Swift course notes

Complete iOS 10 developer course

XCode

2-7 Ctrl-drag to code window to connect

Connect could be of type Outlet or Action or OutletCollection (eg for Button) - action for whatever is to happen - usually placed within ViewController class

import UIKit

2-8 XCode/Preferences - object.method/property - eg. @IBAction func changeButton(\_sender: Any) { labelPJ.text = “Hello” }

skipping some lessons on Basics of Swift

3-19 Classes -

Playground for ‘new ideas’

choice of iOS, macOS or tvOS

// e.g.

class User {

var name = ""

var age = 0

var favouriteMovie = ""

var favouriteWeather = ""

var isDeveloper = false

}

var newUser = User()

newUser.name = "Dashiel Something"

newUser.age = 30

newUser.favouriteMovie = "Pulp Fiction"

newUser.favouriteWeather = "cloudy"

newUser.isDeveloper = false

3-20 Optionals

like nullable variables in C# , suffix ‘?’ to avoid error if variable not set ie ‘nil’ - to be unwrapped (not knowing if any contents in advance)

Alt+3 for hash symbol

//e.g.

//var myName : String = "James"

var myName : String? // optional by '?' - nullable

myName?.uppercased()

<https://cocoacasts.com/what-does-an-exclamation-mark-mean-in-swift/>

TODO: optional binding

optional chaining e.g. person?.sayHello()

person?.name = "John" // not sure if property or method d’exist

Implicitly unwrapped optional - ‘!’ - if certain that values will exist - crash otherwise

class Person {

var name: String!

func sayHello() {

print("hello")

}

}

let person = Person()

person.name = "John"

print(person.name)

3-22

XCode Interface Builder GUI libraries - File template, Code snippet, Object, Media

labels &c in Object, mini-patterns/expressions in Code snippet

right-sidebar Utilites Area has: File Inspector, Quick Help, Identity Inspector, Attributes Inspector, Size Inspector, Connections Inspector

File Inspector: XCode project details

Quick Help,

Identity Inspector: object class, GUID

Attributes Inspector: bespoke to object eg label text, font

Size Inspector: on-screen size

Connections Inspector: outlets & actions / connections for object (event stuff) 2-7

Alt + drag to copy

Assistant Editor oo to see code

Version editor <=> to see XML

Assistant Editor should open ViewController.swift

Ctrl-drag as in 2-7 to connect Text box to class ViewController

(may need to open Main.storyboard in separate window and drag from this window to ViewController.swift, to connect from text box, eg.)

Alternatively Ctrl-drag from ViewController Scene/View/individual components listed

e.g. @IBOutlet weak var firstNumberTxt: UITextField!

@IBAction func plusBtn(\_ sender: Any) {

}

NB weak, strong, unowned - to do with memory management - Automatic Reference Counting for classes (not structs) deallocated when ARC count zero - strong is default - strong increment ARC by 1 - weak won’t increment ARC, automatically set to nil on deallocation, useful for possible 2-way/circular references, must be Optional - unowned doesn’t increment ARC and need not be Optional, not automaticlly set to nil on deallocation, should only be used if object never reset to nil - <https://medium.com/@chris_dus/strong-weak-unowned-reference-counting-in-swift-5813fa454f30>

viewDidLoad is default method when view has loaded

<https://www.tech-recipes.com/rx/52021/how-do-i-fix-the-issue-this-class-is-not-key-value-coding-compliant-for-the-key-in-xcode-6/>

if error e.g.

label.text assign thowing EXC\_BAD\_INSTRUCTION (code=exc\_i386\_invop subcode=0x0)

try deleting the reference to the object (e.g. myLabel) and then (double-clicking Main storyboard to get draggable control) re-drag from object/control (e.g. label) into code to re-make Outlet

<https://stackoverflow.com/a/24091892>

my try at catching non-number inputs:

@IBAction func plusBtn(\_ sender: Any) {

if let \_ = Int(firstNumberTxt.text!) , let \_ = Int(secondNumberTxt.text!)

{ // try to catch non-number string input <https://stackoverflow.com/a/38159489>

result = Int(firstNumberTxt.text!)! + Int(secondNumberTxt.text!)!

// implicitly unwrapped optional - need to be certain that values will exist

resultLabel.text = String(result)

} else

{

resultLabel.text = "Please type numbers"

}

}

3-23 string interpolation \(myVar)

eg “Result is \(String(result))”

[sketchappsources.com](http://sketchappsources.com)

5-29 Layout resizing (in code) for devices

Cmd+R run

eg layout ok in iphone 6 & 7 but off-screen in iphone 5, e.g.

5-30 ‘Align’ button at bottom e.g. ‘horizontally in container’, ‘top edges x pixels’ ‘vertical centers’ &c

click tick box and ‘add 1 constraint’

red indicating error (see message) eg ‘need constraint for Y position’

so…

‘Add Constraint’ button at bottom - values can be entered for width/height/margins &c

‘Resolve auto-Layout’ button at bottom can fill in values, or option to ‘Clear Contraints’

remove constraints if clash (manually or Editor/Resolve Auto-Layout Issues…)

5-31 Project Layout can be set to a device tpe e.g. iphone 5

‘Resolve auto-Layout’ button at bottom can fill in values,as per ‘Reset to Suggested Constraints’

Size Inspector control at top[-right in Utilities has an area for Constraints -

5-33. programmatic placement of objects/controls

override func viewDidLoad() {

super.viewDidLoad()

// Do any additional setup after loading the view, typically from a nib.

