Chapter 1   Java Fundamentals

Xiang Zhang

javaseu@163.com

http://wds.ac.cn/java/

Content
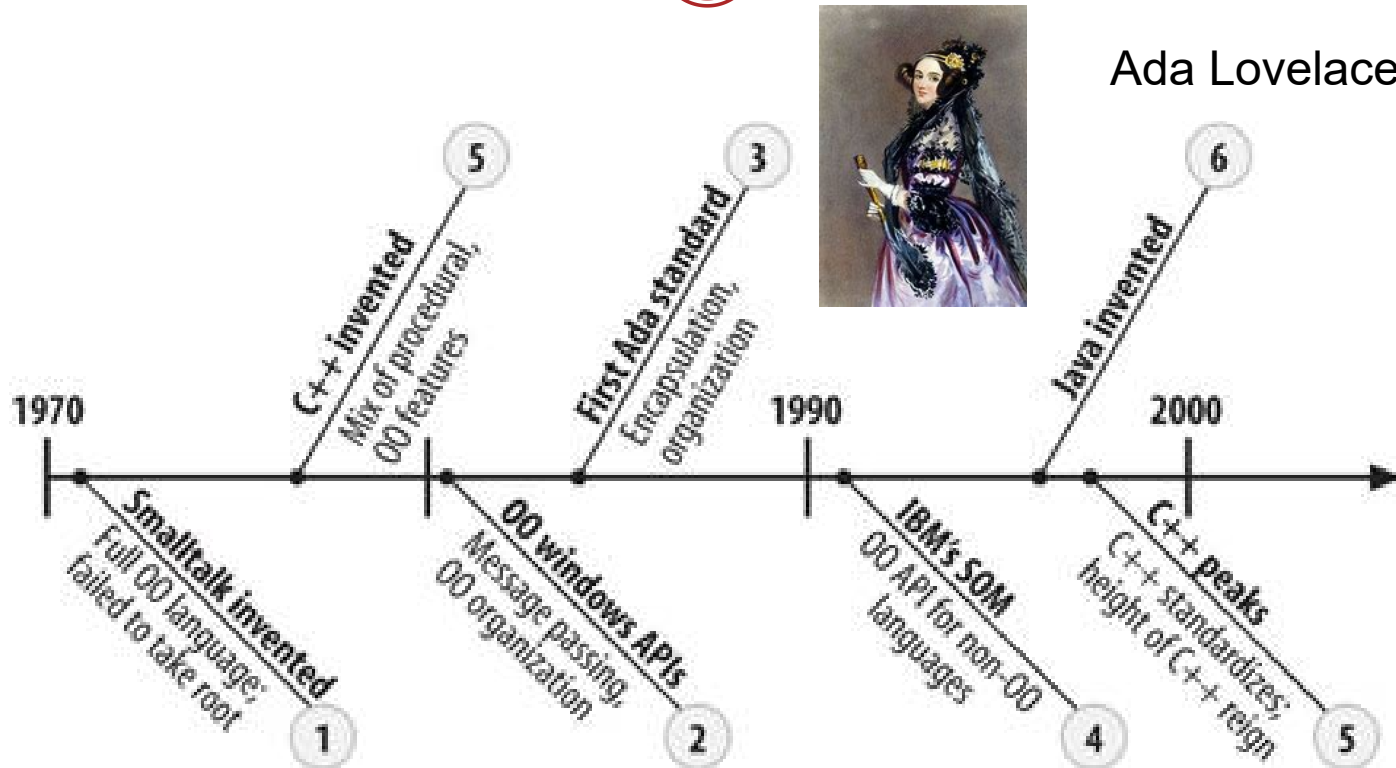
- Evolution of Java

- JDK and JRE

- Java Operating Mechanism

- Java Developing Environment

- Java Primary Data Types

- Java Basic Grammar

Ada Lovelace

from《Beyond Java》
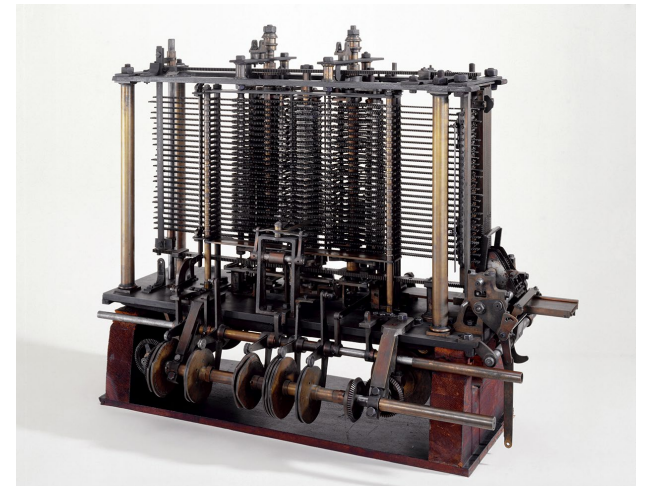
- 10 December 1815 – 27 November 1852)

