Run/Debug Configuration: Application

An *application* run/debug configuration enables you to run applications via the `main()` method.

The dialog box consists of the following tabs:

- Configuration tab
- Code Coverage tab
- Logs tab

Click here for the description of the options that are common for all run/debug configurations.

Configuration tab

Item	Description
Main class	In this text box, specify the fully qualified name of the class to be executed (passed to the JRE). Type the class name manually or click the **Browse** button to open the **Choose Main Class** dialog box, where you can find the desired class by name or search through the project.
Program arguments	In this text box, type a list of arguments to be passed to the program in the format you would use in the command line. If necessary, click the button and type the desired arguments in the **Program Parameters** dialog box. Use the same rules as for specifying the VM options.
Working directory	In this text box, specify the current directory to be used by the running test. This directory is the starting point for all relative input and output paths. By default, the field contains the directory where the project file resides. To specify another directory, click the **Browse** button select the directory in the dialog that opens.
VM options	In this text box, specify the string to be passed to the VM for launching Cucumber tests. Usually this string contains the options such as `-mx`, `-verbose`, etc. If necessary, click and type the desired string in the **VM Options** dialog. When specifying the options, follow these rules: - Use spaces to separate individual options, for example, `-client -ea -Xmx1024m`. - If an option includes spaces, enclose the spaces or the argument that contains the spaces in double quotes, for example, `some" "arg` or `"some arg"`. - If an option includes double quotes (e.g. as part of the argument), escape the double quotes by means of the backslashes, for example, `-Dmy.prop=\"quoted_value\"`. The `-classpath` option specified in this field overrides the classpath of the module.
Environment variables	Click the **Browse** button to open the **Environment Variables** dialog box, where you can create variables and specify their values.

Item	Description
Use classpath of module	From this drop-down list, select one of the modules configured in your project. The classpath and JDK of this module will be used to run your application with the current run configuration.
Use alternative JRE	Select this check box to enable defining another JRE than the JRE used by the current project / module.
Enable capturing form snapshots	Select this check box to enable the GUI Designer to take snapshots of the GUI components, that can be afterwards converted into a form.

Code Coverage tab

Use this tab to configure code coverage monitoring options.

Item	Description
Choose code coverage runner	Select the desired code coverage runner. By default, IntelliJ IDEA uses its own coverage engine with the Sampling mode. You can also choose JaCoCo or Emma for calculating coverage.
Sampling	Select this option to measure code coverage with minimal slow-down.
Tracing	Select this option to collect accurate branch coverage. This mode is available for the IntelliJ IDEA code coverage runner only.
Track per test coverage	Select this check box to detect lines covered by one test and all tests covering line. If this check box is selected, becomes available on the toolbar of the coverage statistic pop-up window. This option is only available for the *Tracing mode* of code coverage measurement for the testing run/debug configurations. Refer to the section Viewing Code Coverage Results.
Merge data with previous results	When you run your unit testing or application configuration several times, use this item to calculate statistics in the **Project View**, taking into account the statistics of each time you have run the configuration. Finally, the line is considered *covered* if it is covered at least once.
Packages and classes to record code coverage data	Click and buttons to specify classes and packages to be measured. You can also remove classes and packages from the list by selecting them in the list and clicking the button.
Enable coverage in test folders.	If this check box is selected, the folders marked as test are included in the code coverage analysis.

Logs tab

Use this tab to specify which log files generated while running or debugging should be displayed in the console, that is, on the dedicated tabs of the Run or Debug tool window.

