B4X Booklets

B4XPages

Cross-platform projects

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 CrossPlatformSourceCode folder.

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

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

B4XPages Cross-platform projects
B4X CustomViews
B4X Graphics
B4X XUI B4**X** **U**ser **I**nterface
B4X SQLite Database
B4X JavaObject NativeObject

B4R Example Projects

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.

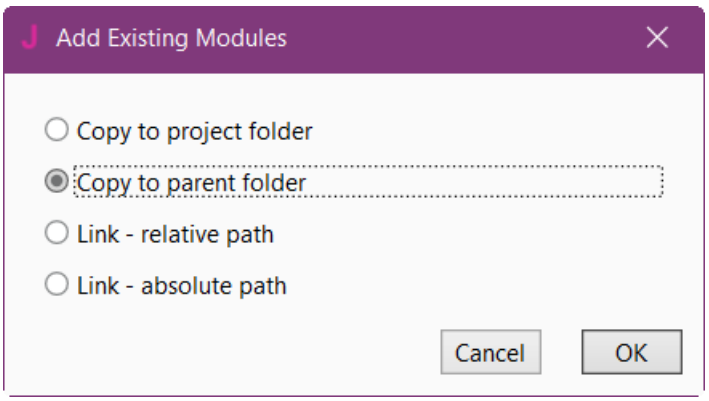
2 First steps

This guide shows the best practices for developing cross-platform projects for B4A, B4i and B4J with the goal to use as much as possible the same code.

What does cross-platform project mean?
The main goal is to have most of the code the same for all three platforms.

To develop cross-platform projects, some simple 'rules' should be followed.

1. Use the B4XPages cross-platform framework to manage pages.
2. Use as much as possible B4XViews, included in the standard *XUI* library.
   These are oversets of the platform specific standard views.
3. Use as much as possible views from the *XUI Views* library.
   These are cross-platform custom views.
4. Use as much as possible XUI CustomViews.
5. When you add a common module:
   a. Copy it to the parent folder if it is a new module for the project.
   b. Link it with a relative path if it is already in the parent folder.
   c. Or link it with an absolute path for modules valid for other projects.
   d. Do not copy it to the project folder, or only if it is specific to the platform.



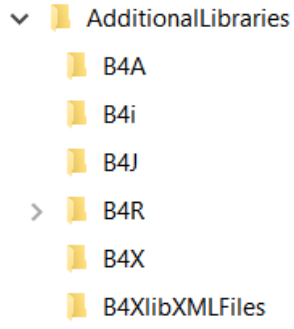
That way, when you modify a common module and save it, the code in the other IDEs is automatically updated.

You should decide directly at the beginning which platforms you want use.
Even for single platform projects it is useful to follow the simple rules above. Because the B4XViews have some properties not exposed directly in the corresponding platform and are cross-platform.

For development, I prefer to start with B4J, because no device needed for testing, and then adapt for the other platforms.

2.1 AdditionalLibraries folder structure

You should setup the AdditionalLibraries folder structure like below, if not yet done.

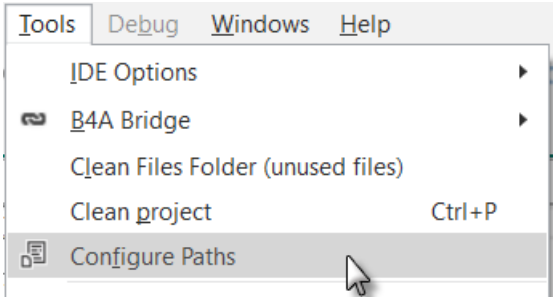


Folder for B4A additional libraries.
Folder for B4i additional libraries.
Folder for B4J additional libraries.
Folder for B4R additional libraries.
Folder for B4X libraries (xxx.b4xlib files).
Folder for B4X libraries XML files.

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

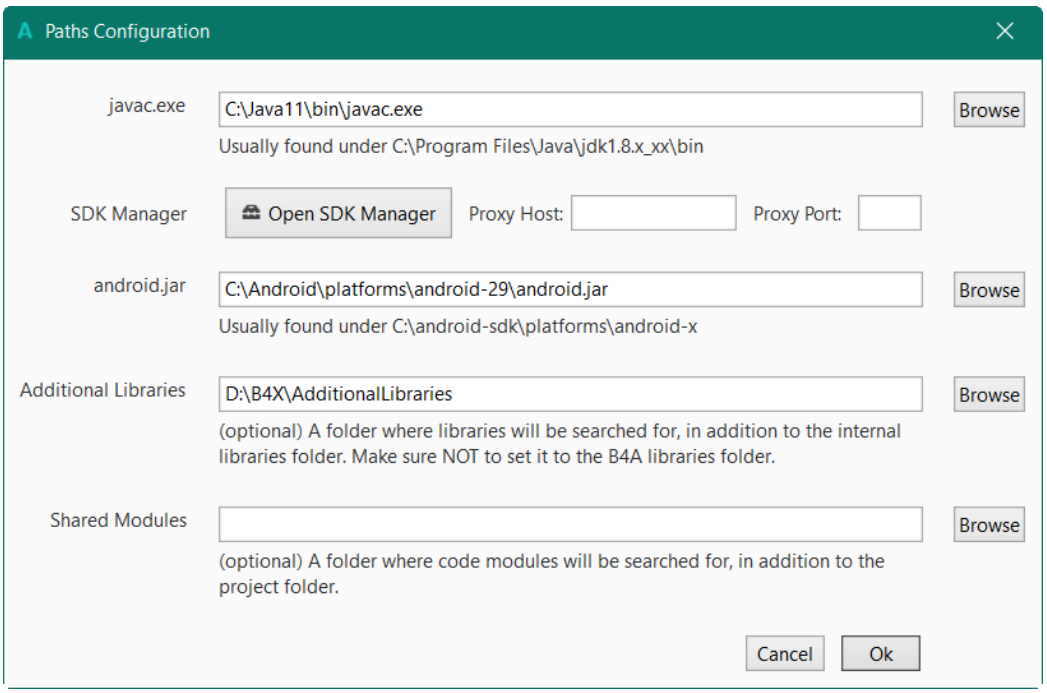
When you install a new version of a B4X platform, all standard libraries are automatically updated, but the additional libraries are not included. The advantage of the special folder is that you do not need to care about them because this folder is not affected when you install the new version of B4X.

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



To setup the special additional libraries folder, click in the IDE menu on Tools / Configure Paths.

In my case D:\B4X\AdditionalLibraries.
The subfolders are automatically considered by the compiler.



In my system, I added a B4XlibXMLFiles folder for XML help files.
The standard and additional libraries have an XML file.
B4X Libraries 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.

To get xml files from b4xlib libraries you can use this utility written by Erel:
[Tool] b4xlib - XML generation.

3 B4XPages

B4XPages is a library that serves two purposes:

1. Make it simple to develop B4A apps by solving almost all of the challenges involved with Android complex activities life cycle.
   B4XPages makes B4A behave more similar to B4J and B4i where the new "B4XPage" element is a regular object that is never paused, never destroyed, can be accessed from anywhere and easy to work with.
2. Provide a cross platform layer above the navigation related APIs.

Before we start:

1. You are not forced to use B4XPages. All the current features behave exactly as before.
2. **It does have some limitations. One notable limitation is that in B4A, the activity that holds all the pages should be locked to a single orientation either portrait or landscape.**
3. It is supported by the latest versions of B4A 10.0+, B4J 8.50+ and B4i 6.80+.

3.1 What exactly does it solve?

B4XPages makes many things simple and even trivial.
As Android developers, we are dealing with these challenges for many years now so it might take us a while to understand how simple things can be.

The advantages of the cross-platform layer are clear. B4XPages hides many of the differences between B4A activities, B4i pages and B4J forms. With B4XPages it is trivial to create a multiple "pages" app where all of the code is shared (except of the template code which is just pasted to the main module).
With activities, it requires creating an activity + shared class + page / form module for each page. A lot of work.

The three most important things regarding the B4XPage classes are:

1. The page classes are 100% regular classes. They do not have a special life cycle and you can do whatever you like with them. There are some B4XPages events, but they do not affect the state of the class itself. This is not the case with activity modules.
2. The page classes are never paused. Nothing special happens when a page is no longer visible or when the app moves to the background. Eventually of course, the whole process will be killed when the app is in the background.
3. The page classes are never destroyed separately. The class global variables and views state will never be reset (until the process is killed).

Even if you are only interested in B4A development, B4XPages can help you a lot. I will list here all kinds of challenges that become simple with B4XPages.

Make sure to use B4A v10.0+, B4J v8.50+, B4i v6.80+.

The 3 points above make many things simple. Some of them are:

1. Events are never missed or queued.
2. Sleep calls are never cancelled. For example, no longer do we need to restart animations in Activity_Resume.
3. The UI state is kept as long as the process lives.
4. In most cases we do not need to use the starter service, but we leave it. Do not remove it.
5. We can directly call public methods of other pages. No need to use CallSub or CallSubDelayed.
6. We can directly access public global variables of other pages.
7. We can directly access and manipulate views of other pages.
8. We can decide whether to create the layouts immediately when a page is created (B4XPages.AddPageAndCreate).
9. We can move views between pages.
10. No need to worry from cases where CallSubDelayed starts the previous activity unexpectedly (usually happens with HttpJobs).
11. We can use the same page class to create many page instances.
12. A single place with a single and simple behavior for the global variables.
13. No need to think what should be initialized when FirstTime = True and what needs to be initialized every time.
14. No need to handle cases where a different activity, other than Main, is started first.
15. Better control over the pages stack as it is implemented in B4X code.
16. Automatic handling of the up indicator. No need to use AppCompat library for this.
17. Quite simple and flexible way to handle the back key.
18. Background / Foreground events that are raised in all pages when the app moves to the background and to the foreground (not so simple to get with activities and required in many cases).
19. No distinguish between classes with "activity context" which must be declared in Globals to other classes that can be declared in Process_Global.
20. No need to split the implementation between a stateless UI layer and a stateful non-UI layer.

With that said, no one is forced to switch to B4XPages. Everything will continue to work exactly as before. It is too soon to rush and convert large working projects.
B4XPages also has some limitations, especially regarding to the locked orientation (in B4A only).

