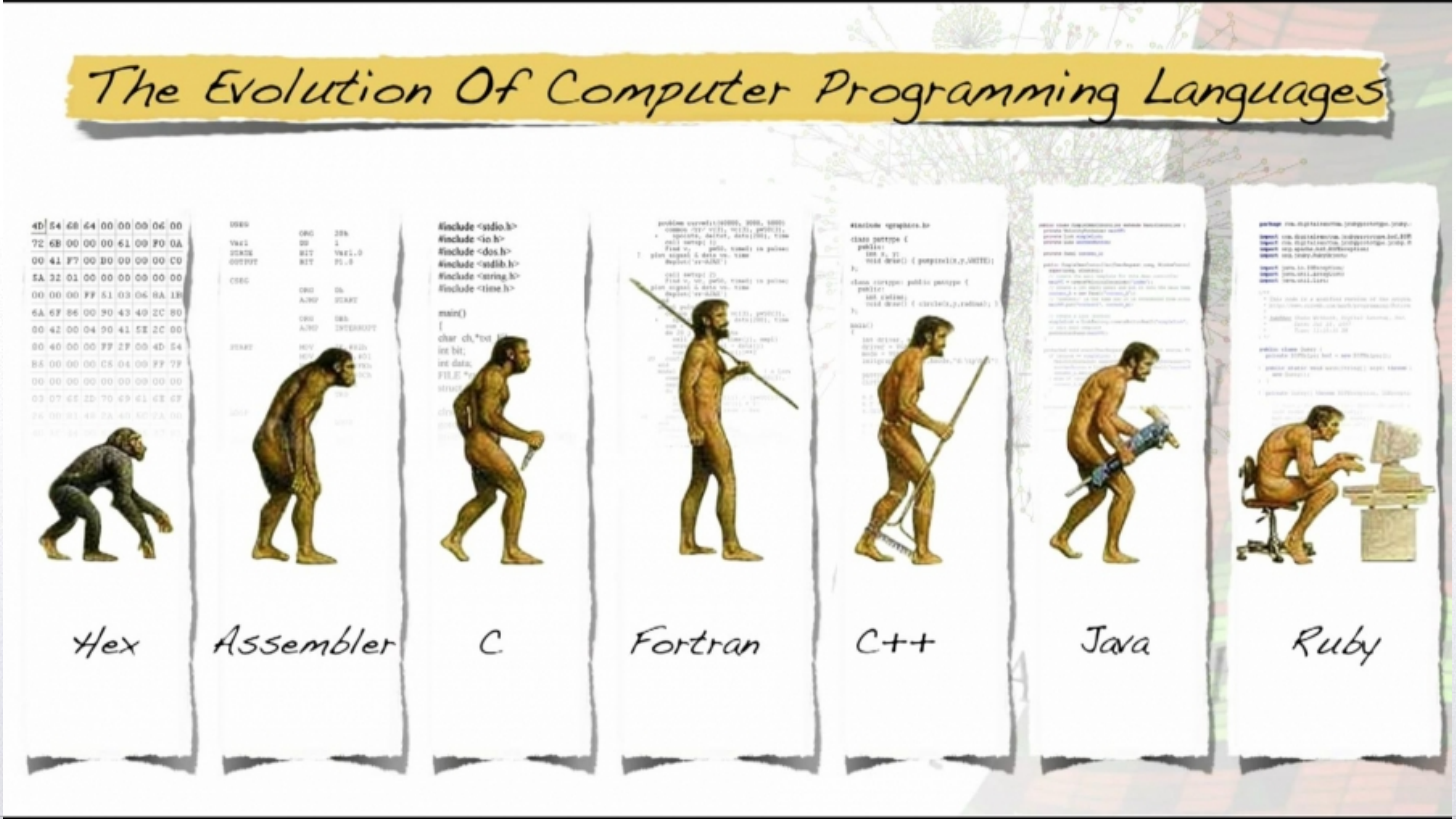
Chapter 1
Java Fundamental

Yang Wang
wyang AT njnet.edu.cn

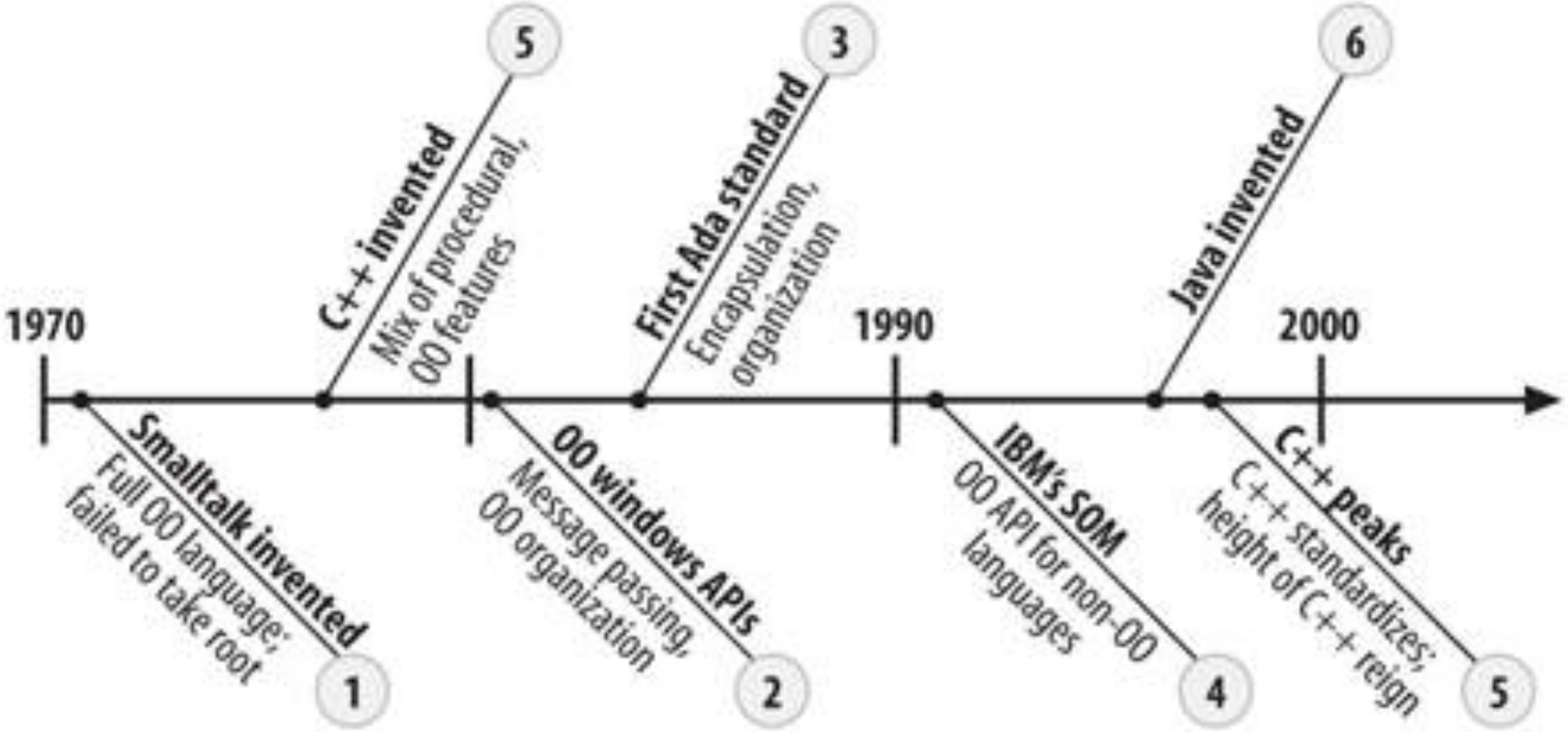
Outline

- Evolution of Java
- Java Operating Mechanism
- Java Developing Environment
- Java Basic Grammar

