Module 1: Installation of useful tools for the course

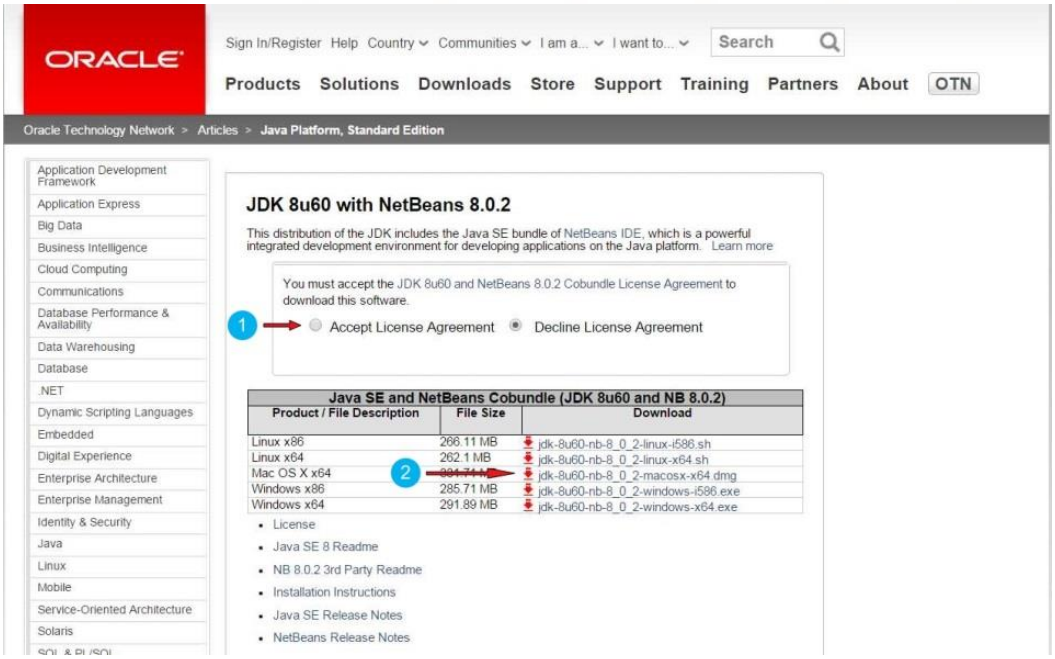
Level of difficulty: ● ○ ○ ○
Estimated time: 5 mn + download.

How to install NetBeans and the Codename One plugin on a MAC

1. **Your MAC needs to run OS X 10.7 (Lion) or a more recent version**.
   Find the OS X version of your MAC.

2. Go to http://www.oracle.com/technetwork/articles/javase/jdk-netbeans-jsp-142931.html

3. Click on "Accept License Agreement" and download the file "jdk-8u60-nb-8_0_2-macosx-x64.dmg" (see below)



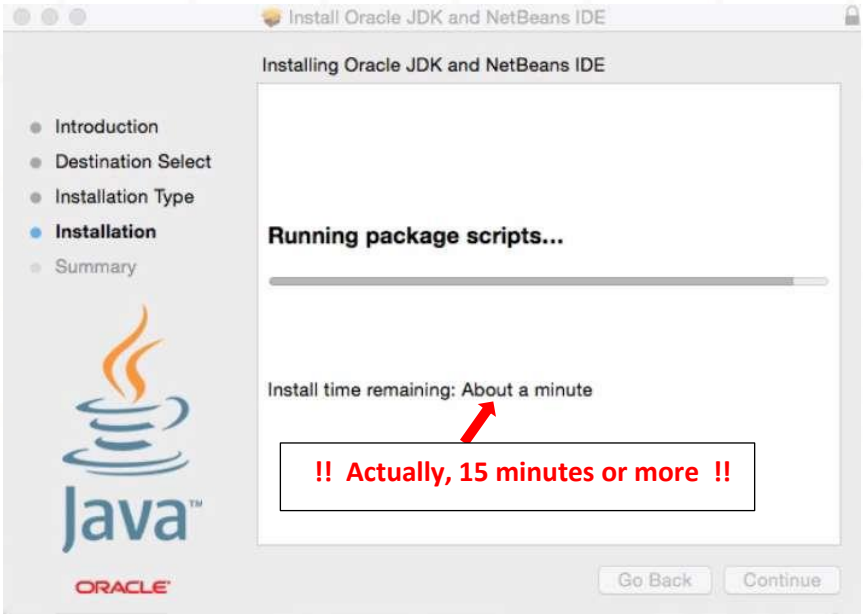
4. Find the file you downloaded on your computer and click on it to install
   Note that you might need administrator rights to install software on your MAC!

5. During the installation, you might be waiting for a long time (15 minutes or more) at this screen, even if this says that just one minute is remaining. This is normal (see following page!)

Module 1: Installation of useful tools for the course

Level of difficulty: ● ○ ○ ○
Estimated time: 5 mn + download.

How to install NetBeans and the Codename One plugin on a MAC



Now that we have NetBeans installed, let's add the Codename One plugin.

6. Open NetBeans. In the menu, select Tools -> Plugins.
   In the windows opening, select "Available plugins". Search for "Codename One" (plugins can be listed alphabetically by clicking on the column "name", that makes it easier to find).
   Select the Codename One plugin by ticking the box on the left. Click on "Install" and follow instructions.
7. Restart NetBeans to finish the installation.

Congratulations, you are ready to create your first mobile app!

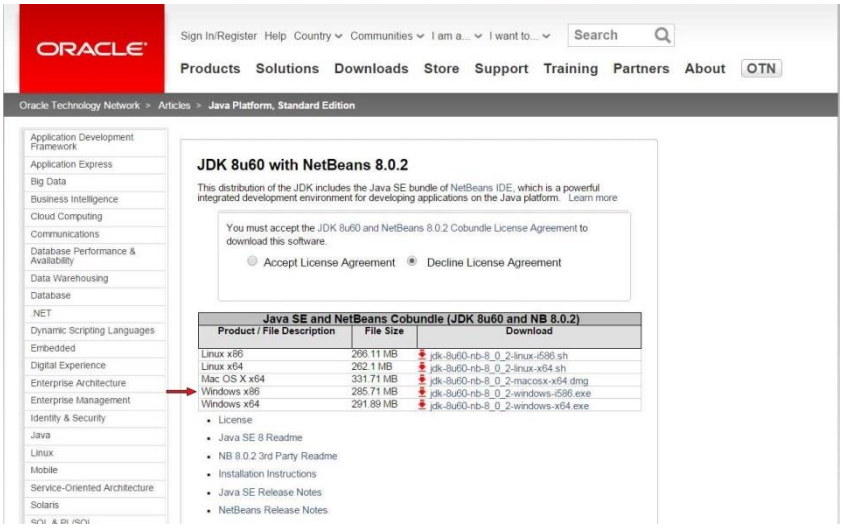
Module 1: Installation of useful tools for the course

Level of difficulty: ● ○ ○ ○
Estimated time: 5 mn + download.

How to install NetBeans and the Codename One plugin on a PC

1. Go to http://www.oracle.com/technetwork/articles/javase/jdk-netbeans-jsp-142931.html

2. Click on “Accept License Agreement”.
   There are two different files for Windows. Which one should you choose?
   ➔ **If you don’t know, choose “jdk-8u60-nb-8_0_2-windows-i586.exe” (see below)**
   ➔ *Advanced users who have a Win64 PC can choose the other file.*



3. Find the file you downloaded on your computer and click on it to install.

   Note that you might need administrator rights to install software on your PC!

   Now that we have NetBeans installed, let’s add the Codename One plugin.

4. Open NetBeans. In the menu, select Tools -> Plugins.
   In the windows opening, select “Available plugins”. Search for “Codename One” (plugins can be listed alphabetically by clicking on the column “name”, that makes it easier to find).
   Select the Codename One plugin by ticking the box on the left. Click on “Install” and follow instructions.
5. Restart NetBeans to finish the installation.

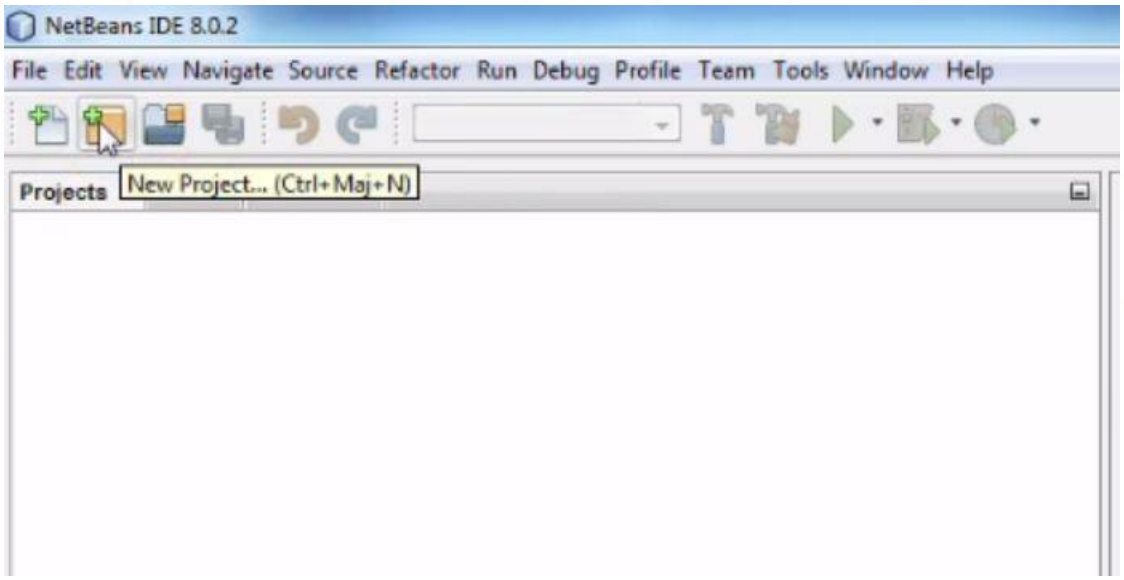
Congratulations, you are ready to create your first mobile app!

Module 1: Installation of useful tools for the course

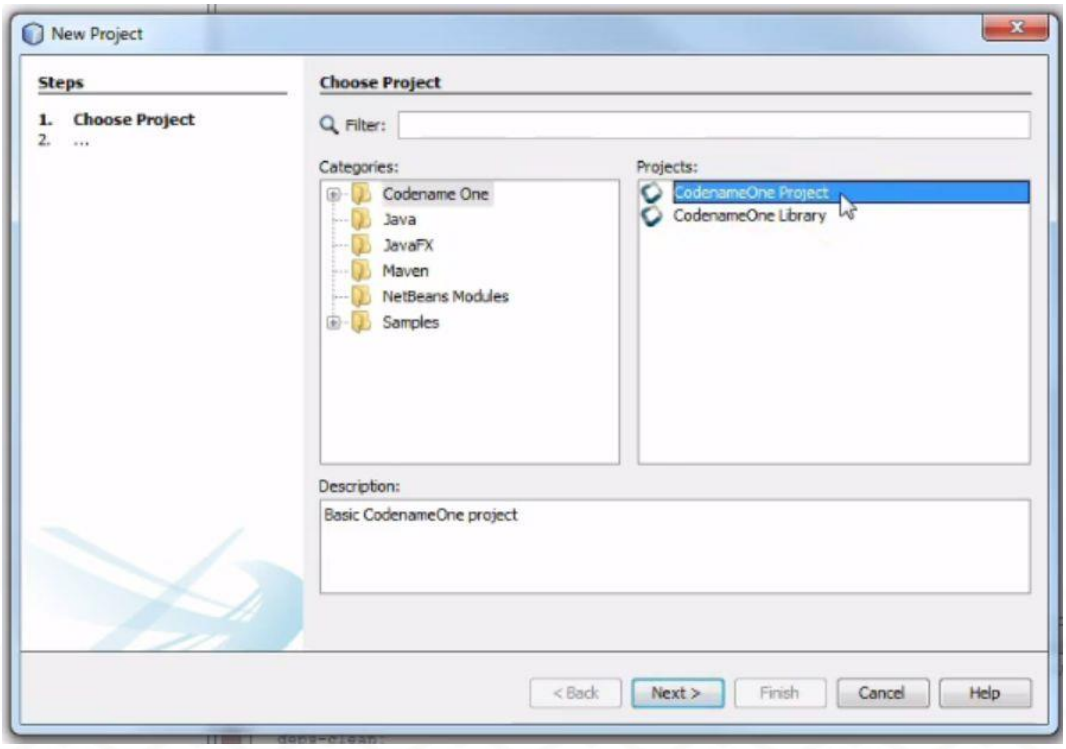
Level of difficulty: ● ○ ○ ○
Estimated time: 5 mn

Creating your first project

1. Click on the second icon from the left



2. Select a « CodenameOne project »

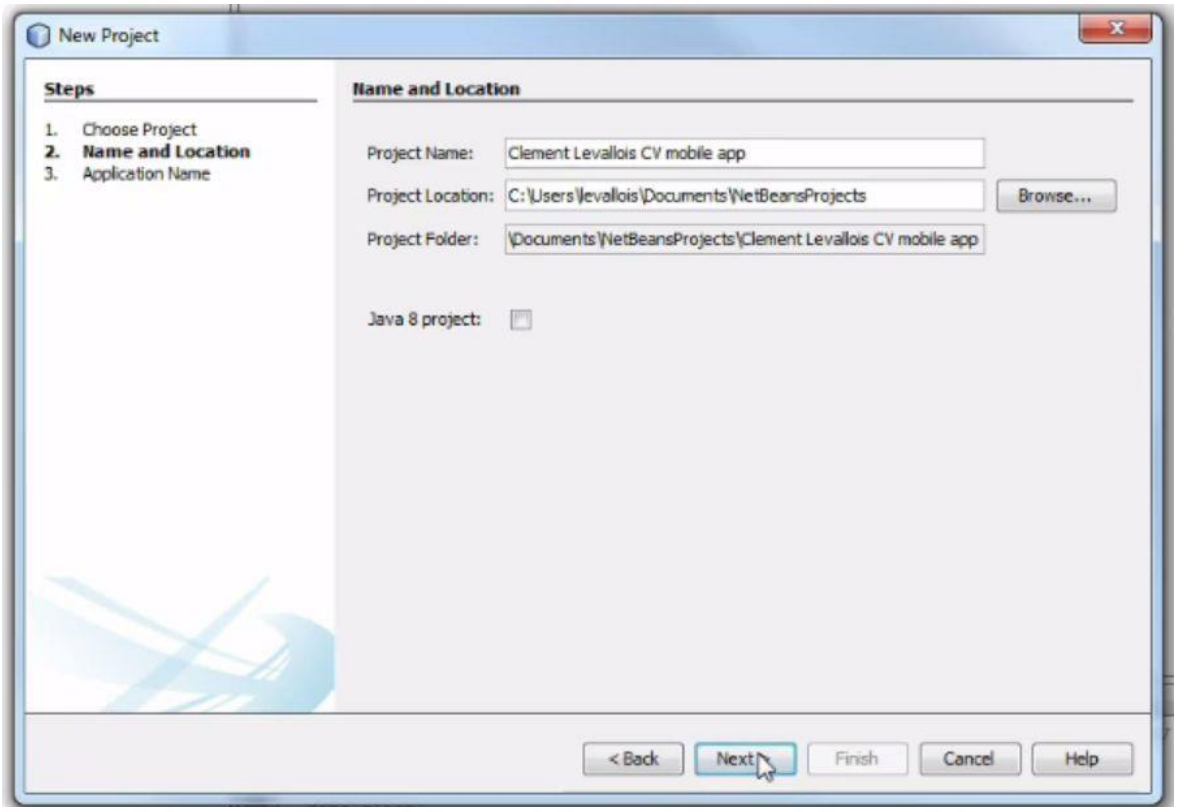


Module 1: Installation of useful tools for the course

Level of difficulty: ● ○ ○ ○
Estimated time: 5 mn

Creating your first project

3. Choose a name for your project
   No special characters like - _# »& @
   This will be the name of your app on the app store so please choose it carefully !
   In this course, we start by building an app showing your curriculum vitae.
   An idea for the name of your app would be « FirstName LastName Mobile CV »



CODAPPS
Coding Mobile Apps for Entrepreneurs

EMLYON business school

Module 1: Installation of useful tools for the course

Level of difficulty: ● ○ ○ ○
Estimated time: 5 mn

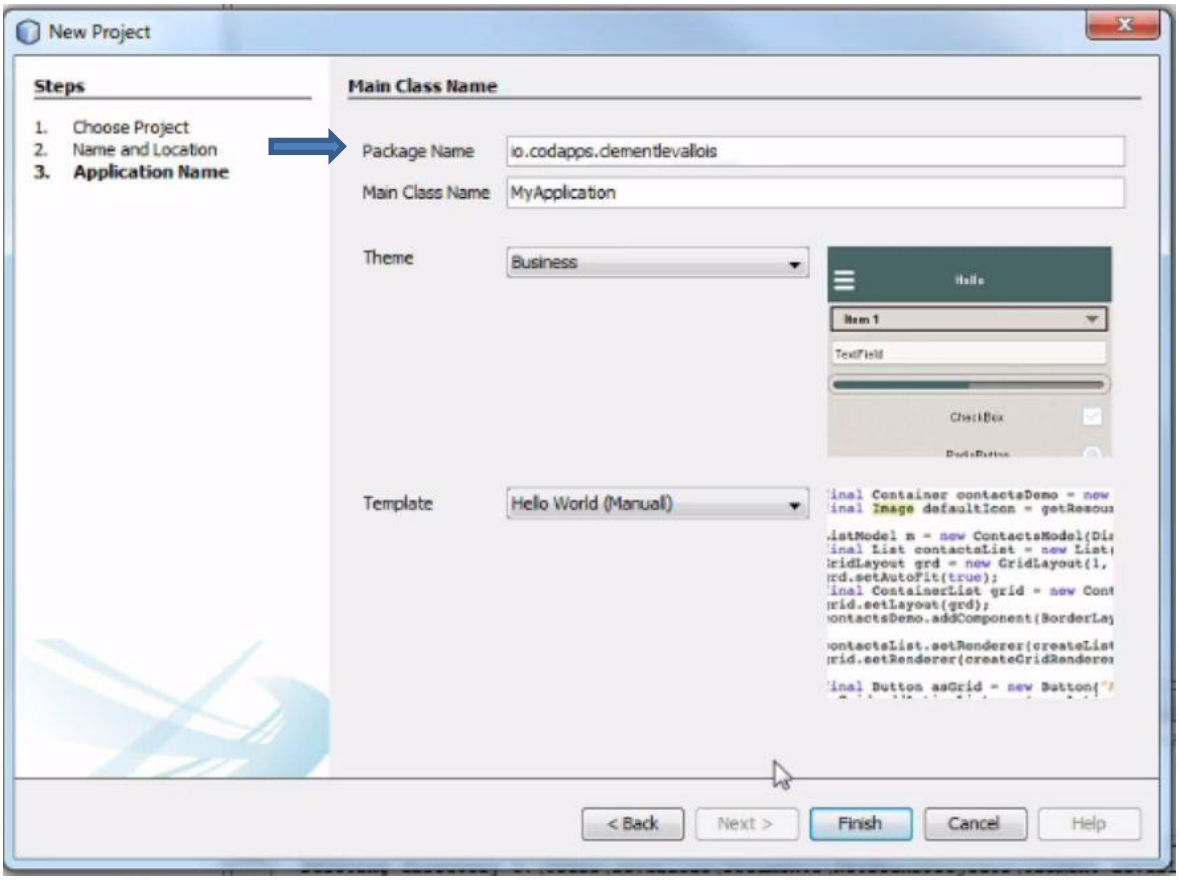
Creating your first project

5. Choose a package name, a Theme and a Template
   Package name : this is the « id card » of your app, it should be unique. How can we find a unique id for an app. The convention is to take the name of your website and reverse it. Like ; if you work at yahoo in France (www.yahoo.fr), you'd have a package name like : fr.yahoo.myamazingapp.
   If you don't have a website, take a nickname you use on social media or Skype, add a dot and then put the name specific to your app, like:
   seinecle.mycv

   EMLYON Students : please use the package name **io.codapps**.something, this is to facilitate access to the Apple iOS University program later in the course.

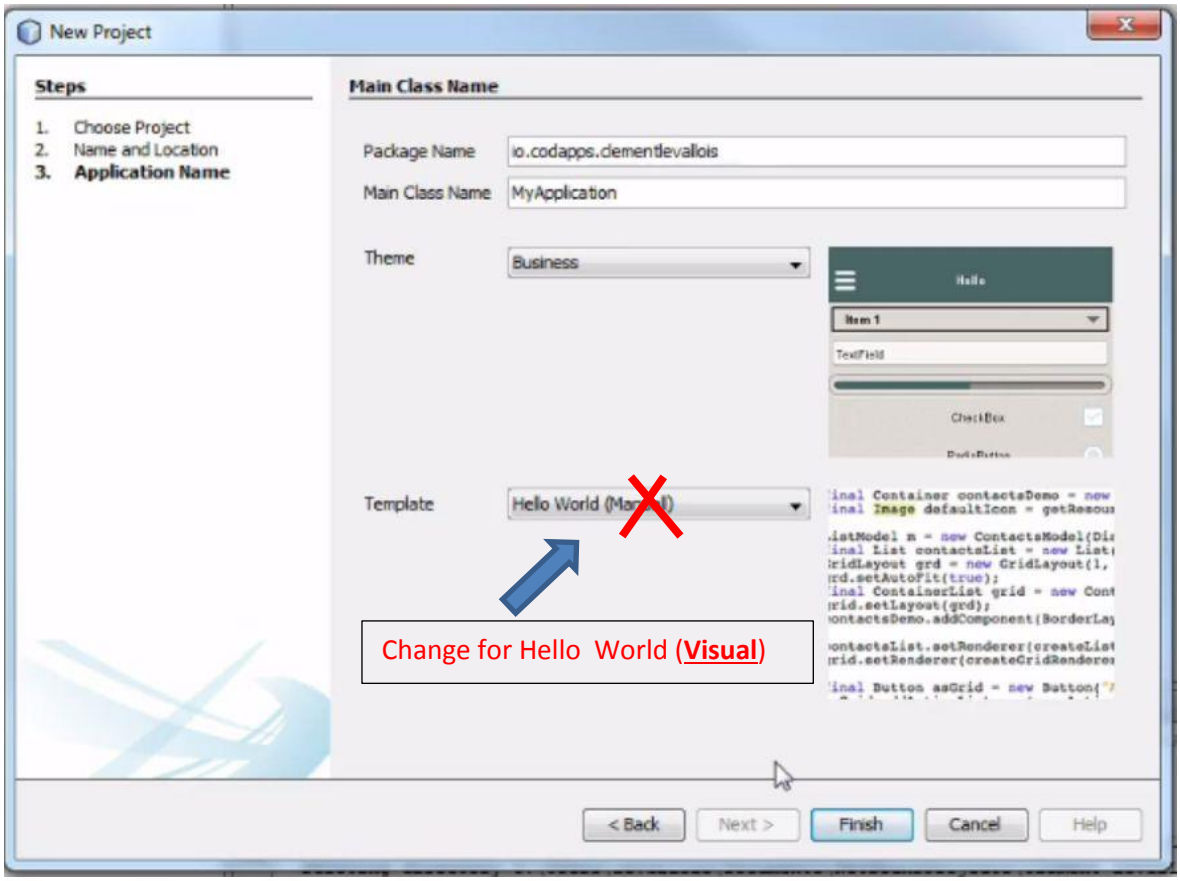
   Final note on packages : use just letters, no space or special characters.

   Then stay on this screen as we need to choose a Theme and a Template



6. Choose a theme, this is how your app will look like (you'll be able to modify that later).
   A special theme is « Native » : this means that your app will have an Apple look on iOS, look like Android style on Android phones, etc.

