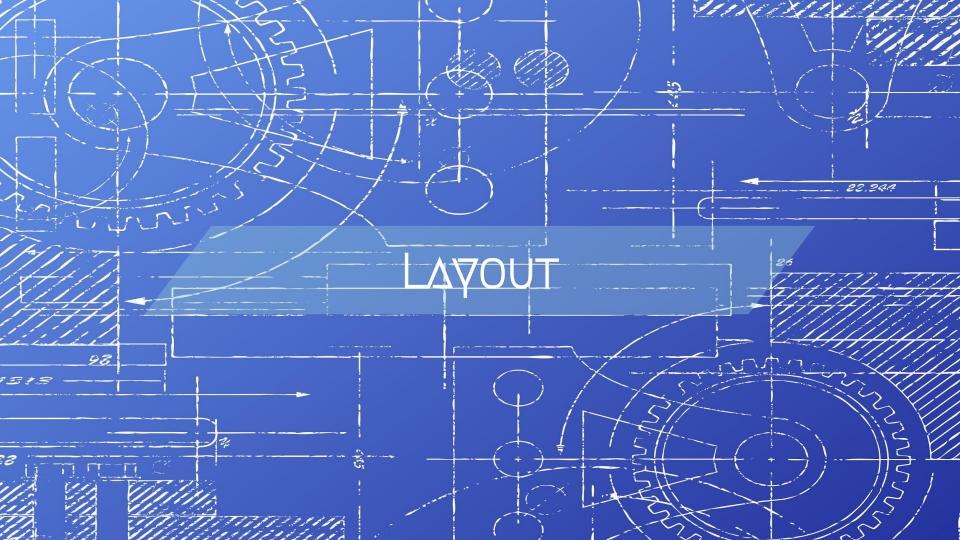# SWIFT

Liesel Wong and Tami Takada

- Programming language created by Apple in 2014 to develop native iOS apps
- Swift was made to replace **Objective-C**
- C-family language (control statements, use of brackets, etc. similar to C)

```swift
class Student: Person {
    let name: String
    var age: Int
    var osis: Int?

    init(name: String, age: Int?, osis: Int?) {
        self.name = name
        self.age = age ?? 17
        self.osis = osis
    }

    func increaseAge() -> Int {
        age += 1
        return age
    }
}
```
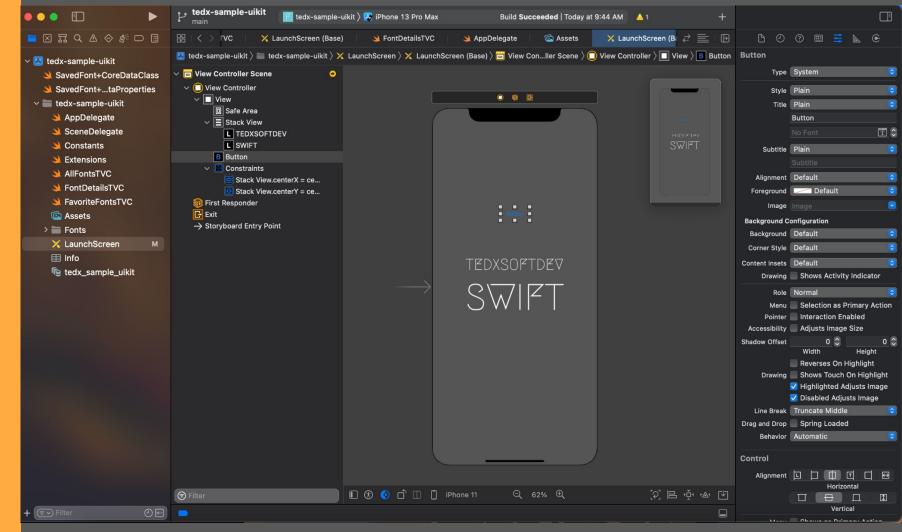
+

- Similar to Java but with many good parts of Python
- Multiple inheritance
- Extensions
- Optionals

-

- Not cross-platform
- No iPhone only option
- Xcode
- $100/yr developer program

```swift
extension Int {
    func getMinutesAndSeconds() -> (Int, Int) {
        let mins = self / 60
        let secs = self % 60
        return (mins, secs)
    }
}
```
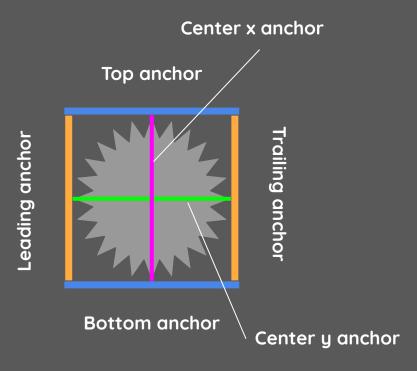
LAYOUT