- English mathematician and writer

- The first to recognize that the machine had applications beyond pure calculation

- She published the first algorithm

mechanical general-purpose computer, the Analytical Engine

Evolution of Java – Life of Java

Open Discussion: Please list some resource-limited devices

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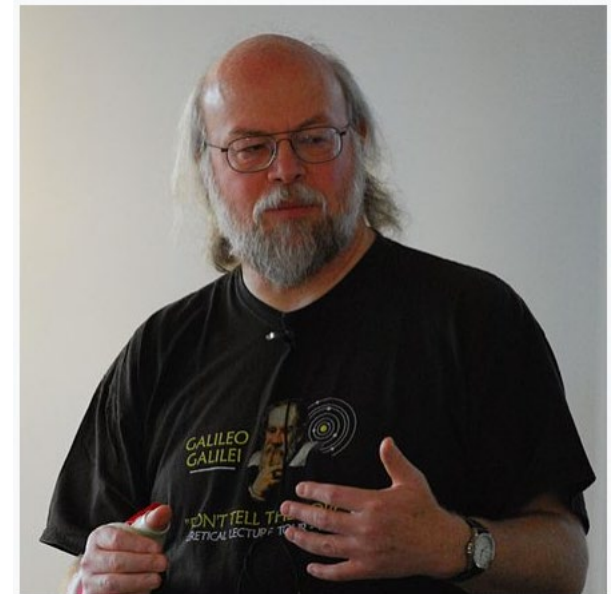 - HotJava

- Present
  - Internet / WWW
  - Enterprise
  - 1st language in industry

- Future
  - Java vs. Dynamic Language
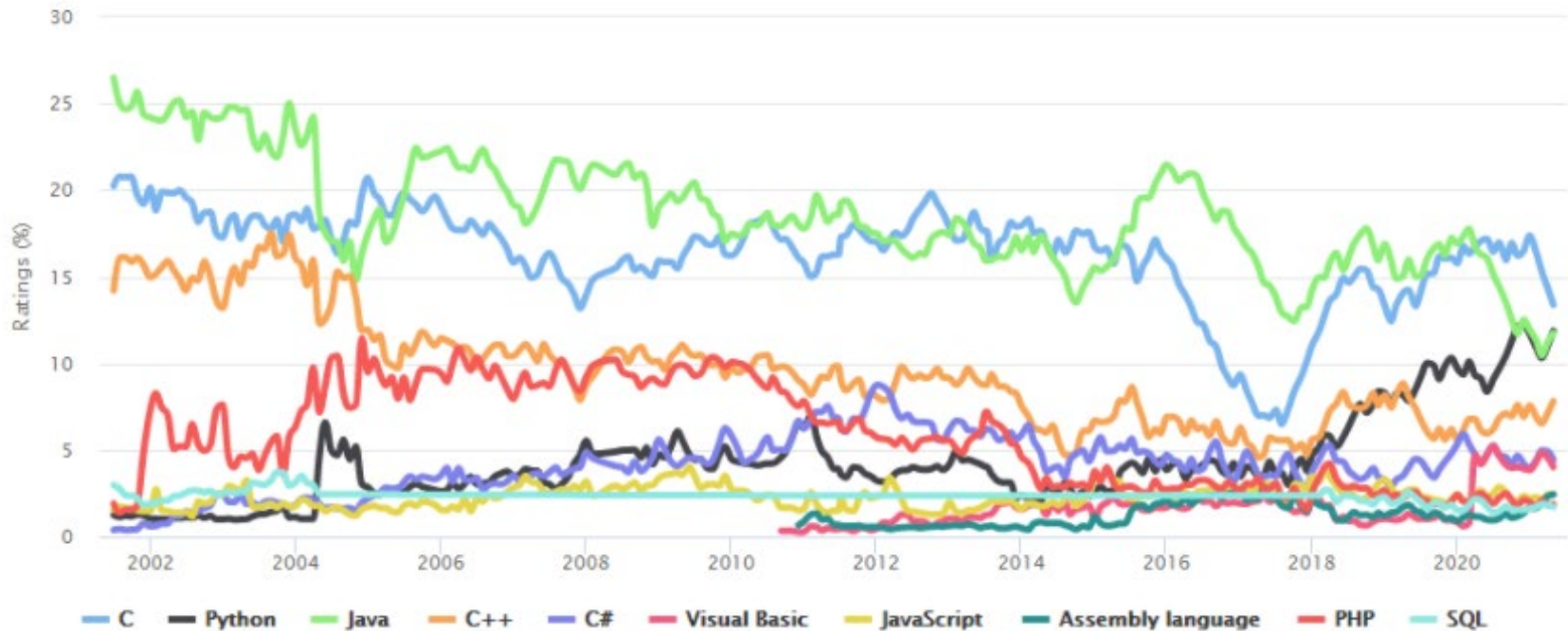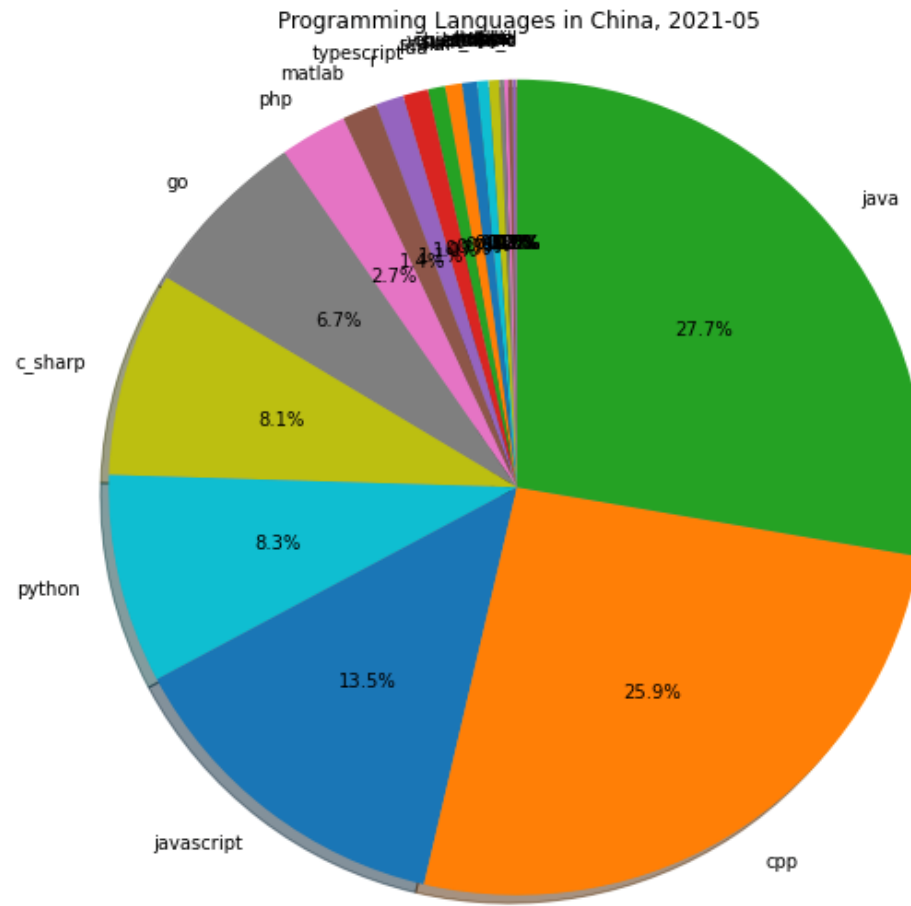 - Java and open source

