JavaScript-Specific Guidelines

IntelliJ IDEA enables creating rich Internet applications and Web applications by providing elaborate support of JavaScript and tight integration with AJAX and other adjacent frameworks and technologies.

In this section:

Prerequisites

Before you start working with JavaScript, make sure that **JavaScript Support** plugin is enabled.

The plugin is bundled with IntelliJ IDEA and activated by default. If not, enable the plugin as described in Enabling and Disabling Plugins.

JavaScript support

JavaScript files are marked with  icon.

JavaScript support in IntelliJ IDEA includes:

- Switching between JavaScript language versions to choose the one that suits the targeted browser:
  - JavaScript 1.5
  - JavaScript 1.6
  - JavaScript 1.7
  - JavaScript 1.8
  - JavaScript 1.8.5
  - ECMAScript 5.1
  - ECMAScript Harmony

- Full coding assistance:
  - Smart, DOM-based, browser-type aware JavaScript code completion for:
    - Keywords, labels, variables, parameters and functions.
    - User defined and built-in JavaScript functions.
    - JavaScript namespaces.
  - Error and syntax highlighting.
  - Code formatting and folding.
  - Numerous code inspections and quick-fixes, including the **Switch to JavaScript <version>** quick-fix.
  - Support of the strict mode standard.
  - Support for ECMAScript.
  - Initial support of ECMAScript Harmony.
- Code Generation
  - Generating code stubs based on file templates during file creation.
  - Inserting, expanding, and generating JavaScript code blocks using live templates.
  - Creating various applications elements via JavaScript and AJAX intention actions.
  - Possibility to create line and block comments (`Ctrl+Slash` or `Ctrl+NumPad //Ctrl+Shift+Slash` or `Ctrl+Shift+NumPad /`).
  - Unwrapping and removing statements.
- Possibility to build and view type, method and call hierarchies
- Refactoring

  IntelliJ IDEA provides both common refactoring types available for all the supported languages and JavaScript-specific refactoring.
  - Common refactorings:
    - Rename a file, function, variable, parameter, or label (both directly and via references).
    - Move/Copy a file.
    - Safe Delete a file.
    - Extract inlined script from HTML to a JS file.
    - Extract function.
  - JavaScript-Specific Extract Parameter Refactoring
- Numerous ways to navigate through the source code, among them:
  - Navigating with Structure Views.
  - Show/Goto Implementation (`Ctrl+Alt+B` or `Ctrl+Alt+Button1 Click`) from overridden method / subclassed class.
- Advanced facilities for searching through the source code.
- Support of the JSDoc format and generating documentation comments.
- Viewing reference information:
  - Definitions, inline documentation, parameter hints.
  - JSDoc comments.
  - Lookup in external JavaScript libraries.

- Running and debugging.
  - Launching applications directly from IntelliJ IDEA by opening the starting application HTML page in the default IntelliJ IDEA browser.
  - A dedicated debug configuration for launching debugging sessions directly from IntelliJ IDEA.
  - A JavaScript-aware debugger that lets you execute applications step by step, evaluate expressions, examine related information and find runtime bugs.
  - Support for JavaScript breakpoints.
- Tight integration with related frameworks and technologies: AJAX, JQuery, YUI, DoJo, Prototype, MooTools, Qooxdoo, and Bindows:
  - Code completion for every framework.
  - DoJo style type annotations to provide more accurate completion and parameter type information.
  - Quick Documentation lookup for DoJo style commands.
- Support for the JSON (JavaScript Object Notation) format:
  - A JSON file type template mapped to `json` file extension.
  - JSON code formatting and folding.

To develop an application that contains JavaScript

Developing an application that contains JavaScript, generally, includes performing the following steps:

1. Make sure the JavaScript Support plugin is enabled. The plugin is bundled with IntelliJ IDEA and activated by default. If it is not, enable the plugin.
2. Create a project to implement your application. On the first page of the New Project wizard, select **Static Web**.
3. On the JavaScript page of the Settings dialog box, choose the JavaScript language version that suits the targeted browser.
4. Download, install, and configure JavaScript frameworks and libraries.
5. Populate the project. Use the following IntelliJ IDEA facilities, where applicable:
   - Coding assistance.
   - Code Generation.
   - Various types of navigation and search through the source code.
   - Viewing reference and generating documentation comments.
   - Look-up in external libraries.
6. Improve the quality and maintainability of your code using various types of refactoring, both common and JavaScript-specific.
7. Run your application by opening its starting HTML page in the IntelliJ IDEA default browser.
8. Debug your application.

   The JavaScript debugging functionality is incorporated in IntelliJ IDEA, so just configure the debugger, whereupon you can start the debugging session and proceed as usual: set the breakpoints, step through them, stop and resume the program, and examine it when suspended.

   > Debugging for JavaScript applications is supported only in the Firefox and Google Chrome browsers.

See Also

Concepts:

- Supported Languages

Language and Framework-Specific Guidelines:

- Node.js

Reference:

- JavaScript Libraries

Web Resources:

- Developer Community