GWT Studio plugin

GWT Studio plugin for IntelliJ IDEA

WATCH DEMO

What is GWT Studio?

GWT Studio plugin for IntelliJ IDEA simplifies development of Web applications using Google Web Toolkit. The plugin is bundled with IDEA.

GWT Studio provides the following features:

- actions to create different elements of GWT application
- GWT specific inspections and quick fixes
- integration with GWT compiler
- specific run configuration for starting and debugging GWT applications in the hosted mode
- "Rename" refactoring is aware of GWT
- "Go To Declaration" action, "Find usage" action and completion for GWT specific references
- GWT native methods support: advanced coding assistance including syntax highlighting, code completion, etc. for editing JavaScript embedded into native GWT methods (see description of IntelliJ IDEA JavaScript editor)
- integration with GWT internationalization support
- support for packaging files generated by GWT compiler into JavaEE application

Actions

Actions are used to generate GWT elements such as entry points, remote services, etc. Each action suggests to configure GWT support if necessary and adds GWT libraries into module classpath.

New GWT Module

Creates GWT application template including:

- directories
    - client
    - server
    - public
- .**gwt.xml** module file
- EntryPoint class
- .html page
- .css file

New GWT Entry Point

Creates GWT entry point template and performs the following operations:

- generates a class derived from EntryPoint
- adds a necessary **<entry-point>** element to the .**gwt.xml** module file

New GWT RemoteService

Creates GWT remote service template and performs the following operations:

- generates an interface derived from RemoteService
- creates an asynchronous version of the interface
- creates RemoteServiceServlet implementing the interface
- adds necessary **<servlet>** element to the .**gwt.xml** module file

New GWT Serializable Class

Creates new class derived from **IsSerializable** with no-argument constructor.

New GWT Test Case

Works under test sources roots. Creates new class derived from GWTTestCase.

New GWT UiBinder class and ui.xml file

Creates new *.ui.xml file with the corresponding Java class

References

"Go to declaration", "Find Usages", "Rename" actions, completion and highlighting works for GWT specific references in the following places:

- parameters of methods UIObject.setStyleName, addStyleName, removeStyleName
- parameter of RootPanel#get method
- @Key annotation
- <inherits> tag in .gwt.xml file
- the return statement in "getModuleName" method in classes derived from GWTTestCase
- parameter of ServiceDefTarget#setServiceEntryPoint method
- @RemoteServiceRelativePath annotation
- <meta> and <script> tags in GWT html file
- tags and attributes in ui.xml files
- @UiHandler annotation

Inspections

GWT Studio have inspections to report the following problems:

- RemoteService-derived interface is not synchronized with its asyncronous version
- Asynchronous version of RemoteService-derived interface is not synchronized with the original interface
- Type of parameter or return type of method in RemoteService-derived interface is not serializable
- GWT Remote service is not registered as a servlet in web.xml
- Classes not from JRE emulation library (and not from client code of inherited modules) are used in client code
- Class marked as serializable contains non-serializable fields
- Style name used as parameter of "addStyleName" or "setStyleName" is not presented in CSS file
- References to Java classes or methods from GWT JavaScript native methods cannot be resolved
- Localizable interface is not synchronized with the corresponding .properties file
- Interfaces derived from Constants contains methods with parameters
- Deprecated @gwt.key JavaDoc tag usages
- Deprecated @gwt.typeArgs JavaDoc tag usages
- AsyncCallback class is used without type parameters
- ServiceDefTarget.setServiceEntryPoint is used instead of @RemoteServiceRelativePath annotation
- Usages of event listeners classes deprecated in GWT 1.6
- Inconsistencies between fields with @UiField annotation and 'ui:field' attributes in ui.xml files
- Wrong signatures of @UiHandler methods

Each inspection suggest a quick-fix to resolve the corresponding problem.

Integration with I18n support

- navigation from method of interface to corresponding property (using icon on an editor gutter or "Go to implementation" action)
- inspection to report inconsistencies between interfaces and property files (with quickfixes to create missing property or method)
- "Rename" refactoring for properties, methods and interfaces renames corresponding methods, properties and properties-files
- "Internationalize" action in IDEA is aware of GWT and automatically creates corresponding methods in interfaces

See also http://blogs.jetbrains.com/idea/2006/10/internationalizing-gwt-applications-at-ease/.

Settings

GWT-related settings are available via GWT Facet (Project Structure | "Modules" item | A module node in the tree | "GWT" node)

Packaging into JavaEE application

An output of GWT compiler can be automatically copied to an artifact on make. Just include the GWT Compiler Output element in the artifact layout. An individual relative output path for each GWT module can be specified in GWT Facet settings as well.

Run/Debug GWT application in the hosted mode

Use "GWT configuration" to run/debug a GWT application in the hosted mode. Also "Run" action is available in the popup menu for GWT entry

point classes and html files allowing to run a GWT application the hosted mode immediately. If AppEngine is used in the project GWT configuration allows to specify AppEngine Dev Server as a target server.

More information

GWT Studio comes bundled with IntelliJ IDEA starting from IDEA 6.0.

To report bug or request feature, visit http://youtrack.jetbrains.net/issues/IDEA