Object Oriented Programming II (Java)
CSE - 2101

Course Group Address:

https://classroom.google.com/c/MTQ1NTI5Mjk2NjEz?cjc=ro6o2ms

Text Book:

Deitel and Deitel: Java How to Program, 9th Edition

Herber Schildt: Java2 ; The Complete Reference, 8th Edition

Ivor Horton: Beginning Java; 7th Edition

Course Outline

Classes : 30

Lab: Object Oriented Programming with Java

Class Participation	10 Marks
In course	20 Marks (2-each of 20 Marks)
Final exam	70 Marks (5 out of 8)

Overview of the Syllabus

- Classes and Objects
- Data types, variables, operator, control structure
- Inheritance
- Polymorphism
- Exception
- Socket Programming
- Applet
- Thread
- Input Output
- Graphics/swing

CSE2101 – Object Oriented Programming

- What is OOP
- Overview of Java
- Versions of Java
- Java API
- Create, Compile and Run a Java Program
- Homework

Programming Languages

- Three types of programming languages
  1. Machine languages
     - Strings of numbers giving machine specific instructions
     - Example:

```
+1300042774
+1400593419
+1200274027
```

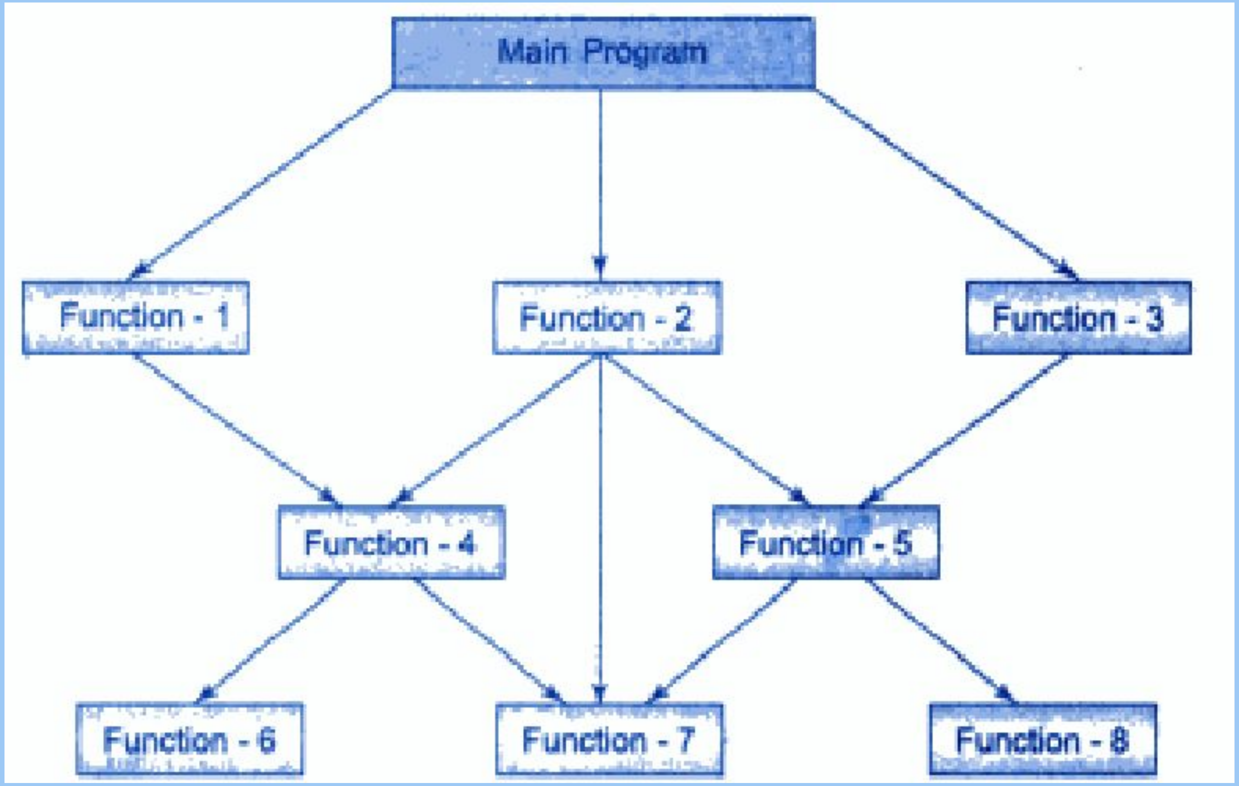
  2. Assembly languages
     - English-like abbreviations representing elementary computer operations (translated via assemblers)
       - Example:

```
LOAD    BASEPAY
ADD     OVERPAY
STORE   GROSSPAY
```

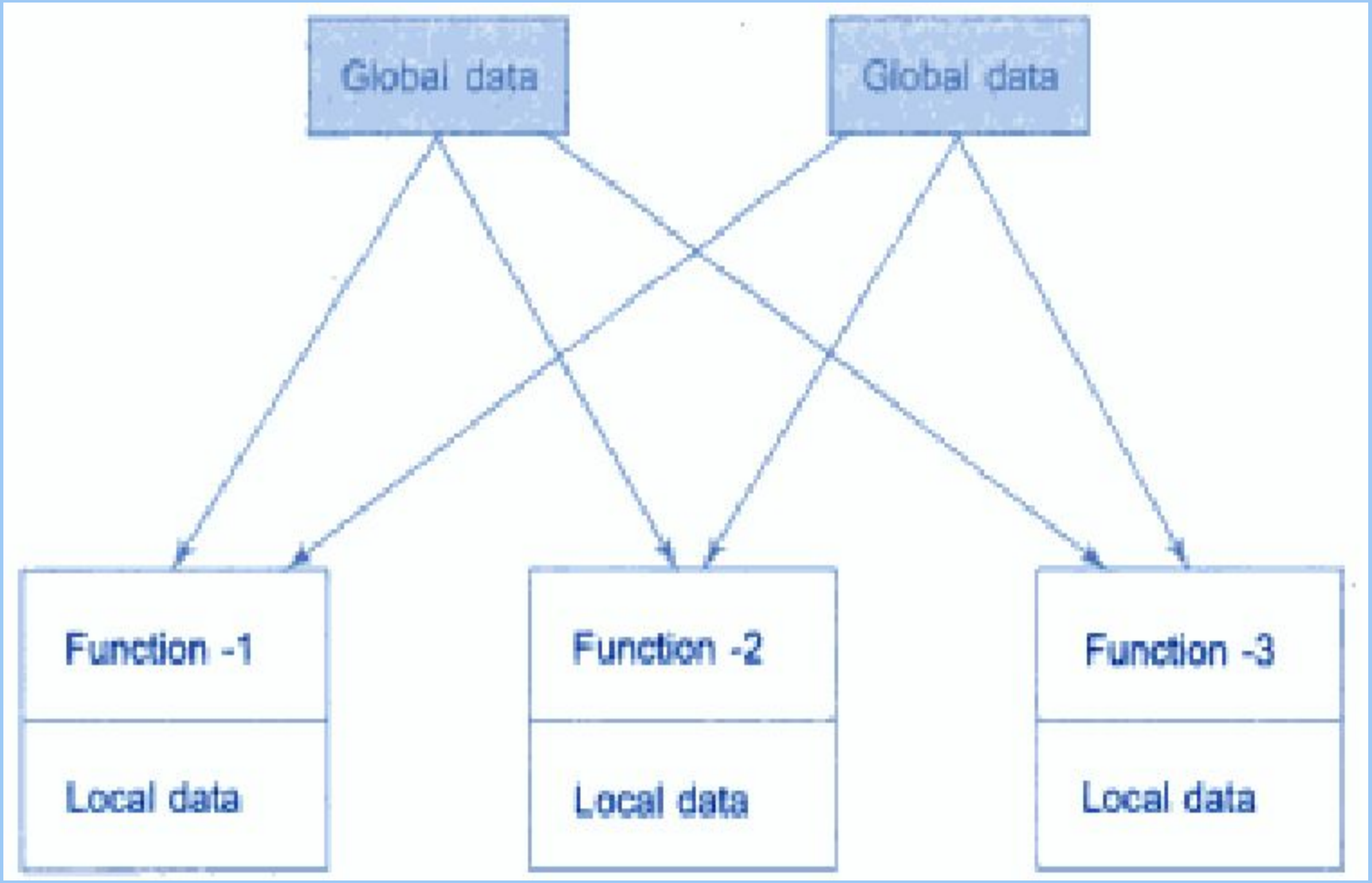
  3. High-level languages
     - Codes similar to everyday English
     - Use mathematical notations (translated via compilers)
     - Example: `grossPay = basePay + overTimePay`

