Lecture-5

- ViewController lifecycle
- Multiple MVC's
  - UITabBarController
  - UISplitViewController
  - UINavigationController
- Demo

View Controller life cycle

- View Controller has a ‘life cycle’

  Sequence of methods that get called at some certain time through out his life cycle
- Why is that important?

  In order to do some certain work you must override some methods in View Controller life cycle
- The start of the life cycle

  Creation of the ViewController.

  Most of the time it’s created in storyboard.

  Very rarely created in code

View Controller life cycle

- Preparation

  This happens if VC is being segued to. Otherwise, it goes with outlet setting after creation

- Outlets get set

- Appearing and disappearing

  This can happen multiple times

- Geometry changes

  This happens whenever our device is changing it's orientation

- Low memory situations

  Almost never happens. If it did, it means your app is consuming a lot memory and you might want to free up some

View Controller life cycle

- After outlets setting, `viewDidLoad` is called

  This is a good place to put all of your set-up code.

```
override func viewDidLoad() {
      super.viewDidLoad()
      // do some set up of your MVC
}
```

One thing you may want to do here is updating your UI, because you are 100% sure that your outlets are set.

DO NOT do here anything related to your geometry. Because you are not sure at this point, what screen size will be established here.

View Controller life cycle

- Right before your view appears on screen, `viewWillAppear` is called

```
override func viewWillAppear(_ animated: Bool) {
      super.viewWillAppear(true)
   // put something here
}
```

This method gets called a lot over time. So DO NOT put here something that should be on viewDidLoad. Otherwise you will be doing unnecessary things

Do something here if things you display is changing, while your screen is OFF. You can also start something expensive in this method, like starting a new thread

View Controller life cycle

- After your view is on screen, viewDidAppear is called

```
override func viewDidAppear(_ animated: Bool) {
      super.viewDidAppear(true)
   // put something here
}
```

Here is not so much to do, but it's may be the good place to start your animation though. Because you know that you're fully on screen, to be interacted with something.

View Controller life cycle

- You also get notified when you will disappear

```
override func viewWillDisAppear(_ animated: Bool) {
     viewWillDisAppear(true)
   // do some clean up code here
   // do not do something time-consuming
}
```

- ‘did’ version of disappearing

```
override func viewDidDisappear(_ animated: Bool) {
     viewDidDisappear(true)
   // usually you undo the things in viewWillAppear
}
```

View Controller life cycle

- Geometry changes?

  Most of the time this is handled by AutoLayout

  But you can get involved with these methods

  ```
  override func viewWillLayoutSubviews()
  override func viewDidLayoutSubviews()
  ```

This methods get called whenever your bounds change, and your view's subviews must be layed-out

AutoLayout happens in between this two methods.

This can happen repeatedly. Not only because of due to the rotation. Even if your bounds don't change, this methods can be called.

View Controller life cycle

- Autorotation

  User can change the orientation switching it to portrait or landscape.

  If you want to participate in autorotation…

```
func viewWillTransition(
          to size: CGSize,
          with coordinator: UIViewControllerTransitionCoordinator
          )
```

Size is the new size for the container's view.

The transition coordinator object managing the size change. You can use this object to animate your changes.

View Controller life cycle

- In low-memory cases, you get notified also …

```
override func didReceiveMemoryWarning() {
     super.didReceiveMemoryWarning()
   // the things that can be recreated should be released here
   // just set your big objects in the heap to be nil
   // this is rarely happens, because iPhones have a lot of memory this days
}
```

View Controller life cycle

- `awakeFromNib`

  This method is sent to all of the objects that come out of the storyboard.
  Happens before viewDidLoad, before outlets are set, even before preparation

View Controller life cycle

- Overview
  - Creation(usually from the storyboard)
  - awakeFromNib
  - segue preparation
  - outlets are set
  - viewDidLoad
    - appearing and disappearing
    - geometry changes
    - didReceiveMemoryWarning

