Embedded Operating Systems

Che-Wei Chang

chewei@mail.cgu.edu.tw

Department of Computer Science and Information Engineering, Chang Gung University

Linux Environment

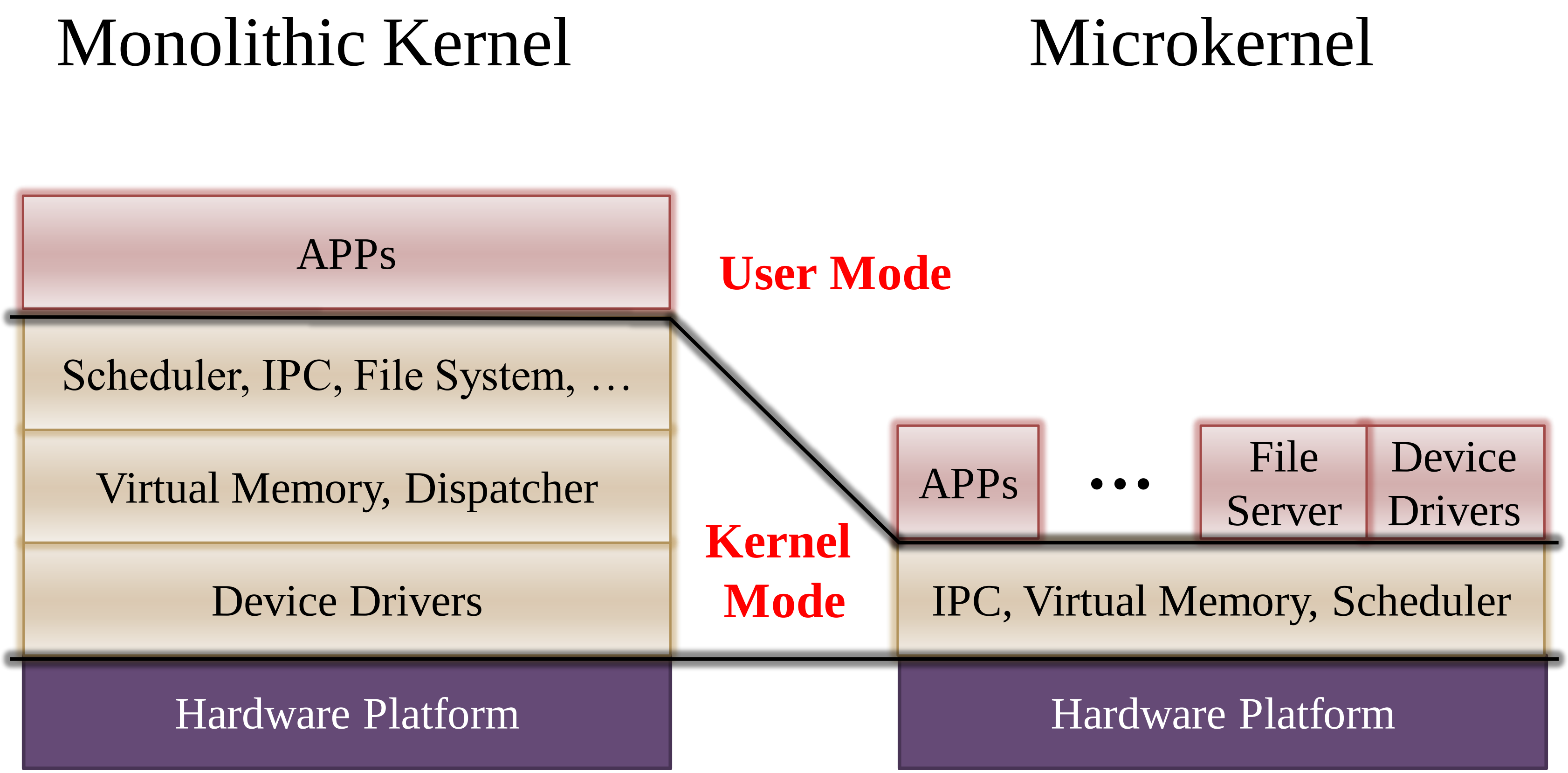
Advantages of Linux

- Linux is free— both in source code and cost, due to the GPL
- Linux is fully customizable in all its components
- Linux can runs on low-end, inexpensive hardware (HW) platforms, e.g., one with 4 MB RAM
- Most Linux systems are stable
- The Linux kernel can be very small and compact
- Linux is highly compatible with many common applications and functions
- Linux is well-supported

Different Type of Operating System Kernels

- Monolithic kernel
  - The entire operating system is working in kernel space
  - All parts of the kernel share the same kernel-level memory
  - Kernel components might affect other components
  - The Linux kernel is an example
- Microkernel
  - Kernel functions are partitioned into components
  - Communications are via inter process communication (IPC) protocol
  - The L4 microkernel is an example

Monolithic Kernel and Microkernel



Device Driver

- Character Devices
  - Sequential access
  - Examples might include printers, scanners, sound boards
  - The same device may have both block and character oriented interfaces
- Block Devices
  - Block devices can support filesystem
  - The block size is from 512B to 4KB and is going to increase in advanced devices
  - For example, disks are commonly implemented as block devices

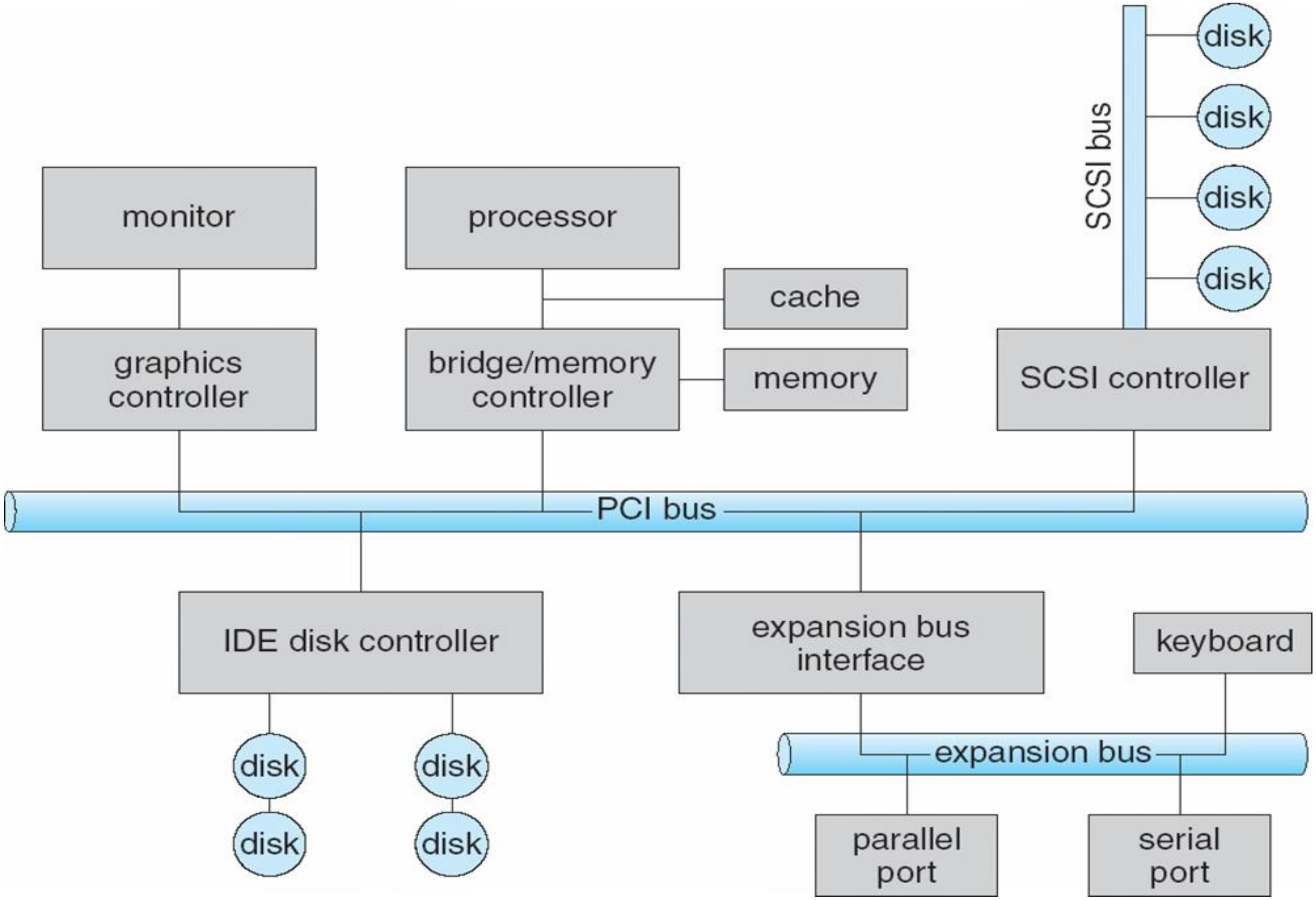
Major and Minor Numbers

- Major number
  - Each device driver is identified by a unique major number
  - This number is assigned by the Linux Device Registrar
- Minor number
  - The number uniquely identifies a particular instance of a device of the same device type
  - If there are three devices with the same device driver, they should have the same major number but different minor numbers
- Command: mknod [device name][bcp] [Major] [Minor]
  - b: block devices
  - c: character devices
  - p: a FIFO file

I/O Hardware

- Incredible Variety of I/O Devices
  - Storage
  - Transmission
  - Human-interface
  - …
- Common Concepts
  - Port: a connection point for a device
  - Bus: can be daisy chain or shared direct access
  - Controller (host adapter): electronics that operate ports, buses, devices

