Chapter 1 Java Fundamentals

Xiang Zhang

javacose@qq.com

Content

- Evolution of Java
- JDK and JRE
- Java Operating Mechanism
- Java Developing Environment
- Java Primary Data Types
- Java Basic Grammar

Evolution of Java – Success of OOP

Ada Lovelace

from《Beyond Java》

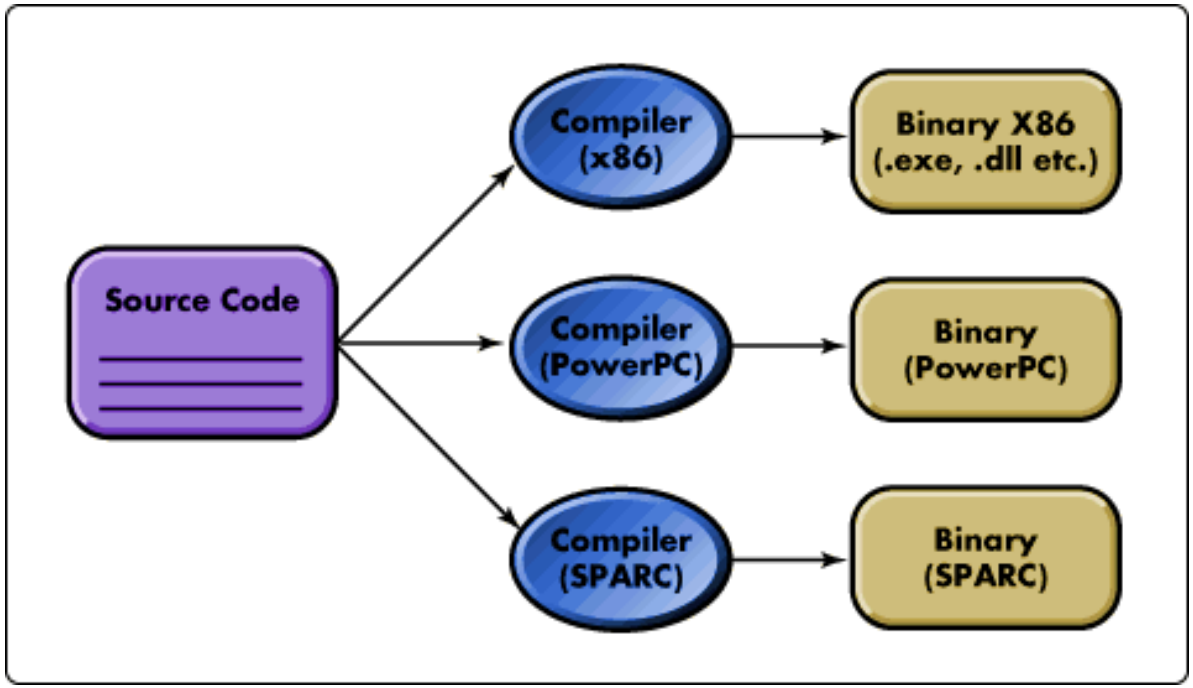
Evolution of Java – Life of Java

