Designing Interactive Systems II

Computer Science Graduate Programme SS 2010

Prof. Dr. Jan Borchers
RWTH Aachen University

http://hci.rwth-aachen.de

Review: Windows 7

- What is special about the Window Manager?
- How is DirectX incorporated in Windows 7?
- What is .NET?
- What is the WPF?
- Which features of .NET are enabled by ...
  - ... the CLR?
  - ... the CTS?

Window Systems for Mobile Devices

Android

- Open Handset Alliance Project
- Free and open platform for mobile devices
- Same API for all applications
  - No differentiation between ‘native’ and ‘third-party’ applications
- Java is the only supported language

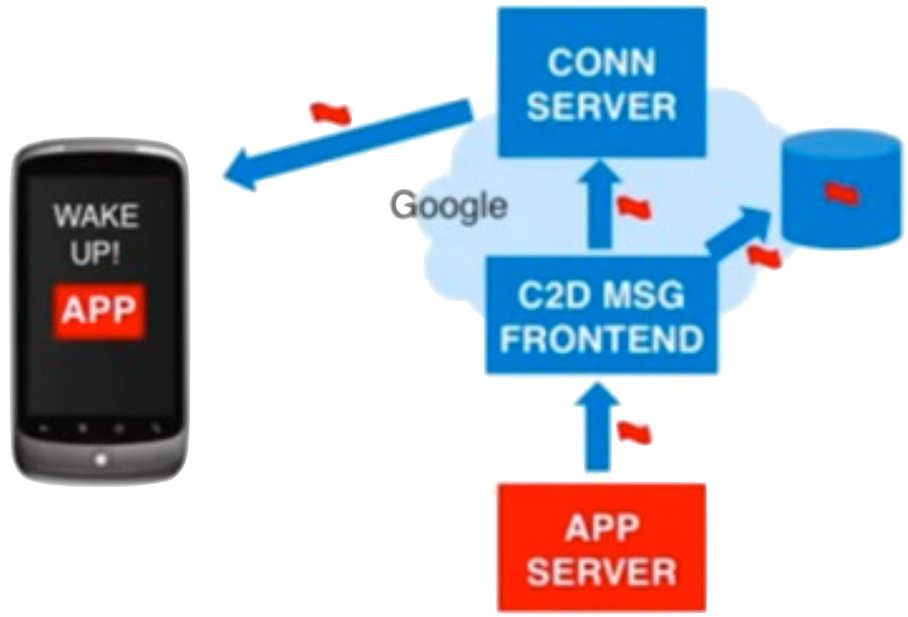
History

- Founding of Open Handset Alliance (Nov 2007)
  - Preview release of Android SDK
- Initial release of Android SDK (Oct 2008)
  - G1 as first Android device
    - 480x320 resolution
    - Qwerty keyboard and capacitive touchscreen
    - GPS, accelerometer, compass

History

- 1.000.000 G1 shipped by HTC (Dec 2008)
- Android 1.5 (Apr 2009)
  - Video recording
  - Direct uploading to YouTube and Picasa
  - Soft keyboard with text prediction
- Android 2.0 (Oct 2009)
  - Microsoft Exchange support
  - HTML 5 support

