Home assignment 01

Due date: 13.07.2022., 23:59

Create Home assignment branch `home-assignment-01` from `main` and create pull request from home assignment branch to `main`. How to create a new branch - take a look at Git & Xcode tutorial doc, last chapter.

Intro && Keywords

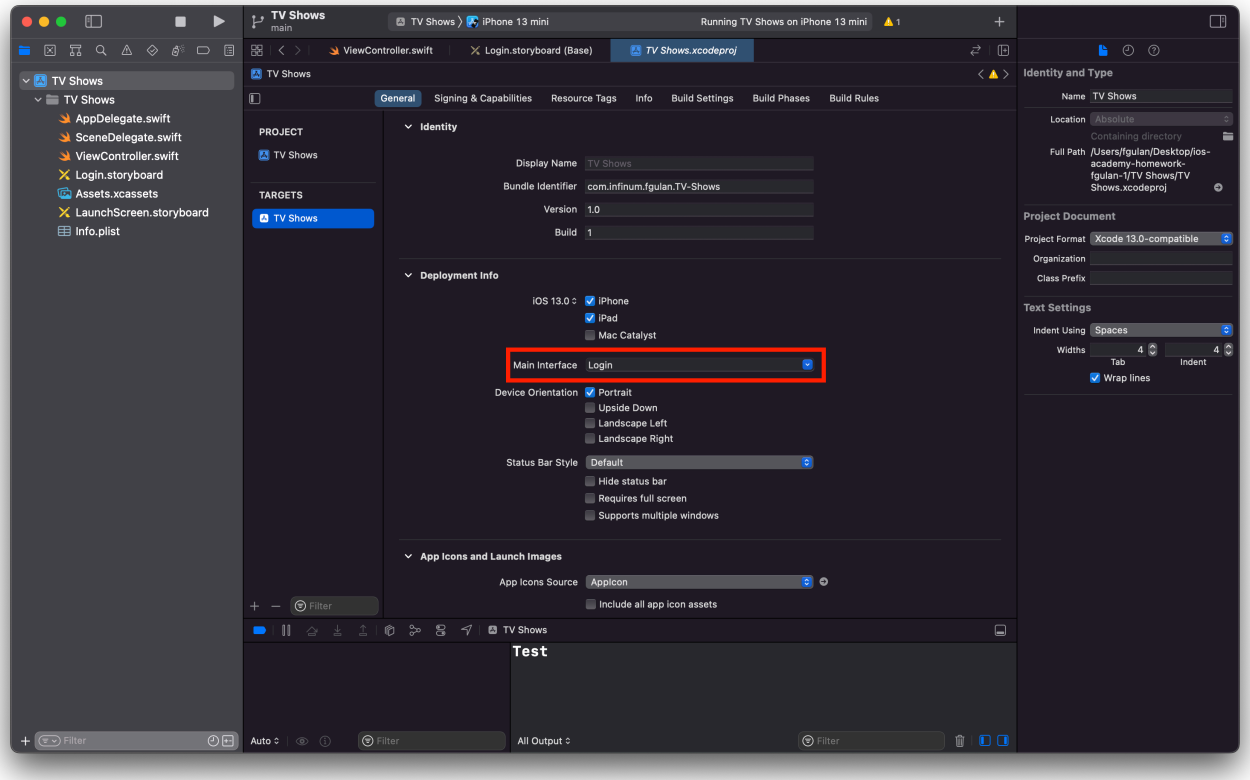
- `@IBAction`
  - function decoration/attribute that will be used so that **Interface Builder** knows how to connect UI element action to the actual code that will handle that.
  - when used with `UIButton` we have two options for connecting
    - `CONTROL + RIGHT CLICK` and then from the drop down menu you would choose `Touch Up Inside` event and drag
    - `CONTROL + LEFT CLICK` on touchpad and drag below `viewDidLoad`
- `@IBOutlet`
  - property decoration/attribute that will be used so that **Interface Builder** knows how to connect UI element to the actual code. The main difference here between `@IBAction` is that this will be used for properties so that you have some sort of reference to the UI element from `Storyboard`. This can come in handy when you need to change the color later on or add rounded edges or something else ...
  - when used with `UIButton` we have two options for connecting
    - `CONTROL + RIGHT CLICK` and then from the drop down menu you would choose `New Referencing Outlet` event and drag
    - `CONTROL + LEFT CLICK` on touchpad and drag above `viewDidLoad` (properties definition always should go on top of the class definition, above all methods)

