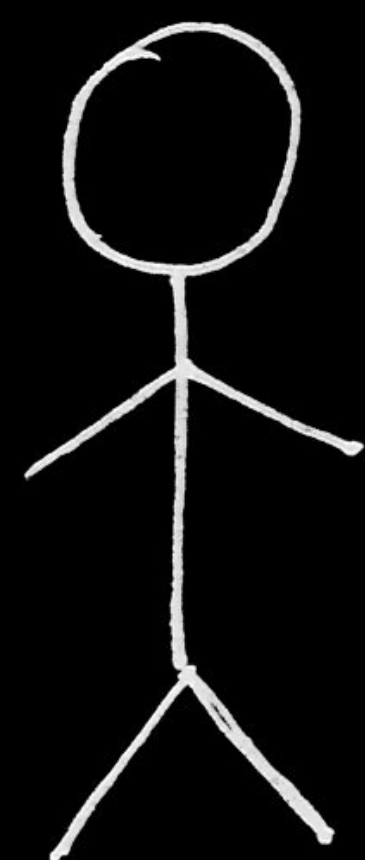


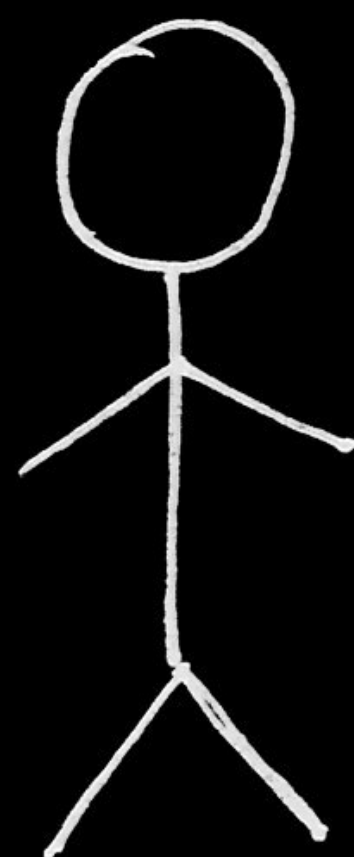
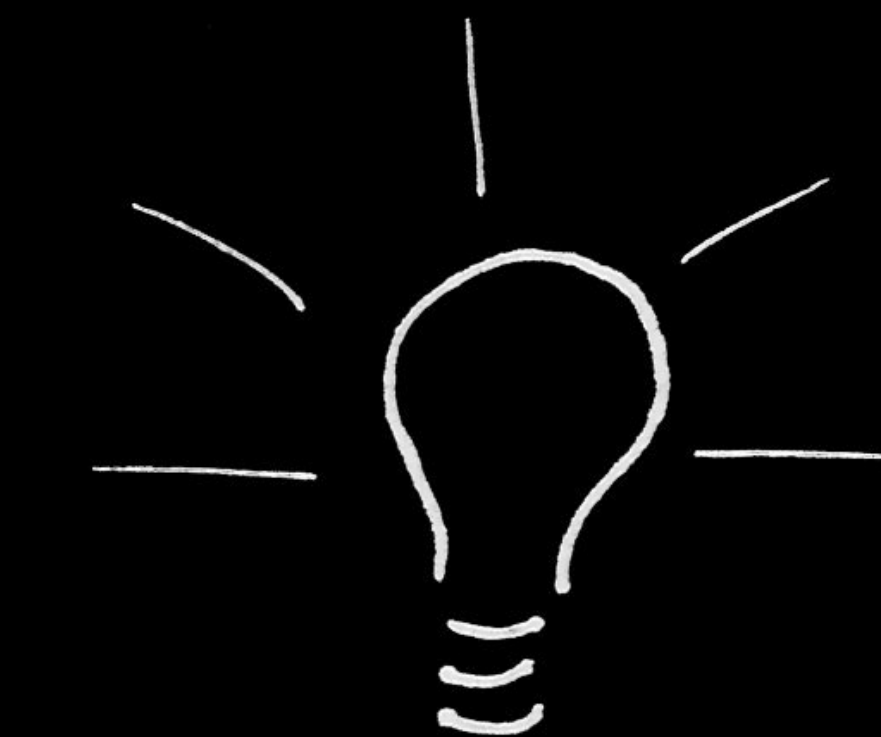
Implementing UI Designs in Interface Builder

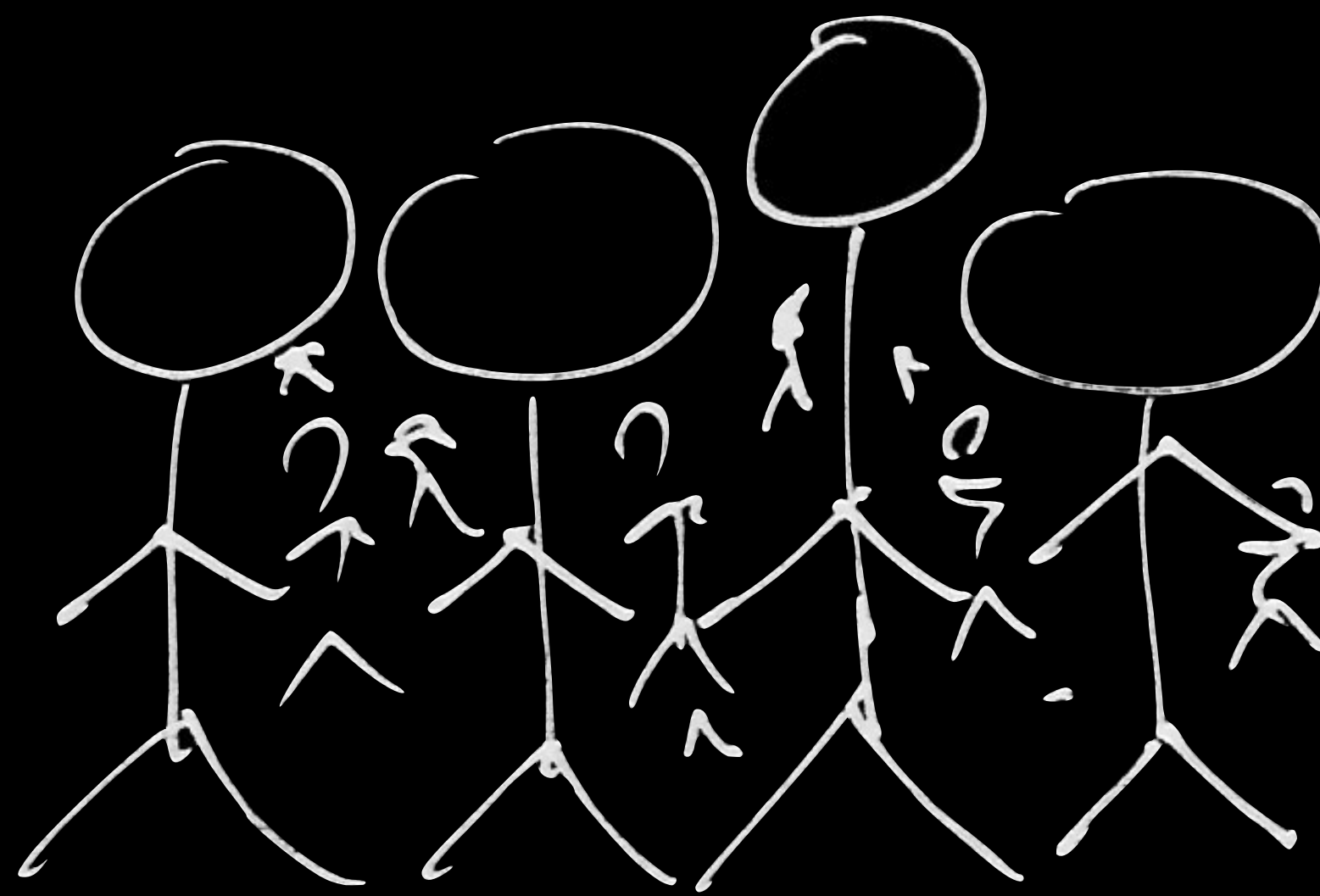
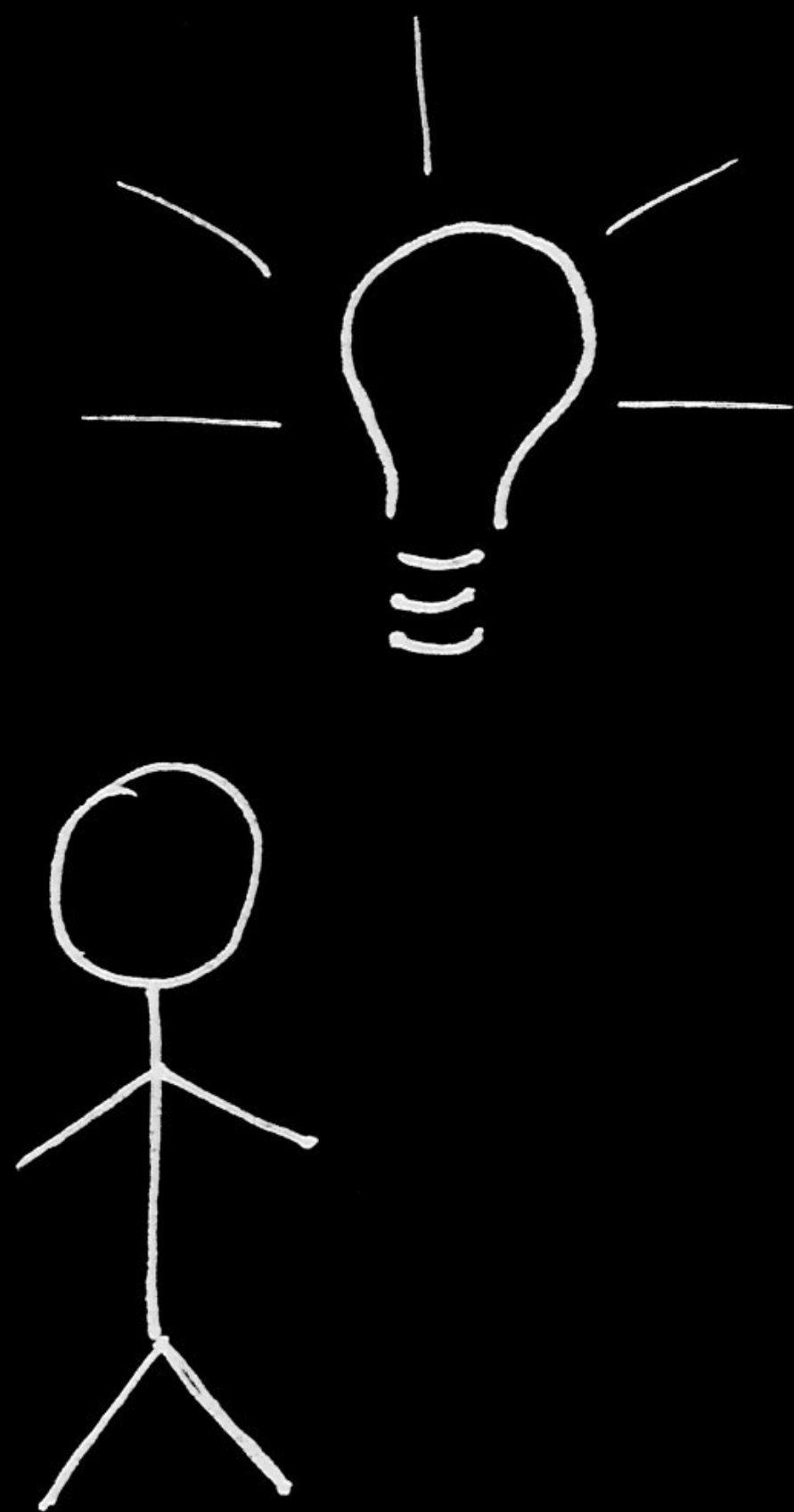
Session 407

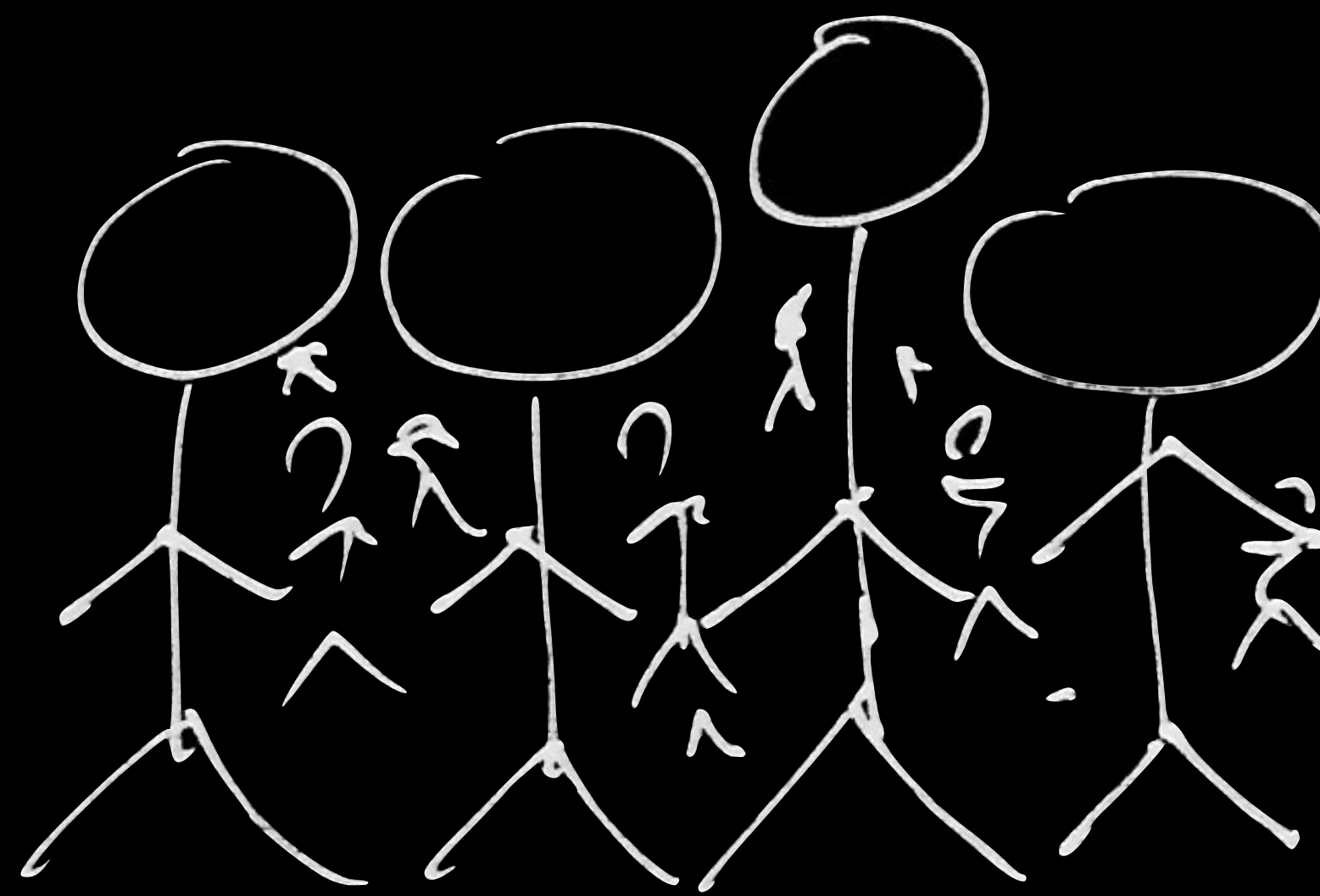
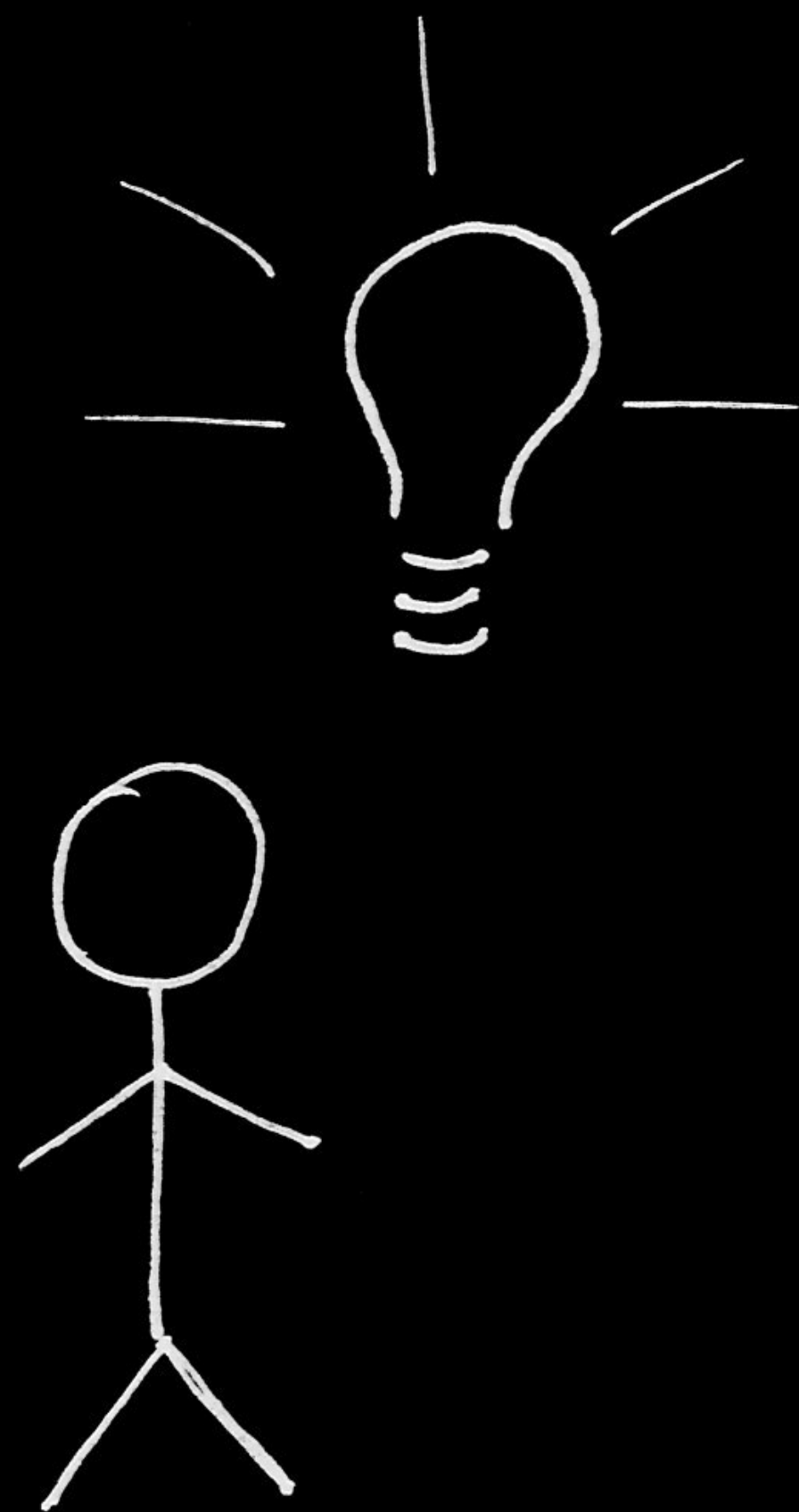
Kevin Cathey Interface Builder Engineer

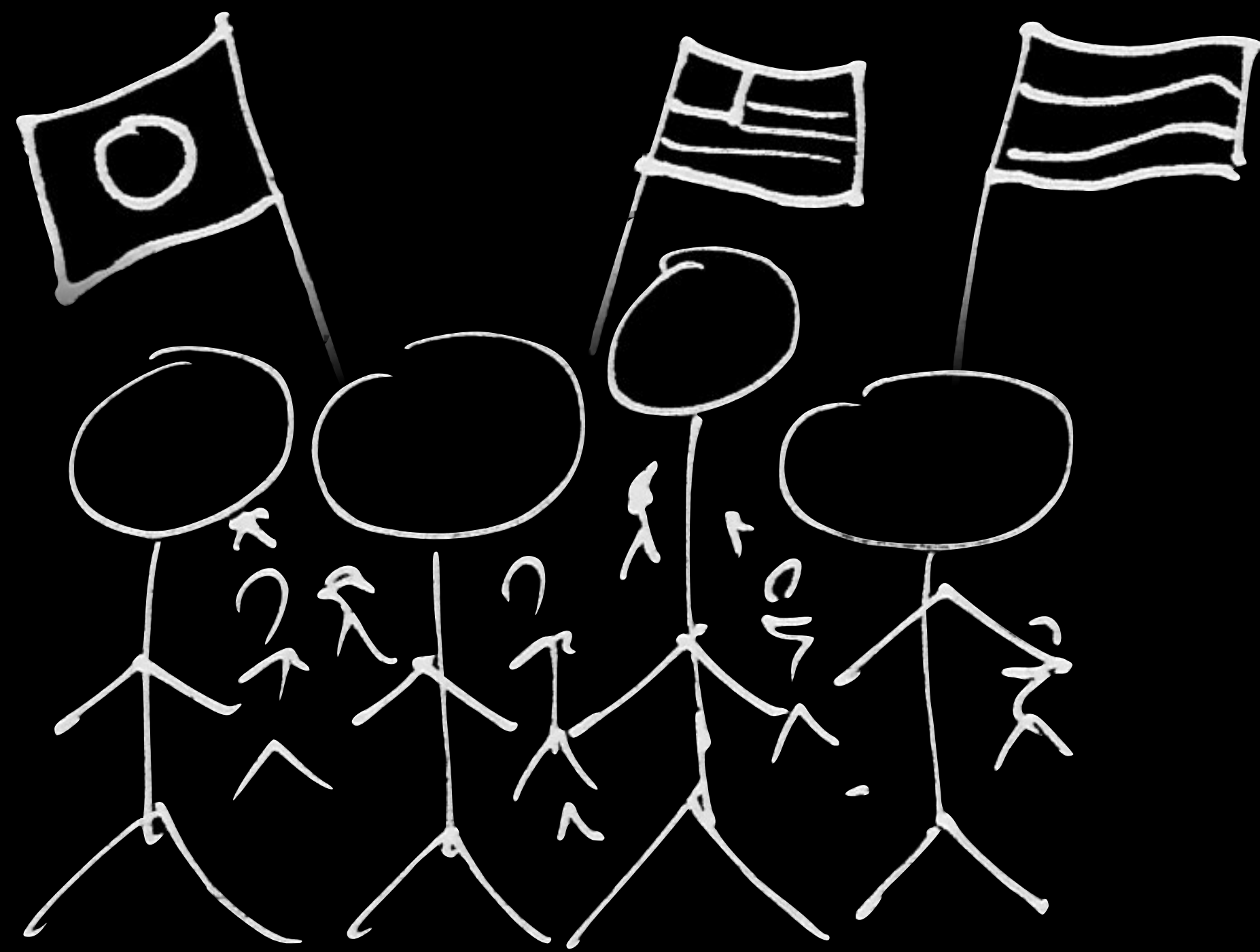
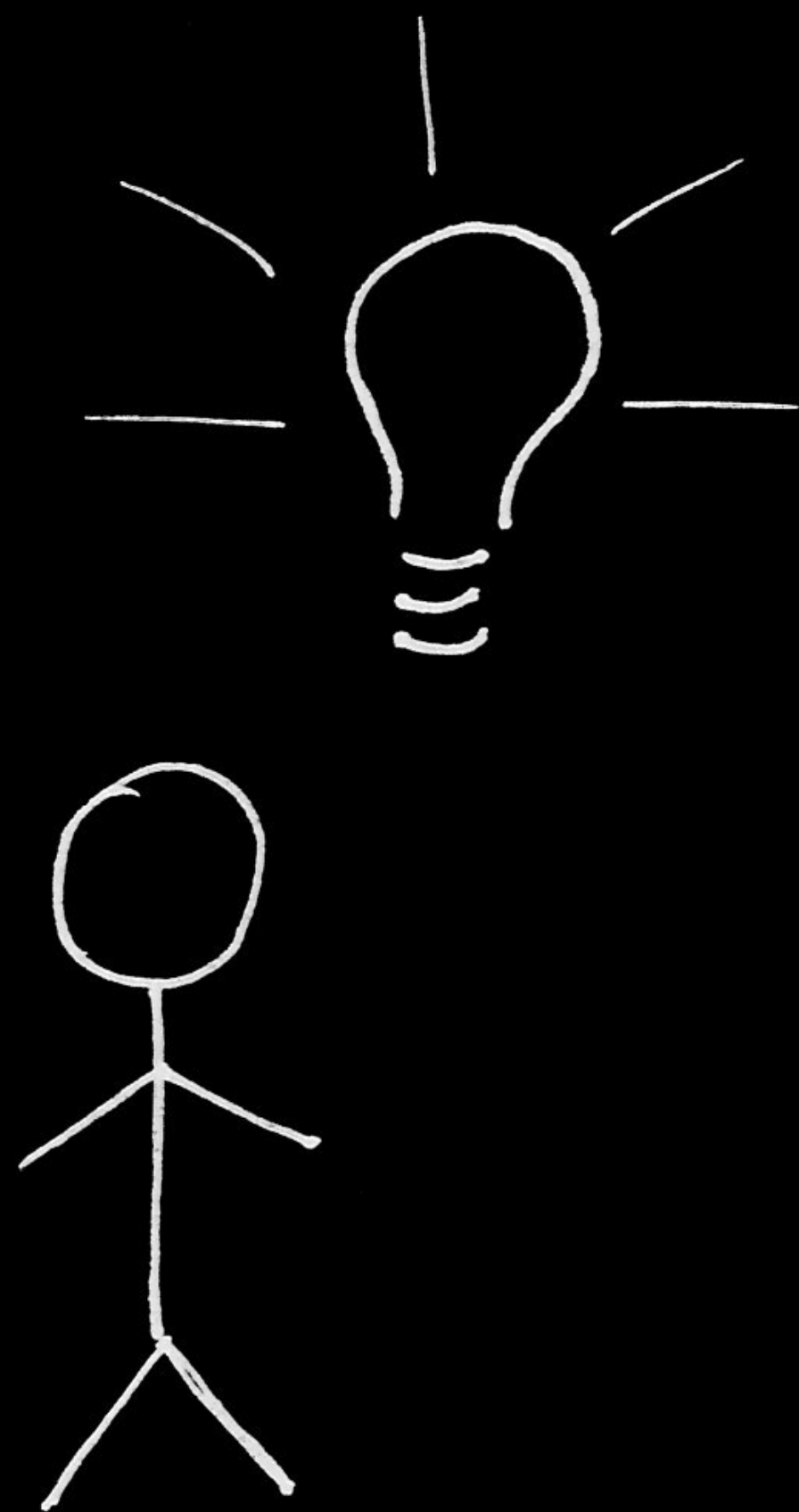
Tony Ricciardi Interface Builder Engineer