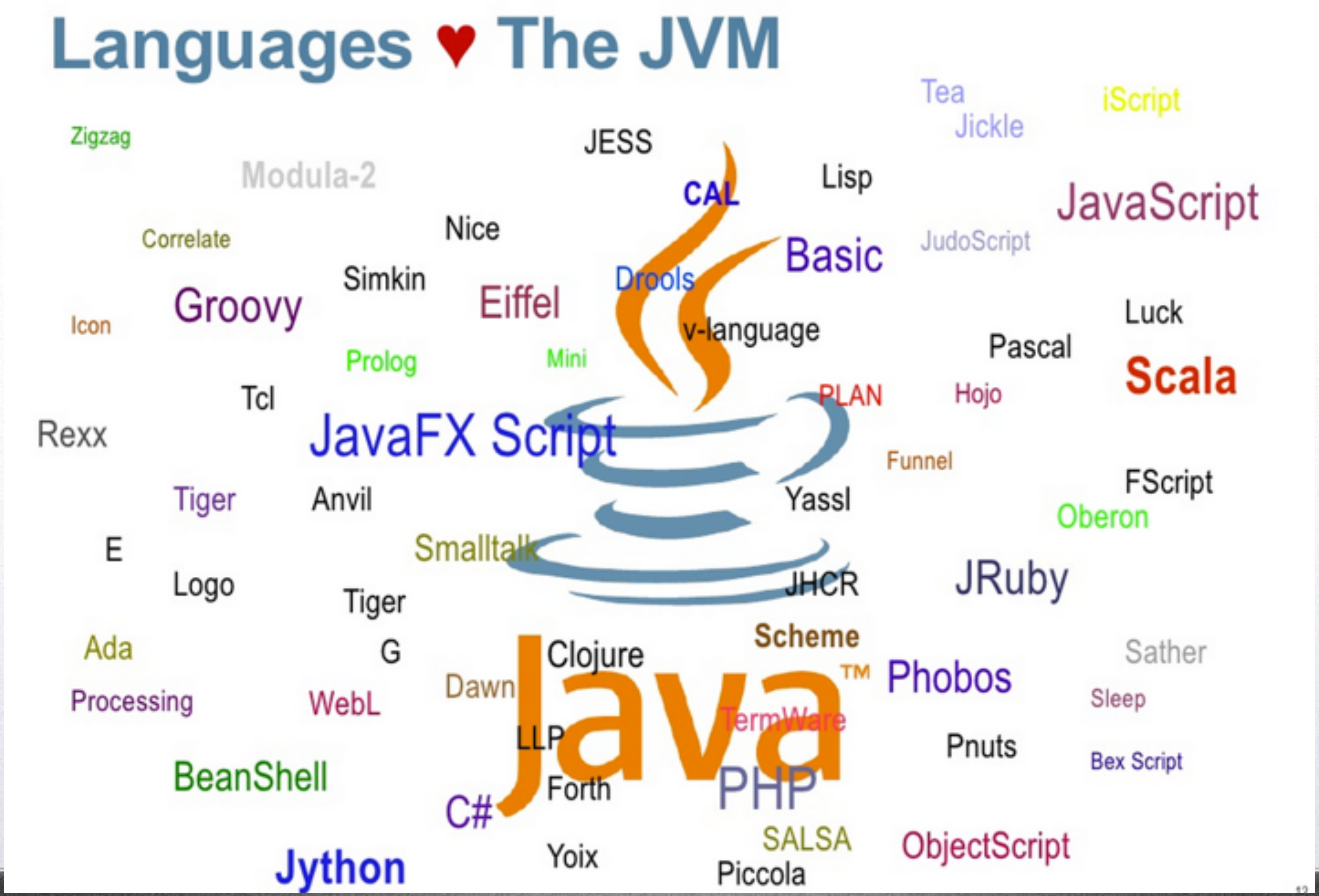
Evolution of Java


The Evolution Of Computer Programming Languages
Hex
Assembler
C
Fortran
C++
Java
Ruby

Evolution of Java – Success of OOP


1970
1990
2000
5
C++ invented
Mix of procedural, OO features
3
First Ada standard
Encapsulation, organization
6
Java invented
1
Smalltalk invented
Full OO language; failed to take root
2
OO windows APIs
Message passing, OO organization
4
IBM's SOM
OO API for non-OO languages
5
C++ peaks
C++ standardizes; height of C++ reign

Language Band



Life of Java

- Past
  - Resource-limited Device
  - C++
  - Green Project
  - Oak
  - Mosaic / Mark Ardreesen
  - HotJava

- Present
  - WWW
  - Internet
  - Mobile
  - Sun与Microsoft
  - Java与开源
  - Java与Oracle