Structure/Procedure Oriented Programming

- C, COBOL, FORTAN are known as Procedure Oriented Programming (POP) language.
- In POP the problem is viewed as a sequence of things to be done such as reading, calculating and printing.
- A number of functions are used to accomplish the task.



Relationship between Data and Functions in POP



Characteristics of POP

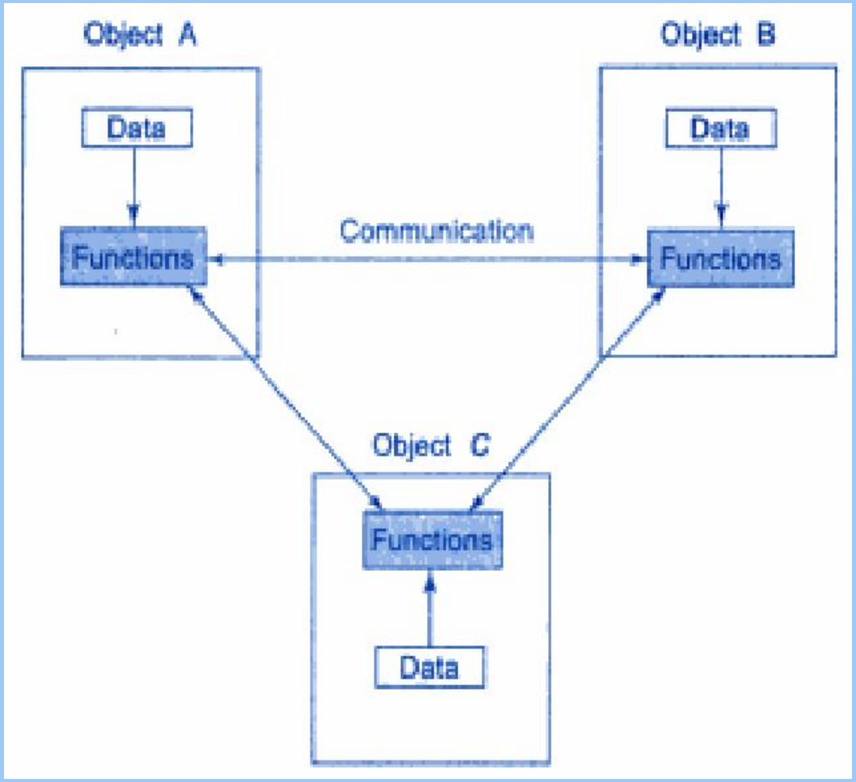
- Emphasis is on doing things (Algorithms).
- Large programs are divided into smaller programs known as Functions.
- Most of the functions share Global data.
- Data move openly around the system from function to function.
- Functions transform data from one form to another.
- Employes top-down approach in program design.

Object Oriented Programming

- Emphasis is on Data rather than procedure.
- Programs are divided into Objects.
- Data structure are designed such that they characterize the objects.
- Function that operate on the data of an object are tied together in the data structure.
- Data is hidden and can not be accessed by external functions.
- Objects may communicate with each other through functions.
- New data and functions can be added whenever necessary.
- Folows the bottom-up approach in program design.

Object Oriented Programming

- Object Oriented Programming is an approach that provides a way of modularizing programs by creating partitioned memory area for both data and functions that can be used as templates for creating copies of such modules on demand.



The Key Software Trend: Object Technology

- Objects
  - Reusable software components that model items in the real world
  - Meaningful software units
    - Date objects, time objects, paycheck objects, invoice objects, audio objects, video objects, file objects, record objects, etc.
    - Any noun can be represented as an object
  - More understandable, better organized, and easier to maintain than procedural programming

Java – An Example of OOP

- Developer: Oracle Corporation
- Originally Developed by Sun Microsystems (James Gosling)
- Originally called ***"Oak"***
- Java, May 20, 1995, Sun World
- A general-purpose object-oriented language
- Based on C/C++
- Designed for easy Web/Internet applications
- Widespread acceptance

Characteristics of Java

- Java is simple
- Java is object-oriented
- Java is distributed
- Java is interpreted
- Java is robust
- Java is secure
- Java is architecture-neutral
- Java is portable
- Java's performance
- Java is multithreaded
- Java is dynamic

Java is Simple?

- Java has automatic memory management
  - Automatically takes out the garbage
  - No dangling pointers. No memory leaks.
- Java simplifies pointer handling
  - No explicit reference/dereference operations
  - Everything is a pointer (like Lisp)
- Syntax is just like C++.

Object-Oriented
(more pure than C++)

- Everything is an object! All methods are virtual.

Network-Savvy

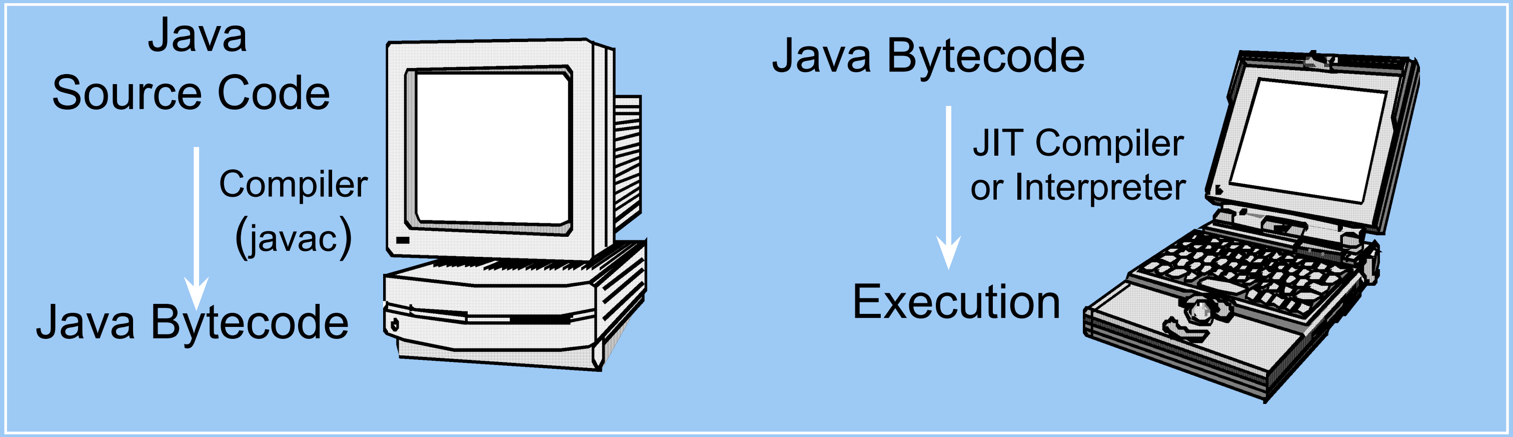
- Can access networks as easily as files.
- Many tools/libraries, run inside browsers.

