Lecture 3

Software, programming language, operating system

Outline

- How to run software on computer?
- What is machine code?
  - Instructions
  - Assembly Language
  - Little man computer
- What are programming languages
  - Why there are many PLs
  - Types of PLs (object-oriented, functional, imperative, declarative, scripts)
- What is operating system?
  - Types of OS
- Types of software

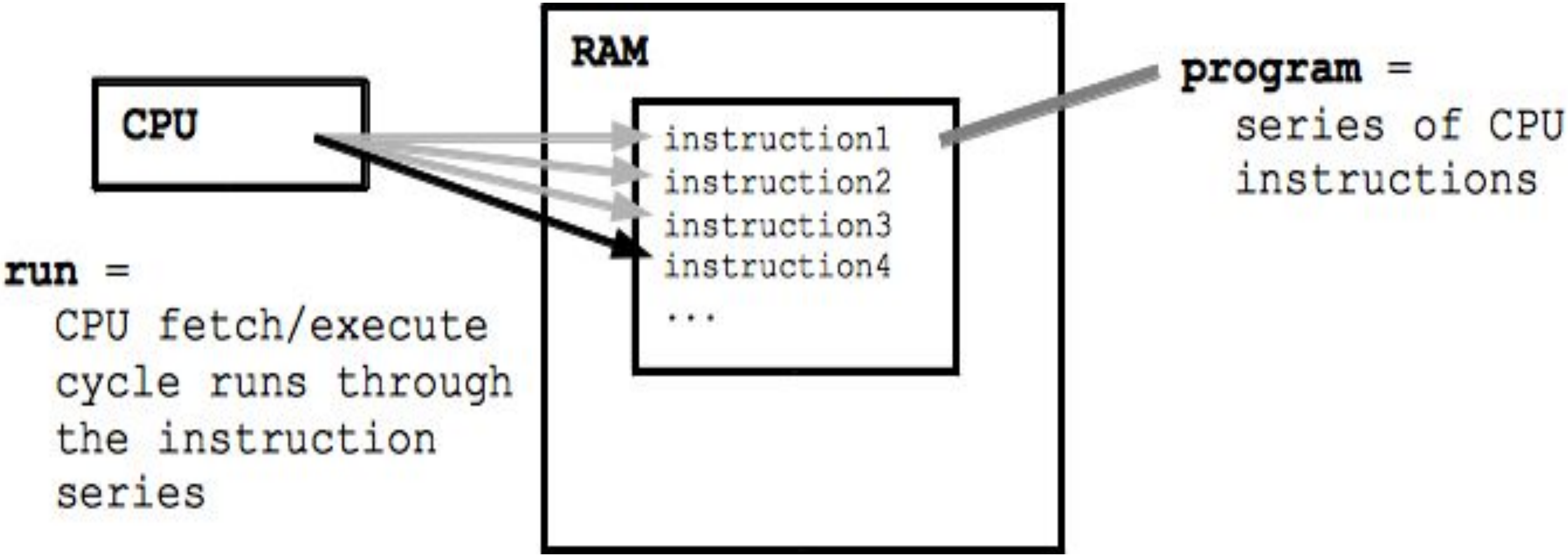
Software concept

Software “runs” on the hardware like music “runs” on the piano.
Hardware
CPU - “brain with a mouth” to eat instructions from memory and produces output out of them...
RAM: memory that CPU can access directly. Access time to RAM is faster than that of hard disk. But RAM is smaller in size and more expensive than hard disks.
Storage : larger memory than RAM but with a slower access and cheaper in cost (e.g. hard disks, flash memory, CD/DVD)

What happens: double-click program?