// default action on loading

let width = view.frame.size.width

let height = view.frame.size.height

let myLabel = UILabel() // let not var as Label will not be changed to another type (immutable) <https://stackoverflow.com/a/24002209> <https://stackoverflow.com/a/31651715>

myLabel.text = "My code-written label"

myLabel.textAlignment = NSTextAlignment.center // key-in '.' to get options

myLabel.textAlignment = .center

//myLabel.frame = CGRect(x:10, y:10, width: 100, height:50)

myLabel.frame = CGRect(x:width \* 0.1, y: height \* 0.3, width: width \* 0.8, height:50)

view.addSubview(myLabel)

}

5-34. Adding button and click event, programatically

Alt+3 for hash symbol

Alt+drag to copy control/object

6-37 UserDefaults to persist data

@IBAction func saveButtonClicked(\_ sender: Any) {

UserDefaults.standard.set(nameText.text, forKey: "name")

UserDefaults.standard.set(birthdayText.text, forKey: "birthday")

UserDefaults.standard.synchronize()

// persist data in key/value pairs using UserDefaults <https://developer.apple.com/documentation/foundation/userdefaults>

cmd+R run

cmd+. stop

cmd+B build

6-39

if let \_ = storedDate as? String { // underscore with let to avoid warning, compiler understands as temporary constant (!)

// if possible to create storedName, cast to String, from (hopefully non-nil) value retrieved...

UserDefaults.standard.removeObject(forKey: "birthday")

UserDefaults.standard.synchronize()

storedDateLabel.text = "Stored date: "

}

6-40 another view

1. drag another ViewControl onto the storyboard
2. File/New/Cocoa Touch Class (e.g.), make a subclass of ‘UI View Controller’ -

* place new file in same directory

1. Select second ViewController in storyboard (left-most icon at top of viewcontroller object), in Utilities (on right) select Identity Inspector, to name storyboard object after code ViewController, and to connect object to object in code in new .swift file.

connections can then be made between the storyboard objects and their owning class (View Controller) in the code

NB - .xib - XML Interface Builder - compiled into .nib NeXT Interface Builder when built - binary - <https://www.c-sharpcorner.com/UploadFile/d49768/difference-between-xib-and-nib-in-iphone/>.

<https://developer.apple.com/library/archive/documentation/General/Conceptual/DevPedia-CocoaCore/NibFile.html>

Cocoa - <https://en.wikipedia.org/wiki/Cocoa_(API)>. -

6-41 segue between ViewController screens in an app

Ctrl+drag from button to other ViewController in storyboard d’give menu ‘Action Seque’ with e.g. Show, Show Detail, Present Modally, Present as Popover, …

also segue from left-most ViewController icon at top (Ctrl+drag)

using e.g. "fromFirstToSecond” name Segue in Identity Inspector

@IBAction func saveBtnClicked(\_ sender: Any) {

performSegue(withIdentifier: "fromFirstToSecond"

}

6-42 data between ViewControllers

in 1st ViewController:

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {

if segue.identifier == "fromFirstToSecond" // NB case-sensitive

{

let destinationVC = segue.destination as! SecondViewController

// 'as!' forced/risky casting/conversion <https://developer.apple.com/swift/blog/?id=23>

// SecondViewcontroller (segue's destinationVC) will be a constant

// ... giving access to variables & other members of segue destination ViewController

// all via prepare

destinationVC.userName = name // use local var

print("preparing-" + name)

}

}

// execute segue previously defined on storyboard

@IBAction func saveBtnClicked(\_ sender: Any) {

name = nameText.text!

// record in local variable when clicked (NB had to use implicit '!')

performSegue(withIdentifier: "fromFirstToSecond", sender: nil)

}

in 2nd ViewController:

nameLabel.text = "Your name is: \(userName)"

to return from a segue, in 2nd ViewController:

self.dismiss(animated: true, completion: nil)

6-43. alerts

let alert = UIAlertController(title: "Error", message: "oh dear", preferredStyle: UIAlertControllerStyle.alert)

let okButton = UIAlertAction(title: "OK", style: UIAlertActionStyle.cancel, handler: nil)

alert.addAction(okButton)

self.present(alert, animated: true, completion: nil)

6-44

6-47 ImageViews

imageViewActual.image = UIImage(named: "hendrix.jpeg")

@IBOutlet weak var imageViewActual: UIImageView!

Stack view. - easier formatting viz constraints - <https://www.raywenderlich.com/508-uistackview-tutorial-introducing-stack-views>

<https://developer.apple.com/library/archive/documentation/UserExperience/Conceptual/AutolayoutPG/LayoutUsingStackViews.html>

subviews stacked, horizontally/vertically

6-49 Gesture recognition -

super.viewDidLoad()

// Do any additional setup after loading the view, typically from a nib.

imageView.isUserInteractionEnabled = true

let gestureRecognizer = UITapGestureRecognizer(target: self,

action:#selector(ViewController.changePicture))

// self is ViewController; NB hash to mark as selector for an action

imageView.addGestureRecognizer(gestureRecognizer)

NB for ‘gesture recognition’, 3 steps: (1) declare/define a UITapGestureRecognizer recogniser, (2) .addGestureRecognizer to ImageView, (3) ensure imageView.isUserInteractionEnabled = true

(4) write code for selector function ie ViewController.changePicture so func changePicture(){….

ie gesture, recognizer & enabled interaction (1-3)

6-51 timers

override func viewDidLoad() {

super.viewDidLoad()

// Do any additional setup after loading the view, typically from a nib.

counter = 10

timerLabel.text = String(counter)

timer = Timer.scheduledTimer(timeInterval: 1, target: self, selector: #selector(ViewController.timerFunction), userInfo: nil, repeats: true)

// 1 second, repeating, select timerFunction as event on timer 'tick'

}

func timerFunction()

{

counter -= 1

timerLabel.text = String(counter)

print("Timer is runnning" + String(counter))

if counter == 0

{

timer.invalidate()

timerLabel.text = "Time's up!"