- 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

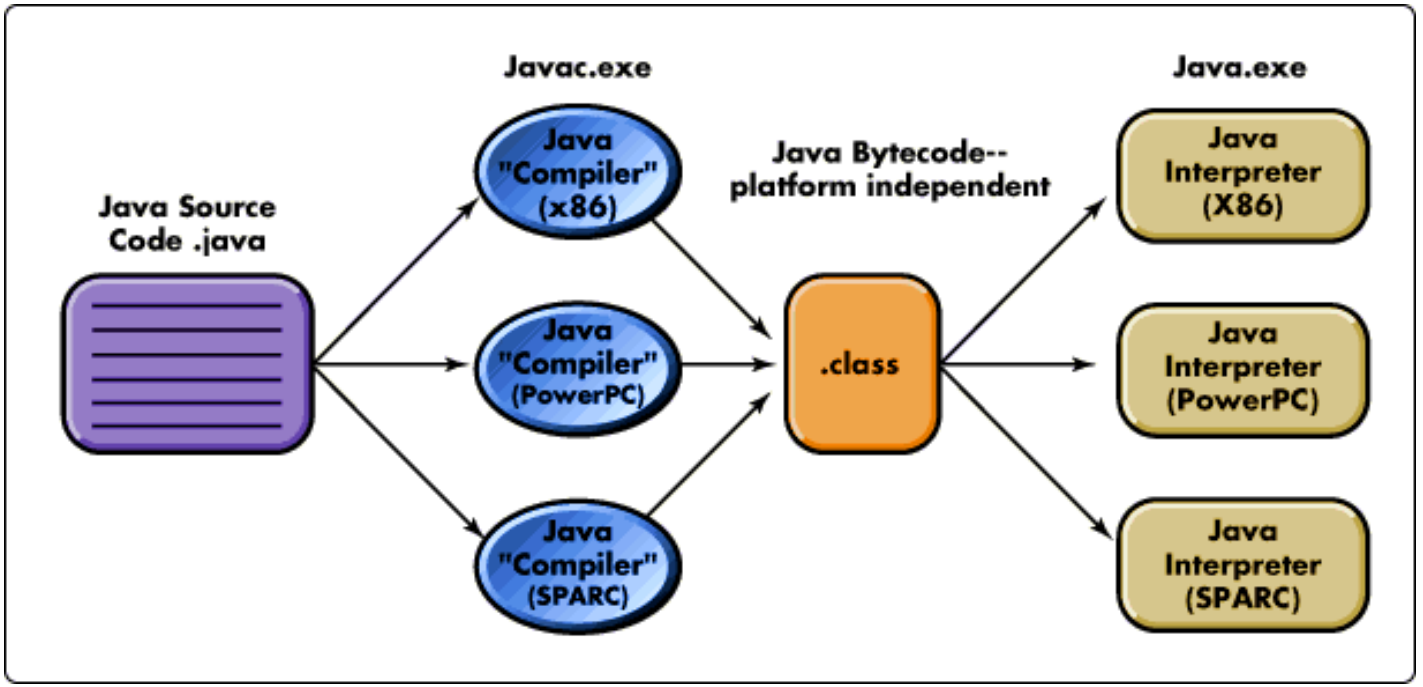
JDK and JRE

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

Java Mechanism – Traditional

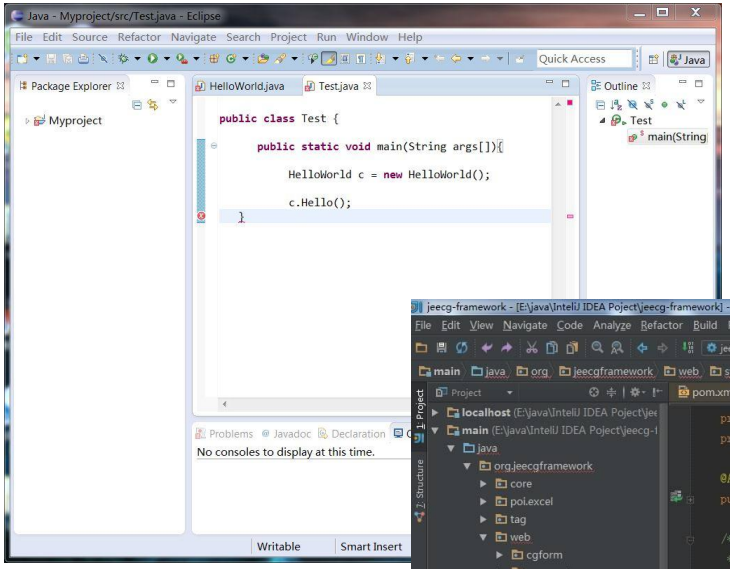


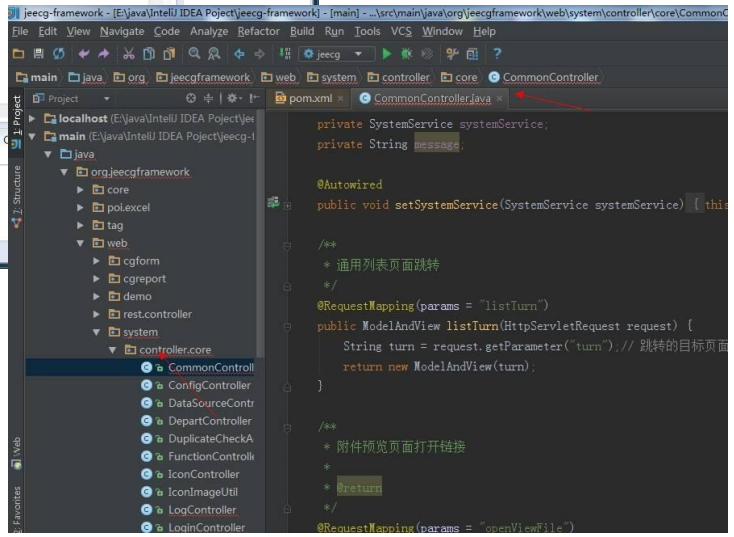
Java Mechanism – Java



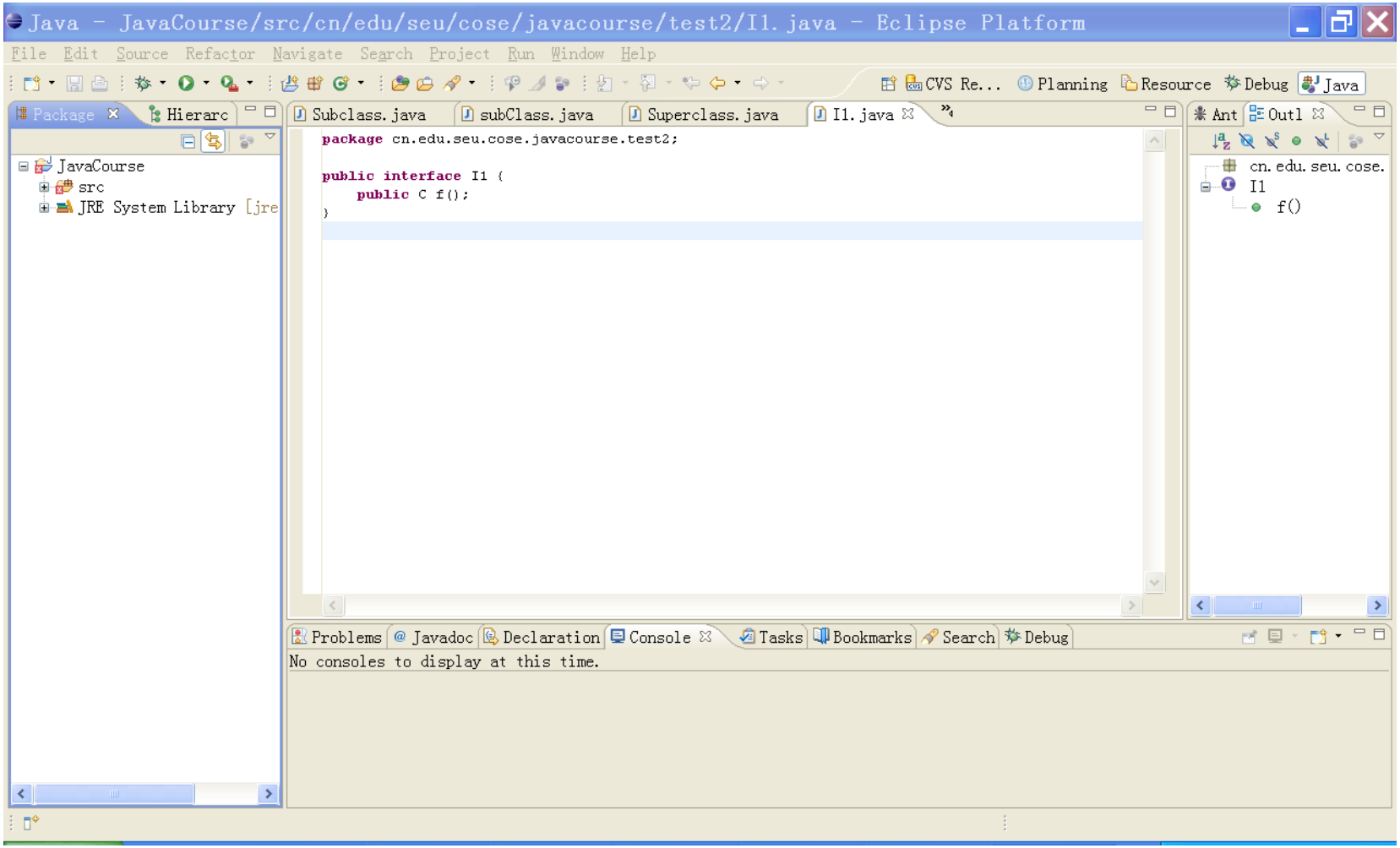