- Used with UIKit (default option)
- Allows user to drag various UI components onto a screen to build a layout
- Good for starting out with, but limited customization

STORYBOARD
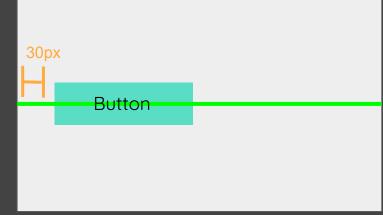
- Layout with constraints
- Can use stacks like flex boxes in CSS
- Easy to center items

Center x anchor

Top anchor

Leading anchor

Trailing anchor

Bottom anchor

Center y anchor

```swift
let b = UIButton()

view.addSubview(b)

b.translatesAutoresizingMaskIntoConstraints = false

b.leadingAnchor.constraint(equalTo: view.leadingAnchor,
constant: 30).isActive = true
b.centerYAnchor.constraint(equalTo:
view.centerYAnchor).isActive = true
```

30px

Button

What is SwiftUI

SwiftUI is another UI Framework (like UIKit) released in 2019
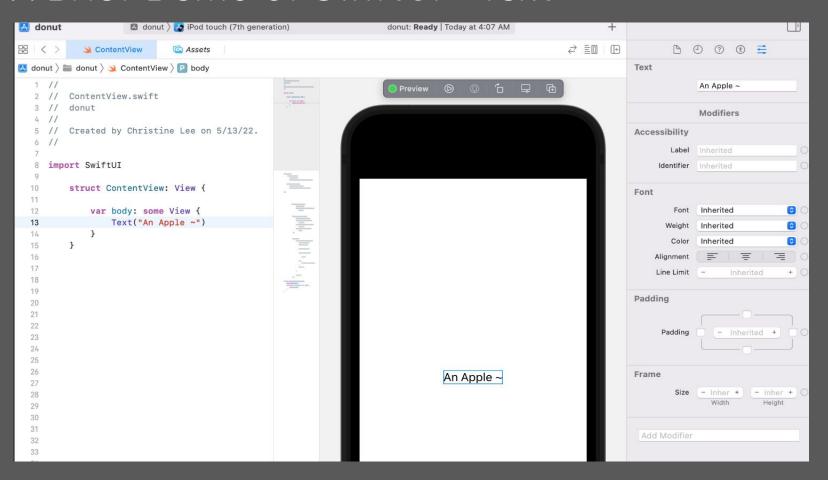
It still uses the Swift language, but…
- Doesn't include Storyboard (aka you can't drag and drop elements onto the fake iPhone)
- Is a Declarative Framework, whereas UIKit is an Imperative Framework

