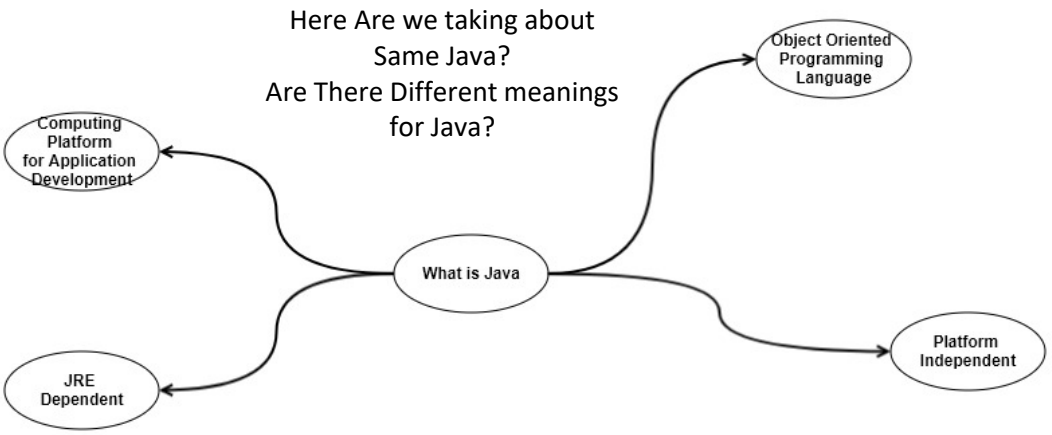
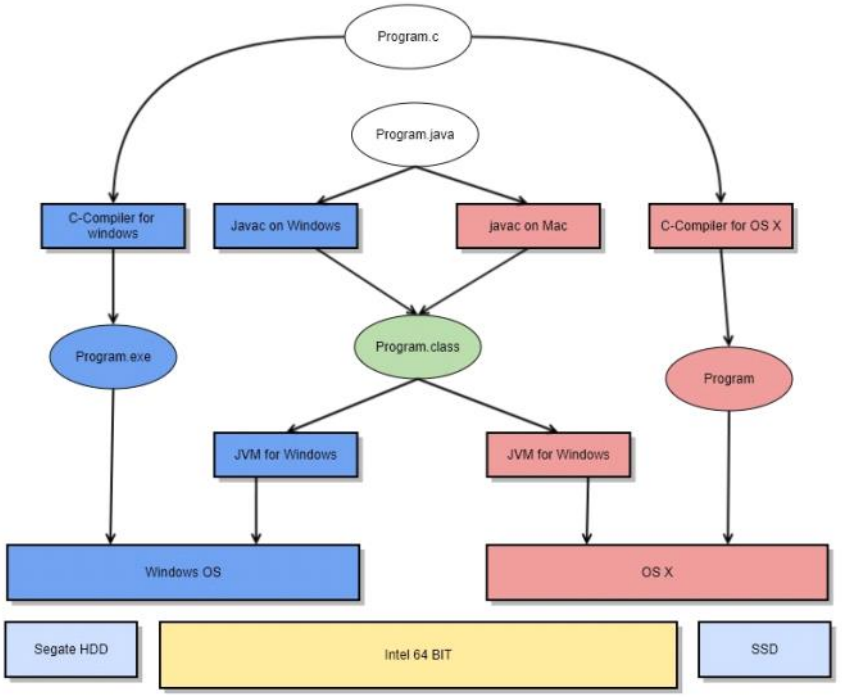
What is Java

Tuesday, July 30, 2019 9:14 AM



Java Vs C (or C++)

Tuesday, July 30, 2019 9:24 AM



Note:

- Even C/C++ programs are not targeting **real machine** or hardware.
- They target a particular OS.
- OS acts as a **virtual machine** for the real hardware exactly in the same way as JVM acts as virtual machine on the top of given OS

Notes:

- **Legends**
  - **Blue ---> Related to windows**
  - **Pink ---> Related to Mac**