// stop counting down at zero

}

}

7-54 layout of game view (& connection into code)

7-55

9 images added and copy/paste/alter code for each as per steps 1-3 of 6-49 re gestures

NB TODO Instrument: profiling template, Activity monitor, File Activity, Cocoa Layout, Leaks <http://www.spotlessicode.com/blog/posts/instruments-tutorial-part-1-profiling-templates-deferred-mode-launch-instruments>

7-56 & 7-57 - timer as per 6-51, and alert as per 6-53

NB alerts have an UIAlertController, and 1 or more UIAlertActions to handle user response (eg ‘ok’)

NB Swift documentation comments

<https://useyourloaf.com/blog/swift-documentation-quick-guide/>

Separate timer so image hidden in intervals different from 1-second count-down (ie faster movement)

/// hide all kennys but 1

func hideKenny()

{

**for kenny in kennyImageArray**

**{**

**kenny.isHidden = true**

**}**

**let randomNumber = Int(arc4random\_uniform(UInt32(kennyImageArray.count - 1)))**

**kennyImageArray[randomNumber].isHidden = false // un-hide 1 randomly selected image**

**}**

7-58 Extra button added to dialogue (extra action added to Alert Controller) - and handler code written for this action to reset values so that user can ‘replay’

func countDown()

{

counter -= 1

timeLabel.text = "\(counter)"

// hideKenny()

if counter == 0

{

timer.invalidate()

hideTimer.invalidate()

let alert = UIAlertController(title: "Time", message: "Time's up!", preferredStyle: UIAlertControllerStyle.alert)

let ok = UIAlertAction(title: "OK", style: UIAlertActionStyle.default, handler: nil)

alert.addAction(ok)

let replay = UIAlertAction(title: "Replay", style: UIAlertActionStyle.default, **handler: {**

**(UIAlertAction) in**

**self.score = 0 // in handler block so refer back out to ViewController class**

**self.scoreLabel.text = "Score: \(self.score)"**

**self.counter = 30**

**self.timeLabel.text = "\(self.counter)"**

**self.timer = Timer.scheduledTimer(timeInterval: 1, target: self, selector: #selector(ViewController.countDown), userInfo: nil, repeats: true)**

**self.timer = Timer.scheduledTimer(timeInterval: 0.5, target: self, selector: #selector(ViewController.hideKenny), userInfo: nil, repeats: true)**

**}) // NB using handler this time**

alert.addAction(replay)

// handle replay interaction with 'replay' action handler code

// for 'replay' button next to 'ok' button

self.present(alert, animated: true, completion: nil)

}

}

7-59. ‘whack-a-mole’ style game

use UserDefaults to persist high score

in ViewDidLoad

{…..

**// check for any high score set already**

**let highscore = *UserDefaults.standard.object*(forKey: "highscore")**

**if let newScore = highscore as? Int**

**{**

**highscoreValueLabel.text = String(newScore)**

**}**

and in countDown

if counter == 0

{

timer.invalidate()

hideTimer.invalidate()

// checking high scores here

**if self.score > Int(highscoreValueLabel.text!)! // NB '!'!**

**{**

***UserDefaults.standard.set*(self.score, forKey: "highscore")**

**highscoreValueLabel.text = String(self.score)**

**}**

8-61 -> 8-63 git & git hub

9-65 TableView into storyboard/ViewController

(i) stretch to whole corners of view and (ii) (Add new constraints) set all margin contraints to 0

9-66

class ViewController: UIViewController, UITableViewDelegate, UITableViewDataSource {

// NB manually subclass/inherit ViewController also from UITableViewDelegate, UITableViewDataSource

override func viewDidLoad() {

super.viewDidLoad()

// Do any additional setup after loading the view, typically from a nib.

tableView.dataSource = self

tableView.delegate = self

// self so as to have more options when subclassing tableview functions - eg numberOfRowsInSection and cellForRowAt

}

// delegate functions

func tableView(\_ **tableView**: UITableView, **numberOfRowsInSection** section: Int) -> Int {

return 10

}

func **tableView**(\_ tableView: UITableView, **cellForRowAt** indexPath: *IndexPath*) -> UITableViewCell {

let cell = UITableViewCell() // instantiate a cell

cell.textLabel?.text = "Checking cell code"

return cell

}

9-67

var landmarkNamesArray = [String]()

var landmarkImagesArray = [UIImage]()

landmarkNamesArray.append("Great Wall")

landmarkImagesArray.append(UIImage(named:"great-wall.jpeg")!)

&c &c

func tableView(\_ tableView: UITableView, **numberOfRowsInSection section: Int) -> Int {**

**return landmarkNamesArray.count**

**// tell table to have e.g. 6 rows for 6 images and names**

}

func tableView(\_ tableView: UITableView, cellForRowAt indexPath: **IndexPath**) -> UITableViewCell {

let cell = UITableViewCell() // instantiate a cell

// cell.textLabel?.text = "Checking cell code" // check rendering ok

cell.textLabel?.text = **landmarkNamesArray[indexPath.row]**

**// get appropriate content for \*this\* cell** [**https://stackoverflow.com/a/50107220**](https://stackoverflow.com/a/50107220)

return cell // this cell for table row(s)

}

9-68 delete cell/row

NB on iphone, swipe left to reveal ‘delete’ button

// add another delegate, for deletion

func tableView(\_ tableView: UITableView, commit editingStyle: UITableViewCellEditingStyle, forRowAt indexPath: IndexPath) {

if editingStyle == .delete

{

landmarkNamesArray.remove(at: indexPath.row)

landmarkImagesArray.remove(at: indexPath.row)

tableView.deleteRows(at: [indexPath], with: UITableViewRowAnimation.fade)

}

}

9-69 ViewController for images (on swipe right)

as per 6-41 & 6-42 with segues,

steps 1-4

(1) add ViewController and (2) add a new Cocoa Touch Class file (3) (subclass UIViewController as default) of class named e.g. say ImageViewController (4) on storyboard, select new ViewController, in Identity Inspector , set ‘Class’ to new .swift filename from 3 eg ImageViewController (view now connected to code)

add & connect imageView and label

select (1st) ViewController, go to Edit/Embed In…/Navigation Controller (also tab controller available) - navigation pane will provide ‘back’ button between 2nd and 1st ViewControllers

if 1st-to-2nd segue popup choice is ‘Show’, navigation pane will be applied to 2nd ViewController (ImageViewController) also; if ‘Present Modally’ selected, the 1st ViewController’s navigation pane *won’t* be applied to 2nd ViewController.