Some issues with SwiftUI are…
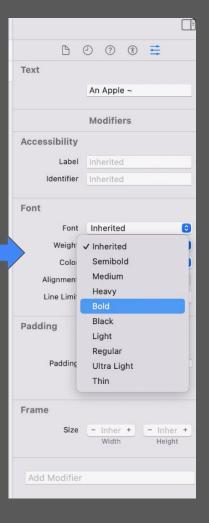- It is newer than UIKit so there is less documentation
- Doesn't work on older computers & operating systems
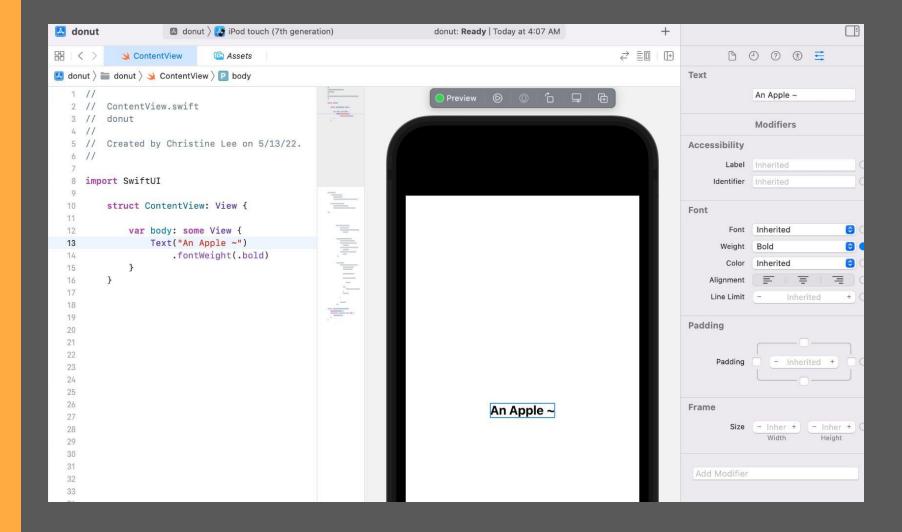
A Brief Demo of SwiftUI - Text
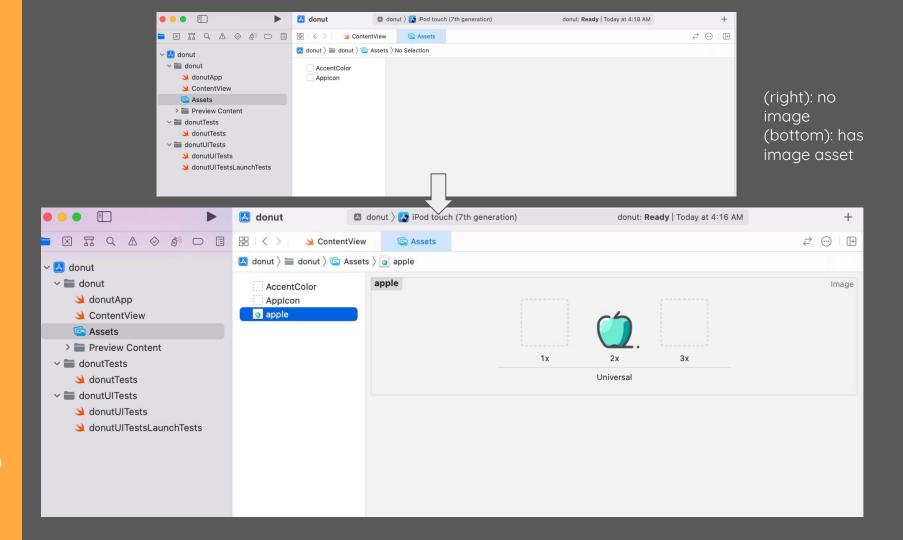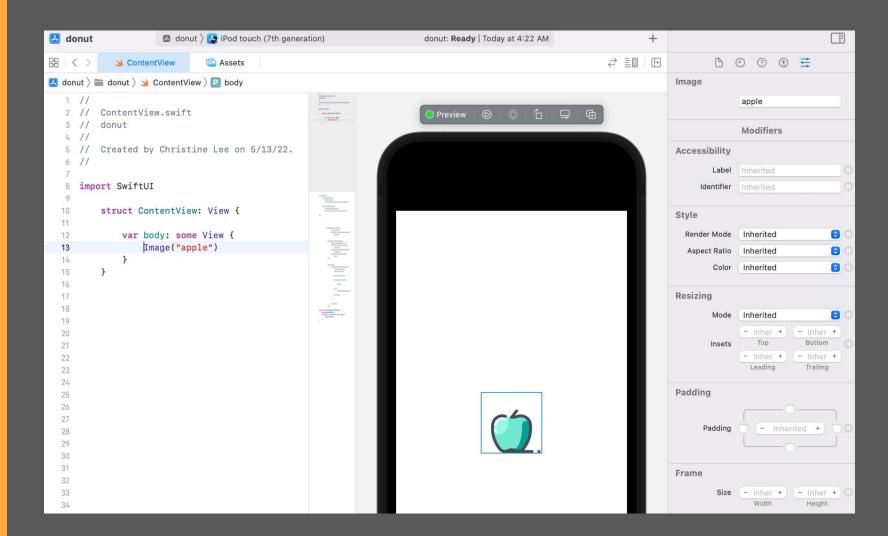
This dropdown has the options for the Weight modifier

TEXT WITH SWIFTUI

```
1    //
2    //  ContentView.swift
3    //  donut
4    //
5    //  Created by Christine Lee on 5/13/22.
6    //
7
8    import SwiftUI
9
10   struct ContentView: View {
11
12       var body: some View {
13           Text("An Apple ~")
14               .fontWeight(.bold)
15       }
16   }
17
```

An Apple ~

Text

An Apple ~

Modifiers

Accessibility

Label   Inherited
Identifier   Inherited

Font

Font   Inherited
Weight   Bold
Color   Inherited
Alignment
Line Limit   Inherited

Padding

Padding   Inherited

Frame

Size   − Inher +   − Inher +
        Width   Height

Add Modifier

(right): no image
(bottom): has image asset

IMAGES WITH SWIFTUI

```
 1  //
 2  //  ContentView.swift
 3  //  donut
 4  //
 5  //  Created by Christine Lee on 5/13/22.
 6  //
 7
 8  import SwiftUI
 9
10  struct ContentView: View {
11
12      var body: some View {
13          Image("apple")
14      }
15  }
```

