

Series 80 Developer Platform 2.0: Designing XHTML/HTML Content

Version 1.1; July 5, 2004

Browsing

NOKIA

Copyright © 2004 Nokia Corporation. All rights reserved.

Nokia and Nokia Connecting People are registered trademarks of Nokia Corporation. Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. Other product and company names mentioned herein may be trademarks or trade names of their respective owners.

Disclaimer

The information in this document is provided “as is,” with no warranties whatsoever, including any warranty of merchantability, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. Furthermore, information provided in this document is preliminary, and may be changed substantially prior to final release. This document is provided for informational purposes only.

Nokia Corporation disclaims all liability, including liability for infringement of any proprietary rights, relating to implementation of information presented in this document. Nokia Corporation does not warrant or represent that such use will not infringe such rights.

Nokia Corporation retains the right to make changes to this specification at any time, without notice.

License

A license is hereby granted to download and print a copy of this specification for personal use only. No other license to any other intellectual property rights is granted herein.

Contents

1	Introduction	5
1.1	Overview of Series 80 Developer Platform 2.0	5
1.2	Conformance to Web Standards.....	5
2	General Guidelines	6
2.1	Display Characteristics	6
2.2	Rocker Manipulated Pointer	6
2.3	Fit-to-Screen Rendering	7
2.4	Font Support	7
2.5	Character Sets and Character Encodings.....	7
2.6	Cache	7
2.7	Network Access.....	7
2.8	Plug-Ins.....	7
3	Support for HTML and XHTML	8
3.1	Supported Image Types	8
3.2	Frames.....	8
3.3	Tables	8
3.4	Tabindex.....	8
3.5	File Upload	8
3.6	File Download.....	9
3.7	mailto:, sms:, fax:URL, tel:.....	9
3.8	Access Keys	10
4	Support for CSS	10
5	Support for HTTP	10
6	Customizing Content	11
6.1	Validating Content	11
6.2	Identifying the Client	11
6.3	Identifying the User.....	11
6.3.1	Cookies.....	11
6.3.2	HTTP authentication	11
7	Terms and Abbreviations	12
8	References	14

Change History

April 8, 2004	Version 1.0	Initial document release.
July 5, 2004	Version 1.1	Chapter 3.8 updated.

1 Introduction

Content developers have tremendous opportunities to reach new clients through the wireless Web. This document provides information needed to make good design decisions when developing Web content for Series 80 Developer Platform 2.0. It is intended to serve as a guide for Web content developers who want to achieve the best possible usability and performance for their Web services in the Series 80 browser. It describes the hardware and software features available on Series 80 Developer Platform 2.0, and presents the features and standards that are supported and not supported in the browser.

1.1 Overview of Series 80 Developer Platform 2.0

Developer platforms provide a consistent implementation of the leading mobile application and content technologies across a diverse range of mobile phones. The Series 80 Developer Platform is a platform for high-end media, games, enterprise applications, and content based on Symbian OS.

Developer platforms offer developers an unrivaled opportunity to leverage their applications and content across a market-leading range of phones while providing consumers with phones that are tightly focused on specific needs such as music, messaging, game playing, or enterprise applications. The Series 80 Developer Platform is the latest extension of the developer platform line, providing developers with a best-of-breed platform for delivering applications and content where a superlative graphics display combined with intuitive user interaction are paramount.

The Nokia 9500 Communicator is the first device that is compatible with Series 80 Developer Platform 2.0.

1.2 Conformance to Web Standards

The Series 80 Developer Platform 2.0 browser supports the following Web standards:

- HTTP 1.0 and 1.1
- HTML 4.01 (with minor exceptions)
- XHTML 1.0 and 1.1
- XHTML Basic
- XHTML Mobile Profile (MP) 1.0
- WML 1.3
- XML
- CSS 1.0 and 2.0 (with minor exceptions)
- CSS Mobile Profile and WAP CSS
- ECMAScript ECMA-262 version 2 and version 3 (with minor exceptions)
- JavaScript™ 1.3 and JavaScript 1.5 (with minor exceptions)

Exceptions for the supported features can be found at <http://www.opera.com/docs/specs/opera6/>.

2 General Guidelines

2.1 Display Characteristics

The UI style for Series 80 Developer Platform 2.0 includes two displays supporting 65,536 colors. The display on the cover is 128 x 128 pixels and on the PDA side 640 x 200 pixels. This document focuses on the content development for the PDA side display.