Java Developing Environment

- Text editor
- IDE
  - Eclipse
  - IntelliJ IDEA
  - Netbeans
  - MyEclipse

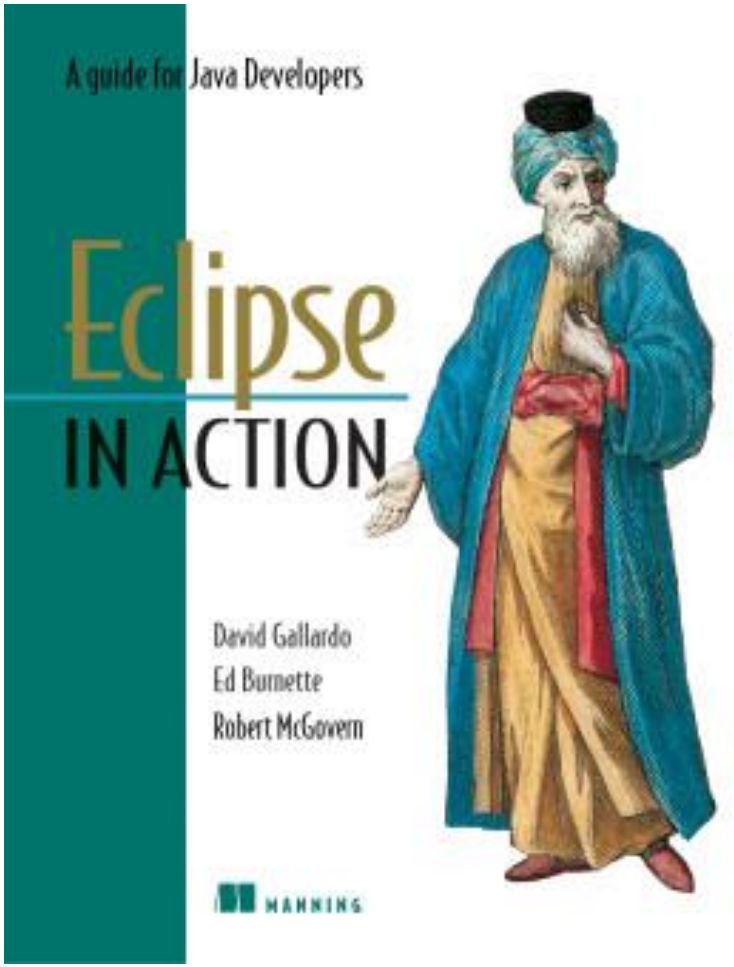




Eclipse


Java - JavaCourse/src/cn/edu/seu/cose/javacourse/test2/I1.java - Eclipse Platform
File Edit Source Refactor Navigate Search Project Run Window Help
CVS Re... Planning Resource Debug Java
Package Hierarc
JavaCourse
src
JRE System Library [jre
Subclass.java subClass.java Superclass.java I1.java
package cn.edu.seu.cose.javacourse.test2;
public interface I1 {
public C f();
}
Ant Outl
cn.edu.seu.cose.
I1
f()
Problems Javadoc Declaration Console Tasks Bookmarks Search Debug
No consoles to display at this time.

