Richard Telford

rt113@duke.edu

Fall 2018

# ECE 564 CLASS HANDBOOK

**v2 (minor updates – Niral's hours and Walker's email id)**
**v3 (minor update – homework due date)**
**v4 (minor update – my office number)**
**v5 (update to team formation hw description)**
**v6 (update to PJ2 and HW5)**
**v7 (updated TOC)**
**v8 (added HW6)**
**v9 (updated code due date)**

# Table of Contents


**ECE 564 CLASS HANDBOOK ..... 0**

**INTRODUCTION ..... 3**

About Me ..... 3

About the TAs ..... 3

Hardware and Software Requirements ..... 4

*iPod / iPhone Distribution ..... 4*

*Software ..... 4*

Text Books (optional) ..... 5

Duke Community Pledge ..... 5

**THE COURSE ..... 5**

Grading ..... 5

*Point Distribution ..... 5*

*Grade Scale ..... 6*

Course Schedule ..... 6

Attendance and Participation ..... 7

**TOOLS ..... 7**

Sakai ..... 7

Git ..... 8

Piazza ..... 8

**HOMEWORK DESCRIPTIONS ..... 8**

Overview ..... 8

HW1 – Storyboard ..... 9

*Description ..... 9*

*Using Git ..... 9*

*Rubric for HW1 ..... 10*

HW2 – View ..... 10

*Description ..... 10*

*Rubric for HW2 ..... 11*

HW3 – Table View Controller ..... 11

*Description ..... 11*

*Rubric for HW3 ..... 13*

HW4 – Drawing and Animation ..... 13

*Description ..... 13*

*Rubric for HW4 ..... 14*

HW5 – Storage ..... 14

*Description ..... 14*

*Rubric ..... 14*

HW6 – Consolidation ..... 15

*Description ..... 15*

*Rubric ..... 16*

**SEMESTER PROJECT ..... 16**

PJ1 – Team Formation / Project Choice ..... 17

- Up to 10 points for the concept itself. If you choose one of the projects off my list (Project List), you receive 10 points automatically. If you come up with your own idea, I will assign a grade based on creativity, difficulty and originality. ..... 17
- Up to 15 points for the actual submission in Sakai – followed directions, on time, complete, readable, clear, concise, interesting, etc. ..... 17

PJ2 – Design Presentation ..... 17

PJ3 – Sponsor Feedback ..... 17

PJ4 – PROJECT PRESENTATION ..... 17
PJ5 – FINAL CODE REVIEW ..... 17

## Introduction

Welcome to ECE 564! By the end of this class you will be an employable app development programmer, assuming you pay attention in class and spend quality time on each of your assignments and project. As always, you will get out of the class what you put into it – my best students have gone on to land jobs in places like Facebook, but they were ones that really focused on becoming a good Mobile Developer.

You can start by reading through this document thoroughly so there is no confusion later. Many of the poorer grades suffered by students were due to simply **not paying attention** to instructions or listening in class. It is also due to students not allocating enough time during the week to read the material and do the homework. I suggest you plan on blocking **8 hours per week minimum** on your calendar now for the first few weeks until you get a feel for how long it takes you to do each homework.

### About Me

My name is Ric Telford (ric.telford@duke.edu) and I am an Executive-in-Residence and Adjunct Associate Professor. I am also the Director, DUhatch Business Incubator and Director, Duke Center for Mobile Development. You can call me Ric.

I received my BS, Computer Science from Trinity University, San Antonio, TX and retired from IBM as Vice President, Cloud Strategy in 2015. Most of my career spent in new technology development, building businesses. I am also the Founder of *Telford Ventures* and co-founder of *Pneumonics, Inc*, a medical device startup.

I live in Morrisville, NC, married with 3 sons, 2 dogs, 3 cats and 3 horses. I enjoy Golf, Swimming and Biking. I know Swift, C and C++ among other lesser-known languages. Most importantly:

- **Office Hours (FITZ 3405)**:
  - MONDAY 9am to 11am and 2pm to 5pm
  - WEDNESDAY 9am to 11am and 1pm to 3pm

### About the TAs

- Walker Eacho
  - Final semester of ECE / CS dual major, Duke University
  - From Chevy Chase, Maryland
  - Likes sailing, climbing and baking
  - Knows Swift, Objective-C, Javascript and C
  - **Office Hours (subject to change – check Piazza for latest)**:
    - TUESDAY 1pm to 3pm
    - FRIDAY 1pm to 3pm
- Niral Shah
  - BS from Rutgers in ECE / CS
  - Currently working on Master of Engineering degree in Computer Engineering
  - From Central New Jersey
  - Likes Computer Vision projects, Tennis and Traveling
  - Knows Swift, Python, Java and C
  - **Office Hours** (subject to change - check Piazza for latest):
    - MONDAY 1pm to 3pm
    - WEDNESDAY 3pm to 5pm

## Hardware and Software Requirements

This class requires an Apple hardware / software development environment which means you will need access to a Mac computer. If you do not have a Mac computer, you have 2 options – use the ones that are on campus (like the ones outside of Teer 106) or borrow a laptop from me. Please let me know if you will be needing a laptop.