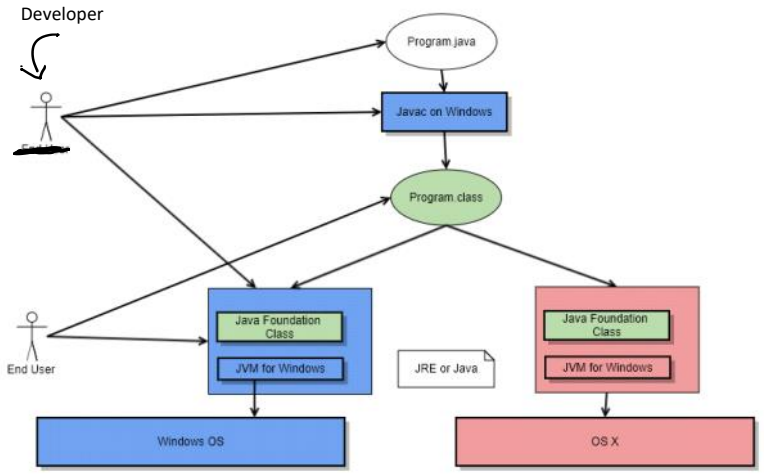
- **C/C++**
  - **Same C code can work on different platforms or virtual machines (such as windows, linux, osX**

  - **We need a compiler for particular OS. The compiler is expected to generate the code that can execute on a given OS.**
  - **Compiler for one OS wont work for other OS**
  - **Code compiled targeting one OS wont work on other OS**

- **Java**
  - **We need to compile the code once**
  - **We would use different compiler on different OS**
  - **The Different compilers are not compiling for (targeting) different OS. They simply run on different OS as an application.**
  - **The compiled code is going to be identical.**
  - **Compiler or JVM are platform dependent. Thy may application written in java platform independent**

Java Distributions

Tuesday, July 30, 2019 9:38 AM



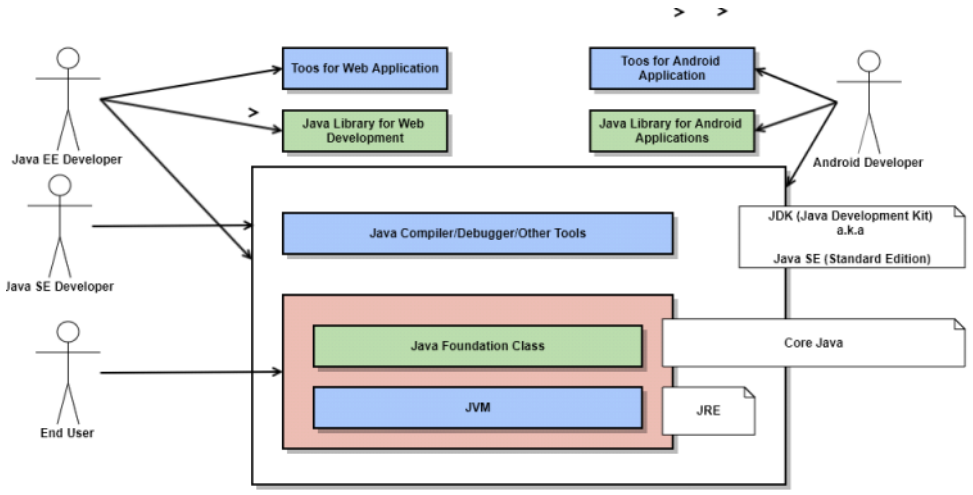
Note:

End Use Perspective

- **An End user needs only**
  - **JVM**
  - **Common Foundation Cass Library used by our application**
- **This JVM + FCL combination is referred as Java Runtime Environment (JRE) or simply Java**

