B4X Booklets

B4X I D E

Integrated **D**evelopment **E**nvironment

 Edition 2.2

Last update : 2021.07.12

Main contributors: Klaus Christl (klaus), Erel Uziel (Erel)

To search for a given word or sentence use the Search function in the Edit menu.

All the source code and files needed (layouts, images etc.) of the example projects in this guide are included in the SourceCode folder.

Updated for following versions:
B4A version 11.0
B4i version 7.50
B4J version 9.10
B4R version 3.71

B4X Booklets:
B4X Getting Started
B4X Basic Language
B4X IDE Integrated Development Environment
B4X Visual Designer
B4X Help tools

B4XPages Cross-platform projects
B4X CustomViews
B4X Graphics
B4X XUI B4X User Interface
B4X SQLite Database
B4X JavaObject NativeObject
B4X Cross-platform projects

B4R Example Projects

You can consult these booklets online in this link [B4X] Documentation Booklets.
Be aware that external links don't work in the online display.

1 B4X platforms

B4X is a suite of BASIC programming languages for different platforms.

B4X suite supports more platforms than any other tool
ANDROID | IOS | WINDOWS | MAC | LINUX | ARDUINO | RASPBERRY PI | ESP8266 | AND MORE...

- **B4A**  **Android**

  B4A is a **100% free** development tool for Android applications, it includes all the features needed to quickly develop any type of Android app.

- **B4i**  **iOS**

  B4i is a development tool for native iOS applications.
  B4i follows the same concepts as B4A, allowing you to reuse most of the code and build apps for both Android and iOS.

- **B4J**  **Java / Windows / Mac / Linux / Raspberry PI**

  B4J is a **100% free** development tool for desktop, server and IoT solutions.
  With B4J you can easily create desktop applications (UI), console programs (non-UI) and server solutions.
  The compiled apps can run on Windows, Mac, Linux and ARM boards (such as Raspberry Pi).

- **B4R** 

Arduino / ESP8266

  B4R is a **100% free** development tool for native Arduino and ESP8266 programs.
  B4R follows the same concepts of the other B4X tools, providing a simple and powerful development tool.
  B4R, B4A, B4J and B4i together make the best development solution for the Internet of Things (IoT).

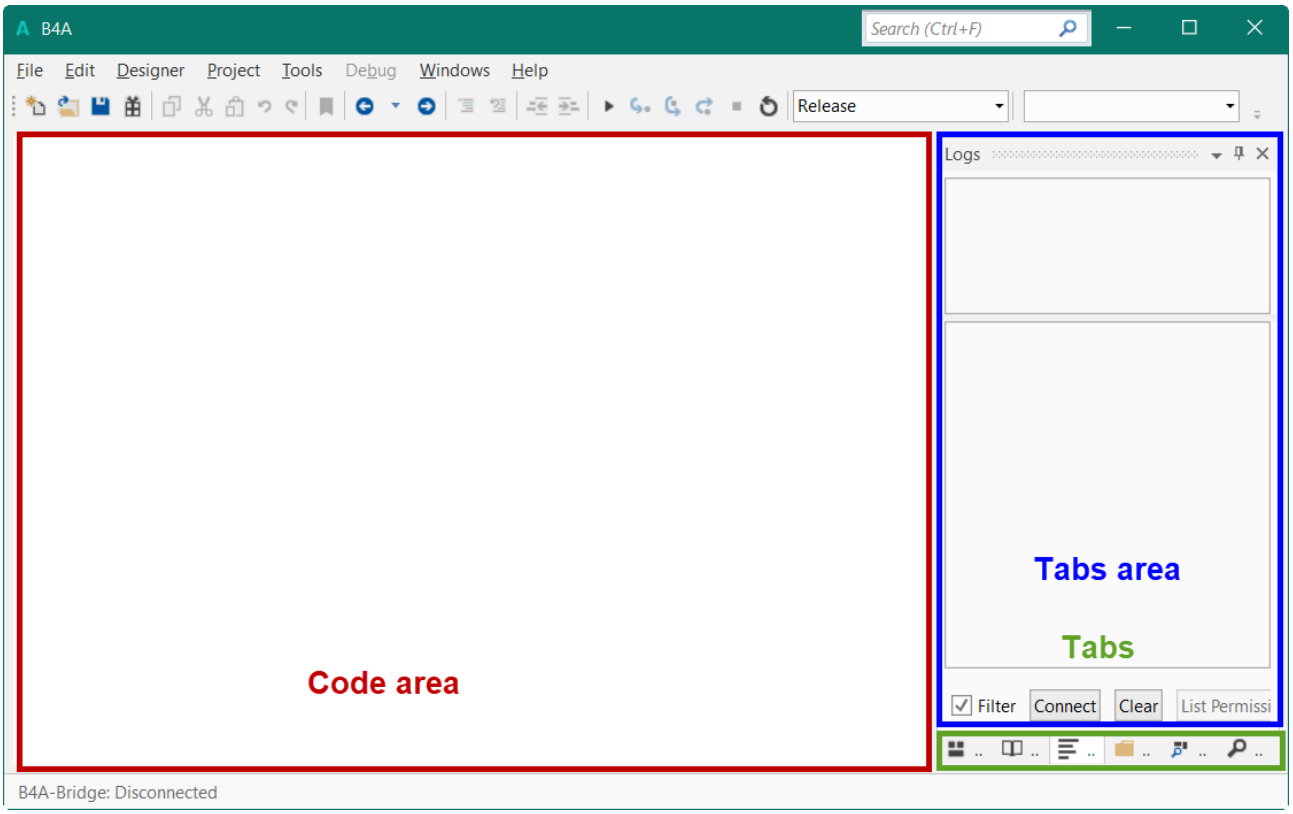
- **B4XPages**

  B4XPages is an internal library for B4A, B4i and B4J allowing to develop easily cross-platform programs.
  B4XPages is explained in detail in the B4XPages Cross-platform projects booklet.
  Even, if you want to develop only in one platform it is interesting to use the B4XPages library it makes the program flow simpler especially for B4A.

2 IDE General

The **I**ntegrated **D**evelopment **E**nvironment.

When you run the IDE you will get a form like the image below.



All the images are made with the B4A IDE.
The IDEs of the other products look similar with different themes.
Specific images are shown if needed.

You see 3 main areas:

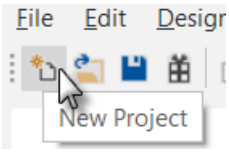
- Code area — The code editor
- Tab area — The content of this area depends on the selected Tab.
- Tabs — Tabs for different settings.

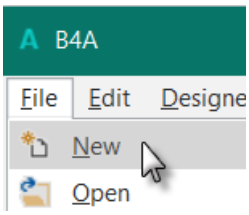
Everything is empty, you can:

- Create a new project.
- Open an existing project.

2.1 Create a new project

To start a new project you must click on:

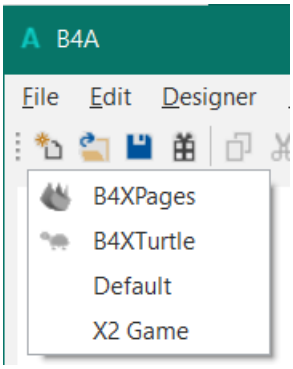
The New Project icon 

or New in the 

File menu.

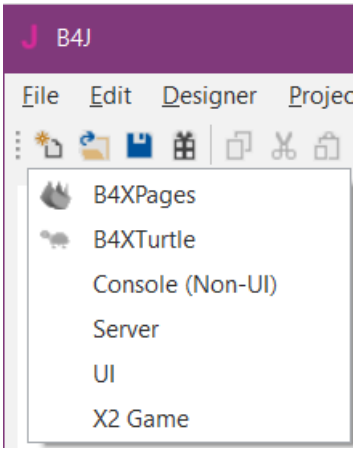
You will be asked what kind of project you want to create.
Specific templates will be loaded depending on the kind or project.

2.1.1 Create a new B4A or B4i project



- B4XPages Explained in the B4X Cross-platform booklet
- B4XTurtle
- Default Default Android project.
- X2 Game X2 Game project

2.1.2 Create a new B4J project



- B4XPages Explained in the B4X Cross-platform booklet.
- B4XTurtle Explained in the Forum tutorial.
- Console (non-User Interace).
- Server Server project.
- UI User Interface project.
- X2 Game X2 Game project

The window below will be shown:

The project type is recalled in the title bar.

Enter the Project Name.

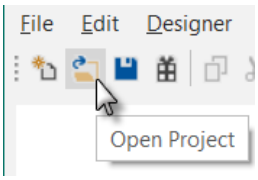
As soon as you type the Project Name it is replicated at the end of the Project Folder name.

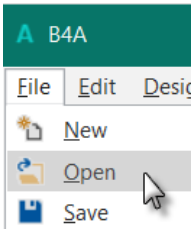
Enter the Project Folder name or use the Browse button to select it.

The Project Folder name is memorized for future projects.
To create a new folder for the project check: Create New Folder.

2.2 Open existing project

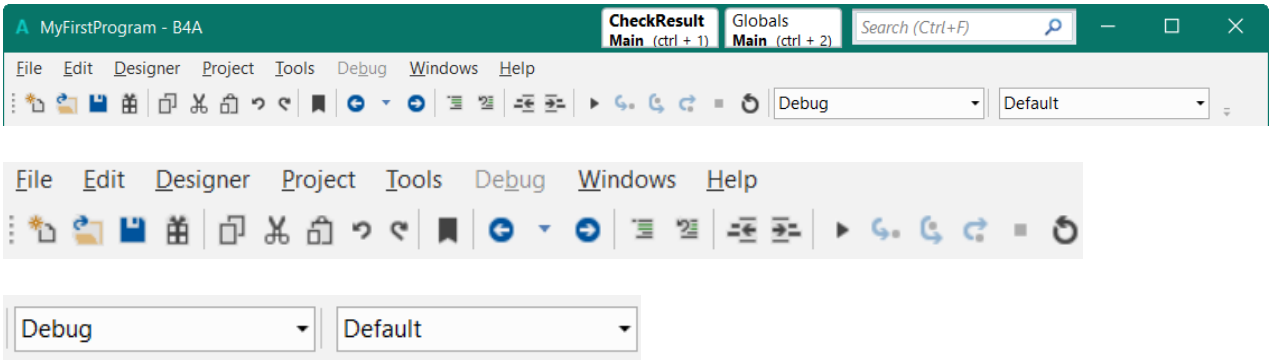
To open an existing project, click on:

The Open Project icon 

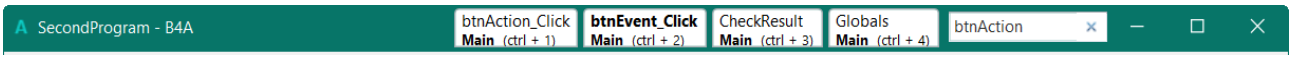
or click on Open in 

the File menu.

3 Titlebar, Menu and Toolbar



3.1 Titlebar



In the Titlebar you find:

- Recent code positions and designer layouts.
- The search field for Quick Search.

3.1.1 Recent code positions (AutoBookmarks)

Recent code positions and designer layouts appear as tabs in the window title.
The IDE decides on the list of tabs based on several factors (recency, modifications and others).
The list is saved together with the project and restored when the project is loaded.

Each field contains the name of the routine `btnAction_Click` and the module **Main**.
In the example above, to move to the third position you can either:

- Click on the third rectangle.
- Press Ctrl + 3

To disable the AutoBookmarks:
1. Close the IDE.
2. Edit either:
   C:\Users\<user name>\AppData\Roaming\Anywhere Software\Basic4andriod\b4xV5.ini
   C:\Users\<user name>\AppData\Roaming\Anywhere Software\B4i\b4xV5.ini
   C:\Users\<user name>\AppData\Roaming\Anywhere Software\B4J\b4xV5.ini
   C:\Users\<user name>\AppData\Roaming\Anywhere Software\B4R\b4xV5.ini
3. Set ShowAutoBookmarks to False.

3.1.2 Quick Search field

Quick search is explained in the Quick Search chapter.

3.2 Toolbar

Generates a new empty project [Ctrl + N].
Loads a project.
Saves the current project [Ctrl + S].
Export As Zip.
Copies the selected text to the clipboard [Ctrl + C].
Cuts the selected text and copies it to the clipboard [Ctrl + X].
Pastes the text in the clipboard at the cursor position [Ctrl + V].
Undoes the last operation [Ctrl + Z].
Redoes the previous operation [Ctrl + Shift + Z].
Toggle Bookmark
Navigate backwards [Alt + Left].
Navigation history [Alt + N].
Navigate forwards [Alt + Right].
Block Comment [Ctrl + Q].
Block Uncomment [Ctrl + W].
Decrease the indentation of the selected lines.
Increase the indentation of the selected lines.
Runs the compiler [F5].

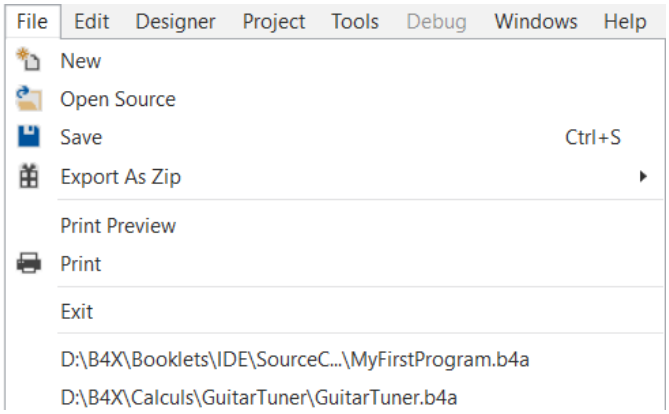
Step In [F8].
Step Over [F9].
Step Out [F10].
Stop.
Restart [F11].

These 5 functions are active only when the debugger is active.

Debug — Compiler options list and Debugging.

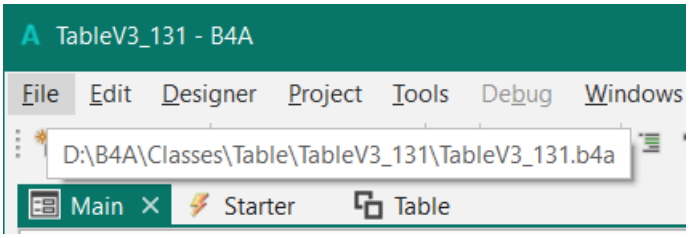
Default — Conditional compiling options.

3.3 File menu



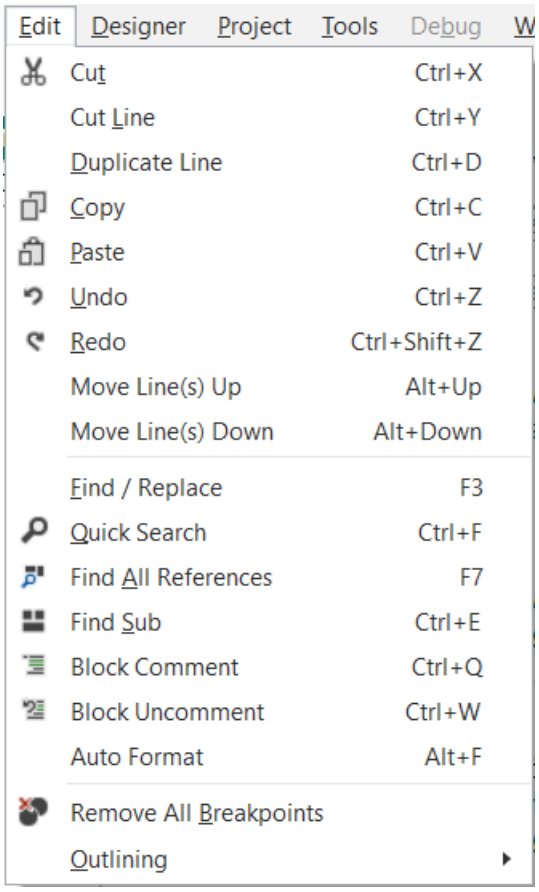
New Generates a new empty project.
Open Source Loads a project.
Save Saves the current project.
Export As Zip Exports the whole project in a zip file.
Print Preview Preview of the print.
Print Prints the whole code of the selected Module.
Exit Leaves the IDE.

List of last loaded programs.



When you hover over the File menu you'll see the full file name.

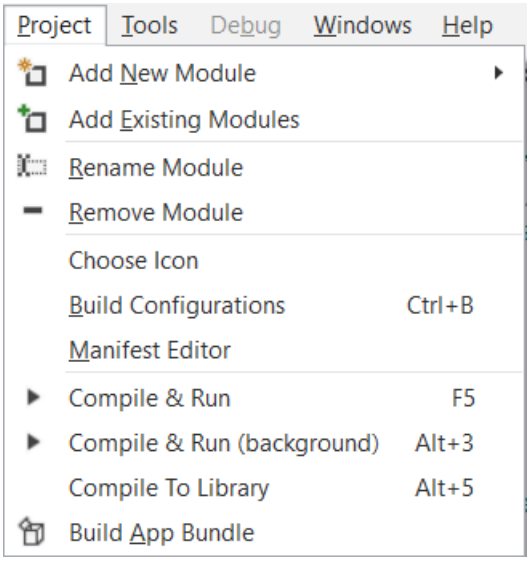
3.4 Edit menu



Cut Cuts the selected text and copies it to the clipboard.
Cut Line Cuts the line at the cursor position.
Duplicate Line Duplicates the line at the cursor position
Copy Copies the selected text to the clipboard.
Paste Pastes the text in the clipboard at the cursor position.
Undo Undoes the last operation.
Redo Redoes the previous operation.
Move Line(s) Up Moves the selected lines upwards.
Move Line(s) Down Moves the selected lines downwards.
Find / Replace Activates the Find and Replace function.
Quick Search Quick Search
Find All References Find All References
Find Sub Find Sub
Block Comment
Block Uncomment
Comment / Uncomment the selected lines.
Auto Format Auto Format
Remove All Breakpoints Breakpoints.
Outlining Collapse the whole code.

3.5 Project menu

B4A



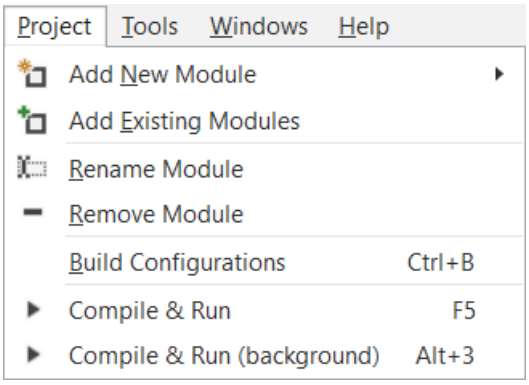
Adds a new module
Adds an existing module

Changes the module name
Removes the current module

Chooses an icon for the program.
Changes the package name.
Runs the Manifest Editor.

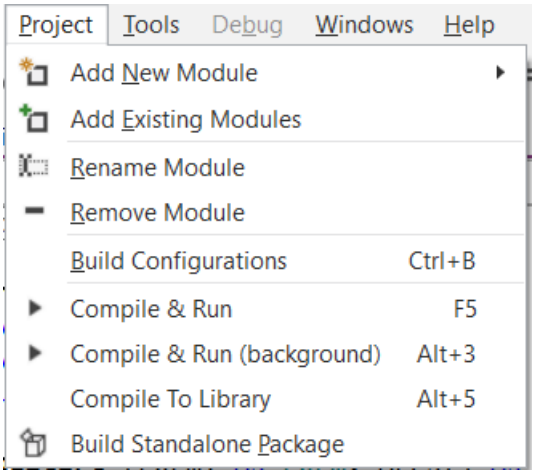
Compile and run the project.
Compile and run the project in the background.
Compile to a library.
Buil App Bundle.

B4i, B4R



Adds a new module
Adds an existing module
Changes the module name
Removes the current module
Changes the package name.
Compile and run the project.
Compile and run the project in the background.

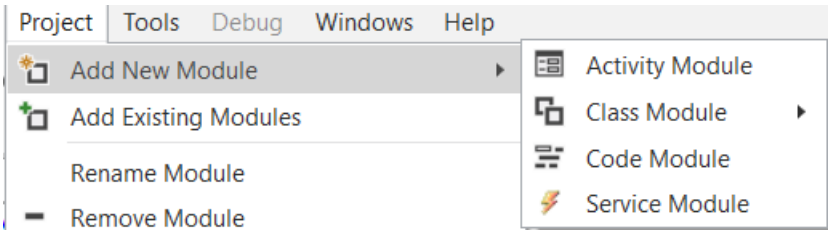
B4J



Adds a new module
Adds an existing module
Changes the module name
Removes the current module
Changes the package name.
Compile and run the project.
Compile and run the project in the background.
Compile to a library.
Buld Standalone Package

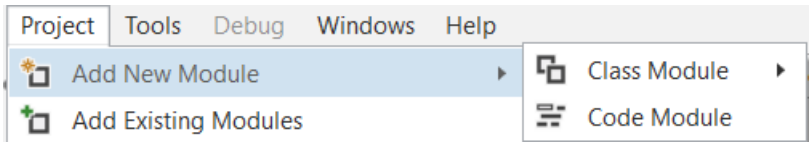
3.5.1 Add a new module

B4A



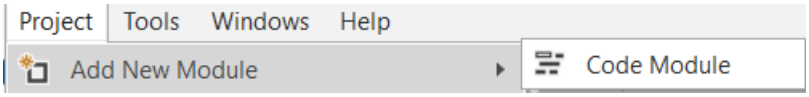
Activity module
Class module
Code module
Service module

B4i, B4J



Class module
Code module

B4R



Only Code module

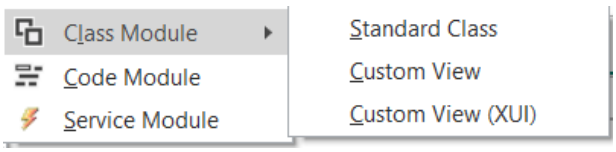
3.5.1.1 Class modules

There exist two Class modules:

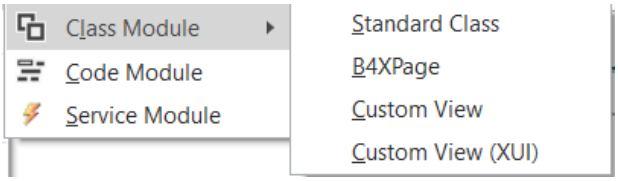
- Standard Class
- Custom View

Custom Views are explained in detail in the B4X CustomViews Booklet.

If you have selected the XUI, jXUI or the iXUI library, then you get another option Custom View (XUI).



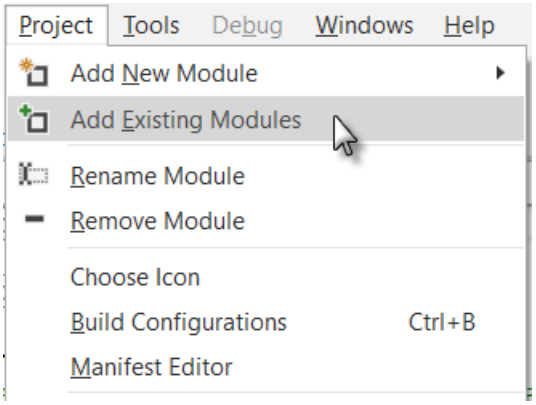
If you have selected the B4XPage library then you get another option B4XPage

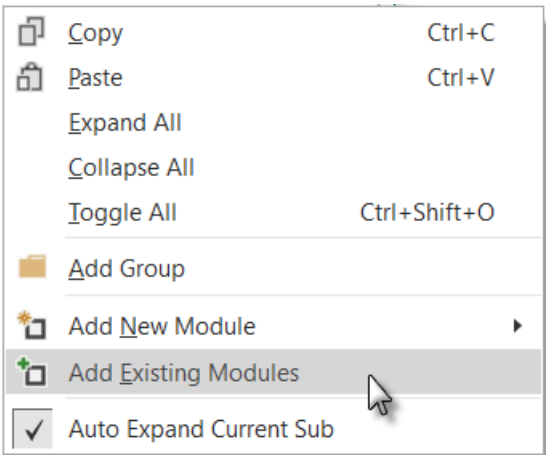


CustomViews XUI are explained in detail in the B4X CustomViews Booklet.
The XUI library is explained in the B4X XUI Booklet.
The B4XPage library is explained in the B4X Cross-platform projects Booklet.

3.5.2 Add an existing module

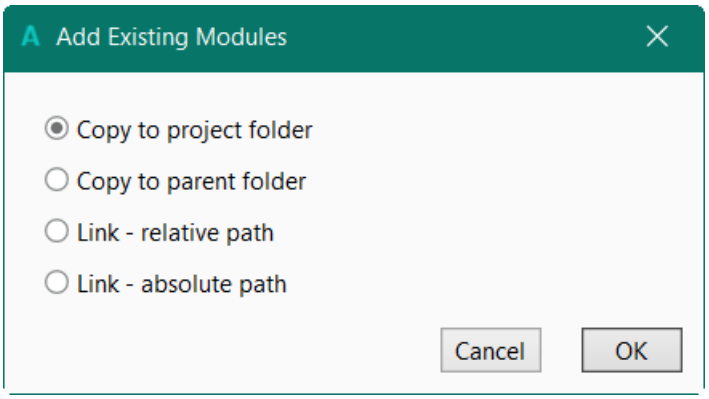
Click on Add Existing Modules in the Project menu, or right click in the Module Tab.





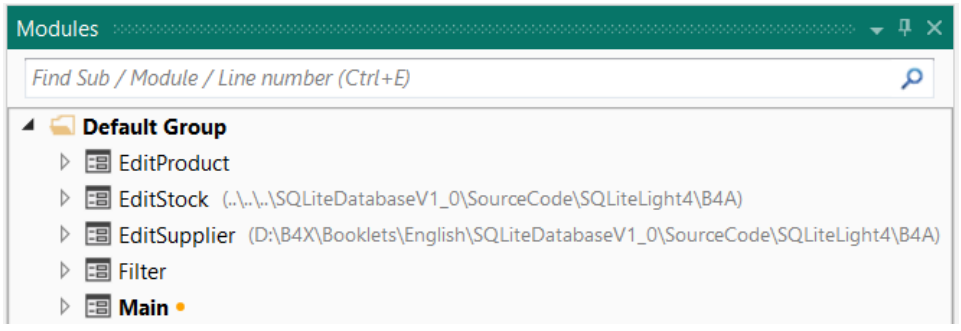
The file chooser will be shown, select the module(s) and click Open.

Then, you will be asked the following:



- Copy to project folder. Copies the file(s) to the Files folder of the project.
- Copy to parent folder. Copies to the parent folder of the project, useful for B4XPages projects.
- Link – relative path. Links the file(s) to a path belonging to the project path. The files are not copied.
- Link – absolute path. Links the file(s) to any path,

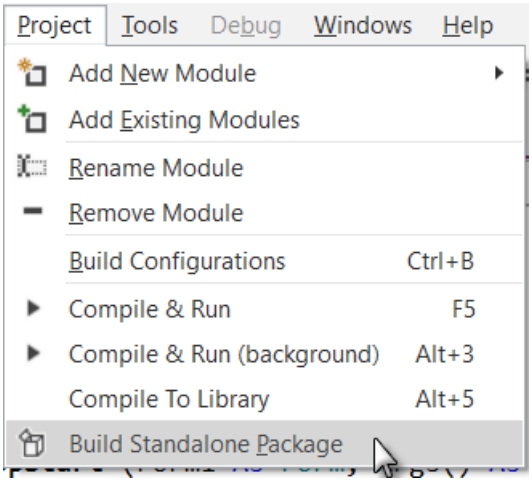
In the Files Tab you will see the difference, for linked modules their path is added.

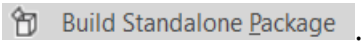


Copied
Relative path
Absolute path
Copied

3.5.3 B4J Build Standalone Package

You can generate a standalone package for your project. It needs Java 11 or Java 14.
If you don't have installed Java 11or Java 14 yet, nor set it in the Tools/Configure Paths menu, you need to do it before being able to build a standalone package.

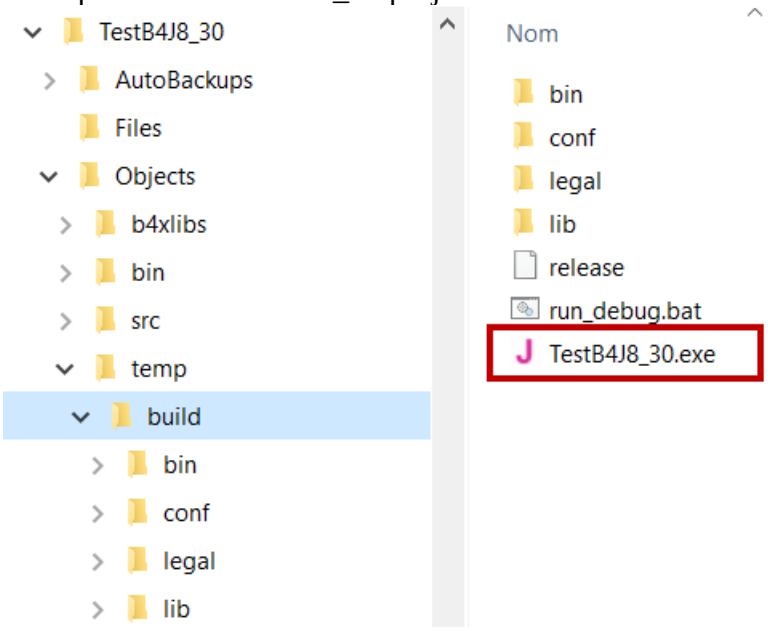


Click on 

.

The *.exe file is saved in the project folder in the Objects\temp\build folder.

Example with a TestB4J8_30 project:



You need to distribute the executable together with the 4 folders.
The run_debug.bat batch file is useful to test the program and see the logs.

An Inno Script template is created in the parent folder. You can use it together with Inno Script to build a single file installer.

The integrated packager creates a Windows package. You can however use the external tool with the generated json file (in the project folder) to create Linux and Mac packages.

The packager supports all kinds of settings. You can set them with the new #PackagerProperty attribute.

For example to set the icon file, assuming that the ico file is in the Files tab and is named turtle.ico:

```
#PackagerProperty: IconFile = ..\Files\turtle.ico
```

Also set the executable name:

```
#PackagerProperty: IconFile = ..\Files\turtle.ico
#PackagerProperty: ExeName = Turtle
```

3.5.3.1 Tips and special cases

- If using jPOI library add:
  ```
  #PackagerProperty: AdditionalModuleInfoString = opens
  schemaorg_apache_xmlbeans.system.sD023D6490046BA0250A839A9AD24C443;
  #PackagerProperty: IncludedModules = jdk.charsets
  ```

- If using WebView add:
  ```
  #PackagerProperty: IncludedModules = javafx.web
  ```

- If using jGoogleMaps add:
  ```
  #PackagerProperty: IncludedModules = javafx.web
  #PackagerProperty: AdditionalModuleInfoString = exports
  com.lynden.gmapsfx.javascript.event;
  ```
  There is an issue with Java 14 and Google Maps. Use Java 11 for now if using jGoogleMaps.

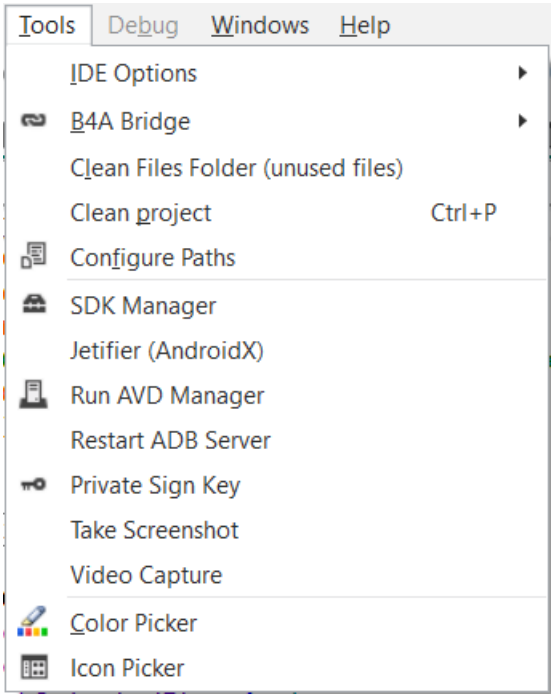
- You can use #CustomBuildAction with the new After Packager step to copy files after the package is built. The default target folder should be: temp\build\bin\

- If using jSerial put the attached jssc.dll file in the project folder and add:
  ```
  #CustomBuildAction: After Packager, %WINDIR%\System32\robocopy.exe, ..\
  temp\build\bin\ jssc.dll
  ```
  Note that it is a Windows 64 bit dll.

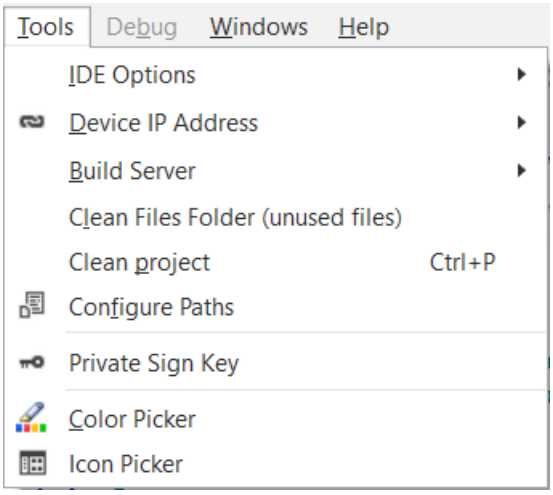
- Each key should appear at most once. So for example if using both WebView and jPOI add:
  ```
  #PackagerProperty: IncludedModules = jdk.charsets, javafx.web
  ```

3.6 Tools menu

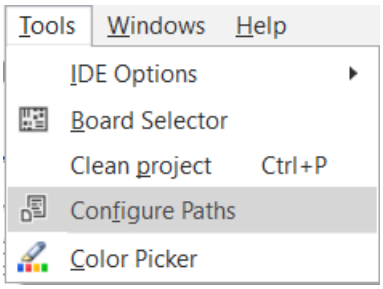
B4A



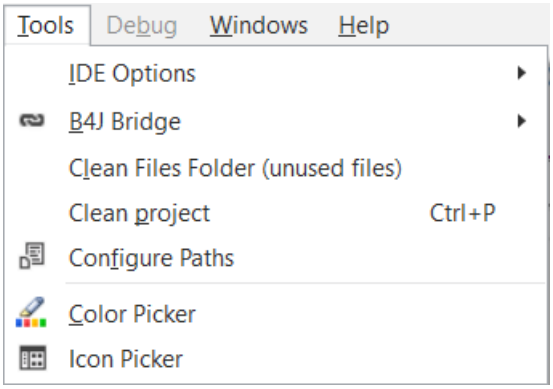
B4i



B4R



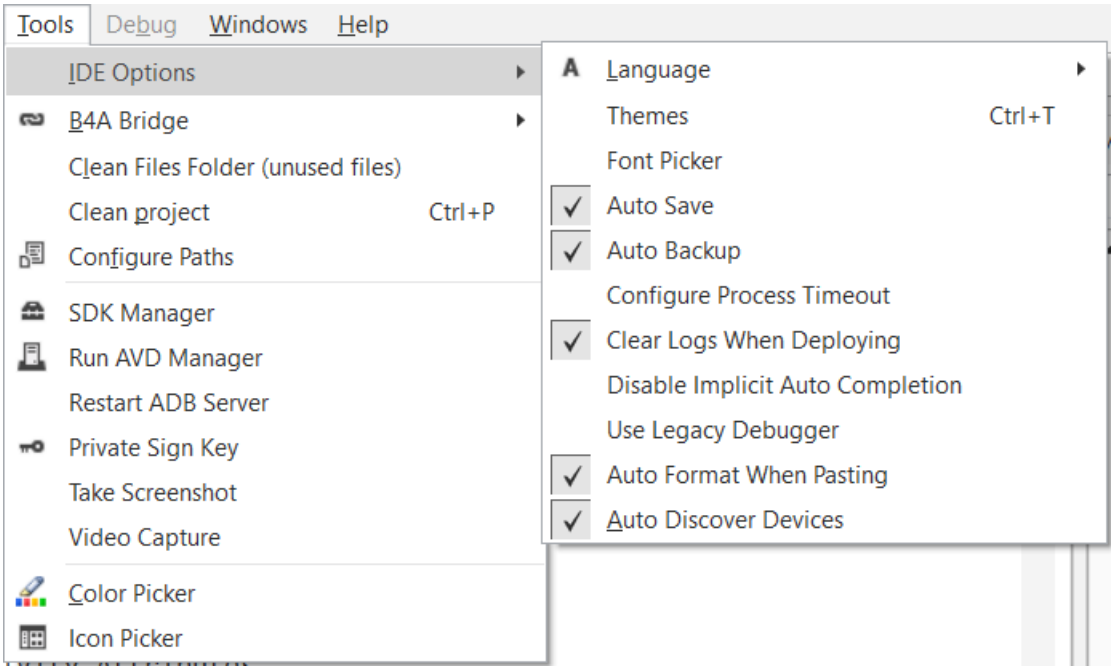
B4J



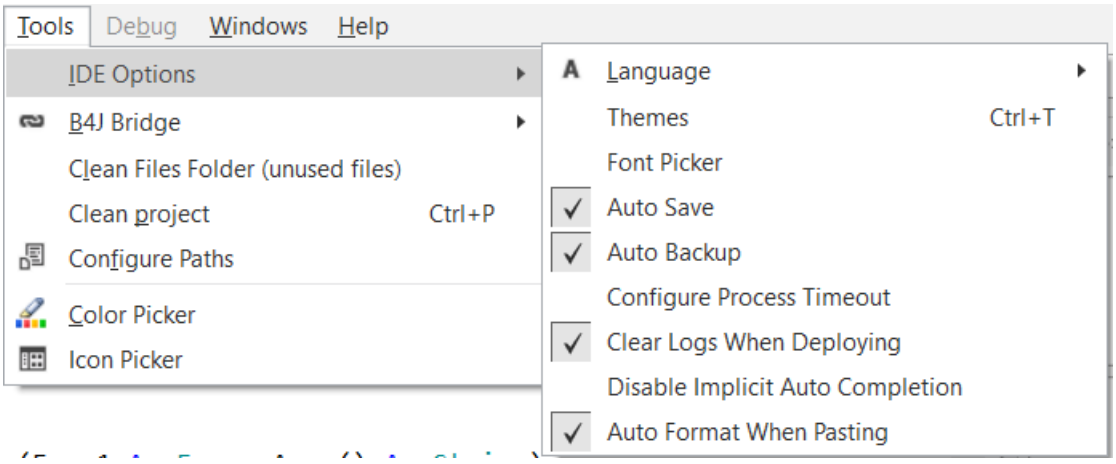
IDE Options	see below	
B4A Bridge, connection with Wifi	B4A	
Clean Files Folder (unused files)	B4A, B4i, B4J	
Clean Project	All	
Configure Paths	All	
SDK Manager	B4A	
Jetifier AndroidX	B4A	
Run ADB Manager	B4A	Used to create Android Emulators, not recommended.
Take Screenshot	B4A	
Capture a video	B4A	
Show the Color Picker	All	
Show the Icon Picker	B4A, B4i, B4J	
Board Selector	B4R	

3.6.1 IDE Options

B4A



B4i, B4J, B4R



All

Language.
Themes.
Font Picker.

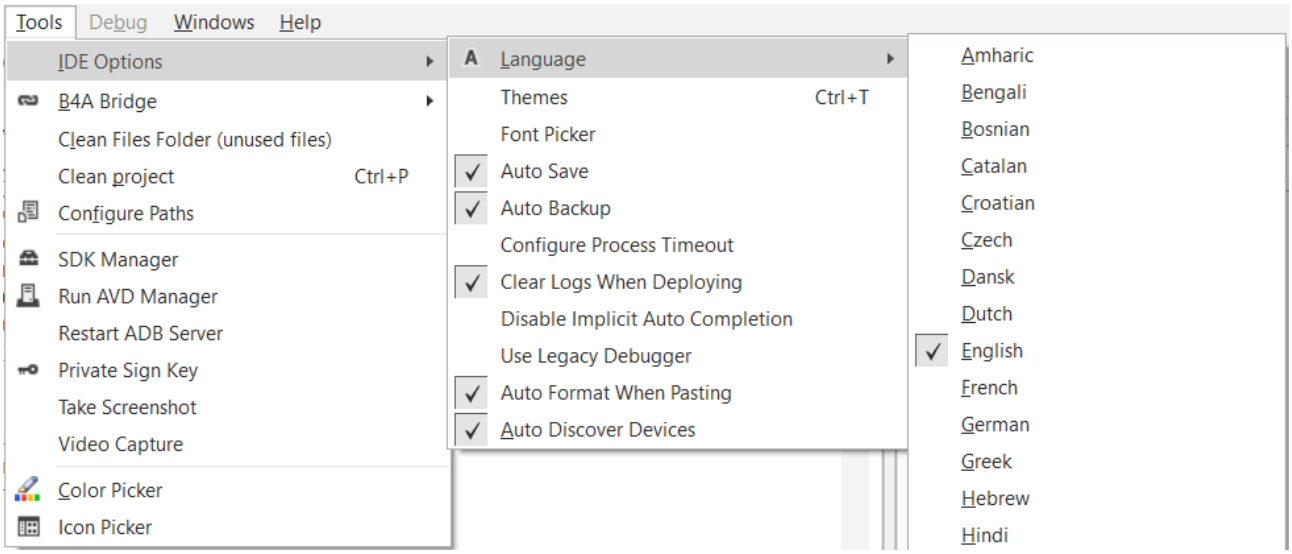
Auto Save	Saves the program every time you run it.
Auto Backup	
Configure Process Timeout	
Clear Logs When Deploying	Removes all Log statements when compiled in Release mode.
Disable Implicit Auto Completion.	
Auto Format When Pasting	

B4A only

Use Legacy Debugger	Use the legacy Debugger instead of the rapid Debugger.
Auto Discover Devices	Detects automatically the connected devices.

