<https://www.jetbrains.com/help/idea/product-educational-tools.html>

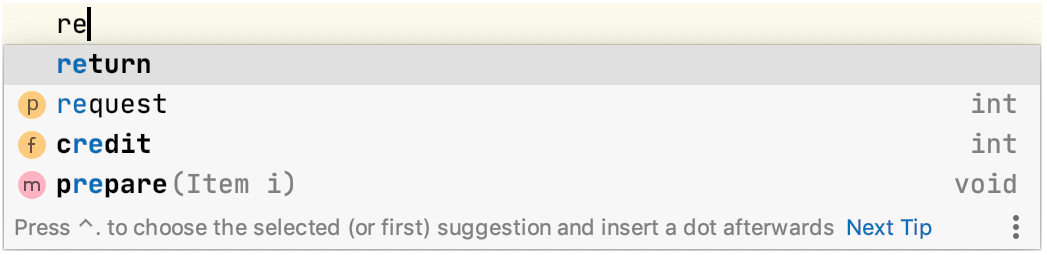
To expand selection, press **Ctrl+W**. Each time you press **Ctrl+W**, the selection expands to other areas of code.

For example, the selection expands from a method name to the expression calling this method, then to the whole statement, then to the containing block, and so on.

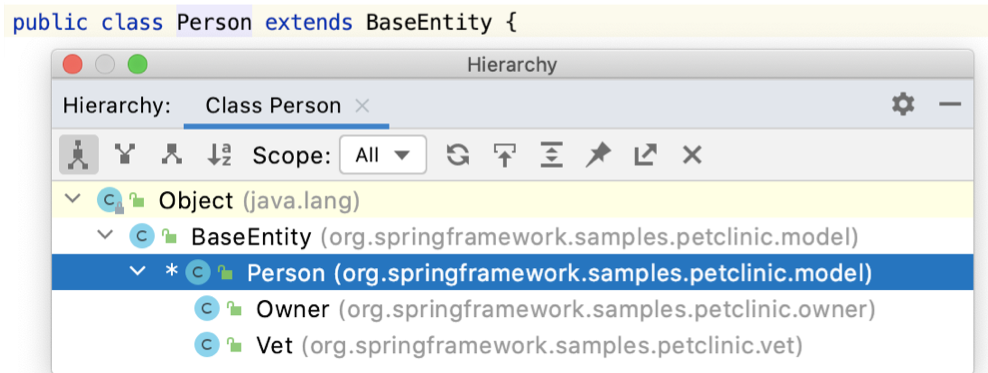
To select multiple fragments in the column mode **Alt+Maj+Insérer**, press and hold **Ctrl+Alt+Shift** (on Windows and Linux) / ⌘⌥⇧ (on macOS), and drag the mouse:



When you use basic code completion **Ctrl+Espace**, type any characters that exist anywhere in an identifier.

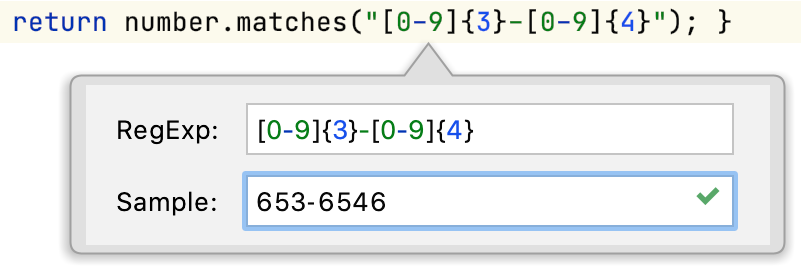


To see the inheritance hierarchy for the selected class, press **Ctrl+H** (Navigate | Type Hierarchy ). You can also invoke the hierarchy view directly from the editor to see the hierarchy for the currently edited class.

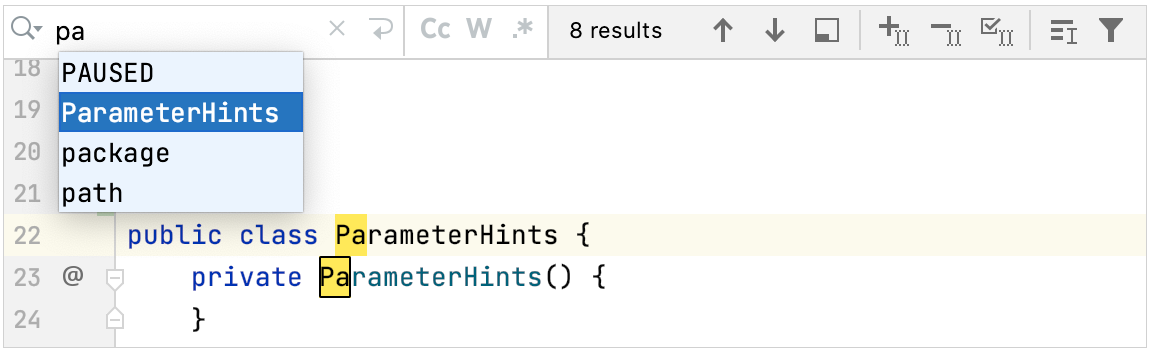


To verify that your regular expression is correct, place the caret within the expression you want to check, press **Alt+Entrée**, and select Check RegExp.

In the popup, type a sample string that should match your regular expression. The icon shows that the match occurred. The icon shows that there is no match or your expression contains a mistake.



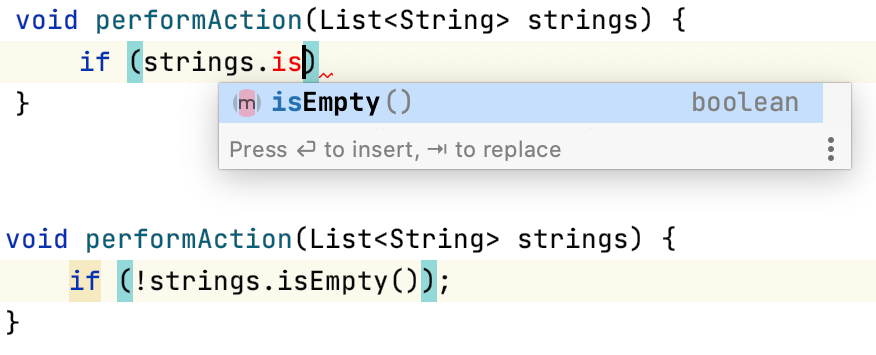
Basic code completion **Ctrl+Espace** is available in the search field when you search for text in the current file Ctrl+F, so there is no need to type the entire string.



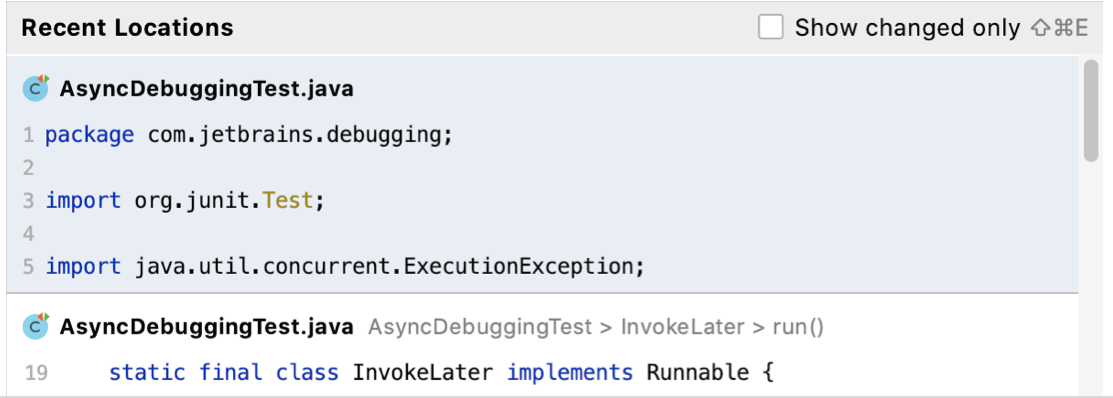
If you position the caret at a symbol and press **Ctrl+Alt+Maj+T,** the list of refactorings that applicable to the current context will open



If you accept a completion suggestion by pressing the exclamation mark (**!**), this expression will be negated.

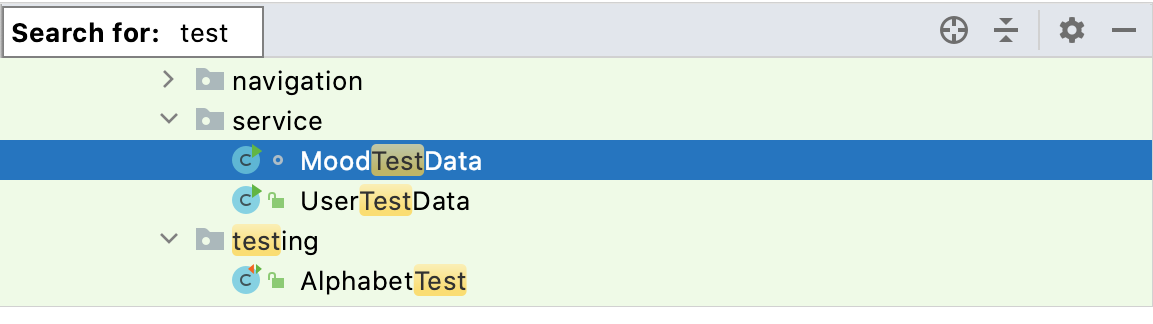


Press **Ctrl+Maj+E** to get a list of recently viewed or changed code fragments.



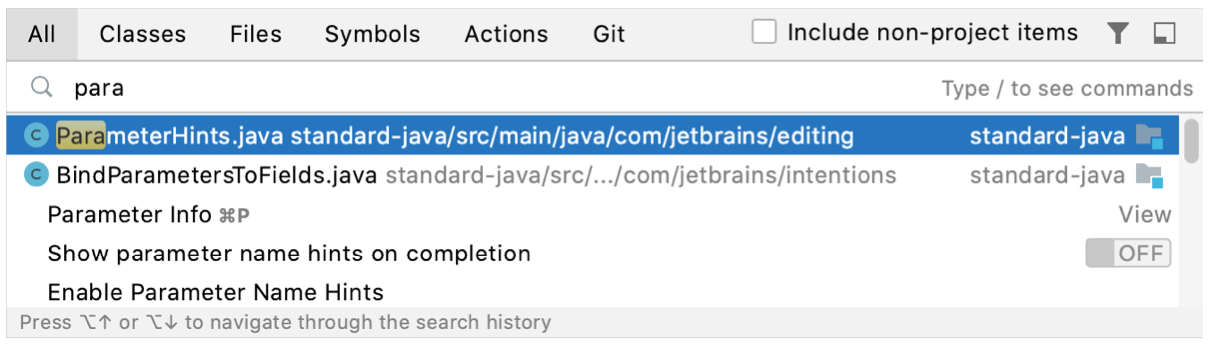
Press **Ctrl+Y** in the editor to delete the whole line at the caret.

Speed search is available in all tree views: start typing, and you'll quickly locate the necessary item.



Press **Shift** twice to search for files, actions, classes, symbols, UI elements, Git branches and comments across your project. Pressing double Shift again, will extend the search to non-project items.

Use tabs or direct shortcuts **Ctrl+N** for classes, **Ctrl+Maj+N** for files, **Ctrl+Alt+Maj+N** for symbols, and **Ctrl+Maj+A** for actions to narrow your search results.



You can use the Extract Variable refactoring on incomplete statements. Press **Ctrl+Alt+V** and choose an expression.

To make your printouts **(File | Print)** nice and informative, use keywords.

For example, use **$DATE$** and **$TIME$** to specify the exact date and time of the printout.

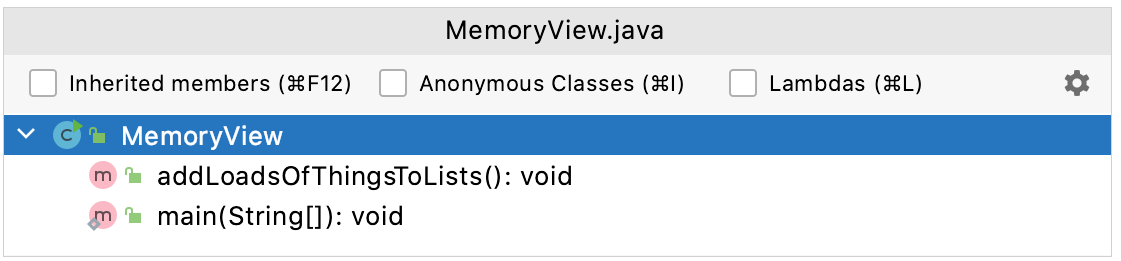
Invoking Type-matching code completion (**Ctrl+Maj+Espace)** twice when a collection type is expected will search for arrays with the same component type and suggest converting them using the **Arrays.asList()** call.



You can quickly navigate in the currently edited file with **Ctrl+F12 (Navigate | File Structure ).**

File structure shows the list of members of the current class. To navigate to an element, select the element and press **Entrée or F4**.