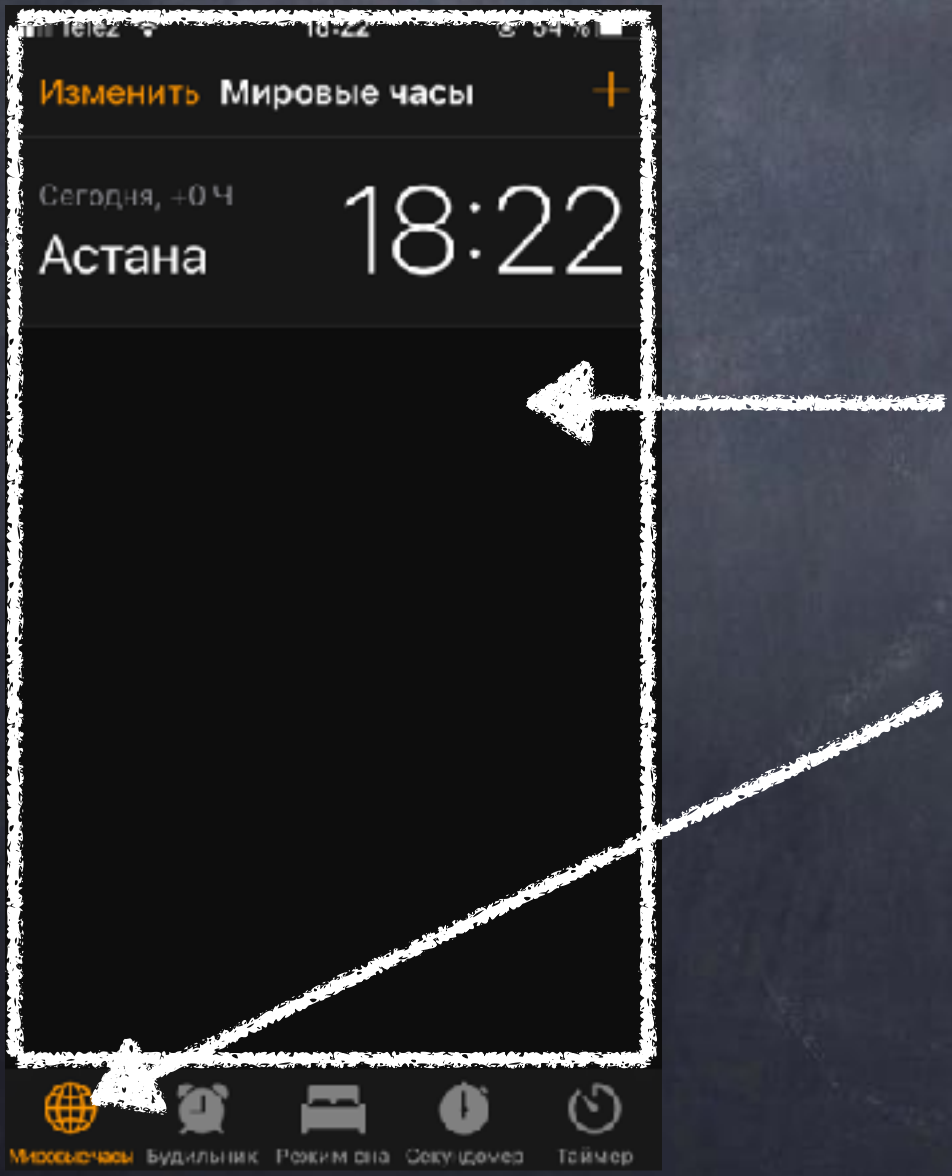
Multiple MVCs

- Time to build more powerful apps
  - to do this must combine MVC's
- iOS actually provides us some multiple MVCs
  - UINavigationController
  - UITabBarController
  - UISplitViewController
- You can create your own multiple MVCs
  - But we will not do this in this class