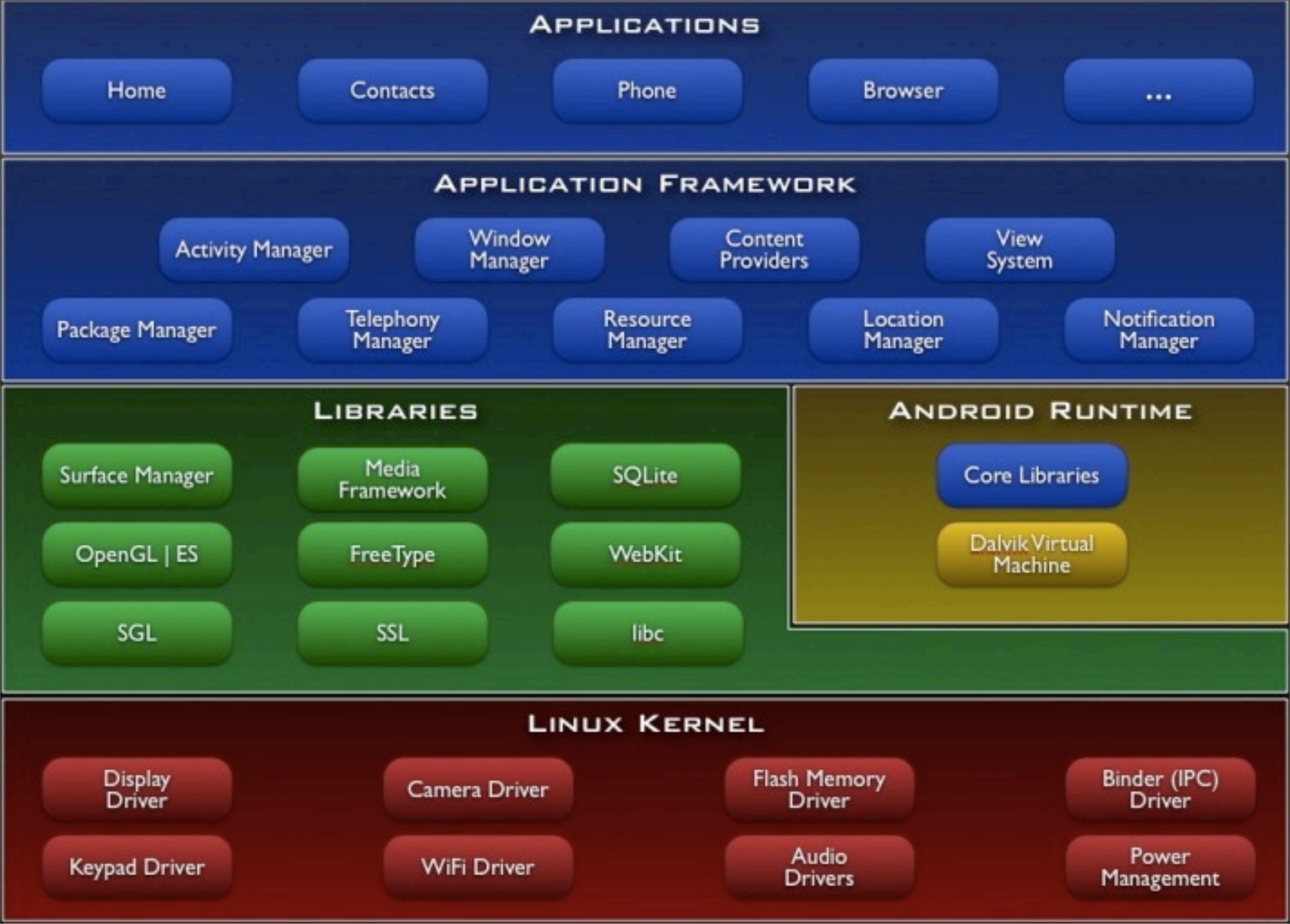
History

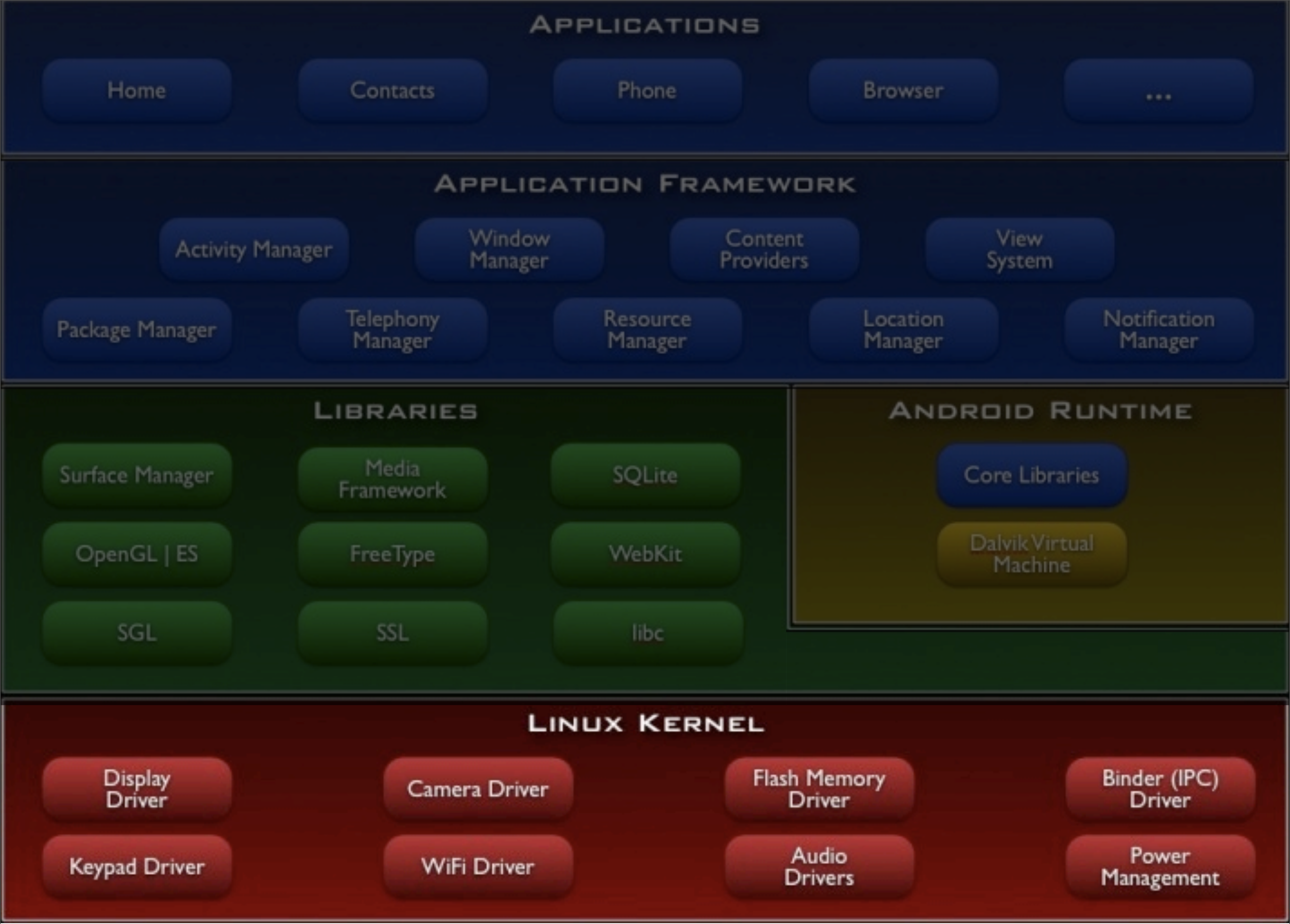


- Android 2.2 (May 2010)
  - Enterprise features
    - Remote wipe
    - Password policies across company devices
    - Improved Microsoft Exchange support
- Cloud to Device Messaging
- Dalvik JIT compiler (2–5x performance increase)