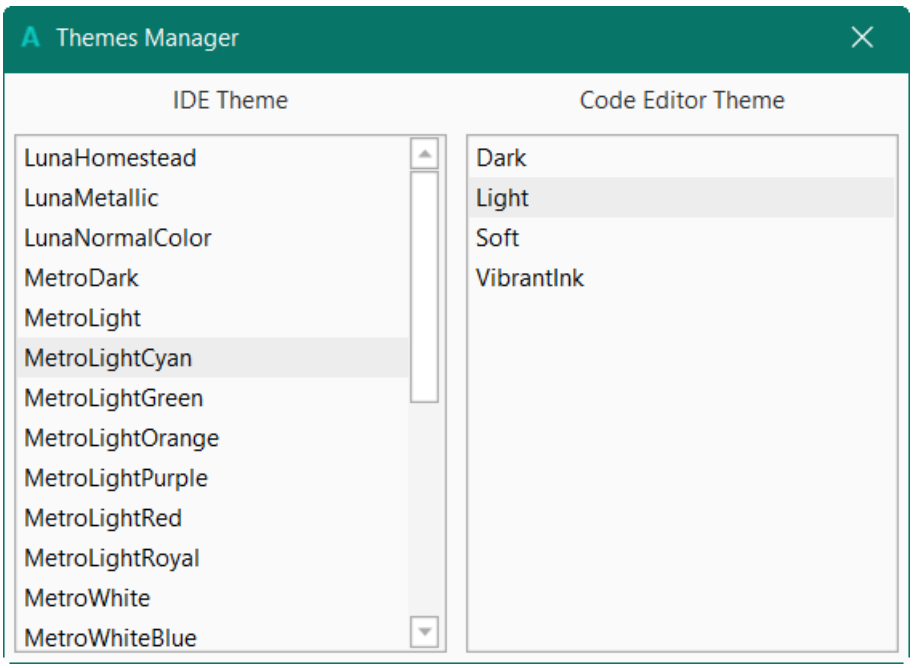
3.6.1.1 Language

You can select the language of the IDE in the menu Tools / IDE Options / Language.



Select the desired language in the list of the currently available languages.

3.6.1.2 Themes

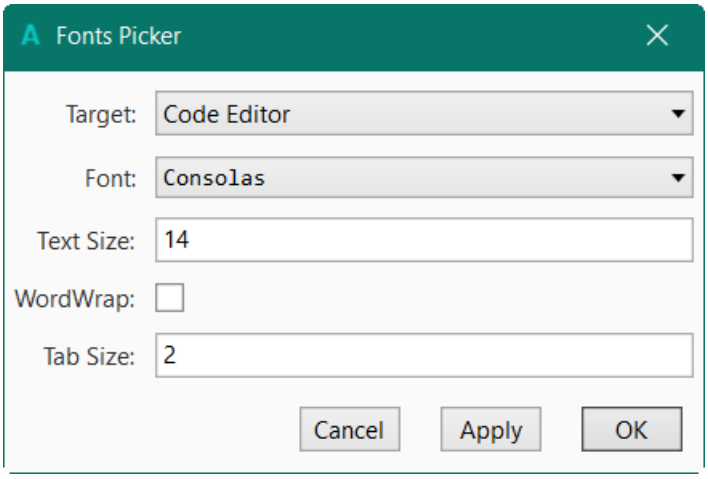


You can select different themes for the IDE.

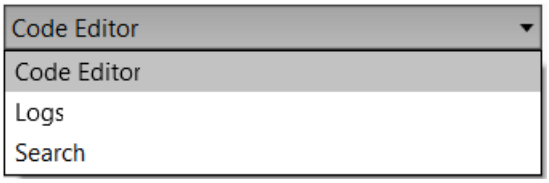
The default theme is different for the different B4X products.

When you select one you see directly the new colors.

3.6.1.3 Font Picker



You can select the target Code Editior, Logs or Search.



Different fonts.
Enter the text size.
Select WordWrap
Enter the Tab size.

3.6.1.3.1 Word wrap

```
53    lblComments.Text = "Enter the result" & CRLF & "and click
54
```

Without word wrap. The end of the line is hidden.

```
53    lblComments.Text = "Enter the result" & CRLF & "and click
on OK"
```

With word wrap. The end of the line is wrapped to the next line.

3.6.1.4 Auto Save

Saves the project at each run when checked.

3.6.1.5 Auto Backup

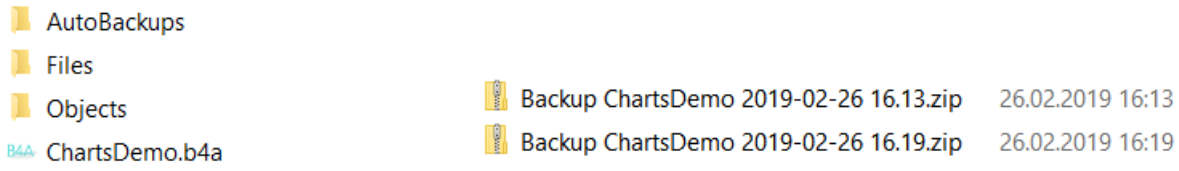
Auto Backup saves a backup project as a zip file.

The zip file created is the same zip that you will get with File - Export as zip.
It creates a new zip every 10 minutes (when there are changes).
It automatically deletes older backups based on a set of internal rules.

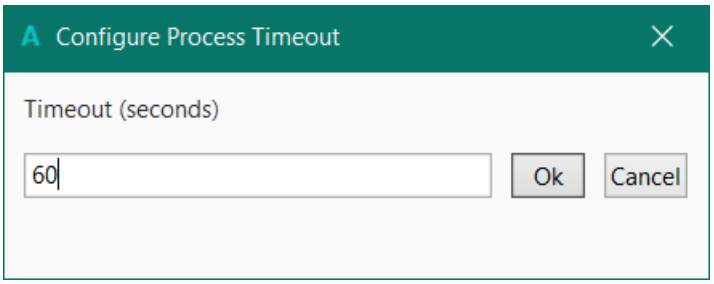
The frequency of kept files is lowered based on the files age. It starts with one file per 10 minutes and ends with one file every two months after 6 months.

It can be disabled if needed.

These backup files are saved in the AutoBackups folder in the project folder.
The folder name includes the program name and the update date.

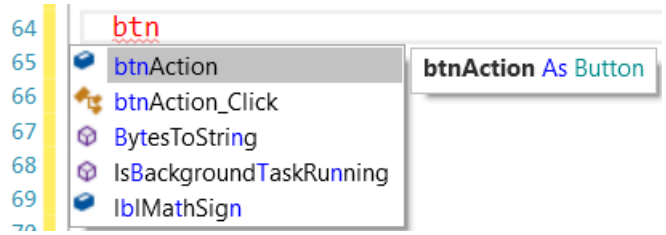


3.6.1.6 Configure Process Timeout



Sometimes the compilation needs more time. If you get a message 'Process timeout' you can increase the time.

3.6.1.7 Disable Implicit Auto Completion



If 

is unchecked you will see a drop down list with possible words during typing.

If checked 

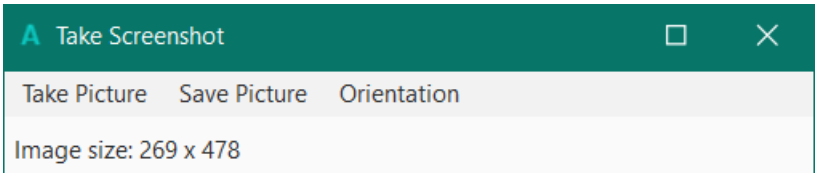
you won't see the auto completion list.

3.6.2 Take Screenshot B4A only

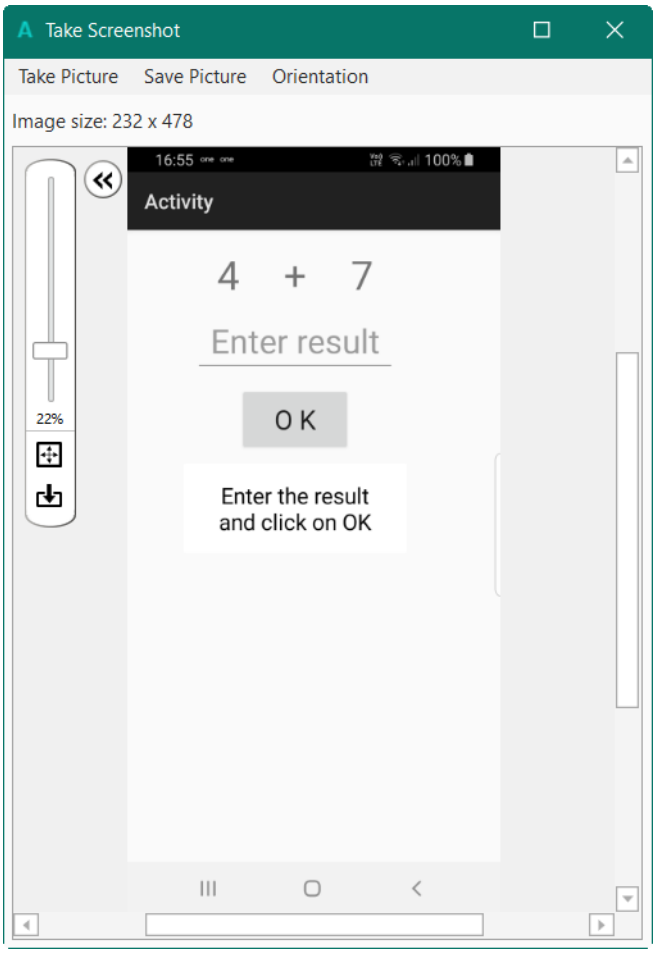
The Take Screenshot function can be called from the:

- Tools menu when the IDE is in edit mode
- Debug menu when the IDE is in debug mode

Note: This function works only with USB connetion not with B4A-Bridge !



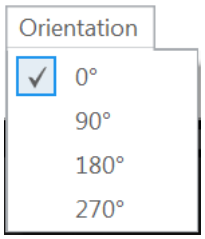
Clicking on Take Screenshot shows this window.



Click on Take Picture to take the screenshot picture from the device.

You can resize the image with the cursor on the left side.

You can save the image with Save Picture as a PNG file.



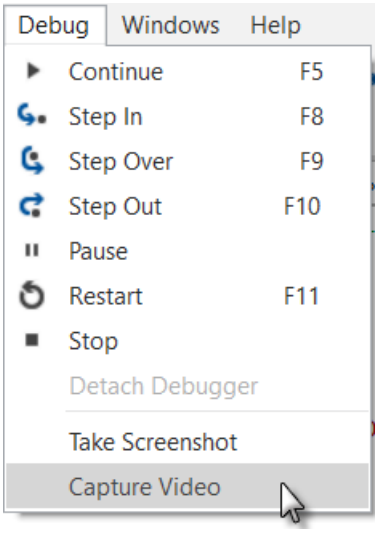
And you can change the orientation of the picture.

Right click on the image to copy the image to the clipboard.

3.6.3 Create Video B4A only

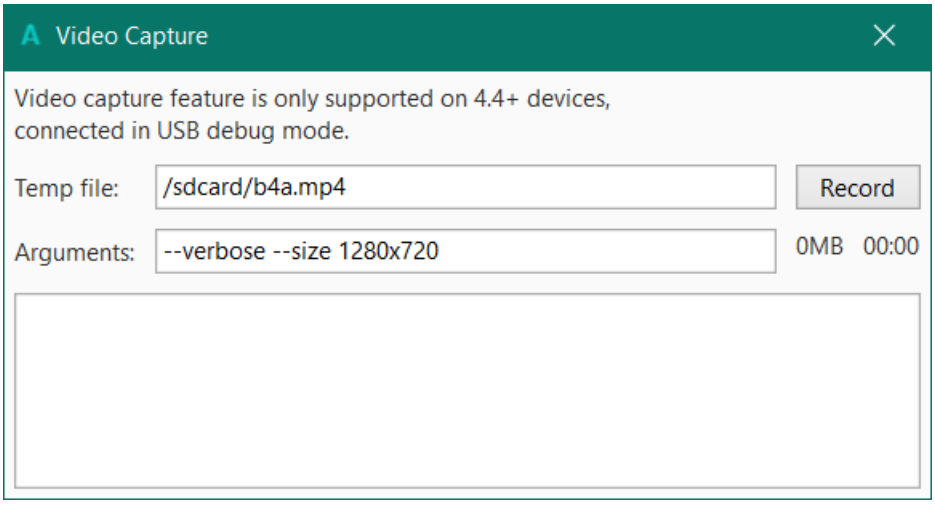
You can run your program and record a video when you use it.

Note: This function works only with USB connetion not with B4A-Bridge !



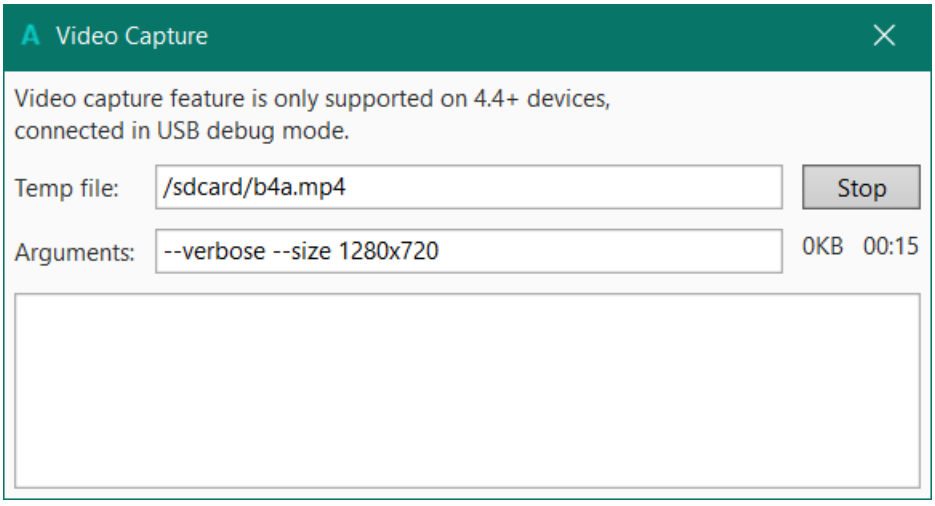
In the Debug menu click on Capture Video.

The sceen below will be dispayed:



Click on Record to begin recording.

A screen similar to this one will be dispaled:



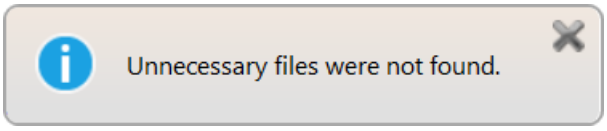
Click on Stop to stop recording.

You will be asked where you want to save the file on the computer.

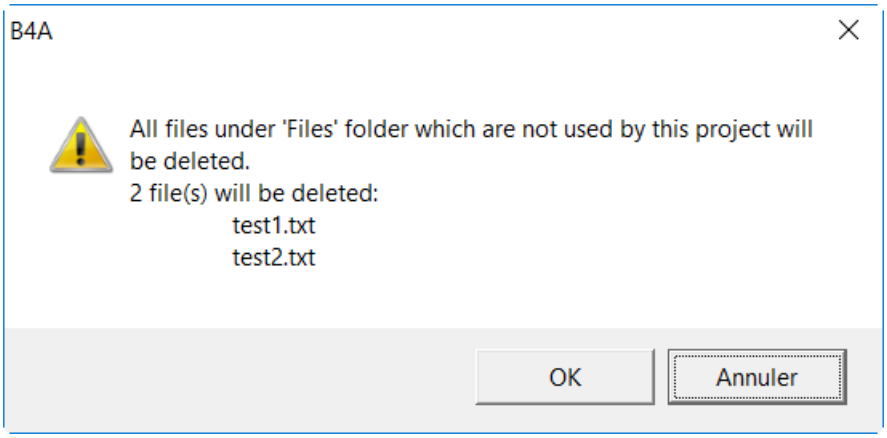
3.6.4 Clean Files Folder (unused files)

Deletes files that are located under the Files folder but are not used by the project (it will not delete any file referenced by any of the project layouts). A list of unused files will be displayed before deletion (and you may cancel the operation).

If there are no unused files the message below will be displayed.



If there are unused files, a window like the one below will be displayed.



3.6.5 Clean Project

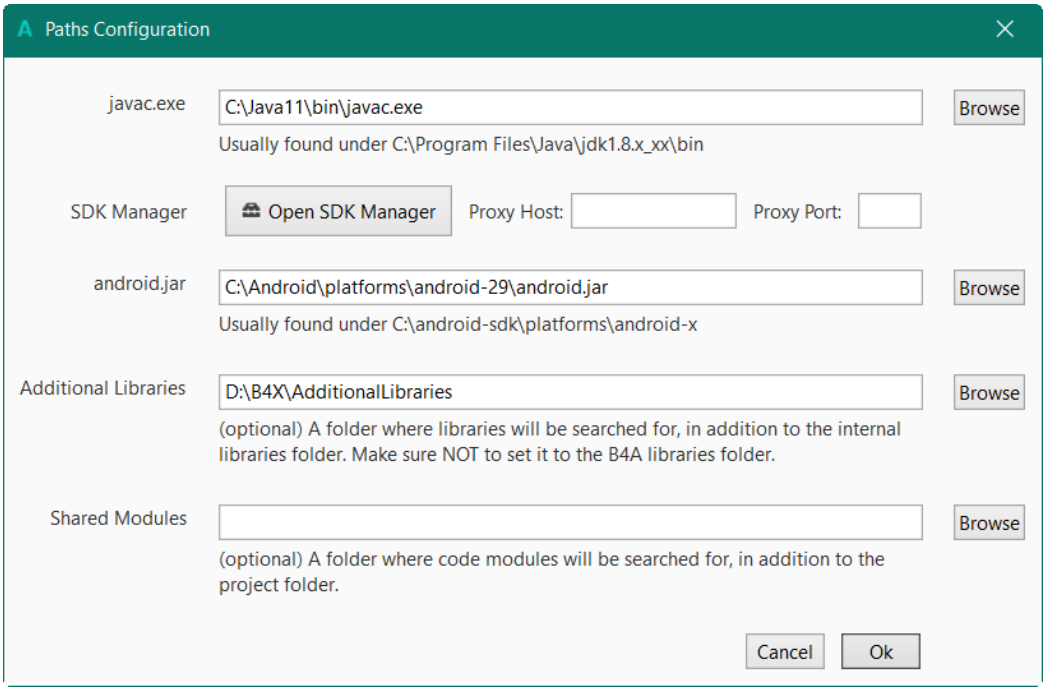
Deletes all files that are generated during compilation in Debug mode.

Sometimes it is useful to ‘clean’ the project if the compilation or the program slows down.
Or if you have two or more IDEs open at the same time and want to run the projects on the same device.

3.6.6 Configure Paths

You need to configtre several paths for the IDE to run.

3.6.6.1 B4A



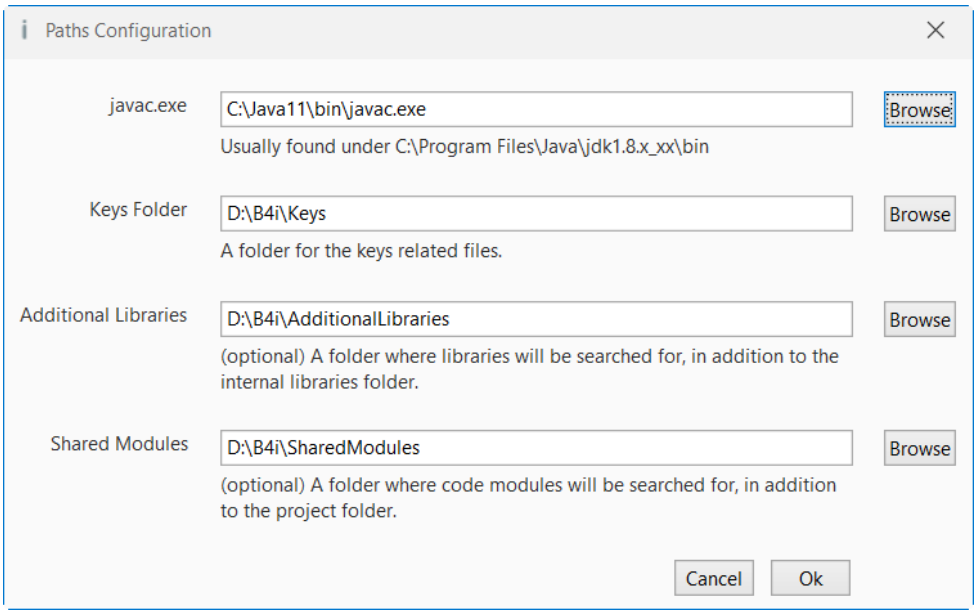
java.exe

Explained in the B4X Getting Started Booklet.

Additional Libraries

Shared Modules

3.6.6.2 B4i



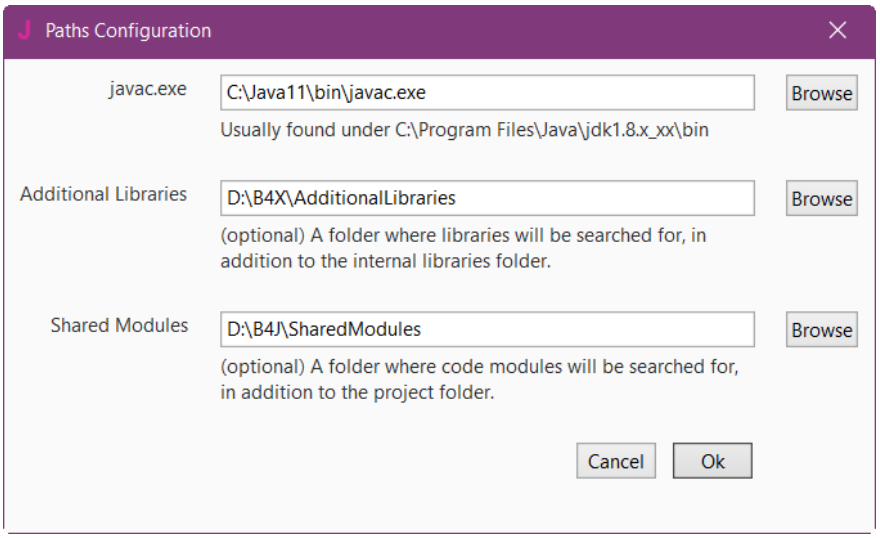
java.exe

Explained in the B4X Getting Started Booklet.

Additional Libraries

Shared Modules

3.6.6.3 B4J

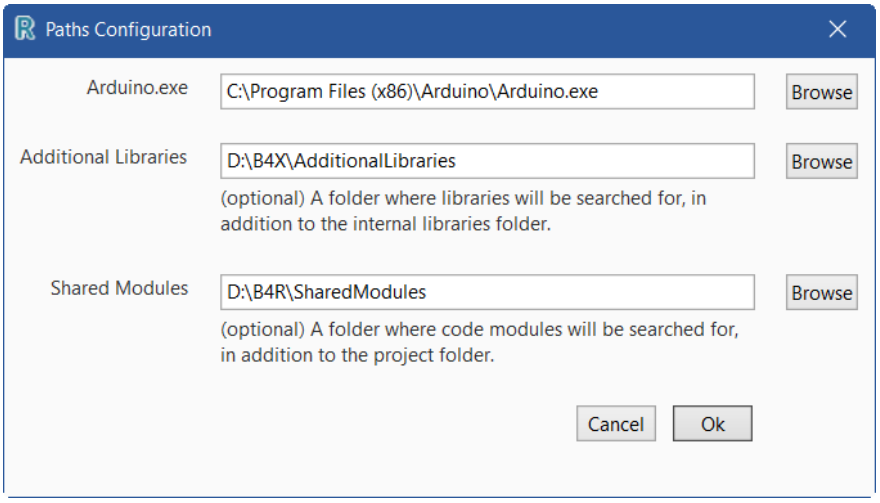


java.exe

Additional Libraries

Shared Modules

3.6.6.4 B4R



Explained in the B4X Getting Started Booklet.

Additional Libraries

Shared Modules

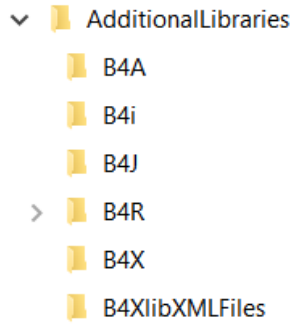
3.6.6.4.1 java.exe B4A / B4i / B4J

This is the loation of the java.exe file.
This has already be done when you installed one of the products.
Depending on which version of Java you installed (8 or 11), it should look similar to this:
Java 11:
C:\Java11\bin\javac.exe,
or *C:\java\bin\javac.exe* depending in which folder you installed Java 11.
Java 8:
C:\Program Files\Java\jdk1.8.x.xxx\bin, 64 bit version.
or *C:\Program Files\Java\jdk1.8.x.xxx\bin* 32 bit version.

3.6.6.4.2 Additional Libraries

You must define a folder for additional libraries.

This folder must have following structure:



Folder for B4A additional libraries.
Folder for B4i additional libraries.
Folder for B4J additional libraries.
Folder for B4R additional libraries.
Folder for B4X libraries.
Folder for B4X libraries XML files.

One subfolder for each product: B4A, B4i, B4J, B4R and another B4X for B4X libraries.

When you install a new version of a B4X product, all standard libraries are automatically updated, but the additional libraries are not included. The advantage of the special folder is that you don't need to care about them because this folder is not affected when you install the new version of B4X. The additional libraries are not systematically updated with new version of B4X.

When the IDE starts, it looks first for the available libraries in the Libraries folder of B4X and then in the additional libraries folders.

In my system, I added a B4XlibXMLFiles folder for XML help files for B4X libraries (b4xlib). The standard and additional libraries have an XML file. B4X Libraies not.

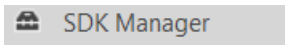
But, if you use the B4X Help Viewer you would be interested in having these help files if they are available. The B4X Help Viewer is explained in the B4X Help tools booklet.

3.6.6.4.3 Shared Modules

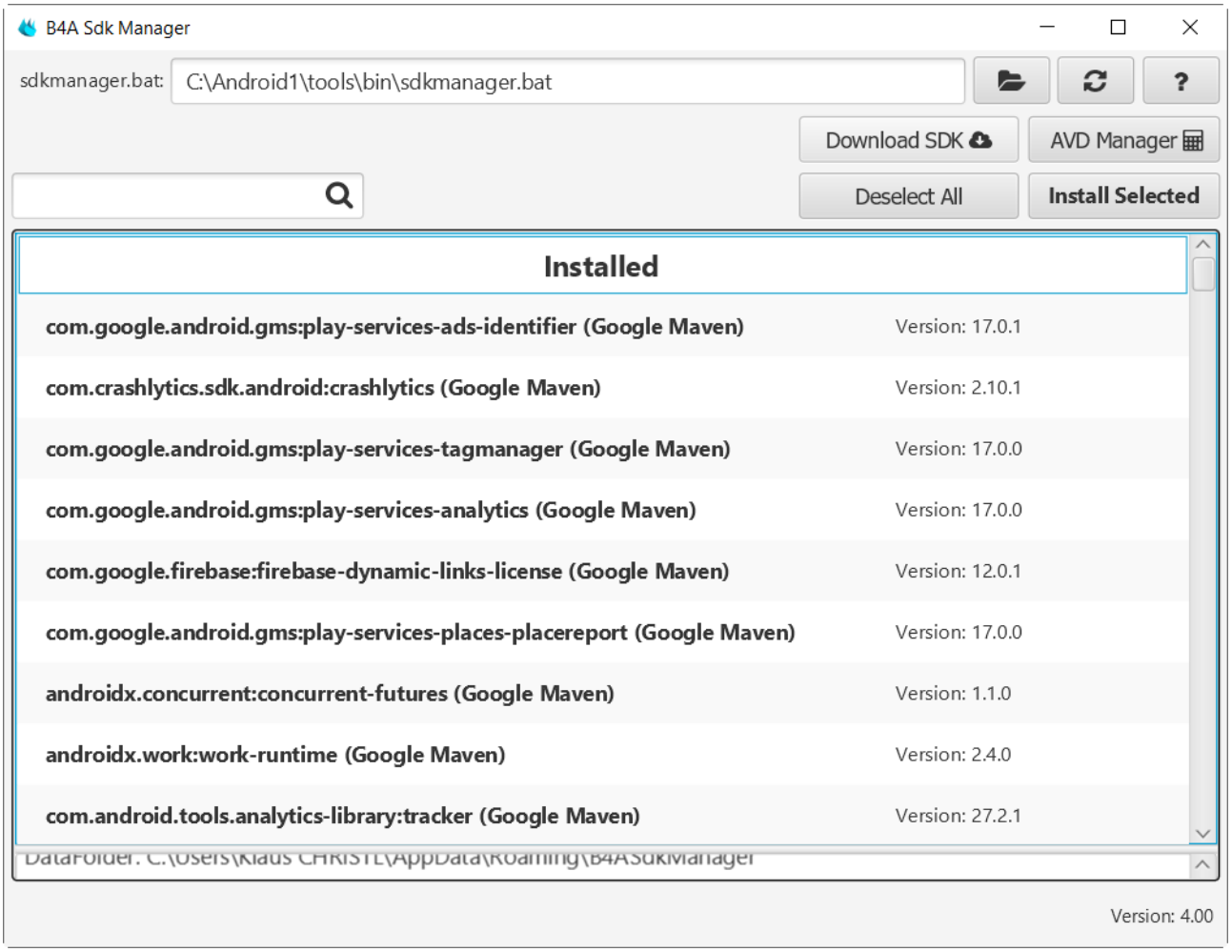
You can add a folder for Shared Modules.
Shared Modules are almost not needed anymore.
This can better be replaced with classes or CustomViews.

3.6.7 SDK Manager B4A only

The SDK Manager can be used to update the Android SDK.

Click on 

to show the SDK Manager window, which can look like the image below:



If you see **Installed** on top, it means that your SDK is up do date.

Even if you see on top Recommended, do not click on 

to install the recommended files, until Erel does recommend it in the Forum !

3.6.8 Jetifier AndroidX B4A only

Android Support Library was implemented and maintained in the last 8 years by Google. It includes a wide range of features and it hides many of the differences between the various Android versions.

Android Support Library was replaced by AndroidX SDK. It is no longer maintained and new versions of Firebase SDK depend on AndroidX. This is a good time to switch to AndroidX.

It is simple to switch to AndroidX as the IDE takes care of most of the things:

- Libraries references (DependsOn / #AdditionalJar) are updated automatically.
- The compiler automatically chooses the androidx libraries if such are available.
- References to support classes in #Extends declarations, JavaObject calls and to a less extent in the manifest editor code are converted during compilation.

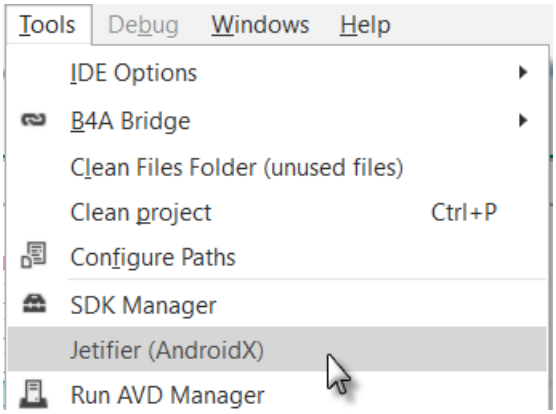
Most of the libraries will work as-is, however libraries that directly call methods from the support library need to be "jetified". This is done with the Jetifier tool (Tools - Jetifier). It will go over all the additional libraries and will jetify all the libraries that need to be jetified. The output of this tool is libraries with .androidx.jar (or .androidx.aar) extension.
The B4A compiler will then use those libraries automatically.

Note that the internal libraries already include the jetified versions.

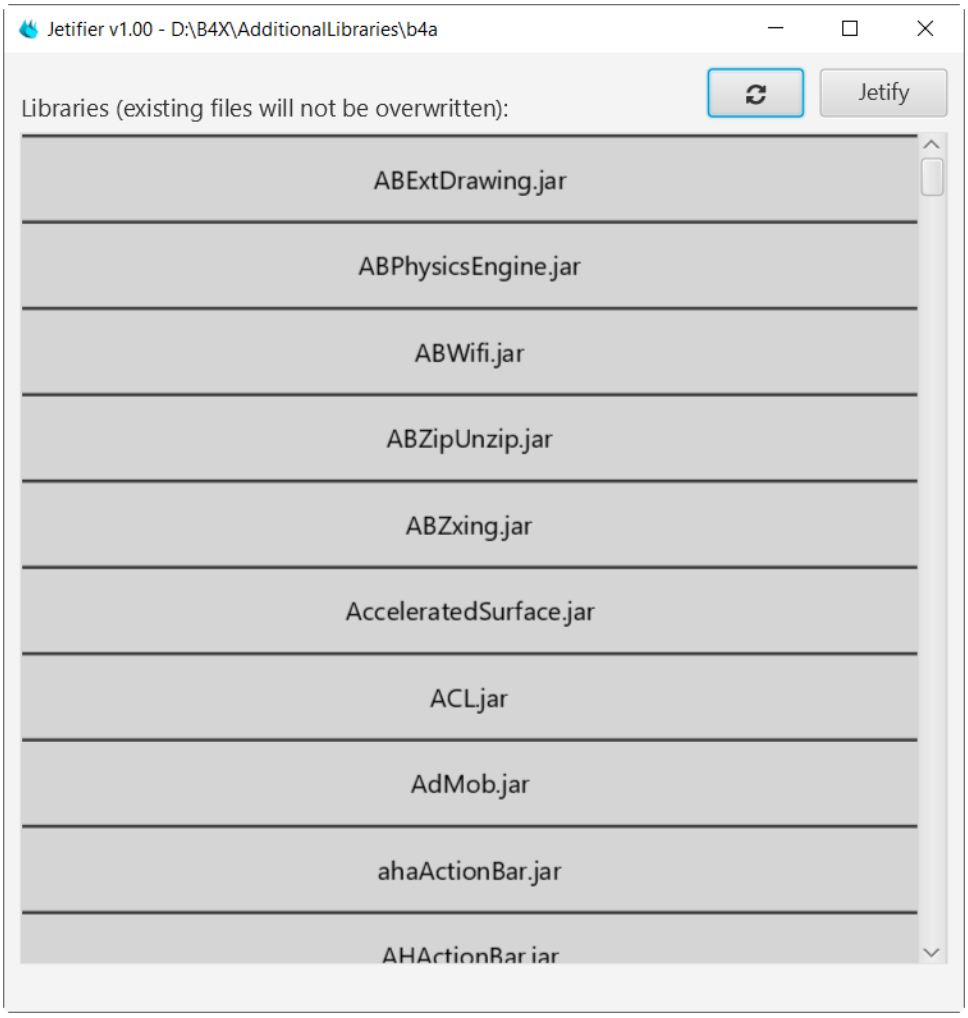
To conclude:

1. Open Tools - B4A Sdk manager.
2. Install all recommended items. This will make the switch to AndroidX. You can always go back to the previous SDK.
3. Open Tools - Jetifier.
4. Click on Jetify. It is possible that the jetifier will fail to jetify a few libraries. In most cases you can ignore it as those libraries don't need to be jetified.
5. Run your project. The compilation dialog will tell you which SDK was used:

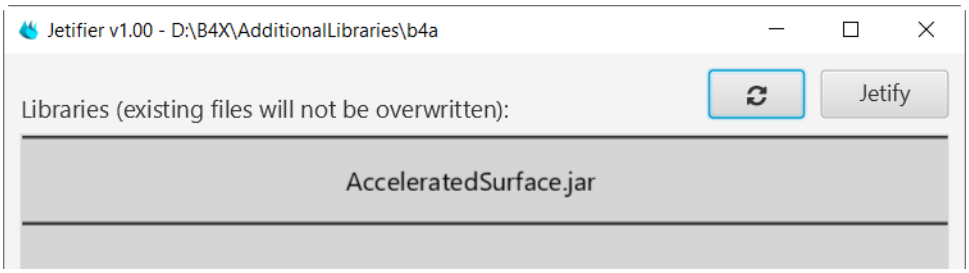
B4A Version: 9.30 BETA #2
Java Version: 11
Parsing code. (0.00s)
Building folders structure. (0.01s)
Compiling code. (0.02s)
Compiling layouts code. (0.01s)
Organizing libraries. (0.00s)
(AndroidX SDK)
Generating R file. (0.96s)
Compiling debugger engine code. (0.73s)
Compiling generated Java code. (1.36s)
Convert byte code - optimized dex. (1.06s)
Packaging files. (0.83s)
Copying libraries resources (0.01s)
Signing package file (private key). (0.61s)
ZipAlign file. (0.04s)
Installing file to device. (0.03s)
Installing with B4A-Bridge.
Completed successfully.



When you click on Jetifier (AndroidX), you will be shown the window below.

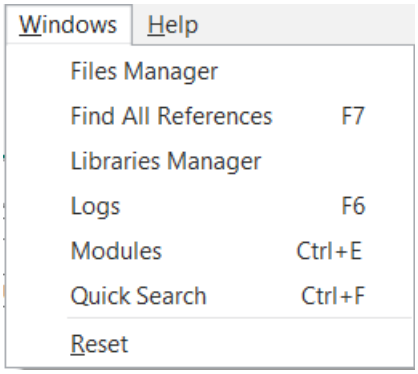


After clicking on , Jetify I got this, the AcceleratedSurface.jar was not 'jetified' because of it's length.



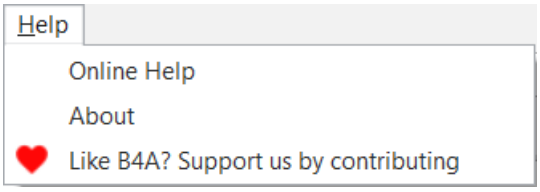
3.7 Windows menu

The windows menu shows the Tabs.



Click on Reset to get the default setting.

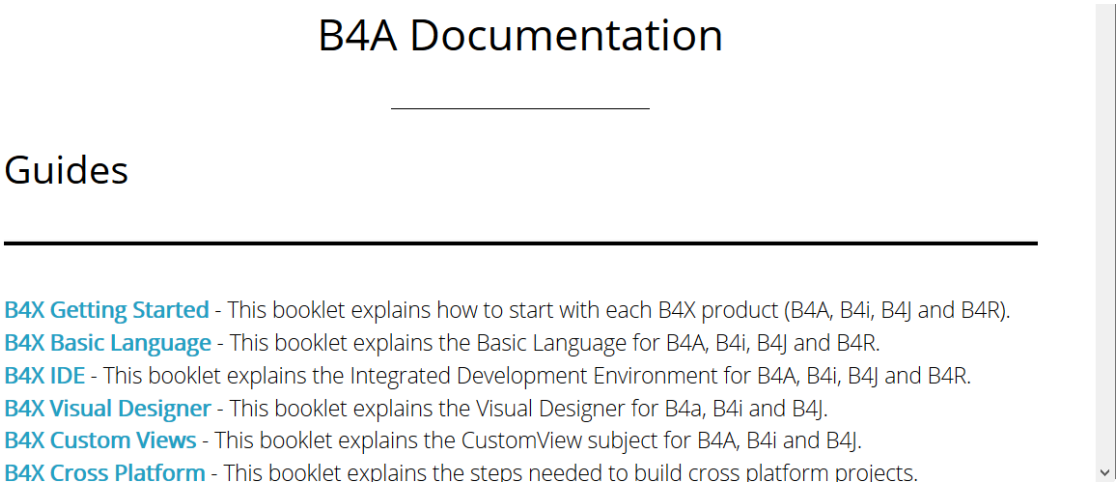
3.8 Help menu



Online Help.
About.
Like B4A? Support us by contributing.

3.8.1 Online Help

Leads you to the Documentation page of B4X site.

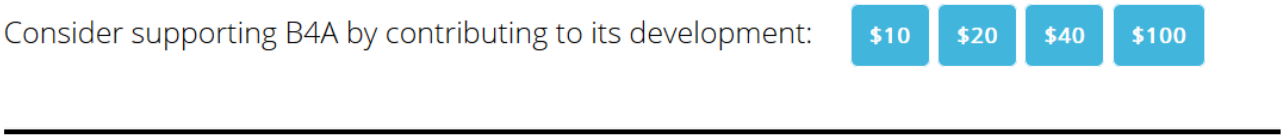


3.8.2 About

Shows the About window with the Version number and a link the the B4X site.

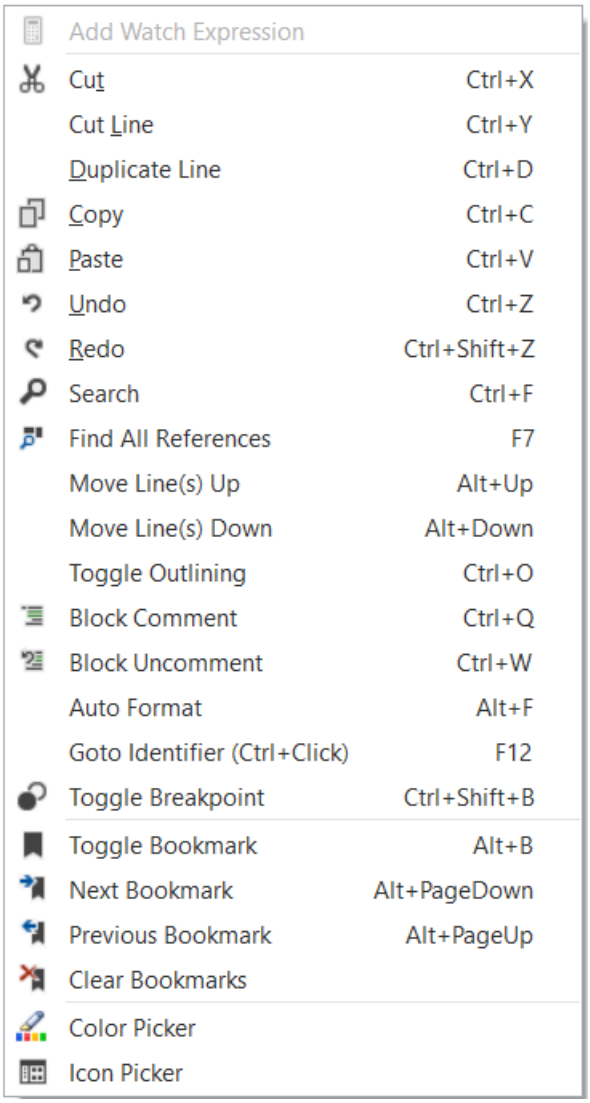
3.8.3 Like B4A? Support us by contributing

Leads you to the main platform page of the B4X site where you can contribute to help Anywhere Software continuing the developmenet of the B4X products.



3.9 Right click menu

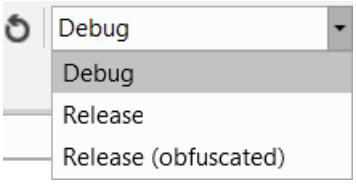
When you right click in the code area the menu below is displayed.