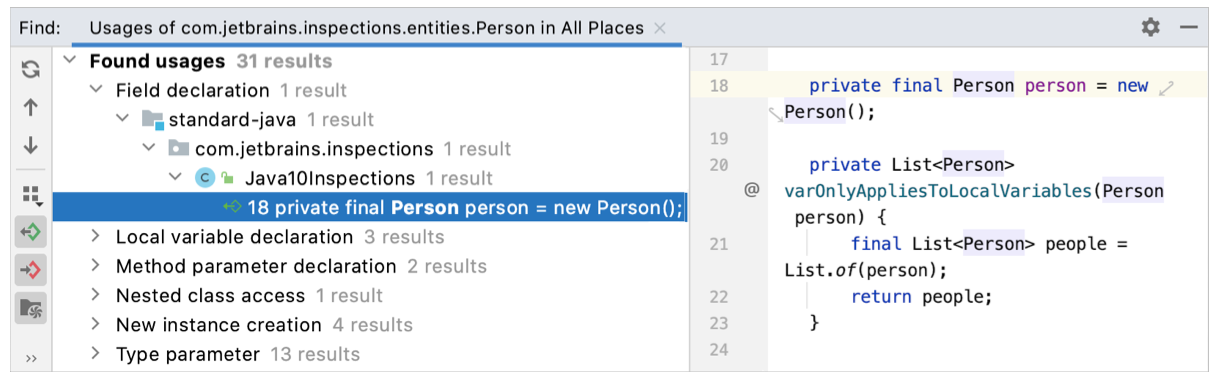
To easily locate an item in the list, start typing its name.



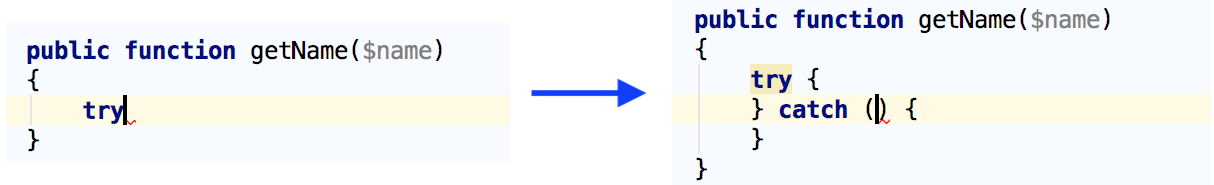
Adjust the IntelliJ IDEA Community tool windows layout to make better use of your screen area.

Toggle between the vertical and side-by-side placement of the tool windows by pressing **Ctrl+Click** (on Windows and Linux) / ⌘+Click (on macOS) on the splitter

Press **Alt+F7** to quickly locate all occurrences of code referencing the symbol at the caret, no matter if the symbol is a part of a class, method, field, parameter, or another statement.



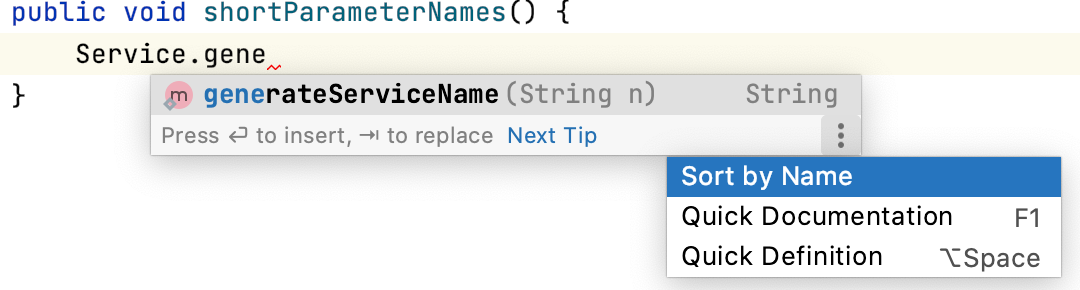
You can create code constructs using statement completion. Start typing a method declaration, a method call or a statement such as **if, do -while, try -catch,** or **return**. **Press Ctrl+Maj+Entrée** to complete the statement into a syntactically correct construct.



You can sort completion suggestions by relevance or alphabetically.

To sort alphabetically, select the Sort by **Name option**.

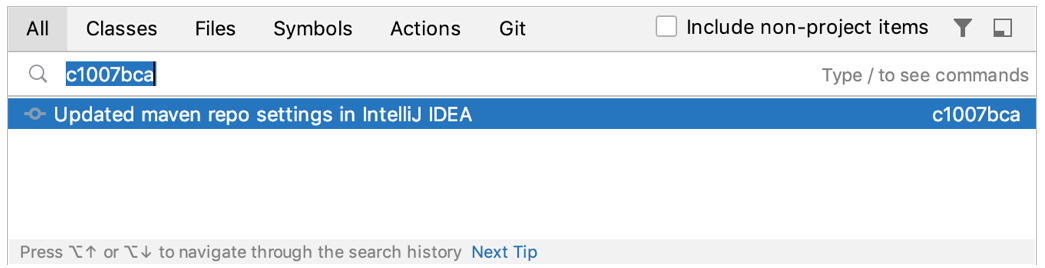
To sort by relevance, clear the Sort by **Name checkbox**.



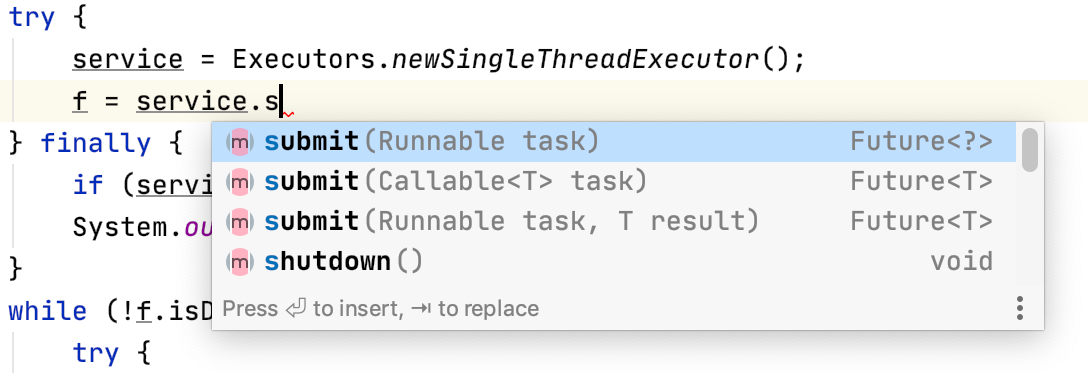
When using Code Completion, you can accept the currently highlighted selection in the popup list by pressing **Tab**.

Unlike accepting with **Entrée**, the selected name will overwrite the rest of the name to the right of the caret. This can be especially useful for replacing one method or variable name with another.

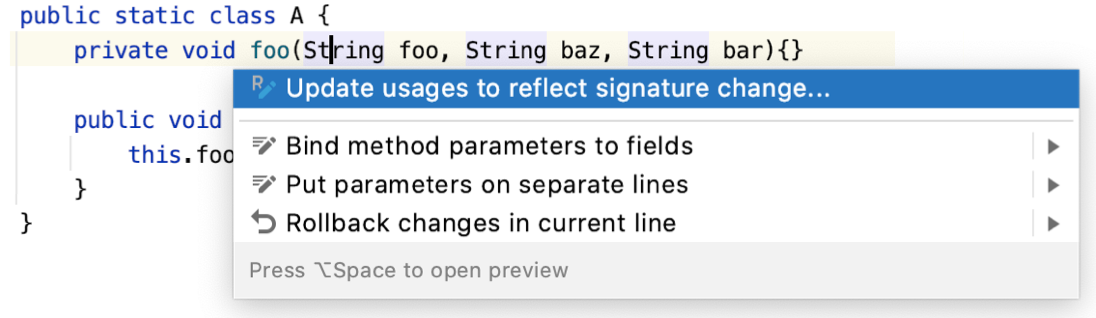
Press **Shift** twice and search for a Git branch, tag, commit hash or message to jump to it in the Log view:



Code Completion **Ctrl+Espace** helps you quickly complete code statements. It works as you type and gives a list of suggestions available from the current caret position:



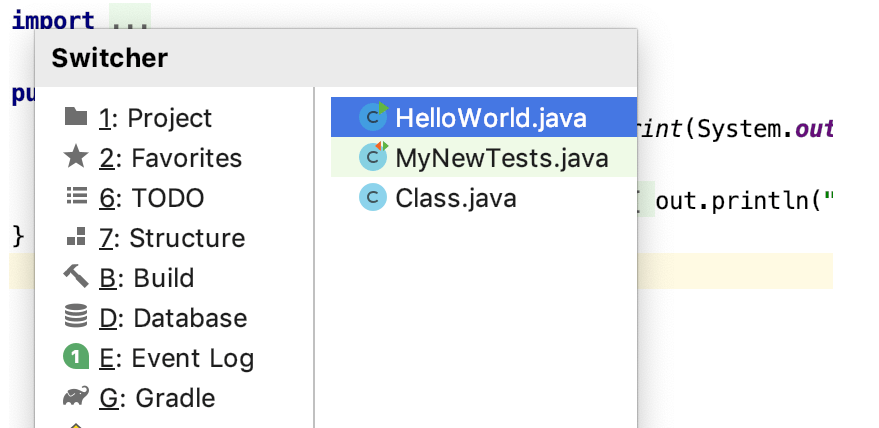
You can move method parameters in both declaration and invocation with **Ctrl+Alt+Maj+Gauche** and **Ctrl+Alt+Maj+Droite**. Moreover, you can propagate such move in a method declaration to the method invocation: press **Alt+Entrée** after the move and choose Update usages to reflect signature change.



To open a file in the editor at a particular line, press **Ctrl+Maj+N (Navigate | File ),** start typing the filename, choose from the suggestion list, and then type : followed by the line number.

To switch between opened files and tool windows, use the Switcher **Ctrl+Tab**. For navigation, press and hold Ctrl (on Windows and Linux) / ⌘ (on macOS) and use the **Up** and **Down** arrow keys or Tab and **Shift+Tab**, and **Alt**.

Press **Delete** or **BackSpace** to close an editor tab or hide a tool window.

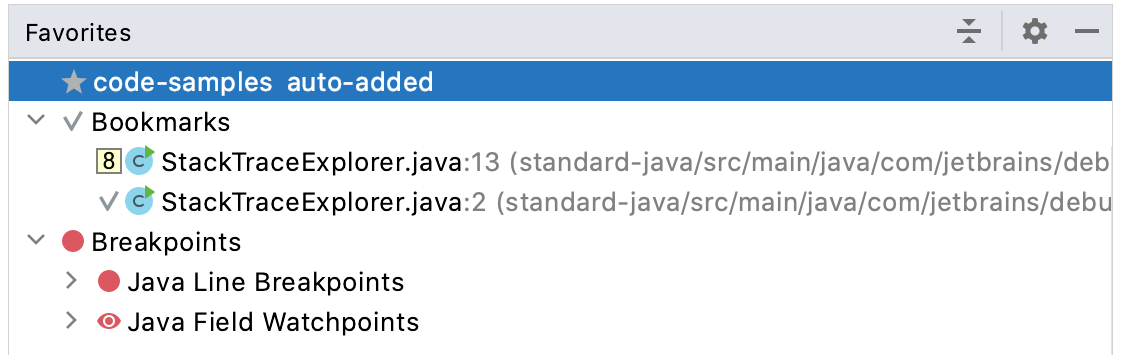


Use **Code | Inspect Code** to run code analysis for the whole project or a custom scope and examine the results in a separate window.

To reformat code according to the style defined on the **Editor | Code Style** page of the Settings/Preferences dialog **Ctrl+Alt+S,** select **Code | Reformat** **Code** from the main menu.

You can also use **Code | Optimize** Imports to organize import statements according to the project code style settings and remove unused imports.

**Alt+Maj+F** lets you add project elements (files, folders, packages, instance, and class members) to a Favorites list Alt+2, which also automatically adds your bookmarks and breakpoints:



You can generate boilerplate code such as getters and setters and implement interface methods using code completion.

Just start typing the would-be name of a method, for example, gn to generate getName() or ct to implement compareTo().



You can start referring to an Ant property or target even if it's not defined yet. Press **Alt+Entrée** and choose Create property from the list of intention actions to create the necessary tag without leaving your current editing location.

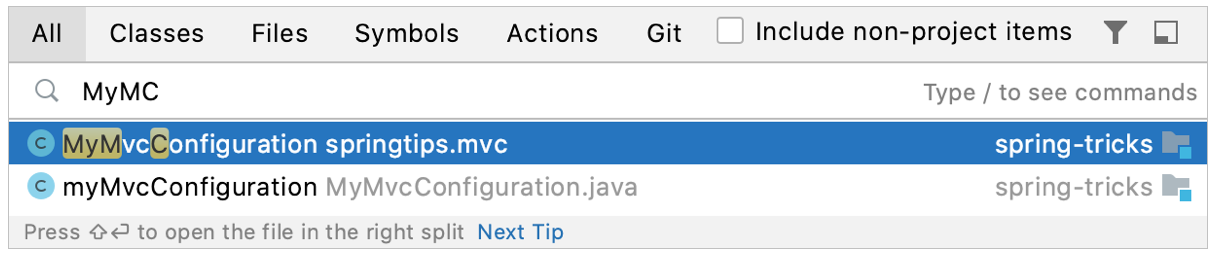
**Echap** in any tool window moves the focus to the editor.