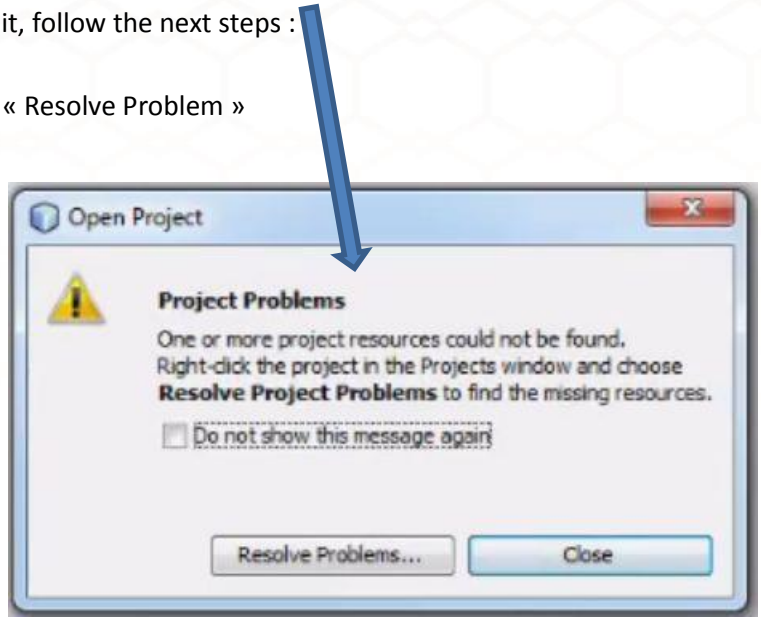
7. In the template, choose « Hello World **(Visual)** », not manual.
   This means that our project will be able that we are going to design our app in a window with the help of the mouse, instead of coding everything by hand. Saves us time !



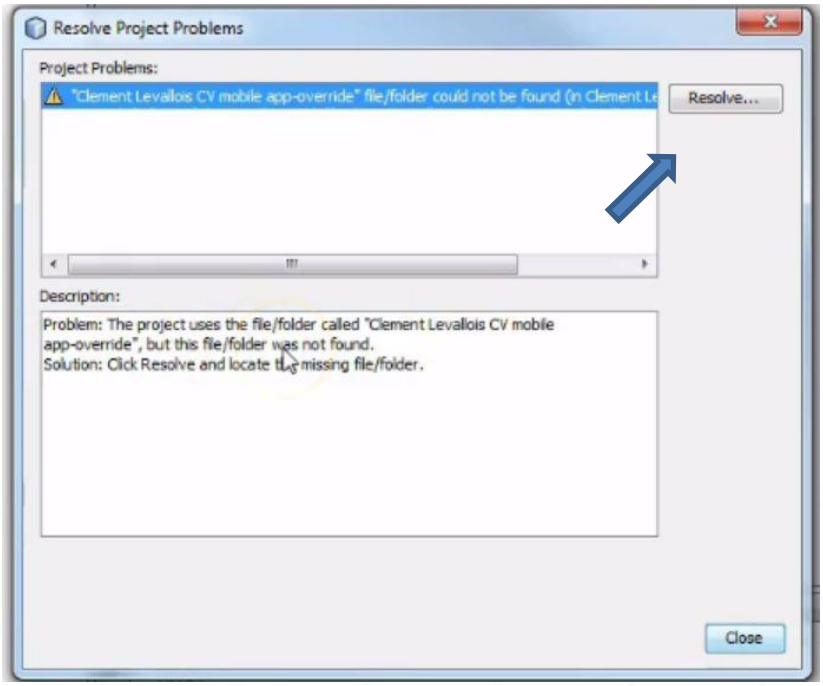
8. **Click on « Finish », your first app is ready !**

9. If you don't see this alert window, you are fine and you have finished this lesson. But if you see it, follow the next steps :

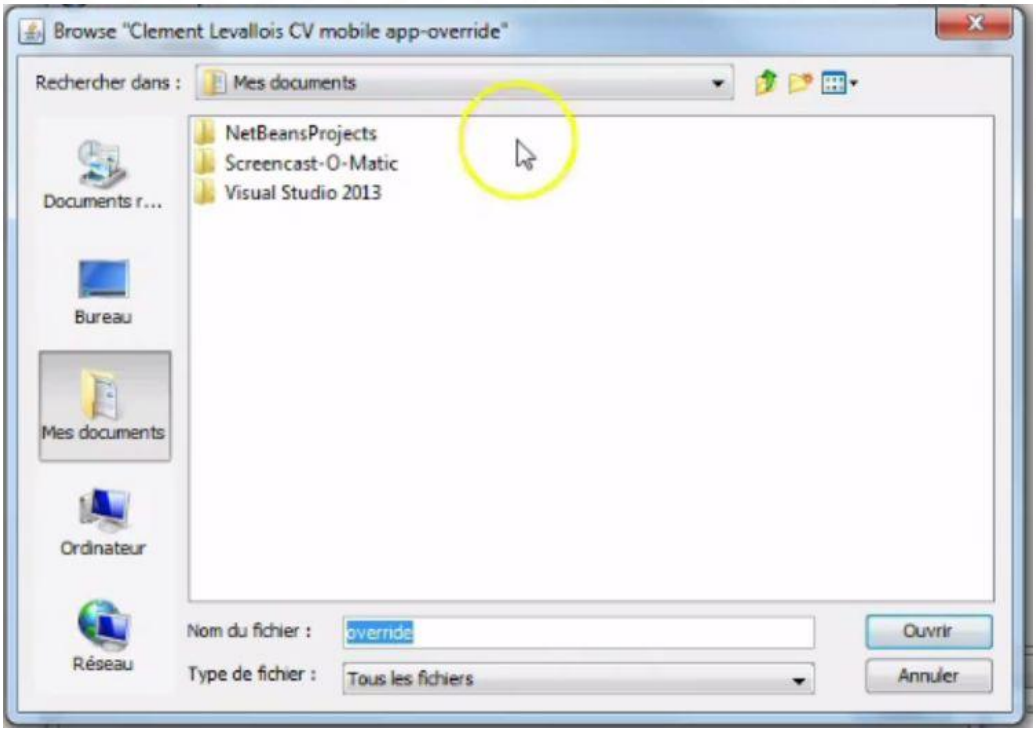
10. Click on « Resolve Problem »



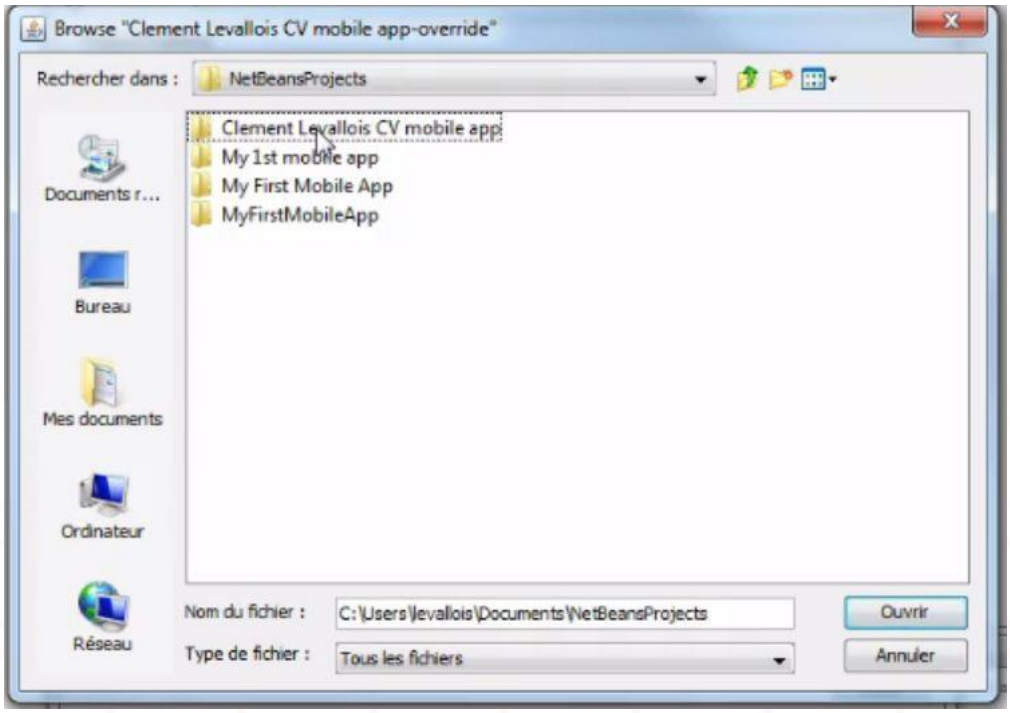
11. A new window opens. Click on « Resolve »



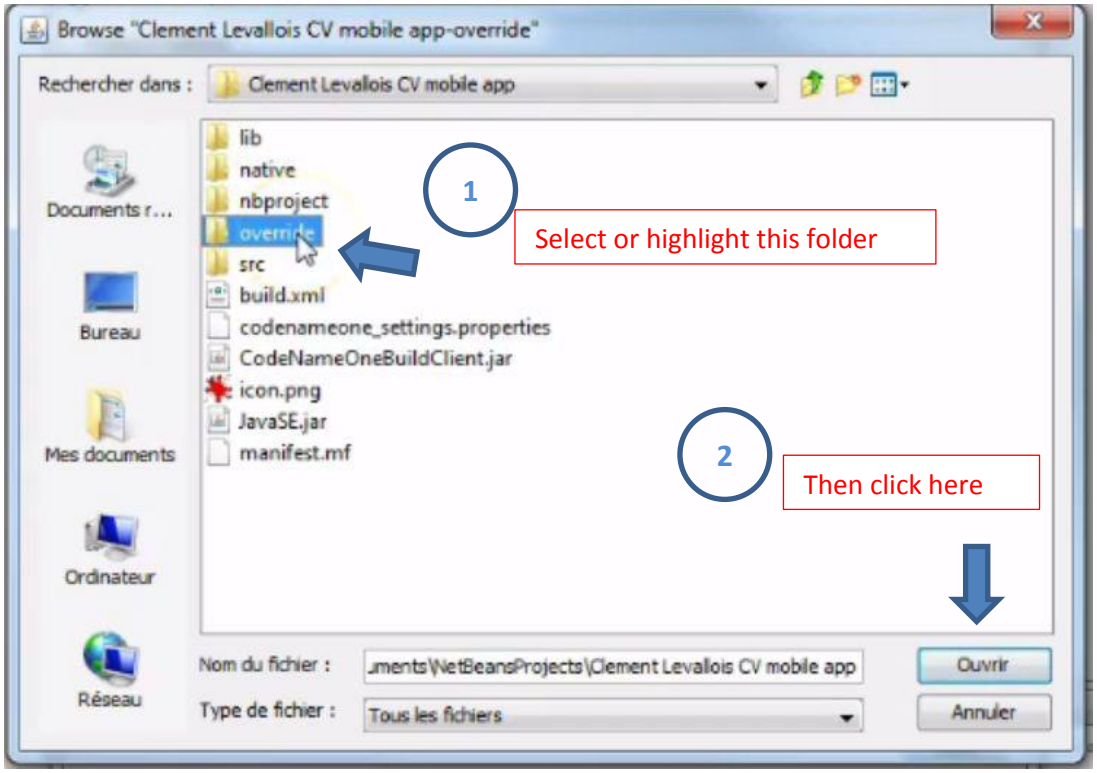
12. A file explorer opens. Open the folder « NetBeans projects »:



13. Then open the folder that has the name of the app we just created:



14. Then select (do not open by double clicking on it) the folder called "override". Click on the button "choose" or "open" and the bug disappears.
    Note: this bug is the only one in the tool we use. After that, we don't deal with such issues.



Congratulations, your first app is ready!

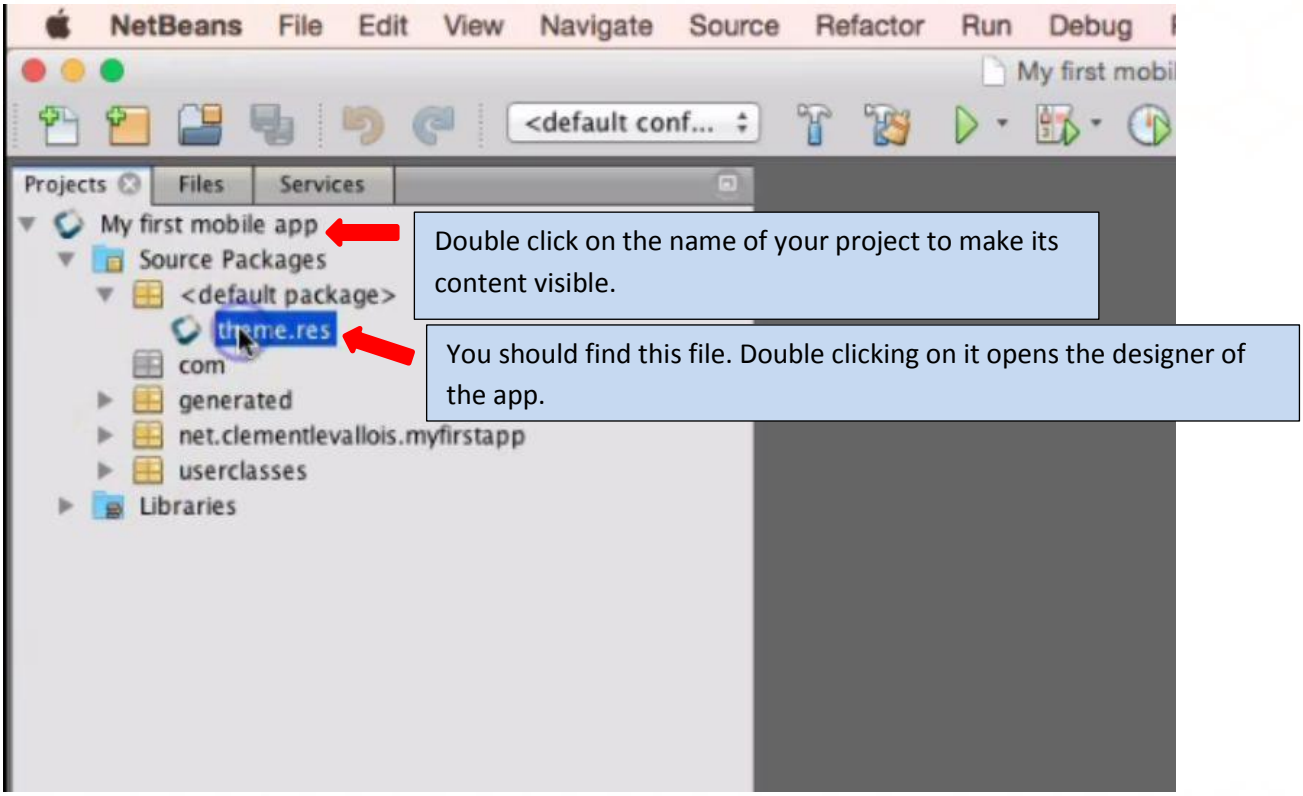
Module 1: Installation of useful tools for the course

Level of difficulty: ● ○ ○
Estimated time: 5 mn

Previewing your app on a phone simulated on your computer

1. Explore the content of your project by double clicking on the name of the project, then by double clicking on the folders (yellow icons) to open them.
   We are looking for the file "**theme.res**" in the folder "**default package**". **We double click on it**.

   Note: make sure you are in the "Projects" tab, not in "Files" or "Services"!



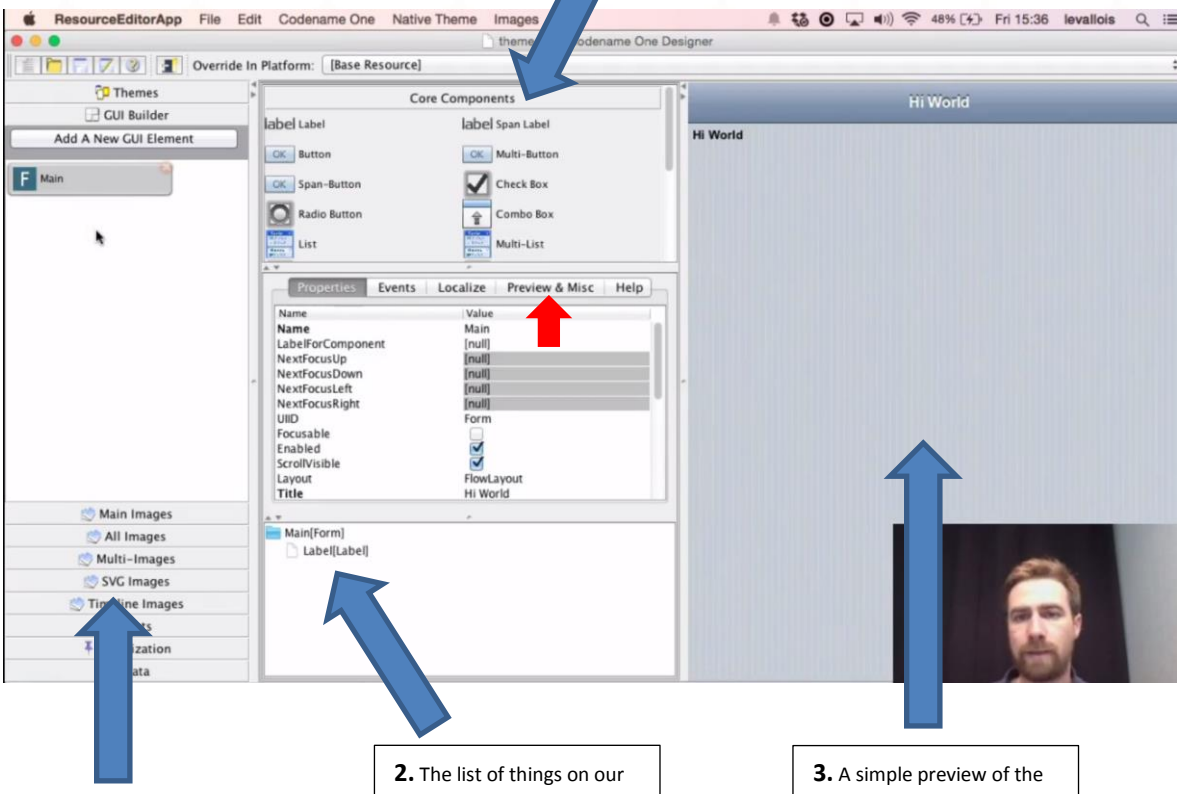
Module 1: Installation of useful tools for the course

Level of difficulty: ● ○ ○ ○
Estimated time: 5 mn

Previewing your app on a phone simulated on your computer

2. The designer has 4 main parts:

4. The catalogue of all the things (called « Components ») that we can use to put on the screens of our app. To add them on the screen, just click once on them.



1. The menu, we don't use it often, mainly to see the list of the screens of our app

We have just one screen at the moment, « F Main ».

2. The list of things on our screen (« Form » means a screen, in Codename One).

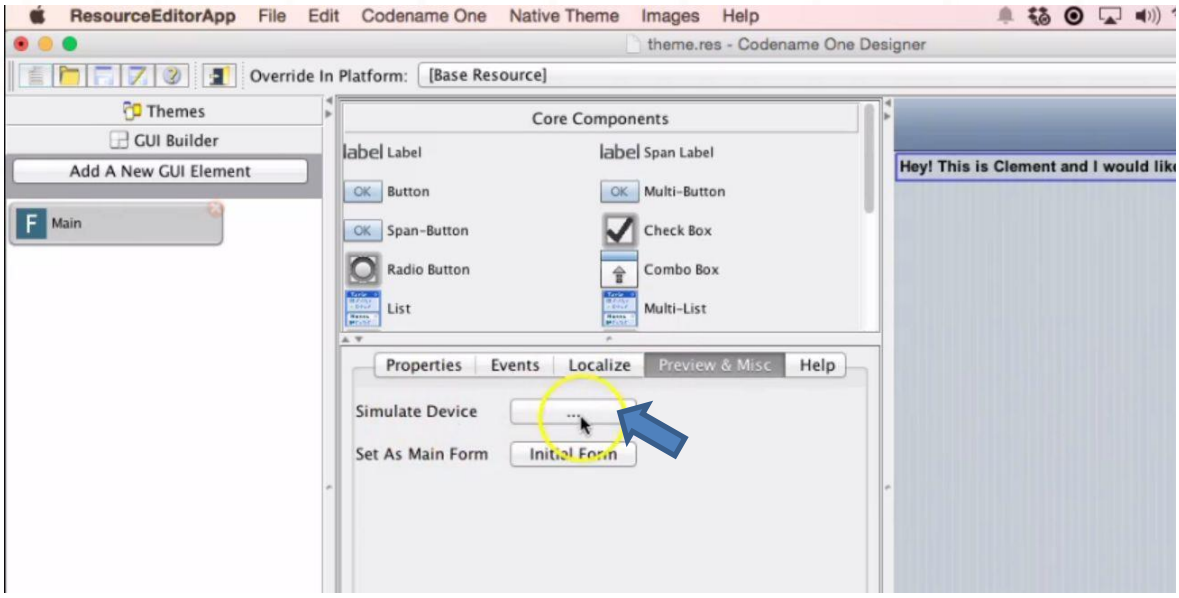
Here we just have a Label on the screen, which means a piece of text

3. A simple preview of the screen of our app that we are currently designing.

3. Click on "Preview and Misc" (red arrow in previous picture)

Previewing your app on a phone simulated on your computer

4. Click on the button on the right of “Simulate Device”.



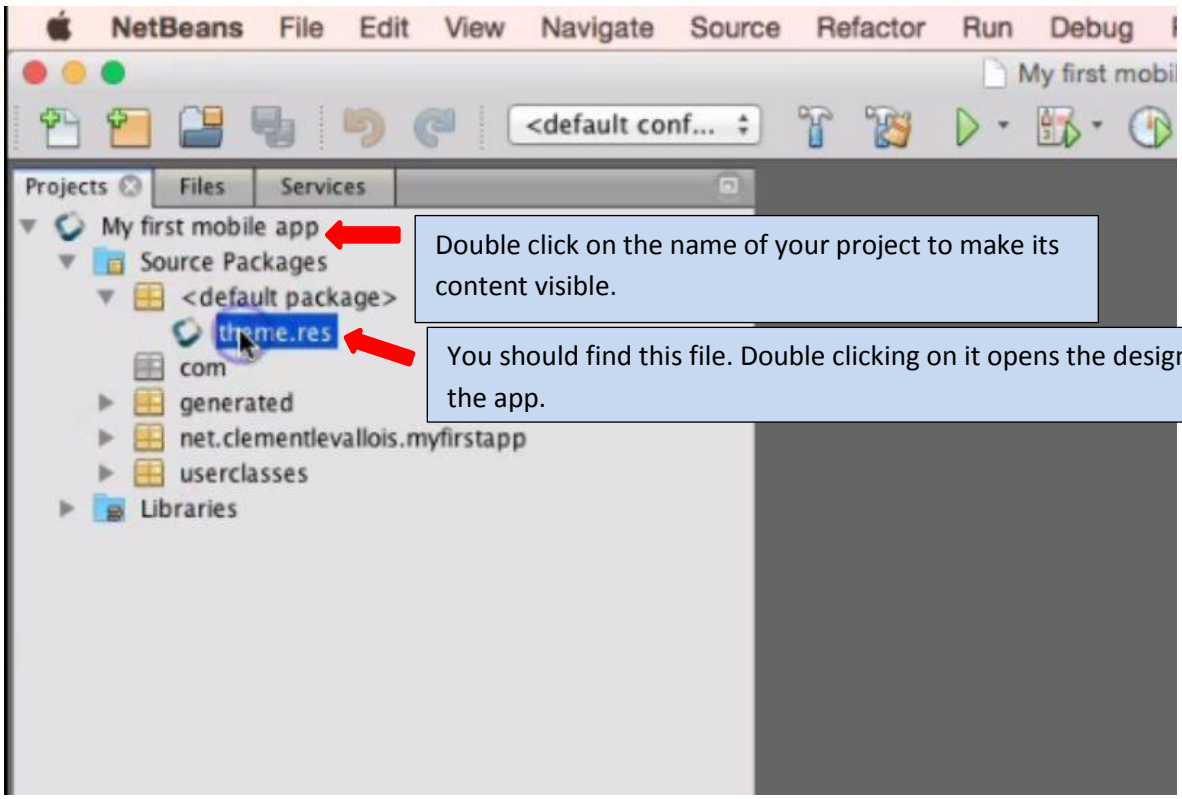
5. A simulator of a phone Appears.
   Use the menu “Skins” to change the brand of the phone
   Use the menu “Simulate” for various tests. “Rotate” is useful to put the phone in portrait or landscape mode.

