

Beginner

101: iOS Apprentice Review Part 1

Part 4: Challenge

iOS Apprentice Review Part 1

Part 4: Challenge Instructions

Copyright © 2014 Razeware LLC.

All rights reserved. No part of this book or corresponding materials (such as text, images, or source code) may be reproduced or distributed by any means without prior written permission of the copyright owner.

This book and all corresponding materials (such as source code) are provided on an "as is" basis, without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose, and noninfringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in action of contract, tort or otherwise, arising from, out of or in connection with the software or the use or other dealings in the software.

All trademarks and registered trademarks appearing in this book are the property of their respective owners.



Validation!

There's nothing worse than an app that requires information, but refuses to hold the user accountable for obtaining that information. This Challenge will focus on validating your users' input.

We have two challenges:

1. **Add an alert view** to give the user feedback on missing data.
2. **Disable the Generate Story button** in order to prevent the user from generating the story without the required data.

Challenge A: Adding An Alert Controller

This is the simplest way to prevent a user from trying to generate a story without the required data is to throw up an alert when they attempt to do so.

Start by adding the following code to `showAlert()` in **ViewController.swift**:

```
let alertController = UIAlertController(title: "Missing Data!",
message: "Hey, you're missing something!", preferredStyle: .Alert)

let defaultAction = UIAlertAction(title: "OK", style: .Default,
handler: nil)
alertController.addAction(defaultAction)

presentViewController(alertController, animated: true, completion:
nil)
```

The code creates a `UIAlertController` with the title and message set. It then presents the alert to the user.

Next, add the following code at the top of `generateStory()`:

```
if (textField1.text.isEmpty || textField2.text.isEmpty) {
    showAlert()
    return
}
```

This code checks for the presence of text in each of the text fields. If no text is present, it calls `showAlert()`.

Build & Run the app to see all the fantastical things you've done!



Challenge B: Disable Buttons

This challenge requires you to disable the generate story button until the user enters the required data. In order to do that, you'll need to use the text field protocol and delegate methods.

To do this, add the following code to `checkValidationStatus()` in

ViewController.swift:

```
if (textField1.text.isEmpty || textField2.text.isEmpty) {
    button.enabled = false
    button.alpha = 0.50
} else {
    button.enabled = true
    button.alpha = 1.0
}
```

Just like before, the validation code checks for the presence of text in the two fields. If text is not present, the button is disabled and its alpha value set to 0.50. If, on the other hand, text is present, the button is enabled and the alpha set to 1.0.

Next, you will need to add the following line of code to `viewDidLoad()`:

```
checkValidationStatus()
```

The reason you added it to `viewDidLoad` is because you want the Generate Story button to be disabled on load. While you could set this in the Interface Builder, sometimes setting values in code is easier (especially if you plan to have it do other things).

Note: You will also need to add that validation check to `resetStory()` since you want the button to be disabled after the story resets.

Finally, in order to detect changes to the text fields, you will need to set the two text fields' delegates. You will also need to populate the two delegate methods (`textFieldDidEndEditing` and `shouldChangeCharactersInRange`) with the following code:

```
checkValidationStatus()
```

Build & Run the app. Congratulations! You did it (I hope!). You should now have a fully functional Story Time app. If you're up for an extra challenge... turn the page.



Look, Ma... No Hands!

With the two previous challenges, steps were given to help you along. With this challenge, however, step-by-step instructions will not be provided. Ready?

Challenge C: Extending The Story

In this challenge, you will need to extend the story to include another character. Here's a brief explanation of what to do:

1. Add another UITextField to capture a second name.
2. Add an IBOutlet and connect it to the newly created UITextField.
3. Verify the input for the second name is included in the validation.
4. Update the story to include the second name.
5. Update all constraints.
6. Verify your screen looks like this:

Carrier 12:35 PM

CHOOSE YOUR STORY

Zombies Vampires Aliens

Names:

Bill Ted

Verb: ran

Number (64)

Monsters Win:

Your Story:

Bill and Ted entered the room and saw 64 aliens! Bill and Ted ran down the hall. Without missing a beat, Bill and Ted killed all of the aliens!

Fantastic!

Our heros will live to fight another day.

Start Over