**Maj+Echap** moves the focus to the editor and hides the current or the last active tool window.

**F12** moves the focus from the editor to the last focused tool window.

To quickly open any class, field, or function in the editor, press **Ctrl+Alt+Maj+N (Navigate | Symbol)** and start typing its name.

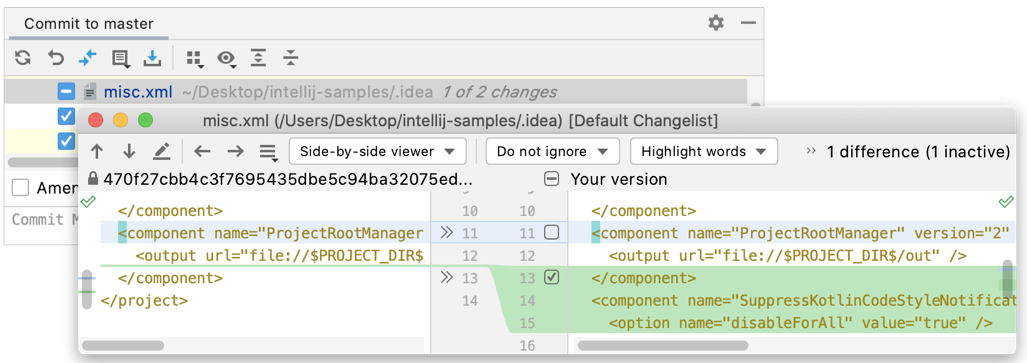
Use camel case in the Search Everywhere popup (double **Shift**) to filter the list of results when searching for a class, file, or symbol.



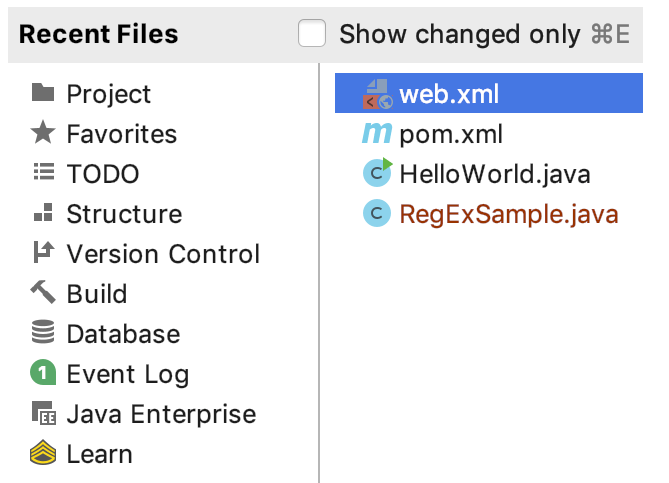
You do not need to open a file in the editor to change its line separator style. Use the Project tool window instead: select one or more files or folders, select **File | File Properties| Line Separators** from the main menu, and then choose the desired line ending style.

For a directory, new line separator applies recursively.

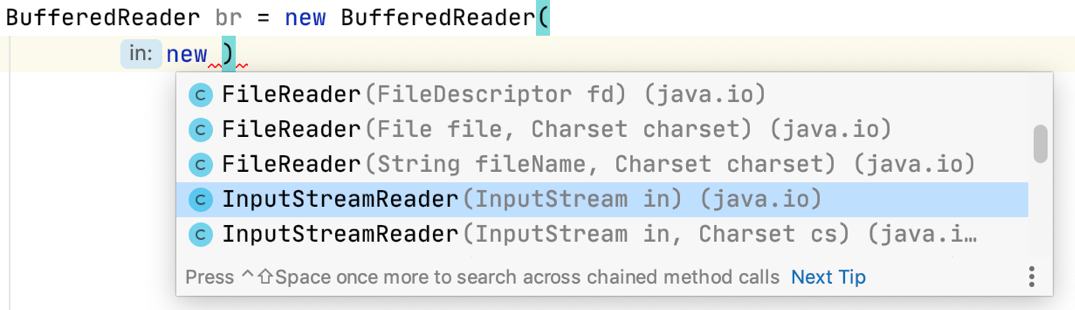
To split changes made to the same file between different commits, in the Commit tool window **Alt+0** **click** **Diff**. Select the checkbox next to each **chunk** of modified or new code that you want to commit and click Commit. Unselected changes will stay in the current changelist.



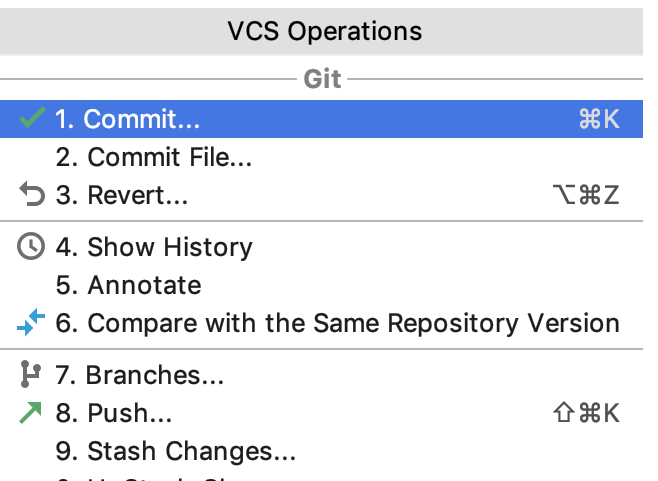
Press **Ctrl+E (View | Recent Files**) to view the list of recently opened files.



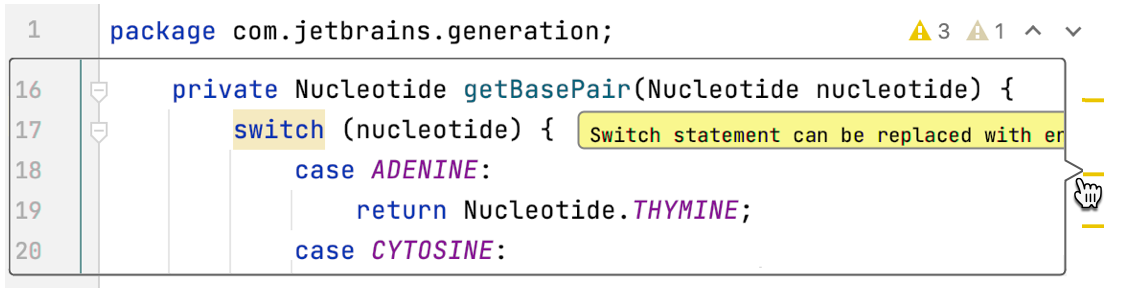
Use the Smart code **completion Ctrl+Maj+Espace** after the new keyword to instantiate an object of the expected type.



To access all VCS-related commands available in the current context, press **Alt+`.**

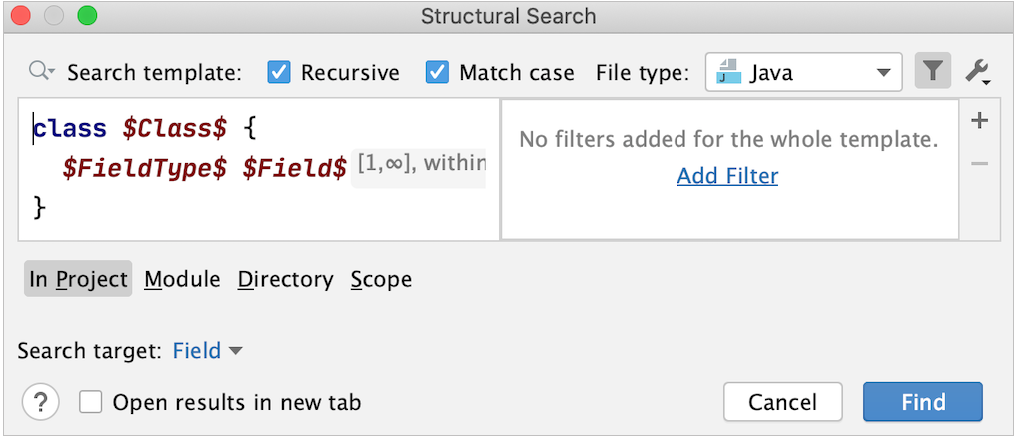


To preview code without actually scrolling to it, hover your mouse pointer over a warning, error stripe, or just some section of source code on the scrollbar, and you will see a lens:



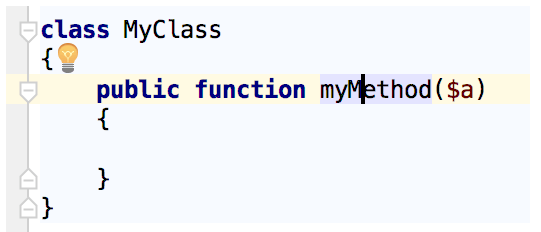
To disable the lens, right-click the code analysis marker at the top of the scrollbar and clear the **Show code lens on scrollbar hover** checkbox.

To search for a code pattern or a grammatical construct, select **Edit | Find | Search Structurally.**

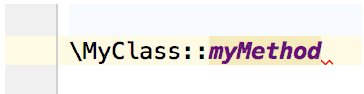


Use the **Edit | Copy | Copy** Reference action to insert a reference to a field/method/class/file into the current position in the editor.

Position the caret within the **myMethod** method name and press **Ctrl+Alt+Maj+C**:

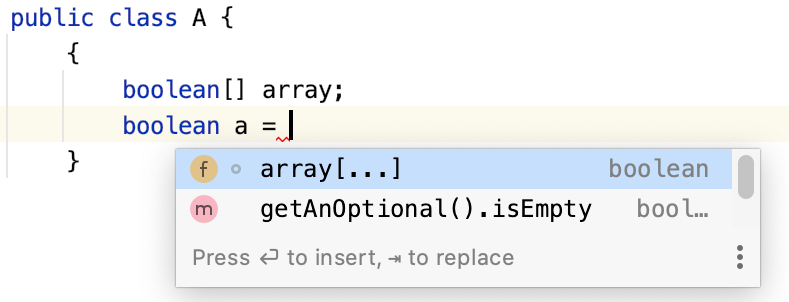


To paste the reference, press **Ctrl+V:**

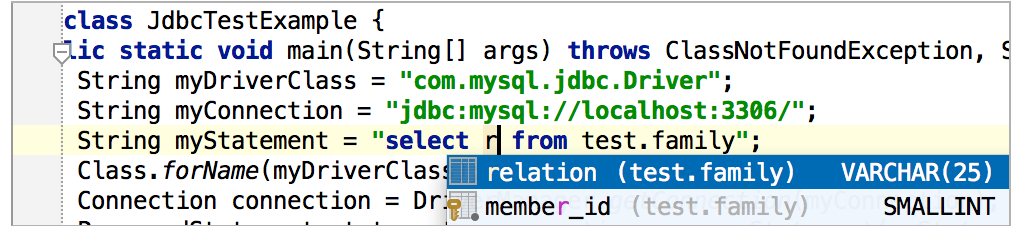


You can also copy references in the **Go to Class/Go to Symbol/Go to File** dialogs. Press **Ctrl+C** on any element in the lookup list.

nvoking Type-matching code completion (**Ctrl+Maj+Espace**) twice when there's an array of the expected type in the context will suggest getting an element from this array.



You can inject SQL into a string literal (**Alt+Entrée** | **Inject language or reference | <SQL dialect>)** and then use coding assistance for SQL.



When using code completion, you can accept the current selection in the suggestions list with **Ctrl+Maj+Entrée**. IntelliJ IDEA Community will not only insert the selected string, but also turn the current code construct into a syntactically correct one (balance parentheses, add missing braces and semicolons, and so on).

Press **F2** or **Maj+F2** to jump to the next or previous error respectively in the current file.

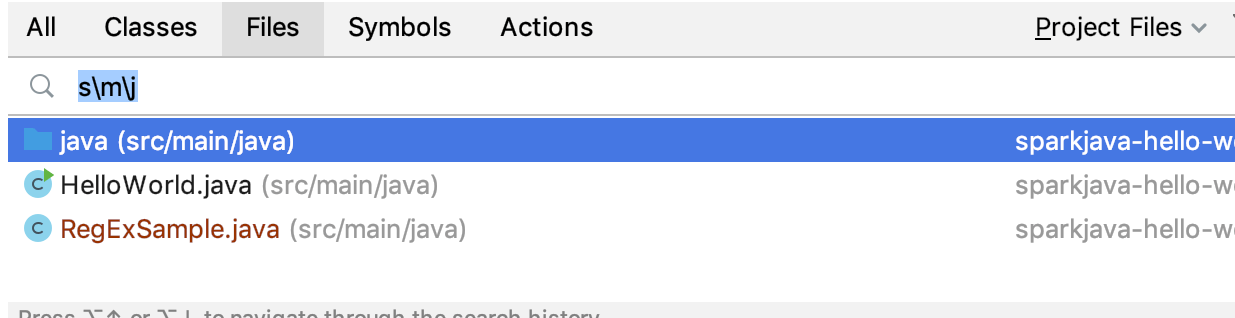
If you notice that IntelliJ IDEA Community works slowly, consider reducing the number of folders under antivirus protection. This may significantly improve performance.

You can exclude any file from your project so that it is ignored by indexing, inspection, and code completion.

In the **Project** tool window, right-click the file you want to exclude and choose **Mark as plain text** from the context menu. If necessary, you can always revert the file to its original type by the **Mark as <file type>** command.

