Created by: Zoltán Richárd Jánki
Date: 17.03.2017

Apple Swift Course
Practice 5

MVC model:

Model: - it contains the functionality of the program (classes, properties, methods and the definitons of the tasks
- e.g.: if we have a Calculator Application the "brain" of the calculator is the model. What kind of tasks does it have to do, etc...

View: - the UI part of our application (UI elements)
- e.g.: if we have a Calculator Application, the View is responsible for the appearance of the display, the buttons at the right position

Controller: - the link between the View and the Model
- add some functionalities to the UI elements
- e.g.: if we have a Calculator Application, after pressing the + button, call the appropriate method (`Add()`)

Example application to use the MVC model (PizzaDemo):

- Let's create a new project! (iOS → Single View Application)

What we get:

~ *AppDelegate.swift:* controls the lifecycle of the application
~ *Assets.xcassets:* the folder of the added external files (e.g.: pictures)
~ *Info.plist:* it contains the informations about our application (e.g.: production name, versions, etc...)
~ *ViewController.swift:* we can implement the Controller of our application here
~ *Main.storyboard:* we can create the View part of our application here

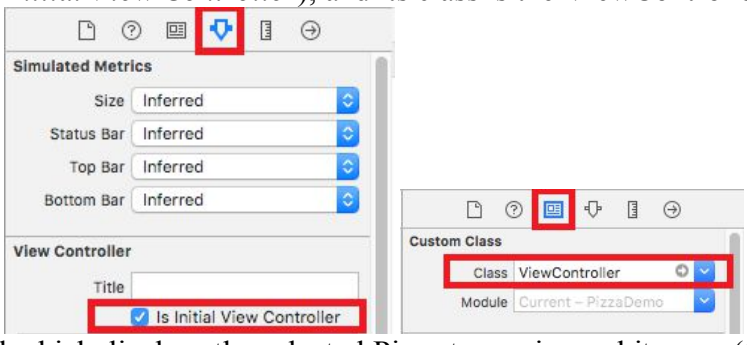
~ The above mentioned files are organized into one folder. After the compilation we get an executable .app file which is in the Products folder.
~ The folders are embedded in the project.
~ If we would like to work with more than one project, we have to push them into a Workspace.

- Let's create a new Swift file, and place it next to the other .swift files! Give the Pizza.swift name to the file!

~ In this file, we should define the Pizza class (`class Pizza {}`), its attributes and its methods.
~ This file contains the *Model* part of the MVC.

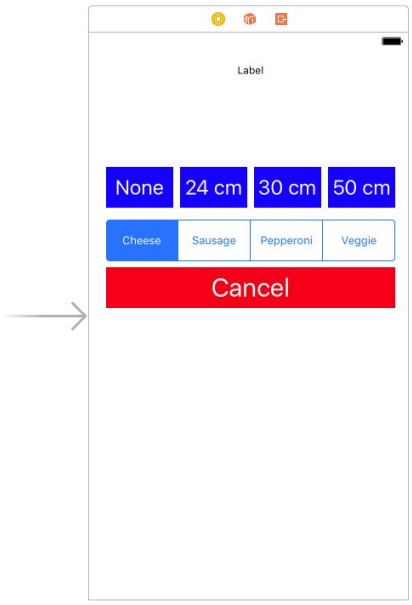
- Let's design the layout of our application in the Main.storyboard!

~ First of all, make sure that the View Controller object on the storyboard is the initial View Controller (*Is Initial View Controller*), and its class is the ViewController.

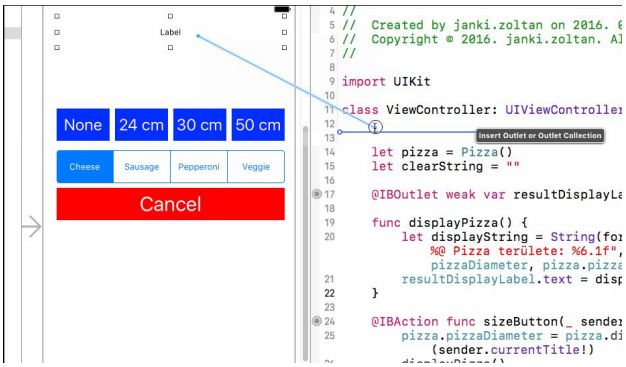


~ Add a Label which displays the selected Pizza type, size and its area (the blue dashed lines help you to decide the positions of the object more precisely). Give an appropriate size to the Label, to make the whole text visible (*Attributes inspector*).

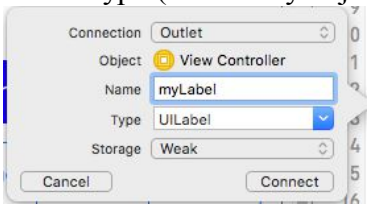
~ Place a button under the label (*Button*), which is used for choosing the size of the pizza. Copy the created button (copy/paste).
~ Under these place a segmented control (*Segmented control*), which helps you to set the type of the pizza. At the attributes of the segmented control, you can give the number of segments and the titles of the segments.
~ At the end, create a new button with the Cancel text and put it under the other elements. The Cancel button will delete the content of the label.



- Let's create references to the UI elements in order to make them accessible in the source code.
~ After opening the storyboard, in the top right corner, switch to the *Assistant editor* (divided screen). With this view, we can see the UI part and the source code at the same time. 
~ The active file in right part should be the **ViewController.swift**, because the View Controller on the storyboard belongs to a class (**`ViewController`**).
~ Let's create references to the UI elements. Right click on the UI element, and drag & drop the element into the source code.



~ In the pop-up window, we have to give Connection type. If the element is used for only visibility things (e.g.: a label shows a text), then it will be an Outlet. If an event will belong to the element (e.g.: clicking on a button), then it will be an Action. Give a reference name to the object and the type of the object is always its own type (never AnyObject!)



- We can give functionalities to the different UI elements by using the references in the source code.

~ We have to do this inside of the ViewController class.

~ The **`viewDidLoad()`** function is actual entry point. If we would like to make some initial view settings, we can set it up here.

- Let's run the application if it's ready!