Interpreted

- Compile for a “Virtual Machine” (VM) based on bytecodes (.class file)
- Have an interpreter (a simulator for the virtual machine).
- “Just-in-time” compiler: translate bytecodes into machine code just before execution

Java is Cross-Platform?

- Truth: Java programs can compile to machine-independent bytecode



Compile Time **Run Time**

- Truth: All major operating systems have Java runtime environments
  - Most bundle it (Solaris, MacOS, Windows 2000, OS/2)

Robust

- A lot more compiler and runtime checks than C++.

(eg. impossible to overwrite memory and corrupt data, exception handling, runtime checks of casts, …)

Architecture-Neutral

- Java executables will run on any machine!
  (provided it has a Java bytecode interpreter)

Portable

- Standard (fixed) data sizes:
  - byte = 8 bits float = 32 bits
  - short = 16 bits double = 64 bits
  - int = 32 bit unicode characters
  - long = 64 bits
- Libraries: Java includes libraries for graphics, sound, etc., and these are implemented on all machines (UNIX, Windows 95, Mac…)
- This means all Java programs are portable!

High Performance

- Java runtimes interpret programs s l o w l y . . .
- However, it is possible to translate the Java bytecodes into native machine code just before a program is run.
- These "just-in-time compilation" runtimes can make Java nearly as fast as C++.

Multi-Threaded

- Several "threads" can run in parallel. Direct standard language support for multitasking.

Dynamic

- Libraries are linked in later than in C++ (Java: at runtime, C++: at compiletime)
- Installing a new version of a library automatically updates all programs!
- Even load classes (=code) while running!

Java Class Libraries

- Java contains class libraries
  - Known as Java APIs (Application Programming Interfaces)
- Classes
  - Include *methods* that perform tasks
    - Return information after task completion
  - Used to build Java programs

The Java APIs

- Oracle constantly adding new features and APIs
- The Core Java API is now very large
- Separate set of extension APIs for specific purposes
  - E.g. Telephony, Web applications, Game programming

JDK (Java Development Kit) Versions

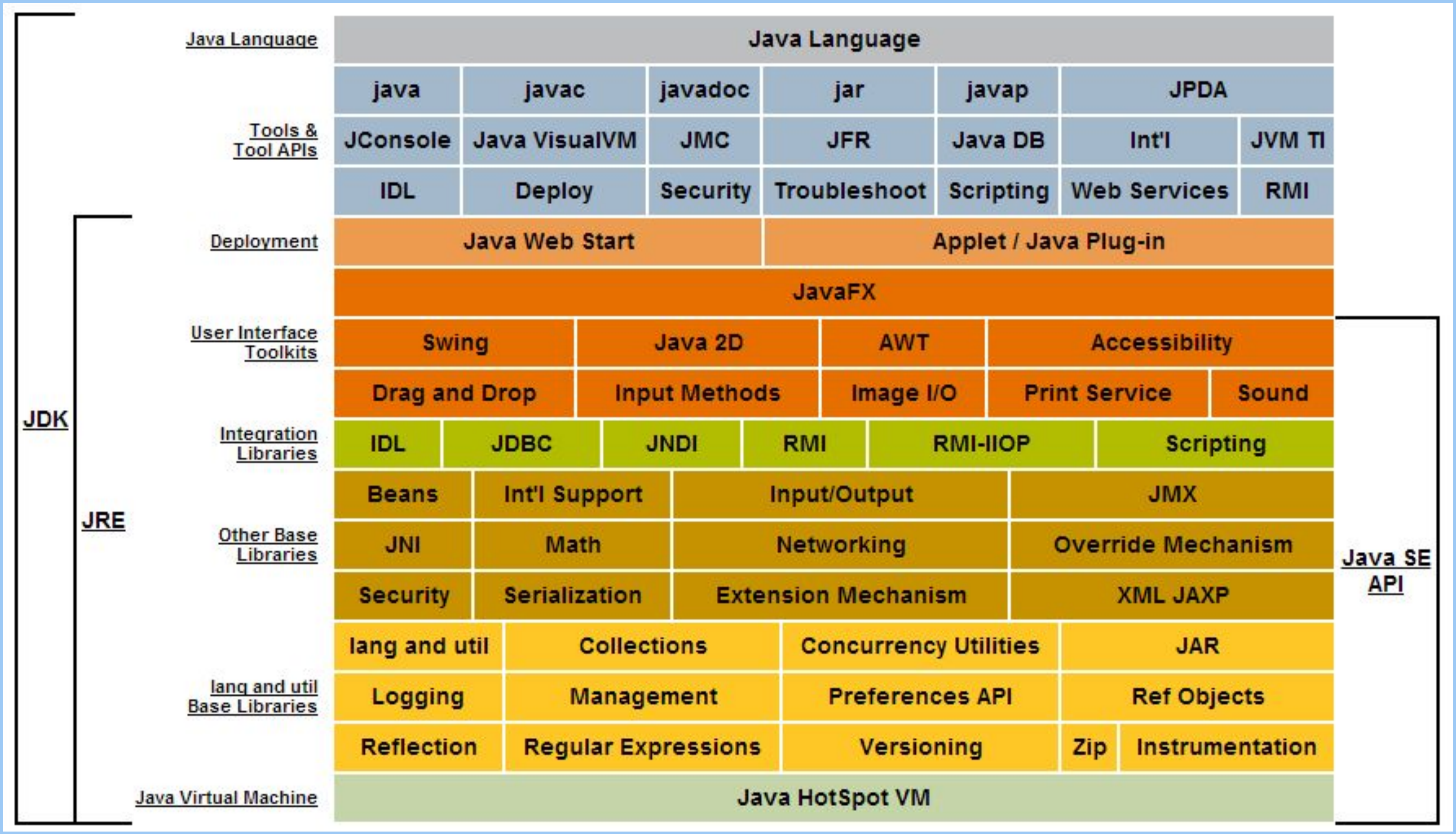
- JDK Alpha and Beta (1995)
- JDK 1.0 (January 23, 1996)
- JDK 1.1 (February 19, 1997)
- J2SE 1.2 (December 8, 1998)
- J2SE 1.3 (May 8, 2000)
- J2SE 1.4 (February 6, 2002)
- J2SE 5.0 (September 30, 2004)
- Java SE 6 (December 11, 2006)
- Java SE 7 (July 28, 2011)
- Java SE 8 (January 19, 2021)

- To Check your java version:
  - Type “java –version” to your command line.