James Gosling: Dr.Java

TOP 10编程语言TIOBE指数走势图 (2002-2020)

Programming Languages in China, 2021-05

Today's Java

Salary：

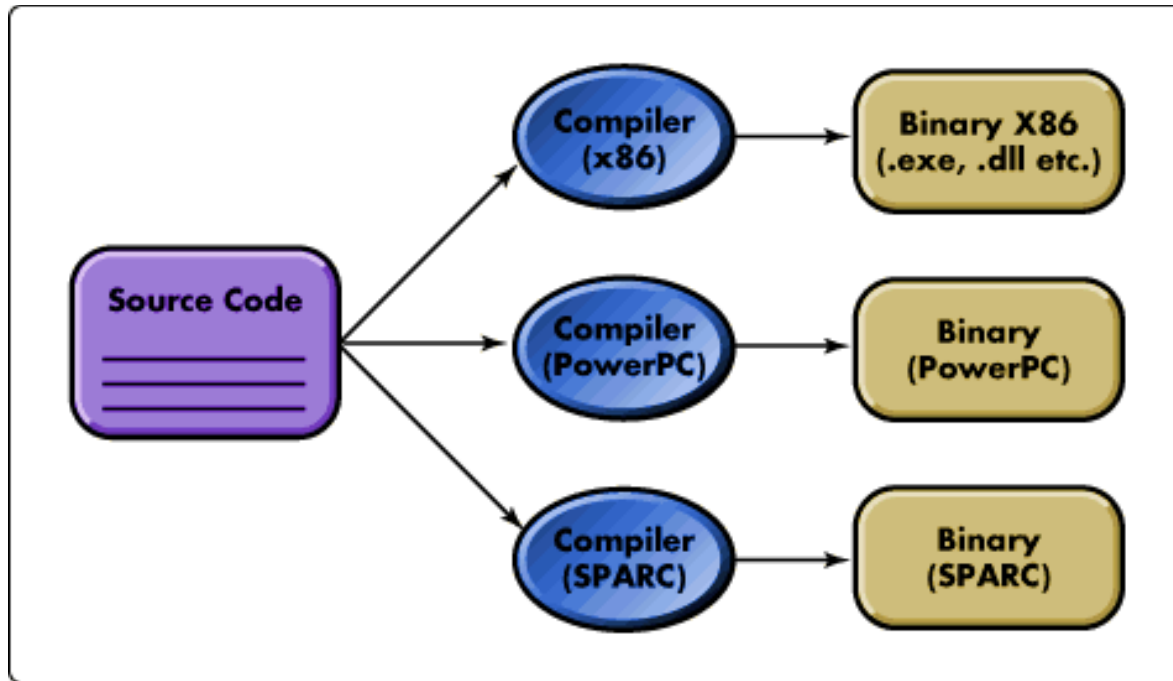| rank	pl_	salary_mean	salary_median	salary_95_min	salary_95_max	head_count	percentage
1	rust	24293	24000	7000	45000	1022	0.23%
2	scala	20041	17500	7000	45000	4158	0.93%
3	julia	19815	20000	12500	27500	54	0.01%
4	python	19329	17500	6500	45000	51652	11.58%
5	go	19246	17500	7000	45000	42051	9.43%
6	lua	19180	17500	6788	45000	4192	0.94%
7	perl	18972	17500	7000	40000	2816	0.63%
8	matlab	18905	17500	7000	37500	8629	1.93%
9	swift	18205	16000	7000	40000	3719	0.83%
10	r	17547	17500	6000	40000	6904	1.55%
11	ruby	17021	15000	5250	37500	1211	0.27%
12	kotlin	16630	15000	5750	40000	2601	0.58%
13	c/c++	16597	15000	6000	37500	161715	36.26%
14	haskell	16250	16250	15000	22500	6	0.00%
15	typescript	16210	15000	7000	36842	6239	1.40%
16	java	15256	13000	5500	37500	172709	38.72%

