

S60 WebKit FAQ Sheet

What open source components are used in the new Nokia Web Browser?

Beginning in S60 3rd Edition, Nokia launches a new web browser for S60, enabling browsing of full Web sites on the Internet. The new browser is based on the WebCore and JavaScriptCore components of Apple's Safari Web Kit, the industry's smallest, open source, full-Web rendering engine for mobile devices that Apple uses in its popular Safari Internet browser. Based on KHTML and KJS from KDE's Konqueror open source project, this software has enabled Nokia to achieve major improvements in Web site usability on smartphones, through the re-use of a proven desktop rendering engine that has been developed and optimized by a large open source community over many years.

In addition, Nokia is open sourcing S60 WebKit, Memory Manager, and a Reference UI implementation so that the open source community can now work with the browser.

For more information about the browser see <http://www.s60.com/s60/html/index.html>.

Why did S60 decide to use WebCore/KHTML and JavaScriptCore/KJS for the browser?

We have been developing a proprietary browser for S60. We found that we have been focusing a lot of resources on dealing with issues like rendering and script execution - issues that have already been solved by open-source components. Since browsing is a complex technology, Nokia decided we should not re-invent the wheel. We started investigating the available open-source solutions and decided to go with a KDE-based solution primarily for 3 reasons:

1. S60 devices are constrained by ROM/RAM. WebCore/KHTML and JavaScriptCore/KJS provide rendering engines that use very small RAM/ROM footprint. That was really a clincher for us. Minimo, the mobile project based on Mozilla's Gecko engine, is unfortunately still too large for the devices we are targeting.
2. WebCore/KHTML and JavaScriptCore/KJS provide a very clean architecture enabling an easy learning curve for our developers.
3. WebCore/KHTML and JavaScriptCore/KJS are very fast. The startup time and general performance of the browser is quite good.

What technical challenges did you face in that work?

Mobile devices are constrained by ROM, RAM, network latency and bandwidth, display, difficulty of input on a phone keypad, and a less powerful CPU compared to the desktop world. We needed to make some changes to take into account these constraints. The code execution behaviors that are correct in the desktop world do not always work well on mobile device.

To give an example, in browsers JavaScript needs to be executed at the position in the markup where it occurs, resulting in blocking the markup parser. External JavaScript files take a much longer time to arrive at the client over wireless connections. In the worst case, the consecutive multiple external JavaScript references cause the requests to be sent serially, each request blocked until the previous response is received. Similar problems are encountered in CSS processing since the rendering engine blocks when waiting for an external CSS file. We do not encounter these problems when the Javascript and CSS is included inline within the HTML file.

What is the open source licensing model applied to open source components?

The WebCore and JavaScriptCore elements are from Apple Computer, which uses them in its popular Safari browser, and are covered by the LGPL open source license. These components in turn are based on KHTML and KJS from KDE's Konqueror open source project. They are covered by the LGPL open source license. See <http://developer.apple.com/opensource/> for details.

The S60 WebKit source code released by Nokia comes under the terms of the open source BSD License, a highly permissive software license with few requirements that is one of the most popular licenses among free software developers worldwide. The source code will be made available to open source developers through the [WebKit Open Source Project](#).

How is the relationship evolving between the open source community and Nokia's project?

We would like to find a model where we can collaborate with the open source community that is focused towards browsing related solutions. With the advent of faster networks and more capable mobile devices we expect to see a lot of development in the mobile software area. We hope that in the future the open source community will be interested to look beyond desktop software and considers taking on projects in the mobile software space. We see Nokia's new open source browser as a good project to spark the open source community's interest in mobile software applications.

We have now open sourced all the components that are required by the open source community to work with the S60 Browser engine. We hope to work with the community to work on the browser development.

Can I port the browser to other mobile devices?

Yes, one can port the same open source core to other mobile devices just as Nokia has done. You would need to port the KWQ layer in WebCore, write a new Web Kit and a new Browser UI to create a browser on a new platform.

What are the mobility enhancements and new components that Nokia is contributing to the open source?

Some of the mobility enhancements done by Nokia are:

Memory manager: designed specifically to handle out of memory situations on the device. WebCore and JavaScript core were not optimized for low memory devices. The memory manager helps to handle low and out of memory situation, which is important for many mobile phones with limited memory.

Usability improvements: layout scaling that fits the text on a mobile page to the display width, rendering of frames as tables, text search capabilities as well as a mouse pointer to give users a desktop-like navigation experience.

Reference user interface: including a number of standard browser user interface features. This can get you easily started on the development of new browser core or user interface features.