## iPod / iPhone Distribution

By week 5 we will be distributing iPod Touches or basic iPhones to everyone ("the device"). This will allow you to test and run your homework and project on a real device. You are welcome to test on your own iPhone as well, but all homework and projects will be graded on how they look/work on the iPhone 6 in the simulator and the device given you. Some things to know when you receive the device:

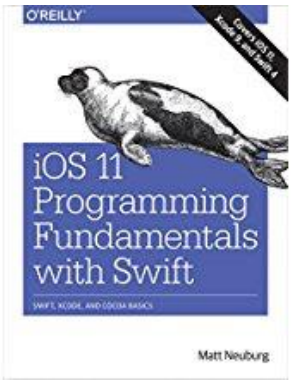
- If it has a password, it is **1234**. Please don't change.
- If you associate an iCloud ID with the device for security reasons, you **MUST** make sure to delete the ID or totally reset the device before returning.
- You **MUST** return the cord with the device.
- I will let you know if you need to bring the device to class.
- See the TAs during office hours if you have any problems with the device given you.
- You will sign a form when you receive the device, which states:
    - *By signing, I understand that I am receiving Duke property for my use during the semester. I am responsible for the safekeeping of this hardware and I am responsible for returning it at the end of the semester. I agree to assume full financial responsibility for all components of the equipment for the time that it is checked out to me. I understand that I will be asked to pay a replacement charge for any damage or loss of the equipment issued to me. I understand that I am responsible for returning the equipment in person, with all parts, on time. I will not leave equipment unattended and understand that I must return the equipment during the time designated by OIT or the professor. I agree to wipe the iPod before returning, especially REMOVING IT FROM MY ICLOUD ACCOUNT and FIND MY IPHONE PAGE*

## Software

- Everyone will need to upgrade their systems to the following:
    - macOS High Sierra
    - Xcode 9.4.1 or later
- If you are assigned an iPhone, make sure you upgrade your iPhone to:
    - iOS 11.4.1
- If you have an iPod Touch, then update it to the last version possible.
- We will be using:
    - Swift 4.2
- You will be getting an email to sign up for an Apple Developer ID. Use your **SHORT NETID** Duke email for this (e.g. rt113@duke.edu)
- With the Developer ID you can log into developer.apple.com to get access to all the information they provide for developers.
- There is a major set of upgrades coming this Fall**. Do NOT upgrade until I tell you** – probably over Fall Break. **Make sure Xcode is NOT on automatic updates**. If you go to a later version of iOS, Swift or Xcode and your homework does not compile on my machine, **you will lose points.**

### Text Books (optional)

You should download the Swift Programming Language 4.0 book from iBooks. It is free. I also strongly encourage you to get the text books – **iOS 11 Programming Fundamentals with Swift** (Matt neuberg, O'Reilly Media) and **Programming iOS 11** (Matt Neuberg, O'Reilly Media). There will be reading assignments for many of the classes. It is a good practice to review / lightly read before class which will make the lecture that much easier to understand. Then, to use the book as reference when doing the assigned homework. If you get the Kindle versions of these books, it is about ½ price.





### Duke Community Pledge

- I will not lie, cheat, or steal in my academic endeavors, nor will I accept the actions of those who do.
- I will conduct myself responsibly and honorably in all my activities as a Duke student.
- I will act if the Standard is compromised.

Please honor the pledge. Copying someone else's work / code is considered cheating and there have been incidents of expulsion because of it. Any reuse of open source code **MUST be attributed with the author's name and any relevant license information**. Keep your eyes on your own paper during tests and do original work on your assignments. I will always let you know when something is an individual vs team assignment.

## The Course

The typical class format is to start with a recap of the previous day, review homework, ask questions and discuss relevant topics (upcoming Midterm, project dates, etc.). Most days will be lecture on new topics, but some will have a guest speaker, presentations by project teams, presentation by a TA, or a hands-on exercise. Each class will end with a look forward to key deliverables and dates. Below is a draft outline of each day's content. Any of this is subject to change.

### Grading

You can earn up to 1000 points during the semester. To know your grade, just move the decimal point over. For example, if you earned 913 points, you will get a 91.3, an A-. There will be **NO** addition of points at the end to "round up" your grade – **so please do not ask!** There will be an opportunity to earn up to 5 points extra credit, which fulfills the same goal as rounding up.

### Point Distribution

- 50 - Attendance / Participation. If you attend all the classes and participate in all requests of mine (surveys / polls on Piazza, questions in emails, etc) you will receive 45 points, an A- .

Higher grades are reserved for those that strongly engage/participate in class and/or answer fellow student questions on Piazza (or proactively post good information to Piazza). I do not always take attendance, but if I do and you are not there, it is a deduction of **5 points,** unless I have given you **written** permission to miss class (illness, job interviews, etc) ahead of time. I may also deduct for tardiness or not following instructions.