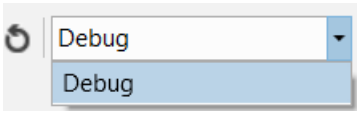


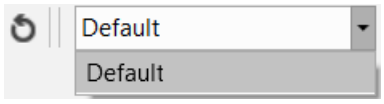
Cut
Cut Line
Duplicate Line
Copy
Paste
Undo
Redo
Search
Find All References
Move Line(s) Up
Move Line(s) Down
Toggle Outlining
Block Comment
Block Uncomment
Auto Format
Goto identifier
Toggle Breakpoint
Toggle Bookmark
Previous Bookmark
Next Bookmark
Clear Bookmark
Color Picker
Icon Picker Not in B4R.

3.10 Compiler mode

Besides the toolbar there is a drop down list to select the compiler mode.







B4A / B4J B4i B4R

Debugging is explained in detail in the Debugging chapter.

3.10.1 B4A and B4J

Compiling modes:
- Debug
- Release
- Release (obfuscated)

3.10.1.1 Release and Release (obfuscated) modes B4A and B4J

To distribute your project you must compile it with:
- Release
  The debugger code will not be added to the apk file.
- Release (obfuscated)
  The debugger code will not be added to the apk file,
  but the program file will be modified. See below.

During compilation B4A generates Java code which is then compiled with the Java compiler and converted to Dalvik (Android byte code format).
There are tools that allow decompilation of Dalvik byte code into Java code.

The purpose of obfuscation is to make the decompiled code less readable, harder to understand and make it more difficult to extract strings like developer account keys.

It is important to understand how the obfuscator works.
The obfuscator does two things:

Strings obfuscation
Any string written in Process_Globals sub (and only in this sub) will be obfuscated, making it much harder to extract important keys. The strings are deobfuscated at runtime.
Note that several keys are used during obfuscation including the package name, version name and version code. Modifying these values with the manifest editor will break the deobfuscation process.

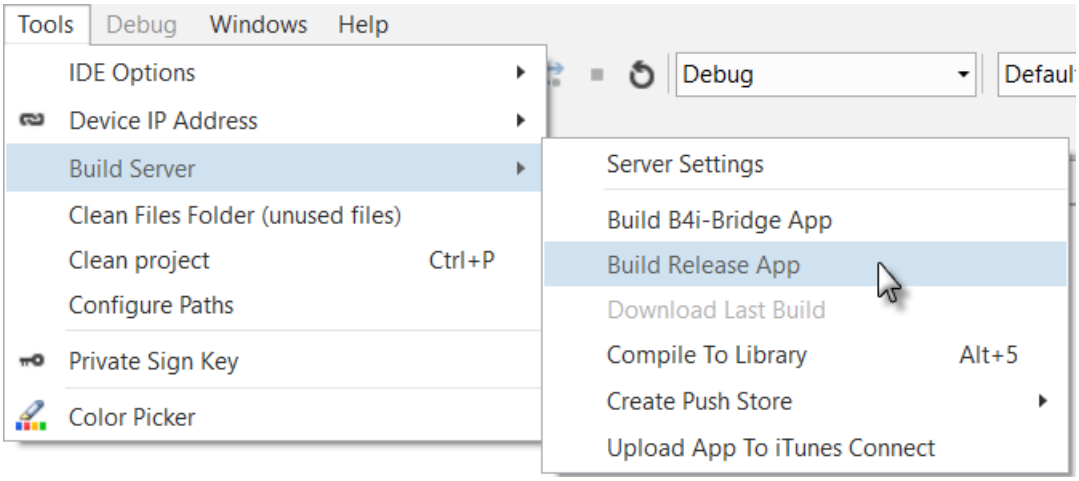
Variables renaming
The names of global variables and subs are converted to meaningless strings. Local variables are not affected as their names are lost anyway during the compilation.
The following identifiers are **not** renamed:
- Identifiers that contain an underscore (required for the events handlers).
- Subs that appear in CallSub statements. When a sub name appears as a static string, the identifier be kept as it is.
- Designer views names.

Tip: If, for some reason, you wish to prevent obfuscation of an identifier, include an underscore character in the name.

A file named ObfuscatorMap.txt will be created under the Objects folder. This file maps the original identifiers names to the obfuscated names. This mapping can be helpful in analysing crash reports.

3.10.2 B4i

To distribute a project you must compile it in Release mode.
Click on Build Release App in the Tools / Build Server menu.



3.10.3 B4R

Only Default mode.

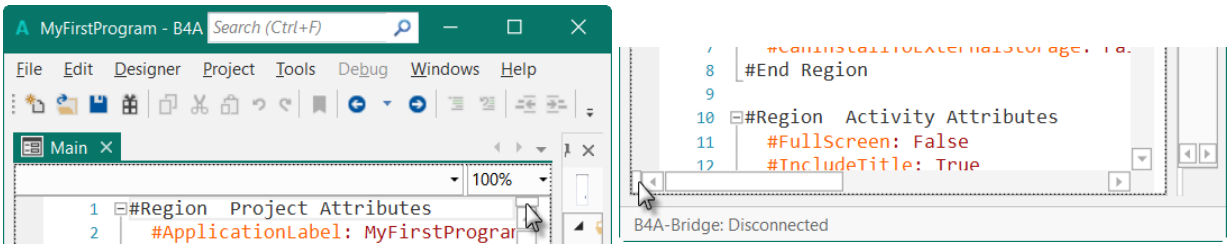
4 Code area

The code of the selected module is displayed in this area and can be edited.
The examples below are based on the code of the SecondProgram in the GettingStarted booklet.

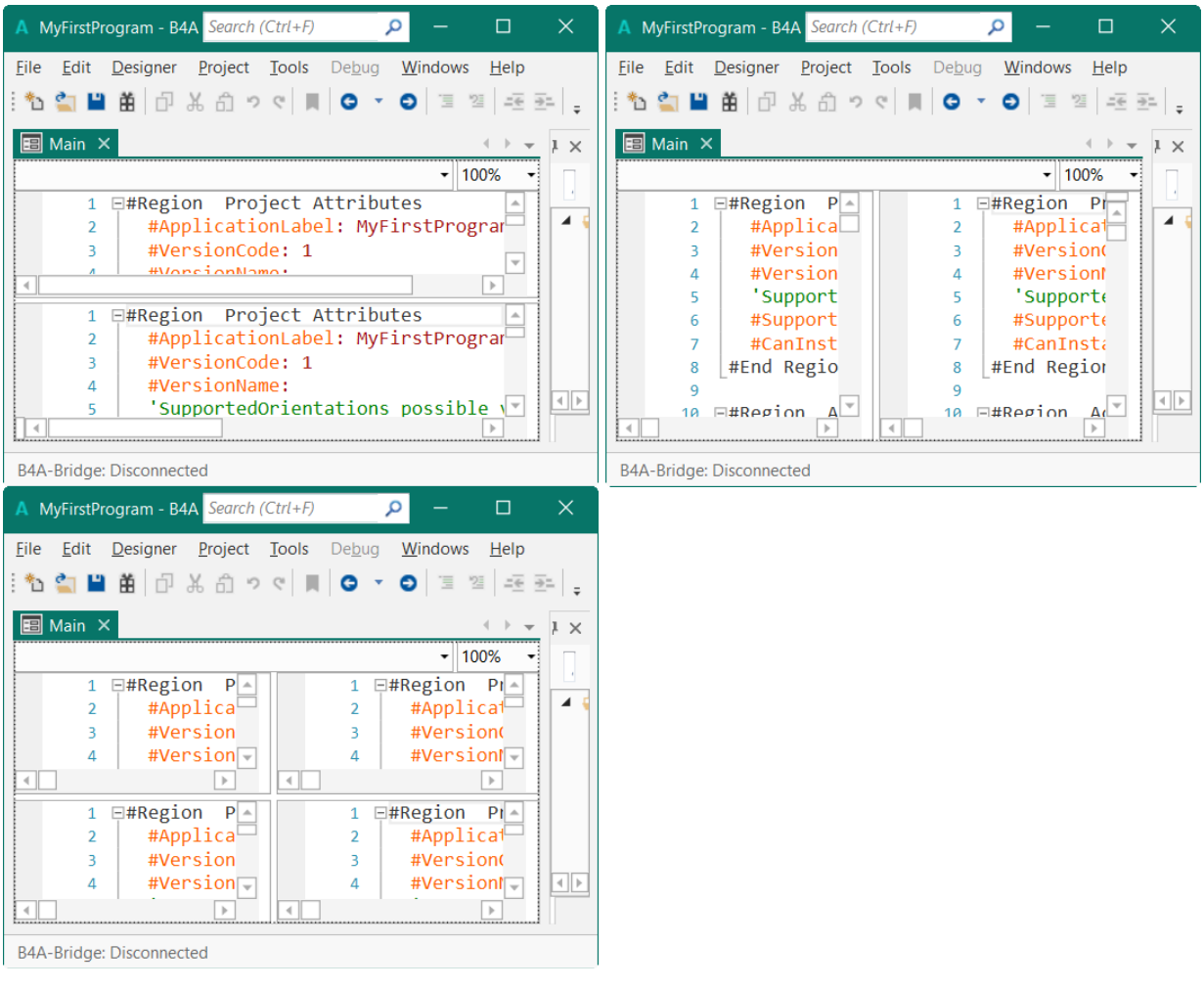
4.1 Split the code area

It is possible to split the code area into two or four parts allowing to edit two or four different code parts at the same time.

Move the small rectangle below the zoom level or in the lower left corner.



And the result.

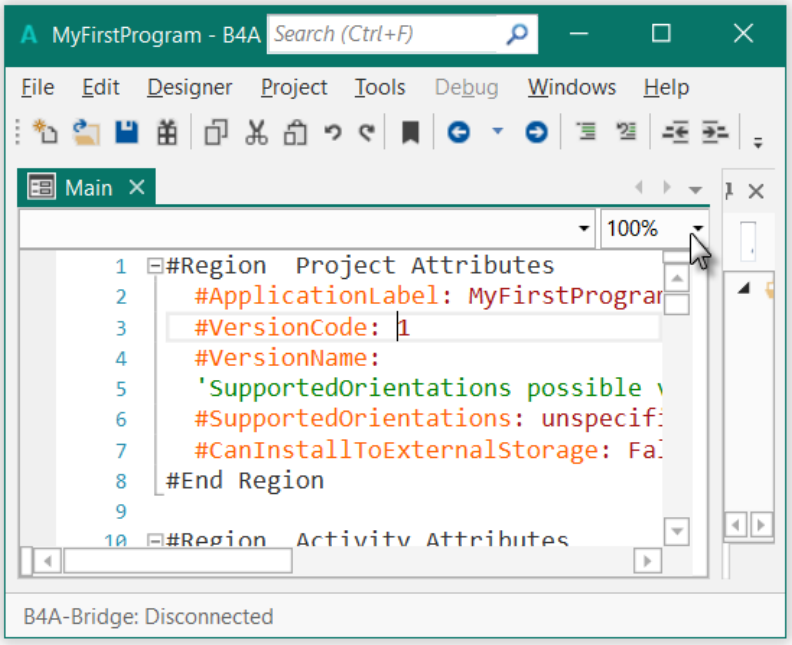


4.2 New version available

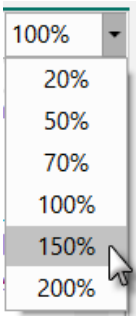
When a new version is available you are informed at the bottom of the IDE.
If you click on **B4A v10.0 is available for download** and you will be led to the download page in the B4X site.

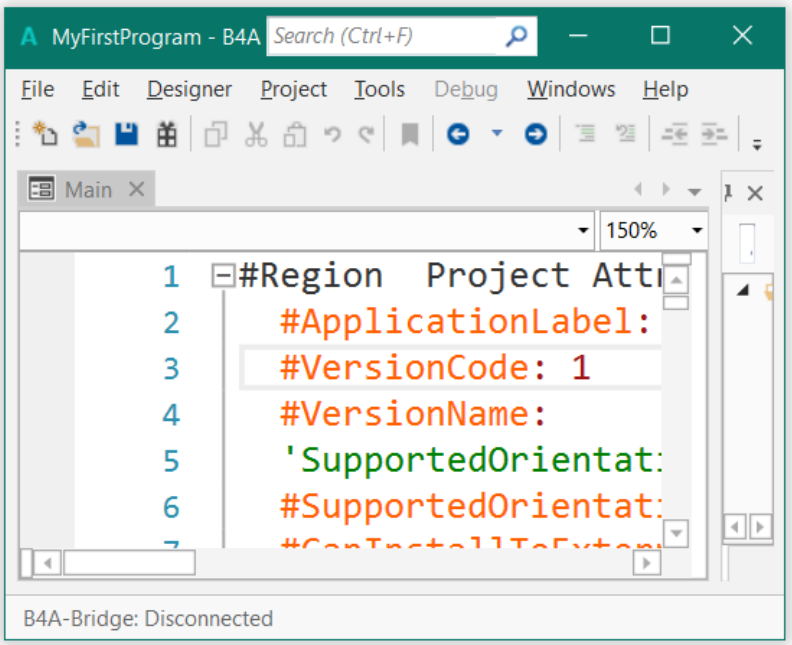
4.3 IDE text size

The IDE text size can be changed with the FontPicker or directly in the IDE:



Click on 100% and select one of the zoom values.





4.4 Code header Project Attributes / Activity Attributes

A code header, with general settings, is added at the beginning of the code.

4.4.1 B4A

4.4.1.1 Project Attributes

Attributes that are valid for the whole project. Displayed only in the Main module.

```
#Region  Project Attributes
   #ApplicationLabel: SecondProgram
   #VersionCode: 1
   #VersionName:
   'SupportedOrientations possible values: unspecified, landscape or portrait.
   #SupportedOrientations: unspecified
   #CanInstallToExternalStorage: False
#End Region
```

`#ApplicationLabel:` The name which will be displayed below the program icon on the device.
`#VersionCode:` The version of the code, it is not displayed.
`#VersionName:` You can add a name for the version.
`#SupportedOrientations:` You can limit the whole program to a given orientation.
`#CanInstallToExternalStorage:` If you want to install the program on an external storage card you must set this attribute to True.

You can add or change the values to your needs.

4.4.1.2 Activity Attributes

Valid for the current activity.

```
#Region  Activity Attributes
   #FullScreen: False
   #IncludeTitle: True
#End Region
```

When you add a new Activity you'll find the Activity Attributes region on top.

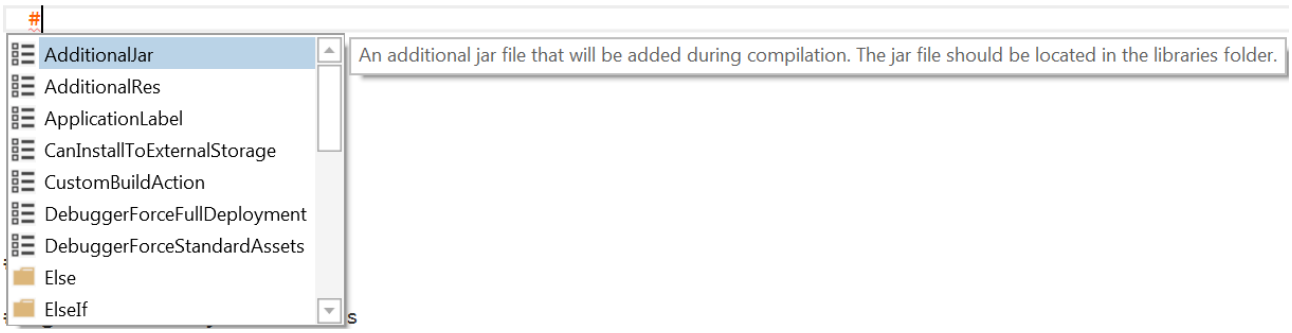
```
#Region  Activity Attributes
   #FullScreen: False
   #IncludeTitle: True
#End Region
```

4.4.1.3 Service Attributes

When you add a new Service you'll find the Service Attributes header.

```
#Region  Service Attributes
   #StartAtBoot: False
#End Region
```

When you want to add a new Attribute you can just write # and the inline help shows all possibilities.



Note the two different icons:

 Attributes.

 Conditional compilation and region keywords.

When you load a project saved with a version of B4A older than 2.5 then the header will look like this:

```
#Region Module Attributes
   #FullScreen: False
   #IncludeTitle: True
   #ApplicationLabel: MyFirstProgram
   #VersionCode: 1
   #VersionName:
   #SupportedOrientations: unspecified
   #CanInstallToExternalStorage: False
#End Region
```

4.4.2 B4i

Only the Attributes below. No other Attributes in modules.

```
 'Code module
#Region  Project Attributes
   #ApplicationLabel: B4i Example
   #Version: 1.0.0
   'Orientation possible values: Portrait, LandscapeLeft, LandscapeRight and
PortraitUpsideDown
   #iPhoneOrientations: Portrait, LandscapeLeft, LandscapeRight
   #iPadOrientations: Portrait, LandscapeLeft, LandscapeRight, PortraitUpsideDown
   #Target: iPhone, iPad
   #ATSEnabled: True
   #MinVersion: 7
#End Region
```

4.4.3 B4J

Only the two Attributes below. No other Attributes in modules.

```
#Region Project Attributes
   #MainFormWidth: 600
   #MainFormHeight: 600
#End Region
```

4.4.4 B4R

Only the Attributes below. No other Attributes in modules.

```
#Region Project Attributes
   #AutoFlushLogs: True
   #CheckArrayBounds: True
   #StackBufferSize: 300
#End Region
```

4.5 Undo – Redo

In the IDE it is possible to undo the previous operations and redo undone operations.
Click on to undo and on to redo.

4.6 Collapse a subroutine

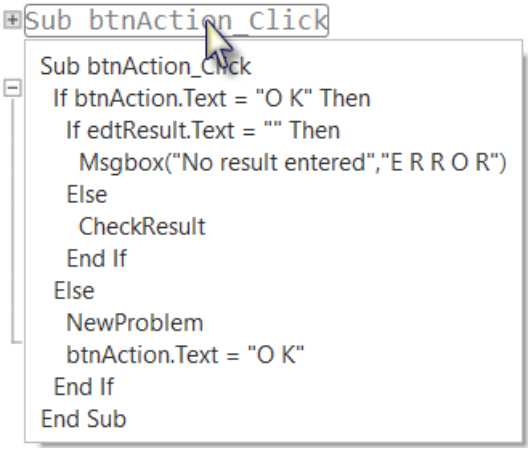
A subroutine can be collapsed to minimize the number of lines displayed.

```
Sub btnAction_Click
  If btnAction.Text = "O K" Then
    If edtResult.Text = "" Then
      Msgbox("No result entered","E R R O R")
    Else
      CheckResult
    End If
  Else
    NewProblem
    btnAction.Text = "O K"
  End If
End Sub
```

The btnAction_Click routine expanded.

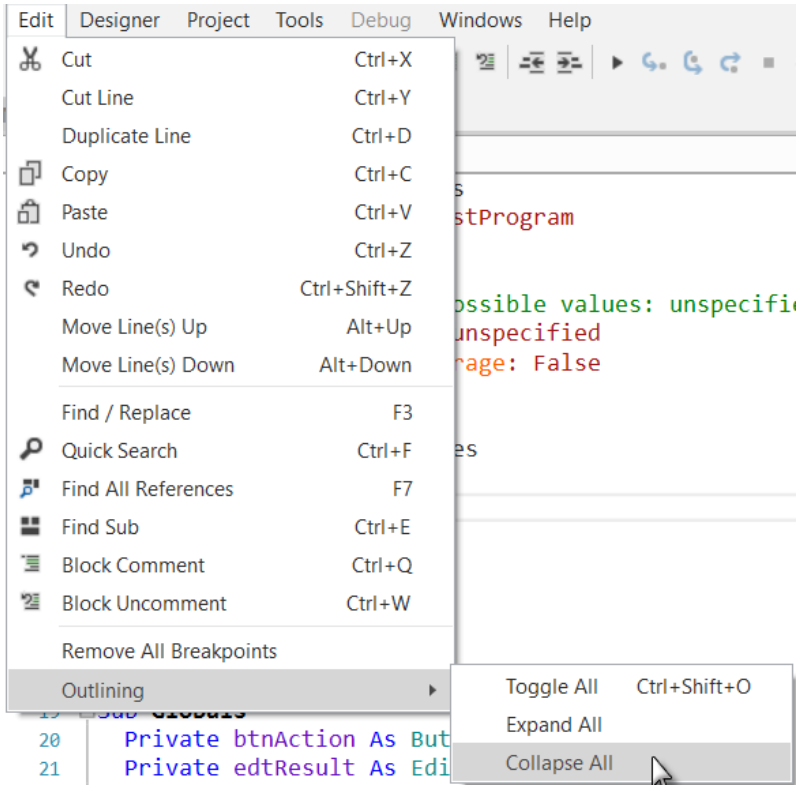
Click on to collapse the subroutine.

The btnAction_Click routine collapsed.



Hovering with the mouse over the collapsed routine name shows its content.

4.7 Collapse the entire code



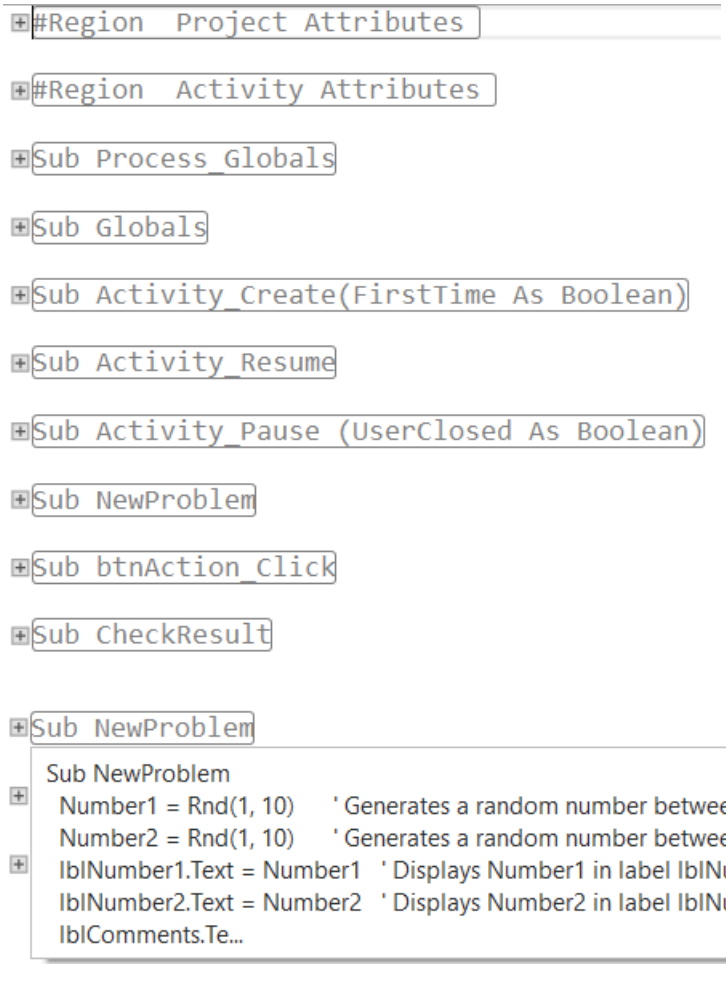
In the Edit / Outlining menu there are three functions:

- Toggle All
Expands the collapsed routines and collapses the expanded routines and regions.

- Expand All
Expands the entire code.

- Collapse All
Collapses the entire code.

Click on Collapse All.



The whole code collapsed.

Hovering with the mouse over a subroutine shows the beginning of its content.

4.8 #Regions

You can define ‘Regions’ in the code, which can be collapsed.

Example:

```
#Region GPS

#End Region
```

`#Region GPS` sets the beginning of a region and `#End Region` the end.

```
#Region GPS
Private Sub Routine1
End Sub
Private Sub Routine2
End Sub
Private Sub Routine3
End Sub
#End Region
```

Then you can add subroutines between the two limits.

```
#Region GPS
Private Sub Routine1
```

Then click on ⊟ to collapse the whole region.

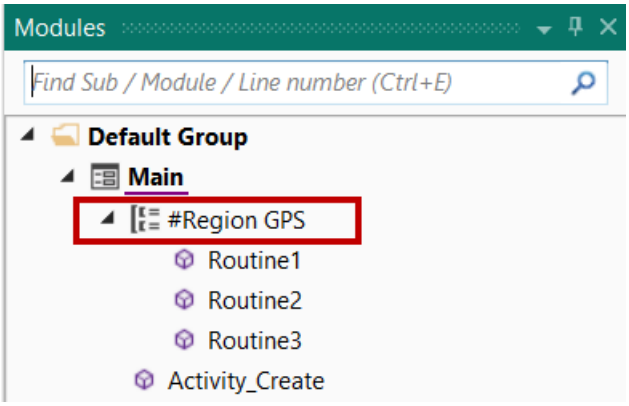
```
#Region GPS
```

Hovering over #Region GPS

```
#Region GPS
#Region GPS
Private Sub Routine1
End Sub
Private Sub Routine2
End Sub
Private Sub Routine3
End Sub
#End Region
```

shows the code. For big regions not all the code is displayed.

The regions are also displayed in the Modules Tab.



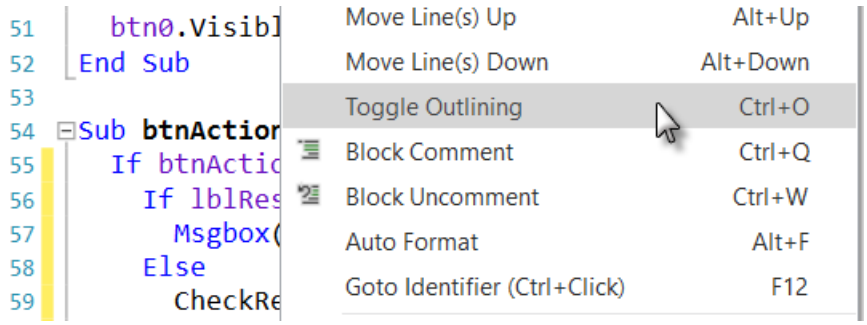
4.9 Toggle Outlining Ctrl + 0

You can toggle code outlining.

Example:

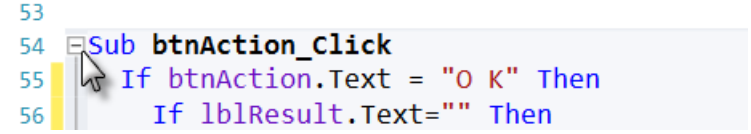
```
Sub btnAction_Click
   If btnAction.Text = "O K" Then
      If lblResult.Text="" Then
       Msgbox("No result entered","E R R O R")
      Else
       CheckResult
      End If
   Else
      NewProblem
      btnAction.Text = "O K"
      lblResult.Text = "" & Chr(0xE632)
   End If
End Sub
```

Click insides the routine and press Ctrl + 0.
Or right click insides the routine to show the pop-up menu and click on Toggle Outlining to collapse the routine.



And the result.

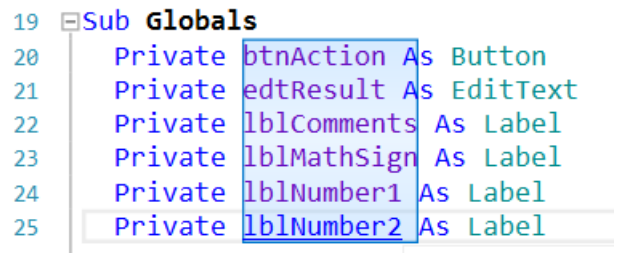
It is the same as clicking on ⊟.



4.10 Copy a selected bloc of text

It is possible to copy a selected bloc of text to the clipboard, not only entire lines.

To select the bloc press Alt and move the mouse cursor.



4.11 Move line(s) up / down Alt + Up / Alt + Down

You can move selected lines up or down.
Either with Alt + Up or Alt + Down.
Or right click on the selected lines and select Move Line(s) Up or Move Line(s) Down.

4.12 Find / Replace

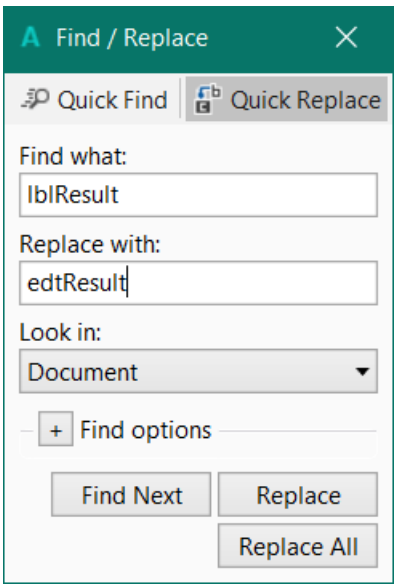
The example uses the code from the SecondProgram project.

Let's replace `lblResult` by `edtResult`.

```
19 Sub Globals
20   Dim btnAction, btn0 As Button
21   Dim lblResult As Label
22   Dim lblComments As Label
```

In the code select `lblResult`.

Press F3 or click on Find / Replace in the Edit menu.

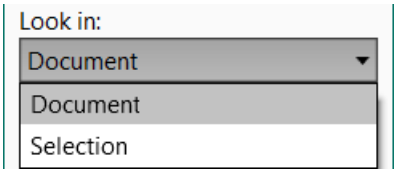


This window will be displayed

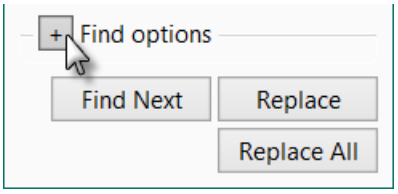
Enter `edtResult` in the 'Replace with' field.

Now, you can either:

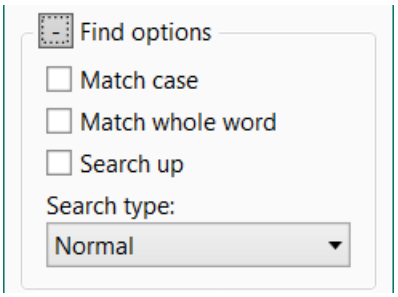
- Find Next find the next occurrence.
- Replace replace the current occurrence and find the next one.
- Replace All replace all occurrencies.



You can search either in a Selection or in the Document, which means in the selected module not the whole document.



You can select Find options, click on +.



These options are self-explanatory.

4.13 Commenting and uncommenting code

A selected part of the code can be set to comment lines or set to normal.

```
20    Private btnAction, btn0 As Button
21    Private lblResult As Label
22    Private lblComments As Label
23    Private lblMathSign As Label
24    Private lblNumber1 As Label
25    Private lblNumber2 As Label
26    Private Number1, Number2 As Int
```

Original code

```
20    Private btnAction, btn0 As Button
21    Private lblResult As Label
22    Private lblComments As Label
23    Private lblMathSign As Label
24    Private lblNumber1 As Label
25    Private lblNumber2 As Label
26    Private Number1, Number2 As Int
```

Select the code.

Click on or Ctrl + Q.

```
20    ' Private btnAction, btn0 As Button
21    ' Private lblResult As Label
22    ' Private lblComments As Label
23    ' Private lblMathSign As Label
24    ' Private lblNumber1 As Label
25    ' Private lblNumber2 As Label
26    ' Private Number1, Number2 As Int
```

The selected lines set as comments.

To set the lines to normal,
select the lines and click on or Ctrl + W.

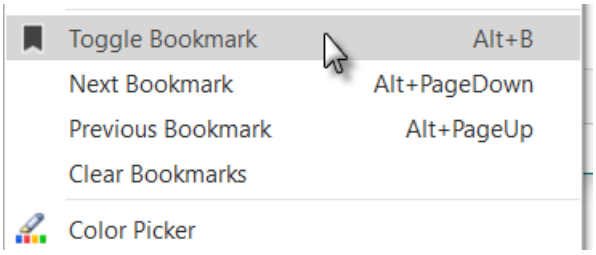
Or right click on the selected code and select Block Comment or Block Uncomment.

4.14 Bookmarks

You can set 'bookmarks' anywhere in the code and jump forward and backwards between these bookmarks.

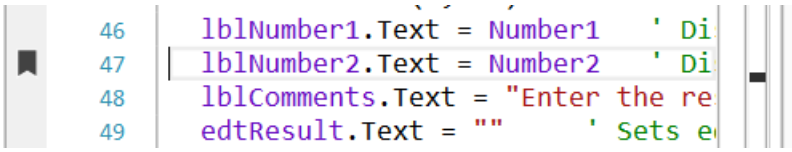
To set or clear a bookmark, select the line and press Alt + B.

Click on in the toolbar, or right click on the line where you want to set a bookmark.



You will get a pop up menu, click on Toggle Bookmark to activate or deactivate a bookmark.

You will see this mark on the left of the line and a small black line in the right slider:

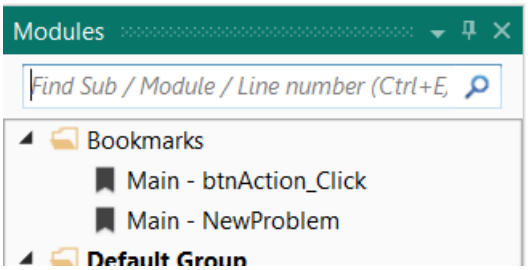


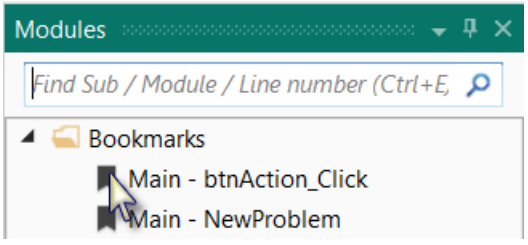
To jump to the next bookmark press Alt + PageDown
or right click and click on Next Bookmark Alt+PageDown

To jump to the previous bookmark press on Alt + PageUp
or right click and click on Previous Bookmark Alt+PageUp

To clear all bookmarks click on in the toolbar or right click and click on Clear Bookmarks

You find the bookmarks also in the Modules Tab.





Click on a bookmark to jump to its line.

4.15 Indentation

A good practice is to use indentation of code parts.
For example for subroutines, loops, structures etc.

You should also have a look at Auto Format.

```
52 Sub btnAction_Click
53 If btnAction.Text = "O K" Then
54 If edtResult.Text = "" Then
55 MsgboxAsync("No result entered","E R R O R")
56 Else
57 CheckResult
58 End If
59 Else
60 NewProblem
61 btnAction.Text = "O K"
62 End If
63 End Sub
```

This code is difficult to read because the structure of the code is not obvious.

```
52 Sub btnAction_Click
53   If btnAction.Text = "O K" Then
54     If edtResult.Text = "" Then
55       MsgboxAsync("No result entered","E R R O R")
56     Else
57       CheckResult
58     End If
59   Else
60     NewProblem
61     btnAction.Text = "O K"
62   End If
63 End Sub
```

This code is much easier to read, the structure of the code is in evidence.

A tabulation value of 2 for the indentation is a good value.

```
52 Sub btnAction_Click
53     If btnAction.Text = "O K" Then
54         If edtResult.Text = "" Then
55             MsgboxAsync("No result entered","E R R O R")
56         Else
57             CheckResult
58         End If
59     Else
60         NewProblem
61         btnAction.Text = "O K"
62     End If
63 End Sub
```

Example with an indentation of 4.

Personally,
I prefer a value of 2.

Whole blocks of code can be indented forth and back at once.

```
20    Dim btnAction, btn0 As Button
21    Dim lblResult As Label
22    Dim lblComments As Label
23    Dim lblMathSign As Label
24    Dim lblNumber1 As Label
25    Dim lblNumber2 As Label
```

Original code.

```
20    Dim btnAction, btn0 As Button
21    Dim lblResult As Label
22    Dim lblComments As Label
23    Dim lblMathSign As Label
24    Dim lblNumber1 As Label
25    Dim lblNumber2 As Label
```

Select the code block.

Click on  .

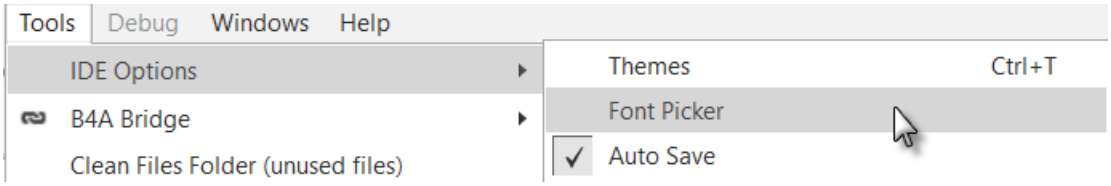
```
20        Dim btnAction, btn0 As Button
21        Dim lblResult As Label
22        Dim lblComments As Label
23        Dim lblMathSign As Label
24        Dim lblNumber1 As Label
25        Dim lblNumber2 As Label
```

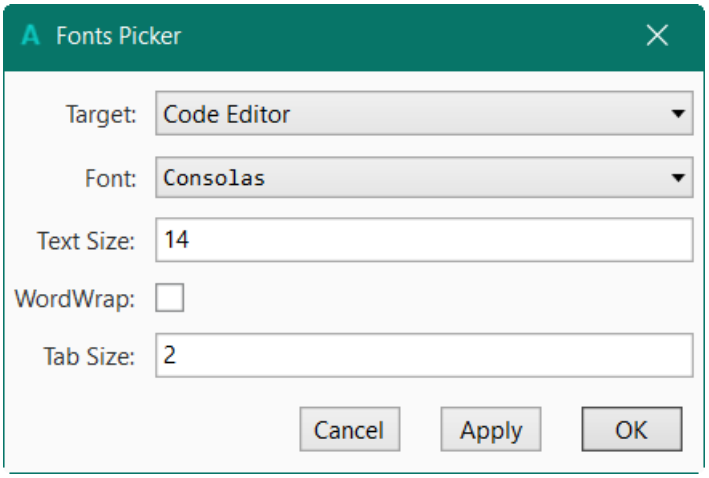
The whole block has moved one tabulation to the right.

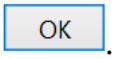
To move a block to the left.

Select the code block and click on  .

The indentation value can be changed in the Tools menu IDE Options / Font Picker.





Enter the value and click on 

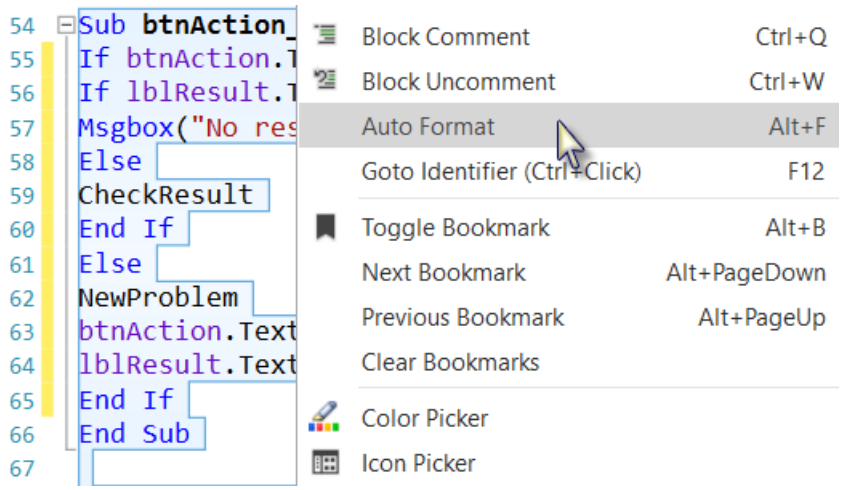
.

4.16 Auto format

You can auto format the code.

This code is not easy to read.

```
54 Sub btnAction_Click
55 If btnAction.Text = "O K" Then
56 If lblResult.Text = "" Then
57 Msgbox("No result entered","E R R O R")
58 Else
59 CheckResult
60 End If
61 Else
62 NewProblem
63 btnAction.Text = "O K"
64 lblResult.Text = "" & Chr(0xE632)
65 End If
66 End Sub
```



Select the code.

Right click in the code area to show this pop-up menu.

And click on Auto Format.

```
54 Sub btnAction_Click
55   If btnAction.Text = "O K" Then
56     If lblResult.Text = "" Then
57       Msgbox("No result entered","E R R O R")
58     Else
59       CheckResult
60     End If
61   Else
62     NewProblem
63     btnAction.Text = "O K"
64     lblResult.Text = "" & Chr(0xE632)
65   End If
66 End Sub
```

And the result.
The Tab size depends on your settings, see previous page.

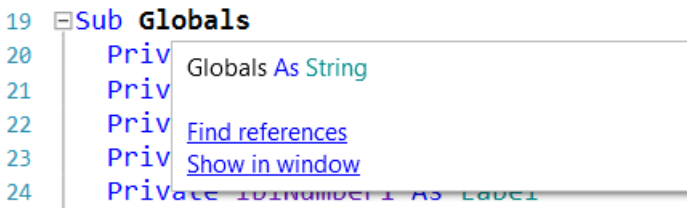
4.17 Documentation tool tips while hovering over code elements

When you hover over code elements the on line helpand other options are displayed.

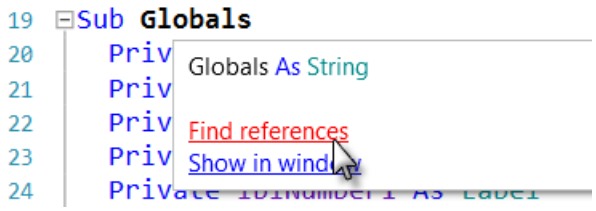
Examples with the MyFirstProgram code:

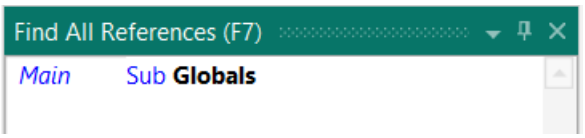
4.17.1 Hovering over a subroutine name

Hovering over **Globals**:



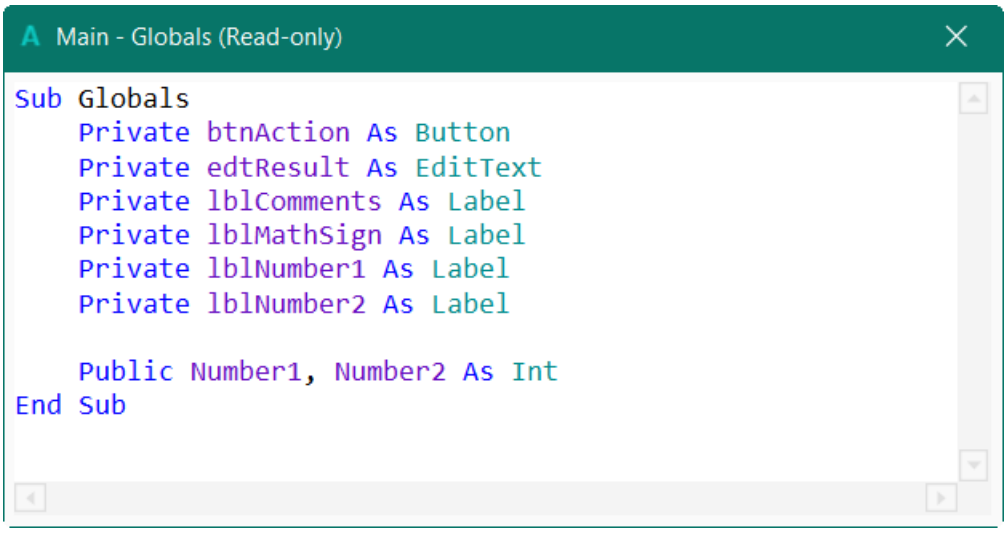
Hovering over Find references or Show in window highlights the link.