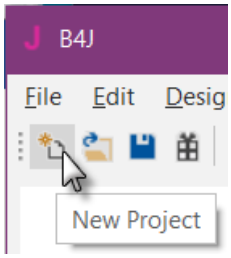
3.2 What is a B4XPage?

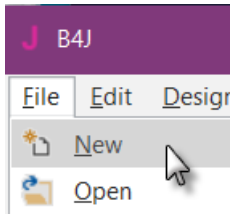
It differs between the three platforms:
B4J – Form
B4A - Panel in a single activity.
B4i - Page in a single NavBarController

3.3 New cross-platform B4XPages project

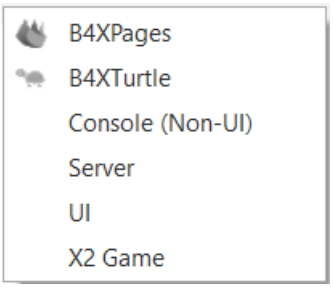
Tip: Begin a new cross-platform project with B4J, it is easier especially for debugging. No need for a device or an emulator!

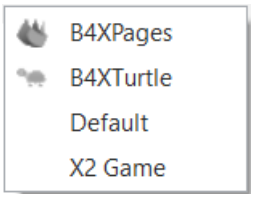
Open the IDE, everything is empty!

Click either on 

or on 

Then you are asked what kind of project you want to create:





B4J **B4A / B4i**

B4XPages	Cross-platform project.
B4XTurtle	B4XTurtle project.
Default	B4A or B4i standard project.
X2Game	X2Game project.
Server	B4J Server project
UI	B4J Standard user interface project.

.

When you select one of the above project types you get a default project template.

Select B4XPages projects.

This form is shown:

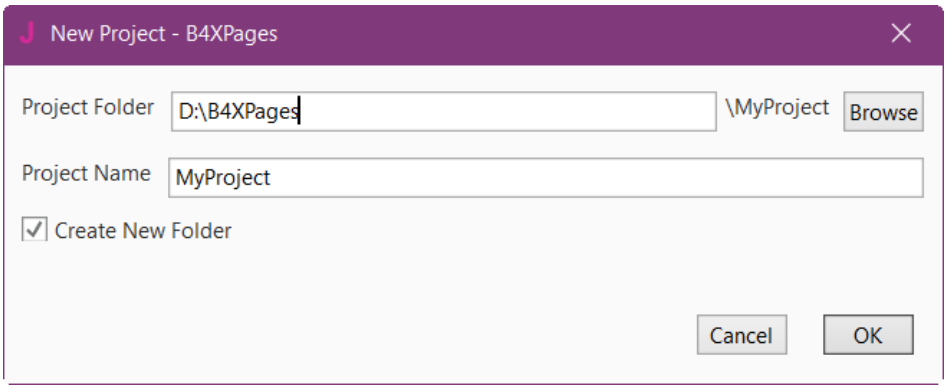
By default, the cursor is positioned in the Project Name field, this is by purpose.

Enter the project name, MyProject in this example.

You see that the project name is automatically added at the end of the Project Folder field.

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

I use *B4XPages* as the generic B4XPages projects folder, this name is memorized for future projects, but you can change it at any time.
Keep ☑ Create New Folder checked because we create a new project.



Click on OK.

Three projects with everything you need are created, one for each platform. Use it even if you are only interested in a single platform.

You see the B4J B4XPages code template.

```
#Region Shared Files
#CustomBuildAction: folders ready, %WINDIR%\System32\Robocopy.exe,"..\..\Shared Files"
"..\Files"
'Ctrl + click to sync files:
ide://run?file=%WINDIR%\System32\Robocopy.exe&args=..\..\Shared+Files&args=..\Files&Fil
esSync=True
#End Region

'Ctrl + click to export as zip: ide://run?File=%B4X%\Zipper.jar&Args=Project.zip

Sub Class_Globals
   Private Root As B4XView
   Private xui As XUI
End Sub

Public Sub Initialize

End Sub

'This event will be called once, before the page becomes visible.
Private Sub B4XPage_Created (Root1 As B4XView)
   Root = Root1
   Root.LoadLayout("MainPage")
End Sub

'You can see the list of page related events in the B4XPagesManager object. The event
name is B4XPage.

Sub Button1_Click
   xui.MsgboxAsync("Hello world!", "B4X")
End Sub
```

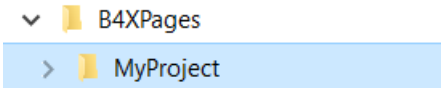
It is worth to add the three lines described in chapter Launch the IDE from another platform.

3.4 B4XPages project structure

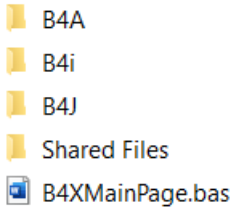
When you create a new B4XPages project the structure below is automatically created.

Example with MyProject.

The *MyProject* project folder is in the *B4XPages* generic folder.



Insides of the *MyProject* folder we have the structure below:

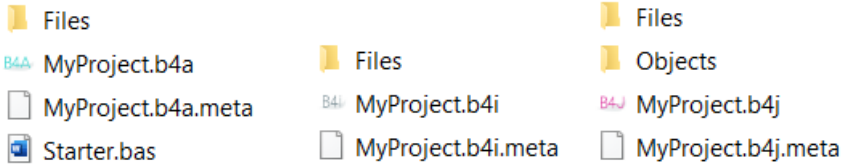


One folder for each platform.

A fourth folder for Shared Files.

And the B4XMainPage.bas file.

Each B4A, B4i and B4J platform folder contains, the usual files and the Files and Objects folders.



B4A **B4i** **B4J**

The Files folder is the 'standard' files folder containing mainly the layout files and maybe other platform dependent files.

The Objects folders are added after compilation, as usual.

The Shared Files folder contains all the files shared between the platforms like image or database files.

Note:
All the files in the Shared Files folder are copied to the Files folders of the different platforms during compilation.
If you make any change in these files, make the changes only in the files in the Shared Files folder, the IDE updates them automatically in the platform specific File folders.

3.5 Code templates

The code templates for all three platforms have following modules:

- **Main** the standard platform specific Main module.
  It contains platform specific code to interface with the B4XMainPage.
  Do not modify any code in the Main module !!!

- **B4XMainPage** this is the mandatory cross-platform main page.
  It contains all cross-platform code and manages the different pages if there are more than one.

When you run the IDE it shows, by default, directly the code in the B4XMainPage module.

3.6 B4XMainPage Code template

B4xMainPage Code template:

```
#Region Shared Files
#CustomBuildAction: folders ready, %WINDIR%\System32\Robocopy.exe,"..\..\Shared Files"
"..\Files"
'Ctrl + click to sync files:
ide://run?file=%WINDIR%\System32\Robocopy.exe&args=..\..\Shared+Files&args=..\Files&Fil
esSync=True
#End Region

'Ctrl + click to export as zip: ide://run?File=%B4X%\Zipper.jar&Args=Project.zip

Sub Class_Globals
   Private Root As B4XView
   Private xui As XUI
End Sub

Public Sub Initialize

End Sub

'This event will be called once, before the page becomes visible.
Private Sub B4XPage_Created (Root1 As B4XView)
   Root = Root1
   Root.LoadLayout("MainPage")
End Sub

'You can see the list of page related events in the B4XPagesManager object. The event
name is B4XPage.

Sub Button1_Click
   xui.MsgboxAsync("Hello world!", "B4X")
End Sub
```

This line:
`#CustomBuildAction: folders ready, %WINDIR%\System32\Robocopy.exe,"..\..\Shared Files" "..\Files"`
Copies the all the files contained in the Shared Files folder to the product specific Files folder.

This line allows, with Ctrl + click, to synchronize the Shared Files folder with the product specific Files folder.
`'Ctrl + click to sync files: ide://run?file=%WINDIR%\System32\Robocopy.exe&args=..\..\Shared+Files&args=..\Files&FilesSync=True`

This line allows, with Ctrl + click, to generate a zip file of the whole B4XPages project.
`'Ctrl + click to export as zip: ide://run?File=%B4X%\Zipper.jar&Args=Project.zip`

You may modify this line with your project name.
Replace `Project` by your project name, `MyProject` in the example.
`'Ctrl + click to export as zip: ide://run?File=%B4X%\Zipper.jar&Args=MyProject.zip`

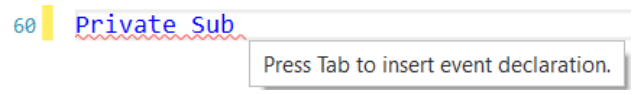
The Button1_Click event routine is only shown as an example, you should remove it.

3.6.1 B4XPage events

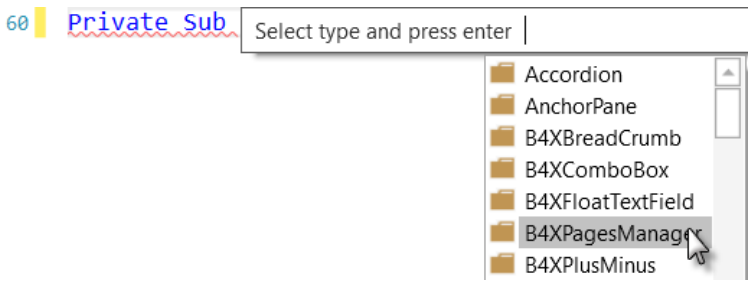
Summary of possible B4XPage event routines which can be used in B4XMainPage.

B4XPage_Created - Called once when the page is created. This will happen before the page becomes visible or after a call to B4XPages.AddPageAndCreate.
B4XPage_Appear - Called whenever the page becomes visible.
B4XPage_Disappear - Called whenever a visible page disappears.
B4XPage_Background - Called when the app is moved to the background. This event will be raised in all pages that implement this sub, not just the top event. This is a good place to save anything that needs to be save as the process might be killed later. Note that in B4J it is raised when the last page is closed.
B4XPage_Foreground - Called when the app moved to the foreground.
B4XPage_Resize (B4J / B4i) - Called when the page is resized.
B4XPage_CloseRequest (B4J / B4A) - In B4A it is called when the user clicks on the back key or on the up indicator. In B4J it is called when the user clicks on the close button.
B4XPage_MenuClick - Called when a menu item or BarButton in B4i is clicked.
B4XPage_KeyboardStateChanged (B4i) - Called when the keyboard state changes.
B4XPage_IconifiedChanged (B4J) - Called when a page is minimized or restored.
B4XPage_PermissionResult (B4A) - Raised after a call to rp.CheckAndRequest.
rp = RuntimePermissions, the B4A RuntimePermissions library.