UITabBarController

- Let's user to choose between different MVCs

  There can be some number of MVCs and they appearances are controller by little tabs at the bottom



WorldClock MVC

This whole space is controlled by WorldClock ViewController, NOT UITabBarController

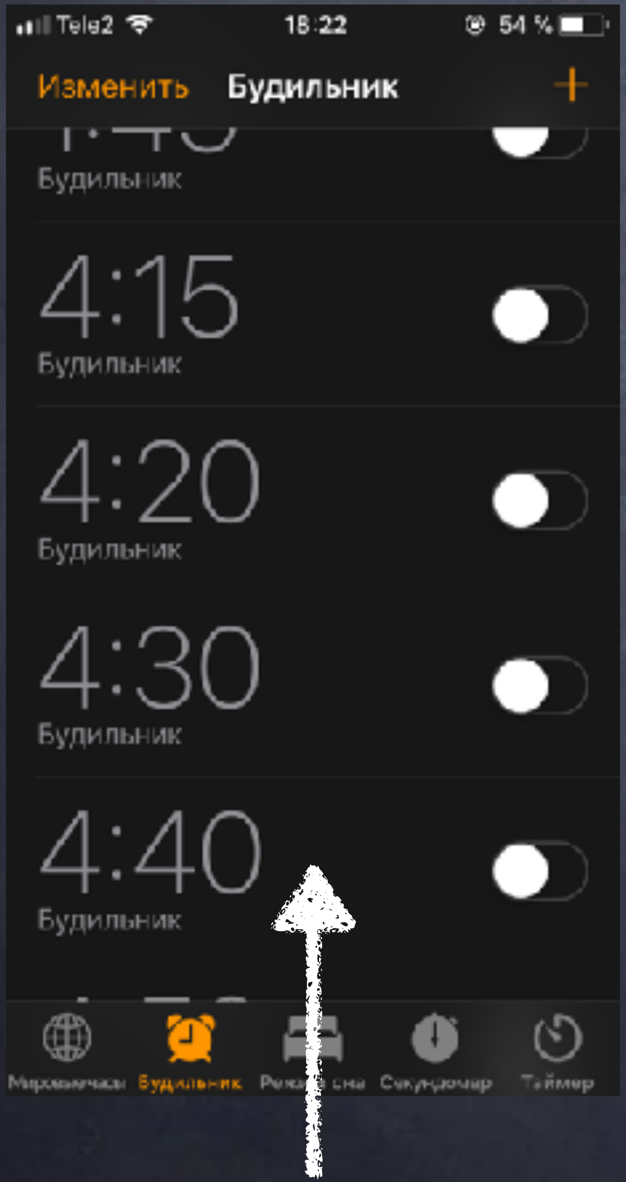
Even the title and the icon of the tab is controlled by WorldClockViewController with special property:

`var tabBarItem: UITabBarItem!`

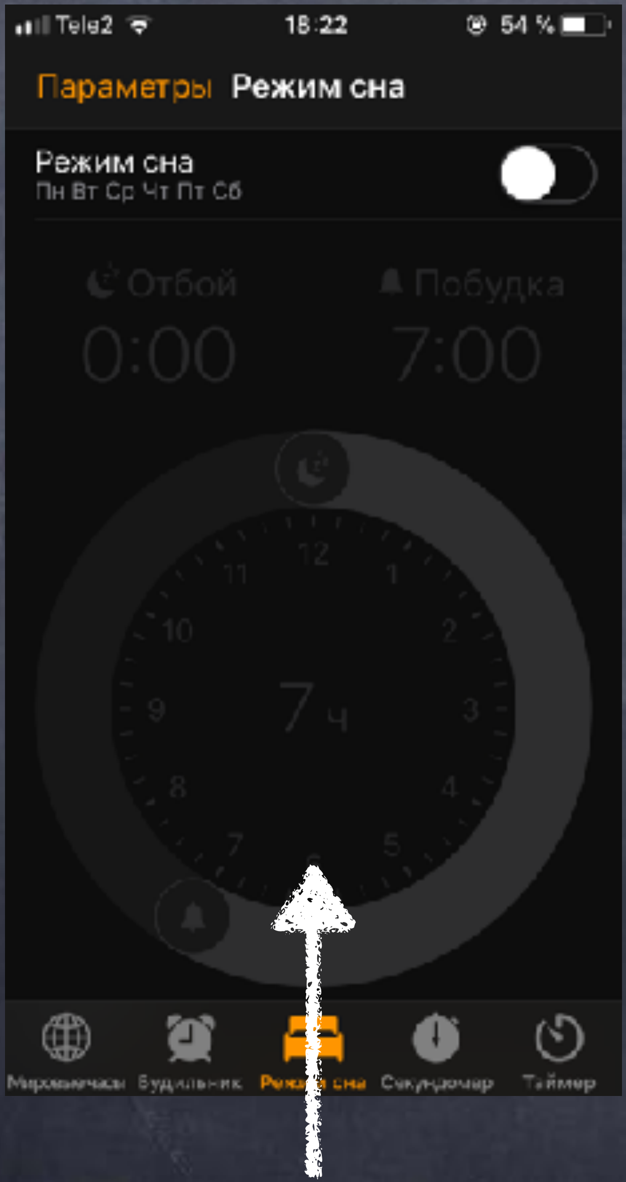
But usually we set them in storyboard

UITabBarController

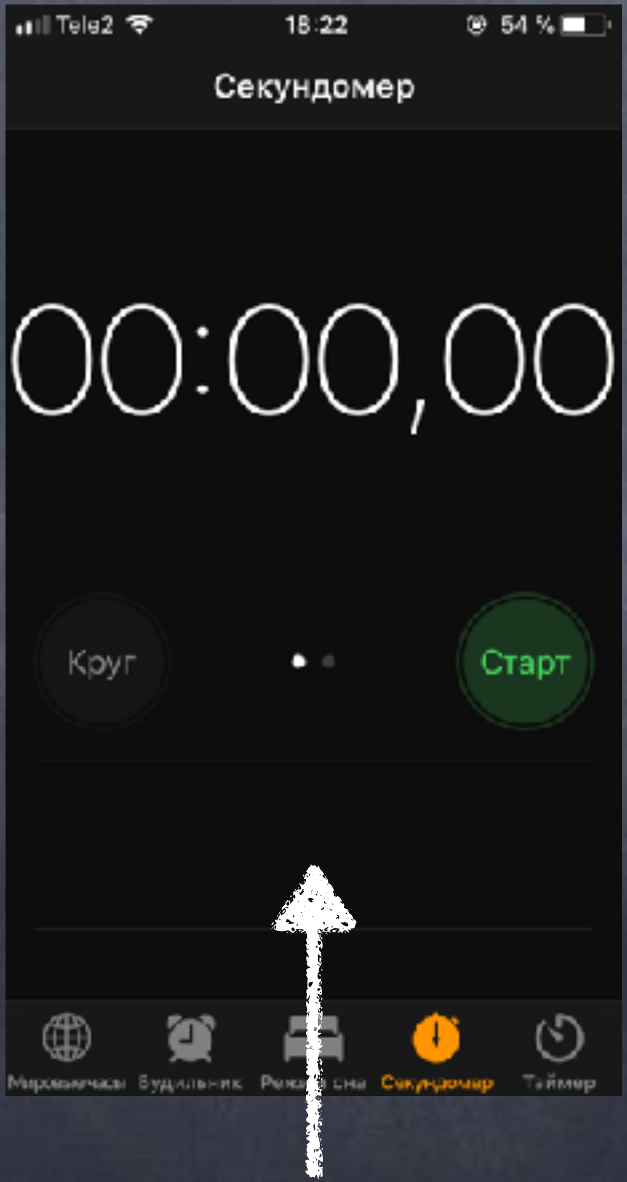
A new tab is a completely difference MVCs



