. As you design your content, you need to make decisions about which users you want to reach, and how best to reach them.

Keep in mind that in BlackBerry Device Software version 4.6, the BlackBerry Browser was redesigned. This redesign resulted in a considerable upgrade in standards support from earlier releases, including full support of HTML 4.01, CSS 2.1, JavaScript® 1.5, and DOM Level 2 (Core, HTML, Styles, and Events).

The difference in standards support between earlier versions of the BlackBerry Device Software and BlackBerry Device Software version 4.6 has certain implications for you as you design content for the BlackBerry Browser.

- A large number of BlackBerry devices are running versions of the BlackBerry Device Software earlier than version 4.6. It is probable that most BlackBerry device users will not have access to a BlackBerry Browser with expanded standards support for several years.
- Earlier versions of the BlackBerry Browser have considerably less support for JavaScript and CSS. If you use these standards extensively, it is probable that you are excluding a considerable portion of BlackBerry users.

It is inevitable that much attention will be given to exploring how best to use this improved web development platform to create richer and more compelling user experiences. However, the decisions you make regarding content should always come back to the users you want or need to reach.

- Know what standards and content types are supported by the BlackBerry Browser.
- Deliver device-appropriate content.
    - Where possible, avoid creating content to suit both BlackBerry devices running BlackBerry Device Software version 4.6 or later and those running earlier versions of the BlackBerry Device Software.

        In earlier versions of the BlackBerry Browser the limited CSS support means that it is more difficult to define presentation using CSS alone. In addition, limited JavaScript support means that web content and application for earlier versions of the BlackBerry Browser are functionally limited.
    - In general, avoid automatically directing BlackBerry devices to mobile web sites. These sites are designed to reach a broad base of wireless devices, and are often too basic in structure and scope for most BlackBerry devices. As a result, these sites do not typically result in a compelling user experience, especially for users with newer BlackBerry devices and the latest BlackBerry Device Software.

        If you are unable to create more device appropriate content, consider providing users with the option to view your desktop-oriented content.

        Alternatively, consider whether directing devices running BlackBerry Device Software version 4.6 or later to your desktop content. If the content is well designed, you may be able to achieve excellent results on these BlackBerry devices.

For more specific information about the standards and content supported by the BlackBerry Browser, refer to the following guides:

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

BlackBerry Browser content support

The following table lists the web formats that the BlackBerry® Browser supports, and the version of the BlackBerry® Device Software in which support was introduced.

Format type	Content type	Component	Partial support in BlackBerry Device Software version	Full support in BlackBerry Device Software version
markup	HTML 4.01		3.2.1	4.6
		tables	3.8	4.6
		image maps	3.8	4.6
		frames	4.2	4.6
	CSS 2.1		3.8	4.6
	SVG		3.7	—
	XHTML-MP		—	3.6
	WML		—	3.2
scripting	DOM L2		4.6	—
		core	—	4.6
		HTML	—	4.6
		style	—	4.6
		events	—	4.6
	AJAX		—	4.6
	JavaScript® 1.5		3.8	4.6
	ECMAScript™ 262 3rd Edition		—	3.8
	WMLScript		3.2.1	—
media	video			
		WMV	—	4.3
		h.263	—	4.6
	audio			
		MP3	—	4.2
		MP4	—	4.2
		WAV	—	4.2
		MIDI	—	3.7
	images			

Format type	Content type	Component	Partial support in BlackBerry Device Software version	Full support in BlackBerry Device Software version
		animated GIF	–	3.8
		TIFF	–	3.7
		BMP	–	3.7
		JPEG	–	3.7
		PNG	–	3.2.1
		GIF	–	3.2.1
web feeds	RSS 0.9, 1.0, 2.0		4.2	–
		RSS enclosures	–	4.3
	Atom™		4.2	–

Design for the characteristics and features of the BlackBerry device

There are a variety of BlackBerry® devices, each with their own unique characteristics and features. Depending on the BlackBerry device on which they are browsing, users might view and interact with your content differently.

Creating content that adapts to different BlackBerry devices

Unless you intend to target a particular BlackBerry® device, your content should be adaptable to all BlackBerry devices. You should create content that supports the multiple screen resolutions and sizes, and the different input and navigation methods available on BlackBerry devices.

- Use CSS styles to control presentation wherever possible. To target specific devices, it is easier to create a single HTML file with multiple CSS style sheets than it is to create multiple HTML files with similar content but different formatting and layout information.
- Test how your content looks on wide and narrow BlackBerry device screens. Keep in mind the following:

  - The BlackBerry® Pearl™ smartphone screen is half the width of the BlackBerry® Bold™ smartphone screen. Designing content that works on both screens may be difficult in some cases.
  - The BlackBerry® Storm™ smartphone can view content in either a portrait orientation (with a screen width of 360 pixels) or a landscape orientation (with a screen width of 480 pixels).

  You should achieve good results on most BlackBerry devices by using style sheets and proportional rather than absolute measures.

If necessary, you might want to consider creating separate style sheets for narrow-screen devices (BlackBerry Pearl), and wide-screen devices (BlackBerry® Curve™, BlackBerry Bold, and BlackBerry Storm). If you properly separate the presentation information from the HTML content, adapting to different screen sizes is simpler.

- Be aware of the different input methods available on BlackBerry devices. Design content that is not dependant on one input method. In most cases, the BlackBerry® Browser translates different user events, such as a trackball click or a touch gesture, to standard mouse or keyboard events. However, you should test your JavaScript® code to be sure that UI events are interpreted as expected on different devices.

Designing around user-configurable settings

Users can configure several settings for the BlackBerry® Browser to control the amount of content that is downloaded and how that content is presented. For example, the user might choose to turn off support for JavaScript® or images. They might also specify how theBlackBerry Browser initially renders content by changing the default view to either Column View or Page View.

These user-configurable settings can have an impact on the way in which the user experiences your content. Although these settings should not guide what content you create or how you implement it, there are some basic design principles you should follow to make sure that the settings the user has specified do not result in unusable content.

