Разработка приложений для OS X

Лекция III
Протоколы. Фреймворки. Foundation.

Hexlet University, 2012

Про домашние задания

Протоколы

- Как interface в Java
- Просто общий список методов для классов
- Просто список! Методы могут быть не реализованы.
- Некоторые методы обязательны, некоторые – опциональны
- Реализация всех обязательных методов – **conforming** и **adopting**

Пример?

Протоколы

```
@protocol ProtocolName
    // required methods
@optional
    // optional methods
@required
    // more required methods
@end

@interface ClassName : ParentClass <ProtocolName>
    //
@end

@implementation ClassName
    // methods implementation
@end
```

Протоколы

- Не связаны с классами. Любой класс может реализовать любой протокол.
- Класс может реализовать несколько протоколов

```
@interface ClassName : ParentClass <ProtocolName, AnotherProtocolName>
```

Проверка реализации

```
if ([myObject conformsToProtocol: @protocol (ProtocolName)] == YES) {
    // can use methods declared in ProtocolName
}
```

Проверка наличия метода

```
if ([myObject respondsToSelector: @selector (myMethod)] == YES){
    // can call myMethod on myObject
}
```

Проверка наличия метода

```
id <myProtocol> myObject;
```

Это сообщает компилятору что myObject будет обращаться к объекту который адаптирует myProtocol. Если myObject будет указывать на объект который не адаптирует тот протокол:

```
warning: class 'yourClass' does not implement the
'myProtocol' protocol
```

Протоколы

- Можно указать несколько протоколов

```
id <myProtocol, anotherOne> myObject;
```

-

-

Протоколы

- Можно указать несколько протоколов

```
id <myProtocol, anotherOne> myObject;
```

- Можно расширять существующие протоколы

```
@protocol anotherProtocol <myProtocol>
```

-

Протоколы

- Можно указать несколько протоколов

```
id <myProtocol, anotherOne> myObject;
```

- Можно расширять существующие протоколы

```
@protocol anotherProtocol <myProtocol>
```

- Категории могут реализовывать протоколы!

```
@interface Fraction (myCategory) <myProtocol, anotherOne>
```