The screen layout contains three basic areas: Indicator Area, Application Area, and Command Button Area (CBA) (see Figure 1). The Indicator Area on the left is reserved for information about the active application, time, and status of the communication. This area is either 92 or 32 pixels wide, depending on the active application. In the Web, the Indicator Area is always the thinner one: 32 x 200 pixels. The Application Area in the middle is reserved for application use. The CBA on the right consists of four commands, each in a dedicated button. Commands within a view/dialog may change depending on the context of use. It is possible to have multiple windows (a maximum of ten) open at the same time for Web content viewing. Web page content can be printed from the menu.



Figure 1: Screen display areas on the Web

Web pages can also be viewed in full-screen mode. This hides the Indicator Area and CBA so that the Application Area is expanded to fill the available space. In full-screen mode, the display area for Web content is 640 x 200 pixels.

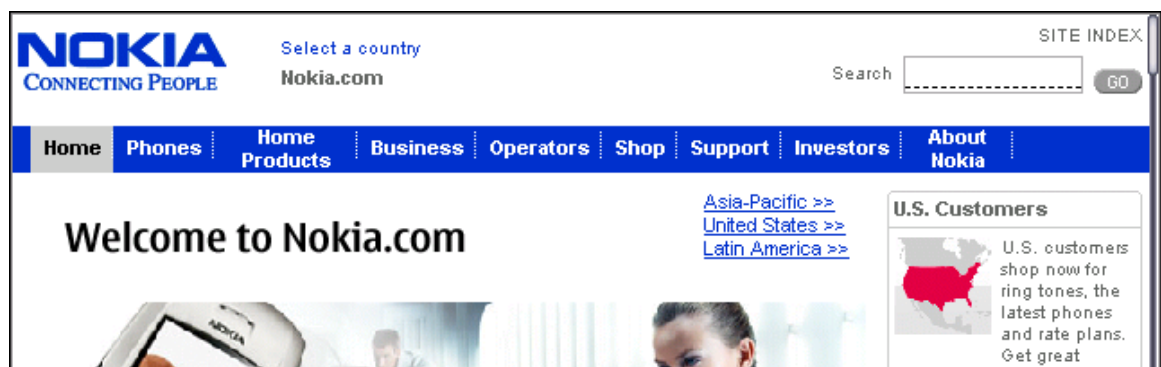


Figure 2: Full-screen mode on the Web

2.2 Rocker Manipulated Pointer

To make Web page usage and such actions as pointing and selecting items more convenient, it is possible to use the Rocker Manipulated Pointer (RMP). The RMP is similar to a mouse pointer except that it is manipulated via keyboard rocker keys instead of a separate mouse device (hardware). The

RMP does not replace the conventional use of a CBA device; rather, it is a user interface extension for using the device. RMP mode is activated/deactivated in the menu or via the shortcut Ctrl + Q.

2.3 Fit-to-Screen Rendering

Most Web sites are written for, and tested exclusively on, desktop computers with large color monitors. Mobile wireless devices typically have much smaller screens, and until recently it was a challenge to present Web pages on these smaller screens. The Series 80 browser's fit-to-screen rendering allows Web sites to fit inside the screen width, thereby eliminating the need for horizontal scrolling. Fit-to-screen mode can be turned on/off by the user, and Web pages can be viewed in normal mode. It is recommended to test Web content on both rendering modes to ensure that the content is usable despite the user's setting.

Fit-to-screen rendering is off by default. The mode is also automatically disabled if the page content is detected by DTD to be XHTML MP or XHTML, or if the page contains CSS definitions "CSS media="handheld" or "CSS media="tv".

2.4 Font Support

The browser supports all fonts available in Symbian OS installed on the device.

2.5 Character Sets and Character Encodings

The browser supports all character sets and character encodings that are available in Symbian OS on the device.

2.6 Cache

The browser places viewed content pages in the cache, but if caching is not desired, developers can use the HTTP header Cache-Control:"no-cache" directive. The expiration time should be set for at least a few days, in order to ensure that content is cached for a suitable time across multiple time zones.

Cache uses the "least-recently-used" algorithm, which means that items used least are removed first. It is recommended that images and external CSS in all HTML pages be reused to ensure that they stay in cache and need not be downloaded each time they are used. The user can also empty cache manually.

2.7 Network Access

The Nokia 9500 Communicator, which is the first phone compatible with Series 80 Developer Platform 2.0, supports High-Speed Circuit Switched Data (HSCSD), General Packet Radio Service (GPRS; also E-GPRS), and wireless LAN (WLAN; 802.11b) connections.