(1) Highlight left-most button ‘ViewController’ at top of View Controller; (2) Ctrl+drag to 2nd View Controller to segue (3) remember to name segue in Attribute Inspector (utilities, top-right) eg ‘toImageVCSegue’ (be ready to copy this string into *performSegue* and into *prepareForSegue*(below))

go for show (add navigation pane to 2nd viewcontroller)

(seen at top of 2nd ViewController)

9-70 set up segues between selection in 1st ViewController and display in 2nd ViewController

var chosenLandmarkName = ""

var chosenLandmarkImage = UIImage()

// 2 variables corresponding with similar variables in other ViewController

*NB typing in part of many-fold function templates eg tableView eg* ***didSelectRow*** *in XCode will bring up desired parameter combination as auto-complete suggestion (easier than finding from typing in tableView(… and wading through all possible suggested options)*

// add yet another delegate, to pass over data of selection in segue to other ViewController, **if row selected by user**

func tableView(\_ tableView: UITableView, **didSelectRowAt** indexPath: IndexPath) {

self.chosenLandmarkName = landmarkNamesArray[indexPath.row] // get selected row data, ready to pass

self.chosenLandmarkImage = landmarkImagesArray[indexPath.row] // get selected row data, ready to pass

performSegue(withIdentifier: "toImageVCSegue", sender: nil)

}

10-73

2 view controllers with a navigation pane embedded-in, tableview in 2nd viewcontroller - named DetailsViewController

segue and new file

as per 9-69 but also

(1) rename ViewController.swift to ‘tableView.swift’ (persistent click & hold?)& (2) change ViewController class name in new detailsVC.swift to tableViewController and (3) in Identity Inspector, change ‘Class’ from generic ‘ViewController’ to renamed ‘tableViewController’

NB save, close project & reopen project if ’cannot find information about suchaclass’

NB git commit —amend

to add 1 more file to commit

(or git command —amend -m “xyzzzz”

for message)

<https://medium.com/@igor_marques/git-basics-adding-more-changes-to-your-last-commit-1629344cb9a8>

NB vi default git editor <https://www.tutorialspoint.com/unix/unix-vi-editor.htm>

ughhh

:q to quit vi

Ctrl+x Ctrl+c to quit emacs

<https://help.github.com/en/articles/associating-text-editors-with-git>

atom was installed, as it happened, so use atom:

**git config —global core.editor “atom —wait**”

PNJ-dev-Mac-09:Simpsons book peterjenkin$ git tag -a v0.5 -m "my version 0.5 just starting"

git push

git push —tags

git push —follow-tags

git push && git push —tags

git config --global push.followTags true

<https://stackoverflow.com/a/5195913>

<https://stackoverflow.com/questions/3745135/push-git-commits-tags-simultaneously/3745250>

PNJ-dev-Mac-09:Simpsons book peterjenkin$ **git tag**

*v0.5*

PNJ-dev-Mac-09:Simpsons book peterjenkin$ **git push origin v0.5**

*Counting objects: 1, done.*

*Writing objects: 100% (1/1), 177 bytes | 177.00 KiB/s, done.*

*Total 1 (delta 0), reused 0 (delta 0)*

*To github.com:pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass.git*

*\* [new tag] v0.5 -> v0.5*

<https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/tags>

* [**1**](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/watchers)
* **Star**[**0**](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/stargazers)
* **Fork**[**0**](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/network/members)

[pjenkin](https://github.com/pjenkin)/[**iOSSwiftTutorial10SimpsonsBookTableClass**](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass)

[Code](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass)

[Issues **0**](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/issues)

[Pull requests **0**](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/pulls)

[Projects **0**](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/projects)

[Wiki](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/wiki)

[Insights](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/pulse)

[Settings](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/settings)

[**Releases**](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/releases)

[**Tags**](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/tags)

**Tags**

[**v0.5**](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/releases/tag/v0.5)**…**

* 8 minutes ago [c31756d](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/commit/c31756dcbd84dccd3ee588de7922b04e140d0f37) [zip](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/archive/v0.5.zip) [tar.gz](https://github.com/pjenkin/iOSSwiftTutorial10SimpsonsBookTableClass/archive/v0.5.tar.gz)

10-74 drag in images

10-75 new class - Simpson

add new file - swift file - Simpson.swift

//

// Simpson.swift

// Simpsons book

//

// Created by Peter Jenkin on 06/04/2019.

// Copyright © 2019 Peter Jenkin. All rights reserved.

//

import Foundation

import UIKit // for UIImage

/// class to represent a Simpson character

class Simpson

{

var name = ""

var occupation = ""

var image = UIImage()

init() // constructor aka init in Swift

{

name = "default Simpsons name"

}

}

10-76 as per 9-66 & 9-67, set up TableView delegates and datasources (extra subclassing &c)

class tableViewController: UIViewController, UITableViewDelegate, UITableViewDataSource {

to match (in ViewDidLoad)

// table view setup

tableView.delegate = self

tableView.dataSource = self

// adding delegates to display rows and their actions

func tableView(\_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int

{

return simpsons.count

}

func tableView(\_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell

{

let cell = UITableViewCell()

cell.textLabel?.text = simpsons[indexPath.row].name

return cell

}

10-77

NB deleting erroneously connected/named label (age instead of occupation):

select label, go to Connections Inspector, delete entry in Referencing Outlets ( ageLabel )-(x DetailsViewCo.. ) by clicking ‘x’