Неформальные протоколы

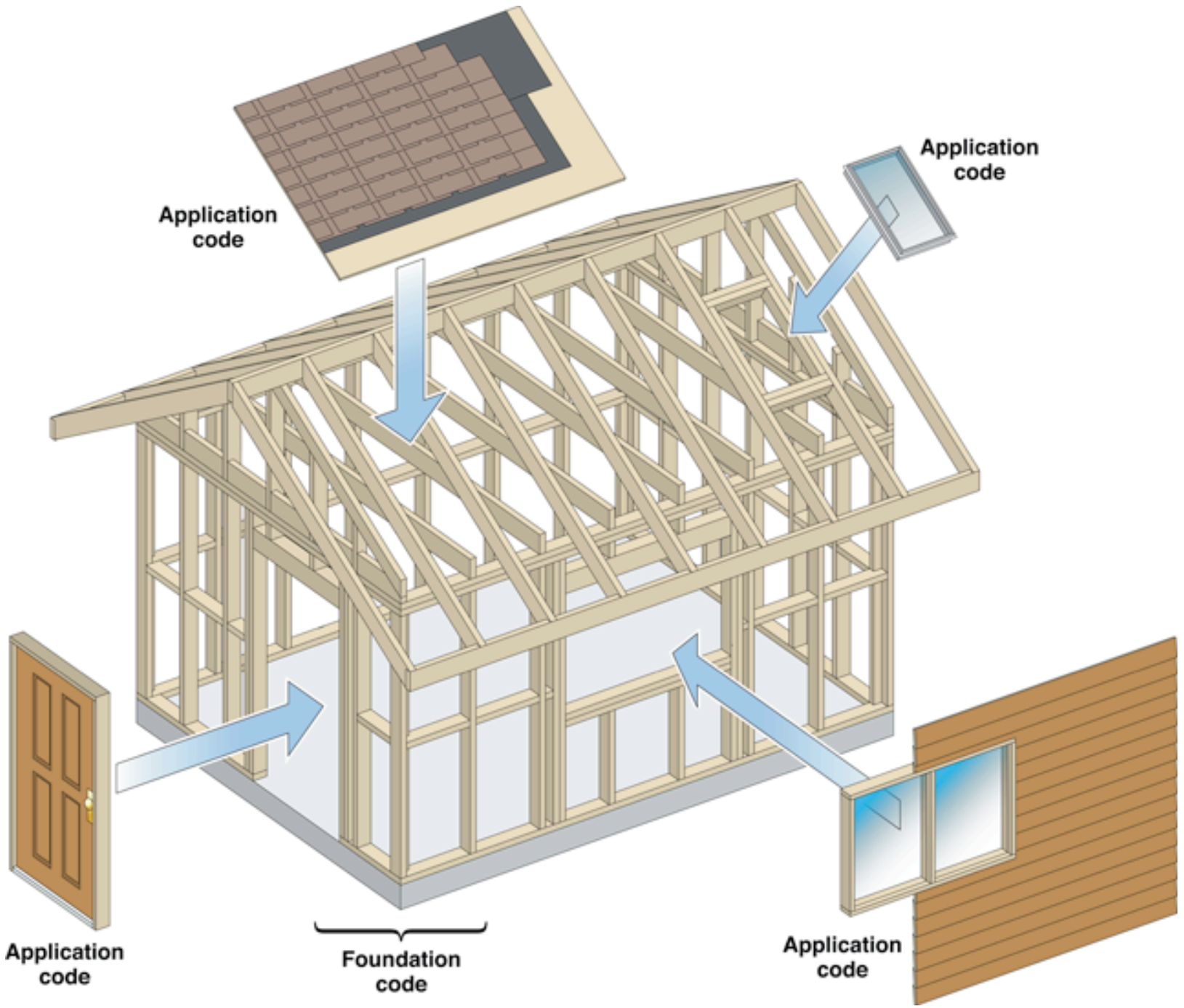
Категория методами. Более не используется, так как теперь есть @optional в протоколах в Objective C 2.0

```
@interface NSObject (NSComparisonMethods)
- (BOOL)isEqualTo:(id)object;
- (BOOL)isLessThanOrEqualTo:(id)object;
- (BOOL)isLessThan:(id)object;
- (BOOL)isGreaterThanOrEqualTo:(id)object; - (BOOL)isGreaterThan:(id)object;
- (BOOL)isNotEqualTo:(id)object;
- (BOOL)doesContain:(id)object;
- (BOOL)isLike:(NSString *)object;
- (BOOL)isCaseInsensitiveLike:(NSString *)object;
@end
```