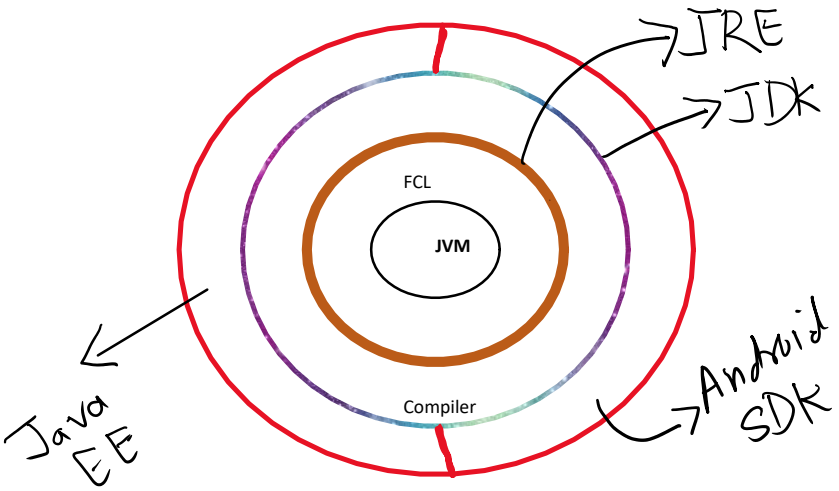
Developer's Perspect

- **Developers apart from JRE also need**
  - **Compiler**
  - **Debugger**
  - **Other development related tools**
- **This bundle is referred as**
  - **JDK or Java SE**



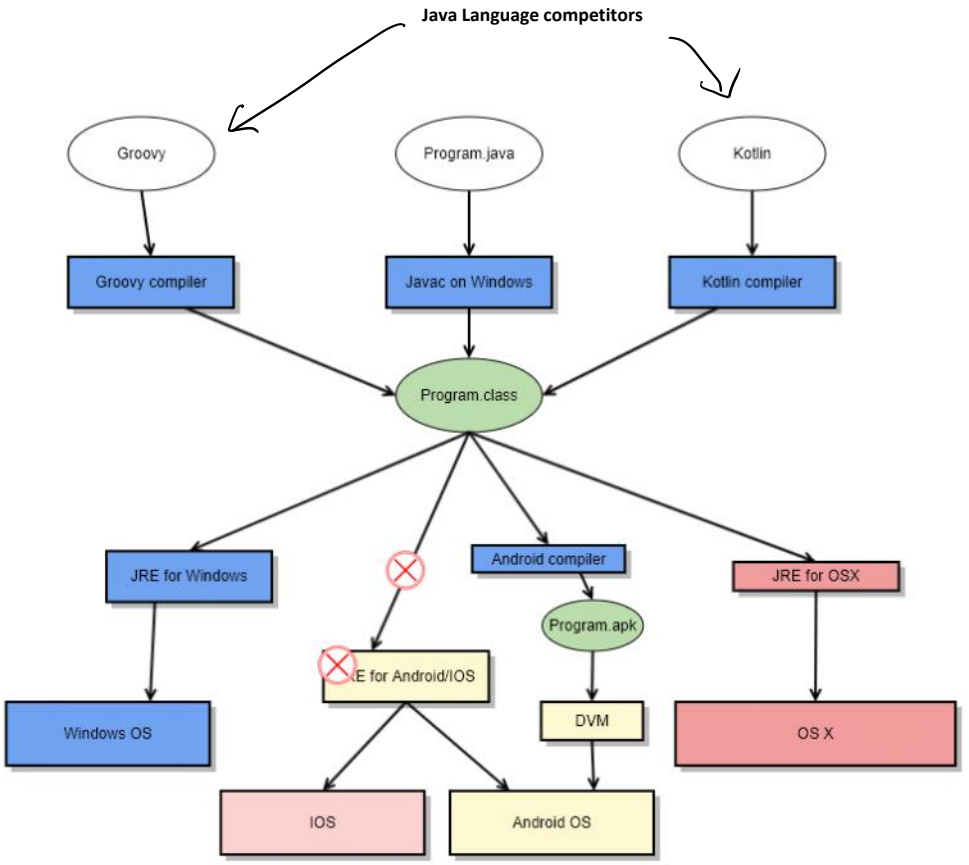
Specialized Variations

- **Apart from SE we can have specialized extensions for different application needs**
- **Enterprise Java is a set of specialized tools for creating distributed enterprise level application**
- **Android SDK is a set of tools and libraries for developing Android based Application**