Find references

Shows all references in the *Find All References* Tab:

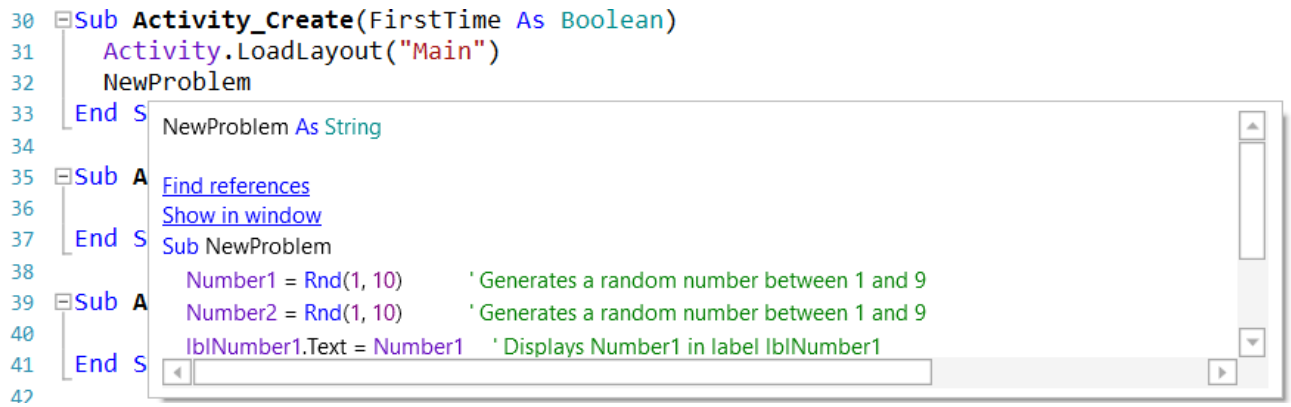


Show in window

Sows the routine in a window:

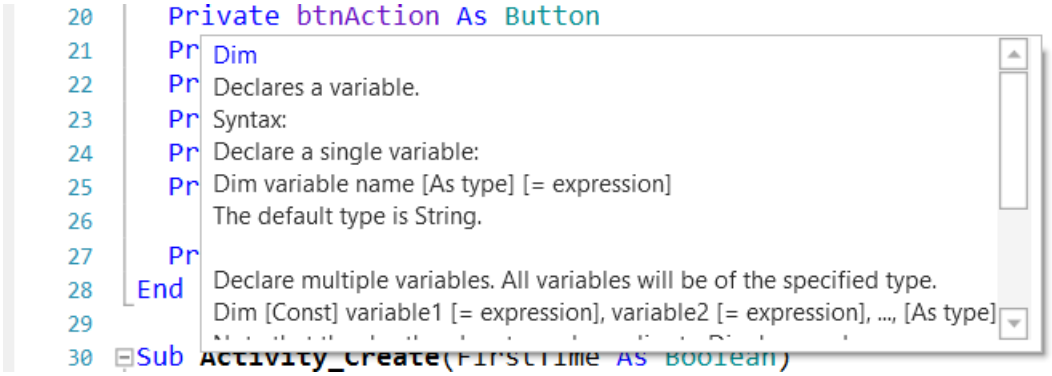
4.17.2 Hovering over a subroutine call

Hovering over **NewProblem**:
Shows Find references and Show in window and shows the content of the routine in the same window.



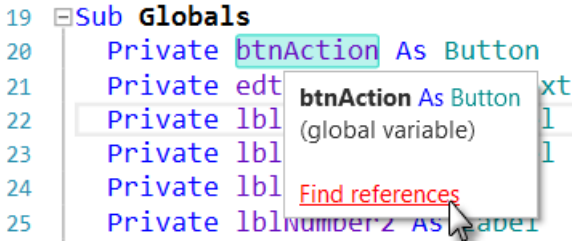
4.17.3 Hovering over a keyword

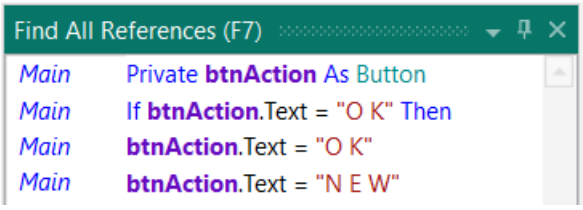
Hovering over **Private** shows the help from the documentation:



4.17.4 Hovering over an object name

Hovering over **btnAction**:



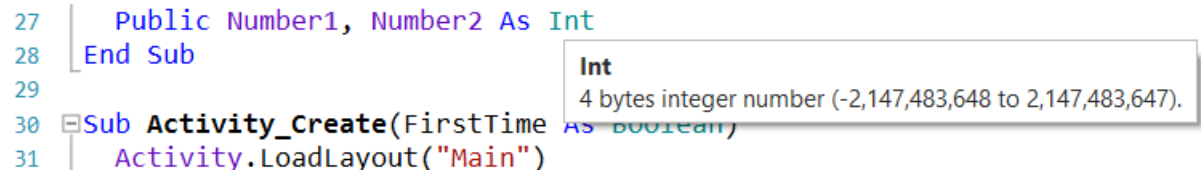


Find references

Shows all lines where btnAction is used.

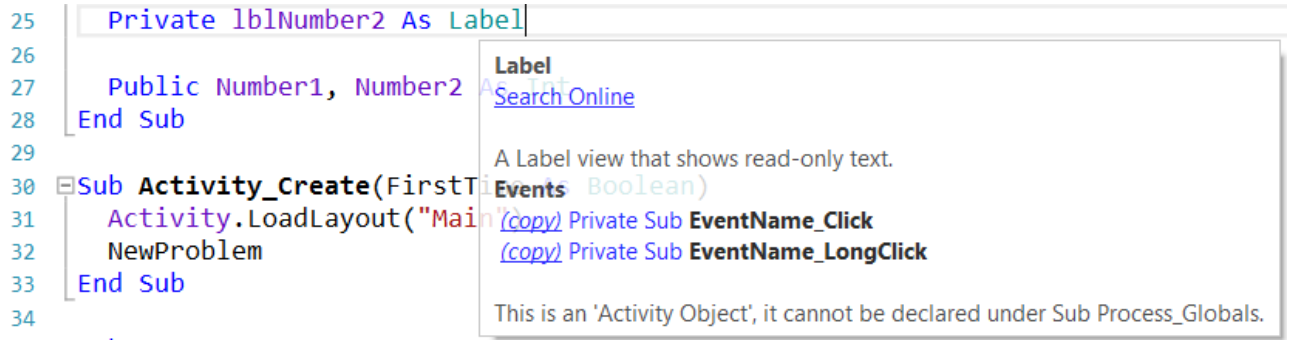
4.17.5 Hovering over a variable type

Hovering over **Int** shows the help:



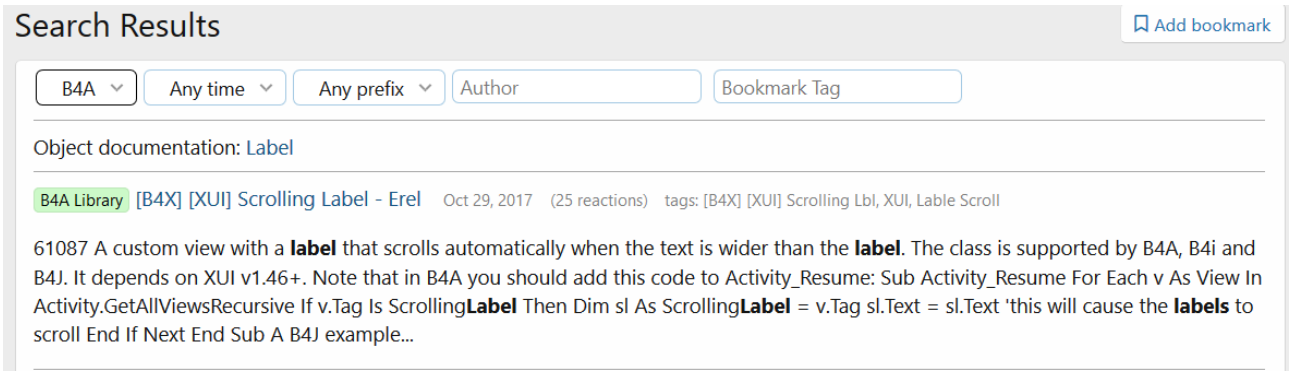
4.17.6 Hovering over a View type

Hovering over **Label** shows the window below:



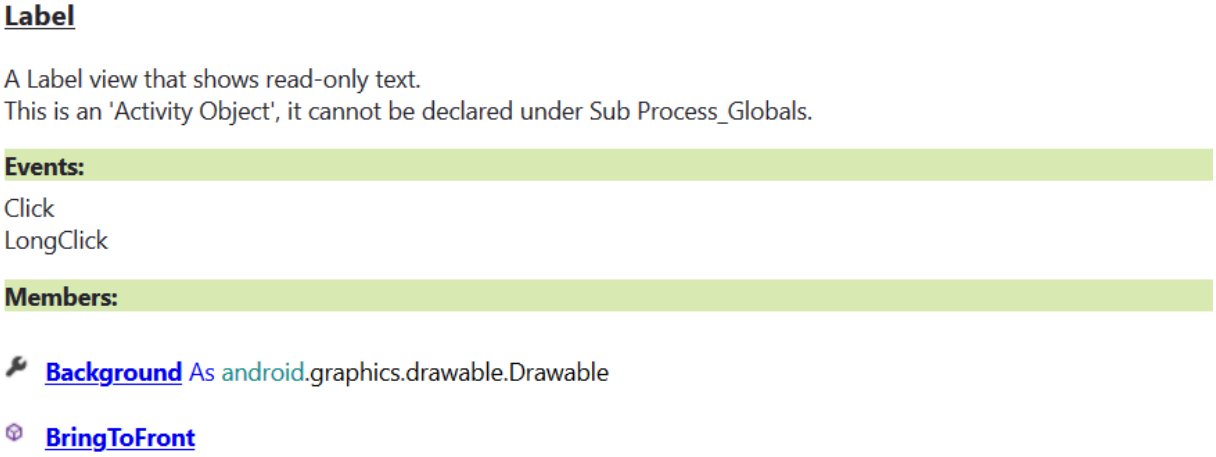
Search Online:

Shows the search result with Label:

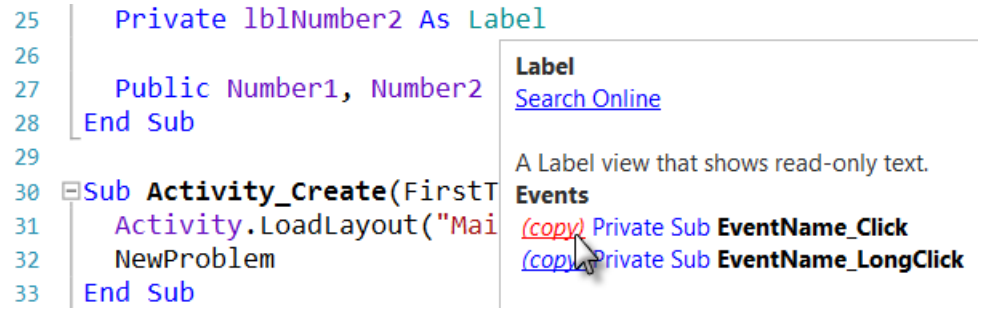


Clicking on 

shows the on line Help:



Clicking on *(copy)* :



Puts `Private Sub EventName_Click` into the clipboard, which you can copy to the code.

The best way to discover all the options is to test this functionality.

4.18 Auto Completion

A very useful tool is the Auto Completion function.

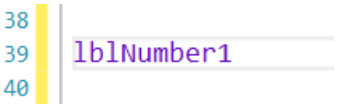
Attention: Make sure that Disable Implicit Auto Completion, in the Tools / IDE Options menu, is not checked !

Example with the MyFirtsProgram code:

Let us write lblN.

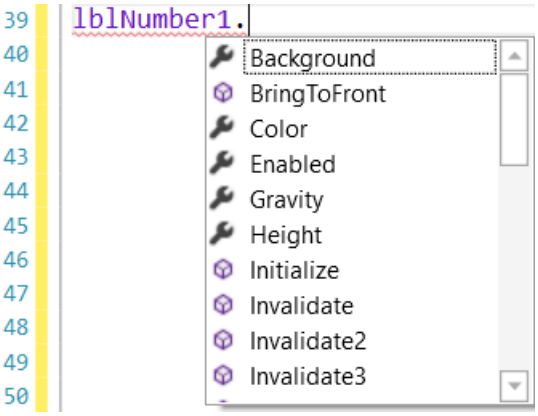
All variables, views and property names beginning with the letters already written are shown in a popup menu with the online help for the highlighted variable, view or property name.

To choose lblNumber1 press Return.

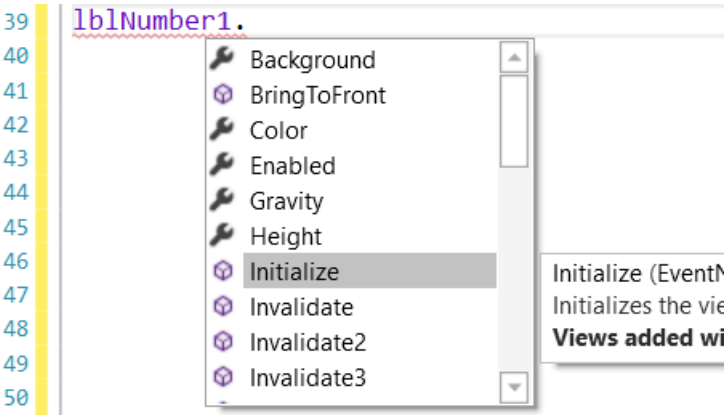


The selected name is completed.

To choose lblNumber2 double click on it or press the down arrow and press Return.



After pressing "." all properties and methods of the view are displayed in a popup menu.



When selecting an item, the internal help is displayed

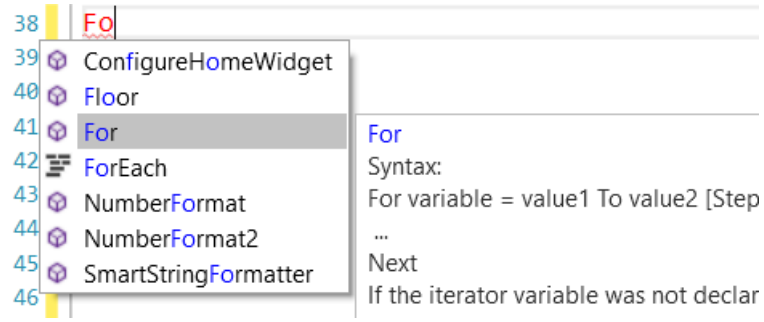
Pressing on the up / down arrows selects the previous or next item with its help.

Pressing a character updates the list and shows the parameter beginning with that character.

Structures are also completed.

Examples:

For / Next



Type Fo

You get For with the help.

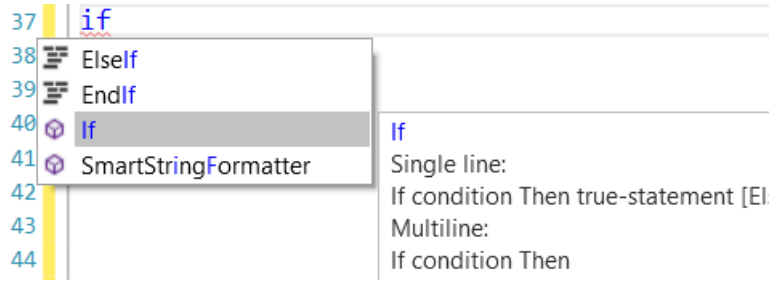
Press Return.

For is completed.

Write the rest of the instruction.
And press Return.

Next is automatically added and the cursor is in the next line idented.

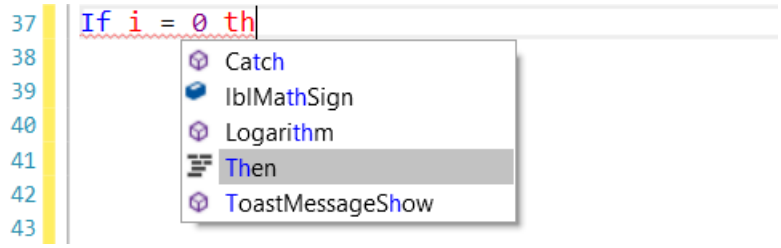
If / Then



Type 'if'.

You get If with the help.

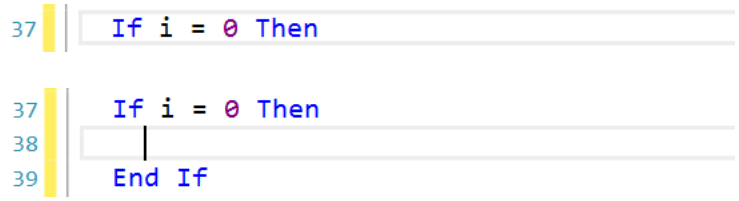
Press Return and continue typing like in the example.



After th you get Then with its help.

Press Return.

And press Return again.

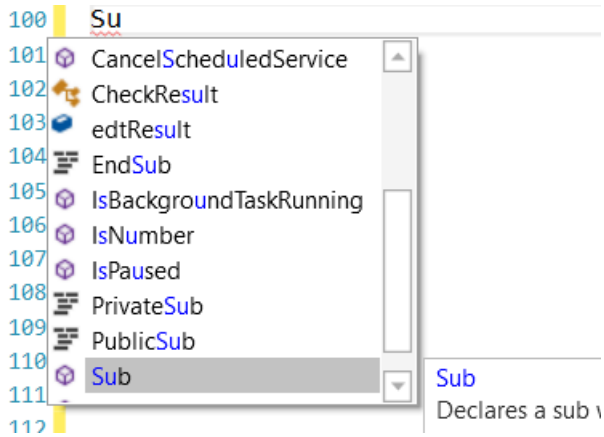


`End If` is automatically added
and the cursor is in the next line idented.

The best way to learn it is to 'play' with it.

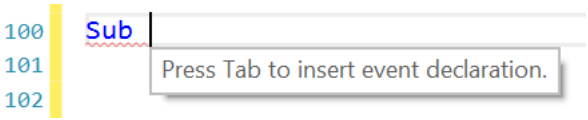
Another very powerful Autocomplete function allows you to create event subroutines.

In the example below we want to create the Click event for the `bntOK` button.
Write '`Su`' and the Auto Completion displays all keywords containing the two characters.

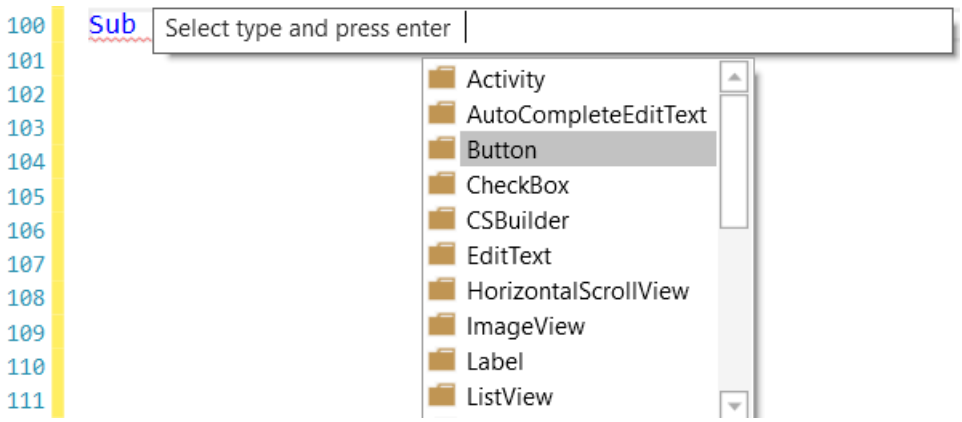


Press Return to select `Sub`.

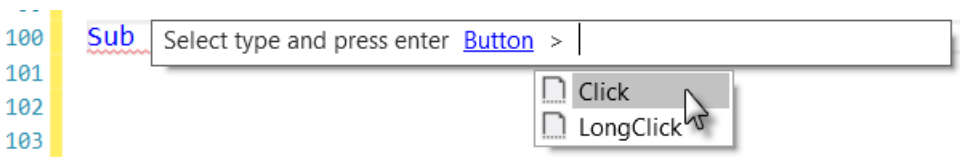
Press blank.



Press Tab and select the view type, select `Button`.



All events for a Button are displayed, select Click.



The subroutine frame is generated.

```
100 Sub EventName_Click
101
102 End Sub
```

Modify 'EventName' to the event name of the button, in our example `btnOK`.

```
100 Sub btnOK_Click
101
102 End Sub
```

Press Return and the routine is ready.

```
100 Sub btnOK_Click
101
102 End Sub
```

4.19 Built in documentation

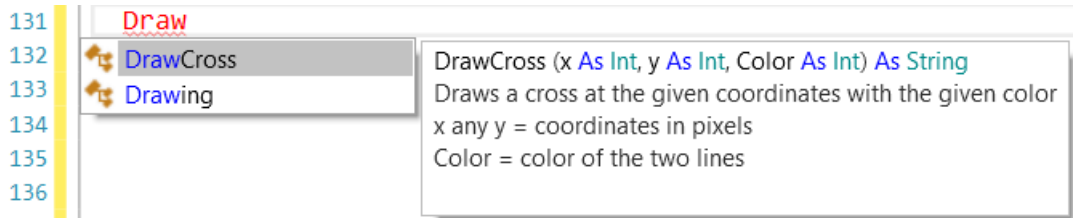
Another useful function is the built-in documentation.

Comments above subs, such as:

```
'Draws a cross at the given coordinates with the given color
'x any y = coordinates in pixels
'Color = color of the two lines
Sub DrawCross(x As Int, y As Int, Color As Int)
   Private d = 3dip As Int

   cvsLayer.DrawLine(x - d, y, x + d, y, Color, 1)
   cvsLayer.DrawLine(x, y - d, x, y + d, Color, 1)
End Sub
```

Will automatically appear in the auto complete pop-up window:

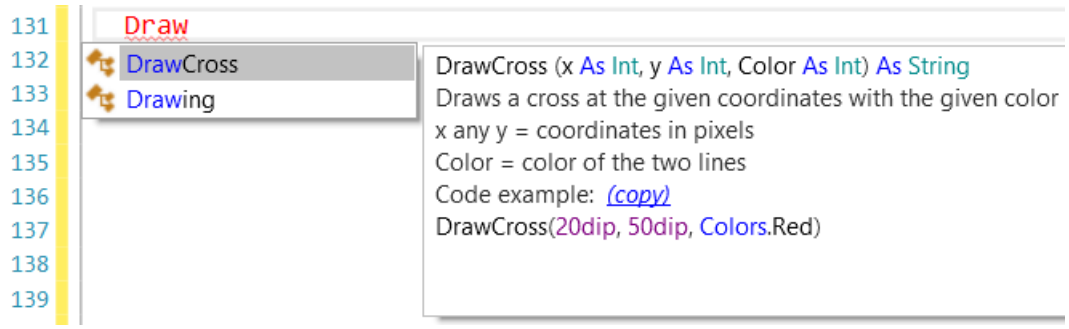


If you want to add a code example you can use <code> </code> tags:

```
'Draws a cross at the given coordinates with the given color
'x any y = coordinates in pixels
'Color = color of the two lines
'Code example: <code>
'DarwCross(20dip, 50dip, Colors.Red)
'</code>
Sub DrawCross(x As Int, y As Int, Color As Int)
   Private d = 3dip As Int

   cvsLayer.DrawLine(x - d, y, x + d, y, Color, 1)
   cvsLayer.DrawLine(x, y - d, x, y + d, Color, 1)
End Sub
```

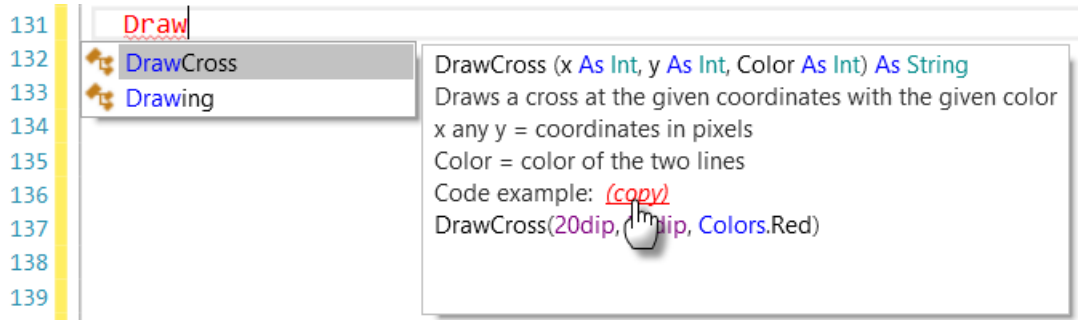
The code will be syntax highlighted:



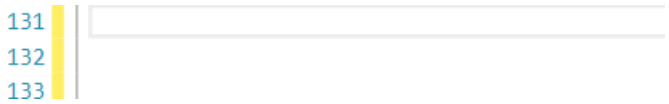
4.19.1 Copy code examples

You can copy the code example in your code.

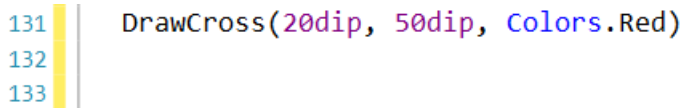
When hovering over (copy) you can copy the code example to the clipboard.



Remove `Draw`



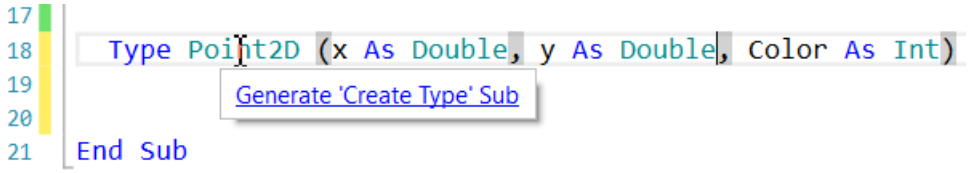
And copy.



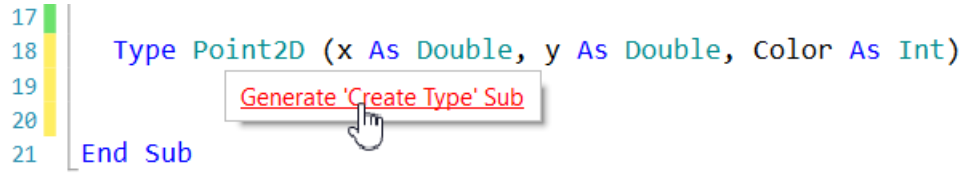
4.19.2 Create Type routine

If you have a Type variable declaration you can create a routine to generate a variable with its content.

When you hover over the Type declaration, you will see the link below.



When you click on the link the CreatePoint3D routine below will be created.




```
Public Sub CreatePoint2D (x As Double, y As Double, Color As Int) As Point2D
   Dim t1 As Point2D
   t1.Initialize
   t1.x = x
   t1.y = y
   t1.Color = Color
   Return t1
End Sub
```

And in the code:

```
Private Point0  As Point2D = CreatePoint2D (10dip, 20dip, xui.Color_Red)
```

4.20 Jump to a subroutine

Sometimes it is useful to jump from a subroutine call to the subroutine definition.
This can easily be done :

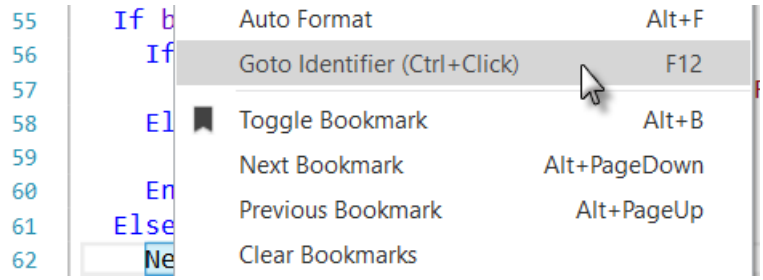
```
61    Else
62       NewProblem
63       btnAction.Text = "O K"
64       lblResult.Text = "" & Chr(0xE632)
65    End If
```

Hover over the text of the subroutine call or select it.
Press Ctrl and Click.

```
43  Sub NewProblem
44     Number1 = Rnd(1, 10)     ' Generate
45     Number2 = Rnd(1, 10)     ' Generate
46     lblNumber1.Text = Number1 ' Displa
47     lblNumber2.Text = Number2 ' Displa
```

And you are there.

Another method.



Select the text of the subroutine call.

Right click on the selected text.

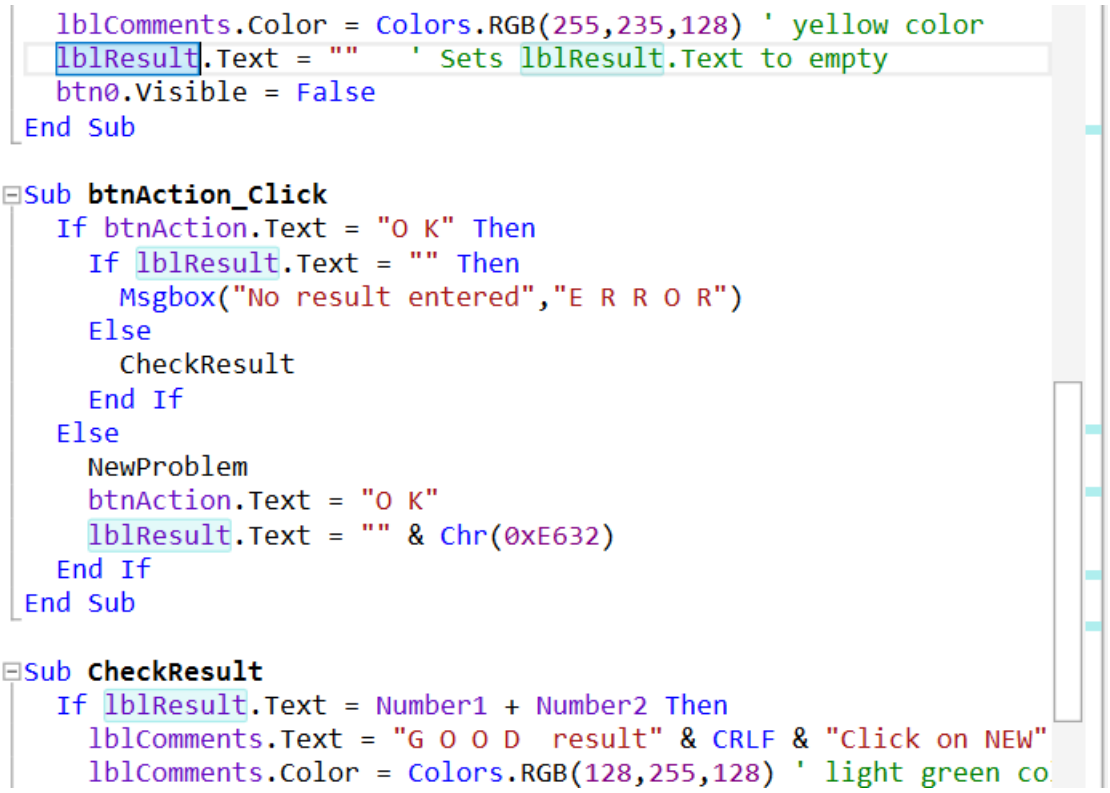
Click on Goto Identifier (Ctrl+Click) .

```
43  Sub NewProblem
44     Number1 = Rnd(1, 10)     ' Generate
45     Number2 = Rnd(1, 10)     ' Generate
46     lblNumber1.Text = Number1 ' Displa
47     lblNumber2.Text = Number2 ' Displa
```

And you are there.

4.21 Highlighting occurrences of words

When you select a single word, it is highlighted in dark blue and all the other occurrences in the code are highlighted in light blue and in the scroll view on the right side.
With the slider you can move up or down the code to go to the other occurrences.



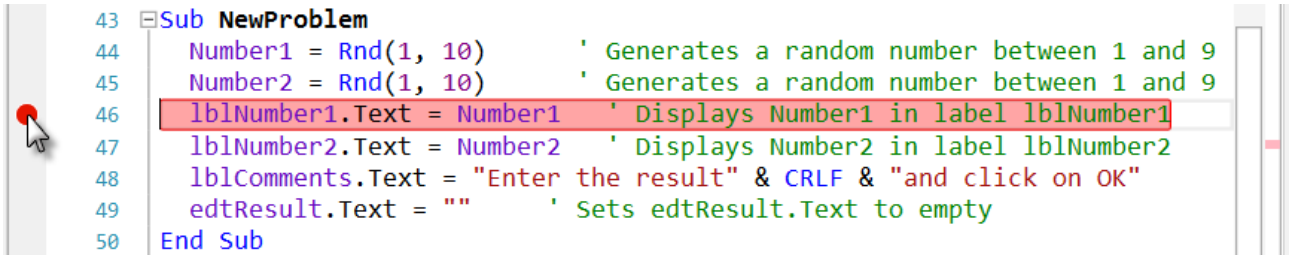
```
  lblComments.Color = Colors.RGB(255,235,128) ' yellow color
  lblResult.Text = ""    ' Sets lblResult.Text to empty
  btn0.Visible = False
End Sub

Sub btnAction_Click
  If btnAction.Text = "O K" Then
    If lblResult.Text = "" Then
      Msgbox("No result entered","E R R O R")
    Else
      CheckResult
    End If
  Else
    NewProblem
    btnAction.Text = "O K"
    lblResult.Text = "" & Chr(0xE632)
  End If
End Sub

Sub CheckResult
  If lblResult.Text = Number1 + Number2 Then
    lblComments.Text = "G O O D  result" & CRLF & "Click on NEW"
    lblComments.Color = Colors.RGB(128,255,128) ' light green co
```

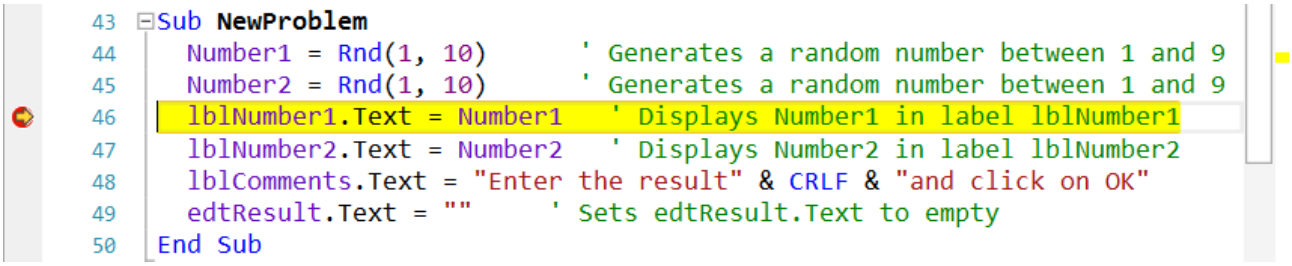
4.22 Breakpoints

Clicking on a line in the left margin adds a breakpoint. When the program is running it stops at the first breakpoint.
Breakpoints are ignored in Globals, Process_Globals and Activity_Pause.

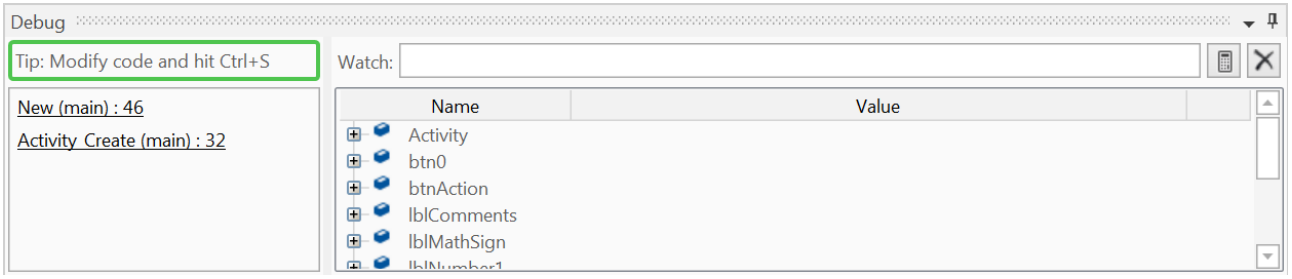


The position of the breakpoints is shown on the right side with a light red bar:

Run the program, the program stops at the breakpoint and the IDE looks like below. The line where the program stops is highlighted in yellow.



At the bottom of the IDE you find other information.



The Debugger is connected. In the left part of the Debugger window we find:

- Tip: Modify code and hit Ctrl+S — A button to update the program after a code modification.
- NewProblem (main) : 46 — The name of the routine where the Debugger stopped the program. New in the module Main in line 46.
- Activity_Create (main) : N/A — Caller of the “New” routine: Activity_Create in the module Main routine in line 32.

Clicking on these links moves the cursor to the given line.

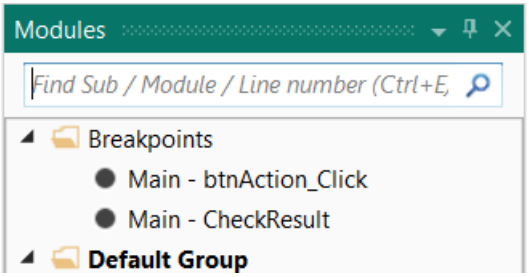
In the right part of the Debugger window we find the list of all Views and Variables with their values.

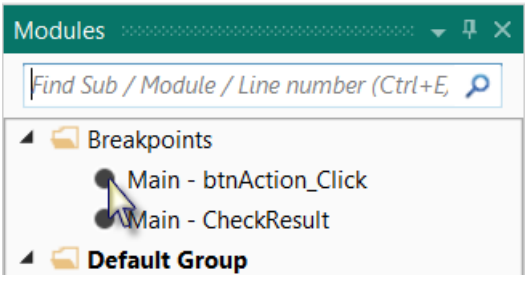
In the Toolbar, at the top of the IDE the navigation buttons are enabled.

	Step In	F8	Executes the next statement.
	Step Over	F9	Steps over the subroutine call, but executes its code.
	Step Out	F10	Executes the rest of the routine and leaves it.
	Stop		Stops the program.
	Restart	F11	Restarts the program.

More details in chapter Debugging.

The Breakpoints are listed in the Modules Tab.



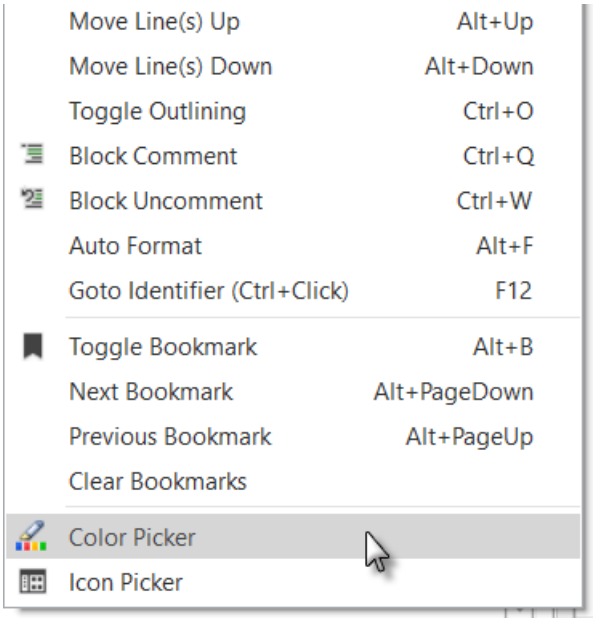


Click on a Breakpoint to jump to its line.

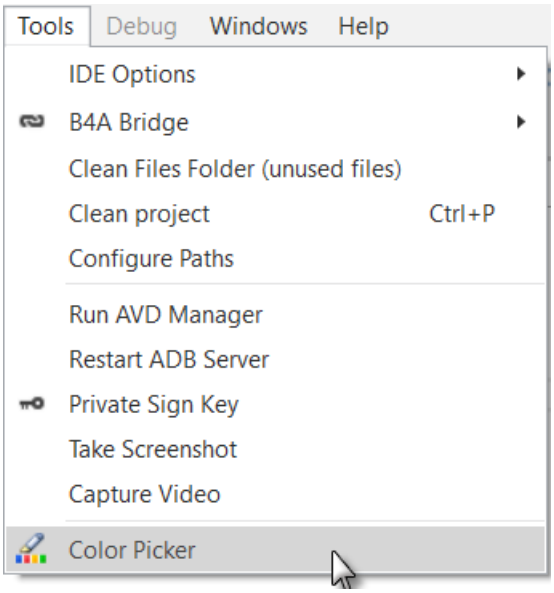
4.23 Color Picker

Color Picker

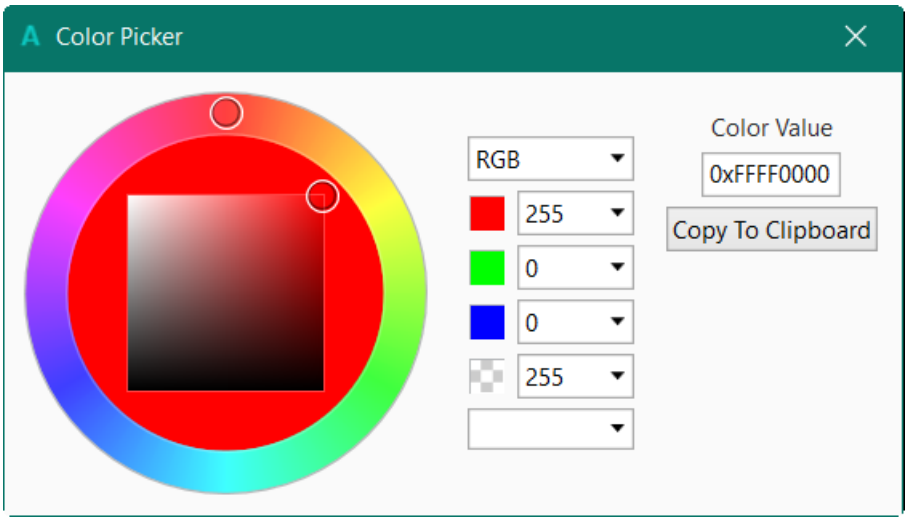
In the code, right click to show the popup menu below.



