Adding a view controller as a subview in another view controller

Asked 10 years ago Modified 2 years, 2 months ago Viewed 171k times

Part of Mobile Development Collective

98

I have found few posts for this problem but none of them solved my issue.

Say like I've..

1. ViewControllerA
2. ViewControllerB

I tried to add ViewControllerB as a subview in ViewControllerA but, it's throwing an error like "`fatal error: unexpectedly found nil while unwrapping an Optional value`".

Below is the code...

ViewControllerA

```
var testVC: ViewControllerB = ViewControllerB();

override func viewDidLoad()
{
    super.viewDidLoad()
    self.testVC.view.frame = CGRectMake(0, 0, 350, 450);
    self.view.addSubview(testVC.view);
    // Do any additional setup after loading the view.
}
```

ViewControllerB is just a simple screen with a label in it.

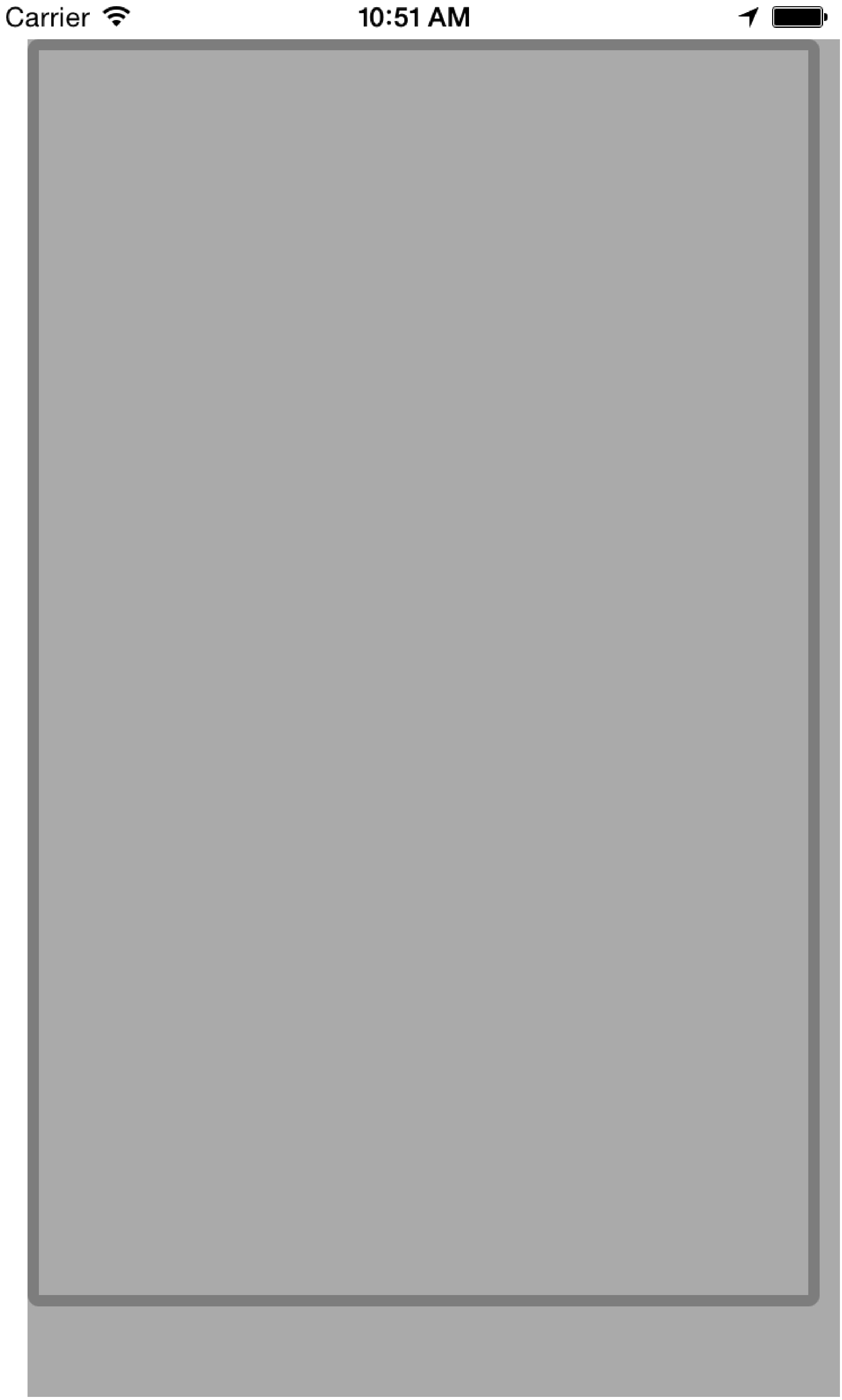
ViewControllerB

```
 @IBOutlet weak var test: UILabel!

override func viewDidLoad() {
    super.viewDidLoad()
    test.text = "Success" // Throws ERROR here "fatal error: unexpectedly found
nil while unwrapping an Optional value"
}
```