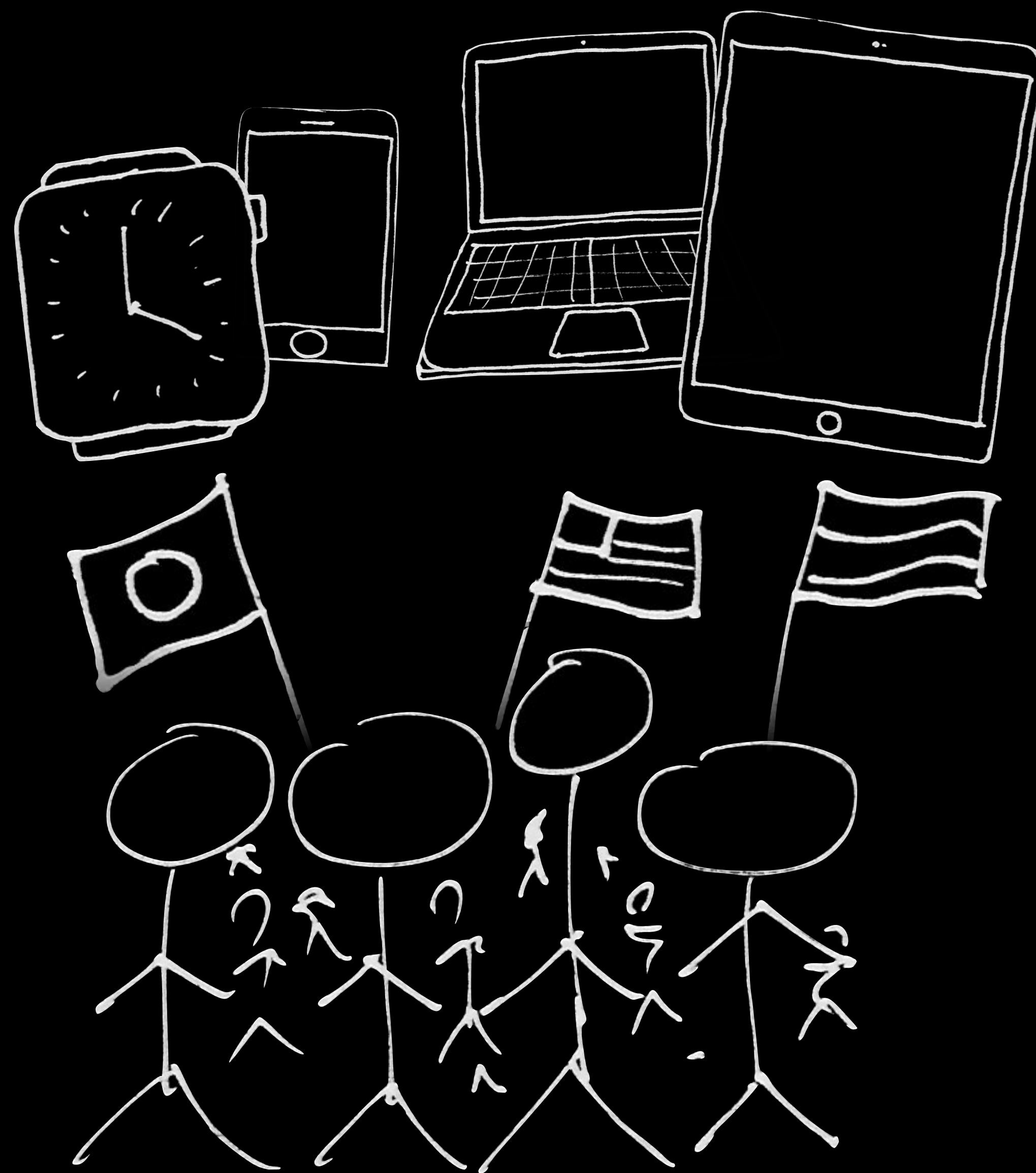
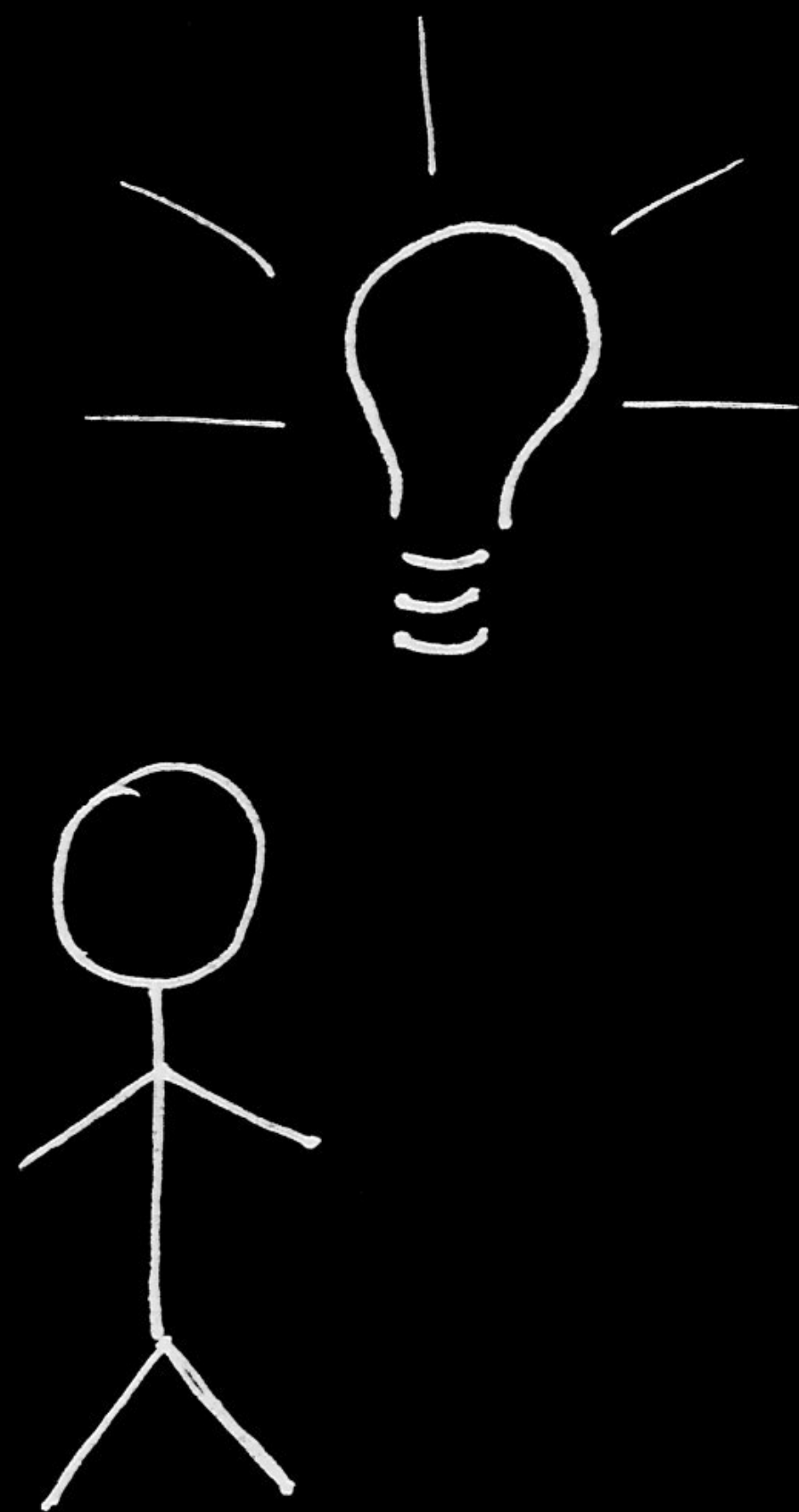


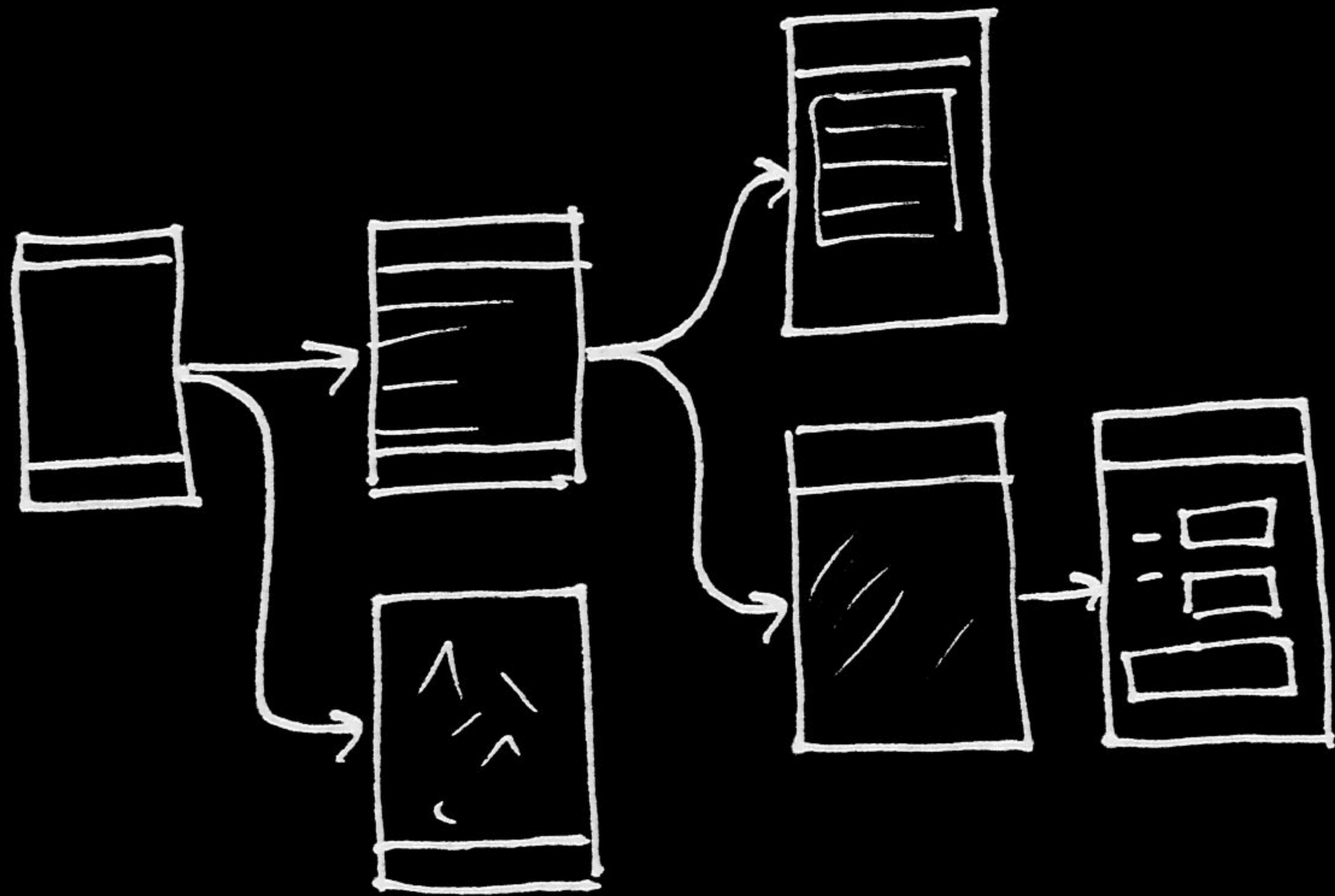












NAME

CIN

SUBMIT

NAME

CIM

SUBMIT

NAME

CIM

SUBMIT



Design Time



Design Time



Build Time



Design Time



Build Time



Run Time



Design Time



Build Time



Run Time

Demo

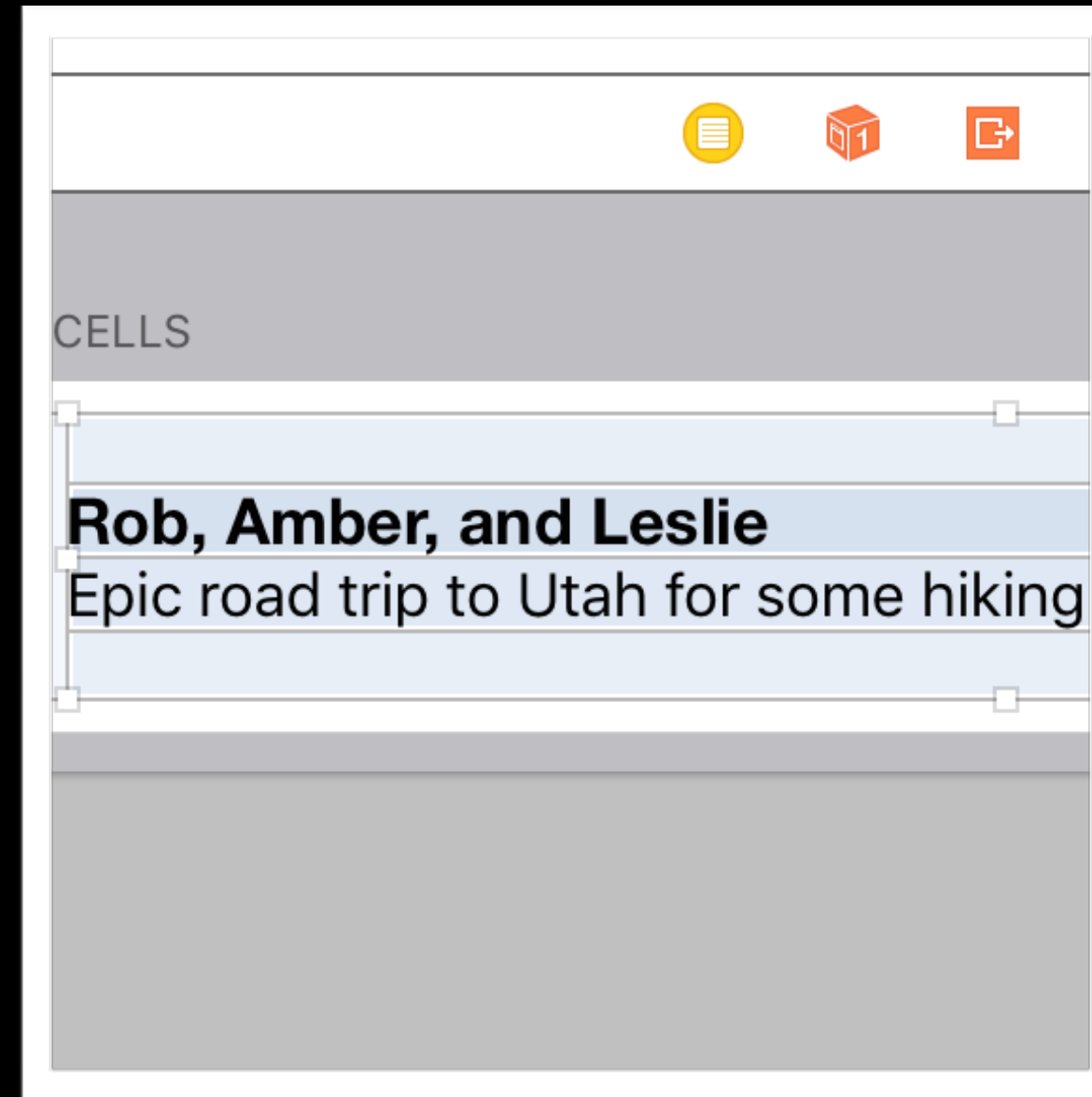
Design Time — Best Practices, Tips and Tricks

