CS193E
Lecture 15

Core Data

Agenda

- Questions
- Final Projects
- Core Data (and a little more bindings as well)

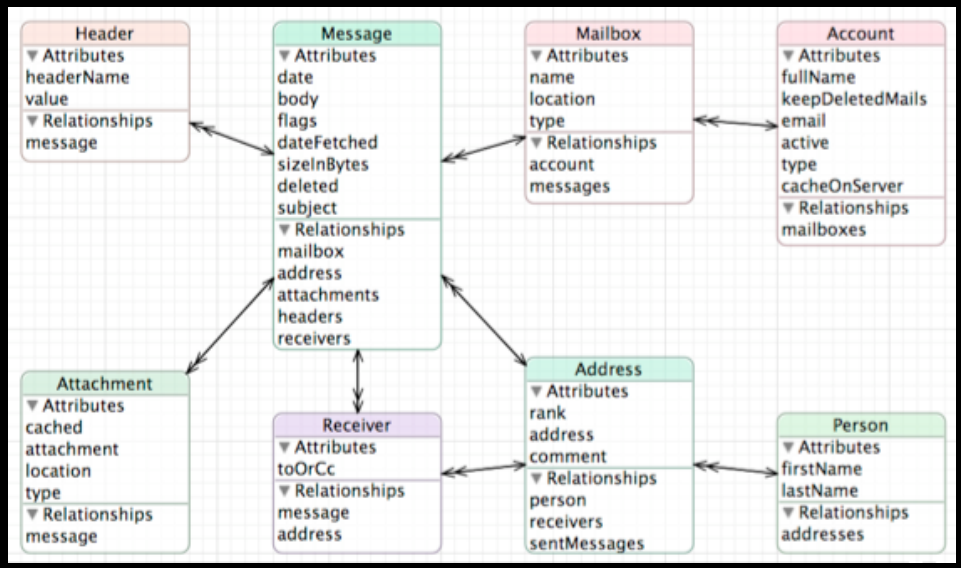
Core Data Introduction

- Continues initiative started with Cocoa Bindings: more functionality with less code
- Core Data addresses a different area of your code
  - Solves much more difficult problems
- Core Data is about fine-grained management of data objects
- Core Data is a "model-driven object graph management and persistency framework"

What?!?

Model-driven Development

- You create a description of your data objects in Xcode



- Core Data handles most aspects of your data objects, both in-memory and on-disk
- You focus on the application logic

What does Core Data do for you?

- Automatic undo/redo
- Scalable object lifecycle management
- Data consistency and correctness
- User interface synchronization through bindings integration

What does Core Data do for you?

Read/write to various types of files

	Scalable to large data sets	Performance	Human Readable	Atomic Access
SQLite	✓	best		
XML		good	✓	✓
Binary		better		✓

Core Data owns the schemas

Example Use Cases

Safari	History, bookmarks
Address Book	Persons, groups database
Mail	Mailboxes files
iPhoto	Music library
iTunes	Photo library
iCal	Calendars
Xcode	Project data

- Every application that manages lots of objects

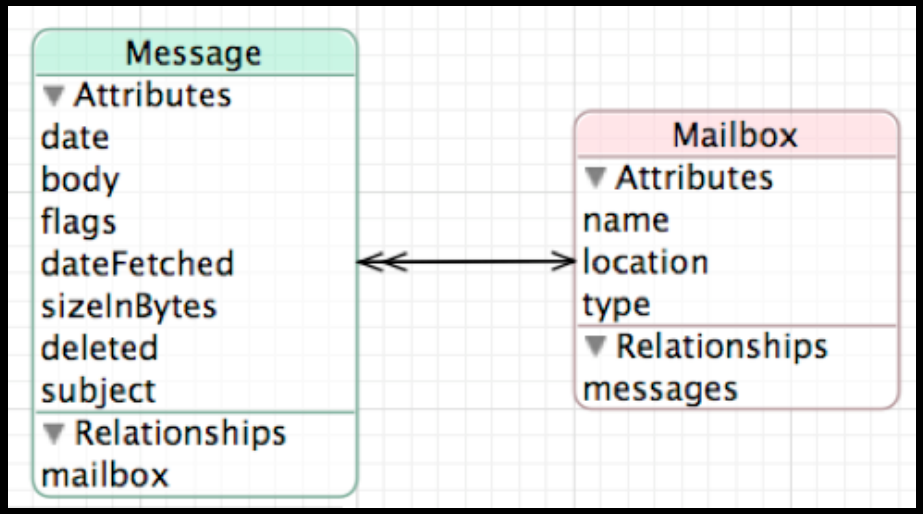
Why is Core Data good for you?