To define these routines, use the IDE internal method:

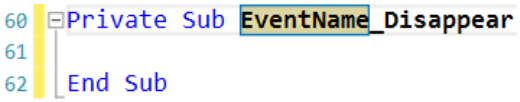


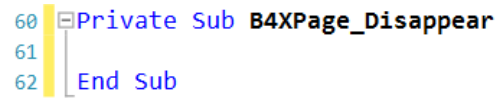
Write Private Sub or Sub plus a space and you will see this, and Press Tab:



Select B4XPagesManager.

Select the event routine you want and enter *B4XPage* for the EventName.
The displayed event list depends on the platform.





3.6.1.1 B4XPage_CloseRequest

In B4A and B4J you can handle the CloseRequest event and cancel the request if needed.

Example:

```
'Return True to close, False to cancel
Private Sub B4XPage_CloseRequest As ResumableSub
   Dim sf As Object = xui.Msgbox2Async("Close?", "Title", "Yes", "Cancel", "No", Null)
   Wait For (sf) Msgbox_Result (Result As Int)
   If Result = xui.DialogResponse_Positive Then
       Return True
   End If
   Return False
End Sub
```

3.6.2 B4XPage methods

- **AddPage** (Id As String, B4XPage As Object) Adds a new page.

```
Private Page1 As B4XPage1
Page1.Initialize
B4XPages.AddPage("Page 1", Page1)
```

Each page is identified with a case insensitive string id. `"Page 1` in the example.
The main page id is "MainPage".

- **ShowPage** (Id As String) Shows a page. In cases where the page is already in the stack

```
B4XPages.ShowPage("Page 1")
```

B4A - The page will be moved to the top of the stack.
B4i - Pages above this page will be removed.
B4J - Not relevant as multiple pages can be displayed.

- **ClosePage** (B4XPage As Object) Closes the page. The page is not destroyed.

```
B4XPages.ClosePage(Me)
```

Use `Me` in the current Page.
In B4i, only the top page can be closed.

- **SetTitle** (B4XPage As Object, Title As Object) Sets the page title. Can be CSBuilder in B4A.

```
B4XPages.SetTitle(Me, "My title")
```

Use `Me` in the current Page.
The first parameter is the Page object, not the Page id!

- **GetPage** (Id As String) Returns the page instance.

```
Private MP As B4XMainPage
MP =B4XPages.GetPage("MainPage")
```

Another example:

```
Private Page2 As B4XPage2
Page2 = B4XPages.GetPage("Page 2")
B4XPages.SetTitle(Page2, "This is page 2"
Page2.btnPage3.Text = "Page 3"
```

You should cast it to the correct type.

- **GetPageId** (B4XPage As Object) Returns the page id from the page object.

```
Private Page3ID As String
Page3ID = B4XPages.GetPageId(Page3)
```

Returns : String

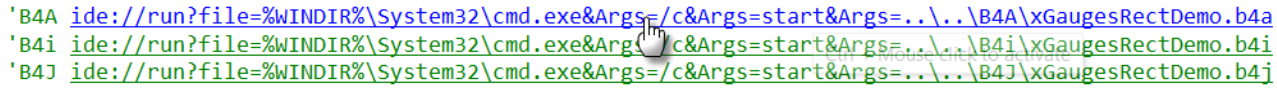
3.6.3 Launch the IDE of another platform

You can add the three lines below on top of the B4XMainPage module if you want to open the IDE of another platform directly from one project.

```
'B4A ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4A\ProjectName.b4a
'B4i ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4i\ProjectName.b4i
'B4J ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4J\ProjectName.b4j
```

Replace ProjectName by your project name.

Then, when you hover over these lines with the Ctrl key pressed the line is highlighted in blue. Click on it to run the selected IDE.



If you have the lines above, you could replace them by these lines:

```
'B4A ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4A\%PROJECT_NAME%.b4a
'B4i ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4i\%PROJECT_NAME%.b4i
'B4J ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4J\%PROJECT_NAME%.b4j
```

Replace `\ProjectName.` by `\%PROJECT_NAME%.` at the end.
`%PROJECT_NAME%` is a reserved key word which represents the project name.

4 Compatibilities B4A B4i B4J XUI

A list of current objects, which can be almost the same, or having different names with similar functionalities and / or the B4XView equivalent or having an equivalent CustomView.

B4J	B4A	B4i	XUI	CustomView
Button	Button	Button	B4XView	---
Canvas	Canvas	Canvas	B4XCanvas	---
CheckBox	CheckBox	Switch	---	B4XSwitch
ComboBox	Spinner	Picker	---	B4XComboBox
ImageView	ImageView	ImageView	B4XView	---
Image	Bitmap	Bitmap	B4XBitmap	---
Label	Label	Label	B4XView	---
ListView	ListView	---	---	xCustomListView
Pane	Panel	Panel	B4XView	
ProgressBar ProgressIndicator	ProgressBar	ProgressView	B4XView	---
ScrollPane	ScrollView HorizontalScrollView	ScrollView	B4XView	---
Slider	SeekBar	Slider	---	B4XSeekbar
TextField	EditText	TextField	B4XView	---
WebView	WebView	WebView	---	---

xCustomListView is a standard library.
B4XSwitch, B4XComboBox and B4XSeekbar are included in the XUI Views.b4xlib library, which is also a standard library.

5 Copy layouts, move views

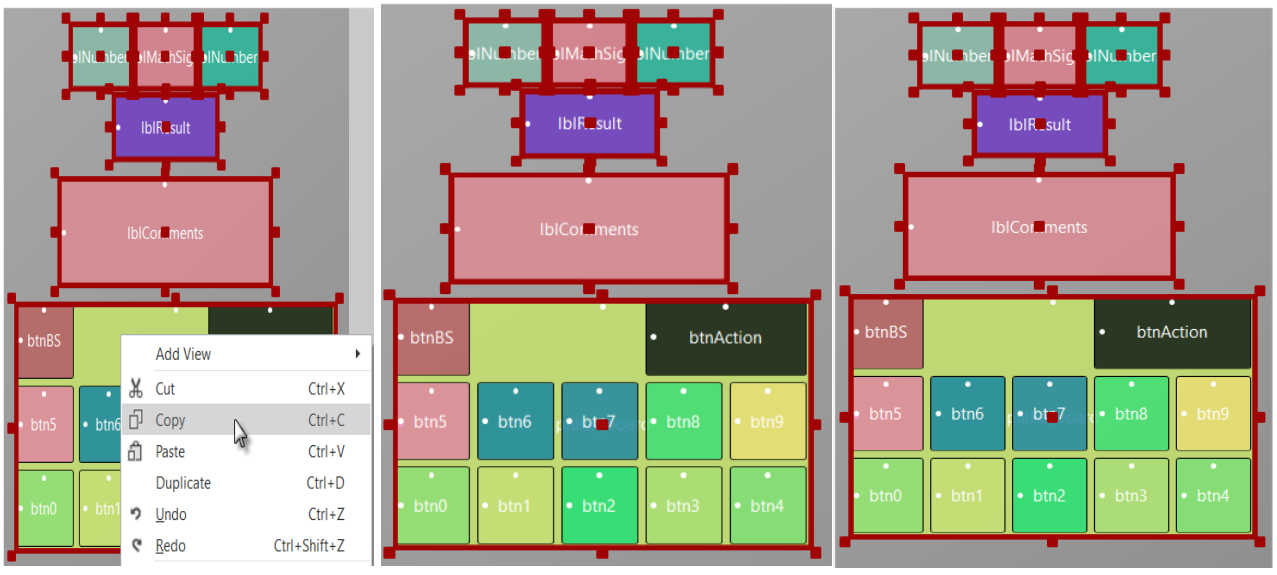
5.1 Copy layouts

You can copy layouts from one platform to another one.

The following types of views can be copied between the platforms:

- CheckBox / Switch
- Button
- Label
- Panel / Pane
- ToggleButton
- SeekBar / Slider
- Spinner / ComboBox / ChoiceBox (B4A and B4i only)
  Better to use B4XComboBox from XUI Views.
- ImageView
- ProgressBar / ProgressView / ProgressIndicator
- RadioButton
- EditText / TextView / TextField / TextArea
- WebView
- Custom Views

Select the views to copy in the Designer of one platform and paste them into the Designer in another platform.
Example from the B4X_SecondProgram project.



Source B4A Designer. Copy in the B4J Designer Copy in the B4i Designer.

It is as simple as this !!!
Maybe you need some adjustments depending on the original dimensions or properties like colors.

Note: If you copy CustomViews, make sure that you have checked the needed libraries before you copy. Otherwise you will lose them!

5.2 Move views between pages

You can move views between pages with the code below.

```
Dim v As B4XView = SomeOtherPage.View1
v.RemoveFromParent
Root.Add(v, 10dip, 10dip, 100dip, 100dip)
```

6 Access objects from another Page

Sometimes it is useful to access an object from another page.

This can easily be done with the code below:

The example below is demonstrated in the B4XPagesNavBar project.

The object must be declared as Public or with Dim in its module, Page2 in the example.

```
Public lblTest As B4XView     'used to show how to access an object in another Page.
```

The `lblTest` object is accessed from Page3 with this code:

```
'to show how to access an object in another Page.
Public Page2 As B4XPage2 = B4XPages.GetPage("Page 2")
Page2.lblTest.Text = "Page 3 was displayed."
```

In `B4XPages.GetPage("Page 2")` the parameter in quotes is the page ID, the first parameter, you gave when you initialized the page in the B4XMainPage module:

```
B4XPages.AddPage("Page 2", Page2)
```

7 B4X Templates

This is a very simple and useful feature. When you create a new project, the project is created based on the chosen template.
It is especially important for cross platform projects where a new project creates three platform specific projects, with the recommended structure and shared code.

The templates are simple zip files, with the b4xtemplate extension. The template files should be in the internal or additional libraries folders.

Files with $APPNAME$ in their name will be replaced with the project name.