Takeaways

Best practices

Takeaways

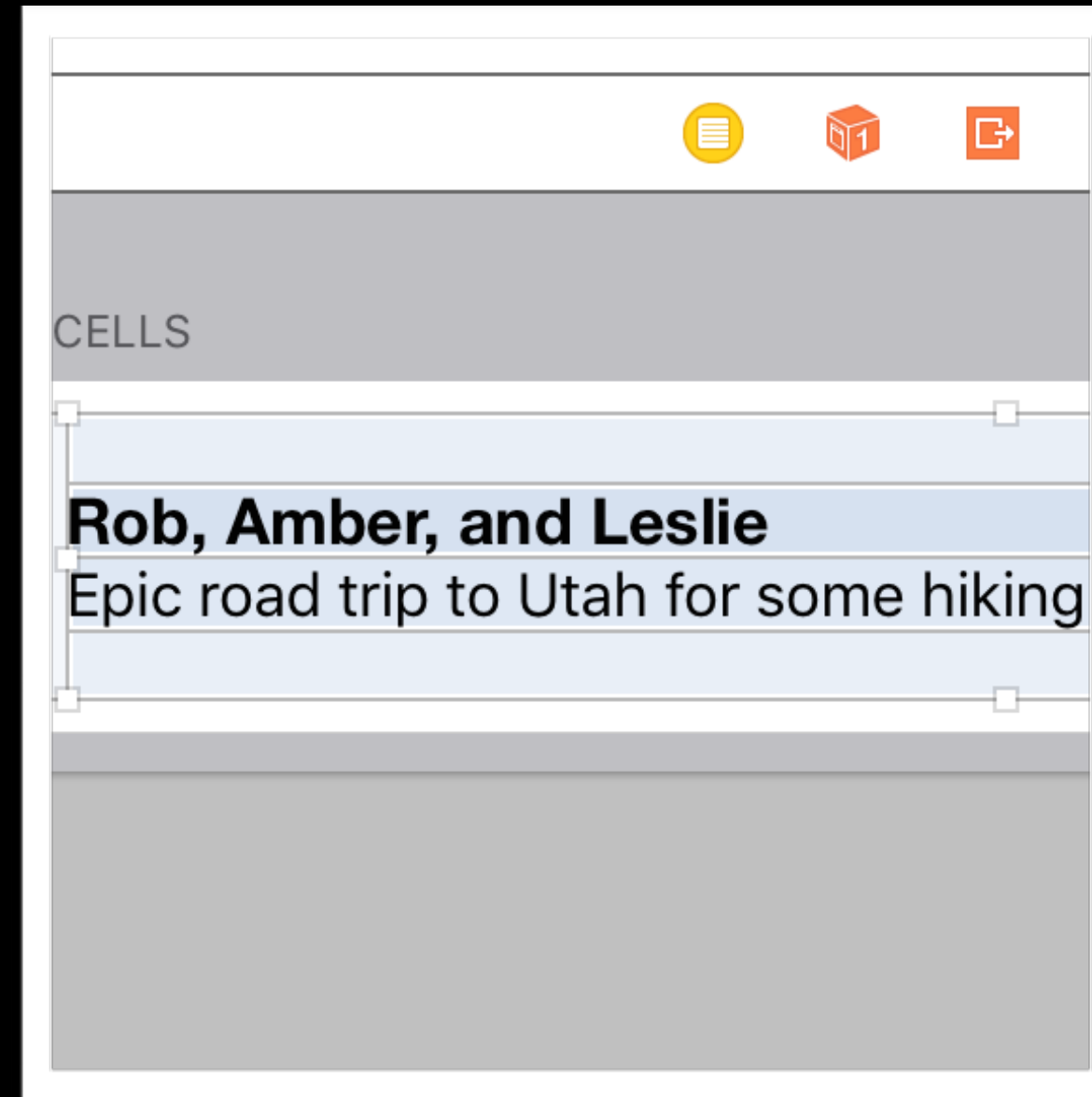
Best practices



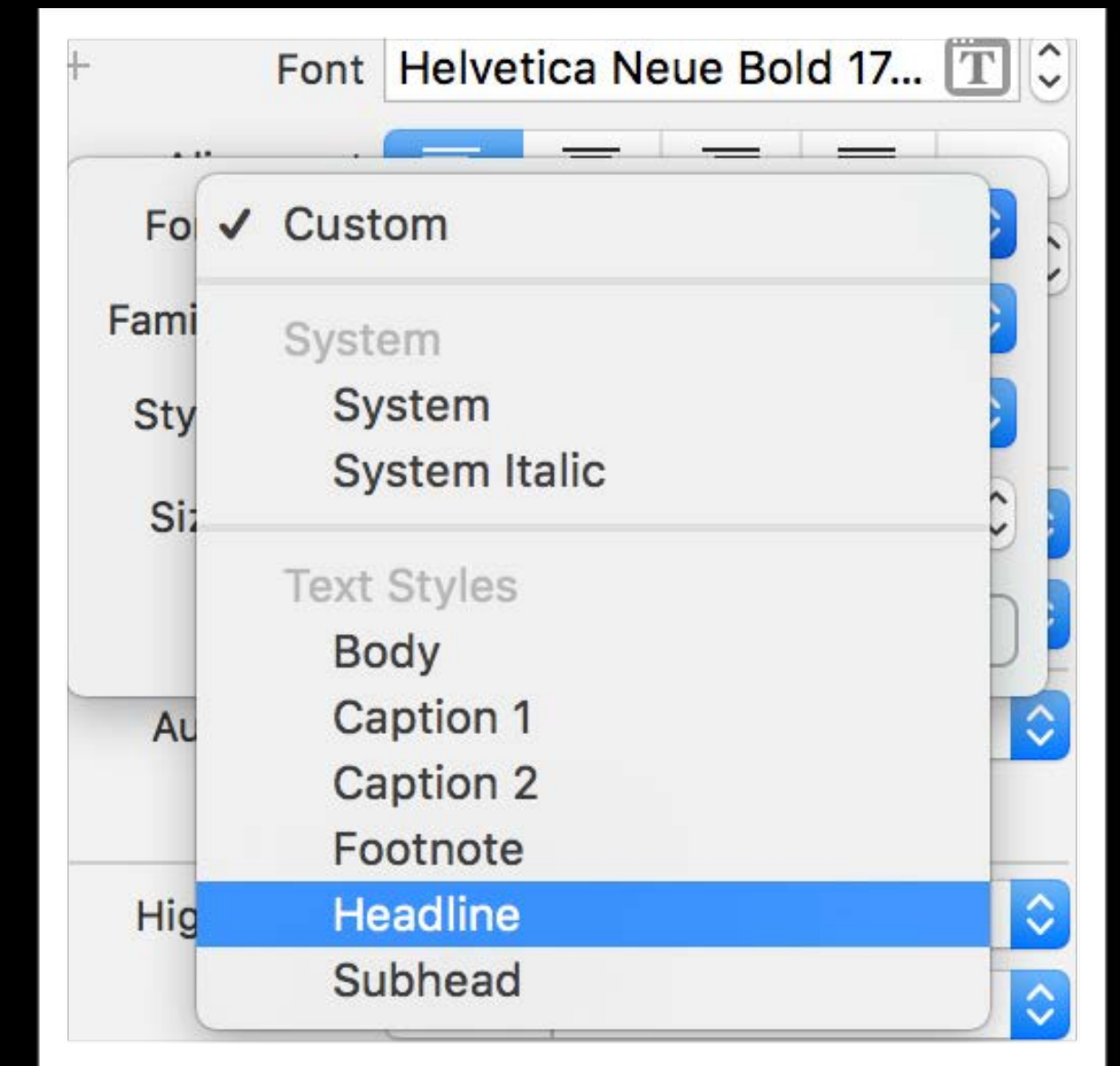
Stack Views

Takeaways

Best practices



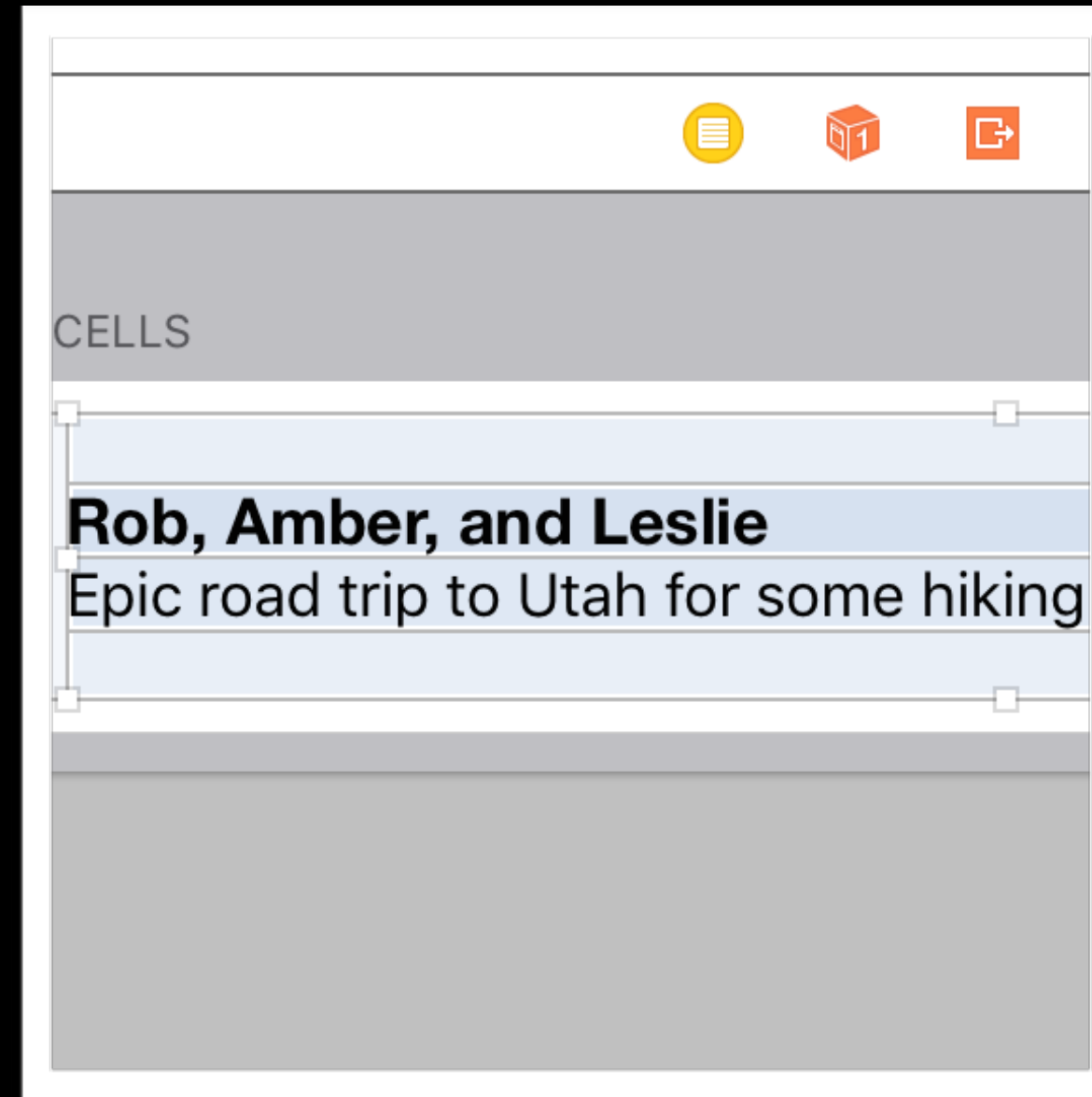
Stack Views



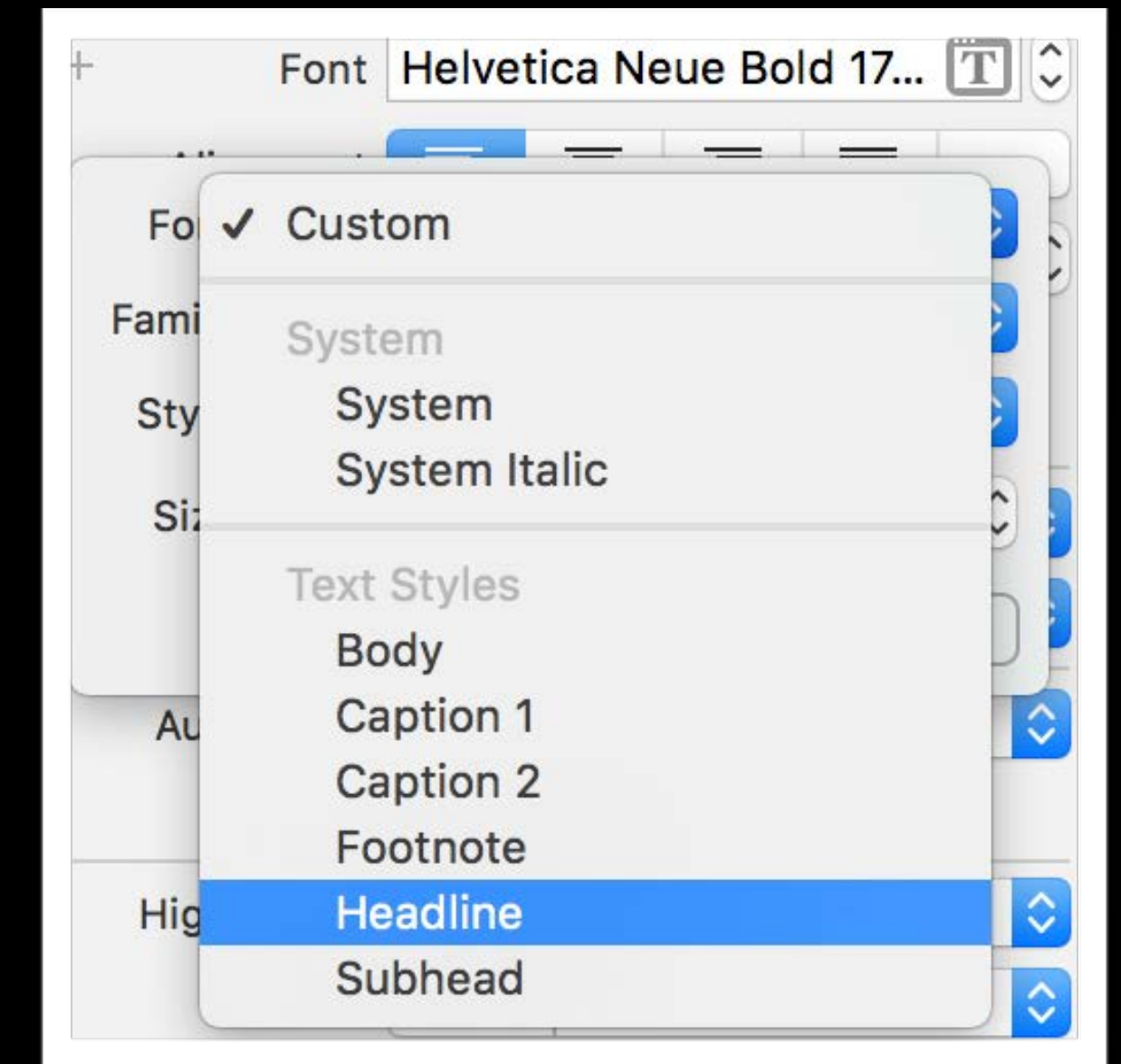
Dynamic Type

Takeaways

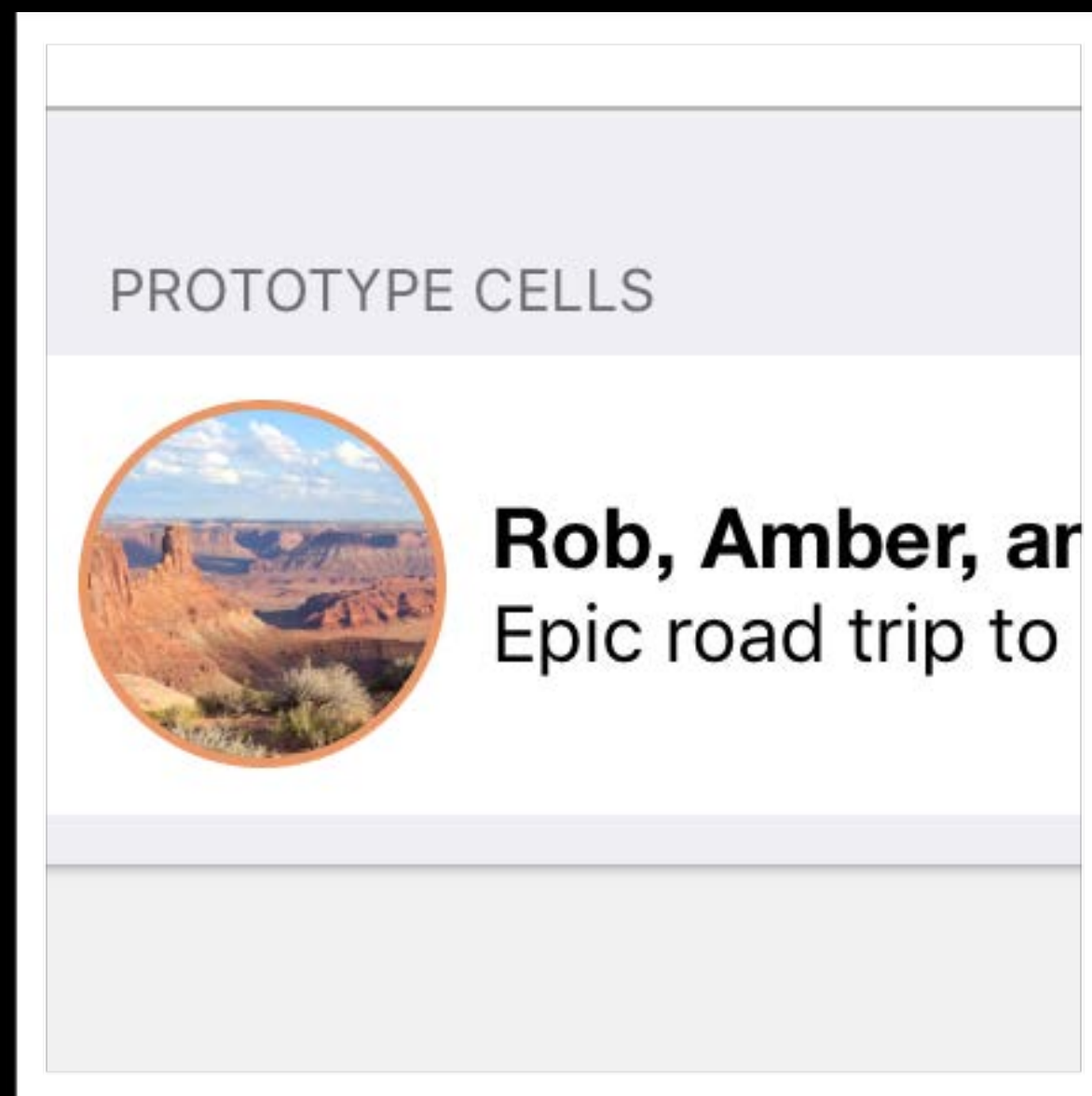
Best practices



Stack Views



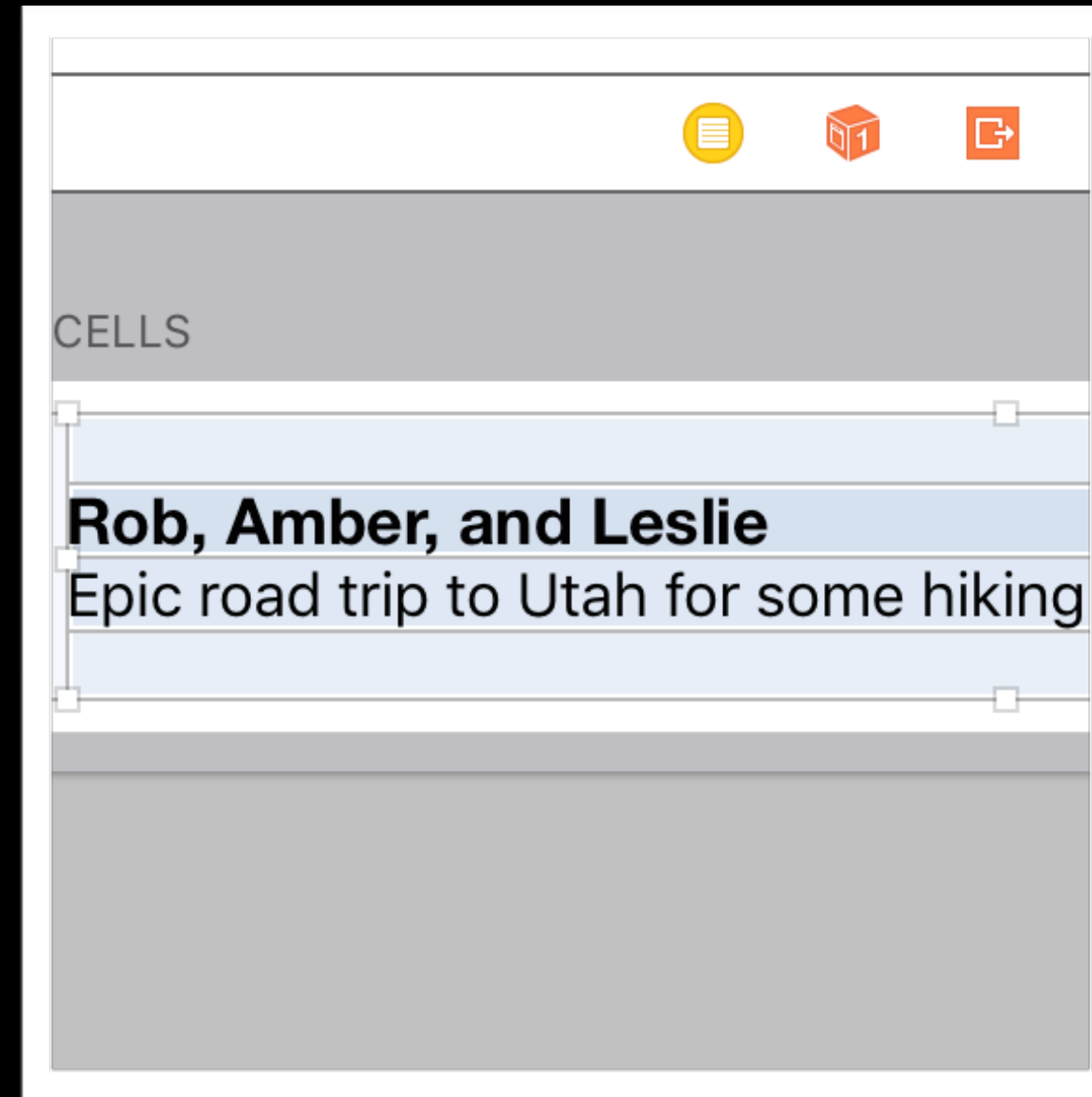
Dynamic Type



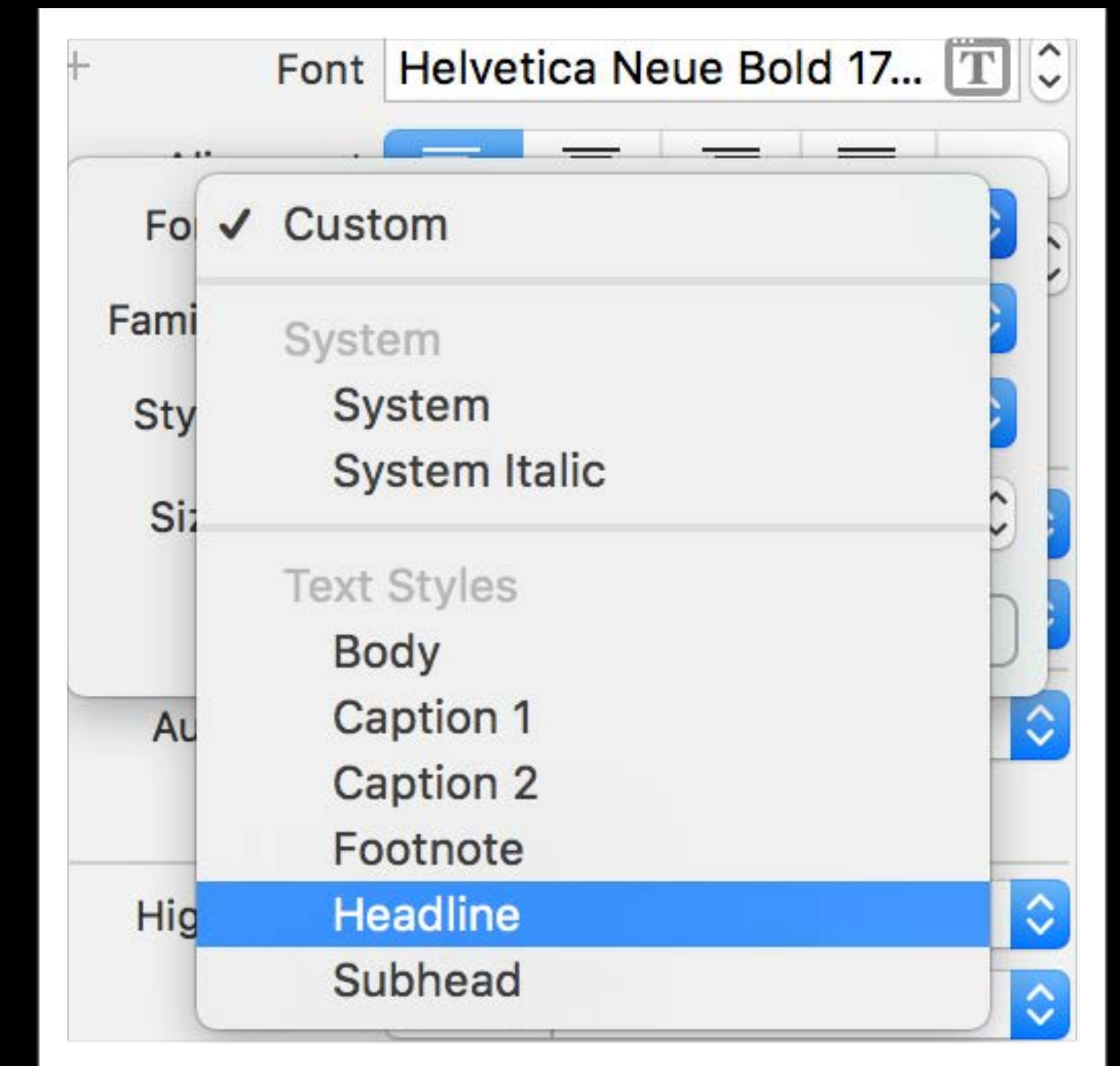
Designables

Takeaways

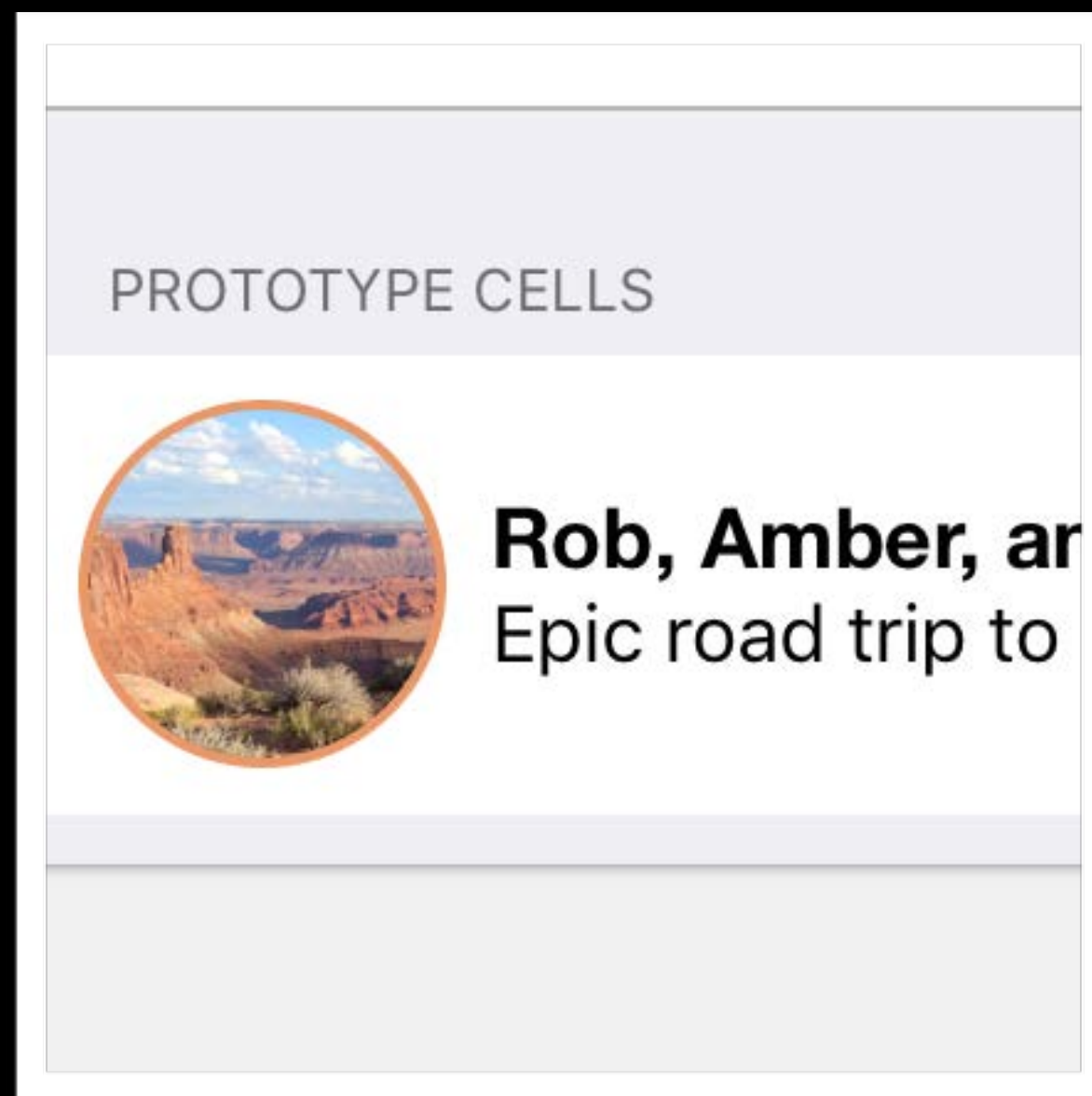
Best practices



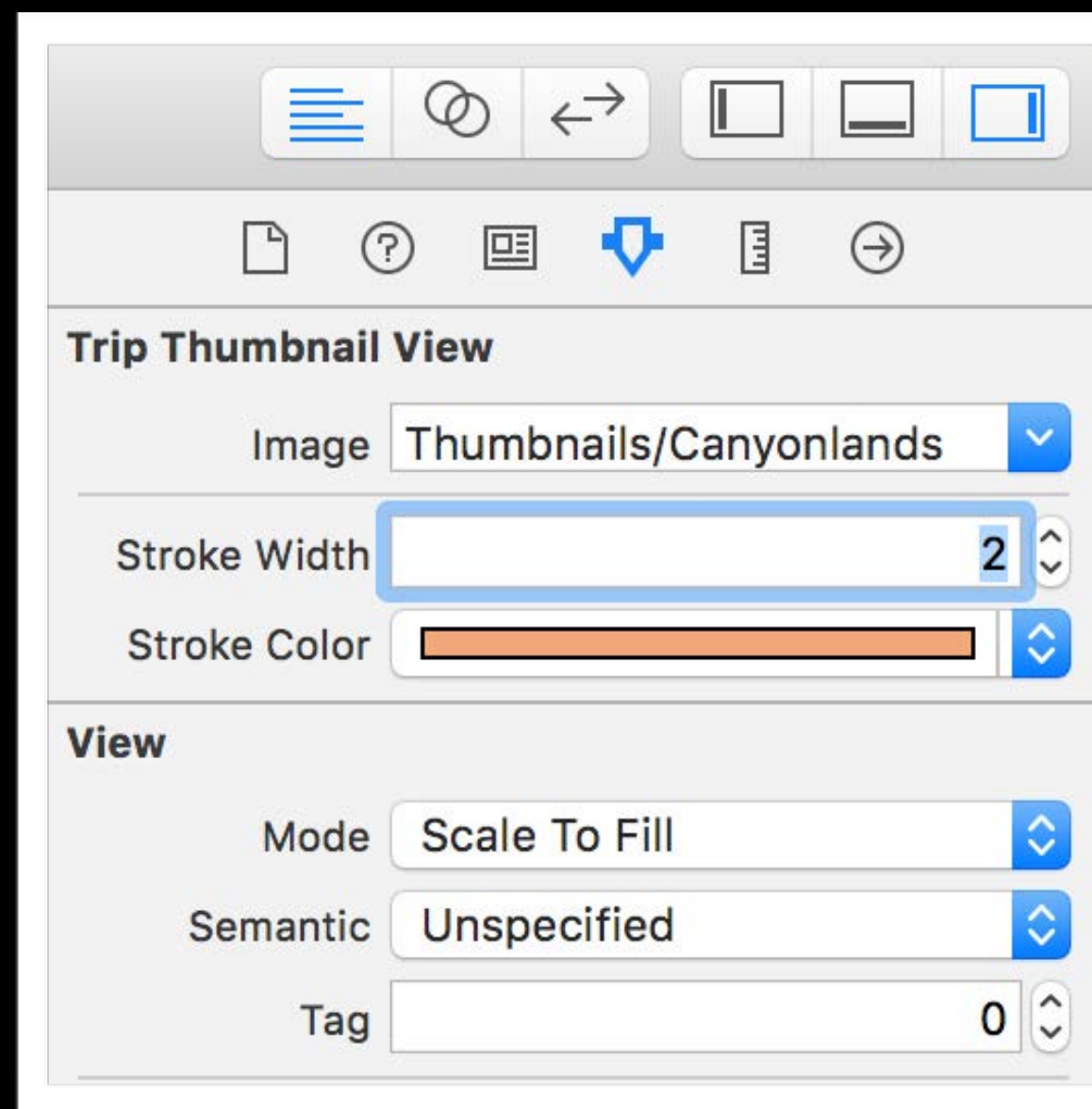
Stack Views



Dynamic Type



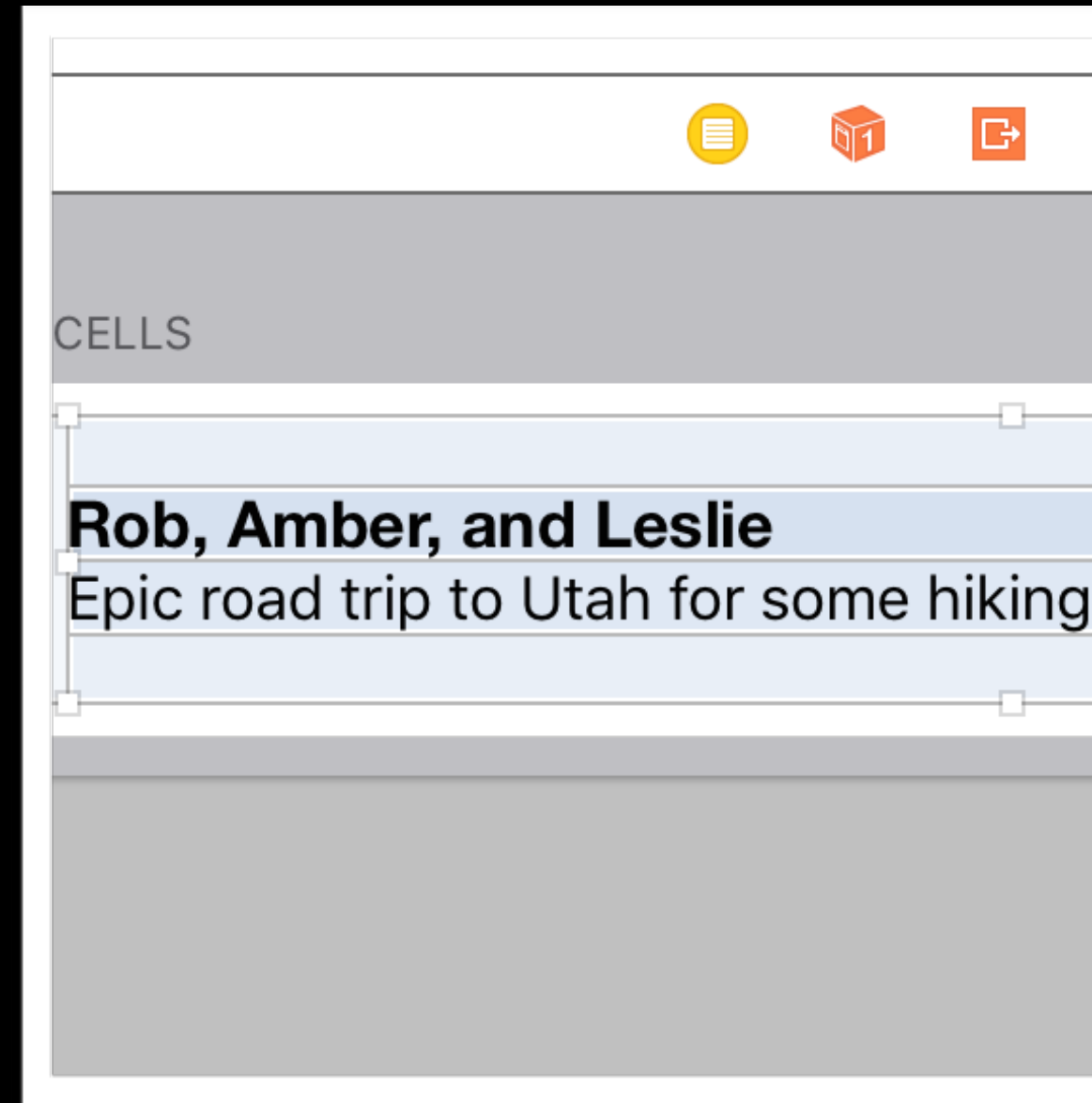
Designables



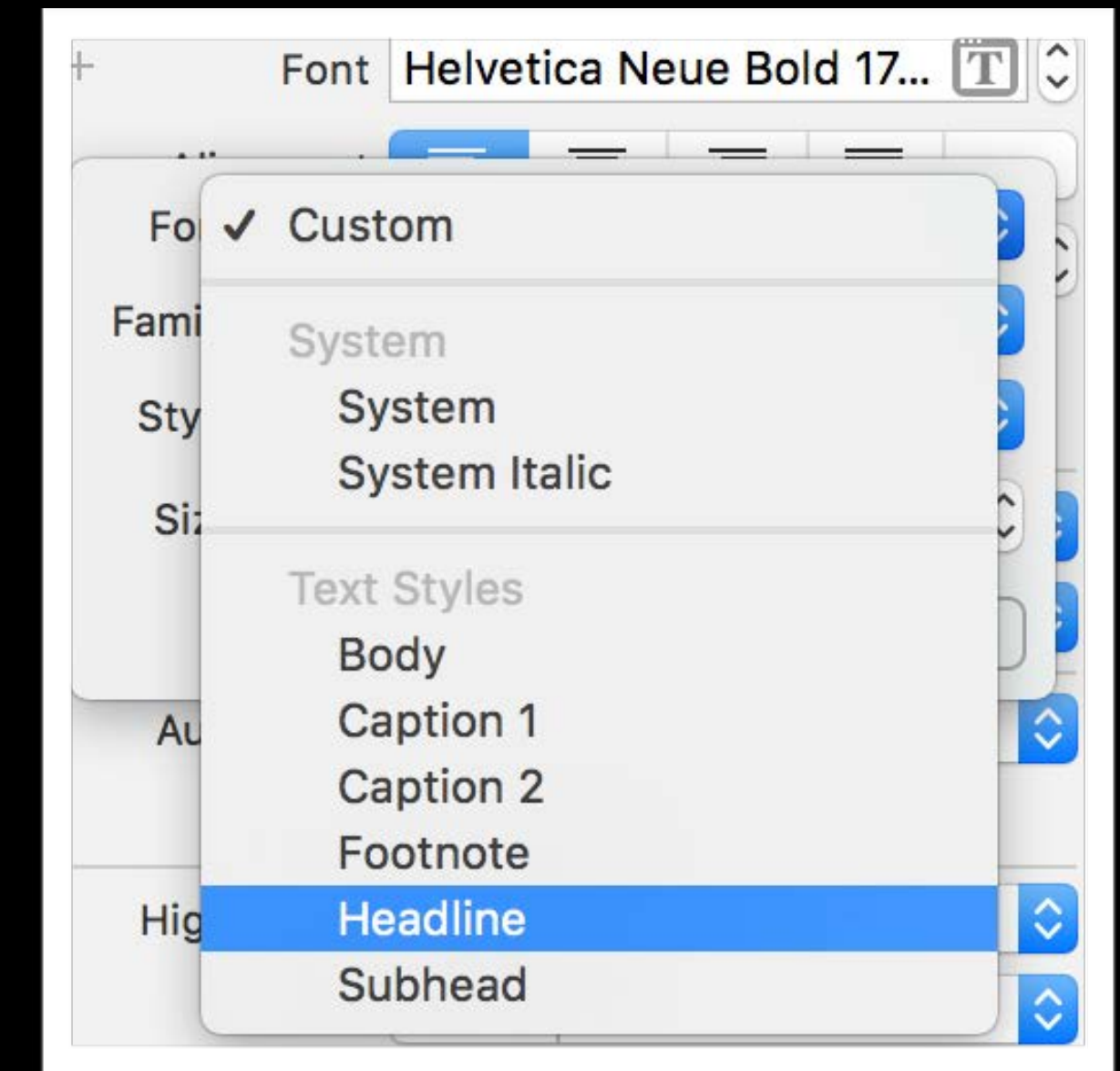
Inspectables

Takeaways

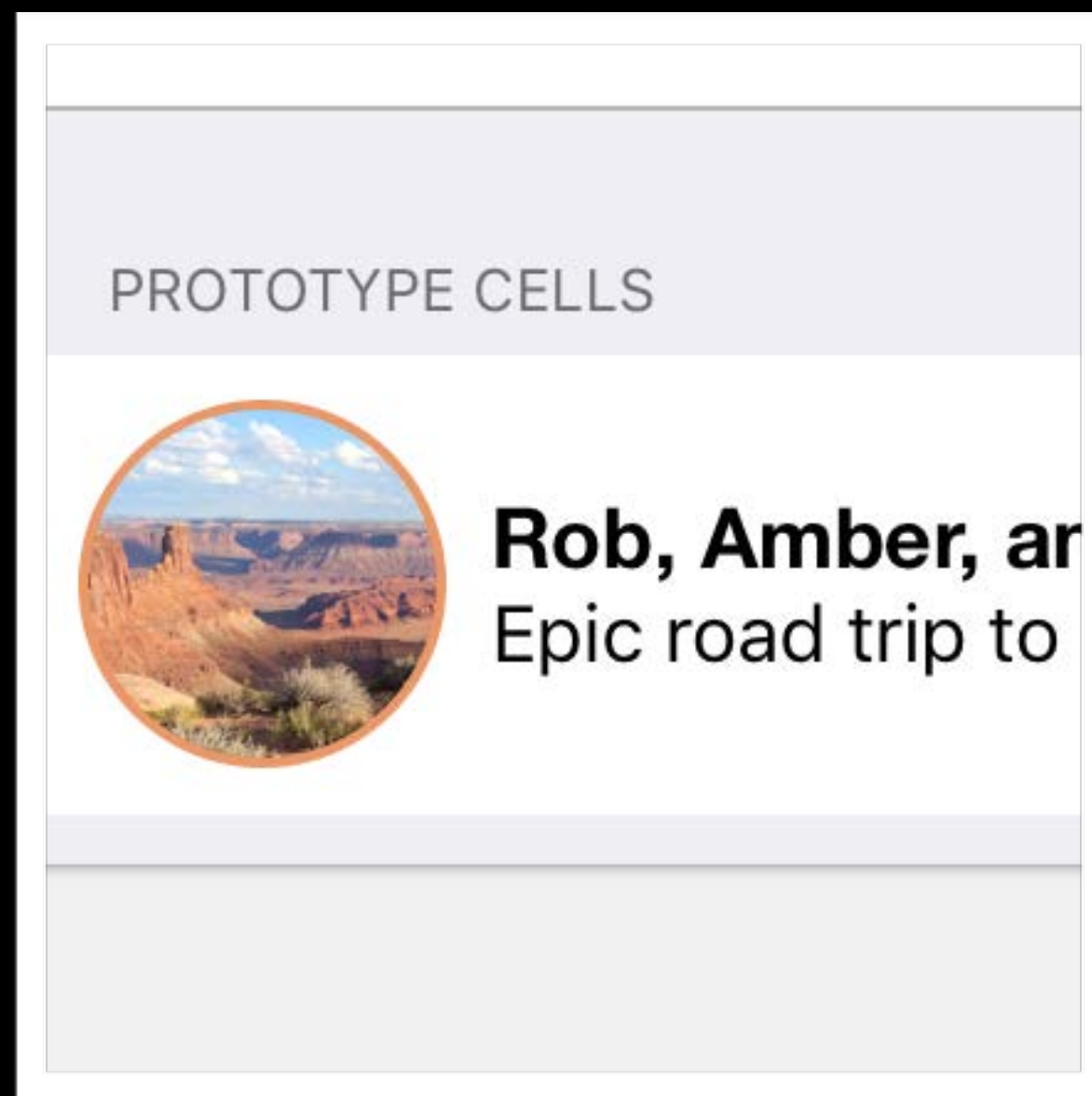
Best practices



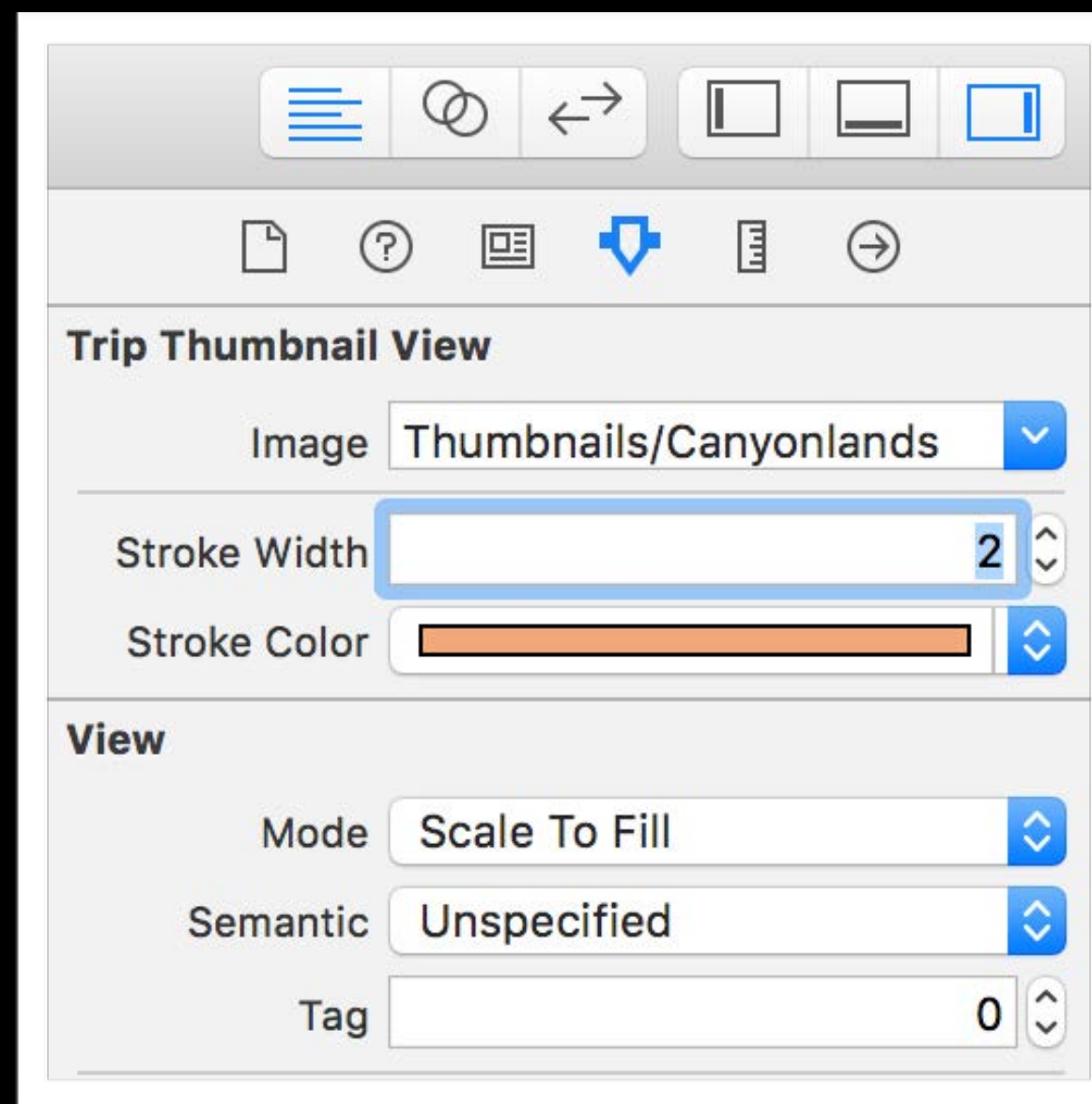
Stack Views



Dynamic Type



Designables



Inspectables



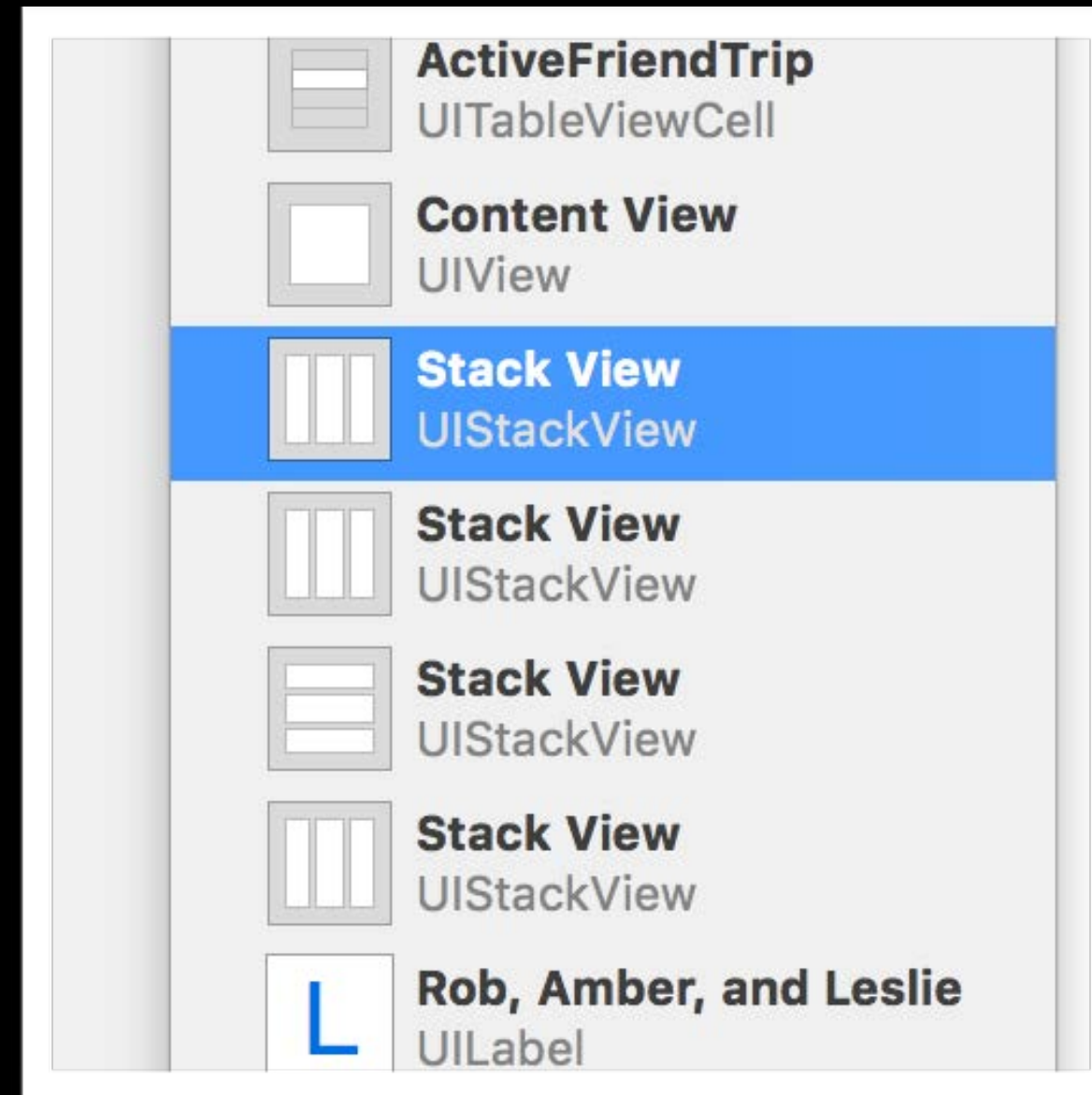
Storyboard References

Takeaways

Tips and tricks

Takeaways

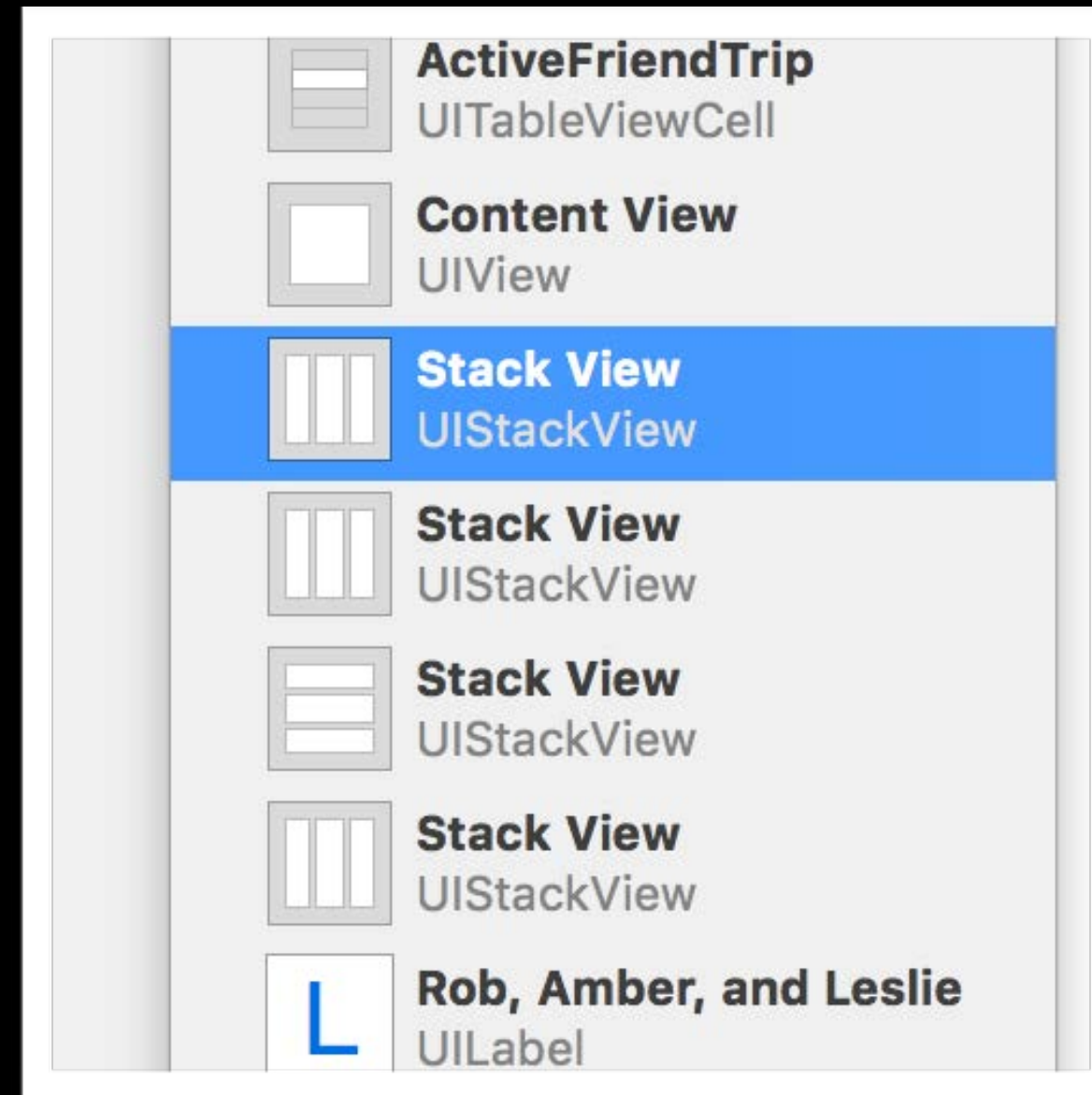
Tips and tricks



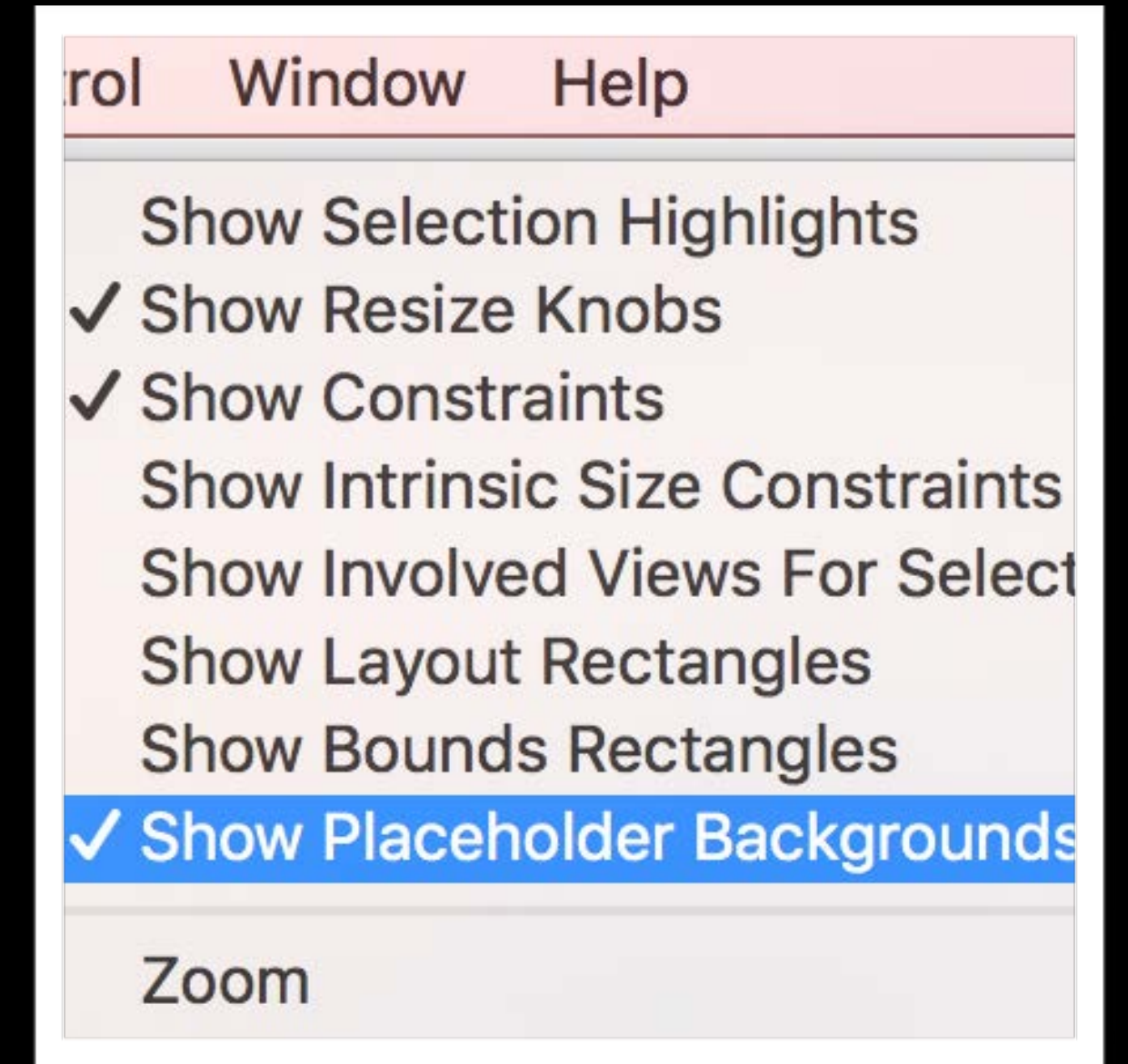
Fast Selection

Takeaways

Tips and tricks



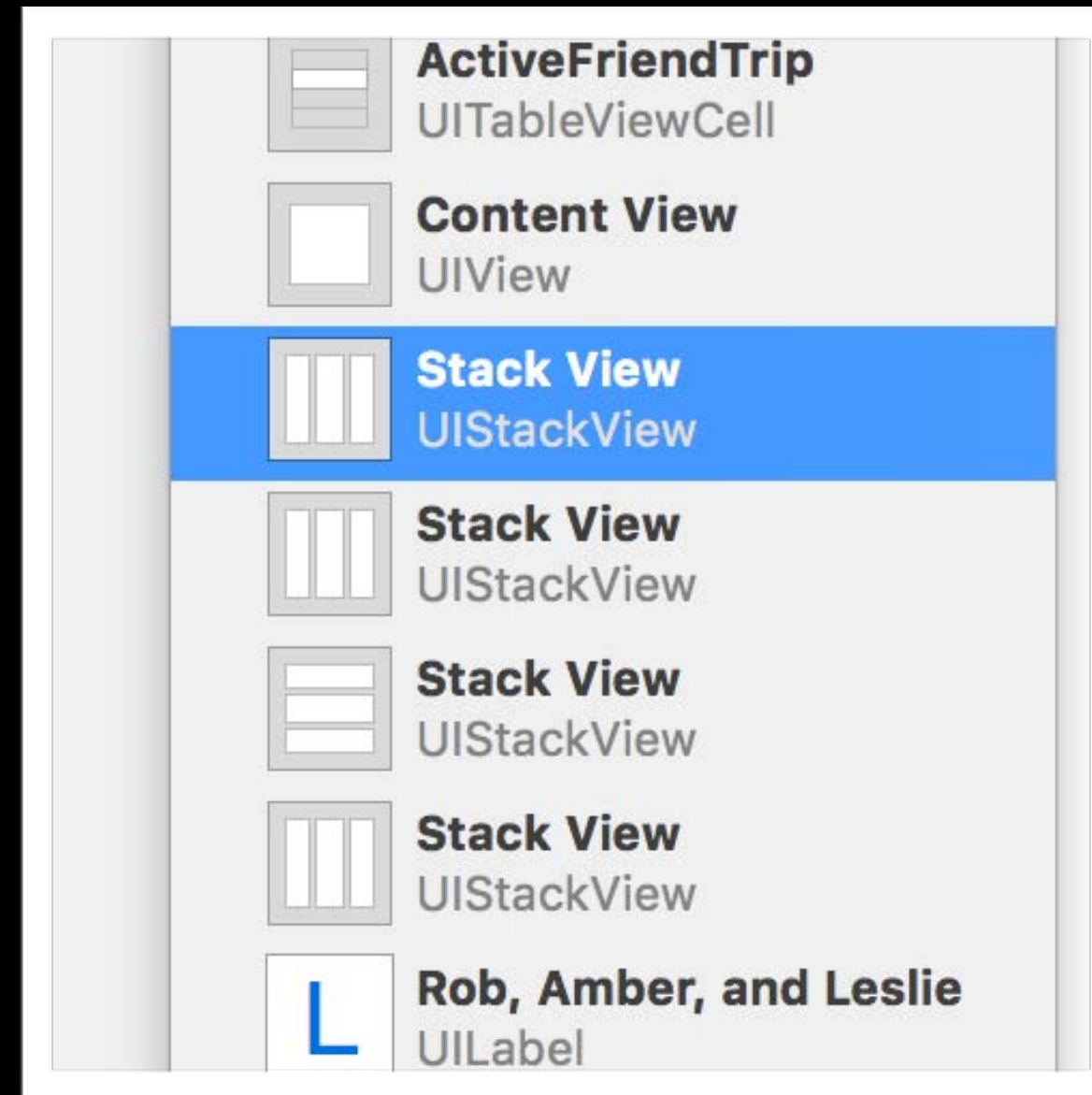
Fast Selection



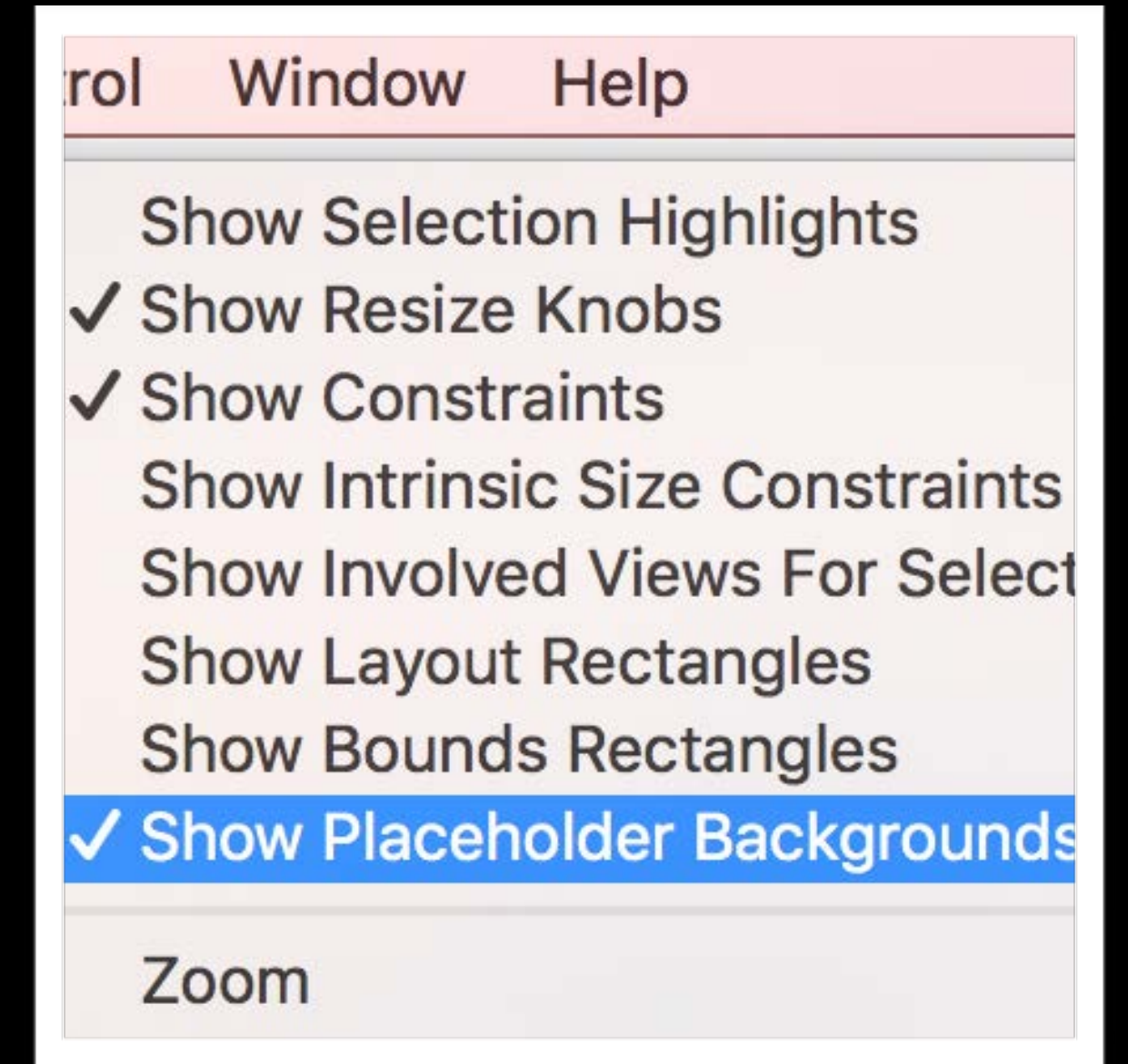
Canvas Customizations

Takeaways

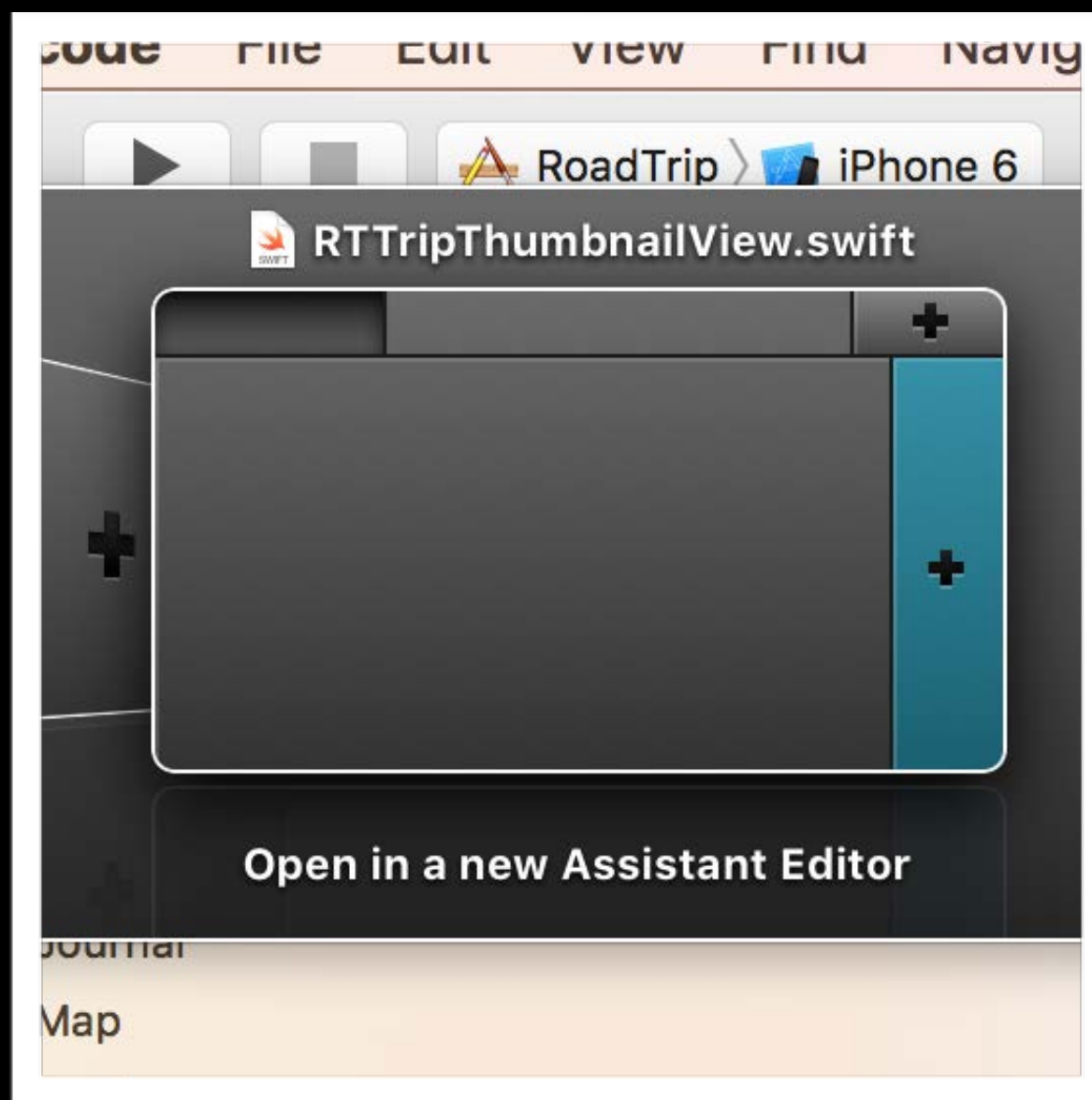
Tips and tricks



Fast Selection



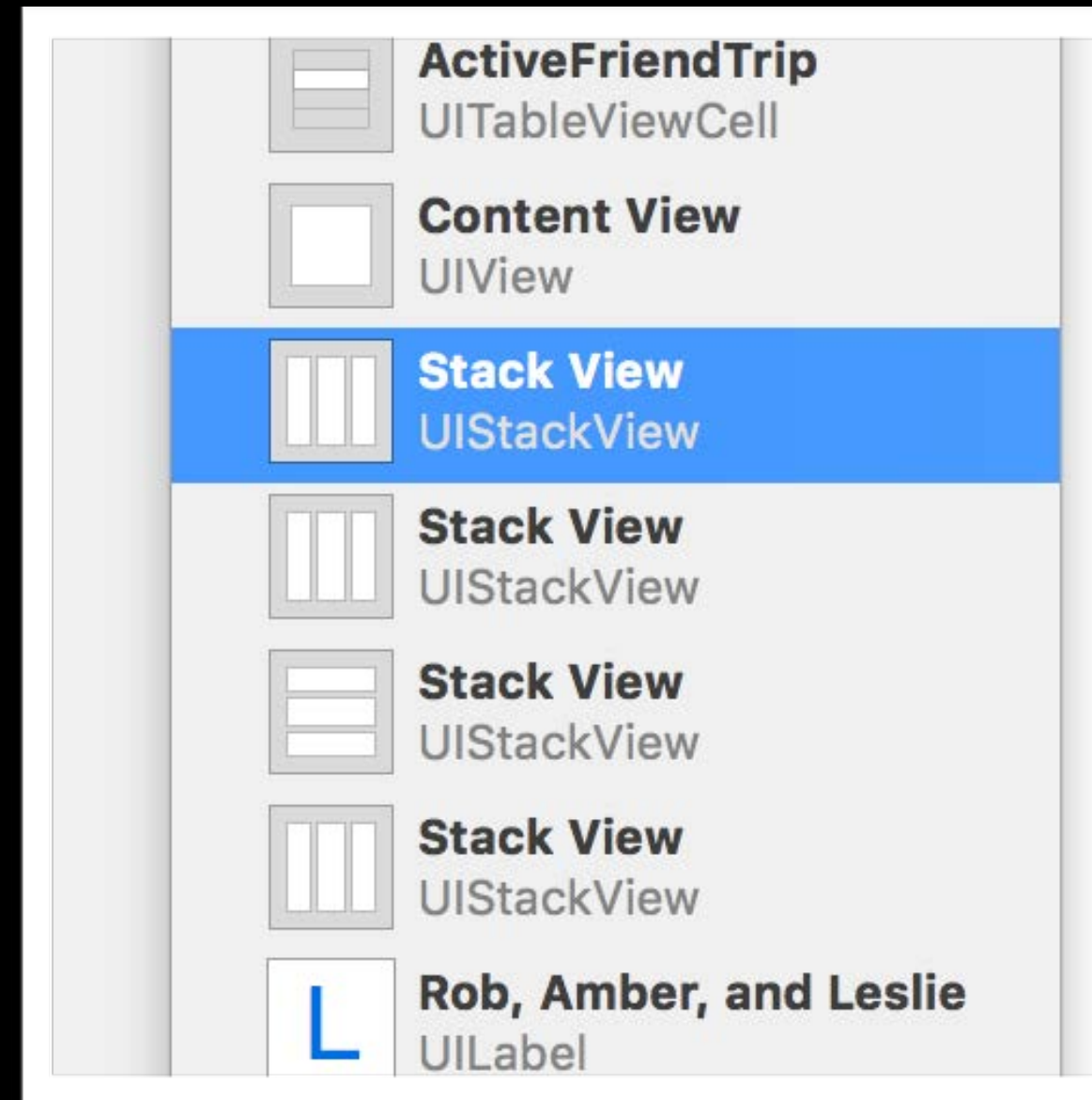
Canvas Customizations



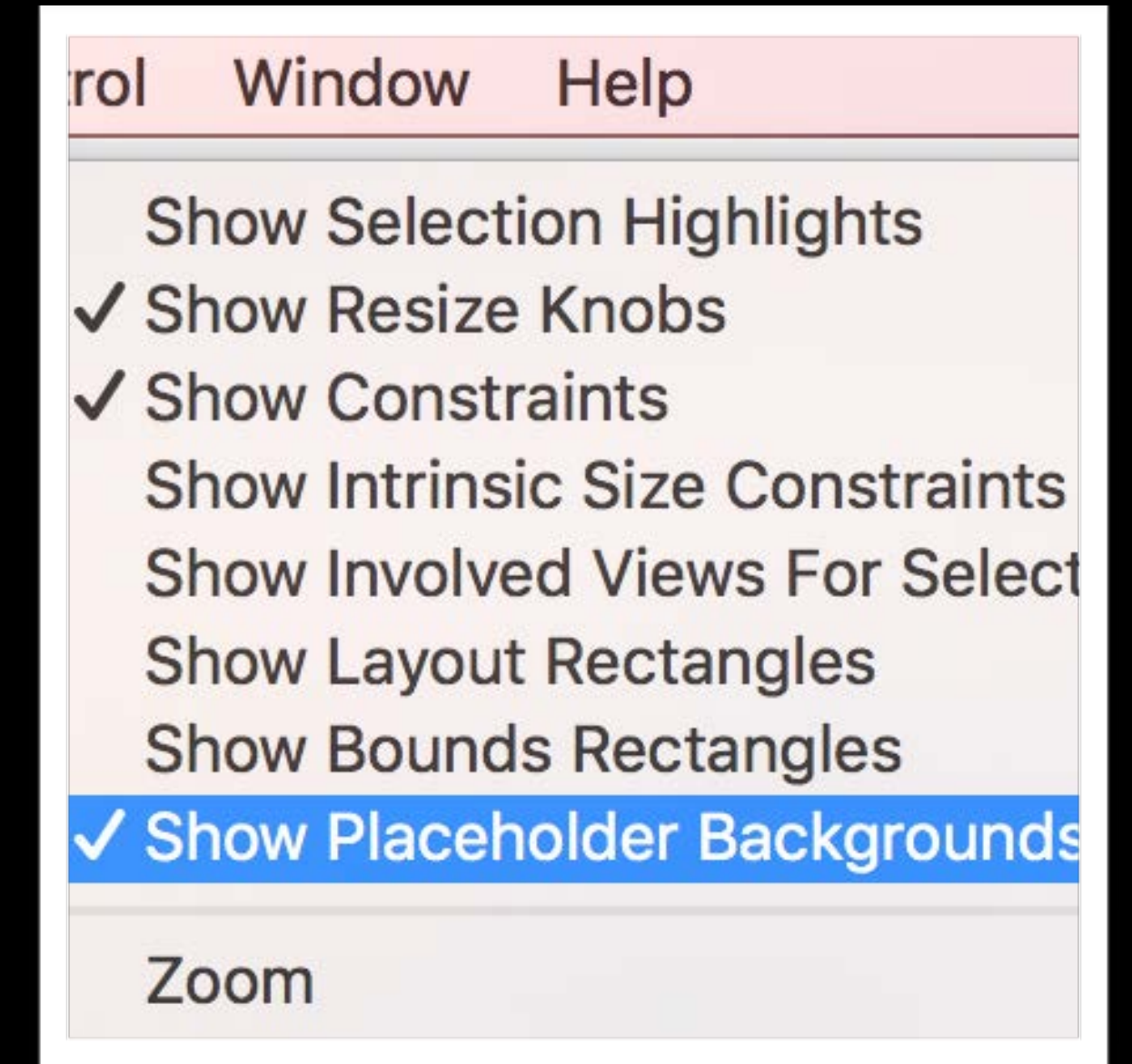
Advanced Navigation

Takeaways