Module 2: How to add text, pics, links etc. to your app

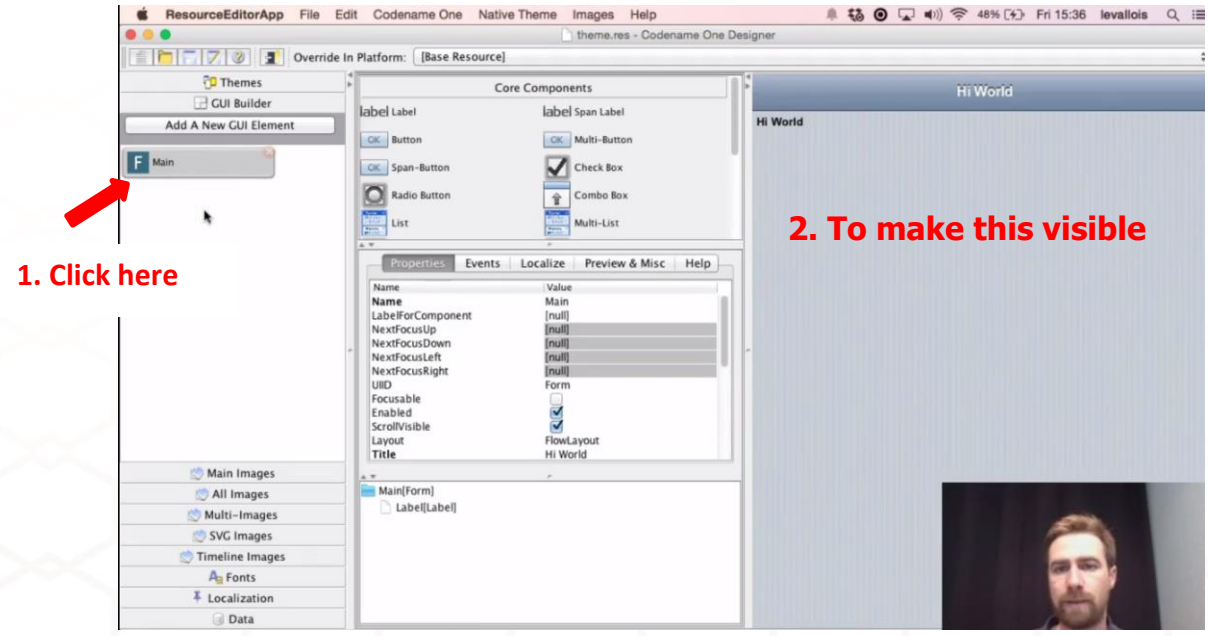
Level of difficulty: ● ○ ○
Estimated time: 5 mn

How to add a link to a webpage

1. Open the designer by double clicking on the file « theme.res » :



2. Make sure your screen is visible by clicking on « F Main » on the left menu :

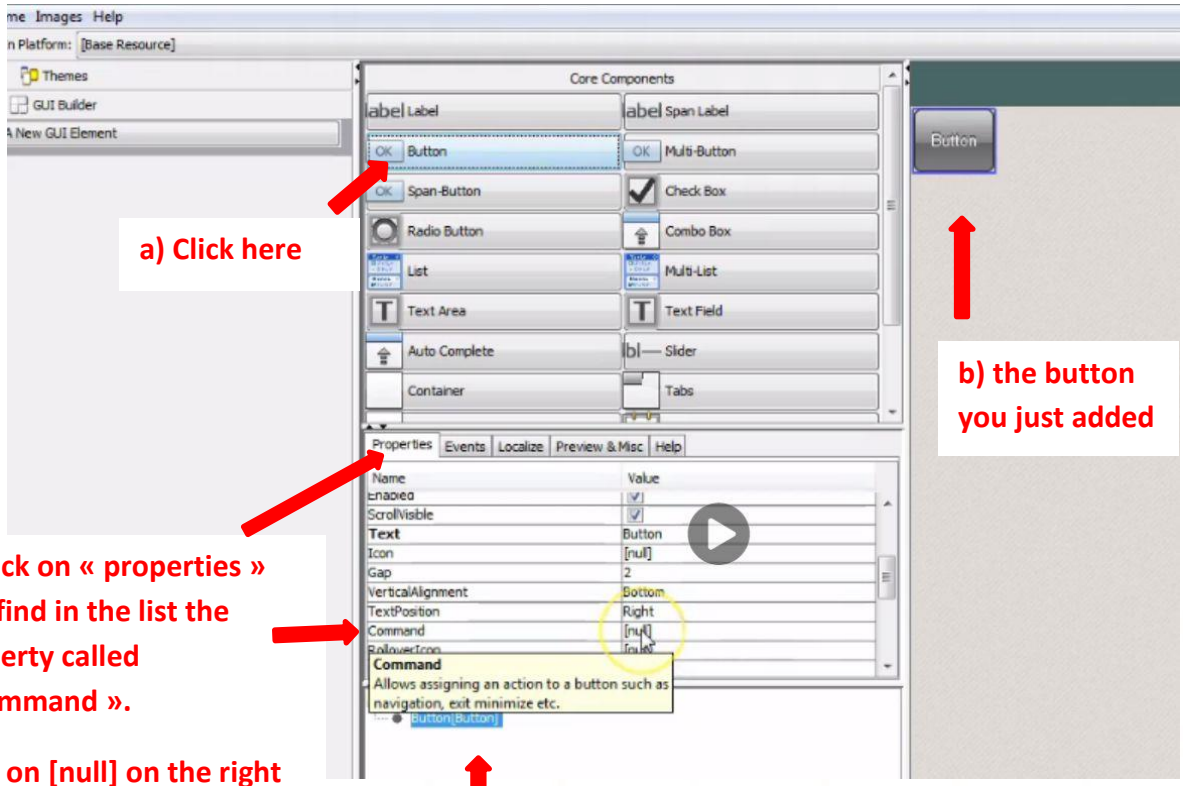


Module 2: How to add text, pics, links etc. to your app

Level of difficulty: ● ○ ○
Estimated time: 5 mn

How to add a link to a webpage

3. To add a link to a web page
   a) start by adding a button :
   b) You can see it on screen
   c) Make sure the button is highlighted in blue in the bottom panel (means you have it selected).
   d) Then click on the tab « properties » to explore the properties of the button. We are going to attach a link to a web page to it. The property we want is « Command ». Click on the right of it, where it says **[null]**



a) Click here

b) the button you just added

d) click on « properties » and find in the list the property called « Command ».

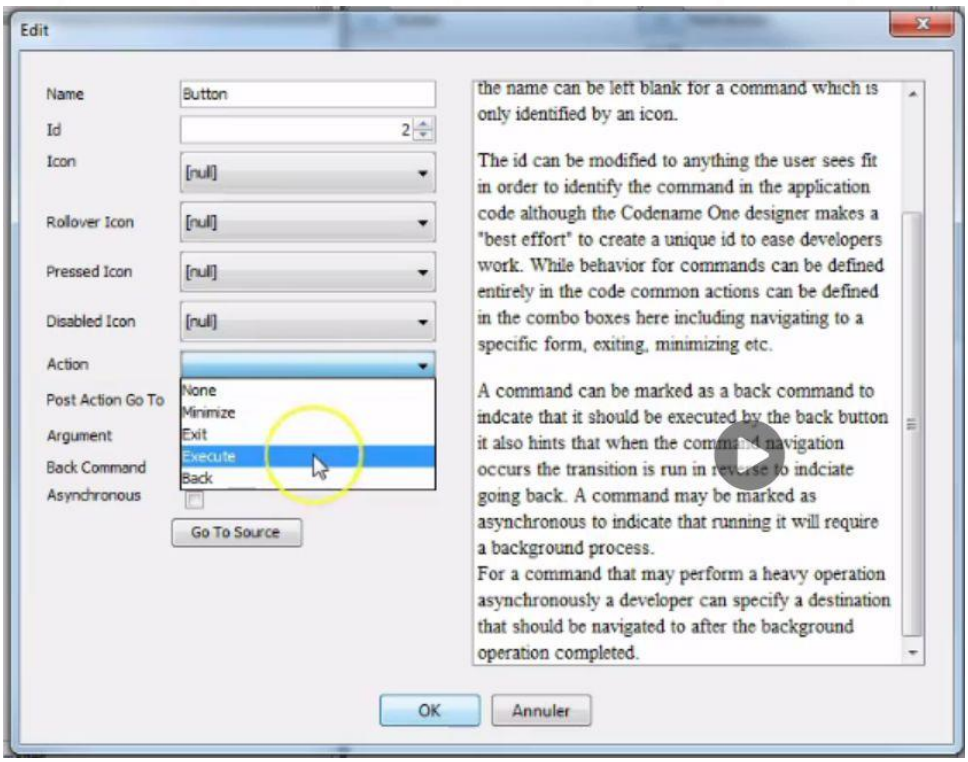
Click on [null] on the right of this line.

c) the button is also listed here.

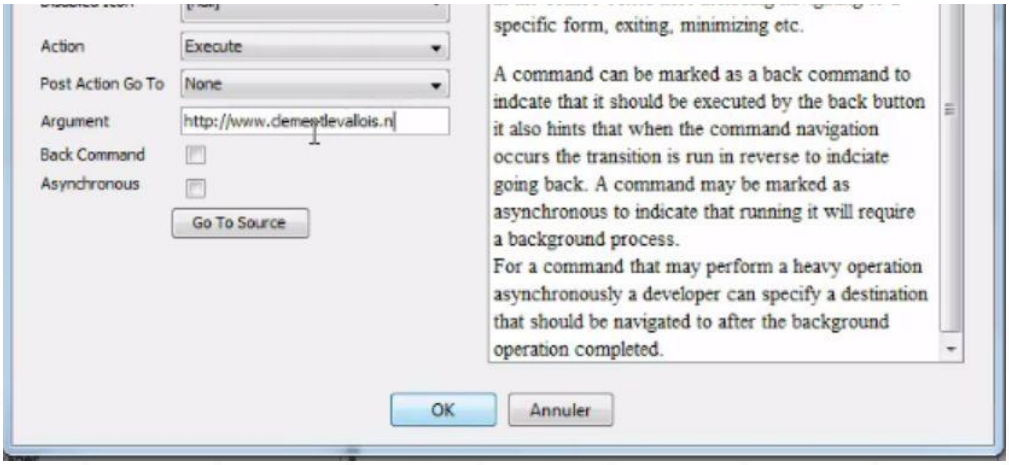
Important : make sure your button is highlighted in blue.

How to add a link to a webpage

4. Clicking on [null] opens a new window. This is where we choose which web page the button is going to open, when a user clicks on it.
   - In the menu « action », choose « Execute »



5. And then write the address of the web page you want to open in the « Execute » text field : **Do not forget to write « http:// » at the beginning of the address.**



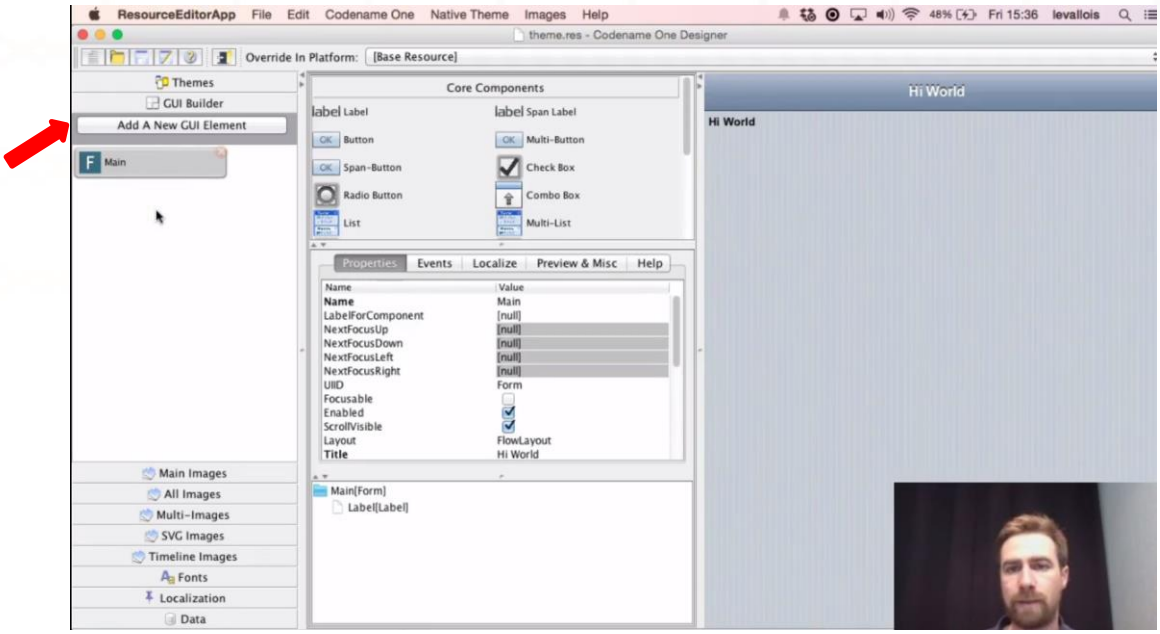
6. Click on « OK » to confirm That's it, your button will open a web page when clicked !

Module 2: How to add text, pics, links etc. to your app

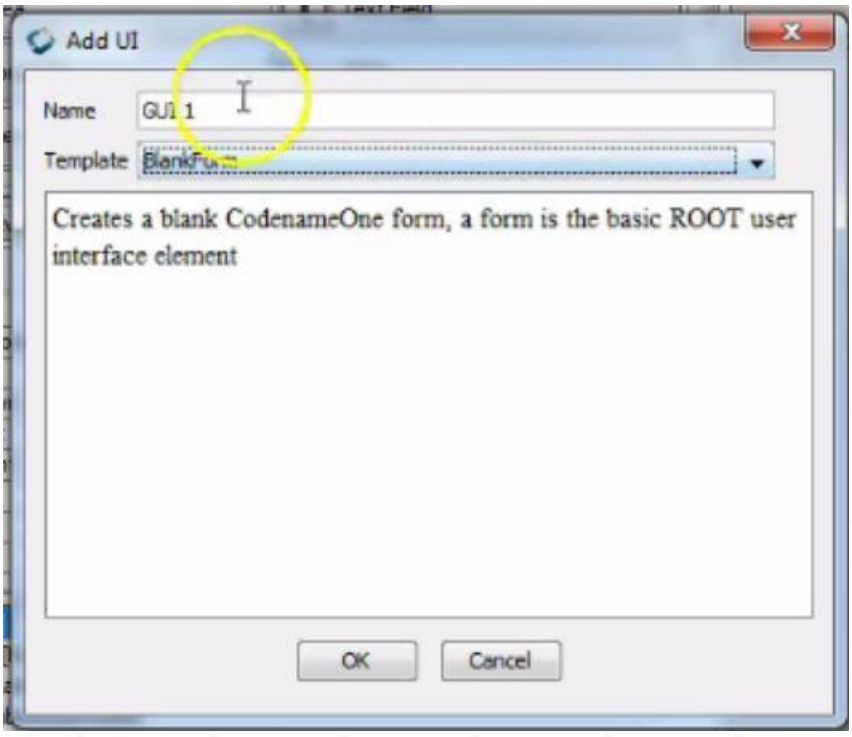
Level of difficulty: ● ○ ○ ○
Estimated time: 5 mn

How to add a screen to the app, and how to navigate between screens

1. You have the designer open. Click on « Add a new GUI Element » :

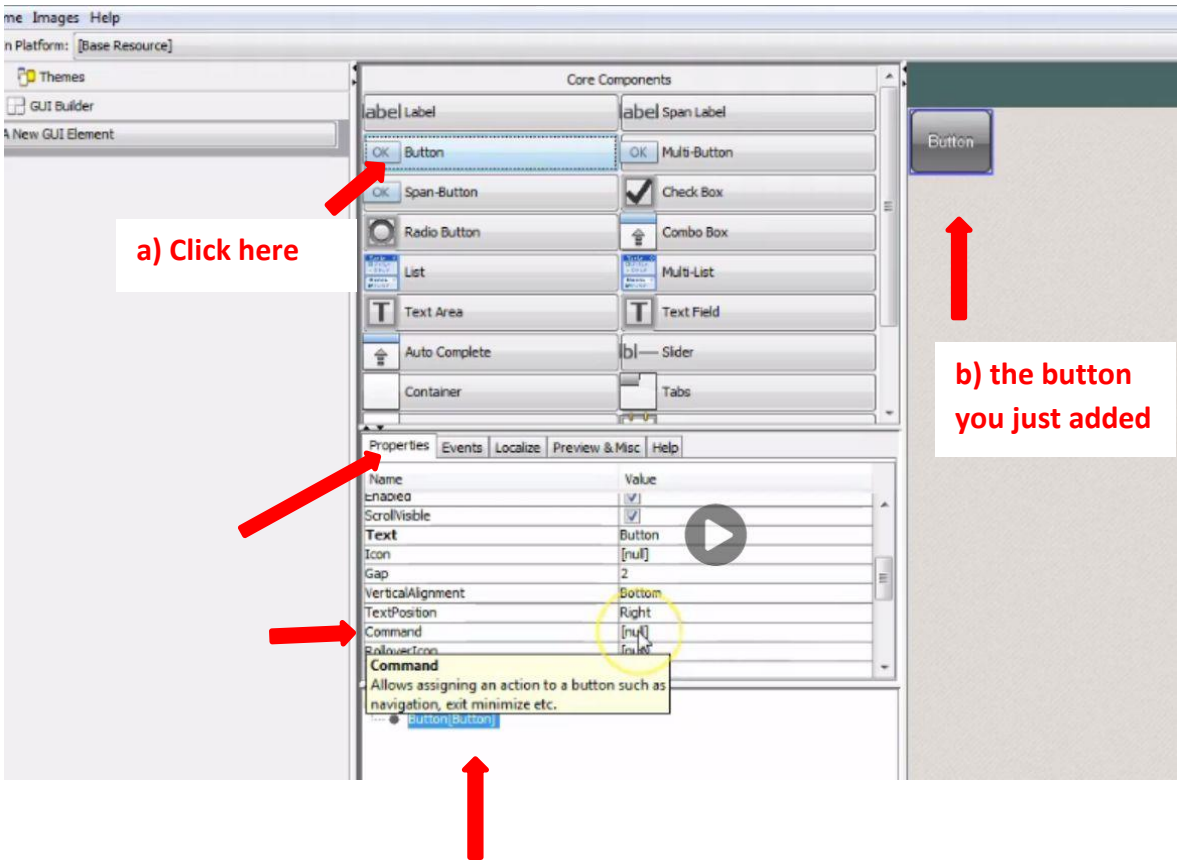


2. This opens a small window. Delete « GUI 1 » and choose a name for your new screen and click « OK »:

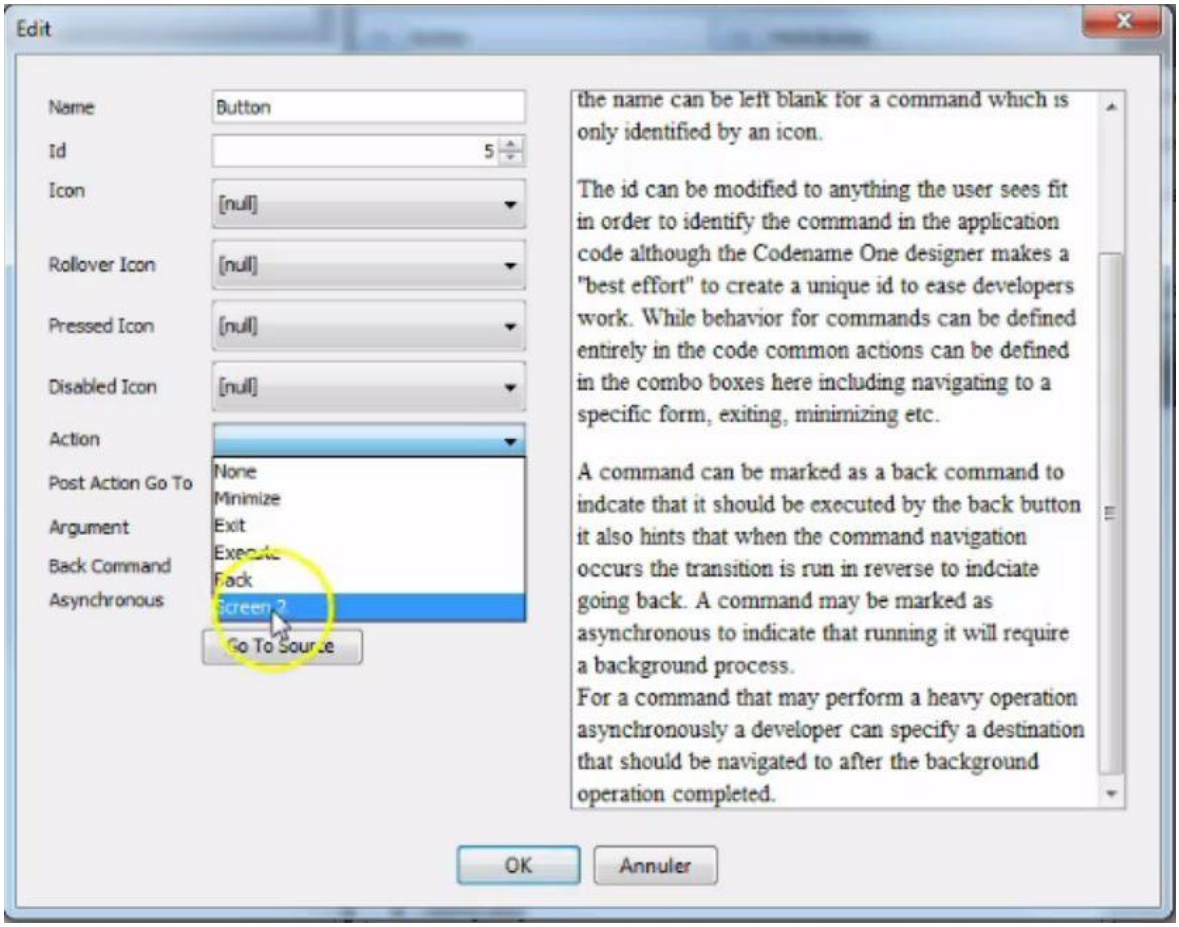


3. You have now a second screen listed in the menu on the left.
   a) Click on the screen « F Main » to make sure we come back to our first screen.

4. Now we are going to add a button to our first screen. When clicked, the app will show the second screen.
   a) Click on « Button » to add one to our first screen.
   b) You can see it on screen
   c) Make sure the button is highlighted in blue in the bottom panel (means you have it selected).
   d) Then click on the tab « properties » to explore the properties of the button.. The property we want is « Command ». Click on the right of it, where it says **[null]**



5. Clicking on [null] opens a new window. This is where we choose which screen is going to be shown, when a user clicks on it:
   ➔ In the menu « action », simply choose the name of your second screen.



