CODAPPS

Essential vocabulary and preliminary notions

Clément Levallois

2017-12-26

Table of Contents

last modified: 2018-01-12

1. A note on components and actions

a. Components

A mobile app can **include** many things. These are like the Lego blocks of our app:

- pictures
- buttons to press
- videos
- places where the user is supposed to write text
- a menu to navigate the app
- etc.

Things included in an app are called **Components**

b. Actions

And the app is also supposed to **do** many things, for example:

- open a web page when the user clicks a button
- change the picture when the user swaps left
- zoom in when the user double taps on the screen
- make Angry Bird fly when the user plays the game
- etc...

All an app can "do", we will call these **Actions**

2. The difficulty of naming things

When building software like mobile apps, it is important to give precise names to the tools we use.

Often, these names don’t make intuitive sense, and this creates an impression of technical difficulty:

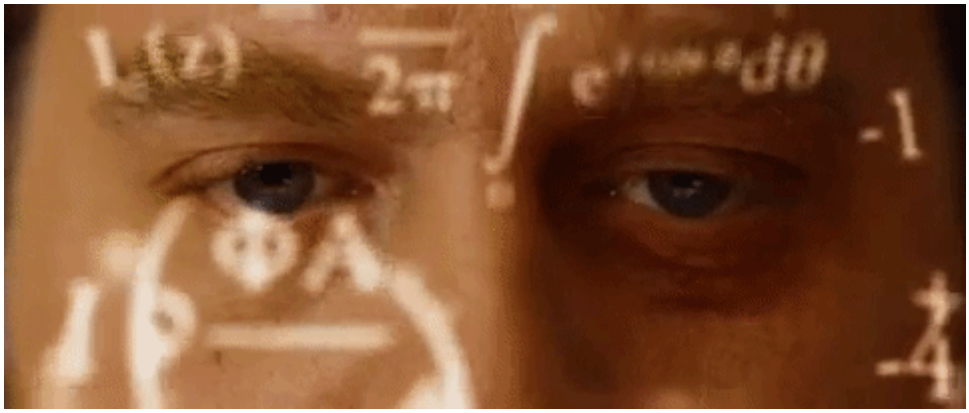

Figure 1. Getting confused by the technical vocabulary

It is not to be annoying, but to make sure we actually get the proper tool when we need it. Every domain of specialty, not just programmers, do this.

For instance:

- we will design the screens of our first mobile application.
- these screens, showing the content of our app, are not simply called "screen"... why?

1. Because `screen` is a name already taken, to name the phone's actual physical screen (the one that breaks when your phone falls on the ground)
2. So, what name should be given instead to the area an app takes on the screen of the phone: "ScreenApp"? "Region"? "FullSizeAppRegion? EmptyScreenOfTheAppWhereToPutThings"? None of this.

→ The name chosen is `Form`.

A Form is the first Component we need when creating a mobile app.

It is the empty region taking all the space of the phone's screen, where we will draw the app (text, buttons, etc.)

Because of this need for precise names, you will see that:

- adding text to the app, we will use a Component named `Label` (not Text)
- adding pictures to the app, we will use a Component named `Image` (not Picture)
- adding buttons to the app, we will use a `Button` Component (this one is intuitive!)

3. Two approaches to work on mobile apps

To sum up what we have seen so far:

- when we start designing an app, we need to create a `Form` first.
- Then we will place Components in the Form: Buttons, Labels, Images, etc.

- Then we can attach actions to each of these components (a "click action" on a button, etc)

How will you do that in practice? Building a new mobile app is like building a new piece of furniture : you can build it yourself from scratch, or buy it at Ikea and just assemble it.

Codename One, the framewok we use, gives you these two options: Do It YourSelf (DYI) or Ikea.

- DYI: write the code to create everything in the app.
- IKEA: use a **"Graphical User Interface" (GUI)**, which means we "draw" our app by click and point (a bit like in Photoshop), with a minimum of code to write.

There are different benefits to both approaches:

	Do It Yourself / writing code	**Ikea approach / using a Graphical User Interface (GUI)**
Benefits	Very flexible! You control every parameter since you write everything yourself	Quick and easy! You just drag and drop things, click and point with the mouse, no need to learn how to code.
Inconvenients	You need to learn how to code. Slow since you write everything yourself.	You get stuck at some point: some features you need for you mobile app are too specific to be available. Writing code is going to be necessary.

In the next lesson of this module, we are going how to see how to create a Form for our app, using the two methods alternatively: by writing code or by using the Graphical User Interface.

The end

Questions? Want to open a discussion on this lesson? Visit the forum here (need a free Github account).

Find references for this lesson, and other lessons, here.

This course is designed by Clement Levallois.

Discover my other courses in data / tech for business: http://www.clementlevallois.net

Or get in touch via Twitter: @seinecle