- 200 – Midterm. The "midterm" will be given around the 8th week and cover Swift, Git, Xcode and some amount of iOS programming. The goal of the midterm is to ensure you are really understanding the concepts of mobile development. It will be around 50 multiple choice / fill in the blank questions.
- 350 – Homework. There will be 7 homework assignments in the first 7 weeks of class worth 50 points each. Detailed descriptions of each follows in this document.
- 400 – Project. The project deliverables occur throughout the semester and culminate in the grading of the final code review (details later in this document):
    - PJ1 – Team Formation / Project Choice - 25
    - PJ2 – Design Presentation – 25
    - PJ3 – Sponsor Feedback – 50
    - PJ4 – Final Presentation – 50
    - PJ5 – Final Code Review – 250

## Grade Scale

- A+ 100 - 97
- A 96 - 93
- A- 92 - 90
- B+ 89 - 87
- B 86 - 83
- B- 82 - 80

## Course Schedule

Below is a draft outline of each day's content. Any of this is subject to change.

- Aug 27 – Intro / Git / Swift
- Aug 29 – Swift (HW1 – Playground)

- Sept 3 – Xcode, Delegation, Protocols, Xtension
- Sept 5 – Single VC (HW2 – View and PJ1)

- Sept 10 – Views, Storyboard
- Sept 12 – View Controllers (HW3 – Table)

- Sept 17 – More Views and VCs
- Sept 19 – Graphics (HW4 – Graphics)

- Sept 24 – Data and Storage
- Sept 26 – Data and Storage (HW5 – Storage and PJ2)

- Oct 1 – Communications / CocoaPods
- Oct 3 – Communications (HW6 – Communications)

- Oct 8 – FALL BREAK
- Oct 10 – Design Presentations (HW7 – Team)

- Oct 15 – Additional Features – Try/Guard, etc & Review
- Oct 17 – Midterm

- Oct 22– Debugging, MARK, TODO, etc
- Oct 24 – Frameworks (PJ 3 – Sponsor Feedback)

- Oct 29 – ARKit
- Oct 31 – In-Class Exercise

- Nov 5 – Additional Frameworks – Mail, Notifications, etc.
- Nov 7 – Queues, Threads, Dispatch

- Nov 12 – Advanced Animation
- Nov 14 – Well Structured Code (PJ4 – Presentation)

- Nov 19 – Sensors
- Nov 21 – Test Framework

- Nov 26 – Bits, Photos, Advanced Graphics
- Nov 28 – Presentations (grad)
- Dec 3 – Presentations (undergrad)

### Attendance and Participation

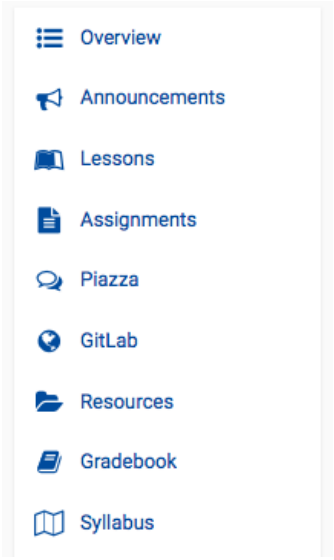
I expect you to attend every class. If you can't make a class, I need you to let me know ahead of time via email (rt113@duke.edu) or text (919-418-0934). I also encourage participation. Feel free to ask questions at any time – some lectures can get very boring without questions to break them up! Also, you can participate on Piazza – students who answer other student's questions on Piazza get points toward their Attendance grade.

## Tools

The class is run on a combination of Sakai, Git, and Piazza. We will discuss Git in class during Weeks 1 and 2, and there are documents in Sakai Resources with detailed Git information. You are expected to know Sakai and Piazza, so if you have not used these tools before please get familiar with them.

### Sakai

Sakai will be the "hub" of this class. From there, you can see your Assignments, get access to Resources, see your grades, launch into Piazza and launch into Git. Those will be the major functions (as you can see in the toolbar below for our Sakai site), but I also use Announcements from time to time.




### Git

All Homework and Project code submissions will be done using GitLab, the Duke internal Git repository. If you have not registered to gitlab.oit.duke.edu, you need to. If you are not familiar with Git, you need to be. I will give an overview in class and there is a document on Resources.

### Piazza

We will use Piazza for technical help information, primarily Q&A. If you have not already, use the link I sent to add yourself to Piazza. Please pay attention to postings – sometimes I may put a poll out there or send info. Helping others on Piazza counts towards Participation grade, so it is best to NOT post anonymously. We will also post Office Hours on Piazza.