Label will no longer be named other than ‘Label’ in assets

Variable (no longer connected) can be deleted from code

(check ok, by cmd+B to build)

// in DetailsViewController (2nd)

selectedSimpson = Simpson()

override func viewDidLoad() {

super.viewDidLoad()

nameLabel.text = selectedSimpson.name

occupationLabel.text = selectedSimpson.occupation

imageView.image = selectedSimpson.image

// in tableViewCOntroller (1st)

var simpsons = [Simpson]() // array of Simpson objects

var chosenSimpson = Simpson()

// add another delegate for if row chosen (to segue)

func tableView(\_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {

self.chosenSimpson = simpsons[indexPath.row] // pass over Simpsons object according to row picked by user

// performSegue(withIdentifier: <#T##String#>, sender: <#T##Any?#>)

self.performSegue(withIdentifier: "toDetailsViewController", sender: nil) // segue name string copy/pasted carefully from Attributes Inspector off of main storyboard

// don't bother with sender info - only 1 route

}

// define bespoke prepare for seque function for this app (override)

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {

if segue.identifier == "toDetailsViewController"

{

let destinationVC = segue.destination as! detailsVC

destinationVC.selectedSimpson = self.chosenSimpson

}

}

11-80 Core Data

tick ‘Use Core Data’ in XCode new project

In place of UserDefaults (key/value pairs), Core Data will persist lots of data - serialised via interface into XML, binary or SQLLite

<https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/CoreData/index.html>

<https://www.raywenderlich.com/7569-getting-started-with-core-data-tutorial>

<https://medium.com/xcblog/core-data-with-swift-4-for-beginners-1fc067cca707>

Core data model in file e.g. Artbook.xcdatamodeld

Entities / Fetch Requests / Configurations

‘Add Entity’ button at bottom left

Entities should be named (in plural, e.g. ‘Paintings’ as a db entity containing several records) can have Attributes, like fields or properties

eg Paintings; artist (string) name (string) year(integer), image (Binary Data) …

11-81

drag TableView and Embed into… a navigation pane

* drag Bar Button into navigation pane (right hand end)

Select Bar Button - in Attribute Inspector in ‘System Item’ list, change from ‘Custom’ to ‘Add’ (changes to + symbol)

connect TableView into code as Collect

connect Bar Button into code as Action - addButtonClicked

drag a 2nd ViewController into Main.storyboard, and

Ctrl-drag from icon to make a segue - in Attributes Inspector, change Identifier ‘toCreateVC’ (needed for *performSegue* and similar methods)

File/new/Cocoa Touch Class - ‘createVC.swift’, then

select 2nd VC and in Identity Inspector, change Class to newly appeared ‘createVC’ at end of list.

Drag onto 2nd VC (createVC) 3 Text Fields (Enter Painting Name, Enter Artist, Enter Year of the Painting) and a button (Save)

constraints - to centre. (NB alter constraint to match canvas where stretched to suitable length bigger than text currently in text field)

use < and > in Assistant editor window bar to see previous and next file

select wanted file from path in bar and use manual instead of automatic to help select if needed

11-82

tableView set up with delegates, subclassing as per 9-66 & 10-76

11-83

make up ‘tap here’ png, simple 200x100 - drag into project

in ImageView Attributes Inspector - ‘Image’ to that image

then

imageView.isUserInteractionEnabled = true

and add gesture recogniser as per 6-49

override func viewDidLoad() {

super.viewDidLoad()

imageView.isUserInteractionEnabled = true

// (3)

// as per 6-49 setup & add gesture recogniser for clicking this image

let gestureRecognizer = UITapGestureRecognizer(target: self, action: #selector(createVC.selectImage))

//(1)

imageView.addGestureRecognizer(gestureRecognizer)

// (2)

// NB not imageView.addGestureRecognizer(gestureRecognizer) sans Tap!

// Do any additional setup after loading the view.

}

// (4)

func selectImage()

{

// selecting image from library

let picker = UIImagePickerController()

picker.delegate = self // required ViewController to subclass UIImagePickerControllerDelegate, UINavigationControllerDelegate

picker.sourceType = .photoLibrary

// source could have been from camera; library for demonstration

picker.allowsEditing = true

//present(<#T##viewControllerToPresent: UIViewController##UIViewController#>, animated: <#T##Bool#>, completion: <#T##(() -> Void)?##(() -> Void)?##() -> Void#>)

present(picker, animated: true, completion: nil)

// show ImagePicker, animated, no handler function on completion

}

// type in didfini ... & auto-complete d'pick right code

func imagePickerController(\_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : Any]) {

imageView.image = info[UIImagePickerControllerEditedImage] as? UIImage

// colossal flag name - typed in nearly all before auto-complete

// then UIImage auto-completed to huge name!

// (happening on video too !)

// get (hopefully) Image from picker cast as UIIMage

//self.dismiss(animated: <#T##Bool#>, completion: <#T##(() -> Void)?##(() -> Void)?##() -> Void#>)

self.dismiss(animated: true, completion: nil)

// info is a dictionary

}

then in Info.plist - like manifest? - authorisation ? - new role, so Add Row before first > at bottom (‘+’ button): list starting ‘Application Category’ - from long list of privileges, select ‘Privacy - Photo Library Usage Description’

giving app permission to use the Photo Library as done in the code - in Value, type in a reason for usage eg ’So that we can use a photo to show as art’

11-84 Core Data

Shift-Cmd-H on simulator to get start menu (**H**ome - see simulator menus)

press & hold to save image/copy

**AppDelegate.swift** - code generated

eg NB lazy var persistentContainer: NSPersistentContainer = {

/\*

The persistent container for the application. This implementation

creates and returns a container, having loaded the store for the

application to it. This property is optional since there are legitimate

error conditions that could cause the creation of the store to fail.