  For all points above there are videos in materials repository with exact steps how and where to drag those connections

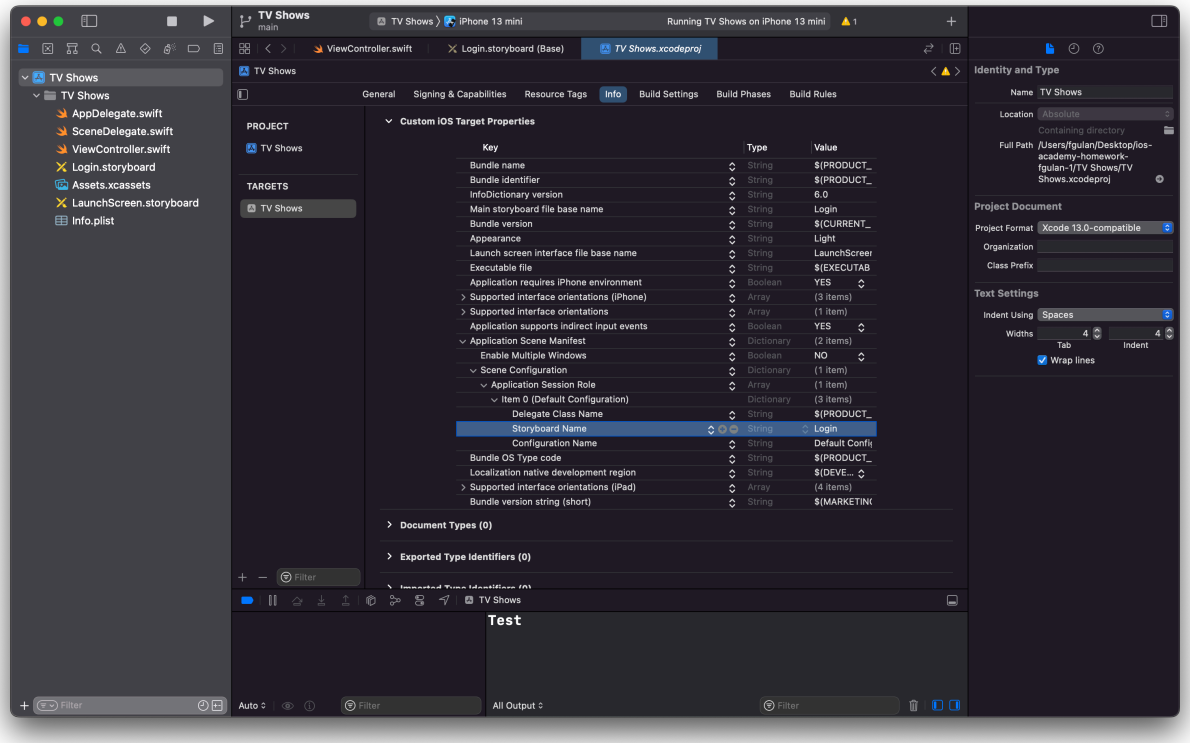
Assignment

- from your current **TVShows** project
  - remove `Main.storyboard` or any other that you added during lecture
  - remove associated `UIViewController` subclasses
  - add new **Group** called **Login**
    - you can do that by `RIGHT CLICK` on the top folder-> `New Group`
  - in that **Group/Folder** add (`RIGHT CLICK` on the folder -> `New File` -> `iOS` on top tab of picker)
    - Login.storyboard
    - LoginViewController.swift