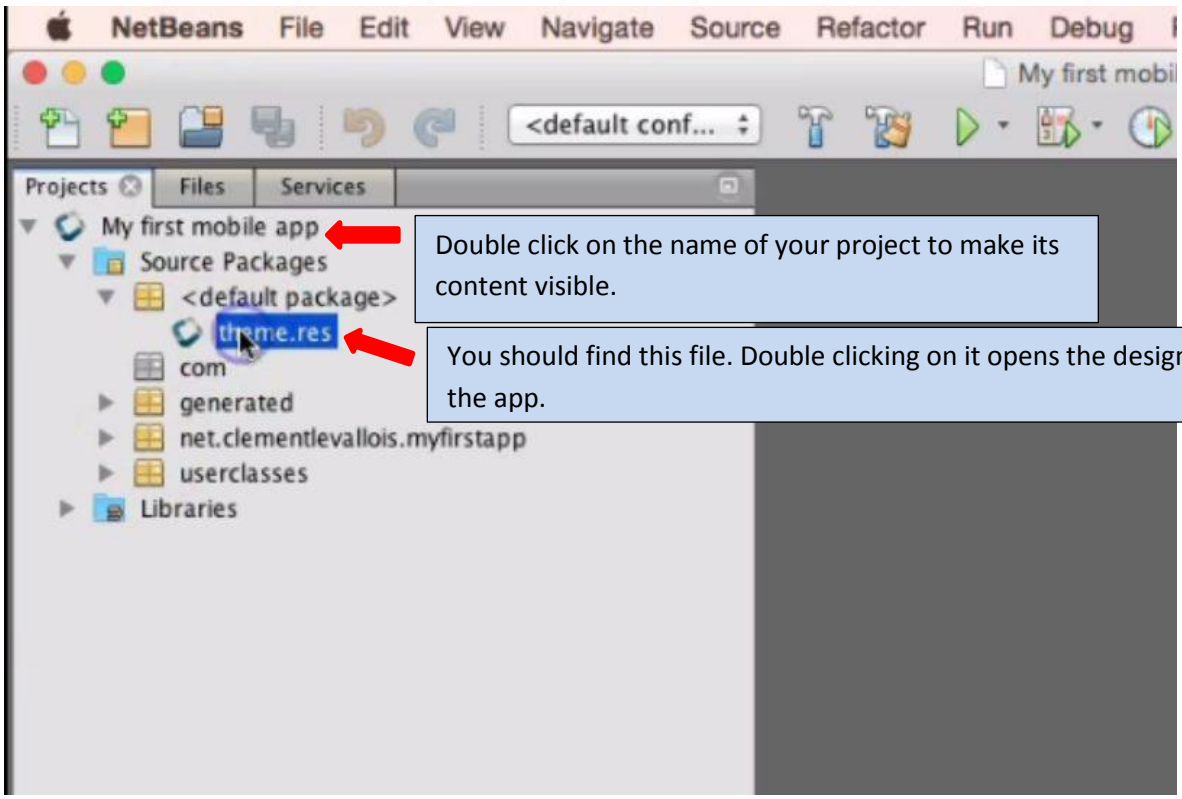
6. Just click on OK to confirm. That's it, clicking on your button will move the user from to the second screen ! (try it in the preview !)

Module 2: How to add text, pics, links etc. to your app

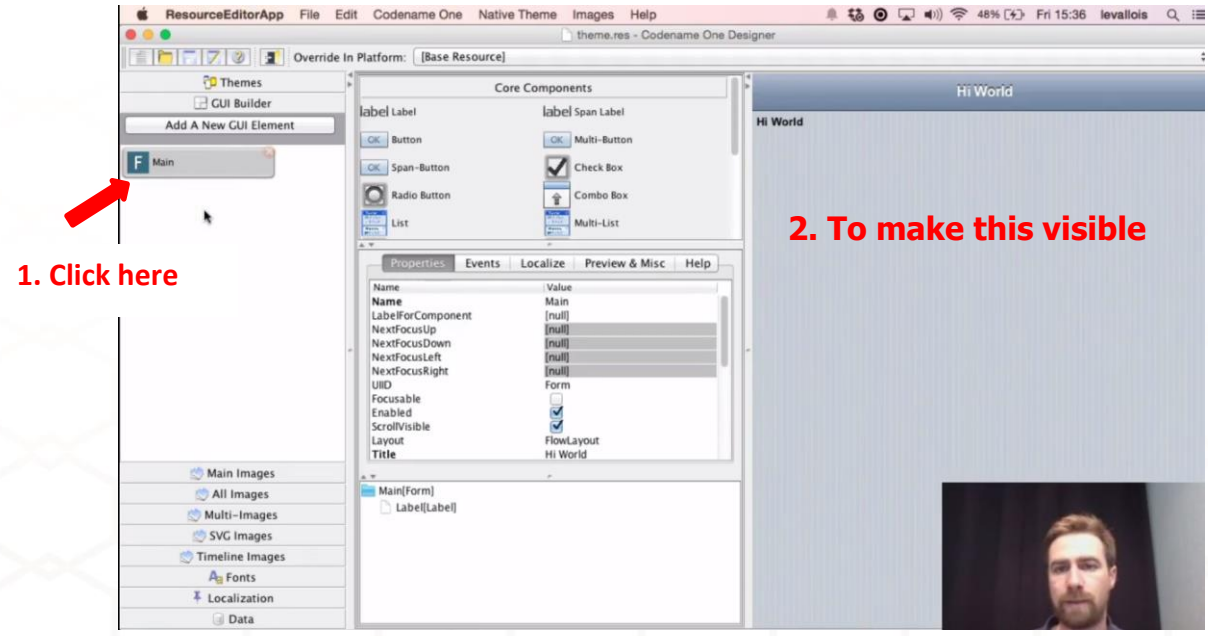
Level of difficulty: ● ○ ○
Estimated time: 5 mn + download.

How to add text to your app

1. Open the designer by double clicking on the file « theme.res » :



2. Make sure your screen is visible by clicking on « F Main » on the left menu :

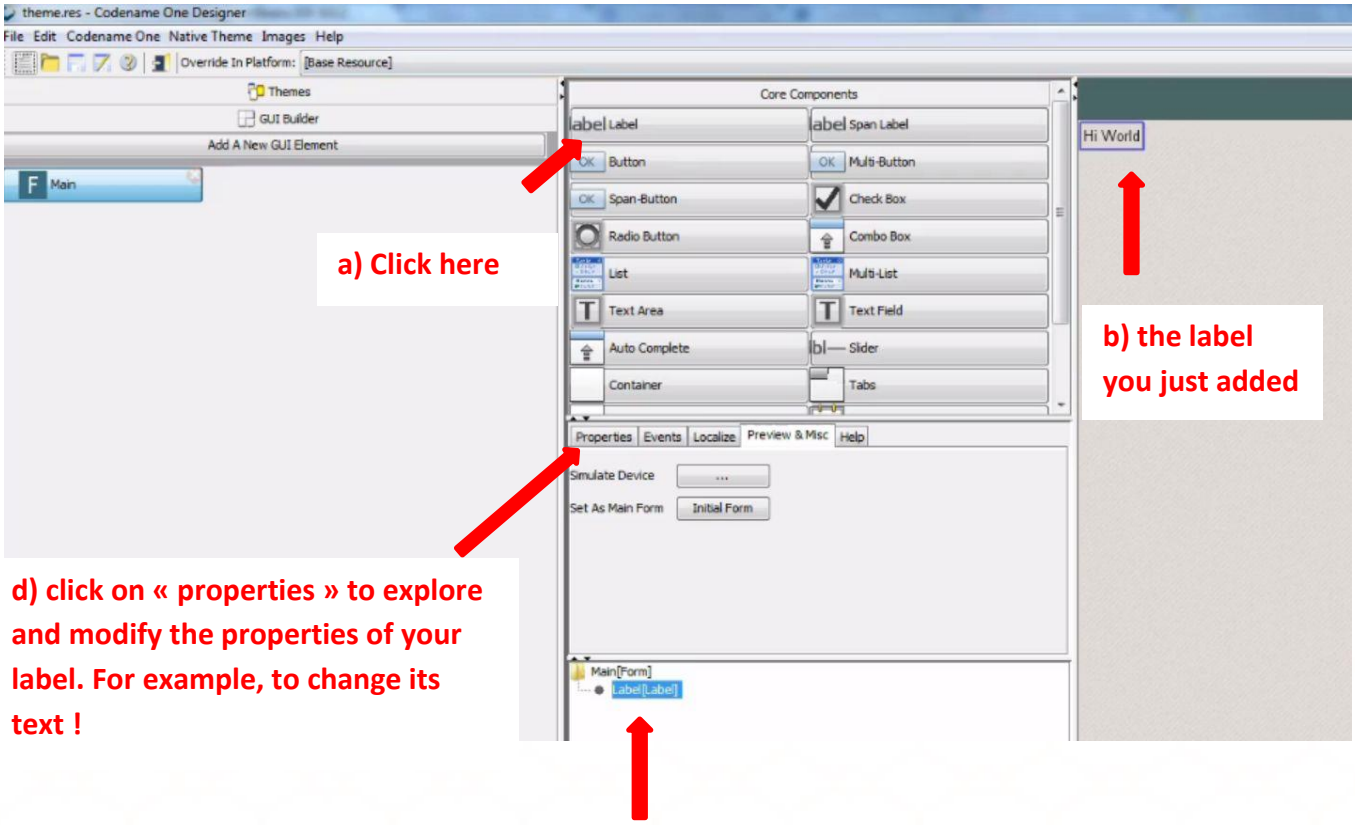


Module 2: How to add text, pics, links etc. to your app

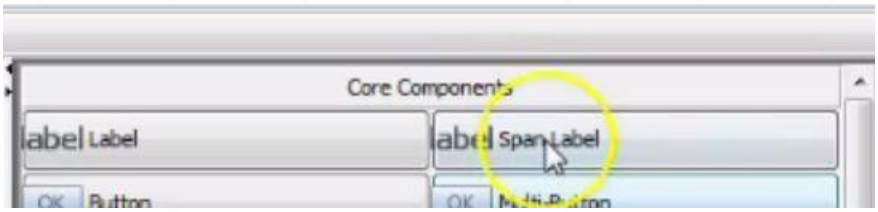
Level of difficulty: ● ○ ○
Estimated time: 5 mn + download.

How to add text to your app

3. To add some text :
   a) Click on « Label » in the zone « Core Components ». This places a bit of text on your screen.
   b) You can see the label on the screen.
   c) You can also see the label in the bottom panel.
   d) When the label is highlighted (in blue) in the bottom panel, this means you can access the properties of the label in the « properties » tab just above. What are properties for ? They list all the parameters of your label. For example the text of your label. Just find the property that says « text », change the text here, and your label on screen will have a different text.



4. Label or SpanLabel ?
   You will see that we can add a Label, or a « SpanLabel » just on the right :



What is the difference between the two ?

➔ The text of a Label gets cut if it is too long for the screen

➔ The text of a SpanLabel breaks nicely into several lines when it is too long for the screen.

Something you should be careful about :

When we add a SpanLabel on screen (by clicking on it), nothing appears on screen. This is because by default, the text of a SpanLabel is empty, so we see nothing on screen.

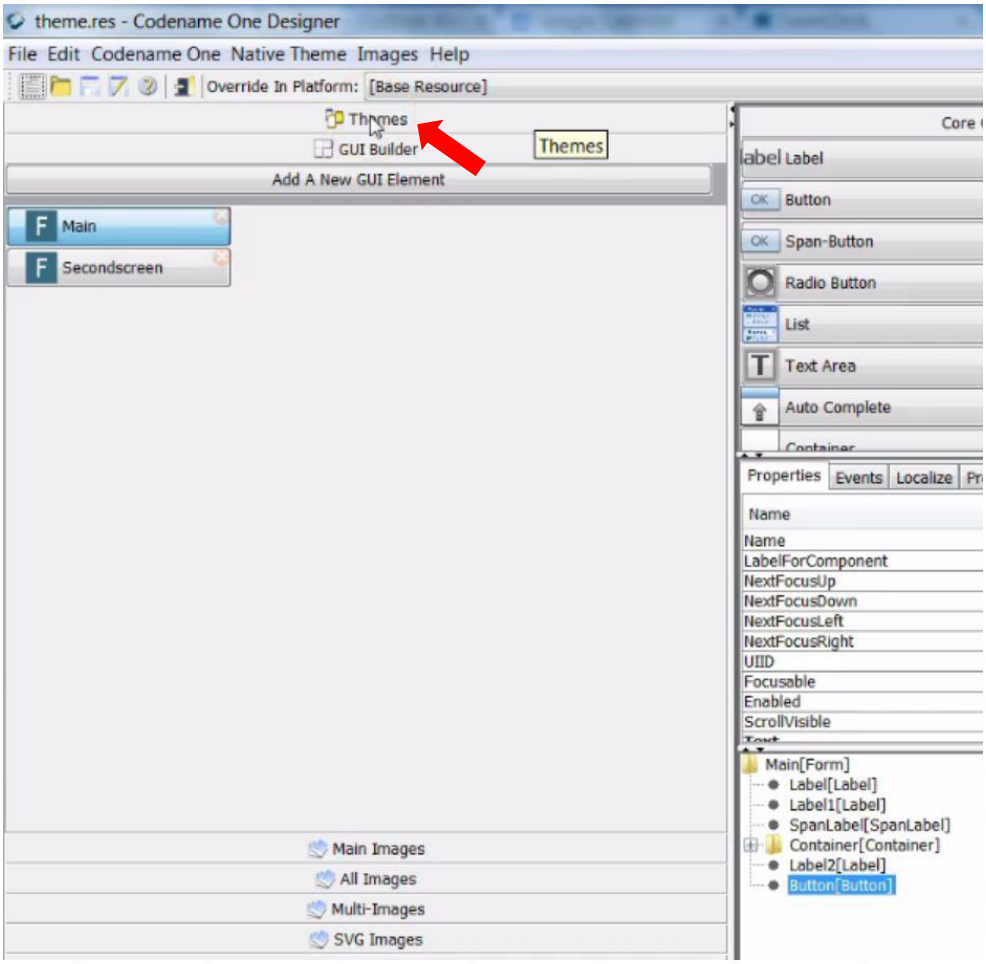
But you know that your SpanLabel is indeed on screen because you can see it listed in the bottom panel. Just select it with the mouse (highlight it in blue), and then you can go to its properties, search for « text » in the properties, and change the text for something you want.

Module 3: Designing your app

How to style components (labels, buttons, pics…)

1. The designer is open, and we have a screen with labels, buttons, etc.
   How do we change the appearance of each component on screen? (color, size, background, shape).
   The general principle to do that is:
   - We add a new style to the Theme of the app. We define this style as we want (color, etc.)
   - We apply this style to the Label, Button… that we want.

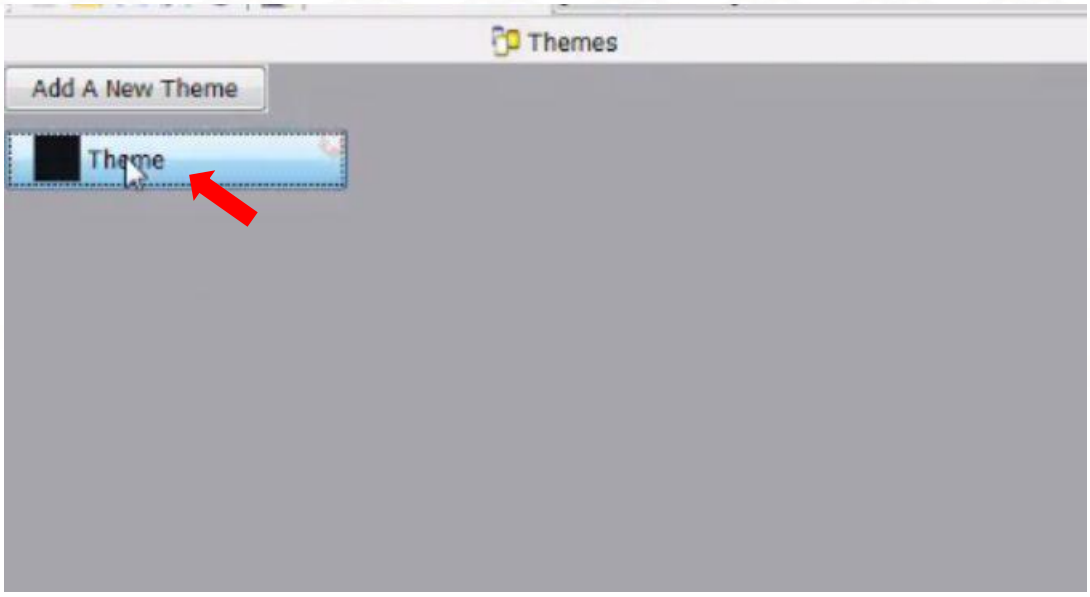
   Let's do it. In the menu on the left, click on "Themes":



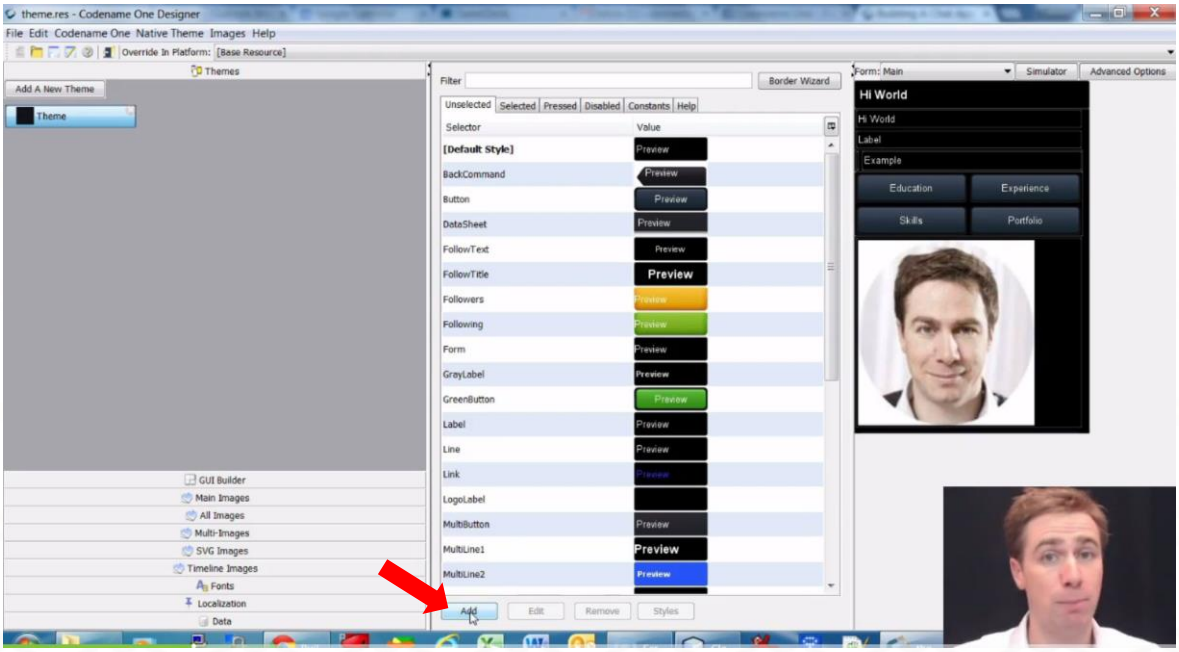
Module 3: Designing your app

How to style components (labels, buttons, pics…)

2. This show this screen, where you should click on “Theme”:



3. This opens a window where your see all the details of the theme of your app. Click on “add” at the bottom to add a new style:

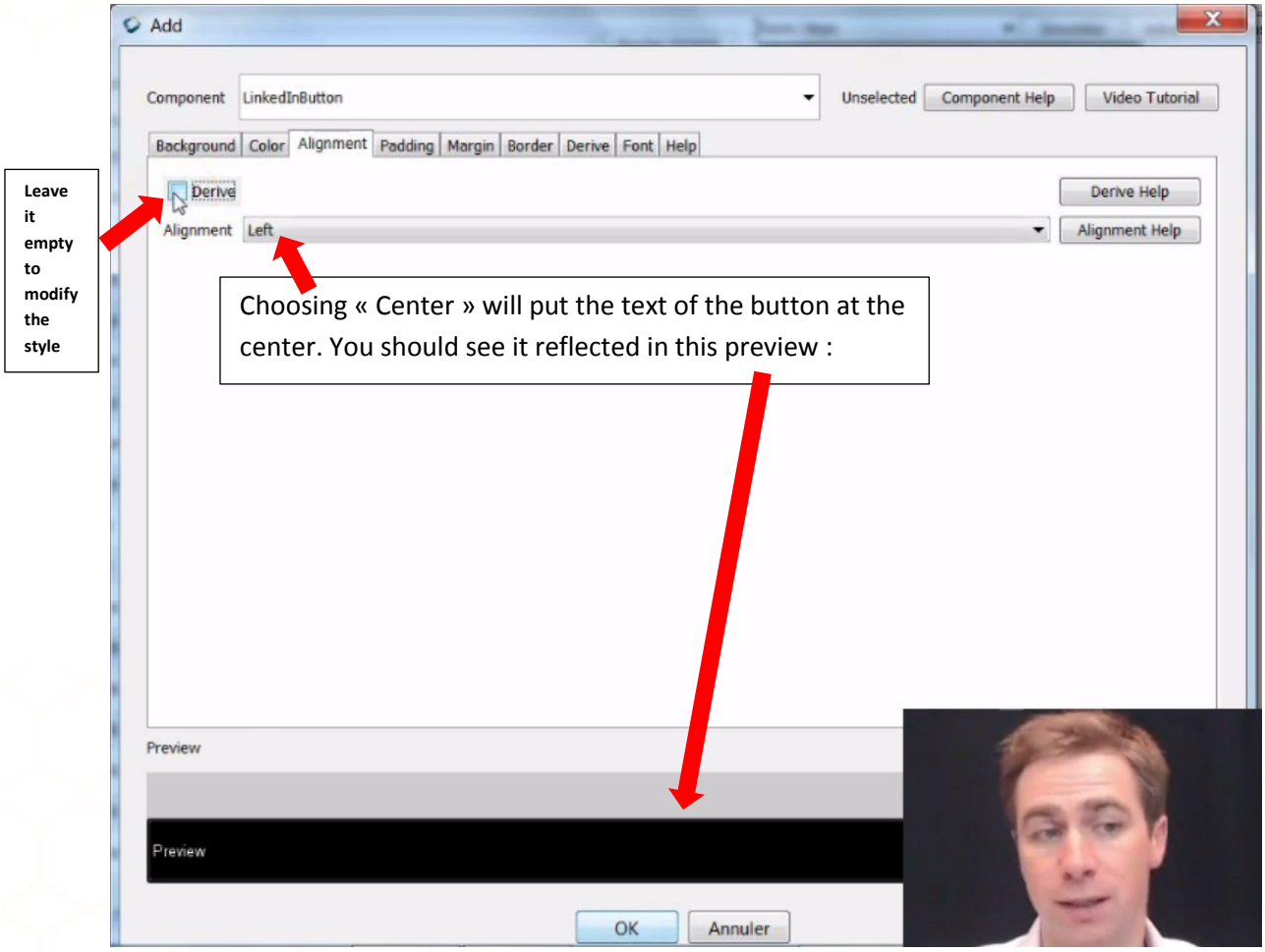


Module 3: Designing your app

How to style components (labels, buttons, pics…)

4. A window opens. On top in the field “Component”, you should give a name for your style. In this example I show how to create a new style for a button which would open my LinkedIn profile, so I call it “LinkedInButton”.
   Note: if you just give a name to your style and click “OK”, your style will not be saved. You need to change at least one thing in the style (color, background… whatever).

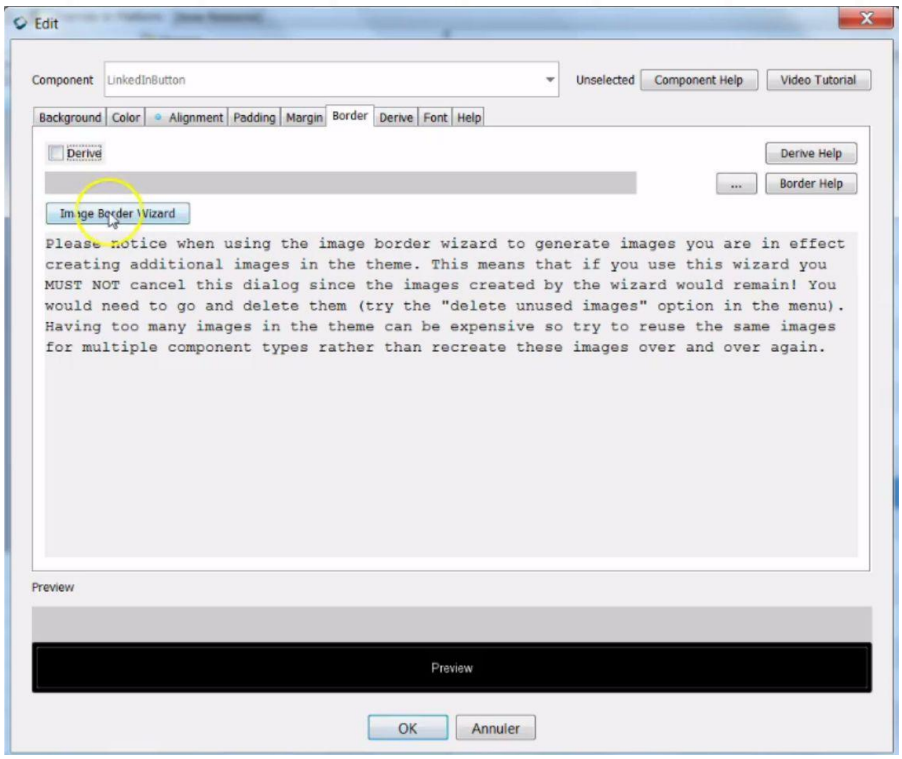
The principle is simple: untick the box “Derive” (remove the mark, as shown on the pic) to apply the changes you want in each category: background, color, font, margins, etc.