- Relieves you from having to reinvent the wheel
  - Faster development cycle
- Robustness
- Flexibility
  - Choice of persistent stores

Models

Models

- High-level description of your data objects
- Entity-relationship diagram



- The more detailed the model, the better Core Data works

Model Files

- Models are entered and modified in Xcode
- File representation separated for Xcode and runtime
  - Xcode manages additional graphical information
  - Runtime model thinned out for best performance

Model Contents

- Entities
  - Types of objects
  - Class name for runtime representation
- Properties
  - Object components
    - Attributes
    - Relationships (unordered NSSet)
    - Fetched Properties
  - Validation rules
  - Delete rules
  - Default values

Model Contents

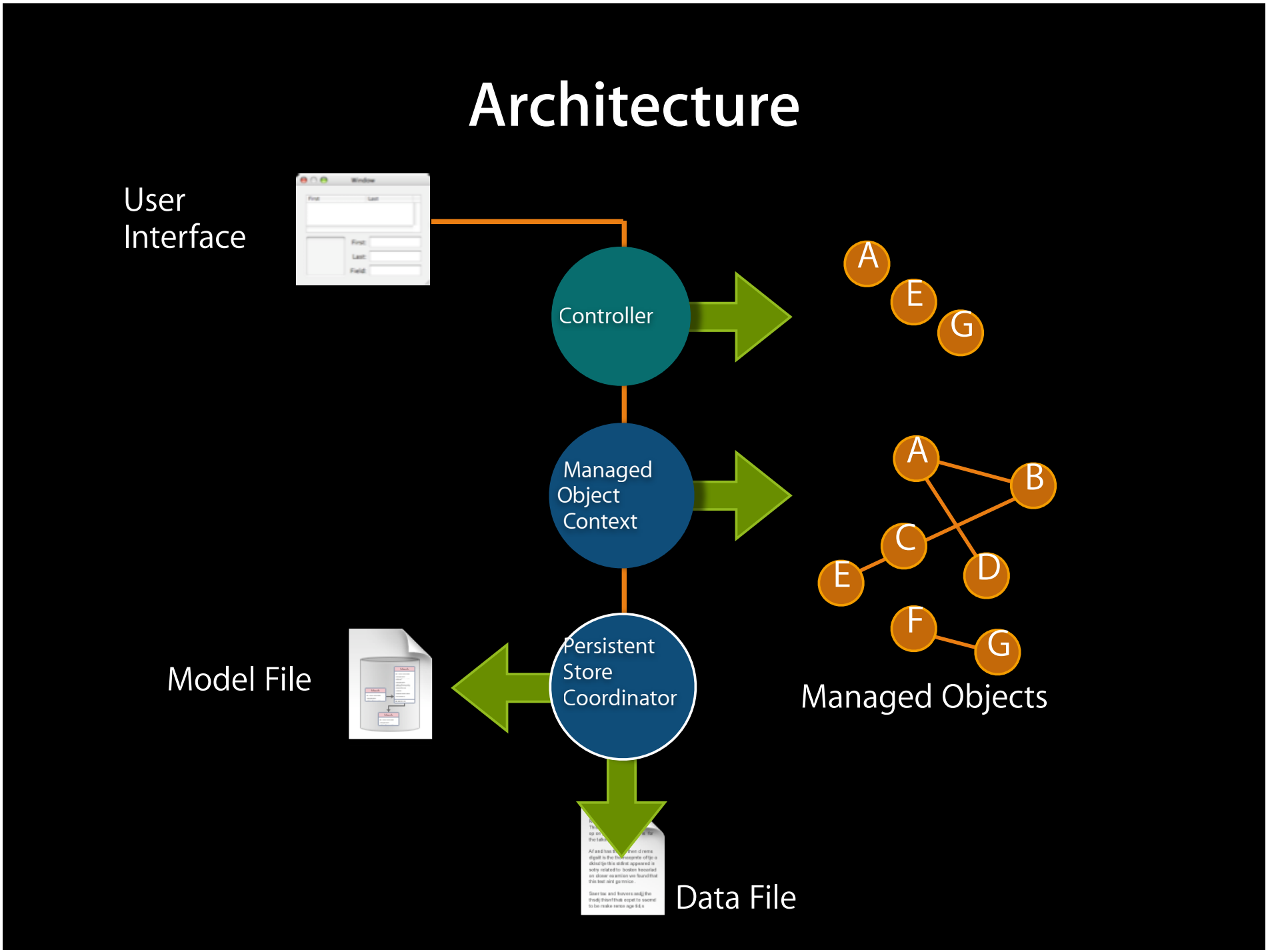
Entity	Class
Attribute	Simple instance variable
Relationship	Managed reference