Android Developer Challenge

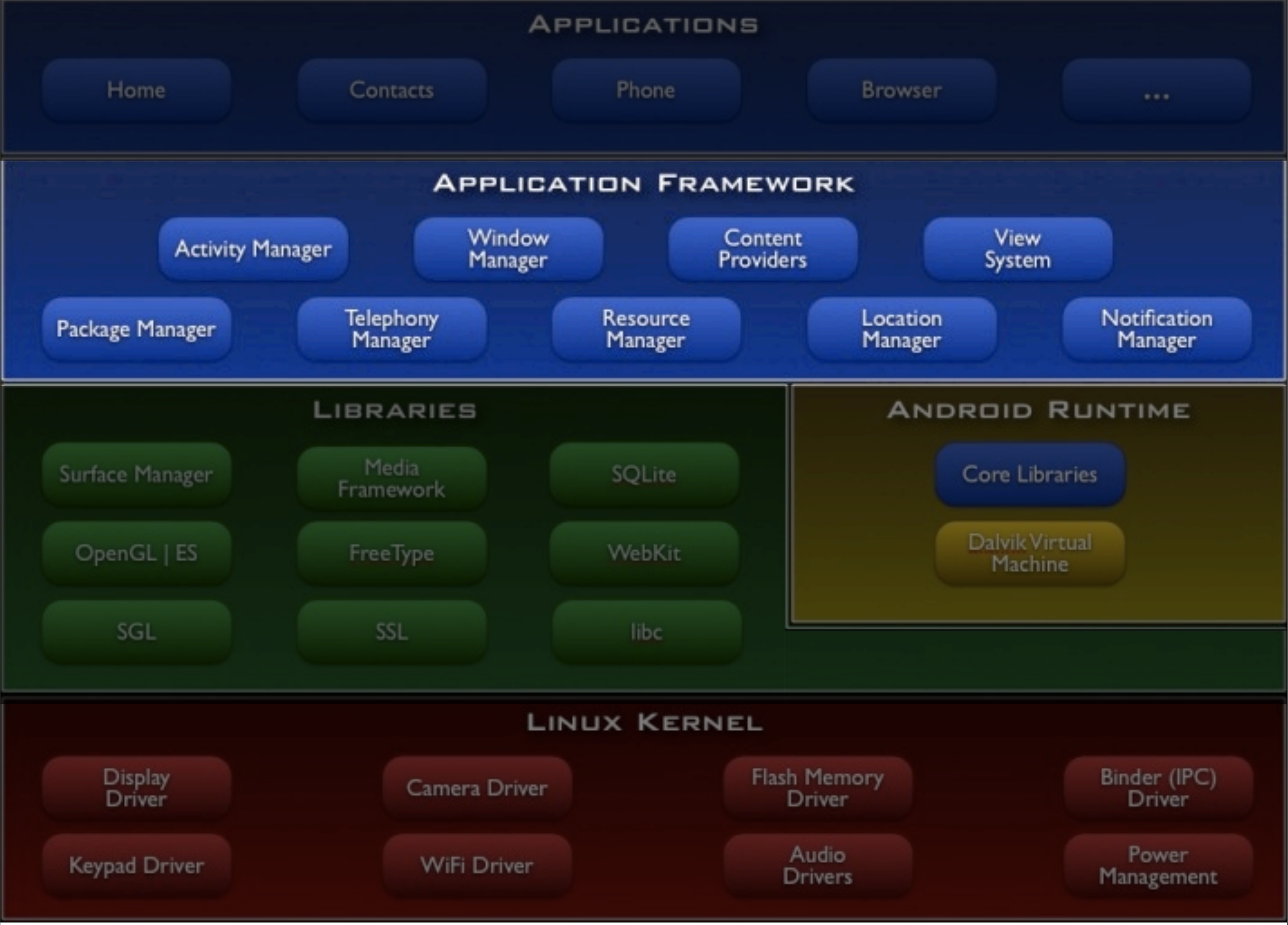
- Jan 2 - April 2008: Android Developer Challenge I
  - roughly $5.000.000 price money
- Second half 2008: Android Developer Challenge II
  - about $2.000.000 price money
- OHA spent millions of dollars to push their platform!