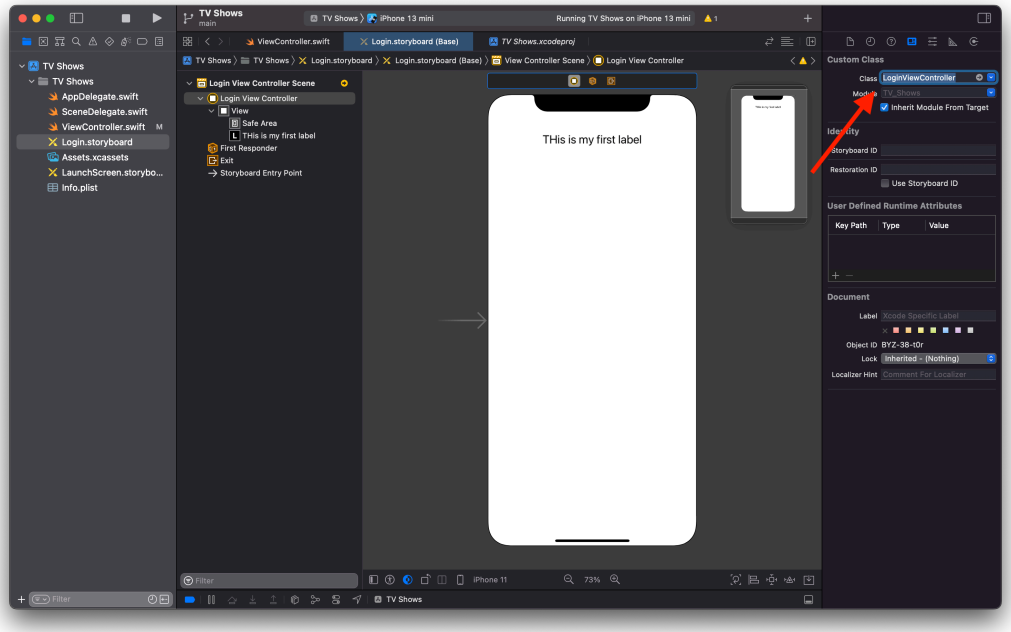
- because we deleted default **storyboard** we need to tell the project which one will be the new default
    - select **TVShows** project on the left (blue icon)
    - under **Targets** select **TVShows**
    - scroll down to the **Main Interface** settings and choose your `Login.storyboard`



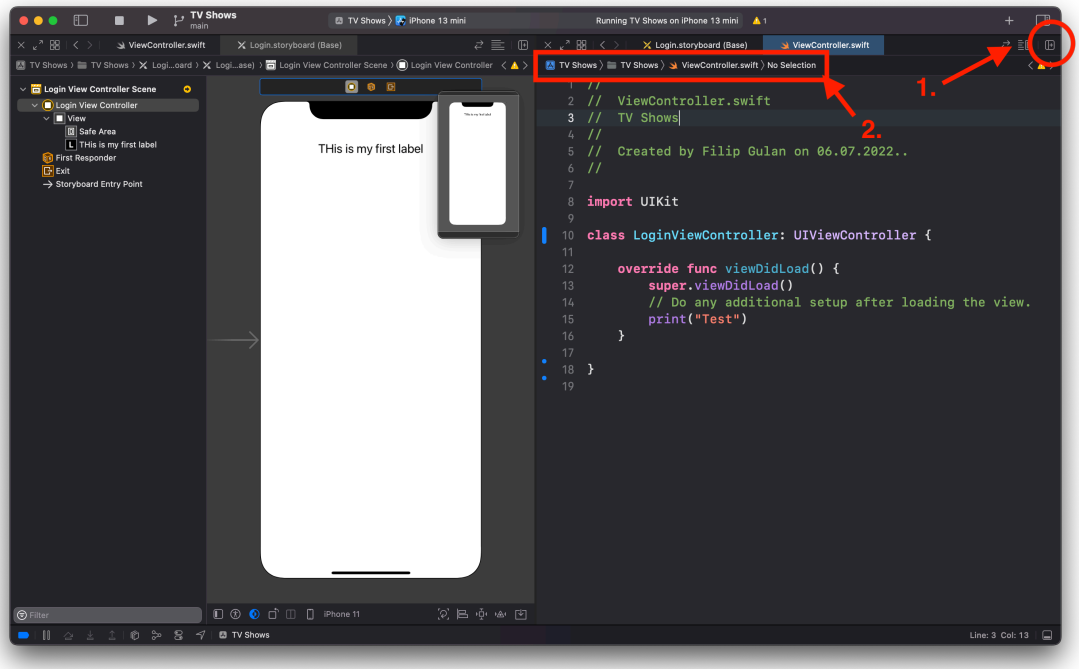
- we'll have to do the same inside `Info` section and change storyboard name to **Login**