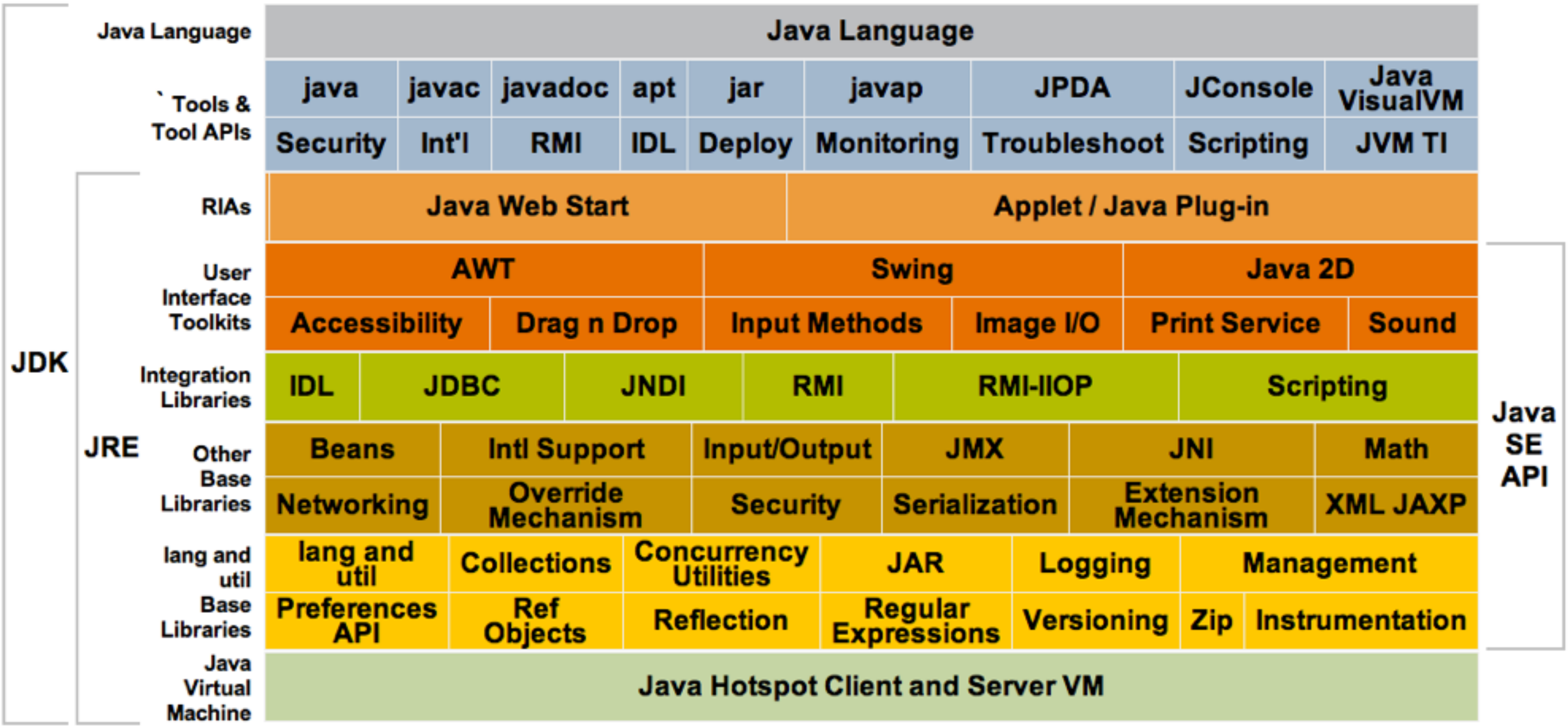
- Future
  - Module System
  - Mutli Language
  - Multicore
  - 64-bit VM

Java Operating Mechanism

JDK and JRE

- JDK – Java Development Toolkit
  - J2SE – Java 2 Standard Edition
  - J2EE – Java 2 Enterprise Edition
  - J2ME – Java 2 Micro Edition
  - Java Card
- JRE – Java Runtime Environment

JDK and JRE



JDK

JRE

Java SE API

Java Language: Java Language

` Tools & Tool APIs: java | javac | javadoc | apt | jar | javap | JPDA | JConsole | Java VisualVM

Security | Int'l | RMI | IDL | Deploy | Monitoring | Troubleshoot | Scripting | JVM TI