There Are Two Java

Tuesday, July 30, 2019 9:48 AM



1. **Java - The Language**
   - **The Object Oriented Language which is expected to run on Java Platform**
   - **The language is Platform Independent**
   - **Comparable to C# language of .NET platform**
   - **Today This language has competitors that can be compiled to run on Java Platform**

2. **Java - the Platform**
   - **This component is responsible for**
     1. **Running Java Application**
     2. **Making it platform independent**
   - **Java can run on only those device that have Java Platform installed**

There is No True Java Platform (JRE) available on mobile devices like IOS or Android

Android Uses Java (language and probably its competitors) and the modifies the byte code to run on DVM rather than JVM

Java Source File

Tuesday, July 30, 2019 9:14 AM

1. A source file should have .java extension
2. A source file name is case sensitive
3. A source file can contain one or more classes

Packages

Tuesday, July 30, 2019 12:23 PM

1. Packages group related classes
2. A package is associated with a folder on the drive
    a. A folder is not a java element
    b. A package is a recognized java element
3. A class marked need to specify its package using a package statement
4. A package statement (if present) must be the first statement in a file
    a. **package furnitures;**
5. Your .java file must be present in a folder matching package name
6. You classpath should include the folder containing your .java/.class or packages
    a. Package folder shouldn't be part of classpath
7. Eleements define within a package is by default accessible only within the package
8. To access any class/method/fields from classes outside current package you must make them **public**
9. Conventionally a package name must be all lowercase.
10. When a class doesn't contain a **package** specification, it is considered to be part of **default global** package

Accessing a packaged element

1. Options 1: Use class qualified names
    - console.Input kb=new console.Input();

Non-Static or Object level members

Tuesday, July 30, 2019 4:45 PM

- Non Static Fields
  - Belongs to individuals object
  - Each object will have its own unique copy of those fields
- Non Static Methods
  - Defines Object's behavior
  - Can access both static and non-static fields and methods
  - Can be accessed only using the object reference

Static Members (a.k.a Shared Members a.k.a class level members)

Tuesday, July 30, 2019 4:41 PM

- **Static Fields**
  - **Contains a single copy of information that is shared among all the objects of a class**
  - **Every object of the class can access it and modify it as if it is its personal copy**
  - **Changes done by an object will reflect in every other object**
  - **Used common sharable information**
    - **E.g. InterestRate is likely to be same for all objects of the class**
    - **E.g. All Card in a deck will have same background.**
- **Static Methods**
  - **can be access using either**
    - **Class Reference**
    - **Object Reference**
  - **doesn't need an object reference (it can use it)**
  - **Can access only other static members (methods or fields)**
  - **Can't directly access the non static members (methods or fields)**

Static Vs NonStatic

Wednesday, July 31, 2019 9:26 AM

```
class BankAccount{

        static int lastId=0;
        static double interestRate=12;

        int accountNumber;
        double balance;
        String password;

        public void deposit(double amount){

        }
}

public static void main(String []args){

        BankAccount a1=new BankAccount("p@ss",1000);
        BankAccount a2=new BankAccount("w0rd",2000);

}
```

```
class Bank{

        static int lastId=0;
        static double interestRate=12;

        BankAccount [] accounts;

        int openAccount(String password, double amount){
                ...
                int accountNumber=++lastId;
                BankAccount a=new BankAccount(accountNUmber,password,amount);

                //add this account to the bank's accounts collection

                return accountNumber;
        }

        public void deposit(int accountNumber, int amount){

                BankAccount a= findAccount(accountNumber);

                a.deposit(amount);

        }


}

public static void main(String []args){

        BankAccount a1=new BankAccount("p@ss",1000);
        BankAccount a2=new BankAccount("w0rd",2000);


        Bank icici=new Bank();

        int a1= icici.openAccount("p@ss",1000);
        int a2=icici.openAccount("w0rd",2000);


        icici.deposit(a1, 2000);
}
```