- For web pages that rely on JavaScript to load properly, use the `<noscript>` element to provide alternative content. If the user turns off JavaScript support, the `<noscript>` element allows your web page to provide some useful content, such as instructions on how to turn on JavaScript support, rather than just displaying an error.
- Provide descriptive alternate text for images. If the user configures the BlackBerry Browser to prevent images from loading, alternate text can help provide some context.
- Create page layouts that can be viewed in Column View and Page View. Design content to fit the screen width, avoid framesets and tables for page layouts (which do not work well in Column View), and use `<meta>` tags to prevent the BlackBerry Browser from erroneously scaling content in Page View.

Making use of integrated BlackBerry device features

The BlackBerry® Device Software is a tightly integrated suite of applications that includes email, phone, and media applications, as well as integrated attachment viewers. Some devices also have GPS functionality. You can make use of the features of BlackBerry devices to provide a richer user experience.

In some cases, this might mean that you create content that targets a specific BlackBerry device or BlackBerry devices with specific feature sets. While feature-specific content may exclude some BlackBerry device users, for others it can result in compelling and useful content or web applications.

- Create links to automatically open the phone and dial a number. The BlackBerry® Browser supports the following types of phone links:
  - WTAI Make Call links (URI form)

    ```
    <a href="wtai://wp/mc;14165551212">Call office</a>
    ```
  - phone links in I-mode format

```
<a href="tel:14165551212">Call office</a>
```

- Direct Connect links on iDEN® networks

```
<a href="dc:234*234*234">Call office</a>
```

- CTI

```
<a href="cti:333333">Call office</a>
```

- On devices that provide GPS support, consider providing unique location-based solutions. You can use the `BlackberryLocation` object to access GPS functionality in your JavaScript® code and create location-based web applications.
- Consider where you can use media to augment your content's message. The BlackBerry Browser supports a number of video and audio codecs, and supports media streaming.

  Keep in mind the user costs associated with downloading large media files and streaming media over wireless connections. These activities are best-suited to BlackBerry devices that support Wi-Fi®. If you present users with the option to download or stream media over the wireless network, consider providing information about the associated costs.

Design content for the wireless network

For many years, the common practice in designing content for wireless networks has been to simply send less content. However, the ever-expanding capabilities of wireless browsers such as the BlackBerry® Browser seem to encourage you to design more complex and feature-rich content. When creating content for wireless networks, the guiding principle you should follow is: send only what you must, and send it as smartly as possible.

When you create content to be delivered over the wireless network, there are two key concerns to keep in mind:

- Avoid increased data transfer costs: Many users must pay for the data transferred over wireless networks. Minimizing the expense of using your content should be one of your design goals.
- Be aware of reduced network bandwidth: Network latency is one of the common frustrations for content developers and users alike. Wireless networks cannot transfer data as fast as wired connections, so the more content you send, the longer the user must wait longer before the content is downloaded and rendered. With efficient content and smart delivery, you can minimize the impact of network latency.

Following the principles of efficient web content design

The first step to designing content for wireless networks is to follow the best practices of standard web content design.

There are a number of accepted principles of standard web content design that improve network efficiency, produce faster and more adaptable web sites, and result in a better overall user experience for users browsing from desktop computers. For example, you can reduce network traffic by properly defining cache controls, using conditional requests, and creating reusable resources. You can create more adaptable content by separating style from substance. You can reduce content rendering time by using efficient markup. The benefits of good web design for desktop-oriented web content are magnified for wireless browsers.

The negative effects of flawed or inefficient design are similarly magnified. In a desktop environment, the cost of poor design is relatively minor; faster networks and greater processing power can help to overcome, or at least mitigate, flaws web content design, and there is typically little or no financial burden passed on to the user. Wireless browsing environments are much less forgiving. Associated costs to the user are more tangible. When the BlackBerry® Browser download poorly designed web content, it can cost users time, money, and even the battery life of their BlackBerry device.

- Use efficient caching practices. Set an appropriate expiry date, use appropriate cache-control headers, and use the `if-modified-since` and `last-modified` HTTP headers to reduce the amount of data that is transferred over the wireless network.
- Use structural HTML markup to create logical blocks of information in your HTML. For example, use `<div>` elements to denote content blocks. Use heading elements (`<h1>`, `<h2>`, and so on) to define informational hierarchies.
- Use CSS to control web page layout and content presentation.
- Create separate files for content, presentation, and interaction components. External CSS and script files can be cached and reused the next time the user visits the web page, and reused when the user visits other web pages from your web site.

    Inline or internal styles and scripts are much less efficient, since, over time, they result in more content being transferred. Each time the user requests that web page, the style and script content must come with it. Internal styles and scripts can be used only by the containing web page, and may not be reused across other web pages from your site.
- Compress content resources using GZIP. The benefit is reduced if a user browses through the BlackBerry MDS Connection Service or the BlackBerry® Internet Service Browsing network gateway. These network gateways will uncompress the components before they reach the wireless network in order to preprocess them and repackage them more efficiently for transmission. However, when content is accessed over WAP or Wi-Fi® connections, using GZIP will reduce network traffic. Using GZIP can also improve transmission over wired networks.

    Although there are many alternative compression algorithms, GZIP allows the network gateway to optimize content and improve network efficiency, which other compression algorithms might not.

Reducing the amount of data that must be transferred

- Apply minimalism to your content. This does not mean stripping web pages to the bare minimum; include what is necessary based on the purpose and audience of your content, and strike the right balance between keeping file sizes small, reducing additional HTTP requests, and producing usable content. Make web pages only as large as they need to be to meet the needs of the audience and to remain functionally useful. For example, sending one large file requires less overhead than sending several small files. If users must visit each of the small web pages to complete a task, then creating smaller files is a false economy.
- Streamline your HTML, CSS, and JavaScript® files.
    - Remove unnecessary or unused elements, styles, and scripts.
    - Reduce the number of CSS files your site uses. Most web sites require only one style sheet to define presentation for every web page. Although this results in a larger file, it is a one time expense, and overall is more efficient than sending several smaller CSS files

- Reduce the number of JavaScript files your site uses. Typically, scripts which affect the initial rendering of the page should be separated from other scripts so you can defer or backload scripts where possible, and rendering is not blocked unnecessarily. Reducing the number of script files also reduces the likelihood that you will inadvertently deliver duplicate scripts to the browser.
- Optimize your use of images and media. Scale images appropriately to reduce file sizes. Carefully consider what images you need to send, and remove any that are deemed unnecessary.