RIAs: Java Web Start | Applet / Java Plug-in

User Interface Toolkits: AWT | Swing | Java 2D

Accessibility | Drag n Drop | Input Methods | Image I/O | Print Service | Sound

Integration Libraries: IDL | JDBC | JNDI | RMI | RMI-IIOP | Scripting

Other Base Libraries: Beans | Intl 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

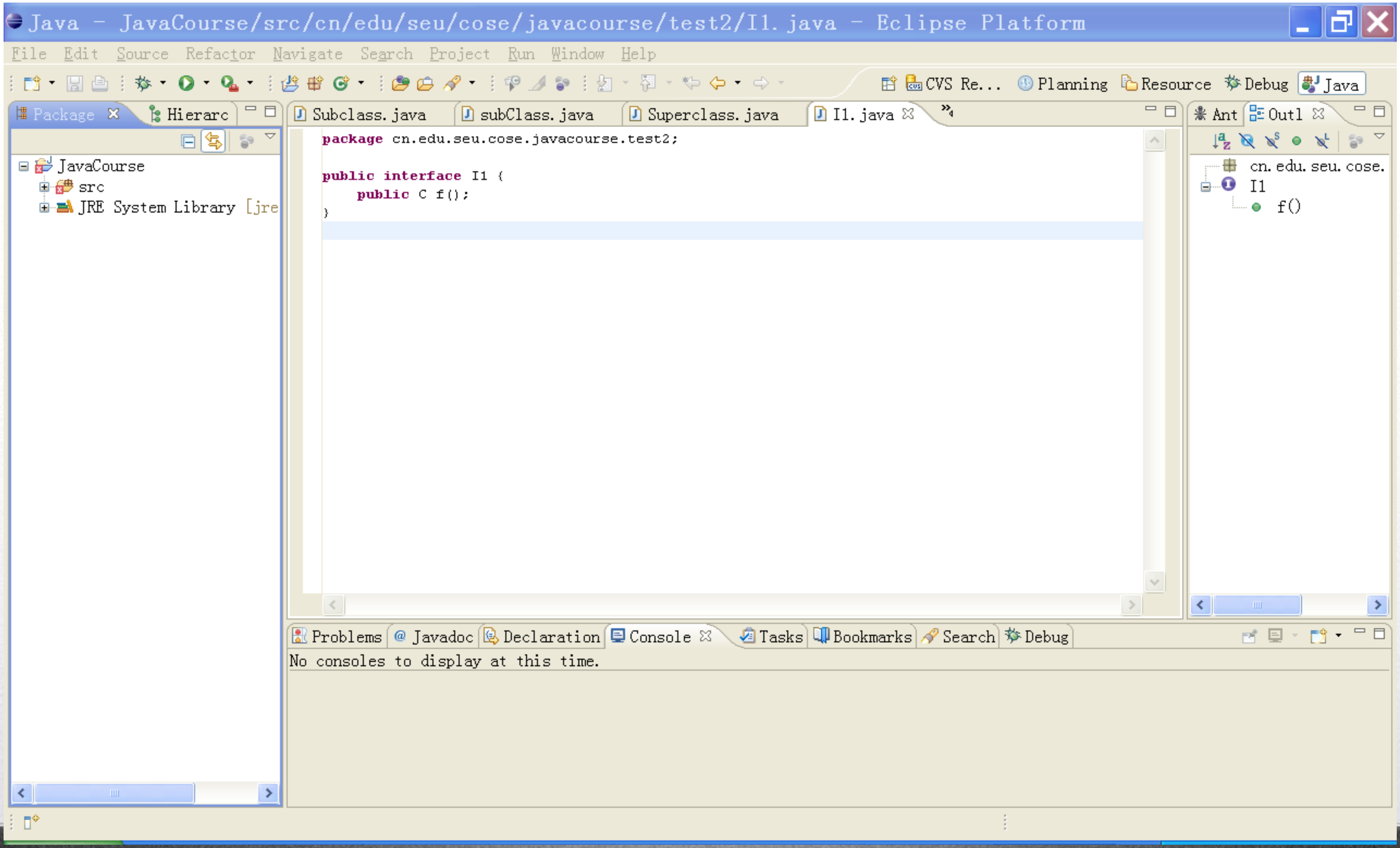
Java Virtual Machine: Java Hotspot Client and Server VM