- What is a program, like Firefox.exe (.exe is a Windows convention)
- Firefox.exe is just a file -- can look at it in the file system
- The file Firefox.exe is basically the bytes of millions of instructions
- Double click Firefox.exe to Run
    a. The instruction bytes are copied up into RAM
    b. The CPU is directed to start running at the first instruction

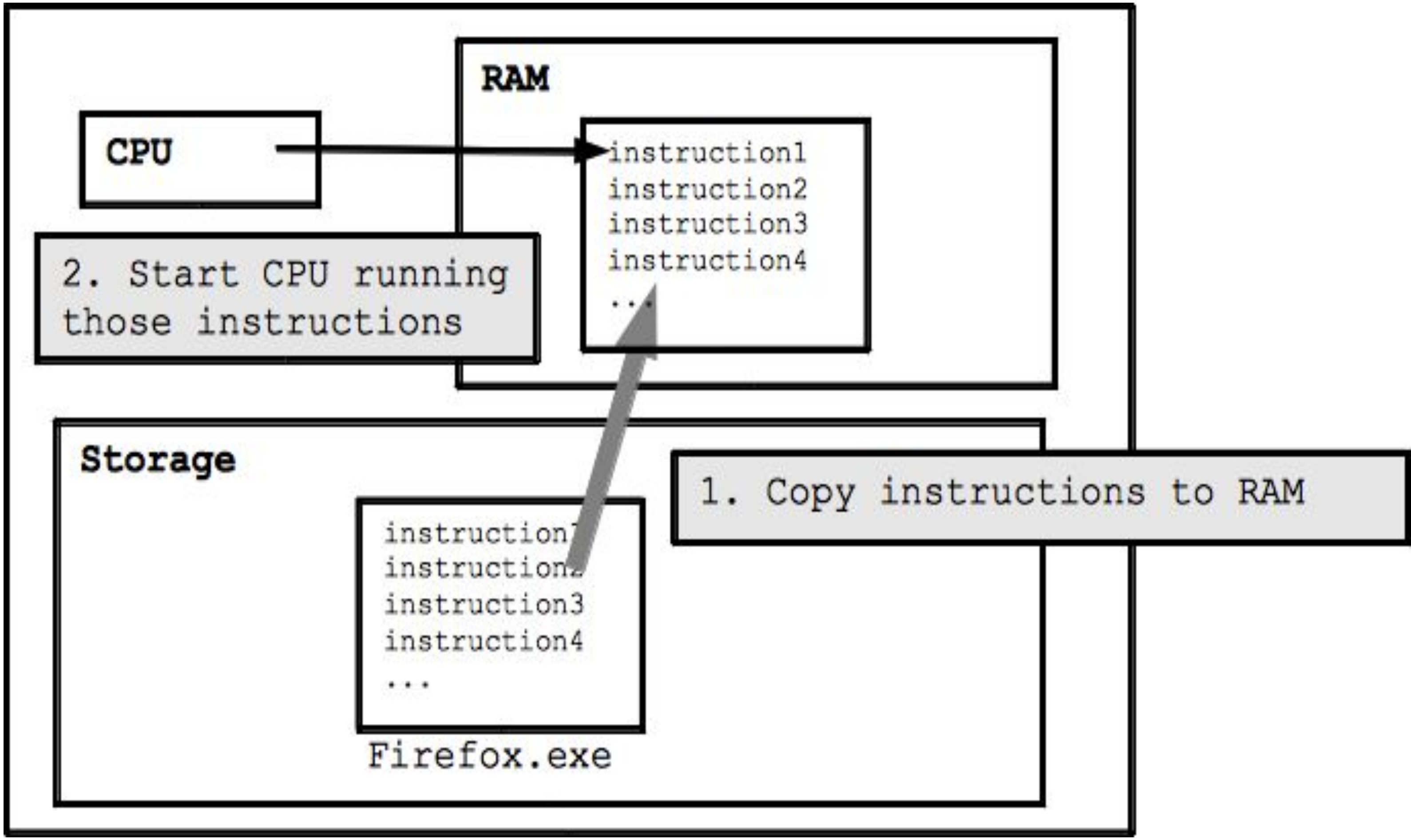
Relation between CPU, RAM and program



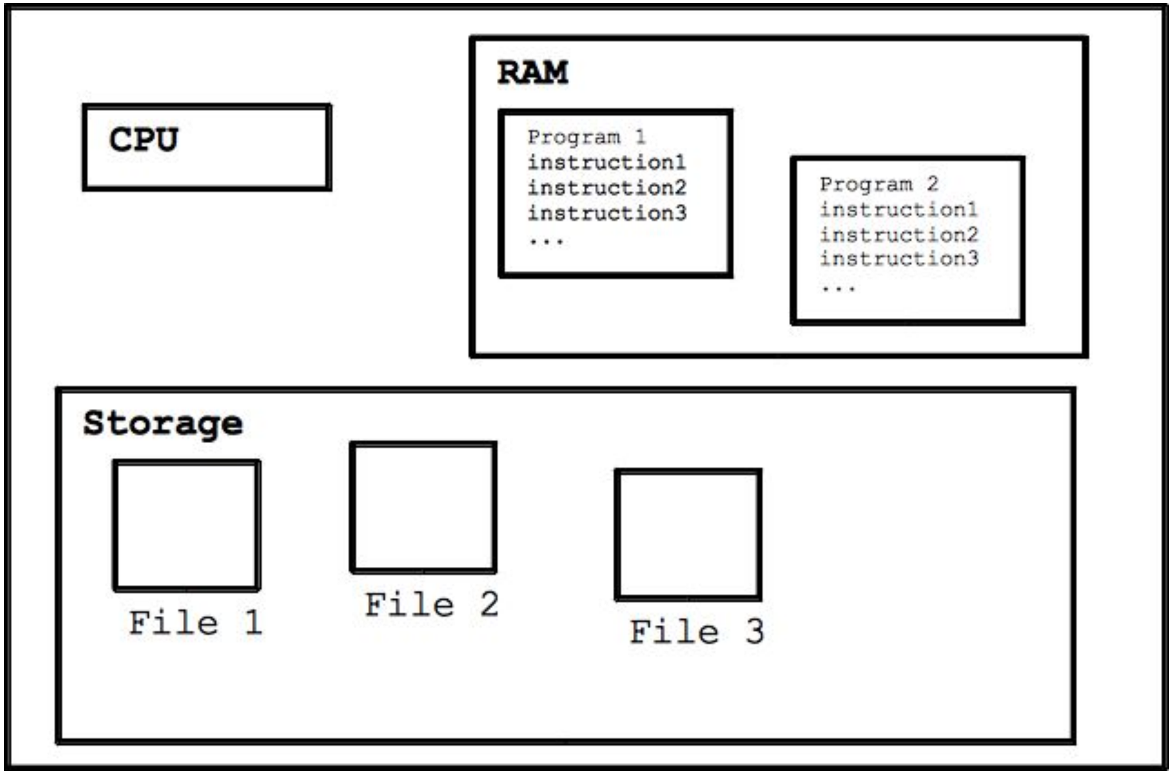
program = series of CPU instructions

run = CPU fetch/execute cycle runs through the instruction series

How program starts



OS manages programs in RAM and files on a storage



Software: machine code

- Software - code that runs on the hardware
- CPU implements "machine code" instructions
- Each machine code instruction is extremely simple
  - e.g. add 2 numbers
  - e.g. compare 2 numbers

Software: machine code

- The language of the machine code is hardwired into the design of the CPU
    - it is not something that can be changed
    - Each family of compatible CPUs (e.g. the very popular Intel x86 family) has its own machine code which is not compatible with the machine code of other CPU families.

Lyrics

CPU is capable of performing simple instructions if you load them into RAM. For example addition/subtraction of two numbers, jumping to another instruction
While CPU is performing an instruction the temporary results are stored inside CPU itself in the fastest, smallest memory locations – registers.
CPU starts by fetching an instruction from RAM and then executes/performs it. Then again fetches and executes... This is all CPU does.
Initially all programs are on hard disk. When user double-clicks a program icon the machine instructions of that program get loaded to RAM where CPU can access them and execute.

What are the instructions?

CPU has list of defined instructions, such as:

- add values
- store values
- copy values
- increment value
- go to command

Example

Signals sent to CPU

(`10110000 01100001`) (read in hex `B0 61`)

`B0` means “Move a copy of the following value into AL (place in memory)”

Value in AL memory is equal to 61 in hexadecimal

What is assembly language

CPU understands only electrical signals, such as:

```
(10110000 01100001)
```

But to be understandable to programmers, assembly languages were created.