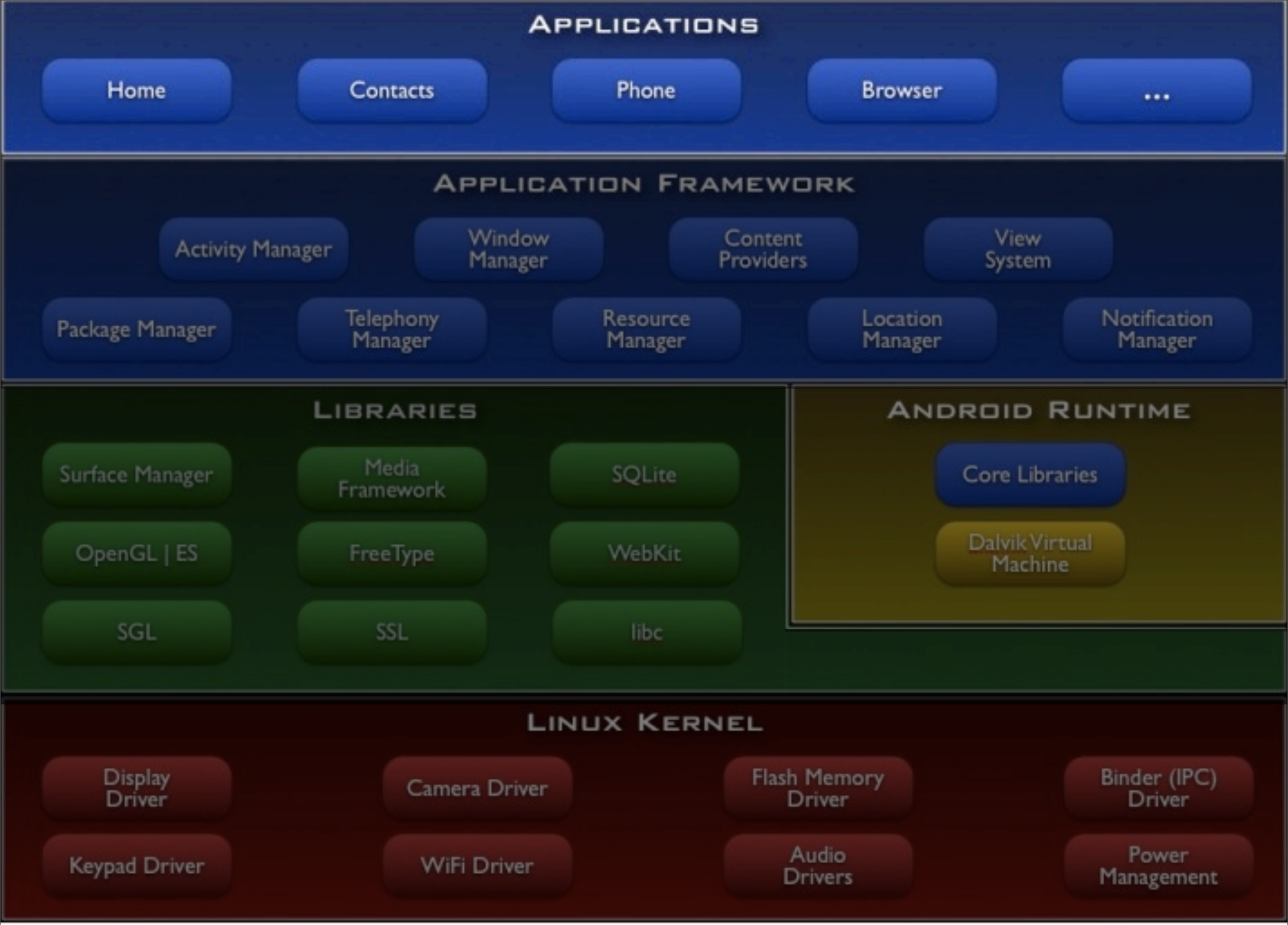

Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

Packages

- Core Packages
  - android.util
  - android.os
  - android.text
  - android.database
  - android.content
  - android.app
  - android.graphics
  - android.view
  - android.widget

App logic

UI

- Domain-specific packages
  - android.provide
  - android.telephony
  - android.webkit

Application Anatomy

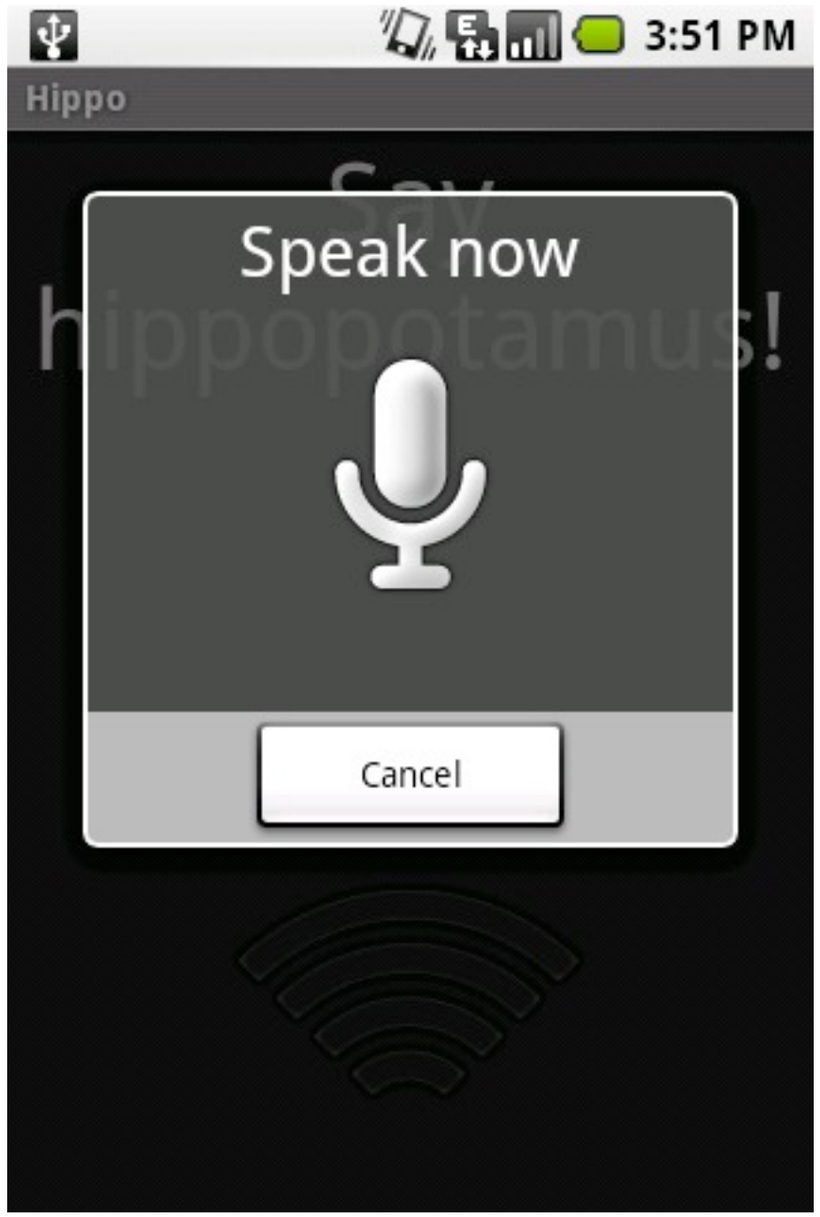
- Idea: share elements of applications
  - No single entry point
- Android applications consist of these components
  - Activities
  - Services
  - Broadcast Receivers
  - Content Providers
- Not every application needs all four