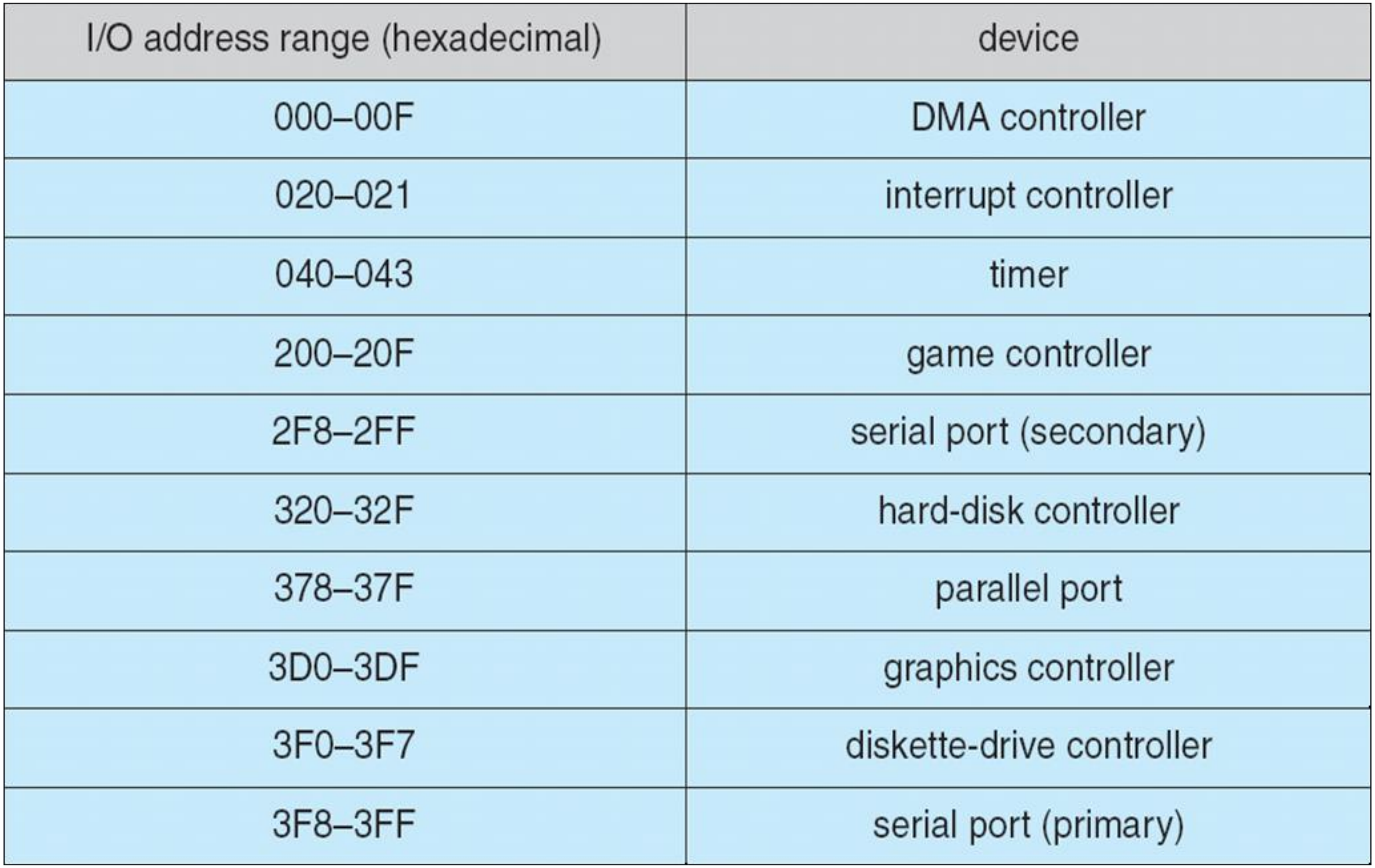
Typical PC Bus Structure



Access to I/O Hardware

- Devices registers which can be accessed by the host
  - The data-in register is read by the host to get the input
  - The data-out register is written by the host to send the output
  - The status register contains bits which indicate device states
  - The control register is written by the host to send commands
- Methods to access devices with their addresses
  - Direct I/O instructions
  - Memory-mapped I/O
    - Device data and command registers mapped to processor address space
    - Especially for large address spaces (graphics)

Device I/O Port Locations on PCs (Partial)



I/O address range (hexadecimal)	device
000–00F	DMA controller
020–021	interrupt controller
040–043	timer
200–20F	game controller
2F8–2FF	serial port (secondary)
320–32F	hard-disk controller
378–37F	parallel port
3D0–3DF	graphics controller
3F0–3F7	diskette-drive controller
3F8–3FF	serial port (primary)

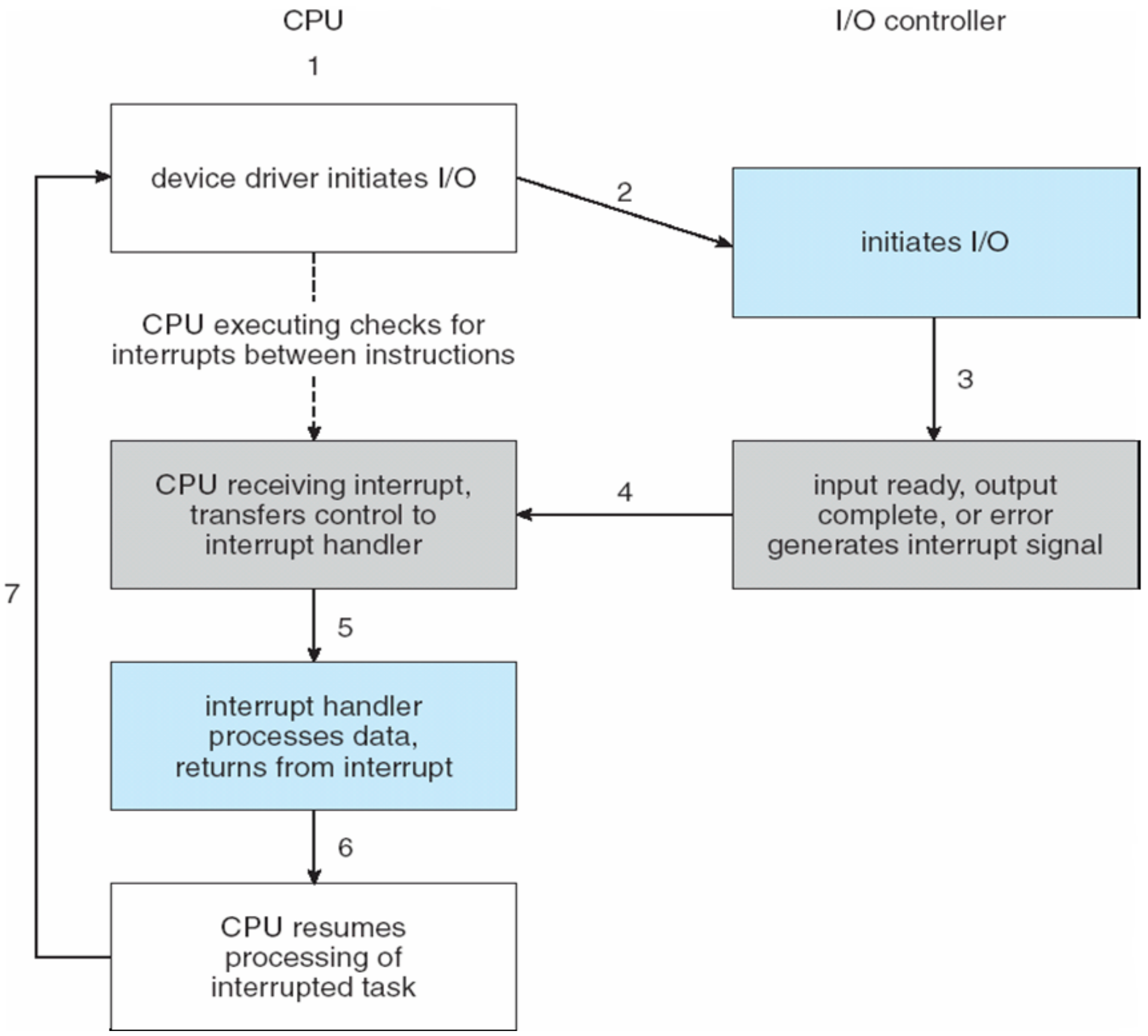
Polling

- An example of polling I/O
  1. Host reads the busy bit from the status register until 0
  2. Host sets read or write bit and copies data into data-out register if it is going to write data
  3. Host sets command-ready bit
  4. Controller sets busy bit, executes the transmission
  5. Controller clears busy bit, error bit, and command-ready bit when the transmission is done
- Step 1 is busy-waiting to wait for I/O from devices
  - Reasonable if device is fast
  - But inefficient if device is slow
    - CPU switches to other tasks?
      - Might miss some data

Interrupts

- CPU interrupt-request line triggered by I/O device
  - Checked by processors (hardware) after each instruction
- Interrupt handler receives interrupts
  - Masked to ignore or delay some interrupts
- Interrupt vector table is used to dispatch interrupt to correct handler
  - Context switches at start and end
  - Based on priority
  - Some nonmaskable
  - Interrupt chaining if more than one device at the same interrupt number