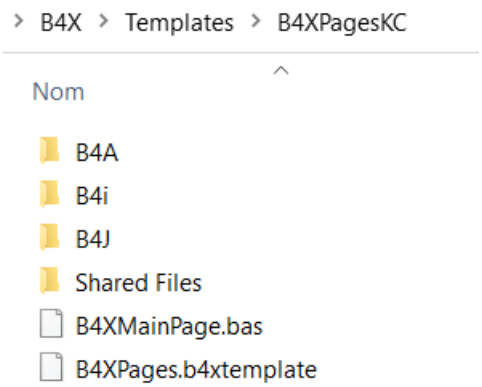
Two templates are currently shipped with the B4X platforms, the files are in the internal libraries folders:

- B4XPages.b4xtemplate
- B4XTurtle.b4xtemplate

To be able to zip a cross-platform project with the correct name and to open the IDE of another platform directly from the B4XMainPage module I made my own B4XPages Template with the modifications below, to do this:

Copy the B4XPages.b4xtemplate file, from the internal libraries folder, into a new folder and unzip it.

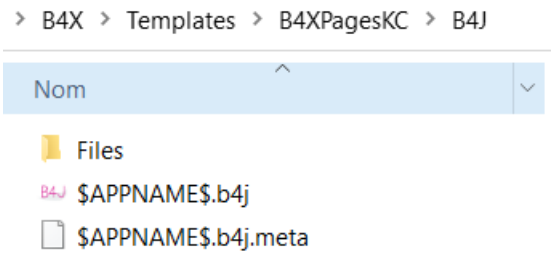
You get the structure below:



One folder for each platform.

The Shared Files folder.
The B4XMainPage.bas file.
The original zip file.

Open the B4J folder, you get the structure below:



The Files folder.
The .b4j file.
The .b4j.meta file.

You see that the name of the *.b4j* file and the *.b4j.meta* file is the reserved word: $APPNAME$.
The Files folder contains the MainPaje.bjl file, the B4J default layout.

Open the B4J project.

You get the code below on top of the B4XMainPage module:

```
#Region Shared Files

'Ctrl + click to export as zip: ide://run?File=%B4X%\Zipper.jar&Args=ProjectName.zip
```

I replaced jar&Args=ProjectName.zip by jar&Args=\%PROJECT_NAME%.zip.

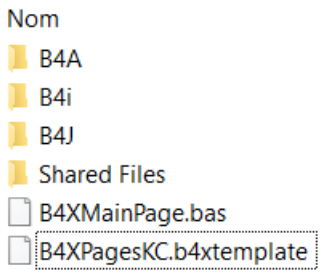
And I added the three lines below:

```
'B4A ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4A\%PROJECT_NAME%.b4a
'B4i ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4i\%PROJECT_NAME%.b4i
'B4J ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4J\%PROJECT_NAME%.b4j
```

These lines allow you to lunch the project of another platform directly from the B4XMainPage.

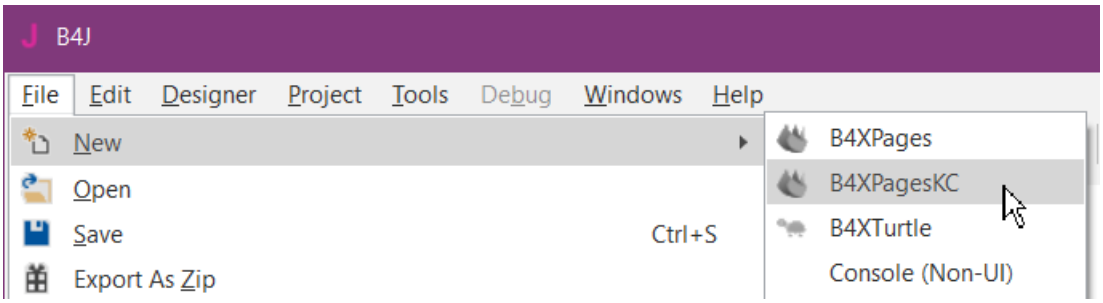
Save the project and leave the IDE.

Zip the entire structure with a new name, *B4XPagesKC* in my case, and with the b4xtemplate extension.

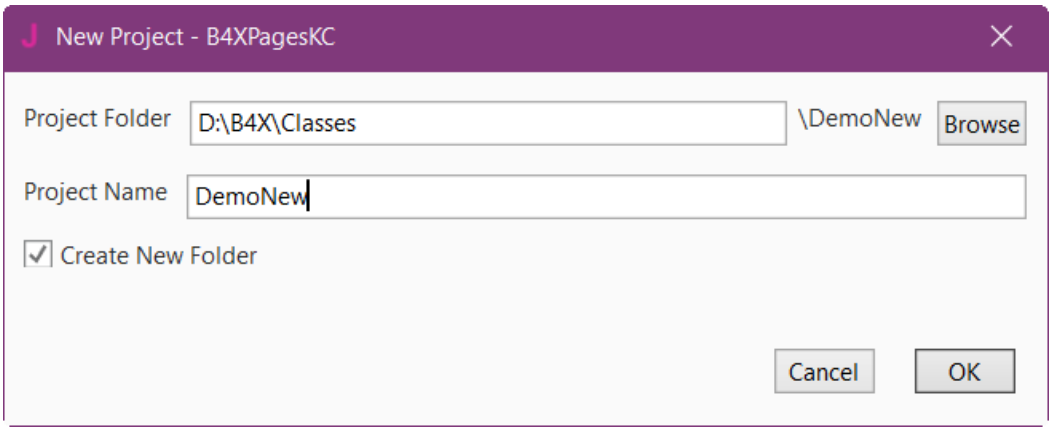


Then copy the B4XPagesKC.b4xtemplate file into the AdditionalLibraries\B4X folder.
The same folder where you copy B4X Library files.

When you now open the IDE, select New in the File menu, you will see the new B4XPagesKC template.



Create a new project, I used the name DemoNew as an example:



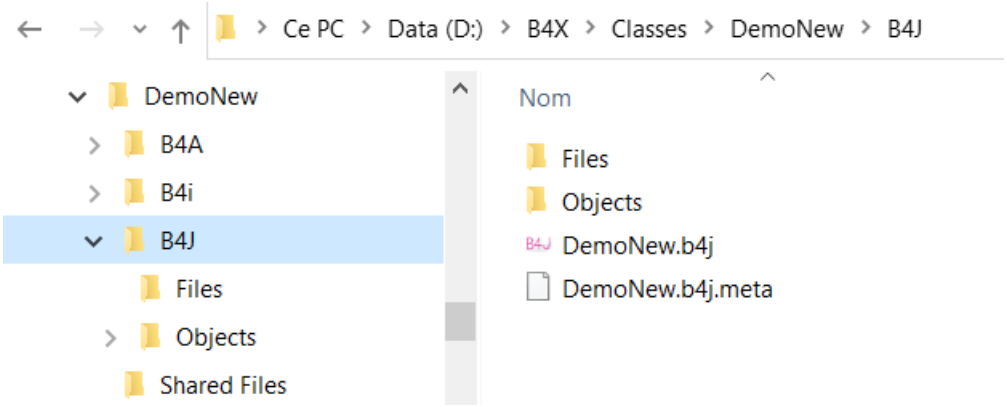
You get the new template with the modifications.

```
#Region Shared Files

'Ctrl + click to export as zip: ide://run?File=%B4X%\Zipper.jar&Args=%PROJECT_NAME%.zip

'B4A ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4A\%PROJECT_NAME%.b4a
'B4i ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4i\%PROJECT_NAME%.b4i
'B4J ide://run?file=%WINDIR%\System32\cmd.exe&Args=/c&Args=start&Args=..\..\B4J\%PROJECT_NAME%.b4j
```

And you can check the new project. The project folder and the files have the correct name.



You can make any other template with the same principle.

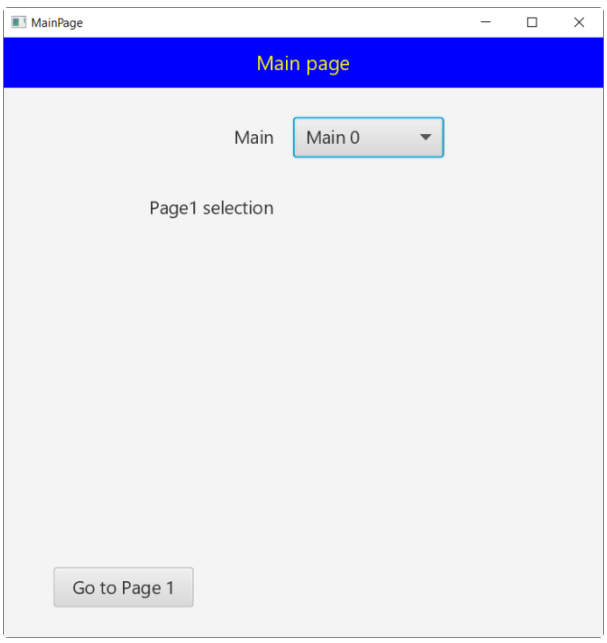
8 Project with 2 pages B4XPagesTwoPages

This is a simple project to show how to handle projects with two Activities / Pages /Forms.

The term ‘page’ stays for *Activity* in B4A, *Page* in B4i and *Form* in B4J.

Project name: B4XPagesTwoPages.

It is a quite simple project containing two pages with:



Main Page:
A B4XComboBox for a selection.
A Label to show the selection of Page1.
A Button to show Page1.
There is no Page1 selection yet.

Page 1:
A Label to show the selection of MainPage.
A B4XComboBox for a selection.
This one will also be displayed in MainPage.

The two B4XComboBoes are used to make a selection in each page.
These selections are displayed in the other page just to show the interaction between the two pages.

8.1 Steps to follow

These are the steps to follow to design the B4XPagesTwoPages cross-platform project:

1. Run the B4J IDE and setup the project.

   The structure of the whole project is ready for all three platforms.

2. Modify the Package name in Build Configurations.
3. Add the Page1 Standard Class, save it to the parent folder, for it is a cross-platform module.

4. Define the two layouts 'MainPage' and 'Page1', not explained, they are quite simple.
   We need to check the XUI-Views library because we use a B4XCombobox.

5. Code for B4XMainPage
6. Code for Page1

   You can now run the B4J program.