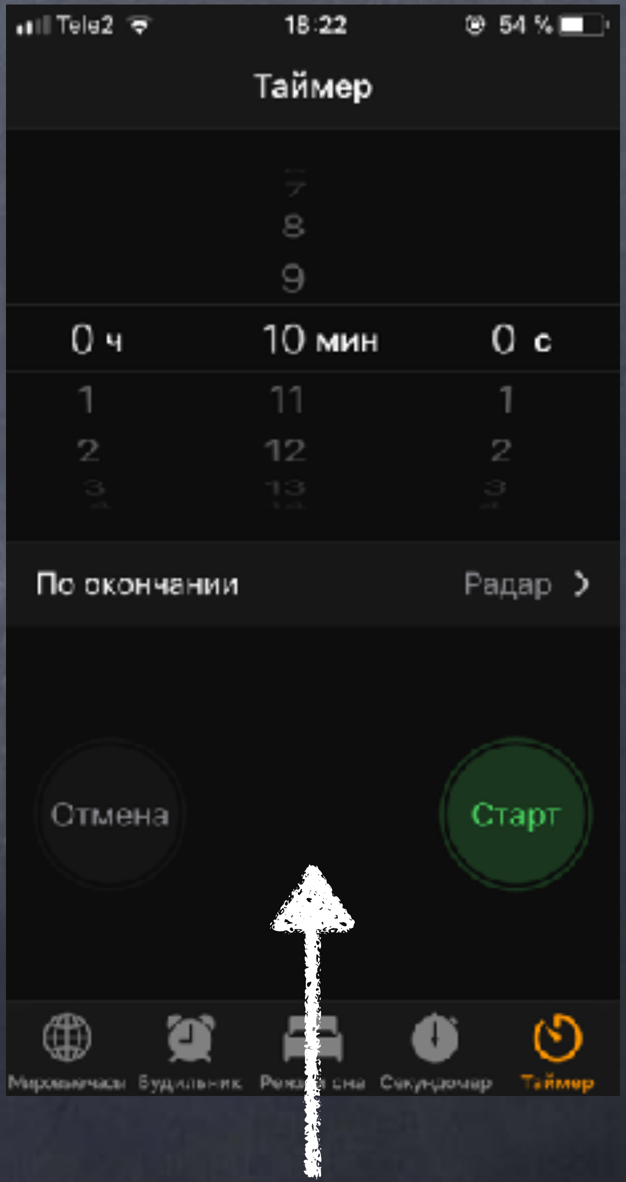
Alarm MVC



BedTime MVC



StopWatch MVC

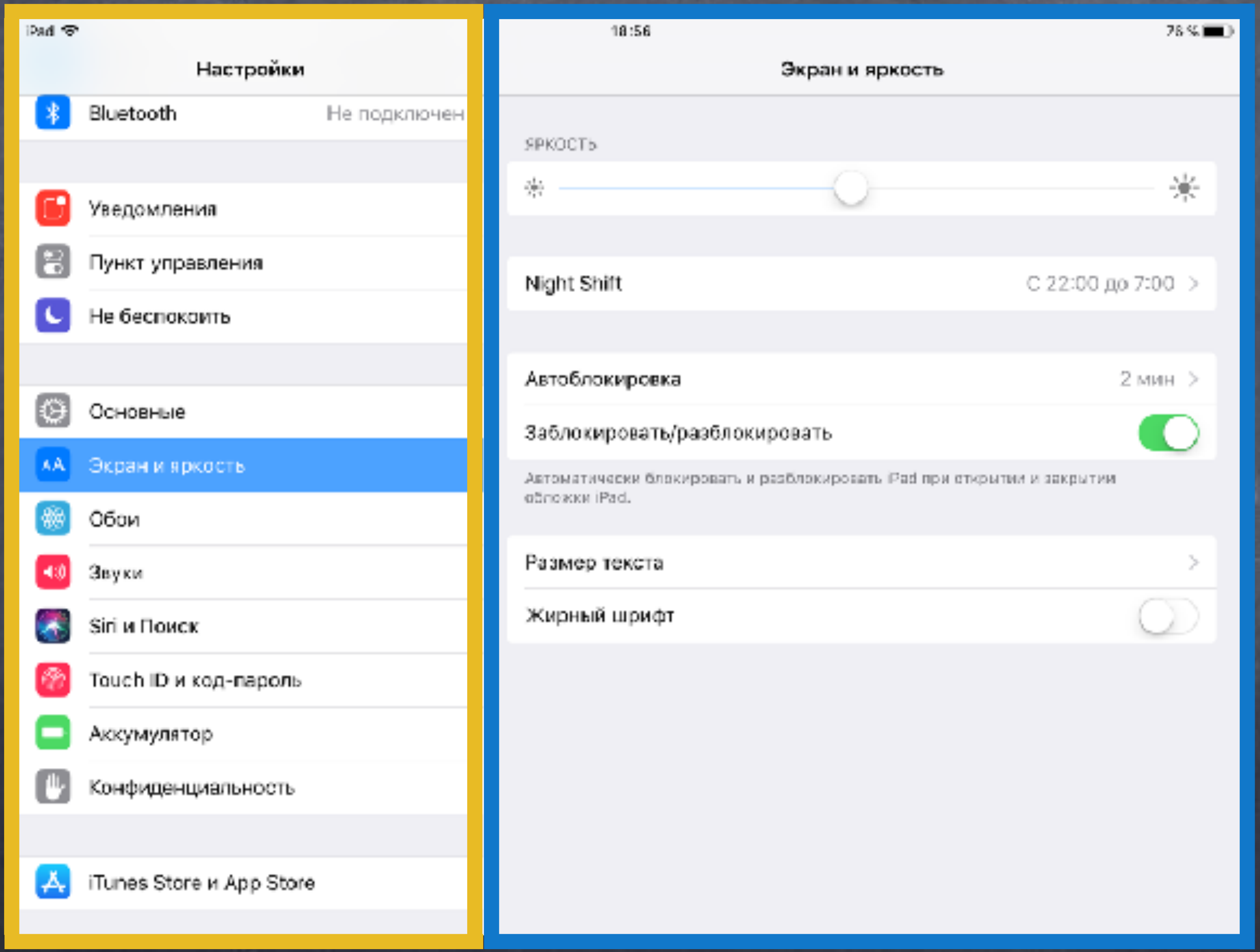


Timer MVC

UISplitViewController

Puts two MVCs side by side

Master


iPad

Настройки

Bluetooth — Не подключен

Уведомления

Пункт управления

Не беспокоить

Основные

Экран и яркость

Обои

Звуки

Siri и Поиск

Touch ID и код-пароль

Аккумулятор

Конфиденциальность

iTunes Store и App Store

18:56 — 78 %

Экран и яркость

ЯРКОСТЬ

Night Shift — С 22:00 до 7:00

Автоблокировка — 2 мин

Заблокировать/разблокировать

Автоматически блокировать и разблокировать iPad при открытии и закрытии обложки iPad.