Minimizing the impact of slower wireless networks

- Design your web site so that users make fewer requests to reach the content they need. Fewer content requests translates into less time spent waiting for content to be transferred and rendered. This requires clear site navigation, clear labels, and a well designed web site hierarchy. By effectively anticipating where a user might want to go from a specific page and providing target links, you can minimize wait times.
- Consider using AJAX scripting techniques to progressively render a page. Progressive page rendering refers to loading web page content in phases. You can reduce wait times for the user by sending a portion of the content to the BlackBerry® device, and then using the `XMLHttpRequest` object to request and receive additional content after the BlackBerry® Browser has rendered the page. These requests occur in the background, so they do not block the user from viewing or working with the page.

  Using AJAX has limitations: AJAX is only supported for BlackBerry devices running BlackBerry® Device Software version 4.6 or later, and users can turn off the browser's JavaScript® support, which is required for AJAX requests to be sent.
- Consider whether pushing content to the BlackBerry device proactively is an option. Content is sent to the BlackBerry Browser cache without a user requesting and waiting for it. For example, you could push a resource to the BlackBerry device so that when the user browses to content that uses that resource, the browser does not need to download it over the wireless network.

  Pushing content is network gateway-dependent. The BlackBerry MDS Connection Service can push content to BlackBerry devices that are associated with a BlackBerry® Enterprise Server. Other push strategies supported by the BlackBerry Browser are push/pull mechanisms, in which the user receives a notification that new content is available on the server so that they can request the content.

Detecting BlackBerry device information

Detecting device information can be a useful practice. There are instances in which it is helpful to have information about the requesting BlackBerry® device and version of the BlackBerry® Device Software it is running.

For example, having information about the BlackBerry device that is making a content request is useful in the following cases:

- if you want to send images that are optimized for particular BlackBerry devices, or for particular BlackBerry device screen sizes
- if you want to send device-specific style sheets to improve presentation on certain BlackBerry devices

- if you want to determine whether the requesting BlackBerry device supports a particular feature (for example, GPS) that is required for your content.

When the BlackBerry® Browser makes an HTTP request, it includes the `Profile` header. This header provides the URL for the location of the User Agent Profile for the BlackBerry device making the request. This URL uses the following form:

```
http://www.blackberry.net/go/mobile/profiles/uaprof/ <BlackBerry-model>/
<software-version>.rdf.
```

where:

- `<BlackBerry-model>` is the BlackBerry device model number, for example, 8220 or 9000.
- `<software-version>` is the BlackBerry Device Software version, for example, 4.3.1 or 4.6.0.

You can create a script to extract the `Profile` header information and let the content server return device-appropriate content in the HTTP response.

Caching BlackBerry Browser web resources efficiently

Effectively caching web resources on the BlackBerry® device can reduce the amount of data that must be transferred over the wireless network. First-time visitors must wait for all the necessary components to be transferred. If components are cached, the next time the user visits your web site, the BlackBerry® Browser can render content more quickly, since the amount of data that must be retrieved from the server is reduced. The BlackBerry Browser respects all caching directives specified in either the content header, or within HTML `<meta>` tags.

The BlackBerry Browser adheres to the following rules:

- If a component is secure or authenticated, the browser does not cache that component.
- If the `cache-control` header for a component has a value of `no-cache`, the browser does not cache that component.
- If an expiry time or an age directive is specified, and it is still valid, the component remains fresh and can be retrieved directly from the cache.
- If an object is no longer fresh, the browser attempts to check with the origin server to determine whether the component is still valid. If the component has not changed since it was cached, the browser considers the component fresh, and will retrieve it from the cache.

Although specifying caching directives correctly is the best thing you can do to reduce data transfer and to speed content rendering, be aware that the BlackBerry device might clear the cache from time to time in order to free up device memory, with expired content cleared first.

Best practices: Caching web resources effectively

- For static components, use the `Expires` header to specify an expiry date that is far into the future to keep the component fresh longer. so that the BlackBerry® Browser does not need to waste time attempting to validate content that is unlikely to change. If you change the component before the expiry date, you cannot force the browser to ignore the expiry date and load the updated version. Rename the component to make sure it is loaded.
- For frequently updated components, specify an appropriate `Expires` value. For example, if you have a web site that provides sports scores, specifying an `Expires` value of 12 hours should be enough to make sure that when new results are available, cached content are expired and the BlackBerry Browser will retrieve the new content.
- Use SSL only for components that must be secure. Secure components are never cached. Downloading components that are not security-sensitive over a secure connection prevents the components from being cached appropriately.
- Change files only when necessary. Remember that each time you post a file, the `Last-Modified` value is updated. An updated value can cause components in the browser cache to become invalid earlier than they should, and the BlackBerry Browser to download those components unnecessarily. By only posting files that have changed, the `Last-Modified` header value can provide an accurate account of a component's validity.
- Use consistent references for external components. If one resource is referenced using multiple URLs, the browser considers the resource to be two different resources, and the cached version is not used.

Making content requests conditional

If you effectively use cache-control headers to define cache behavior for your content, you can use the `if-modified-since` and `last-modified` headers to maximize your network efficiency. When you make requests conditional based upon whether the requested content is new, the BlackBerry® Browser downloads content only when content in the cache is out of date.

Header	Description
If-Modified-Since	This header is used with a GET method to request that the web server transfer content only if it has been modified since the specified date and time. For example: `If-Modified-Since: Fri, 6 May 2005 12:00:00 GMT` The request is managed as follows: - If the requested content has been modified since the specified date, or if the specified date is invalid (such as a date that is later than the server's current time), the content will be downloaded. - If the requested content has not been modified, the server sends a 304 (Not Modified) response, and the browser retrieves the content from the cache.