Assembly languages use words instead of binary commands, as:

```
MOV AL, 61h
```

Little Man Computer

- The Little Man Computer (LMC) is an instructional model of a computer.
- The LMC is generally used to teach students, because it models a simple architecture computer - which has all of the basic features of a modern computer.
- It can be programmed in machine code or assembly code

Try it: Little Man Computer

https://en.wikipedia.org/wiki/Little_man_computer
https://www.youtube.com/watch?v=kCyyZI1GgsQ

http://robowriter.info/little-man-computer/

http://peterhigginson.co.uk/LMC/

Battle Tank:
http://pddring.github.io/cpu-battle-tank/

PL vs Assembler

```
if(a > 2)
    b = 3;
else
{
    b = 5;
    c = 8;
}
a = 8;
```

```
MOV A, 200
LGR #2
JZ ELSE
MOV 201, #3
JMP END
ELSE:
MOV 201, #5   ; b = 5;
MOV 202, #8   ; c = 8;
END:
MOV 201, #8   ; a = 8
```

Programming languages (PL)

- PL are translated into machine code
- PL were created to make developing software simple
- Programming Languages has more abstractions
  - Like arrays, lists, classes
  - functions and etc
- Programming Languages has more extra lines of code

Why there are many PLs?

People take ideas from different languages and combine them into a new languages. Some features are improved, some are added, some are removed.

Some are modifications of previous languages, like C++ is next step of C, and Java is some modification of C++

Important PLs

Java - used in web applications, software systems, where software needs security, and frequent changes

C++ - used in games, and software where speed is the most important. (Windows is written in C++)

Python - writing software is much faster. Used by scientists because it is easy.

Important PLs (2)

Javascript - to perform operations in web-browser

PHP - develop fast web-applications

Ruby - easy to understand, and write complex applications

PL comparison

This table has ***imaginary*** numbers.

But this numbers shows some intuition

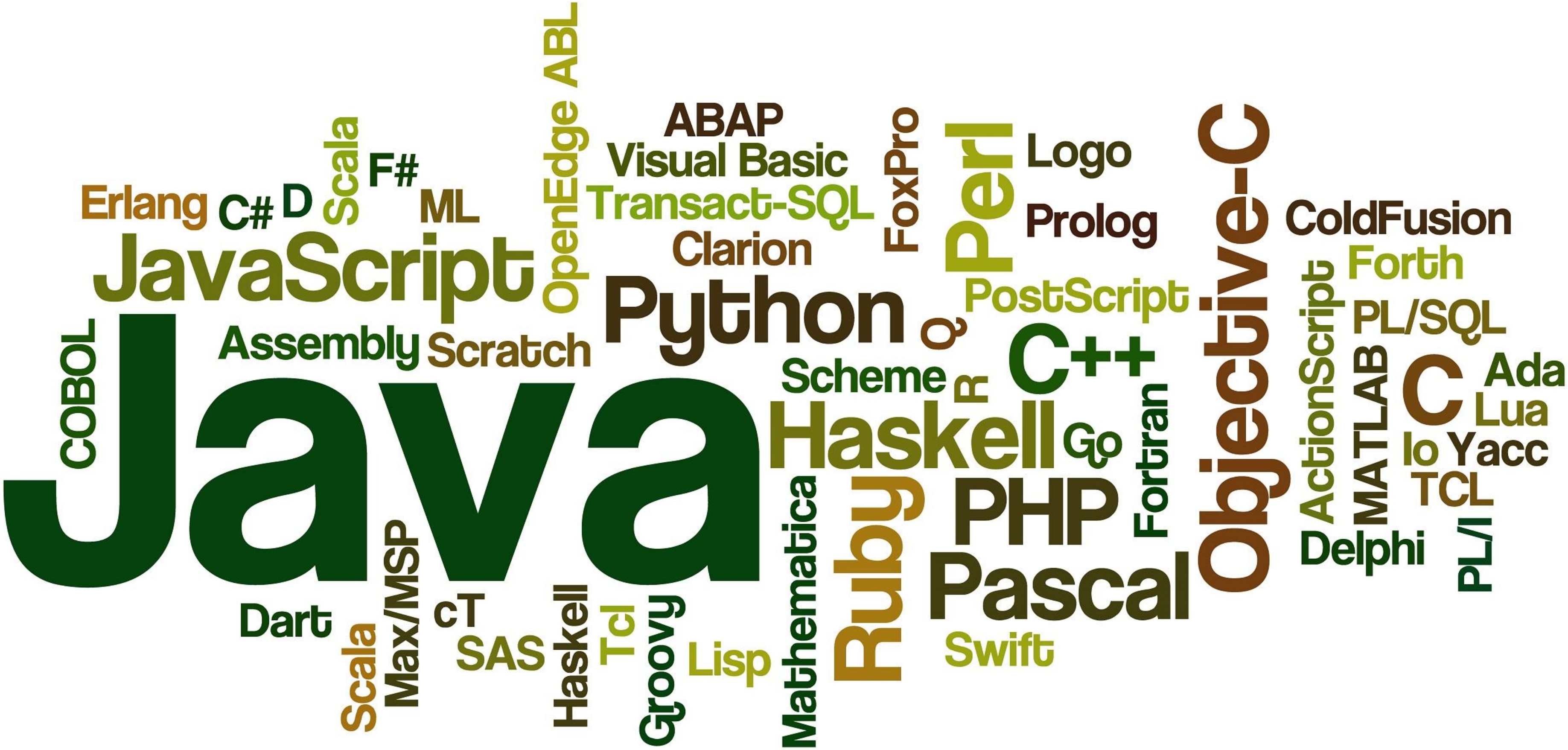
Programming Language	**Execution time**	**Development time**	**Learning time**
Assembler	5ms	14 days	2 years
C	15ms	8 days	1 year
C++	25ms	4 days	1 year
Java	100ms	1 day	6 month
Python	500ms	6 hours	4 month

