Debug Tool Window. Variables

The **Variables** pane enables you to examine the values stored in the objects of your application.

When a stack frame is selected in the Frames pane, the **Variables** pane displays all the data within its scope (method parameters, local and instance variables).

In this pane you can set labels for the objects, inspect objects, evaluate expressions, add variables to watches and more.

In this topic:

Context menu options

Item	Shortcut	Description
Adjust Range		This command is available for the arrays and lists and lets you view the contents of an array within the specified range of indices.
Inspect		This command is available for fields, local variables and reference expressions, and opens a non-modal **Inspection** window, where you can concentrate on a particular reference. You can open as many **Inspection** windows as required. The view in the **Inspection** window is the same as in the Watches pane, but requires less screen space.
Mark Object	F11	Adds object label.
Set Value	F2	This command enables you to change the runtime value of a field or variable.
Jump to Source	F4	This command opens the source code of the selected variable or field in the editor and places the caret on a proper line.
Jump to Object Source	Shift+F4	This command opens the source code of the type of selected variable or field in the editor.

Item	Shortcut	Description
View as		Use this command to select the layout of the values. For the numeric values, you can toggle between the decimal and hexadecimal formats. The objects are usually represented by their class name and instance identifier, but you can optionally show them in string format. ■ **Auto**: Available for items where different layout is possible. With this layout IntelliJ IDEA looks through all available renderers searching for the first suitable layout for the current item. If none is found, the default layout is applied. ■ **Hex**: This layout is available for numeric variables. If checked, the variable is shown in hexadecimal format. ■ **Primitive**: This layout is available for primitive type variables. It shows the value appropriate for the primitive type. ■ **toString()**: This layout is available for all reference types where `toString()` is overridden except for arrays, and shows the node's `toString()` value in the tree. ■ **Array**: Available for arrays. ■ **Object**: Default layout. Available for all non-primitive type nodes. ■ **Map**: Show as a map. ■ **Collection**: Show as a collection. ■ **User-defined renderer**: Available for renderers created by the user in the Debugger | Type Renderers or in the Customize Data Views dialogs. The corresponding renderer name is shown in the menu.
Add to Watches		This command is available for all nodes except static. Use this command to create an expression that references the node and add this expression to the Watches pane.
Add Field Watchpoint		Use this command to create a new Field Watchpoint for a field and define its options in the Breakpoints dialog.
Copy Value		Use this command to copy the value of a node to the Clipboard. If a string value is too long to fit in the stack frame view, it is truncated. You can use this command to copy the value to the Clipboard, and then paste it to the editor, where you can examine the contents. Alternatively, hover your mouse cursor over the value and view the contents at the tooltip.
Customize Data Views		Use this command to customize how objects are displayed in the debugger. In the **Customize Data Views** dialog, specify the items to be shown in the tab and their view mode.

Types of variables

Icon	Description
	Static variable
	Field
	Array
	Primitive type
	Object

See Also

Concepts:

- Running, Debugging and Testing

Procedures:

- Debugging

Reference:

- Debug Tool Window. Frames
- Debug Tool Window. Console
- Debug Tool Window. Variables

Web Resources:

- Developer Community