Eclipse is not only an IDE



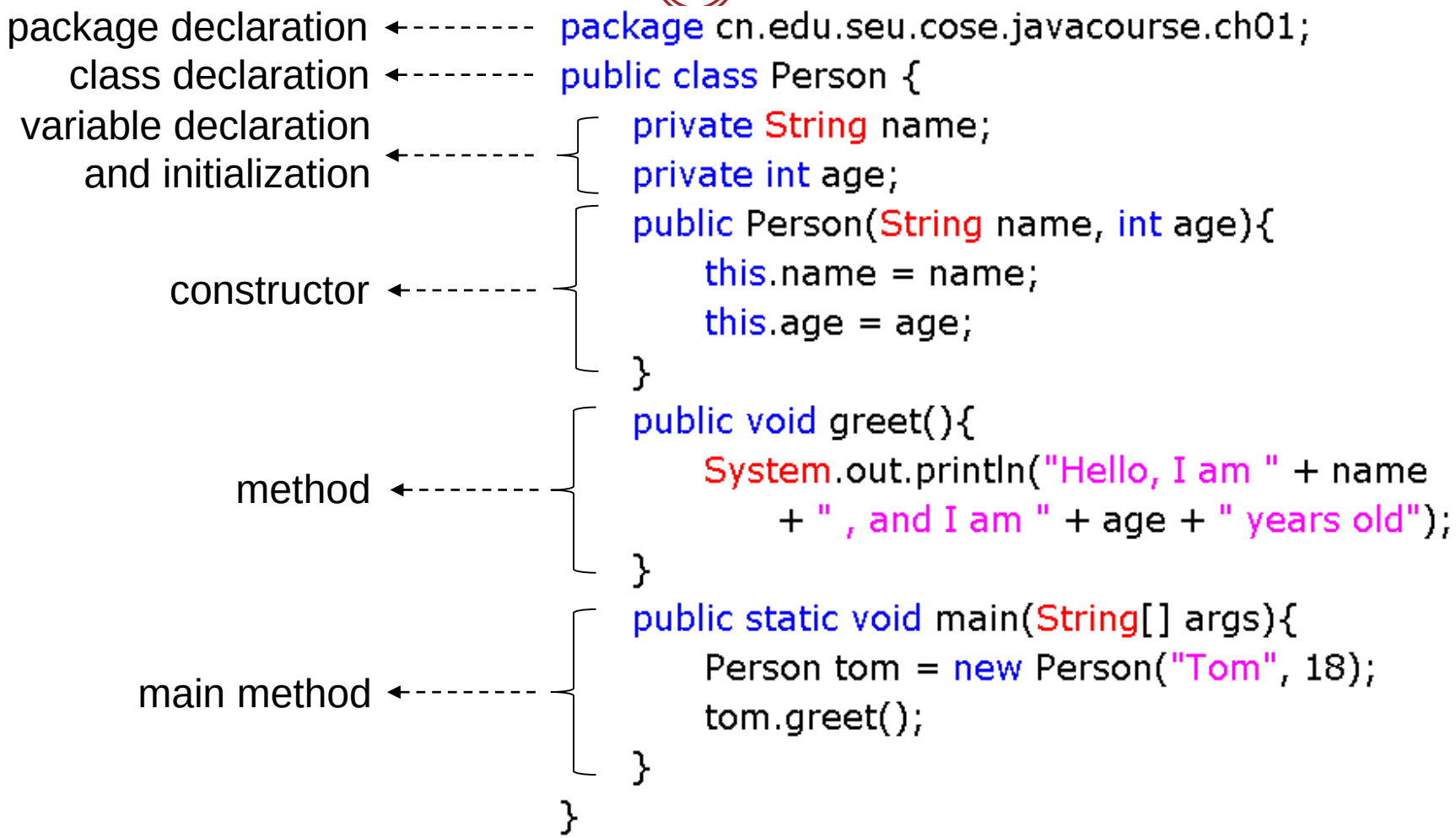
Java Features

- Simplicity: simple grammar, rich library
- Pure OO: everything is object!
- Security: memory access, garbage collection, exception
- Portability: Java Virtual Machine
- Interpreted execution: Bytecode

Exploring 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



How many errors?

```
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();
    }
```