Or, in the menu Tools.



Click on Color Picker to show the Color Picker.

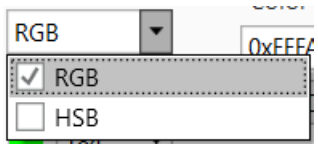


You can:
- Move the cursor in the outer circle and in the square to select the color.

- Enter directly
  A R G B values or
  A H S B values.

- Copy the value to the Clipboard.

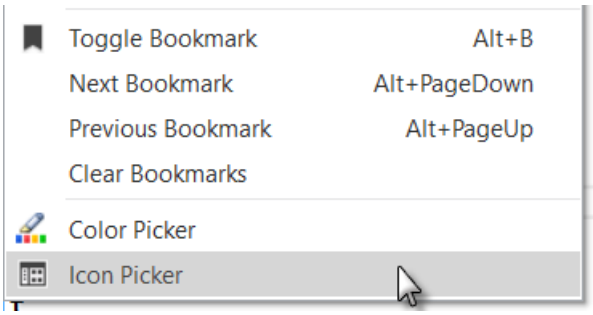
You can then paste the value into the code.



You can select either RGB or HSB values.

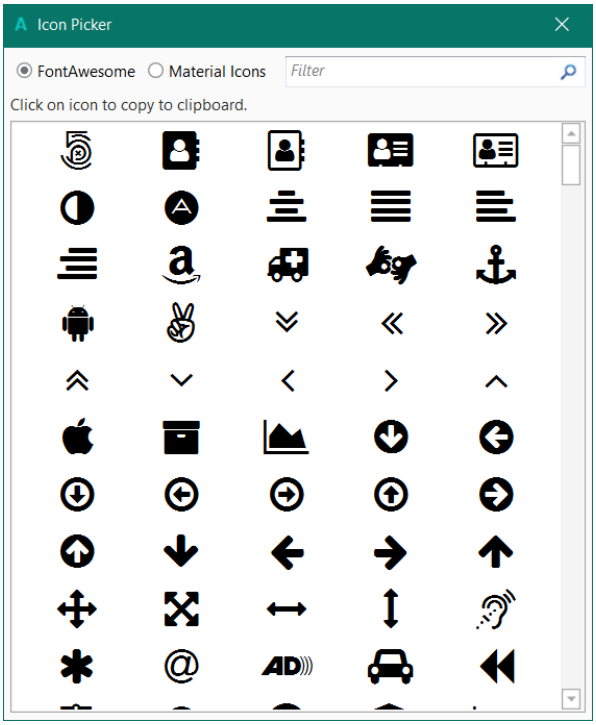
RGB Red, Green, Blue
HSB Hue, Saturation, Lightness

4.24 Icon Picker

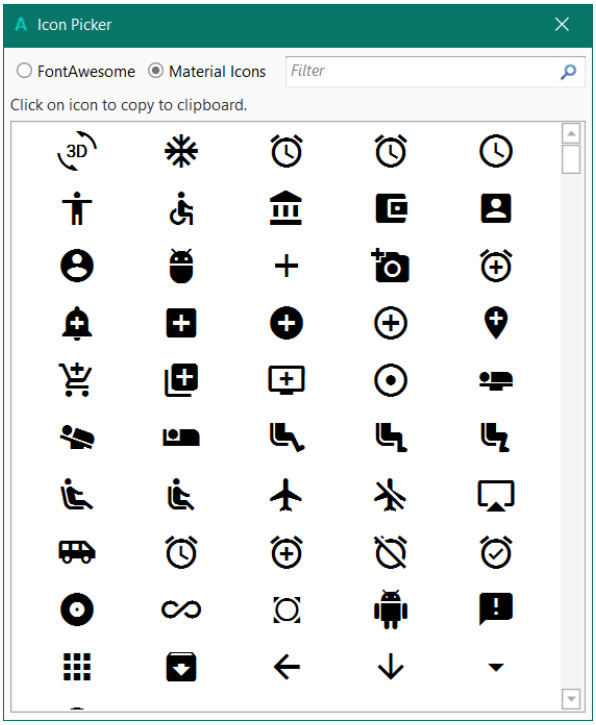


Right click in the IDE code area to show the pop-up menu and click on Icon Picker.

You can schoose between Font Awesome and Material icons.



Font Awesome icons.



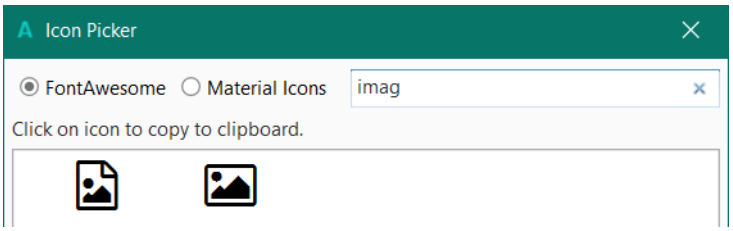
Material icons.

Click on an icon to copy it to the clipboard.
Then you can paste it into the code like below.
The icon is given with its character number, `Chr(0xE632)`.

```
lblResult.Text = Chr(0xE632)
```

We need also to change the font type to:

```
lblResult.Typeface = Typeface.FONTAWESOME
or  lblResult.Typeface = Typeface.MATERIALICONS
```



You can filter the icons.

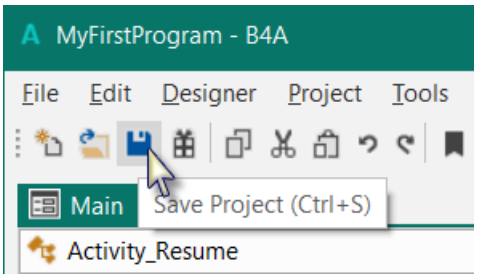
4.25 Colors in the left side

Sometimes, you will see yellow or green vertical lines in the left side od the IDE.

As soon as you modify a line it will be marked with a yellow vertical line on the right of the line number meaning that this line was modified.

```
67 Sub CheckResult
68    If edtResult.Text = N
69      lblComments.Text =
70      btnAction.Text = "N
71    Else
72      lblComments.Text =
73    End If
74 End Sub
```

If we click on  to save the project the yellow lines become green showing a modified code but already saved. You can also press Ctrl + S to save the project.




```
67 Sub CheckResult
68    If edtResult.Text = Nu
69      lblComments.Text = '
70      btnAction.Text = "N
71    Else
72      lblComments.Text = '
73    End If
74 End Sub
```

If we leave the IDE and load the project again the green lines disappear.

4.26 URLs in comments and strings are ctrl-clickable

URLs in comments and strings are ctrl-clickable.

In a comment:

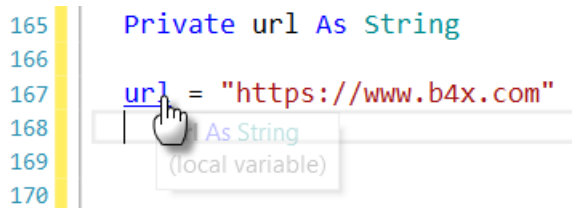
```
162  'https://www.b4x.com
```

If the cursor is on the line and you press Ctrl the url is highlighted in blue and if you click on it the url it is executed. Hovering over the line with Ctrl pressed does also highlight the url.

In a String:

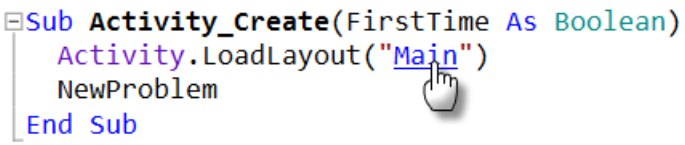
```
165    Private url As String
166
167    url = "https://www.b4x.com"
```

The cursor must be over the String variable and not over text.



4.27 Ctrl + Click on layout file name opens the Desiger

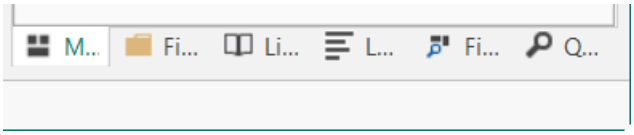
When you hover over a layout filename in the code, with CTRL key pressed like this:



The layout filename color is changed to blue, the mouse cursor becomes a hand and if you click on it, the Designer is opened with that layout file.

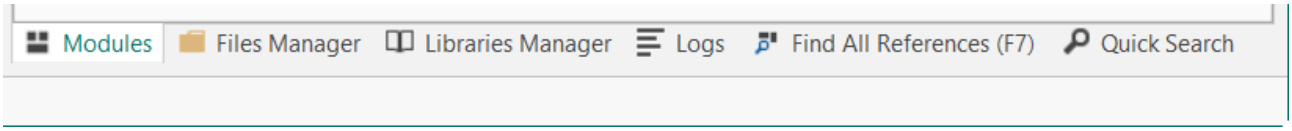
5 Tabs

There are 6 tabs at the bottom right corner of the IDE that displays different windows.



The short version.

The wide version.

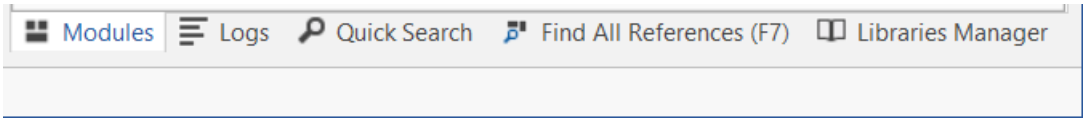


The 6 Tabs are:

- Modules
- Files Manager
- Libraries Manager
- Logs
- Find All References
- Quick Search

Each Tab has its own window.
By default they are displayed in the Tab area on the right side of the IDE, only one at the same time.
These windows can be closed, hidden or floating, see next chapter.

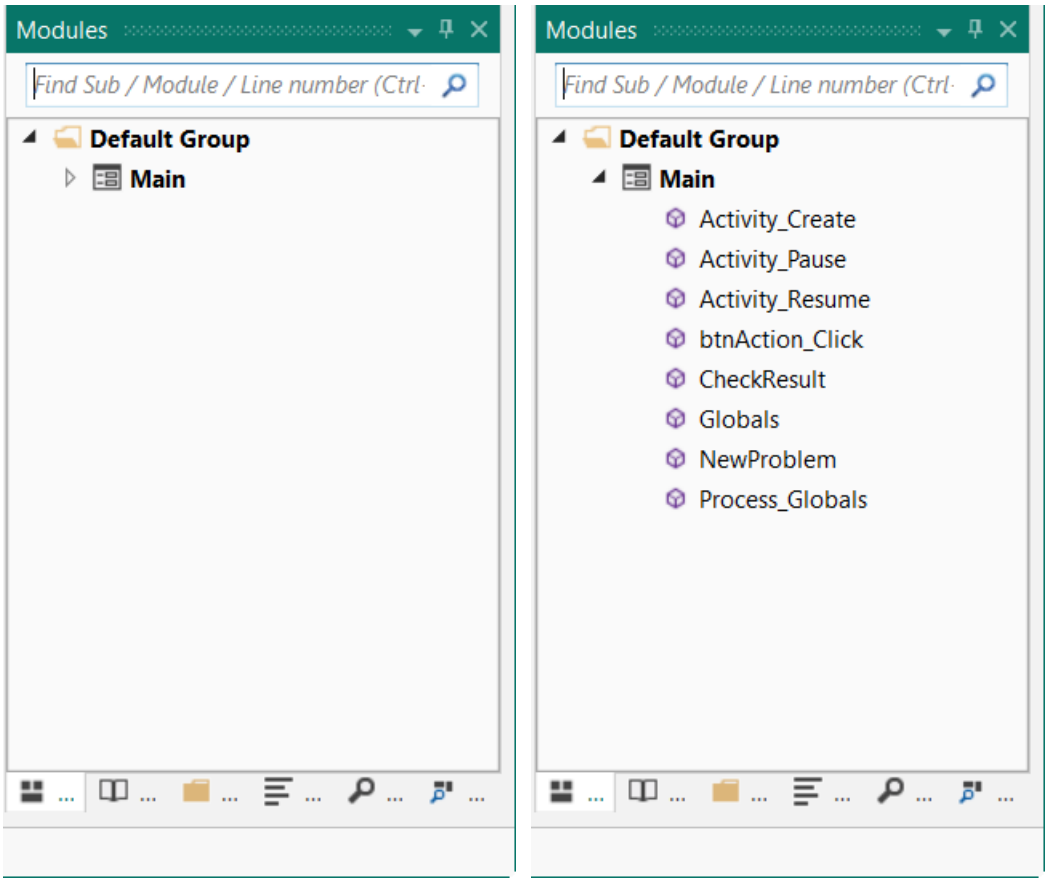
B4R



Only 5 Tabs, no Files Manager Tab

5.1 Floating Tab windows

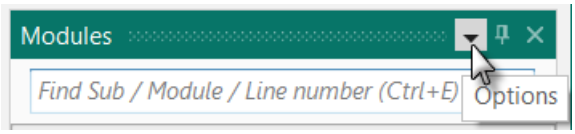
When you start the default IDE all Tab windows are docked in the Tab area.



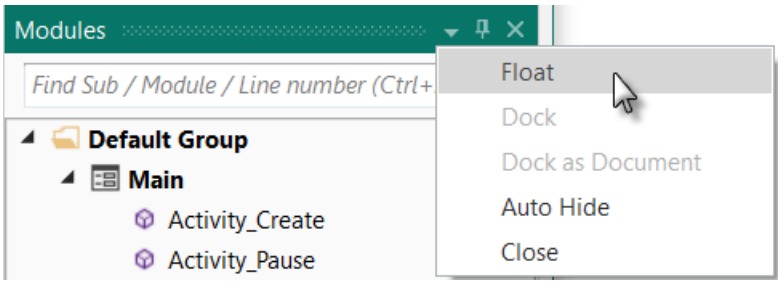
You can set each Tab window as a separate floating window.

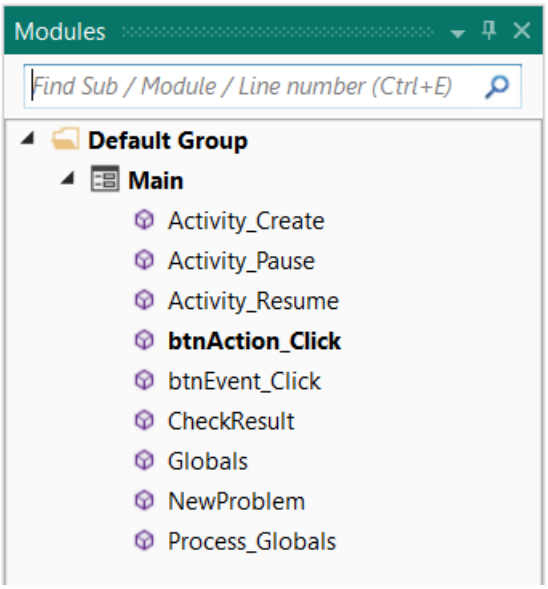
5.2 Float

To set the Modules Tab window to floating click in the title on .



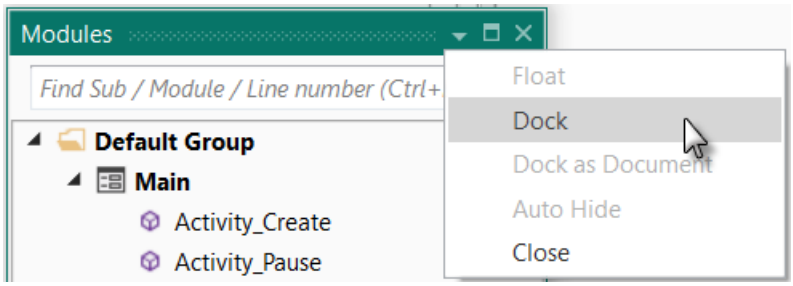
Click on Float.





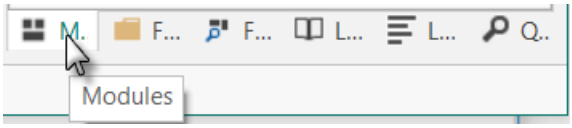
The Modules Tab Window is now floating, you can place it where you want on the screen even on a second monitor.

To dock it back to the Tab area click on Dock.

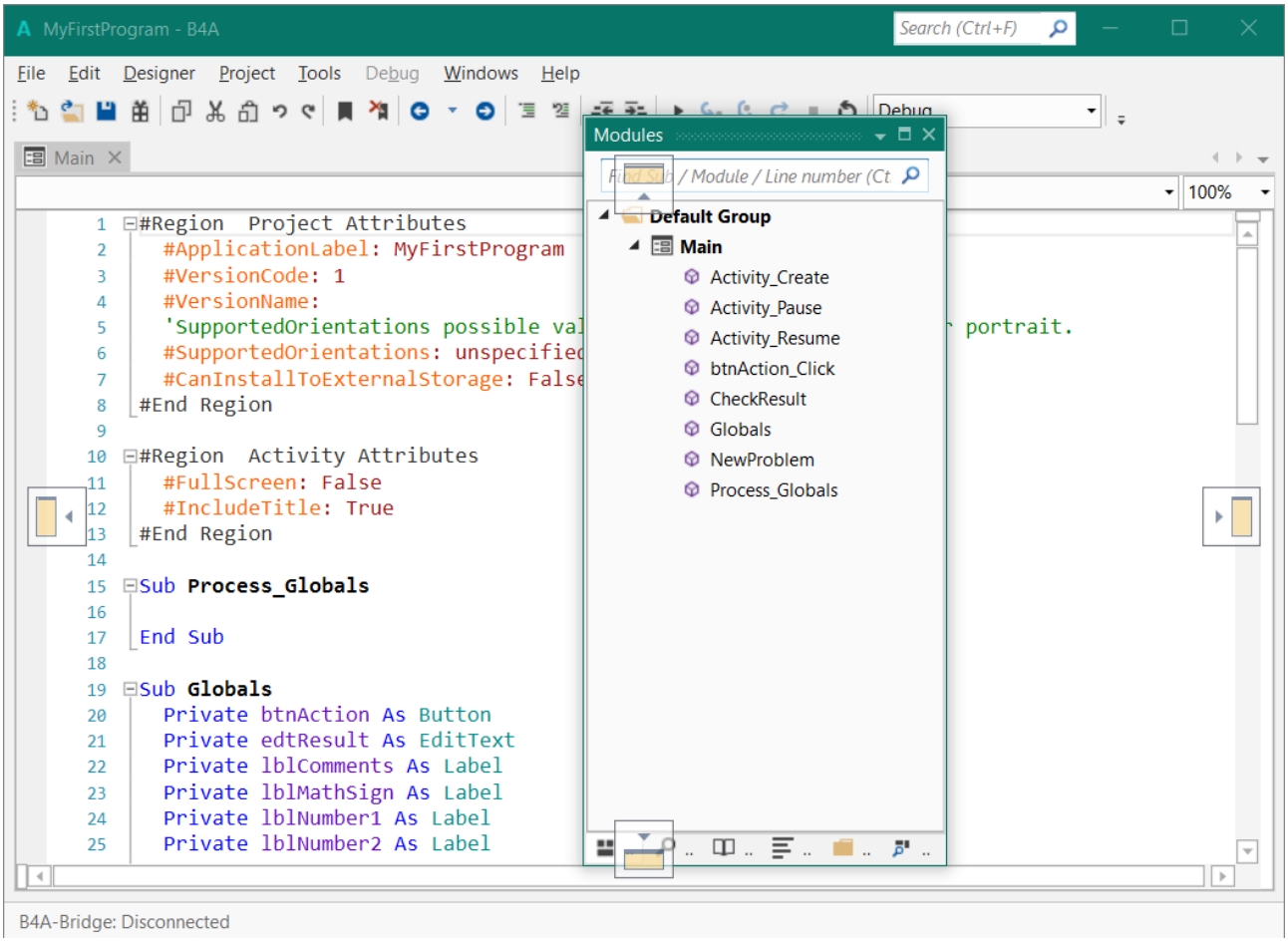


To show the Tabs again click either on Dock in the Options or on Reset in the IDE Window menu.

You can also click on a Tab and while maintaining the mouse down, move the Tab.



This will show you all the possible ‘docking’ areas.



Docking areas:

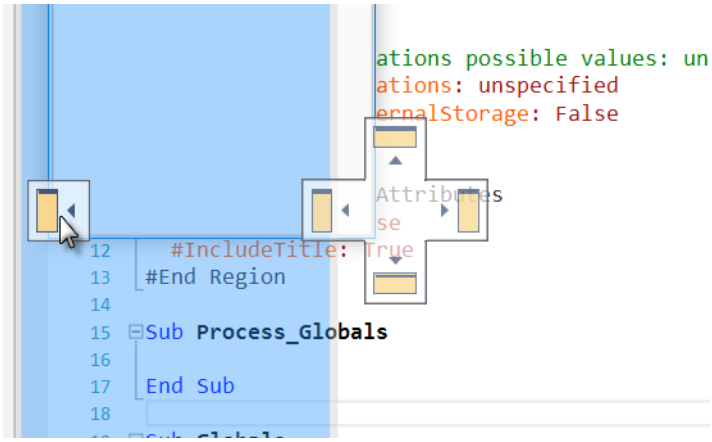

Top


Left

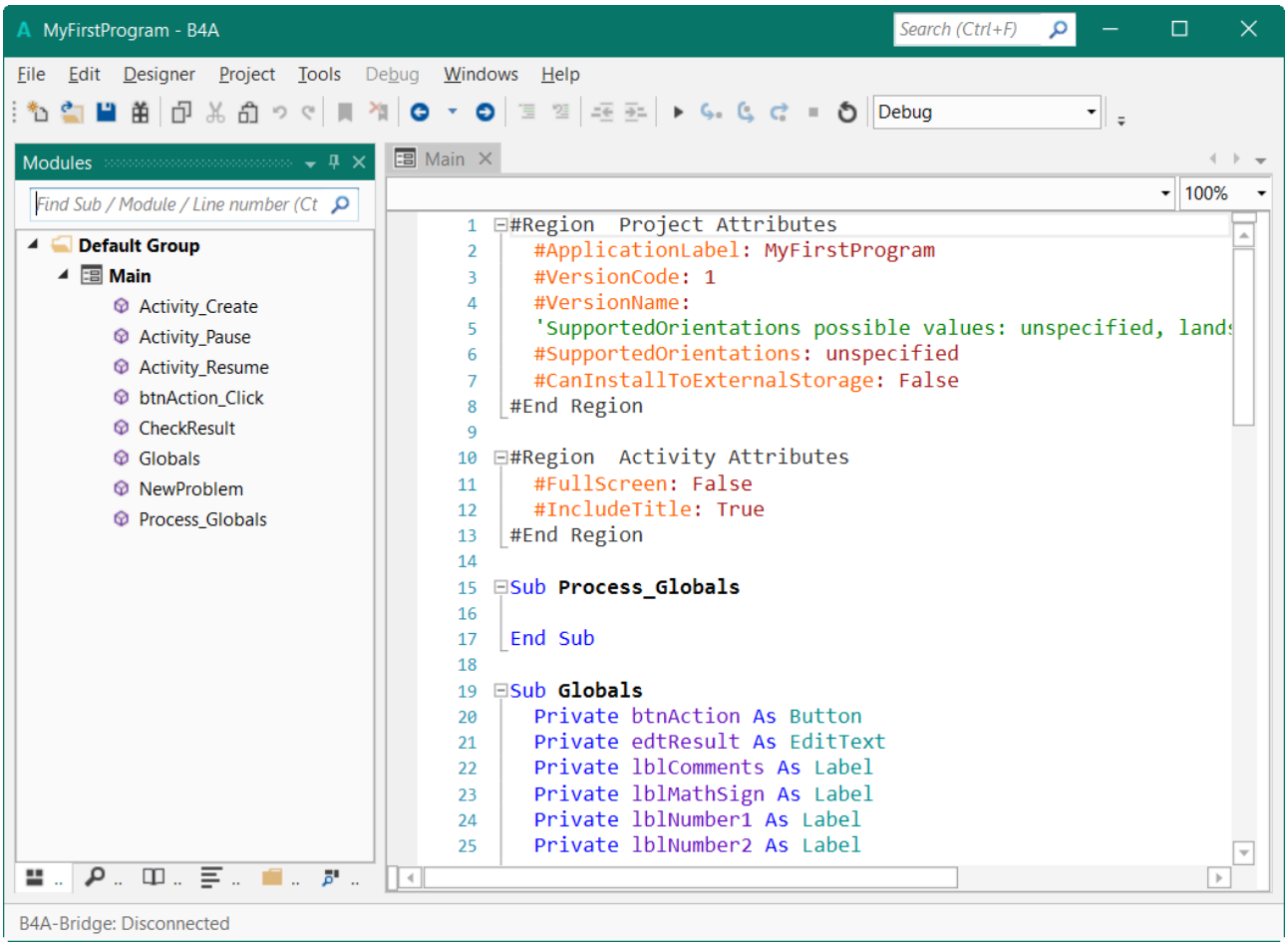

Right

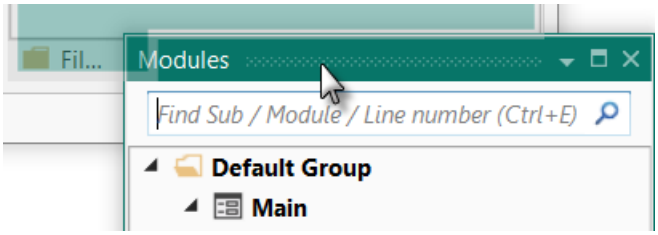

Bottom

If you mouve the mouse onto one of the docking area symbol, the Tab window will be either on top, on the left, the right or on the bottom.



And the result.

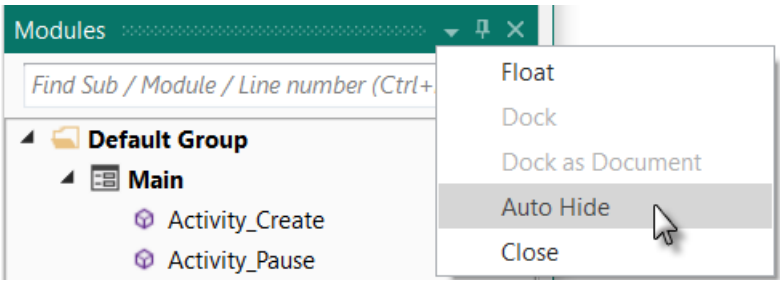


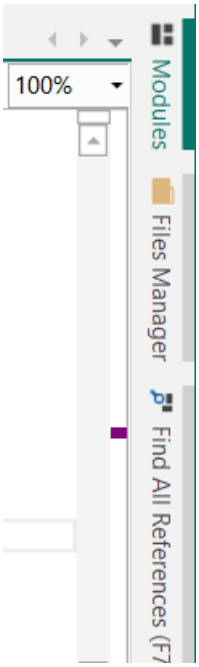


To bring it back to the Tabs, click on the window title and move it back to the Tabs.

5.3 Auto Hide

Click on in the title or click on Auto Hide in the Options.

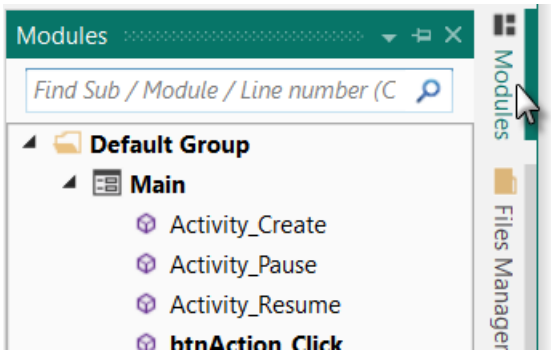




The Tabs move from the bottom of the screen vertically on the right side of the screen and the Tab window is hidden.

Hovering over a Tab highlights it in green.

Click on a Tab to show it.

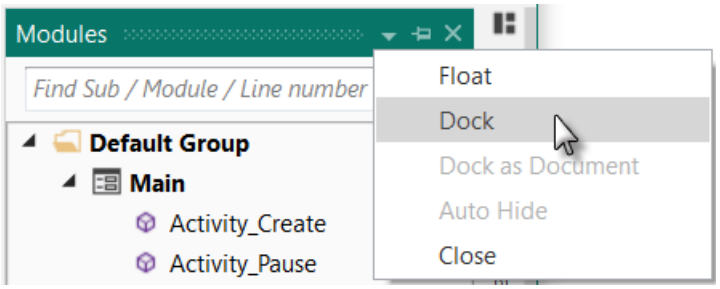


The selected Tab is displayed.

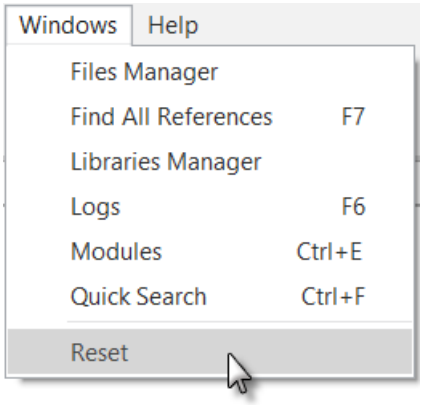
As soon as you click somewhere else in the IDE the Tab is hidden again.

To move the Tabs back to the lower right corner:

Click on Dock in the Options.



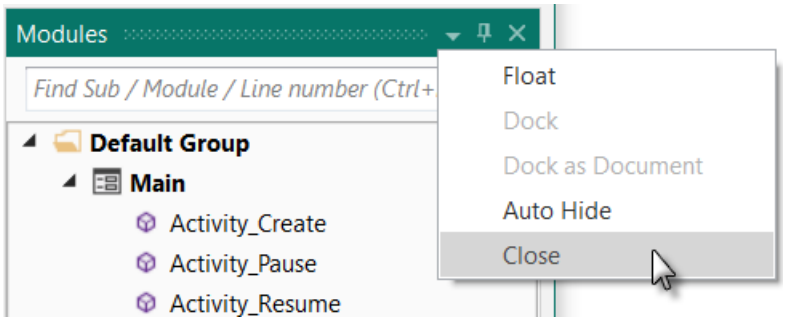
Or click on Reset in the IDE Windows menu.

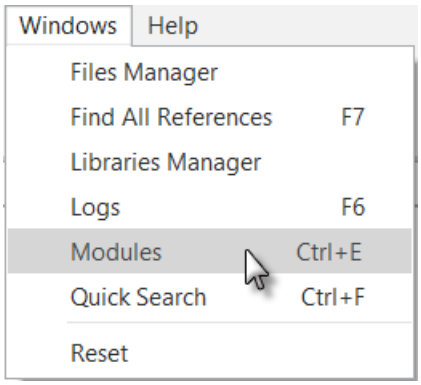


5.4 Close

You can close a window, hide it.

Click on  in the title or on Close in the Options.

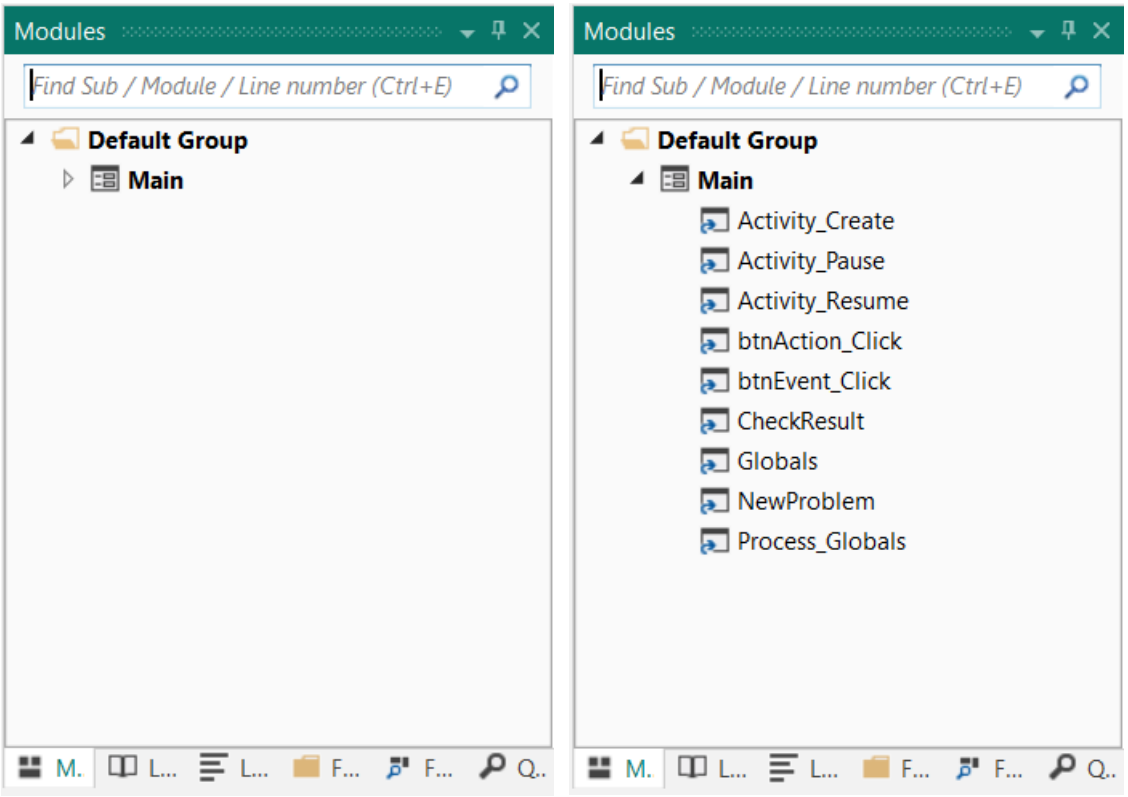




To show it again, in the Windows menu click on the module name you want to show, Modules in our example.

5.5 Modules and subroutine lists Modules

All the modules of the project and all subroutines of the selected module are listed in the Modules window. The picture below has been reduced in height.



On top you see Default Group and Main. Click on Main to show the routines contained in Main.

Find Sub / Module / Line number (Ctrl + E)

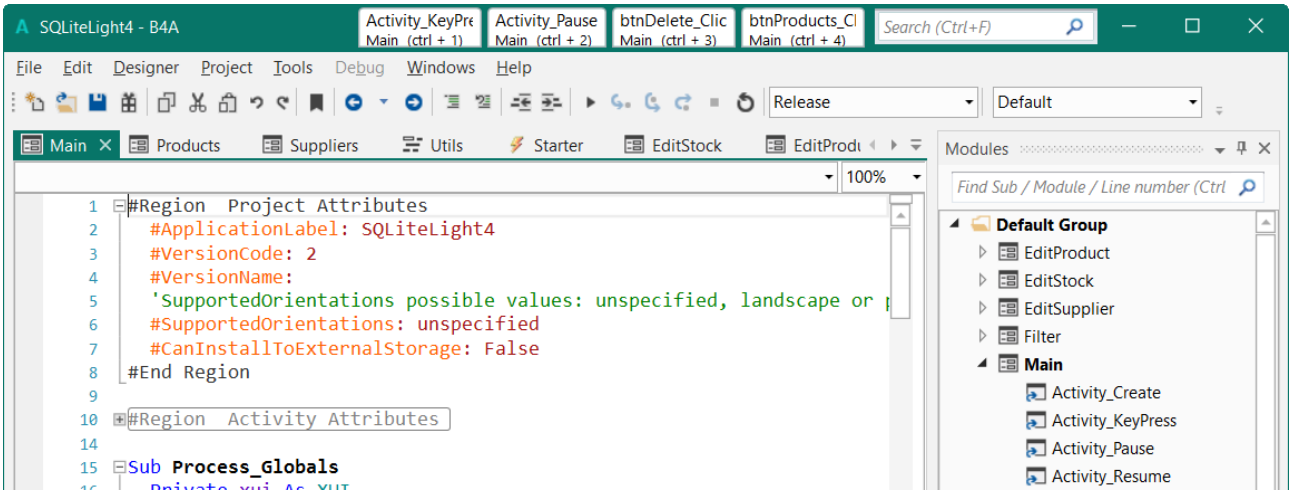
Module list on top.
Clicking on a module shows its code in the code area.

Find Sub Tool (Ctrl + E) see below

Subroutine list of the selected module.
Clicking on a subroutine shows its code in the middle of the code area.

To show a hidden module, click on the module name in the module list.

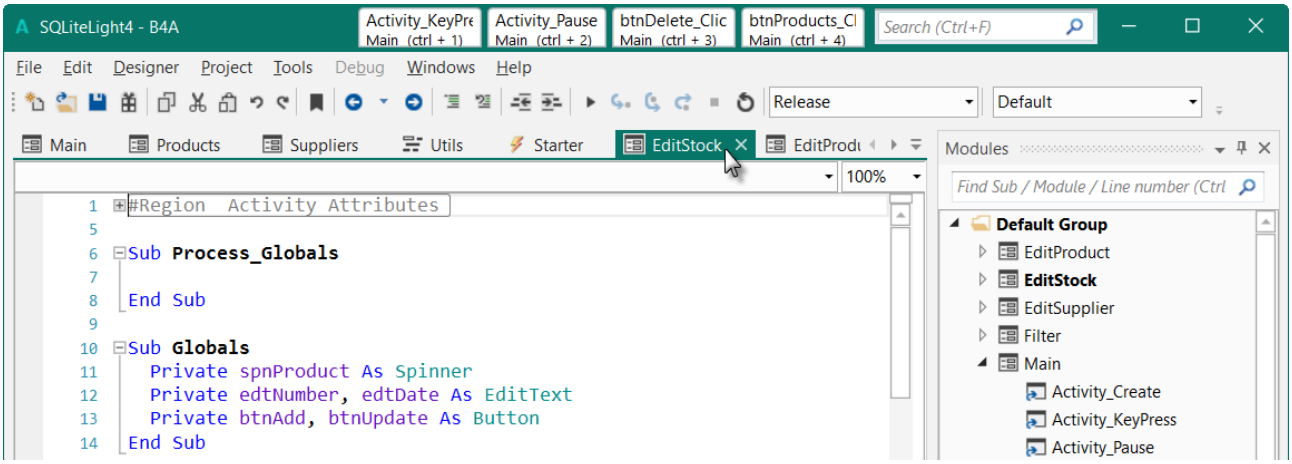
Example with several Modules:



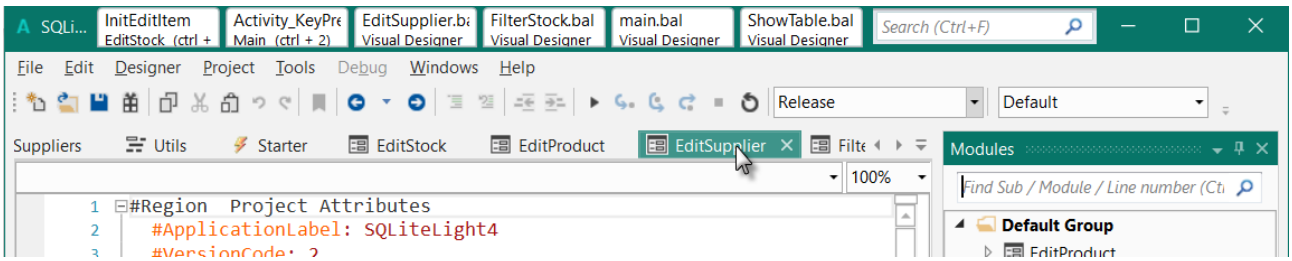
In the Modules Tab you find all the modules listed.

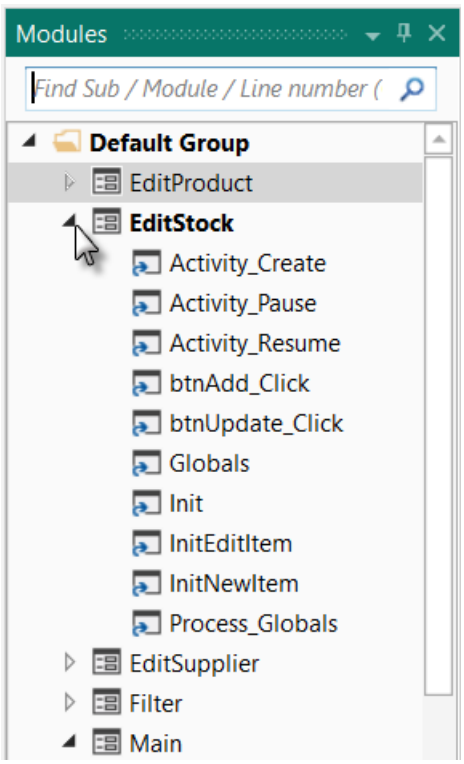
The active module is highlighted.

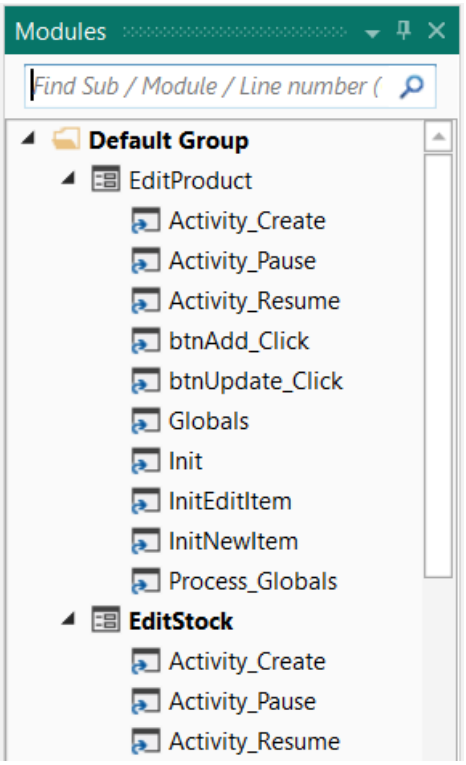
Clicking on a module shows it in the code area.



When the cursor is on top of a Tab like in the picture above and you move the mouse wheel, the Tabs scroll horizontally.