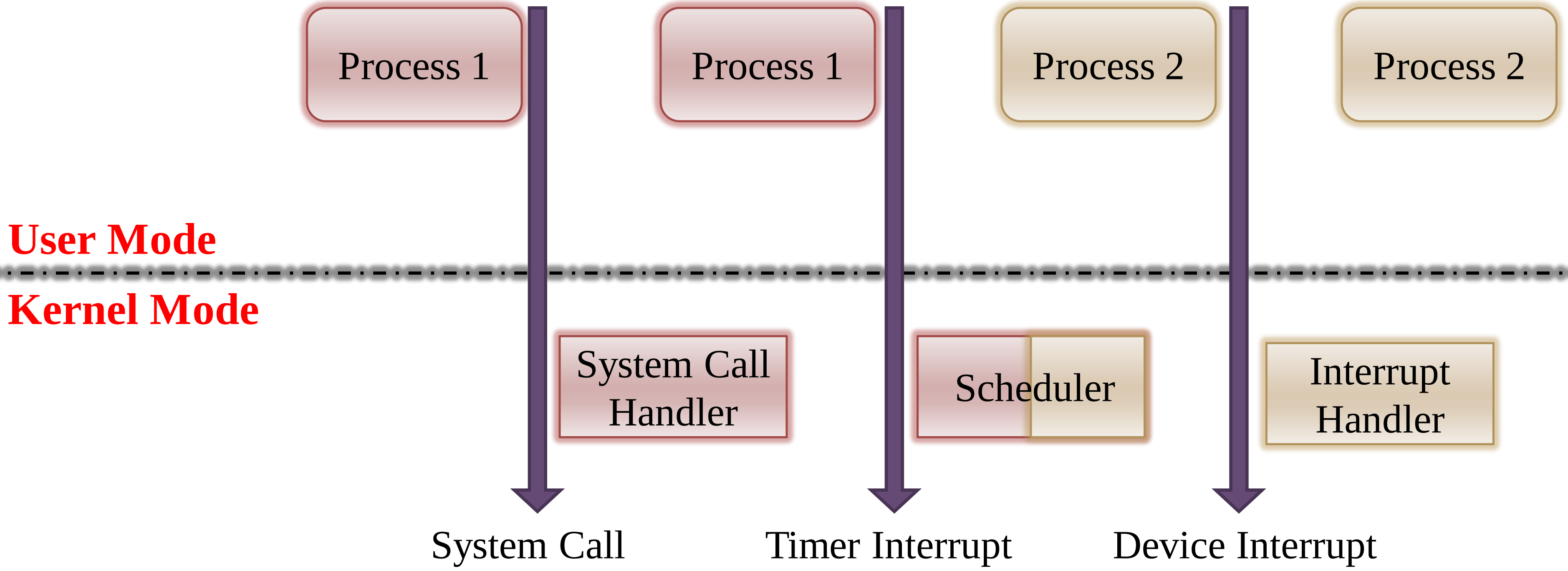
Interrupt-Driven I/O Cycle


CPU
I/O controller
1
device driver initiates I/O
2
initiates I/O
CPU executing checks for interrupts between instructions
3
CPU receiving interrupt, transfers control to interrupt handler
4
input ready, output complete, or error generates interrupt signal
5
interrupt handler processes data, returns from interrupt
6
CPU resumes processing of interrupted task
7

Interrupt Usage

- Interrupt vector table is used to identify which device sent out the interrupt
  - When multiple devices share an interrupt number, the handlers are checked one by one
- Interrupt mechanism is also used for exceptions
  - Terminate process or crashed subsystem due to hardware error
  - Page fault executes when there is some memory access error
  - System call executes via a trap to trigger the kernel to execute some request
- Multi-CPU systems can process interrupts concurrently
  - If operating system designed to handle it

Transitions between User and Kernel Modes in Linux



Process 1

Process 1

Process 2

Process 2

User Mode

Kernel Mode

System Call Handler

Scheduler

Interrupt Handler

System Call

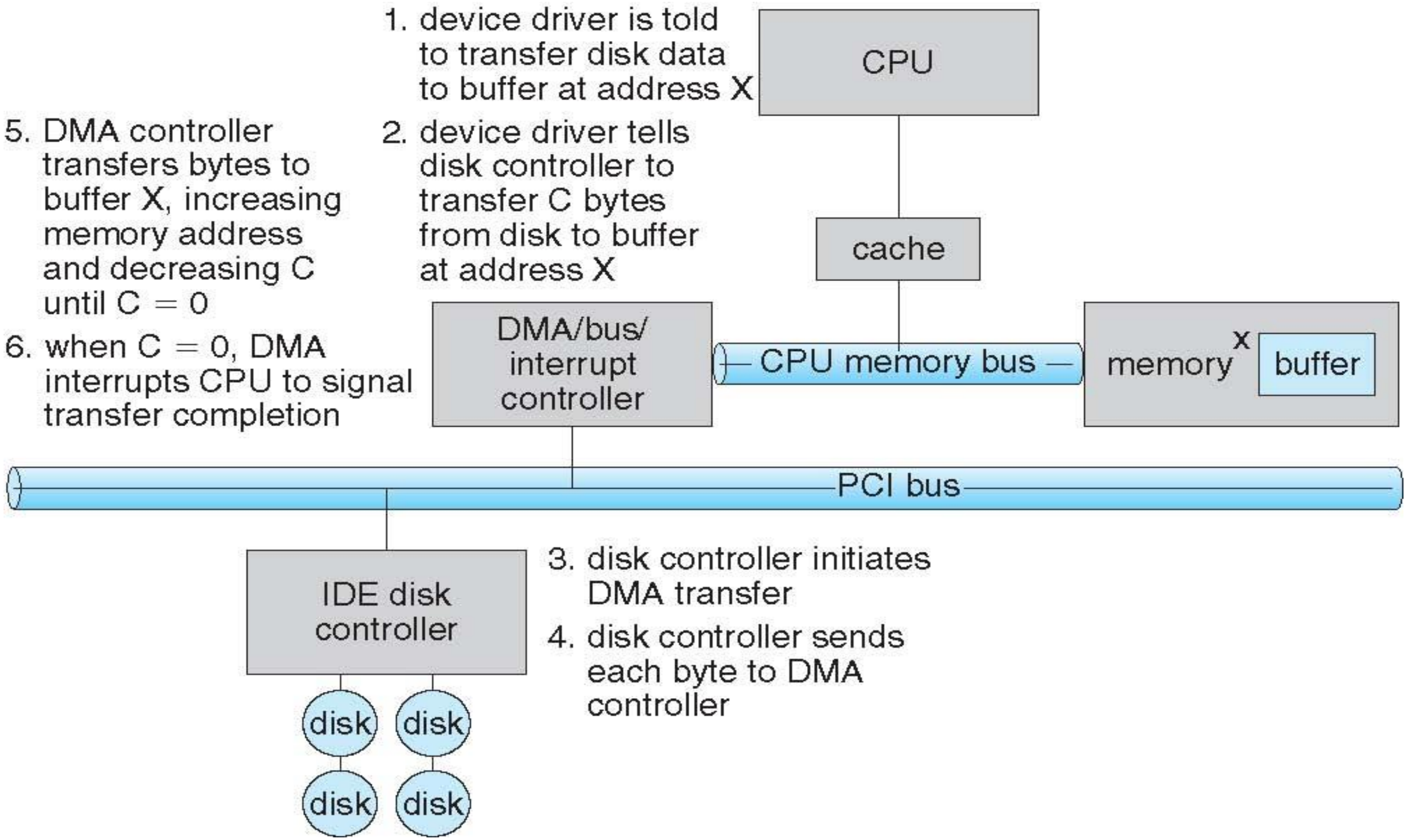
Timer Interrupt

Device Interrupt

Direct Memory Access

- Used to avoid programmed I/O (one byte at a time) for large data movement
- Requires a DMA controller
- Bypasses the CPU to transfer data directly between I/O devices and memory
- OS writes a DMA command block into memory
    - Source and destination addresses
    - Read or write mode
    - Number of bytes
- OS writes the location of the command block to the corresponding DMA controller
    - Bus mastered by the DMA controller – grabs bus from CPU
    - When transmission is done, the DMA controller sends an interrupt

DMA Transfer



1. device driver is told to transfer disk data to buffer at address X
2. device driver tells disk controller to transfer C bytes from disk to buffer at address X
3. disk controller initiates DMA transfer
4. disk controller sends each byte to DMA controller
5. DMA controller transfers bytes to buffer X, increasing memory address and decreasing C until C = 0
6. when C = 0, DMA interrupts CPU to signal transfer completion

Getting Started

- Installing Linux is now easier than installing MS Windows
- Doing it on a virtual machine can be harmless
- Many distributions are there for you

- Which Linux distribution is better?
  - If you ask this question, it means “it doesn't matter for you”
  - Just use the distribution with the most supports you can find

Android Environment

History of Android

- Android was founded in Palo Alto, California in October 2003
- Google acquired Android in August 2005
- The Open Handset Alliance started in November 2007
- The first commercially available smartphone running Android was the HTC Dream, released on October 22, 2008
- The latest released version is Android Pie 9.0.0, which was released on August 6, 2018