## Homework Descriptions

### Overview

There will be 7 homework assignments, assigned each week for the first 7 weeks. The first 6 are individual homework assignments and the last one will be a Team assignment. The first 4 homework are described below and are already open in Sakai, for those that want to work ahead. This document will be updated later in the semester to cover the final 3 homework. By default, homework will be due at **2pm on the Wednesday following the week it was assigned, but always check Sakai "Due Date" to get the correct Due Date and Time.** In Sakai, there is also an "Accept Until" date. Anything submitted after the "Due Date" but before the "Accept Until" date will be accepted but with -3 points for being late. Anything after the "Accept Until" date will be graded as a **0** unless I give you a **written** extension (email me and I will respond).

- There will be 7 homework assignments each worth 50 points, the final being a Team homework.
- The "base grade" for doing homework is 45. That is, if you do everything you were asked and nothing more, you get an A-
- If you do less than what was asked, or if the code has errors or doesn't run/compile/etc, points come off
- If you do more than what was asked, points added on to a max of 50
- Example:
  - Jane turns in her code and it had a warning. Jane's grade is lowered 1 points to a 44.
  - But, Jane also added a novel slide control, nice layout and fonts, good use of color, and an additional function. Jane's grade is increased 7 points so she ends up maxed out at 50.
- Each Homework will have a Rubric with the pluses and minuses (see below)

# HW1 – Storyboard

## Description

The purpose of Homework 1 (HW1) is to get comfortable with writing pure Swift code. We won't be doing any iOS programming and we won't be doing any compiling in this homework – the code will be written in the Xcode "Playground" – an interactive coding environment that runs the code real time. The final instructions for the assignment are in the header of the playground itself, which you will download from GitLab (see "Using GitLab" below). Please refer to the playground file for final instructions, but here is an overview:

- In this first assignment, you are going to create a base data model for the Students, TAs and Professors in this class, and then can search an array of those objects by writing a `whoIs` function.
- I suggest you create a new class called `DukePerson` and have it subclass `Person`. You also need to abide by 2 protocols: BlueDevil and CustomStringConvertible
- I also suggest you try to follow good OO practices by containing any properties and methods that deal with `DukePerson` within the class definition.
- In addition to the properties required by `Person`, `CustomStringConvertible` and `BlueDevil`, you need to have include the following information about each person:
    - Their degree, if applicable
    - Up to 3 of their best programming languages as an array of `String` (like `hobbies` that is in the protocol)
- I suggest you create an array of `DukePerson` objects, and you must have at least 4 entries in your array for me to test:
    1. Yourself
    2. Me (my info is in this document!)
    3. The TAs (also in this document)
- Your program must contain the following:
    - You must include 4 of the following - array, dictionary, set, class, function, closure expression, enum, struct
    - You must include 4 different types, such as string, character, int, double, bool, float
    - You must include 4 different control flows, such as for/in, while, repeat/while, if/else, switch/case
    - You must include 4 different operators from any of the various categories of assignment, arithmetic, comparison, nil coalescing, range, logical

## Using Git

- Download **ECE564_HOMEWORK** from the **ALL** Subgroup of the **ECE564** Group (NOTE: This Subgroup comes up by default from our Sakai site).
- Add your code to this project in the HW1.playground. Do not add anything to the other files in the project.
- Create a New Project on your GitLab ID (not in any Group) named **ECE564_HOMEWORK** (all upper case)
- You will continue to use this same project for the first 6 homework, just keep adding to it using Branches in Git and then Merging before the homework is due.
- Once we switch to the Team homework (HW 7) we will use Groups
- Click on **Settings**, then **Members**, add "rt113", "dwe8" and "ns247" as "Reporter" status
- Push your project to this GitLab repo before the due date.
- NOTE: If you are doing any additional functions that add points to your grade (see list below), you **MUST** list what you did in a **README.md** file that you create for your project. You will continue to add to this **README.md** for every homework assignment.

## Rubric for HW1

Here are the types of things that would cause grade deductions:

- Turning it in late
- Crashes
- Not meeting the functional requirements of the assignment
- Lack of simple error checking (like giving a message if person not found)
- Not using the required amount of Swift capabilities

Here are the types of things that can add points to grade:

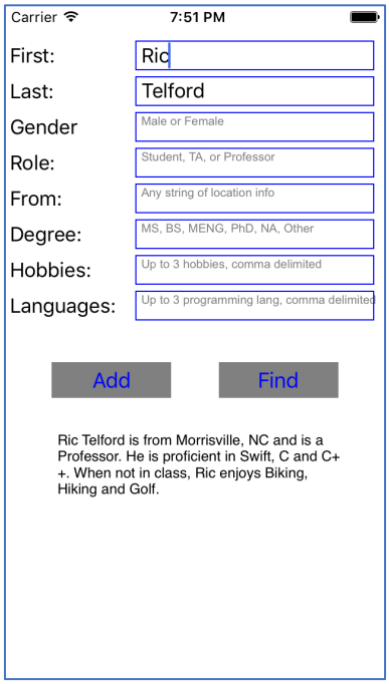
- Compact code, well-structured
- Broader use of Swift functions (more than 4 per category)
- Ability to pass back several results from whoIs using blanks / wildcard (*)
- Additional functions (in addition to whoIs) to search other fields (like degree, from, etc)
- Any other creative additions