7. Run the B4A IDE, modify the Package name in Build Configurations.
8. Modify the Application Label, `#ApplicationLabel: B4XPagesTwoPages`.
   In the IDE we see some lines in red, why?

```
12    Private Page1 As B4XPage1
13    Private lblPage1 As B4XView
14    Private xcbxMainPage As B4XComboBox
```

   The XUI Views library is missing, so we must check it, some of the red lines are now OK.

```
12    Private Page1 As B4XPage1
13    Private lblPage1 As B4XView
14    Private xcbxMainPage As B4XComboBox
```

   And we need to add the B4XPage1 class module to the project.

9. Add B4xPage1 as an Existing Module from the project folder with a relative link.

```
12    Private Page1 As B4XPage1
13    Private lblPage1 As B4XView
14    Private xcbxMainPage As B4XComboBox
```

10. Copy the two layouts 'MainPage' and 'Page1' from B4J to B4A.

    You need to adapt them for the sizes and perhaps for colors.
    If you do not see the B4XCombobox view when you copy the layout, this means that you have not checked the XUI Views library!

    No code to change !!!

    You can now run the B4A program.
    The layouts may need some fine tuning.

11. Run the B4i IDE

    The following steps are the same as for B4A.

12. Modify the Application Label.
13. Modify the Package name in Build Configurations.
14. Copy the two layouts 'MainPage' and 'Page1' from B4A to B4i.
    It is better to copy the layouts from B4A to B4i because the screen sizes are similar.
    You may need to adapt them for sizes or colors.

15. Add B4XPage1 as an Existing Module from the project folder with a relative link.

    No code to change !!!

    You can now run the B4i program.
    The layouts may need some fine tuning.

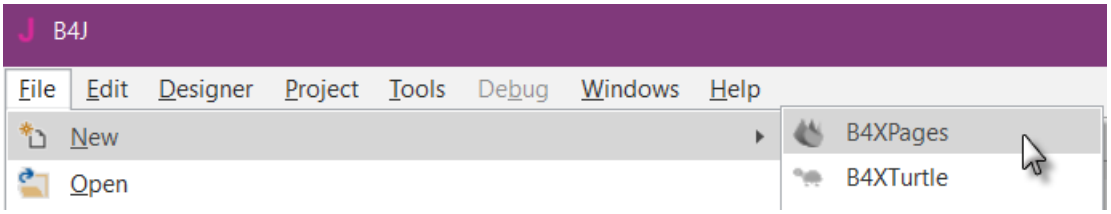
16. Conclusion.
    We see that, in this simple example, all the code written in the B4J project is reused in B4A and B4i without any change nor adaptation.

    If you have two or the three platform IDEs open and you change the code in any one of them and save the project the code in the cross-platform modules is automatically updated.

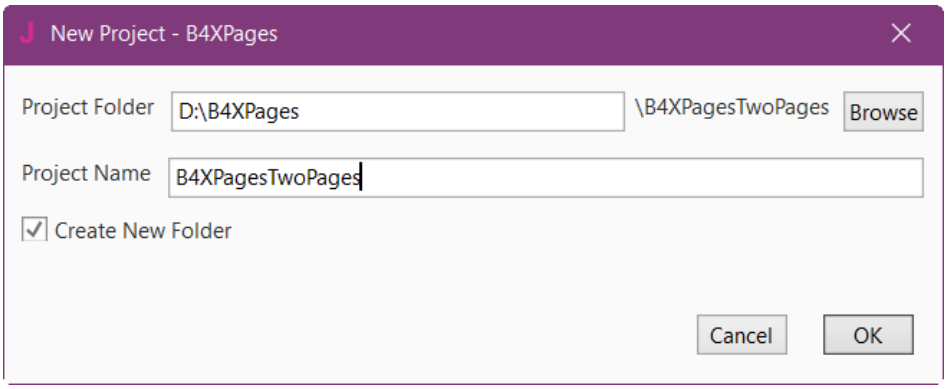
    Only the layouts need some adaptations.

8.2 Setup of a project

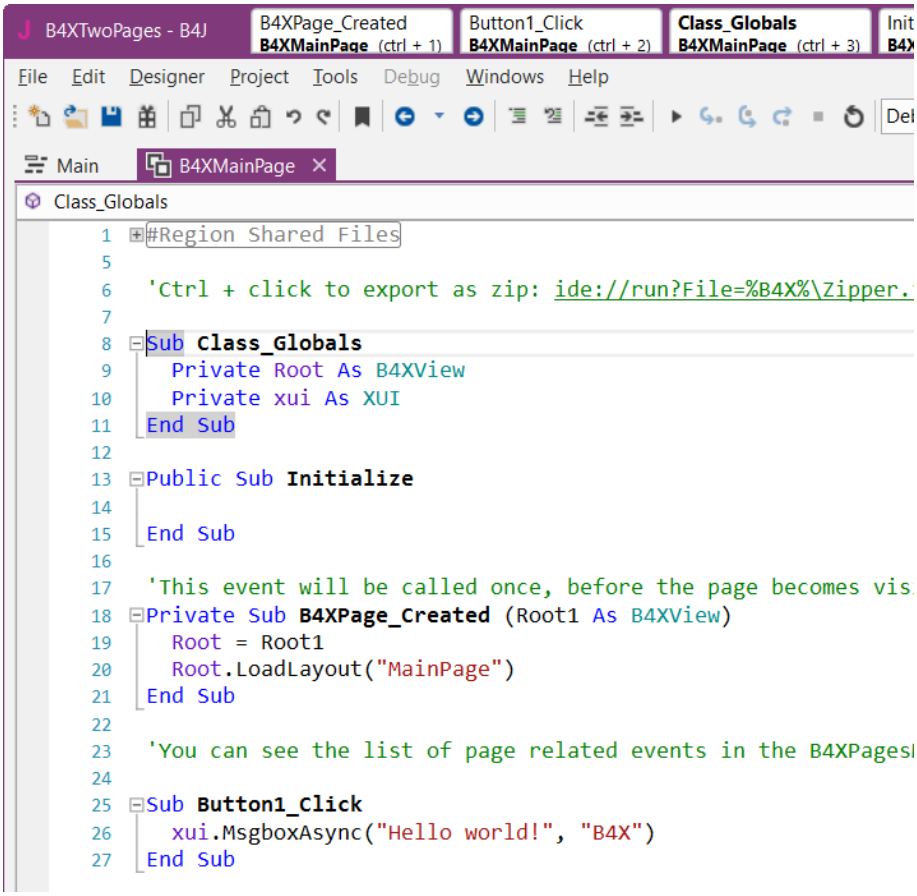
To create a new project, we run the B4J IDE:



Enter the Project Folder: *D:\B4XPages*, in my case, enter the project name, *B4XPagesTwoPages* in the example, check Create New Folder, and click on OK:

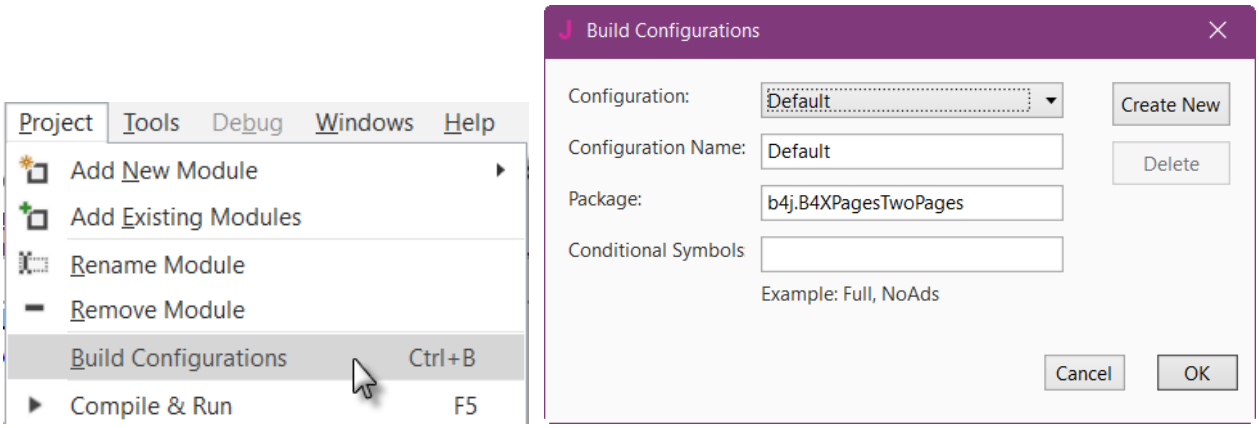


We arrive directly in the B4XMainPage.



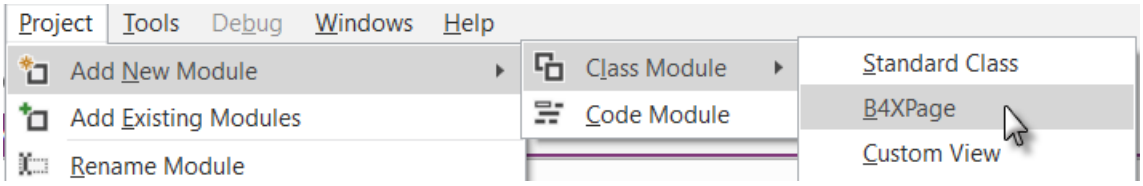
8.3 Modify the Package name in Build Configurations

As usual, I change the Package name in Build Configuration directly at the beginning:

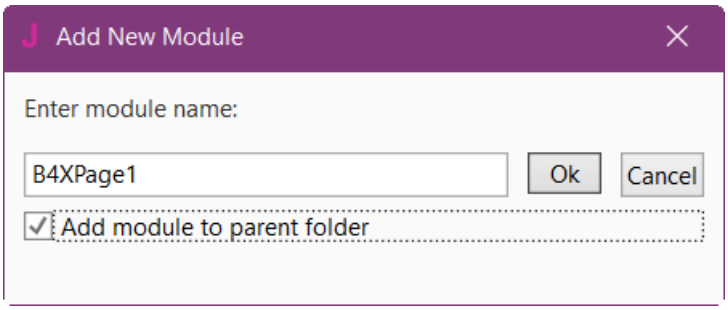


8.4 Add a new B4XPage to the project

Click Project / Add New Module / Class Module / B4XPage.