Android Versions

Cupcake
Android 1.5

Donut
Android 1.6

Eclair
Android 2.0/2.1

Froyo
Android 2.2

Honeycomb
Android 3.0-3.2

Ice cream Sandwich
Android 4.0+

Jelly Bean
Android 4.1-4.3

KitKat
Android 4.4

Lollipop
Android 5.0-5.1

Marshmallow
Android 6.0-6.0.1

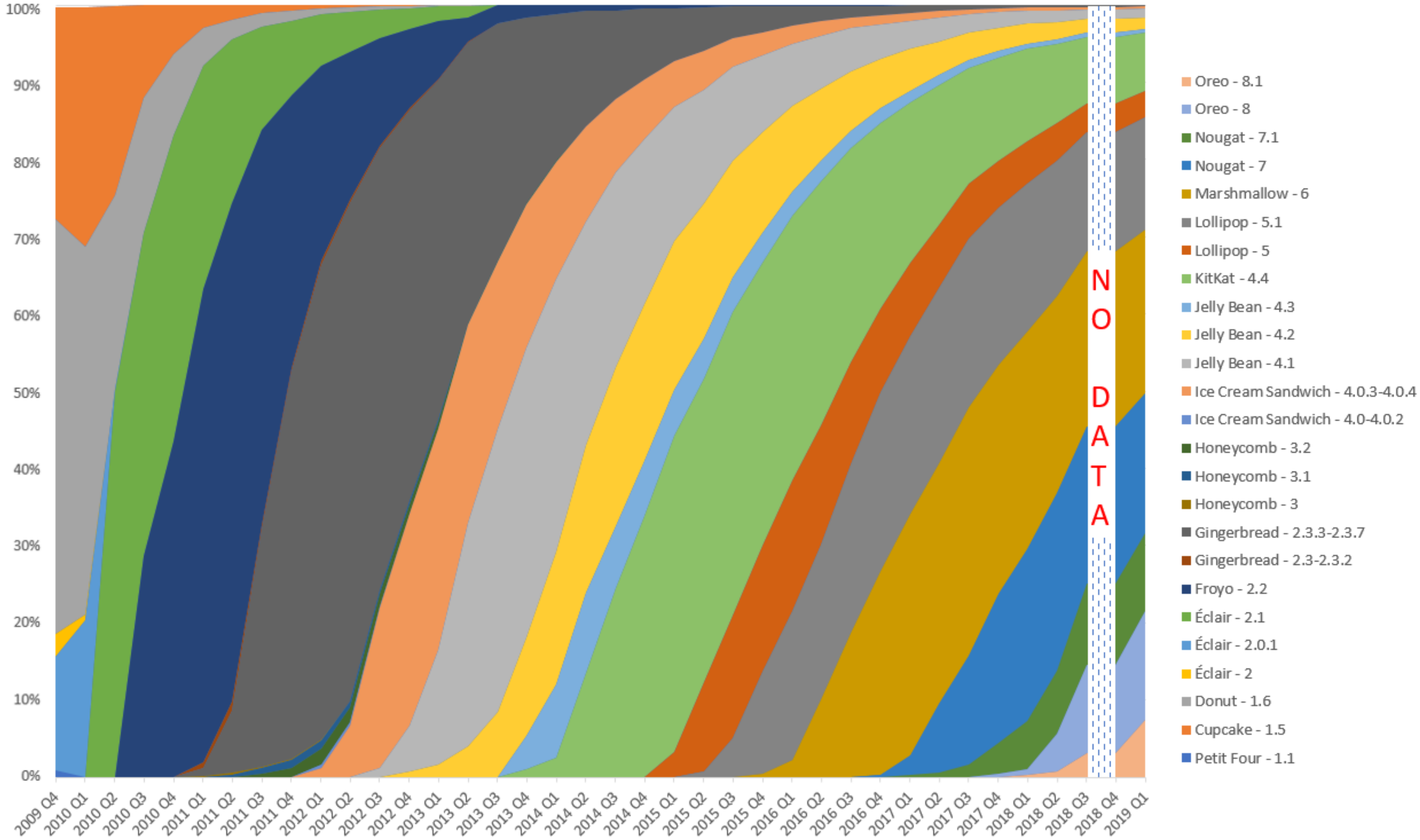
Nougat
Android 7.0 – 7.1.2

Oreo
Android 8.0 - 8.1

Pie
Android 9.0

Android 10
Android 10

Android Distribution

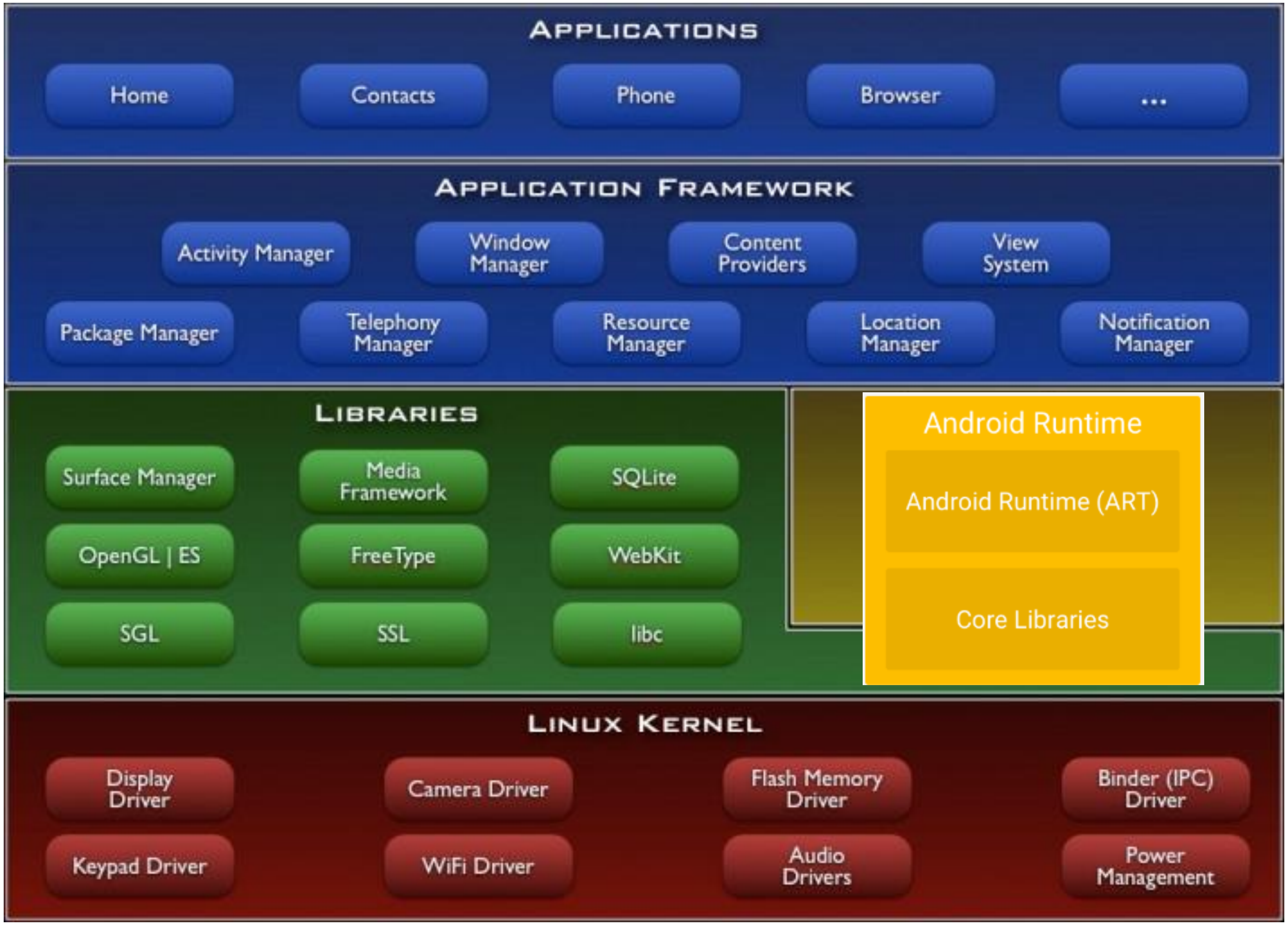


Source: https://en.wikipedia.org/wiki/Android_version_history

Google Android

- A software stack for mobile devices
  - An operating system
  - Middleware
  - Key Applications
- Linux for core system services
  - Security
  - Memory management
  - Process management
  - Power management
  - Hardware drivers

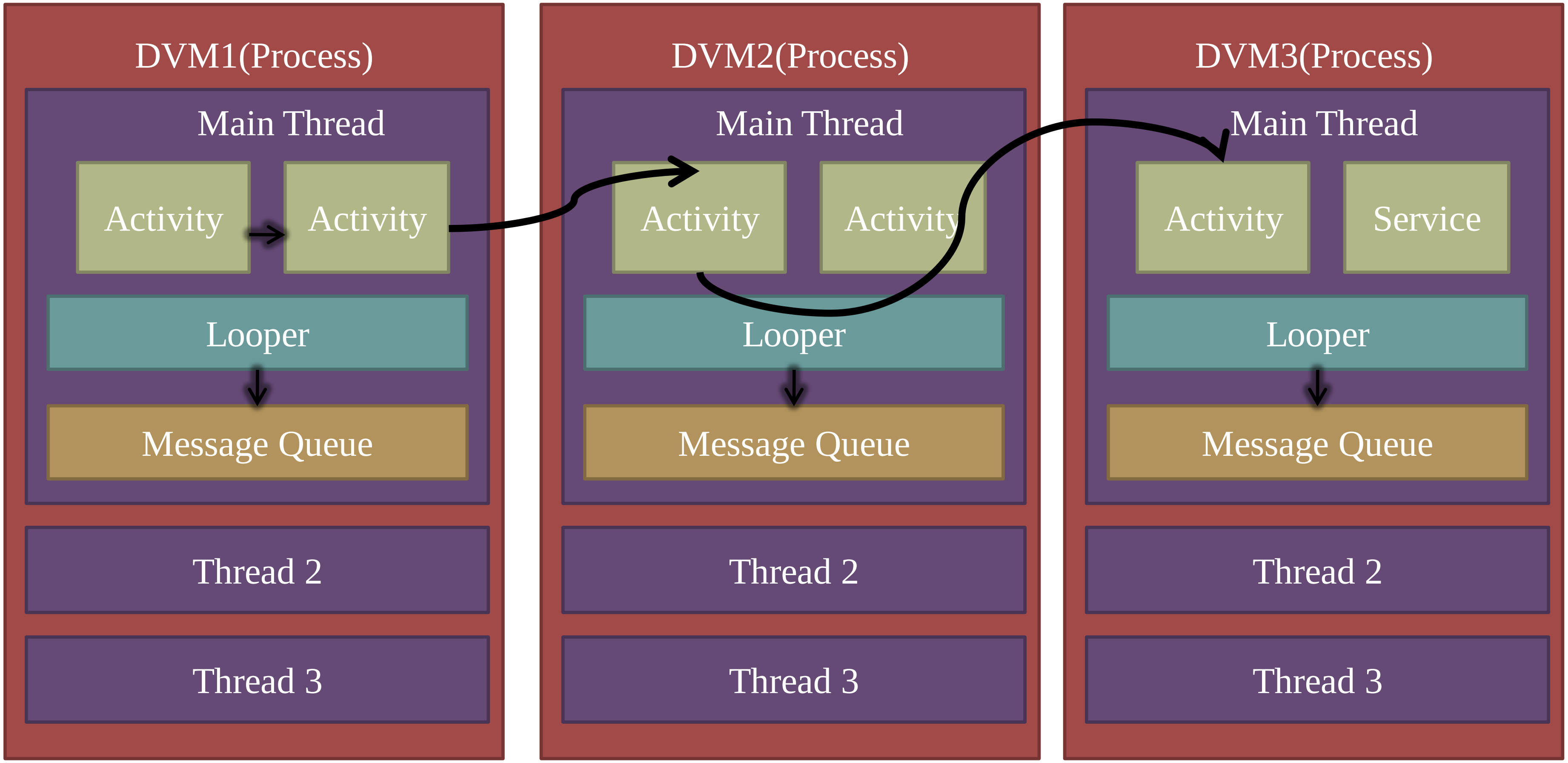
Android Architecture



Mobile Devices

- Advantages
  - Always with the user
  - Typically have Internet access
  - Typically GPS enabled
  - Typically have accelerometer & compass
  - Most have cameras & microphones
- Disadvantages
  - Limited screen size
  - Limited battery life
  - Limited processor speed
  - Limited web browser functionality