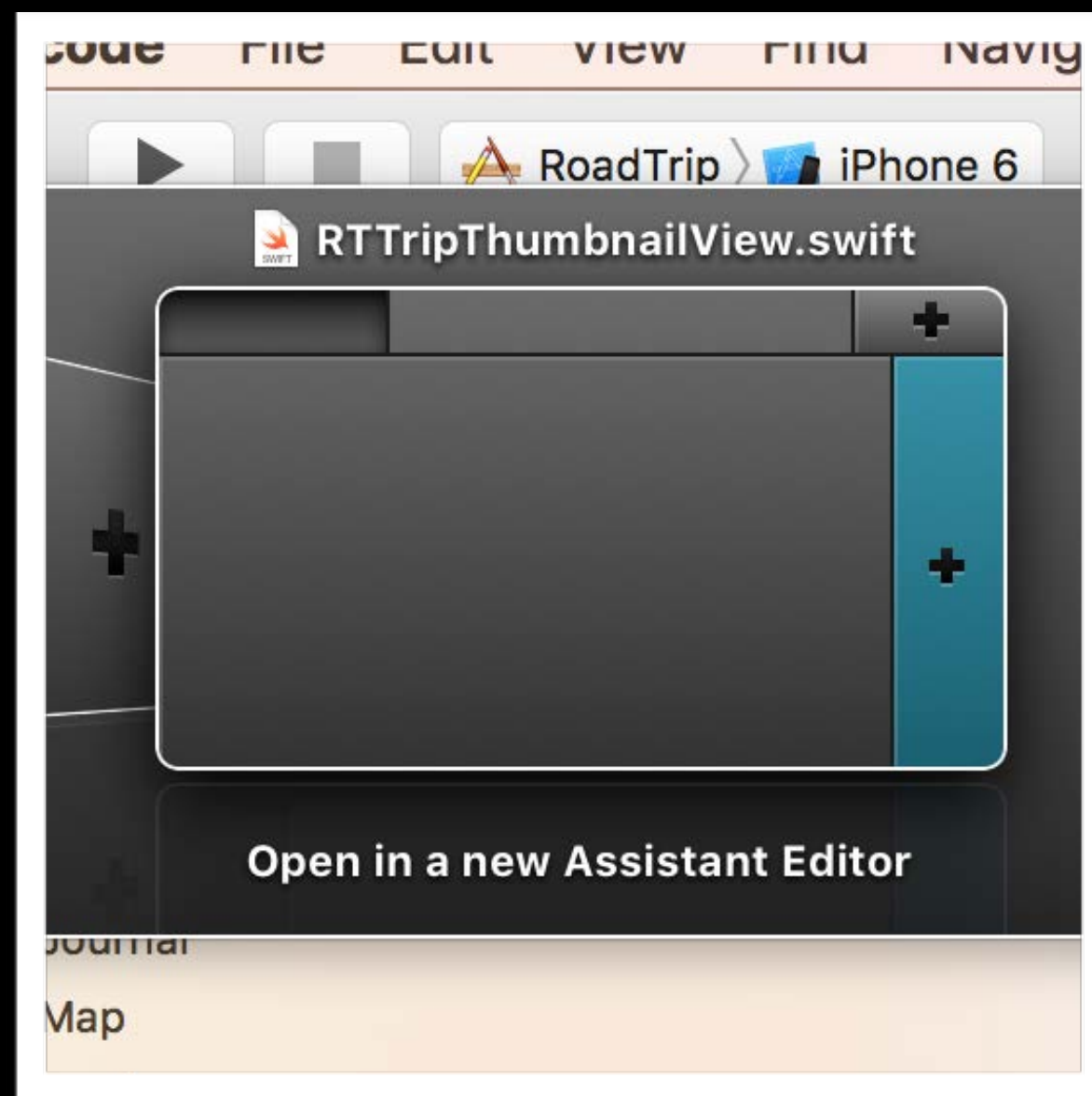
Tips and tricks



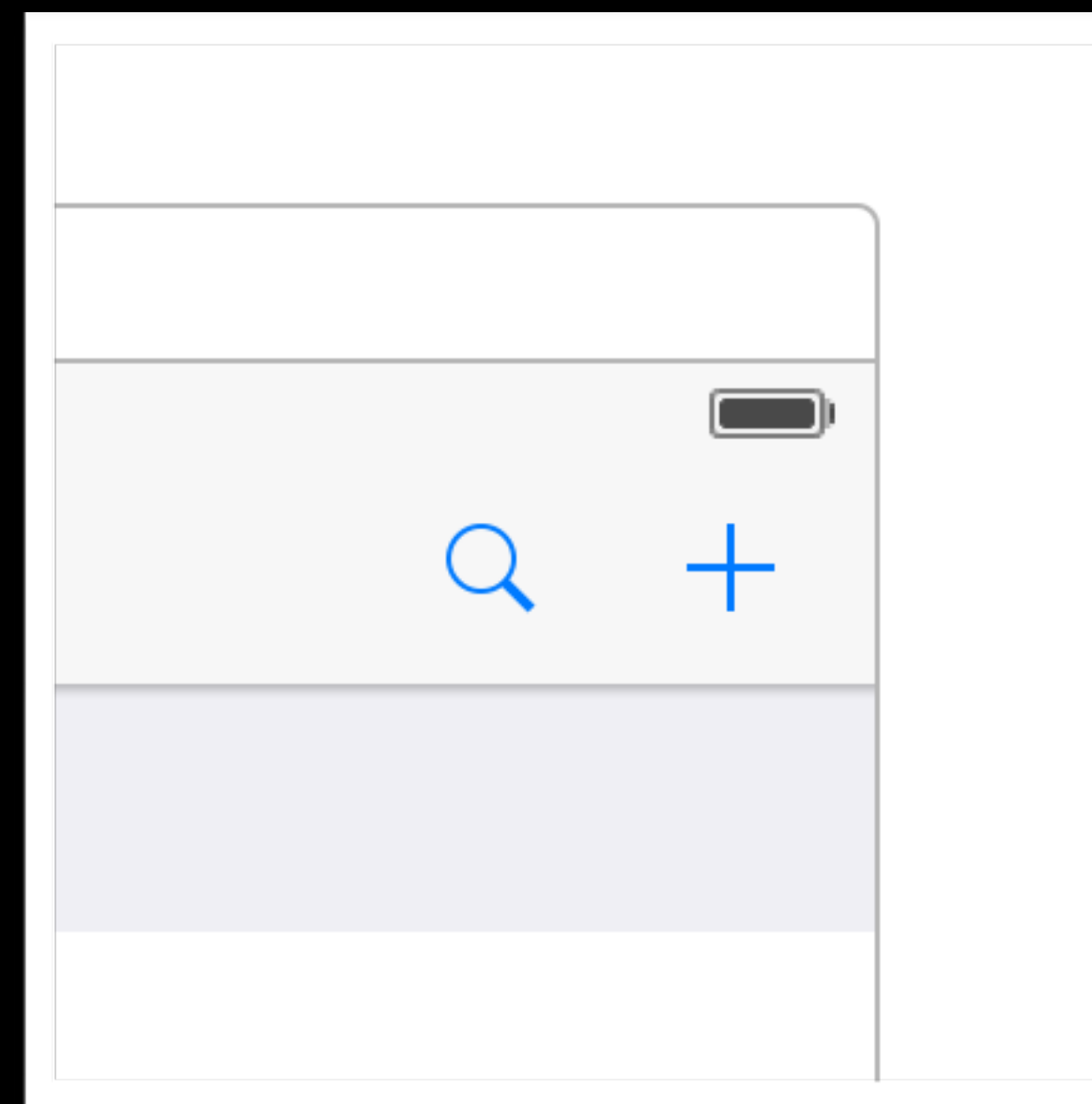
Fast Selection



Canvas Customizations



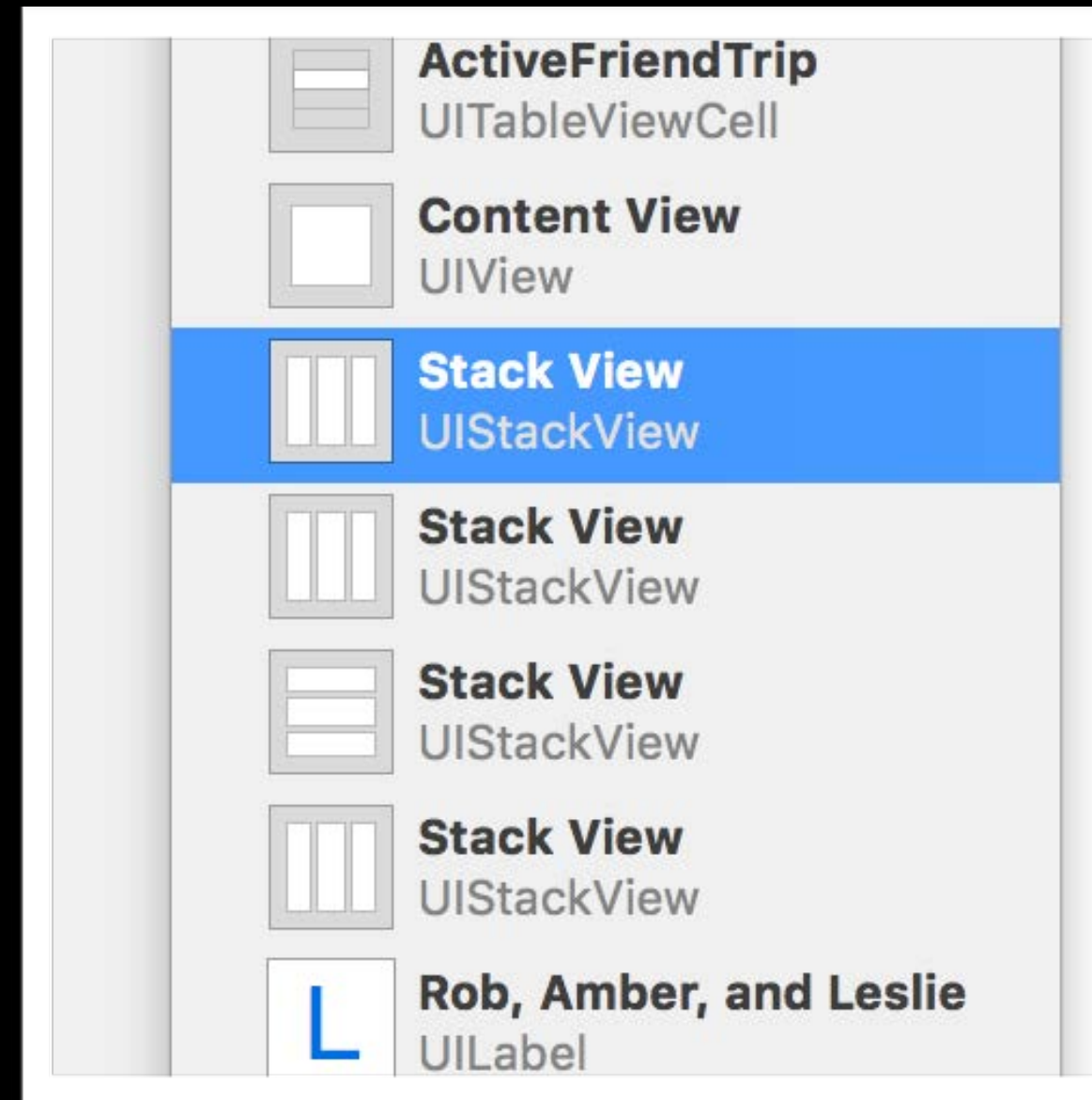
Advanced Navigation



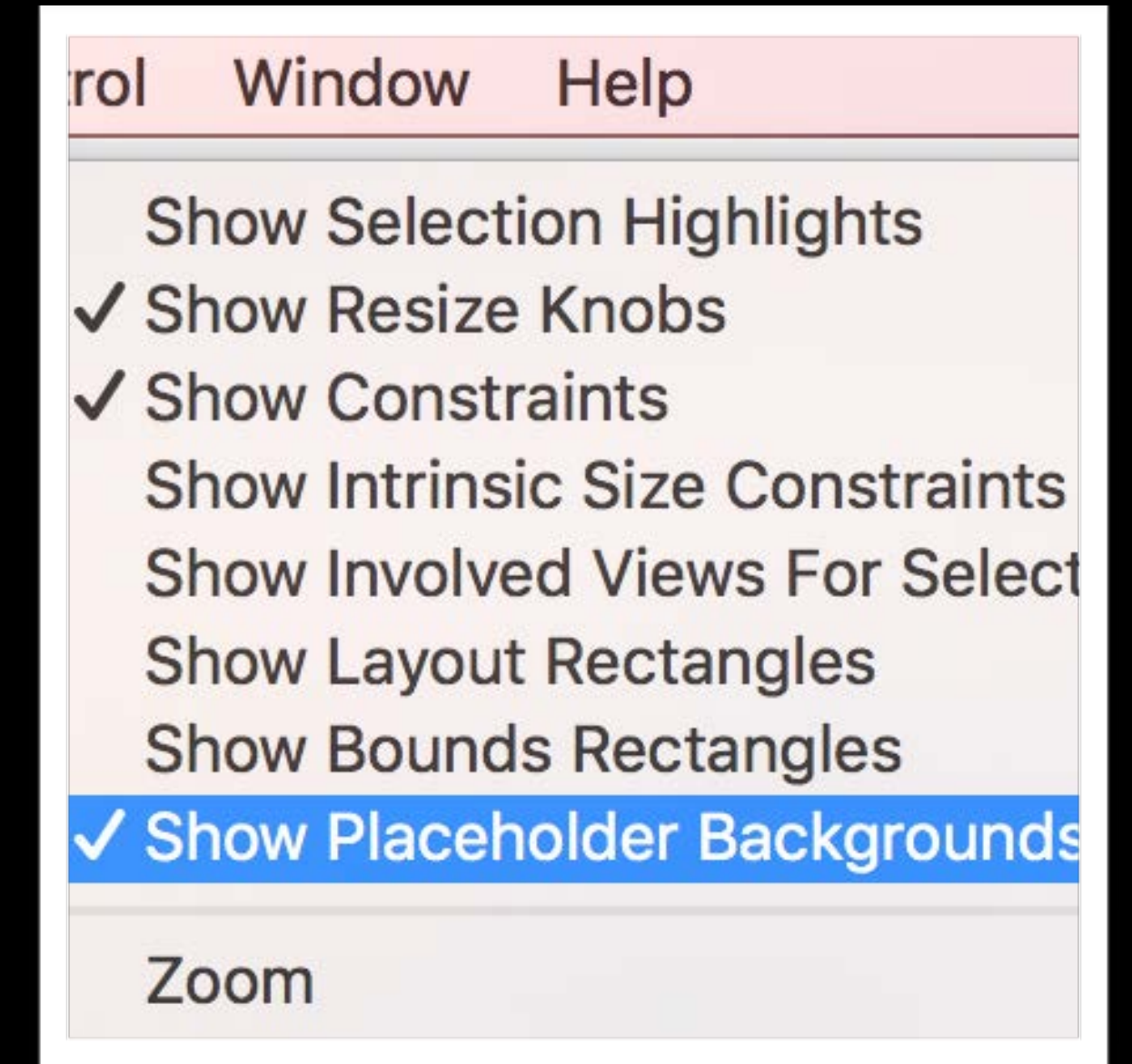
Multiple Bar Items

Takeaways

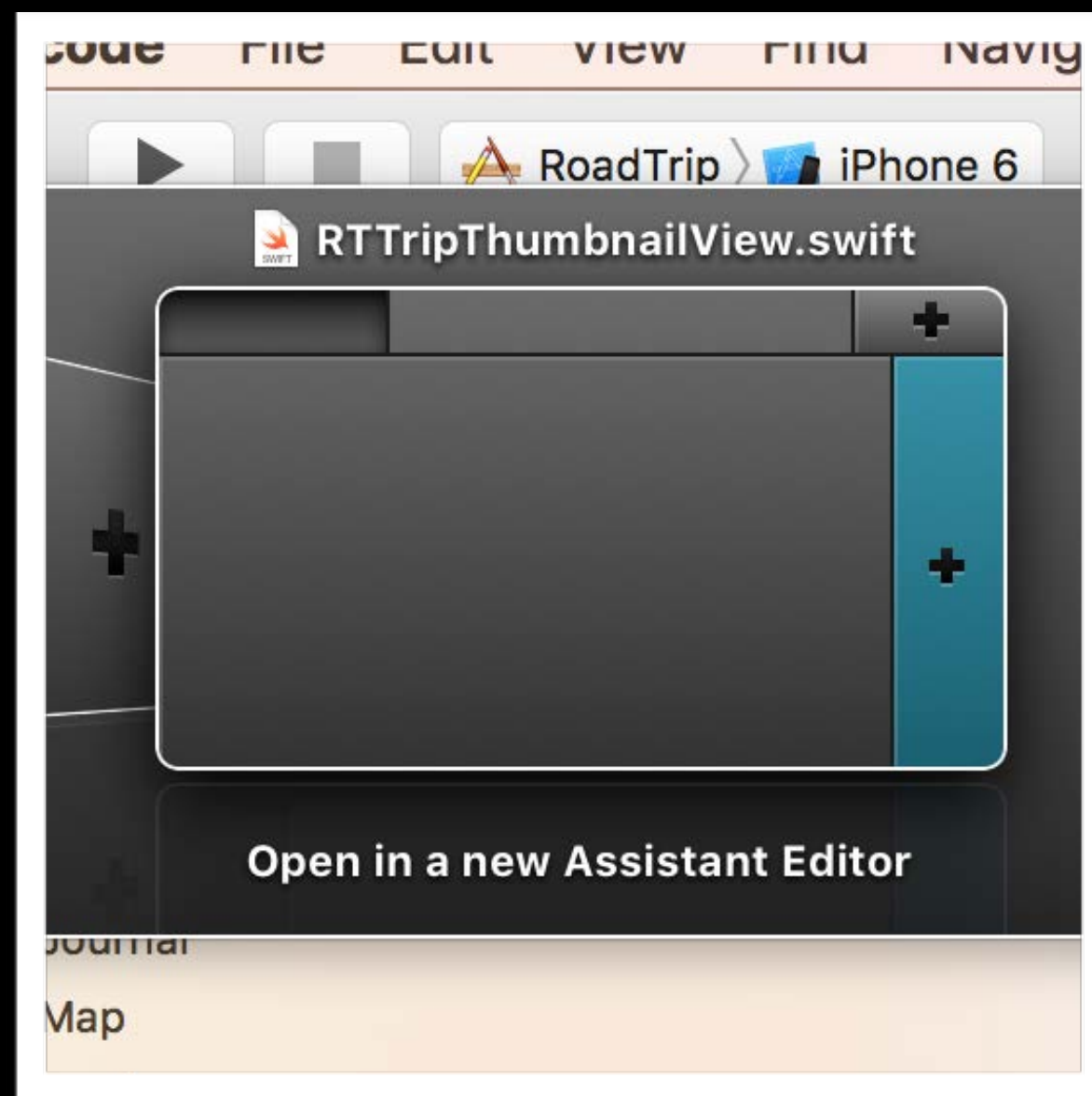
Tips and tricks



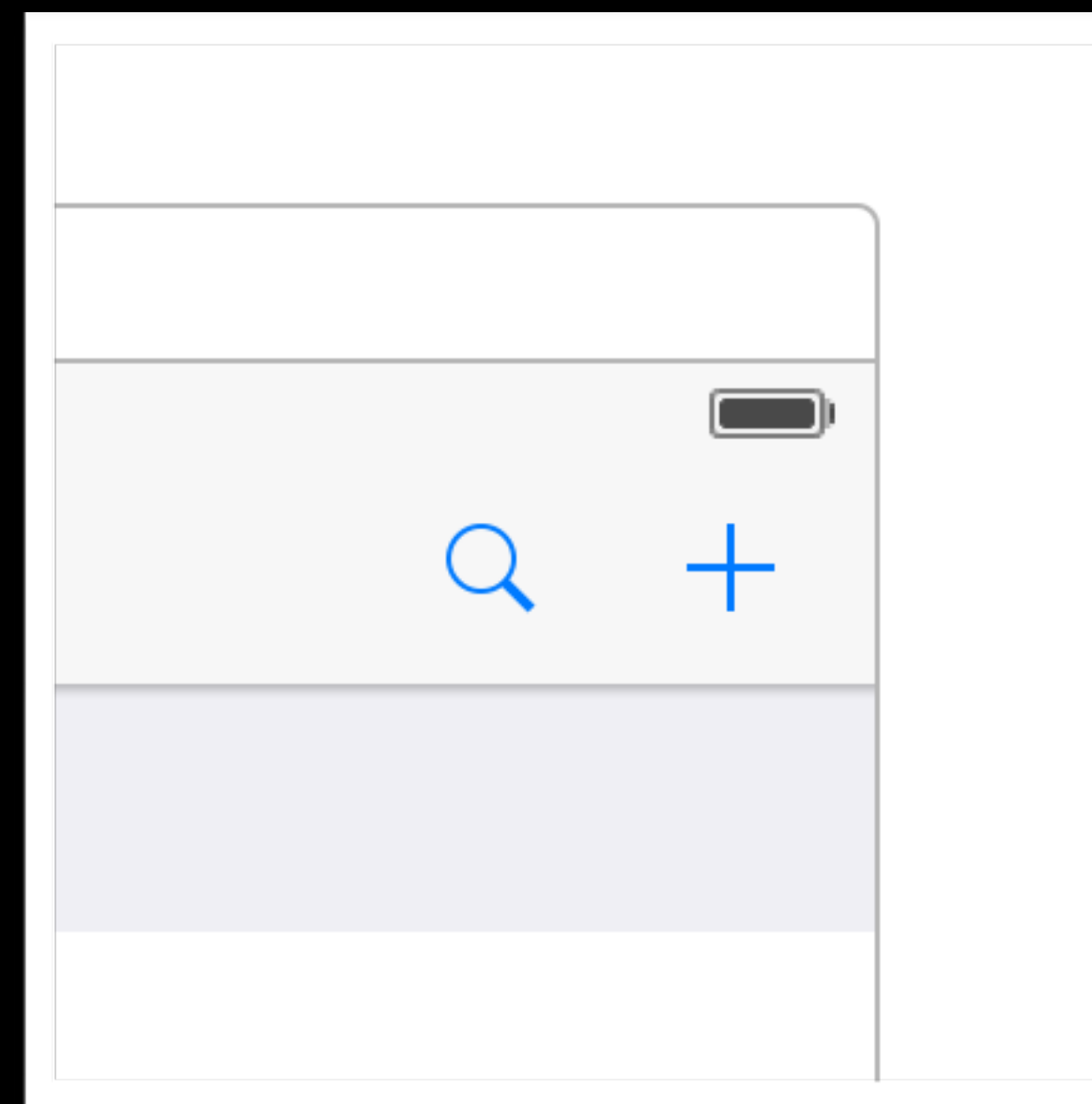
Fast Selection



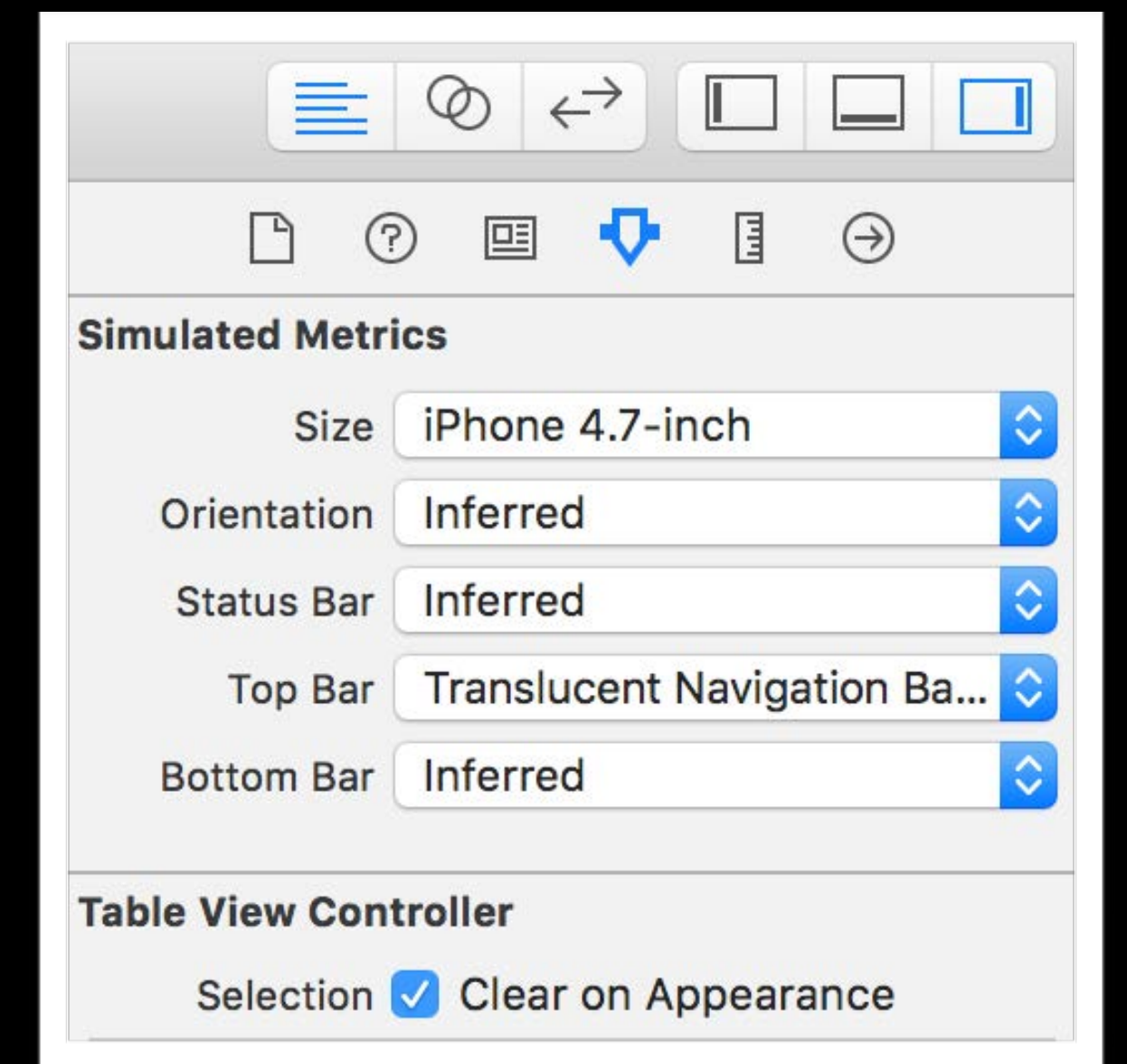
Canvas Customizations



Advanced Navigation



Multiple Bar Items



Simulated Metrics



Design Time



Build Time



Run Time



Design Time



Build Time



Run Time



Design Time



Build Time



Run Time



Design Time
XML Documents



Build Time



Run Time



Design Time
XML Documents

Build Time
ibtool

Run Time



Design Time
XML Documents



Build Time
ibtool

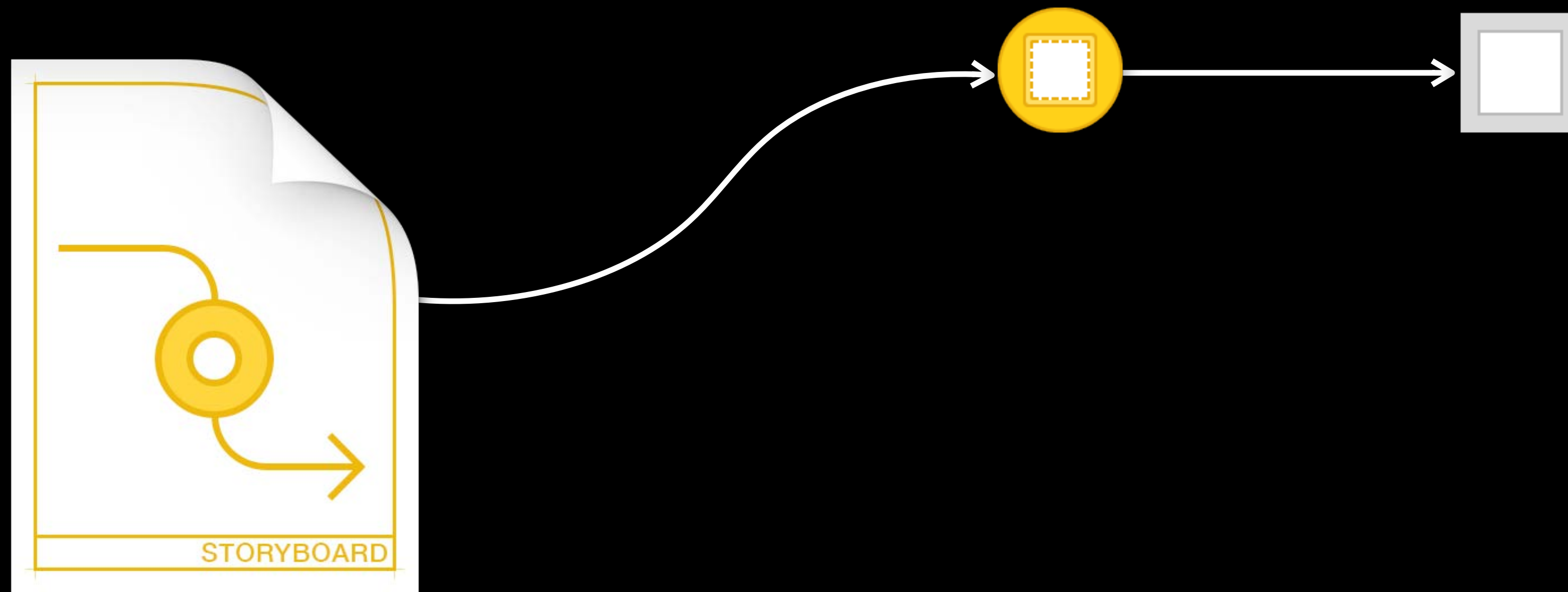


Run Time
Nib Files

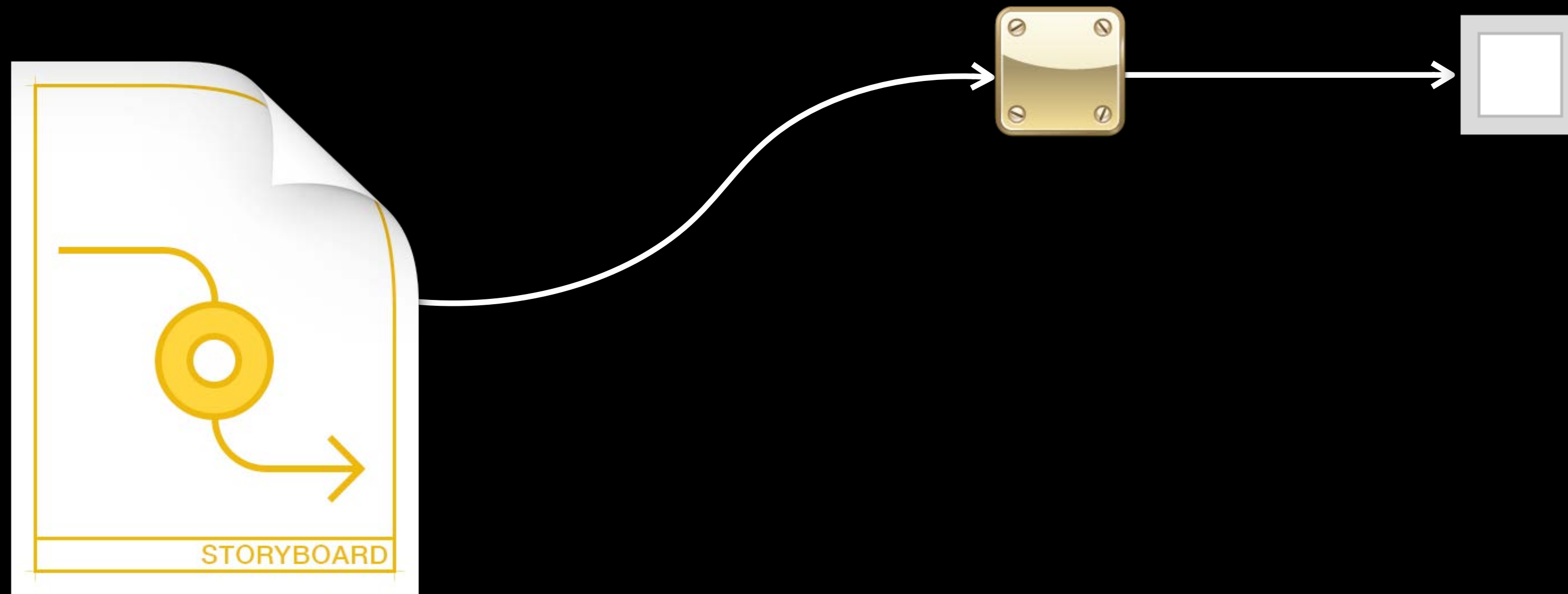
Compiling Storyboards



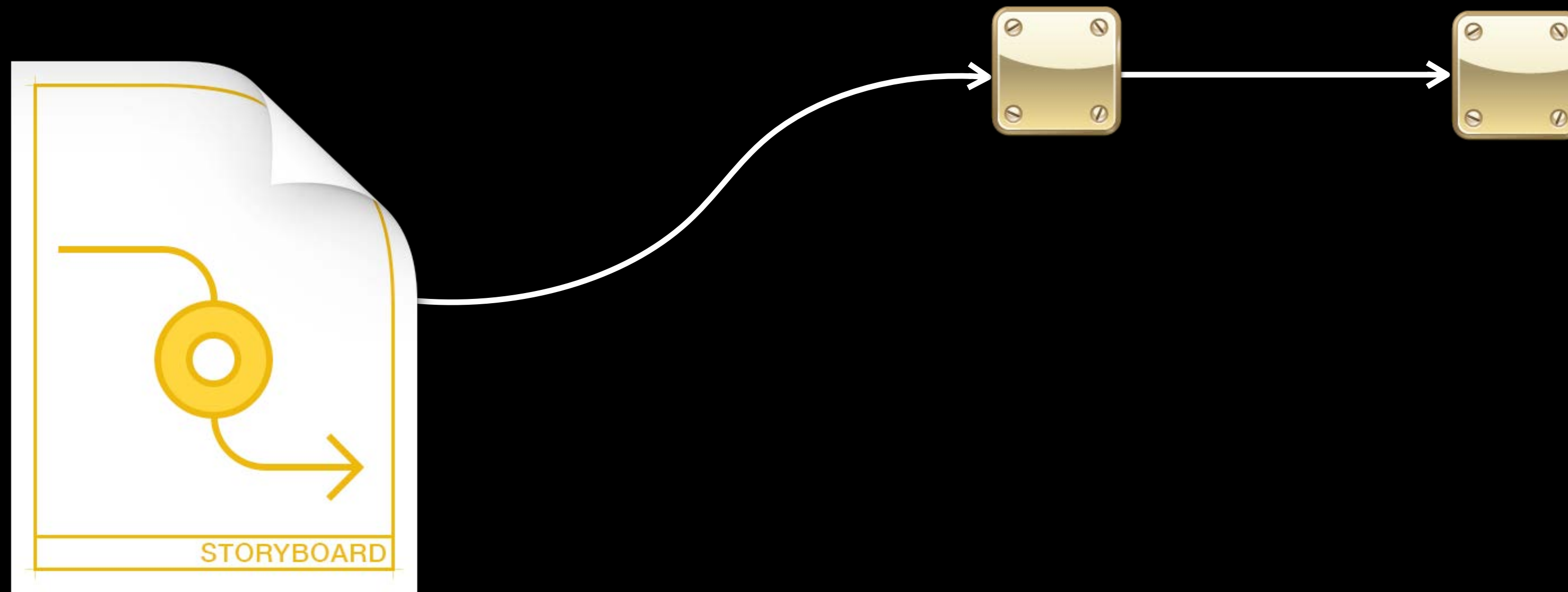
Compiling Storyboards



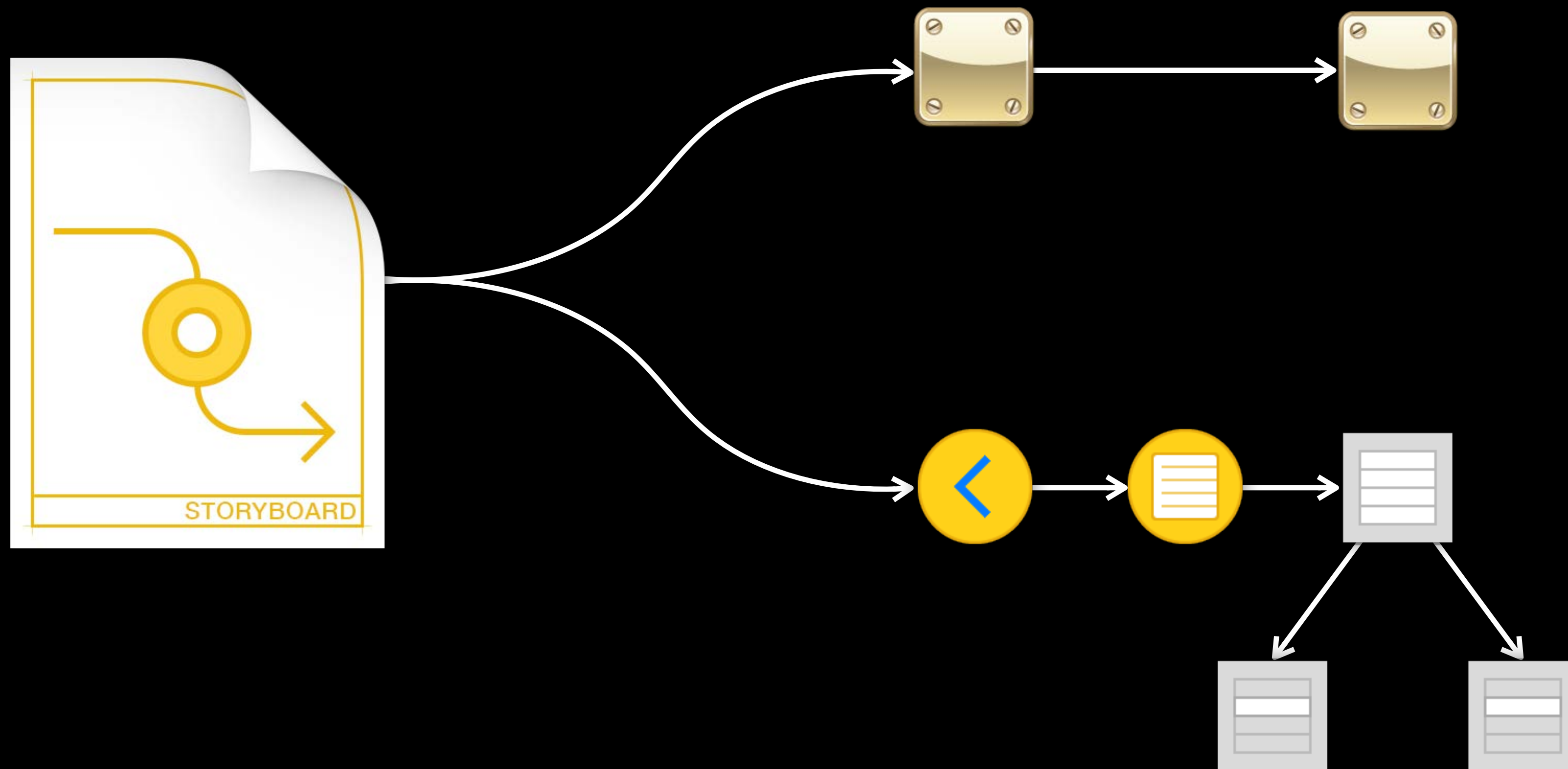
Compiling Storyboards



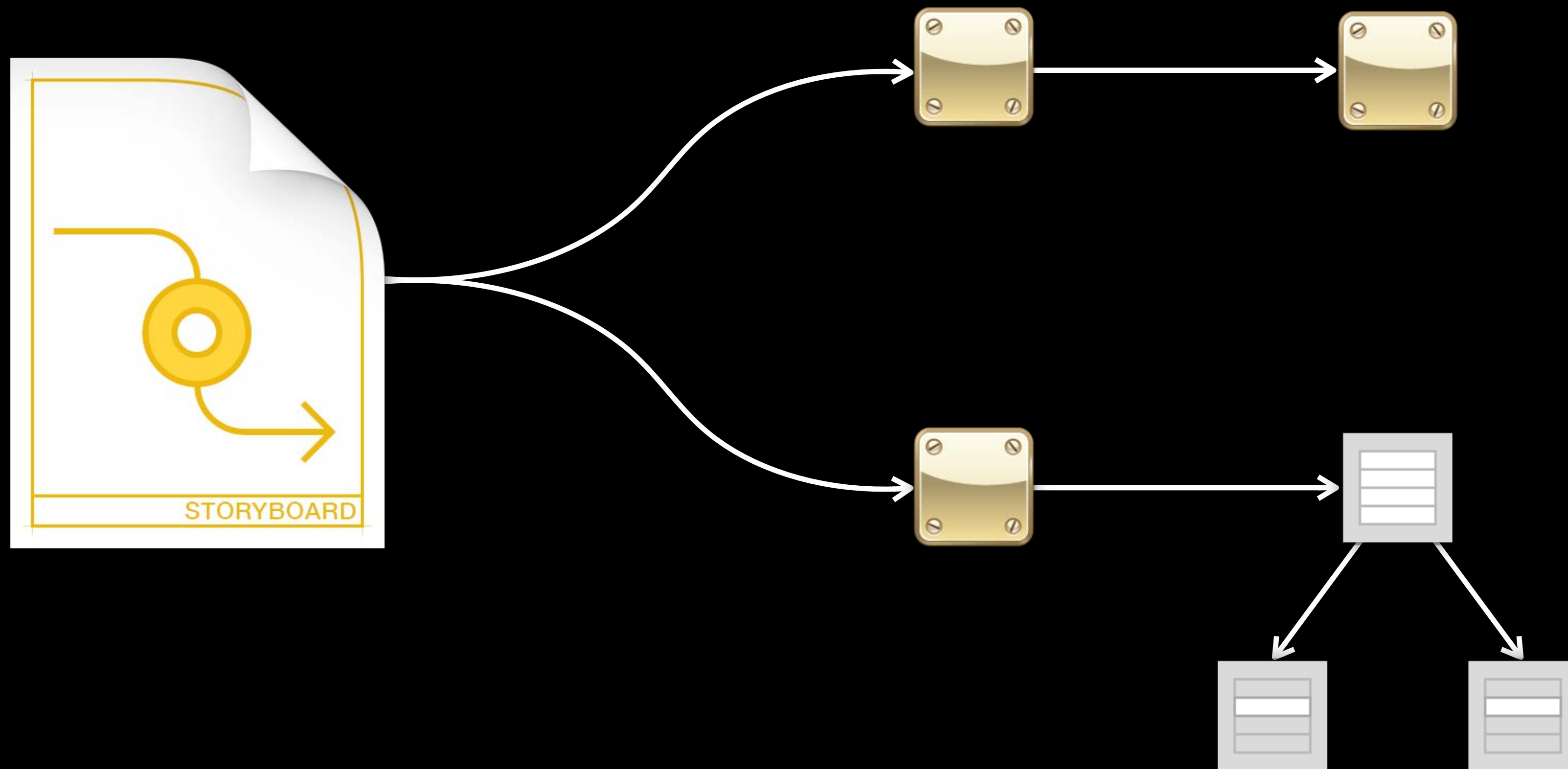
Compiling Storyboards



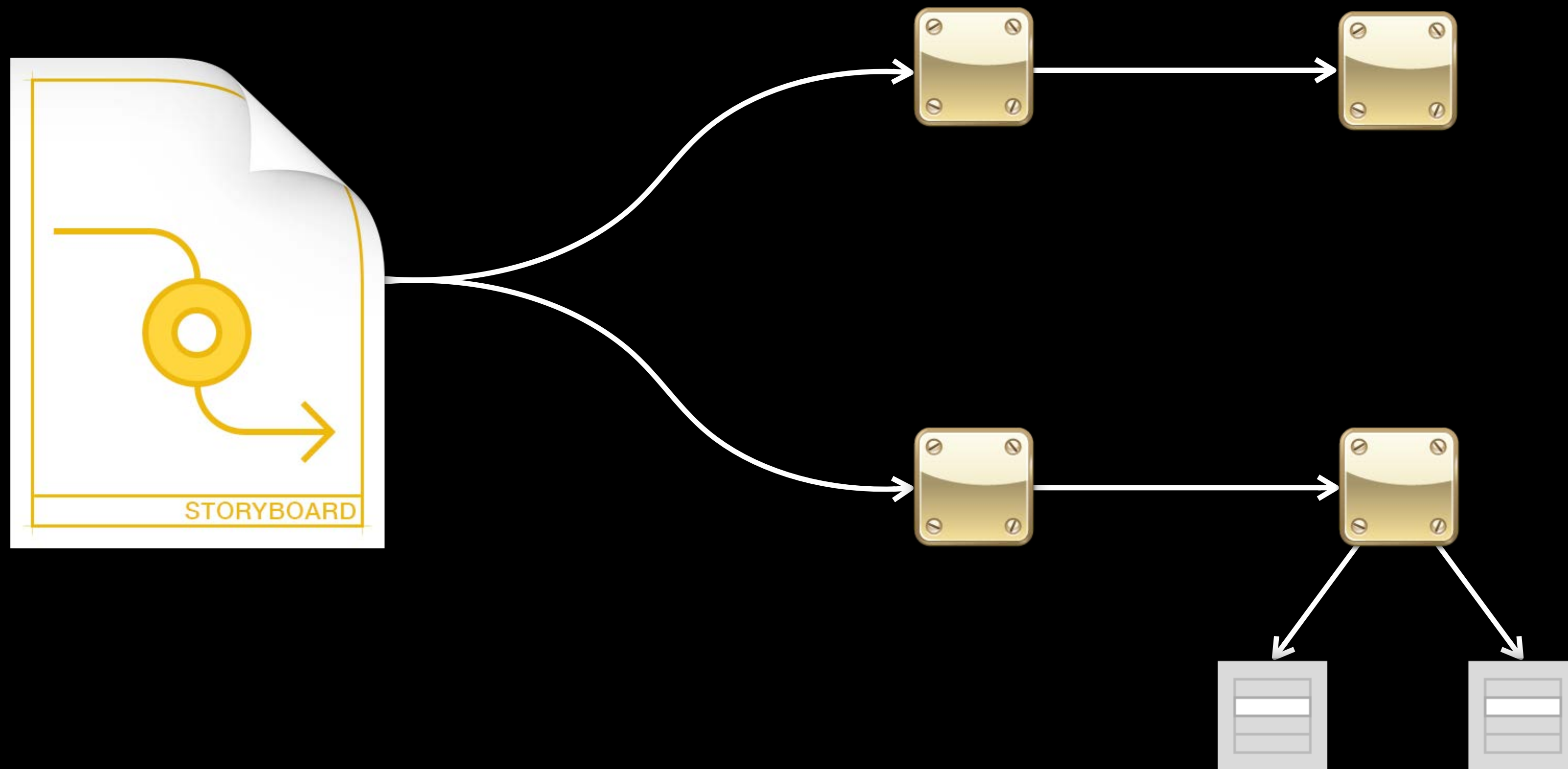
Compiling Storyboards



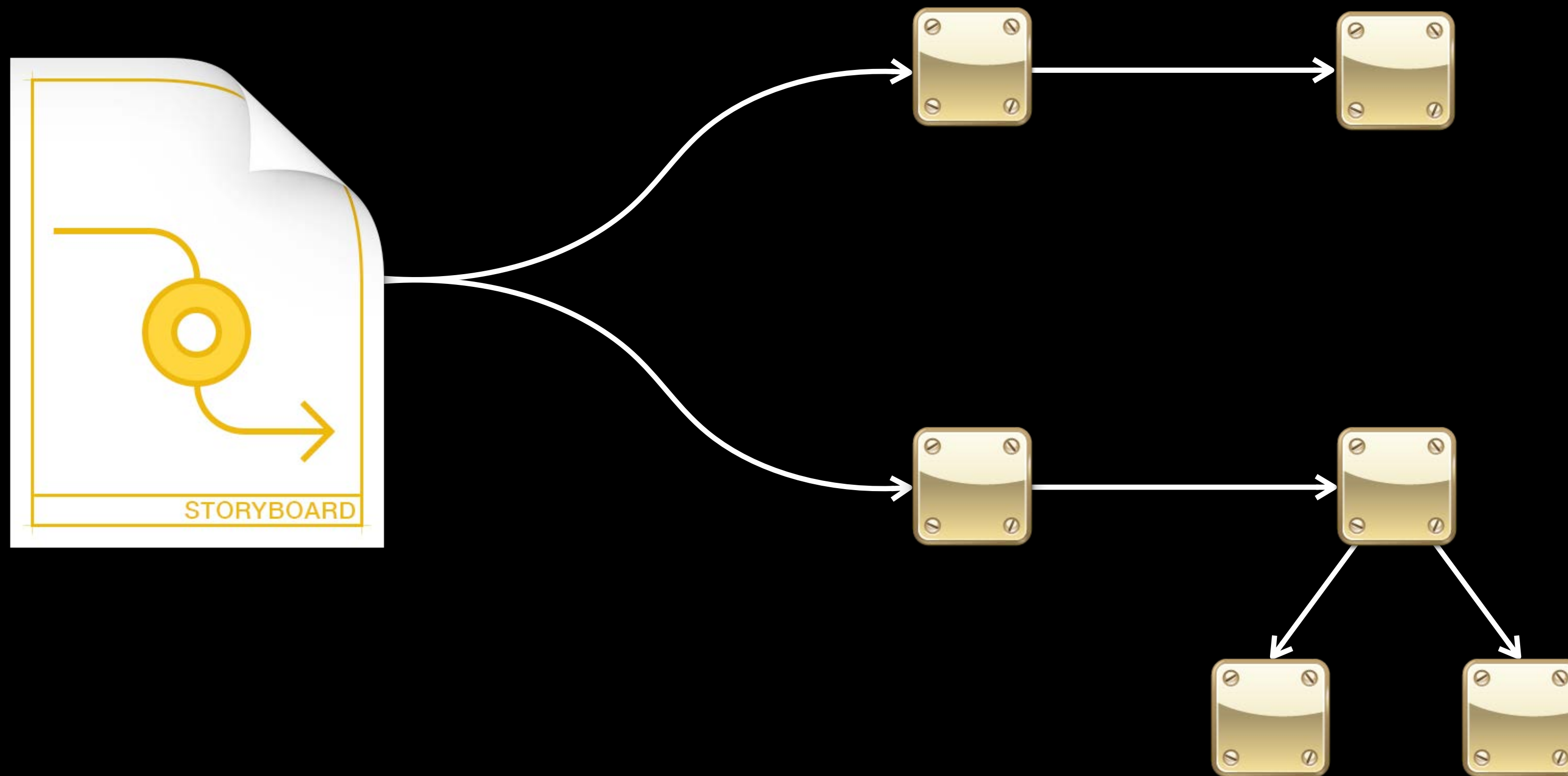
Compiling Storyboards



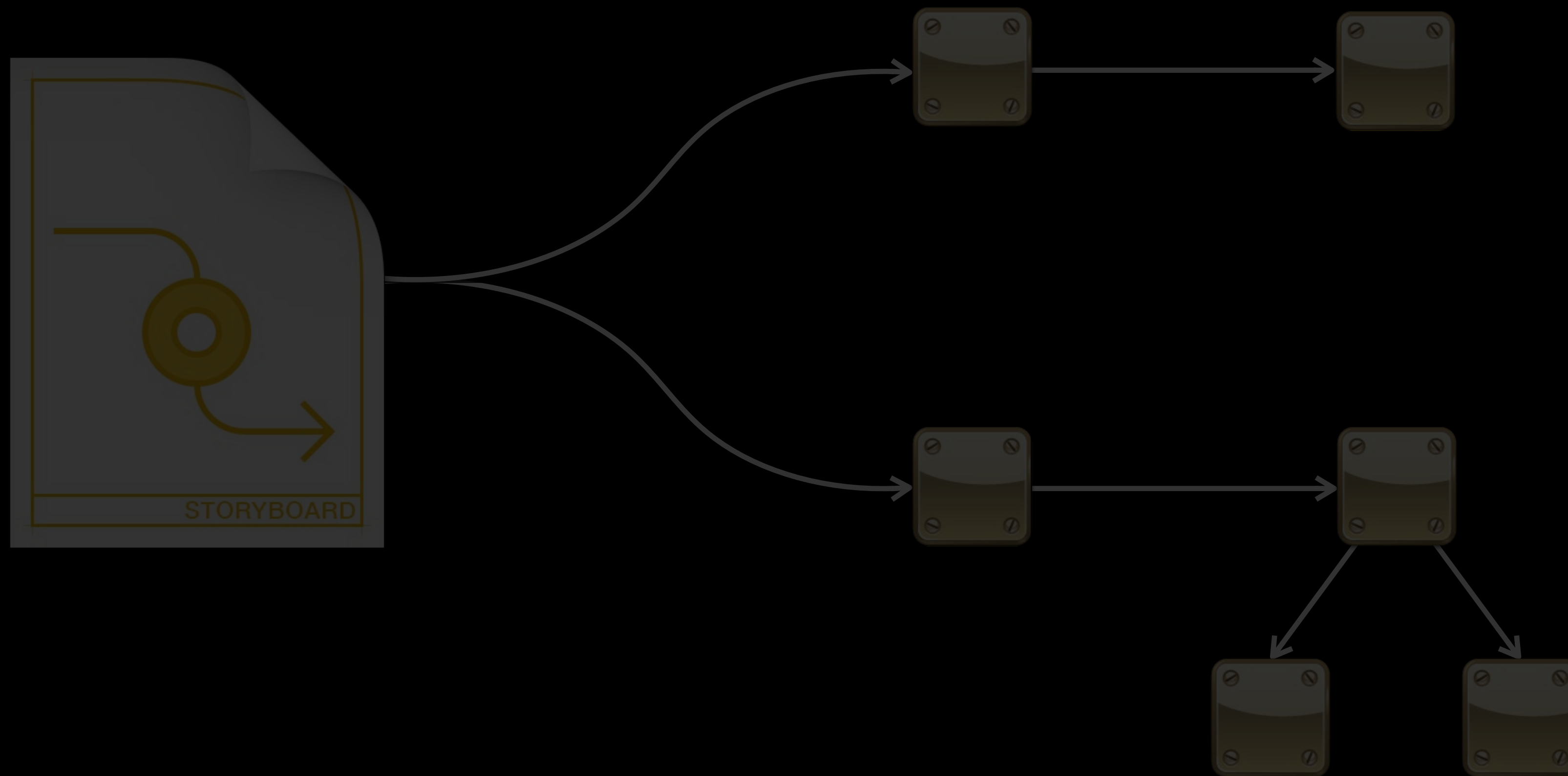
Compiling Storyboards



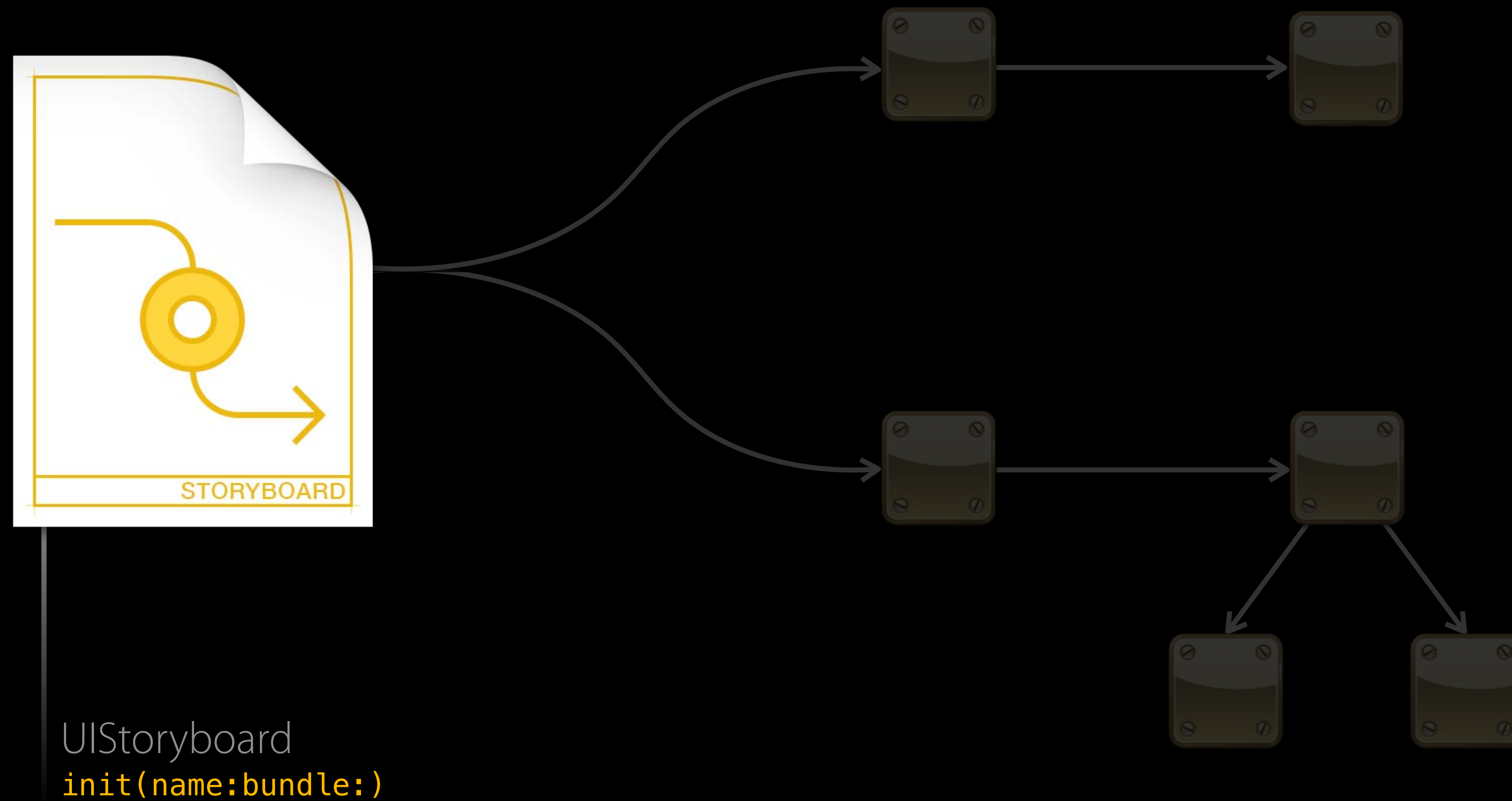
Compiling Storyboards



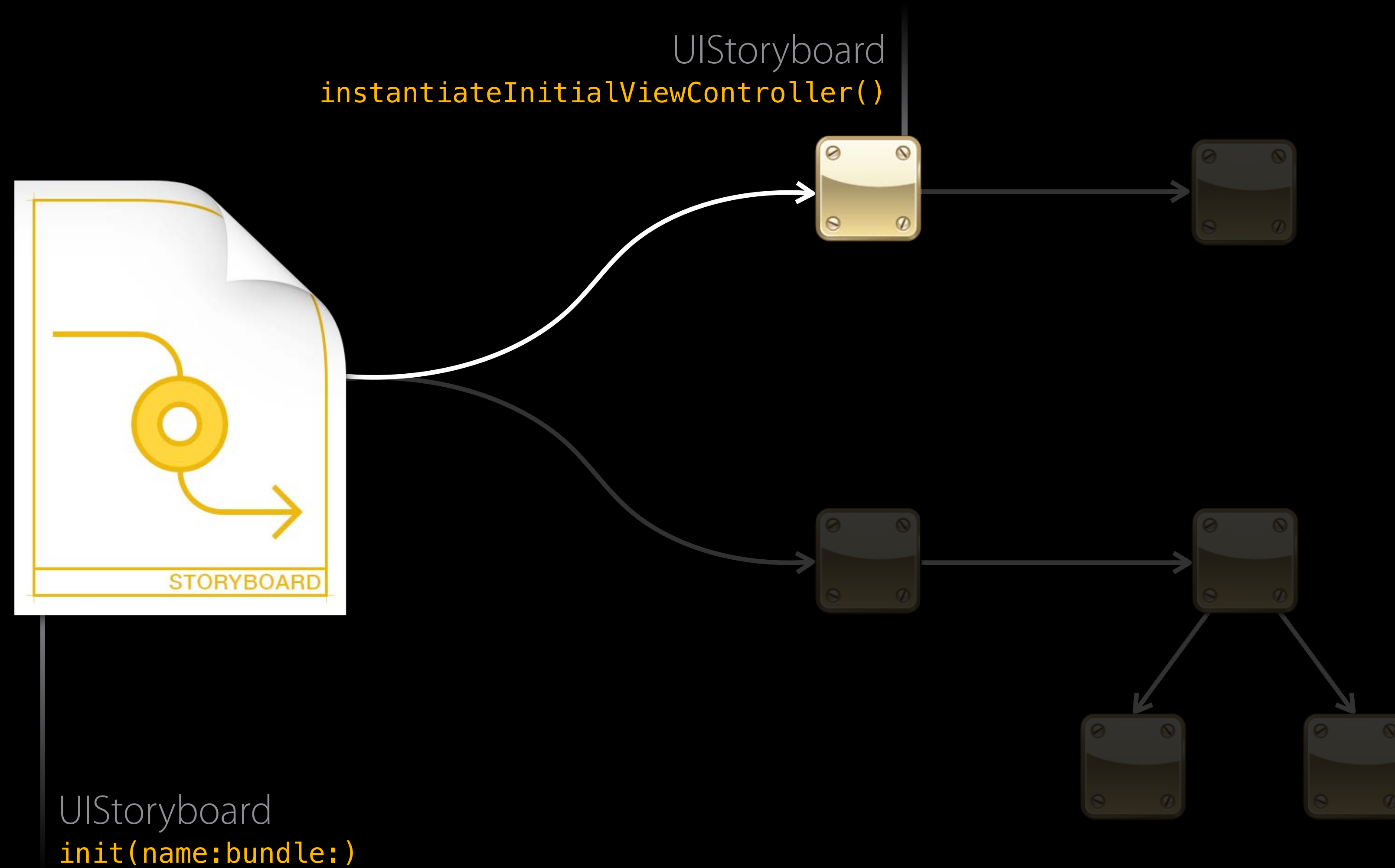
Loading Storyboards At Run Time



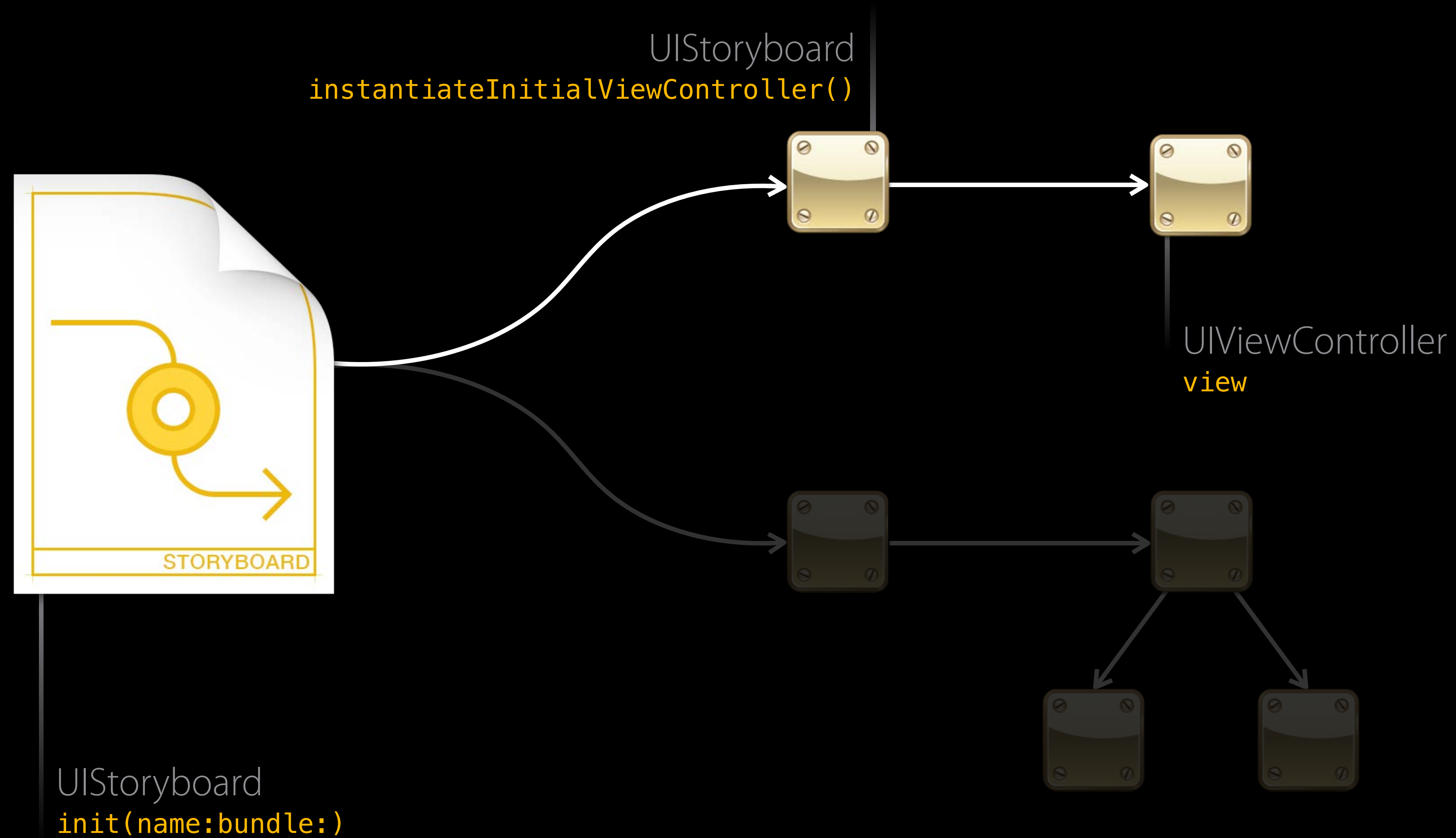