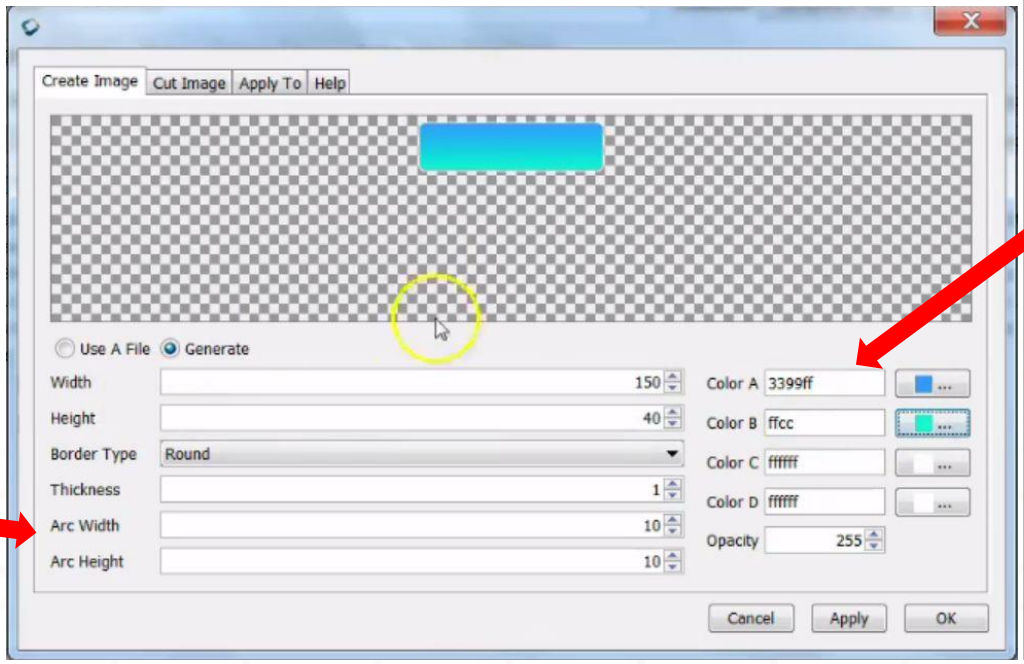
Module 3: Designing your app

How to style components (labels, buttons, pics…)

5. A special tab is “Border”. There, untick “derive” and click on “Image Border Wizard”. This opens a new window, where you can change the shape of the component you are designing.



6. Explanations of the settings available in the Image Border Wizard:



Arc width and height: Try values of 25 to get a button with round corners.

Color A: color of the top of the button

Color B: color of the bottom of the button

Color C: color of the top of the border

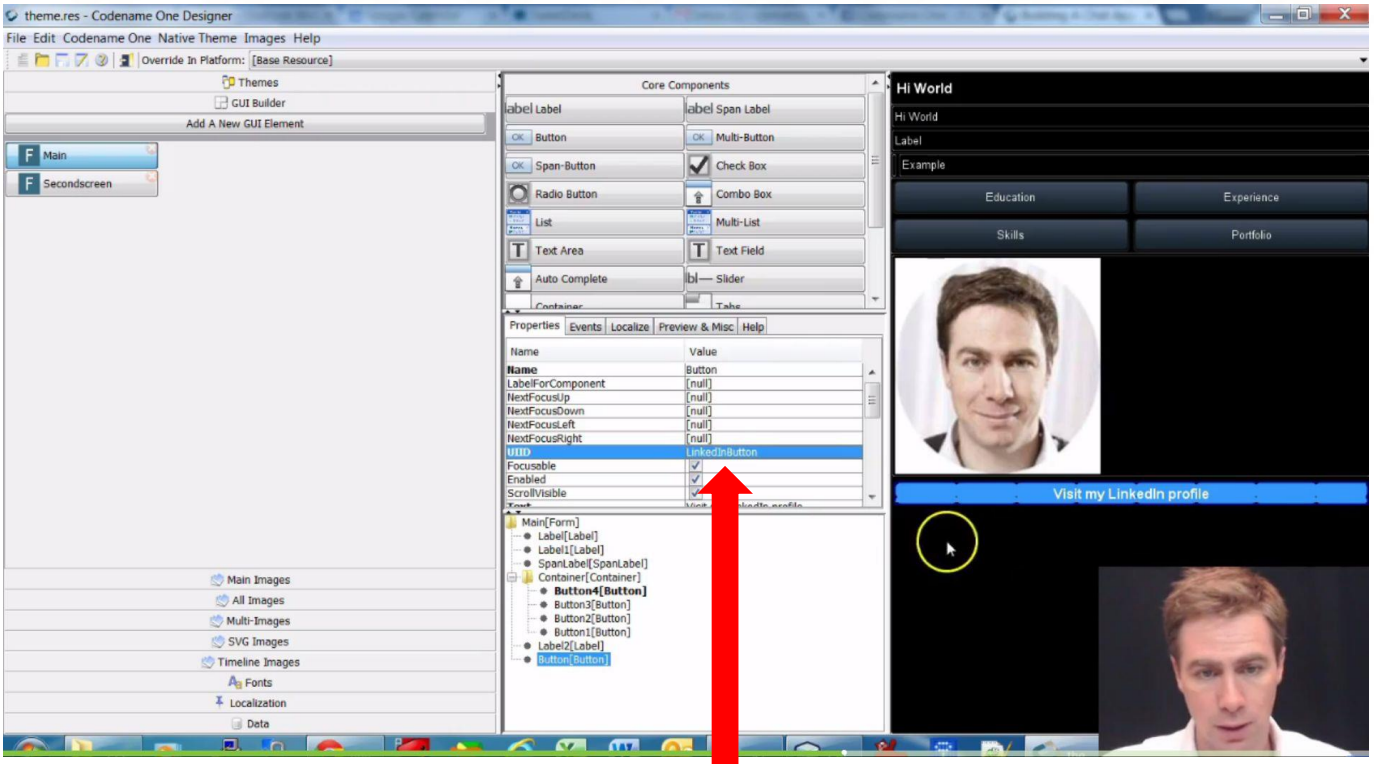
Color D: color of the bottom of the border

Module 3: Designing your app

How to style components (labels, buttons, pics…)

7. When we have finished defining the style of our buton, label or else:

We come back to the GUI Builder, and we put back the screen of our app on display. We select the button we want to style:



… and in the properties of the button, we find the property « UUID ».

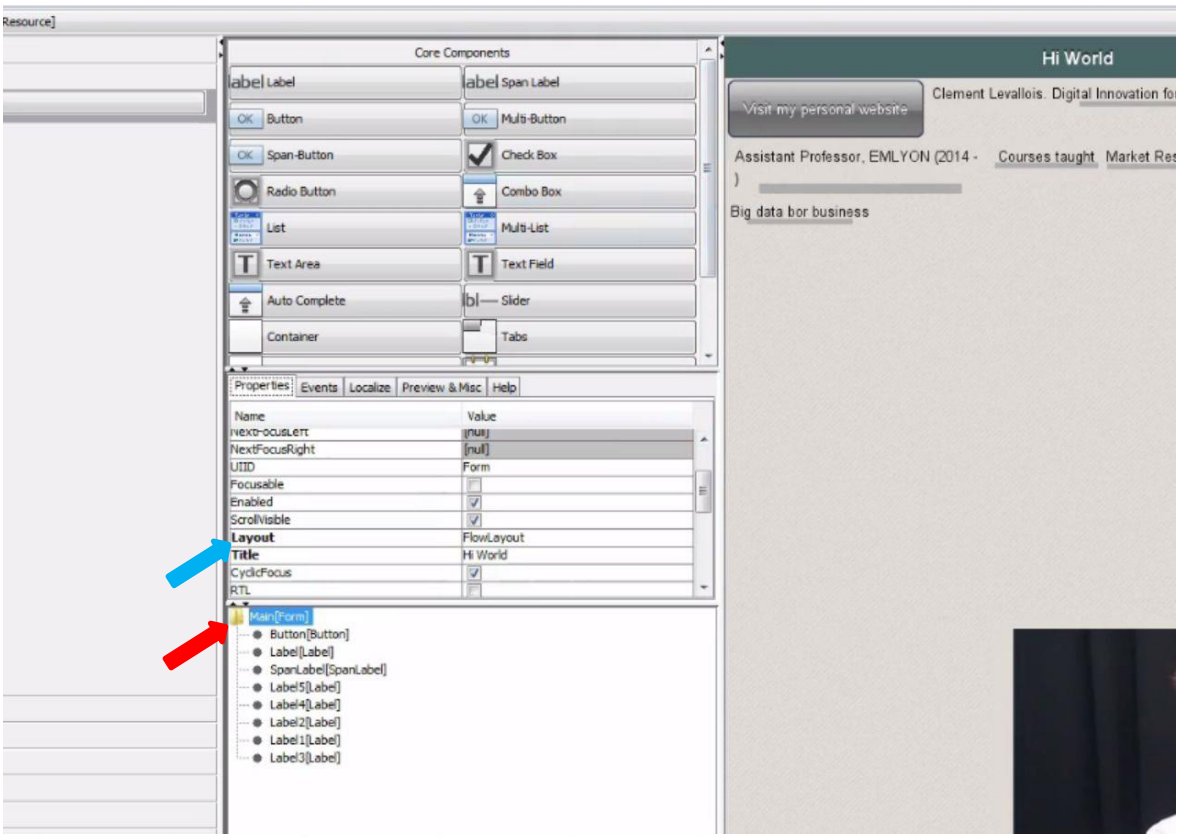
On the right of this property, there is a drop down menu. We select the name of the style we just created, and our button gets the new style.

Module 3: Designing your app

Level of difficulty: ● ○ ○
Estimated time: 10 mn

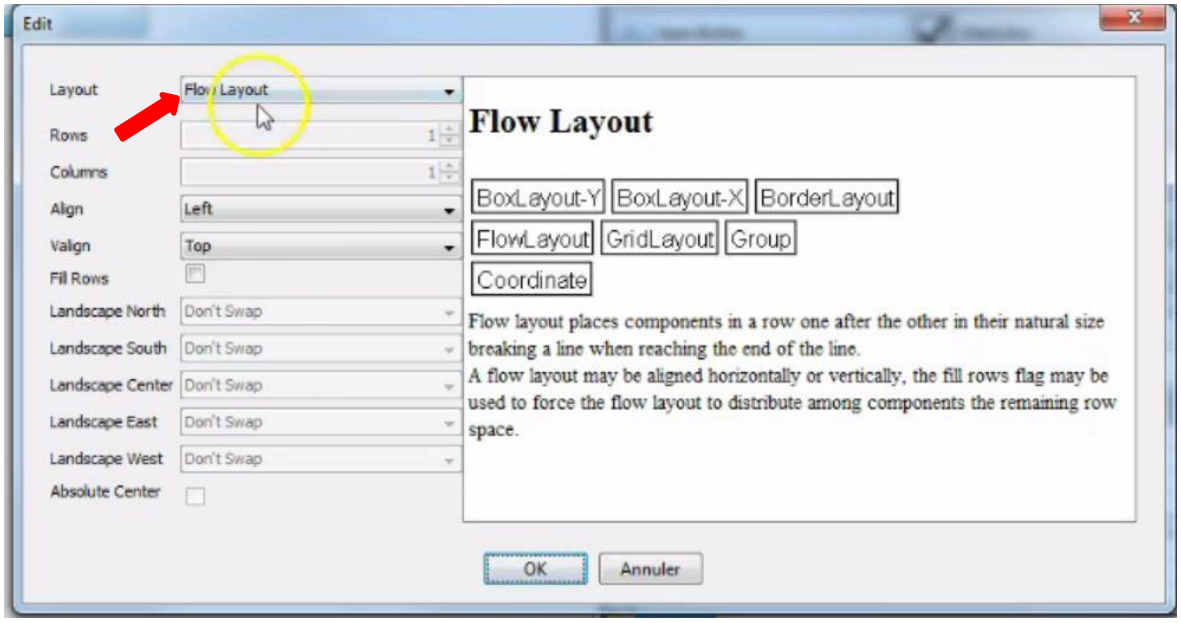
How to place things where we want on a screen: layouts

1. The designer is open, and we have a screen with labels, buttons, etc.
   The screen is not organized : any label or button we have added is just placed in line, next to each other from left to right.
   We can change that.
   For example, we might want to have evetything lined up vertically. Let's do it. Select the screen in the bottom panel. This is the first line in the bottom panel (**red arrow**) :



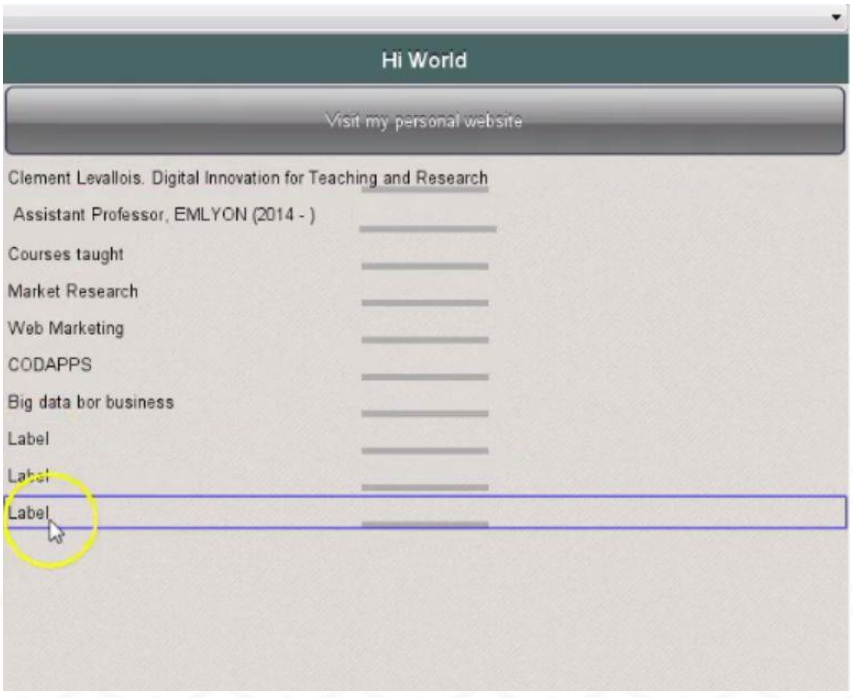
2. Go to the property tab and find the property « Layout » (**blue arrow** in the picture above). Click on « Flow Layout » on the right.

3. This opens a new window. It looks complicated, but to change how the screen is organized you just need to click the menu « Layout » :



In the drop-down menu, choose « Box Layout Y », then close the window by clicking on “OK”.

This will change the place of all things on the screen: they are now placed vertically, from top to bottom :



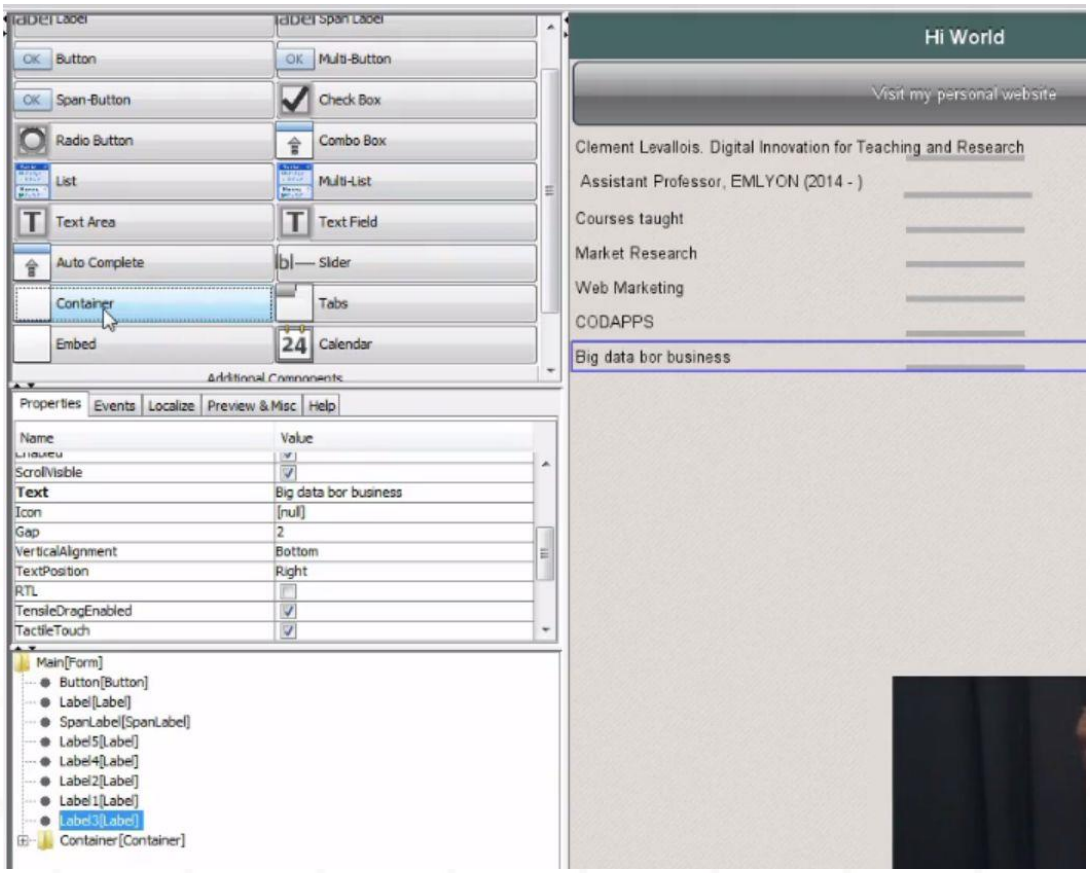
How to place things where we want on a screen: layouts

4. So we have a way to change how the screen is organized. But this is not good enough: we would prefer to have <u>some</u> regions of the screen organized in one way (for example : vertically), and some other regions of the screen where things are placed in a different way (for example : in a grid).

   How do we give different layouts to different regions of the screen?
   The solution is:
   - We put in a box the things we want to place in a certain way.
   - We apply a layout (vertical, grid…) to this box.
   - Other things that we want to organize in a different way: we put them in a different box, and we apply a different layout to this box. Simple!

The boxes that can contain Labels, Buttons, pictures, etc… and they can have a layout just for them. Boxes are called **<u>Containers</u>**. So we add containers to our screens, put stuff in them, and apply layouts to them:

5. Adding a Container :



1. Click here

2. You will see the container added to your screen

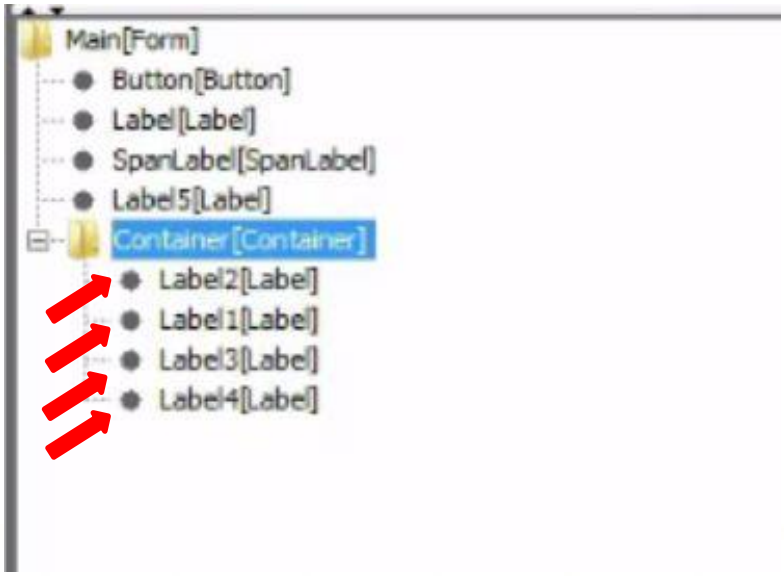
6. So, in this container we'll put things we'd like to arrange in a grid. A grid is a layout where things are place in rows in columns. If we choose a 2 x 2 grid layout, things will be placed like that:

So we'll place 4 things in the container, and they should become placed like the grid above. To put things in a container, we just go to the bottom panel and use the mouse to drag things (Labels, Buttons, etc.) inside the container (here watching the video for this lesson would help see how to drag things in a container).

You know that a Label (or a Button, etc.) is in the Container when it appears slightly offset to the right :

Four Labels inside a container.

We know they are inside it because they are offset to the right, « nested » in the container.



7. Then we select the Container (highlight it in blue, like in the picture above),
   - go to its properties.
   - Select « Layout », click on « Flow Layout »
   - In the window that opens, select « Grid Layout »
   - Make sure we put 2 in « Rows » and 2 in « Column », then click on OK.

As a result, we should now have only these 4 labels in the container organized as in a table. The rest of the screen stays vertically aligned.

Conclusion: using containers, we can achieve exactly the design we want for our app!

Module 4: Testing your app on a phone and distributing it

Level of difficulty: ● ○ ○ ○
Estimated time: 45 mn

How to test your app on a Windows phone

NOTE 1: this tutorial needs to be executed from a PC. It won't work with a MAC computer. You will also need a USB cable to plug your phone to the USB of your computer (the cable you use to recharge your phone via USB is the one you need).

NOTE 2: this tutorial is to install your app on Windows Phone version 8.1, which is the latest version of Windows Phone in Summer 2015. To check which version is your Windows Phone, see this link: https://www.windowsphone.com/en-us/how-to/wp8/basics/which-version-of-windows-phone-do-i-have

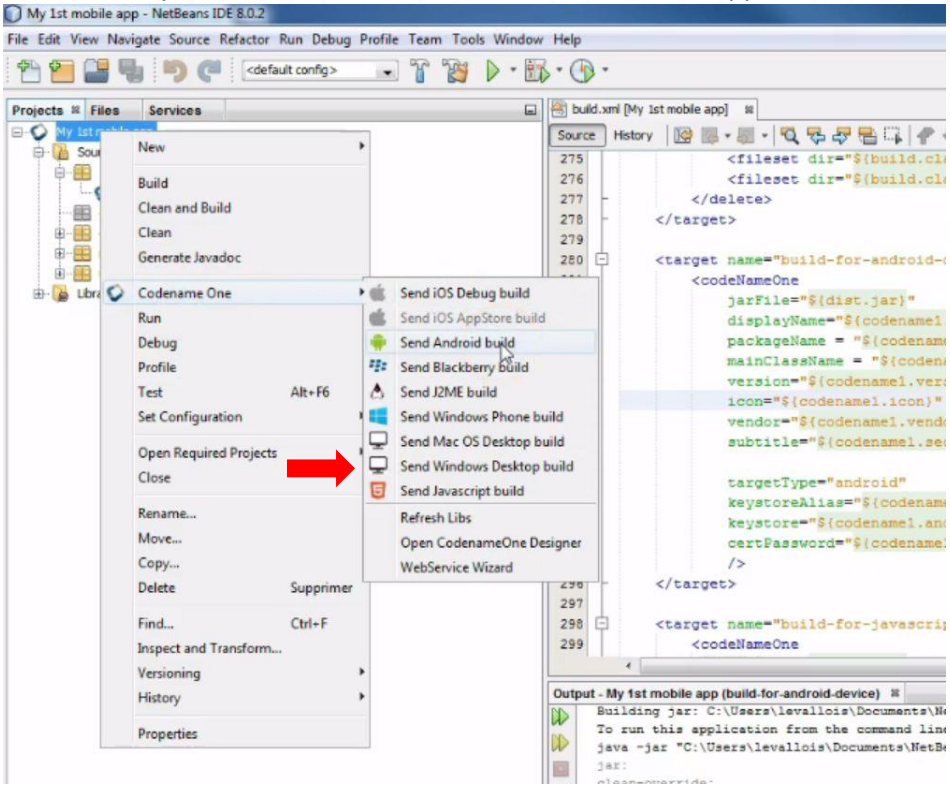
1. You need first to create a free account with Codename One (the tool we use to create apps).
   To do so, visit their website at www.codenameone.com and select "Sign up".
   Choose the free version. You are not obliged to choose to receive the newsletter.
   Please open the confirmation email you will receive and click on the link in it to confirm the creation of your account.