JDK and JRE

- JDK – Java Development Toolkit

  - J2SE – Java 2 Standard Edition

  - J2EE – Java 2 Enterprise Edition

  - J2ME – Java 2 Micro Edition
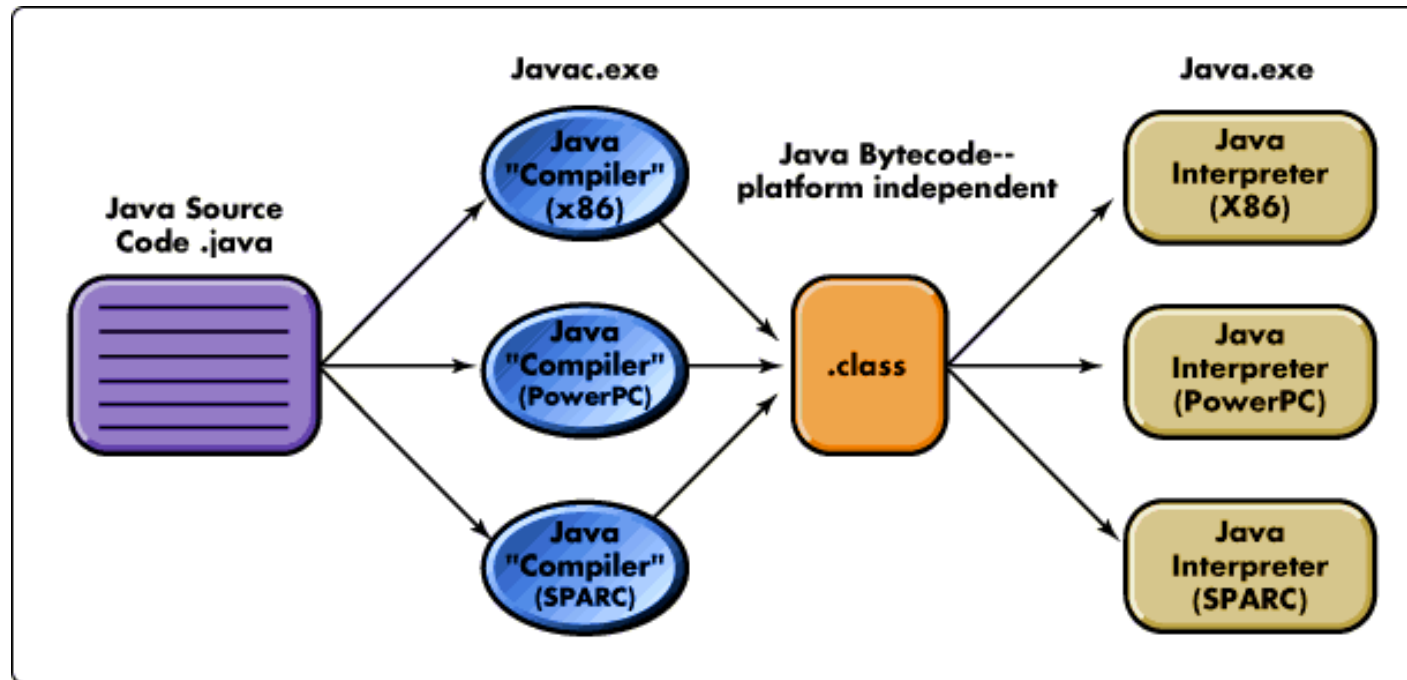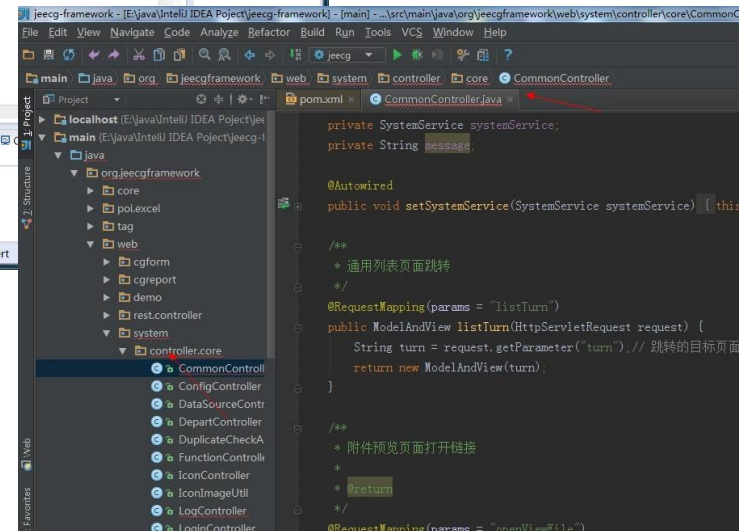
- JRE – Java Runtime Environment

