Creation of Intention Action

This document is a work in progress.

This topic describes the conditionalOperatorConverter sample plugin that adds a new intention action to the IDEA Intensions list. In addition, the sample plugin contains a `JUnit`-based test. Basing on this information, you can develop your own plugins using similar techniques.

About Intention Actions

The IntelliJ IDEA interface analyzes your code and helps you handle the situations that may result in errors. When a possible problem is suspected, IntelliJ IDEA suggests appropriate intention actions, denoted with the special icons. For more information, refer to Intention Actions in IntelliJ IDEA Web Help.
You can view a list of all available intention actions using the Intention List provided by IntelliJ IDEA interface.

To display Intention List

1. On the toolbar, click .
   The Settings dialog box opens.
2. Under IDE Settings, click Intentions.
   This displays the list of all intention actions currently available in IntelliJ IDEA. The intention actions are grouped according to the areas of their use.
   To enable/disable an intention action, select/clear the check box to its left.
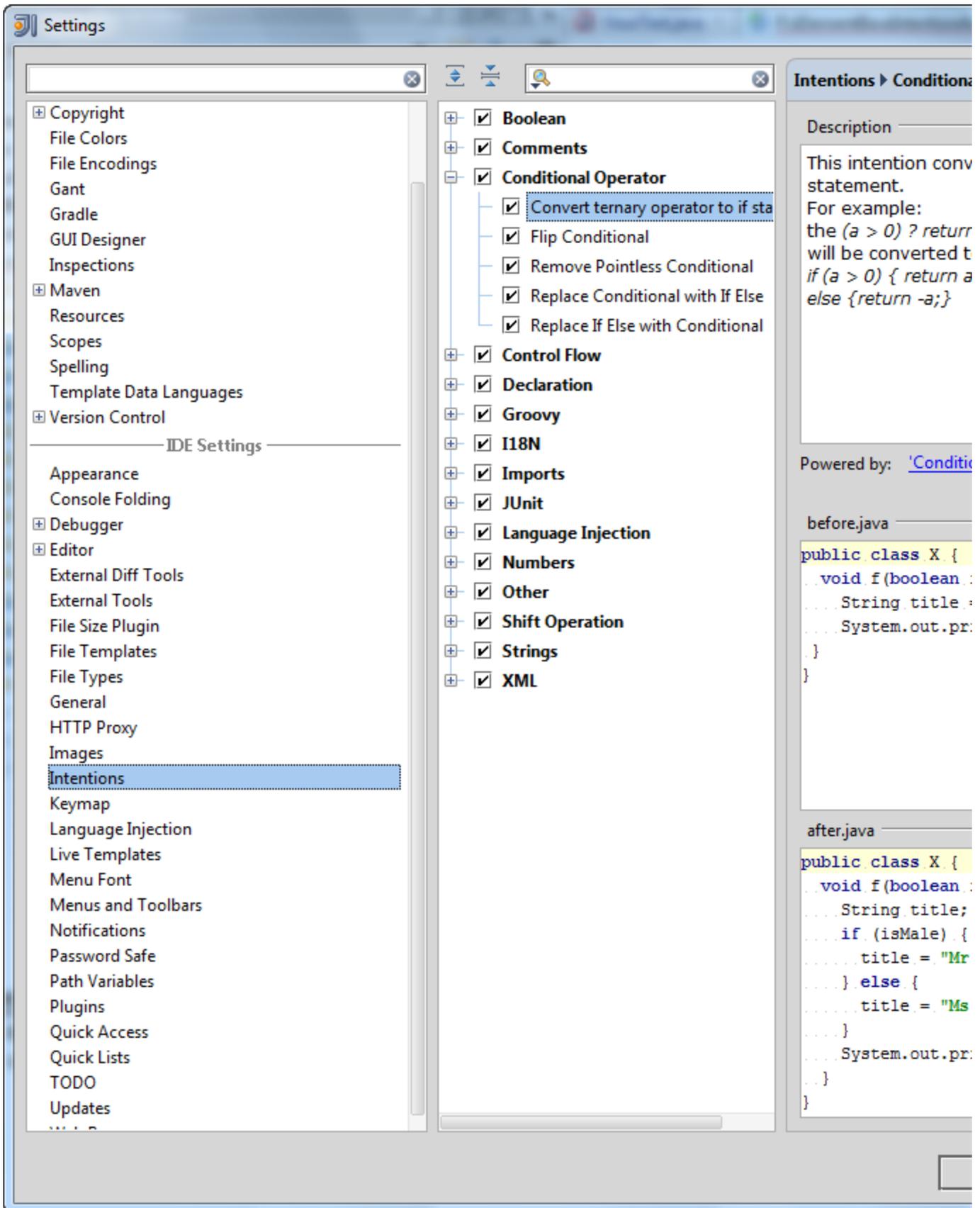
Used Techniques