2.8 Plug-Ins

Series 80 Developer Platform 2.0 supports Macromedia Flash 5 as a plug-in. Audio and video files can be launched in their appropriate players. A plug-in for Open Mobile Alliance (OMA) Download and Digital Rights Management (DRM) content is also available. The default plug-in handles all embedded content that is not supported by the browser natively, or any other plug-in. Embedded content is downloaded only upon user request.

3 Support for HTML and XHTML

Series 80 Developer Platform 2.0 supports HTML 4.01 with minor exceptions. XHTML differs from HTML because it is an XML-based markup language, which enforces some strict syntax rules for Web documents. The browser supports full XHTML 1.0 and XHTML 1.1 when the content is delivered with text/html MIME type. XHTML Basic and XHTML Mobile Profile (XHTML MP), which are subsets of XHTML 1.1 targeted at mobile content development, are also supported. HTML, XHTML, and XHTML Basic are specified by the World Wide Web Consortium (W3C), whereas XHTML MP specification work is maintained in the OMA.

This chapter outlines only those features that are supported beyond standard HTML and XHTML specifications, and covers anything notable in Series 80 Developer Platform 2.0 browser implementation as compared to standards.

3.1 Supported Image Types

Natively, the browser supports image formats GIF, animated GIF, JPG, PNG, BMP, WBMP, and ICO.

It is recommended to specify the correct image width and height in img elements. This speeds up the layout process because the layout engine can reserve the right amount of space on the screen even before downloading the image, which avoids unnecessary screen refreshes.

3.2 Frames

Web pages with frames are of limited use on a mobile browser and therefore frames should be avoided. In particular, pages with a complicated frame structure featuring multiple framesets and frames don't provide good usability on a small display.

3.3 Tables

The browser supports the table elements table, caption, tr, th, and td. The table element, <table>, is used together with <tr> and <td> elements to create sets of rows and columns of data, such as text, images, links, and so on. The cells are shown in bordered rows and columns. The presentation attributes of a table, for example, border width, border color, and text alignment, can be defined in a style sheet.

3.4 Tabindex

The tabindex attribute is used to specify the order in which the form controls in a form should be focused when navigating to the "next element" (usually Tab on desktop browsers). The tabindex attribute is supported in elements a, area, button, input, object, select, and textarea.

The elements with the tabindex attribute are navigated first. Navigation proceeds from the element with the lowest tabindex value to the element with the highest value. The elements that do not have the tabindex attribute (or have it, but assign it a value of "0") are navigated next. These elements are navigated in the order in which they appear in the code.

3.5 File Upload

The browser supports uploading content, such as images, from the device to the Web server. Content is sent from a device to the server as a Multipart Message; the server-side implementation of the content upload is similar to the HTTP upload mechanism for standard Internet browsers. The process of implementing the upload mechanism in the server side is:

1. An XHTML page with a <form> element is created. Inside the <form> element there are at least <input type=file> and <input type=submit> elements. The page is deployed to the Web server.
2. When the page is opened with the terminal browser, the user can select the content to upload.
3. After the content has been selected and the Submit button has been activated, the form is posted to the URL defined in the form. Selected content is part of this Multipart Message.
4. On the server side, the Multipart Message is parsed with an application (written with, for example, CGI, PHP, ASP, JSP, or Java™ Servlet) and content can be saved to the file system of the service provider's server.

3.6 File Download

The browser can be used to download various kinds of content to the phone. The downloadable content support includes:

- Wallpaper/screen savers/animations (Gif87a/89a)
- Images
- Operator logo and calling line identification logo icons
- Java Mobile Information Device Profile (MIDP) applications (MIDlets)
- Symbian installation files (.sis)
- Ring tones, video, audio

There are different technologies for providing the download of the content, depending on, for example, the content type to deliver. The download technologies are:

- OMA Download (forward-lock)
- Java Application Description (JAD) files
- Direct links

For further information about content download methods, see *DRM Developer's Guide For Nokia Devices* (the chapter entitled "Download Methods") available through <http://www.forum.nokia.com/>.

3.7 mailto:, sms:, fax:URL, tel:

The browser supports mailto:, sms:, fax:URL, and tel: schemes if the accompanying software has been installed. It is also possible to add third-party URL handlers with add-on software. When there is an e-mail address defined with the mailto: scheme on a Web page, tabbing the mailto: URL link launches the messaging application. The parameters (e-mail addresses) given in the link are passed to the messaging API and are printed to the recipient text field in the messaging application. The same applies to sms: scheme. When a Web page contains a phone number defined with tel: scheme, a confirmation note is displayed and the user can initiate a phone call directly by accepting the confirmation.