Java Developing Environment

- Text editor

- IDE

  ○ Eclipse

  ○ IntelliJ IDEA

  ○ Netbeans

  ○ MyEclipse

Eclipse

A guide for Java Developers

Eclipse
IN ACTION

David Gallardo
Ed Burnette
Robert McGovern

MANNING
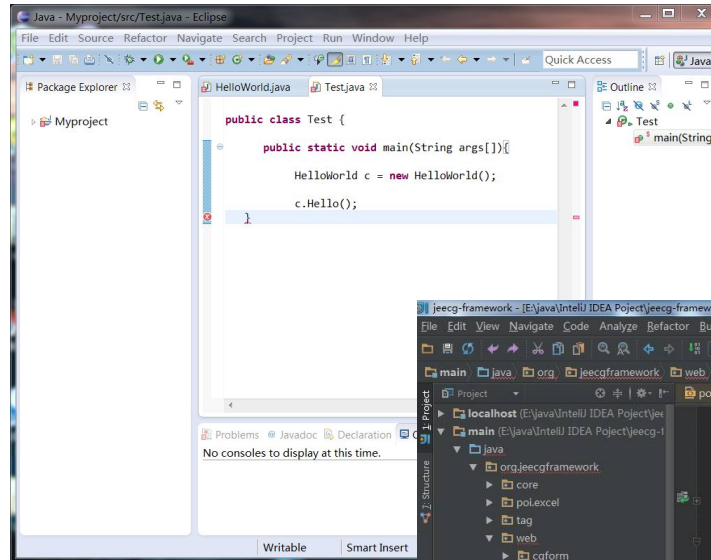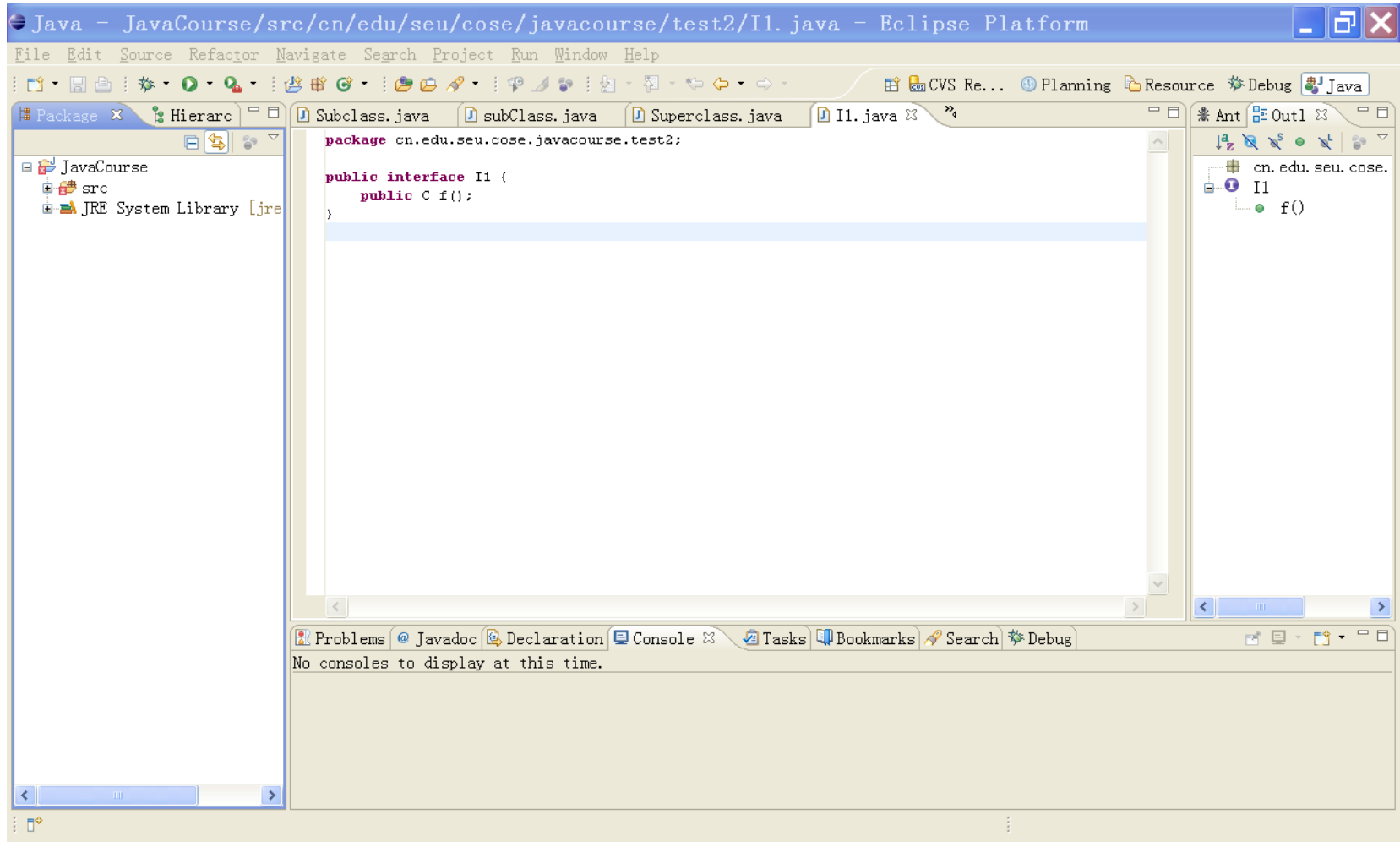
Java Features