When the **Show suggestions as you type** option is enabled for code completion, **Ctrl+Bas** and **Ctrl+Haut** will close it and move the caret down or up in the editor.

You can jump to a file located in a deeply nested directory by pressing **Ctrl+Maj+N** and typing several characters of the enclosing directories and filename. Use either a slash or a backslash as a delimiter.

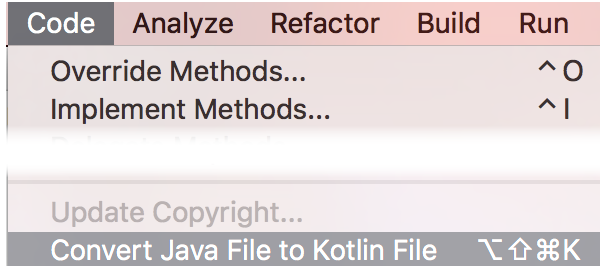


Press **Ctrl+F** to display the search pane. Press **Ctrl+R** to add another field where you can type the replace string.

In the **Find in Files** dialog, you can switch to replace by pressing **Ctrl+Maj+R**. Similarly, press **Ctrl+Maj+F** to hide the Replace with field and switch to regular search.

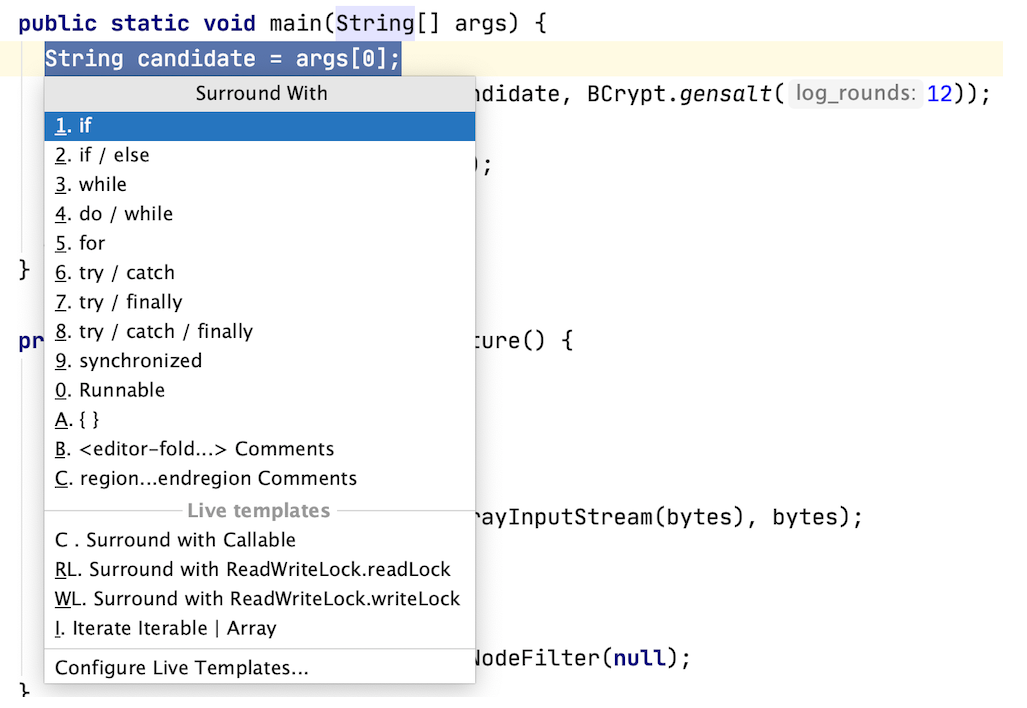
One can easily convert any Java class to the Kotlin one with the same semantics.

To do that, just choose **Code | Convert Java File to Kotlin** File on the main menu:



To quickly wrap a code block in useful constructs, select it in the editor and press **Ctrl+Alt+T (Code | Surround With** ).

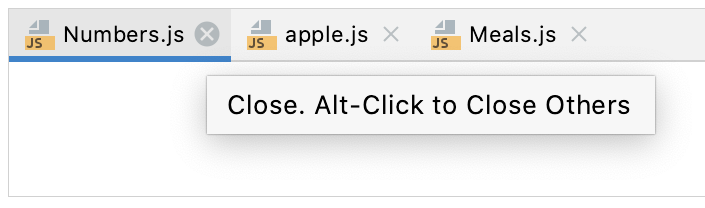
The list of available options or wrappers is context-sensitive and depends on the language. For example, you can surround **html** blocks with tags, and so on.



To compare active editor with the Clipboard contents, right-click anywhere in the editor and choose **Compare with Clipboard** from the context menu.

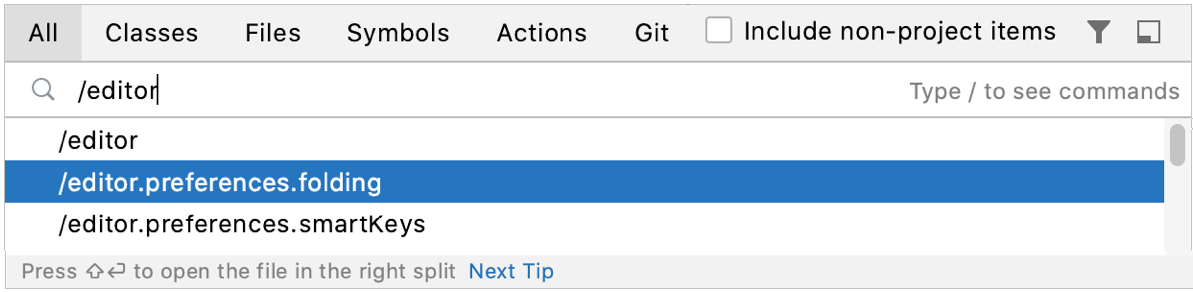
Press **Ctrl+D** in the editor to duplicate the selected block, or the current line when no block is selected.

To close all editor tabs except the current one, keep **Alt** (on Windows and Linux) / ⌥ (on macOS) pressed and click the cross icon for the current editor tab.



Enter "**/**" in the search field of the Search Everywhere (**Shift** twice) window to search for a list of settings, their options, and plugins.

You can also search for the URL mappings entering "**/**" before the part of the URL mapping you are searching for.



If you don't like the color of your code, press **Ctrl+Maj+A**, type **Jump**, and click **Jump to Colors and Fonts**. You will be taken to the settings page where you can modify the color of the code element at the caret.

Invoking Type-matching code completion (**Ctrl+Maj+Espace**) twice will search for chained expressions of the expected type.

Press **F2/ Maj+F2** to jump between the highlighted syntax errors.

Press **Ctrl+Alt+Haut/ Ctrl+Alt+Bas** to jump between error messages or search results.

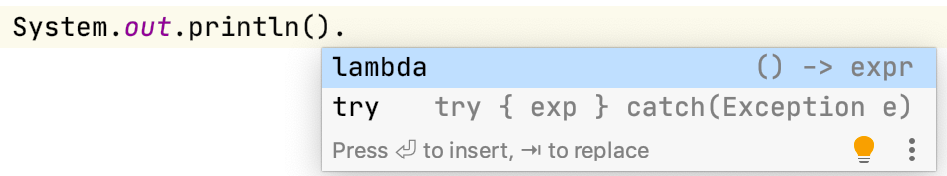
To skip warnings, right-click the validation side bar / marker bar and **choose Go to high priority problems only**.

To scroll a file horizontally, turn the mouse wheel while keeping **Shift** pressed.

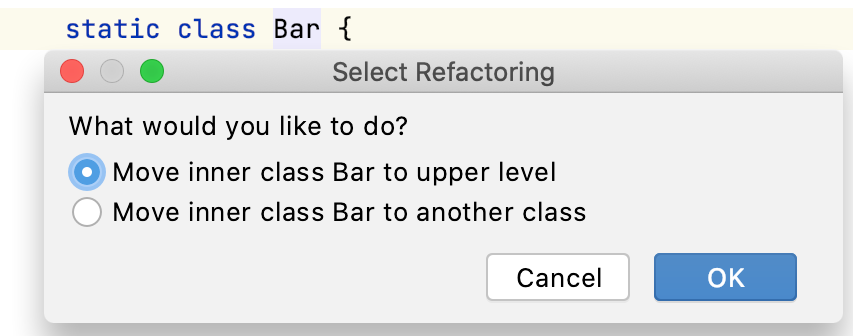
Press **Ctrl+Maj+V** to select the text fragment that you have previously copied to the clipboard.

Postfix code completion is applied from right to left to avoid backward caret jumps when coding. Type a dot . after the code fragment that you want to change and select the desired option.

To see the suggestion list with postfix templates, you can also press **Ctrl+J.**



When you invoke the Move refactoring **F6** on an inner class that is declared **static**, you are prompted to either make it a top-level class, or move it to another class.



Use shortcuts to comment and uncomment lines and blocks of code:

* **Ctrl+/**: for single line comments (**//...**)
* **Ctrl+Maj+/**: for block comments (**/\*...\*/**)

When using Code Completion, you can select the first item by pressing **Ctrl+**..

IntelliJ IDEA Community will insert the selected item followed by a dot or **->** depending on the current context.

You can use the Preview area of the Find in Files dialog (**Ctrl+Maj+F**) for quicker search without leaving the dialog. The **Preview** dialog displays the first 100 results.

Use your favorite shell from the built-in Terminal.

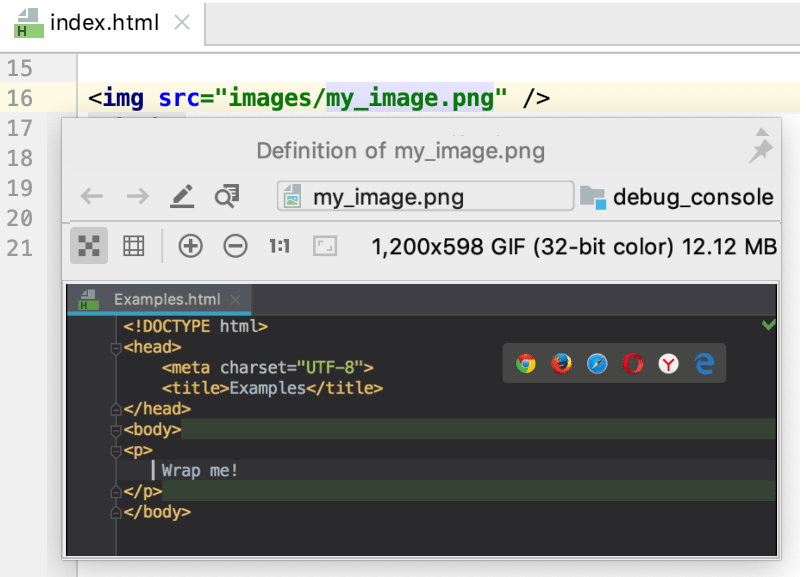
In the **Settings/Preferences** dialog **Ctrl+Alt+S**, go to Tools | Terminal and specify the path to your shell executable.

The **Code | Move Statement Up/Down** actions are useful for reorganizing code lines, for example for bringing a variable declaration closer to the variable usage.

Select a code fragment and press **Ctrl+Maj+Haut** or **Ctrl+Maj+Bas.**

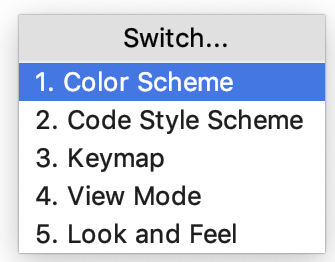
When nothing is selected in the editor, the line at the caret position will be moved.

To preview a referenced image in a popup instead of in a separate editor tab, press **Ctrl+Maj+I.**



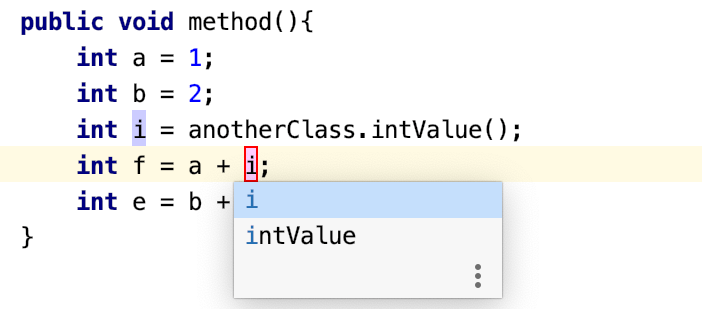
You can invoke the **Quick Definition Viewer** **Ctrl+Maj+I** for items in code completion lists and the class, file, or symbol navigation commands.

You can apply a different code style or coloring scheme or keymap with a single keystroke right from the editor. Press **Ctrl+`** (**View | Quick Switch Scheme**) to specify the scheme you want to switch to.