Header	Description
Last-Modified	This header identifies the date and time when the content was most recently modified. For example: `Last-Modified: Fri, 6 May 2005 12:00:00 GMT` The exact meaning of this header field depends on the implementation of the origin server and the nature of the original resource. • For files, the value of `last-modified` might be the file system last-modified time. • For entities with dynamically included parts, the value of `last-modified` might be the last-modified time of the most recently modified component. • For database gateways, the value of `last-modified` might be the last-update time stamp of the record.

Designing web page content for the BlackBerry Browser   2

Best practices: Designing information for the BlackBerry Browser

- Design content so that users can quickly find the content they require.
    - Practice minimalism to avoid sending unnecessary content. Web pages should include only what is necessary for the user to complete the intended task. Remove content that does not further the purpose of the web page.
    - Place important information at the top of the page where it is most quickly accessible.
    - Reduce the vertical space occupied by top-of-page header components, such as banner images and navigation bars. On BlackBerry® devices with landscape-oriented screens, large headers can take up valuable screen space.
    - Label links carefully, and clearly identify the target of each link, to help users decide which links they want to follow.
- Consider the length of your web pages carefully. The amount of content on a web page has several implications. Sending more content can reduce the number of requests the user makes, but it can also increase the time it takes to download and render content and the amount of scrolling users must do. Sending less content can result in quicker download and rendering times and more focused content, but it can increase the number of HTTP requests the user makes to complete their task.

    It is typically better to provide slightly more content on a web page and reduce the number of HTTP requests for which the user must wait.

    To improve download and rendering speeds, you can use JavaScript® and AJAX requests to progressively render the page. Using AJAX requests can result in an improved user experience, since perceived wait times are reduced. However, to benefit from AJAX, the user must have JavaScript support turned on.
- Consider using dynamic HTML effects to show and hide content dynamically. Using dynamic HTML can allow you to place more information on the page without using up extra screen space. For example, key headings could be clustered at the top of the page. The user can then click a heading to expand a topic of interest to read more. Use dynamic HTML judiciously, however; keep in mind that for your dynamic HTML to be functional, the user must have JavaScript support turned on.

Best practices: Using fonts effectively in your web content

- Modify fonts judiciously. Use the user-defined default font for the BlackBerry® Browser where possible, unless you have a specific need to select a different font. Although using the user-defined default font makes it more difficult to control layout and appearance, it avoids issues of choosing a font face or font size that the user finds hard to read. Users will either change the default BlackBerry Browser font to a suitable font face or and font size, or accept the theme-specified default font.
- Avoid absolute font sizes where possible. If you change the font size, use relative sizes, such as larger, smaller, xx-large, and so on. In standard text flows, such as a series of paragraphs, changing the font size is unnecessary, since text will wrap. Keep in mind that the user can specify the smallest font size for the BlackBerry Browser, so relative or absolute font sizes that are less than the specified smallest font size will produce no effect on a user's device.
- Verify that the specified font is legible on BlackBerry devices.

- Use CSS to define fonts and font style information. Avoid using the `<font>` or `<basefont>` elements to define fonts for your content. These elements have been deprecated. Avoid using HTML elements such as `<bold>` or `<italic>`, to define fonts. Instead, use semantic elements that suggest the type of information that is enclosed, for example `<em>`, then set a style for that element using CSS.
- Use bold fonts for headings and button text. Consider carefully the need for bold fonts elsewhere in your content. Excessive use of bold fonts for emphasis can be distracting and impair the user's ability to read your content. Keep in mind that although the BlackBerry Browser supports all values for the CSS font-weight property, it translates specified values into one of two font weights: normal and bold.
- Use italics and underlined text judiciously. These font styles can make text hard to read.
- Use colored text carefully. Consider the legibility of text when choosing colors. Text becomes illegible when it appears on a background of similar color or brightness.

Designing web page layout

With extensive support of web standards such as HTML 4.01, CSS 2.1, JavaScript® 1.5, and DOM Level 2, the BlackBerry® Browser can capably render and display most content. Most content, however, is not optimized for viewing on BlackBerry devices.

The web page layouts that you use desktop-oriented web content are typically too complex for small screens. Many web pages use two-column or three-column layouts, with the principle content placed in one column, and other columns typically devoted to site navigation, advertisements, related links, and other non-essential elements.

Most standard web content is designed at widths of between 900 and 1100 pixels, while most BlackBerry devices have screens between 240 and 480 pixels wide. You need to decide what content is essential for the purposes of the web page.

Typically, there are three or four structural components common to every web page.

- A page header. The header typically contains branding information such as a logo or banner image.
- A navigation structure. The navigation is often in a side bar, or in a navigation bar located below the header.
- The page content.
- A page footer. Footers often include secondary navigation links or contact information.

Because of small screen width on BlackBerry devices, when you create content for the BlackBerry Browser, stacking the structural components vertically is the most effective layout.

Best practices: Designing web page layouts for BlackBerry Browser content

- Keep web page layouts simple. Avoid complex, multi-column layouts which do not present well on small screens. Instead, stack structural components vertically.
- Use CSS to define web page layouts, where possible. Using CSS allows you flexibility in terms of altering presentation to suit different browsers.

- Avoid using tables to lay out content, where possible. Although tables can be used to define the layout of basic page components, using tables hard codes the layout and reduces your ability to adapt content to different screen sizes. Keep in mind that in BlackBerry® Device Software version 4.6 and later, when displaying content in Column View, the BlackBerry® Browser ignores table layout and places one cell in each row.
- Design web page components that adapt to different screen widths. Avoid using absolute measures to specify element widths. Using proportional measures allows the BlackBerry Browser to size elements proportionally to the screen width.

Controlling initial web page scaling

On BlackBerry® devices with BlackBerry® Device Software version 4.6 or later, by default, the BlackBerry® Browser rendered desktop-oriented content as it would be displayed on a desktop-computer, then scales the content to fit the BlackBerry device screen. For most desktop-oriented content, the BlackBerry Browser renders it at 20% to 40% of its original size.

Content that is designed for the BlackBerry Browser should be presented to the user unscaled. You must inform the BlackBerry Browser that content is designed for the BlackBerry device screen by adding a `<meta>` tag to control the initial scale of your web page. Otherwise, the BlackBerry Browser will scale your content in error.

