

# Quick Guide To Optimizing XHTML Mobile Profile Services

Version 1.0; March 22, 2004

Usability

**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
2	Keep user task flow fluent and be reasonable with image use.....	6
3	Make the structure easy for novices but don't forget power users .....	6
4	Provide sufficient information on a page.....	6
5	Provide informative feedback for user actions .....	7
6	Minimize the amount and size of images.....	7
7	Specify image height and width attributes.....	7
8	Consider the use of tables carefully.....	7
9	Consider options for adding style definitions .....	8
10	Remove unnecessary white space and comments inside code .....	8
11	Enable caching of pages using HTTP header directives.....	8
12	Use Unicode 2.0 character sets for XHTML content.....	9
13	Use correct MIME types and validated XHTML code .....	9
14	Use multipart/mixed to fasten XHTML page download .....	9

## Change History

March 22, 2004	Version 1.0	Initial document release

## 1 Introduction

This document lists recommendations for optimizing XHTML Mobile Profile (XHTML MP) content for Nokia mobile browsers. It is intended to serve as a guideline for content developers to ensure the best usability and functionality of XHTML MP-based mobile browsing services.

Devices and services are often described as easy to use, but seldom is there an explanation of what that actually means. There are several issues to take into consideration when developing an XHTML service that is "usable." The user must be considered from the very beginning of service creation: Service must answer the users' questions, and information must be provided in such a way that it decreases user memory load. The user interface should be designed to support user tasks and task flow and provide proper information about the selections and actions.

To ensure that a service is easy to use, we recommend that the usability process be followed in service design and that the user interface be designed with detailed guidelines.

## 2 Keep user task flow fluent and be reasonable with image use

Colorful pages look great but may not *feel* great when images slow down service. According to usability studies<sup>1</sup>, users are less enthusiastic about services if images disrupt their task flow. In particular, large images aren't appreciated when users are navigating toward the target page. Images that have information value are appreciated, but in many cases users either turn off images to save time and money, or proceed to the next page without waiting for the images to download. It is important to allow users to continue navigation even before all images are downloaded.

Large tables may cause similar problems—that is, the user is stuck on a page until it is downloaded or can't find the way to proceed before the page is fully downloaded. Because phone displays are different sizes, developers should make sure that data tables are readable even on the smallest displays; often they must be squeezed to fit the display.

## 3 Make the structure easy for novices but don't forget power users

In mobile services it often seems that a shallow structure is easier to understand than a deep structure. Links and pages should feature descriptive names to help the user find the information s/he needs.

It is hard to say how many links should be provided on one link list page. If the links clearly belong together and are easy to browse (one line per link, alphabetical or otherwise logical order so that the user does not have to read through each link), it is better to provide 30 links on a single page rather than five links on six pages. If there are tens of links, it may be a good idea to provide sorting options before showing them. If the link can fit on one row, it makes the selection clear and the page look better.

There are no <do> elements in WAP 2.0; instead, they have been replaced with accesskeys. However, most users seem to be unaware of accesskeys and unable to find them. To help users to understand the concept of accesskeys, developers should make sure that they are visible on the screen and in a form that resembles phone keys.

If possible, a search function should be provided. Power users appreciate it, as do novices navigating large sites.

## 4 Provide sufficient information on a page

Interactive pages should be short; informative pages long.<sup>2</sup>

In XHTML, information is downloaded as pages and not as decks as in WML. This means that it is even more important to provide necessary information to users on a single page to support their task flow. Going back and forth between pages may take more time in XHTML, because the pages are downloaded separately. In backwards navigation this applies especially to cases where pages are not cacheable, e.g., in systems related to paying or where private information is provided.

It is difficult to scroll up and down pages, so interactive pages with forms should not be too long, since users may not be sure if they have filled in every field on a long form. Users can easily lose the feeling of control if the form takes more than two screenfuls.

<sup>1</sup> Roto, V., Kaikkonen, A.: *Acceptable Download Times in the Mobile Internet*. In Stephanidis, C. (ed): *Universal Access in HCI*. Volume 4 of the Proceedings of HCI International 2003.

<sup>2</sup> Kaikkonen, A., Roto, V.: *Navigating in a Mobile XHTML Application*. Proceedings of CHI2003 conference

There should be sufficient information on the destination page to which the user is headed. For example, if the target page contains a story or instructions, the entire contents should be presented on one page. Subtitles that take the user to points within the page help when browsing a long, informative page.

The fact that information is not downloaded as decks but as individual pages is the biggest individual change that affects navigation and structural differences between WML and XHTML.

## 5 Provide informative feedback for user actions

Developers should provide proper feedback for user actions and for error and problem situations. For example, after the user clicks a link, the page title should be the same as the link name. Minimizing steps in navigation should not increase the user's feeling of insecurity, e.g., confirmation pages for user actions may be needed even if they require an extra click. If the confirmation page is missing, the user may feel s/he needs to check to see if the action has taken place—this leads to even more clicks. User should feel that they are controlling the system.

If problems arise, users should be informed what to do next. Errors can be prevented by explaining to users the format of expected input, and by marking mandatory fields.

## 6 Minimize the amount and size of images

The amount and size of images on an XHTML should be considered carefully. Each image on a page causes a separate round trip, which in turn makes the rendering of an entire page slower. Therefore the number of round trips should be minimized. Also it should be noted that each time an image arrives at the mobile device, the entire containing page may be rearranged, which takes time and processor resources. Thus, a page with just a few images may load faster than one with many smaller images. If possible, it is recommended to use the same images on various pages throughout the service; then a specific image is downloaded only once and saved to cache. For example, if custom images are used for bullets, the same ones should be used throughout the service

TCP/IP connection may result in different download speeds for pages, even though the amount of data is the same. For example, downloading an XHTML page containing four images at 2 KB each is faster than downloading a page with eight images at 1 KB each.