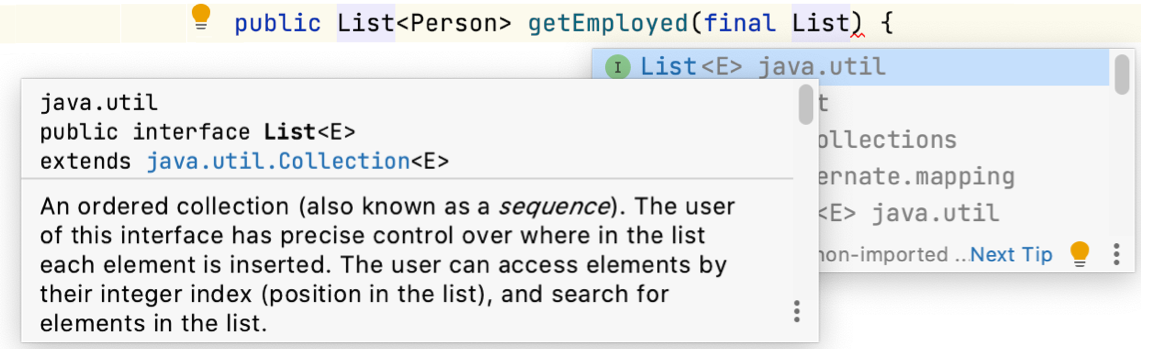
You can easily rename your classes, methods, and variables with automatic correction of all places where they are used.

Position the caret at the symbol you want to rename, and press **Maj+F6** (**Refactor | Rename** ). Type the new name and press **Entrée**.

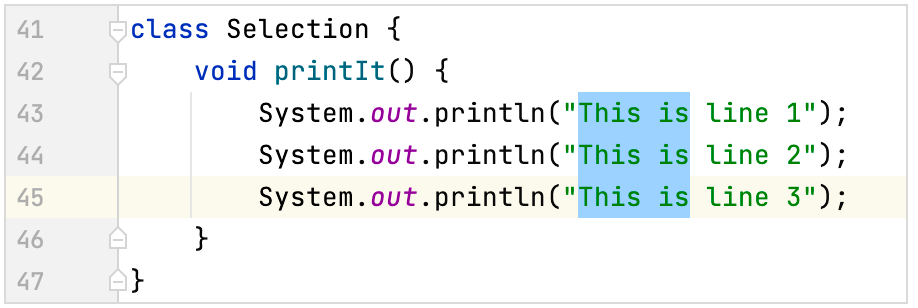


Press **Maj+F6** again to open the **Rename** dialog that will allow you to search for the provided text in comments and strings.

You can use **Ctrl+Q** (**View | Quick Documentation** ), **Ctrl+P** (**View | Parameter Info** ), **Ctrl+B** (**Navigate | Declaration** ), and similar shortcuts not only in the editor but also in the suggestions list while using code completion.



To select a rectangular piece of code, drag your mouse while pressing **Alt** (on Windows and Linux) / ⌥ (on macOS).

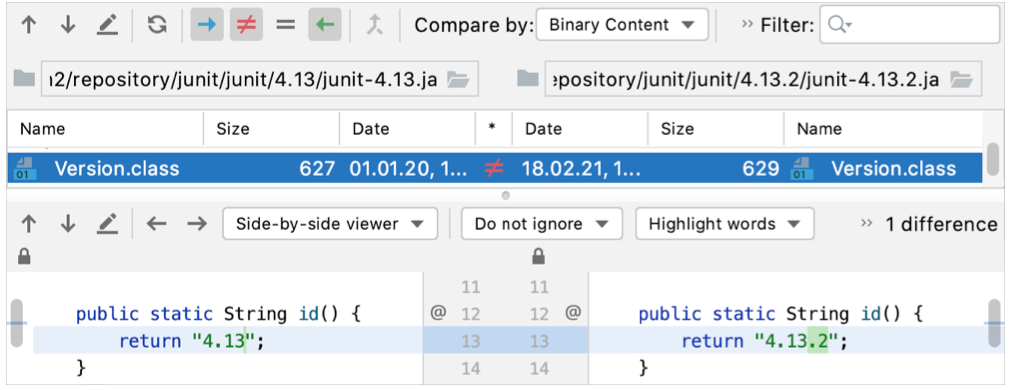


You can easily override methods of the base class by pressing **Ctrl+O** (**Code | Override Methods** ).

To implement methods of the interfaces (or of the abstract base class) that the current class implements, press **Ctrl+I** (**Code | Implement Methods** ).

To compare two **.jar** files or even files inside a **.jar** archive, select them in the Project tool window and press **Ctrl+D**.

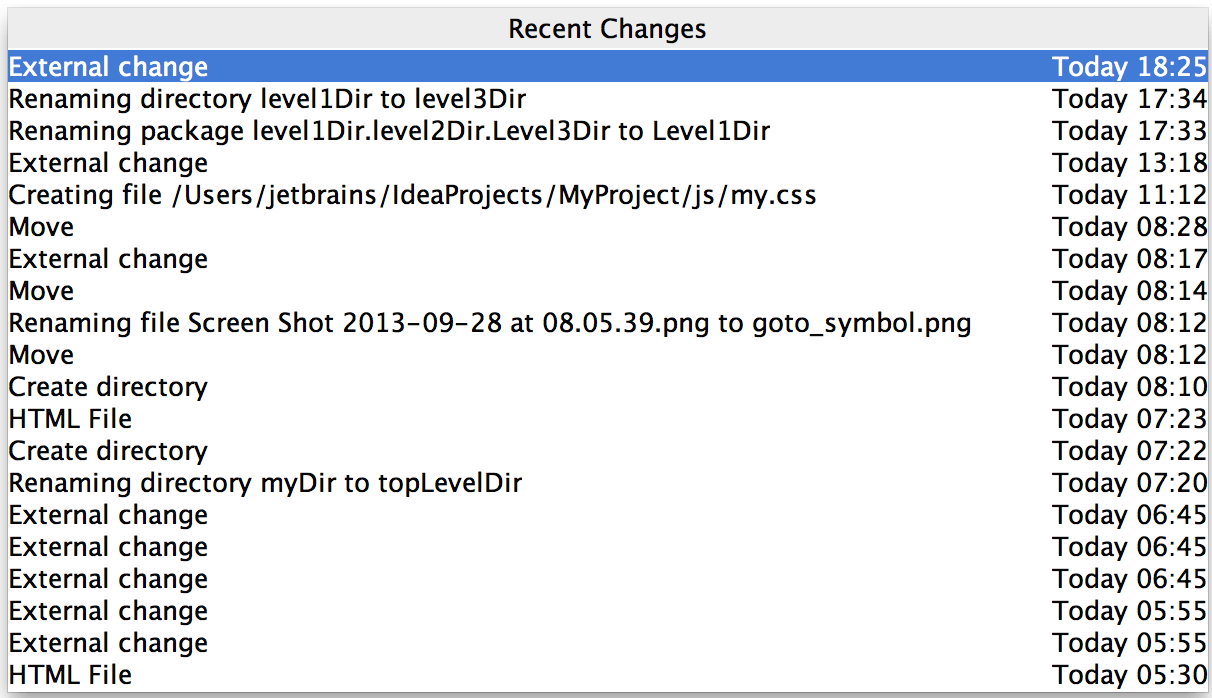
The **Compare Archives** feature is integrated with the Java Bytecode Decompiler and allows you to see what exactly has changed between two different versions of a library.



To open your browser with the documentation page for the element at the caret, press **Maj+F1** (**View | External Documentation** ).

To use this feature, a browser must be selected on the **Tools | Web Browsers** page of the **Settings/Preferences** dialog **Ctrl+Alt+S**

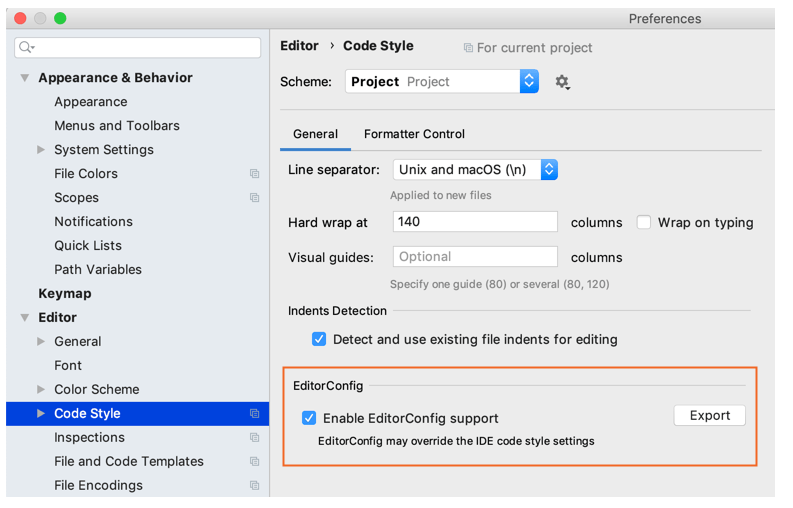
Use **Alt+Maj+C** to quickly review recent changes to the project



You can override IDE code style settings with settings from EditorConfig (https://editorconfig.org/ ).

To use **EditorConfig** settings, you must install and enable the EditorConfig plugin. To configure the style in EditorConfig, open settings (**Ctrl+Alt+S** ), navigate to Editor | Code Style and select the **Enable EditorConfig support** checkbox.

To export the current IDE code style settings into the **.editconfig** file, click **Export**.

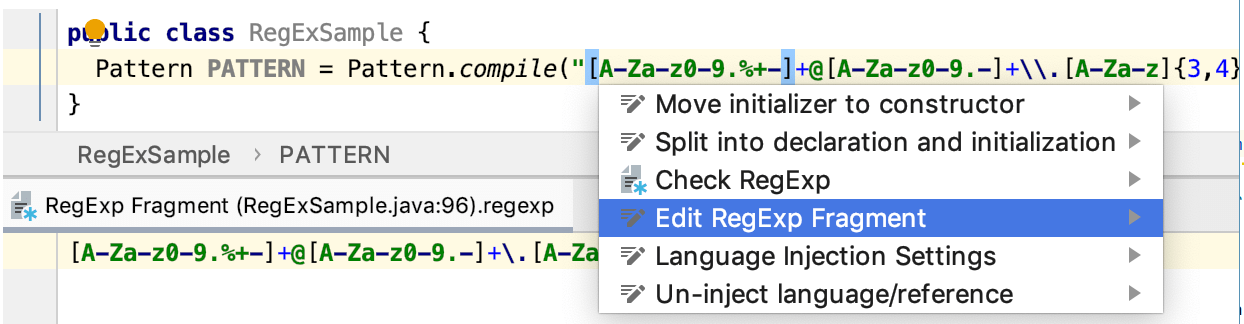


To quickly find and run an inspection, press **Ctrl+Alt+Maj+**I and start typing the name of the inspection or its group. Choose an inspection from the suggestion list and specify the scope.

You can edit a language injection in your code using a dedicated editor.

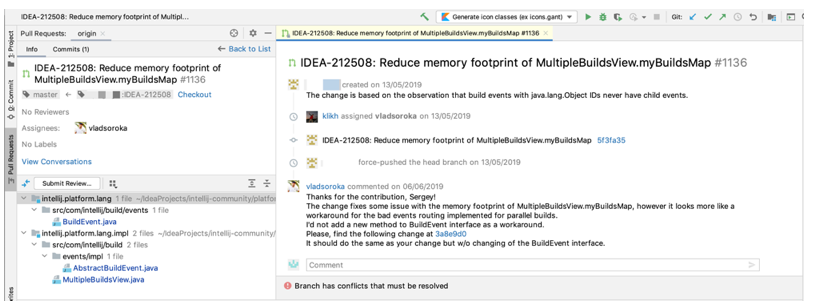
For example, to edit a regular expression, start typing it, press **Alt+Entrée** and choose **Edit RegExp Fragment**. The regular expression opens in a separate tab in the editor, where you can type backslashes as is.

All changes are synchronized with the original regular expression, and escape characters are presented automatically. When ready, press **Echap** to close the regular expression editor.

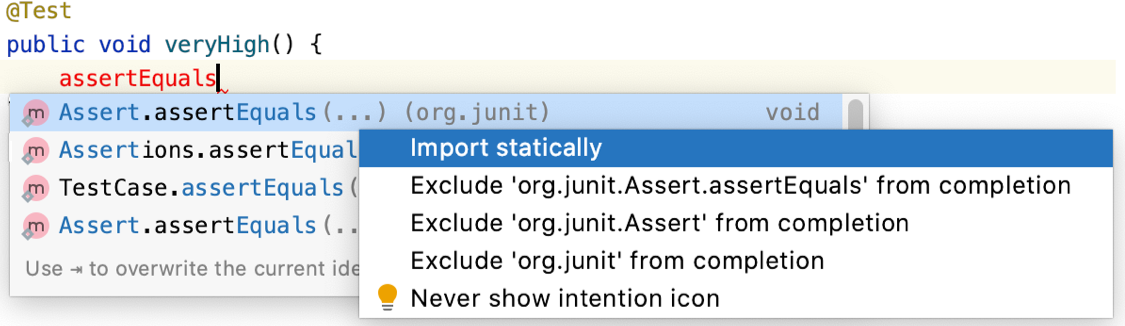


To compare any two files or folders, select them in the Project tool window and press **Ctrl+D**.