Versions of Java

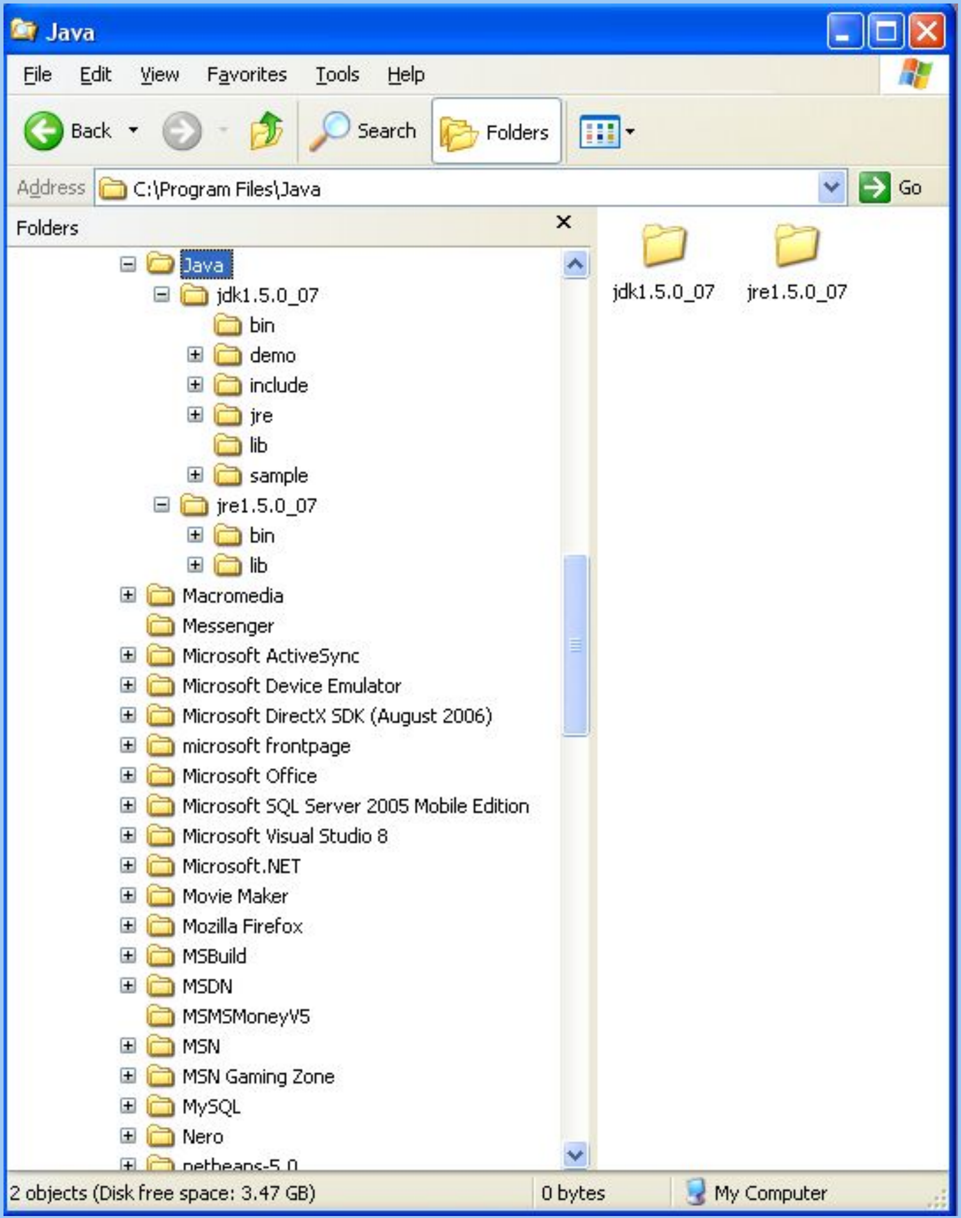
- Three versions of the Java 2 Platform, targeted at different uses
  - Java 2 Micro Edition (J2ME)
    - Very small Java environment for smart cards, pages, phones, and set-top boxes
    - Subset of the standard Java libraries aimed at limited size and processing power
  - Java 2 Standard Edition (J2SE)
    - The basic platform
    - J2SE can be used to develop client-side standalone applications or applets.
  - Java 2 Enterprise Edition (J2EE)
    - For business applications, web services, mission-critical systems
    - Transaction processing, databases, distribution, replication

J2SE 8.0


JDK
JRE
Java Language
Java Language
Tools & Tool APIs
java
javac
javadoc
jar
javap
JPDA
JConsole
Java VisualVM
JMC
JFR
Java DB
Int'l
JVM TI
IDL
Deploy
Security
Troubleshoot
Scripting
Web Services
RMI
Deployment
Java Web Start
Applet / Java Plug-in
JavaFX
User Interface Toolkits
Swing
Java 2D
AWT
Accessibility
Drag and Drop
Input Methods
Image I/O
Print Service
Sound
Integration Libraries
IDL
JDBC
JNDI
RMI
RMI-IIOP
Scripting
Other Base Libraries
Beans
Int'l Support
Input/Output
JMX
JNI
Math
Networking
Override Mechanism
Security
Serialization
Extension Mechanism
XML JAXP
lang and util Base Libraries
lang and util
Collections
Concurrency Utilities
JAR
Logging
Management
Preferences API
Ref Objects
Reflection
Regular Expressions
Versioning
Zip
Instrumentation
Java Virtual Machine
Java HotSpot VM
Java SE API

JRE and JDK

- **JRE: J2SE Runtime Environment**
  - provides
    - libraries,
    - Java virtual machine,
    - other components necessary for you to *run* applets and applications
- **JDK: J2SE Development Kit**
  - includes
    - JRE
    - command-line development tools such as compilers and debuggers



JVM

- **JVM: Java Virtual Machines**
  - The Java virtual machine is an abstract computing machine that has an instruction set and manipulates memory at run time.
  - The Java virtual machine is ported to different platforms to provide hardware- and operating system-independence.

Key Java Packages and Protocols

- Core Technologies
  - JDBC
  - RMI
  - Jini (Device Networking)
  - JavaBeans
  - Swing
  - Java 2D
- Standard Extensions
  - Servlets and JavaServer Pages (JSP)
  - Enterprise Java Beans
  - Java 3D

Java Packages and Protocols: JDBC (Java DataBase Connectivity)

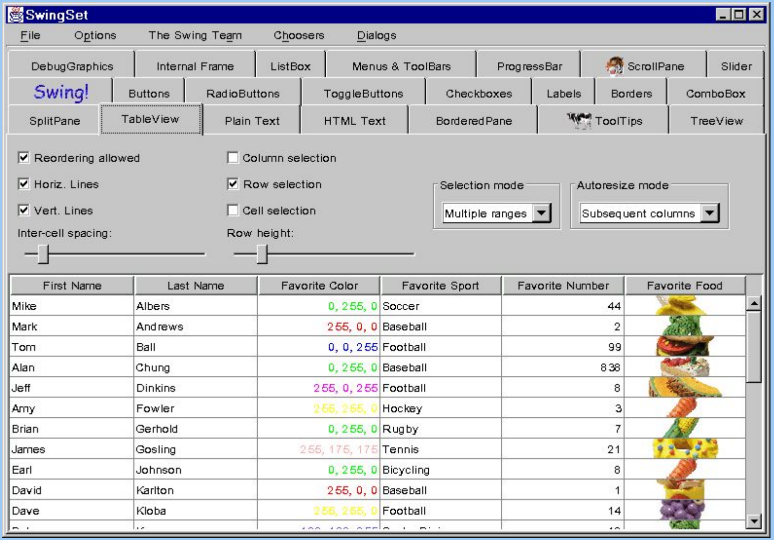
- API to access database
    - Requires server-specific driver on client. No change to server.