Android Applications

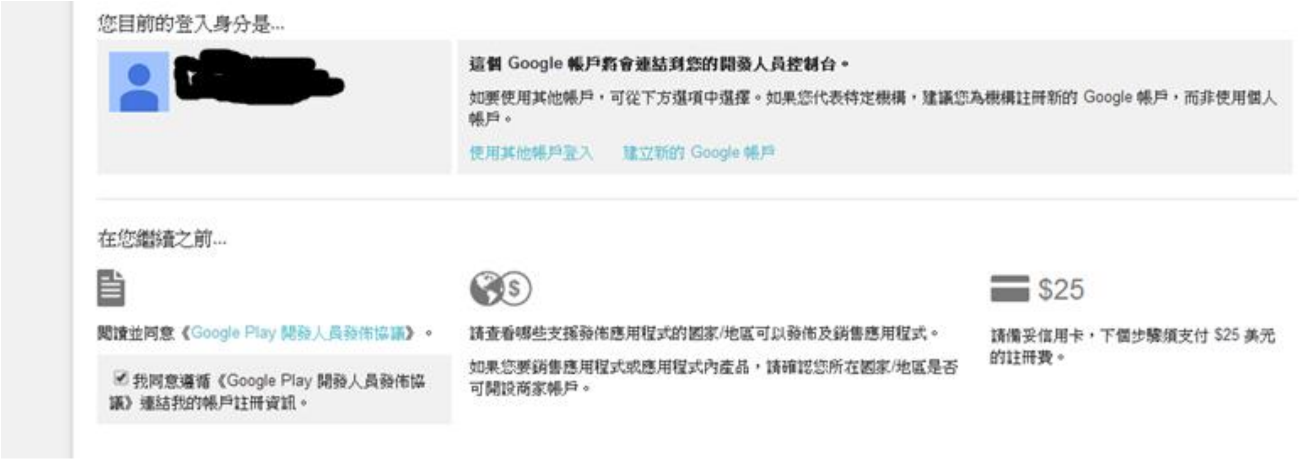


Android Market / Google Play

- Has various categories, allows ratings
- Have both free/paid apps
- Featured apps on web and on phone
- Initial release: October 23, 2008, as Android Market
- Development status:
  - 1+ million apps, as of July, 2013
  - 1.3+ million apps, as of July, 2014
  - 1.5+ million apps, as of Q1, 2015
  - 1.9+ million apps, as of Q1, 2016
  - 2.7+ million apps, as of Q1, 2017

Publish Your APP

- Link to an Account
  - Developer Account: $25 fee
  - Link to your checking account
  - Developer take 70% of app purchase price



您目前的登入身分是...

[redacted]

這個 Google 帳戶將會連結到您的開發人員控制台。

如要使用其他帳戶，可從下方選項中選擇。如果您代表特定機構，建議您為機構註冊新的 Google 帳戶，而非使用個人帳戶。

使用其他帳戶登入　建立新的 Google 帳戶

在您繼續之前...

閱讀並同意《Google Play 開發人員發佈協議》。

☑ 我同意遵循《Google Play 開發人員發佈協議》連結我的帳戶註冊資訊。

請查看哪些支援發佈應用程式的國家/地區可以發佈及銷售應用程式。

如果您要銷售應用程式或應用程式內產品，請確認您所在國家/地區是否可開設商家帳戶。

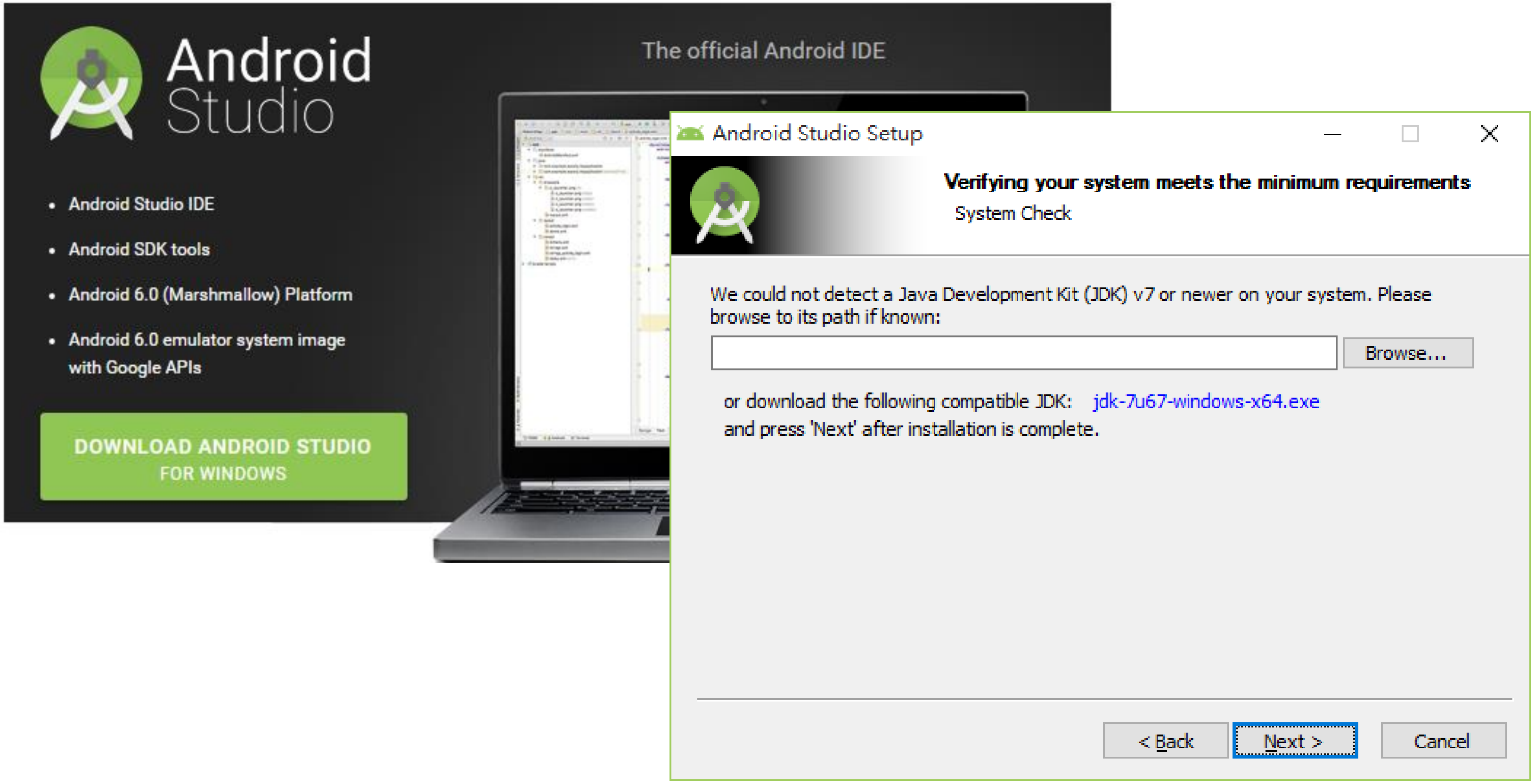
$25

請備妥信用卡，下個步驟須支付 $25 美元的註冊費。

Android Environment

- Eclipse + ADT (Android Developer Tools) Plugin
- Android SDK (System Development Kit) Tools
- Android Platform-Tools
- The Latest Android Platform Configuration
- The Latest Android System Image for the Emulator

Android Studio



- Android Studio IDE
- Android SDK tools
- Android 6.0 (Marshmallow) Platform
- Android 6.0 emulator system image with Google APIs

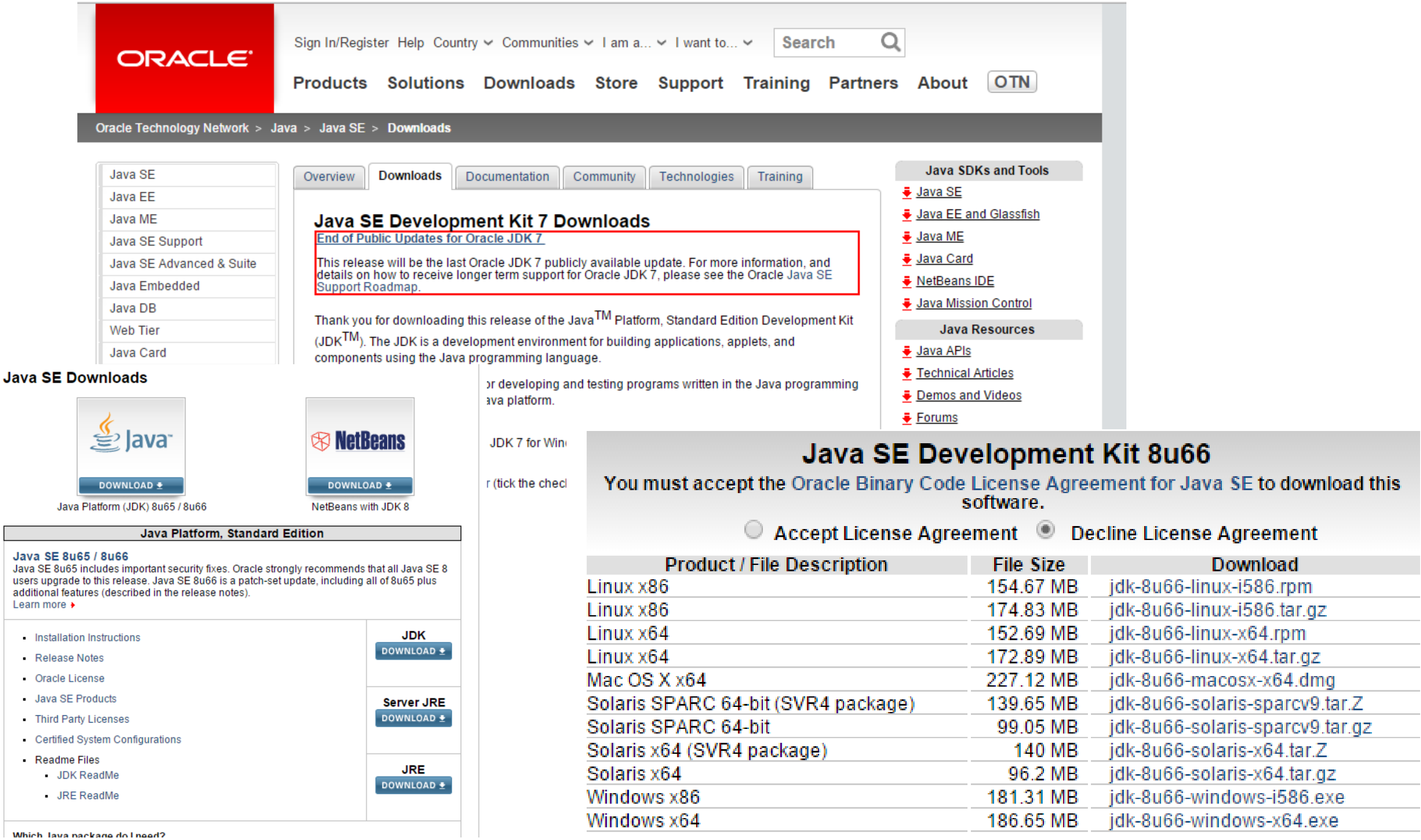
Verifying your system meets the minimum requirements

