iOS Foundations II
Session 2

- AutoLayout
- Size Classes
- NSUserDefaults
- NSKeyedArchiver

Data Persistence

	Core Data	NSKeyedArchiver	NSUserDefaults
Entity Modeling	Yes	No	No
Querying	Yes	No	No
Speed	Fast	Slow	Slow
Serialization Format	SQLite, XML, or NSData	NSData	Binary Plist
Migrations	Automatic	Manual	Manual
Undo Manager	Automatic	Manual	Manual

Data Persistence

	Core Data	NSKeyedArchiver	NSUserDefaults
Persists State	Yes	Yes	Yes
Pain in the Ass	Yes	No	No

NSUserDefaults

- “NSUserDefaults allows an app to customize its behavior based on user preferences”
- Think of it as an automatically persisting plist that is easily modified in code.
- Use the standardUserDefaults class method to return the shared defaults object.
- Setting values inside of it is as easy as these methods:
  - setBool:ForKey:
  - setObject:ForKey:
  - setInteger:ForKey:

NSUserDefaults

- Each app has its own database of user preferences
- Used to store and retrieve an object
- Objects must be NSCoding-compliant
- Primitives may be stored as-is (Float, Int, Bool, String, etc.)

NSUserDefaults – Demo

NSKeyedArchiver

- NSKeyedArchiver and NSKeyedUnarchiver provide a convenient API to read / write objects directly to / from disk.

Archiving

```
NSKeyedArchiver.archiveRootObject(books, toFile:"/path/to/archive")
```

Unarchiving

```
NSKeyedUnarchiver.unarchiveObjectWithFile("/path/to/archive")
```

NSCoding

- NSCoding is a protocol that your objects must conform to if you plan on archiving/unarchiving them
- NSCoding all-the-way-down

Unarchiving

```
init(decoder: NSCoder) {…}
```

Archiving

```
func encodeWithCoder(encoder: NSCoder) {…}
```

NSCoding — Demo