

This article is archived because it is not considered relevant for third-party developers creating commercial solutions today. The article is believed to be still valid for the original topic scope.



Developing widgets for WidSets requires learning a Java-like, strongly typed, proprietary scripting/programming language. One could ask why not just develop the code directly in Java, even WidSets itself is written in Java. The reason lies behind the fact that the Mobile Information Device Profile (MIDP) does not support dynamic classloading. Therefore it is not possible to load new Java classes to a running MIDP application and all functionality needs to reside inside a JAR file.

The WidSets approach has, however, a certain benefit: compiled WidSets Scripting Language (WSL) code will be packed down to smaller bytecode size than the same amount of Java bytecode (even when obfuscated). WidSets widgets therefore consume less storage space in the phone memory. Smaller bytesize means less data traffic when the code is being transferred to a mobile phone. There are also tools to monitor the amount of traffic sent between client-server on the client software or at the WidSets Web site. Traffic limitations can also be set.

WidSets servers take care of content being modified to fit the mobile usage on the client side. Images can be scaled down and image format can be converted to work separately on each different device model. The service also enables filtering the relevant content from Web pages or XML documents, as requested by the widget. Servers automatically poll for feeds and content streams, and notify the mobile client and widget when there is new data available.

## How to start developing

Start reading from the [Developing your first WidSets widget](#) article. It describes the widget development process. Check the [stylesheet](#) syntax and XML-side UI construction, and have a look at the [Syndication](#) and [PicViewer](#) template models, which are widely used in the widgets in the system. These types of widgets are easy to create and work well with existing feed aggregation-based web technologies.

For more detailed information on content fetching, there are multiple articles available [at content fetching with the available services](#). There are also detailed instructions and references on how to use the advanced filtering technology, which is the key component in transforming your favourite web service to a WidSets widget. You can also find out how the variety of more advanced technologies and technical details such as [WidSets bookmarking](#) work, or how to enable the [sharing via e-mail](#) functionality in a widget.

See also the article called [Integrating WidSets with Web Sites](#) which describes how to bridge the gap between the WidSets-ready services and the system itself. You can also find ways on how to promote your widgets within the system and how to build external links to a widget in the library. If security is a concern, the article also describes the advanced security models that can be implemented between a partner and a single widget.

The article [Porting from MIDP to WidSets Scripting Language](#) explains how the WidSets platform differs from Java ME/MIDP, and what actions need to be done when porting existing MIDP Java applications to WidSets Scripting Language (WSL) so that they can run on the WidSets platform.

This documentation along with the [WidSets API docs](#) brings the world of functional mobile widgets to the developers. If you like your creation, take the time to publish it in our widget library, share it out to the

community, and promote it.

## See also

- **Introduction to developing WidSets widgets**
- [Widget files](#)
- [Mobile developing and design guidelines](#)
- [Publishing your own widgets](#)
- [Widget Configuration 2.0](#)
- [Stylesheet](#)