## HW2 – View

### Description

The purpose of HW2 is to get your first exposure of using Xcode to create an iOS app. You will be creating an app for the "iPhone 6" Simulator. You will be using the same **ECE564_HOMEWORK** project as in HW1 as your base, just moving the code to the template files provided. You will then create a single view application to enter data, save it, search it, and display it. The details:

1. Start from where you left off on the HW1 project. If you start from the same local repo, make sure you are in sync with the GitLab repo. You can optionally create a Branch and do the work in the Branch before merging. The third option is just to do an Xcode Source Control->Check Out and start a new local repo from a different folder.
2. Copy and Paste your code from HW1 playground into the proper files (Data Model, View Controller, etc). This will form the basis of your new app.
3. The app will be one screen – we will call it the **Information** view - that accepts input for the following fields: First Name, Last Name, Gender, Role, From, Degree, Hobbies, Programing Languages.
4. Somewhere on the screen, have a Label field which displays the output string produced by the *description* property in HW1.
5. Have two buttons on the screen – one for Adding the info to your data model and one for Finding the Information on that person given First and Last name (similar to whoIs )
6. When the buttons are pressed, extract the data from the input fields and handle any potential problems in the input (look out for nils, etc). You don't have to do full validity checking – that would be extra (see below). You can use the Label field as needed, though, to put error messages when there is a problem.
7. Do NOT use a Storyboard or. xib files – this should all be done in code.




## Rubric for HW2

All the types of problems listed in HW1 Rubric apply as well to HW2. In addition, you need to avoid the following:

- Compiler warning messages. If you don't know how to fix a warning message, please ask on Piazza or ask the TA or me.
- Data entry that causes a crash. You don't need to do extensive data entry error checking, but you need to make sure any data that is entered does not crash the app.
- Text field being behind the keyboard and preventing input. We won't be grading this on the device, but it is good to dismiss the keyboard when the user clicks outside of a text field.

Below are the ways to add additional points to your grade (NOTE: remember to put the additional work in your README.md):

- Use controls other than Text Field and Label. For example, a Picker control for Gender or Degree.
- Extensive Error Checking
- Nicer looking layout
- Use of color, style
- Add an ImageView for a picture. Since we are not testing on the device yet you should not put in any camera code (it will crash in the Simulator) but you can show an image or avatar for each person.
- Make the "Add" button an "Add/Update" button. If the name already exists, it will update that person's information.
- Clever ways to make the code less verbose (tighter code). This homework can have a lot of repetitive code but there are ways to make it more concise.

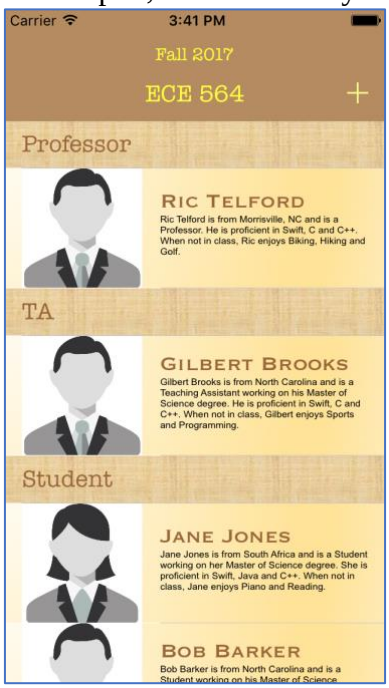
## HW3 – Table View Controller

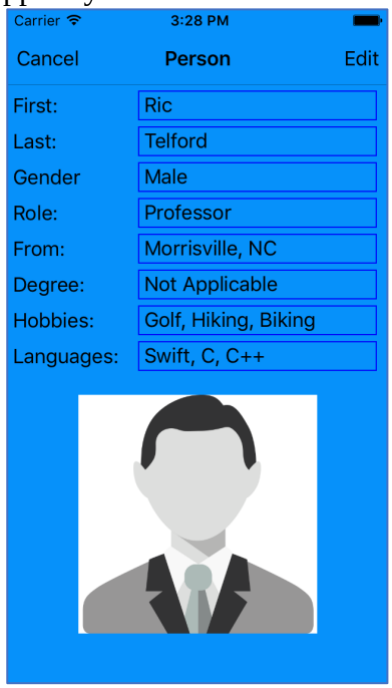
### Description

The goal of HW3 is to create a multi-view app that uses a Table View Controller on the first view we will call the **Table** view. HW3 also introduces the concept of a storyboard – using the Xcode Interface Builder. The Storyboard will just lay out flow using Navigation Controllers, Segues, etc. You do not

need to rewrite your HW2 View Controller for the storyboard, but you are welcome to for additional points. The details:

- Implement a Table View Controller as the first screen of your app.
- Pre-populate the Table with hardcoded "Person"s. At a minimum, have yourself, your teammates, the Professor and the TA(s).
- On the **Table** View, have an "Add" or "+" button on the Navigation Bar.
- When "Add" is pressed on the **Table** View, present your screen to add a new "Person" to the class.
- Have two cell types – one for displaying the "Person" and one to serve as a "Separator". Different the two cell types clearly – by height, background color, style, etc. You choose. The "Separator" cell type will be used for your Team (display your Team Name), the TAs (the separator should say "TAs") and the Professor (the separator should say "Professor"). The "Person" cell type will be used, of course, to display the different people. For now, you only need display the name of the person and the Person.description information in the cell.
- HINT: You need to think of a way to index your array of Persons so that you can easily map from the rows in the Table to the rows in the Person array. This may require another data structure.
- When a "Person" cell is selected, show the View from HW2 (**Information** View), with full information about that "Person but **without** the "Add/Update" button, the "Find" button and the "Person.description" Label (see below for what to do with these).
- In that space on the bottom of the screen, you will put an ImageView instead, for showing the Person's picture. You don't have to show actual pictures in this homework, but can use an avatar for now. As mentioned above, you can use your HW2 code as the base here, or rewrite the View using storyboard objects.
- The **Information** View needs to appear in "view mode" only – that is, the data cannot be edited.
- Make the rightBarButtonItem be "Edit". When pressed, the fields become Editable and the Button changes to "Save". When "Save" is pressed, the information about the Person described in the "First Name" and "Last Name" fields are updated accordingly.
- The functions you removed do not go away – they are moved as follows:
    - The "Person.description" information is moved to the Table View cell
    - The "Find" function will move to a Search bar on the Table View in a later homework.
    - The "Add/Update" function will move to a "Save" function as described above.
- Compile, Run and Test your app on your iPod Touch. This is the device we will be testing on.



### Rubric for HW3

All the issues defined in HW1 and HW2 that are applicable to HW3 are ways to lose points. In addition, look out for the following:

- Circular segues. You will want to use an "unwind" segue to get back to the table view. HINT: You can give unique Identifiers to different unwind segues.
- Poorly formatted text and images – make sure you maintain the aspect of any pictures.
- Missing files, especially pictures, that don't get copied up to GitLab. Sometimes when you add files to a project it creates a link but does not copy the file into the folder.
- Data model problems – index gets off when you Add so wrong data is displayed, fields don't get properly initialized, etc.
- Information View allows editing when it should be in "view mode" and vice versa
- Information gets added when it should have been updated, leaving 2 cells for same Person
- Save / Cancel / Edit buttons don't work as expected

Below are the ways to add additional points to your grade:

- Add a "Team" field to the Person information screen and to your data model. This will be required later but if you want to do it now you can get additional points. Team only applies to Students (not TAs or Professor) and you will need to search this field for displaying the teams on the Table View properly.
- Add Picture support. That is, when you Add a new Person, have a button to take their picture. Save the picture and display on both the Table View and the Information View.
- Creativeness in the design of your Table View and Cells. For example, add pictures or multiple text fields.
- Additional functions of your own creation
- Clever code – a unique approach or algorithm

## HW4 – Drawing and Animation

### Description

The purpose of HW4 is to have hands-on experience writing to some of the lower-level View and Layer functions – Drawing and Animation. You use these capabilities when you want to fill a View with your own content – not just use the pre-made Views like a Table View or a Text View. You will create the "flip side" of the **Information** View about yourself. Much like a trading card, one side has your information, the other is a picture, but in this case the picture will be an animated scene of one of your Hobbies. Here are the specifics:

- Start from your last homework (HW3) code base as described earlier
- On the **Information** view, add a button or some other way to "Flip" to your new "Drawing" page/View.
- When the "Flip" is done, the View will be replaced by a screen-sized Drawing you created. Make sure you have a way to "Flip" back!
- The Drawing **MUST** reflect something about one of the 3 hobbies you listed on your Information View. Not following this will lead to loss of points. For example, if "Soccer" was one of your Hobbies, then animating a soccer ball going through a goal would be perfect.
- Finally, the Drawing **MUST** contain the following elements:
    - A UIImage
    - A Graphic Context with **VECTOR** graphic drawing orders (like a Bezier curve, Rect, Ellipse, Path, etc.), NOT just an image.
    - Animation – using any of the various animation techniques, make something move in the view.

### Rubric for HW4

Once again, anything mentioned in earlier homework that is applicable here can cause loss of points. In addition:

- Make sure you have a way to get back to Information view from Drawing view
- Don't use the same View Controller to remove the Information screen then add the Drawing screen. You need to launch into a different View Controller
- Make sure you test on the iPod Touch – that is what we will be grading on.
- You need to ensure the drawing is related to one of the 3 hobbies you chose
- Make sure you have a Drawing that is not overly simple - just drawing a box or a circle won't cut it!
- Don't just copy something you find on the Internet. You can look at examples in tutorials, etc to get ideas but do your own Drawing in your own code.