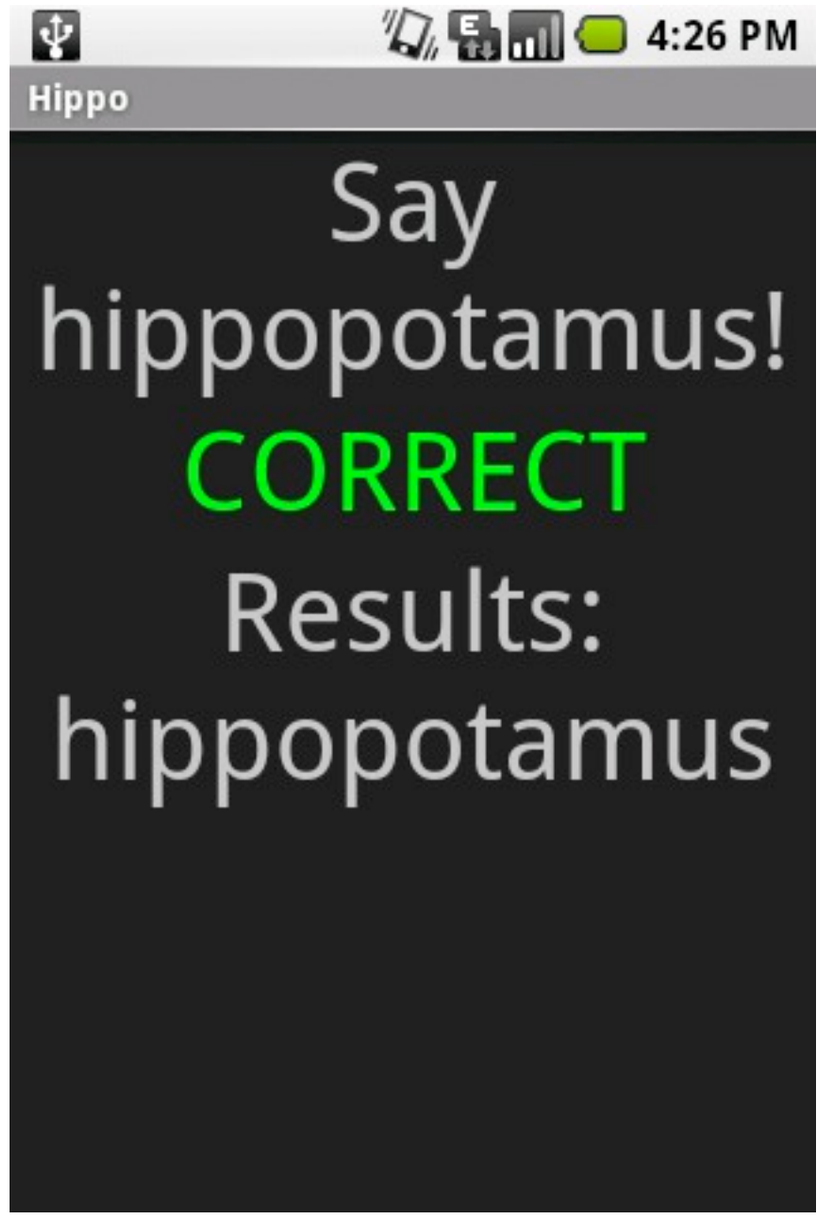
Activities

- One screen of your application's UI
- Activities publish their capabilities using an IntentFilter
- Activities are started when it receives an Intent (name + data URI)

Activities


3:51 PM
Hippo
Say
hippopotamus!

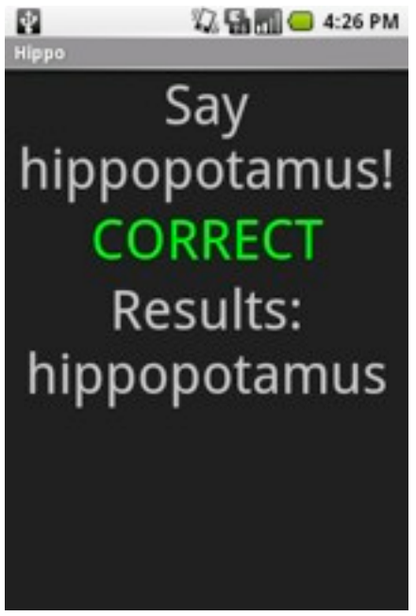

3:51 PM
Hippo
Speak now
Cancel


4:26 PM
Hippo
Say
hippopotamus!
CORRECT
Results:
hippopotamus

Activities

```
Intent intent = new Intent("android.speech.action.RECOGNIZE_SPEECH");
StartActivityForResult(intent, 0);


protected void onActivityResult int request,int result, Intent data) {
   //...code for checking result and data intent...
   ArrayList<String> recognized = data.getExtras()
                                  .getStringArrayList("results");
   //do stuff with the recognized words ...
}
```



Activities

- If a new Activity is started the old one is paused and goes onto a history stack
- Dynamic resolution of Intents makes Android extremely flexible:
  - Applications can reuse functionality from other components simply by making a request in the form of an Intent
  - Activities can be replaced at any time by a new Activity with an equivalent IntentFilter

Broadcast Receiver

- Executes in reaction to external events (even when the App is not running)
  - Incoming phone call, network becomes available, midnight
- No own UI but can use the NotificationManager for alert messages

Service

- Long-lived, non-UI code that runs in the background
  - E.g., music player
- Is kept alive by the system even if the starting Activity has finished executing
- Activities can connect to running services and talk to them through an interface exposed by the Service
  - E.g. play, pause, restart, ...

Content Provider

- Implements a standard set of methods to share data with other applications
  - Independent of how data is actually stored
- Only way to share data between packages
- Built-in providers for audio, video, images, contact information, etc.

Application Life Cycle

- Applications do not control their life cycle themselves
- System decides upon a combination of factors
  - Memory usage, importance to the user, parts of the app running
- Priority list (kill from bottom to top)
  - Foreground processes (current Activity, Service in callback)
  - Visible processes (running Activities behind dialogs)
  - Service processes
  - Background processes (stopped Activity)
  - Empty processes (only cached for faster start-up)

Demo:
Creating an Android Application

Android Evaluation

- Availability: growing fast
- Productivity: high for app functionality, medium for GUI design (no graphical editor)
- Parallelism: external: no, internal: yes up to a certain degree
- Performance: android 2.1: medium (one virtual machine per process)
  android 2.2: better because of new JIT compiler
- Graphics model: RasterOp

Android Evaluation

- Appearance: exchangeable through ‘themes’
- Extensibility: low (WS is not extensible but API is still changing)
- Adaptability: high (XML resource system)
- Resource sharing: none between applications
- Distribution: no

More information on ...