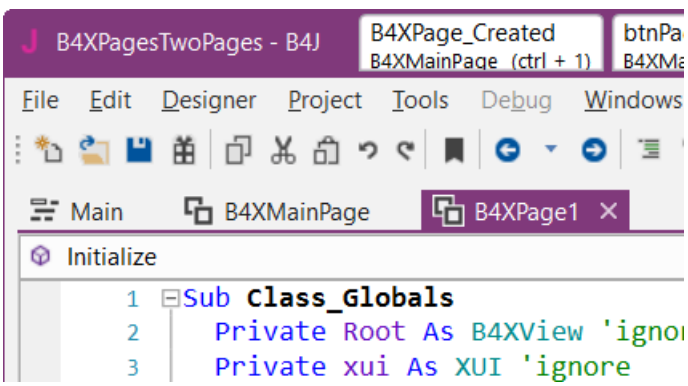


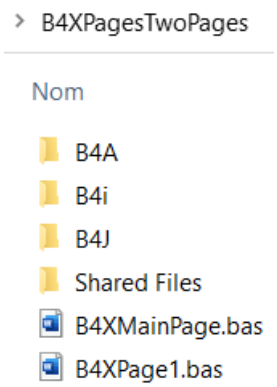
Enter the page name in the window below, *B4XPage1* in our example.



Check ☑ Add module to parent folder because B4XPage1 is a cross-platform module.

Now, we see the B4XPage1 module in the IDE and we can see that the B4XPage1.bas file is added in the B4XPagesTwoPages folder.





8.4.1 Code of B4XMainPage

In Class_Globals we:

- Declare Page1.
- Declare the views.
- Declare two public variables for the selection.

```
Sub Class_Globals
   Private Root As B4XView
   Private xui As XUI

   Private Page1 As B4XPage1
   Private lblPage1 As B4XView
   Private xcbxMainPage As B4XComboBox

   Public SelectedIndex As Int   'Selected index from the xMain B4XComboBox
   Public SelectedItem As String 'selected item from the xMain B4XComboBox
End Sub
```

Note: Do not use a same name for the Page and the module like, `Private Page1 As Page1.`
Because in B4i it is not allowed to use a variable name the same as a module name.
In B4J and B4A it would work with a warning, in B4i it throughs an error !

In B4XPage_Created we:

- load the layout, initialize, and add Page1.
- Fill the B4XCombobox.

```
'This event will be called once, before the page becomes visible.
Private Sub B4XPage_Created (Root1 As B4XView)
   Root = Root1
   Root.LoadLayout("MainPage")

   Page1.Initialize 'initializes Page1
   B4XPages.AddPage("Page 1", Page1)   'adds Page1 to the B4XPages list

   FillComboBox
End Sub
```

And then, we have:

- FillCombobox, fills the combobox.
- btnPage1_Click, button click event to show Screen1.
- xcbxMainPage_SelectedIndexChanged, the B4XCombobox event.

```
'Initialize the B4XComboBox
'Select the first item
Private Sub FillComboBox
   Private i As Int
   Private lst As List

   lst.Initialize
   For i = 0 To 10
       lst.Add("Main " & i)
   Next
   xcbxMainPage.SetItems(lst)

   xcbxMainPage.SelectedIndex = SelectedIndex
   SelectedItem = xcbxMainPage.SelectedItem 'We memorize the index in the variable
End Sub

'Display Page1
Private Sub btnPage1_Click
   B4XPages.ShowPage("Page 1")
End Sub

Private Sub xcbxMainPage_SelectedIndexChanged (Index As Int)
   SelectedIndex = Index  'We memorize the index in the variable
   SelectedItem = xcbxMainPage.SelectedItem 'We memorize the index in the variable
End Sub
```

We add a last public routine to Update the MainPage:

```
'Updates the MainPage
Public Sub Update
   lblPage1.Text = Page1.SelectedItem
End Sub
```

This routine is called from Page 1 to update the display of the MainPage.

8.4.2 Code of B4XPage1

In Class_Globals we:

- Declare the two views.
- Declare two global variables for the selection.

```
Sub Class_Globals
   Private Root As B4XView 'ignore
   Private xui As XUI 'ignore

   Private lblMain As B4XView
   Private xcbxPage1 As B4XComboBox

   Public SelectedIndex As Int
   Public SelectedItem As String
End Sub
```

In B4xPage_Create we:

- Load the layout
- Fill the B4XCombobox

```
Private Sub B4XPage_Created (Root1 As B4XView)
   Root = Root1
   Root.LoadLayout("Page1")

   FillComboBox
End Sub
```

In B4XPage_Appear we:

- Update the display of the MainPage selection.

```
Private Sub B4XPage_Appear
   lblMain.Text = B4XPages.MainPage.SelectedItem
End Sub
```

In B4XPage_Disappear we:

- Call the Update routine of MainPage to update the display of the selection in Page1.

```
Private Sub B4XPage_Disappear
   B4XPages.MainPage.Update
End Sub
```

Then we have:

- FillComobox, fills the B4XCombobox.
- xcbxPage1_SelectionIndexChanged, the B4XCombobox event.

```
'Initialize the B4XComboBox with Page1 & i
'Select the first item
Private Sub FillComboBox
	Private i As Int
	Private lst As List

	lst.Initialize
	For i = 0 To 10
		lst.Add("Page1 " & i)
	Next
	xcbxPage1.SetItems(lst)

	xcbxPage1.SelectedIndex = 0	'We memorize the Index in the variable
	SelectedItem = xcbxPage1.SelectedItem  'We memorize the Item in the variable
End Sub

Private Sub xcbxPage1_SelectedIndexChanged (Index As Int)
	SelectedIndex = Index	'We memorize the Index in the variable
	SelectedItem = xcbxPage1.SelectedItem  'We memorize the Item in the variable
End Sub
```

These are almost the same as in B4XMainPage.

8.5 Create a B4XPages project zip file

To create a zip file for the complete project including all files we use feature in included in line 6 on top of the B4XMainPage code:

```
6   'Ctrl + click to export as zip: ide://run?File=%B4X%\Zipper.jar&Args=Project.zip
```

By default, the name of the zip file is Project.zip.

To get the name of our project, B4XTwoPages in our example, we can change Project.zip into B4xTwoPages.zip.

```
6   'Ctrl + click to export as zip: ide://run?File=%B4X%\Zipper.jar&Args=B4XTwoPages.zip
```

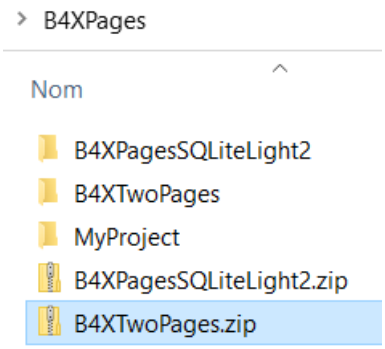
When you press on Ctrl and hover over the line, the right part is highlighted.

```
5
6   'Ctrl + click to export as zip: ide://run?File=%B4X%\Zipper.jar&Args=B4XTwoPages.zip
7
8   Sub Class Globals
```

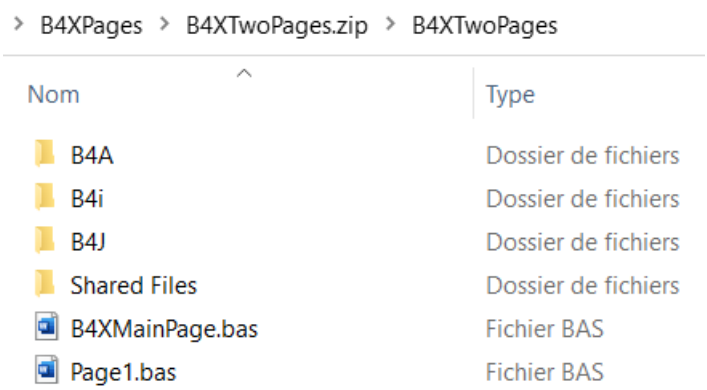
Ctrl + Mouse click to activate

To create the zip file, simply press on Ctrl + Mouse click.

You will get the project zip file.



And its content.



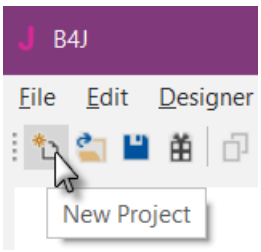
9 B4XPages version of the SecondProgram project

The goal of this chapter is to show how to modify three simple platform specific projects into one B4XPages cross-platform project.

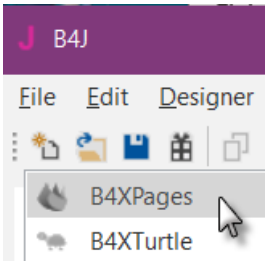
We use the three SecondProgram projects from the GettingStarted SourceCode.

9.1 Project structure

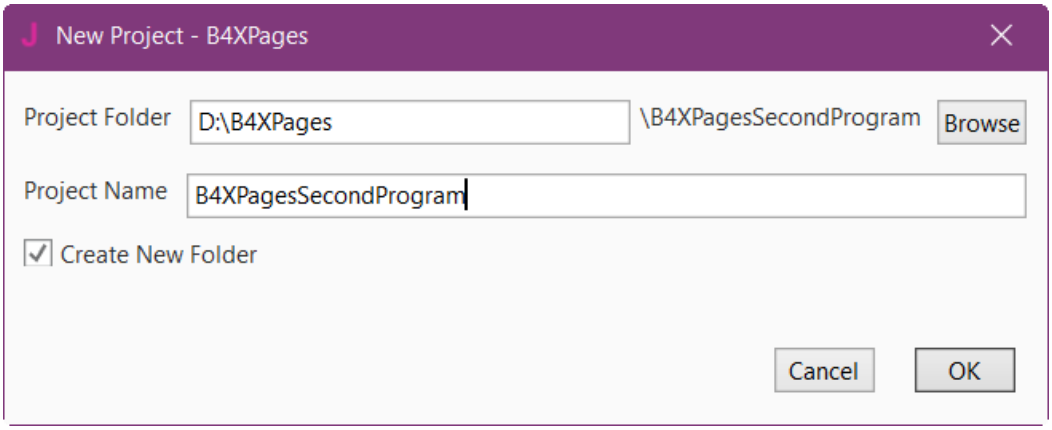
First, we create a new project.