The BlackBerry Browser supports the following methods to control the initial display scale with the `<meta>` tag:

- `HandheldFriendly`: The `HandheldFriendly` processing instruction informs the BlackBerry Browser that the content contained within the document is designed for small screens.
- `viewport`: The `viewport` definition allows you to specify the specific dimensions of the intended viewport.

Although there are differences in the implementation of each method, you can achieve the same result using either method.

Controlling initial scale using the HandheldFriendly processing instruction

The `HandheldFriendly` processing instruction informs the BlackBerry® Browser that the content is optimized for small screens.

When content defined as `HandheldFriendly` is rendered in Page View, the BlackBerry Browser renders it on a canvas equal to the size of the BlackBerry device screen.

The `HandheldFriendly` processing instruction is defined in a `<meta>` element within the `<head>` section of an HTML document.

```
<meta name="HandheldFriendly" content="true" />
```

Controlling initial scale using the viewport definition

The `viewport` definition is used to specify the dimensions of the viewport in which the content is designed to be displayed. The BlackBerry® Browser renders the content on a canvas that is equal to the specified dimensions, then scales it to fit the BlackBerry device screen.

The BlackBerry Browser supports the `height` and `width` properties for this tag. You can specify absolute values for these properties, or use the relative `device-height` and `device-width` constants. For best results, you should specify `device-width` as the value of the `width` property. Specifying the height is typically unnecessary.

The `viewport` definition is specified in a `<meta>` element within the `<head>` section of an HTML document.

```
<meta name="viewport" content="width=device-width" />
```

Designing site navigation for the BlackBerry Browser

Effective navigation is vital for BlackBerry® Browser content. Users need to quickly find the content they need, and with as few wrong turns as possible. Following links erroneously in a wireless environment is costly, both in potential data charges, and in the time wasted retrieving and rendering an unwanted web page.

However, navigation structures used in desktop-oriented web content are not appropriate for smaller screens.

You can do a lot to improve site navigation as you plan your web site. Begin by creating an effective site hierarchy. A consistent hierarchy of web pages helps to make your site more predictable, and as a result, easier to navigate. Your web pages should be organized based on their purpose to minimize the number of clicks—and as a result, the amount of time—that it takes users to find information or perform tasks. In addition, provide a short, descriptive title for each web page. The BlackBerry Browser lists web pages in the History list by their title. Clear titles can help users retrieve previously visited pages from the History list.

As you create content, within each web page, reduce the primary navigation links to the bare minimum. Commonly used sidebar navigation is not appropriate for display on most BlackBerry device screens. Instead, a simple horizontal navigation bar with a few targeted links in your content header work best. If necessary, consider adding a footer to your content to containing additional navigational links.

In all cases, label links carefully. Clearly identify the target of each link to help users to decide which links they want to follow, and to avoid having them follow links in error.

Using JavaScript®, it is possible to create dynamic effects such as drop-down menus to provide additional navigation without using additional screen space. However, dynamic navigation structures are not considered a good web design practice, and are not effective for BlackBerry Browser content that must be accessible by all BlackBerry device users. Consider the following implications:

- Dynamic content relies on JavaScript support being turned on. Navigation should not rely on JavaScript support. If you create dynamic navigation structures, make sure that users can still access at least top level links if JavaScript support is turned off.
- Dynamic content is only supported in BlackBerry® Device Software version 4.6 and later. If you create dynamic navigation structures, users with BlackBerry devices running earlier versions of the BlackBerry Device Software will not be able to access links.

Best practices: Designing your web site navigation

- Create an effective site hierarchy. If web pages are organized well, users can intuitively find the information they need.
- Provide a short, descriptive title for the page. The BlackBerry® Browser displays the title in a title bar above the content, and uses the title to identify the web page in the History and Go To lists. From the page title, the user should be able to clearly identify the content and purpose of each web page.
- Provide consistent navigation aids. However you choose to present your navigation aids, you should keep them consistent across your web site. Inconsistent navigation aids force users to spend time trying to understand navigation behavior and can cause the user to follow links in error. Consistent navigation structures give your web pages a predictability that helps users navigate through your web site.
- Consider providing primary navigation links in a horizontal navigation bar. Links should be displayed on a single line at the top of the page.
- Simplify your site navigation. Reduce the number of links on a web page to those that are most relevant.
- Include a link to the Home page at the top of every page.
- Consider adding a Search field at the top of the page to allow users to search your web site for content.
- If necessary, include secondary navigation links at the bottom of the page.
- Avoid sidebar navigation. Additional right or left columns are inappropriate for small screens.
- Avoid displaying breadcrumbs at the top of the screen. Although breadcrumbs can help users to orient themselves within your web site and quickly return to previously viewed pages, on small screens, breadcrumbs occupy too much screen space. Look for alternative ways to present this information to the user.
- Be cautious about using dynamic HTML to create drop-down lists for navigation bars. Although you can use these techniques to provide more navigation from the navigation bar without using more screen space, they rely on the user having JavaScript® support turned on. Ideally, navigation aids should not rely on user configurable settings.

  Also, be aware that the BlackBerry Browser does not support dynamic pseudo-classes such as `:hover`.

Using tables in your BlackBerry Browser content

In BlackBerry® Device Software version 4.6 or later, the BlackBerry® Browser provides full support for tables. However, tables should be used sparingly in content designed for the BlackBerry Browser. Although tables are effective in providing structure to content, tables can often result in presentation problems on small screens, since they often require more space than is available.

Unless the table is small, the entire table might not be viewable unless the user scales the content to fit, in which case it may not be legible, or scrolls horizontally or vertically. The result is that the scannability of data, one of the main benefits of tables, is diminished.

It is possible to improve table presentation in content designed for BlackBerry devices by designing tables to fit the width of the screen. If you decide you need tables, you should be aware of the implications of table presentation in the BlackBerry Browser.