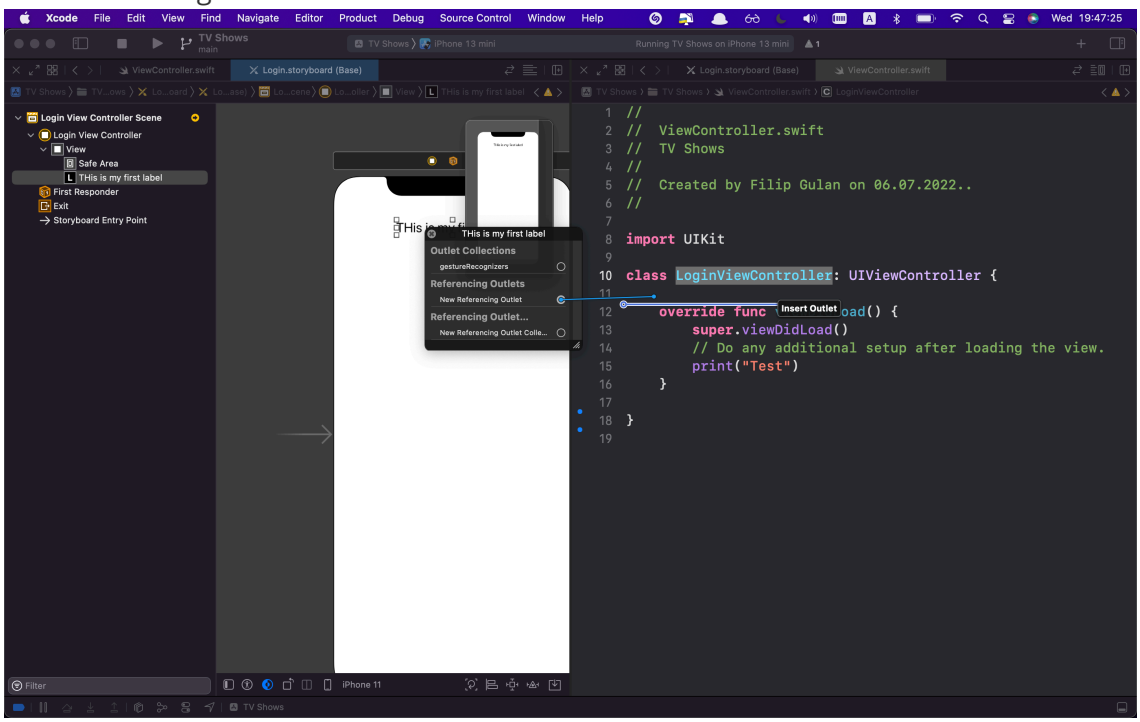


- in `Storyboard` add:
  - view controller UI element
  - button UI element
  - label UI element
- in `Storyboard`:
  - mark view controller UI element as `Is Initial View Controller`
    - you can do that by selecting either `View Controller Scene` or `View Controller` from the left
    - when a view controller is selected you can find that property on the right side, the fifth icon in the row (Attributes Inspector)
    - if successful your view controller will have a right-pointing arrow on its left side :)
  - assign you `UIViewController` subclass called `LoginViewController` to the view controller UI element
    - you can do that by selecting either `View Controller Scene` or `View Controller` from the left
    - when a view controller is selected you can find that property on the right side, the fourth icon in the row (Identity Inspector)
      - `Custom Class` -> `Class`



- in `Storyboard`:
  - select either `View Controller Scene` or `View Controller` from the left
  - press **Add Editor on Right** button that you can find on the topmost bar of Xcode (Toolbar)
    - when you press it you'll get another editor where you can select another file - in this case our LoginViewController.swift



- we need this so that we can connect our UI elements with code ...
- if all is good, you will have side by side view of your storyboard on the left, and code editor on the right
- 

- create `@IBAction` for button's `Touch Up Inside` event
  - that will be the method that will print any string you like to the console
- create `@IBOutlet` for label
  - that will be the property/reference to the label UI element
- update your `@IBAction` function so that on each tap, it will increment the number of taps, and show that in the label UI element
  - you will most likely need to store the number of taps somewhere in the `LoginViewController` in the form of a property
  - `UILabel` element has a property called `text`, which you can use to **set** updates on

each button tap

Assignment Extra (Optional)

- add some bling to your application
  - color the view controller
  - color the button
  - add rounded edges to the button
  - change font and size on the label
  - try to add icon or image to the button by first adding it to the `Assets.xcassets`
- add `Activity Indicator View` UI element and try to figure out how to start/stop
  - add start/stop option on button tap
  - add automatic start when the view appears, and stop 3 seconds later

Notes

- `let` immutable, `var` mutable
  - you will need this for counting number of taps
- we encourage you to:
  - use **StackOverflow** and/or **Apple** documentation
  - `COMMAND + CLICK` on any element in the code editor, and choosing `Jump To Definition`
    - depending who owns the class (You, Apple or somebody else), you will be taken to the header file, where you will have an overview of all the possibilities that are available to you for this class, explore that a bit, and read the descriptions of the functions
  - ask us for help ;)