2. You must also use a Microsoft account. You have a Microsoft account if you have a Hotmail, MSN or Live address. If not, you can create a Microsoft account for free here: https://www.microsoft.com/en-us/account

3. You then need a developer account for Windows. It costs around 20$, this is a one-time fee and you can then have as many apps as you want for the Windows Store. Use the email address of your Microsoft account to register for a developer account, here: https://dev.windows.com/en-us/programs/join

4. Ok, we are ready to download the only tool we need, "**Visual Studio Community 2013 with Update 5**", here: https://www.visualstudio.com/en-us/downloads#d-express-windows-8?CR_CC=200395106

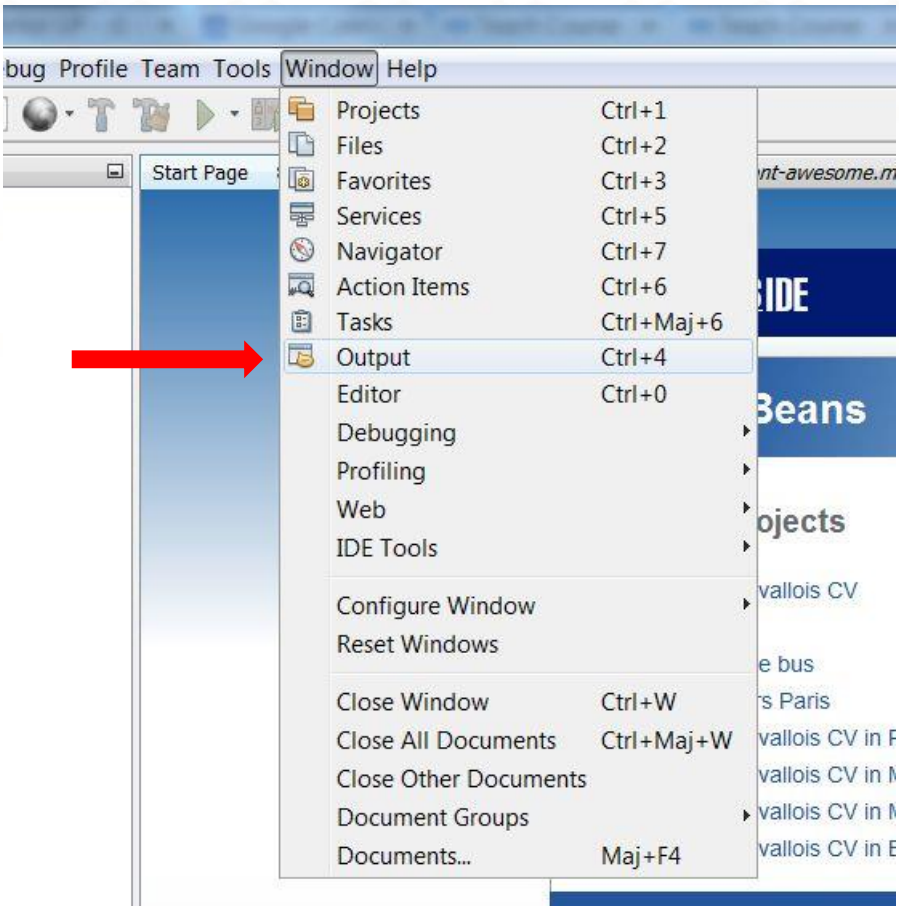
5. We are ready to create the Windows Phone version of our app. In NetBeans, right click on



6. This starts the creation of package ready to be sent to Codename One. One alert windows will open, asking for the login and password of your Codename One account (that you just created). Please enter them.

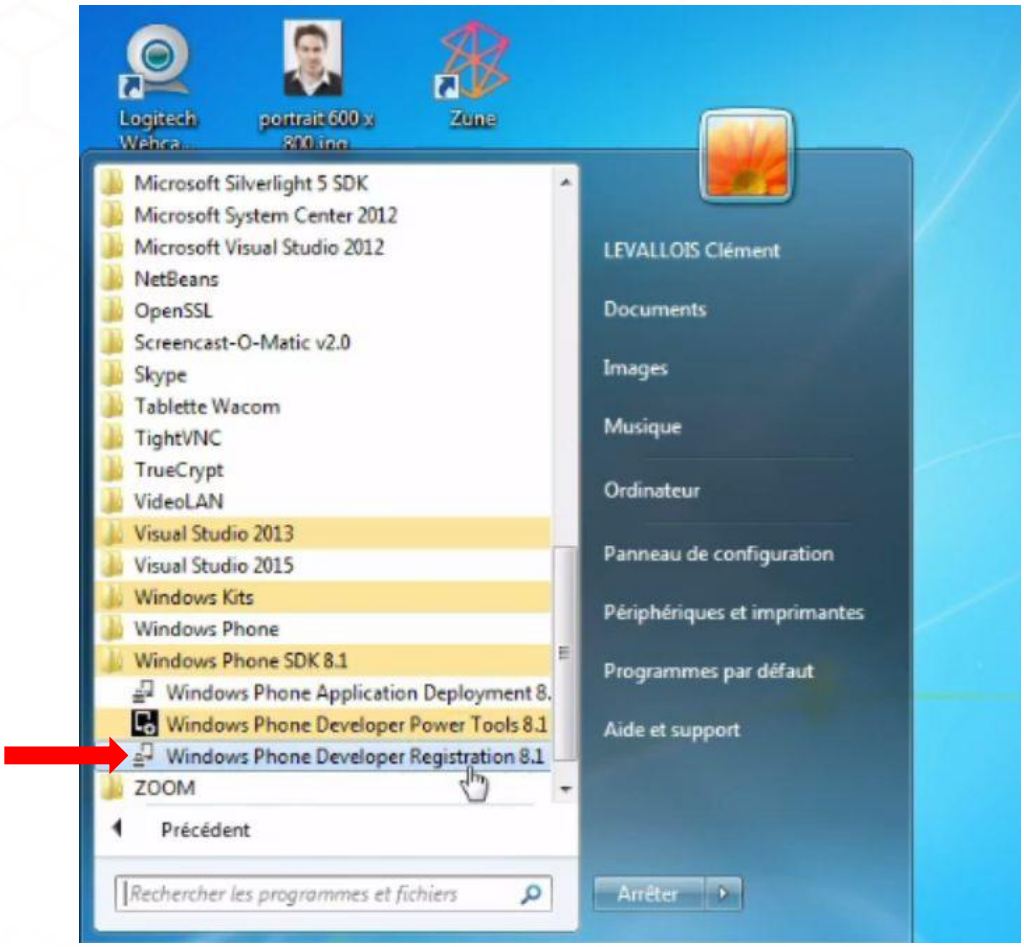
7. You know the process finished successfully when you see this in the bottom panel of NetBeans:

Note: you don't see this output window at the bottom of NetBeans? In this case, in the menu of NetBeans, select "Windows" then "Output", it should make it appear:



8. You can now go back to the website of Codename One, where your app has been sent to create the Windows Phone version of it. Login, then go the the Dashboard (in the menu on top of the page).
9. You will see the status of your app:
   - blue -> in progress
   - green -> finished, ready to be used
   - red: something went wrong, your Android app could not be created.
   -
10. While the app is being built, you can register your Windows Phone, this is the process that will allow it to install apps you develop. To register your phone, you need to use a

program we have downloaded. Find it in the Start Menu of yuour PC; as shown in this screenshot:



11. Plug your phone with a USB cable to your computer. You should unlock the screen of your phone and your phone should have an Internet connection (wifi enabled, or you have a data plan), so that the computer can detect your phone.
    When your phone is detected, you are invited to register it. Type in your Windows dev account (email and password) to confirm.

12. Almost finished! We get back to www.codenameone.com, where our Windows app must be finished creating by now (can take 5 to 10 minutes,much longer than for Android apps).

When the status turns green, click on it. A series of option appears to install your app on your Windows phone:

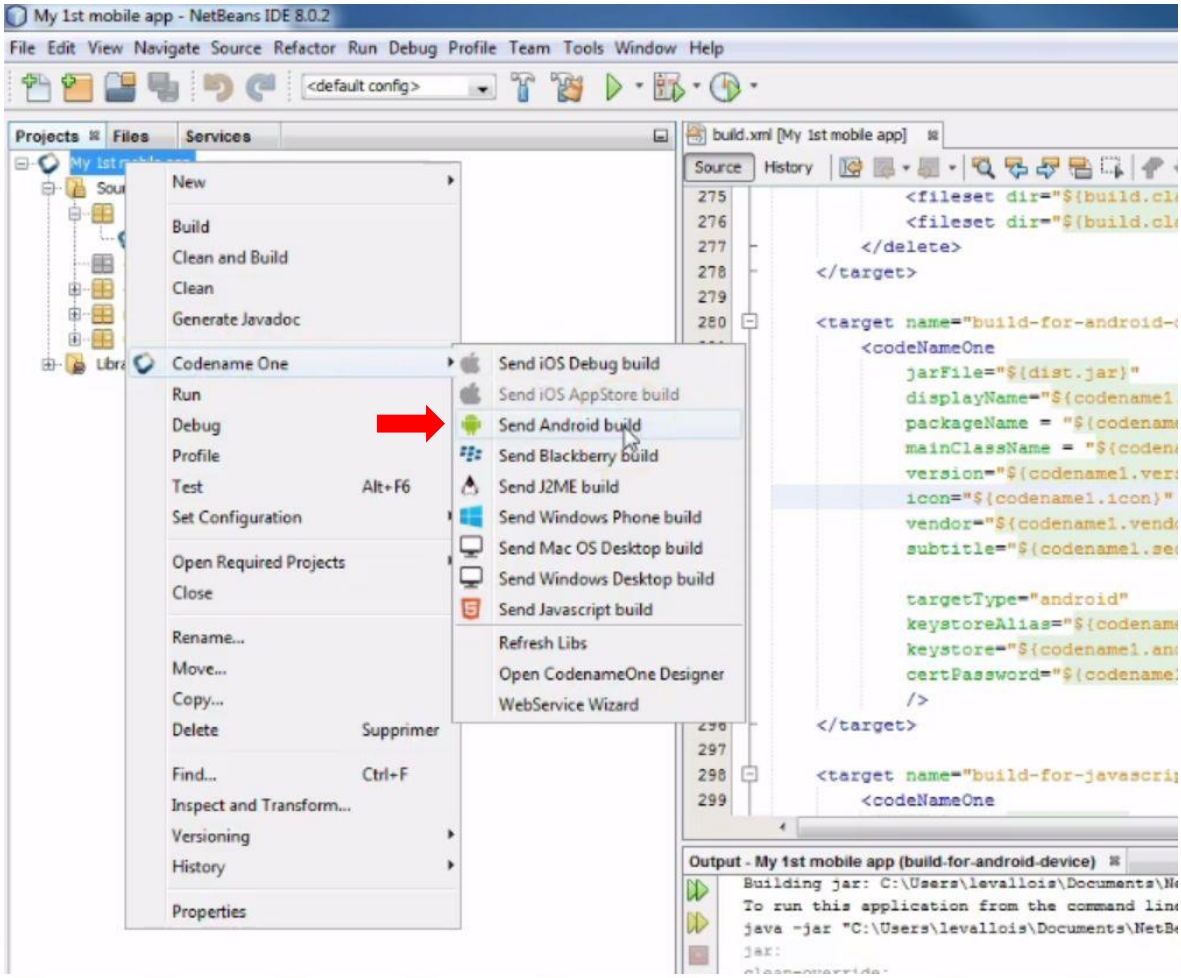
- A QR code. Flash the QR code from your Windows phone, this will download the app on your phone and you'll be able to install it.
- An email link: click here, and an email will be sent to the email address of your Codename One account. Open this email from your phone, click the link in it and this will start the download + installation process.

Module 4: Testing your app on a phone and distributing it

How to test your app on an Android phone

1. You need first to create a free account with Codename One (the tool we use to create apps).
   To do so, visit their website at www.codenameone.com and select "Sign up".
   Choose the free version. You are not obliged to choose to receive the newsletter.
   Please open the confirmation email you will receive and click on the link in it to confirm the creation of your account.

2. In NetBeans, right click on the title of your project and select "Send Android Build"



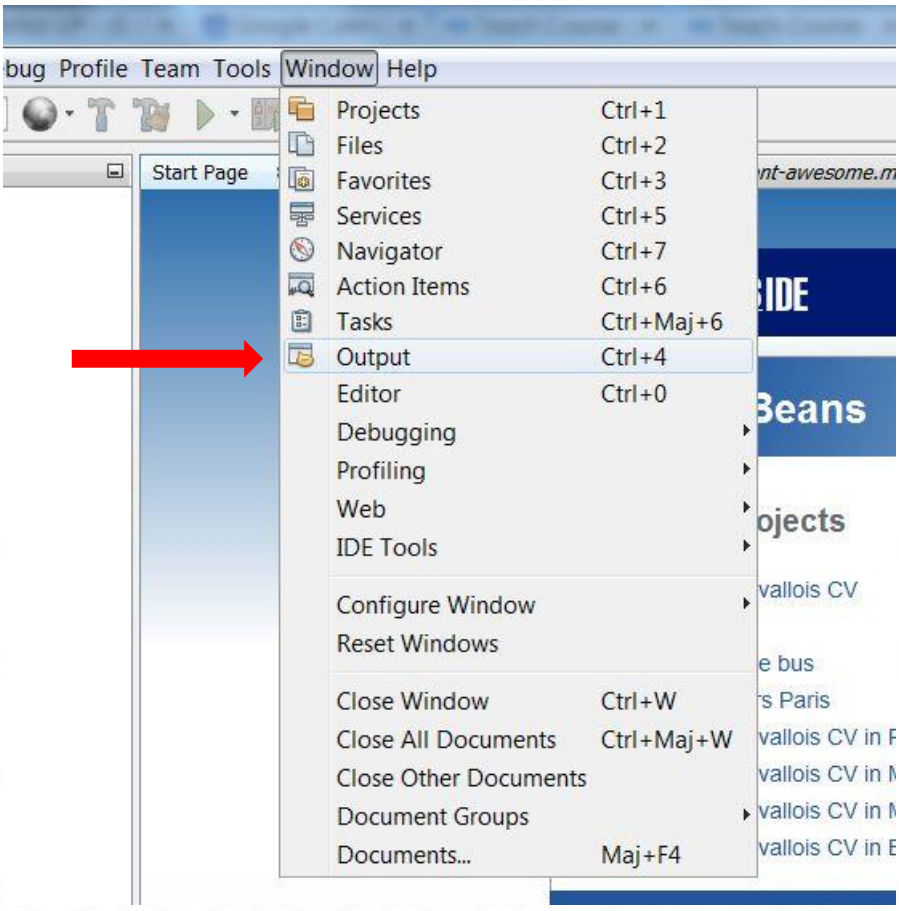
3. This starts the creation of package ready to be sent to Codename One. Two alert windows will open:
   - One asking for the login and password of your Codename One account (that you just created)
   - One telling you that your app has a provisional certificate. This is ok.

How to test your app on an Android phone

Level of difficulty: ● ○ ○ ○
Estimated time: 15 mn

4. You know the process finished successfully when you see this in the bottom panel of NetBeans:

Note: you don't see this output window at the bottom of NetBeans? In this case, in the menu of NetBeans, select "Windows" then "Output", it should make it appear:



Module 4: Testing your app on a phone and distributing it

How to test your app on an Android phone

5. You can now go back to the website of Codename One, where your app has been sent to create the Android version of it. Login, then go the the Dashboard (in the menu on top of the page).
6. You will see the status of your app:
   - blue -> in progress
   - green -> finished, ready to be used
   - red: something went wrong, your Android app could not be created.

7. When the status turns green, click on it. A series of option appears to install your app on your Android phone:
   - A QR code. Flash the QR code from your Android phone, this will download the app on your phone and you'll be able to install it.
   - An email link: click here, and an email will be sent to the email address of your Codename One account. Open this email from your phone, click the link in it and this will start the download + installation process.

Module 4: Testing your app on a phone and distributing it

Level of difficulty: ● ○ ○ ○
Estimated time: 45 mn

How to test your app on an iOS phone

NOTE 1: this tutorial can be executed from a PC, MAC or Linux. You also need to have 99$ ready, this is the cost of an Apple Developer account.

1. You need first to create a free account with Codename One (the tool we use to create apps).
   To do so, visit their website at www.codenameone.com and select "Sign up".
   Choose the free version. You are not obliged to choose to receive the newsletter.
   Please open the confirmation email you will receive and click on the link in it to confirm the creation of your account.

2. You must also use an Apple Id. You have an Apple Id if you use any Apple product. If not, you can create an Apple Id for free here: https://appleid.apple.com/

3. You then need a developer account for Apple. It costs 99$, and you can then have as many apps as you want for the Apple Store, for one year. Use your Apple Id to register for a developer account, here: https://developer.apple.com/programs/ (click on the "enroll" button on the right).

4. If you don't already have it, download and install iTunes: http://www.apple.com/itunes/

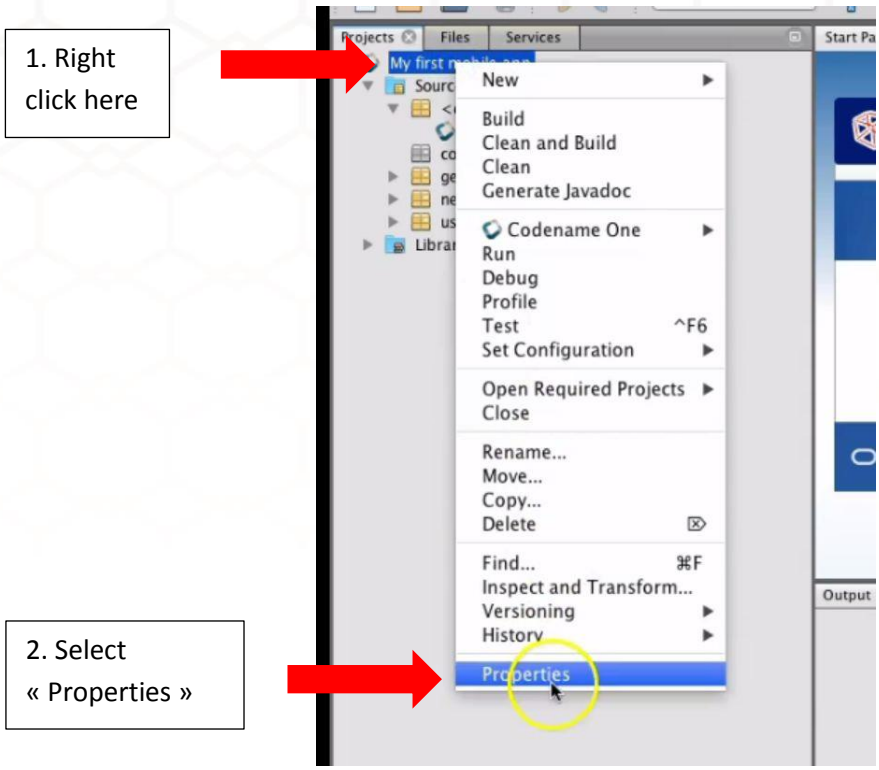
5. Plug your iPhone to your computer with a cable. Open iTunes and show the content of your phone.
   The serial number of the phone is shown. Click on it to switch to the UDID of your phone. You can copy this value by doing Control + Click on the UDID. Paste the value in a text editor of your choice.

6. Back to NetBeans. There, we right click (Control + click on Mac) on the title of our project and click on "Properties" in the menu that opens:

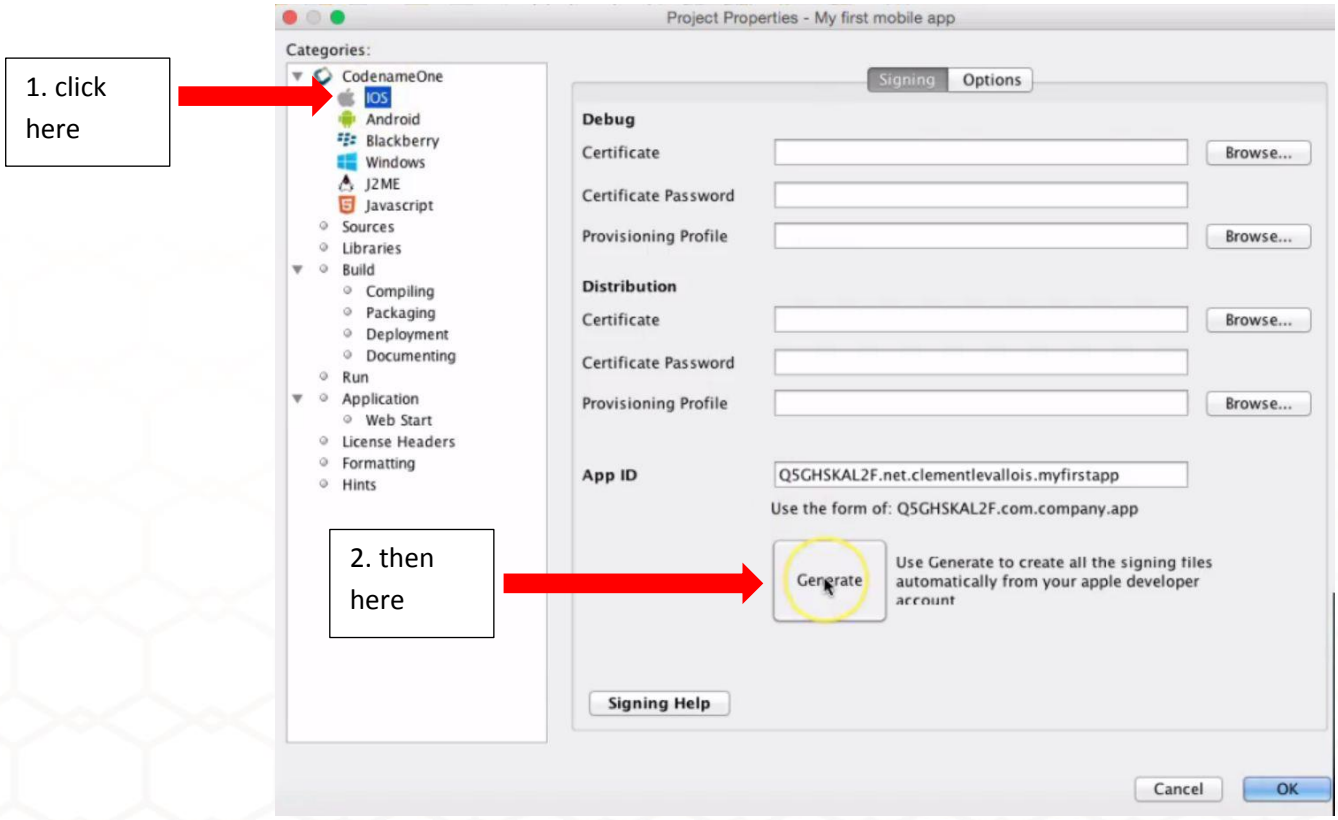
Module 4: Testing your app on a phone and distributing it