Java Developing Environment

Java Developing Environment

- Text editor
  - vi with extra packages
- IDE
  - JCreator
  - JBuilder
  - *Eclipse
  - *Netbeans
  - IntelliJ IDEA
  - MyEclipse

Eclipse

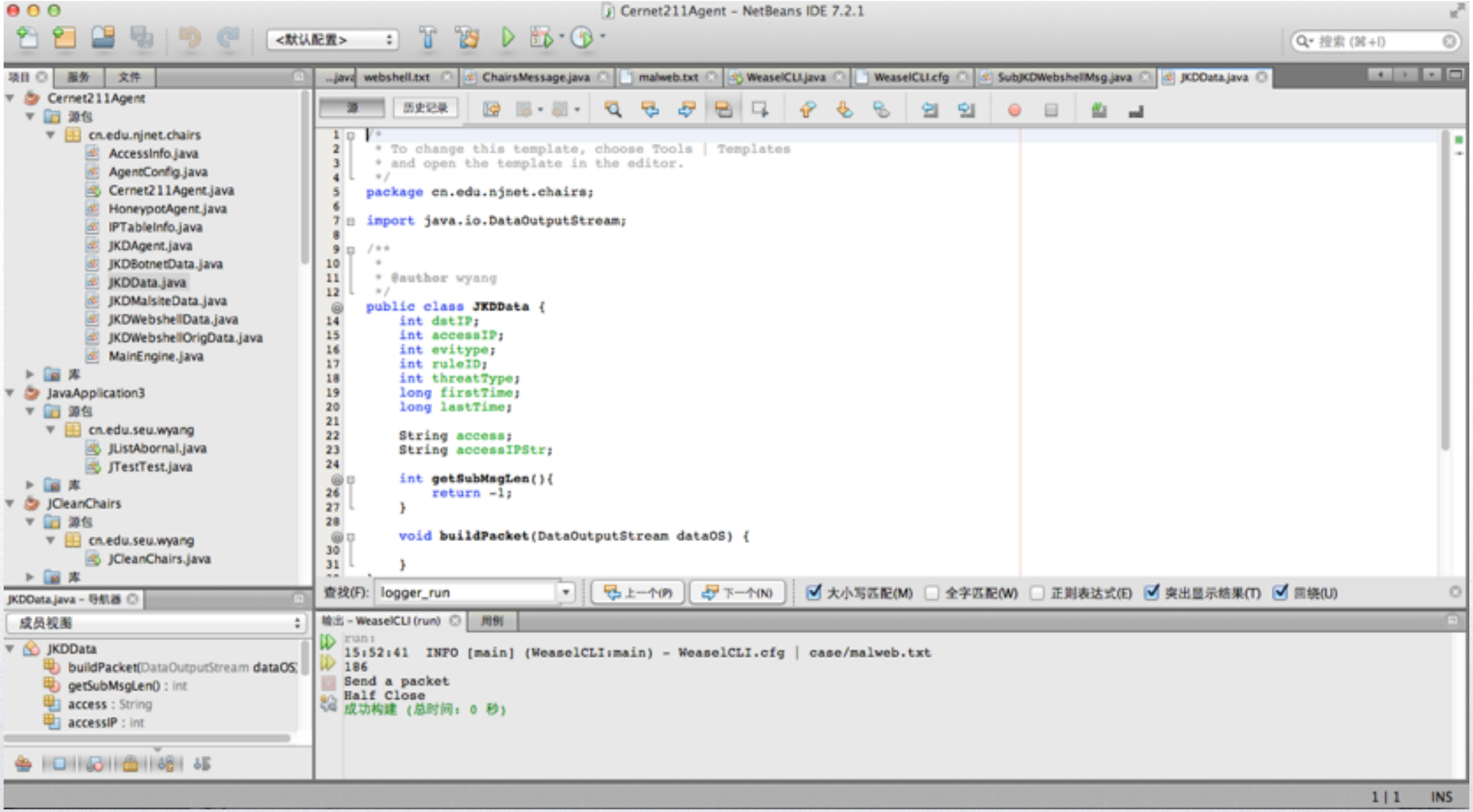


```java
package cn.edu.seu.cose.javacourse.test2;

public interface I1 {
    public C f();
}
```

```
No consoles to display at this time.
```