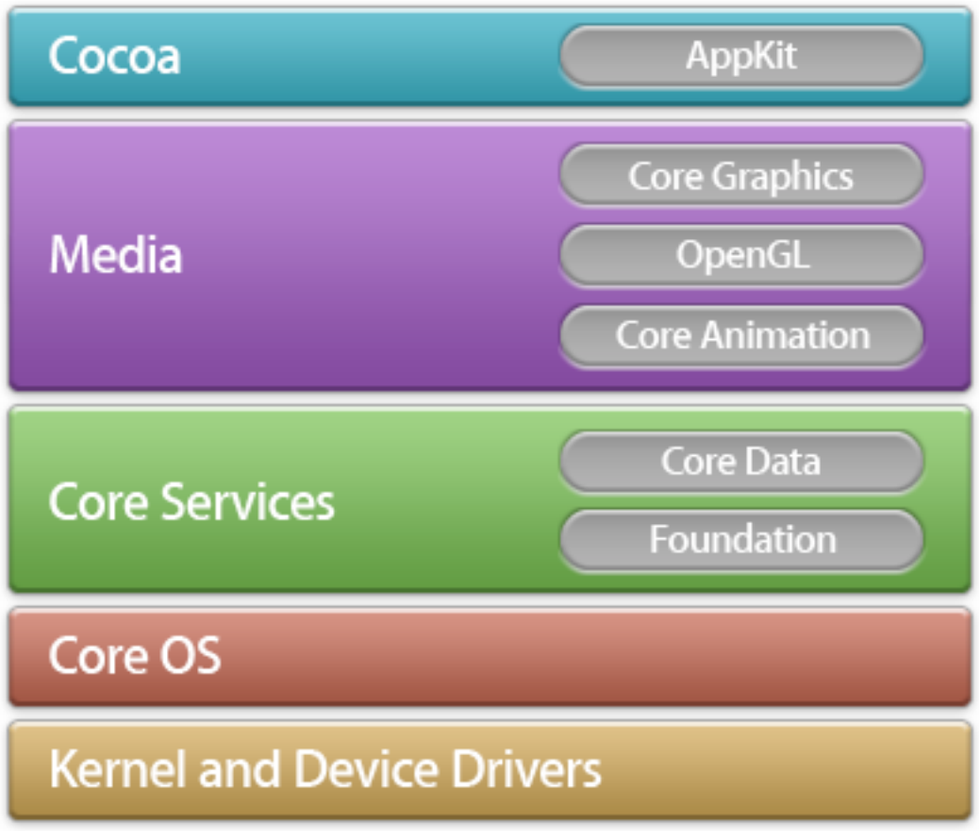
Фреймворки

Фреймворк

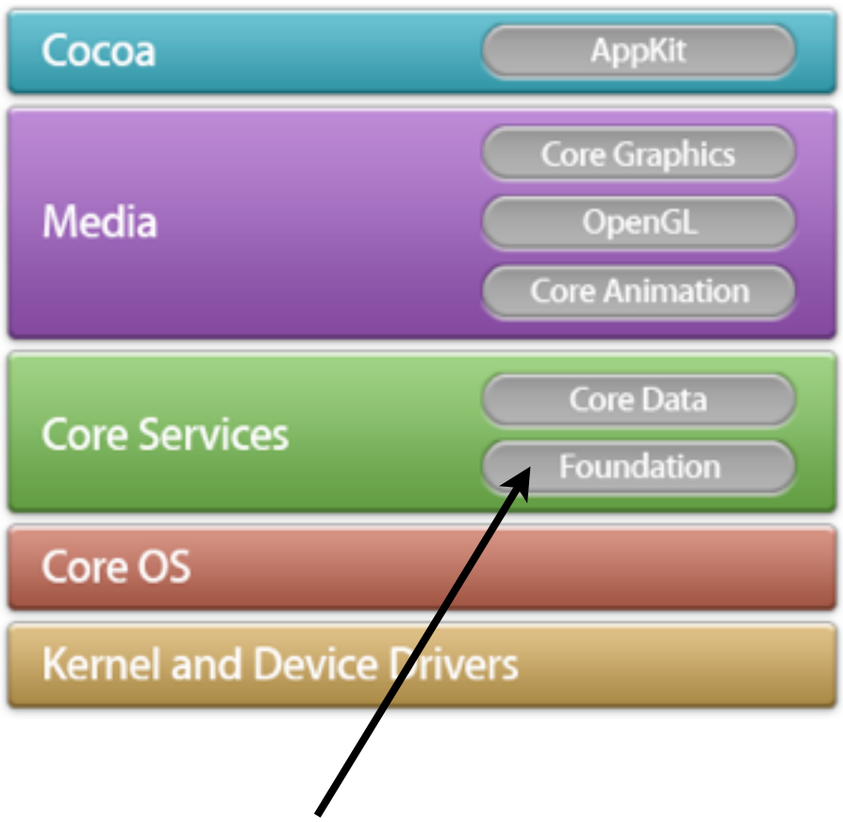
Набор классов, методов, функций и документации, сгруппированных в логическую структуру для упрощения разработки.