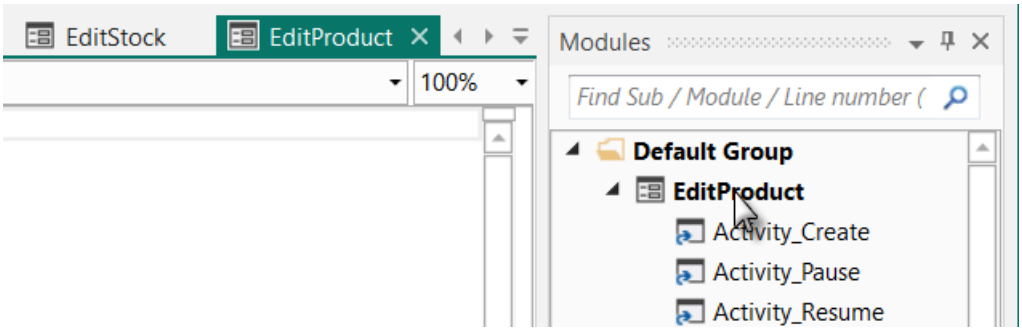




Click on ▷, to show the routines in a given module.

Once you 'opened' a module, it remains open. You can scroll through the list.

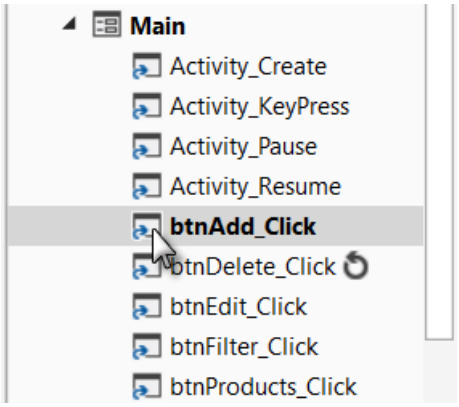
A double click on a module, sets this module as the active one and shows its content.

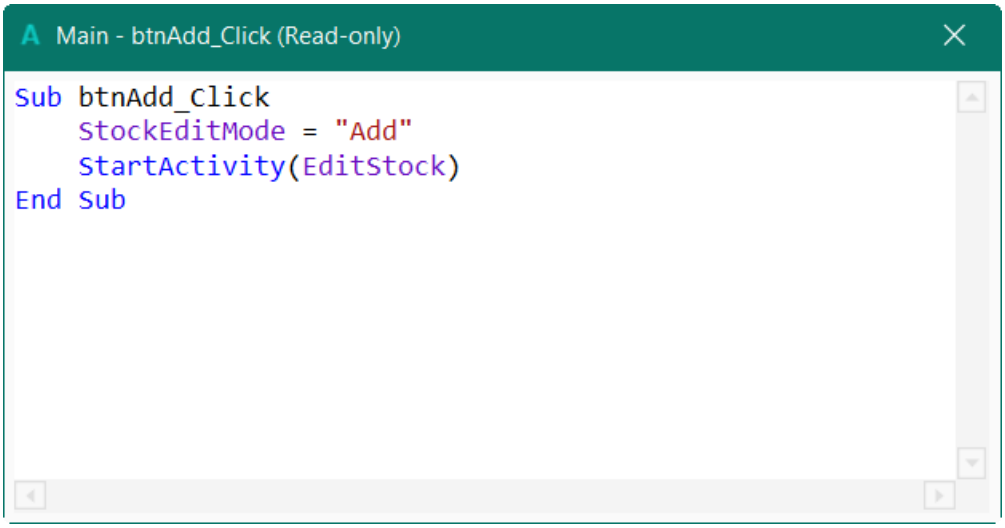


Clicking on a routine, even in a none active module, sets the module as the active one and shows the routine in the editor.

Show a routine in a separate window.

Click on the small icon  near the routine name to display it.

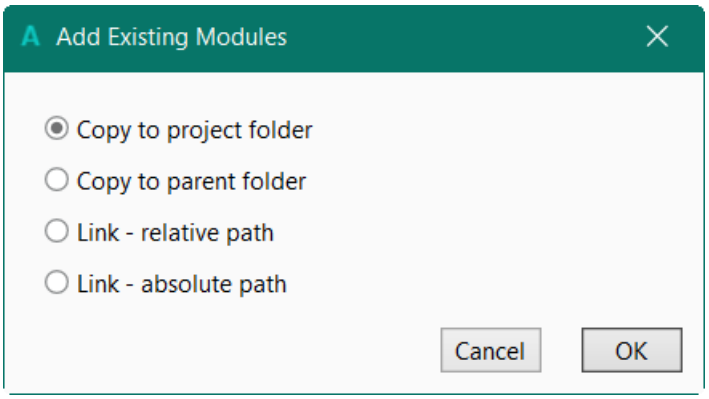




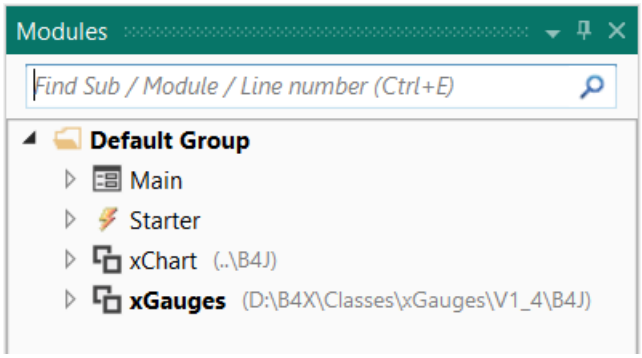
You can modify the code in this window, it will also be modified in the main window.

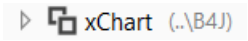
5.5.1 Modules with relative or absolute links

When you add an existing module, you will be asked what kind of link you want.

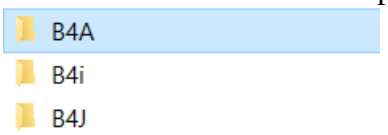


Three option are available.
If you choose one of the links, you will see it in the Modules Tab.





Is a relative link. In the example, another folder at the same level as the project.



The project is in the B4A folder.

The xChart Module is in the B4J folder.

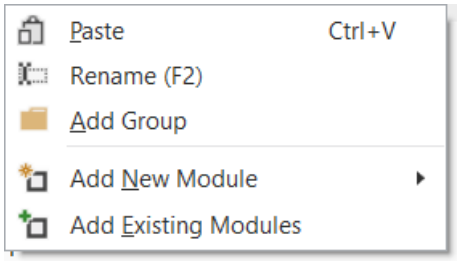
Is an absolute link, with the full name.

5.5.2 Context menus

What you can do:

- Right click on a Group 

:

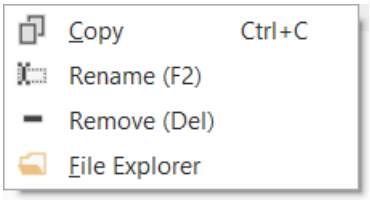


Paste a selected Module.
Rename the Group.
Add a Group.

Add a new Module (same as in the Project menu).
Add an existing Module (same as in the Project menu).

- Right click on a Module 

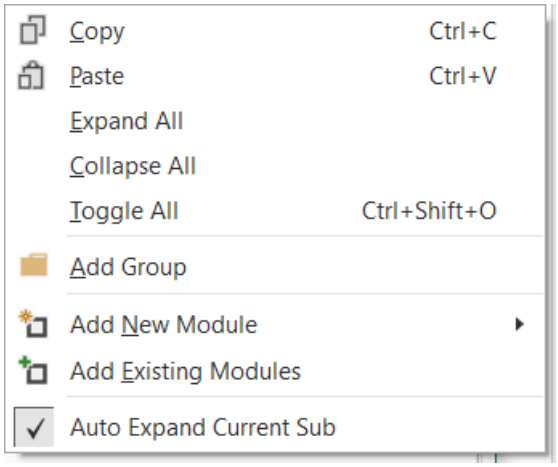
:



Copy the Module.
Rename the Module.
Remove the Module.
Open the File Explorer to select any Module.

- Right click on a Subroutine 

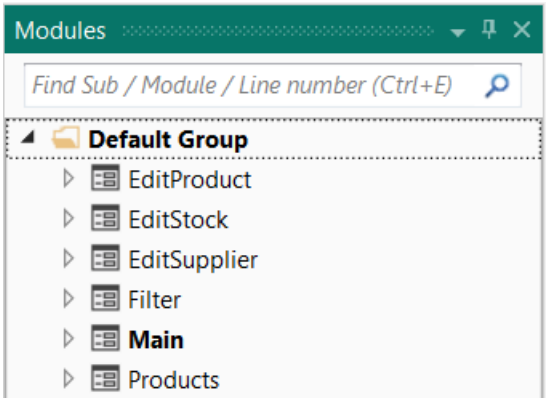
, or somewhere else:



Copy the Subroutine
Paste a Subroutine from the clipboard.
Expand all the Tab content.
Collapse all the Tab content.
Toggle all the Tab content.

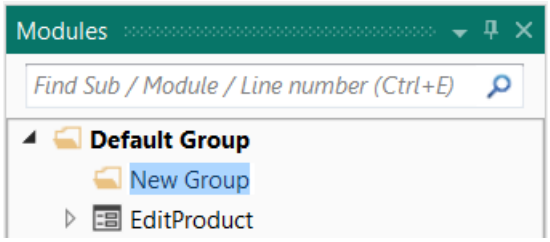
Add a Group
Add a new Module (as in the Project menu).
Add an existing Module (as in the Project menu).
Auto expand the current sub.
Expands automatically the sub when you click on it.

5.5.2.1 Add a Group

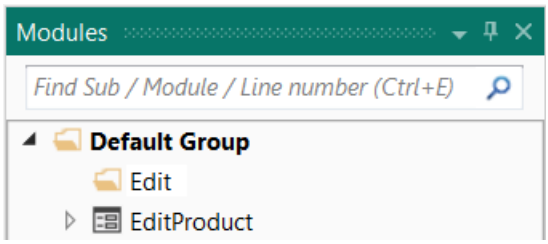


There are several modules for the 'Edit' function.

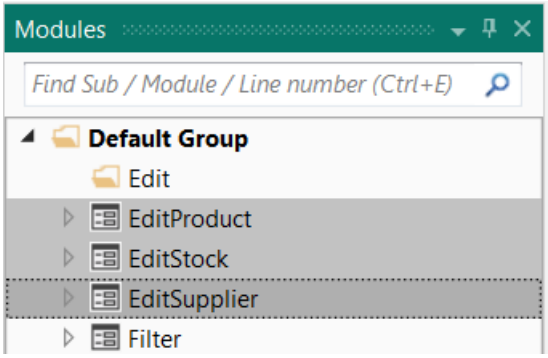
We want to create a group for these modules.



Right click on Default Group and click on Add Group.

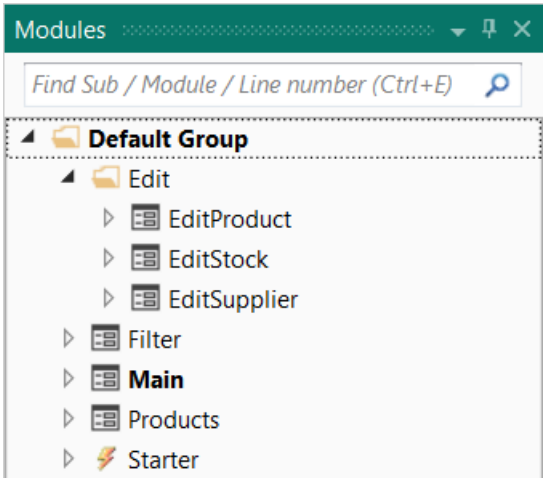


Enter 'Edit'.

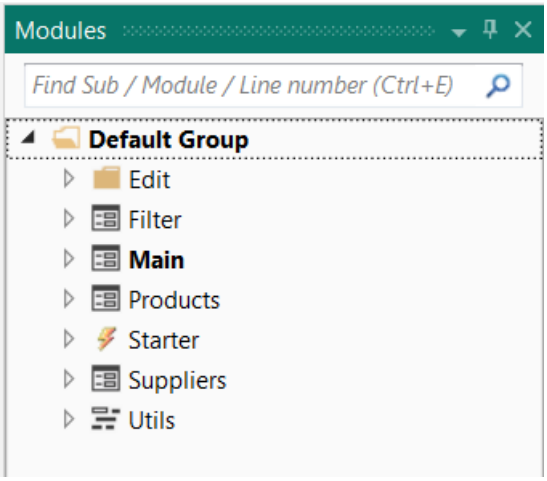


Select the three 'Edit…' modules and move them to the Edit Group.

The files remain in the Files folder of the project, they are not moved somewhere else.



The result expanded



and collapsed.

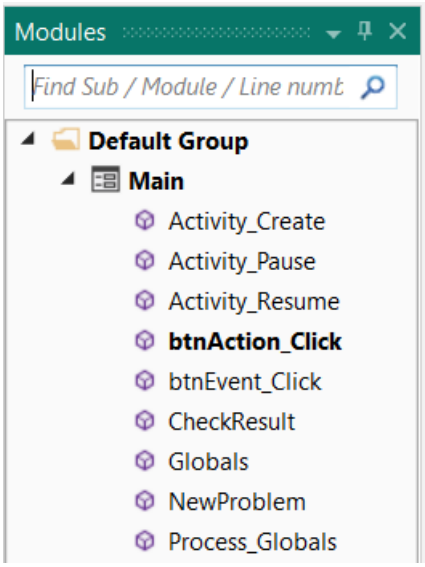
5.5.3 Find Sub / Module / Line number (Ctrl + E)

The *Find Sub / Module / Line number* function is a search engine, on the Top of the Modules Tab, to find subroutines or Modules with a given name or with a given part of the name.

You can press Ctrl + E in the code to select the Modules Tab with the *Find Sub / Module* function.

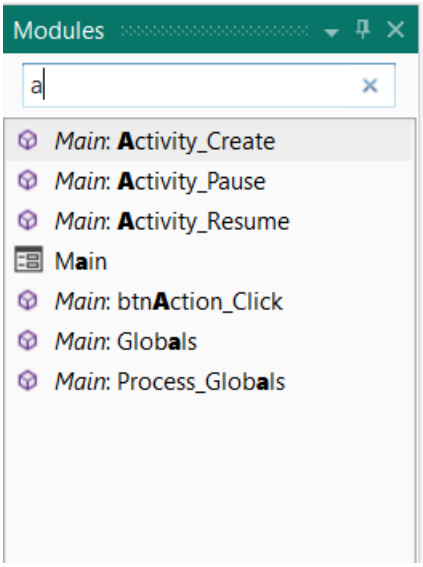
Example with the code of the SecondProgram example.

No text



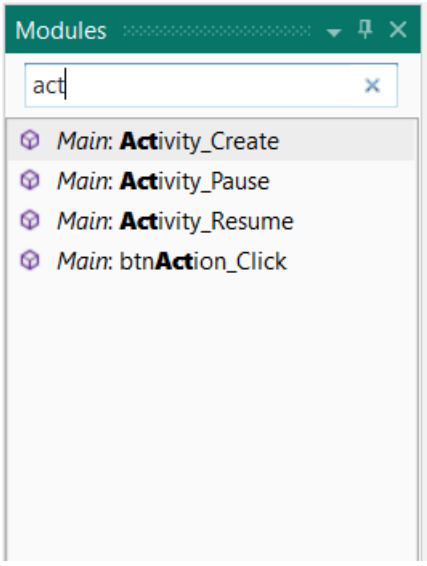
Shows all modules and all routines of the selected Module.

only the character ‘a’



Shows all modules and routines containing ‘a’.

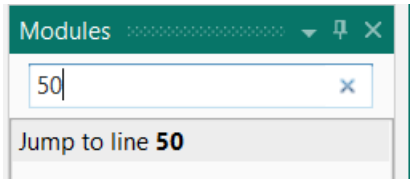
text ‘act’



Shows all modules and routines containing ‘act’.

Clicking on one item shows the code of the selected module or routine.

To jump to a given line number, enter the line number:

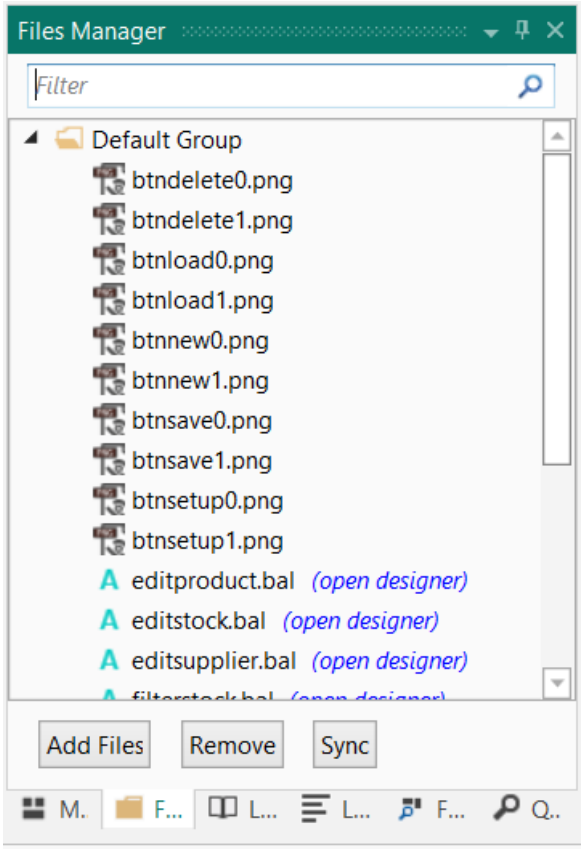


Then, press Return or click on Jump to line **50** .

5.6 Files Manager Files Manager B4A, B4i and B4J only

This window lists all the files that have been added to the project with the Add Files button.
These files are saved in the 'Files' subfolder under your main project folder.
These can be any kind of files: layouts, images, texts, etc.

All files you need in your project must be added with the Add Files button, just copying any file in the projects Files folder is not enough.

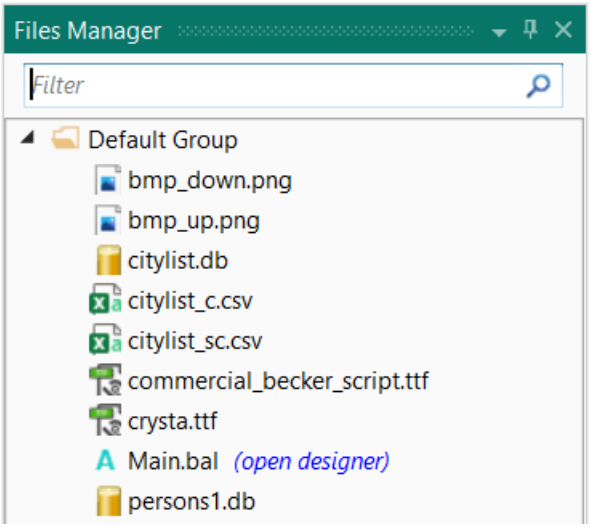


You can add, remove, synchronize or filter files.

Or click on Sync to add all the files from the projects Files folder into the File Tab.

For layout files, you can click on (open designer) to open the Designer with the selected file.

Different file types have different file icons.



png file
db database file

csv file

ttf font file

bal layout file

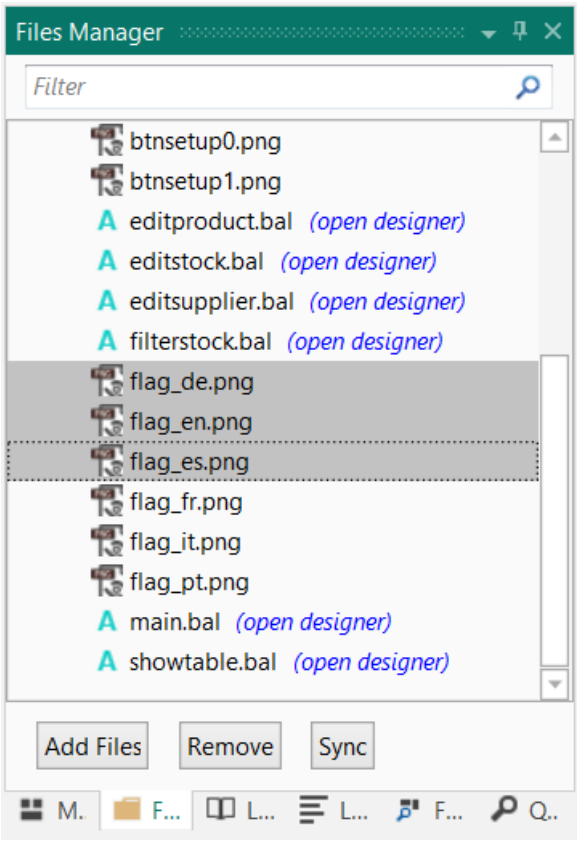
5.6.1 Add files

Click on Add Files to add files to the list.

The files in that subfolder can be accessed from your program by using the reference File.DirAssets.

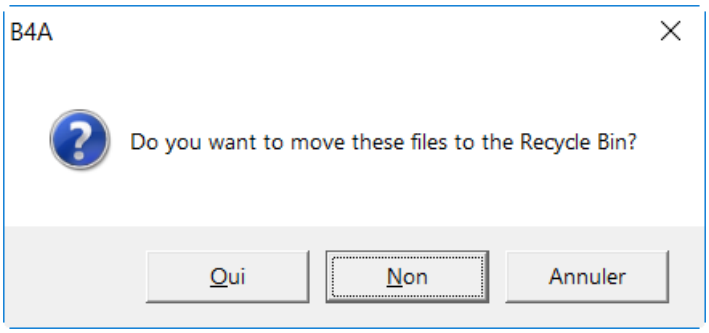
The file chooser will be shown. Select one or more files and click on Open.

5.6.2 Remove files



To delete files, select the files you want to delete and click on the Remove button.

Clicking on this button removes the selected files from the list and, if you want, from the Files folder of the project.



You are asked if you want to move the files from the 'Files' folder to the Recyxle Bin.

Oui = Yes
Non = No
Annuler = Cancel

The removed files are moved to the Recycle Bin and, if necessary can be recuperated from there.

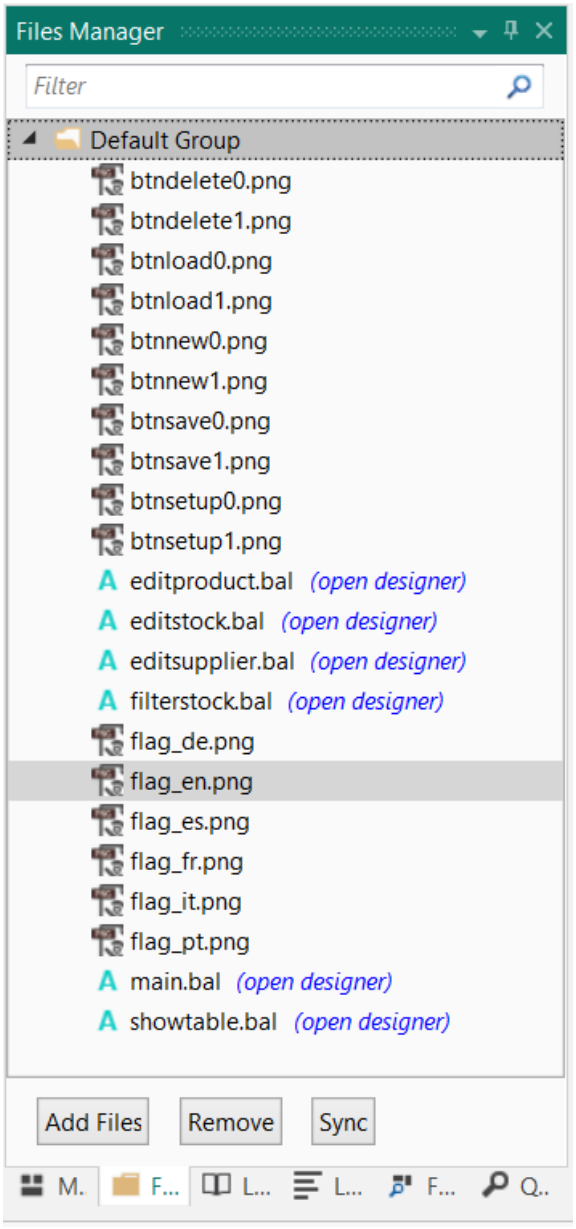
5.6.3 Synchronize files.

If you have added files into the projects Files folder from outsides the IDE, you can add those to the IDE Files Tab with the Sync button.

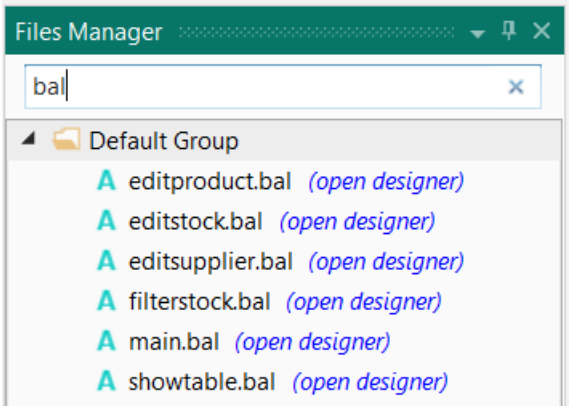
All files in the projects Files folder will be added to the Files Tab of the IDE.

5.6.4 Filter files.

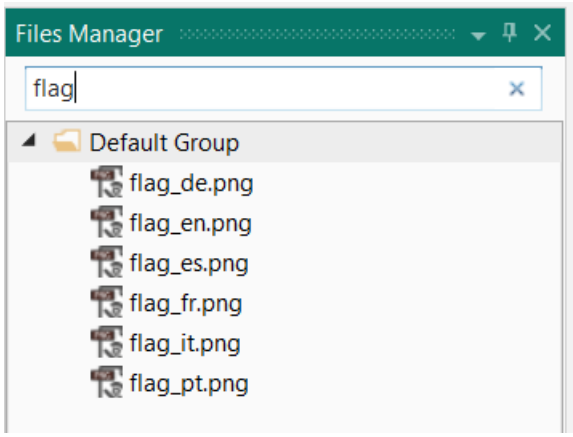
On top of the Files Manager window you can filter the files list.



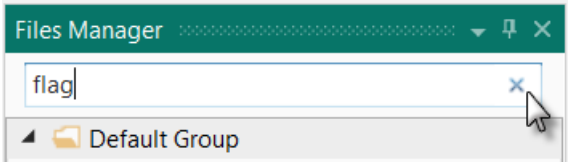
Enter ‘bal’ to filter all layout files, and the result:



Or, enter ‘flag’, and the result:

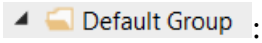


To remove the filter, click on ×.

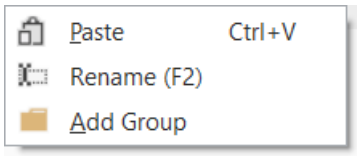


5.6.5 Context menus

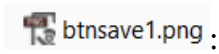
What you can do:

- Right click on a Group 

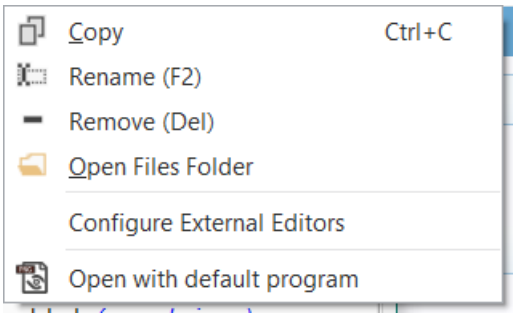
:



Paste a selected Module.
Rename the Group.
Add a Group. Adds a new group into the selected group.

- Right click on a file 

:

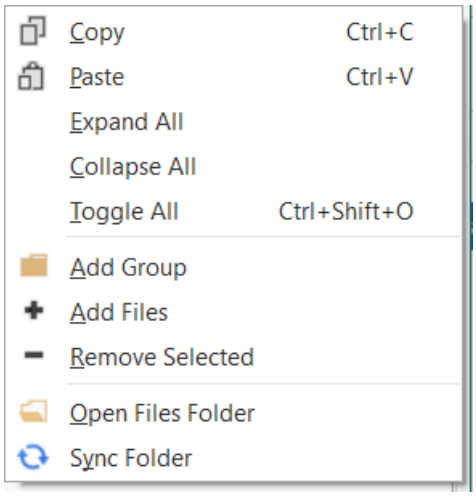


Copy the file.
Rename the file.
Remove the file.
Open the File Explorer to select any files.

Configure external editors.

Open the file with its default program

- Right click on an empty area of the Files Tab



- :

Copy the Subroutine
Paste a Subroutine from the clipboard.
Expand all the Tab content.
Collapse all the Tab content.
Toggle all the Tab content.

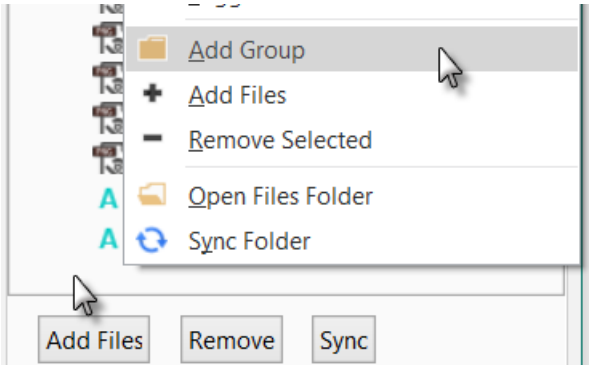
Add a Group. Adds a new group on the top level.
Add Files
Remove Selected Remove the selected files.

Open the projects Files folder.
Sync Folder.

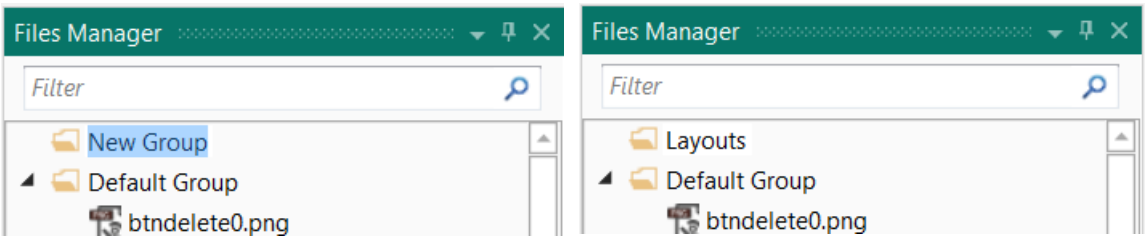
5.6.6 Add a Group

To add a group:

- Right click in an empy part of the Files Tab to add a group at the top level.
- Right click on a Group insides the Tab to add a 'subgroup' in the selected group.



Example:
Right click on an empty area of the Files Tab and click on Add Group.

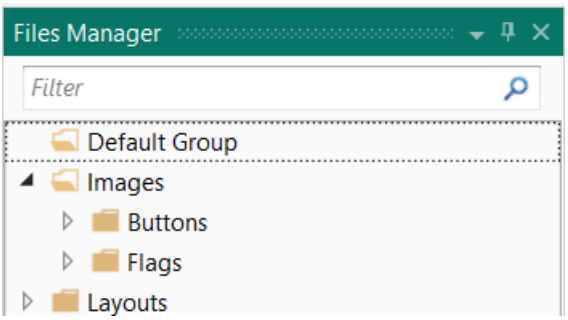


Enter 'Layouts' and move all the layout files into the Layout group.

And the result:

Expanded and collapsed.

Example with several groups:



Images, on the top level with two subgroups

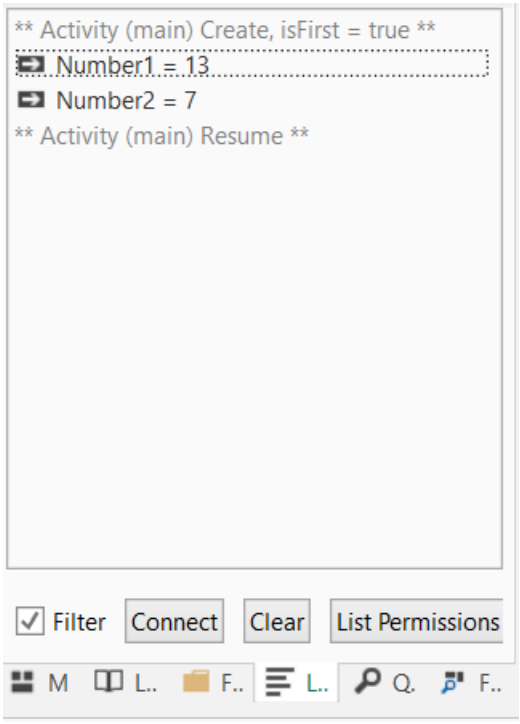
Layouts, on the top level.

5.7 Logs

Display of Log comments generated by the program when it is running.

We add the two lines 44 and 46 in the program 'SecondProgram' in the 'New' routine.
The number of the lines may be different from yours.

```
43 Sub NewProblem
44   Number1 = Rnd(1, 20)     ' Generates a random number between 1 and 9
45   Log("Number1 = " & Number1)
46   Number2 = Rnd(1, 20)     ' Generates a random number between 1 and 9
47   Log("Number2 = " & Number2)
48   lblNumber1.Text = Number1 ' Displays Number1 in label lblNumber1
49   lblNumber2.Text = Number2 ' Displays Number2 in label lblNumber2
50   lblComments.Text = "Enter the result" & CRLF & "and click on OK"
51   lblComments.Color = Colors.RGB(255,235,128) ' yellow color
52   lblResult.Text = ""    ' Sets lblResult.Text to empty
53   btn0.Visible = False
54 End Sub
```



Run the program.

Click on Connect to connect the logger.

The top area of the window shows Compile Warnings see next page.

In the lower area of the window we see the flow of the program.

** Activity (main) Create, isFirst = true **
Number1 = 9 First log message
Number2 = 1 Second log message
** Activity (main) Resume **

Filter When *Filter* is checked you will only see messages related to your program. When it is unchecked you will see all the messages running in the system. If you are encountering an error and do not see any relevant message in the log, it is worth unchecking the filter option and looking for an error message

Click on Clear to clear the Logs window.

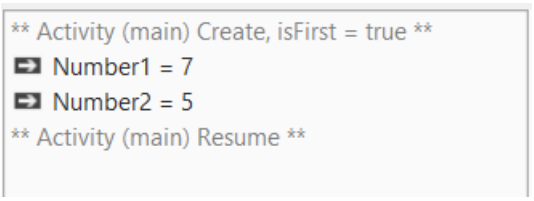
The arrow at the beginning of the Log allows to jump to the code line of the Log, see next page.

5.7.1 Jump to …

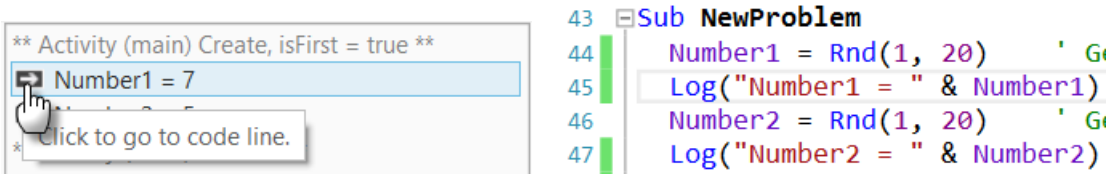
Logs in the code:

```
43  Sub NewProblem
44      Number1 = Rnd(1, 20)     ' G
45      Log("Number1 = " & Number1)
46      Number2 = Rnd(1, 20)     ' G
47      Log("Number2 = " & Number2)
48      lblNumber1.Text = Number1 '
```

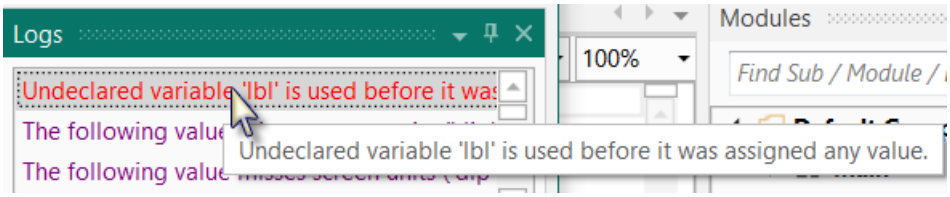
And in the Logs you will see this:

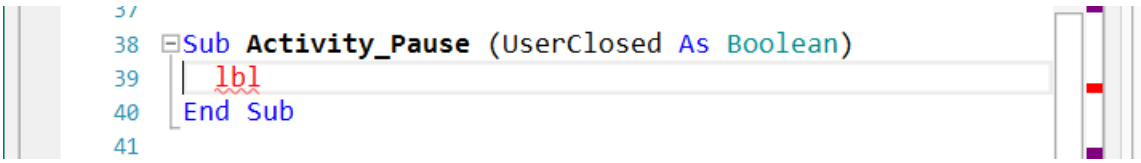


Click on an arrow ➡ to jump to the Log line in the code.



On top of the Logs Tab, if there are warnings or errors, hovering over a line shows the full message and clicking on it jumps to the concerned code line in the editor.

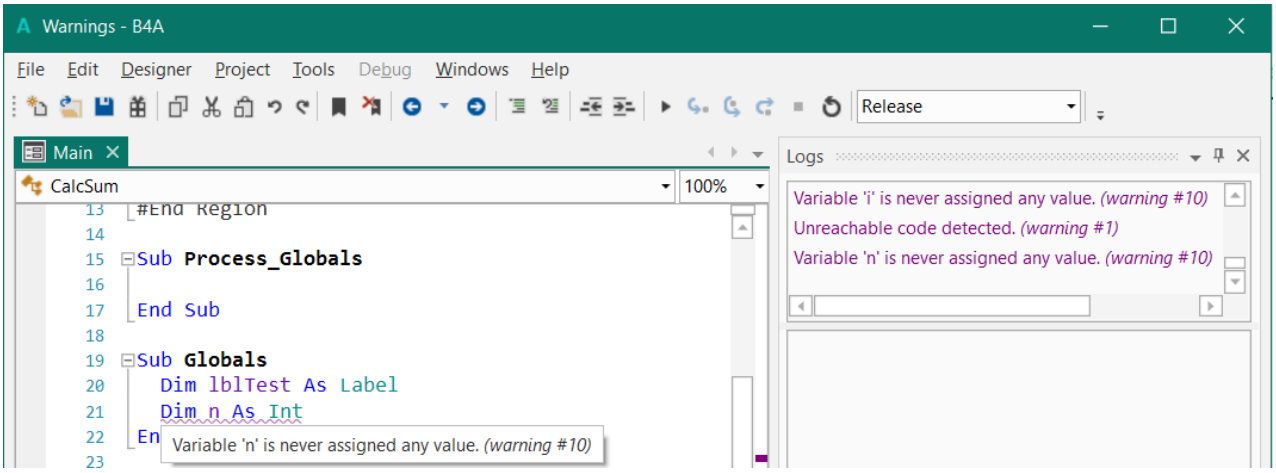




5.7.2 Compile Warnings

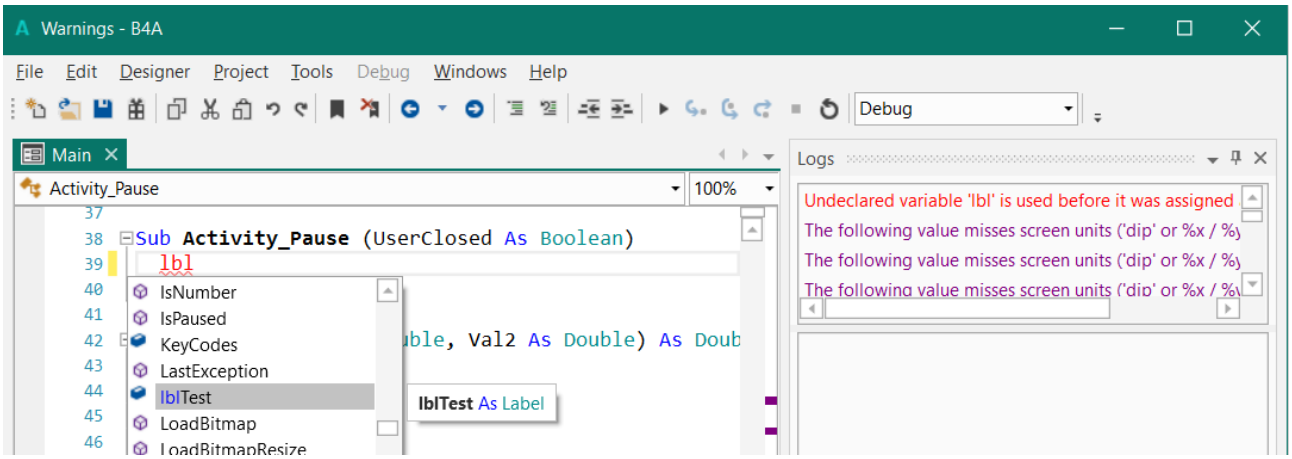
B4X includes a warning engine. The purpose of the warning engine is to find potential programming mistakes as soon as possible. The examples are from the Warnings project.

The compile-time warnings appear above the logs and in the code itself when hovering with the cursor above the code line.
The code lines which cause a warning are underlined like this `Dim i As Int`.



Clicking on the warning in the list will take you to the relevant code.

The warning engine runs as soon as you type.



Typing for example ‘lbl’ at the beginning of a line shows immediately:

- `lbl` in red, because lbl was not declared.
- a warning Undeclared variable 'lbl' is used before it was assigned any value.
- the auto complete pop up window with suggestion containing the written characters.

5.7.2.1 Ignoring warnings

You, as the developer, can choose to ignore any warning. Adding an "ignore" comment will disable all the warnings for that specific line:

You can also disable warnings from a specific type in the module by adding the #IgnoreWarning on the top in one of the code in the Attribute regions.

For example, to disable warnings #10 and #12:

```
#Region  Activity Attributes
   #FullScreen: False
   #IncludeTitle: True
   #IgnoreWarnings: 10, 12
#End Region
```

You find the warning numbers at the end of each warning line.

5.7.2.2 List of warnings

The warning may be different in the four products, the list is not exhaustive.