- Screen widths among different BlackBerry devices can vary, from between 240 to 480 pixels. Tabular configurations that work on wide screens may present poorly on narrow screens.
- On BlackBerry devices with BlackBerry Device Software version 4.6 or later, if users view content in Column View, the BlackBerry Browser ignores table layout, and instead renders tables vertically with each cell on its own line. As a result, if you include tables, you may be requiring users to change their content view.

  You can probably expect that most users will view content primarily in Page View, and therefore accept that a small number of users will need to change the content view to view table data as intended.

Best practices: Using tables

- Minimize the use of tables in your content. Where possible, look for alternative ways to present table data. For example, you might be able to effectively present tabular data in definition lists or some other list format.
- Use tables for data presentation only. Avoid using tables to define layout of pages or for page structures such as forms. Use CSS for layout, instead of tables, wherever possible.
- Design tables that do not exceed the screen width. To allow the browser to resize the table to fit different screen sizes, use proportional measures, rather than specifying fixed widths.
- Design tables to use the simplest layout possible. If tables are overly complex, consider whether the information can be split into separate tables. Avoid nested tables.
- Use CSS styles to visually define table structure and improve the scannability of table data. Clearly differentiate header information from data.
- Make tables stylistically minimal. On small screens, excessive styling can cause visual clutter and can make table data harder to scan.

  Minimize the use of borders. Heavy borders can hinder legibility rather than improve it. Defining table structure using background shading and color is typically more effective.
- Consider collapsing borders. By default, table borders are defined with a value of `separate`, meaning each cell has its own borders, rather than sharing borders with adjacent cells. Collapsed borders remove unnecessary spacing between cells and present a more streamlined appearance. To collapse cells, apply the following style in your style sheet:

```
border-collapse: collapse;
```

Designing forms for the BlackBerry Browser

The BlackBerry® Browser is well-suited to collecting information from the user using forms. However, the smaller screen sizes and wireless connections of BlackBerry devices mean that the layout and information design of forms takes on greater importance. Excessive screen clutter, scrolling, or wait times can all impair the user's ability to complete your form and can result in a poor user experience.

In general, well-structured, minimalist layouts work best for forms on BlackBerry devices. Text input fields, selection lists and other form controls should be aligned rather than positioned beside the label to create a consistent and clean presentation. Forms with proper alignment are easier to read and complete than those with ragged layouts.

To reduce the amount of vertical space required, use a two-column layout, with labels aligned to the right or left in one column, and form controls left aligned in the other column. If a narrow screen width prevents a manageable two-column layout, a single-column layout is a good alternative, with labels above the form control and all form content left aligned. Single-column layouts require more vertical space, and therefore more scrolling by the user, but the ordered structure of single-column layouts makes them a better choice than ragged layouts in which form controls are not aligned.

Consider the following examples:

Layout	Description
two-column right-aligned labels	Advantages: • Two-column layouts use less vertical space. • Right-aligned labels are clearly associated with the corresponding form controls. • The proximity of the labels makes it easy to see the relationship between grouped elements. Disadvantages: • Right-aligned labels are less easily scanned. • Right-aligned labels require more horizontal space. • Limited label space means labels must be brief.
two-column left-aligned labels	Advantages: • Two-column layouts use less vertical space. • Left-aligned labels are easily scanned. • The proximity of the labels makes it easy to see the relationship between grouped elements. Disadvantages:

Layout	Description
	• Left-aligned labels are less clearly associated with the corresponding form controls.
	• Right-aligned labels require more horizontal space.
	• Limited label space means labels must be brief.
one-column labels aligned above the form control	Advantages: • Single-column layouts can be processed by the user quickly. • Labels are clearly associated with the corresponding form controls. • Extra horizontal space for labels makes the form more flexible for complex or localized labels. Disadvantages: • Single-column layouts require more vertical space. As a result, fewer form controls are viewable on the screen, and more scrolling is necessary. • Top-aligned labels make it more difficult to identify relationships between grouped elements.

Best practices: Creating usable form designs

Form layouts should be simple, structured, and flexible.

- Use a two-column layout where horizontal space permits, with labels in the left column and form controls in the right column. This layouts uses less vertical space and is generally better suited to BlackBerry® device screens.
- Use a single-column, vertical layout where the screen width prevents a usable two-column layout.
- Use CSS to control form layout, rather than using tables. Because of the variation in screen width on different BlackBerry devices, you might want to consider presenting device-specific layouts for narrow and wide screens. Using CSS provides greater flexibility in presentation.
- Left-align form controls. Aligning form controls, such as text fields or list boxes, improves the readability of the form and helps to reduce visual clutter.
- Space form controls appropriately. Provide enough space between form controls so that a user can easily select a control using the device pointer or a finger.
- Group related form elements together. Organizing elements into groups provides a visual and logical structure to form content, which allows the user to process form content more quickly.
- Minimize the visual impact of your form. Reduce the number of visual elements you use to provide organization. For example, avoid using borders around groups of form elements, where a single horizontal rule separating groups is sufficient. Similarly, if you use a background color to group content, keep it subtle.

Adding form controls

Best practices: Adding text input fields

You can create text input fields using the `<input type="text">`, `<input type="password">`, or `<textarea>` elements.

- Provide fields that are sized appropriately for the data they are intended to accept. The size of a field provides a visual cue to the user about the input required. For example, if the user is required to enter a ZIP code, create a field that is 5 characters wide.
- Where the length of text input is unknown, use a consistent field width. Any `<textarea>` elements should conform to this width.
- Define a border for all text input fields. Keep in mind that users must click the field to activate it and enter text. Borders clearly define the clickable area for the user.
- Provide a default value for text input fields, where possible. Default text provides a cognitive cue to the user about the input required. For example, you can use a default value to provide some hint text, provide a brief description of the kind of information the field is intended to collect, or assist with field completion (such as providing HTTP:// in a field intended to collect a URL).
- Define an input mask for text input fields, where possible. The BlackBerry® Browser supports the `-wap-input-format` WAP CSS extension. This CSS style allows you to define the kind of input the field can accept. Defining input masks also provides field level input validation without requiring scripts.