Application code
Application code
Application code
Foundation code
Application code

Структура Mac OS X

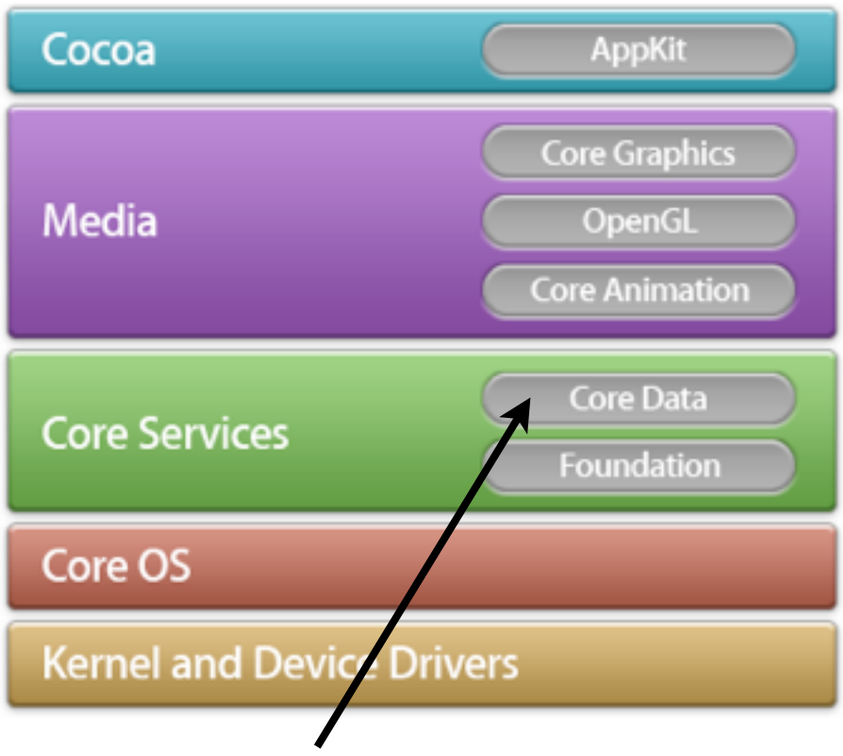


Foundation Framework



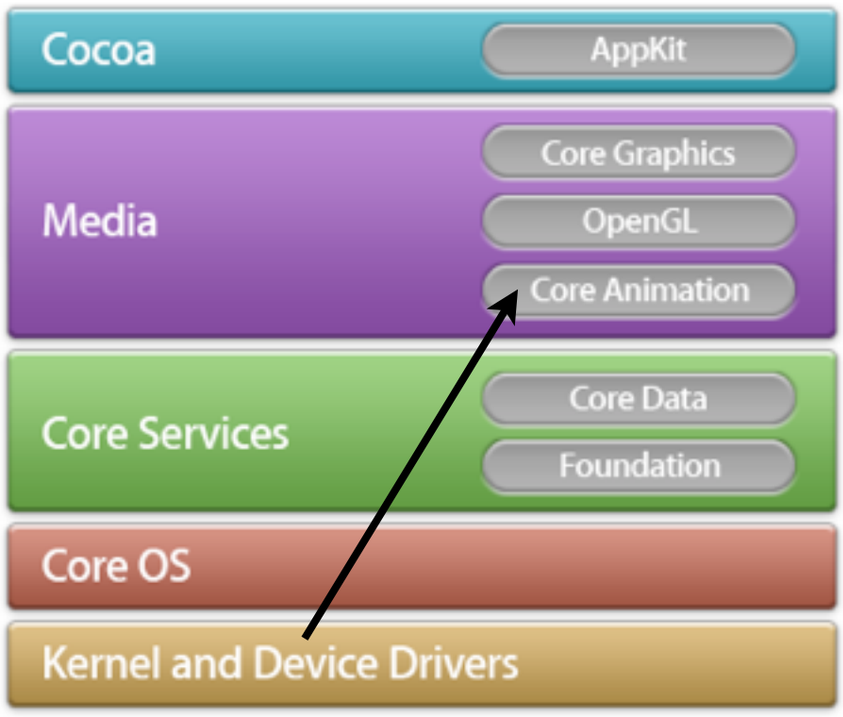
- Простые типы (числа, строки)
- Коллекции – массивы, множества, словари, etc.
- Дата, время
- Файловая система
- Хранение объектов
- Геометрические 2Д-объекты (points, rectangles, etc)