Manage incoming GitHub pull requests directly from IntelliJ IDEA Community: from the main menu select **Git | GitHub | View Pull Requests**. IntelliJ IDEA Community lets you assign and merge pull requests, view the timeline and inline comments, submit comments and reviews, and accept changes without leaving the IDE:

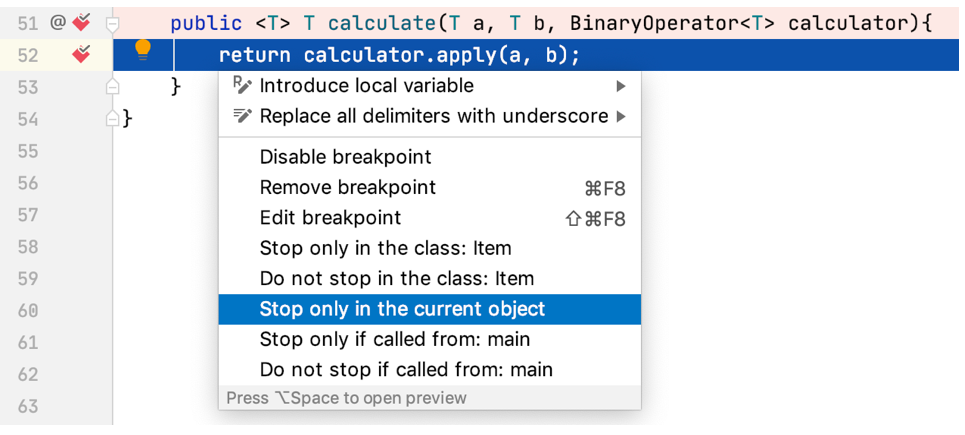


To quickly complete a method call of a static method located anywhere in your project, a library or a JDK, enter a prefix and press **Ctrl+Espace** twice. You can press **Alt+Entrée** to import the selected method.

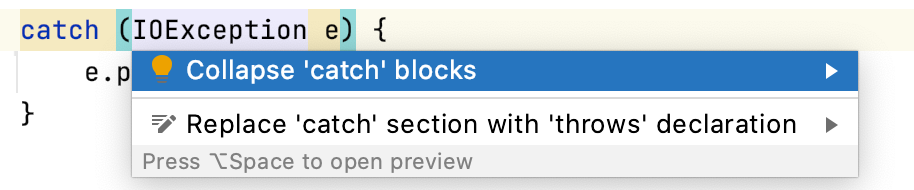


If nothing is selected in the editor, press **Ctrl+C** to copy the whole line at the caret to the clipboard.

Get access to the most common breakpoint actions and filters through intention actions (**Alt+Entrée**)

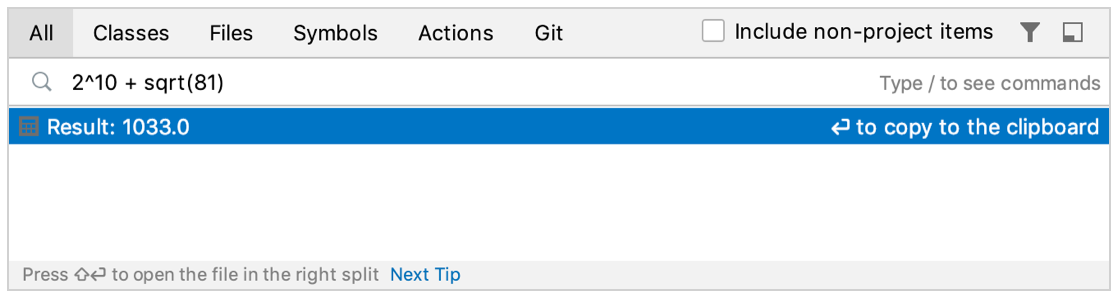


Press **Alt+Entrée** in the editor to fix a highlighted error or warning, improve, or optimize a code construct. For some intention actions, you can open a preview by pressing **Ctrl+Maj+I** (**View | Quick Definition** ).



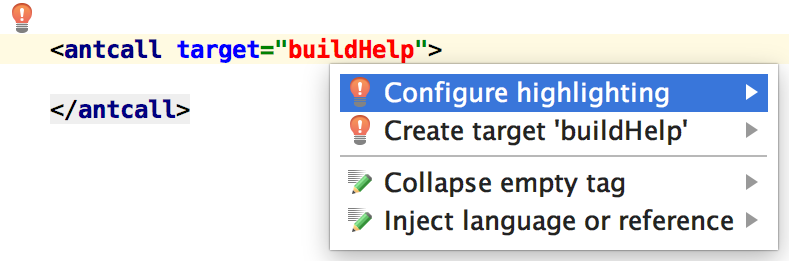
You can evaluate simple arithmetical expressions with the **Search Everywhere** window (**Shift** twice).

Enter an expression you want to evaluate in the search field. The result will be displayed in the search results.



To quickly select the currently edited element (a class, file, method, or field) in another view, press **Alt+F1** or call **Navigate | Select In**

Refer to a non-existing target tag in your Ant build file, and IntelliJ IDEA will suggest you to automatically create the corresponding tag. This intention action will not even make you change your current editing location.



Press **Alt+1** to open the **Project** tool window and switch focus to it.

When you need to cast an expression value to the required type, use Smart type completion. For example, type

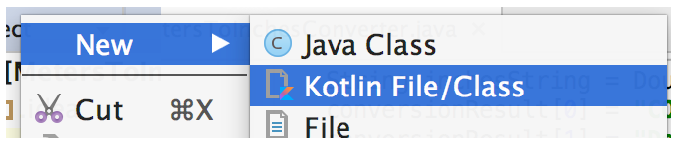
**String s = (**

and press **Ctrl+Maj+Espace** to see what happens

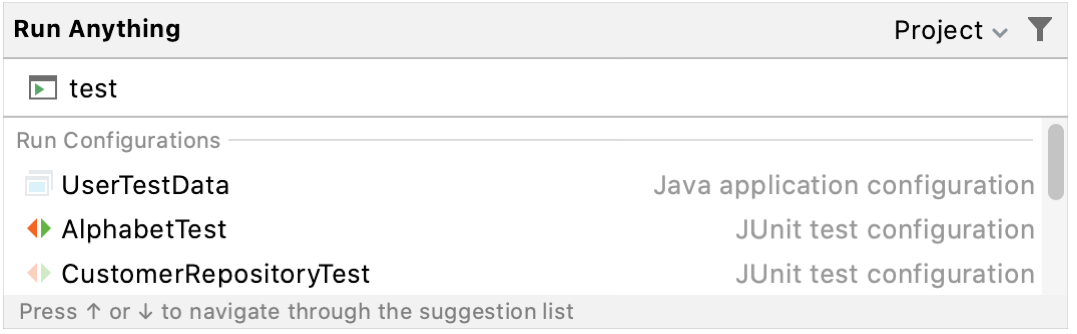
Kotlin is a powerful, concise and expressive language crafted by JetBrains.

IntelliJ IDEA supports it out-of-the-box.

To create a Kotlin file, right-click the target folder, and choose the corresponding option on the New menu:

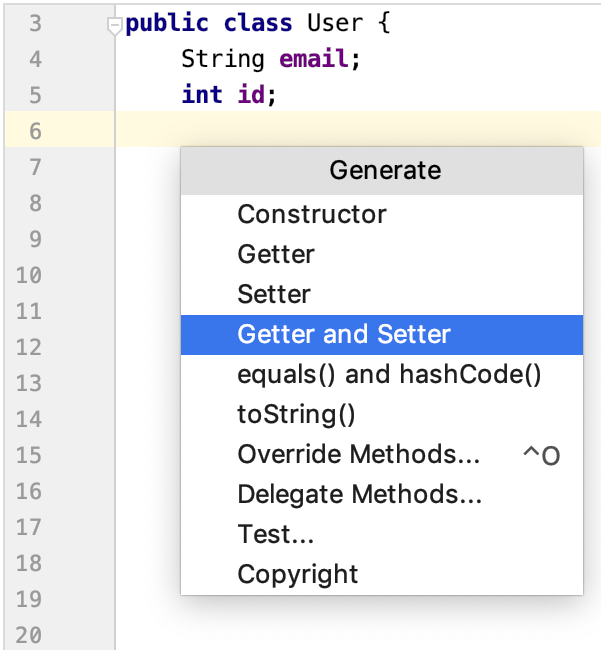


Press **Ctrl** twice to get quick access to numerous actions, such as opening a project, launching a run/debug configuration, running a command-line utility, and so on.

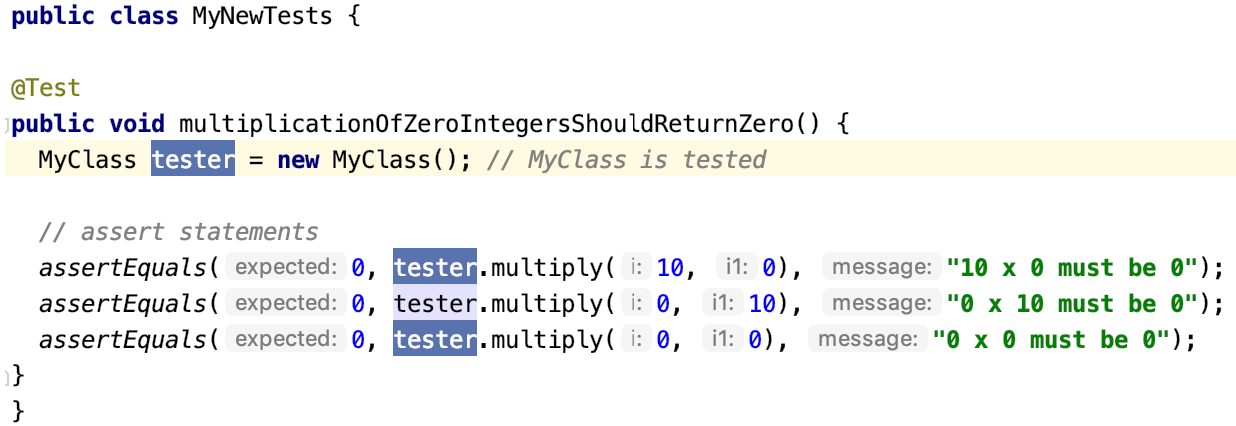


To join two lines into one and remove unnecessary spaces, press **Ctrl+Maj+J**.

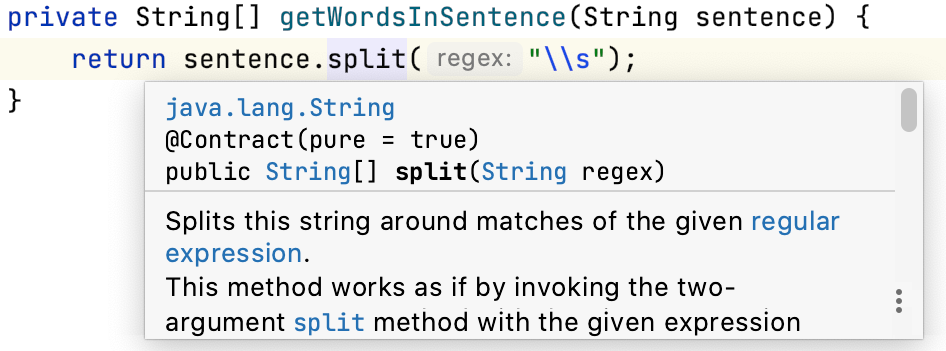
IntelliJ IDEA Community can generate getter and setter methods for fields in your class. With the caret inside the class, **press Alt+Insérer** (**Code | Generate** ).



To select several words, press and hold ⇧⌥ (on Windows and Linux) / ⇧⌥ (macOS), and double-click each word you want to select.



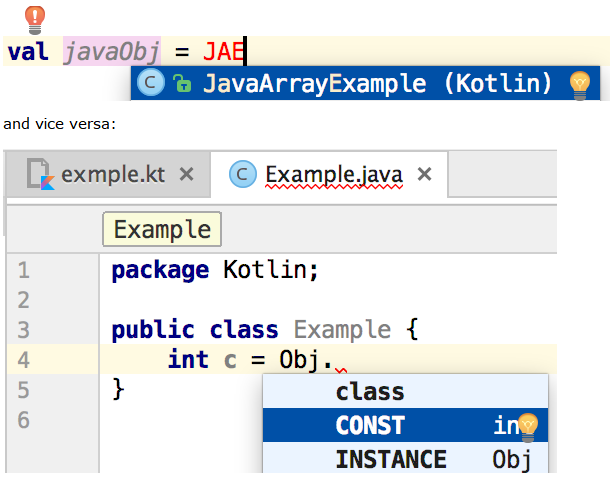
To quickly see the documentation for a class or method at the caret, press **Ctrl+Q** (**View | Quick Documentation** ).



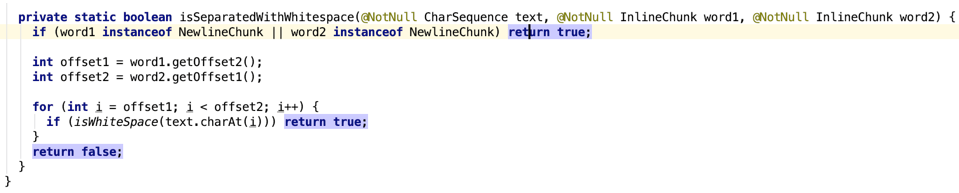
You can view all statements within the method where certain exceptions can be **thrown**. Place the caret at the throws statement and press **Ctrl+Maj+F7**.