Loading Storyboards At Run Time



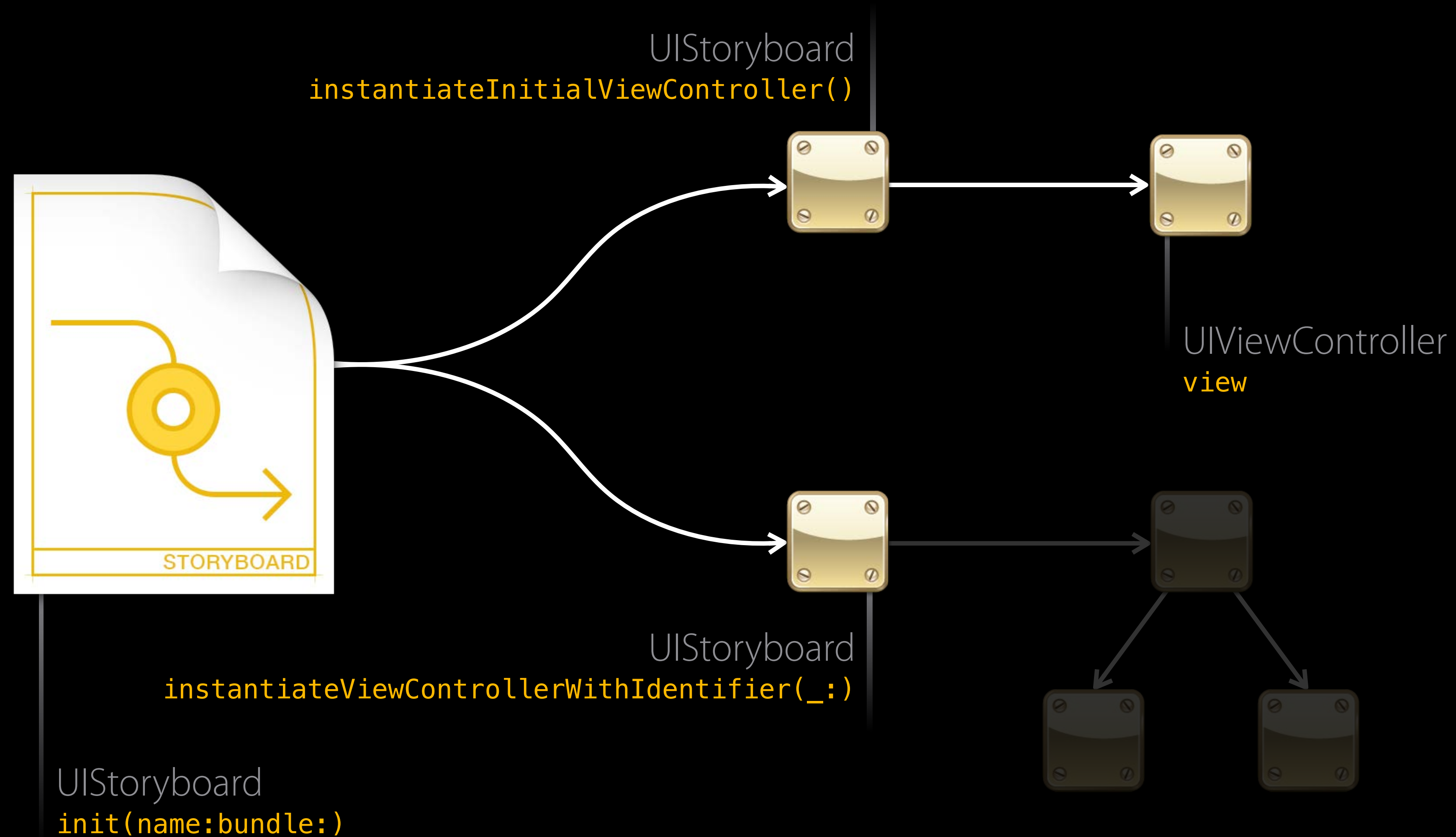
Loading Storyboards At Run Time



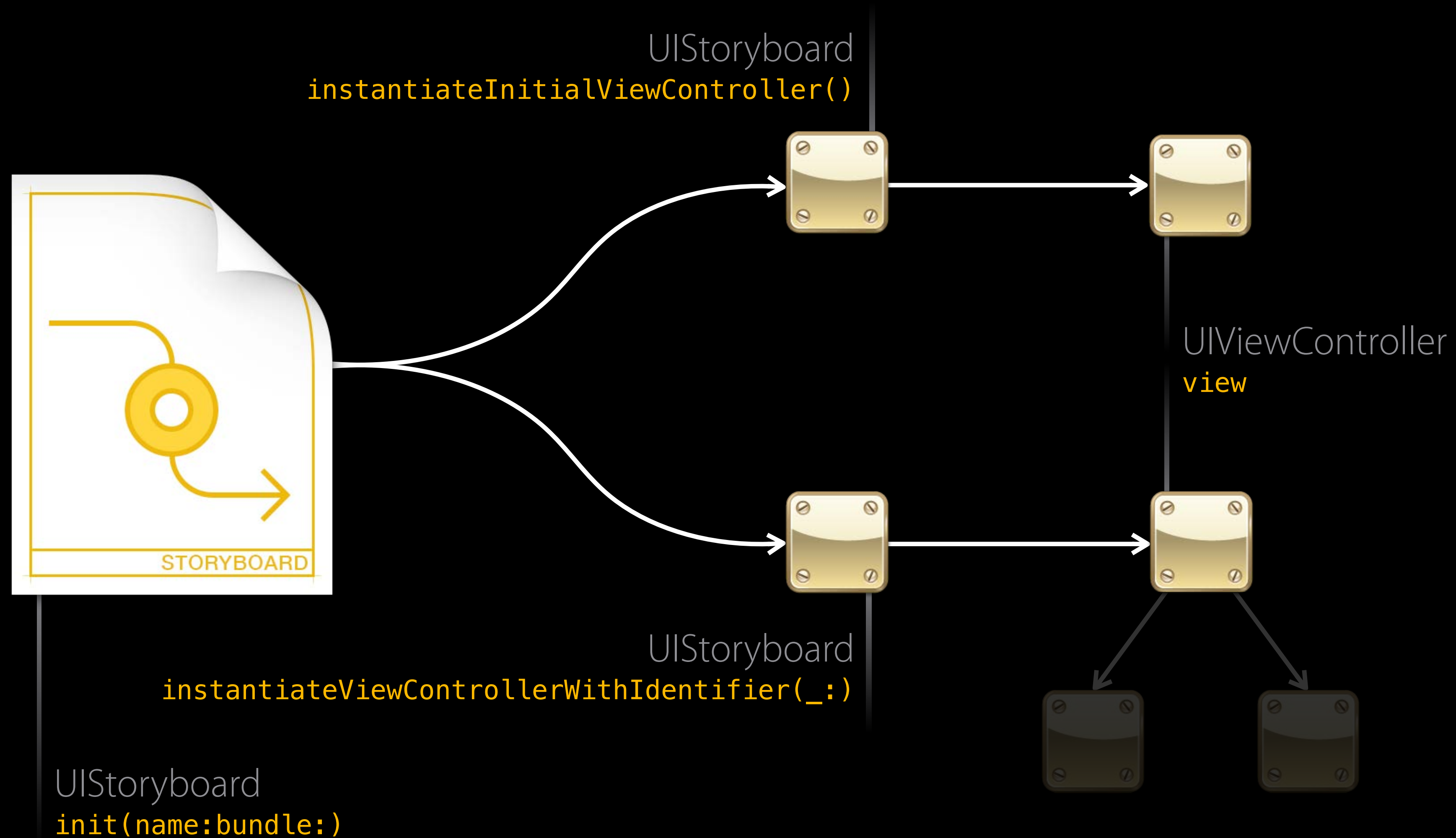
Loading Storyboards At Run Time



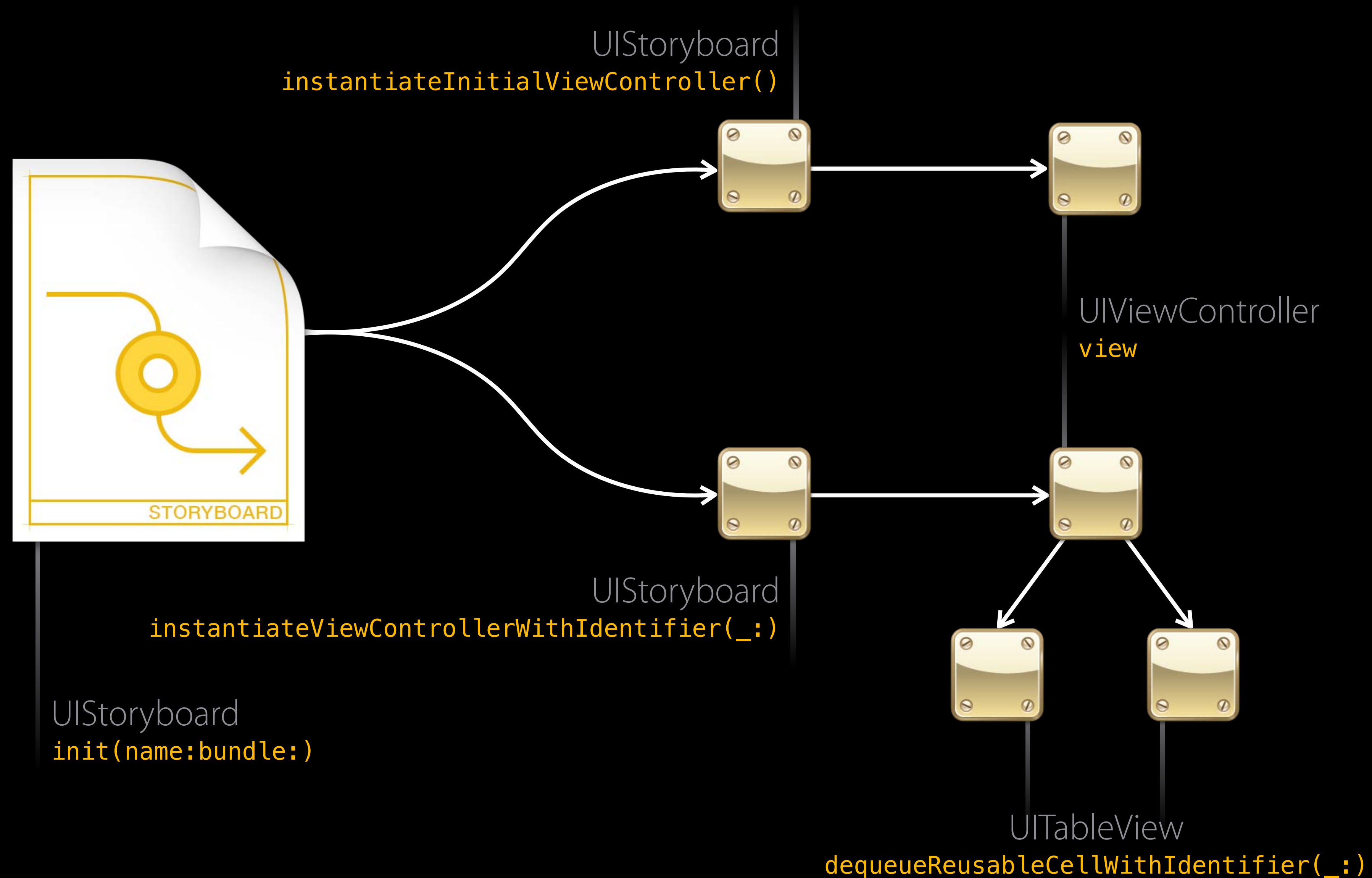
Loading Storyboards At Run Time



Loading Storyboards At Run Time



Loading Storyboards At Run Time



Takeaways

Takeaways

Performance. Nib files loaded on demand.

Takeaways

Performance. Nib files loaded on demand.

Reuse. Nib files enable reuse.

Takeaways

Performance. Nib files loaded on demand.

Reuse. Nib files enable reuse.

Life cycle. Know when objects are created.



Design Time



Build Time



Run Time



Design Time



Build Time



Run Time



Design Time



Build Time



Run Time



Design Time



Build Time



Run Time

Interface Builder at Run Time

Tony Ricciardi



Design Time



Build Time



Run Time



Design Time



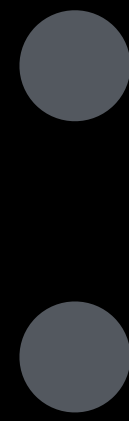
Build Time



Run Time



Run Time



Connections



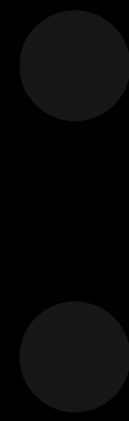
API



Adaptability



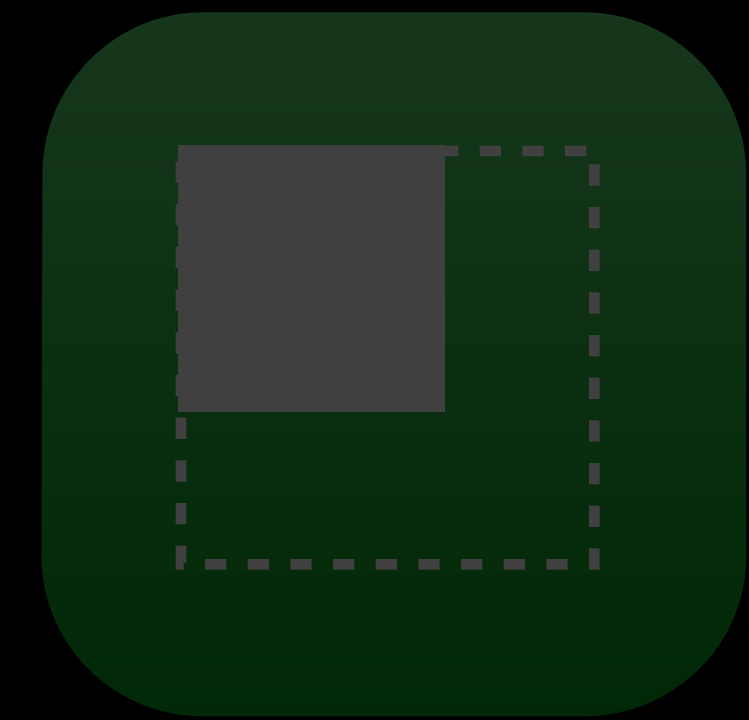
Run Time



Connections



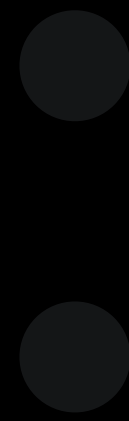
API



Adaptability



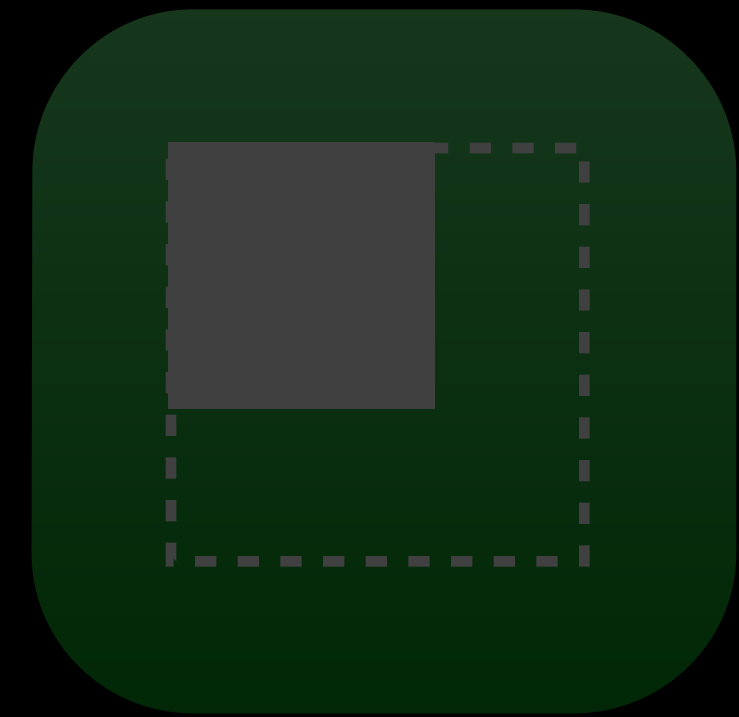
Run Time



Connections



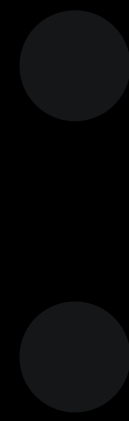
API



Adaptability



Run Time



Connections



API



Adaptability

Connections

```
class AccountViewController : UIViewController {  
  
    @IBOutlet var usernameLabel: UILabel!  
  
    override func viewDidLoad() {  