- Simplicity: simple grammar, rich library

- Pure OO: everything is object!

- Security: memory access, garbage collection, exception

- Portability: Java Virtual Machine

- Interpreted execution: Bytecode

```java
package cn.edu.seu.cose.javacourse.ch01;
public class Person {
    private String name;
    private int age;
    public Person(String name, int age){
        this.name = name;
        this.age = age;
    }
    public void greet(){
        System.out.println("Hello, I am " + name
            + " , and I am " + age + " years old");
    }
    public static void main(String[] args){
        Person tom = new Person("Tom", 18);
        tom.greet();
    }
}
```
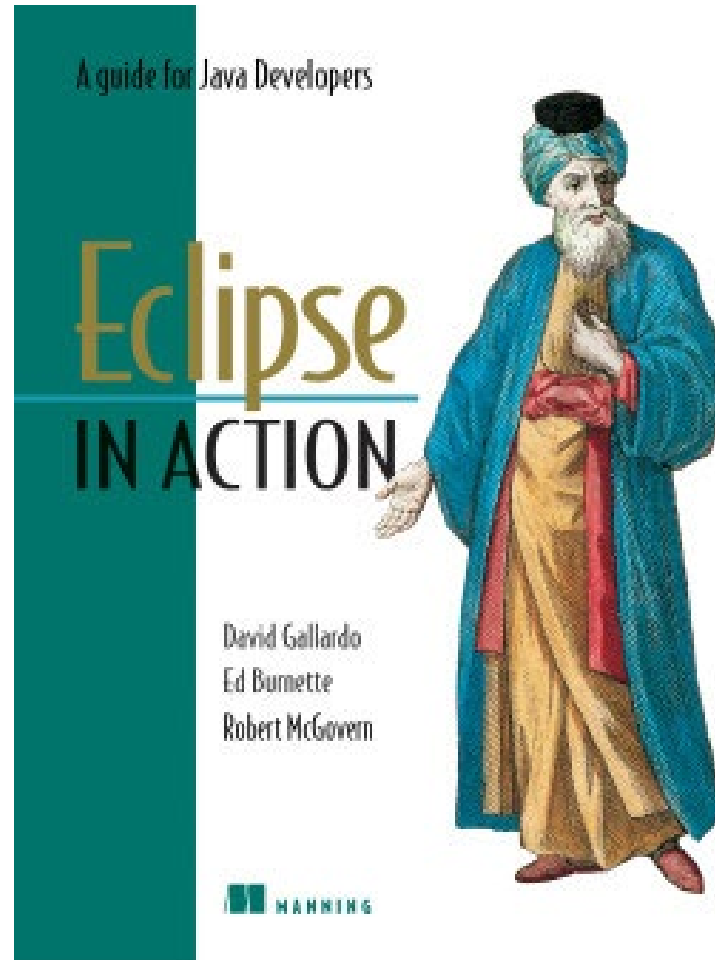
The Structure Of Java Programs

package declaration ◂------- `package cn.edu.seu.cose.javacourse.ch01;`

class declaration ◂------- `public class Person {`

variable declaration
and initialization ◂-------
```java
    private String name;
    private int age;
```

constructor ◂-------
```java
    public Person(String name, int age){
        this.name = name;
        this.age = age;
    }
```

method ◂-------
```java
    public void greet(){
        System.out.println("Hello, I am " + name
                + " , and I am " + age + " years old");
    }
```

main method ◂-------
```java
    public static void main(String[] args){
        Person tom = new Person("Tom", 18);
        tom.greet();
    }
}
```

```java
public class Person {

    privat String name;
    privat int age;

    System.out.println("the program begins.");

    public void person(int age){
        this.age = age
    }

    public int greet{
        System.out.println("Hello, I am Tom, and I am "
                    + age + " years old");
    }

    private static main(String arg){
        Person tom = new Person("18");
        tom.greet();
    }
```

How many errors?