## 7 Specify image height and width attributes

It is recommended that content developers explicitly specify the height and width of images in markup to allow the browser to reserve the proper space for the image. Width and height should correctly match the actual image's size so the browser doesn't have to spend time resizing the image. When width and height parameters are used in image tags, the XHTML browser is able to reserve the space for images before the images are downloaded. Thus the page is displayed to the user before images are downloaded and images become visible when they are downloaded. This doesn't change the complete download and processing time of the XHTML page, but it improves the user experience substantially because a user can read the page before the images are downloaded. For example:

```

```

## 8 Consider the use of tables carefully

The browser supports the use of tables and nested tables in the XHTML page. Developers should be careful when defining cell widths, particularly with nested tables.

The CSS single-pass (fixed) algorithm is used in laying out tables in order to optimize CPU usage. Unlike the CSS multi-pass (dynamic) table layout algorithm, however, the fixed table layout algorithm determines the number and size of columns based on the first row. Consequently, optimal performance is achieved by rectangular tables with explicit column widths.

When using nested tables, content developers should avoid specifying the parent table width as a percentage when the child table widths are explicitly dimensioned. Because there are devices with different screen sizes, percentages will not always represent the same number of pixels. Therefore it is recommended that absolute widths (pixels) be used for both parent and nested tables, to ensure the content will be rendered properly. Care should be taken to ensure that the total table width is the same as the sum of the widths of the individual columns plus borders and cell spacing. In general, as table nesting levels increase, so does the complexity of the page and the required processing time to display the page. In order to ensure the timely display of pages, very deeply nested tables should be avoided.

Tables should also not have borders that are too thick, since in a device of limited display size the border width can easily consume enough pixels to make the actual content area too small to be useful.

## 9 Consider options for adding style definitions

Developers can specify their own styles in various ways: in an external style sheet, in a style element in the document head, or by using inline style attribute in 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 XHTML page is faster when style definitions are inside XHTML code, because the browser always redraws the screen when it processes an external style sheet (even if it is cached). However, the use of external style sheets offers a convenient way to change styles throughout the service. A good solution is to insert the style sheet inline in the document's style element. If an external style sheet is linked, then 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.

## 10 Remove unnecessary white space and comments inside code

It is important to make sure that there is no extra white space inside the code. Although white space may have zero displayed size, it is still processed, as the browser parses, formats, and applies CSS to white space. All white space is saved in the browser in case CSS is applied to it later, which would make it visible, e.g., `white-space:pre`.

The number of comments in XHTML code should be minimized to keep the code as compact as possible.

## 11 Enable caching of pages using HTTP header directives

The browser places viewed XHTML pages in the cache; however, content developers should not assume that pages are cached by default. Explicit cache headers should be sent with documents to ensure that pages are cached on the client, if possible. In addition, an 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.

Placing cache directives within Meta tags (e.g., using HTTP-EQUIV) is not supported but caching can be controlled using HTTP headers. The "no-cache" HTTP header directive can be set by the HTTP server hosting the pages to define that pages are not cached.

Cache uses the “least recently used” algorithm, which means that items used least are removed first. It is recommended to reuse images and external CSS in all XHTML pages to ensure that they stay in cache and need not be downloaded again when they are used later.

## 12 Use Unicode 2.0 character sets for XHTML content

The Nokia XHTML browser supports ASCII and Unicode 2.0 character sets. Therefore to ensure the greatest level of interoperability, all XHTML content should be created by using Unicode for non-Latin languages. For Latin languages, ASCII can also be used. Some gateways and proxies can convert local character sets to Unicode, but not all. Therefore the only way to guarantee the terminal receives Unicode is to create the content in Unicode. More information about Unicode and non-Latin languages can be found in the following books:

Lunde, Ken. (December 1998) CJKV Information Processing. 1st edition. O'Reilly & Associates.

Graham, Tony. (March 2000) Unicode: A Primer. John Wiley & Sons.

## 13 Use correct MIME types and validated XHTML code

The preferred MIME type for XHTML MP content is "application/xhtml+xml", which should be used for serving XHTML MP documents to XHTML user agents. In addition, "application/vnd.wap.xhtml+xml", which is specified by OMA, can be used. The MIME type "text/html" is available too, but for XHTML, use of this type should be reserved for the purpose of rendering on existing desktop HTML user agents. It should be noted that XHTML documents served as "text/html" will not be processed as XML. Using correct XHTML mime types or DOCTYPE header tells the browser to not reformat the page into narrow screen mode, because the page is already designed for mobile devices. Authors who wish to support both XHTML and HTML user agents may utilize content negotiation by serving HTML documents as "text/html" and XHTML documents as "application/xhtml+xml".

It is recommended to use file extension \*.xhtml in all XHTML MP content.

XHTML code should be validated to avoid any interoperability problems and to enhance performance. XHTML content can be validated, for example, with the W3C validator at <http://validator.w3.org>. When creating XHTML content dynamically, the generated code should be valid DTD XHTML MP 1.0.

## 14 Use multipart/mixed to fasten XHTML page download

Multipart is used to request and deliver XHTML page content in a single multipart message, instead of the current multiple independent page object requests. This makes page downloading faster. For example, if an XHTML page contains text, seven images, and a link to an external style sheet, the whole content can be requested in one request instead of ten separate requests. To use this feature, multipart must be supported both in the Web server and in the browser. Content developers must take care of encoding all displayable content in a page to a multipart message.

For information about Nokia phones supporting multipart/mixed MIME type, see *Browser MIME Types In Nokia GSM Phones* at [www.forum.nokia.com/documents](http://www.forum.nokia.com/documents).