        usernameLabel.text = username  
    }  
  
    var username: String? {  
        didSet {  
            usernameLabel?.text = username  
        }  
    }  
}
```

Connections

```
class AccountViewController : UIViewController {  
  
    @IBOutlet var usernameLabel: UILabel!  
  
    override func viewDidLoad() {  
        usernameLabel.text = username  
    }  
  
    var username: String? {  
        didSet {  
            usernameLabel?.text = username  
        }  
    }  
  
}
```

Connections

```
class AccountViewController : UIViewController {  
  
    @IBOutlet var usernameLabel: UILabel!  
  
    override func viewDidLoad() {  
        usernameLabel.text = username  
    }  
  
    var username: String? {  
        didSet {  
            usernameLabel?.text = username  
        }  
    }  
  
}
```

Connections

```
class AccountViewController : UIViewController {  
  
    @IBOutlet var usernameLabel: UILabel!  
  
    override func viewDidLoad() {  
        usernameLabel.text = username  
    }  
  
    var username: String? {  
        didSet {  
            usernameLabel?.text = username  
        }  
    }  
  
}
```


Connections

```
class LoginViewController : UIViewController {  
  
    @IBAction func toggledAutoLoginSwitch(sender: UISwitch) {  
        UserSettings.autoLogin = sender.on  
    }  
  
    @IBAction func tappedLoginButton() {  
        if attemptLogin() {  
            performSegueWithIdentifier("unwindAfterLogin", sender: nil)  
        } else {  