Java Packages and Protocols: Remote Method Invocation (RMI)

- Built-in Distributed Object Protocol
  - RMI lets a developer access a Java object and manipulate it in the normal manner. Behind the scenes, each function call really goes over the network to a remote object.
  - restricted to Java-to-Java communication

Java Packages and Protocols: Swing

- Standard GUI-control (widget) library in Java 2
- Many more built-in controls
- Much more flexible and customizable
- Includes many small features aimed at commercial applications
  - Tooltips, tabbed panes, on-line help, HTML support, dockable toolbars, multi-document interface, etc.
- Look and feel can be changed at run time



[✓] Reordering allowed
[✓] Horiz. Lines
[✓] Vert. Lines
[ ] Column selection
[✓] Row selection
[ ] Cell selection
Inter-cell spacing:
Row height:
Selection mode: Multiple ranges
Autoresize mode: Subsequent columns

First Name	Last Name	Favorite Color	Favorite Sport	Favorite Number	Favorite Food
Mike	Albers	0, 255, 0	Soccer	44	
Mark	Andrews	255, 0, 0	Baseball	2	
Tom	Ball	0, 0, 255	Football	99	
Alan	Chung	0, 255, 0	Baseball	838	
Jeff	Dinkins	255, 0, 255	Football	8	
Amy	Fowler	255, 255, 0	Hockey	3	
Brian	Gerhold	0, 255, 0	Rugby	7	
James	Gosling	255, 175, 175	Tennis	21	
Earl	Johnson	0, 255, 0	Bicycling	8	
David	Karlton	255, 0, 0	Baseball	1	
Dave	Kloba	255, 255, 0	Football	14	

Software required:

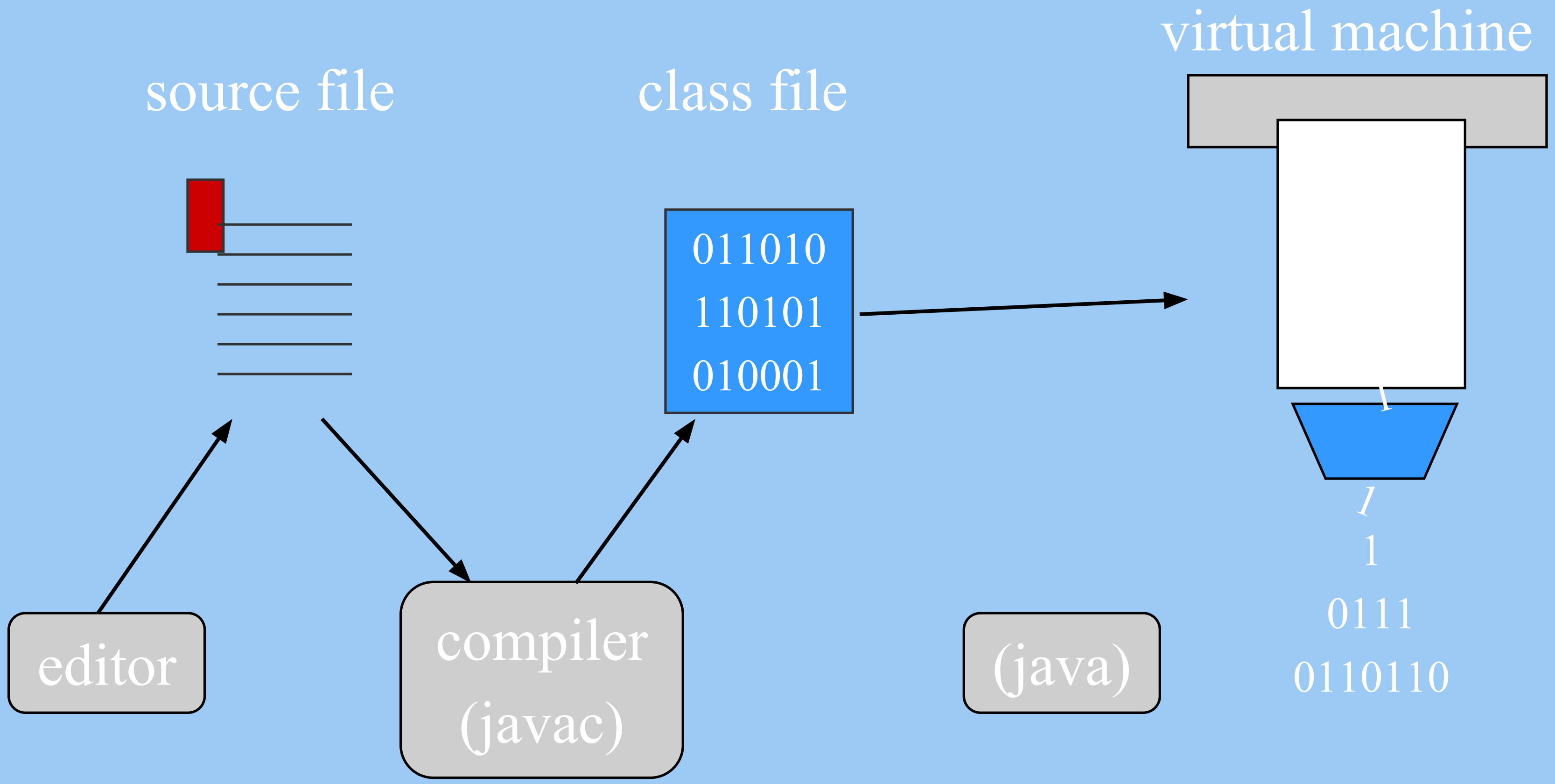
- Java Compiler:
  - JDK (Java Development Kit):
    - J2SE 8.2
  - Download from:
    - http://www.oracle.com/technetwork/java/javase/downloads/index.html

- Java Editor
  - JCreator 3.00 or JCreator 3.50 (simple)
  - NetBeans (little bit slow, good for software development)
  - JBuilder
  - J#
  - IntelliJ
  - Eclipse
  - TextPad
  - BlueJ

Getting Started with Java Programming

- A Simple Java Application
- Compiling Programs
- Executing Applications

The edit-compile-execute cycle



Standard Java files

- **source files: *.java**
  Java source files contain the source code in readable form, as typed in by the programmer.

class files: *.class
Java class files contain byte code (a machine readable version of the class). They are generated by the compiler from the source file.

