GENERAL ASSEMBLY

MOBILE DEVELOPMENT

DATA PERSISTENCE WITH CORE DATA

Rudd Taylor
Founder, SALT

LEARNING OBJECTIVES

- Breakdown the Core Data stack at a high level.
- Create, read, update, delete (CRUD) data with Core Data.
- Discuss when to/not to use Core Data over plists/flat files/user defaults.

BUILDING A TODO APP

EXPENSE APP – "EXPENSETRACKER"

- View Controller should have a table view and navigation bar.
- The UITableView should have a single prototype cell with type "Right detail."
- Data model should have an Expense class and ExpenseManager singleton.
- Expense class should have a title (String) field and an amount (Double) field.
- The View Controller should also have a "+" button in the upper right, and should show an alert dialog asking for a title and amount of a new Expense.
- The ExpenseManager class should have "create" and "delete" methods. "create" accepts a title and amount to create a new Expense and add it to the list of Expenses. "delete" should accept an Integer index and remove it from the list.
- The navigation bar's title should be the sum of the amounts of all Expenses.

CORE DATA

CORE DATA

Core Data is an object persistence framework provided by iOS.

Instead of a file system, Core Data utilizes a database to enable us to write persistent code like writing a data model of classes.

- Very powerful, very complicated.
- Lots of "boilerplate" code required.
- Provides a lot of additional services like change (undo/redo) management, schema migration, lazy object loading, relationship maintenance, etc.

- https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/CoreData/index.html

CORE DATA

- Managed object model (MOM): a file that represents the data model, essentially the database schema.
- Entity: essentially a class definition in Core Data.
- Attribute: a property of an entity (a member variable).
- Relationship: link between two entities. This is where entity (table) relationships are defined.
- Managed object: an entity that we want to store in Core Data. Its instances are placed in managed object context.
- Managed object context: this is the virtual representation of our data. This instance of data can be manipulated and saved when ready.

CORE DATA

- We always work on the Managed Object Context.
- A series of operations are performed on the MOC (insert, fetch & update, delete), then saved when we want them to persist.

CORE DATA RETROFIT

ADDING CORE DATA

- Use Core Data to store note information instead of "ephemeral" properties.
- Edit the Core Data data model to add an Expense entity.
- Create properties in your ExpenseManager for the AppDelegate, Managed Object Context, and Expense entity.
- Make Expense a subclass of NSManagedObject.
- In the initializer, fetch all the Expenses.
- In the "create" method, initialize your new Expense using the NSManagedObject initializer.
- In the delete method, add a Core Data method to remove it from the store.
- Write a save() method and use it liberally throughout the app.