Now let's add images and text on the same screen!

```
VStack {

    Image("apple")
        .resizable()
        .aspectRatio(contentMode: .fit)

    Text("An Apple ~")
        .fontWeight(.heavy)

}
```

```
HStack {

    Image("apple")
        .resizable()
        .aspectRatio(contentMode: .fit)

    Text("An Apple ~")
        .fontWeight(.heavy)

}
```

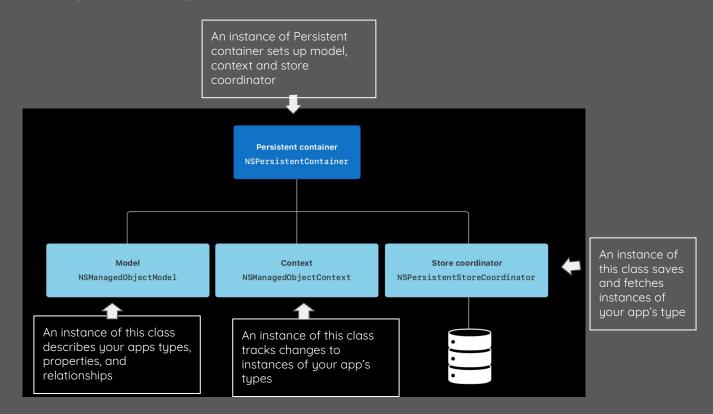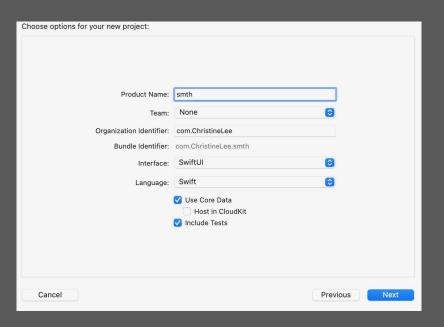# CORE DATA

What Is CoreData?

CoreData works with Swift to manage your app's database through a Persistent Container Class

An instance of Persistent container sets up model, context and store coordinator

Persistent container
NSPersistentContainer

Model
NSManagedObjectModel

Context
NSManagedObjectContext

Store coordinator
NSPersistentStoreCoordinator

An instance of this class saves and fetches instances of your app's type

An instance of this class describes your apps types, properties, and relationships

An instance of this class tracks changes to instances of your app's types

When Accesssing Core Data in Your Project:

1) Allow Core Data to be used (image of checking off)

Choose options for your new project:

Product Name: smth
Team: None
Organization Identifier: com.ChristineLee
Bundle Identifier: com.ChristineLee.smth
Interface: SwiftUI
Language: Swift
☑ Use Core Data
☐ Host in CloudKit
☑ Include Tests

Cancel   Previous   Next

2) Computer will create an instance of the Persistant Container
(so you can use all the classes on previous slide)

3) Add the classes you're using into a file related to the NSManagedObjectModel (allows you to describe an instance of the classes in your app through a table format)

4) In your code, reference the Persistent Container and NSManagedObjectContext (tracks instances of your app's classes and changes to your app data) through viewContext

5) Then you can fetch you app data in your code, and display it

FONTBOOK DEMO

FONTBOOK

Enter text here

Academy Engraved LET ★
Almost before we knew it, we had left the ground.

Al Nile ☆
Almost before we knew it, we had left the ground.

American Typewriter ★
Almost before we knew it, we had left the ground.

Apple Color Emoji ☆
Almost before we knew it, we had left the ground.

Apple SD Gothic Neo ☆
Almost before we knew it, we had left the ground.

Apple Symbols ☆
Almost before we knew it, we had left the ground.

American Typewriter

Regular
Almost before we knew it, we had left the ground.

Light
Almost before we knew it, we had left the ground.

Semibold
Almost before we knew it, we had left the ground.

Bold
Almost before we knew it, we had left the ground.

Condensed
Almost before we knew it, we had left the ground.

CondensedLight
Almost before we knew it, we had left the ground.

CondensedBold
Almost before we knew it, we had left the ground.

FAVORITE FONTS

Academy Engraved LET
Almost before we knew it, we had left the ground.

American Typewriter
Almost before we knew it, we had left the ground.

Bradley Hand
Almost before we knew it, we had left the ground.

Charter
Almost before we knew it, we had left the ground.

Cochin
Almost before we knew it, we had left the ground.