Netbeans



```java
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package cn.edu.njnet.chairs;

import java.io.DataOutputStream;

/**
 *
 * @author wyang
 */
public class JKDData {
    int datIP;
    int accessIP;
    int evitype;
    int ruleID;
    int threatType;
    long firstTime;
    long lastTime;

    String access;
    String accessIPStr;

    int getSubMsgLen(){
        return -1;
    }

    void buildPacket(DataOutputStream dataOS) {

    }
```

```
run:
15:52:41  INFO [main] (WeaselCLI:main) - WeaselCLI.cfg | case/malweb.txt
186
Send a packet
Half Close
成功构建 (总时间: 0 秒)
```

Java Basic Grammar

Basic Grammer

- First Contact with Class
- Main Function
- Variable
- Primary Types
- Operators
- Control Flow
- Keywords
- Comments

First Contact with Class

- Nearly Everything is Class and Object
- Each class Correspond to a File
- Class Name <-> File Name
- No Header File
- Something like but not equal to “include”

First Contact with Class

- HelloWorld.java

```
import java.util.*;
Class HelloWorld {
    public static void main(String args[]) {
        System.out.println(“Hello World”)
    }
}
```

First Contact with Class

- javac HelloWorld.java
  - HelloWorld.java => HelloWorld.class
- java HelloWorld
  - JVM initialized
  - Load HelloWorld.class
  - Execute HelloWorld.main()

Main Function

- A project has multiple class
- each class can has its own main function
- you can tell JVM which class is main class

Main Function

- A special function
- must be public, static, void
- param must be String[]
- type is important, name is not

Java Primary Data Types

Type	Size	Range	Wrapper
boolean			Boolean
char			Character
byte			Byte
short			Short
int			Integer
long			Long
float			Float
double			Double
void			Void

Java Primary Data Types

Type	Size	Range	Wrapper
boolean	1	true/false	Boolean
char	16	Unicode	Character
byte	8	[-128, 127]	Byte
short	16	[-2^{15}, 2^{15}-1]	Short
int	32	[-2^{31},2^{31}-1]	Integer
long	64	[-2^{63},2^{63}-1]	Long
float	32	$3.4*10^{38}$	Float
double	64	$1.7*10^{308}$	Double
void			Void

Encoding for Integer

- True Code
  - 3 : 00 00 00 11; -3: 10 00 00 11
- One's Complement Code
  - 3 : 00 00 00 11; -3 : 11 11 11 00
- Two's Complement Code
  - 3: 00 00 00 11; -3 : 11 11 11 01

Encoding for Float

- M * B^E
- M : 尾数 1.xxxxx
- B : 基数 2
- E : 阶数
- 特殊的数 :NaN : Not a Number 0.0f/0.0f

Conversion Between Values

- From Low Accuracy to High Accuracy: Auto
  - double d = 10;
- From High Accuracy to Low Accuracy: Cast
  - int t = (int)10.2 ;

Wrappers

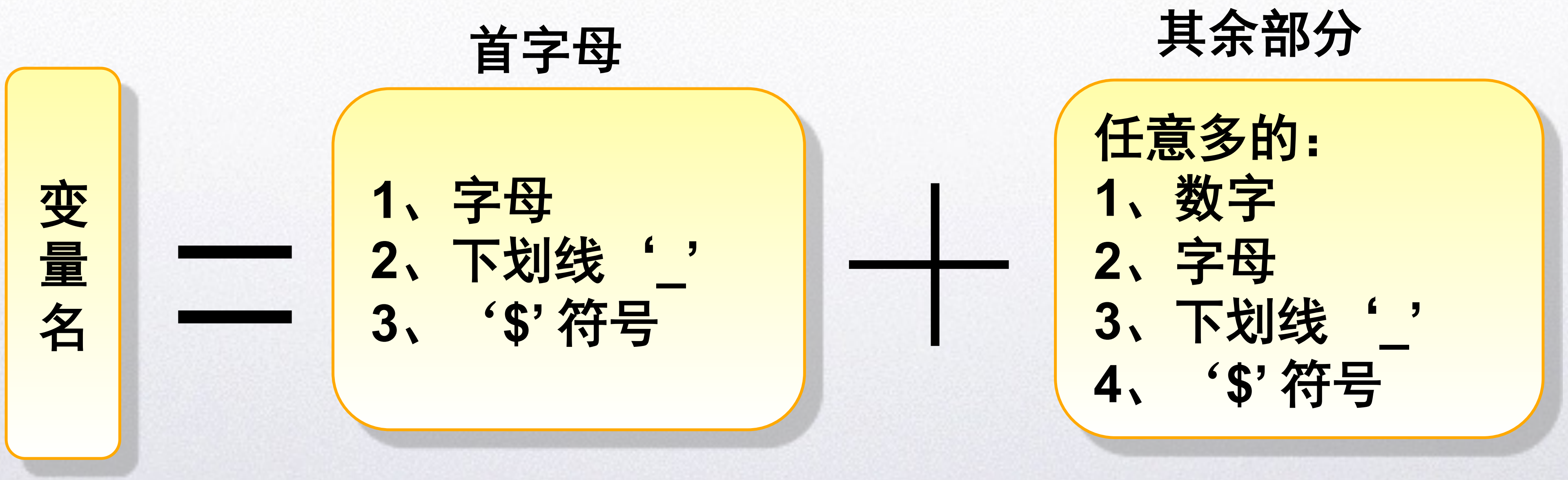
- Values of Primary Types are NOT Objects!
- Each Primary type has a corresponding wrapper to wrap a value into an object:
    - Integer a = 473;
    - System.out.println(a.compareTo(new Integer(472)));