System Check

We could not detect a Java Development Kit (JDK) v7 or newer on your system. Please browse to its path if known:

or download the following compatible JDK: jdk-7u67-windows-x64.exe
and press 'Next' after installation is complete.

Java Development Kit



Java SE Development Kit 7 Downloads

End of Public Updates for Oracle JDK 7

This release will be the last Oracle JDK 7 publicly available update. For more information, and details on how to receive longer term support for Oracle JDK 7, please see the Oracle Java SE Support Roadmap.

Thank you for downloading this release of the Java^{TM} Platform, Standard Edition Development Kit (JDK^{TM}). The JDK is a development environment for building applications, applets, and components using the Java programming language.

›r developing and testing programs written in the Java programming
›va platform.

JDK 7 for Win

r (tick the checl

Java SE Downloads

Java Platform (JDK) 8u65 / 8u66

NetBeans with JDK 8

Java Platform, Standard Edition

Java SE 8u65 / 8u66
Java SE 8u65 includes important security fixes. Oracle strongly recommends that all Java SE 8 users upgrade to this release. Java SE 8u66 is a patch-set update, including all of 8u65 plus additional features (described in the release notes).
Learn more

- Installation Instructions
- Release Notes
- Oracle License
- Java SE Products
- Third Party Licenses
- Certified System Configurations
- Readme Files
  - JDK ReadMe
  - JRE ReadMe

JDK DOWNLOAD

Server JRE DOWNLOAD

JRE DOWNLOAD

Which Java package do I need?

Java SE Development Kit 8u66

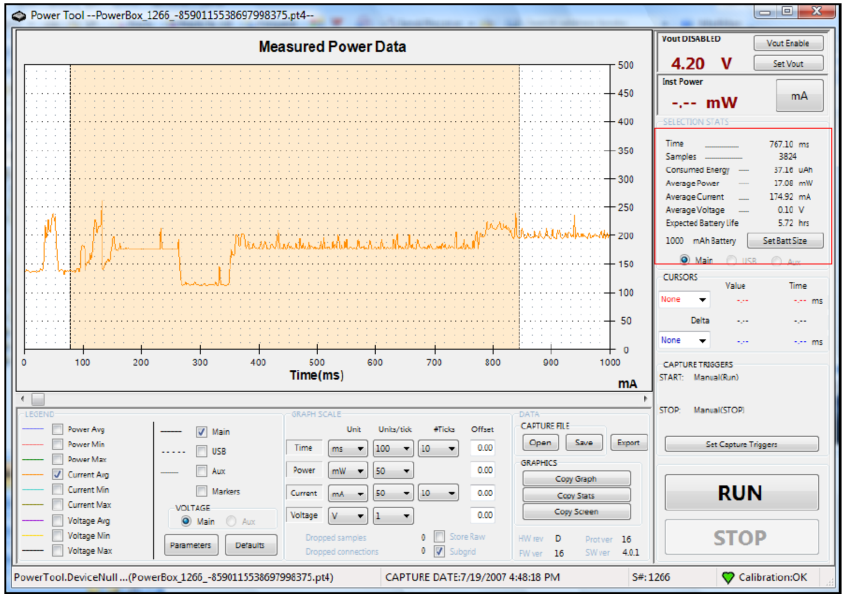
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

○ Accept License Agreement ● Decline License Agreement

Product / File Description	File Size	Download
Linux x86	154.67 MB	jdk-8u66-linux-i586.rpm
Linux x86	174.83 MB	jdk-8u66-linux-i586.tar.gz
Linux x64	152.69 MB	jdk-8u66-linux-x64.rpm
Linux x64	172.89 MB	jdk-8u66-linux-x64.tar.gz
Mac OS X x64	227.12 MB	jdk-8u66-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	139.65 MB	jdk-8u66-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.05 MB	jdk-8u66-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	140 MB	jdk-8u66-solaris-x64.tar.Z
Solaris x64	96.2 MB	jdk-8u66-solaris-x64.tar.gz
Windows x86	181.31 MB	jdk-8u66-windows-i586.exe
Windows x64	186.65 MB	jdk-8u66-windows-x64.exe

Power Monitor

- Power measurement for any device with a single lithium battery


Power Tool --PowerBox_1266_-8590115538697998375.pt4--

Vout DISABLED

4.20 V

Inst Power

-.-- mW

SELECTION STATS

Time	767.10 ms
Samples	3824
Consumed Energy	37.16 uAh
Average Power	17.08 mW
Average Current	174.92 mA
Average Voltage	0.10 V
Expected Battery Life	5.72 hrs

1000 mAh Battery

● Main ○ USB ○ Aux

CURSORS

	Value	Time
None	-.--	-.-- ms
Delta	-.--	-.--
None	-.--	-.-- ms

CAPTURE TRIGGERS

START: Manual(Run)

STOP: Manual(STOP)

LEGEND

□ Power Avg
□ Power Min
□ Power Max
☑ Current Avg
□ Current Min
□ Current Max
□ Voltage Avg
□ Voltage Min
□ Voltage Max

☑ Main
□ USB
□ Aux
□ Markers

VOLTAGE
● Main ○ Aux

GRAPH SCALE

	Unit	Units/tick	#Ticks	Offset
Time	ms	100	10	0.00
Power	mW	50		0.00
Current	mA	50	10	0.00
Voltage	V	1		0.00

Dropped samples 0 □ Store Raw
Dropped connections 0 ☑ Subgrid

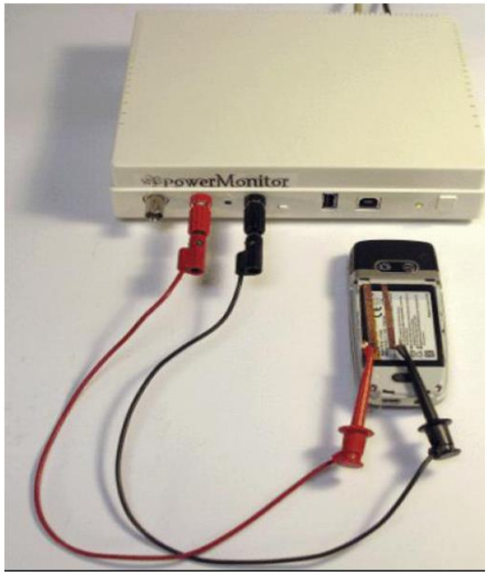
DATA

CAPTURE FILE

GRAPHICS

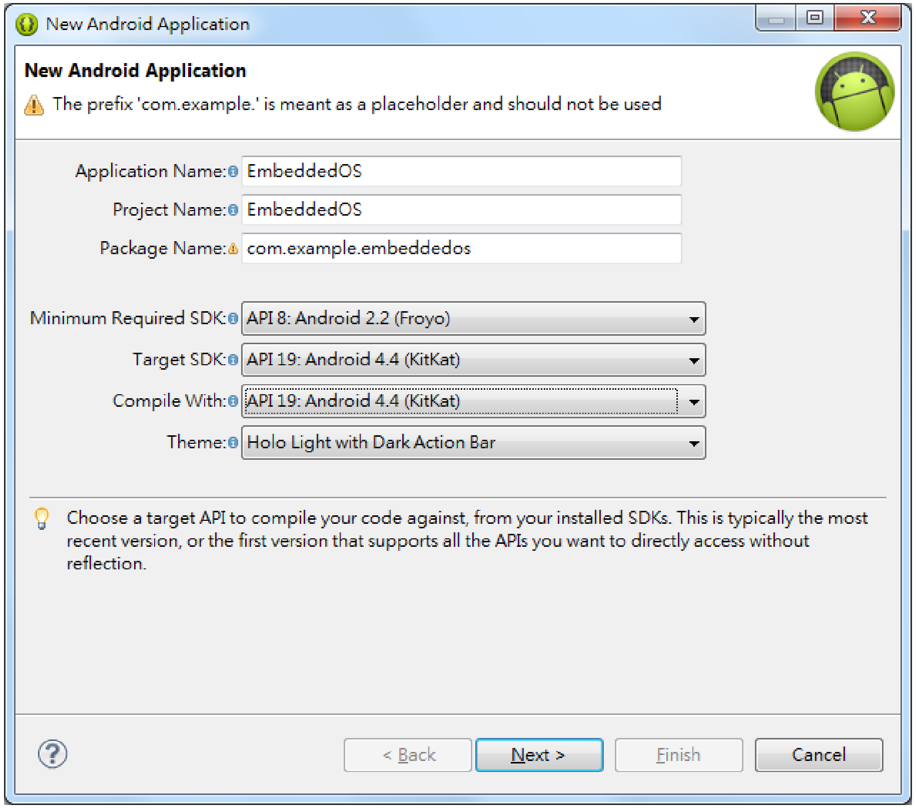
HW rev D Prot ver 16
FW ver 16 SW ver 4.0.1