\*/

@IBAction func saveButtonClicked(\_ sender: Any) {

let appDelegate = UIApplication.shared.delegate as! AppDelegate // NB AppDelegate.swift

let context = appDelegate.persistentContainer.viewContext // use this context (in this app) for saving data into model

let newArt = NSEntityDescription.insertNewObject(forEntityName: "Paintings", into: context)

// use new entity for CoreData defined in Artbook.xcdatamodeld

// need to import CoreData

newArt.setValue(nameText.text, forKey: "name") // use attributes of above entity, as just now defined in Artbook.xcdatamodeld, coming from control in ViewController

// NB for all TextField attributes 'Correction' and 'Check Spelling', set to 'No' <https://stackoverflow.com/a/42752380>

newArt.setValue(artistText.text, forKey: "artist")

//newArt.setValue(artistText.text, forKey: "style") // not using 'style' field/attribute at mo

if let year = Int(yearText.text) // checking of text cast-able as integer

{

newArt.setValue(year, forKey: "year")

}

//let data = UIImageJPEGRepresentation(<#T##image: UIImage##UIImage#>, <#T##compressionQuality: CGFloat##CGFloat#>)

let data = UIImageJPEGRepresentation(imageView.image!, 0.5)

newArt.setValue(data, forKey: "image") // set binary data for 'image' attribute/field

// now save values

**do**

{

**try** context.save()

print("Saved successfully")

}

**catch**

{

print("Error trying to save")

}

}

// **NB** for all TextField attributes 'Correction' and 'Check Spelling', set to '**No**' <https://stackoverflow.com/a/42752380>

from ‘Default’ using Attribute Inspector else error: *Unable to copy asset information from https://mesu.apple.com/assets/ for asset type*

11-85 CoreData retrieved

*// set up arrays in class:*

var nameArray = [String]()

var yearArray = [Int]()

var artistArray = [String]()

var imageArray = [UIImage]()

*// call retrieveInfo in viewDidLoad:*

@IBOutlet weak var tableView: UITableView!

override func viewDidLoad() {

super.viewDidLoad()

// Do any additional setup after loading the view, typically from a nib.

**tableView.dataSource = self**

**tableView.delegate = self**

// function for retrieving CoreData

retrieveInfo() // populate table with persisted CoreData results

}

/// function for retrieving Core Data records

func retrieveInfo()

{

self.nameArray.removeAll() // clear out previous before fetching new

self.yearArray.removeAll()

self.artistArray.removeAll()

self.imageArray.removeAll()

**let appDelegate = UIApplication.shared.delegate as! AppDelegate**

**let context = appDelegate.persistentContainer.viewContext**

**let fetchRequest = NSFetchRequest<NSFetchRequestResult>(entityName: "Paintings") // need to import CoreData**

// NB template of NSFetchRequestResult, to pass to context to receive CoreData results

fetchRequest.returnsObjectsAsFaults = false // **have to write this**

**do**

{

let results = **try** context.fetch(fetchRequest)

if results.count > 0 // NB results in an array

{

**for result in results as! [NSManagedObject]** // cast as NSManagedObject, to use .value(forKey...

{

if let name = result.value(forKey: "name") as? String

{

self.nameArray.append(name)

}

if let year = result.value(forKey: "year") as? Int

{

self.yearArray.append(year)

}

if let artist = result.value(forKey: "artist") as? String

{

self.artistArray.append(artist)

}

if let imageData = result.value(forKey: "image") as? Data

{

let image = UIImage(data: imageData)

self.imageArray.append(image!) // '!' else error/warning - image should exist cos of if check above

}

self.tableView.reloadData() // refresh the table view - delegate cellForRowAt should display new data

}

}

}

**catch**

{

print("There was an error while retrieving")

}

}

11-86 prepare for segue to pass chosen variable back to first ViewController

add delegates &c and set up

selected/chosen variable correspondence as in 10-72, 10-76, 10-77

11-87 predicates for filters in CoreData model queries

in createVC

override func viewDidLoad() {

super.viewDidLoad()

if chosenPainting != ""

{

print(chosenPainting) // diagnostic

// prep context for another CoreData query

let appDelegate = UIApplication.shared.delegate as! AppDelegate

let context = appDelegate.persistentContainer.viewContext

let fetchRequest = NSFetchRequest<NSFetchRequestResult>(entityName: "Paintings")

// fetchRequest.predicate = NSPredicate(format: <#T##String#>, <#T##args: CVarArg...##CVarArg#>)

fetchRequest**.predicate** = **NSPredicate**(format: "name = %@", self.chosenPainting) // NB predicate for CoreData query

fetchRequest.returnsObjectsAsFaults = false

do

{

// could refactor this into a function

let results = try context.fetch(fetchRequest)

if results.count > 0

{

for result in results as! [NSManagedObject]

{

if let name = result.value(forKey: "name") as? String

{

nameText.text = name // populate textField with retrieved value for name (according to key/value query from other ViewController)

}

if let year = result.value(forKey: "year") as? Int

{

yearText.text = String(year) // populate textField with (cast) retrieved value for name (according to key/value query from other ViewController)

}

if let artist = result.value(forKey: "artist") as? String

{

artistText.text = artist // populate textField with retrieved value for name (according to key/value query from other ViewController)

}

if let imageData = result.value(forKey: "image") as? Data

{

let image = UIImage(data: imageData)

imageView.image = image // populate textField with retrieved value for name (according to key/value query from other ViewController)

// do we need self.imageView.image as in video here??

}

}

}

}

catch

{

print("Error trying to fetch using predicate")

}

}

imageView.isUserInteractionEnabled = true

// as per 6-49 setup & add gesture recogniser for clicking this image

let gestureRecognizer = UITapGestureRecognizer(target: self, action: #selector(createVC.selectImage))

imageView.addGestureRecognizer(gestureRecognizer)

// Do any additional setup after loading the view.