Demo

Code-free Document-based Core Data Cocoa Application

Architecture

Architecture



Managed Object Context

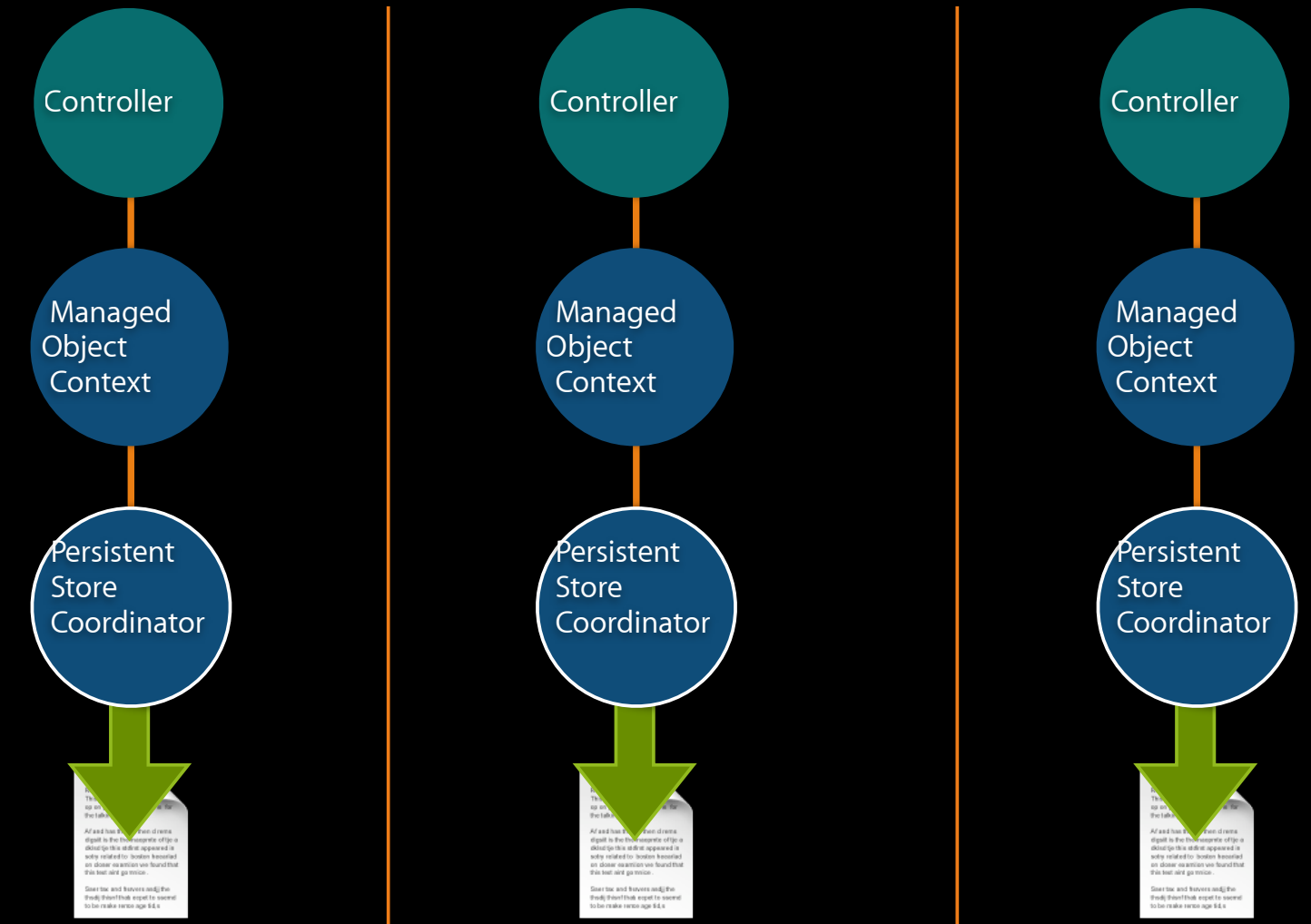
- The center of your new world
- Manages “everything”
  - Tracks changes
  - Maintains inverse relationships
  - Triggers validation
  - etc.
- Serves as gateway to persistent store coordinator
- Has access to an undo manager
- Acts as “scratchpad”

Persistent Store Coordinator

- Provides the model
- Synchronizes access to persistent stores
  - Unified representation for multiple stores

Architecture

Document Based Application



NSPersistentDocument

- Subclass of NSDocument
- Adds ability to work with a model and managed object context
- Makes document persistence ‘automatic’

Demo Revisted

Code You Have To Write

Managed Objects

- You still have to write your application logic
- Core Data requires you to use NSManagedObjects
  - Generic data object
  - Sufficient for many cases
- Custom subclasses for application logic
  - Key-value coding/observing (KVC/KVO)
  - Validation
  - Reacting to various state changes

Questions?