EDIT

With the suggested solution from the user answers, ViewControllerB in ViewControllerA is going off the screen. Grey border is the frame I have created for the subview.



Carrier
10:51 AM

MD ios swift uiviewcontroller

Share

Improve this question

Follow

edited Mar 9, 2016 at 2:47

asked Dec 3, 2014 at 16:17

Srujan Simha Adicharla

3,713 ● 8 ● 44 ● 61

8 Answers

Sorted by: Highest score (default)

197

✓

A couple of observations:

1. When you instantiate the second view controller, you are calling `ViewControllerB()`. If that view controller programmatically creates its view (which is unusual) that would be fine. But the presence of the `IBOutlet` suggests that this second view controller's scene was defined in Interface Builder, but by calling `ViewControllerB()`, you are not giving the storyboard a chance to instantiate that scene and hook up all the outlets. Thus the implicitly unwrapped `UILabel` is `nil`, resulting in your error message.

   Instead, you want to give your destination view controller a "storyboard id" in Interface Builder and then you can use `instantiateViewController(withIdentifier:)` to instantiate it (and hook up all of the IB outlets). In Swift 3:

   ```
   let controller = storyboard!.instantiateViewController(withIdentifier: "scen
   storyboard id")
   ```

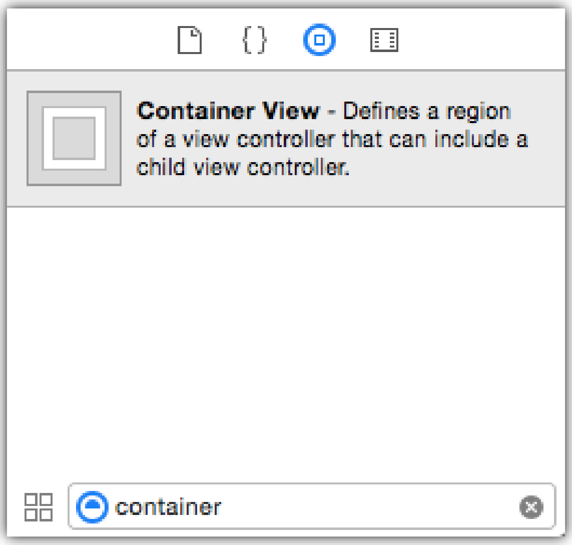
   You can now access this `controller`'s `view`.

2. But if you really want to do `addSubview` (i.e. you're not transitioning to the next scene), then you are engaging in a practice called "view controller containment". You do not just want to simply `addSubview`. You want to do some additional container view controller calls, e.g.:

   ```
   let controller = storyboard!.instantiateViewController(withIdentifier: "scen
   storyboard id")
   addChild(controller)
   controller.view.frame = ...  // or, better, turn off
   `translatesAutoresizingMaskIntoConstraints` and then define constraints for
   this subview
   view.addSubview(controller.view)
   controller.didMove(toParent: self)
   ```

> For more information about why this `addChild` (previously called `addChildViewController`) and `didMove(toParent:)` (previously called `didMove(toParentViewController:)`) are necessary, see WWDC 2011 video #102 - Implementing UIViewController Containment. In short, you need to ensure that your view controller hierarchy stays in sync with your view hierarchy, and these calls to `addChild` and `didMove(toParent:)` ensure this is the case.
>
> Also see Creating Custom Container View Controllers in the *View Controller Programming Guide.*

By the way, the above illustrates how to do this programmatically. It is actually much easier if you use the "container view" in Interface Builder.



Container View - Defines a region of a view controller that can include a child view controller.

container

Then you don't have to worry about any of these containment-related calls, and Interface Builder will take care of it for you.