PowerTool.DeviceNull ...(PowerBox_1266_-8590115538697998375.pt4) CAPTURE DATE:7/19/2007 4:48:18 PM S#:1266 Calibration:OK



Set Information of the Project

- Application Name is the app name that appears to users
- Project Name is the name of your project directory and the name visible in Eclipse
- Package Name is the package namespace for your app (following the same rules as packages in the Java programming language)



New Android Application

⚠ The prefix 'com.example.' is meant as a placeholder and should not be used

Application Name: EmbeddedOS

Project Name: EmbeddedOS

Package Name: com.example.embeddedos

Minimum Required SDK: API 8: Android 2.2 (Froyo)

Target SDK: API 19: Android 4.4 (KitKat)

Compile With: API 19: Android 4.4 (KitKat)

Theme: Holo Light with Dark Action Bar

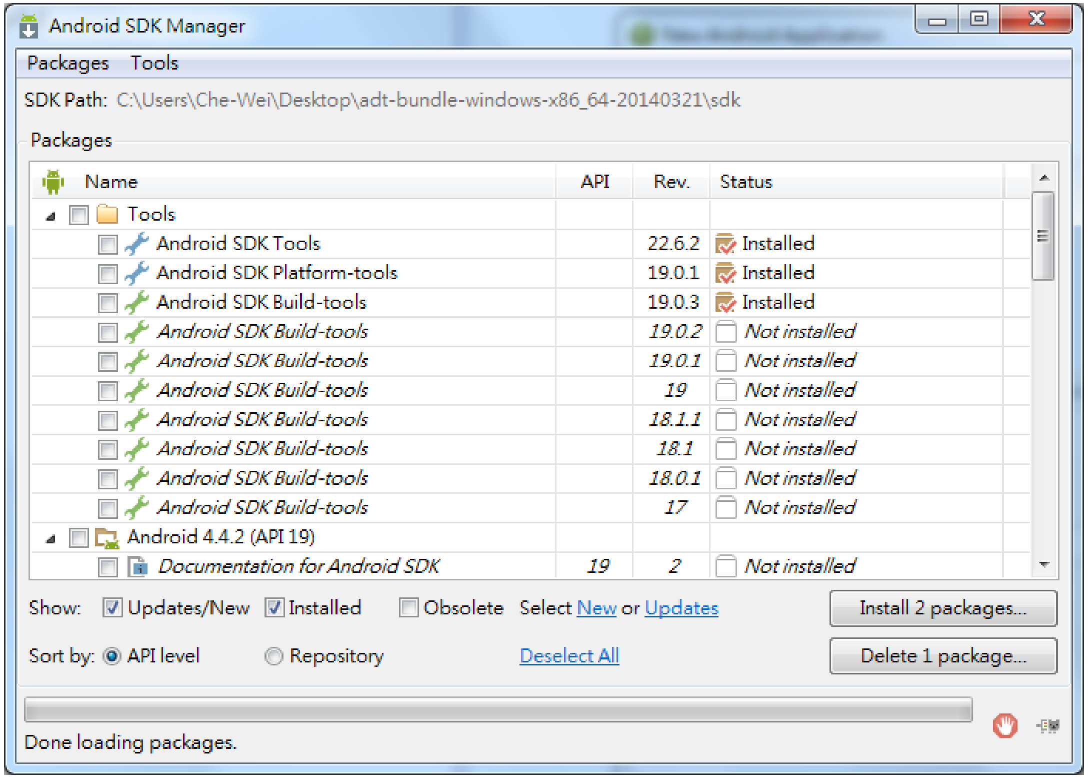
Choose a target API to compile your code against, from your installed SDKs. This is typically the most recent version, or the first version that supports all the APIs you want to directly access without reflection.

API Support

- Minimum Required SDK is the lowest version of Android that your app supports
- Target SDK indicates the highest version of Android
- Compile With is the platform version against which you will compile your app
  - By default, this is set to the latest version of Android available in your SDK
- Theme specifies the Android UI style to apply for your app

SDK Manager

- If you want to install more libraries for different Android versions or different function supports



Android SDK Manager

SDK Path: C:\Users\Che-Wei\Desktop\adt-bundle-windows-x86_64-20140321\sdk

Packages

Name	API	Rev.	Status
□ Tools			
□ Android SDK Tools		22.6.2	✓ Installed
□ Android SDK Platform-tools		19.0.1	✓ Installed
□ Android SDK Build-tools		19.0.3	✓ Installed
□ *Android SDK Build-tools*		*19.0.2*	□ *Not installed*
□ *Android SDK Build-tools*		*19.0.1*	□ *Not installed*
□ *Android SDK Build-tools*		*19*	□ *Not installed*
□ *Android SDK Build-tools*		*18.1.1*	□ *Not installed*
□ *Android SDK Build-tools*		*18.1*	□ *Not installed*
□ *Android SDK Build-tools*		*18.0.1*	□ *Not installed*
□ *Android SDK Build-tools*		*17*	□ *Not installed*
□ Android 4.4.2 (API 19)			
□ *Documentation for Android SDK*	*19*	*2*	□ *Not installed*

Show: ☑ Updates/New ☑ Installed □ Obsolete Select New or Updates

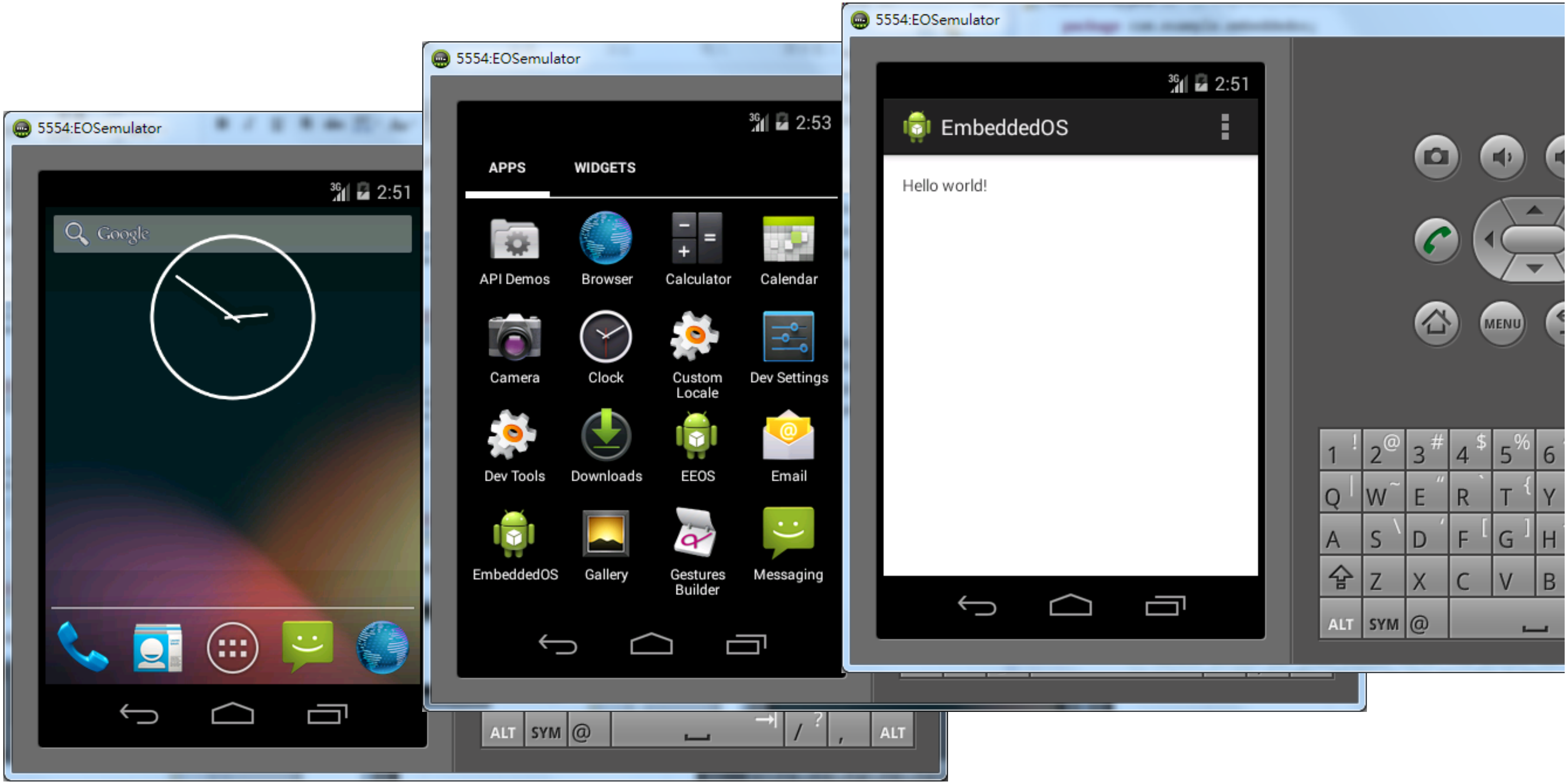
Sort by: ● API level ○ Repository Deselect All

Done loading packages.

Execute APP on an Android Device

- Enable USB debugging on your device
  - On most devices running Android 3.2 or older, you can find the option under Settings → Applications → Development
  - On Android 4.0 and newer, it's in Settings →Developer options
  - On Android 4.2 and newer, Developer options is hidden by default
    - To make it available, go to Settings → About phone → tap Build number (版本號碼 or 軟體版本) seven times
      - It might be different for different Android devices
    - Return to the previous screen to find Developer options
- Developer Options → Enable USB debugging
- Down and install the USB driver and install it

Hello World



Hello world!