Open the B4J IDE and select New.

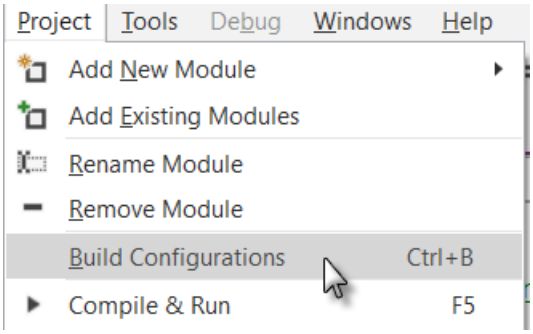


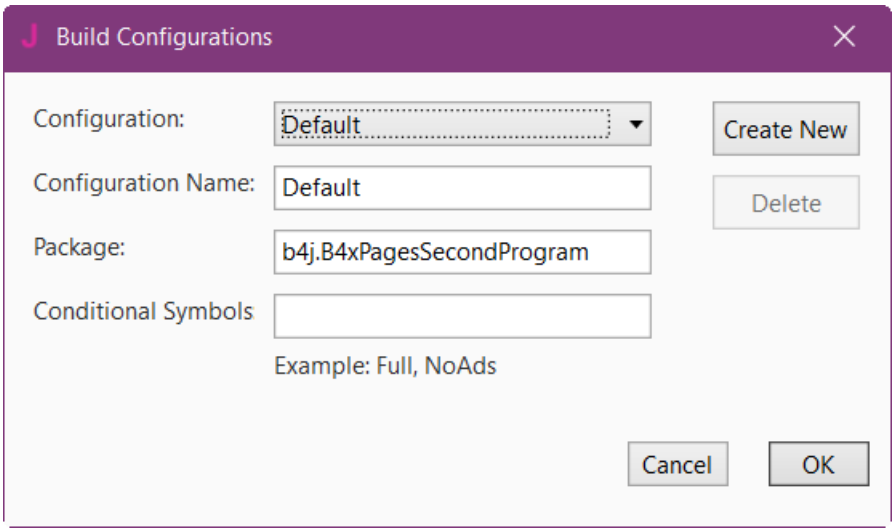
And select B4XPages.



Enter B4XPagesSecondProgram in the Project Name field.
Check Create New Folder to create a new folder.

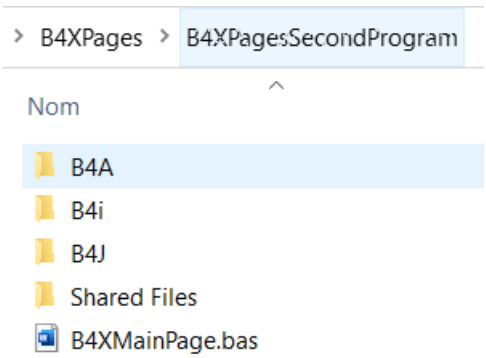
In the IDE click on Build Configurations.



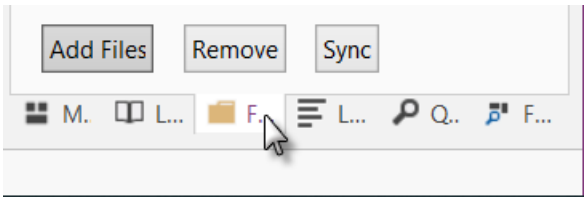


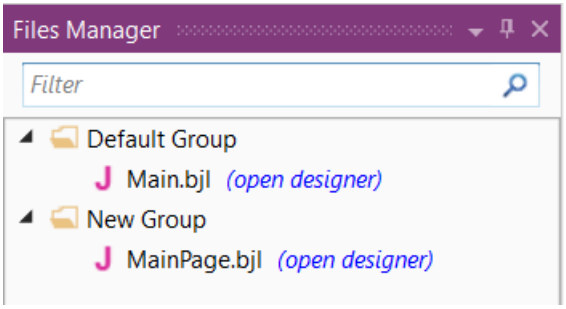
Enter *b4j.B4xPagesSecondProgram* in the Package field.

And we have the project structure.



As we already have made the layout, we load Main.bjl from the SecondProgram project in the GettingStarted source code folder.



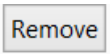


We now have the layout file in the project.

We can remove the default MainPage.bjl layout file.

Click on the MainPage.bjl file 

.

And click on 

.

If you open the Designer, you will see the layout.

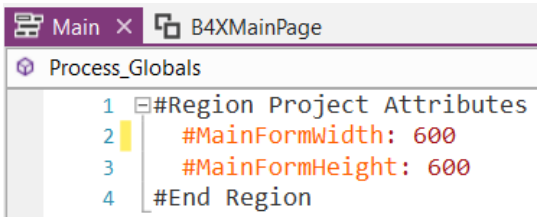
In the IDE Change the layout file name from MainPage to Main.
Remove the Button1_Click event routine.

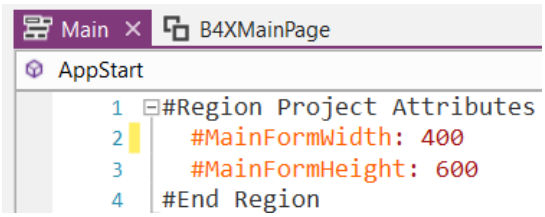
Now you should have this:

```
17  'This event will be called once, before the page
18 Private Sub B4XPage_Created (Root1 As B4XView)
19    Root = Root1
20    Root.LoadLayout("Main")
21  End Sub
22
23  'You can see the list of page related events in
24
```

9.2 B4J code adaptation

Modify on top in the Main module the value of #MainFormWidth: from 600 to 400.





Open a second instance of the B4J IDE and open the SecondProgram B4J project, located in the GettingStarted folder.

Copy lines 9 to 15

```
6  Sub Process_Globals
7    Private fx As JFX
8    Private MainForm As Form
9    Private btnAction, btn0 As Button
10   Private lblComments As Label
11   Private lblMathSign As Label
12   Private lblNumber1 As Label
13   Private lblNumber2 As Label
14   Private lblResult As Label
15   Private Number1, Number2 As Int
16 End Sub
```

Into Class_Globals in the B4XMainPage module and change `Button` and `Label` into `B4XView`. We use XUI views instead of platform specific views.

```
8  Sub Class_Globals
9    Private Root As B4XView
10   Private xui As XUI
11
12   Private btnAction, btn0 As B4XView
13   Private lblComments As B4XView
14   Private lblMathSign As B4XView
15   Private lblNumber1 As B4XView
16   Private lblNumber2 As B4XView
17   Private lblResult As B4XView
18   Private Number1, Number2 As Int
19 End Sub
```

Then, in B4XPage_Created we add the call to NewProblem.

```
25  'This event will be called once, before the page
26 Private Sub B4XPage_Created (Root1 As B4XView)
27   Root = Root1
28   Root.LoadLayout("Main")
29
30   NewProblem
31 End Sub
```

From the SecondProgram code, copy all the code beginning with Sub btnAction_MouseClicked to the B4XPagesSecondProgram project.

Now we must modify the B4J specific code to make it cross-platform.

Change these two lines:
```
Private Sub btnAction_MouseClicked (EventData As MouseEvent)
Private Sub btnEvent_MouseClicked (EventData As MouseEvent)
```

Into these:
```
Private Sub btnAction_Click
Private Sub btnEvent_Click
```

We need to replace the B4J specific **MouseClicked** event by the cross-platform **Click** event without any parameters.

We need to change the B4J specific MsgBox call into the cross-platform MsgBox call:
We replace this
```
fx.Msgbox(Null, "No result entered", "E R R O R")
```

By this:
```
xui.MsgboxAsync("No result entered", "E R R O R")
```

We need to modify the three B4J specific color setting calls into cross-platform settings.
Replace these:lines
```
CSSUtils.SetBackgroundColor(lblComments, fx.Colors.RGB(255,235,128))' yellow color
CSSUtils.SetBackgroundColor(lblComments, fx.Colors.RGB(128,255,128))' light green color
CSSUtils.SetBackgroundColor(lblComments, fx.Colors.RGB(255,128,128))' light red color
```

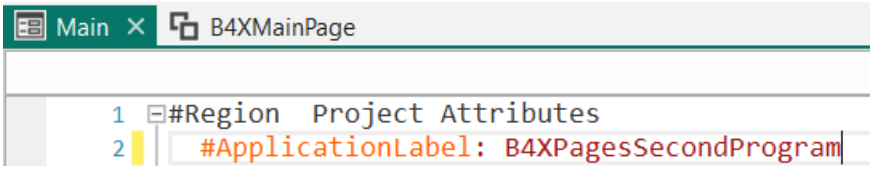
By these:
```
lblComments.Color = xui.Color_RGB(255,235,128)  ' yellow color
lblComments.Color = xui.Color_RGB(128,255,128)  ' light green color
lblComments.Color = xui.Color_RGB(255,128,128)  ' light red color
```

That is ALL!

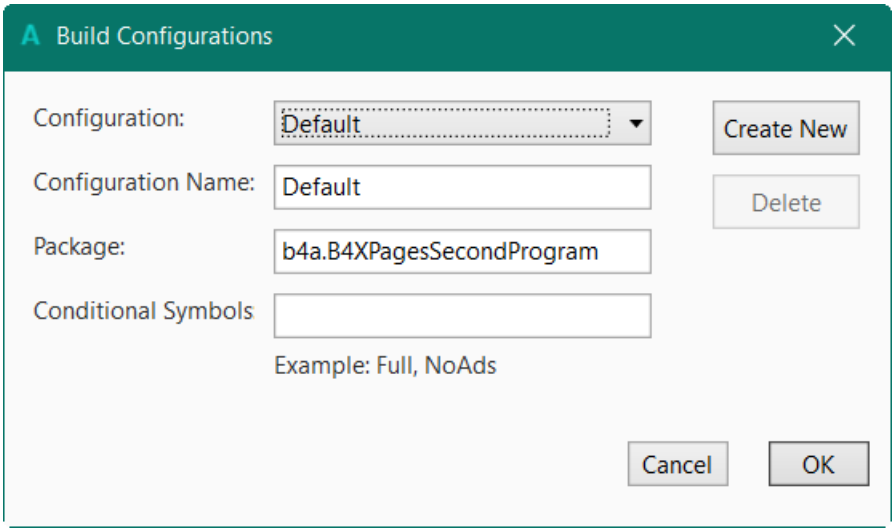
9.3 B4A adaptation

As for B4J, we need to copy the Main.bal file from the B4A SecondProgram project.
Remove the MainPage.bal file.