Command line invocation

- compilation and execution of Java in JDK are done from a command line
- On Microsoft systems: DOS shell
- On Unix: Unix shell

First Java Program

Comments

```
/* My first simple Java program */
```

All Java programs have a main function; they also start at main

```
public class Hello
{
    public static void main (String[] args)
    {
        System.out.println ("Hello World");
    }
}
```

Function to print to screen

What to print

Braces indicate start and end of main

End of statement

Compiling

- Name of the JDK compiler: **javac**
- To invoke:

```
javac <source name>
```

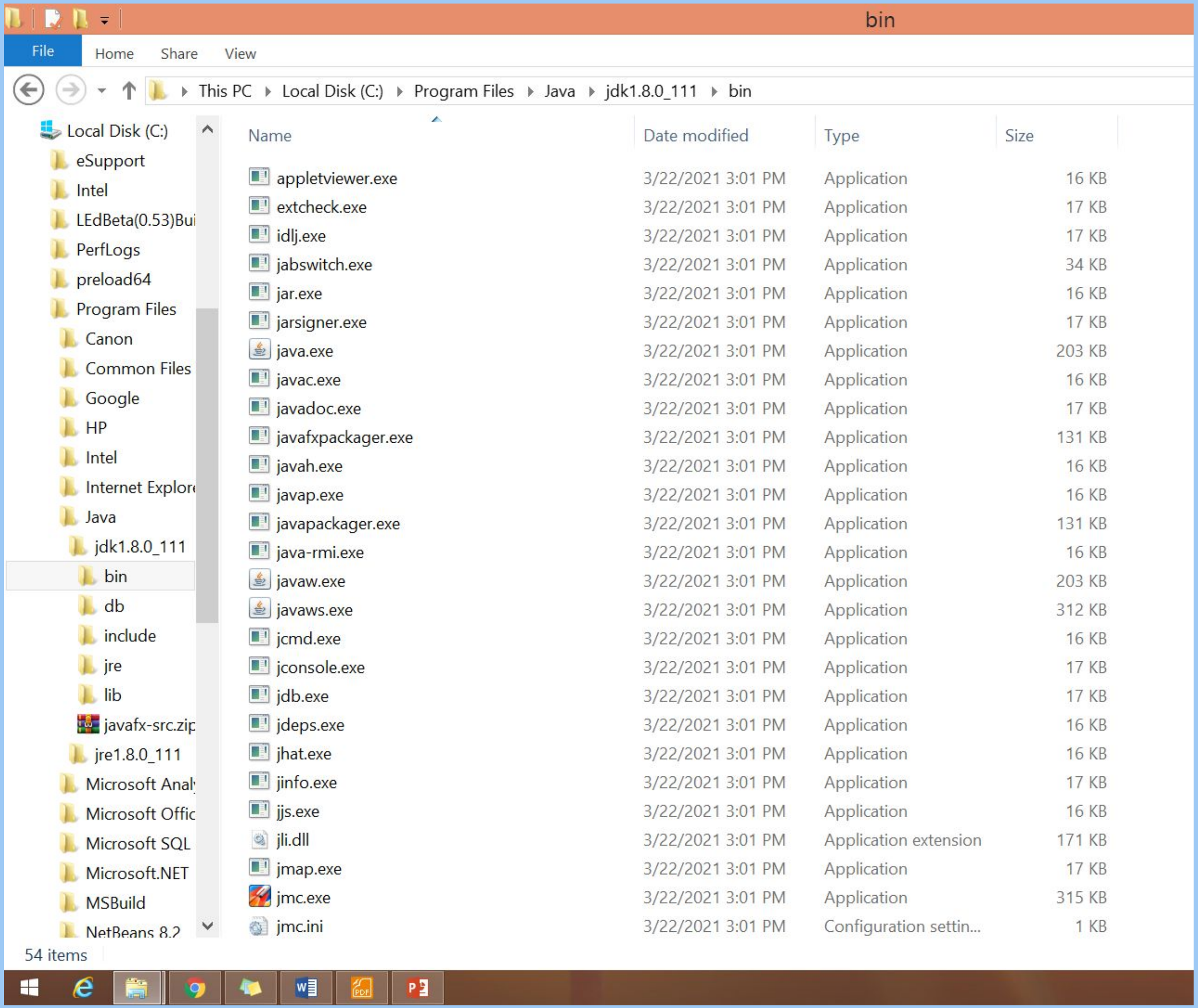
- compiles <source name> and all classes it depends on
- Example:

```
cd C:\example
javac Hello.java
```

Execution

- `C:\example> java Hello`
- "java" starts the Java virtual machine
- Wrong! `> java Hello.class`
- The named class is loaded and execution is started
- Other classes are loaded as needed
- Only possible if class has been compiled

Java/jdk1.8.0_111/bin



Name	Date modified	Type	Size
appletviewer.exe	3/22/2021 3:01 PM	Application	16 KB
extcheck.exe	3/22/2021 3:01 PM	Application	17 KB
idlj.exe	3/22/2021 3:01 PM	Application	17 KB
jabswitch.exe	3/22/2021 3:01 PM	Application	34 KB
jar.exe	3/22/2021 3:01 PM	Application	16 KB
jarsigner.exe	3/22/2021 3:01 PM	Application	17 KB
java.exe	3/22/2021 3:01 PM	Application	203 KB
javac.exe	3/22/2021 3:01 PM	Application	16 KB
javadoc.exe	3/22/2021 3:01 PM	Application	17 KB
javafxpackager.exe	3/22/2021 3:01 PM	Application	131 KB
javah.exe	3/22/2021 3:01 PM	Application	16 KB
javap.exe	3/22/2021 3:01 PM	Application	16 KB
javapackager.exe	3/22/2021 3:01 PM	Application	131 KB
java-rmi.exe	3/22/2021 3:01 PM	Application	16 KB
javaw.exe	3/22/2021 3:01 PM	Application	203 KB
javaws.exe	3/22/2021 3:01 PM	Application	312 KB
jcmd.exe	3/22/2021 3:01 PM	Application	16 KB
jconsole.exe	3/22/2021 3:01 PM	Application	17 KB
jdb.exe	3/22/2021 3:01 PM	Application	17 KB
jdeps.exe	3/22/2021 3:01 PM	Application	16 KB
jhat.exe	3/22/2021 3:01 PM	Application	16 KB
jinfo.exe	3/22/2021 3:01 PM	Application	17 KB
jjs.exe	3/22/2021 3:01 PM	Application	16 KB
jli.dll	3/22/2021 3:01 PM	Application extension	171 KB
jmap.exe	3/22/2021 3:01 PM	Application	17 KB
jmc.exe	3/22/2021 3:01 PM	Application	315 KB
jmc.ini	3/22/2021 3:01 PM	Configuration settin...	1 KB

Problems on compiling:

- **Compile the program:**
  - compile Hello.java by using the following command:

```
javac Hello.java
```

it generates a file named Hello.class

☹

```
'javac' is not recognized as an internal or  external
command, operable program or hatch file.
 javac: Command not found
```