http://code.google.com/android/

iPhone

- Hardware, OS, UI and core apps from single vendor
- Introduced multi-touch to a broad audience
  - Also changed the role of the provider
- App Store concept
- Mobile Safari browser
- iPhone 4: Apple A4 CPU, Retina Display, gyroscope, two cameras

iPhone History

iPhone
June 2007
Multitouch

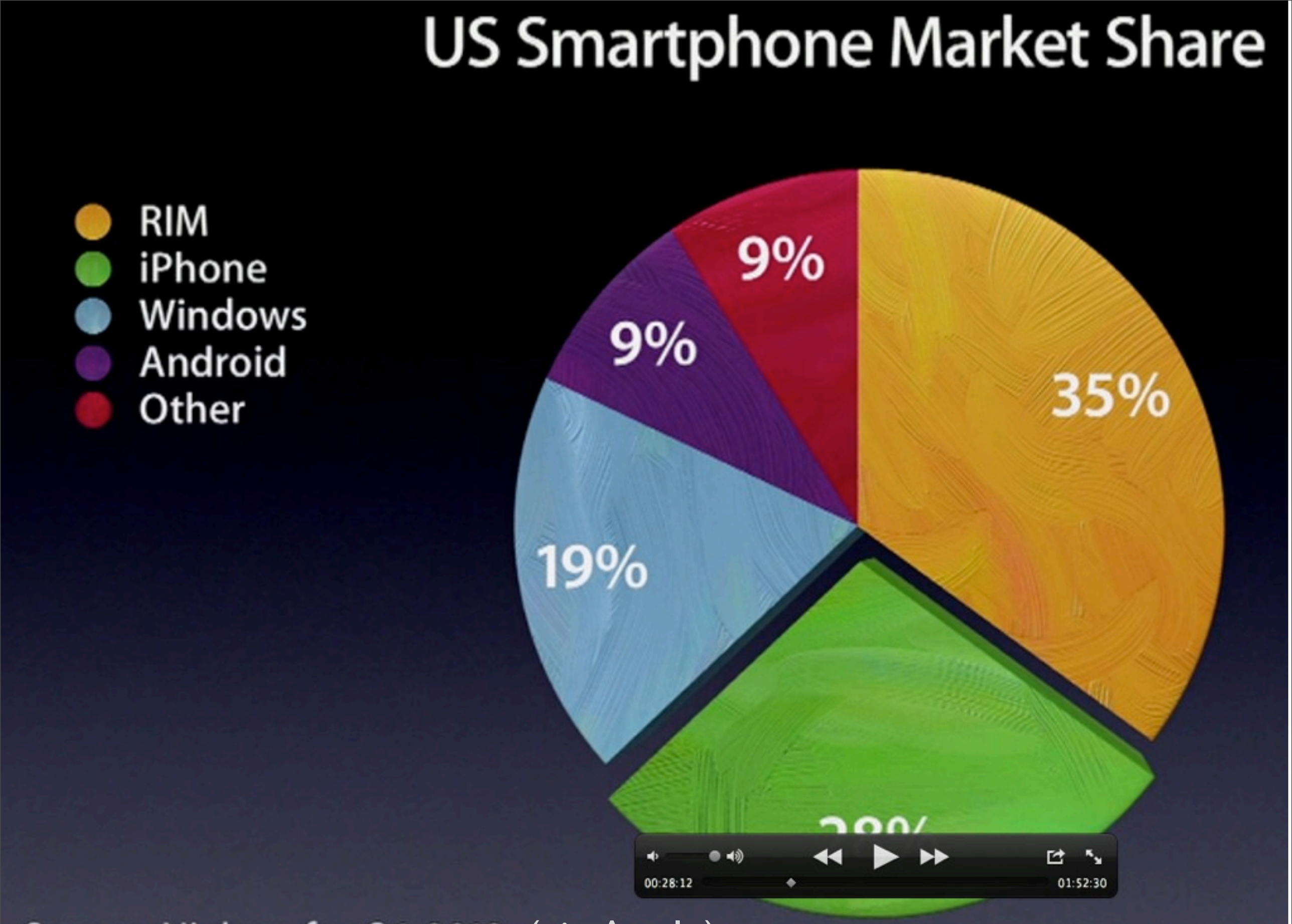
iPhone 3G
July 2008
App store

iPhone 3GS
June 2009
A/V recording

iPad
June 2007
New device class

iPhone 4
June 2010
Multitasking

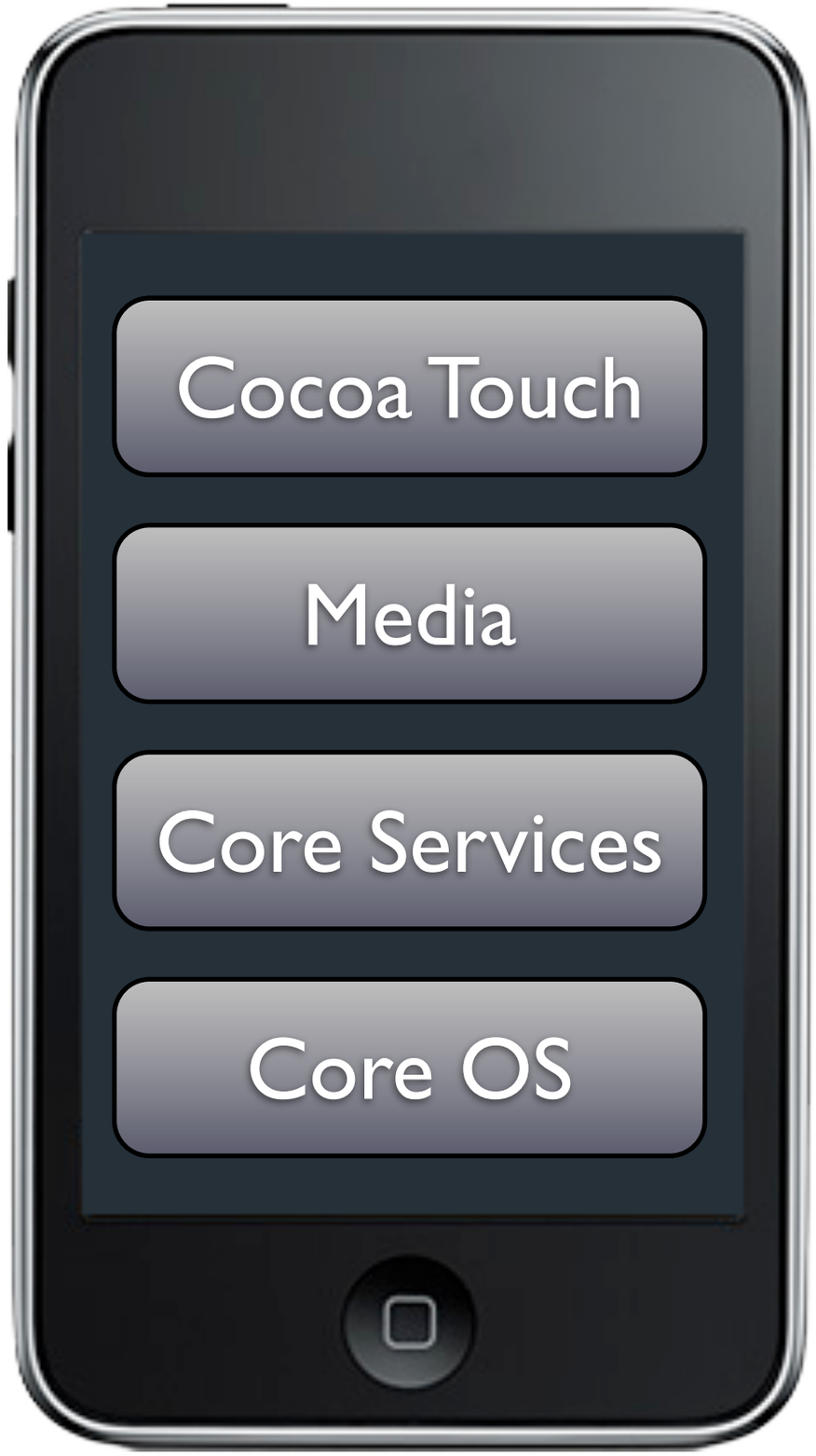
US Smartphone Market Share



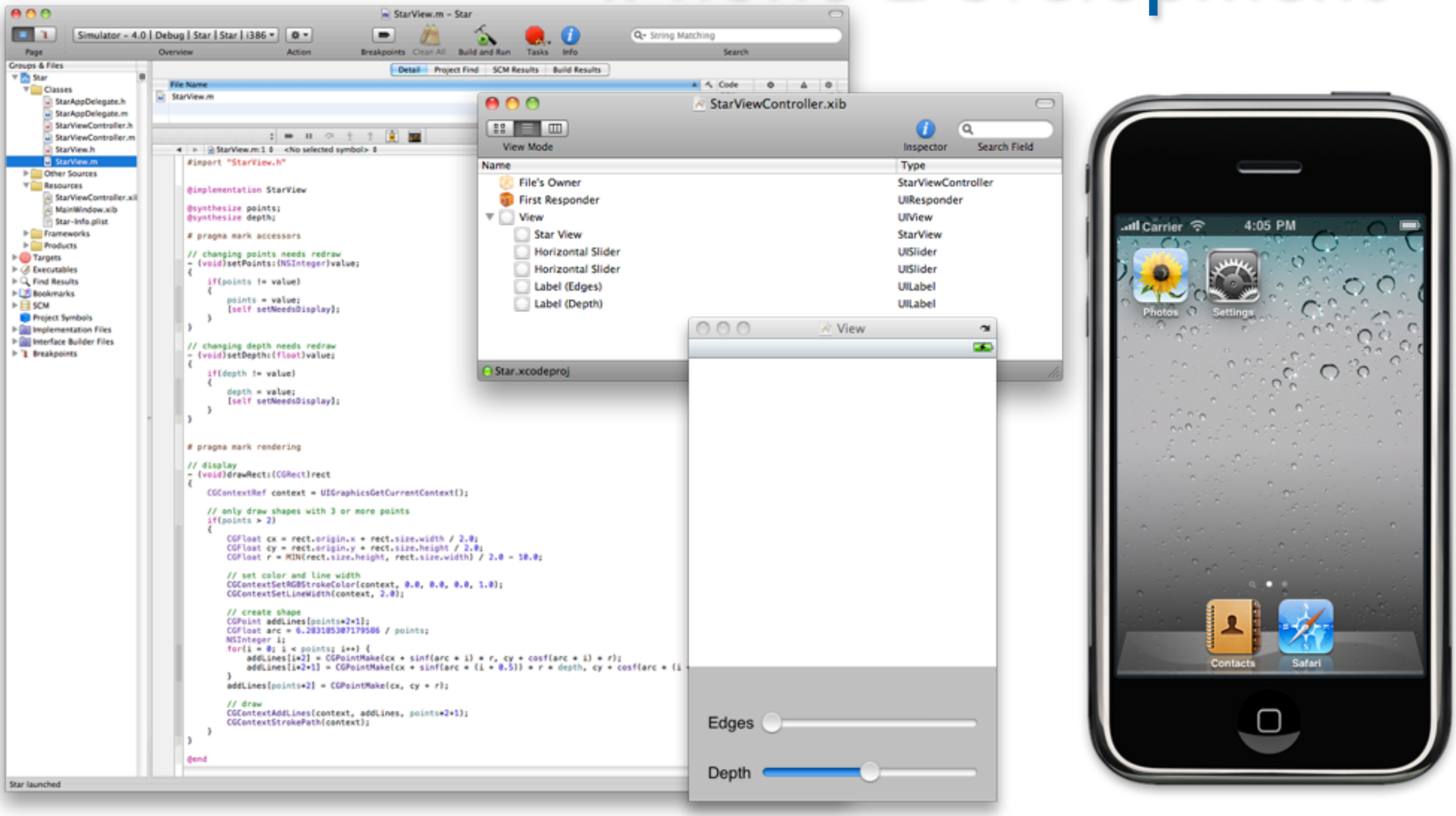
Source: Nielsen for Q1 2010 (via Apple)