Modify the `#ApplicationLabel:` on top of the Main module.



Modify the Package in Build Configurations.



All the code in B4XMainPage has already been made cross-platform, therefore ne change needed.

This is ALL!

9.4 B4i adaptation

Copy the Main.bil file and remove the MainPage.bil file.
Modify the `#ApplicationLabel:` on top of the Main module, the same as for B4A.
Modify the Package in Build Configurations, the same as for B4A.

This is ALL!

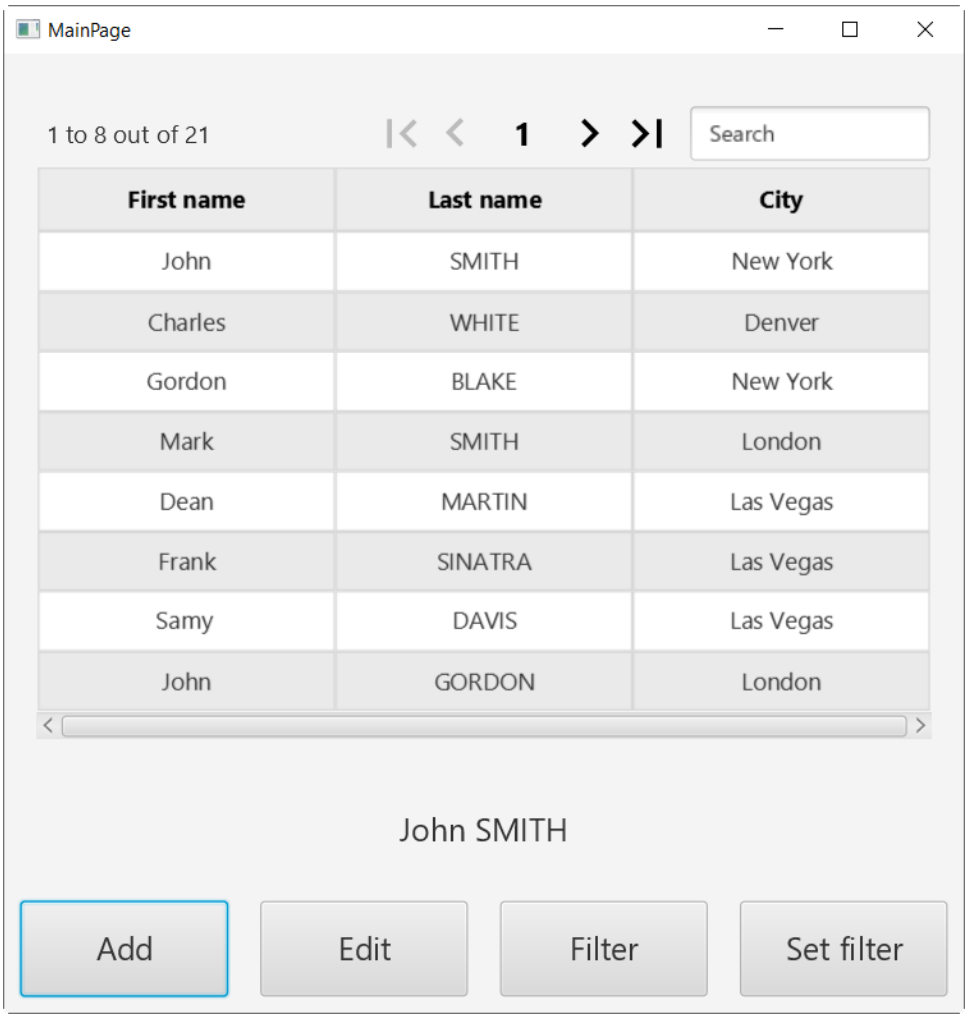
9.5 Conclusion

It was quite easy to make this project cross-platform. The layout files did already exist, nothing to do. The B4J code did need some modification to replace B4J specific code into cross-platform code. The platform specific views (nodes) declaration had to be changed into B4XView in their declaration.

10 B4XPagesSQLiteLight2 project

This project is almost the same as the SQLiteLight2 projects from the B4X SQLite database booklet. But this one is cross-platform using as much as possible common code, XUI Views and, of course the XUI library.

The WebView Table has been replaced by the B4XTable view from the XUI Views library.



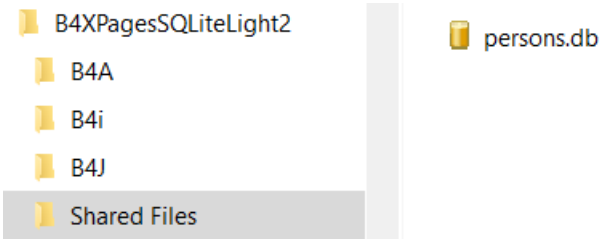
10.1 Project structure

The program has three pages:

- B4XMainPage
- PageEdit
- PageFilter

PageEdit and PageFilter are called only from B4XMainPage.

The program uses a database file, *persons.db*, which is saved in the Shared Files folder of the project.



This file is automatically copied into the Files folder of each product.

Then, in the code, this file must be copied to another folder because SQLite database files cannot be used directly from the project Files folder. This is explained below.

10.2 Comments about the code

I will not explain how to make a new project, this has already been explained before.
I will only highlight some more specific aspects.

10.2.1 Copy a file from the Files folder

In this case we use: `xui.DefaultFolder`.

This folder is:

- B4A same as File.DirInternal
- B4i same as File.DirDocuments
- B4J same as File.DirData. You must first call SetDataFolder once before you can use this folder. In our case: `xui.SetDataFolder("B4XPagesSQLiteLight2")`

And the code:

```
#If B4J
   xui.SetDataFolder("B4XPagesSQLiteLight2")
#End If

   If File.Exists(xui.DefaultFolder, "persons.db") = False Then
      'copy the default DB
      File.Copy(File.DirAssets, "persons.db", xui.DefaultFolder, "persons.db")
   End If
```

10.2.2 Initialize the SQLite database

There is a difference between B4A / B41 and B4J, in B4J you must specify SQLite.

The code in the B4XMainPage:

```
#If B4J
  SQL1.InitializeSQLite(xui.DefaultFolder, "persons.db", True)
#Else
  SQL1.Initialize(xui.DefaultFolder, "persons.db", True)
#End If
```

11 B4XPages Navigation through pages and program flow

The B4XPagesThreePages project shows different ways to navigate through the different pages.
It shows also, in the Logs, the program flow and which event is raised when.

The project has three pages:

- MainPage has two Buttons one for each Page.
- Page2 has one Button for Page3.
- Page3 no Button.

Navigation possibilities:

- 1 > 2 > 3 > 2 > 1
  When we go from Page2 to Page3 and then close Page3 we come back to Page2.
  And go from MainPage to Page 2 and back, or go from MainPage to Page 3 and back.

- 1 > 2 > 3 > 1
  When we go from Page2 to Page3 and then close Page3 we go directly back to MainPage.
  Unfortunately, this mode does not work in B4i.

  For this we add `B4XPages.ClosePage(Me)` in the **`btnPage3_Click`** routine after `B4XPages.ShowPage("Page 3").` This removes Page2 from the stack list.

```
Sub btnPage3_Click
   B4XPages.ShowPage("Page 3")
   If MP.NavigationIndex = 1 Then       '1 > 2 > 3 > 1
      B4XPages.ClosePage(Me)            'this does not work in B4i !!!
   End If
End Sub
```

- 1 > 2 > 1 1 > 3 > 1
  We can only go from MainPage to Page 2 and back or go from MainPage to Page 3 and back.
  For this, we set btnPage3.Visible = False.
  In a ‘normal’ case there should not be any button in the layout of Page2.
  I added it to show the different methods.

Just test it and play a bit with it.

12 B4XPagesNavBar

This project contains four pages and a NavBar class, displayed at the bottom of each page, and used to navigate between the different pages.

The NavBar has four buttons, one for each page.

The NavBar is initialized the B4XPage_Created routine of each page with;

```
NavBar1.Initialize(Root)
```

In B4XMainPage we define a global variable holding the current active page.

```
Public CurrentPage As Object
```

Code of the NavBar class.

In the Initialize routine we transmit the Parent view.
Set the variable MP for the B4XMainPage
And load the NavBar layout onto the parent view.
Close the current Page, if it is not the MainPage, when a new one is opened.

```
Public Sub Initialize(Parent As B4XView)
   pnlParent = Parent

   MP = B4XPages.MainPage
   pnlParent.LoadLayout("NavBar")
End Sub
```

Code for the button events.

```
Private Sub btnNavBar_Click
   Private btn As B4XView
   Private Index As Int

   btn = Sender
   Index = btn.Tag

   If MP.CurrentPage <> MP Then
      B4XPages.ClosePage(MP.CurrentPage)
   End If

   Select Index
      Case 1
         MP.CurrentPage = MP
         B4XPages.ShowPage("MainPage")
      Case 2
         MP.CurrentPage = MP.Page2
         B4XPages.ShowPage("Page 2")
      Case 3
         MP.CurrentPage = MP.Page3
         B4XPages.ShowPage("Page 3")
      Case 4
         MP.CurrentPage = MP.Page4
         B4XPages.ShowPage("Page 4")
   End Select
End Sub
```

The project includes the access of an object from another page explained in chapter:
Access objects from another Page.

13 Other projects in the forum

Below a small list of available projects in the forum:

- [B4X] ThreePagesExample
  A basic example.

- [B4X] B4XPages + B4XDrawer
  Creates a B4XPages example using B4XDrawer.

- [B4X] [B4XPages] Splash Screen
  B4A, B4i and B4J splash screen implementation. The implementation is different in each platform.

- [B4X] [B4XPages] Pleroma / Mastodon Client
  Roughly speaking, Mastodon is an open source, distributed, social network a bit similar to Twitter.

To find other B4XPages projects in the forum, type [B4X] [B4XPages] in the search field and you get a list of projects.