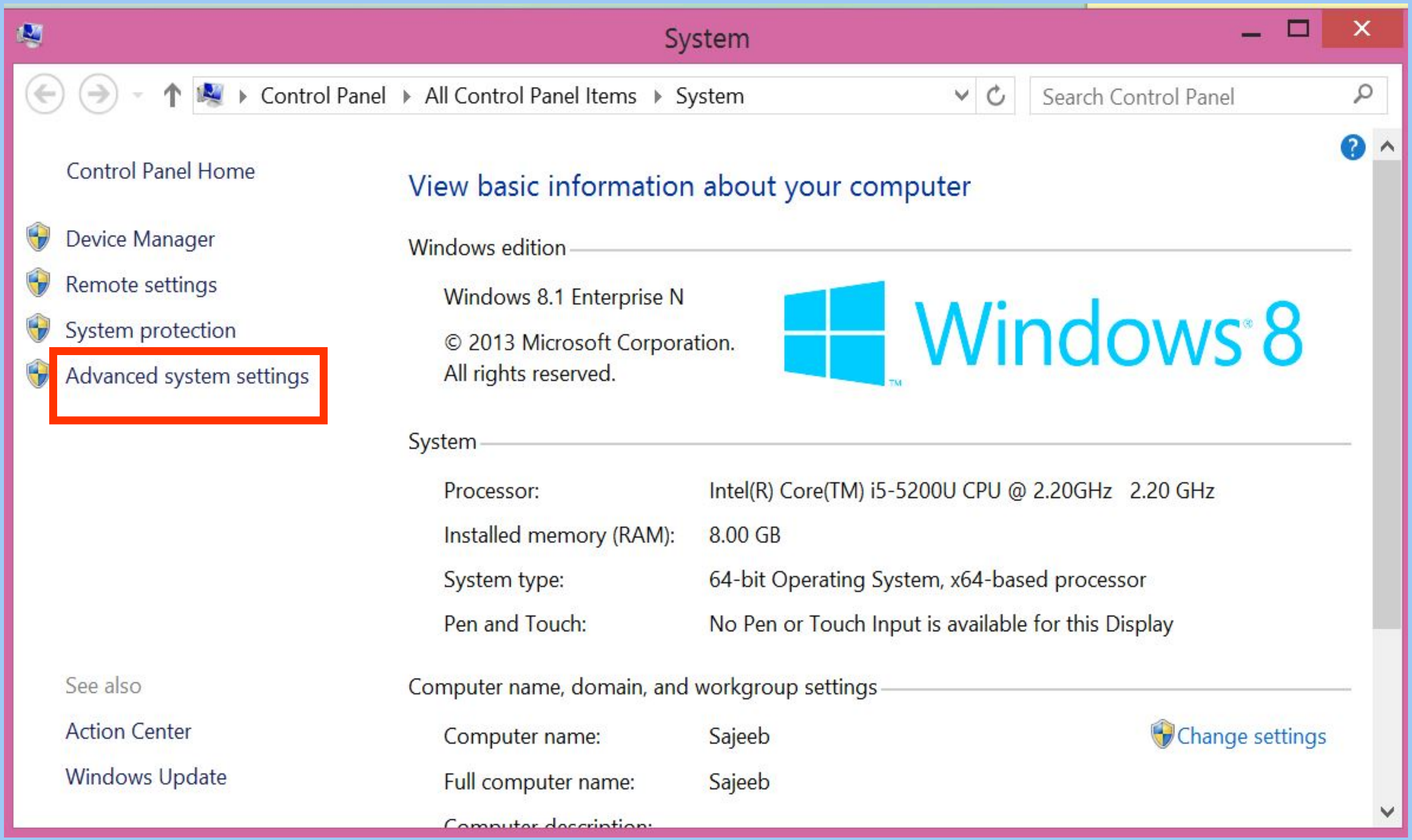
if you see one of these errors, you have two choices:
1) specify the full path in which the `javac` program locates every time. For example:

```
C:\program files\java\jdk1.5.0_07\bin\javac Hello.java
```

2) set the PATH environment variable

Windows Procedure to Set Path

- Step 1
  - Choose Control Panel-> System.
  - Choose Advanced System Settings



View basic information about your computer

Windows edition

Windows 8.1 Enterprise N
© 2013 Microsoft Corporation. All rights reserved.

System

Processor:	Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz 2.20 GHz
Installed memory (RAM):	8.00 GB
System type:	64-bit Operating System, x64-based processor
Pen and Touch:	No Pen or Touch Input is available for this Display

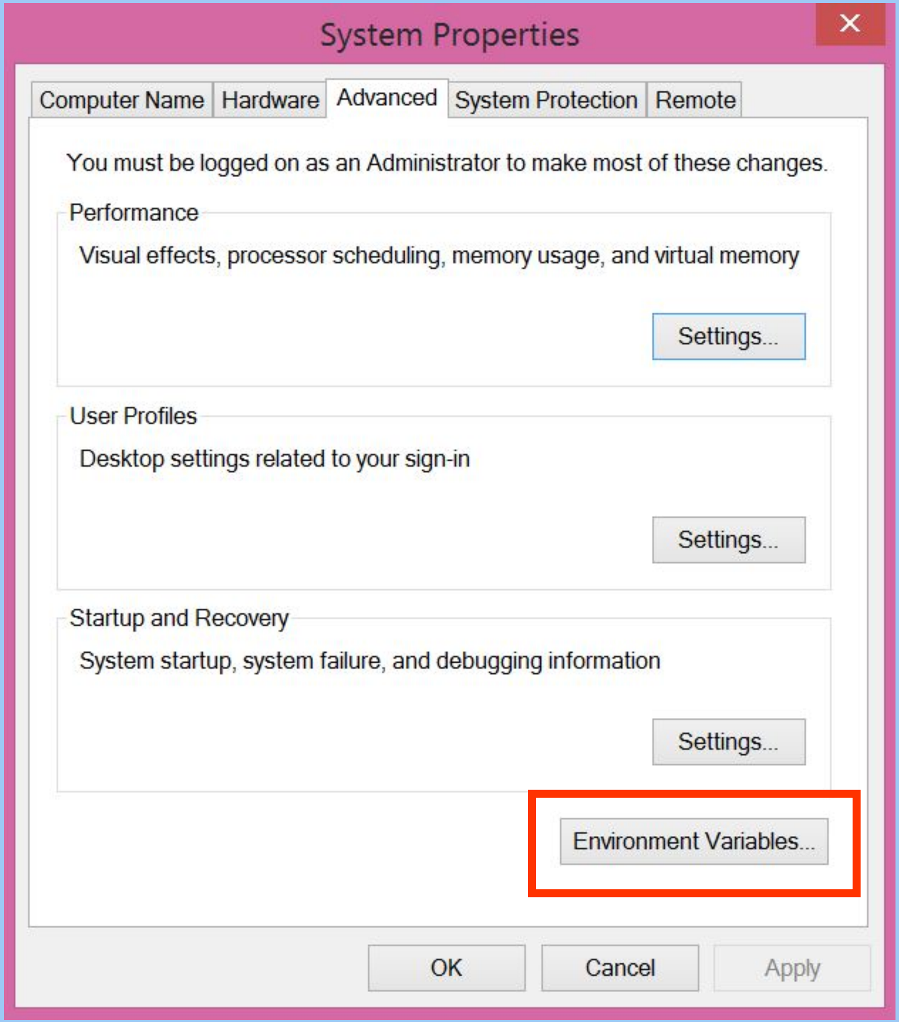
Computer name, domain, and workgroup settings

Computer name:	Sajeeb
Full computer name:	Sajeeb
Computer description:	

Windows Procedure to Set Path

- Step 2

Choose
Environment Variables.



