



Taking S60 Java Platform user interface capabilities to the next level

eSWT for S60 3rd Edition FP 2 is a major update to the Java UI frameworks, providing more flexibility and a richer UI experience, as Aleksi Uotila, Software Chief Engineer for S60 Java UI explains.

The low-level graphics capabilities of the S60 Java Platform have improved steadily with each release. The Java base platform APIs from MIDP 2.0 have been providing basic 2D graphics features from S60 2nd Edition, while Mobile 3D Graphics API was introduced in S60 2nd Edition FP 2 and Scalable 2D Vector Graphics in 3rd Edition FP 1. The latter provides access to the Scalable Vector Graphics (SVG-T) engine of the S60 platform.

While the graphics capabilities of S60 Java platform have been very powerful, the component functionality of the user interface has lagged behind, as the only API has been the Limited Capability Device User Interface API (LCDUI) of MIDP.

The MIDP LCDUI was originally designed in the era of black and white monochrome displays and very limited functionality UIs such as that for the Nokia Series 30. The LCDUI hasn't seen any significant improvements in Java standardization recently because it must also be available on low-end devices and the portability of LCDUI applications could not be sacrificed. Currently, the LCDUI does not provide the UI functionality of modern smartphone user interface frameworks such as the S60 UI.

So far, the capability of the S60 Java platform UI toolkit has been on the level of LCDUI. However, the S60 3rd Edition FP 2 changes this dramatically by introducing a completely new UI component toolkit to the Java Platform called eSWT. Standing for embedded Standard Widget Toolkit, eSWT is a significant facelift for Java UI frameworks.

The eSWT API provides:

- A rich user interface component set
- Flexible and scalable layout system via layout managers
- Rich user interface events
- Access to native UI functionality on par with smartphone UI frameworks.

Since eSWT on the S60 has been implemented using the normal S60 UI services, applications built using it will automatically have the standard look and feel of S60, including S60 themes and internationalization features.



In some respects, eSWT provides even more than the S60 native UI services, as all of the components in eSWT are freely locatable and resizable. The toolkit automatically implements a focus traversal scheme for applications, which can, if needed, be customized by the application. Since eSWT is a Java API, it is also makes development quick and easy. There are UI creation tools available, both free open source and commercial alternatives.



An email application using eSWT shows the possibilities of eSWT for complex UI layouts

Background of eSWT

The eSWT API was jointly developed by IBM and Nokia in the Eclipse Foundation eRCP project [1]. However, most of the eSWT API is actually a direct subset of desktop SWT API, itself a very popular Java UI API for desktop Java. Many applications use SWT, from commercial to freeware. For example, the Eclipse IDE itself uses SWT to create its user interface, while another example is the Lotus Symphony free office suite. Existing SWT developers will view eSWT as their second home as it shares all its principles with SWT, as well as its core components and even some advanced ones.

Power and flexibility of desktop GUI on a phone

eSWT is as flexible as any modern GUI toolkit. Developers can easily create UI views that automatically adapt to the device's resolutions, providing a very easy way to create scalable user interfaces without being limited to using simple layouts in the application view designs.



The screen shots reproduced on this page show an eSWT e-mail application created by Lauri Jääskelä to demonstrate the layout capabilities of eSWT. The toolkit makes it easy to create complex layouts using the UI components as building blocks. If LCDUI was used, the application would be forced to render everything itself but wouldn't have access to platform themes or input methods to perform text editing. These issues are solved using eSWT.

Compared to Java LCDUI, eSWT provides much finer control for nearly everything. It has rich UI events from any part of the user interface: focus, traversal, keyboard and pointer presses. Applications can use standard dialogs for files, directories, colors, font, messages and queries. eSWT also has a Mobile Extensions package for user interface elements common on S60 that cannot be found on the desktop. It provides access to features such as S60 style tabbed panes, different list box layouts, text input capabilities found on mobile devices and so on.

eSWT also has a Browser package that can be used to instantiate the S60 Browser control within Java applications. This allows all sorts of interesting use cases such as easy ways to do online help, messaging applications (e-mail content is often HTML) and even mashups using AJAX applications from a Browser widget for which the more complex business logic is done in Java.

Tools for the UI toolkit

The basic tool set for eSWT developers are normal Java development tools for which there are plenty of alternatives. When a developer wants to see and test the eSWT application running on an actual S60 platform, he can use the S60 MIDP SDK emulator [4] from Nokia, which supports eSWT.

More advanced GUI creation tools are also available. Even if the desktop SWT GUI creation tools can also be used to some extent for eSWT applications, there are tools for eSWT. Instantiations, a long-time supplier of SWT GUI tools, has an eSWT UI designer product [2] that can be used to create eSWT application user interfaces in WYSIWYG form. The tool provides device skins for many vendor devices.

Innovation done in open source

eSWT, including the S60 eSWT implementation, is open source just like any other Eclipse based technology. The API and implementations are developed in close co-operation within the Eclipse eRCP project. This allows multi-platform, multi-vendor UI APIs to be defined very quickly.

The open source way of working also has a significant impact on quality, as implementation aspects are discussed openly between teams from different companies, for example, to resolve compatibility issues. As well as implementation code, test cases are also being shared in the project.



eSWT API is evolving all the time. The S60 3rd Edition FP 2 implementation is based on v1.0 of the eSWT API specification but the open source project has already developed a minor upgrade v1.1 and the next release is being designed. Luckily the eSWT implementation is also updatable to newer versions on S60 devices.

Cross platform wonder

SWT was designed from the start as a cross platform GUI toolkit. SWT itself has been implemented in nearly all modern desktop windowing systems: Windows, Mac OS X Carbon, Linux, FreeBSD/AIX on GTK+ and Motif, QNX Photon and Pocket PC.

eSWT aims to reach more portable device platforms and currently there are implementations by IBM for Windows Mobile and WinCE and by Nokia for Series 80 and S60. SonyEricsson is implementing it for its mass volume feature phone platform. And we expect, of course, more platforms in the future.

The Eclipse RAP project [5] expands the supported eSWT/SWT platforms to modern web browsers – the RWT module of RAP is an implementation of SWT on AJAX. To demonstrate the ease of porting, Görkem Ercan, the lead of the Eclipse eRCP project and S60 eSWT implementation, has demonstrated [3] a single application, initially developed for S60 device using eSWT, being deployed to desktop and web browser environments. To create the other deployment packages took him about an hour.

In summary, eSWT is an ideal toolkit for application UIs for those applications that need the cross platform capabilities of Java but still need more UI functionality and fine grain control. eSWT is also a powerful toolkit for development aimed at only S60 use, because of the ease of use and high development efficiency compared to S60 native C++ development. The tools for eSWT development are now publicly available and when S60 3rd Edition FP 2 devices start to ship, eSWT will be ready for prime time.

[1] eRCP Project at Eclipse: <http://www.eclipse.org/ercp>

[2] Instantiations eSWT GUI builder: <http://www.instantiations.com/images2/forum/eRCP.gif>

[3] eSWT demos: <http://www.eclipse.org/ercp/demos.php>

[4] S60 Java MIDP SDK: <http://www.forum.nokia.com>

[5] Eclipse Rich Ajax Platform (RAP): <http://www.eclipse.org/rap>