Best practices: Adding labels to form controls

- Enclose text for form control labels in the `<label>` element, and define the `for` attribute to associate the label and the corresponding form control. This represents well-structured, accessible HTML, which offers more versatility when defining CSS presentation properties and can be used with screen reader technology.
- Provide clear, concise label text.
- Use title case capitalization for labels.
- Punctuate labels for form controls using a colon (:).
- Identify optional or required form controls. If your form contains a mix of optional and required form fields, you need to communicate this to the user.

  - Indicate whichever is least prevalent, the required or optional elements. That is, if you have fewer required fields than optional fields, indicate the required fields. Where required and optional fields are relatively evenly split, indicate the required fields. Identifying the smaller number of fields helps to reduce visual clutter.
  - Use text or a symbol such as an asterisk (*) as an indicator. Associate the indicator with the label that corresponds to the form control. Text provides greater clarity. However, to save space, symbols are preferable on BlackBerry® devices.
  - Consider using color to highlight special-case indicators.

- Provide a legend to describe what the indicator symbol denotes. If you use a color symbol, use the same color for the legend text to visually associate the legend and the symbol.

Best practices: Adding buttons

You can create buttons using the `<input type="submit">`, `<input type="reset">`, or `<button>` elements. Buttons allow users to perform ane action such as submit the form, move to a new screen, or save form data.

- Add buttons judiciously. Typically, each page should have a Submit button or a button that moves users forward through the completion process, and a Save button. You can add a button to allow users to move back to previous screens, or a Reset or Cancel button. Before you add other buttons, consider whether they are necessary and will be helpful to the user.
- Present the buttons for secondary or destructive actions, such as the cancel function, in a manner that is visually distinct from, and less conspicuous than, buttons for primary actions, such as submit or save function. By creating a visual differentiation, you draw the user's attention away from negative actions and allow primary action buttons to have greater focus.
- Include a Save button on every form. Adding a Save button helps the user save unsubmitted or partially completed form data if they cannot complete the form in one continuous block of time.
- Use one-word labels for buttons, where possible. A descriptive verb that identifies the action the button performs is ideal. For example, Save, Submit, Cancel, Reset.
- Limit the use of custom images for buttons. In general, text labels are more reliable than images. Images work best when they are universally understood symbols. For example, in a long form, you could use arrow images for buttons which allow the user to navigate back and forth through form screens.
- Position buttons appropriately based on their importance. Most web design theorists suggest that the primary action button, typically Submit or Continue, should be aligned at the left edge of the form control.

  Aligning the primary button with other form controls creates a visual path from the top of the form to the point of completion.
- Avoid using punctuation with buttons.

Creating wireless-friendly forms

Although forms generally work very well in the BlackBerry® Browser, there are several characteristics of the BlackBerry browsing environment to consider as you develop content.

A loss of network coverage can be a problem for users trying to complete and submit forms. Unless the browser session is kept open, input can be lost. On wireless devices, loss of network connectivity can occur more frequently than on desktop computers. BlackBerry devices can lose network connectivity for a number of reasons, for example, if battery power is too low, or the user moves into an area without wireless network coverage.

With slow network speeds, there are design choices you can make to minimize the impact of the network, particularly when lengthy forms are required. If users complete a form page and then must request additional form content from the server, the time it takes to complete and submit the form increases, and can result in a poor user experience.

Consider the following guidelines to mitigate the impact of the wireless network.

- Validate form input on the device, where possible, to prevent wasting time and bandwidth transferring data that is invalid or unusable.
- Define form submission queues. To reduce the impact of loss of wireless network connectivity, set up form submission queues to allow the BlackBerry Browser to queue a form submission request, and submit the form data when the device regains a network connection.
- Provide a Save button. Allow users to save the form input if they are unable to complete it at once.
- Reduce the number of HTTP requests the user must make and for which they must wait for responses. Look for ways to download and display form content that minimize wait times. For example, you could use AJAX requests to retrieve content in the background and progressively display it as it is received. Multiple HTTP requests are still required, but because data is retrieved as the user completes the form, wait times can be reduced.

Creating form submission queues

To reduce the impact of loss of wireless network connectivity, the BlackBerry® Browser allows you to set up form submission queues. By configuring form submission queues, users can still complete and submit several forms (possibly for different queues), even if their BlackBerry device is outside of a wireless coverage area.

When the user attempts to submit a form without being connected to the wireless network, the BlackBerry Browser queues the submit requests until the network connection is re-established. As soon as the BlackBerry device returns to a wireless coverage area, the browser submits its queued forms in the order in which they were queued, without any user intervention. After the queued forms are submitted, user responses are stored by the browser in the same queue list. Users can open the queue list, and click the form queue to view the response. The BlackBerry Browser only queues submitted form data when the BlackBerry device is outside a wireless coverage area.

You can create form submission queues in the following ways.

- Create an HTTP property file: Creating an HTTP property file in which you define the queuing parameters lets you create and manage multiple form queues in a single location. You must set your web server to send the headers when the web page that contains the form is requested. You can add queuing parameters for additional forms within the same header file.

  For example, the following Apache Tomcat™ configuration file defines a queue for a form on a web page named mypage.xhtml:

  ```
  <Files mypage.xhtml>

  Header set cache-control max-age=2592000

  Header set x-rim-queue-id Register

  Header set x-rim-request-title "Stock Monitor"

  Header set x-rim-next-target success.xhtml

  </Files>
  ```

- Add the queuing parameters directly to the HTML content: In HTML or XHTML, queuing parameters are added using hidden `<input>` elements.

```
<input type="hidden" name="x-rim-queue-id" value="Register" />
```

```
<input type="hidden" name="x-rim-request-title" value="Stock Monitor" />
```

Form submission queue headers

Use the following headers to define form submission queues.

Parameter	Required	Value	Description
x-rim-queue-id	yes	*text_string*	This parameter specifies the offline form queue to which the BlackBerry® Browser sends any GET or POST requests from the web page when the BlackBerry device is outside a wireless coverage area.
x-rim-next-target	no	*text_string*	This parameter specifies the next web page to send after sending any GET or POST requests to the form queue.
x-rim-request-title	no	*text_string*	This parameter specifies the label used to identify this request in the Queue view page. By default, the title of page is used to identify the request.
x-rim-request-id	no	True|False	This parameter specifies whether the browser will generate a unique ID and add the ID as an HTTP header that is sent with each GET or POST request that results from this web page. By default, the value is True.
x-rim-request-date	no	True|False	This parameter specifies whether the browser will generate a time stamp and add it as an HTTP header for each offline request resulting from the page. By default, the value is True.

