ATLS 4120/5120: Mobile Application Development
Week 4: Delegation

Tip Calculator
(tipcalculator)
Create a single view application, iPhone (or iPad), named tipcalculator.

Go into the storyboard and add 3 text fields with labels next to them(check amount, tip %, # of people), and then 3 labels(tip, total, total per person) with labels next to those to match mine. Think about where you should put the text fields so users can use the keyboard.
Text fields should have a font size of around 15 and use right alignment. Set the keyboard for the textfields to numbers and punctuation (look at number pad as well).
Set some defaults if you want.
You can also set the text fields to clear when you start editing (check box).
You can also customize the return key to say Done if you want.

Now let's make the connections for the text fields and labels on the right. They are all outlets.
checkAmount, tipPercent, people, tipDue, totalDue, totalDuePerPerson
Now these variables are defined in our swift file.

If you run it you might not see some of your fields.
Go back into the storyboard to set the constraints
Editor | Resolve Auto Layout Issues | All Views | Add Missing Constraints
Based on your layout Xcode will add what it thinks are the right constraints.
If you don't see your bottom row of labels remove the constraint they each have to the bottom layout guide. Now they should appear.
If you move any elements you need to update their constraints
This at least helps in portrait mode for now so we can keep working. We can tweak the constraints as needed later.

Snapshot: UI

Try taping one of the text fields. Even with no code the keyboard shows up but you can't get rid of it since this keyboard doesn't have a dismiss keyboard button.
We need to adopt the UITextField protocol.

```
class ViewController: UIViewController, UITextFieldDelegate
```

Implement the UITextFieldDelegate method that is called when the return button is pressed

```
    func textFieldShouldReturn(textField: UITextField) -> Bool {
        textField.resignFirstResponder()
        return true
    }
```

Assign our controller as the delegate for each UITextField. Then the text field will tell its delegate when certain events, like the return key being tapped, take place.

```
override func viewDidLoad() {
        checkAmount.delegate=self
        tipPercent.delegate=self
        people.delegate=self
```

```
        super.viewDidLoad()
}
```
viewDidLoad is called automatically after the views have been initialized but before it's displayed.
Run the app and see if the return key dismisses the keyboard.
If you also want to dismiss the keyboard by tapping the background, the book covers it in chapter 4.

Note where your keyboard comes up. Is it blocking any of the text fields? Do you need to fix the layout of your UI so it works with a keyboard?

Snapshot: textfield delegate

(remove text from the computed labels)

Now it's time to write the method that calculates the tip, total price, and price per person.

```
    func updateTipTotals() {
        let amount = (checkAmount.text as NSString).floatValue
        let pct = (tipPercent.text as NSString).floatValue/100
        let numberOfPeople=people.text.toInt() //returns an optional
        let tip=amount*pct
        let total=amount+tip
        var personTotal : Float = 0.0
        if numberOfPeople != nil {
            if numberOfPeople! > 0 {
                personTotal = total / Float(numberOfPeople!)
            }
        }
        var currencyFormatter = NSNumberFormatter()
        currencyFormatter.numberStyle=NSNumberFormatterStyle.CurrencyStyle
        tipDue.text=currencyFormatter.stringFromNumber(tip)
        totalDue.text=currencyFormatter.stringFromNumber(total)

totalDuePerPerson.text=currencyFormatter.stringFromNumber(personTotal)
    }
```

When should we call this method?
We want the tip and totals fields to be updated when any of the text fields are changed. We could use textFieldShouldReturn but that is only called when the user hits Return. What if they just click in one text field and then another?
The UITextFieldDelegate has another method called textFieldDidEndEditing that is called when the user finishes editing a text field (pressing Done or switching to another field). So let's call updateTipTotals from textFieldDidEndEditing so the tip labels will update as soon as the user finishes editing any text field.

```
    func textFieldDidEndEditing(textField: UITextField) {
        updateTipTotals()
    }
```

Now that your labels have values if they're not lined up, select them all and add a leading edge alignment.

Snapshot: tip calculated 2

Alert

Although we want to use alerts sparingly because they interrupt the user experience, using them to warn the user of errors is a good use of them.
So let's add an alert if the user has number of people at 0.

We update `updateTipTotals` (add else statement to the if)

```
else {
    //create a UIAlertController object
    let alert=UIAlertController(title: "Warning", message: "The number of
people must be greater than 0", preferredStyle:
UIAlertControllerStyle.Alert)
    //create a UIAlertAction object for the button
    let cancelAction =UIAlertAction(title: "Cancel",
style:UIAlertActionStyle.Cancel, handler: nil)
    alert.addAction(cancelAction) //adds the alert action to the alert
object
    let okAction =UIAlertAction(title: "OK", style:
UIAlertActionStyle.Default, handler: { action in
                        self.people.text="1"
                        self.updateTipTotals()
                    })
    alert.addAction(okAction)
    presentViewController(alert, animated: true, completion: nil)
                } //end else
```

The handler parameter takes a closure. A closure is a block of code, like a function without a name. (You can actually write a function instead and just call it here) This is similar to an anonymous function.

Test it with different values for people.
You might get a warning, just ignore it.

Snapshot: alert

Update your launch screen and add app icons

Snapshot: app icons 2

Try the app in landscape for different size phones, does it work?
If you can't get the constraints to work for all, change the size class to wAny hCompact and modify the constraints for iPhone landscape.

Snapshot: landscape constraints