Core Data Framework



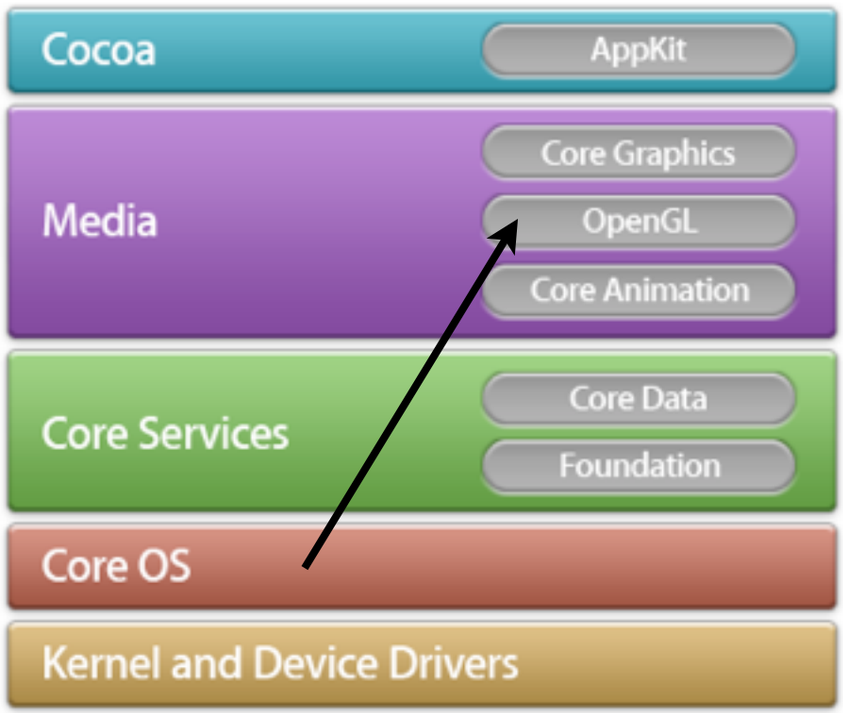
- Сохранение и загрузка объектов
- Простые undo/redo
- Автоматическая валидация значений
- Фильтрация, группировка и организация данных в памяти
- Поддержка document-based apps (об этом – в лекции 5-6)

Core Animation



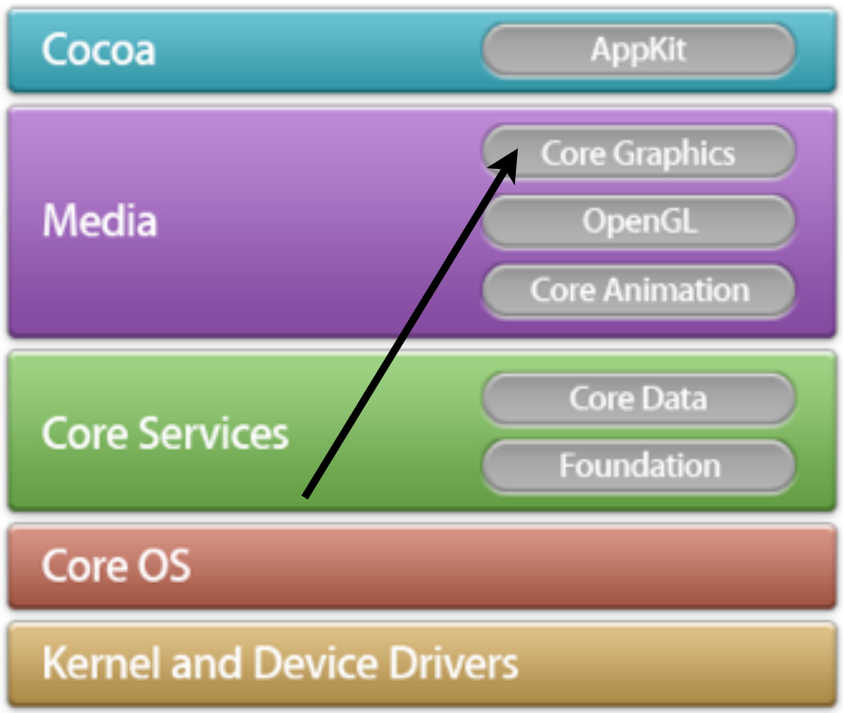
- Собственная анимация
- Таймер и анимация
- Покадровая анимация
- Специальные графические ограничения
- Группировка изменений из нескольких слоев в одно микро-событие