}

11-88 **Notification Center** used to coordinate view actions (using **viewWillLoad**) according to user actions

override func **viewWillAppear**(\_ animated: Bool) {

// look out for "paintingRecorded" from saveButtonClicked in other ViewController

// **NotificationCenter.default.addObserver**(<#T##observer: Any##Any#>, selector: <#T##Selector#>, name: <#T##NSNotification.Name?#>, object: <#T##Any?#>)

// use polymorph with selector, so as to connect this code to another (handler-seque) function: the bespoke rerieveInfo function

NotificationCenter.default.addObserver(self, selector: #selector(ViewController.retrieveInfo), name: NSNotification.Name(rawValue:"paintingRecorded"), object: nil)

// NB viewWillAppear called \*every\* time view shown; viewDidLoad only once, on first showing of view

}

**NB // NB viewWillAppear called \*every\* time view shown; viewDidLoad only once, on first showing of view**

constraints - compression resistance priority (1000 for required). https://stackoverflow.com/a/42324608

NB dashed line in ‘add constraint’ dialogue click line to add constraint, viz separation

NB **git log** to see commits & tags

12-91 Map Kit View into view & code

mapView.delegate = self, & subclass as MKMapViewDelegate

12-92 subclass CLLocationManagerDelegate &

override func viewDidLoad() {

super.viewDidLoad()

// Do any additional setup after loading the view, typically from a nib.

mapView.delegate = self

locationManager.delegate = self

locationManager.desiredAccuracy = kCLLocationAccuracyBest // NB battery drain with accuracy

locationManager.requestWhenInUseAuthorization() // how often to as for permission (cf adding Privacy-Location when in use’ in Info.plist - )

locationManager.startUpdatingLocation() // NB function didUpdateLocation function

}

// function to handle location udate NB parameters

//func locationManager(\_ manager: CLLocationManager, didUpdateLocations locations: [CLLocation]) {

// code

//}

func locationManager(\_ manager: CLLocationManager, didUpdateLocations locations: [CLLocation]) {

//let location = CLLocationCoordinate2D(latitude: <#T##CLLocationDegrees#>, longitude: <#T##CLLocationDegrees#>)

// NB locations is a parameter in this 'handler' for updated location on device

let location = CLLocationCoordinate2D(latitude: locations[0].coordinate.latitude, longitude: locations[0].coordinate.longitude)

let span = MKCoordinateSpan(latitudeDelta: 0.05, longitudeDelta: 0.05) // zoom level (standard 0.05)

let region = MKCoordinateRegion(center: location, span: span)

self.mapView.setRegion(region, animated: true)

}

‘Privacy-Location when in use’ in Info.plist





Simulator - Debug/location/ Custom location - by default, Apple office at Cupertino (old office?)

12-93 Import CoreLocation library

Also need to select project (eg ‘Travel Map’), and from tabs ‘**Build Phases**’ - ‘**Link Binary with Libraries**’ - ‘+’ button, in list find ‘CoreLocation.framework’. ‘Add’ this (framework/library).

(these not added by default to save space) - also

import CoreLocation

12-94 Annotations

Annotations are tags within maps; a different gesture recogniser used (long press)

in viewDidLoad

locationManager.startUpdatingLocation() // NB function didUpdateLocation function

// declare gesture recognition

let recognizer = **UILongPressGestureRecognizer**(target: self, action: #selector(ViewController.chooseLocation(gestureRecognizer:)))

recognizer.minimumPressDuration = 3 // 3 seconds press

mapView.addGestureRecognizer(recognizer)

and

func chooseLocation(gestureRecognizer: UILongPressGestureRecognizer) // bespoke handler function

{

if gestureRecognizer.state == UIGestureRecognizerState.began

{

let touchedPoint = gestureRecognizer.location(in: self.mapView)

// let chosenCoordinates = self.mapView.convert(<#T##point: CGPoint##CGPoint#>, toCoordinateFrom: <#T##UIView?#>)

let chosenCoordinates = self.mapView.convert(touchedPoint, toCoordinateFrom: self.mapView)

let annotation = MKPointAnnotation()

annotation.coordinate = chosenCoordinates

annotation.title = "A new annotation"

annotation.title = "This is the point chosen"

self.mapView.addAnnotation(annotation)

}

}

12-95 title and subtitle of annotation from textfields on VIewController

12-96 CoreData to save the location

12-96 Saving location information to Coredata context

new Entity in TravelMap.xcdatamodelid

as per 11-84

12-97

new ViewController, as firstViewController

(Cocoa Touch .swift & in Main.storyboard)

-> arrow is entry point into application (draggable)

drag firstViewController to left of other VC

segue with identifier “toMapVC”

drag TableView to firstVC and BarButton (System Item ‘Add’) to firstVC nav pane - drag into code

then (as ever) subclass and

tableView.delegate = self

tableView.dataSource = self

}

// delegates added for tableView

func tableView(\_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {

return 5 // 5 rows, say

}

func tableView(\_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {

let cell = UITableViewCell()

cell.textLabel?.text = "checking text"

return cell

}

@IBAction func addBtnClicked(\_ sender: Any) {

performSegue(withIdentifier: "toMapVC", sender: nil)

}

12-98 Retrieve Coredata of location info (as per 11-85)

func fetchData()

{

let appDelegate = UIApplication.shared.delegate as! AppDelegate

let context = appDelegate.persistentContainer.viewContext

// NB autocomplete trying to write AppDelegate not appDelegate

let request = NSFetchRequest<NSFetchRequestResult>(entityName: "Locations")

// cf Entity entry in Travel\_Map.xcdatamodeld

// NB add <NSFetchRequestResult> template specifier manually - ensure autocomplete doesn't pick wrong one!