Below are the ways to add additional points to your grade:

- Use a Page Controller instead of a button to transition from the Information View to the Drawing View.
- Creativity / Complexity in the drawings inside the graphic context. This is the best way to get full credit – put some effort into the actual drawing itself.
- Add media – Audio, Video, Attributed Text are all good additions.

## HW5 – Storage

### Description

The main purpose of HW5 is to persist your student information to storage so your list of people information goes from session to session. In addition, you will need to add a few new functions to make your app really usable (assuming you have not already implemented). Here is the list of functions to add in HW5:

- As mentioned above, you need to be able to save and load your array of DukePersons so that it persists from session to session. There are several methods to persist all fields to disk. You can use whatever method you wish (even invent your own), but as discussed in class the assumed approach is to JSONEncoder method or NSKeyArchiver.
- Team support – In the Edit mode of the Information View, be able to prompt for Team name for the Person. It is an optional field. On the Table View, sort students under their Team Name cell, which is a separator cell that takes the place of "Student"
- On the Table View, enable the "Swipe left to Delete" function. If the user swipes left and selects Delete, remove the entry from the table.

### Rubric

In addition to the types of things mentioned in previous HW rubrics, make sure:

- You test from both a new install and an existing install. Apps have been known to crash after changes are made but a "fresh install" is not re-tested.
- All fields from the Information view save correctly
- All fields on the information get loaded correctly after saving.

Ways to earn points towards full credit:

- Search support in the Table View. Enable the search bar, then filter what is shown in the Table View based on what is typed in the Search bar. You can decide if they can only search on Name Field or do a General search, etc. The more complex the search options the better.-
- Create and store additional information (like job experience) about each person.
- Implement a different way to save/retrieve other than using JSONEncoder or NSKeyArchiver.

# HW6 – Consolidation

## Description

The purpose of Homework 6 is to get you used to developing in a team environment and using GitLab accordingly. This will prep you for doing your team project. It also is our first attempt at using iOS 12 and Xcode, so go ahead and make the switch over Fall Break!

For Homework 6, you will create 1 clean version of code for your team and use the new Group / Project in GitLab. There is already a group in GitLab with your team name and project HW6. Everything you have done so far in homework needs to carry forward into this consolidated code base. The minimum function set is defined below. It needs to be clean, well-documented and error/warning free. Here is a checklist you can follow:

- **Camera / picture support** - In the Edit mode of the Information View, be able to take a picture and include it in the information about the Person. If you need a device to test this on, let me know. Show the picture on the Table View as well.
- **Communications support** – More info to follow on this, but you will be using HTTP / REST to send your student information to a web server in the ECE 564 JSON Grammar.
- Ability to **Search, Add, Delete and Edit** from the Table Screen.
- All features / functions from HW1 – HW5.
- A clean project in Xcode 10. Project Name, bundle name, etc. should all be "*teamname*HW" where teammate is your team name.
- Clean, well-structured code with no compiler errors or warnings
- Comments in the code. Use "// MARK:" to show the major sections for any long files.
- Nice App Icon
- Nice looking Table Cells with things like background color, image, subtext line, pictures, fonts, etc.
- Ability to add both new teams and new students
- When a student is added, make sure there is a prompt for a team name. Every student is assigned a team, and only 1 team. If they don't type team name, you can just put in "Student"
- Ability to Save all info, including the Picture.
- Easy to navigate and use fields on the Edit screen, including good management of dismissing the Keyboard appropriately.
- Looks good / works on an iPhone SE. As mentioned above, I have some of these you can test with.
- Consistent and sensible way to go from Information View to that person's Graphic/Animation page. That is, use the same method to "flip" from the Information View to the Animation View. All team member's Graphic Screens need to be able to "flip" from their Information View Screen.

Suggested steps, but you can do what you want:

1. Decide which of your individual project code bases will serve as the majority base code for HW6.
2. Create a clean, new project in XCode 10 and call it *teamname*HW. **FOR EXAMPLE: MoveITHW6 or VeniceHW6 or HHCHW6.**
3. Create **new** files and copy/paste code into new files. I don't suggest trying to move existing files – sometimes that causes issues in Xcode.
4. Recreate the storyboard, .xibs, etc.
5. Push it to your ECE564/teamname/HW6 project in Gitlab
6. Have a strategy of how you will divide up any additional work.
    a. Everyone should have a role and everyone should do Pushes to GitLab.

    b. I will be looking at Activity trail to ensure everyone participates. This is a team grade only, so everyone loses points if there is not team participation.
    c. Some of the work that needs to be done includes moving Animation Views into new base, adding comments, cleaning up / restructuring code, etc.
    d. If there are files that multiple people need to touch, you should consider using GitLab branches and later Merges
7. Have schedule for when everyone needs to have code submitted in GitLab.
8. Have everyone do a clean download from GitLab and run tests. I suggest using GitLab **Issues** to document the bugs you find and assign them to the person who needs to fix.
9. Fix bugs, repeat step 7 above. Iterate until you feel you have a clean product.
10. We will do the Git PULL at 10am on October 22nd.