Examples of using mailto:, sms:, and tel: schemes:

E-mail to one recipient:

```
<a href="mailto:john@homeserver.com">Send e-mail to John</a>
```

E-mail to multiple recipients:

```
<a href="mailto:john@homeserver.com,mike@homeserver.com">Send e-mail to John and Mike</a>
```

Short Message Service (SMS) message to one recipient:

```
<a href="sms:+3581234567">SMS to +358 (123) 4567</a>
```

SMS message to multiple recipients:

```
<a href="sms:050-1234567,(040)9876543">SMS 050-123 4567 and (040) 987 6543</a>
```

Initiate a phone call:

```
<a href="tel:(040) 1234567">Call (040) 123 4567</a>
```

3.8 Access Keys

The browser supports the access key attribute. It can be used with the elements `<a>` and `<input>`. This attribute can be used to assign hardware keys to elements in the page, such as hot links and text areas, thus easing navigation.

On the PDA side, keys 0-9 and a-z can be used for assigning access key attributes. The specified access key is opened by selecting Ctrl + Chr + specified access key. Adding the label of the key to an element indicates to the user which key should be pressed in order to activate that element. For example, if the developer associates the key 1 with a link, the number 1 can be added to the label of the link.

```
1. <a href="text.html" accesskey="1"> Text 1 </a>
```

Other way to indicate the shortcut defined with access key to users is to use emphasis, e.g. underlining. For example, if the developer associates the key N with a link, the character N can be underlined so that users know it's a shortcut.

```
<a href="text.html" accesskey="N"> <u>N</u>ext </a>
```

4 Support for CSS

Style sheets enable the content of a page to be separated from its presentation, thus allowing easy control of how services look on the display. Every aspect of a Web page's appearance – positioning, fonts, colors, text attributes, borders, margins, and alignment – can be defined in a style sheet.

The Series 80 Developer Platform 2.0 browser has a default style that specifies how all HTML/XHTML elements will be displayed in the event that developers do not specify their own style. Developers can specify their own style in various ways: in an external style sheet, in a style element in the document head, or by using an inline style attribute in a specific element. Although in general it is a good practice to use external style sheets whenever possible to separate style from markup, it should be noted that there are tradeoffs to consider. Rendering of the HTML/XHTML page is faster when style definitions are inside HTML/XHTML code, but the use of external style sheets offers a convenient way to change styles throughout the service. The same external style sheet should be used throughout the service to avoid downloading multiple style sheets to the phone. The external style sheet is downloaded only once and saved in cache.

The Series 80 browser supports all features defined in CSS 1.0. CSS 2.0 features are supported with some exceptions. The unsupported CSS elements can be found at <http://www.opera.com/docs/specs/opera6/#css>. The CSS specifications from W3C can be found at <http://www.w3.org/TR/REC-CSS1> (CSS 1.0) and <http://www.w3.org/TR/REC-CSS2> (CSS 2.0).

5 Support for HTTP

Series 80 Developer Platform 2.0 supports network protocols HTTP 1.0 and HTTP 1.1. Supported features include:

- Persistent connections (multiple request/response through one connection)
- Resume download, provided the server supports it
- Secured connections (HTTPS://) using Secure Sockets Layer (SSL) 3.0 and Transport Layer Security (TLS) 1.0. SSL and TLS are provided with 128-bit encryption (RSA key exchange only). Secured connection is also supported through proxy/firewall.
- Proxy for HTTP
- Automatic Proxy Configuration

6 Customizing Content

6.1 Validating Content

HTML/XHTML code should be validated to avoid any interoperability problems and to enhance performance. Valid code is always less prone to incompatibilities and errors than pages that contain erroneous syntax. There are several HTML/XHTML validators that validate documents against correct DTD. Content can be validated, for example, with the W3C validator at <http://validator.w3.org>.

6.2 Identifying the Client

Users do not generally want to select which browser they are using. They certainly don't want to be told that their browser is not supported, and that they should upgrade. In a modern Web site hosting environment, it is usually possible to detect the browser and supply correct content transparently, without user interaction. The client can be identified from the user agent field, which is sent in the HTTP header when the client fetches content from the origin server. Developers can use the user agent field to define the type of browser and then build logic on the origin server to serve suitable content.

The user agent name for Nokia devices can be found in the *Browser Characteristics In Nokia GSM Phones* document at <http://www.forum.nokia.com/documents>.

The Series 80 browser supports User Agent Profile (UAProf), which provides another means to customize content for a specific device. The UAProf file contains capability information about the client device. The XML format UAProf file is stored in a repository server and a URL pointing to the file is sent in the HTTP header when the client requests content from the Web server.