Creating and using images

Images are often used in web content with no meaningful effect. In wireless web content, reduced bandwidth of wireless networks means that you must use images judiciously and optimize images for wireless browsing or risk creating a poor user experience. For example, large images require considerable bandwidth, can render slowly, and can result in higher costs to the user.

When a BlackBerry® device requests a web page through the BlackBerry MDS Connection Service or the BlackBerry® Internet Service Browsing network gateways, images are automatically optimized for rendering on the BlackBerry device. The network gateways compress images and deliver them in segments to the BlackBerry® Browser. On the BlackBerry device, the browser renders images progressively, so that content flow can be rendered before the entire image is downloaded.

Not all BlackBerry users will access your content through these network gateways, however. As a result, it is essential that you optimize images before you send them to BlackBerry devices.

You might decide that want to detect the BlackBerry device model that is requesting your content and send images that are optimized for the screen dimensions of the BlackBerry device. Sending device-appropriate images reduces the volume of data that is transferred to the BlackBerry device and allows you to make sure images are used to their best effect for each screen size.

Best practices: Using images

- Use images judiciously. Make sure each image you add to your web page adds value and meaning.
- Scale images appropriately for the screen size before sending them to BlackBerry® devices. Because of the variance in screen width among different BlackBerry device models, sending images optimized for a particular screen size is ideal, where possible.
- Crop images to provide a meaningful impact on small screens. Details visible when an image is displayed on a desktop computer can often be lost when that content is scaled for display on a BlackBerry device.

  By cropping images to remove unnecessary detail while retaining meaningful information, you increase the image's value to the user.
- Avoid using text in images. Text in images can become illegible if the image is scaled, and text that is embedded as part of the image cannot be localized.
- Use the `alt` attribute to provide context when users do not download images, or for users who are sight-impaired. If an image is purely decorative, and there is no need for alternative text, provide a null value ("").
- Explicitly specify the dimensions of the image. When content is accessed over WAP or Wi-Fi® connections, specifying image dimensions allows the BlackBerry® Browser to determine how text will flow around the image without requiring it to load each image first. When content is accessed through the BlackBerry MDS Connection Service or BlackBerry® Internet Service Browsing network gateways, specifying image dimensions is typically unnecessary, since those network gateways can specify image dimensions for the BlackBerry Browser while preprocessing the content.

  Image dimensions can be specified in your style sheet, or you can use the `height` and `width` attributes of the `<img>` tag.
- Beware of using icons to represent navigational links. You may be tempted to provide icon-based navigation to reduce the space required for links. However, if the user cannot easily identify what the icon represents, icons can reduce clarity. Providing clear text-based links is typically a safer option.

  If you decide to use icons, limit their use to easily identifiable images. Consider using icons similar to those used in popular web content, desktop applications, or operating systems, with which users are likely to be familiar.

Best practices: Using JavaScript with BlackBerry Browser content

Best practice	Description
Make JavaScript® files external.	Linking to external JavaScript files requires an additional HTTP request, which might initially increase the time to retrieve and load the web page. However, because external JavaScript files are cached, over time, linking to external JavaScript files can result in time savings. Inline scripts must be loaded every time the page is retrieved, whereas external script files result in time savings, since if the script file is cached this front-end cost is outweighed by the savings experienced
	To reduce the impact of first-time visits, you can inline a subset of JavaScript on the first page, and to load the JavaScript after the page is loaded. Load time is reduced for the Home page, but other web pages can reference the cached script resources.
Use the `<noscript>` element.	JavaScript support on the BlackBerry® device can be turned on or off by the user or by an IT policy.
	You can use the `<noscript>` element to make sure that your content is still usable when JavaScript support is turned off. The `<noscript>` element defines an alternate block of HTML content to be executed if the BlackBerry® Browser is unable to run the script.
Avoid blocking page rendering unnecessarily.	When the BlackBerry Browser parses a web page, the browser processes JavaScript as it is encountered and page rendering is blocked. The browser must first determine if the script affects the initial page content. If the script does not effect the initial page content, then blocking page rendering is an unnecessary delay.
	Consider the following guidelines to avoid unnecessarily blocking page rendering.
	• Separate the scripts that affect page content from those that do not.
	• Load scripts at the end of the page, rather than from within the `<head>` element, or use the `defer` attribute for scripts that do not affect page content.
	```<script defer SRC="http://www.your_company.com/ js/my_script.js">```

Best practice	Description
Avoid using `eval()` functions to execute JavaScript code.	Using `eval()` functions to execute JavaScript code can negatively affect performance. The BlackBerry MDS Connection Service network gateway and the BlackBerry® Internet Service Browsing network gateway precompile and compress scripts before sending them to the BlackBerry Browser. The BlackBerry Browser does not compile script as quickly as the network gateways. Precompiling the script can reduce the time it takes to execute the script. The network gateways do not evaluate the content of the JavaScript file. As a result, the contents of `eval()` method is treated as a string. If the method contains JavaScript code, the code is passed to the BlackBerry Browser uncompiled, and the browser must compile the code, increasing the time it takes to execute the script.
Avoid calling additional resources from within scripts.	Unless the requests are asynchronous, requesting additional resources from within a script can cause delays for the user. Typically, the BlackBerry MDS Connection Service network gateway and the BlackBerry Internet Service Browsing network gateway speed up resource retrieval by requesting resources on behalf of the BlackBerry Browser as they process the content. However, the network gateways do not evaluate the contents of scripts; they merely precompile the code and then send it to the BlackBerry Browser. If you request a resource such as another JavaScript file from within a script, the BlackBerry Browser must make an HTTP request over the wireless network to retrieve the resource, and pause the execution of the script until the resource is retrieved.