Item	Description
Is Active	Select check boxes in this column to have the log entries displayed in the corresponding tabs in the Run tool window or Debug tool window.
Log File Entry	The read-only fields in this column list the log files to show. The list can contain: ▪ Full paths to specific files. ▪ Ant patterns that define the range of files to be displayed. ▪ Aliases to substitute for full paths or patterns. These aliases are also displayed in the headers of the tabs where the corresponding log files are shown. If a log entry pattern defines more than one file, the tab header shows the name of the file instead of the log entry alias.
Skip Content	Select this check box to have the previous content of the selected log skipped.
Save console output to file	Select this check box to save the console output to the specified location. Type the path manually, or click the browse button and point to the desired location in the dialog that opens.
Show console when standard out changes	Select this check box to activate output console and bring it forward, if associated process writes to Standard.out.
Show console when standard error changes	Select this check box to activate output console and bring it forward, if associated process writes to Standard.err.
+	Click this button to open the Edit Log Files Aliases dialog where you can select a new log entry and specify an alias for it.
	Click this button to edit the properties of the selected log file entry in the Edit Log Files Aliases dialog.
−	Click this button to remove the selected log entry from the list.
	Click this button to edit the select log file entry. The button is available only when an entry is selected.

Toolbar

Item	Shortcut	Description
+	Alt+Insert	Click this button to add new configuration to the list.
−	Alt+Delete	Click this button to remove the selected configuration from the list.

Item	Shortcut	Description
	Ctrl+D	Click this button to create a copy of the selected configuration.
	Edit defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
 or	Alt+Up or Alt+Down	Use these buttons to move the selected configuration or group of configurations (folder) up and down in the list. The order of configurations or folders in the list defines the order, in which configurations appear in the **Run/Debug** drop-down list on the main toolbar.
		Use this button to create a new folder. If one or more run/debug configurations have the focus, then the selected run/debug configurations are automatically moved to the newly created folder. If only a category has the focus, then an empty folder is created. Move run/debug configurations to a folder using drag-and-drop, or  buttons.

Common options

Item	Description
Name	In this text box, specify the name of the current run/debug configuration. This field does not appear for the default run/debug configurations.
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Share	Select this check box to make the run/debug configuration available to other team members. If the directory-based project format is used, the settings for a run/debug configuration are stored in a separate xml file in the `.idea\runConfigurations` folder if the run/debug configuration is shared and in the `.idea\workspace.xml` file otherwise. If the file-based format is used, the settings are stored in the `.ipr` file for shared configurations or in the `.iws` file for the ones that are not shared. This check box is not available when editing the run/debug configuration defaults.

Item	Description
Before launch	Specify which tasks should be carried out before starting the run/debug configuration. The specified tasks are performed in the order that they appear in the list. **Item** | **Keyboard shortcut** | **Description**

Item	Item	Keyboard shortcut	Description
	+	Alt+Insert	Click this icon to add a task to the list. Select the task to be added: ■ **Run External tool.** Select this option to run an application which is external to IntelliJ IDEA. In the dialog that opens, select the application or applications that should be run. If the necessary application is not defined in IntelliJ IDEA yet, add its definition. For more information, see Configuring Third-Party Tools and External Tools. ■ **Make.** Select this option to have the project or module compiled. The Make Module command will be carried out if a particular module is specified in the run/debug configuration, and the Make Project command otherwise. If an error occurs during the compilation, IntelliJ IDEA won't attempt to start the run/debug configuration. ■ **Make, no error check.** The same as the **Make** option but IntelliJ IDEA will try to start the run/debug configuration irrespective of the compilation result. ■ **Build Artifacts.** Select this option to have an artifact or artifacts built. In the dialog that opens, select the artifact or artifacts that should be built. See also, Configuring Artifacts. ■ **Run Another Configuration.** Select this option to have another run/debug configuration executed. In the dialog that opens, select the configuration to be run. ■ **Run Ant target.** Select this option to have an Ant target run. In the dialog that opens, select the target to be run. For more information, see Ant. ■ **Generate CoffeeScript Source Maps.** Select this option to have the source maps for your CoffeeScript sources generated. In the dialog that opens, specify where your CoffeeScript source files are located. For more information, see CoffeeScript Support. ■ **Run Maven Goal.** Select this option to have a Maven goal run. In the dialog that opens, select the goal to be run. For more information, see Maven. ■ **Run Remote External tool**: Add a remote SSH external tool. Refer to the section Remote SSH External Tools for details.

See Also

Concepts:

- Run/Debug Configuration

Procedures:

- Creating and Editing Run/Debug Configurations

Reference:

- Run/Debug Configurations
- Debugger

Web Resources:

- Developer Community