1: Unreachable code detected.
2: Not all code paths return a value.
3: Return type (in Sub signature) should be set explicitly.
4: Return value is missing. Default value will be used instead.
5: Variable declaration type is missing. String type will be used.
6: The following value misses screen units ('dip' or %x / %y): {1}.
7: Object converted to String. This is probably a programming mistake.
8: Undeclared variable '{1}'.
9: Unused variable '{1}'.
10: Variable '{1}' is never assigned any value.
11: Variable '{1}' was not initialized.
12: Sub '{1}' is not used.
13: Variable '{1}' should be declared in Sub Process_Globals.
14: File '{1}' in Files folder was not added to the Files tab.\nYou should either delete it or add it to the project.\nYou can choose Tools - Clean unused files.
15: File '{1}' is not used.
16: Layout file '{1}' is not used. Are you missing a call to Activity.LoadLayout?
17: File '{1}' is missing from the Files tab.
18: TextSize value should not be scaled as it is scaled internally.
19: Empty Catch block. You should at least add Log(LastException.Message).
20: View '{1}' was added with the designer. You should not initialize it.
21: Cannot access view's dimension before it is added to its parent.
22: Types do not match.
23: Modal dialogs are not allowed in Sub Activity_Pause. It will be ignored.
24: Accessing fields from other modules in Sub Process_Globals can be dangerous as the initialization order is not deterministic.
25: Sub '{0}' not found.
26: Add android:targetSdkVersion="19" to the manifest editor (after minSdkVersion).
27: AndroidManifest.xml is read-only or Do not overwrite manifest file option is checked. Use the manifest editor instead.
28: It is recommended to use a custom theme or the default theme.
Remove SetApplicationAttribute(android:theme, “@android:style/Theme.Holo”) from the manifest editior.
32: Library '{0}' is not used.
33: DoEvents is deprecated. It can lead to stability issues. Use Sleep(0) instead (if really needed).
34: Msgbox and other modal dialogs are deprecated. Use the async methods instead.
35: Comparison of Object to other types will fail if exact types do not match.\nBetter to put the object on the right side of the comparison.

Runtime warnings
1001: Panel.LoadLayout should only be called after the panel was added to its parent.
1002: The same object was added to the list. You should call Dim again to create a new object.
1003: Object was already initialized.
1004: FullScreen or IncludeTitle properties in layout file do not match the activity attributes settings.

1: Unreachable code detected.

There is some code which will never be executed.
This can happen if you have some code in a Sub after a Return statement.

2: Not all code paths return a value.

```
Sub Calc(Val1 As Double, Val2 As Double, Operation As String) As Double
   Select Operation
   Case "Add"
      Return (Val1 + Val2)
   Case "Sub"
      Return (Val1 - Val2)
   Case "Mult"
      Return (Val1 * Val2)
   Case "Div"

   End Select
End Sub
```

In the `Case "Div"` path no value is returned !

Other example:
Wrong code

```
Sub Activity_KeyPress(KeyCode As Int) As Boolean
   Private Answ As Int
   Private Txt As String

   If KeyCode = KeyCodes.KEYCODE_BACK Then' Checks if the KeyCode is BackKey
      Txt = "Do you really want to quit the program ?"
      Answ = Msgbox2(Txt,"A T T E N T I O N","Yes","","No",Null) ' MessageBox
      If Answ = DialogResponse.POSITIVE Then  ' If return value is Yes then
       Return False ' Return = False  the Event will not be consumed
      Else                ' we leave the program
       Return True        ' Return = True    the Event will be consumed to avoid
      End If              ' leaving the program
   End If
End Sub
```

Correct code

```
Sub Activity_KeyPress(KeyCode As Int) As Boolean
   Private Answ As Int
   Private Txt As String

   If KeyCode = KeyCodes.KEYCODE_BACK Then' Checks if the KeyCode is BackKey
      Txt = "Do you really want to quit the program ?"
      Answ = Msgbox2(Txt,"A T T E N T I O N","Yes","","No",Null) ' MessageBox
      If Answ = DialogResponse.POSITIVE Then  ' If return value is Yes then
       Return False ' Return = False  the Event will not be consumed
      Else                ' we leave the program
       Return True        ' Return = True    the Event will be consumed to avoid
      End If              ' leaving the program
   Else
      Return True         ' Return = True    the Event will be consumed to avoid
   End If                 ' leaving the program
End Sub
```

3: Return type (in Sub signature) should be set explicitly.

Wrong code
```
Sub Calc(Val1 As Double, Val2 As Double, Operation As String)
```

Correct code
```
Sub Calc(Val1 As Double, Val2 As Double, Operation As String) As Double
```
The return type must be declared !

4: Return value is missing. Default value will be used instead.

Wrong code
```
Sub CalcSum(Val1 As Double, Val2 As Double) As Double
   Private Sum As Double

   Sum = Val1 + Val2
   Return
End Sub
```

Correct code
```
Sub CalcSum(Val1 As Double, Val2 As Double) As Double
   Private Sum As Double

   Sum = Val1 + Val2
   Return Sum
End Sub
```

5: Variable declaration type is missing. String type will be used.

Wrong code
```
Sub Calc(Val1, Val2 As Double, Operation As String) As Double
```

Correct code
```
Sub Calc(Val1 As Double, Val2 As Double, Operation As String) As Double
```

In sub declarations each variable needs its own type declaration.
But in Private, Public or Dim declarations it's allowed, in the line below both variables are Doubles:
```
Private Val1, Val2 As Double
```

6: The following value misses screen units ('dip' or %x / %y): {1}.

Wrong code
```
Activity.AddView(lblTest, 10, 10, 150, 50)
```

Correct code
```
Activity.AddView(lblTest, 10dip, 10dip, 150dip, 50dip)
```

In the example above you will get four warnings, one for each value.
For view dimensions you should use dip, %x or %y values.

7: Object converted to String. This is probably a programming mistake.

8: Undeclared variable '{1}'.

Wrong code
```
Sub SetHeight
   h = 10dip
End Sub
```

Correct code
```
Sub SetHeight
   Private h As Int
   h = 10dip
End Sub
```
The variable `h` was not declared. You see it also with the red color.

9: Unused variable '{1}'.

```
Sub SetHeight
   Private h As Int
   h = 10dip
End Sub
```

This warning tells that the variable `h` is not used.
It is declared and assigned a value, but it is not used !

This code gives no warning because variable `h` is used:
```
Sub SetHeight
   Private h As Int
   h = 10dip
   lblTest.Height = h
End Sub
```

10: Variable '{1}' is never assigned any value.

```
Sub Test
   Private h As Int

End Sub
```
This warning shows that the variable `h` is declared but not assigned any value.
Correct code see above.

11: Variable '{1}' was not initialized.

Wrong code
```
   Private lst As List
   lst.Add("Test1")
```

Correct code
```
   Private lst As List
   lst.Initialize
   lst.Add("Test1")
```

Variables (objects) like List or Map must be initialized before they can be used.
Views added by code must also be initialized before they can be added to a parent view.

12: Sub '{1}' is not used.

This warning is displayed if a Sub routine is never used.

13: Variable '{1}' should be declared in Sub Process_Globals.

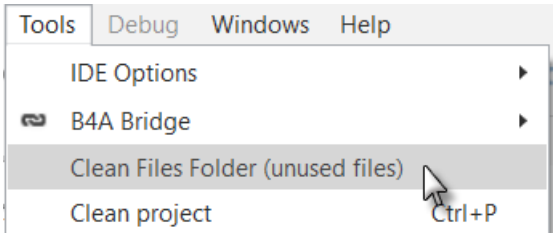
Wrong code :
```
Sub Globals
   Public Timer1 As Timer
   Public GPS1 As GPS
```

Correct code :
```
Sub Process_Globals
   Public Timer1 As Timer
   Public GPS1 As GPS
```

Certain objects like Timers and GPS should be declared in Process_Globals, not in Globals.

14: File '{1}' in Files folder was not added to the Files tab.

You are using a file which is in the Files folder, but was not added to the Files tab.
You should:
- Delete it from the Files subfolder. Don't forget to make a backup copy before deleting it.
- Add it to the project in the Files tab.
- Use Clean Files Folder (unused files) in the Tools menu.

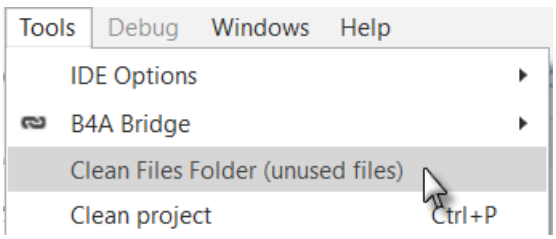


15: File '{1}' is not used.

You have files in the Files folder that are not used.
You should remove them from the Files folder.
Or you can clean the Files folder from within the Tools menu (see above).

16: Layout file '{1}' is not used. Are you missing a call to Activity.LoadLayout?

You have a layout file in the Files folder that is not used.
You should add LoadLayout or you can remove the layout file from the Files folder.
Or you can clean the Files folder in the Tools menu.



17: File '{1}' is missing from the Files tab.

The given file is in the Files tab but is missing in the Files folder. You should add it.
See chapter 4.3.2 Files

18: TextSize value should not be scaled as it is scaled internally.

Wrong code
```
lblTest.TextSize = 16dip
```

Correct code
```
lblTest.TextSize = 16
```

TextSize values are pixel and density independent. Their unit is the typographic point, a typographic unit, and must be given absolute values and not dip values.

19: Empty Catch block. You should at least add Log(LastException.Message).

Wrong code
```
   Try
      imvImage.Bitmap = LoadBitmap(File.DirRootExternal, "image.jpg")
   Catch

   End Try
```

Correct code
```
   Try
      imvImage.Bitmap = LoadBitmap(File.DirRootExternal, "image.jpg")
   Catch
      Log(LastException.Message)
   End Try
```

It is recommended to add at least `Log(LastException.Message)` in the Catch block instead of leaving it empty.

20: View '{1}' was added with the designer. You should not initialize it.

A View defined with the Designer in a layout file must not be initialized !
Only views added by code need to be initialized.

21: Cannot access view's dimension before it is added to its parent.

You must add a view to a parent view before you can access its dimensions.
When you add a view by code its dimensions are defined when you add it with AddView.

22: Types do not match.

23: Modal dialogs are not allowed in Sub Activity_Pause. It will be ignored.

Modal dialogs like MessageBox should not be used in the Activity_Pause routine.

24: Accessing fields from other modules in Sub Process_Globals can be dangerous as the initialization order is not deterministic.

25: Sub '{0}' not found.

The specified sub has not been found.

26: Add android:targetSdkVersion="19" in the ManifestEditor (after minSdkVersion).

<uses-sdk android:minSdkVersion="5" android:targetSdkVersion="19"/>
Instead of:
<uses-sdk android:minSdkVersion="5""/>

27: AndroidManifest.xml is read only. Use the Manifest Editor.

28: It is recommended to use a custom theme or the default theme.
Remove SetApplicationAttribute(android:theme, "@android:style/Theme.Holo") from the manifest editior.

This was set automatically in older versions of B4A. No more needed.

32: Library 'xxxx' is not used.

Remove the unused library.

33: DoEvents is deprecated.

It can lead to stability issues. Use Sleep(0) instead (if really needed).

34: Msgbox and other modal dialogs are deprecated.

Use the async methods instead.

35: Comparison of Object to other types will fail if exact types do not match.

Better to put the object on the right side of the comparison.

Runtime warnings:

1001: Panel.LoadLayout should only be called after the panel was added to its parent.

1002: The same object was added to the list. You should call Dim again to create a new object.

1003: Object was already initialized.

1004: FullScreen or IncludeTitle properties in layout file do not match the activity attributes settings.

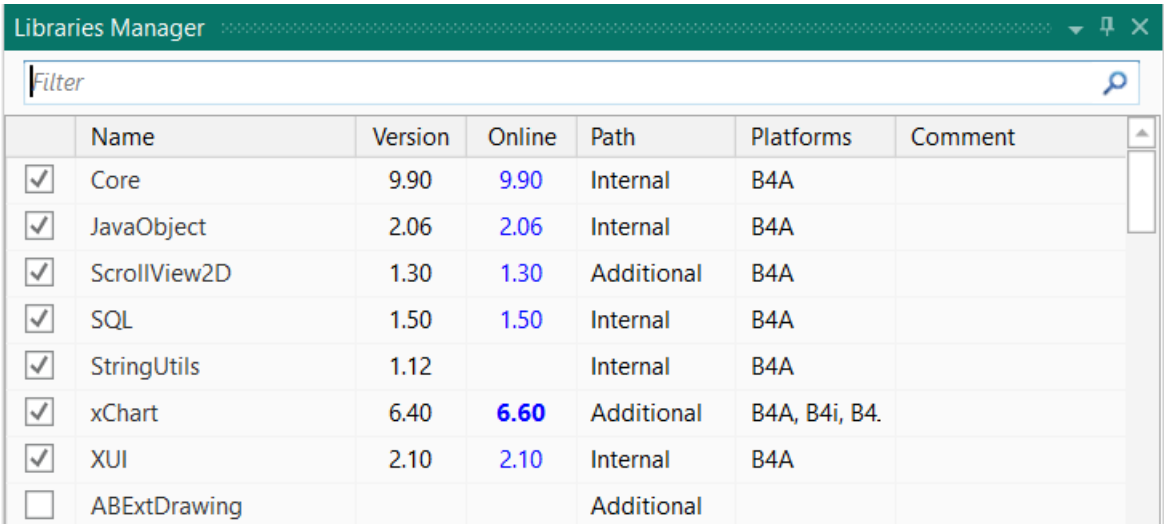
5.8 Libraries Manager

The “Libraries Manager” Tab contains a list of the available libraries that can be used in the project.

The ontenet of the list depends on the available libraries in the given IDE.
The images are an example with B4A.

Libraies are explained in detail in the
B4X Basic Language Booklet.

Check the libraries you need for your project.
Make sure that you have the latest version of the libraries.
If there does exist a newer version in the forum, the version number in the Online column is displayed in bold characters, like the xChart library in the image.



	Name	Version	Online	Path	Platforms	Comment
☑	Core	9.90	9.90	Internal	B4A	
☑	JavaObject	2.06	2.06	Internal	B4A	
☑	ScrollView2D	1.30	1.30	Additional	B4A	
☑	SQL	1.50	1.50	Internal	B4A	
☑	StringUtils	1.12		Internal	B4A	
☑	xChart	6.40	**6.60**	Additional	B4A, B4i, B4.	
☑	XUI	2.10	2.10	Internal	B4A	
☐	ABExtDrawing			Additional		

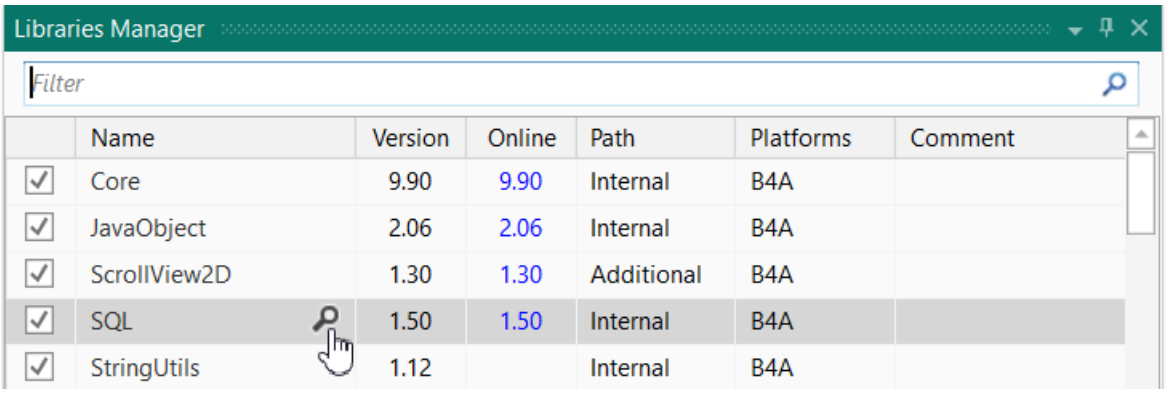
The used libraries are shown on top of the list.
As soon as you select one it moves to the top of the list.

Additional information is provided for all selected libraries:

- Name Name of the library.
- Version Version number of the library on your computer.
- Online Shows the number of the latest version in the forum.
- Path Path of the library.
  - Internal Internal library, shipped with the platform.
  - Additional Additional library you copied to your AdditionalLibraries folder.
- Platforms The library is valid for the given platforms.
- Comment Comment of the library.
  Comments can be added by the library developpers.

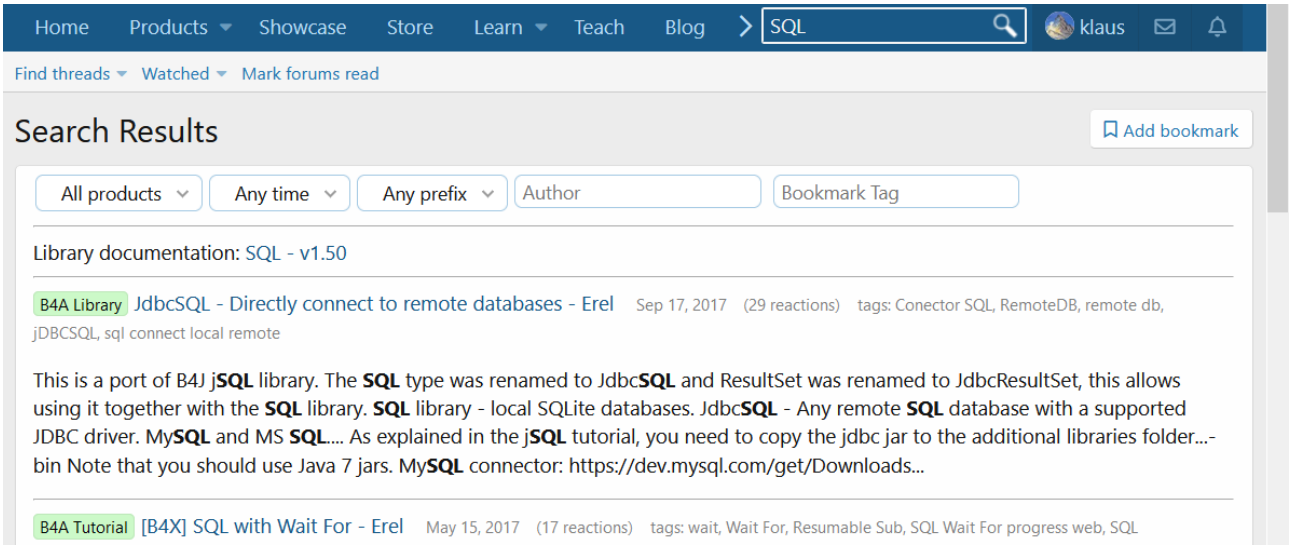
5.8.1 Search function

When you hover over the library names you see this:

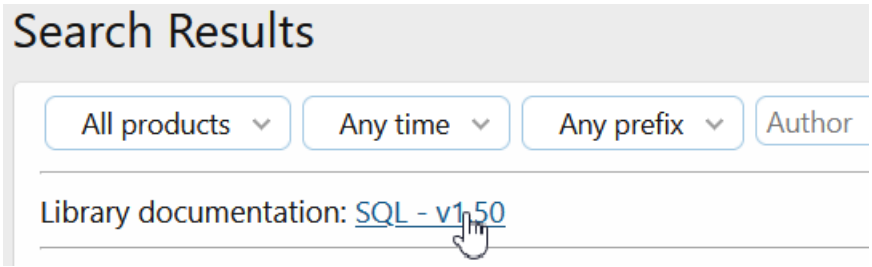


When you click on  , the search function of the forum is launched.

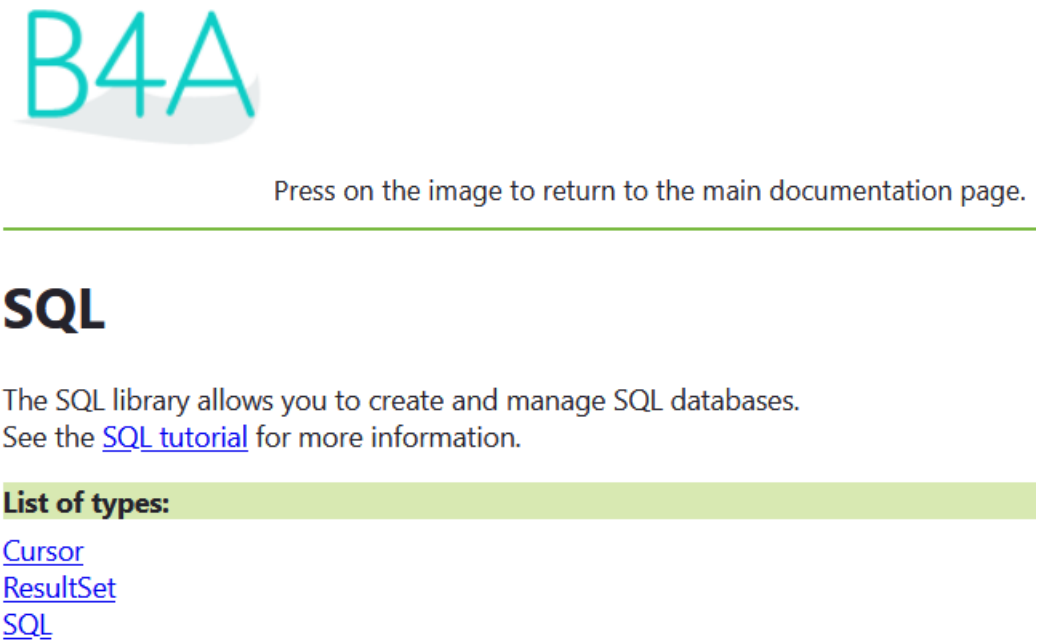
The result is the same as if you searched in the forum with: 



You may see on top of the list: Library documentation : Library name



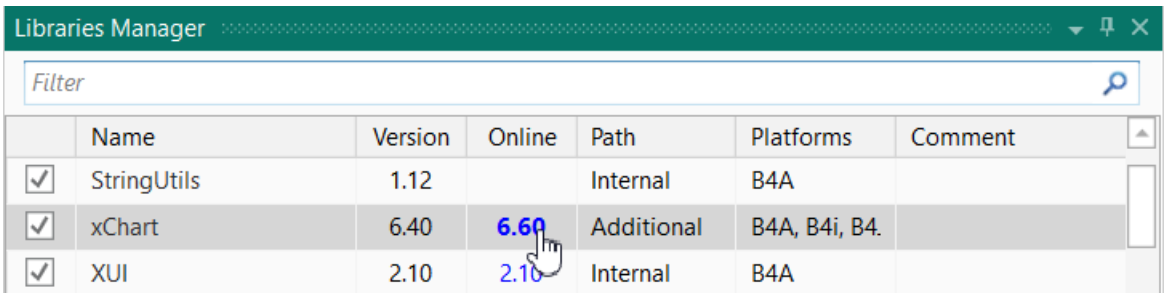
Click on the link to show to the online documentation of the library.



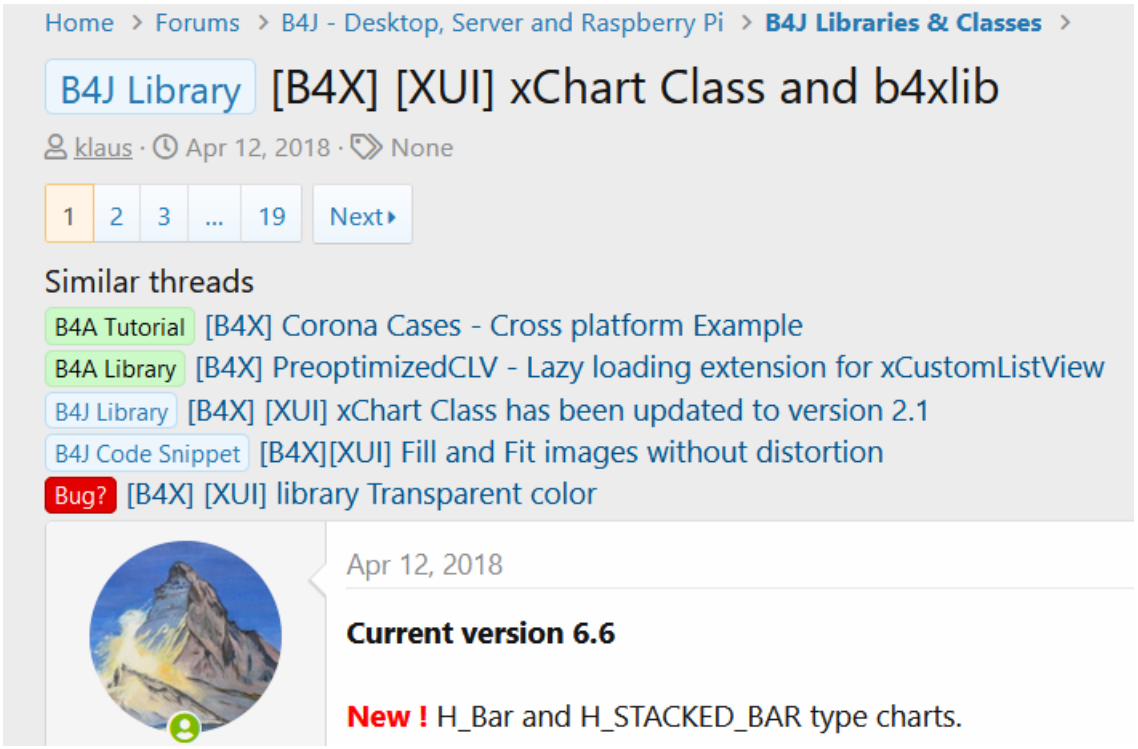
5.8.2 Online version number

When you see an Online version number in bold characters, this means that there is a newer version in forum !

Click on the version number.



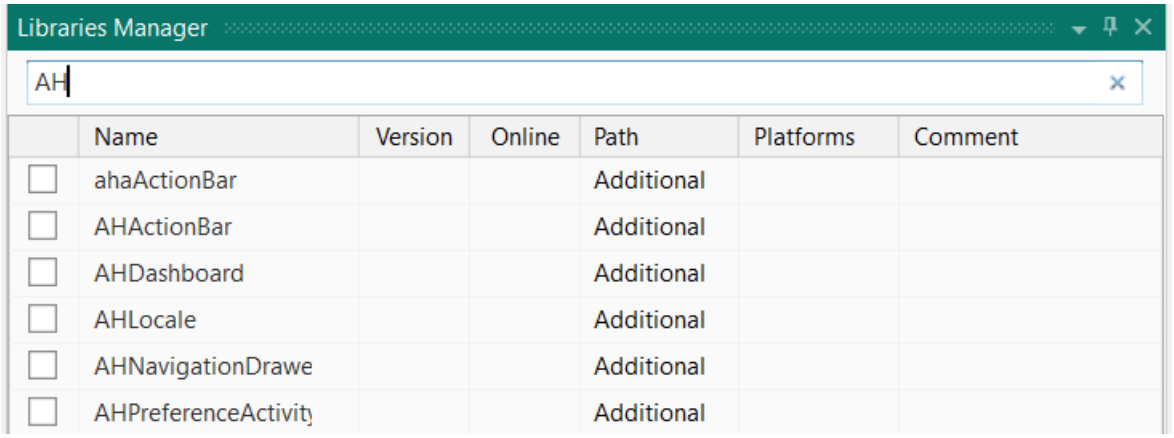
You get directly to the first post of the library thread where you can download the new version from.



5.8.3 Filter function

On the top of the Tab you find a field to filter the libraries.

Enter 'AH' and you get all libraries containing the text AH.

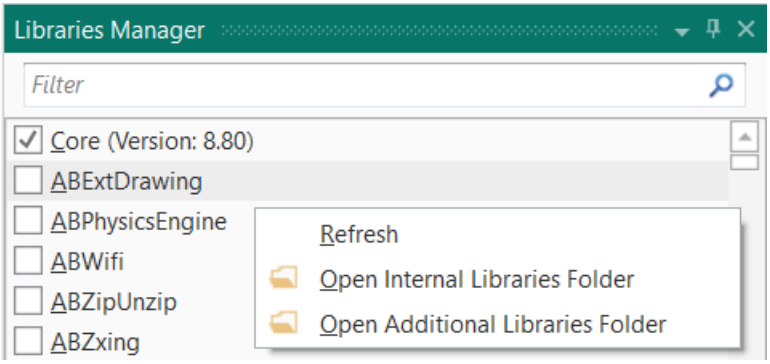


The list of all additional libraries can be found here:
B4A, B4i, B4J, B4R

Clicking on a link in the list shows the documentation.

5.8.4 Context menu

Right click in the Libraies Manager Tab:



You can:

- Refresh the Tab content. Useful if you updated an internal or additional library.
- Open the Internal Libraries Folder.
- Open the Additional Libraries Folder.

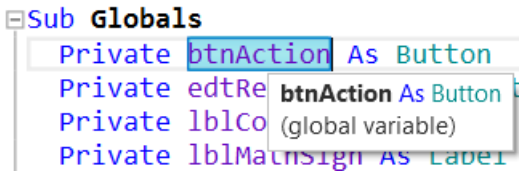
5.9 Quick Search

Quick Search allows to search for any text occurrences in the code of the whole project.
Examples with the SecondProgram code.

Several possibilities to select the Quick Search function:

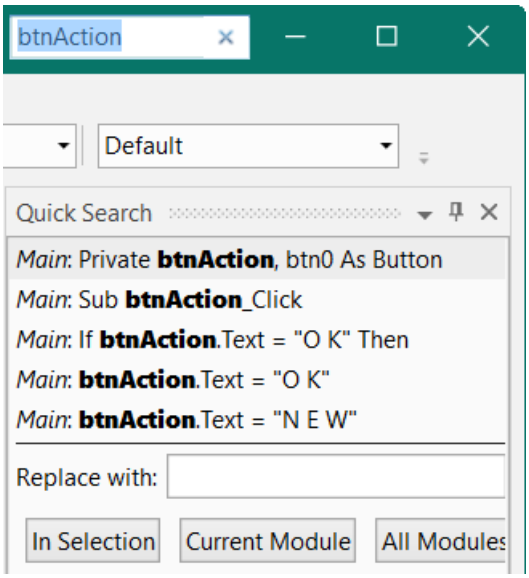
- Ctrl + F, the easiest and most efficient way.
- Click on the Quick Search Tab in the lower right corner of the IDE.
- Click on Quick Search Ctrl+F in the Edit menu.

Example:



In the code double click on `btnAction` to select it and press Ctrl + F.

You get the window below in the Tab area.

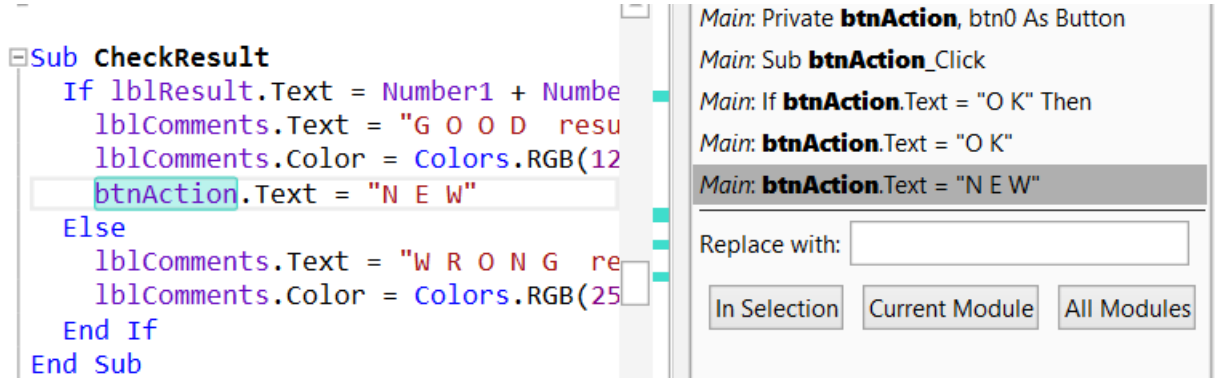


The selected text is shown in the title bar.

The list shows the occurrences in all Modules.

In each line you find the Module name and the line content.

Clicking on a line in the list moves the cursor directly to the selected occurrence in the code.



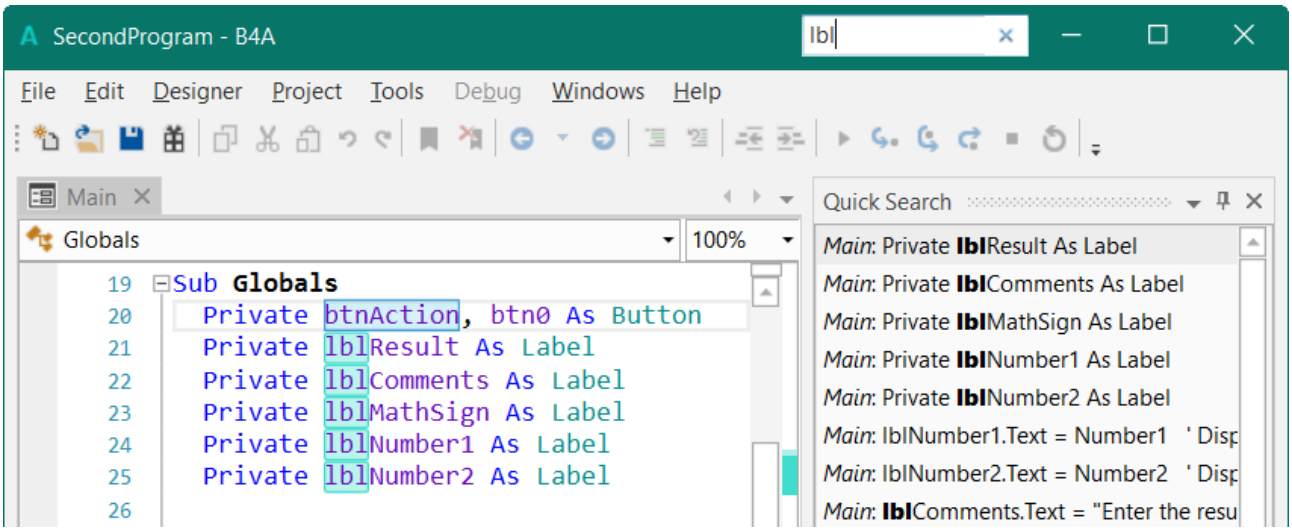
To remove the selection click on on the top right of the title bar.

You can also enter any text in the search field:

As an example, enter *lbl* in the Search field and you get the window below where you find all lines containing the text you entered, *lbl* in this example.

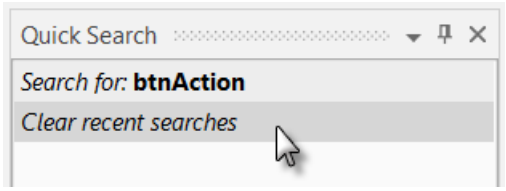
The search text is highlighted in all code lines containing this text.

Clicking on one of the lines in the list jumps directly to this line in the IDE.



Click on  to remove a search.

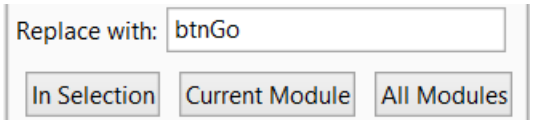
You will see a list of the last searches.



Click on Clear recent searches to remove all recent searches.

Items are added to the recent items when:
1. You select one of the results or click Return which selects the first result.
2. You select text in your code and click on Ctrl + F to search for it.

You can replace a text either in the selected code, in the current module or in all modules.
Enter the new name and click either on In Selection, Current Module or All Modules.



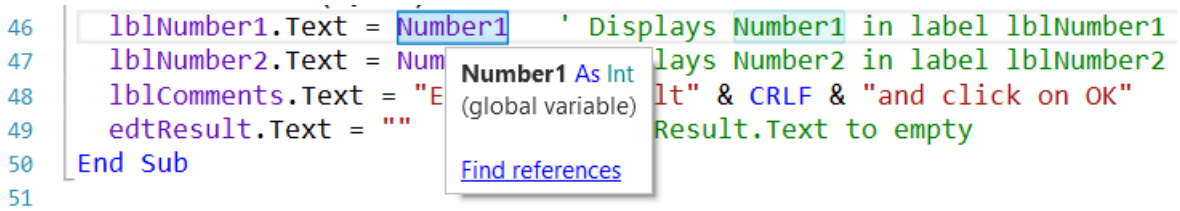
5.10 Find All References (F7)

This is a search engine to find all references for a given object (view, variable).

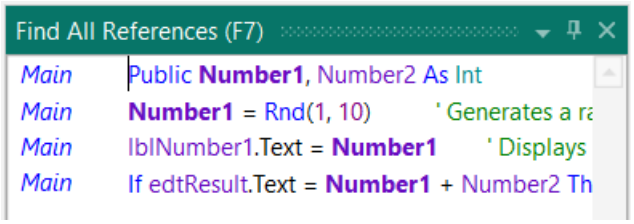
Click on the Find All References (F7) Tab or press F7 to get the screen below showing a list of all code lines with the selected reference or the first object in the current line.

Example with the code of SecondProgram.

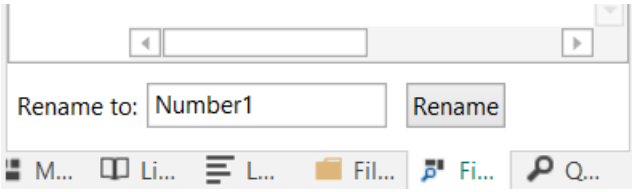
Select in the code in line 46 `Number1`.



Click on Find references or press F7 and you get the list below with all code lines containing the selected object.
The first words is the module name. Clicking on it, jumps to that line in the IDE code area.



Clicking on a line in the list shows that line in the middle of the IDE code area.



At the bottom of the Tab you can Rename the selected object.

Enter a new name and click on Rename.

6 Navigation in the IDE

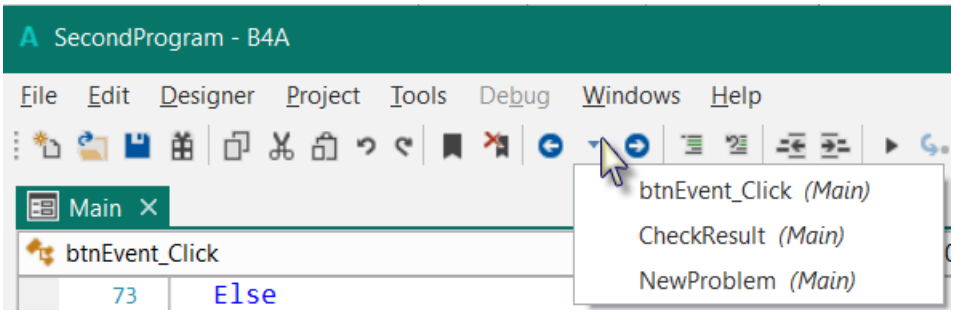
Advices given by Erel in the forum

6.1 Alt + Left / Alt + Right Move backwards and forwards

Moves backwards and forwards based on the navigation stack. This is useful to jump back and forth between the last recent subs.

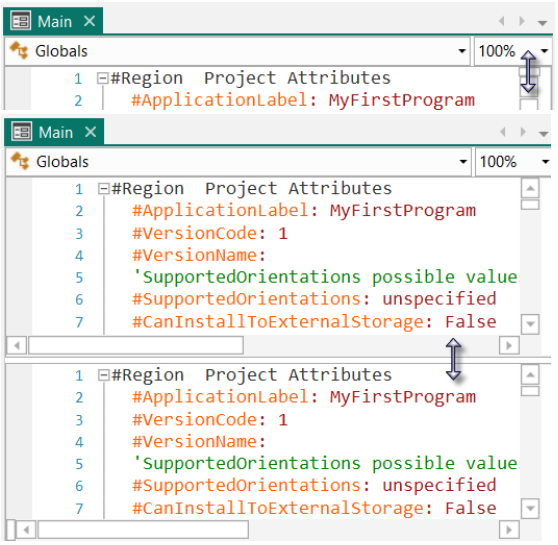
6.2 Alt + N Navigation stack menu

Opens the navigation stack menu. You can then choose the location with the up and down keys.

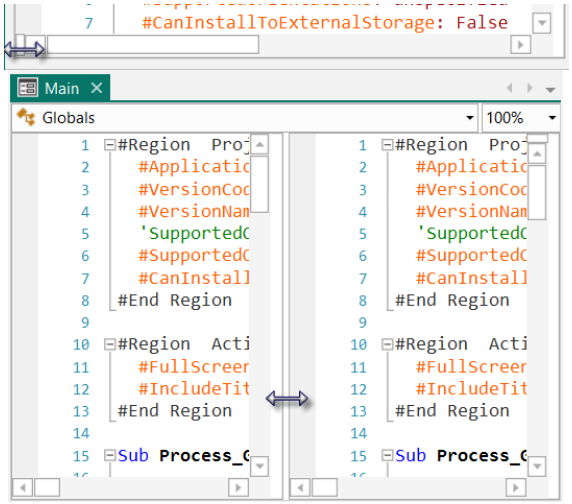


6.3 Split the screen

If you are working on two locations in the same module then you can split the code editor (it can be split again vertically):

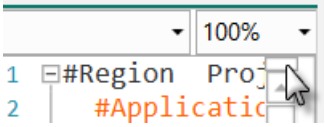


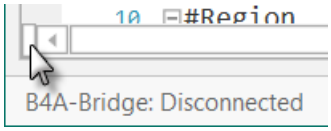
Horizontally



Vertically

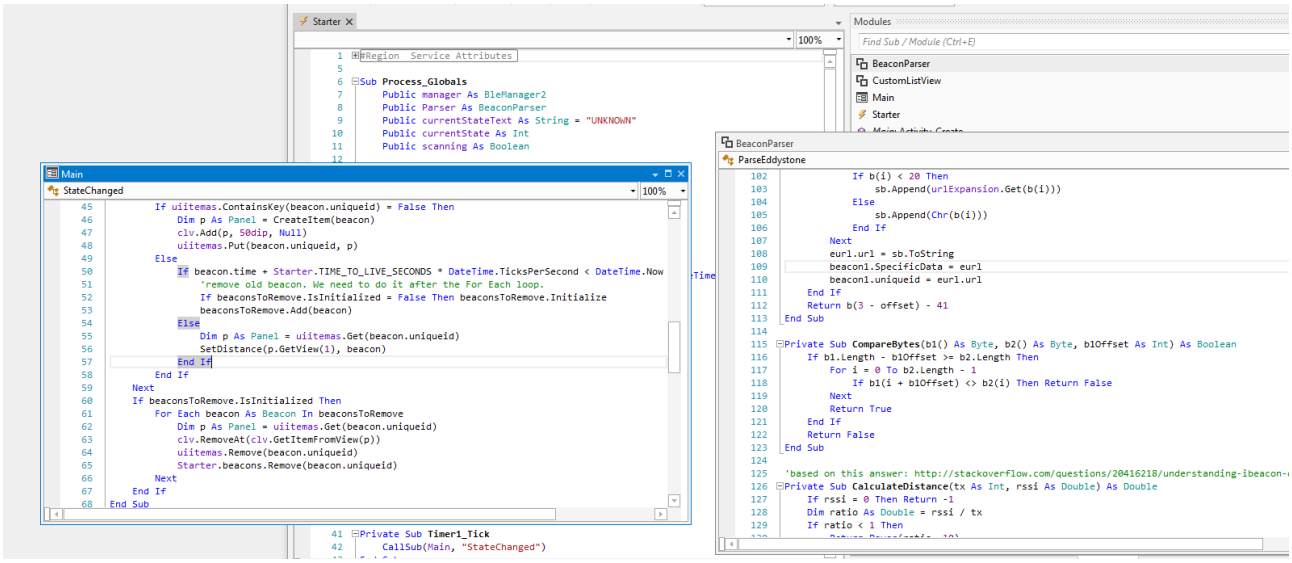
You can also double click on the small rectangles to split the screen.