The list of available UAProf files for Nokia devices can be found at http://nds.nokia.com/uaprof/uaprof_list.txt and a specific UAProf file can be accessed at <http://nds1.nds.nokia.com/uaprof/<filename>>.

6.3 Identifying the User

6.3.1 Cookies

Cookies enable storage of data, such as user information, which eases browsing by reducing the amount of information the user must enter. For example, log-in can be configured dynamically so that it is displayed only if the application cannot identify the user. Cookies should be used whenever possible to reduce the number of times a user must re-enter user credentials.

6.3.2 HTTP authentication

The browser supports HTTP Basic Authentication. Digest authentication is also supported, except for integrity check on body.

7 Terms and Abbreviations

Term or Abbreviation	Meaning
BMP	Bitmap.
CSS	Cascading Style Sheet. Mechanism for adding style (e.g., fonts, colors, spacing) to Web documents.
CSD	Circuit Switched Data.
DRM	Digital Rights Management. Governs how mobile content is used and allows the development of new end-user features and new kinds of mobile content services for content providers, service developers, operators, and service providers.
ECML	Electronic Commerce Modeling Language. A specification that defines a common vocabulary for electronic wallets and merchant Web pages.
GIF	Graphic Interchange Format. Image format for small images, like icons and buttons, and black-and-white images.
GPRS	General Packet Radio Service. Mobile service that gives a packet switched access over GSM to external data networks with high peak transfer capacity.
HSCSD	High-Speed Circuit Switched Data.
HTML	Hypertext Markup Language. The markup language that is used for publishing hypertext on the Web.
HTTP	Hypertext Transfer Protocol. An application-level protocol for transferring HTML documents on the Internet.
JAD	Java Application Descriptor.
JPEG	Joint Photographic Experts Group. Image format for large images or images with a lot of detail, scanned images, and photographs.
OMA	Open Mobile Alliance. The center of mobile service standardization work, helping the creation of interoperable services across countries, operators, and mobile terminals that will meet the needs of the user.
PDA	Personal Digital Assistant.
PNG	Portable Network Graphic.
RMP	Rocker Manipulated Pointer.
SSL	Secure Socket Layer. The industry standard method for protecting Web communications.
TLS	Transport Layer Security. Security specification based on SSL 3.0.
UAProf	User Agent Profile. A device profile in XML format that contains capability information about the client device.
URL	Uniform Resource Locator. String that identifies resources on the Web.
WAP	Wireless Application Protocol.
WAP CSS	WAP Cascading Style Sheet. CSS specification optimized for wireless devices.

WML	Wireless Markup Language. Markup language based on Extensible Markup Language (XML), which is intended for use in specifying content and user interface for narrowband devices, including cellular phones and pagers.
WTAI	Wireless Telephony Applications Interface. Feature that provides the means to create telephony applications by using a WAE user-agent with the appropriate WTAI function libraries.
W3C	World Wide Web Consortium. An organization that develops interoperable technologies for the Web.
XHTML	Extensible Hypertext Markup Language. XHTML uses the definitions in HTML 4.01 but requires proper XML syntax.
XHTML Basic	A subset of XHTML for small devices that omits features, such as frames, which are inappropriate for small screens.
XHTML MP	XHTML Mobile Profile. XML-based markup language that contains XHTML Basic and a few additional elements from full XHTML 1.1.
XML	Extensible Markup Language.

8 References

Browser Characteristics in Nokia GSM Phones <http://www.forum.nokia.com>

Cascading Style Sheets 1 Specification <http://www.w3.org/TR/REC-CSS1>

Cascading Style Sheets 2 Specification <http://www.w3.org/TR/REC-CSS2/>

DRM Developer's Guide For Nokia Devices <http://www.forum.nokia.com>

HTML 4.01 Specification <http://www.w3.org/TR/html4/>

Introduction to User Agent Profile <http://www.forum.nokia.com>

Series 80 UI Style Guide <http://www.forum.nokia.com>

WAP Cascading Style Sheet Specification <http://www.openmobilealliance.org>

Web Specifications Supported in Opera 6 <http://www.opera.com/docs/specs/opera6/>

XHTML Mobile Profile 1.0 Specification <http://www.openmobilealliance.org>

XHTML 1.0 Specification <http://www.w3.org/TR/xhtml1/>

XHTML 1.1 – Module-Based XHTML Specification <http://www.w3.org/TR/xhtml11/>

XHTML Basic Specification <http://www.w3.org/TR/xhtml-basic/>