OpenGL Framework



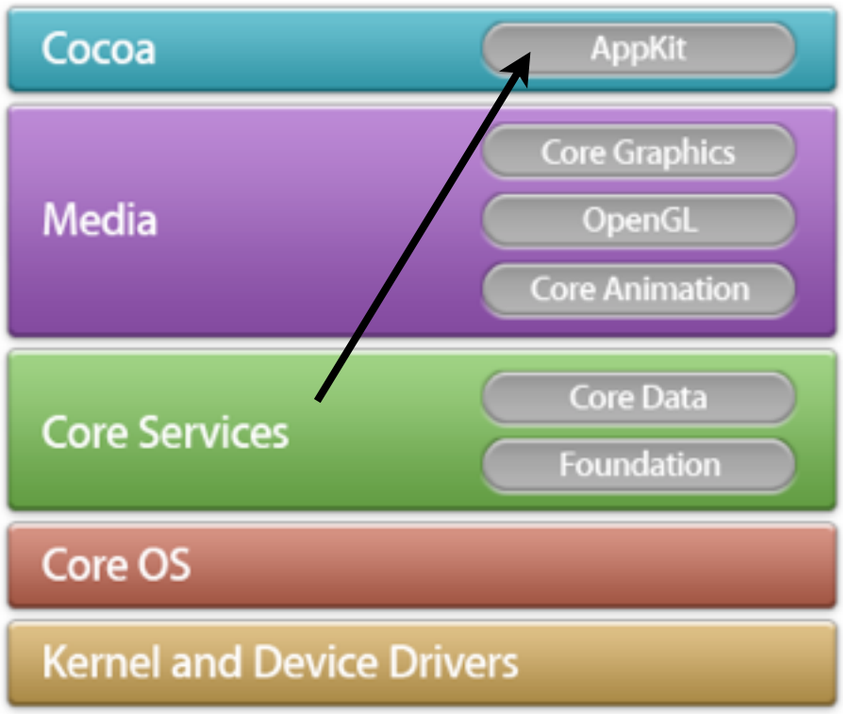
- 2D и 3D графика
- Игры, симуляция и так далее
- Использует несколько потоков (threads)
- Доступ к графическому железу

Core Graphics



- Рисование с помощью путей (paths)
- Antialiasing
- Градиенты, фильтры, изображения
- Трансформации в координатной плоскости

AppKit Framework



- Создание и управление пользовательским интерфейсом
- Обработка пользовательских событий
- Шрифты, цвета, картинки
- Простая анимация
- Базовые свойства приложений – Spaces, Help Support, несколько аккаунтов
- Кастомные элементы интерфейса

Документация Foundation

- Полная документация в XCode
- Option+Click на слове в коде
- Quick help в панели утилит (справа)
- Два главных сайта

Foundation: Числа

```
NSNumber *myNumber, *floatNumber, *intNumber;
NSInteger myInt;

// integer value
intNumber = [NSNumber numberWithInteger: 100];
myInt = [intNumber integerValue];
NSLog (@"%li", (long) myInt);

// long value
myNumber = [NSNumber numberWithLong: 0xabcdef];
NSLog (@"%lx", [myNumber longValue]);

// float value
floatNumber = [NSNumber numberWithFloat: 100.00];
NSLog (@"%g", [floatNumber floatValue]);

// double
myNumber = [NSNumber numberWithDouble: 12345e+15];
NSLog (@"%lg", [myNumber doubleValue]);
```

NSLog

```
NSString *str = @"Why u no..?";
NSLog (@"%@", str);
```

Можно задать собственный формат вывода объекта на экран с помощью @ перезаписав метод description.

метод description

```
-(NSString *) description {
    return [NSString stringWithFormat: @"%i/%i", numerator, denominator];
}
```

Попробуем!