request.returnsObjectsAsFaults = false

do

{

let results = try context.fetch(request)

if results.count > 0

{

// clear all arrays

self.titleArray.removeAll(keepingCapacity: false)

self.subtitleArray.removeAll(keepingCapacity: false)

self.latitudeArray.removeAll(keepingCapacity: false)

self.longitudeArray.removeAll(keepingCapacity: false)

for result in results as! [NSManagedObject]

{

if let title = result.value(forKey: "title") as? String

{

self.titleArray.append(title)

}

if let subtitle = result.value(forKey: "subtitle") as? String

{

self.subtitleArray.append(subtitle)

}

if let latitude = result.value(forKey: "latitude") as? Double

{

self.latitudeArray.append(latitude)

}

if let longitude = result.value(forKey: "longitude") as? Double

{

self.longitudeArray.append(longitude)

}

print("got here")

self.tableView.reloadData() // refresh table view with retrieved results

}

}

}

catch

{

}

}

// delegates added for tableView

func tableView(\_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {

// return 5 // 5 rows, say

return titleArray.count // however many as are in the titleArray

}

func tableView(\_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {

let cell = UITableViewCell()

//cell.textLabel?.text = "checking text"

cell.textLabel?.text = titleArray[indexPath.row]

return cell

}

// and call fetchData from didViewLoad

12-99 pass data between VIewCotntrollers

corresponding sets of array variables in ViewControllers

in firstViewCOntroller:

// additional delegate for when a tableView row (ie location title) selected

func tableView(\_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {

chosenTitle = titleArray[indexPath.row] // pass over data to other ViewController (ie the map view)

chosenSubtitle = subtitleArray[indexPath.row]

chosenLatitude = Double(latitudeArray[indexPath.row])

chosenLongitude = Double(longitudeArray[indexPath.row])

}

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {

if segue.identifier == "toMapVC"

{

let destinationVC = segue.destination as! ViewController

// pass over/transmit data to the other ViewController

destinationVC.transmittedTitle = self.chosenTitle

destinationVC.transmittedSubtitle = self.chosenSubtitle

destinationVC.transmittedLatitude = self.chosenLatitude

destinationVC.transmittedLongitude = self.chosenLongitude

}

}

12-101

// customising annotation, to show new kind of non-standard button bespoke for this app

func mapView(\_ mapView: MKMapView, viewFor annotation: MKAnnotation) -> MKAnnotationView? {

if annotation is MKUserLocation {

return nil

}

let reuseId = "myAnnotation"

var pinView = mapView.dequeueReusableAnnotationView(withIdentifier: reuseId) as? MKPinAnnotationView

// reuse just the 1 annotation view (good for memory resources)

if pinView == nil

{

pinView = MKPinAnnotationView(annotation: annotation, reuseIdentifier: reuseId)

pinView?.canShowCallout = true

pinView?.pinTintColor = UIColor.red

let button = UIButton(type: .detailDisclosure) // .detailDisclosure == UIButton.detailDisclosure

pinView?.rightCalloutAccessoryView = button

}

else // if pinView already existent (already defined, and reused)

{

pinView?.annotation = annotation

}

return pinView

}

12-102 handler function for mapView Map View so that, if the map is not being used to display current location (ie is used in browsing other locations instead) then alter to show a bespoke annotation which, when clicked, will show driving directions

// 1 of 2 - customising annotation, to show new kind of non-standard button bespoke for this app

func mapView(\_ mapView: MKMapView, viewFor annotation: MKAnnotation) -> MKAnnotationView? {

//

if annotation is MKUserLocation {

return nil

} // don't do anything if the map is opening for the purpose only of showing the current user location (because directions to current location would be pointless!)

let reuseId = "myAnnotation"

var pinView = mapView.dequeueReusableAnnotationView(withIdentifier: reuseId) as? MKPinAnnotationView

// reuse just the 1 annotation view (good for memory resources)

if pinView == nil

{

pinView = MKPinAnnotationView(annotation: annotation, reuseIdentifier: reuseId)

pinView?.canShowCallout = true

pinView?.pinTintColor = UIColor.cyan

let button = UIButton(type: .detailDisclosure) // .detailDisclosure == UIButton.detailDisclosure - detailDisclosure is 'i' in circle

pinView?.rightCalloutAccessoryView = button // on right hand side (could've been on left)

}

else // if pinView already existent (already defined, and reused)

{

pinView?.annotation = annotation

}

return pinView

}

// 2 of 2 - for custom annotation, to pull up driving directions from current location to that location

//func mapView(\_ mapView: MKMapView, annotationView view: MKAnnotationView, calloutAccessoryControlTapped control: UIControl) {

//<#code#>

//} // had to auto-complete by calloutAccess.....

func mapView(\_ mapView: MKMapView, annotationView view: MKAnnotationView, calloutAccessoryControlTapped control: UIControl) {

if transmittedLatitude != 0 && transmittedLongitude != 0

{

self.requestCLLocation = CLLocation(latitude: transmittedLatitude, longitude: transmittedLongitude) // to use in reverse geolocation below

}

//CLGeocoder().reverseGeocodeLocation(<#T##location: CLLocation##CLLocation#>, completionHandler: <#T##CLGeocodeCompletionHandler##CLGeocodeCompletionHandler##([CLPlacemark]?, Error?) -> Void#>)

CLGeocoder().reverseGeocodeLocation(requestCLLocation)

{

(placemarks, error) in

if let placemark = placemarks

{

if placemark.count > 0 // if successfully reverse geo-coded

{

let newPlaceMark = MKPlacemark(placemark: placemark[0])

let item = MKMapItem(placemark: newPlaceMark)

item.name = self.transmittedTitle

// launch a screen showing travel directions to/fro

let launchOptions = [MKLaunchOptionsDirectionsModeKey : MKLaunchOptionsDirectionsModeDriving]

item.openInMaps(launchOptions: launchOptions)

}

}

}

}