Wrappers More

- Auto Boxing
- Parsing from different Source
  - Integer.parseInt()
- Constants
  - MIN, MAX

Naming of Variables

- Principle:
  - A names should reflect the meaning of a variable
  - Precise
  - First letter in lower case (different with classes)



变量名 = 首字母 + 其余部分

首字母

1、字母
2、下划线 ‘_’
3、 ‘$’ 符号

其余部分

任意多的：
1、数字
2、字母
3、下划线 ‘_’
4、 ‘$’ 符号

Java Operator

- Arithmetic operator
- Comparison operator
- Logical operator
- Bitwise operator
- Assignment operator
- Others

Arithmetic Operator

- +、- 、* 、/ 、%
- ++ 、- -

Comparison Operator

- > 、>=
- < 、<=
- == 、!=
- instanceof

Logical Operator

- &、|
- &&、||
- !
- ^

Bitwise Operator

- & | !
- <<
- >>   有符号
- >>> 无符号

Assignment Operator

- =
- += 、-= 、*= 、/= 、% =
- >>= 、<<= 、>>> =

Others

- ? :
- .
- new
- []

Magic Endless Loop

```
int start = ?;

for(int i = start; i<= start+1; i++){

        Thread.sleep(1000);

}
```

Magic Endless Loop

```
? i = ?;
while(i != 0 && i == -i){
    Thread.sleep(1000);
}
```

Magic Endless Loop

```
? i = ?;

while(i == i+1){

    Thread.sleep(1000);

}
```

Magic Endless Loop

```
? i = ?;
while(i != i){
    Thread.sleep(1000);
}
```

Magic Endless Loop

```
? i = ?;

while(i != i+0){

    Thread.sleep(1000);

}
```

Magic Endless Loop

```
? i = ?;

while(i !=0){

    i >>>= 1;

    Thread.sleep(1000);

}
```

Magic Endless Loop

```
? i = ?;

? j = ?;

while(i <=j && i >= j && i !=  j){

    Thread.sleep(1000);

}
```

Control Flow

- if-else
- switch
- while、do-while
- for
- break
- continue
- return

Keywords

```
abstract
assert
boolean
break
byte
case
catch
char
class
const
continue
default
do
double
else
enum
extends
false
final
finally
float
for
goto
if
implements
import
instanceof
int
interface
long
native
new
null
package
private
protected
public
return
short
static
strictfp
super
switch
synchronized
this
throw
throws
transient
true
try
void
volatile
while
```

Comments

- Single Line

```
// It is a single line comment
```

- Multi Line

```
/* It is a multi line comment
 * It is a multi line comment
 */
```

- javadoc

```
/*
 * @param age
 * @return
 */
public void count(int age) {
}
```

Self-teaching

- Javadoc
    - What is Javadoc?
    - How to add comments in program for making a Javadoc?
    - How to generate Javadoc in HTML format?
    - How to search in Javadoc?

Forecast

- OO Concepts
- Class and Objects
- Package
- Field
- Method
- Main method
- Object
- Construct and Initialization
- Access Control