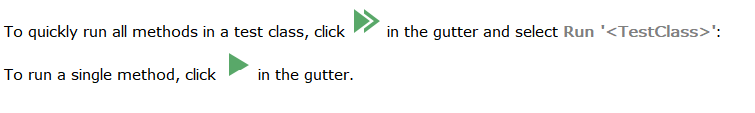
IntelliJ IDEA allows you to easily call a Java code from Kotlin:



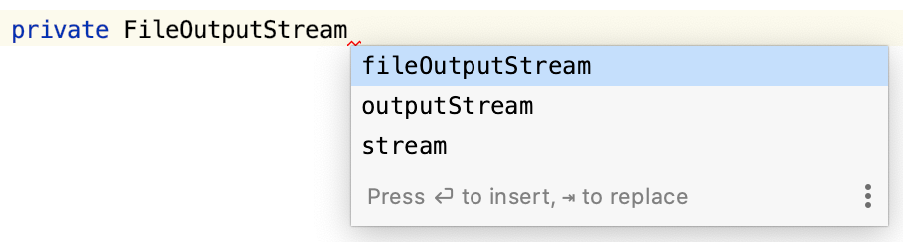
To view all exit points of a method, place the caret at one of them, for example the **return** statement, and press **Ctrl+Maj+F7**:



To navigate to the implementations of an abstract method, position the caret at its usage or its name in the declaration and press **Ctrl+Alt+B**.

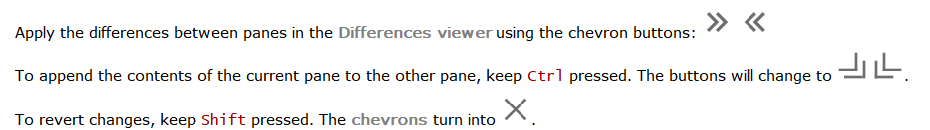


Code Completion can suggest a name for a variable when you declare it. For example, start typing **private** **FileOutputStream** and press **Ctrl+Espace**.



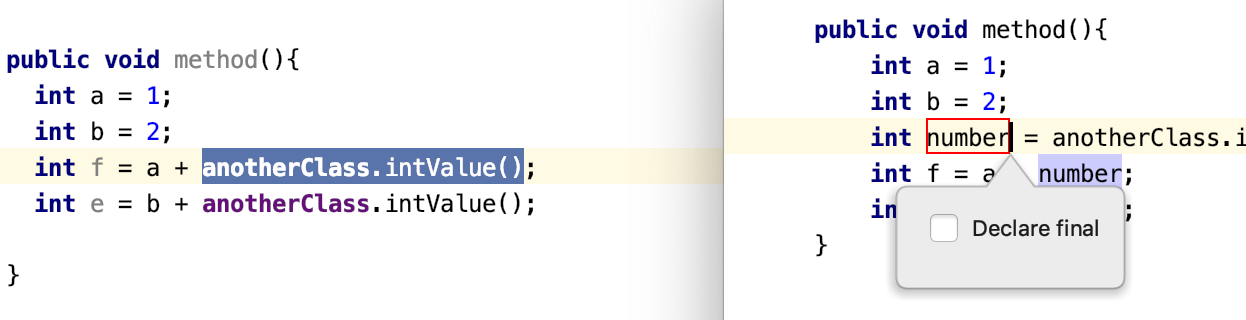
You can customize name prefixes for local variables, parameters, instance and static fields in the **Settings/Preferences** dialog **Ctrl+Alt+S** under **Code Style**.

Local history lets you track all changes to files, classes, methods, or any code fragments and roll back to any stable point if necessary. To view local history, from the main menu choose **VCS | Local History | Show History**.



The **Extract Variable** refactoring wraps a selected expression into a variable. It adds a new variable declaration and uses the expression as an initializer. To invoke this refactoring, select the expression and press **Ctrl+Alt+V** (**Refactor | Extract | Variable** ):

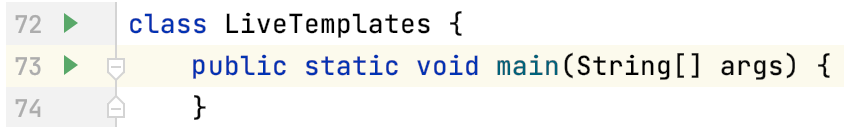
This will result in the following:



Use live templates to insert frequent code constructs.



When you select a live template from the suggestion list, the template expands to the variant that is defined on **the Editor | Live Templates** page of the Settings/Preferences dialog **Ctrl+Alt+S.**



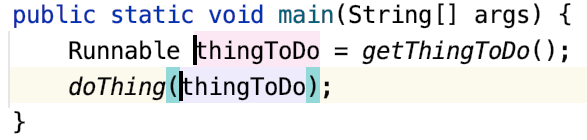
Use **Tab** to move between the template fields.

You can accept the current selection in the code completion suggestions list with the period key (.), comma (,), semicolon (;), space, and some other characters.

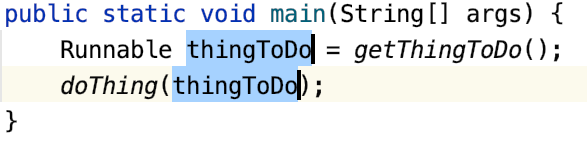
Use text patterns in Search Everywhere (double **Shift**) when searching for a class, file, or symbol. Use **\*** and **space**:

* **\*** stands for any number of arbitrary characters.
* **space** marks the end of a pattern. The preceding string is considered not just a prefix but a whole pattern.

In IntelliJ IDEA Community, you can have multiple carets. Press and hold **Shift+Alt** (on Windows and Linux) / ⇧⌥ (on macOS) and then click at different positions to set additional carets in the editor.

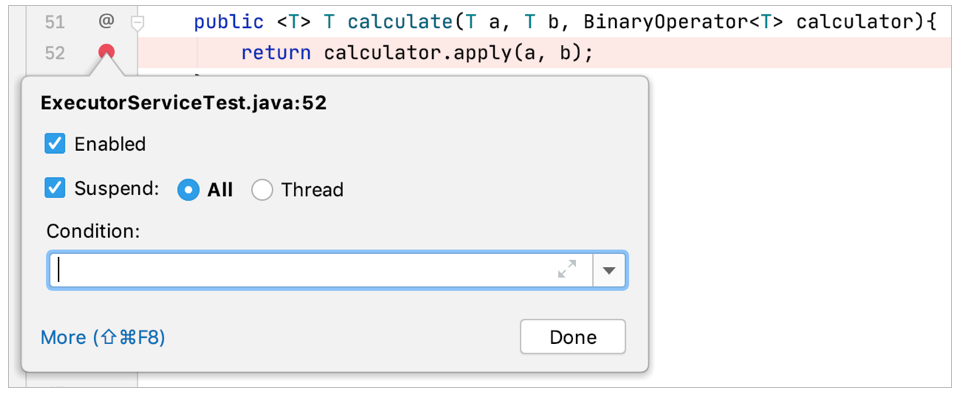


You can then press **Shift** with the left or right arrow keys to select multiple text fragments.



Press **Echap** to remove all the carets except for the recently added one.

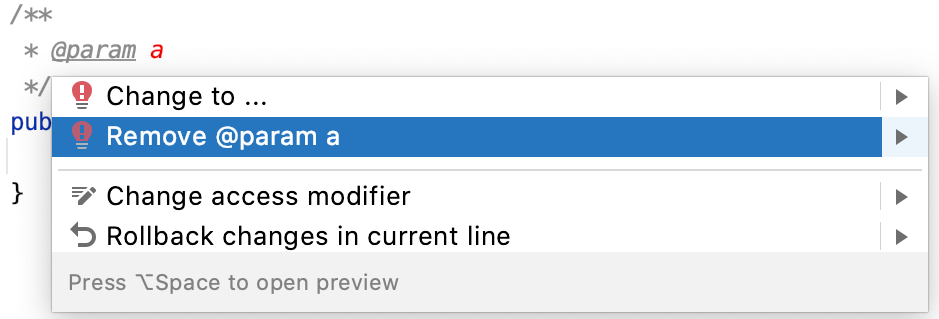
Right-click a breakpoint marker in the gutter to quickly enable/disable the breakpoint or adjust its properties.



In IntelliJ IDEA Community, you can configure different copyright notices for different sets of files. For convenience, you can use variables. For example, use **$today.year** to keep the year up to date.

In the Settings/Preferences dialog **Ctrl+Alt+S**, select **Editor | Copyright | Copyright Profiles**, create a new profile, and add the text for your copyright notice using variables

If a method signature has changed, IntelliJ IDEA Community highlights the documentation comment tags that ran out of sync and suggests a quick-fix:



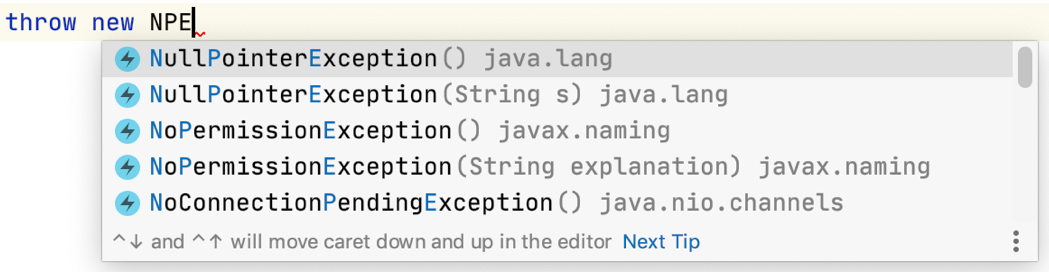
When some words are missing in the pre-defined dictionaries, you can create your own. A custom dictionary is a **.dic** text file containing each word on a new line.

All you have to do is add the directories where your dictionaries are stored **in Settings/Preferences | Editor | Proofreading | Spelling**.

You can view multiple files side by side in the editor. Right-click the desired editor tab and select how you want to split the editor (**Split Vertically** or **Split Horizontally** ).

Alternatively, drag a tab to any area of the editor to activate split-screen mode.

You can narrow down the list of code completion suggestions by using camel case prefixes.

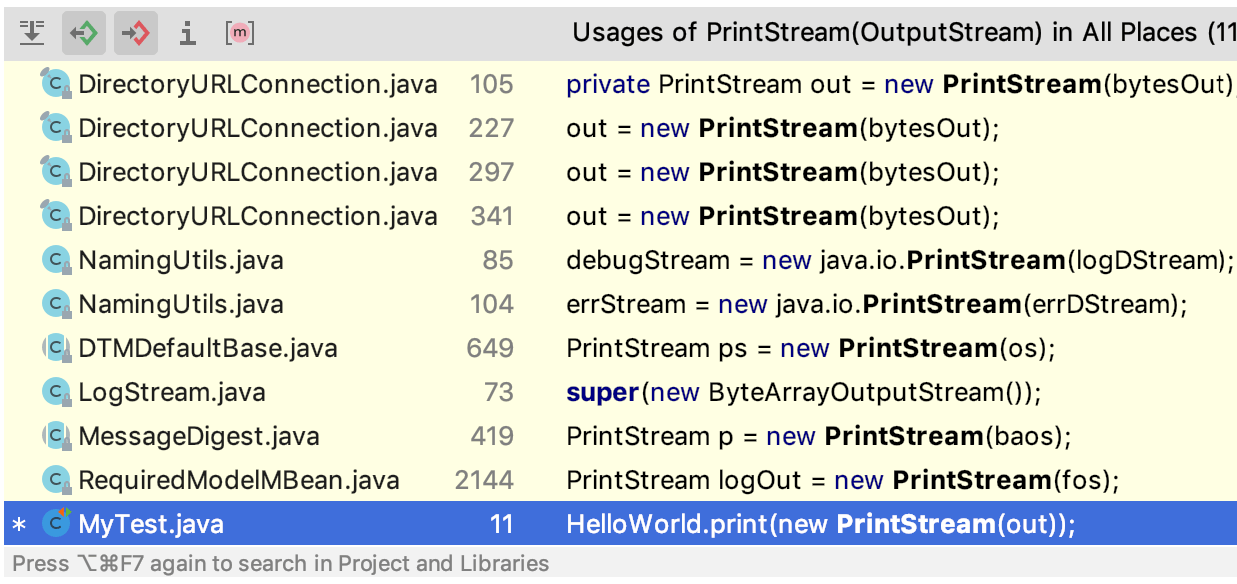


To close all editor tabs, right-click a tab and select **Close All**.

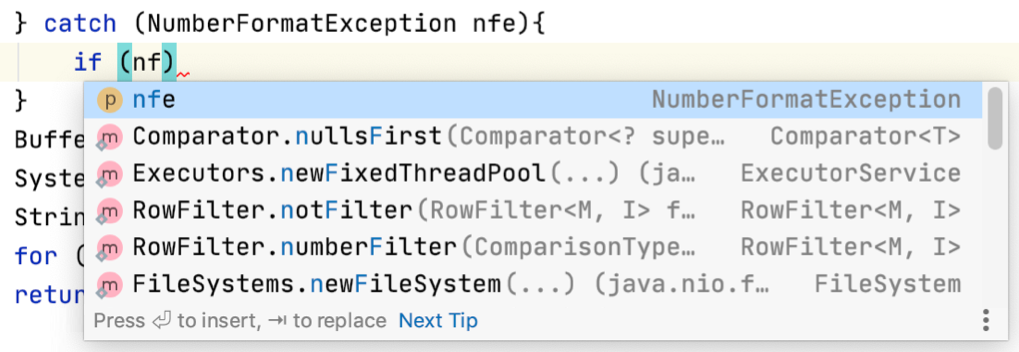
To close all tabs except the active one, press **Alt** (on Windows and Linux) / ⌥ (on macOS) and click the Close button on the active tab

You can view the list of all usages of a class, method or variable across the whole project, and quickly navigate to the selected item. Place the caret at a symbol and press **Ctrl+Alt+F7** (**Edit | Find Usages | Show Usages** ).