Cocoa
Media
Core Services
Core OS


Cocoa
Media
Core Services
Core OS


Cocoa Touch
Media
Core Services
Core OS

iPhone Development



iPhone Development

- Generally very similar to using Foundation and AppKit on OS X
- Some distinct differences, though (AppKit)
  - Views are remodeled to work well on limited screen space
  - Event handling is concerned with touch gestures not with mouse clicks
  - Target-Action is no longer 1:1 but 1:n
  - No main menu for applications

iPhone Development

- Differences in the foundation framework
  - No Cocoa bindings
  - No garbage collection
  - No distributed objects
- Frameworks are adapted for the mobile platform

iPhone Development

Core Data

WebKit

GameKit

GameCenter

StoreKit

Core Location

CoreMotion

Graphics

EventKit

Core Audio

Multitasking

Push Notifications

Bonjour

Multitouch Handling

- Track a set of touches
  - 5 touches on iPhone
  - 10 touches on iPad

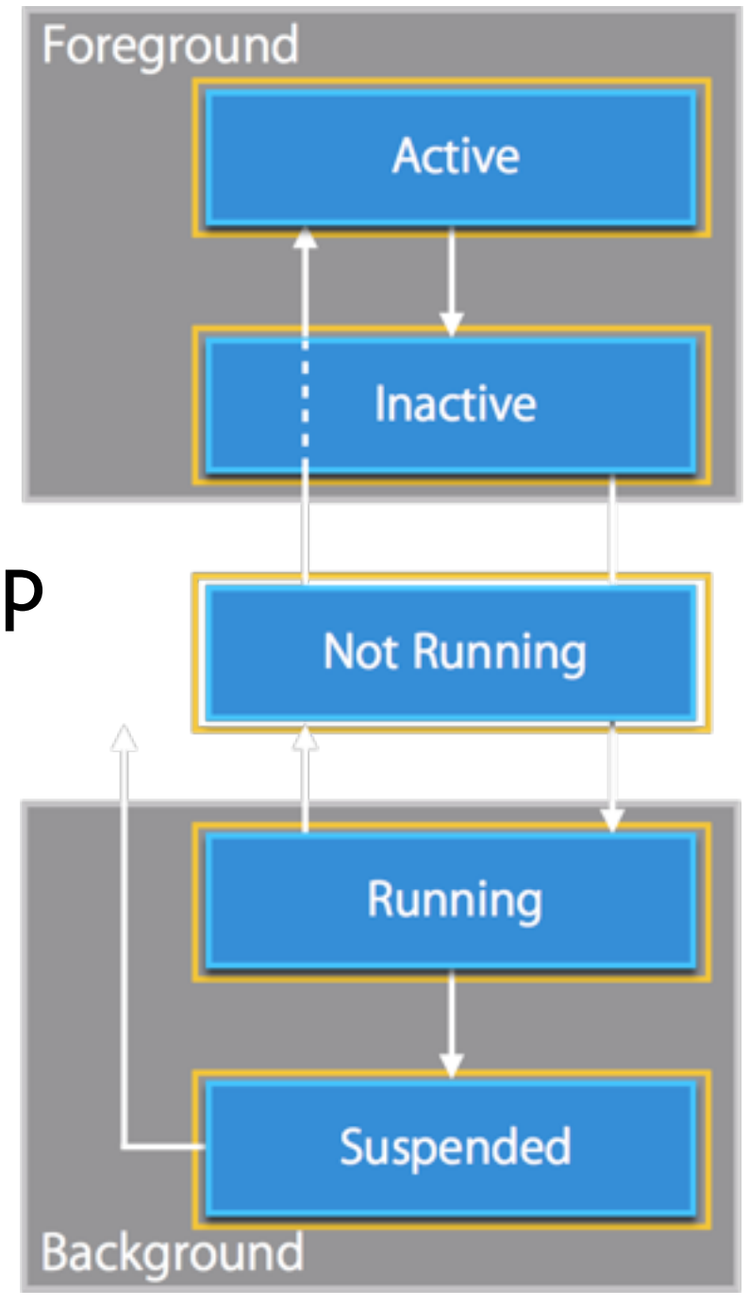
- Event defines the type: touch or motion

```
-(void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
-(void)touchesCancelled:(NSSet *)touches withEvent:(UIEvent *)event
-(void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event
-(void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event
```

Gesture Recognition

- UIGestureRecognizer
- Provides six default gesture recognition patterns
  - Tapping (any number of taps)
  - Pinching in and out
  - Panning or dragging
  - Swiping
  - Rotating
  - Long press
- Simple interface to create your own recognizer

Multitasking in iOS 4



- No implicit termination by the user
- The OS decides when to terminate an app
  - Mostly depending on memory footprint
- The app is not aware of the termination
  - Dramatic change in programming paradigm

Demo:
Creating an iPhone Application

iPhone OS Development

- Different iPhone OS devices have different screen resolutions
  - iPhone 3GS, iPhone 4, iPad
- One app-bundle for all devices (fat binary)
  - Only UI-related parts have to be modified (MVC)
  - Store different XIB files and resources

iPhone OS SDK

- Public frameworks available to developer
- However, not all the phone's functionality is accessible
  - Keep user experience consistent
  - Avoid harmful behavior
  - Limits the possibilities for exciting apps
- HTML 5 vs. Adobe Flash
  - Who defines the standards
  - New ideas, new technologies

Evaluation: iPhone

- Availability: medium - runs on every iPhone OS device but nowhere else
- Productivity: very high for a mobile platform! If you know Cocoa you are almost ready to go
- Parallelism: limited, running applications in parallel is possible, background computation is restricted.
- Performance: tailored to a small set of hardware platforms so performance is good

Evaluation: iPhone

- Graphics model: point/pixel-based
- Appearance: fixed to the style of UIKit
- Extensibility: like Mac OS X
- Adaptability: high, by modifying xib files
- Resource sharing: system resources are implicitly shared
- Distribution: no

"*The first time in many years that coding for a mobile device was actually **fun** again.*" — *Newton/Palm/Pocket PC/Windows Mobile/iPhone developer, WWDC lunch talk*