Размер текста

Жирный шрифт

Detail

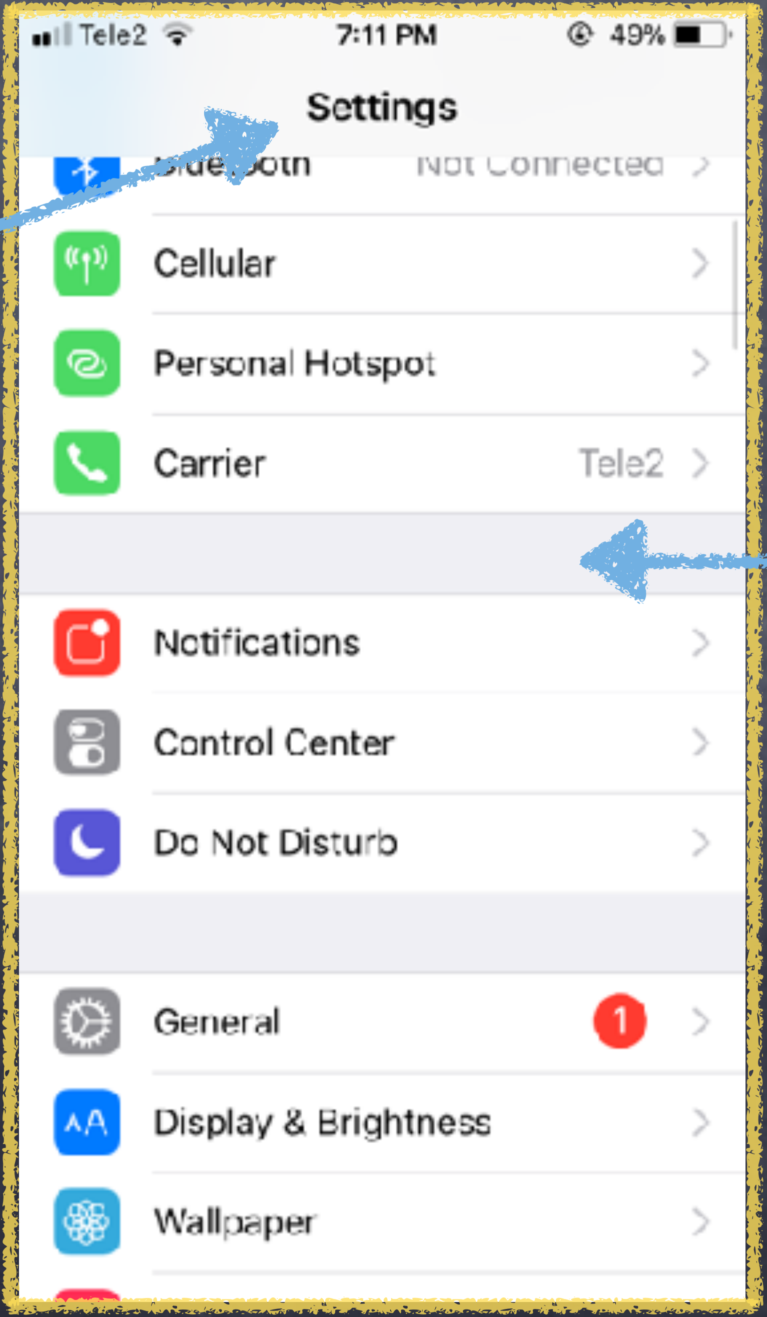
UINavigationController

Pushes and pops the MVCs off of a stack of cards

This title appearance is controlled by navigationController

But it's content is controlled by currently Displayed ViewController

Each MVC communicates With this title content by UIViewController's navigationItem property