Level of difficulty: ● ○ ○ ○
Estimated time: 45 mn

How to test your app on an iOS phone



7. Then we click on « iOS », then on « Generate ». We are going to generate the certificates for your app : the ones to install the app on your phone, but also to distribute the app on the Apple Store :



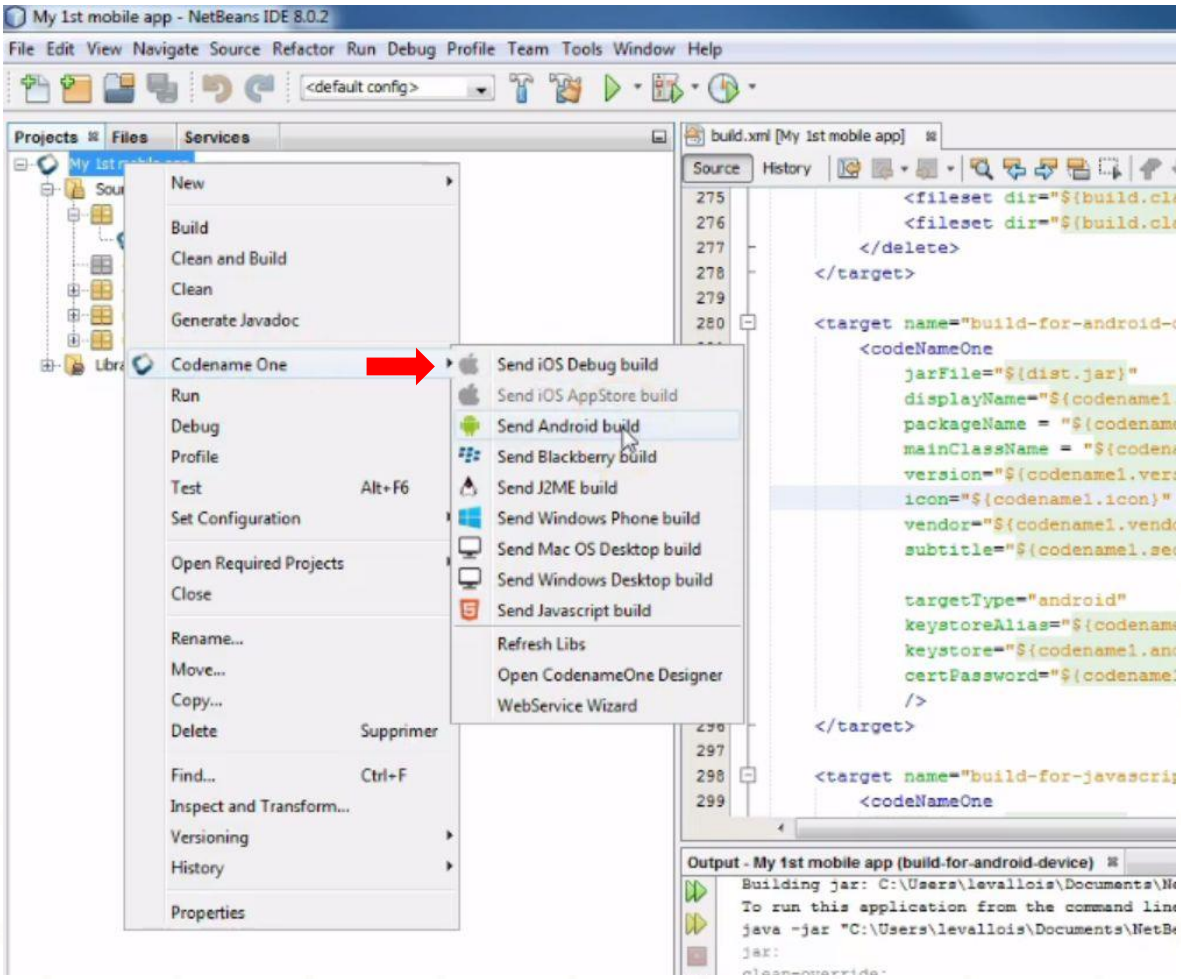
The first screen asks for your iTunes Connect login and password. These are the same as your Apple Developer account.

8. In the next screen, choose « Add a new device ».Give it a name of your choice (« My iPhone 4 »…) and paste the UDID we had gotten from iTunes (see preceding steps).

9. If you see any window asking to overwrite existing certificates, choose « Yes ».

10. When asked, install or save the certificates on your computer, choose the folder of your NetBeans project. We are now done with creatin the certificates for your app !

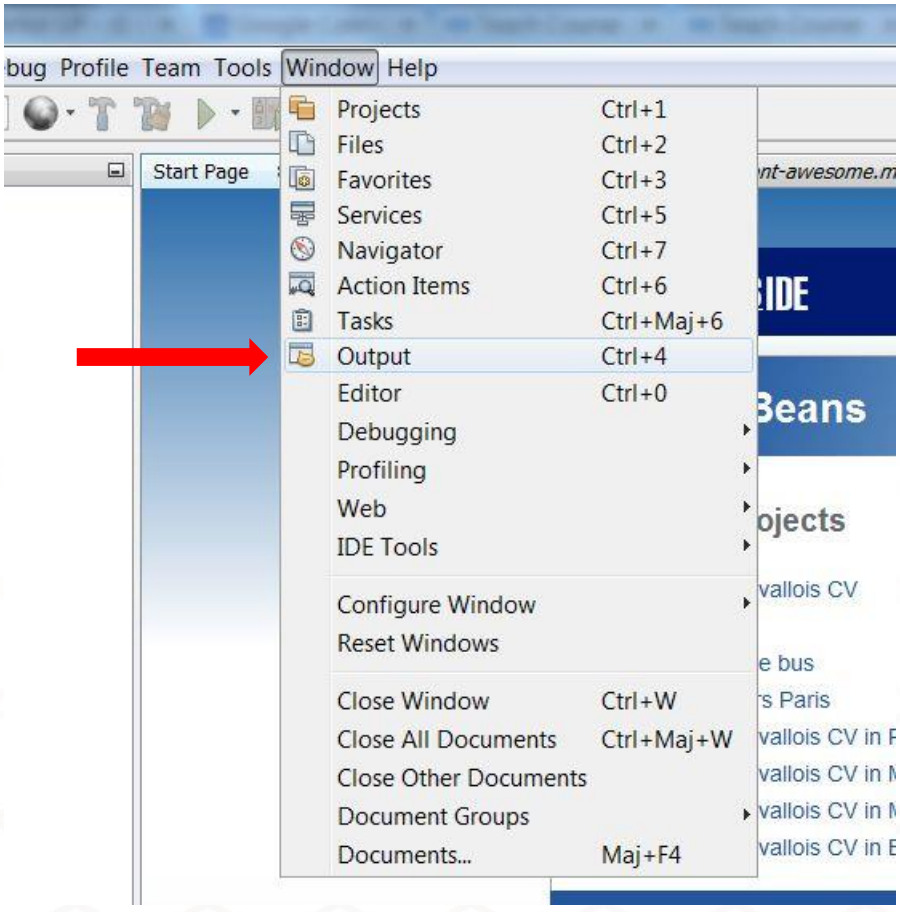
11. In NetBeans, right click on the title of your project and select “Send iOS Debug Build



12. This starts the creation of package ready to be sent to Codename One. An alert windows will ask you for the login and password of your Codename One account, please enter them.

13. You know the process finished successfully when you see this in the bottom panel of NetBeans:

Note: you don't see this output window at the bottom of NetBeans? In this case, in the menu of NetBeans, select "Windows" then "Output", it should make it appear:



14. You can now go back to the website of Codename One, where your app has been sent to create the iOS version of it. Login, then go the the Dashboard (in the menu on top of the page).

15. You will see the status of your app:
    - blue -> in progress
    - green -> finished, ready to be used
    - red: something went wrong, your Android app could not be created.

16. When the status turns green, click on it. A series of option appears to install your app on your iOS phone:
    - A QR code: flash the QR code from your iPhone, this will download the app on your phone and you'll be able to install it.
    - An email link: click here, and an email will be sent to the email address of your Codename One account. Open this email from your phone, click the link in it and this will start the download + installation process

Note:

This is a case where the videos are easier to understand than the pdfs alone.

The videos comment the slides step by step, this helps build your understanding.

Don’t read and rush. Watch the video instead!

To create an app, we use lots of boxes to organize all the objects that we need to build the app.
We create a box by giving it a name.

nameOfTheUser

nameOfTheUser;

Note:
- no space in the name
- no special character like !@#
- The first letter must be in lower case

Lines finish with a ;

Boxes are specialized: they contain just one type of object.

Example: a box created to contain text will always contain text, never numbers or pictures

So when we create a box, we must explain **in front of it** the kind of objects that can be put in it.

We say « String » for textual objects. Don't ask me why.
Please note that object names always start with an uppercase letter

String nameOfTheUser;

Text and numbers are the most simple objects that can be put in boxes.
Numbers are separated in different categories.

A box only for round numbers!

Integer ageOfTheUser;

A box which can contain decimal numbers!

Float averageRatingOfTheApp;

« Boolean » is another cool type of object. It has two values: true or false

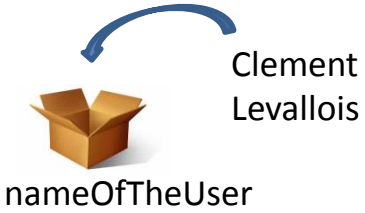
Creating a box which will contain just true or false values

Boolean isTheUserRegistered;

To put objects in boxes, we use the sign « = »

```
ageOfTheUser = 36;
```

When we put text in a box this is a bit special, it needs to be written between double quotes when we add it to a box:




```
nameOfTheUSer = “Clement Levallois”;
```

When we put new objects in boxes, it replaces what was there before:

```
ageOfTheUser = 37;
```

Now there is the number 37 in the box, 36 has been deleted.

Objects can have actions

(what?? What does it mean?? You said text – or String as you call it – is a type of object, I don’t see how text can have actions???)

Well, Strings can have lots of actions!
Example: turn the text to uppercase. Replace a letter by another. Find if the text contains a letter. Etc…

All actions of the object can be found by adding a dot (“.”) to the box that contains the object.

This action transforms the text in the box into CAPITAL LETTERS:

```
String nameOfTheUser;
nameOfTheUSer = “Clement Levallois”;
```

Why some brackets? We’ll explain that in just a moment.

Wow, should I know all these actions by heart???
No, just write the dot after the name of the box and a menu with all possible actions will appear.

```
nameOfTheUser.toUpperCase();
```

Name of the box

The dot!!

Name of the action

```
nameOfTheUser.concat(“ you are welcome!”);
```

This is what brackets are for. Sometimes actions need some object to work, and we put this object between the brackets.

What kind of objects can we put between the brackets? How many of them? It depends, all actions are different.

So here, if we had « Julie » in the box, after the action we will have: « Julie you are welcome! »

RECAP

We create 2 empty boxes, specialized in containing text

```
String greetings;

String nameOfTheUser;
```

We add « Clement » to one box → `nameOfTheUser = “Clement”;`

We change the text inside the box to upper case, **and we add it back into the box*** → `nameOfTheUser = nameOfTheUser.toUpperCase();`

we apply the action « concat » to the box nameOfTheUser and we put the result of the action in the box « greetings » → `greetings = nameOfTheUser.concat(“, hello!”);`

Now we have the box “greetings” which contains: CLEMENT, hello!

* Why do we need to add it back to the box? Why not just writing nameOfTheUser.toUpperCase() and the content of the box would be immediately updated?
Good question… this is those who have invented the programming language who have chosen to do it this way.

Note:

This is a case where the videos are easier to understand than the pdfs alone.

The videos comment the slides step by step, this helps build your understanding.

Don’t read and rush. Watch the video instead!

So we used so far boxes which contain objects like String (text) or Integer (round numbers)

```
String nameOfTheUser;
Integer ageOfTheUser;
```

The good news is, boxes can contain anything, not just text or numbers. That's actually 2 good news:

1. **We can create new kinds of objects, with their own actions, and put them inside boxes, and this will enable us to invent the functions of our app**

2. **People all around the world have already created many different objects, with many actions, and they share them freely and we can use them in our apps**

Actually Codename One, the tool we use in this MOOC, is made of objects and we can use them and their actions in our apps to connect to the Internet, access the camera of the phone, etc.

How to put these objects we created (or somebody else) in a box?

Let's take the example of a Label (you know, these pieces of text we put on the screen of the mobile phone in the designer)

myText

First, we create a box which I call "myText":

```
myText;
```

And we remember to put "Label" in front of it to say that it is a box specialized in containing Labels:

```
Label myText;
```

And finally we create a new label and put it in the box. Contrary to String and Integers, which are very simple objects, almost all other objects need to be created like that when we put them in the box:

new Label()

myText

```
myText = new Label();
```

Don't forget the upper case initial in the name of the object!

The same method works if we want to create a new screen for our mobile app:

First, we create a box called “myScreen” which will contain screens:

The box “myScreen”, specialized in containing Forms. Yes, screens are called « forms » by codename one…

```
Form myScreen;
```

Then we create a screen and put it in the box:

```
myScreen = new Form();
```

Now, we can explore what kind of actions we can do with a screen. Maybe show it in the app? So easy, write a dot after myScreen and explore all possible actions. The one we want is:

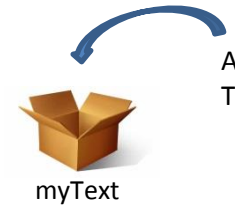
```
myScreen.show();
```

If you write that in the code of your app (we’ll show you where exactly), you will be able to change which screen is shown to the user. Yes, you know how to code the navigation from one screen to another!!

Special actions

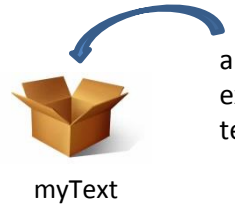
This page is a bit complicated, you can skip it and come back later if you want

We have seen that objects have values and actions. For example:
We create a box specialized in containing text, and we add some text to it:



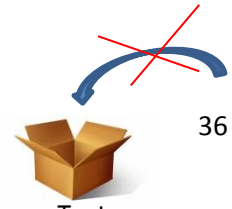

```
String myText;
myText = “An EXAMPLE of Text”;
```

And we can apply some actions to the content of the box (this one puts every letter in lower case):




```
myText = myText.toLowerCase();
```

Well, there are other special actions we can use. These special actions can be found by writing the type of object (here, String) and adding a dot to it.
For example, this would not work because a number cannot go into a box made for text:




```
myText = 36;
```

What we can do is using a **special action** by adding a dot to the type of object of the box (here, String – don’t forget the upper case “S!”). This action “valueOf” takes a number (36), transform it into text, and adds it to the box:


```
myText = String.valueOf(36);
```

Note:

This is a case where the videos are easier to understand than the pdfs alone.

The videos comment the slides step by step, this helps build your understanding.

Don’t read and rush. Watch the video instead!

What are curly braces made for? { }

These curly braces are used to pack together several lines of code.
There are 3 important situations where we use them:

1) "if this, then do all that!"

Let's say we create a box for text, and we put an empty text in it:

```
String myText = "";
```

We might want to do:
<u>if</u> the box contains an empty text, then
1) add some text in the box
2) put it in uppercase.

this is true or false

```
if ( myText.isEmpty() ) {
   myText = "here is some text";
   myText = myText.toUpperCase();
}
```

if myText is empty, everything inside the curly braces will happen.

If myText is not empty, what is written inside the curly brackets is ignored, it won't happen.

What are curly braces made for? { }

2) "for each object in a list, do some actions" (creating a "loop")

To show that, let's use an object we have not seen before. This is an ArrayList, which is like a shopping list where many objects can be added and deleted. Let's create a box specialized in containing ArrayLists, and let's add an ArrayList to it:

```
ArrayList myList ;
myList= new ArrayList();
```

One useful action that lists can do is to put objects in it, like Strings. Let's do it:

```
myList.put("my first item in the list");
myList.put("my second item in the list");
```

What if we want to apply an action on the 2 items of the list? We use curly brackets { }:

this means: take each object of type String that we have added to the list

We « **loop** » through the items of the list →

```
for (String item: myList) {
   item = item.replace("my", "your");
   item = item.toUpperCase();
}
```

← **This means that for every item in the list, we replace « my » by « your » (convenient action!) and we put it in UPPERCASE**

What are curly braces made for? { }

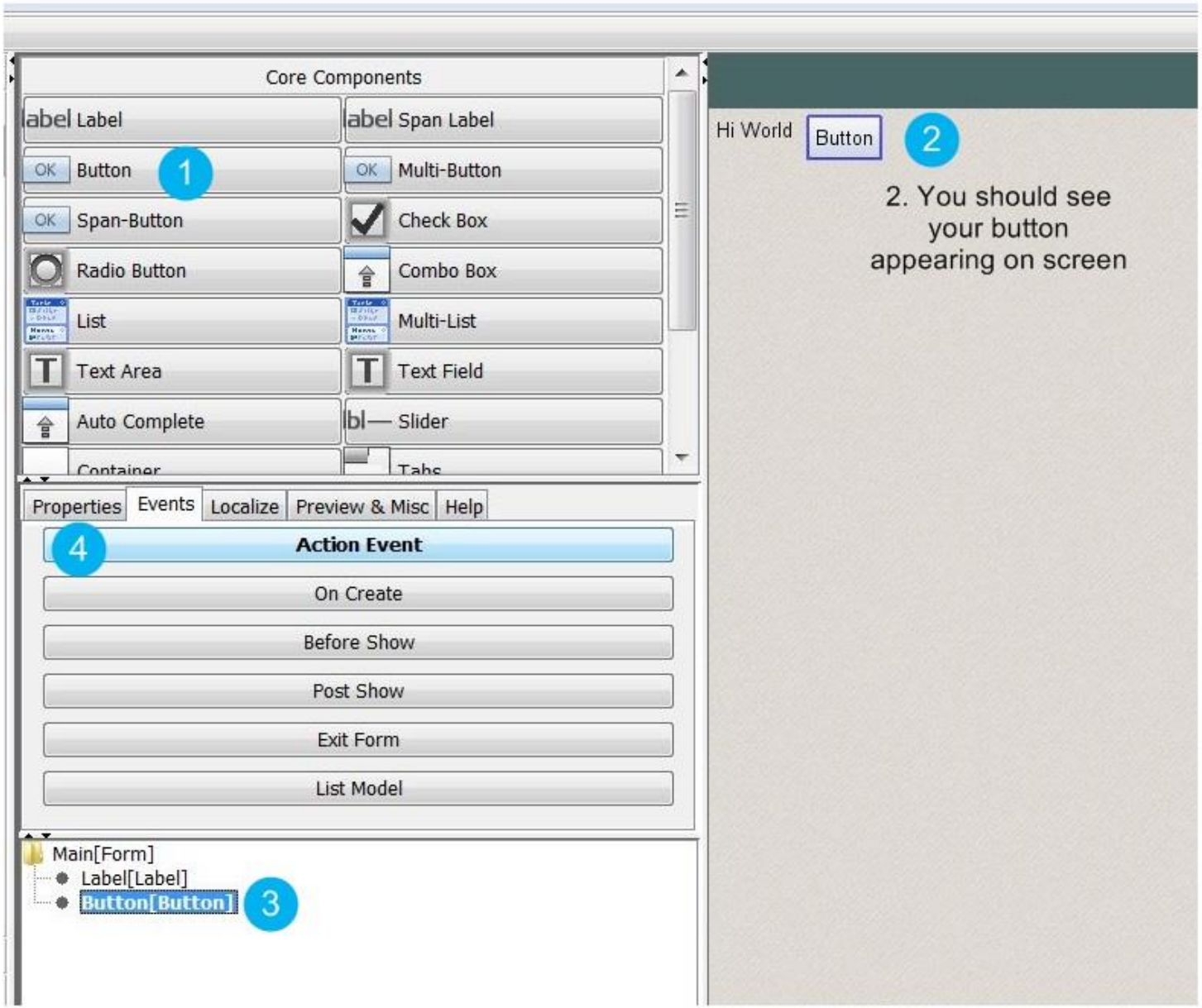
3) Finally, we use curly braces when we create our own objects and their actions

It is actually supra easy to create new objects, and to invent actions for them, but you had enough for one session.

We are lucky that the designer of Codename One can create new actions just by clicking on stuff, without coding.

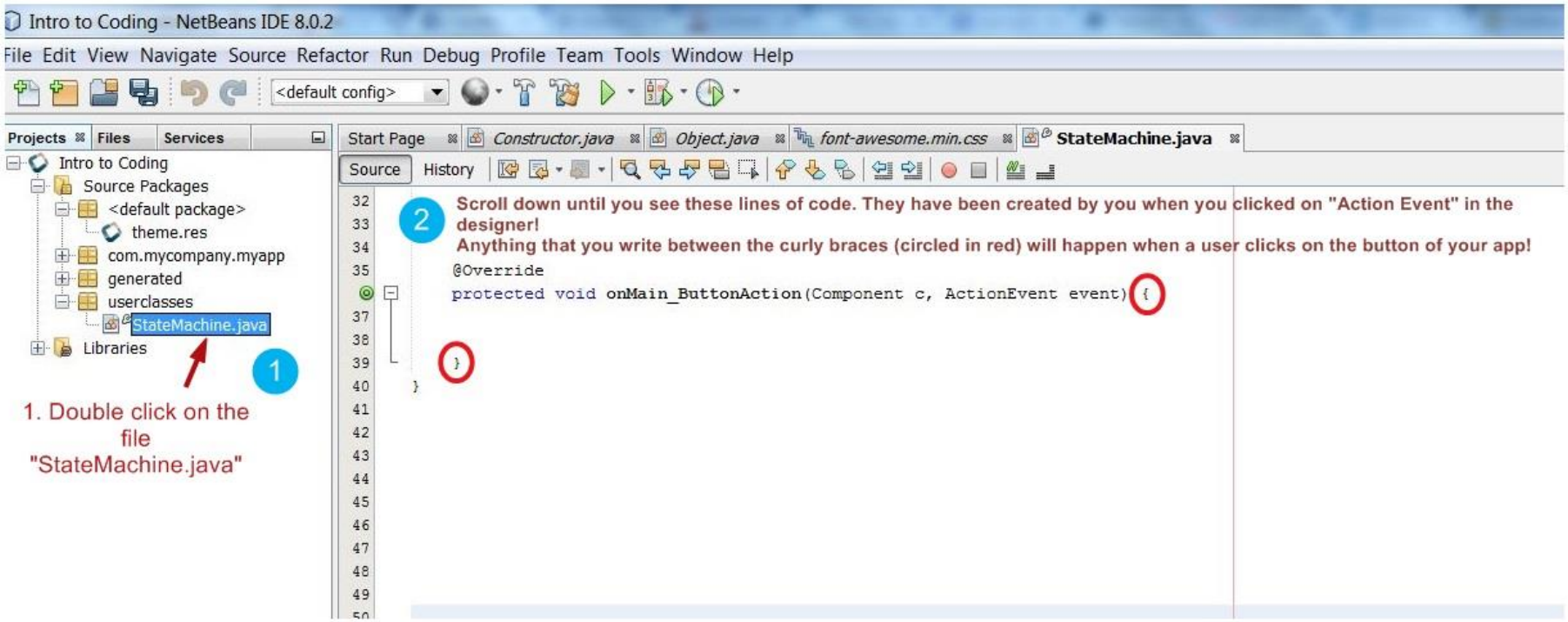
See the next slide showing how to create an action without coding!