Java Primary Data Types

Java Primary Data Types

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

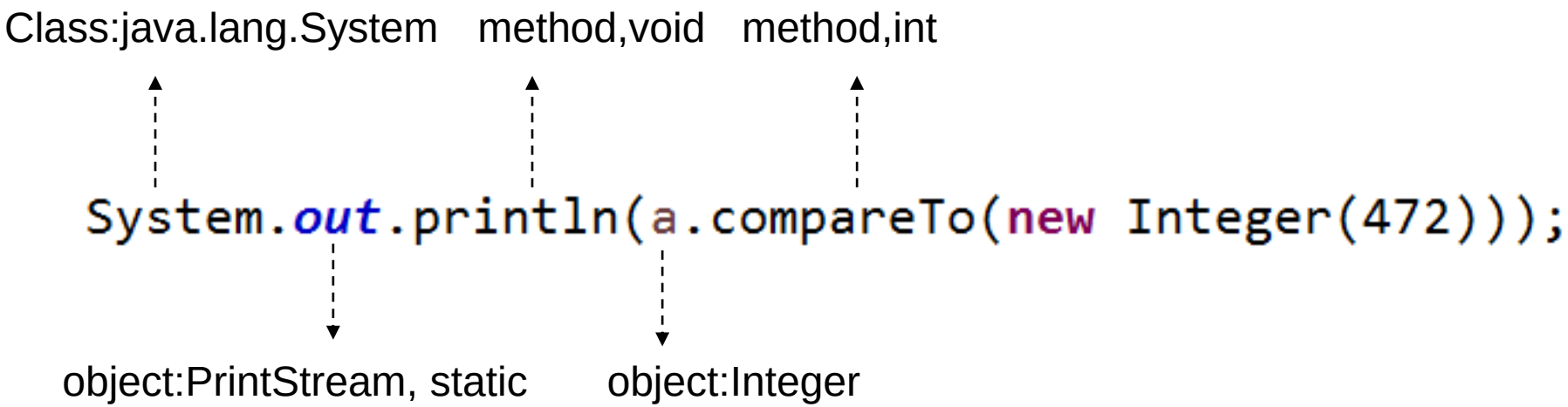
Conversion Between Values

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

Primary Types and Wrapper

- 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)));

More About This Statement



Print and Format

- System.out.println()
- String Formatter

```
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

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

- Lifecycle and Hidden Variables

```
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);
    }
}
```

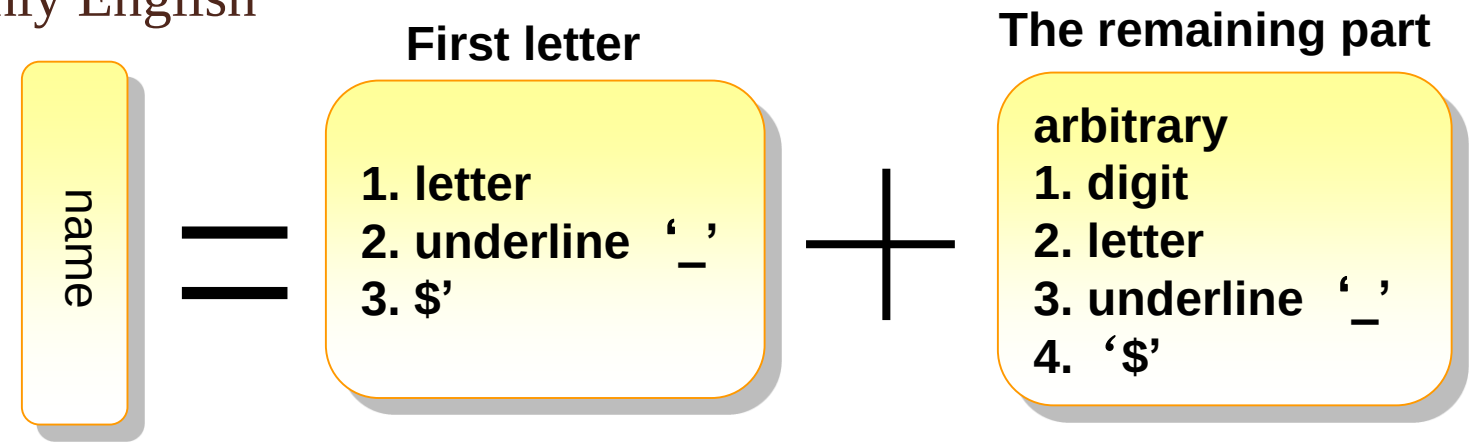
Notice!

- Different with C++

```
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



Naming

- project

  demo

- package

  package efrei.java;

- class

  public class Person;

- variable

  int age = 20;

- method

  void greet(){};

- constant

  final double PI =3.14;

Java Operator

- Mathematical operator
- Relational operator
- Logical operator
- Bitwise operator
- Assignment operator
- Others

Mathematical Operator

- +、-、*、/、%
- ++、--

Relational Operator

- > 、 >=
- < 、 <=
- == 、 !=
- instanceof

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

Logical Operator

- &、|
- &&、||
- !
- ^

Bitwise Operator

- <<
- >>
- >>>

Assignment Operator

- =
- +=、-=、*=、/=、%=
- >>=、<<=、>>>=

Others

- ? :
- .
- new
- []