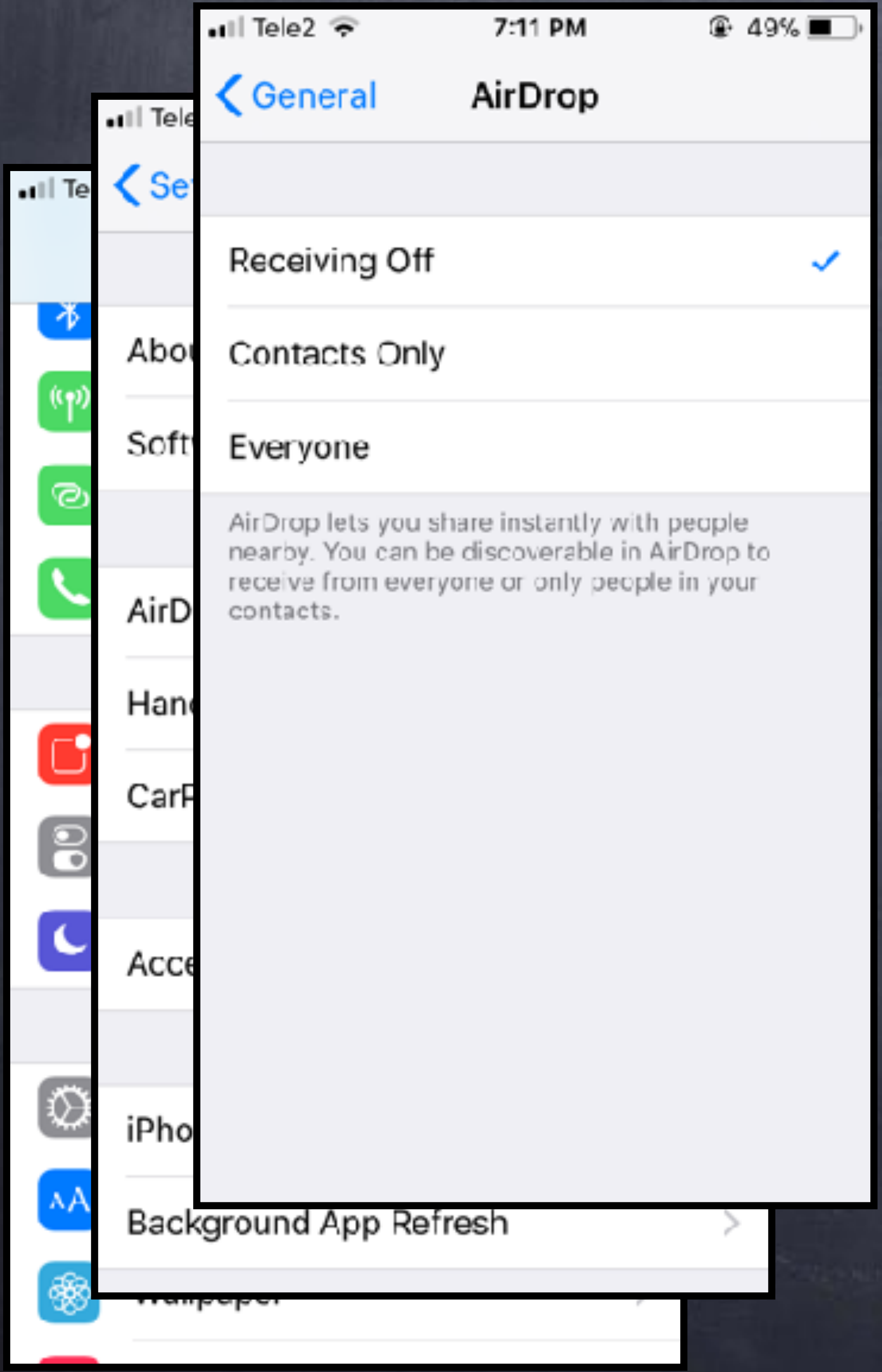

Tele2 7:11 PM 49%

Settings

Bluetooth	Not Connected
Cellular	
Personal Hotspot	
Carrier	Tele2
Notifications	
Control Center	
Do Not Disturb	
General	1
Display & Brightness	
Wallpaper	

All settings MVC

UINavigationController


Tele2 7:11 PM 49%

General AirDrop

Receiving Off ✓

Contacts Only

Everyone

AirDrop lets you share instantly with people nearby. You can be discoverable in AirDrop to receive from everyone or only people in your contacts.

- Now my stack (UINavigationController) has 3 MVCs on top of each other
- Whenever we segue to another MVC, It gets pushed to the stack
- Whenever we go back, my current MVC gets poped from the stack FOREVER

Demo