System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

Performance

Visual effects, processor scheduling, memory usage, and virtual memory

Settings...

User Profiles

Desktop settings related to your sign-in

Settings...

Startup and Recovery

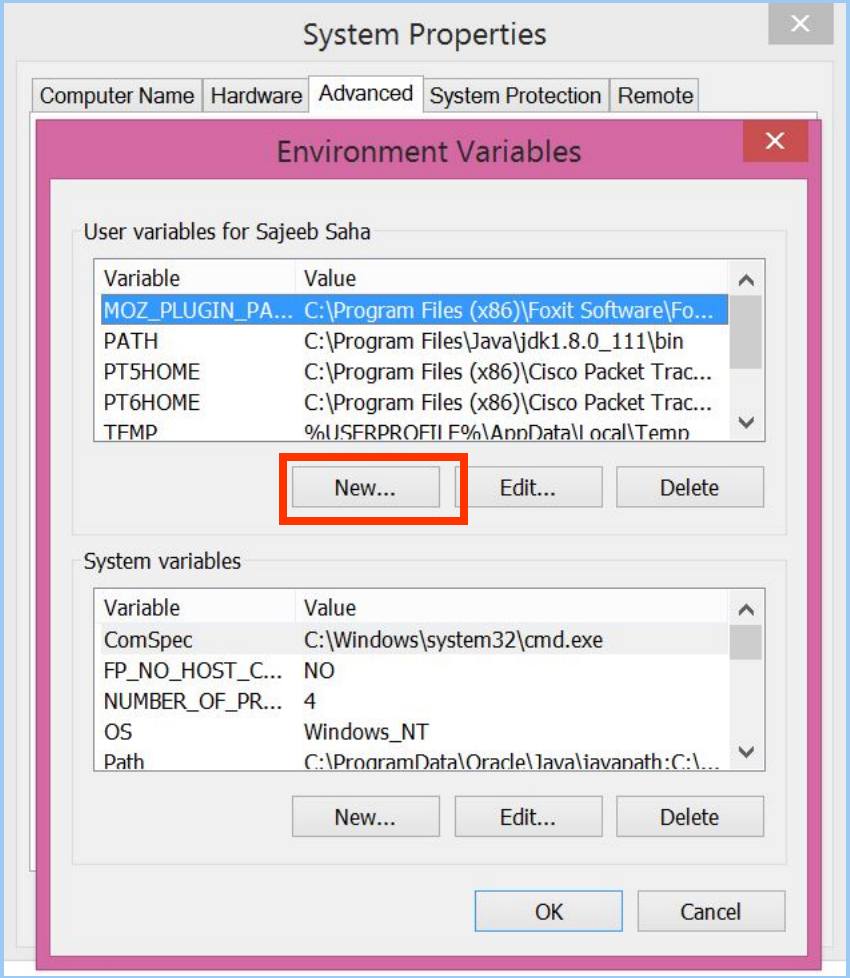
System startup, system failure, and debugging information

Settings...

Environment Variables...

OK | Cancel | Apply

Windows Procedure to Set Path

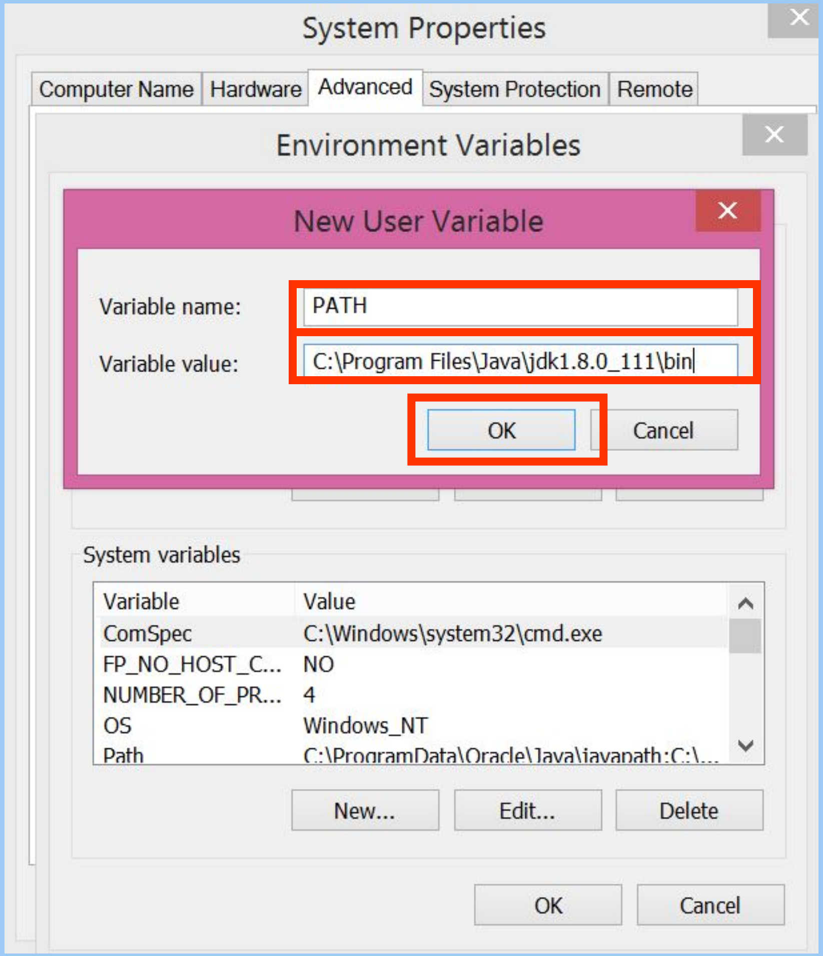


User variables for Sajeeb Saha

Variable	Value
MOZ_PLUGIN_PA...	C:\Program Files (x86)\Foxit Software\Fo...
PATH	C:\Program Files\Java\jdk1.8.0_111\bin
PT5HOME	C:\Program Files (x86)\Cisco Packet Trac...
PT6HOME	C:\Program Files (x86)\Cisco Packet Trac...
TEMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_PR...	4
OS	Windows_NT
Path	C:\ProgramData\Oracle\Java\javapath;C:\...



Variable name: PATH

Variable value: C:\Program Files\Java\jdk1.8.0_111\bin

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_PR...	4
OS	Windows_NT
Path	C:\ProgramData\Oracle\Java\javapath;C:\...

Reading:

- Java: The Complete Reference - Harbert Schildt
  - Chapter 1

Homework:

- Write a program that writes the following to the screen:
  - Your Name
  - Your Roll
  - Your Email address
- Find the difference between System.out.println, System.out.print
- What would be the result for the following:
  - System.out.printf("%s\n%10s\n%-10s\n","Hello","Hello","Hello");
- Try to print an double variable in various format using printf and println function:
  - (to declare double variable you can use: double abc=10.57645;)
  - Various format means: exponent format, 2 digits after decimal point, 0-padded format, left justified format, right justified fomat