Ternary if-else operator

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

Java Grammar

- Package
- Import
- Class
- Field
- Method

```
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();
    }
}
```

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
```

Java Comments


```
//   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;
    }
```

Lab Work 1

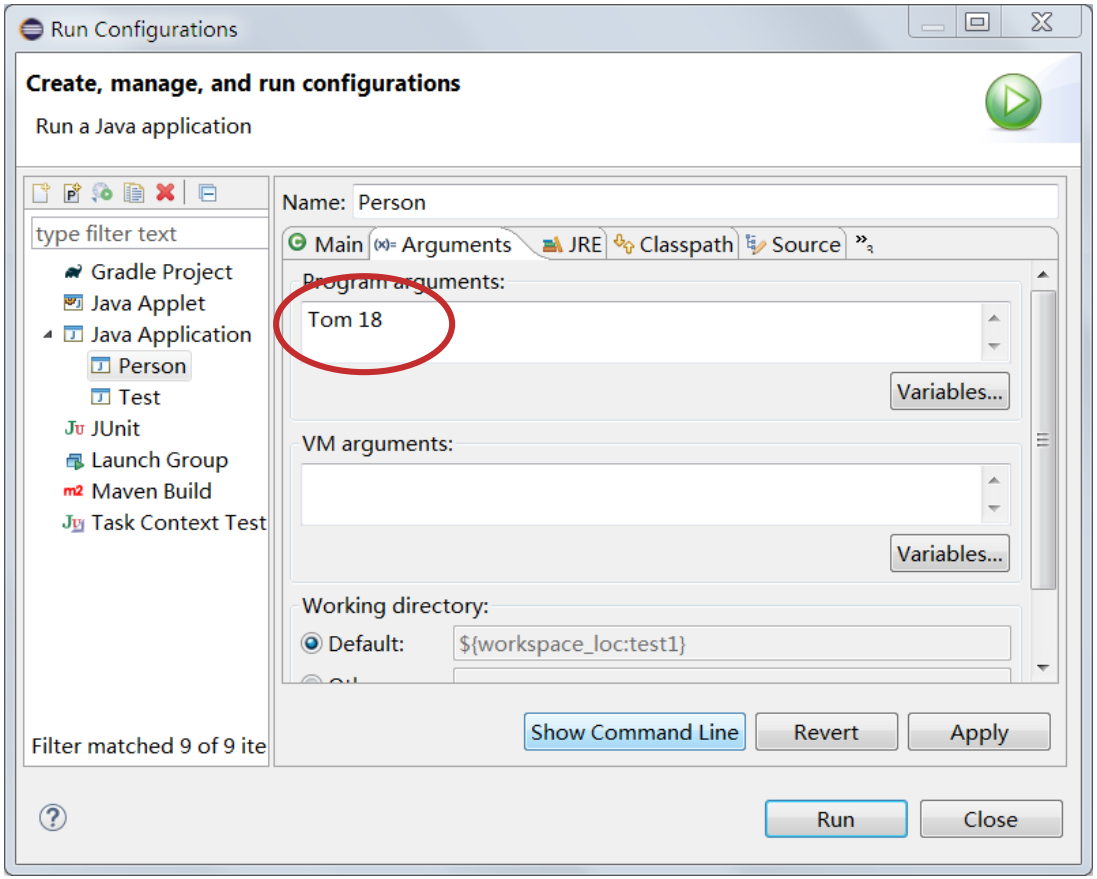

```
public class Person {

    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

Lab Work 1

- Two ways to avoid hard-coding
  - String[] args
  - Scanner

String[] args


Run Configurations
Create, manage, and run configurations
Run a Java application
type filter text
Gradle Project
Java Applet
Java Application
Person
Test
JUnit
Launch Group
Maven Build
Task Context Test
Filter matched 9 of 9 ite
Name: Person
Main
Arguments
JRE
Classpath
Source
Program arguments:
Tom 18
Variables...
VM arguments:
Variables...
Working directory:
Default:
${workspace_loc:test1}
Show Command Line
Revert
Apply
Run
Close

Scanner


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

hint:

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

import java.util.Scanner;

public class Person {
```

Lab Work 2

- 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()!!

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