Java Primary Data Types

19

Java Primary Data Types

Open Discussion: Why we need wrappers when we already have primary type? And why we need primary type when we have wrappers?

Type	size(bit)	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

- 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)));

  - int a = 473;

  - System.out.println(a.compareTo(472));

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;

Class:java.lang.System   method,void   method,int

```
System.out.println(a.compareTo(new Integer(472)));
```

object:PrintStream, static   object:Integer

- System.out.println()

- String Formatter

```java
double pi = 3.1415926;
String result = String.format("%.2f", pi);
System.out.println(result);
// print pi with specific digits of fractional part
```

Variables and Constants

- Declare and use

- Lifecycle and Hidden Variables

```java
int a = 10;
final int B = 20;
```

```java
public class Test {
    int t = 0;
    public void hideT(){
        int t = 10;
        int s = 9;
        System.out.println(t);
    }
    public void printT(){
        System.out.println(t);
    }
}
```

Notice!

- Different with C++

Duplicate local variable i

```java
int i = 0;
for(int j=0; j<10; j++){
    int i = 10;  // not allowed in java
}
```
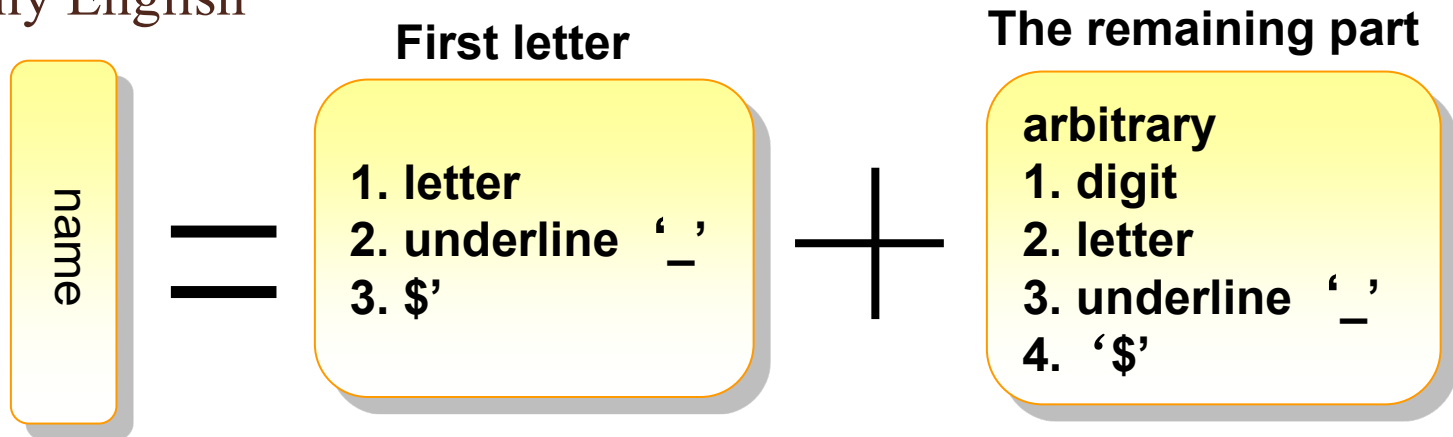
Naming

- ## Basic Principle:

  - A names should reflect the meaning of a class/package/variable…

  - Different with Java keywords

  - **Different with java.lang.* // not restricted

  - Only English

name = **First letter** + **The remaining part**

First letter
1. letter
2. underline '_'
3. $'

The remaining part
arbitrary
1. digit
2. letter
3. underline '_'
4. '$'

Naming

- project

  demo

- package

  package cn.edu…;

- class

  public class Person;

- variable

  int age = 20;

- method

  void greet(){};

- constant

  final double PI =3.14;

- Mathematical operator

- Relational operator

- Logical operator

- Bitwise operator

- Assignment operator

- Others

Mathematical Operator

- +、 -、 *、 /、 %

- ++、 --

Relational Operator

- > 、 >=

- < 、 <=

- == 、 !=

- instanceOf

```java
Person tom = new Person("Tom", 18);
System.out.println(tom instanceof Person);
```

- &、 |

- &&、 ||

- !

- ^

- <<

- >>

- >>>

Assignment Operator

- =

- += 、 -= 、 *= 、 /= 、 % =

- >>= 、 <<= 、 >>> =

- ? :

- .