For Swift 2 implementation, see previous revision of this answer.

Share
Improve this answer
Follow

edited Jan 18, 2019 at 21:42

answered Dec 3, 2014 at 18:25
Rob
436k ● 74 ● 832 ● 1.1k

1 Thank you for the detailed explanation. When I tried adding `ViewControllerB` to `ViewControllerA`, `ViewControllerB` is going off the screen. I have edited my post with the screenshot of the simulator. – Srujan Simha Adicharla Dec 4, 2014 at 17:01

That's possible. That's why, in my example, I set the `frame` manually. Or if you turn off `translatesFrameIntoConstraints` (or whatever it's called), and you can probably add constraints programmatically, too. But if you're adding the subview, you're responsible for setting its frame, one way or the other, just like you are for all programmatically added subviews. – Rob ✪ Dec 4, 2014 at 18:31

1 This is how you can add bounds `controller.view.frame = UIScreen.mainScreen().bounds` – Codetard Mar 3, 2016 at 9:14

3 When doing view controller containment, you really should reference the superview, not the screen. Frankly, now that we have split screen multitasking, doing anything that references the screen is generally inadvisable. – Rob Mar 3, 2016 at 15:42

1 @Honey - I'm not sure what you mean by "assume all that viewController's view is just one subview of the parentViewController". By definition, when you do `addSubview`, the child controller's root view is a subview of the view to which you added it. All you do is add constraints between the child controller's root view and the view to which you just added it as a subview. – Rob Nov 27, 2017 at 20:14

61

Thanks to Rob. Adding detailed syntax for your second observation :

```
let controller:MyView =
self.storyboard!.instantiateViewControllerWithIdentifier("MyView") as! MyView
controller.ANYPROPERTY=THEVALUE // If you want to pass value
controller.view.frame = self.view.bounds
self.view.addSubview(controller.view)
self.addChildViewController(controller)
controller.didMoveToParentViewController(self)
```

And to remove the viewcontroller :

```
self.willMoveToParentViewController(nil)
self.view.removeFromSuperview()
self.removeFromParentViewController()
```

Share

Improve this answer

Follow

edited Nov 2, 2020 at 17:56

Shebuka
3,218 ● 1 ● 27 ● 44

answered Jun 18, 2015 at 5:59

Sunil Lohar
2,242 ● 4 ● 31 ● 46

1 When you do controller.ANYPROPERTY = THEVALUE..I am guessing that AnyProperty is defined in the childViewController. I tried it and it was giving me an error. Any idea how to rectify that. – Anuj Arora Oct 8, 2015 at 8:15

@Anuj Arora your guess is right. ANYPROPERTY is defined in child viewController.You can check ANYPROPERTY in child viewcontroller but in ViewDidAppear not in ViewDidLoad. – Sunita May 19, 2016 at 11:17

14

This code will work for Swift 4.2.

```
let controller = self.storyboard!.instantiateViewController(withIdentifier:
"secondViewController") as! SecondViewController
controller.view.frame = self.view.bounds
self.view.addSubview(controller.view)
self.addChild(controller)
controller.didMove(toParent: self)
```

Share

Improve this answer

Follow

edited Jun 9, 2021 at 13:23

budiDino

13.5k ● 8 ● 99 ● 96

answered Mar 27, 2019 at 18:11

Nirbhay Singh

1,288 ● 1 ● 12 ● 19

5

For Add and Remove ViewController

```
 var secondViewController :SecondViewController?

  // Adding
 func add_ViewController() {
    let controller  =
self.storyboard?.instantiateViewController(withIdentifier:
"secondViewController")as! SecondViewController
    controller.view.frame = self.view.bounds
    self.view.addSubview(controller.view)
    self.addChild(controller)
    controller.didMove(toParent: self)
    self.secondViewController = controller
}

// Removing
func remove_ViewController(secondViewController:SecondViewController?) {
    if secondViewController != nil {
        if self.view.subviews.contains(secondViewController!.view) {
             secondViewController!.view.removeFromSuperview()
        }

    }
}
```