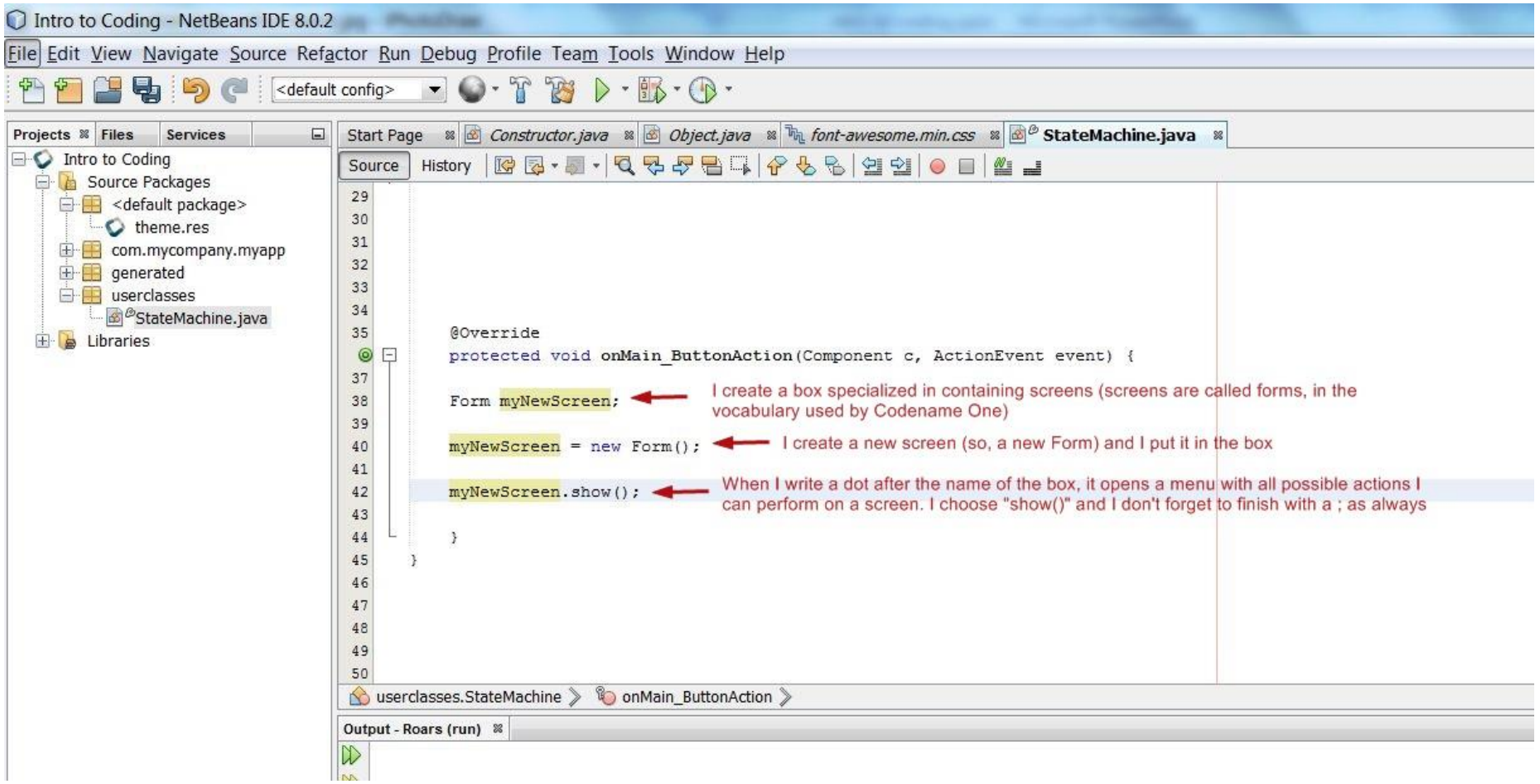
1. Click here to add a button on screen

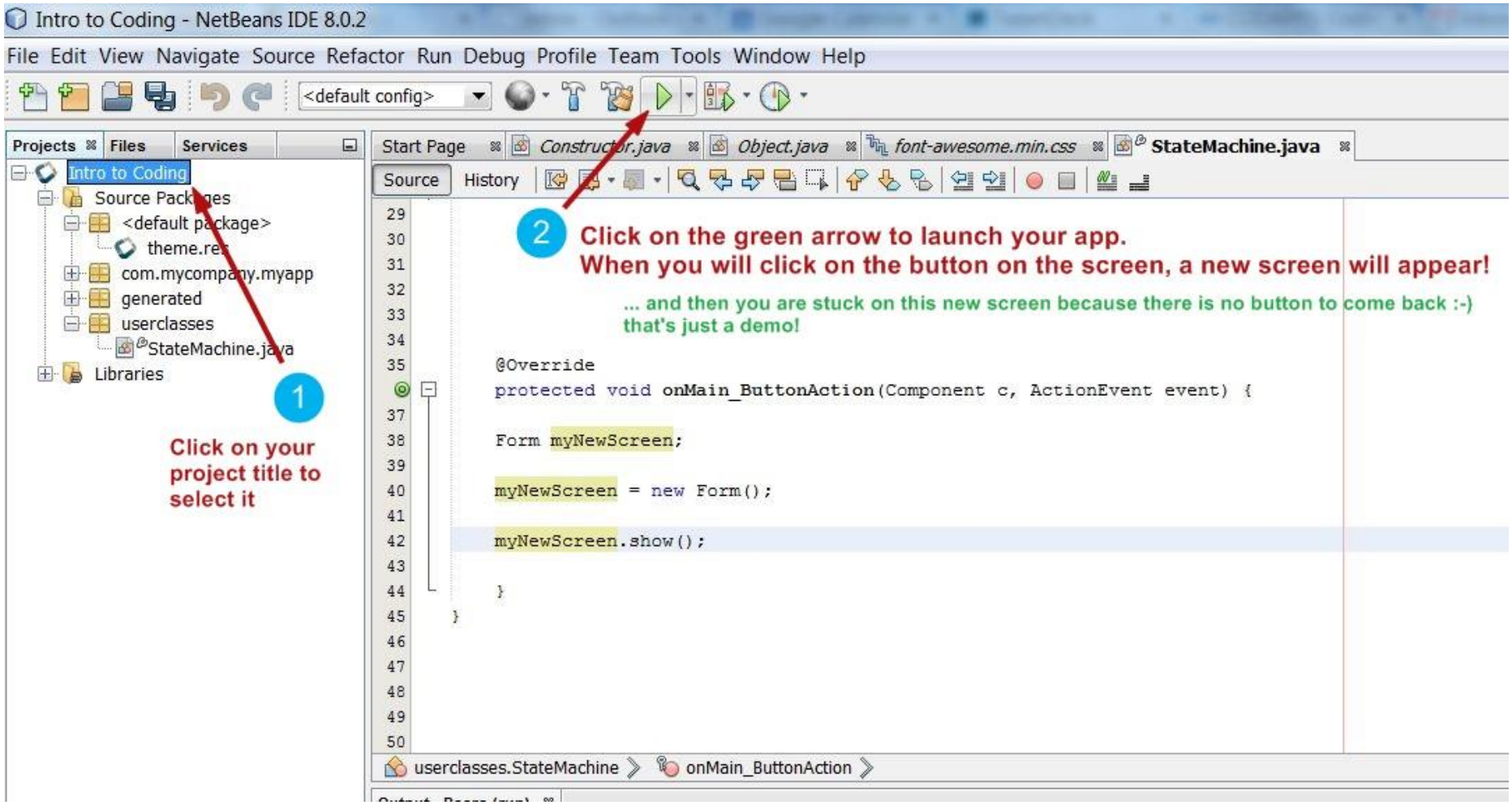


4. Click here once. This has for effect to create a new action for the button. We will find it in NetBeans.

3. Make sure you select your button in this bottom panel


Intro to Coding - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config>
Projects Files Services
Intro to Coding
Source Packages
<default package>
theme.res
com.mycompany.myapp
generated
userclasses
StateMachine.java
Libraries
1
1. Double click on the file "StateMachine.java"
Start Page
Constructor.java
Object.java
font-awesome.min.css
StateMachine.java
Source
History
2
Scroll down until you see these lines of code. They have been created by you when you clicked on "Action Event" in the designer!
Anything that you write between the curly braces (circled in red) will happen when a user clicks on the button of your app!
@Override
protected void onMain_ButtonAction(Component c, ActionEvent event) {
}
}


Intro to Coding - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config>
Projects Files Services
Intro to Coding
Source Packages
<default package>
theme.res
com.mycompany.myapp
generated
userclasses
StateMachine.java
Libraries
Start Page
Constructor.java
Object.java
font-awesome.min.css
StateMachine.java
Source History
@Override
protected void onMain_ButtonAction(Component c, ActionEvent event) {
Form myNewScreen;
I create a box specialized in containing screens (screens are called forms, in the vocabulary used by Codename One)
myNewScreen = new Form();
I create a new screen (so, a new Form) and I put it in the box
myNewScreen.show();
When I write a dot after the name of the box, it opens a menu with all possible actions I can perform on a screen. I choose "show()" and I don't forget to finish with a ; as always
}
}
userclasses.StateMachine
onMain_ButtonAction
Output - Roars (run)



Any issue on this step? The app does not launch? Probably a very easy thing to fix, post a screenshot on the forum of Coursera to get help!

Everything works? Congrats, we have learned enough to build an app in the next session!

Note:

This is a case where the videos are easier to understand than the pdfs alone.

The videos comment the slides step by step, this helps build your understanding.

Don’t read and rush. Watch the video instead!

Coding quicker (or how to code like a boss)

So far we learned that:

- We create a box, specialized in a type of object. For example:

```
Form firstScreen;
```

- We put an object in it:

```
firstScreen = new Form();
```

To save time and write less code, we can do the same in one line:

```
Form firstScreen = new Form();
```

The box specializes in containing screens (Form)

Name of the box

and we put a new Form in the box

If we want to create a label:

```
Label welcomeMessage = new Label();
welcomeMessage.setText("Welcome to this app");
```

To save time and write less code, we can already decide the text of the label when we create it:

```
Label welcomeMessage = new Label("Welcome to this app");
```

Then imagine we want to add the welcome message to the screen:

```
Form firstScreen = new Form();
Label welcomeMessage = new Label("Welcome to my app!");
firstScreen.addComponent(welcomeMessage);
```

« addComponent » is the action we need when we want to add stuff to a screen or to a container

But we can write it in an even shorter way:

```
Form firstScreen = new Form();
firstScreen.addComponent(new Label("Welcome to my app!"));
```

Rooky mistake: forgetting to open or close all brackets!

If we want to get the height of the welcome message on the screen:

We start by getting the label and putting in in the box “welcomeMessage”:

```
Label welcomeMessage = firstScreen.getComponent(0);
```

getComponent is the action we need to select a component on the screen. The number in the bracket is to indicate which one we want. The 1st component on screen (on top) is number zero. The 2nd component is number 1, etc.

Then we write another line of code to get the height of this message:

```
Integer heightOfTheLabel = welcomeMessage.getHeight();
```

But we can write all that in an even shorter way:

```
Integer heightOfTheWelcomeMessage = firstScreen.getComponent(0).getHeight();
```

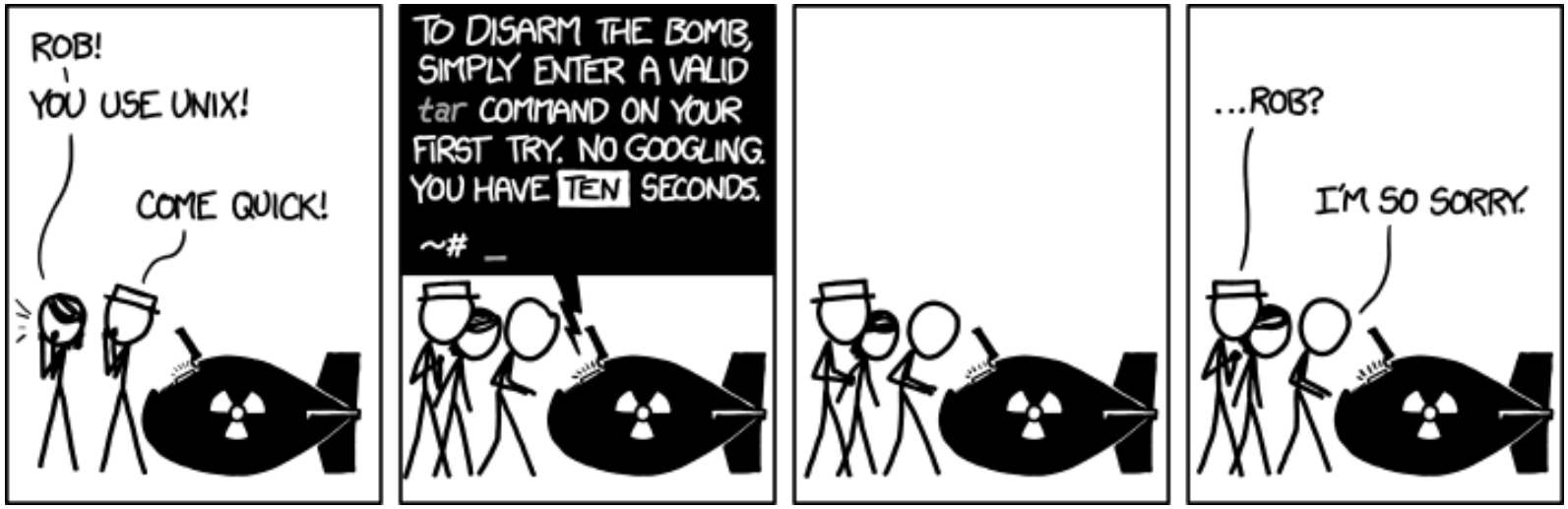
This is called « chaining » the actions because we put them in a chain, each new action comes after a new dot

Any secret tricks?

1) How do I know what code I should write to make my app do this and that?
-> Use the online documentation!
For Codename One: https://www.codenameone.com/manual/

2) But how do I learn all that?
-> Start with very simple things, get the reward of seeing it work
-> Then try more complex things, then Google your question when it does not work!

3) Black magic is OK
We learnt the basics of coding. Sometimes, we will see other things (strange curly brackets, strange words like “protected” or “final”…) that we don’t understand.
In this case, **just copy-pasting these lines of code (what I call “black magic”) is OK. Everybody does it, even the experts.**



To get this joke: the « tar » command is one of the most simple lines of code in unix. Rob, who is an expert in unix, must confess he needs to Google « how to use the tar command » when he needs to use it…
So relax! Even experts keep using Google to write code! **Source: https://xkcd.com/1168/**

Note:

This is a case where the videos are easier to understand than the pdfs alone.

The videos comment the slides step by step, this helps build your understanding.

Don’t read and rush. Watch the video instead!

JSON, what an awful beast (but we need it)

Let's say that Julie, a user of your app, wants to send a message to another user.
You think it would be like:
"hey, how are you today?"

But no. That would be too simple.
When we exchange messages, they are organized in a different way.
We write instead:

The JSON way of exchanging info on the Internet

```
{
 "author": "Julie",
 "message": "hey, how are you today?"
}
```

The JSON way of exchanging info on the Internet

```
{
 "author": "Julie",
 "message": "hey, how are you today?"
}
```

Why??

It is convenient:

- We can send different stuff at once (the name of the author, the message…), they all stay nicely organized
- If everybody uses this JSON thing, we can use standard objects to write or read stuff in Json and exchange data easily with everybody!

Converting some text in JSON before sending it to the Internet:

```
Hashtable tableOfMessages = new Hashtable();
tableOfMessages.put("author", "Julie");
tableOfMessages.put("message", "hey, how are you today?");

String textFormattedInJson = BlackMagic.convert(tableOfMessages);
```

… and then send this box textFormattedInJson to the Internet!

The JSON way of exchanging info on the Internet

```
{
 "author": "Julie",
 "message": "hey, how are you today?"
}
```

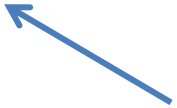
And when our app receives some JSON stuff from the Internet, it can be used like that:

Using some JSON we received from the Internet:

```
JSONObject stuffInJson = new JSONObject(whatIReceivedFromInternet);

String nameOfAuthor = stuffInJson.get("author");
String message = stuffInJson.get("message");
```


The box « nameOfAuthor » contains now « Julie »



The box « message » contains now « hey, how are you today? »

So, JSON is actually a convenient way to exchange text over the Internet.

The dark side of JSON

The nice easy way of JSON:

```
{
 "author": "Julie",
 "message": "hey, how are you today?"
}
```

The dark side of JSON:

```
{
 "author": "Julie",
 "message": "my friends says \"hi\"! ",
 "recipients": ["Mike", "Liu", "Mohamed"],
 "timestamp": {
    "date": "23/09/2015",
    "time": "16:23:02"
  }
}
```

We need to use the special character \ in front when we use " " in our text.

Here, inside our JSON object (which is in green) we have another JSON object (in orange)!!

We can have lists of stuff (called ARRRAYS) between square brackets, that's confusing

So how can we retrieve the time when the message has been sent by Julie?

```
JSONObject stuffInJson = new JSONObject(whatIReceivedFromInternet);

JSONObject timestamp = stuffInJson.get("timestamp");
String timeWhenMessageWasSent = timestamp.get("time");
```

Roar part 1 :

Sending a roar from the app to the Internet

```
String roar;
```

We create a box «roar », specialized in containing text.

This is in this box that we will store the roar that the user will write

```
@Override
protected void onMain_TextAreaAction(Component c, ActionEvent event) {
    roar = findTextArea().getText();
}
```

We put in the box « roar » what the user has typed in the TextArea that we have added to the designer (watch the video !)

```
@Override
protected void onMain_ButtonAction(Component c, ActionEvent event) {
```

What is roar here? It is the content of **this** box

```
Hashtable infoToBeSent = new Hashtable();
infoToBeSent.put("roar", roar);
infoToBeSent.put("author", "seinecle");
```

title — value

We create a box « infoToBeSent », specialized in containing objects « Hashtable ». Hashtables are objects which can store pairs of titles + values.

With these lines of code, we put the roar and the name of the author in the Hashtable object.

```
final String infoInString = Result.fromContent(infoToBeSent).toString();
```

This line of code transforms the box infoToBeSent into some text formatted in JSON. Convenient !

The Internet address where we will send the roars.

```
String firebase = "https://roar.firebaseIO.com/listofroars.json";
```

```
ConnectionRequest request = new ConnectionRequest() {
    @Override
    protected void buildRequestBody(OutputStream os) throws IOException {
        os.write(infoInString.getBytes("UTF-8"));
    }
};
request.setUrl(firebase);
request.setPost(true);
request.setHttpMethod("POST");
request.setContentType("application/json");

NetworkManager.getInstance().addToQueueAndWait(request);
```

Boring ? Difficult ? No, I just did a copy / paste from the Codename One documentation

This creates a « ConnectionRequest » object and we put it in the box « request ». Then between the green curly braces, we have strange lines of code, which basically prepare infoInString to be sent to Internet.

We apply actions to the box « request ». These actions are necessary to prepare the sending of « InfoInString » to the Internet.

This line of code actually sends infoInString to the Internet. Done !

```
}
```

If you think these lines of code are particularly hard to understand, and certainly can't be memorized, I agree with you. Coders don't learn that by heart. You read the documentation of Codename One, copy the code you need into your app, and do one or two adjustments to adapt the code to your particular needs. That's how I did it here.

Clément Levallois

Roars part 2 :

Receiving a roar from the app to the Internet

Note : for better undersanding of this lesson, the tutorial on JSON should be fresh in your mind

```
String roars = "https://roar.firebaseIO.com/listofroars.json";
```
The website where we will get the roars from

```
ConnectionRequest request = new ConnectionRequest();
request.setUrl(roars);
request.setPost(false);
request.setHttpMethod("GET");
request.setContentType("application/json");

NetworkManager.getInstance().addToQueueAndWait(request);
```
The object « ConnectionRequest » in the box « request », with all actions necessary to connect to the website above

```
ByteArrayInputStream allRoarsInBytes = new ByteArrayInputStream(request.getResponseData());
```
This line actually gets the data from the Internet

```
String responseInString = Util.readToString(allRoarsInBytes, "UTF-8");
```
Converting the data received to text

```
JSONObject allRoarsInJsonFormat = new JSONObject(responseInString);
```
Converting the text into a JSON object, which is is a list of roars

The JSON object we received looks like that (here, the example of receiving just 2 roars):

```
{"-JywLB4G8OErGN3dsI_c" :
        {"author":"seinecle",
        "roar":"This is an example of a roar"},
"-JywNGfPCEUDmGxJ3AXw" :
        {"author":"seinecle",
        "roar":"This is a test"},
}
```

```
JSONArray listOfRoarIds = allRoarsInJsonFormat.names();
```
Collecting all the names (or « titles ») of the roars.

So listOfRoarIds contains: `-JywLB4G8OErGN3dsI_c` and `-JywNGfPCEUDmGxJ3AXw`

```
Form wallScreen = c.getComponentForm();
```
We are getting access to the screen where we want to put the roars. We put this screen in a box to use it later

```
Container myContainerForAllRoars = new Container();
Layout myLayout = new BoxLayout(BoxLayout.Y_AXIS);
myContainerForAllRoars.setLayout(myLayout);
```
Creating a Container, creating a Box Y Layout, then saying that the Container should have this layout.

This container is going to contain all roars

Now, we are going to take each of the roars we received and put them on screen.

Clément Levallois

In simple words, the lines of code below are doing the following:

1. "We'll start counting roars we received from Internet, starting at roar number zero.
2. While our counter of roars is lower than the total number of roars,
   - 3 - Get the title corresponding to the roar (so, for the 1st roar that is `-JywLB4G8OErGN3dsI_c`)
   - 4 - Get the JSON object corresponding to this title. So, for the 1st roar that is
     ```
     {"author":"seinecle",
      "roar":"This is an example of a roar"}
     ```
   - 5 - Get the value corresponding to the title "author" ("seinecle" for the 1st roar), put it in a box "author"
   - 6 - Get the value corresponding to the title "roar" ("This is an example of a roar" for the 1st roar), put it in box "roar"
   - 7 - Create a container
   - 8 - Create a label with the text of the box "author"
   - 9 - Create a label with the text of the box "roar"
   - 10 - Put these labelsin the container we just created
   - 11 - Add the container to the container "myContainer" created on the previous page (see #)
   - 12 - Move on to the next roar by adding 1 to our counter of roars

When we have finished looping through all the roars we have received,

- 13 - Add the bigger container (now containing all the labels representing the roars and their authors) to the screen. We add it in the last position, which is position "number of components already present on the screen". This number is obtained with the method getComponentCount() from the screen
- 14 - Refresh the screen so that the roars we added can appear"

```
1  Integer counterOfRoars = 0;

2  while (counterOfRoars < allRoarsInJsonFormat.length()) {
3          String idOfOneRoar = listOfRoarIds.getString(counterOfRoars);
4          JSONObject oneRoarInJsonFormat = (JSONObject) allRoarsInJsonFormat.get(idOfOneRoar);

5          String author = oneRoarInJsonFormat.getString("author");
6          String roarText = oneRoarInJsonFormat.getString("roar");

7          Container myRoarContainer = new Container();

8          Label myLabelForAuthor = new Label(author);
9          Label myLabelForRoar = new Label(roarText);

10         myRoarContainer.addComponent(myLabelForAuthor);
           myRoarContainer.addComponent(myLabelForRoar);

11         myContainerForAllRoars.addComponent(myRoarContainer);

12         counterOfRoars = counterOfRoars + 1;

   }  <-- End of loop, go back to 2

13 wallScreen.addComponent(wallScreen.getComponentCount(), myContainerForAllRoars);
14 wallScreen.revalidate();
```

Clément Levallois

Eventbrite

Meetup

Participate in meetups in your city!

Join a coworking space

Apply for an incubator

Release early,

Release often

post your questions on the forum!

Google group
+
website (where there is
a manual and videos)

Inspiration to design your mobile app

The largest forum for all questions
related to coding.

NATIVE:

You create an app for Android → You get an app for Android phones (Samsung, Nexus, Motorola, etc.)

Google

You create an app for Windows → You get an app for Windows Phones (Nokia)

Windows Phone

You create an app for Apple → You get an app for iPhones

iOS

3 different technologies to learn

Takes time

Excellent result: we get 3 « real apps »

HYBRID:

We create <u>one</u> project, using the same kind of methods as for creating web pages

The app is actually contained in the web browser of the phone. It looks like a native app.

Web brower used by any phone

ionic

Just one project to develop

Techno is well known

Fit for most purposes

NO NAME:

We create a project, using a programming language called Java

Java

C#

Codename One takes your project and create from it « real » apps for Apple, Android and Windows Phones.

CODENAME ONE

Xamarin

You get an app for Android phones (Samsung, Nexus, Motorola, etc.)

You get an app for Windows Phones (Nokia)

You get an app for iPhones