- new

- []

Ternary if-else operator

```
return i < 10 ? i * 100 : i * 10;
```

Java Grammar

- Package

- Import

- Class

- Field

- Method

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

public class Person {

    private String name;
    private int age;

    public Person(String name, int age){
        this.name = name;
        this.age = age;
    }

    public void greet(){
        System.out.println("Hello, I am " + name
                + " , and I am " + age + " years old");
    }

    public static void main(String[] args){
        Person tom = new Person("Tom", 18);
        tom.greet();
    }
}
```

- Write a Java Class Student

  - id, name, gender

  - A sample greet() outputs: "Hello, I am Xiang Zhang. I am a male student, and I am from Class 3."

  - Open discussion:

    - Is is good to use a String for gender? Any better type?

    - Don't put everything in main(), why?

Java Statement

- if-else

- switch

- while、 do-while

- for

- break

- continue

- return

Java 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

```
//   This is a simple lined comment

/*   This is a multiple lined comment
 *   This is a multiple lined comment
 *   This is a multiple lined comment
*/


/**
 * @param age
 * @return
 */
    public int count(int age){
        return 0;
    }
```

```java
public class Person {

    private String name;
    private int age;

    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public void greet() {
        System.out.println("Hello, I am " +
                name + " , and I am " + age + " years old");
    }

    public static void main(String[] args) {
        Person tom = new Person("Tom", 18);
        tom.greet();
    }
}
```

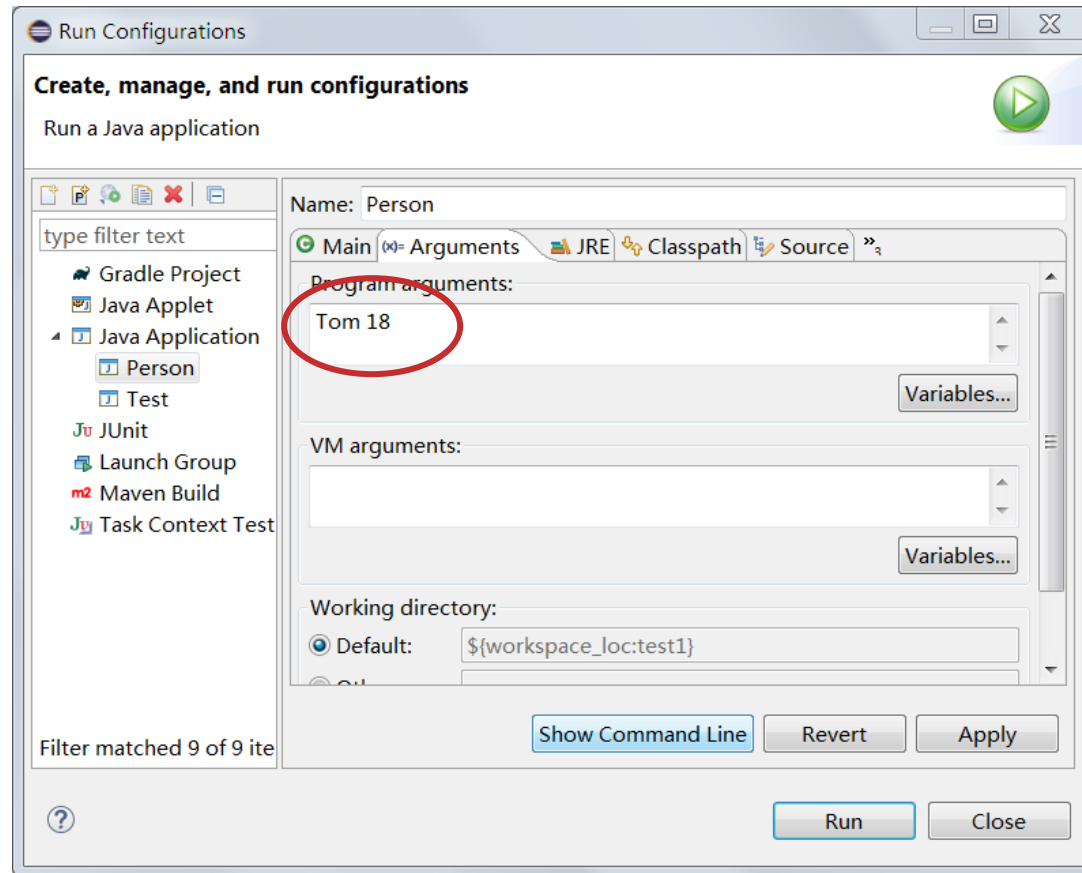bad code!
data is hard-coded, which is hard to modify

- Two ways to avoid hard-coding

  ○ String[] args

  ○ Scanner

```java
Scanner sc = new Scanner(System.in);
```

hint:

```java
package cn.edu.seu.java;

import java.util.Scanner;

public class Person {
```

- A simple version of ATM

  - Single user

  - Deposit / Withdrawal / Query Balance

  - Using Scanner to get user request and amount of money

  - An user interface like this:

```
Please select your transaction:
1: Deposit
2: Withdrawal
3: Query Balance
```

  - Try NOT to write all the codes in main()!!

- 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