Programming Languages

When you need web-application, that have operations with database, and have small amount of time for development. (You can choose Ruby or Python)

When you need program that calculates very big amounts of numbers, and you want to make it very fast (You can choose Assembler or C)

When you need program that makes less bugs (You can choose Java)



Programming languages

Web-site with information about different PLs

http://programming.dojo.net.nz/welcome/index

Computer languages

Programmers write code in a “high” level programming language whereas CPU understands very simple “low” level language

Programmers say more in less words

Need for translation from high to low level

Source code = code in high level language like C++, Java, Python etc

Compiler = translator from high level to low level CPU/machine language

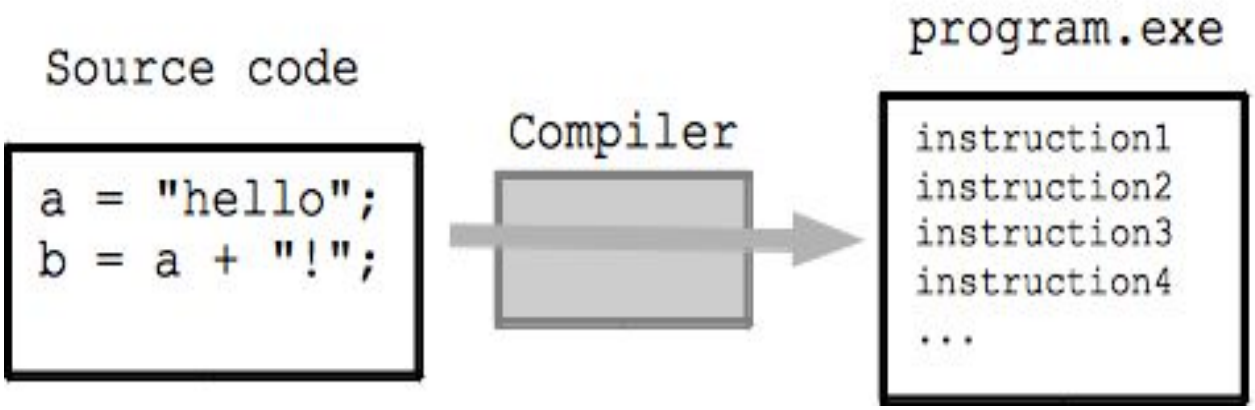
Declarative and Imperative languages

Imperative - The focus is on what steps the computer should take rather than what the computer will do (ex. C, C++, Java).

