CODAPPS

Essential vocabulary and preliminary notions

Clément Levallois

2017-12-26

Table of Contents

last modified: 2017-12-27

1. A note on components and actions

a. Components

A mobile app can include many things:

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

We will call these things **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:

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

Like: to cut fabric, do you need a rotary cutter, scissors or shears?

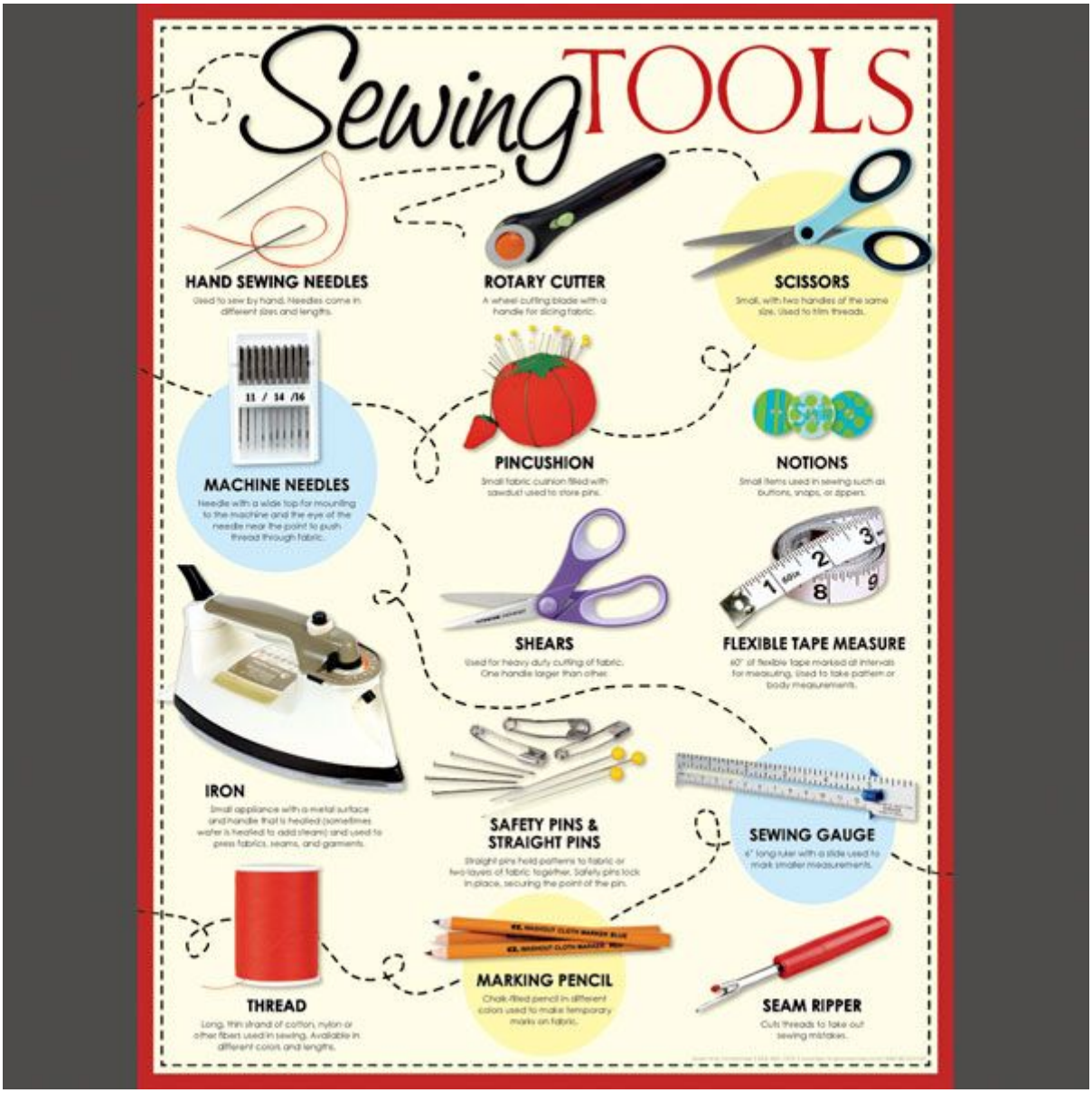

Figure 1. Tools need precise names

Back to mobile app development:

- To build a mobile app, the first Component we need is an empty region, the size of the screen of the phone, where we will place all other Components.
- Is this Component going to be named a `Screen`, simply?
- no, 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)
- So, what name should be given instead? ScreenApp? Region? FullSizeAppRegion?

EmptyScreenOfTheAppWhereToPutThings? None of this.

- The designers of the tool chose the name `Form`.

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

It is the empty region of the app where we will place all other Components.

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 designing 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 do we do that in practice? There are two ways.

a. DIY or the Ikea way?

Building a new mobile app is like building a new piece of furniture :

- "Do it yourself (DIY)": buy wooden planks and nails, cut and design everything at the right size, and assemble the pieces.
- "The Ikea way": buy a furniture in detached pieces already at the right size, and do the assembly yourself with the aid of a user guide.

Codename One, the framewok we use, gives you these two options: DYI or Ikea.

- DYI: write the code to create everything in the app.
- IKEA: use a **"Graphical User Interface" (GUI)**, which means you drag and drop things to build the app with a minimum of code to write.

There are 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: to add features to your mobile app, not everything is in the GUI. Writing code is going to be necessary.

b. An example: creating a form by writing code, or with a GUI.

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