Dalvik Virtual Machine

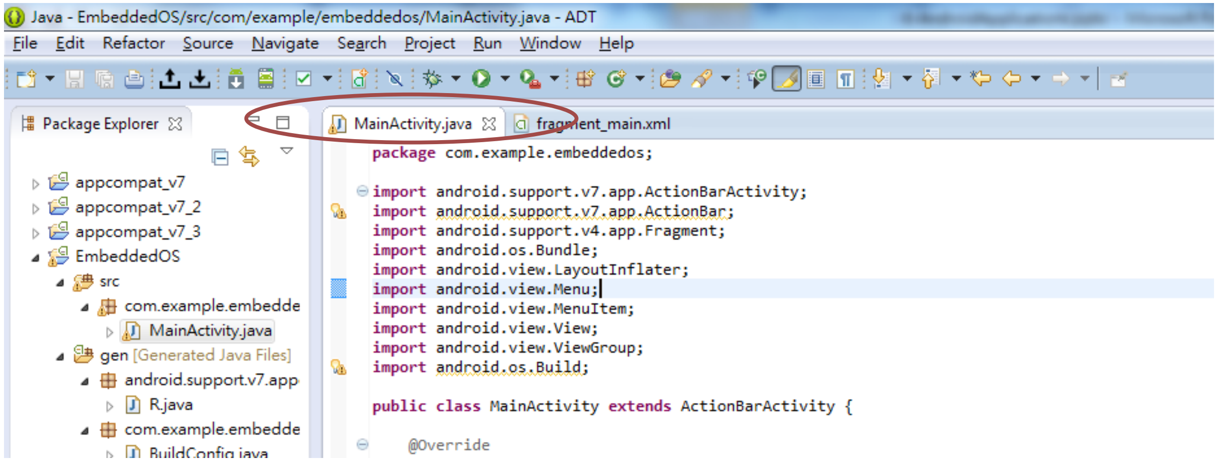
- Providing environment on which every Android application runs
  - Each Android application runs in its own process, with its own instance of the Dalvik Virtual Machine (DVM)
  - Register-based virtual machine
- Executing the Dalvik Executable (.dex) format
  - .dex format is optimized for minimal memory footprint
- Relying on the Linux Kernel
  - Multi-threading
  - Low-level memory management

Android Runtime (ART)

- Android Runtime (ART) is an application runtime environment
- ART is provided to replace Dalvik
- ART introduces the use of ahead-of-time (AOT) compilation
- AOT compiles entire applications into native machine code upon their installation
- Android 4.4 has alternatives to use ART or Dalvik
- After Android 5.0, Dalvik was entirely replaced by ART

Activities

- Activities are the basis of android applications
- An Activity defines a viewable screen
- Multiple Activities for an application are allowed
- Each activity is a separate entity
- They have a life cycle
  - Events happen either via touching buttons or programmatically



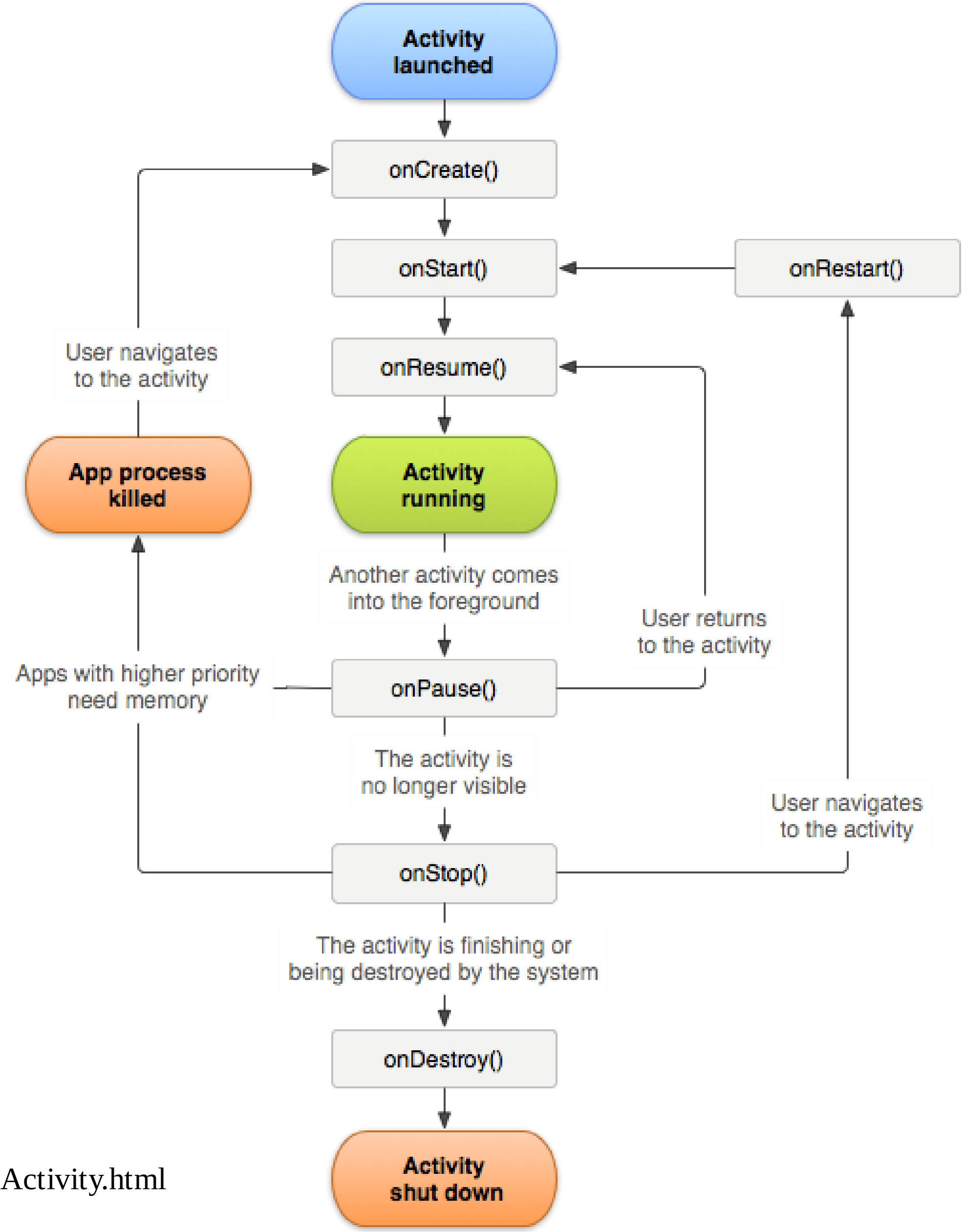
```
package com.example.embeddedos;

import android.support.v7.app.ActionBarActivity;
import android.support.v7.app.ActionBar;
import android.support.v4.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.os.Build;

public class MainActivity extends ActionBarActivity {

    @Override
```

Activity Lifecycle

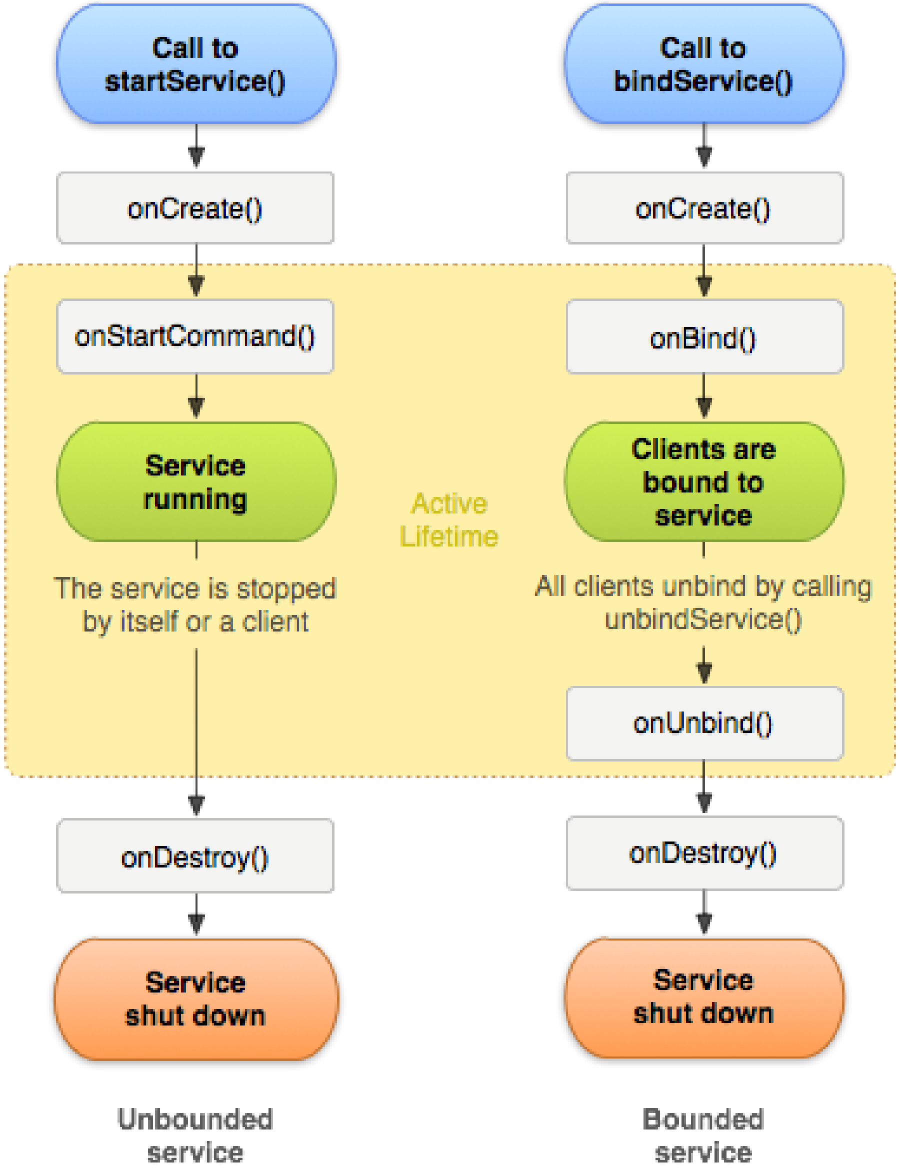

Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

Source: http://developer.android.com/reference/android/app/Activity.html

Services

- Run in the background
  - Should be used if something needs to be done while the user is not interacting with application
  - Should create a new thread in the service to do work in
- Can be bound to an application
  - It will terminate when all applications bound to it have unbound
  - Multiple applications can communicate with each other via a service
- Needs to be declared in manifest file

Service Lifecycle



Source: http://developer.android.com/guide/components/services.html