            performSegueWithIdentifier("presentLoginError", sender: nil)  
        }  
    }  
  
}
```

Connections

```
class LoginViewController : UIViewController {  
  
    @IBAction func toggledAutoLoginSwitch(sender: UISwitch) {  
        UserSettings.autoLogin = sender.on  
    }  
  
    @IBAction func tappedLoginButton() {  
        if attemptLogin() {  
            performSegueWithIdentifier("unwindAfterLogin", sender: nil)  
        } else {  
            performSegueWithIdentifier("presentLoginError", sender: nil)  
        }  
    }  
}
```

Connections

```
class LoginViewController : UIViewController {  
  
    @IBAction func toggledAutoLoginSwitch(sender: UISwitch) {  
        UserSettings.autoLogin = sender.on  
    }  
  
    @IBAction func tappedLoginButton() {  
        if attemptLogin() {  
            performSegueWithIdentifier("unwindAfterLogin", sender: nil)  
        } else {  
            performSegueWithIdentifier("presentLoginError", sender: nil)  
        }  
    }  
  
}
```

API

UINavigationController:

```
init(name:bundle:)
```

```
func instantiateInitialViewController()
```

```
func instantiateViewControllerWithIdentifier(_:)
```

UIViewController:

```
var storyboard: UIStoryboard? { get }
```

API

UIViewController:

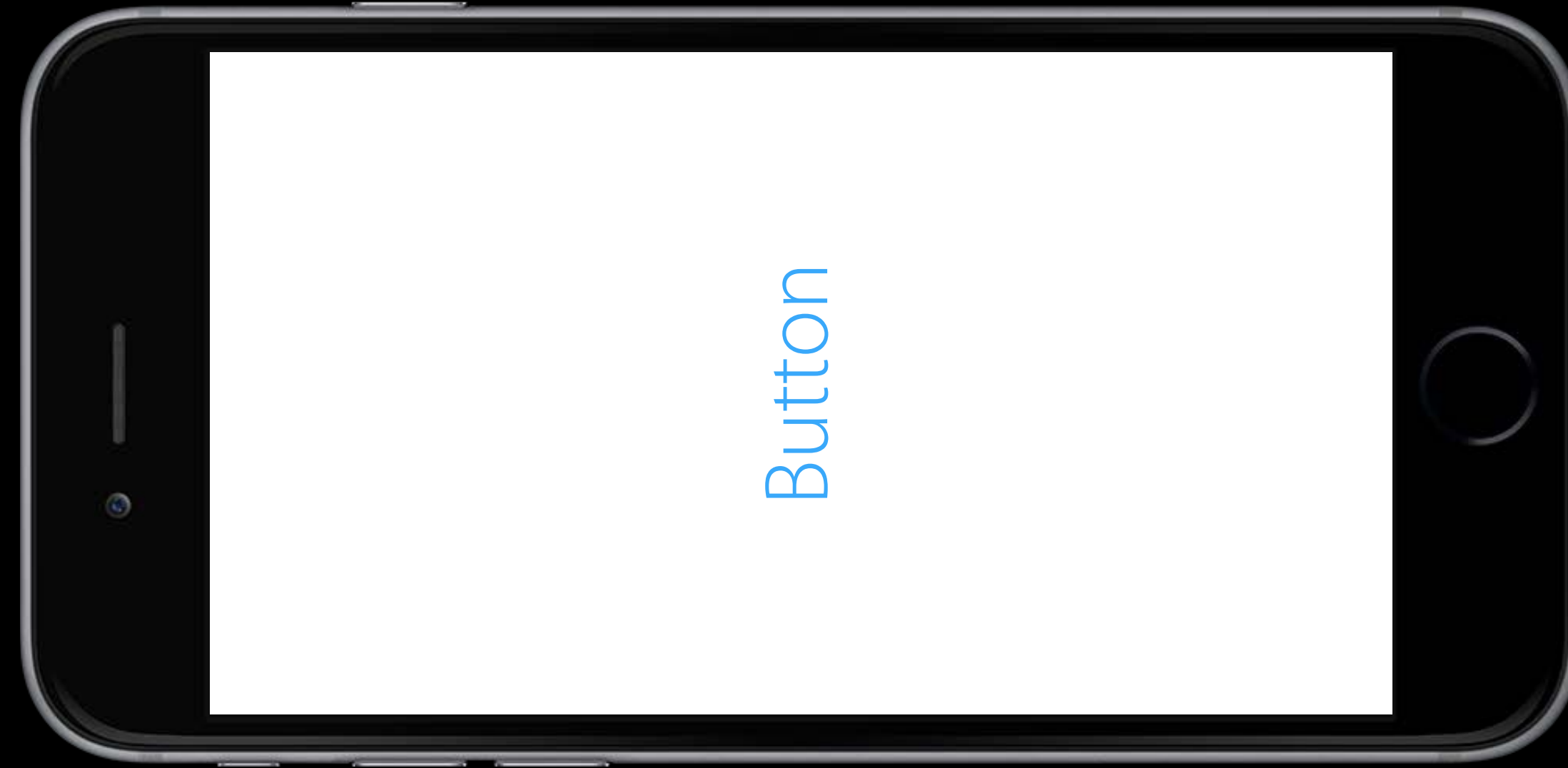
```
func prepareForSegue(_:sender:)  
func performSegueWithIdentifier(_:sender:)  
func shouldPerformSegueWithIdentifier(_:sender:) -> Bool  
func unwindForSegue(_:towardsViewController:)
```

UIStoryboardSegue:

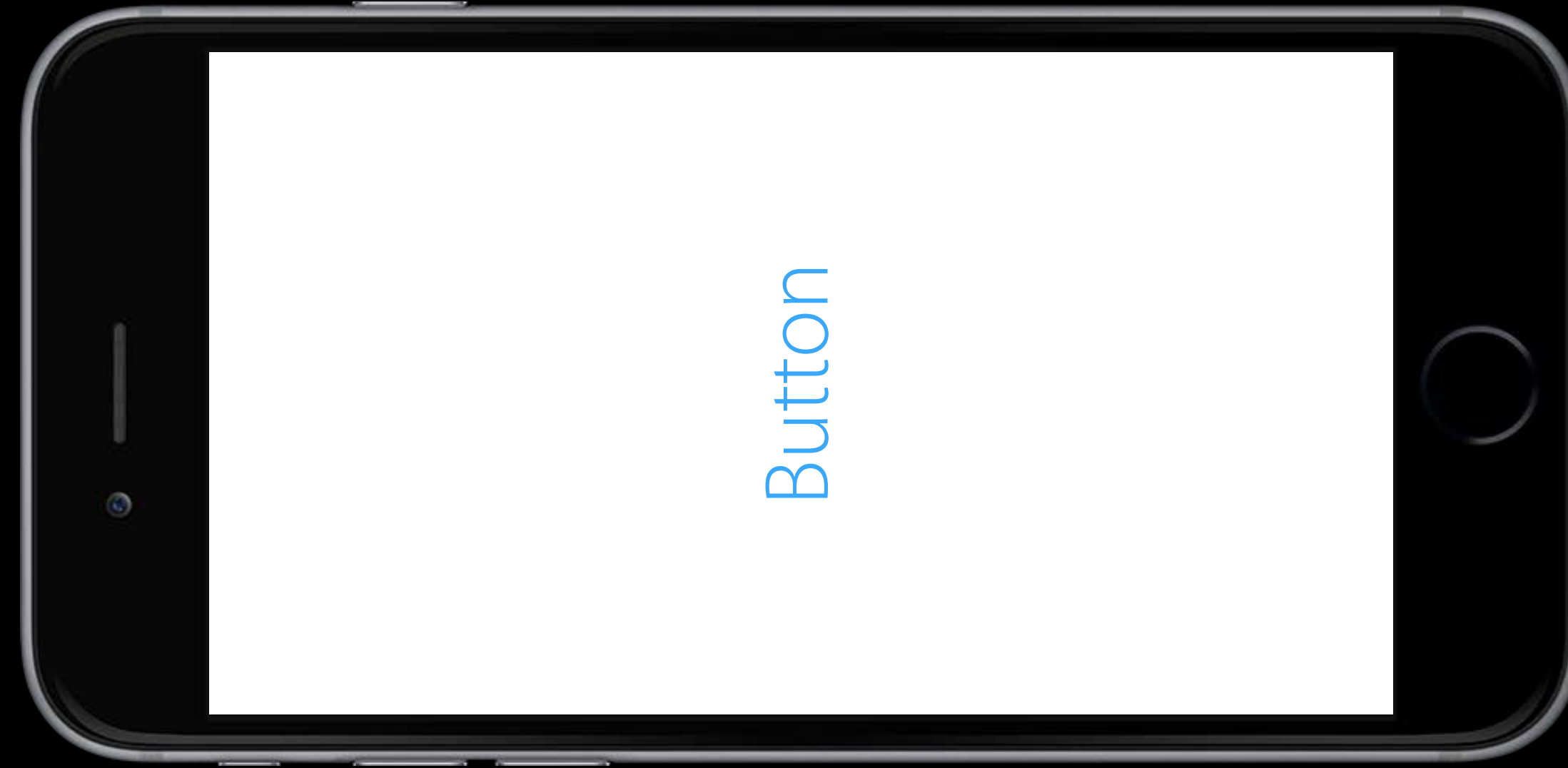
```
func perform()
```

Adaptability

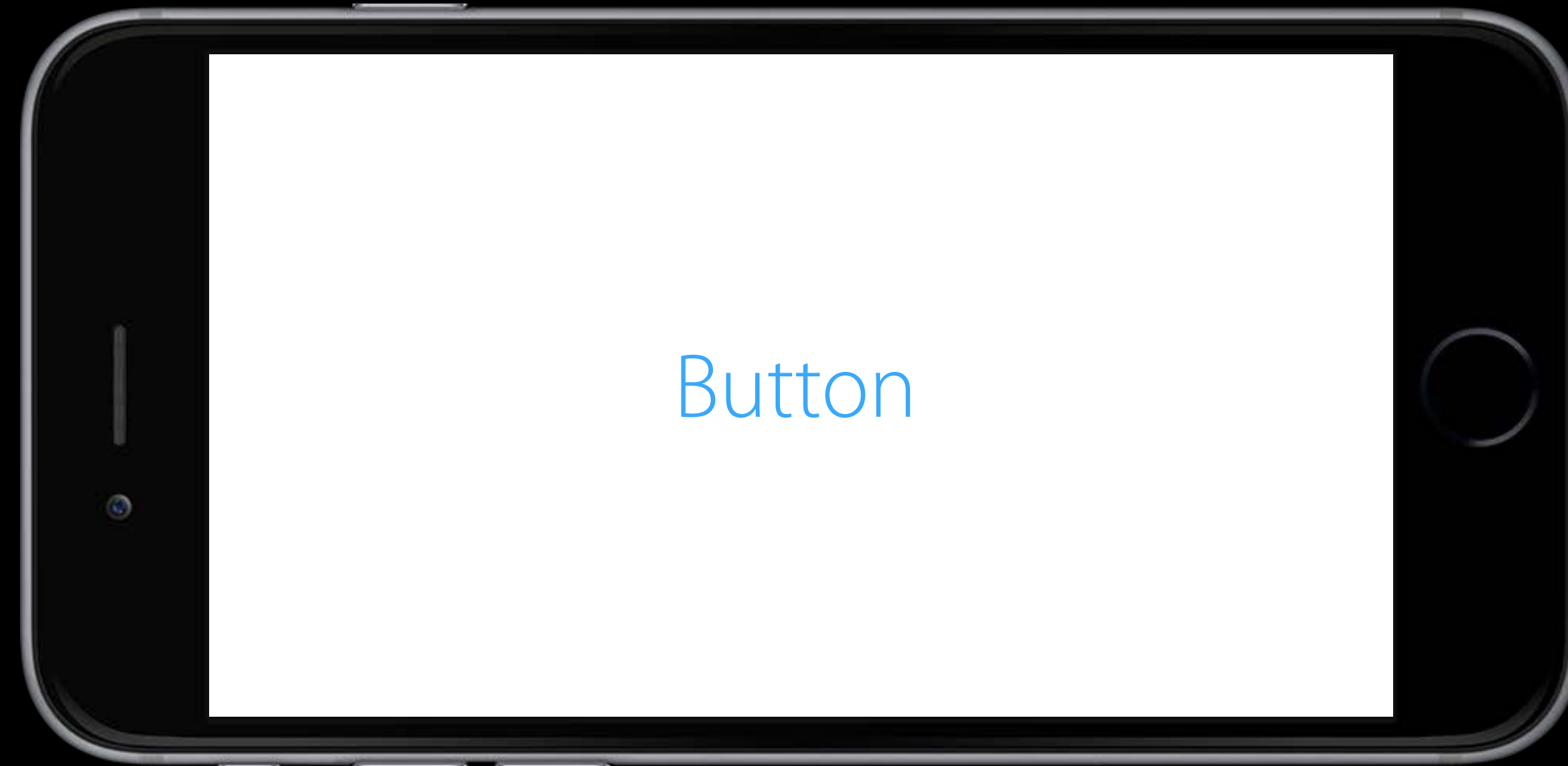
Adaptability



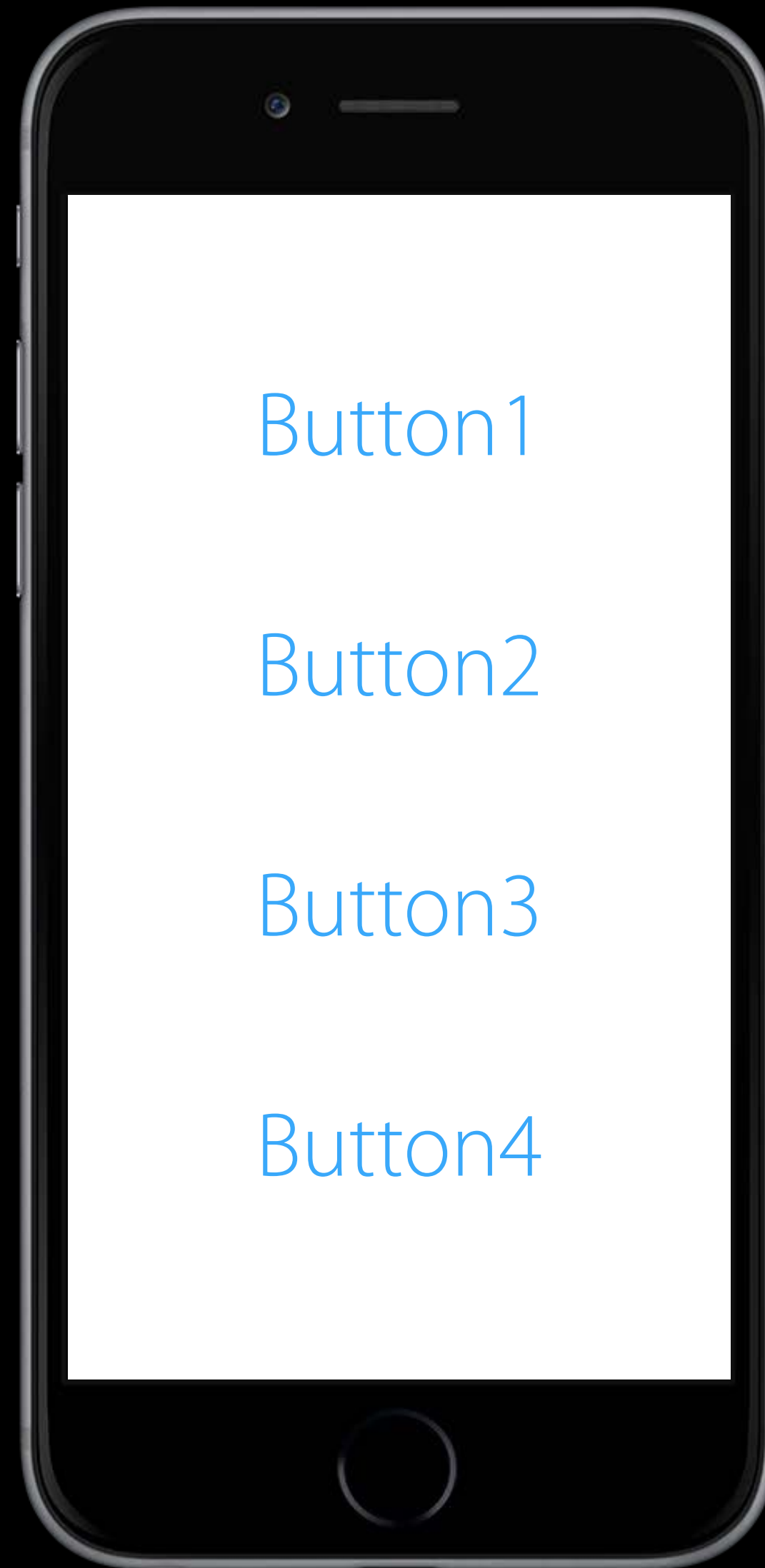
Adaptability



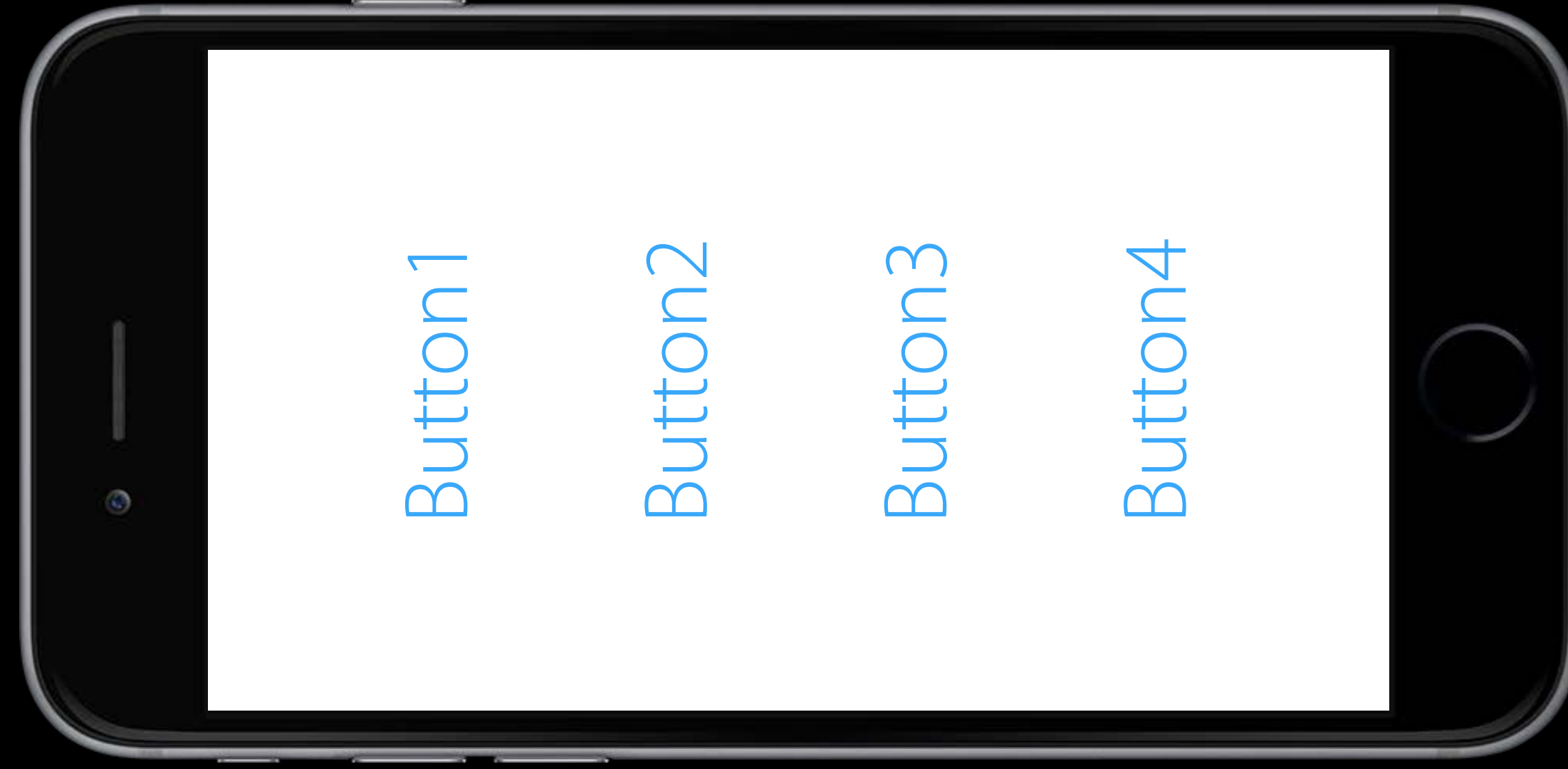
Adaptability



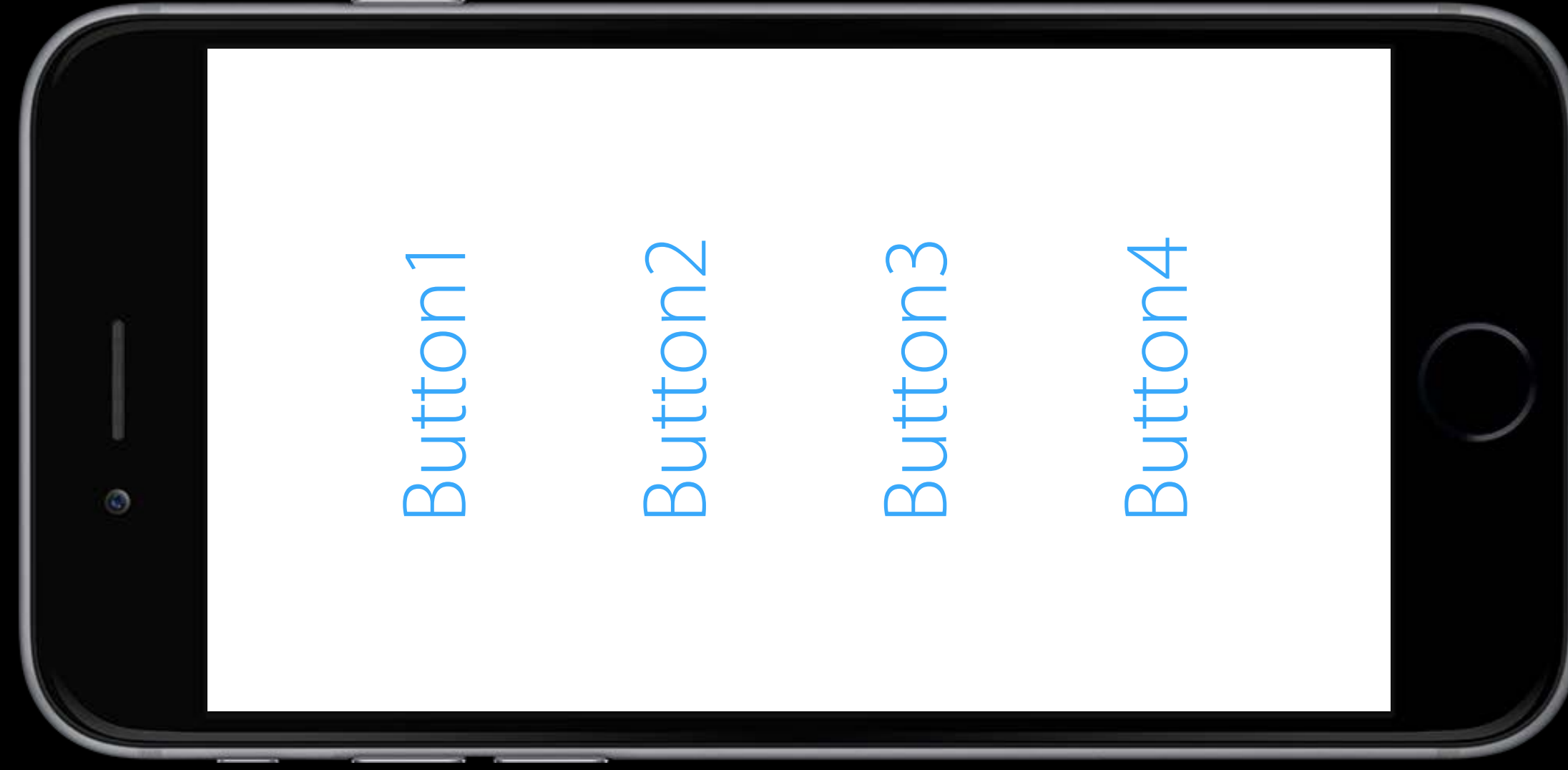
Adaptability



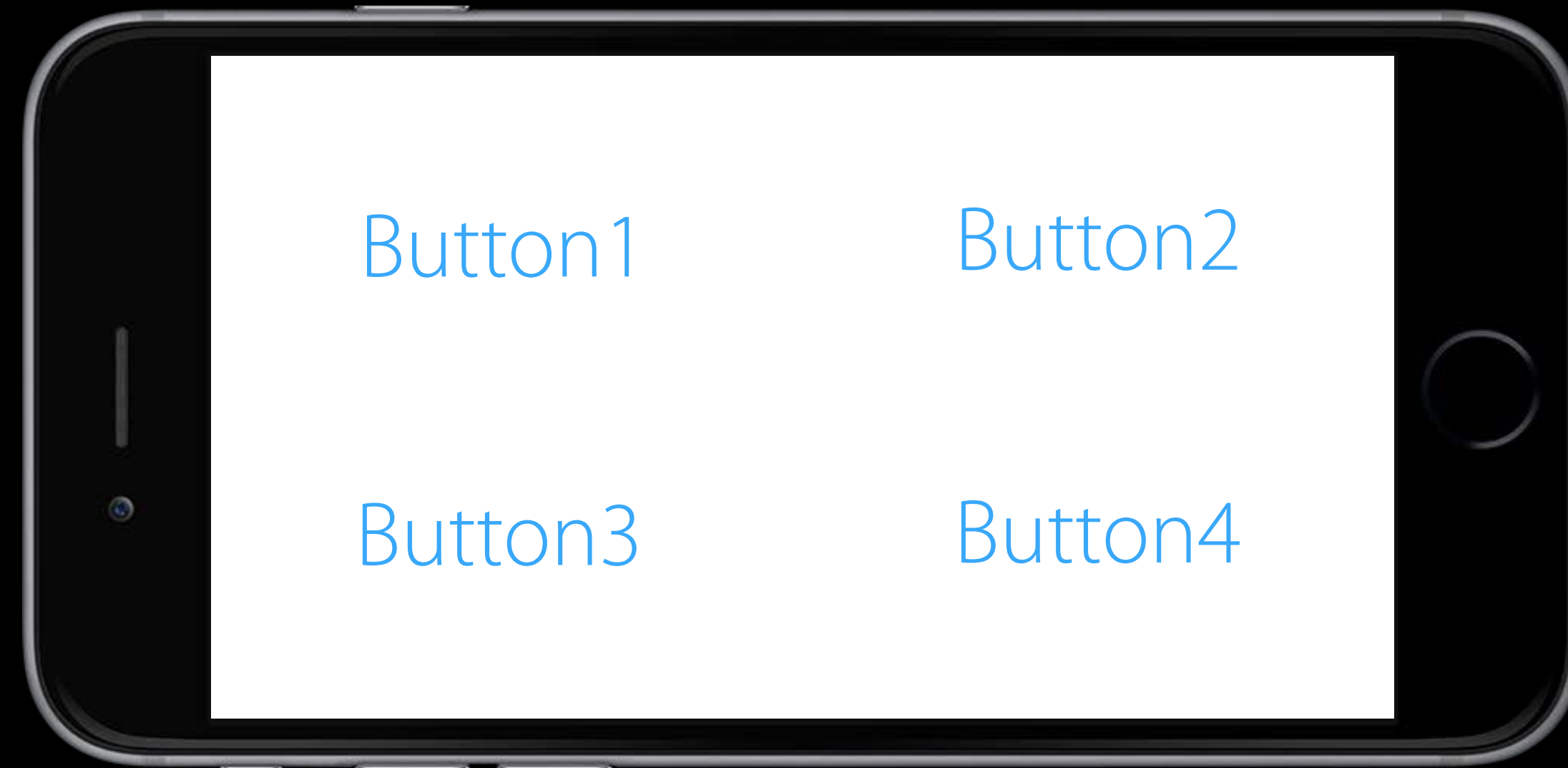
Adaptability



Adaptability



Adaptability





Connections



API



Adaptability

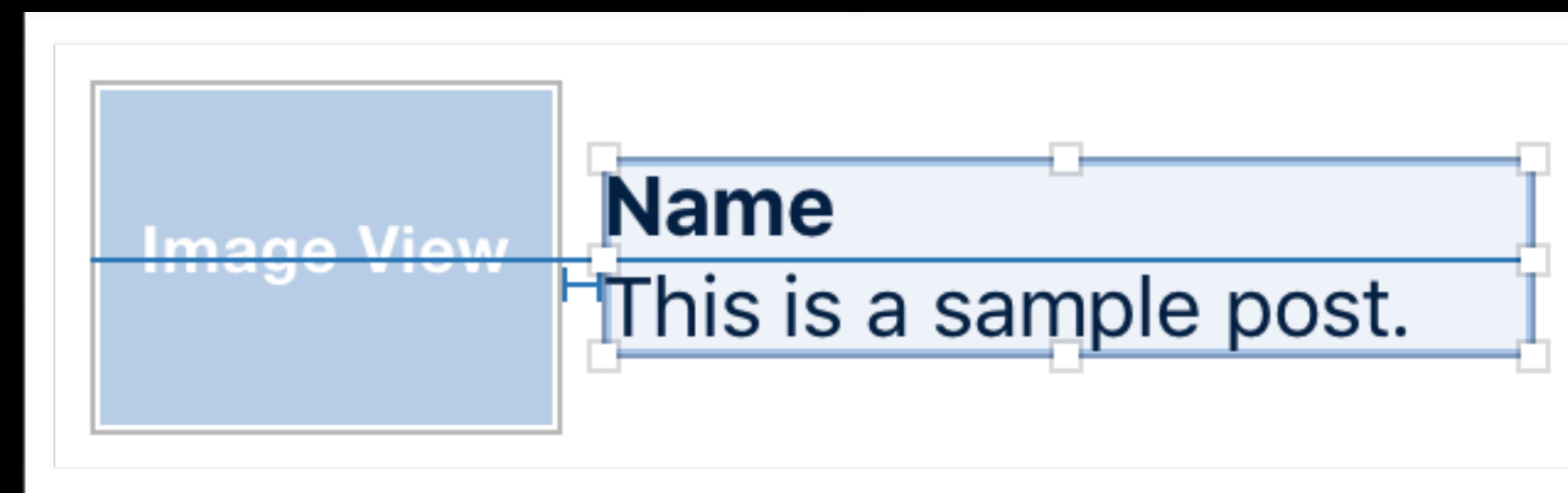
Demo

Interface Builder at Run Time

Summary

Summary

Design a flexible UI with constraints and stack views



Summary

Design a flexible UI with constraints and stack views

Rapidly iterate with designable views



Rob, Amber, and Leslie

Epic road trip to Utah for some hiking in Canyonlands!

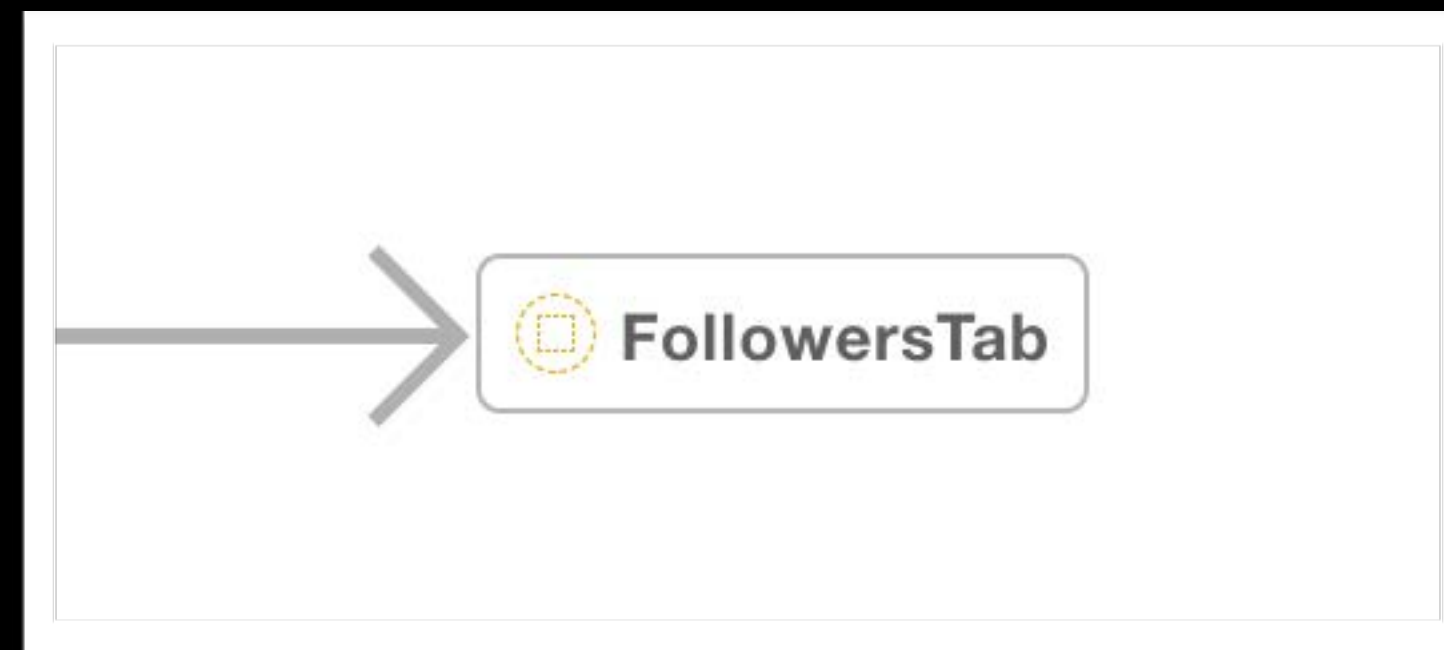
7 PHOTOS, 3 LOCATIONS

Summary

Design a flexible UI with constraints and stack views

Rapidly iterate with designable views

Modularize your UI with Storyboard References



Summary

Design a flexible UI with constraints and stack views

Rapidly iterate with designable views

Modularize your UI with Storyboard References

Reuse content with the storyboard API

`instantiateViewControllerWithIdentifier(_:)`

Summary

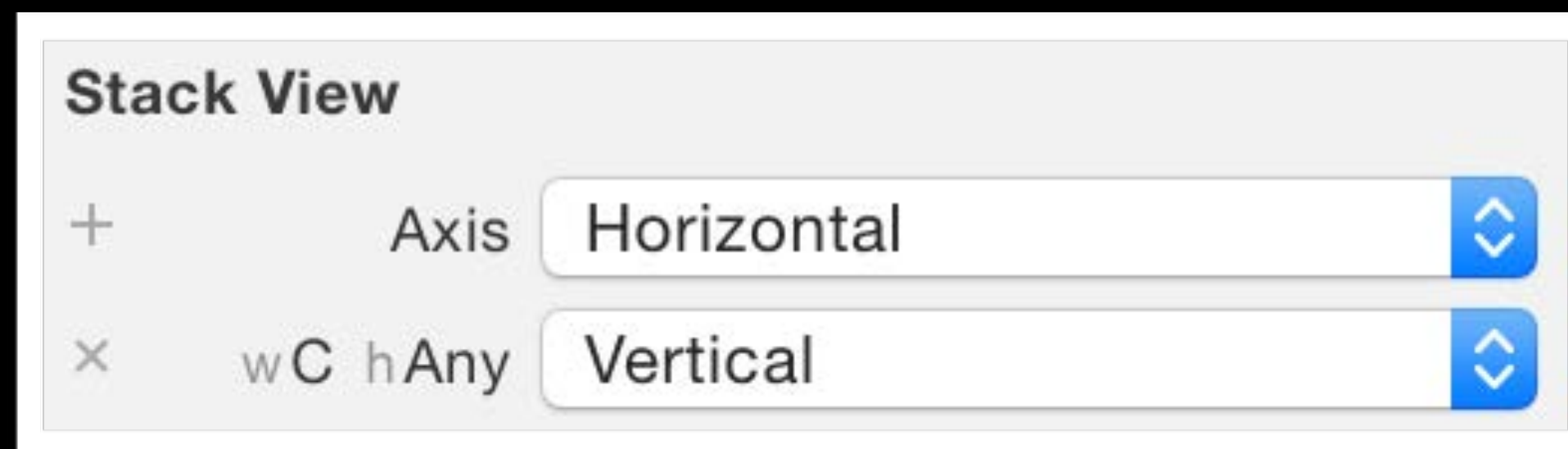
Design a flexible UI with constraints and stack views

Rapidly iterate with designable views

Modularize your UI with Storyboard References

Reuse content with the storyboard API

Make your UI adaptive with Size Classes



More Information

Apple Developer Forums

<http://developer.apple.com/forums>

Stefan Lesser

Developer Tools Evangelist

sless@apple.com

Related Sessions

What's New in Storyboards	Mission	Thursday 9:00AM
Mysteries of Auto Layout, Part 1	Presidio	Thursday 11:00AM
Mysteries of Auto Layout, Part 2	Presidio	Thursday 1:30PM
Building Adaptive Apps with UIKit		WWDC14
Taking Control of Auto Layout in Xcode		WWDC13

Related Labs

Interface Builder and Auto Layout

Developer Tools Lab C Thursday 2:30PM