### Rubric

- 100 points possible, with deductions for not implementing any of the checklist items above. For example:
    - -1 for EACH warning when compiling for iPhone SE
    - -5 if it doesn't compile out of GitLab for any reason
    - -2 for poorly structured code
    - -2 for absence of comments
    - -2 for poorly formatted View / Edit screens
    - -2 for problems viewing on the iPhone SE in Portrait (such as keyboard blocking the input fields).
    - -2 for any broken or missing functions from previous homework. This needs to be a superset of all functions developed and delivered in previous homework.
    - -2 for anything I noted in previous homework grading that has not been fixed.
    - -5 for not having an AppIcon
    - -5 or more if there is it appears that not every team member participated equally in the work. This is determined by looking at each individual's PUSH and MERGE REQUEST activity.
    - Other as will be noted in Rubric
- Everyone on your team can earn up to 5 points **EXTRA CREDIT** if you do the following:
    - Design an approach that defines the interface to your "Information View Flip Mechanism".
        - Specifically, you could have a design pattern, protocol, delegate method, or similar approach that defines how to plug in an animation view.
        - If someone adds code to your project and follows your design approach, then anyone should be able to write a VC with animation and have it registered and executed when your "Flip" option (button, page view controller, etc) is called.
        - Design approach should include the ability to "register" the fact that that Information View has an Animation view, and then that fact is notated somehow on your Table View.
        - All your team's entries should, of course, use your design approach so it is consistent across those.
        - **Clearly document** the interface in your README.md so that the TAs and I can try it out and try to insert our own Animations in your code.

## Semester Project

The semester project will be team-based. You will be forming teams of 3 (or 2 or 4 with permission from me). You will choose a project from a list of suggestions that I have placed on Sakai Resources or you may choose your own (with my approval). The project grade breaks down into 5 deliverables as detailed below.

## PJ1 – Team Formation / Project Choice

This will be due within the first few weeks of class. You will form a team and choose a project. You should create a slide or a document with all the information about the team members, the project, and why you chose the project you did. Post this document to Sakai on time to avoid late penalty. This assignment is worth 25 points which breaks down as follows:

- Up to 10 points for the concept itself. If you choose one of the projects off my list (Project List), you receive 10 points automatically. If you come up with your own idea, I will assign a grade based on creativity, difficulty and originality.
- Up to 15 points for the actual submission in Sakai – followed directions, on time, complete, readable, clear, concise, interesting, etc.

## PJ2 – Design Presentation

On the Due Date specified in Sakai Assignment PJ2, your team will give a quick 5 min presentation on your project – concept, design and plan. Create a presentation on your project for the class to hear and give you feedback. Just 3 slides are required - you can do more if you want as long as you stay to 5 minutes:

- Concept / Function Overview (text)
- System Design / Architecture (components, network protocols, database, server OS, etc) (text or diagram)
- User Interface screen flow / storyboard (diagram – using a tool, or Powerpoint, or even a picture of a **clearly drawn** diagram on whiteboard)

Not everyone needs to talk but please have the entire team up front. We will have a short Q&A after each presentation. Worth up to 25 points - grading based on completeness and clarity of presentation, content of the slides, and the overall concept. See Sakai Resources for examples, under "EXAMPLE Design Presentations". These is also a link to these presentations from the Sakai Assignment.

## PJ3 – Sponsor Feedback

You will need to meet with your Sponsor sometime before final presentations to review your near-final app with them and take them through the features / functions. Attached in the Sakai Assignment you will find the rubric I will be sending each sponsor to fill out. The data sent back to me from the sponsor will inform the grade for this assignment - 50 points possible.

**You need to respond/submit to the Sakai Assignmen**t once you have conducted your interview so I know to send out the Rubric.

## PJ4 – Project Presentation

Your team will give a 10-15-minute presentation at the end of the semester showing the actual working code, the final design/code and a demo. Worth up to 50 points. The presentation should be:

- 10-15 minutes long
- Show a few slides on how the project went, final functions, architecture overview
- A demo of your running code (simulator or I will have an Apple TV)

Attached to the Sakai Assignment are some example presentation from last year. Due by 2pm before class on Nov 28th. **Respond in Sakai with your presentation attached.**

## PJ5 – Final Code Review

Your final code needs to be in GitLab by 9am, Friday, December 7th. The final 250 points will be from a final code review conducted by myself and the TAs. We will review all the code for completeness, clarity, structure, conciseness, and overall programming style. We will also run the program and look for bugs, UI issues, and overall appeal. **Make sure you respond/submit in Sakai when your code is ready to be graded – any response after Dec 7th will be downgraded for being late.**

Some additional notes:

- If you have backend server code, it needs to be running throughout the week (Dec 7th – 13th) so we can grade.
- Apps need to build cleanly out of GitLab. We won't build the backend, but we would like to see the code in GitLab somewhere – let us know if you want another project open
- Make sure you have a clear README.md that explains key features/functions and any config / setup / login info we need to test. This can also serve as a "user's guide" if there is usage information we need to know.