Share

Improve this answer

Follow

edited Nov 2, 2020 at 17:57

Shebuka

3,218 ● 1 ● 27 ● 44

answered Aug 14, 2019 at 5:46

krishnan muthiah pillai

2,721 ● 2 ● 30 ● 35

3

Thanks to Rob, Updated Swift 4.2 syntax

```
let controller:WalletView =
self.storyboard!.instantiateViewController(withIdentifier: "MyView") as!
WalletView
controller.view.frame = self.view.bounds
self.view.addSubview(controller.view)
self.addChild(controller)
controller.didMove(toParent: self)
```

Share

Improve this answer

edited Nov 2, 2020 at 17:57

answered Dec 13, 2018 at 12:25

Follow

Shebuka
3,218 ● 1 ● 27 ● 44

Emre Gürses
2,170 ● 1 ● 25 ● 32

2 use "controller.view.frame = self.view.bounds" instead of "controller.view.frame = self.view.frame" works for me! – Sabrina Jan 6, 2019 at 17:39

2

func callForMenuView() {

```
    if(!isOpen)

    {
        isOpen = true

        let menuVC : MenuViewController =
self.storyboard!.instantiateViewController(withIdentifier: "menu") as!
MenuViewController
        self.view.addSubview(menuVC.view)
        self.addChildViewController(menuVC)
        menuVC.view.layoutIfNeeded()

        menuVC.view.frame=CGRect(x: 0 - UIScreen.main.bounds.size.width, y: 0,
width: UIScreen.main.bounds.size.width-90, height:
UIScreen.main.bounds.size.height);

        UIView.animate(withDuration: 0.3, animations: { () -> Void in
            menuVC.view.frame=CGRect(x: 0, y: 0, width:
UIScreen.main.bounds.size.width-90, height: UIScreen.main.bounds.size.height);
    }, completion:nil)

    }else if(isOpen)
    {
        isOpen = false
      let viewMenuBack : UIView = view.subviews.last!

        UIView.animate(withDuration: 0.3, animations: { () -> Void in
            var frameMenu : CGRect = viewMenuBack.frame
            frameMenu.origin.x = -1 * UIScreen.main.bounds.size.width
            viewMenuBack.frame = frameMenu
            viewMenuBack.layoutIfNeeded()
            viewMenuBack.backgroundColor = UIColor.clear
        }, completion: { (finished) -> Void in
            viewMenuBack.removeFromSuperview()

        })
    }
```

Share Improve this answer Follow

answered Aug 11, 2017 at 8:37

jaya
29 ● 1

Swift 5.1

1

To Add:

```
let controller = storyboard?.instantiateViewController(withIdentifier:
"MyViewControllerId")
addChild(controller!)
controller!.view.frame = self.containerView.bounds
self.containerView.addSubview((controller?.view)!)
controller?.didMove(toParent: self)
```

To remove:

```
self.containerView.subviews.forEach({$0.removeFromSuperview()})
```

Share

Improve this answer

Follow

edited Oct 6, 2022 at 6:46

answered Oct 6, 2022 at 0:41

Naval Hasan

1,480 ● 1 ● 15 ● 21

Please also check the official documentation on implementing a custom container view controller:

0

https://developer.apple.com/library/content/featuredarticles/ViewControllerPGforiPhoneOS/ImplementingaContainerViewController.html#//apple_ref/doc/uid/TP40007457-CH11-SW1

This documentation has much more detailed information for every instruction and also describes how to do add transitions.

Translated to Swift 3:

```
func cycleFromViewController(oldVC: UIViewController,
               newVC: UIViewController) {
   // Prepare the two view controllers for the change.
   oldVC.willMove(toParentViewController: nil)
   addChildViewController(newVC)

   // Get the start frame of the new view controller and the end frame
   // for the old view controller. Both rectangles are offscreen.r
   newVC.view.frame = view.frame.offsetBy(dx: view.frame.width, dy: 0)
   let endFrame = view.frame.offsetBy(dx: -view.frame.width, dy: 0)

   // Queue up the transition animation.
   self.transition(from: oldVC, to: newVC, duration: 0.25, animations: {
        newVC.view.frame = oldVC.view.frame
        oldVC.view.frame = endFrame
    }) { (_: Bool) in
        oldVC.removeFromParentViewController()
        newVC.didMove(toParentViewController: self)
    }
}
```

Share Improve this answer Follow

answered Apr 20, 2017 at 10:43

Simon Backx

1,362 ● 14 ● 16