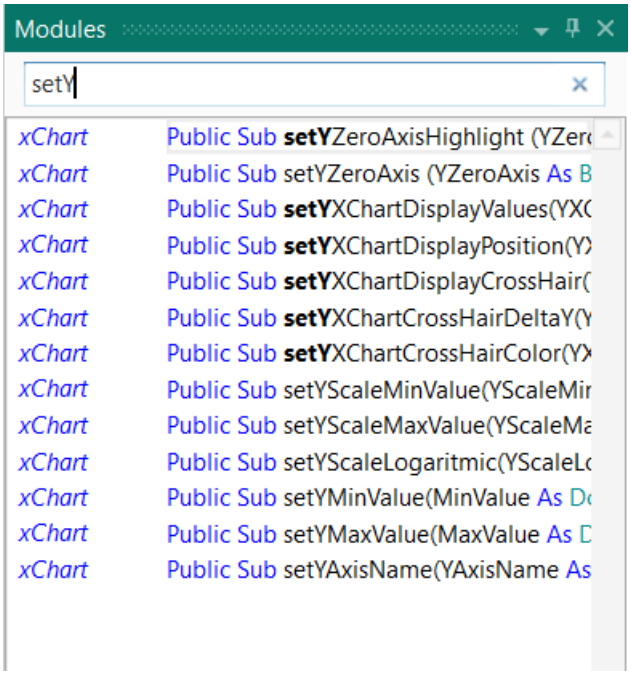
6.4 Multiple windows

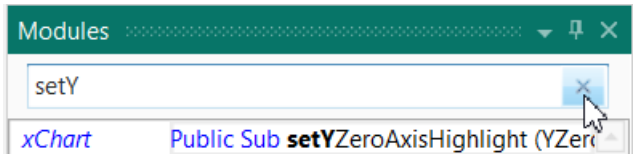
If you are working with multiple modules you can move the modules out of the main IDE as separate windows.



6.5 Ctrl + E Search for sub or module

Ctrl + E - searches for sub or module. Very useful when working with large projects.
On top of the Modules Tab enter characters to search for subroutines.
In the list you find the module name and the subroutines containing the characters you entered.





Click on to unfilter.

6.6 Ctrl + Click on any sub or variable

Ctrl + Click on any sub or variable to jump to the declaration location.

In the example above, YXChart1 is a CustomView Class module xChart and DrawChart is a subroutine in this module, and the editor jumps directly to the DrawChart subroutine in the xChart module.

6.7 F7 - Find all references

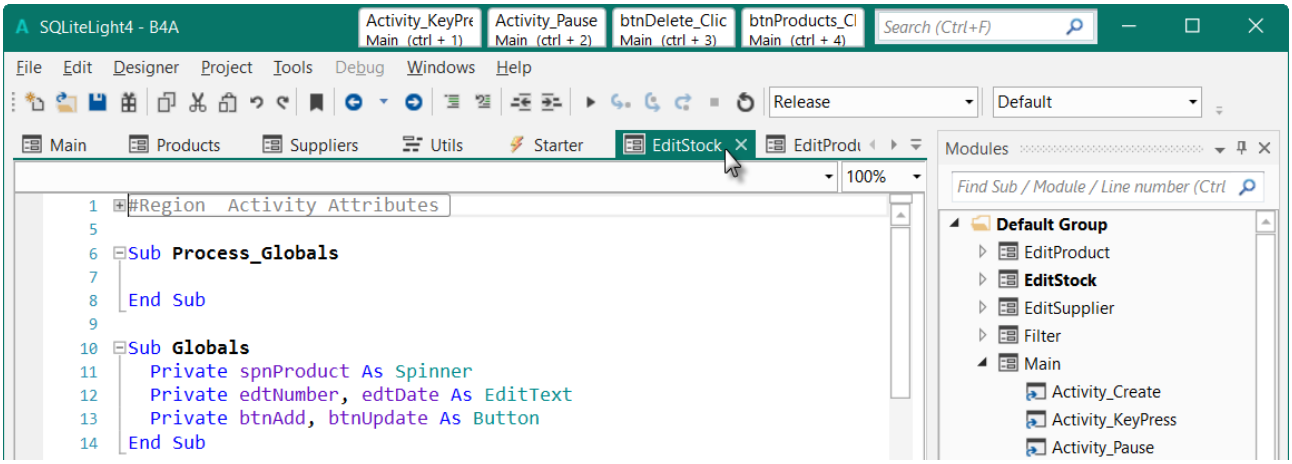
Not exactly related to navigation but is also useful when working with large projects.
Details in Find all references.

6.8 Ctrl + F Quick Search

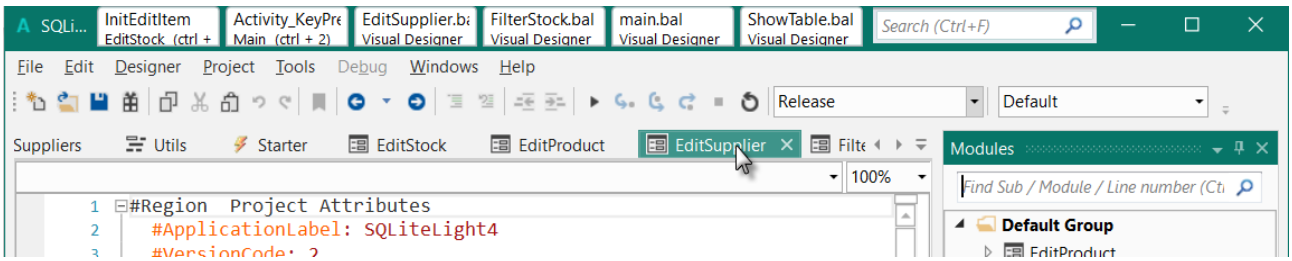
Ctrl + F - Index based quick search. Details in Quick Search.

6.9 Scrolling module Tabs

In a project with a certain number of modules, not all are visible at the same time.
Point the mouse cursor on a mosle Tab like in the picture.



Then move the mouse wheel, the Tabs will scroll horizontally.

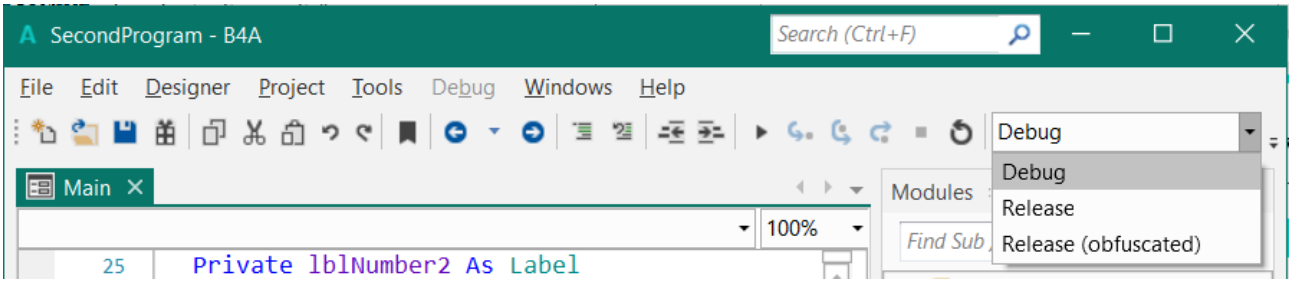


7 Debugging B4A, B4i, B4J

Debugging is an important part when developing.
Debugging is different in B4R than in B4A, B4i and B4J.

7.1 B4A, B4i, B4J

To allow debugging you must activate the debugging mode *Debug* on top of the IDE.



Notes about the debugger (B4A only):

- Breakpoints in the following subs will be ignored: Globals, Process_Globals and Activity_Pause.
- Services - Breakpoints that appear after a call to StartService will be ignored. Breakpoints set in Service_Create and Service_Start will pause the program for up to a specific time (about 12 seconds). This is to prevent the OS from killing the Service.
- Events that fire when the program is paused will be executed. Breakpoints in the event code will be ignored (only when the program is already paused).
- The data sent from the device to the IDE is limited in size. Long strings may be truncated.

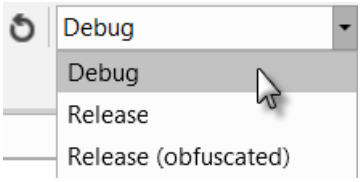
The two major utilities for debugging are:

Breakpoints - You can mark lines of codes as breakpoints. This is done by pressing on the grey area left of the line.
The program will pause when it reaches a breakpoint and will allow you to inspect the current state.

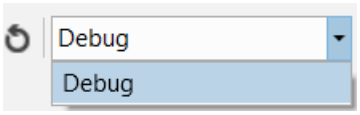
Logging - The Logs tab at the right pane is very useful. It shows messages related to the components life cycle and it can also show messages that are printed with the Log keyword. You should press on the Connect button to connect to the device logs. Note that there is a Filter checkbox. When it is checked you will only see messages related to your program. When it is unchecked you will see all the messages running in the system. If you are encountering an error and do not see any relevant message in the log, it is worth unchecking the filter option and looking for an error message.
Note that the log is maintained by the device. When you connect to a device you will also see previous messages.

7.1.1 Debug mode

The debugging modes are different in the in the products:



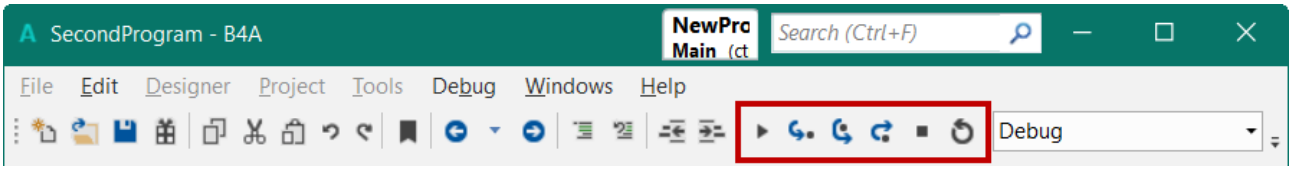
B4A, B4J



B4i, only Debug

7.1.1.1 Debug Toolbar

The debug toolbar is at the right side of the IDE toolbar.



Debug Toolbar:

▶	Run the program	F5	Runs the program, no action in Debug (rapid)
	Step In	F8	Executes the next statement.
	Step Over	F9	Executes a routine without jumping in it.
	Step Out	F10	Finishes executing the rest of a routine.
■	Stop		Stops the program.
	Restart	F11	Restarts the program.

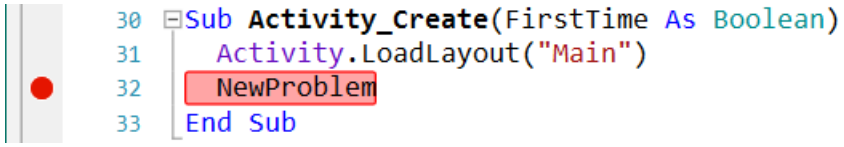
The examples below are shown in the SecondProgram project.

7.1.1.1.1 Run ▶ F5

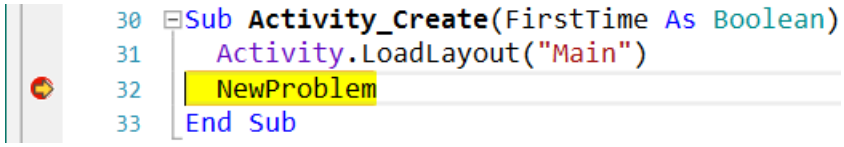
Runs the program.
If the program is stopped at a breakpoint the program runs until the next breakpoint or completes running.

7.1.1.1.2 Step In F8

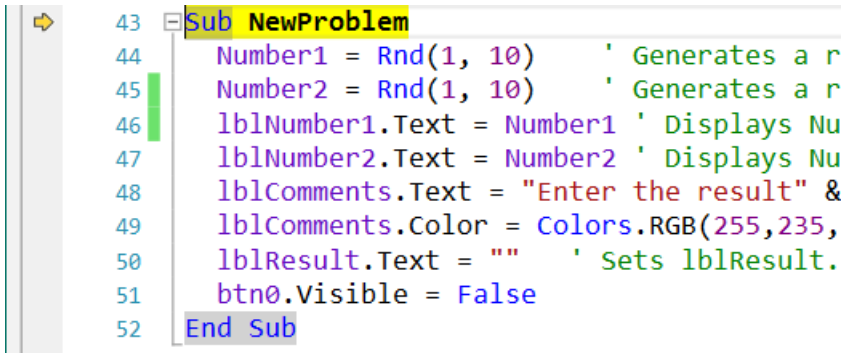
The debugger executes the code step by step.



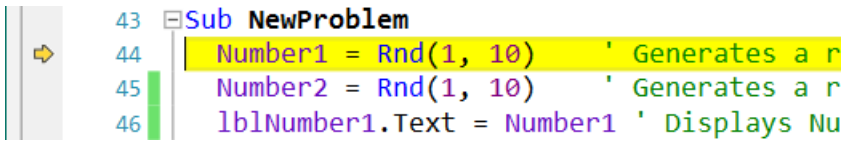
In the SecondProgram project we set a Breakpoint at line 32 `New`.



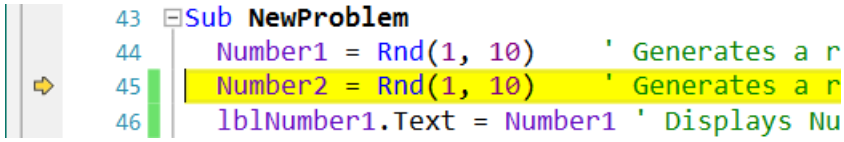
We run the program, it will stop executing at line 32 `New`.



Click on .
The debugger executes the next line, `Sub New` in this case.



Click once more on .
The debugger executes the next line, `Number1 =...`



Click once more on .
The debugger executes the next line, `Number2 =...`

7.1.1.1.3 Step Over F9

If the current line is a sub calling line the debugger executes the code in this subroutine and jumps to the line after the calling line.

```
30 Sub Activity_Create(FirstTime As Boolean)
31   Activity.LoadLayout("Main")
32   NewProblem
33 End Sub
```

In the SecondProgram project we set a Breakpoint at line 32 `New`.

```
30 Sub Activity_Create(FirstTime As Boolean)
31   Activity.LoadLayout("Main")
32   NewProblem
33 End Sub
```

We run the program, it will stop executing at line 32 `New`.

```
30 Sub Activity_Create(FirstTime As Boolean)
31   Activity.LoadLayout("Main")
32   NewProblem
33 End Sub
```

Click on .
The debugger executes the code in `New` and jumpes directly to the next line which is `End Sub` of `Activity_Create`.

7.1.1.1.4 Step Out F10

If the current line is in a subroutine the debugger finishes executing the rest of the code and jumps to the next line after the subs' calling line.

```
30 Sub Activity_Create(FirstTime As Boolean)
31   Activity.LoadLayout("Main")
32   NewProblem
33 End Sub
```

In the SecondProgram project we set a Breakpoint at line 32 `New`.

```
30 Sub Activity_Create(FirstTime As Boolean)
31   Activity.LoadLayout("Main")
32   NewProblem
33 End Sub
```

We run the program, it will stop executing at line 32 `New`.

```
43 Sub NewProblem
44   Number1 = Rnd(1, 10)     ' Generates a r
45   Number2 = Rnd(1, 10)     ' Generates a r
46   lblNumber1.Text = Number1 ' Displays Nu
```

We go step by step with to a line in the subroutine.

```
30 Sub Activity_Create(FirstTime As Boolean)
31   Activity.LoadLayout("Main")
32   NewProblem
33 End Sub
```

Click on .
The debugger executes the rest of the code in the subroutine and jumps to the next line which is `End Sub` of `Activity_Create`.

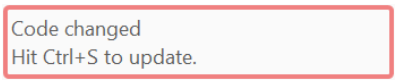
7.1.1.1.5 Stop ■

Stops the program and leaves the Rapid Debugger.

7.1.1.1.6 Restart F11

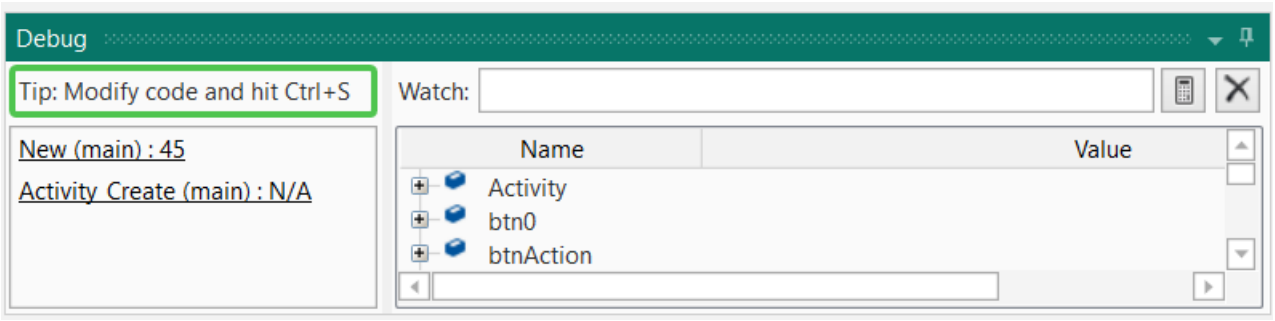
Restarts the program remaining in the Rapid Debugger.
Executes:
B4A Process_Globals, Globals, Activity_Create and reloads the layout.
B4i Process_Globals,
B4J Process_Globals,

This is useful if you changed a layout file.

It is different from 

explained in the next chapter.

7.1.2 Debug window



In the debug window we have, example with the SecondProgram, and a breakpoint in line 45:

7.1.2.1 The status button

Tip: Modify code and hit Ctrl+S

Shows that the program is running, the button border is green.

Code changed
Hit Ctrl+S to update.

When you change the code the button border changes to red.
To update the code click on the button or hit Ctrl + S.

7.1.2.2 The breakpoint window

New (main) : 45
Activity_Create (main) : N/A

The breakpoint window shows where the program has stopped.

```
43 Sub NewProblem
44   Number1 = Rnd(1, 10)     ' Generates
45   Number2 = Rnd(1, 10)     ' Generates
46   lblNumber1.Text = Number1 ' Display
47   lblNumber2.Text = Number2 ' Display
48   lblComments.Text = "Enter the resul
49   lblComments.Color = Colors.RGB(255,
50   lblResult.Text = ""    ' Sets lblRes
51   btn0.Visible = False
52 End Sub
```

New (main) : 45

The program stopped in line 45,
in routine New in the main module.

```
30 Sub Activity_Create(FirstTime As Boolean)
31   Activity.LoadLayout("Main")
32   NewProblem
33 End Sub
```

AppStart (main) : N/A

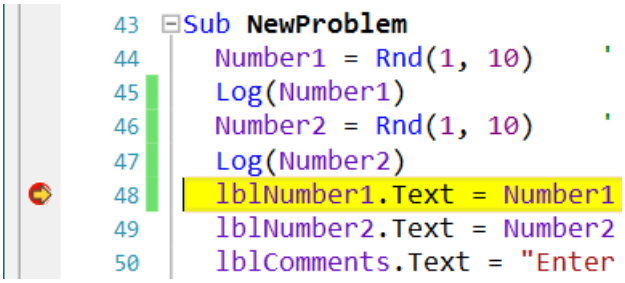
The calling routine is AppStart,
and the calling line is not shown.

NewProblem (main) : 45
Activity_Create (main) : N/A

When you click on one of the lines the cursor jumps to that line.

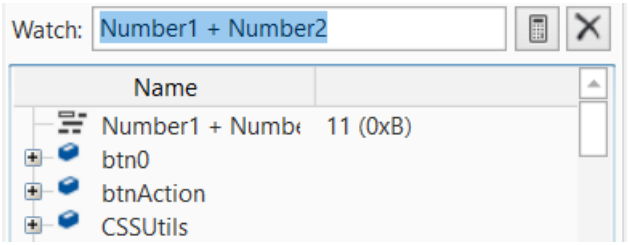
7.1.2.3 The Watch window

The Watch window allows to check more complex functions for testing and debugging.



In the SecondProgram code add two Log lines and set a breakpoint in line 47.

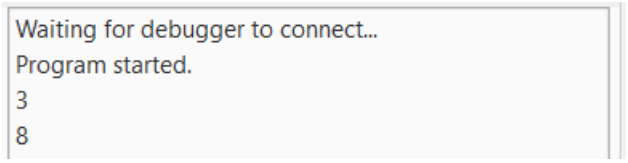
Run the program.



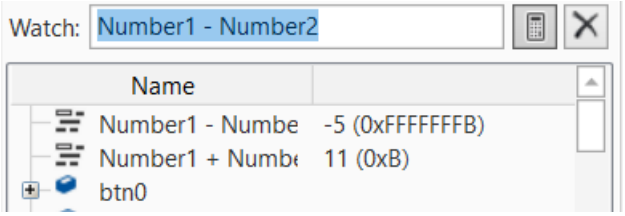
In the Add Watch field enter:

```
Number1 + Number2
```

Click on  to show the result on top of the list.



As we left the two Log lines in the code we still see the values of Number1 and Number2.



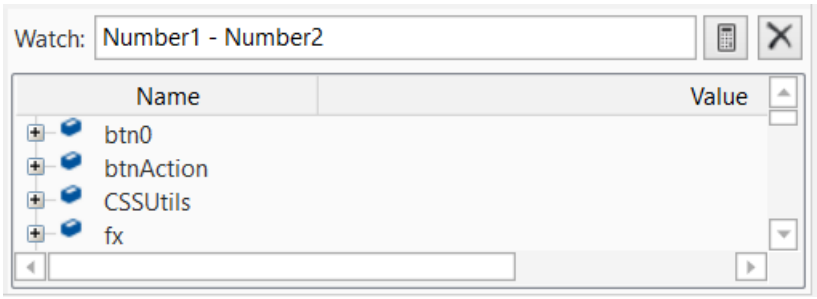
You can enter a new watch line
`Number1 - Number2`
and show it.

Click on  to remove the watch functions. This removes all the functions.

We could, of course, also have done this test with Logs :

```
Log(Number2 + Number2)
Log(Number2 - Number2)
```

7.1.2.4 The object window



Shows all variables and objects in the list ordered by alphabetical order.

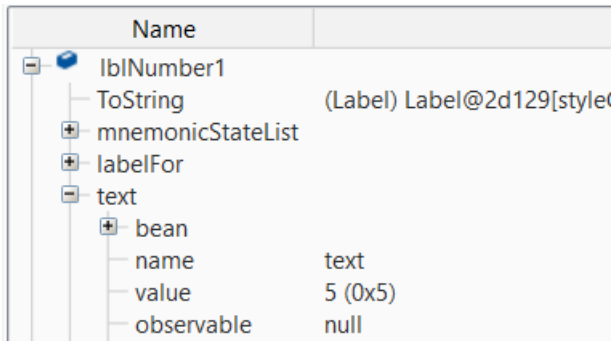
Click on ⊞ to show the details of the object:

Examples:

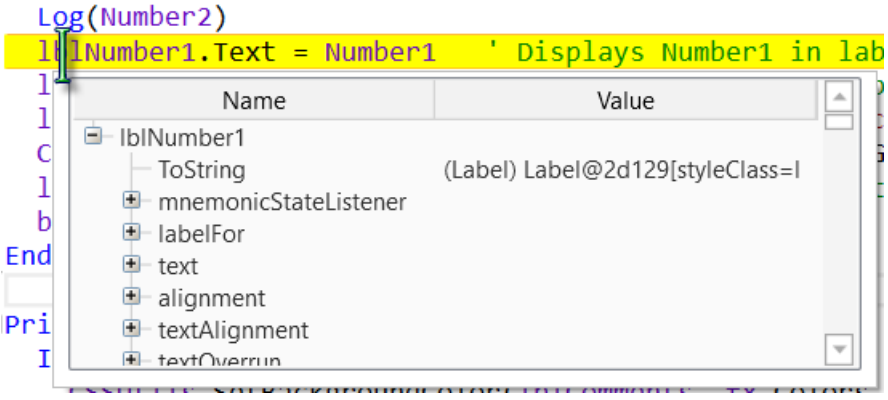
- Number1

Shows the current value (3).

- lblNumber1

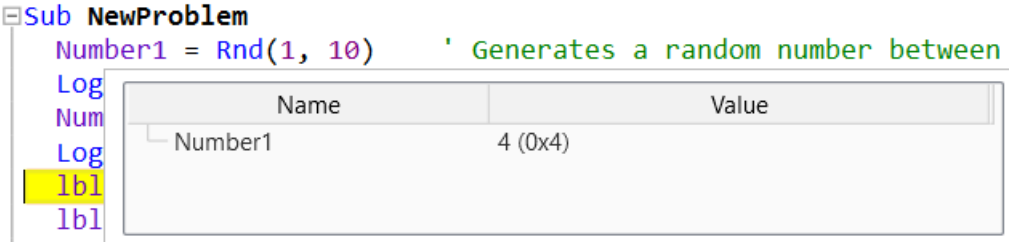


Shows all properties of the object, a Label in the example.



You get the same information when you hover over the object in the code:

lblNumber1

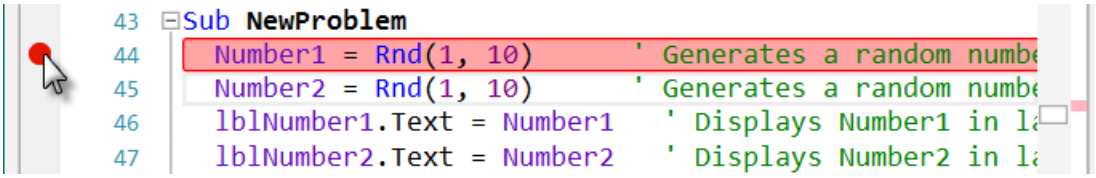


Number1

7.1.3 Breakpoints

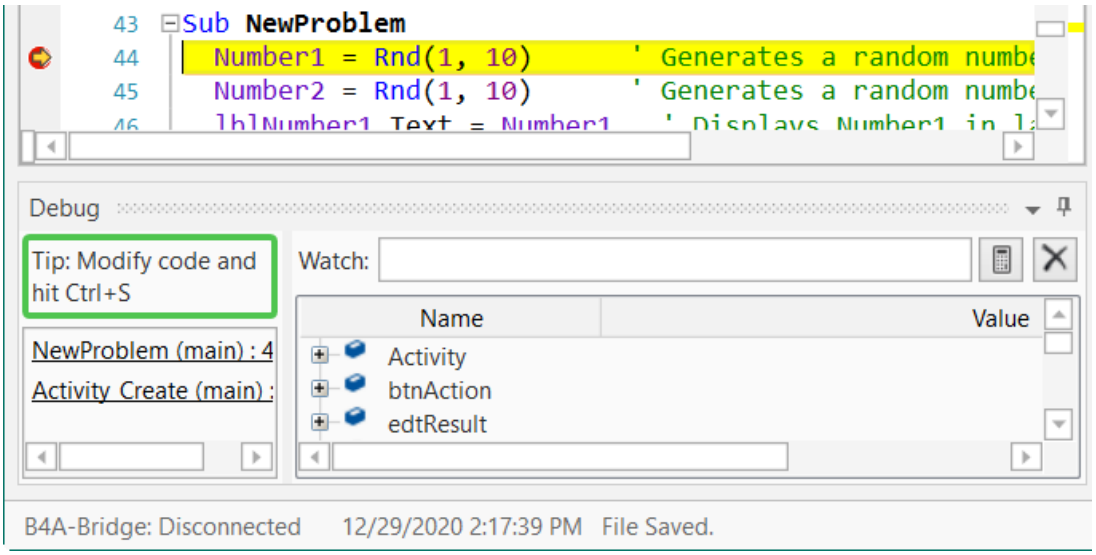
One important feature to make debugging easier are breakpoints. You can set breakpoint almost wherever you want in the code.
Breakpoints, in Process_Globals are ignored.

Clicking on a line in the left margin adds a breakpoint. When the program is running it stops at the first encountered breakpoint.



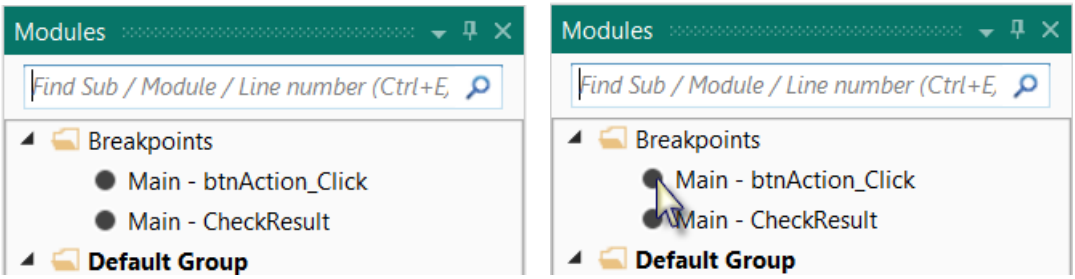
The breakpoints are shown on the right side with a small light red line: 

Run the program, the program stops at the breakpoint and the IDE looks like below. The breakpoint line is highlighted in yellow.



On the bottom of the window you see the debug window.

The Breakpoints are listed in the Modules Tab.

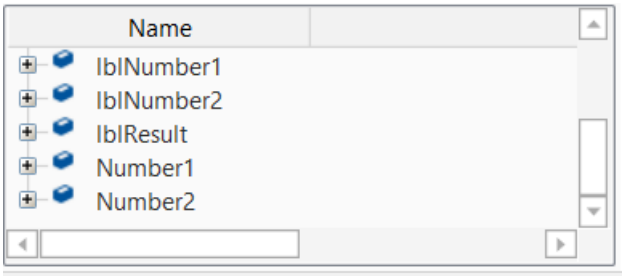


Click on a Breakpoint to jump to its line.

```
42
43 Sub NewProblem
44   Number1 = Rnd(1, 10)
45   Number2 = Rnd(1, 10)
46   lblNumber1.Text = Number1
```

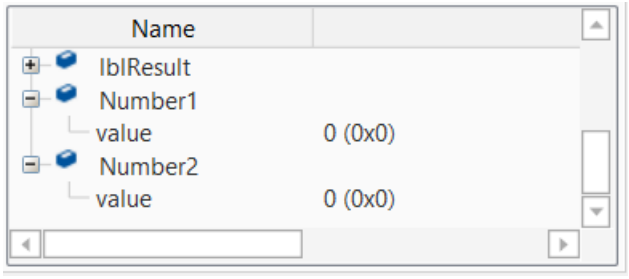
Example with the SecondProgram:

Set a breakpoint in line 44 and run the program.



In the variable window look at Number1 and Number2:

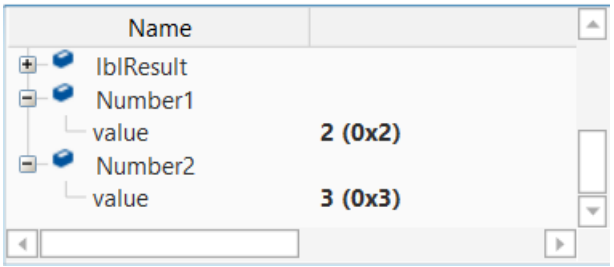
```
43 Sub NewProblem
44   Number1 = Rnd(1, 10)
45   Number2 = Rnd(1, 10)
```



The values are 0 for both.
If you see this ⊞ at the left side of Number1 or Number2 click on it to show the details.
Click on ⤵.
The program jumps to the next line.

Click on ⊞.
You see that the value of Number1 has changed.

```
43 Sub NewProblem
44   Number1 = Rnd(1, 10)
45   Number2 = Rnd(1, 10)
46   lblNumber1.Text = Number1
47   lblNumber2.Text = Number2
```



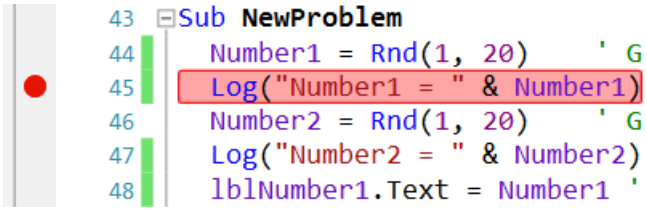
Click on ⤵ again.
The program jumps to the next line.

Click on ⊞.
You see that the value of Number2 has changed.

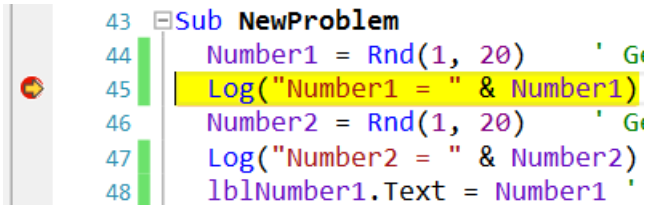
The best way to learn debugging is testing, testing and testing!

7.1.4 With Logs

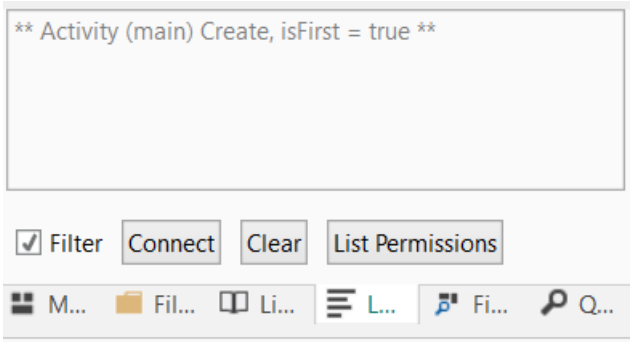
Example with the SecondProgram.



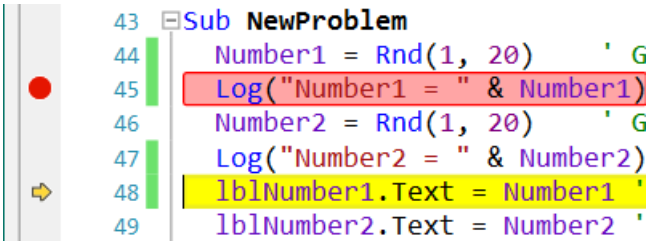
We add the two lines with the Log keyword to display the two numbers in the Log Tab.
We and add a breakpoint in line 44 to watch what happens.



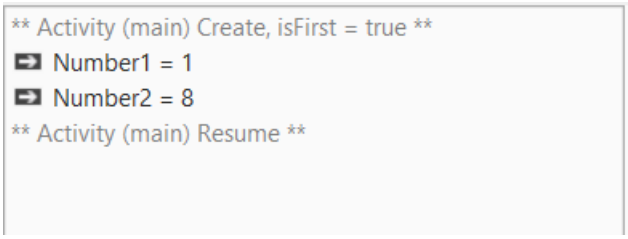
Run the program, it stops at line 44.



In the Log Tab we see at the moment only
`Waiting for debugger to connect...`
and `** Activity (main) Create, isFirst = true **` telling that the program has started.

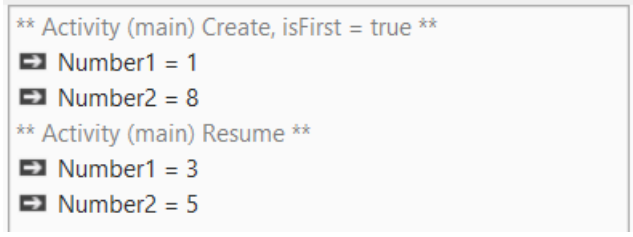


Click four times on  till the program reaches line 48.



In the Log Tab we see the values of the two variables.

Click on  to run to the end.
Nothing new is displayed



When you are using the program the two new values will be shown every time the program runs the New routine.

The arrow  at the beginning of the Log allows to jump to the code line of the Log, see Jump to.

7.1.5 Modifying code in the Debugger

It is possible to change the code in the Debugger and see the new behavior without restarting the program.

Still with SecondProgram and the two Logs and the breakpoint in line 47.

```
43 Sub NewProblem
44   Number1 = Rnd(1, 20)     ' G
45   Log("Number1 = " & Number1)
46   Number2 = Rnd(1, 20)     ' G
47   Log("Number2 = " & Number2)
48   lblNumber1.Text = Number1 '
49   lblNumber2.Text = Number2 '
```

Run the program till it stops at the breakpoint.

```
43 Sub NewProblem
44   Number1 = Rnd(1, 20)     ' G
45   Log("Number1 = " & Number1)
46   Number2 = Rnd(1, 20)     ' G
47   Log("Number2 = " & Number2)
48   lblNumber1.Text = Number1 '
49   lblNumber2.Text = Number2 '
```

In the two lines with `Rnd(1, 10)`
we change the numbers from 10 to 20.

Code changed
Hit Ctrl+S to update.

The status button color has changed confirming a code change.
To rerun the program click on Ctrl + S.

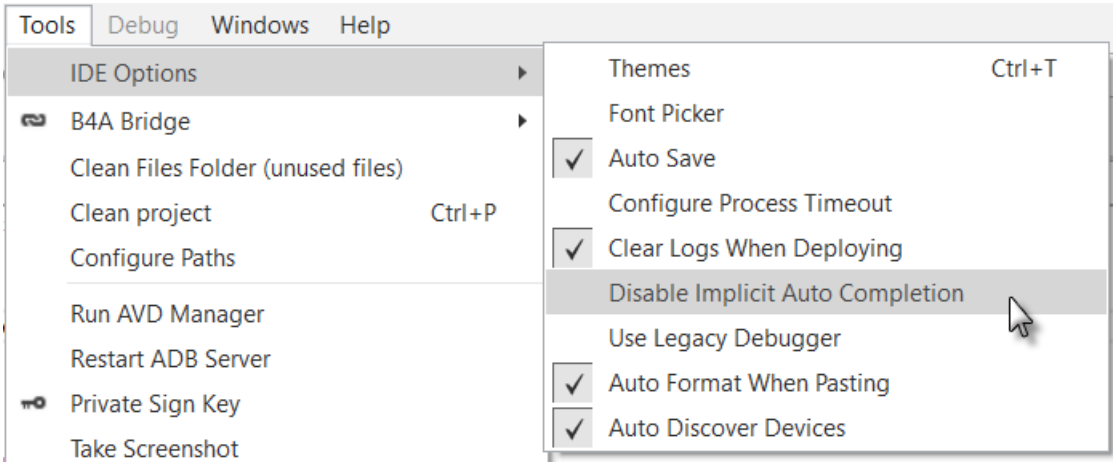
Using the program we see now that the numbers can be between 1 and 19.

```
** Activity (main) Create, isFirst = true **
Number1 = 8
Number2 = 6
** Activity (main) Resume **
** Activity (main) Resume **
Number1 = 4
Number2 = 14
```

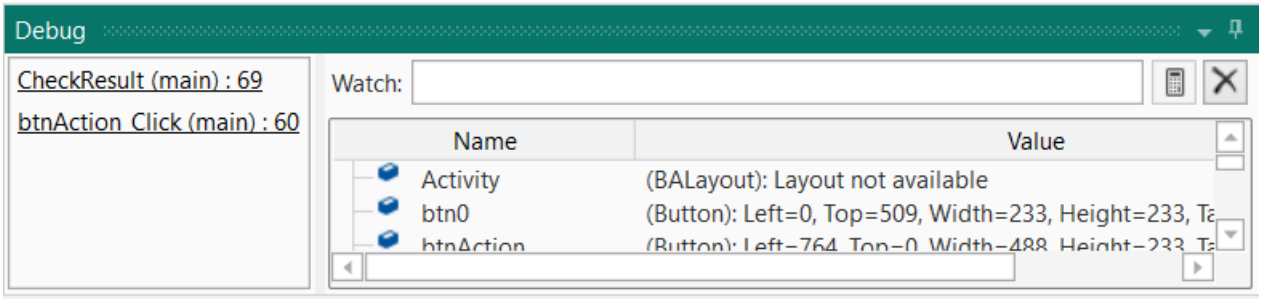
The arrow ⮕ at the beginning of the Log allows to jump to the code line of the Log, see Jump to.

7.1.6 Debug (legacy) mode B4A only

In some cases the legacy Debugger can be useful, you can select it in the Tools menu under IDE options.



Debug(legacy): When this option is selected then the compiled code will contain debugging code.
The debugging code allows the IDE to connect to the program and inspect it while it runs.
The name of the compiled APK file will end with _DEBUG.apk. You should not distribute this apk file as it contains the debugging code which adds a significant overhead.
To distribute files you must select the *Release* or the *Release (obfuscated)* option.



The navigation buttons in the Toolbar are enabled  .
These work similar to the Debug (rapid) mode.

7.2 Debugging B4R

Debugging is an important part when developing.

In B4R there is no Debug mode like in the other B4X languages.

Debugging can only be done with Logs.

The Logs tab in the right pane shows messages related to the components life cycle and it can also show messages that are printed with the Log keyword. You should press on the Connect button to connect to the device logs.

7.2.1 Debug example with the TrafficLight project

In the TrafficLight project I added several Log statements which show the evolution of the program.

When we run the program the Logs is empty.

Then:
We press the button > `State: 0`
The light is set to ON > `Light: 1` red light ON
We release the button > `State: 1`

The logs of the timers:
In TimerGreenRed_Tick change from red to green:
> `TimerGreenRed_Tick LightGreen`
We set the light to yellow and enable TimerYellow:
> `TimerGreenRed_Tick LightYellow`
In TimerYellow_Tick we set yellow OFF and red ON
> `TimerYellow_Tick LightRed`
End of first cycle
Begin of next cycle: > `TimerGreenRed_Tick LightGreen`

We press the button > `State: 0`
The light is set to OFF > `Light: 0`
We release the button > `State: 1`