The conditionalOperatorConverter sample plugin illustrates the use of the following techniques:

- How to analyze a PSI tree .
- How to find a Java token of interest in the PSI tree.
- How to invoke a quick fix action for a token element under cursor using the PsiElementBaseIntentionAction class.
- How to create a `JUnit` test for this plugin using the IdeaTestFixtureFactory class.

Sample Plugin

The ConditionalOperatorConverter sample plugin is available in the <%IDEA project folder%>/community/samples/ConditionalOperatorConvertor> directory. When launched, this plugin adds the Convert ternary operator if statement item to the Conditional Operator node in the IDEA Intensions list:

Settings

```
Copyright                              ☑ Boolean
    File Colors                        ☑ Comments
    File Encodings                     ☐ Conditional Operator
    Gant                                   ☑ Convert ternary operator to if sta
    Gradle                                 ☑ Flip Conditional
    GUI Designer                           ☑ Remove Pointless Conditional
    Inspections                            ☑ Replace Conditional with If Else
Maven                                      ☑ Replace If Else with Conditional
    Resources                          ☑ Control Flow
    Scopes                             ☑ Declaration
    Spelling                           ☑ Groovy
    Template Data Languages            ☑ I18N
Version Control                        ☑ Imports
──────────── IDE Settings ─────────    ☑ JUnit
    Appearance                         ☑ Language Injection
    Console Folding                    ☑ Numbers
Debugger                               ☑ Other
Editor                                 ☑ Shift Operation
    External Diff Tools                ☑ Strings
    External Tools                     ☑ XML
    File Size Plugin
    File Templates
    File Types
    General
    HTTP Proxy
    Images
    Intentions
    Keymap
    Language Injection
    Live Templates
    Menu Font
    Menus and Toolbars
    Notifications
    Password Safe
    Path Variables
    Plugins
    Quick Access
    Quick Lists
    TODO
    Updates
```

Intentions ▸ Conditiona

Description ─────

This intention conv
statement.
For example:
the *(a > 0) ? retur*
will be converted t
if (a > 0) { return a
else {return -a;}

Powered by: 'Conditi

before.java ─────
```
public class X {
   void f(boolean :
      String title :
      System.out.pr:
   }
}
```

after.java ─────
```
public class X {
   void f(boolean :
      String title;
      if (isMale) {
         title = "Mr
      } else {
         title = "Ms
      }
      System.out.pr:
   }
}
```

Running the Plugin
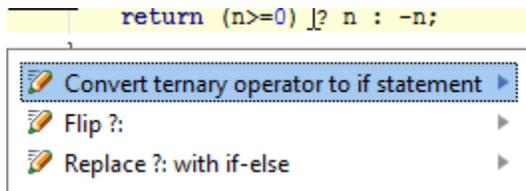
To run the sample plugin

1. Start IntelliJ IDEA and open the conditionalOperatorConvertor plugin project saved into the <%IDEA project folder%>/community/samples/conditionalOperatorConvertor directory.
2. Ensure that the project settings are valid for your environment. If necessary, modify the project settings.

    To view or modify the project settings, on the toolbar, click , and then complete the Project Structure dialog box that opens.
3. Run the plugin by choosing the Run | Run on the main menu.
    If necessary, change the Run/Debug Configurations .

How It Works?

The plugin analyzes symbols under the cursor in your code opened in the IDEA editor. If the cursor is positioned on the "?" conditional operator, IntelliJ IDEA proposes to replace this conditional (ternary) operator with the "if-then-else" statement:

In this example, the code:

```
return (n>=0) ? n : -n;
```

will be replaced with the code:

```
if ((n>=0)) {
        return n;
    } else {
        return -n;
    }
```

Testing the Plugin

The sample plugin contains the `YourTest` Java class in the testSource/testPlugin package and the test data in the <plugin directory>/testData directory.
To perform the plugin test, run the `YourTest.test()` method.

For detailed information about testing and all related procedures, refer to Testing and Testing Support in the IntelliJ IDEA Web Help.