Declarative - The focus is on what the computer should do rather than how it should do it

Declarative languages: SQL, Scala, Haskell, Erlang

Compilation Process



```
a = "hello";
b = a + "!";
```

```
instruction1
instruction2
instruction3
instruction4
...
```

Operating system

- What starts Firefox?
  - Operating System
- Set of supervisory programs, run when computer first starts
- Administration behind the scenes
- Starting/managing/ending other programs
  - Modern computers can run multiple programs at the same time
  - Operating system keeps each program run isolated
  - Program has its own RAM, its own windows on screen
  - vs. accidental or malicious action between programs

Modern operating systems: functions

- Program execution
- Memory management
- Multitasking
- Disk access and file systems
- Networking
- Security

Operating system

Operating System is the set of supervisory programs that run when computer first starts

Administers behind the scene – like government with programs like citizens and malicious programs (viruses) as criminals

Starts/manages/ends other programs

Keeps each program run isolated

Provides access to peripheral devices via drivers

Manages file system on storage devices like hard disks, USB memory sticks etc.

Operating systems: FAQ

Do all computers should have OS?

No, some computers may work only with one program

What does operating system do?

Actually, it runs other programs

Operating systems: Desktop

There are three main families of operating systems:

- Linux
- Windows
- OSX

Linux

- Fedora, Ubuntu, RedHat, Suse
- mostly free
- mostly open-source (customizable)

Windows

- Most widespread operating system
- Windows 3.1, Windows 98, Windows ME, Windows XP, Windows Vista, Windows 7, Windows 8
- Operating system is a property of Microsoft

OSX

- bundled in Mac computers
- cannot work with other computers
- well-known for its’ pretty and intuitive user interface

Mobile operating systems

- Android
- IOS
- Windows Phone
- Ubuntu Touch OS
- BlackBerry OS

Mobile OS: features

Android

- developed by Google
- free to manufacturers
- to create applications. pay only 20$ and immediately publish

IOS

- developed by Apple
- only in Apple products
- pay 100$ and then wait for approvement of developed application

Mobile operating systems

Ubuntu touch OS:

- couldn’t find money for publishing
- main idea: one OS on mobile phone and desktop computer

BlackBerry:

- was popular for its ciphering technology

Windows Phone:

- becoming popular in last years

2015 Q2 shipments (in millions)

Android - (85%)

IOS - (13.9%)

Windows Phone - (2.6%)

BlackBerry OS - (0.3%)

Other - (0.4%)

Drivers

A device driver or software driver is a computer program allowing OS to interact with a hardware device.

Command-line interface

Operating systems have computer-line interface (CLI) to control OS through predefined commands.

In Windows:

In Run command write cmd

Command-line interface: commands (Windows)

cd - change directory

dir - show contents of directory

mkdir - create new directory

move - move file to folder

copy - copy file to folder

del - delete file

CLI - Examples

cd ../ - go one directory up

copy ../a.txt ./ - copy file a.txt from up folder to current directory

Open-source

Some software are published not in form of executable file but in form of code, that is called open-source.

Anyone can change code, and produce his own version of product.

List of notable open-source software: Ubuntu, Firefox, GIMP, Blender, Android, LibreOffice, MySQL

Motivation to produce open-source

http://en.wikipedia.org/wiki/Open-source_movement#Motivations_of_Programmers

Altruism - helping others

Community sharing and improvement

Recognition

Creative Expression

Proprietary

Get Started with Ease

Support and Documentation

Better User Experience

Accountability

http://www.codagenic.com/company/blog/open-source-vs-proprietary.html

Open-source

Free to Use

Cost of Ownership

No Vendor Lock-in

Quality Code