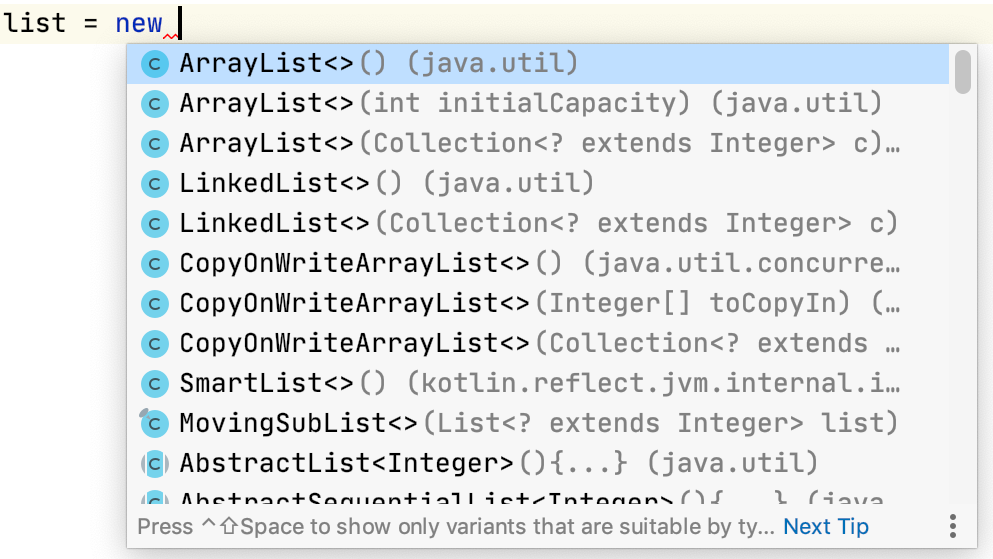
To jump to a usage, select it from the list and press **Entrée**.



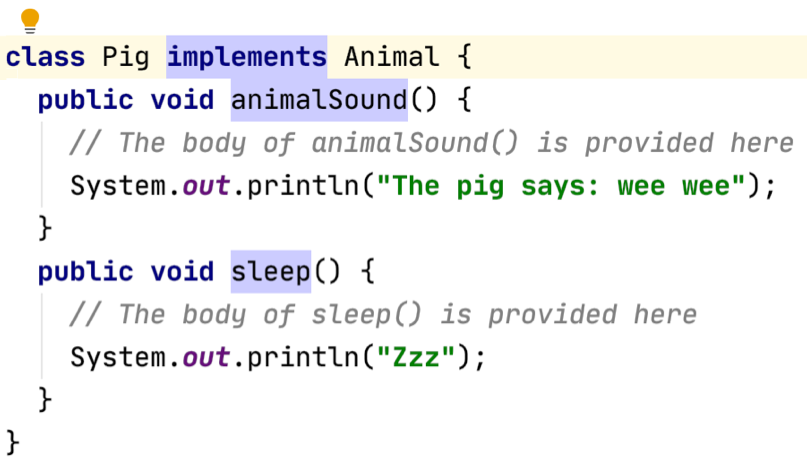
When you invoke Basic Completion **Ctrl+Espace**, IntelliJ IDEA Community suggests the choices that are reachable from the current caret position. Press **Ctrl+Espace** again to also see inaccessible classes and members.



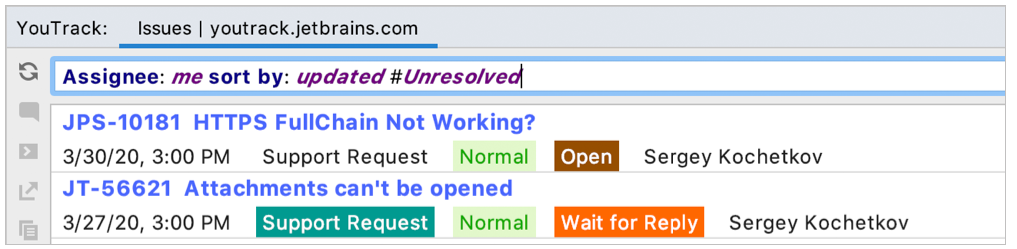
The SmartType code completion analyzes the expected type of the whole expression and helps to find methods and variables that are applicable in the current context. It works after the **return** keyword, in an assignment, in the argument list of a method call, and other places. Press **Ctrl+Maj+Espace** (**Code | Completion | SmartType**) to get the completion list filtered.



To view all methods of the **implemented** interfaces in a class, place the caret at the **implements** keyword in the class declaration and press **Ctrl+Maj+F7**:



Manage your projects efficiently with YouTrack. If you have a YouTrack instance, you can use the YouTrack Integration plugin to work with your issues directly in IntelliJ IDEA Community.



YouTrack is free for teams of 10 forever. Try YouTrack.

Press **Ctrl+Maj+I** (**View | Quick Definition**) to preview the definition or content of the symbol at the caret, without opening it in a separate editor tab

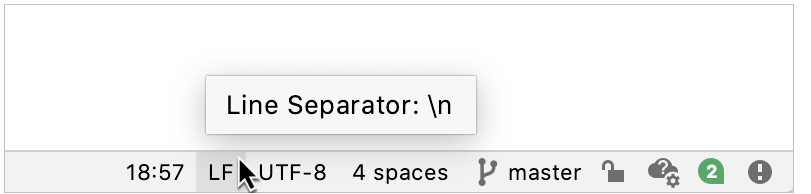
Using options in **Settings/Preferences | Editor | Reader Mode** settings you can configure how to display code and comments in read-only files and libraries to make them easier to read and understand.

You can enable inlay hints, font ligatures, adjust line height, toggle rendered view of comments, and so on.

Use **Refactor | Copy** to create a class which is a copy of the selected class.

This is useful when you need to create a class similar to an existing one, and it's not feasible to put shared functionality in a common superclass

To check which line separators are used in the current file, look at the status bar:



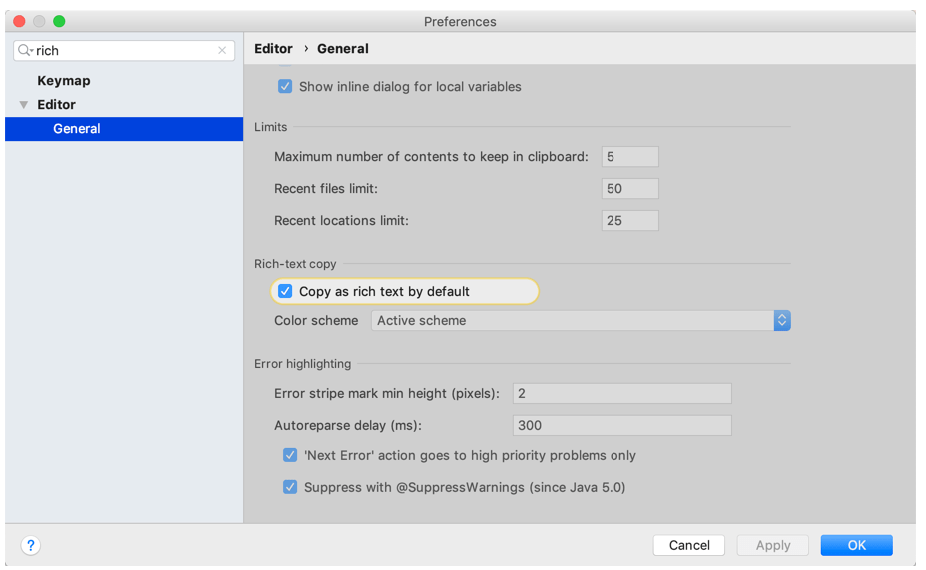
Scratch files are temporary files that let you experiment and prototype in the editor, without creating any project files.

To create a scratch file, press **Ctrl+Alt+Maj+Insérer**, and then select the language to use.

In the editor, press **Ctrl+W**/ **Ctrl+Maj+W** to extend or shrink your current code selection.

You can copy text from the editor as rich text to paste it into any other editor that recognizes RTF.

Make sure the **Copy as rich text** checkbox is selected on the **Editor | General** page of the **Settings/Preferences** dialog **Ctrl+Alt+S**.



Press **Ctrl+Maj+F7** (**Edit | Find Usages | Highlight Usages in File**) to quickly highlight usages of a certain variable in the current file.

Press F3 and Maj+F3 to navigate through the highlighted usages.

Press **Echap** to remove highlighting.

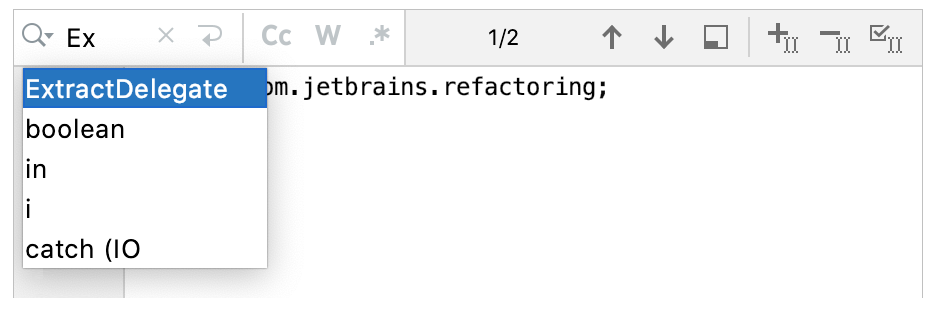
If you do not remember a live template abbreviation, press **Ctrl+J** to see a list of suggestions for the current context. For example, type it and press **Ctrl+J** to see what happens.



Git annotations show detailed information on the origin of each line of code (right-click the gutter and select **Annotate with Git Blame** ).

Right-click an annotation and choose **Show Diff** to review the differences between the current and the previous version of the file.

When searching for a text string in a file, use recent search history. Press **Ctrl+F** to open the search pane and then press **Alt+Bas** to show the list of recent entries.

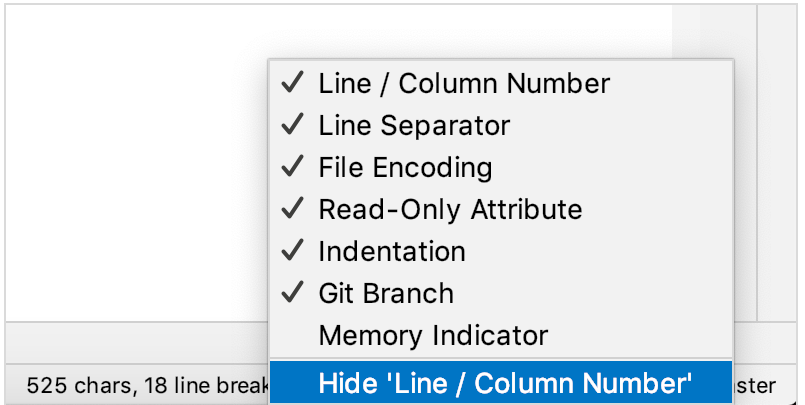


To avoid accidentally removing breakpoints, you can choose to remove them by dragging them from the gutter. Press **Ctrl+Alt+S**, go to Build, Execution, **Deployment | Debugger** and select **Drag to the editor or click with middle mouse button**.

Click a breakpoint to enable or disable it.

You can drag an external file from Explorer or Finder and drop it onto the Favorites tool window **Alt+2**.

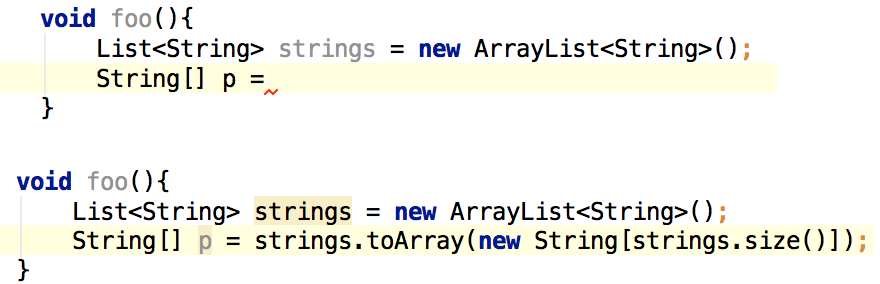
You can configure the set of widgets that appear in the status bar. To do this, right-click a widget and select the items you want to see.



Welcome to IntelliJ IDEA Community 2021.1

You can get familiar with the main features of the IDE by following these tips. You can try out the features without closing this dialog. If you close it, you can always get back to it by selecting **Help | Tip of the Day** from the main menu

Invoking Type-matching code completion (**Ctrl+Maj+Espace**) twice when an array type is expected will search for collections with the same component type and suggest converting them using the toArray() call.



To select multiple text fragments and modify them, press and hold **Shift+Alt** (on Windows and Linux) / ⇧⌥ (on macOS) and drag your mouse across the text:

