JAVA 程式設計入門

Wen-Pinn Fang

Outline

- introduction to Java and VM
- variable, class and object oriented programming
- input and output
- condition and branch
- loop
- function
- exception
- multithread
- data structure
- file and text parser
- reflection
- case study

INTRODUCTION TO JAVA

Introduction to Java

Introduction to Java

- James Gosling辦公室的窗外，正好有一棵橡樹(Oak)
- Oak具備安全性、網路通訊、物件導向、Garbage Collected、多執行緒等等。
- Oak主要的目的是撰寫在star 7上的應用程式。

Introduction to Java

Star 7不被當時的消費性市場所接受。

當小組快要被SUN裁撤時，全世界第一個全球資訊網瀏覽器--Mosaic誕生了。

Introduction to Java

Java就以它優異的功能，在全球資訊網的平台上撰寫高互動性的網頁程式，我們稱為Applet。

因為那時沒有其它的程式語言能夠做到，所以原本坐以待斃的Java，又在全球資訊網上開啟了另一片天空。在1995年5月23 號，JDK(Java Development Kits)1.0a2版本正式對外發表。

Introduction to Java

當Oak要去註冊商標時，發現已經有另外一家公司已經先用了Oak這個名字。

OOak這個名字不能用，工程師們邊喝著咖啡討論著，看看手上的咖啡，突然靈機一動，就叫Java好了。

Introduction to Java

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Multithreaded
- Robust
- Dynamic
- Secure

怎麼樣可以讓程式執行

作業系統是什麼?

作業系統可以執行編譯好的程式

作業系統會和硬體相關

寫一個可以讓作業系統執行的程式

這個程式可以執行另一個寫好的程式

這就是VM

但是為了讓所有廠商可以做這個程式

內容無法加密而且執行速度比較慢

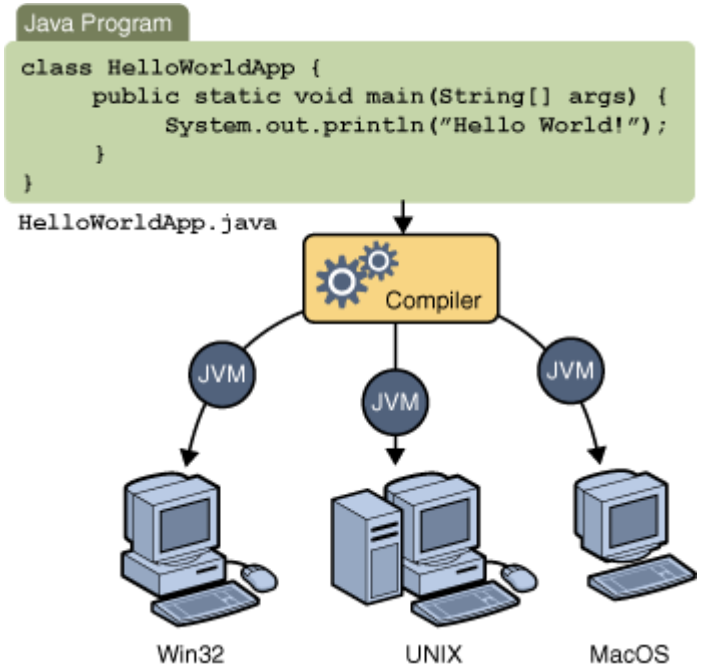
HTTP://WWW.JAVADECOMPILERS.COM/

INTRODUCTION TO VM

Introduction to VM

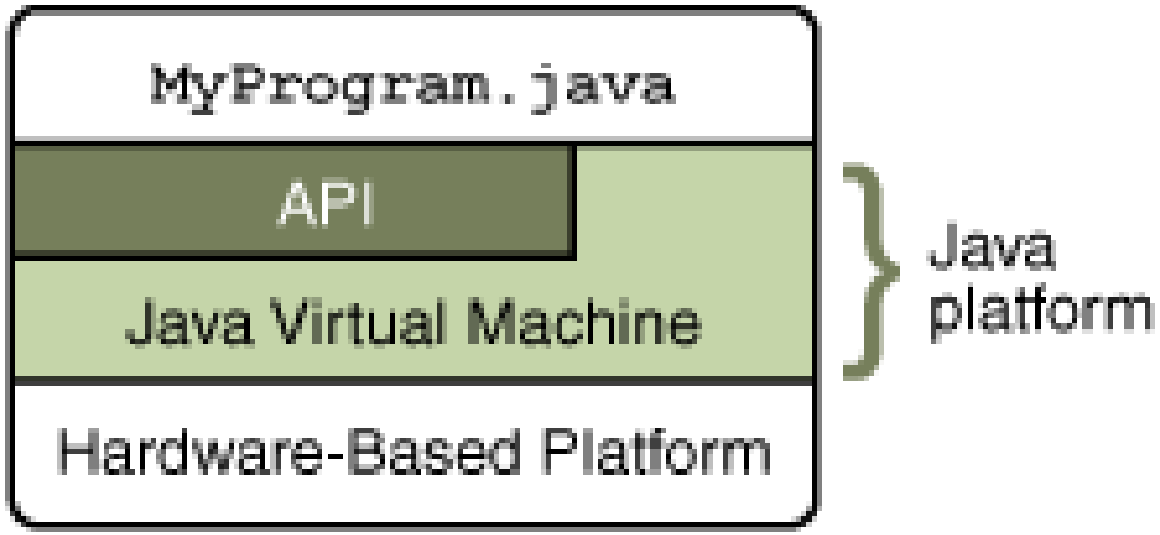
An overview of the software development process.

Introduction to VM



Through the Java VM, the same application is capable of running on multiple platforms.

Introduction to VM



The API and Java Virtual Machine insulate the program from the underlying hardware.

How to comiple

The simplest example

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

什麼是變數?

放資料的地方

來討論以下式子

- A=B

- A存的內容和B一樣
- A就是B

注意

- Java 認大小寫
- Java 類別名稱和檔案名稱要一樣
- Java package和目錄名稱要一樣

Compile and execute

```
C:\wpfang\teach\java\demo>dir
 磁碟區 C 中的磁碟沒有標籤。
 磁碟區序號:  1610-99B4

 C:\wpfang\teach\java\demo 的目錄

2017/04/30  上午 09:40    <DIR>          .
2017/04/30  上午 09:40    <DIR>          ..
2017/04/30  上午 09:40               143 simplest.java
               1 個檔案            143 位元組
               2 個目錄  258,470,457,344 位元組可用

C:\wpfang\teach\java\demo>javac simplest.java

C:\wpfang\teach\java\demo>java simplest
the simplest example

C:\wpfang\teach\java\demo>
```

javadoc

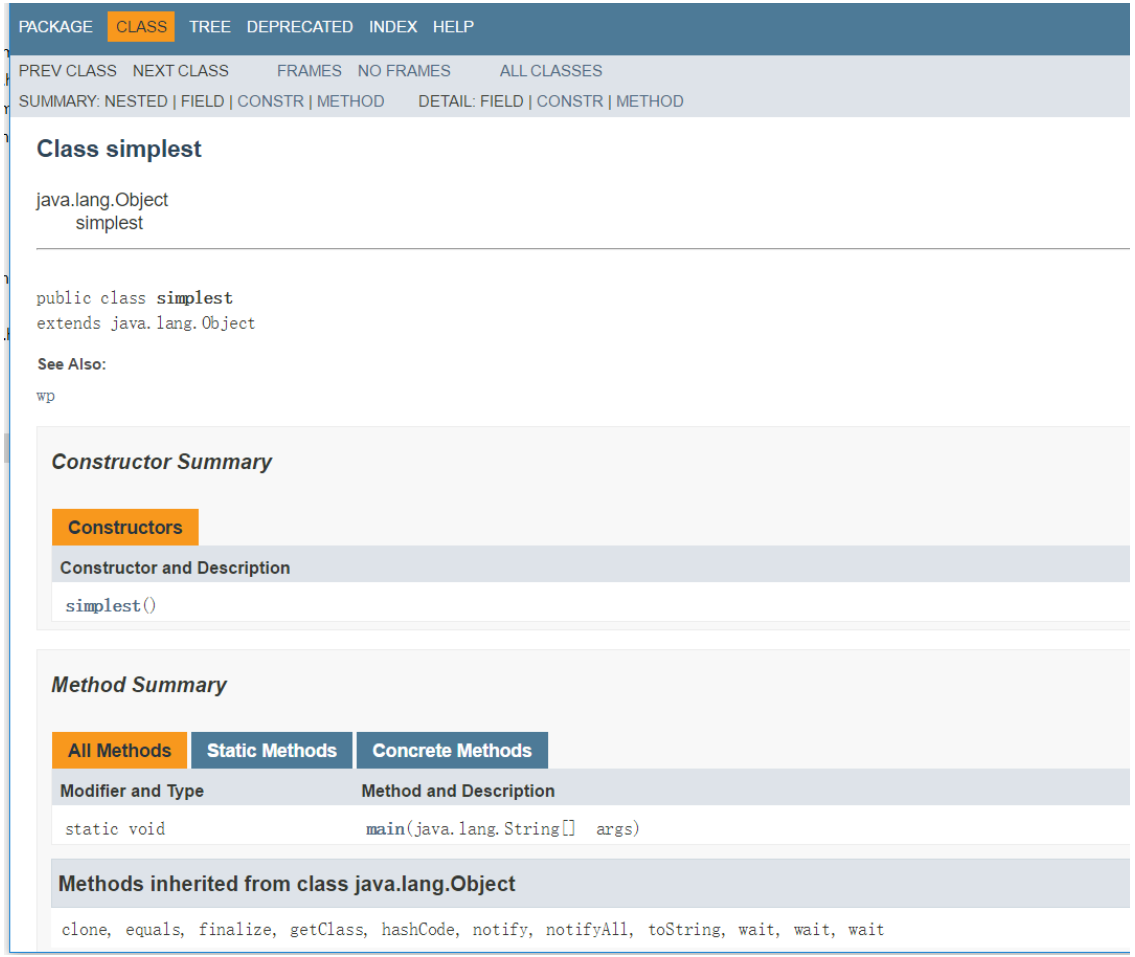
```
/**
 * @author  wpfang
 * @version %I%, %G%
 * @see     <a href="140.138.146.85">wp</a>
 */
public class simplest
{
   public static void main(String [] args)
   {
        System.out.println("the simplest example");
   }
}
```

javadoc

```
C:\wpfang\teach\java\demo>javadoc simplest.java
Loading source file simplest.java...
Constructing Javadoc information...
Standard Doclet version 1.8.0_111
Building tree for all the packages and classes...
Generating .\simplest.html...
Generating .\package-frame.html...
Generating .\package-summary.html...
Generating .\package-tree.html...
Generating .\constant-values.html...
Building index for all the packages and classes...
Generating .\overview-tree.html...
Generating .\index-all.html...
Generating .\deprecated-list.html...
Building index for all classes...
Generating .\allclasses-frame.html...
Generating .\allclasses-noframe.html...
Generating .\index.html...
Generating .\help-doc.html...

C:\wpfang\teach\java\demo>
```

javadoc



PACKAGE CLASS TREE DEPRECATED INDEX HELP

PREV CLASS NEXT CLASS FRAMES NO FRAMES ALL CLASSES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Class simplest

java.lang.Object
    simplest

public class **simplest**
extends java.lang.Object

See Also:

wp

Constructor Summary

Constructors

Constructor and Description
simplest()

Method Summary

All Methods **Static Methods** **Concrete Methods**

Modifier and Type	Method and Description
static void	main(java.lang.String[] args)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

JAVAC

- Usage: javac <options> <source files>
- where possible options include:
  - -g Generate all debugging info
  - -g:none Generate no debugging info
  - -g:{lines,vars,source} Generate only some debugging info
  - -nowarn Generate no warnings
  - -verbose Output messages about what the compiler is doing
  - -deprecation Output source locations where deprecated APIs are used
  - -classpath <path> Specify where to find user class files and annotation processors
  - -cp <path> Specify where to find user class files and annotation processors
  - -sourcepath <path> Specify where to find input source files

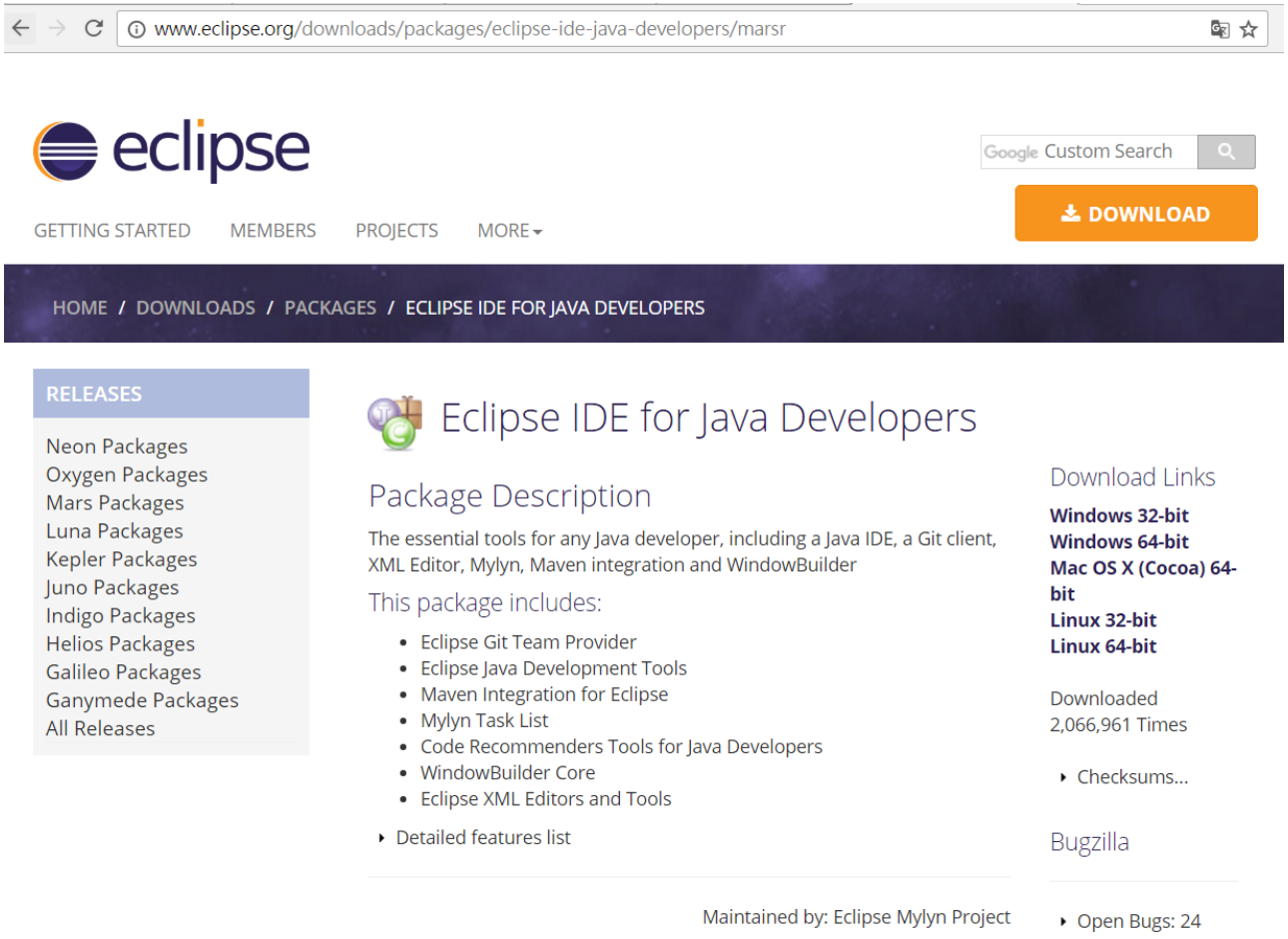
JAVAC

- Usage: javac <options> <source files>
- where possible options include:
  - -bootclasspath <path> Override location of bootstrap class files
  - -extdirs <dirs> Override location of installed extensions
  - -endorseddirs <dirs> Override location of endorsed standards path
  - -proc:{none,only} Control whether annotation processing and/or compilation is done.
  - -processor <class1>[,<class2>,<class3>...] Names of the annotation processors to run; bypasses default discovery process
  - -processorpath <path> Specify where to find annotation processors
  - -parameters Generate metadata for reflection on method parameters
  - -d <directory> Specify where to place generated class files
  - -s <directory> Specify where to place generated source files
  - -h <directory> Specify where to place generated native header files

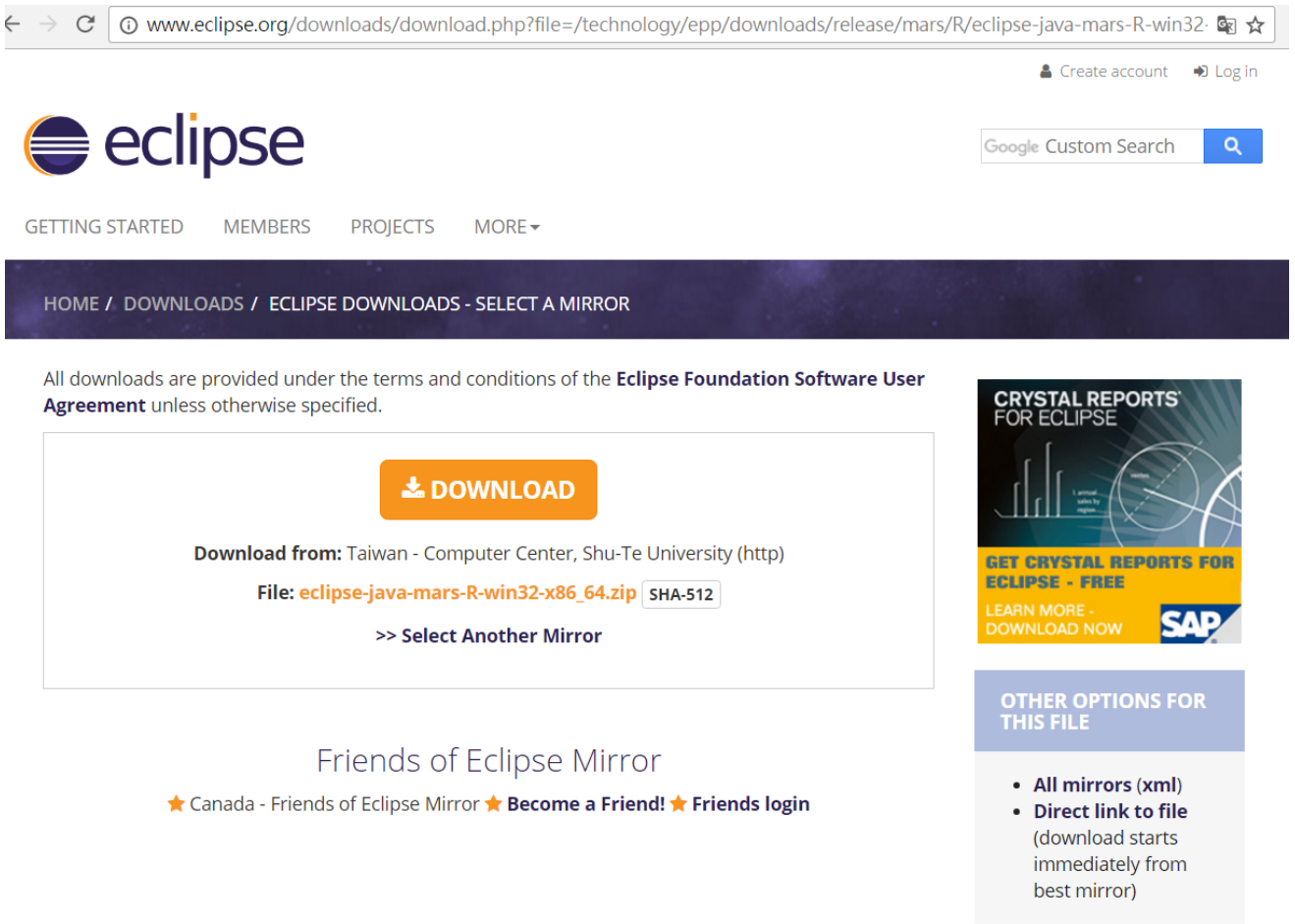
JAVAC

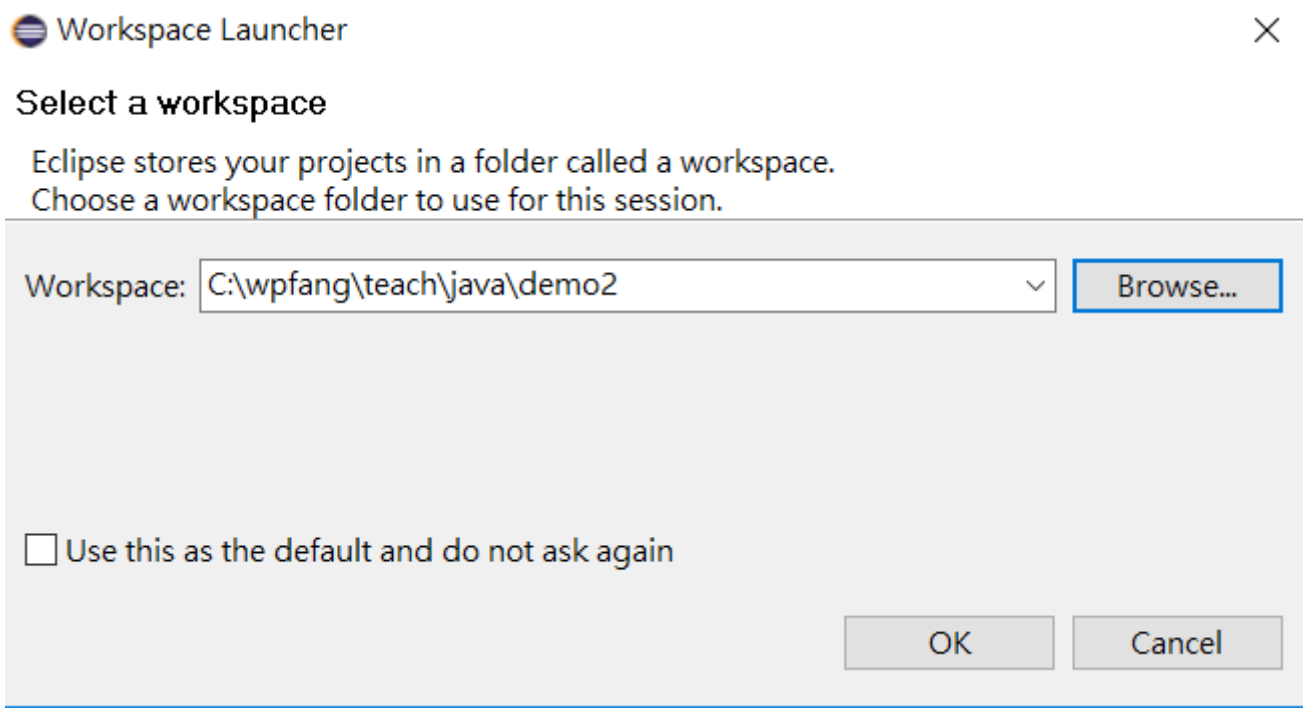
- Usage: javac <options> <source files>
- where possible options include:
  - -implicit:{none,class} Specify whether or not to generate class files for implicitly referenced files
  - -encoding <encoding> Specify character encoding used by source files
  - -source <release> Provide source compatibility with specified release
  - -target <release> Generate class files for specific VM version
  - -profile <profile> Check that API used is available in the specified profile
  - -version Version information
  - -help Print a synopsis of standard options
  - -Akey[=value] Options to pass to annotation processors
  - -X Print a synopsis of nonstandard options
  - -J<flag> Pass <flag> directly to the runtime system
  - -Werror Terminate compilation if warnings occur
  - @<filename> Read options and filenames from file

Eclipse


www.eclipse.org/downloads/packages/eclipse-ide-java-developers/marsr
eclipse
Google Custom Search
GETTING STARTED
MEMBERS
PROJECTS
MORE
DOWNLOAD
HOME / DOWNLOADS / PACKAGES / ECLIPSE IDE FOR JAVA DEVELOPERS
RELEASES
Neon Packages
Oxygen Packages
Mars Packages
Luna Packages
Kepler Packages
Juno Packages
Indigo Packages
Helios Packages
Galileo Packages
Ganymede Packages
All Releases
Eclipse IDE for Java Developers
Package Description
The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven integration and WindowBuilder
This package includes:
Eclipse Git Team Provider
Eclipse Java Development Tools
Maven Integration for Eclipse
Mylyn Task List
Code Recommenders Tools for Java Developers
WindowBuilder Core
Eclipse XML Editors and Tools
Detailed features list
Maintained by: Eclipse Mylyn Project
Download Links
Windows 32-bit
Windows 64-bit
Mac OS X (Cocoa) 64-bit
Linux 32-bit
Linux 64-bit
Downloaded 2,066,961 Times
Checksums...
Bugzilla
Open Bugs: 24

Eclipse

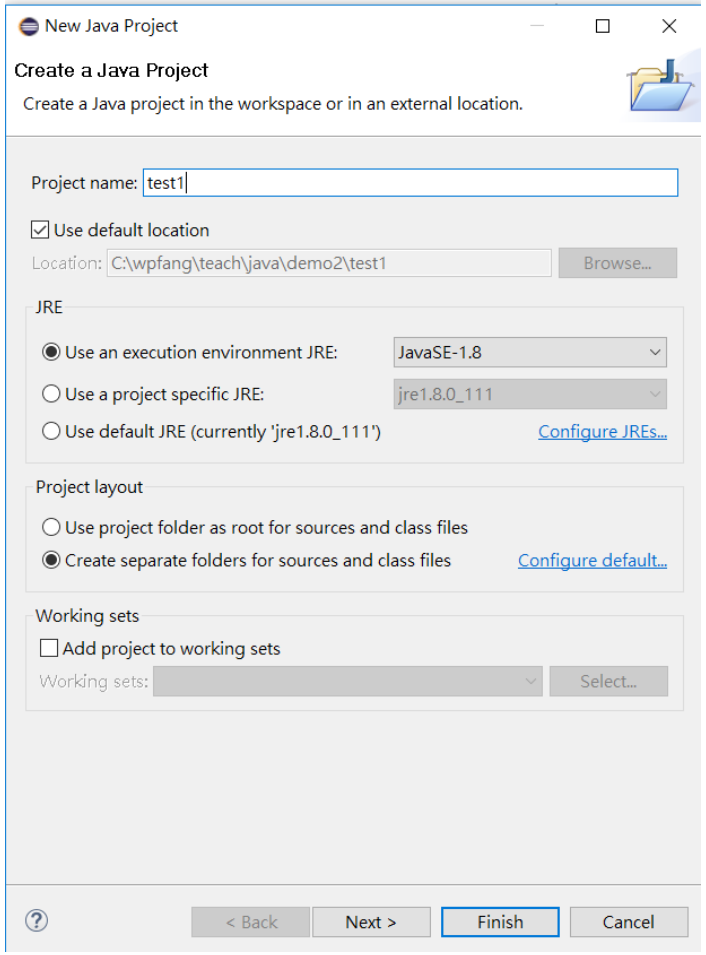

www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/mars/R/eclipse-java-mars-R-win32-
Create account
Log in
eclipse
Google Custom Search
GETTING STARTED
MEMBERS
PROJECTS
MORE
HOME / DOWNLOADS / ECLIPSE DOWNLOADS - SELECT A MIRROR
All downloads are provided under the terms and conditions of the Eclipse Foundation Software User Agreement unless otherwise specified.
DOWNLOAD
Download from: Taiwan - Computer Center, Shu-Te University (http)
File: eclipse-java-mars-R-win32-x86_64.zip SHA-512
>> Select Another Mirror
Friends of Eclipse Mirror
Canada - Friends of Eclipse Mirror Become a Friend! Friends login
CRYSTAL REPORTS FOR ECLIPSE
GET CRYSTAL REPORTS FOR ECLIPSE - FREE
LEARN MORE - DOWNLOAD NOW
SAP
OTHER OPTIONS FOR THIS FILE
All mirrors (xml)
Direct link to file (download starts immediately from best mirror)


Workspace Launcher
Select a workspace
Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.
Workspace:
C:\wpfang\teach\java\demo2
Browse...
Use this as the default and do not ask again
OK
Cancel

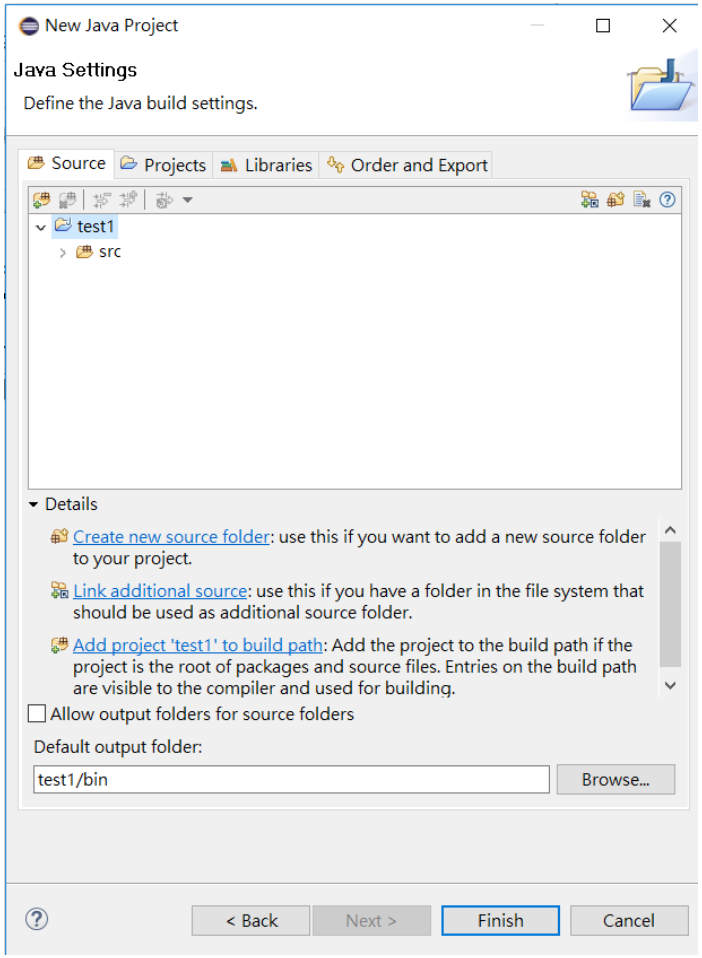
Eclipse

Eclipse

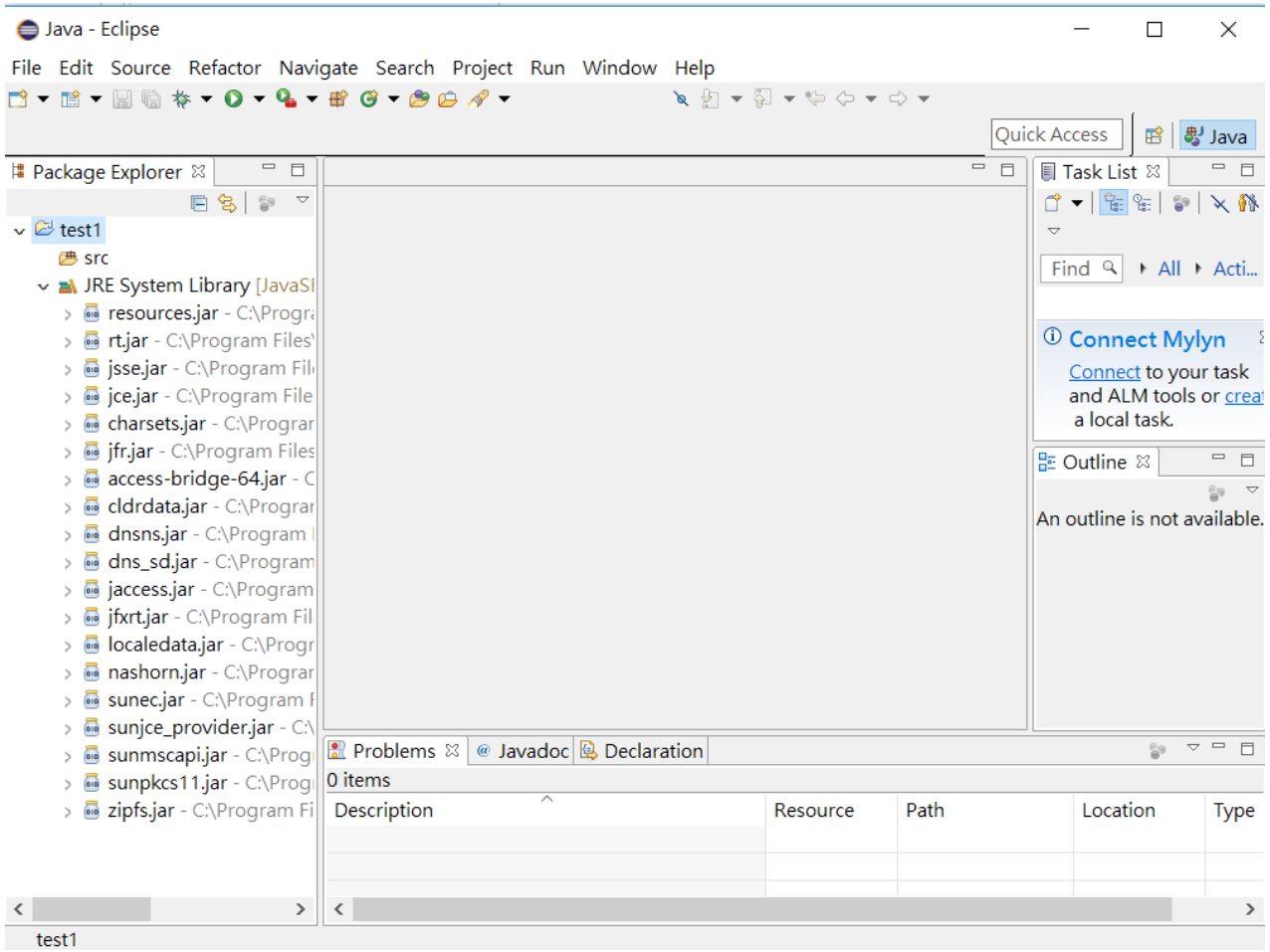

Java - Eclipse
File Edit Navigate Search Project Run Window Help
Welcome
Workbench
Welcome to the Eclipse IDE for Java Developers
Overview
Get an overview of the features
Tutorials
Go through tutorials
Samples
Try out the samples
What's New
Find out what is new
Setup check: (1%)

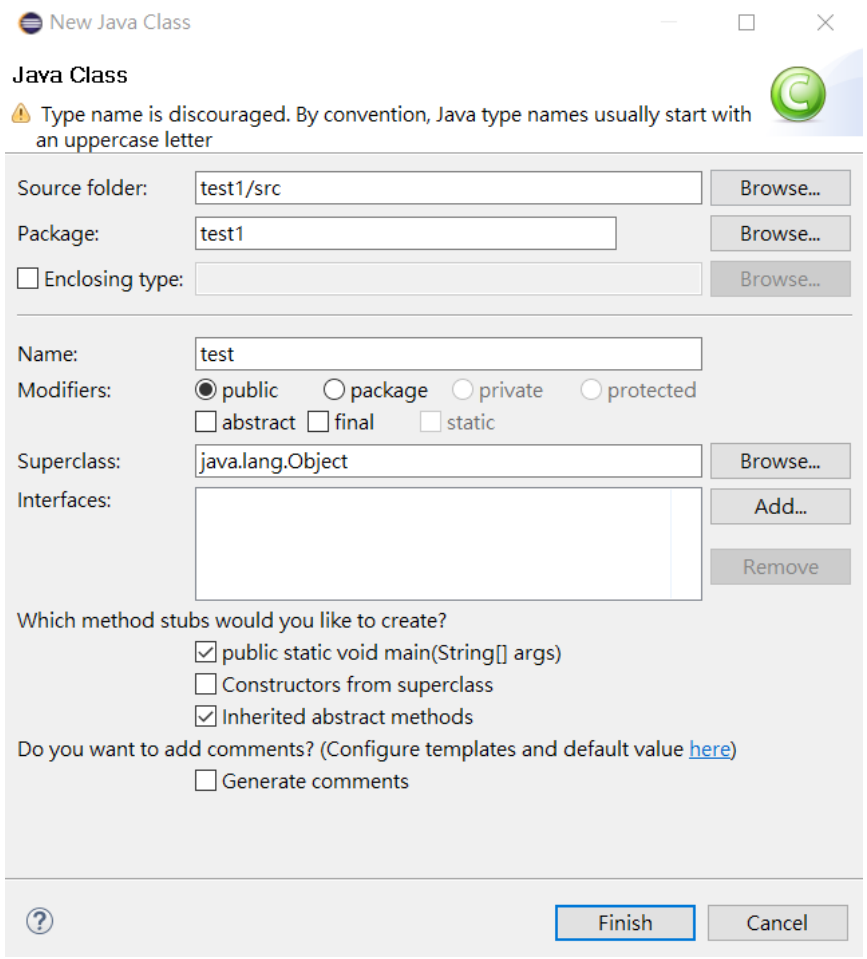

New Java Project
Create a Java Project
Create a Java project in the workspace or in an external location.
Project name: test1
Use default location
Location: C:\wpfang\teach\java\demo2\test1
Browse...
JRE
Use an execution environment JRE:
JavaSE-1.8
Use a project specific JRE:
jre1.8.0_111
Use default JRE (currently 'jre1.8.0_111')
Configure JREs...
Project layout
Use project folder as root for sources and class files
Create separate folders for sources and class files
Configure default...
Working sets
Add project to working sets
Working sets:
Select...
< Back
Next >
Finish
Cancel

Eclipse


New Java Project
Java Settings
Define the Java build settings.
Source
Projects
Libraries
Order and Export
test1
src
Details
Create new source folder: use this if you want to add a new source folder to your project.
Link additional source: use this if you have a folder in the file system that should be used as additional source folder.
Add project 'test1' to build path: Add the project to the build path if the project is the root of packages and source files. Entries on the build path are visible to the compiler and used for building.
Allow output folders for source folders
Default output folder:
test1/bin
Browse...
< Back
Next >
Finish
Cancel

Eclipse

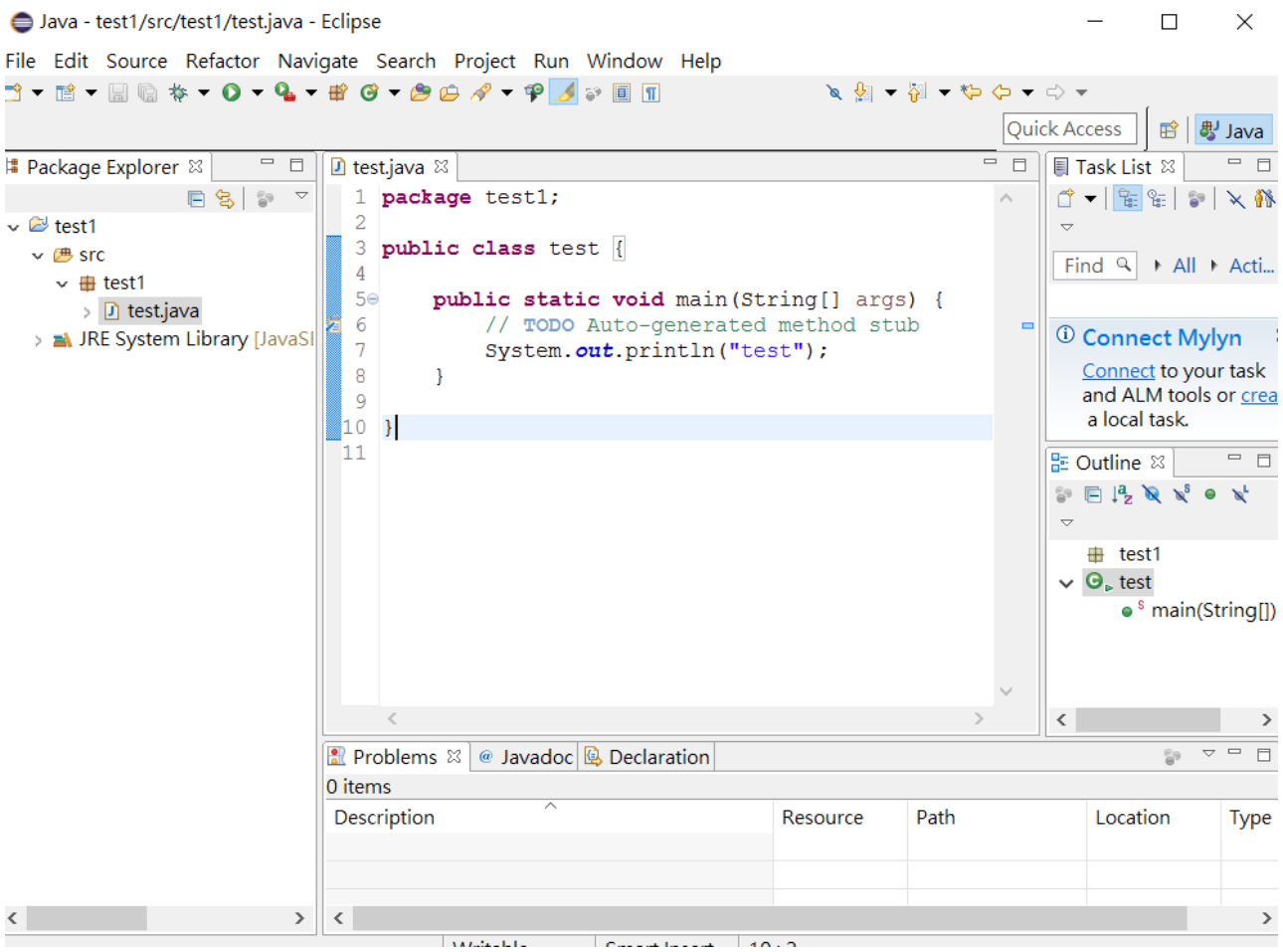

Java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Java
Package Explorer
test1
src
JRE System Library [JavaSE
resources.jar - C:\Progra
rt.jar - C:\Program Files\
jsse.jar - C:\Program Fil
jce.jar - C:\Program File
charsets.jar - C:\Prograr
jfr.jar - C:\Program Files
access-bridge-64.jar - C
cldrdata.jar - C:\Prograr
dnsns.jar - C:\Program
dns_sd.jar - C:\Program
jaccess.jar - C:\Program
jfxrt.jar - C:\Program Fil
localedata.jar - C:\Progr
nashorn.jar - C:\Prograr
sunec.jar - C:\Program
sunjce_provider.jar - C:\
sunmscapi.jar - C:\Progr
sunpkcs11.jar - C:\Progr
zipfs.jar - C:\Program Fi
Task List
Find
All
Acti...
Connect Mylyn
Connect to your task
and ALM tools or creat
a local task.
Outline
An outline is not available.
Problems
Javadoc
Declaration
0 items
Description
Resource
Path
Location
Type
test1


New Java Class
Java Class
Type name is discouraged. By convention, Java type names usually start with an uppercase letter
Source folder:
test1/src
Browse...
Package:
test1
Browse...
Enclosing type:
Browse...
Name:
test
Modifiers:
public
package
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
Remove
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Inherited abstract methods
Do you want to add comments? (Configure templates and default value here)
Generate comments
Finish
Cancel

Eclipse

- Alt+/

Eclipse


Java - test1/src/test1/test.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Java
Package Explorer
test1
src
test1
test.java
JRE System Library [JavaSI
test.java
1 package test1;
2
3 public class test {
4
5 public static void main(String[] args) {
6 // TODO Auto-generated method stub
7 System.out.println("test");
8 }
9
10 }
11
Task List
Find
All
Acti...
Connect Mylyn
Connect to your task and ALM tools or crea a local task.
Outline
test1
test
main(String[])
Problems
Javadoc
Declaration
0 items
Description
Resource
Path
Location
Type

Package and class

當遇上類別名稱(**class name**)相同的情況，這時候的權宜之計就是採用**package**（套件、套裝）的概念。

使用方法宣告一個**package**把類別包起來。就是在原來類別(**class**)程式的最前面再加上一行**package**宣告，這樣就不會再受到類別名稱相同的困擾了。

```
package package名稱 ;
```

Java 簡介

在**package**中納入**class**的程序：

在原來的工作目錄下建立一個與**package**名稱相同的資料夾，再把原始檔案儲存在這裡。

在原來的工作目錄下，輸入「**javac** 目錄名稱＼原始檔案名稱」，並加以編譯。

在原來的工作目錄下，輸入「**javac package**名稱**.**類別名稱」即可執行。

Java 簡介

即使類別本身原本已經是一個獨立的檔案，還是可以讓它們隸屬於同一個**package pa**之下。

```
package pa;

class CAR
{
   public static void main(String args[])
   {
      Car car1 = new Car();
      car1.show();
   }
}
```

CAR類別包含於package pa之中

package

從自己所屬的**package**內存取到其他**package**內的類別，有**2**件事您必須先完成：

1. 被存取的類別，前面必須加上public

2. 存取別人的類別的程式中，要清楚指定「被存取的package名稱.類別名稱」

package

在一個大型的**Java**程式中，由於會存取大量的類別，因此類別名稱重複的機會很大，因此面對重複的類別名稱時應該如何加以區別，在**Java**內部有一塊空間專門用來蒐集類別名稱，這個空間被稱為**namespace**（名稱空間），只要隸屬於不同**package**的類別就會被存放在不同的**namespace**當中。由於這個機制，只要**package**名稱不同，就不必擔心類別名稱是否相同的問題。

package

透過這種方式，在程式中如果要存取該類別時，完全不必再加上**package**名稱，只要直接使用類別名稱即可，上述這種方式在**Java**當中稱為匯入**(import)**。

```
import package名稱.類別名稱;
```

直接匯入其他package下的類別

```
import pc.Car
Car car1 = new Car();
```

此後終於不必再寫package名稱了

package

當**package**愈來愈多時，可以按照類別的角色規劃給不同的**package**，最後形成分類、階層化的**subpackage**，這種事先分類的做法對於未來進一步利用**package**撰寫程式有莫大的幫助。

package

- public static final double PI
- = 3.141592653589793;
- public static double cos(double a)
- {
- ...
- }

- import **static** java.lang.Math.PI

package

```
/**
 * @author  wpfang
 * @version %I%, %G%
 * @see     <a href="140.138.146.85">wp</a>
 */
package test;
public class simplest
{
   public static void main(String [] args)
   {
        System.out.println("the simplest example");
   }
}
```

package

```
C:\wpfang\teach\java\demo>dir
 磁碟區 C 中的磁碟沒有標籤。
 磁碟區序號:  1610-99B4

 C:\wpfang\teach\java\demo 的目錄

2017/04/30  上午 10:04    <DIR>          .
2017/04/30  上午 10:04    <DIR>          ..
2017/04/30  上午 09:59               257 simplest.java
               1 個檔案            257 位元組
               2 個目錄  258,458,763,264 位元組可用

C:\wpfang\teach\java\demo>javac -d . simplest.java

C:\wpfang\teach\java\demo>dir
 磁碟區 C 中的磁碟沒有標籤。
 磁碟區序號:  1610-99B4

 C:\wpfang\teach\java\demo 的目錄

2017/04/30  上午 10:05    <DIR>          .
2017/04/30  上午 10:05    <DIR>          ..
2017/04/30  上午 09:59               257 simplest.java
2017/04/30  上午 10:05    <DIR>          test
               1 個檔案            257 位元組
               3 個目錄  258,458,763,264 位元組可用

C:\wpfang\teach\java\demo>java -cp . test.simplest
the simplest example
```

- jar cf jar-file input-file(s)

Jar command

Option	Description
v	Produces verbose output on stdout while the JAR file is being built. The verbose output tells you the name of each file as it's added to the JAR file.
0 (zero)	Indicates that you don't want the JAR file to be compressed.
M	Indicates that the default manifest file should not be produced.
m	Used to include manifest information from an existing manifest file. The format for using this option is:jar cmf existing-manifest jar-file input-file(s) See Modifying a Manifest File for more information about this option.Warning: The manifest must end with a new line or carriage return. The last line will not be parsed properly if it does not end with a new line or carriage return.
-C	To change directories during execution of the command. See below for an example.

```
C:\wpfang\teach\java\demo>jar cvf test.jar test
已新增資訊清單
新增: test/ (讀=0)(寫=0)(儲存 0%)
新增: test/simplest.class (讀=435)(寫=291)(壓縮 33%)

C:\wpfang\teach\java\demo>dir
 磁碟區 C 中的磁碟沒有標籤。
 磁碟區序號:  1610-99B4

 C:\wpfang\teach\java\demo 的目錄

2017/04/30  上午 10:12    <DIR>          .
2017/04/30  上午 10:12    <DIR>          ..
2017/04/30  上午 09:59               257 simplest.java
2017/04/30  上午 10:05    <DIR>          test
2017/04/30  上午 10:12               849 test.jar
               2 個檔案          1,106 位元組
               3 個目錄  258,458,308,608 位元組可用
```

decompiler

- http://www.javadecompilers.com/

VARIABLE, CLASS AND OBJECT ORIENTED PROGRAMMING

variable, class and object

- Object
  - Objects have states and behaviors. Example: A dog has states - color, name, breed as well as behavior such as wagging their tail, barking, eating. An object is an instance of a class.
- Class
  - A class can be defined as a template/blueprint that describes the behavior/state that the object of its type supports.
- Methods
  - A method is basically a behavior. A class can contain many methods. It is in methods where the logics are written, data is manipulated and all the actions are executed.
- Instance Variables
  - Each object has its unique set of instance variables. An object's state is created by the values assigned to these instance variables.

variable

- data type variable [= value][, variable [= value] ...] ;

variable

- int a, b, c
- int a = 10, b = 10;
- byte B = 22;
- double pi = 3.14159;
- char a = 'a';

variable

- int [] A={1,2,3};

variable

- Local variables
- Instance variables
- Class/Static variables

Local variable

- declared in methods, constructors, or blocks.
- created when the method, constructor or block is entered and the variable will be destroyed once it exits the method, constructor, or block.
- Access modifiers cannot be used for local variables.
- visible only within the declared method, constructor, or block.
- implemented at stack level internally.
- There is no default value for local variables, so local variables should be declared and an initial value should be assigned before the first use.

Instance Variables

- declared in a class, but outside a method, constructor or any block.
- When a space is allocated for an object in the heap, a slot for each instance variable value is created.
- created when an object is created with the use of the keyword 'new' and destroyed when the object is destroyed.
- hold values that must be referenced by more than one method, constructor or block, or essential parts of an object's state that must be present throughout the class.
- can be declared in class level before or after use.
- Access modifiers can be given for instance variables.
- visible for all methods, constructors and block in the class. Normally, it is recommended to make these variables private (access level). However, visibility for subclasses can be given for these variables with the use of access modifiers.
- have default values.
  - For numbers, the default value is 0,
  - for Booleans it is false,
  - for object references it is null.
  - Values can be assigned during the declaration or within the constructor.
- Instance variables can be accessed directly by calling the variable name inside the class.
- within static methods (when instance variables are given accessibility), they should be called using the fully qualified name. *ObjectReference.VariableName*.

Class/Static Variables

- Class variables also known as static variables are declared with the static keyword in a class, but outside a method, constructor or a block.
- There would only be one copy of each class variable per class, regardless of how many objects are created from it.
- Static variables are rarely used other than being declared as constants. Constants are variables that are declared as public/private, final, and static. Constant variables never change from their initial value.
- Static variables are stored in the static memory. It is rare to use static variables other than declared final and used as either public or private constants.
- Static variables are created when the program starts and destroyed when the program stops.
- Visibility is similar to instance variables. However, most static variables are declared public since they must be available for users of the class.
- Default values are same as instance variables. For numbers, the default value is 0; for Booleans, it is false; and for object references, it is null. Values can be assigned during the declaration or within the constructor. Additionally, values can be assigned in special static initializer blocks.
- Static variables can be accessed by calling with the class name *ClassName.VariableName*.
- When declaring class variables as public static final, then variable names (constants) are all in upper case. If the static variables are not public and final, the naming syntax is the same as instance and local variables

Object-Oriented

- Polymorphism
- Inheritance
- Encapsulation
- Abstraction
- Classes
- Objects
- Instance
- Method
- Message Parsing

Encapsulation

Modifier	Class	Package	Subclass	World
public	Y	Y	Y	Y
protected	Y	Y	Y	N
no modifier	Y	Y	N	N
private	Y	N	N	N

INPUT AND OUTPUT

input and output

```
import java.util.Scanner;
public class io2
{
   public static void main(String [] args)
   {
         String user;
         Scanner s=new Scanner(System.in);
         user=s.next();
         System.out.println("Hello "+user);
   }
}
```

input and output

```
import java.util.Scanner;
public class io1
{
   public static void main(String [] args)
   {
            double inch,cm;
            Scanner s=new Scanner(System.in);
            inch=s.nextDouble();
            cm=inch*2.54;
            System.out.println(String.valueOf(cm)+" cm = "+inch+"\"");
   }
}
```

input and output

```
import java.io.File;
public class io3 {
    public static void main(String[] args) {
    File f = null;
    String[] strs = {"io1.java", "io2.java"};
    try {
      for(String s:strs ) {
        f = new File(s);
        boolean bool = f.canExecute();
        String a = f.getAbsolutePath();
        System.out.print(a);
        System.out.println(" is executable: "+ bool);
      }
    }catch(Exception e) {
      e.printStackTrace();
    }
  }
}
```

Filestream

```
import java.io.*;
public class CopyFile {
  public static void main(String args[]) throws IOException {
    FileInputStream in = null;
    FileOutputStream out = null;
    try {
      in = new FileInputStream("input.txt");
      out = new FileOutputStream("output.txt");
      int c;
      while ((c = in.read()) != -1) {
        out.write(c);
      }
    }finally {
      if (in != null) {
        in.close();
      }
      if (out != null) {
        out.close();
      }
    }
  }
}
```

CONDITION AND BRANCH

condition and branch

- if-else if –else
- switch

examples

LOOP

loop

- for –loop
- while –loop
- do-while –loop
- break
- continue

exampes

Loop and array

```
public class array
{
        public static void main(String [] args)
        {
                int [] A={1,2,3,4,5};
                int i;
                for(i=0;i<A.length;i++)
                        System.out.print(A[i]);
                System.out.println();
                for(int j : A)
                        System.out.print(j);
        }
}
```

FUNCTION

function

- Call by value
- Call by reference

Call by value

```
public class function1{
 static void change(int data){
            data=data+100;
            System.out.println("in "+ data);
 }

 public static void main(String args[]){
            int x=100;
            System.out.println("before "+ x);
            change(x);
            System.out.println("after "+ x);
 }
}
```

before 100

in 200

after 100

Call by reference

```
public class function2{
 static void change(T data){
                data.x=data.x+100;
                System.out.println("in "+ data.x);
 }

 public static void main(String args[]){
                T x=new T();
                x.x=100;
                System.out.println("before "+ x.x);
                change(x);
                System.out.println("after "+ x.x);
 }
}
class T
{
   int x;
 }
```

before 100

in 200

after 200

CLASS RECALL

Modifiers

- Access Control Modifiers
  - Visible to the package, the default. No modifiers are needed.
  - Visible to the class only (private).
  - Visible to the world (public).
  - Visible to the package and all subclasses (protected).
- Non-Access Modifiers
  - The *static* modifier for creating class methods and variables.
  - The *final* modifier for finalizing the implementations of classes, methods, and variables.
  - The *abstract* modifier for creating abstract classes and methods.
  - The *synchronized* and *volatile* modifiers, which are used for threads.

Abstract Class

開頭第1個字是使用關鍵字abstract做為修飾子，像這樣以 abstract 做為開頭的類別，我們稱它是抽象類別 (abstract class)。

```
abstract class 類別名稱
{
  宣告field;
  abstract 傳回值資料型態 method名稱(參數…);
}
```

關於抽象類別有二點您必須特別注意：

1. 和一般的類別不同，抽象類別不能拿來直接產生新物件。
2. 抽象類別內部的method沒有定義處理的方式。

Abstract Class

雖然抽象類別不能直接產生物件，但是準備好抽象類別的變數、陣列之後，仍然可以讓其中的陣列元素指向子類別。

抽象類別可以繼續往下繼承延伸出其他子類別，不過如果您還是想要利用抽象類別的子類別產生新物件的話，必須注意：

原本在抽象類別當中沒有定義抽象方法的處理方式，因此繼承到子類別之後要加以明確定義，也就是加以改寫(overriding)。

使用**instanceof**運算子，可以知道變數所指向的物件，其所屬的究竟是哪一個類別**(class)**。

介面 (interface)

介面的宣告，和宣告一般類別不同的是，原本宣告類別時一開始應該使用的「**class**」，在宣告介面時則改成「**interface**」，其他的部份和一般類別宣告一樣。

```
interface 介面名稱
{
   資料型態 field名稱 = 運算式 ;
   傳回值的資料型態 method名稱();
}
```

field一定要先進行初值化

這裏也是沒有定義method的處理方式

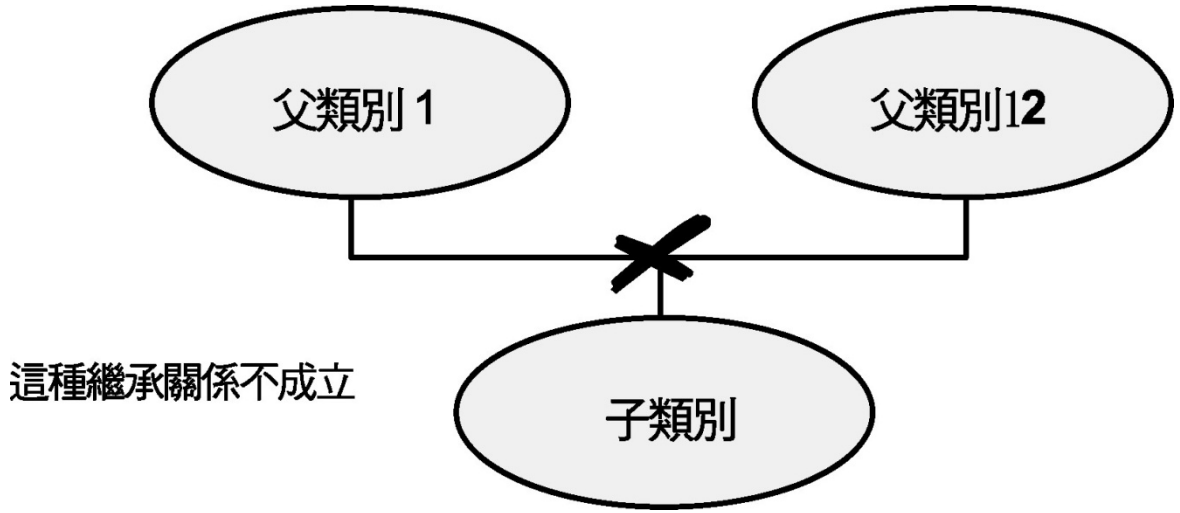
使用介面設計程式

使用介面的方式是把介面和類別融合在一起，特別是把介面納入到類別當中，這種情況稱為介面的實作(implementation)。(See Sample3.java)

```
class 類別名稱 implements 介面名稱
{
   …
}
```

Multiple Inheritance

製作大型程式時，會需要組合多個類別或介面，也可能想繼承多個父類別，但Java (跟C++ 不同) 不允許一個子類別延伸2個父類別，雖然如此，Java可實做2個或以上的介面，仍然可達到部分多重繼承(multiple inheritance)的目的。



實作2個以上的介面

1個類別可以和2個以上的介面實作在一起。雖然Java並不直接支援多重繼承，但是透過實作2個以上介面的方式，仍然可以達成多重繼承method名稱的目的。 (See Sample4.java)

```
class 類別名稱 implements 介面名稱1,介面名稱2,..
{
    …
}
```

可以實作2個以上的介面

介面的延伸

介面和一般類別一樣，可以透過延伸(**extends**)的方式產生新的介面。原來的介面稱為父介面(**superinterface**)，衍生出來的稱為子介面(**subinterface**)，和先前的經驗一樣，介面的延伸也是要透過關鍵字**extends**。

```
interface 子介面名稱 extends 父介面1, 父介面2
{
   …
}
```

介面也可以加以延伸

Class and object

```
class CCAR
{




}
```

```
public class WORLD
{
public static void main(String[] string)
{




}
}
```

Class and object

```
class CCAR
{




}
```

```
public class WORLD
{
public static void main(String[] string)
{




}
}
```

EXCEPTION

Exception

- 將異常由被呼叫者轉為呼叫者處理

Exception

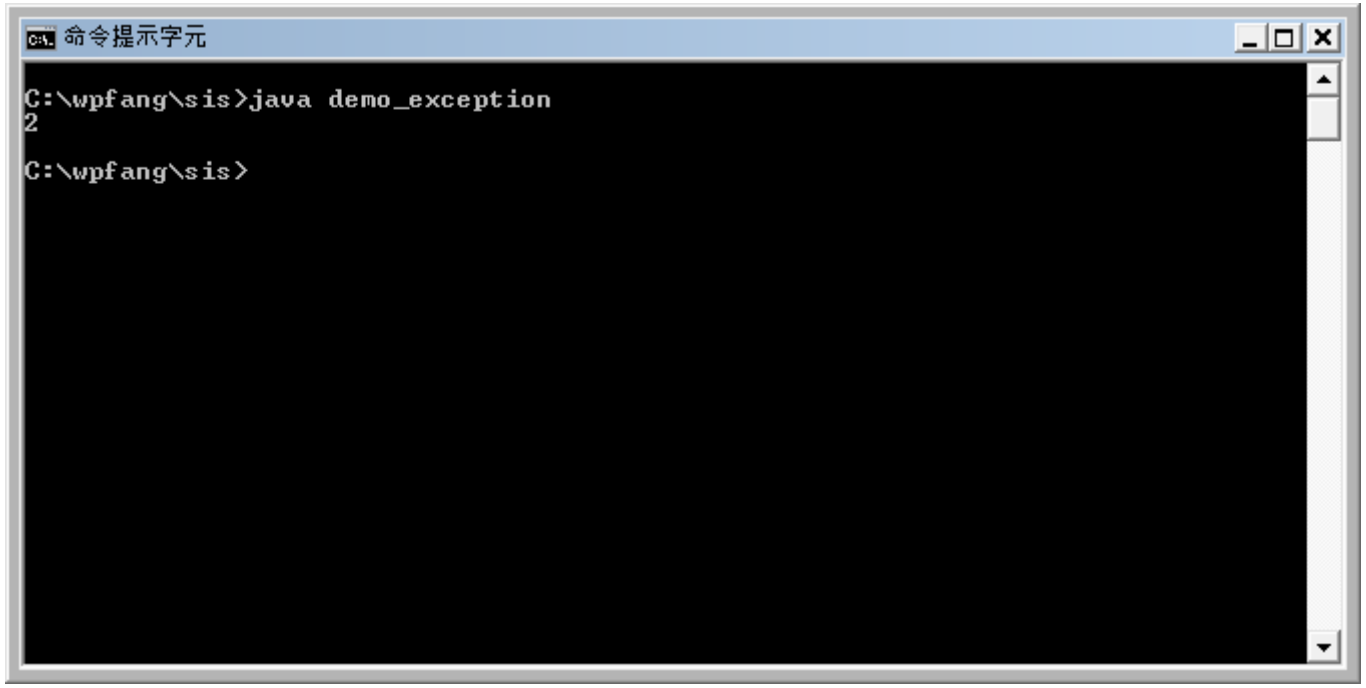
- Example:

  a/b=?

Exception

```
public class demo_exception
{
  public static int div(int a,int b)
  {
        return a/b;
   }
  public static void main(String [] args)
  {
      System.out.println(div(4,2));
  }
}
```

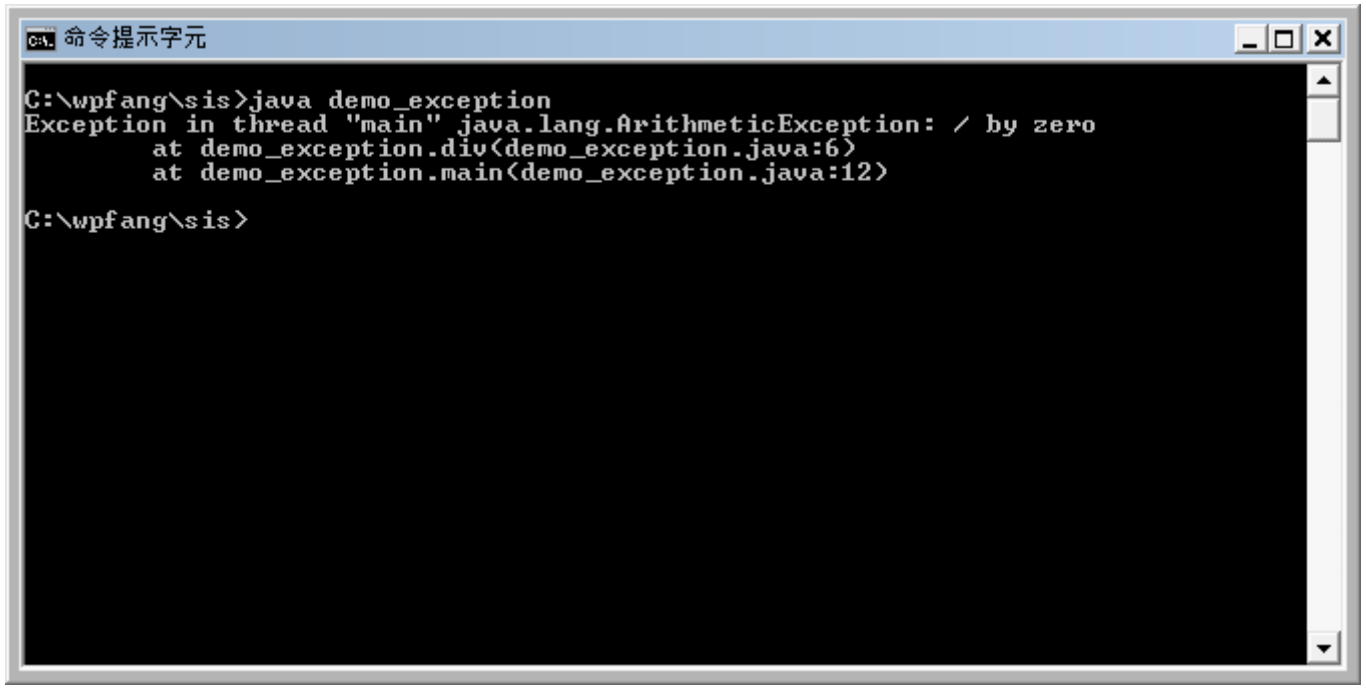
Exception



Exception

```
public class demo_exception
{
  public static int div(int a,int b)
  {
        return a/b;
   }
  public static void main(String [] args)
  {
     System.out.println(div(4,0));
  }
}
```

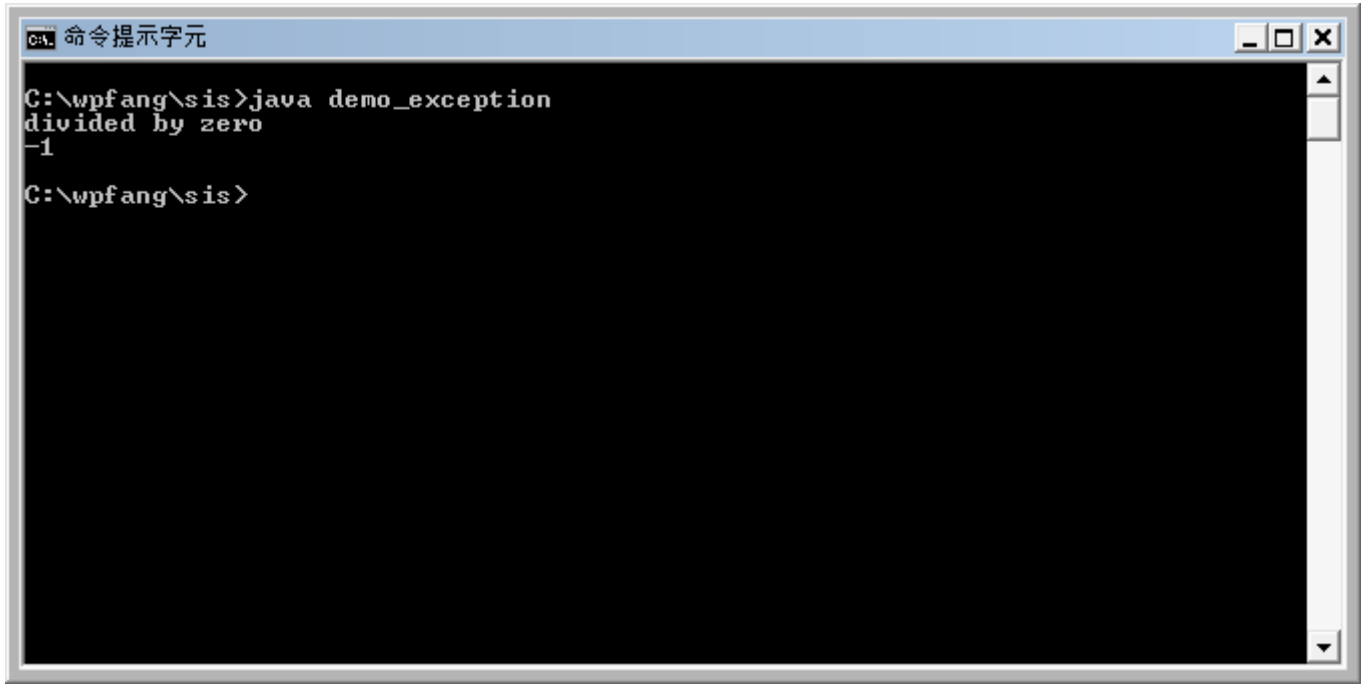
Exception


命令提示字元
C:\wpfang\sis>java demo_exception
Exception in thread "main" java.lang.ArithmeticException: / by zero
at demo_exception.div(demo_exception.java:6)
at demo_exception.main(demo_exception.java:12)
C:\wpfang\sis>

Exception

```
public class demo_exception
{
  public static int div(int a,int b)
  {
        if(0==b) {System.out.println("divided by zero");return -1;}
         return a/b;
   }
  public static void main(String [] args)
  {
      System.out.println(div(4,0));
  }
}
```

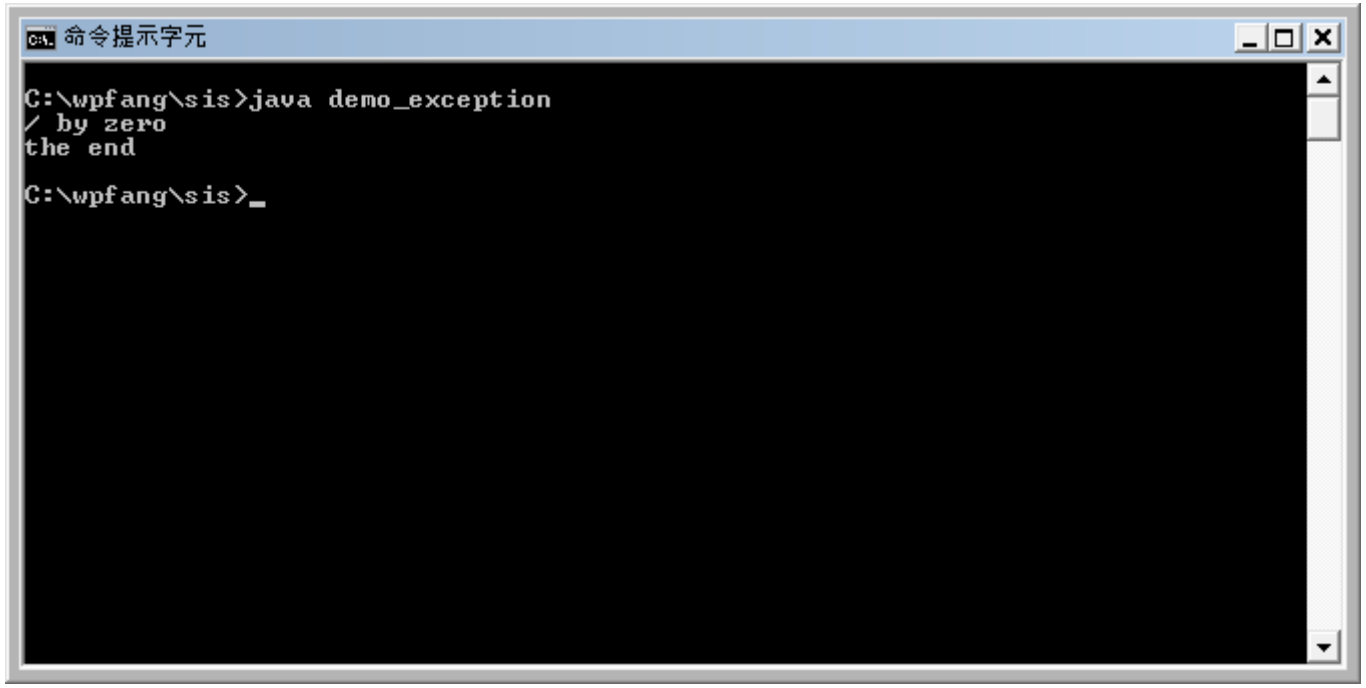
Exception


命令提示字元
C:\wpfang\sis>java demo_exception
divided by zero
-1
C:\wpfang\sis>

Exception

```
public class demo_exception
{
  public static int div(int a,int b)
  {
        return a/b;
   }
  public static void main(String [] args)
  {
       try
      {
       System.out.println(div(4,0));
       }
       catch(ArithmeticException e)
       {
             System.out.println(e.getMessage());
       }
        System.out.println("the end");
  }
}
```

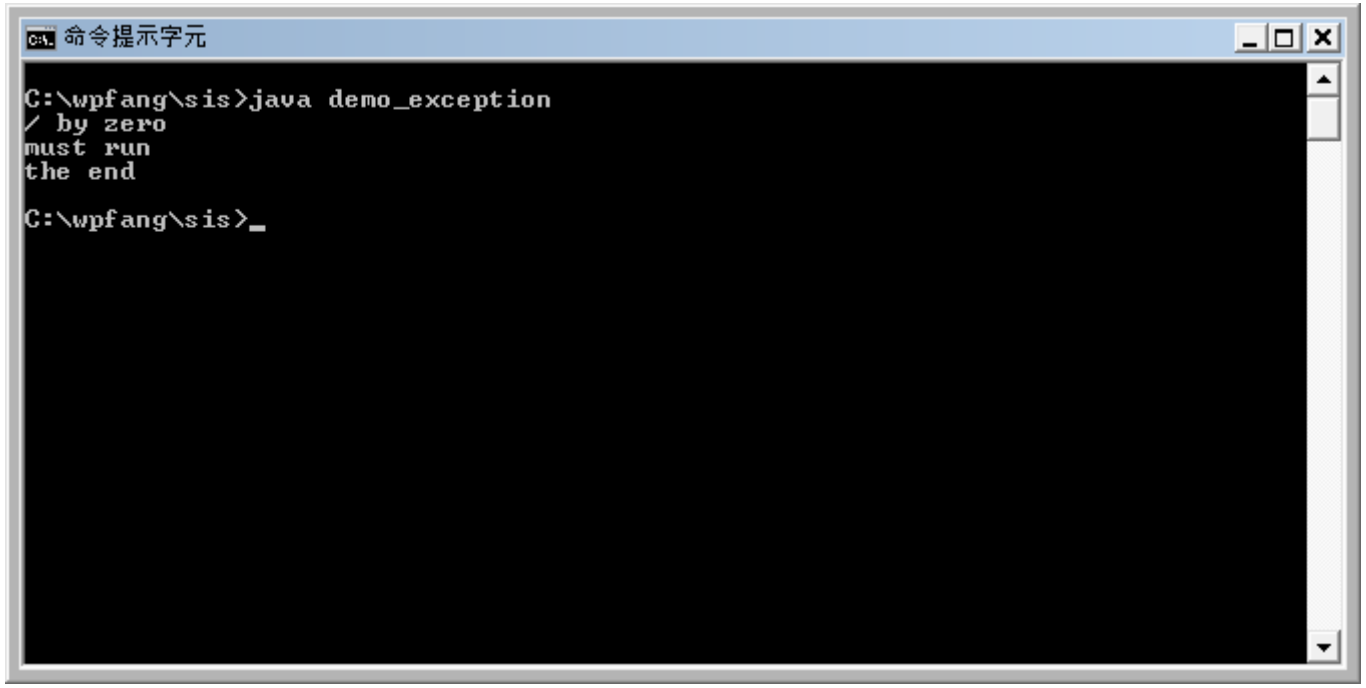
Exception


命令提示字元
C:\wpfang\sis>java demo_exception
/ by zero
the end
C:\wpfang\sis>

Exception

```
public static void main(String [] args)
 {
      try
     {
      System.out.println(div(4,0));
       }
       catch(Exception e)
       {
              System.out.println(e.getMessage());
       }
       finally
       {
                 System.out.println("must run");
        }
            System.out.println("the end");
  }
}
```

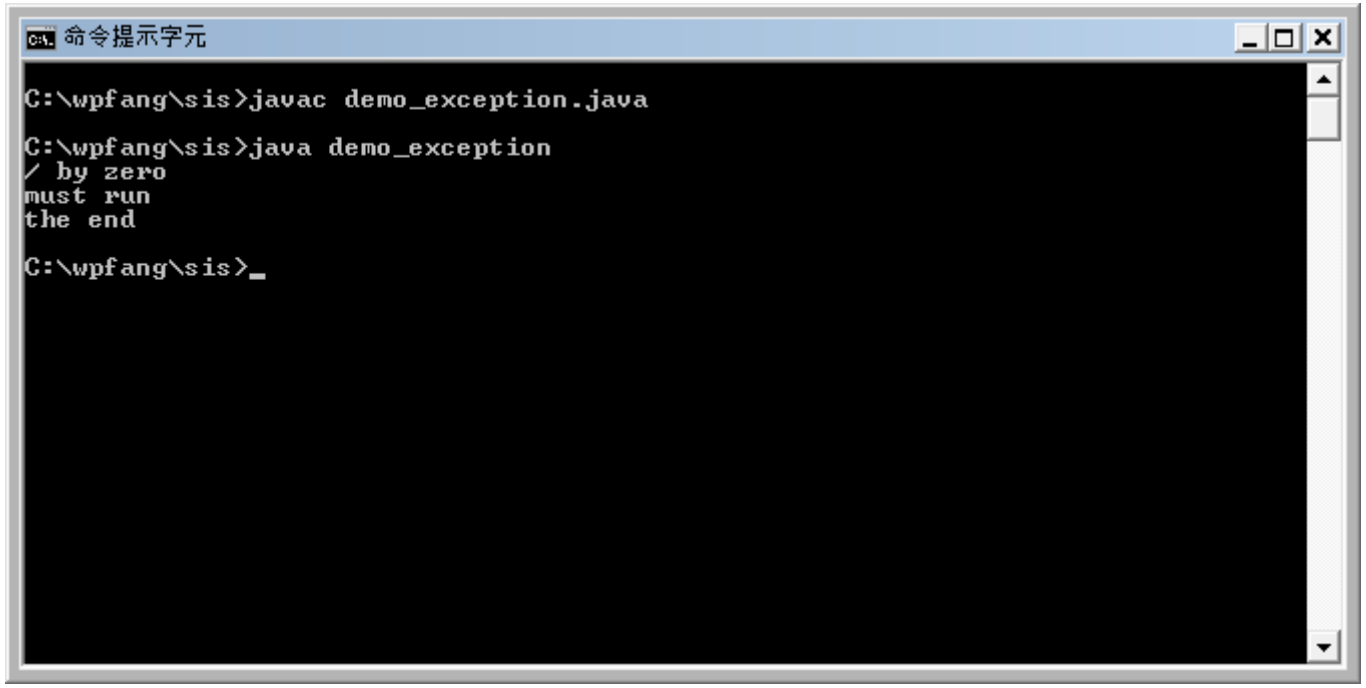
Exception


命令提示字元
C:\wpfang\sis>java demo_exception
/ by zero
must run
the end
C:\wpfang\sis>

Exception

```
public class demo_exception
{
  public static int div(int a,int b)
  {
        return a/b;
   }
  public static void main(String [] args)
  {
       try
      {
       System.out.println(div(4,0));
       }
       catch(ArithmeticException e)
       {
             System.out.println(e.getMessage());
       }
       finally
       {
              System.out.println("must run");
        }
        System.out.println("the end");
  }
}
```

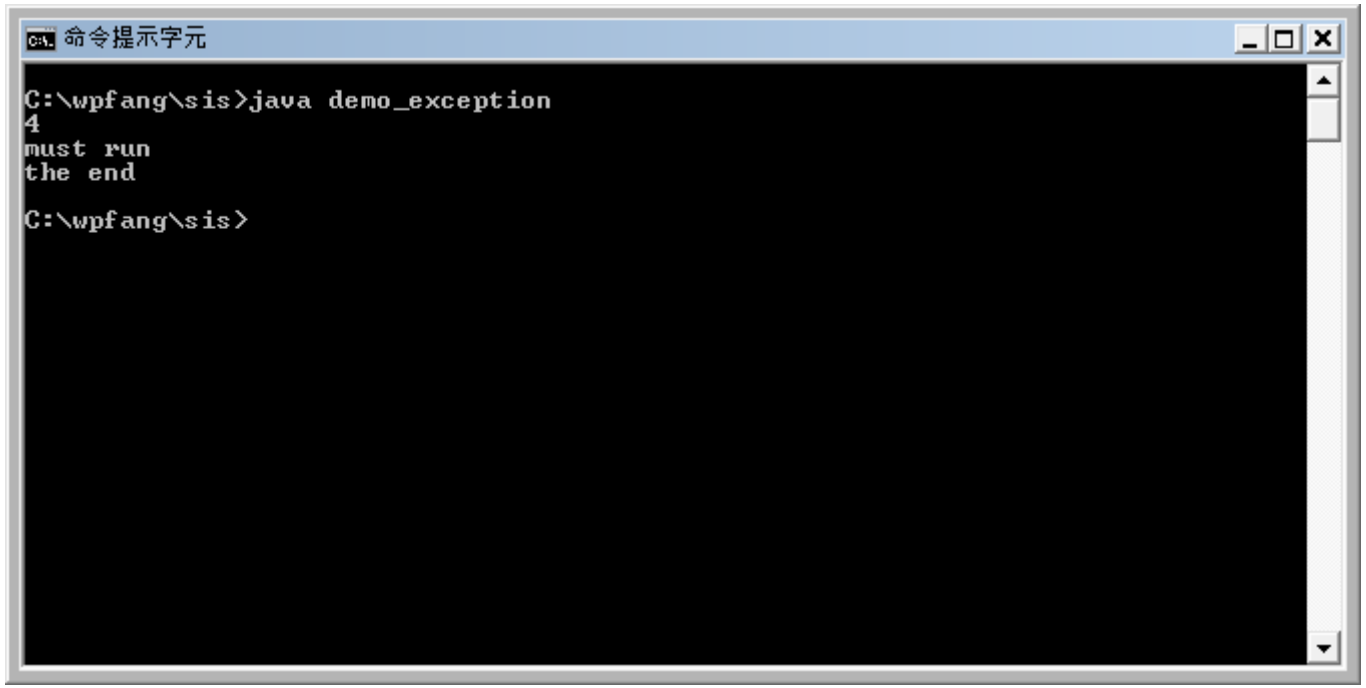
Exception


命令提示字元
C:\wpfang\sis>javac demo_exception.java
C:\wpfang\sis>java demo_exception
/ by zero
must run
the end
C:\wpfang\sis>

Exception

```
public class demo_exception
{
  public static int div(int a,int b)
  {
        return a/b;
   }
  public static void main(String [] args)
  {
       try
      {
       System.out.println(div(4,1));
       }
       catch(ArithmeticException e)
       {
             System.out.println(e.getMessage());
       }
       finally
       {
              System.out.println("must run");
        }
        System.out.println("the end");
  }
}
```

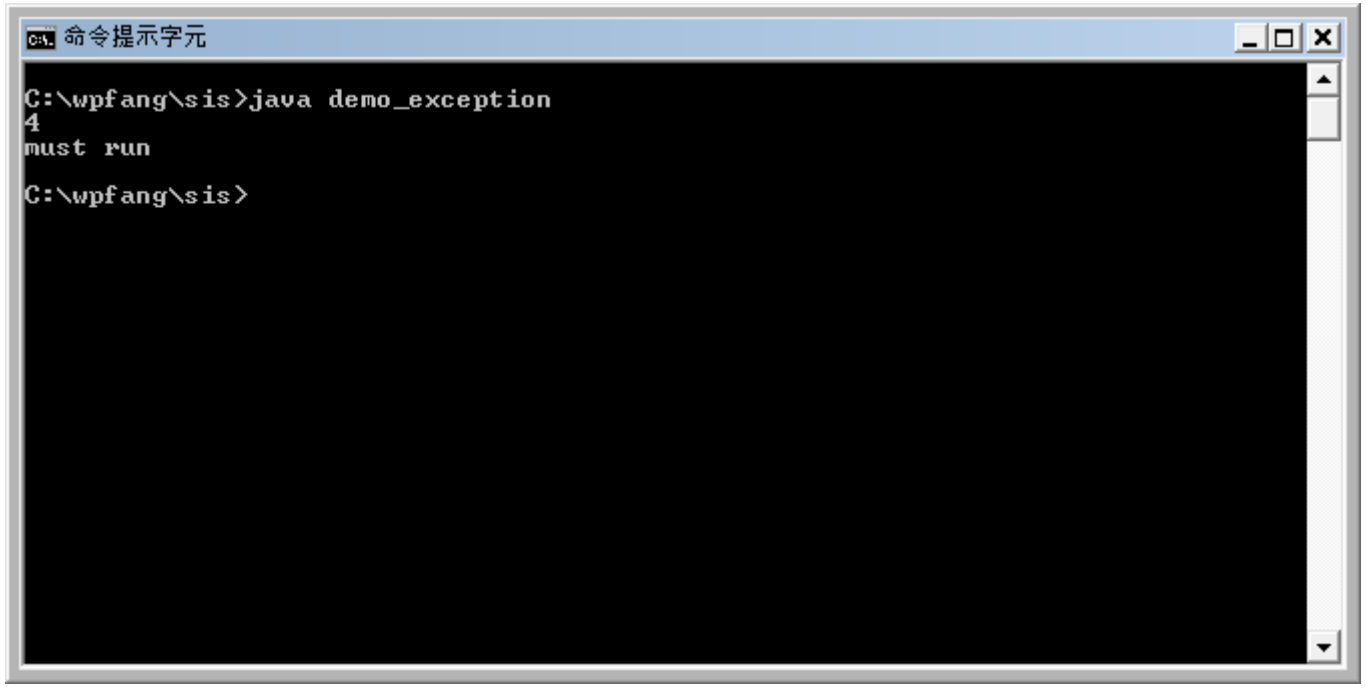
Exception


命令提示字元
C:\wpfang\sis>java demo_exception
4
must run
the end
C:\wpfang\sis>

Exception

```
public class demo_exception
{
  public static int div(int a,int b)
  {
        return a/b;
   }
  public static void main(String [] args)
  {
       try
      {
       System.out.println(div(4,1));
        }
        catch(ArithmeticException e)
        {
              System.out.println(e.getMessage());
        }
        finally
        {
                System.out.println("must run");
         }
         System.exit(0);
         System.out.println("the end");
  }
}
```

Exception


命令提示字元
C:\wpfang\sis>java demo_exception
4
must run
C:\wpfang\sis>

CheckException

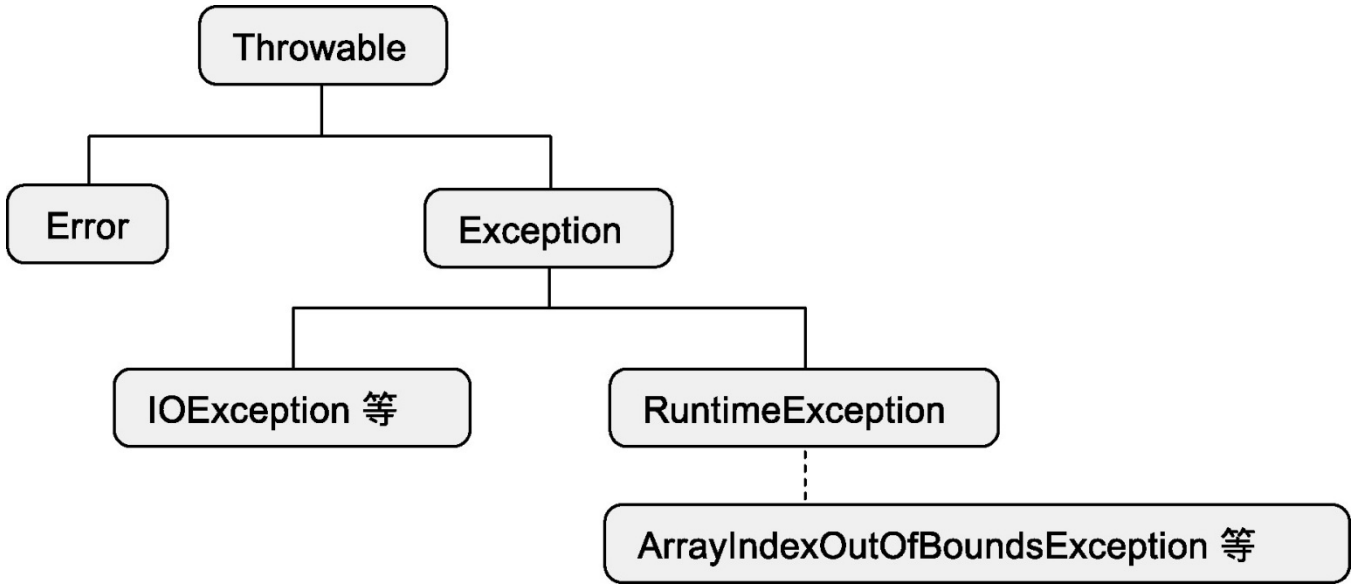
- ClassNotFoundException
- FileNotFoundException
- CloneNotSupportException
- InterrupteException
- IOException

RuntimeException

- ArithmeticException
- ArrayIndexOutOfBoundsException
- ArrayStoreException
- ClassCastException
- IndexOutOfBoundsException
- IllegalArgumentException
- NullPointerException
- NumberFormatException
- SecurityException
- (可以沒有try-catch)

Exception

從下圖可以看出，從**Throwable**類別延伸出**Error**類別和**Exception**類別。**Error**類別是指那些造成程式無法繼續執行的錯誤，通常也是比較嚴重的錯誤，在此我們討論到的例外處理都是屬於**Exception**類別。



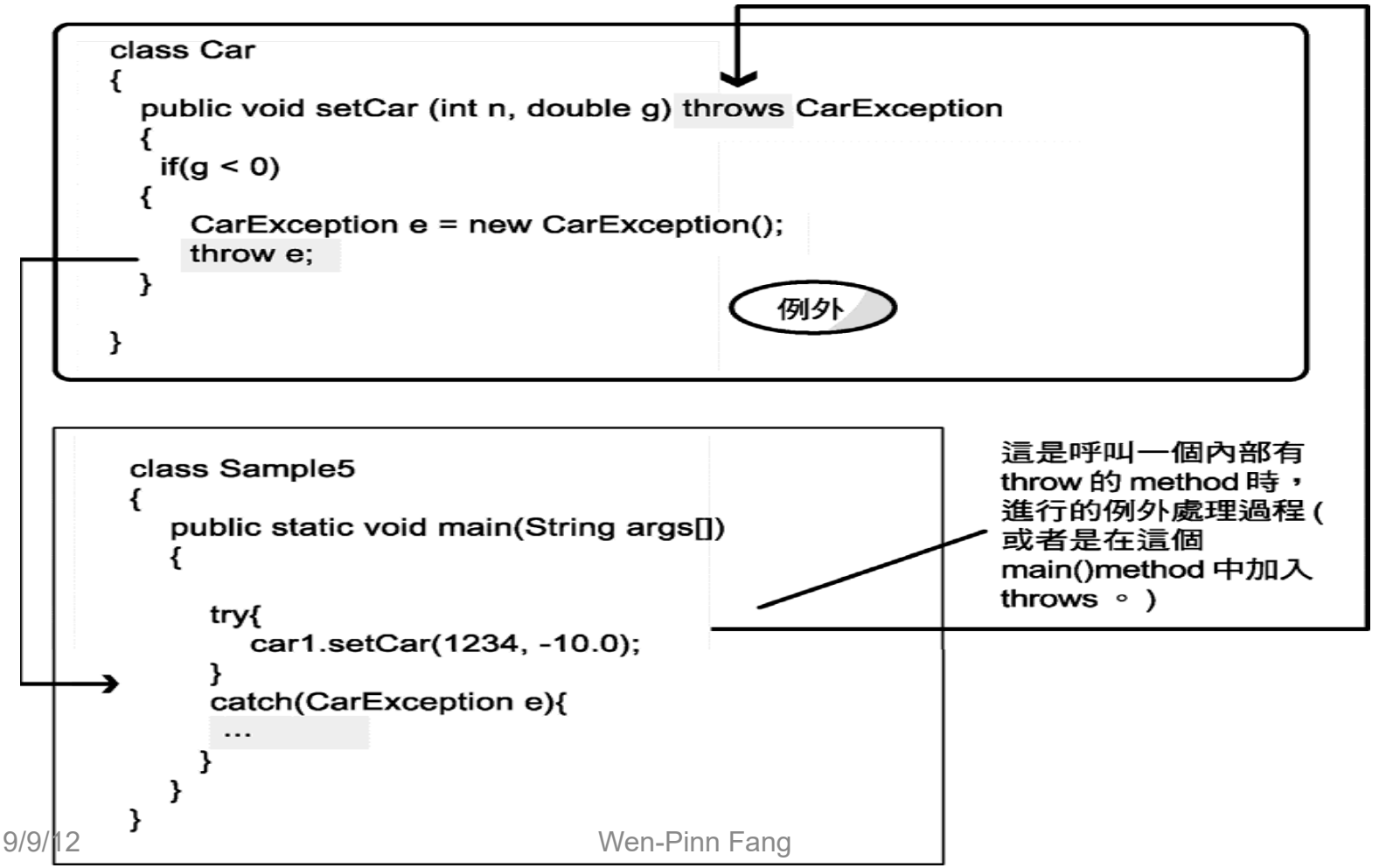
Exception

必須藉由**throw**這個關鍵字才能把例外拋出。

```
public void setCar (int n, double g) throws CarException
{
 if(g < 0)
 {
   CarException e = new CarException();
   throw e;
 }
}
```

Exception

在**main()**當中呼叫**setCar()method**的時候，**setCar()** 能夠對例外狀況拋出例外類別。所以必須在呼叫**setCar()**的**main()**中處理例外。



Exception

撰寫一個可能會送出例外的**method**時，原則上：

使用**try～catch**，在**method**內直接處理例外，或是**...**

透過**throws**，把處理例外的工作交給原呼叫程式所在的**method**。

選擇第一種方法的時候，因為在**method**內直接處理例外，並不需要特別記述什麼。選擇第二種方法的時候，要在**method**中加入**throws**，才能表示把例外交給原始程式所在的**method**處理。

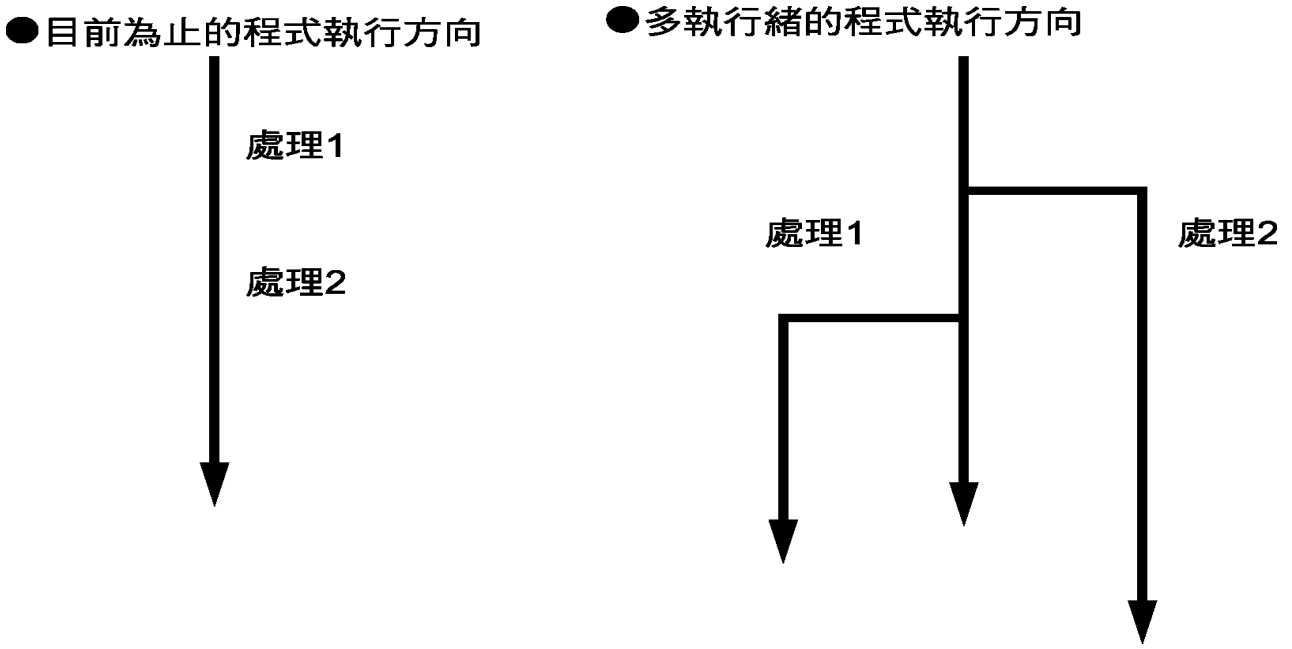
Exception

```
public class MyStackTrace
{
public static void main(String[] args)
{
     try
    {
        a();
    }catch(HighLevelException e)
    {
       e.printStackTrace();
    }
}
```

MULTITHREAD

關於執行緒(Thread)

對Java來說，可以同時擁有多個「程式處理流程」。在程式內部好像具有執行多個處理 (Task) 的架構，其中的每一個程式處理流程都被稱為是一個執行緒 (thread)。



啟動執行緒

啟動執行緒之前，必須做好執行緒的準備工作，也就是從類別庫的**Thread**類別當中，延伸出一個子類別。

```
class Car extends Thread
{
   public void run()

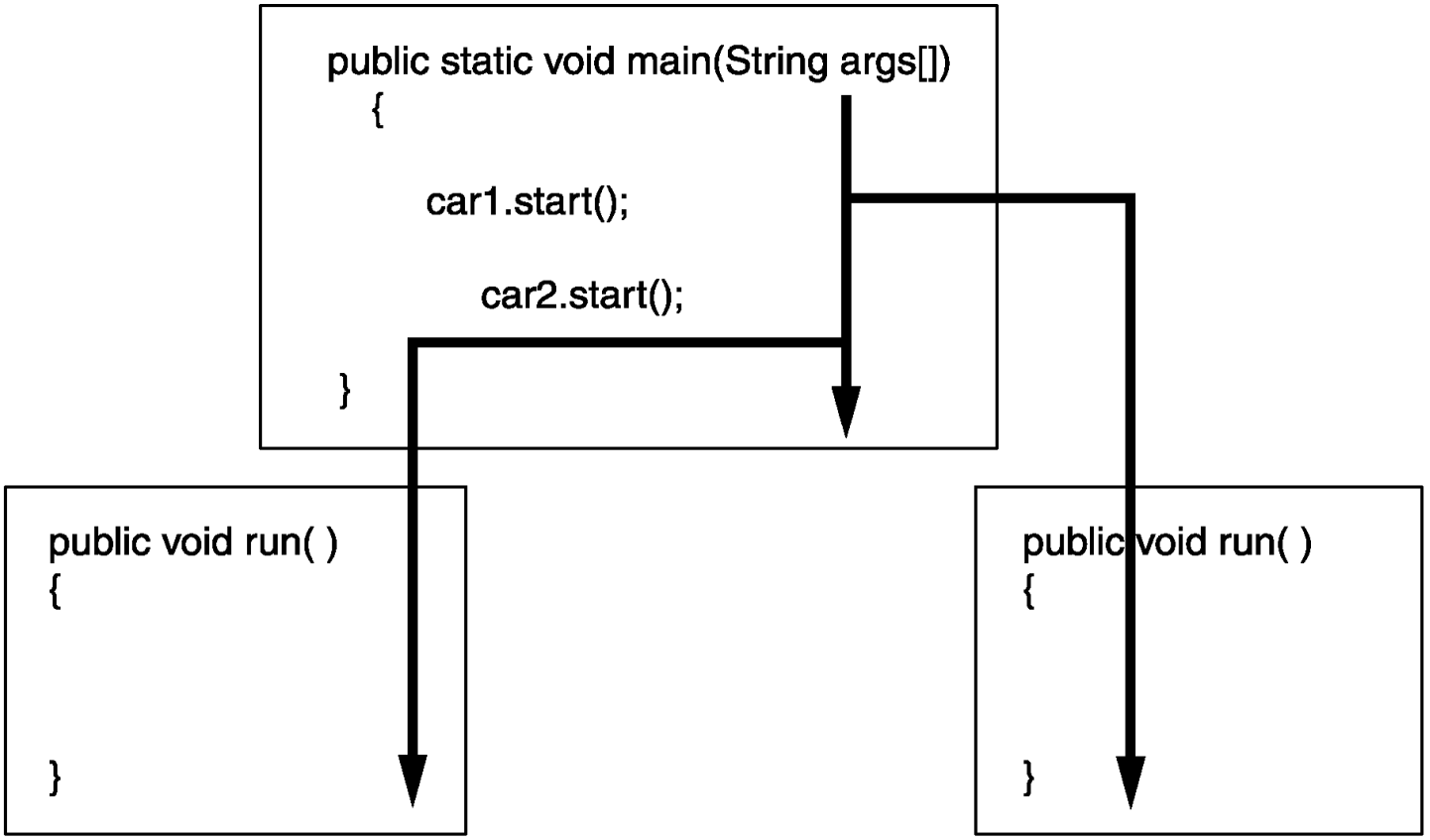
   {
      以其他執行緒執行處理;
      ...
   }
}
```

從Thread類別延伸出子類別

定義其中的run() method

啟動多個執行緒

您可以視程式上的需要增加執行



暫停、等待執行緒

我們可以透過 **sleep() method**，讓執行中的執行緒暫時停下來。

由於**sleep() method**是繼承自**Thread**類別下的子類別，因此，即使寫成**Thread.sleep(1000)**的形式，還是可以暫停執行緒。

我們可以利用**join()**這個**method**，做到讓某一個執行緒等待另一個執行緒執行結束後，自己再開始執行。

另一種建立執行緒的方法

如果本身已經是**Thread**類別的子類別，卻又不得不同時從其他類別加以繼承的話，此時可使用類別庫當中的**Runnable**介面，透過它，還是可以正常啟動執行緒。

```
Class Car extends Vehicle implements
      Runnable
{
    ...
}
```

啟動執行緒共有下列二種方式：

- 建立 Thread 類別的子類別的方式
- 實作 Runnable 介面的方式

關於「同步 (synchronization)」

為了避免資料互相抵觸，必須做到當某個執行緒正在執行匯入處理時，其他執行緒絕對不能同時進行匯入處理；因此，**add() method**需改寫。

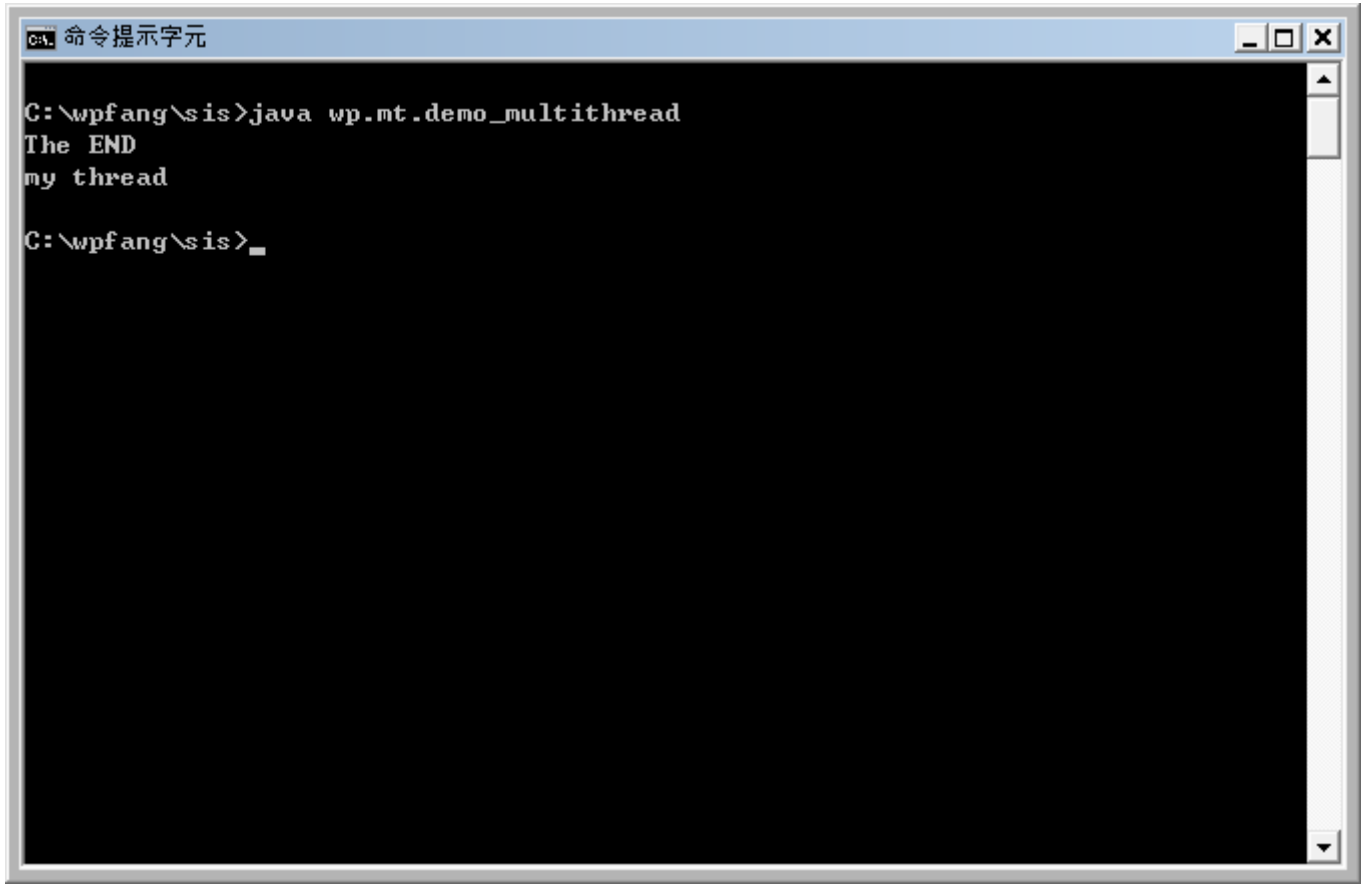
```
public synchronized void add(int a)
{
    ...
}
```

加上這個關鍵字

透過 **synchronized** 在時間點上取得協調一致的現象，稱為同步**(synchronization)**。

extends Thread

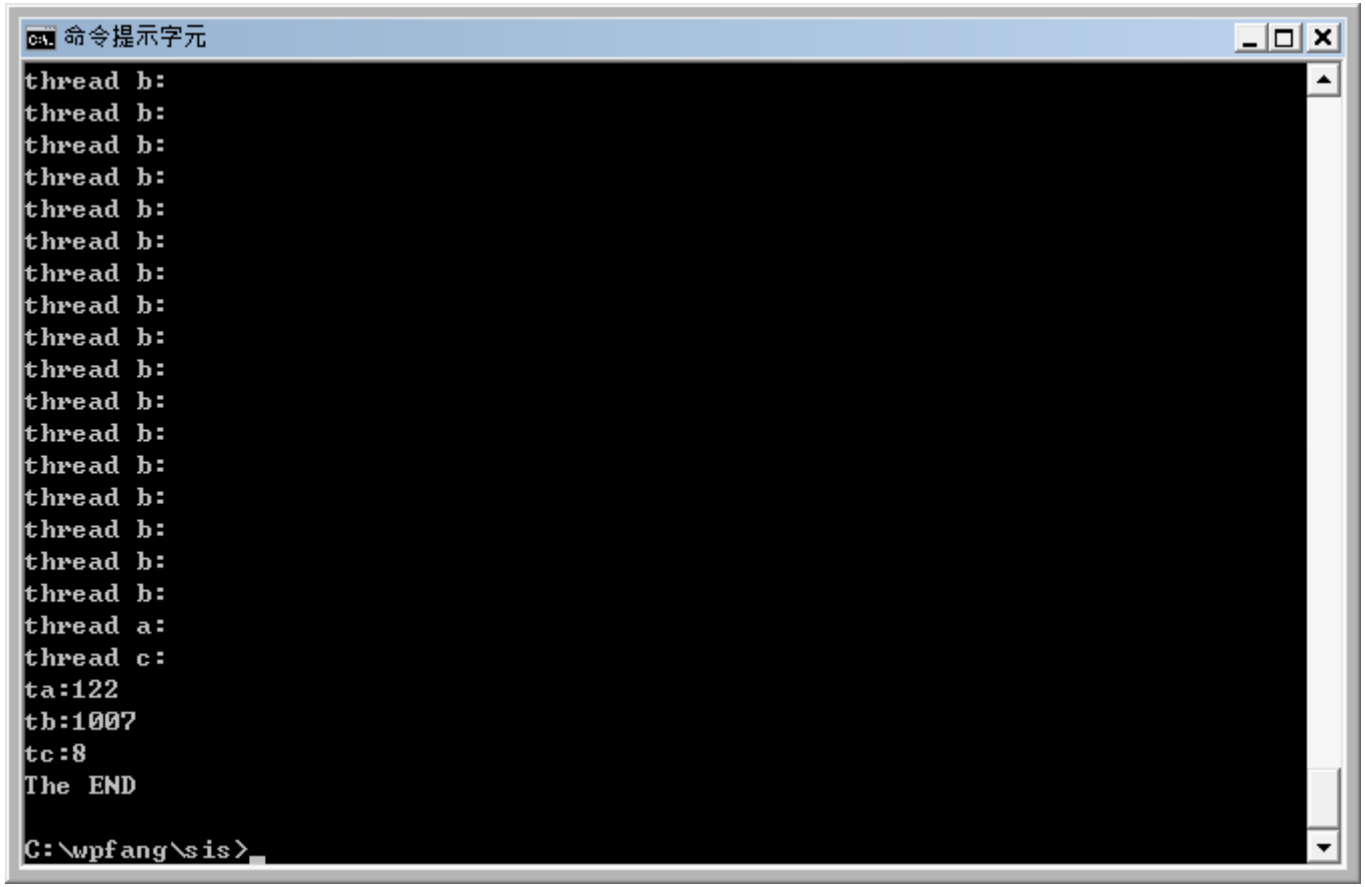
```
package wp.mt;
import java.lang.Thread;
class MyThread extends Thread
{
   MyThread()
   {
                start();
   }
   public void run()
   {
        System.out.println("my thread");
   }
}
public class demo_multithread
{
    public static void main(String [] args)
    {
                MyThread t=new MyThread();
      System.out.println("The END");
    }
}
```


命令提示字元
C:\wpfang\sis>java wp.mt.demo_multithread
The END
my thread
C:\wpfang\sis>

setPriority

```
package wp.mt;
import java.lang.Thread;
class MyThread implements Runnable
{
   String name;
   Thread t;
   boolean flag=true;
   public int count=0;
   MyThread(String name,int priority)
   {
      this.name=name;
      t=new Thread(this,name);
      t.setPriority(priority);
              t.start();
   }
   public void run()
   {
              while(flag)
     {
          System.out.println(name);
                                    count++;
     }
   }
   public void stop()
   {
       flag=false;
   }
}
```

```
public class demo_multithread2
{
   public static void main(String [] args)
   {
              MyThread ta=new MyThread("thread a:",8);
              MyThread tb=new MyThread("thread b:",7);
              MyThread tc=new MyThread("thread c:",4);
     try
     {
        Thread.sleep(1000);
     }
     catch(InterruptedException e)
     {
          System.out.println(e.toString());
     }
     ta.stop();
     tb.stop();
     tc.stop();
     System.out.println("ta:"+ta.count);
     System.out.println("tb:"+tb.count);
     System.out.println("tc:"+tc.count);
     System.out.println("The END");
   }
}
```


命令提示字元
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread b:
thread a:
thread c:
ta:122
tb:1007
tc:8
The END
C:\wpfang\sis>

implements Runnable

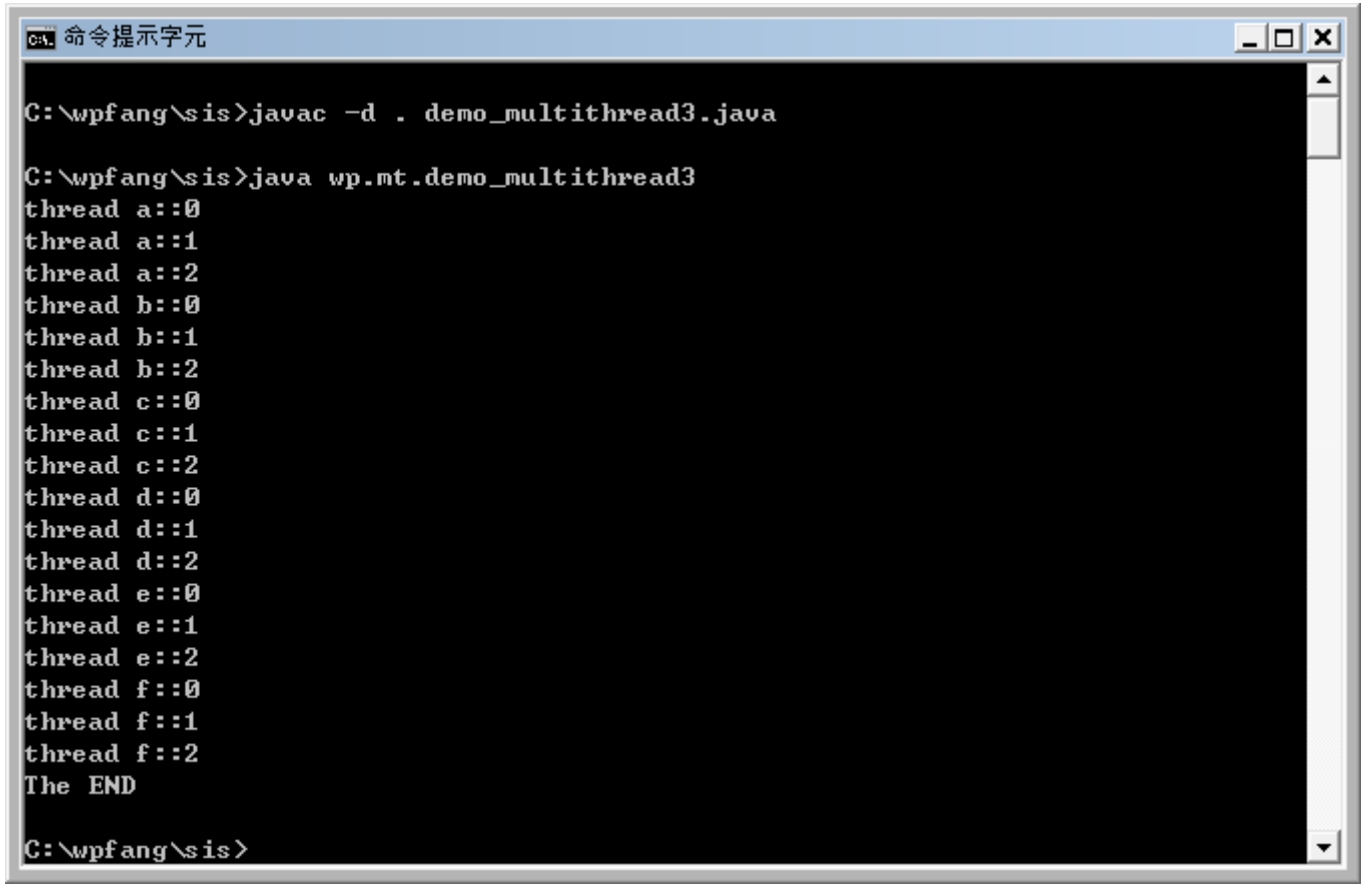
```
package wp.mt;
import java.lang.Thread;
class MyThread implements Runnable
{
   String name;
   Thread t;
   boolean flag=true;
   public int count=0;
   MyThread(String name,int priority)
   {
      this.name=name;
      t=new Thread(this,name);
      t.setPriority(priority);
                  t.start();
      try
      {
        t.join();
      }
      catch(InterruptedException e)
      {
         System.out.println(e.toString());
      }
   }
   public void run()
   {

       for(int j=0;j<3;j++)
       {
        for(int i=0;i<100000;i++)
            t.yield();
        System.out.println(name+":"+j);
       }
   }

}
```

```
public class demo_multithread3
{
    public static void main(String [] args)
    {
          MyThread ta=new MyThread("thread a:",9);
          MyThread tb=new MyThread("thread b:",5);
          MyThread tc=new MyThread("thread c:",4);
          MyThread td=new MyThread("thread d:",3);
          MyThread te=new MyThread("thread e:",2);
          MyThread tf=new MyThread("thread f:",1);

      System.out.println("The END");
    }
}
```


命令提示字元
C:\wpfang\sis>javac -d . demo_multithread3.java
C:\wpfang\sis>java wp.mt.demo_multithread3
thread a::0
thread a::1
thread a::2
thread b::0
thread b::1
thread b::2
thread c::0
thread c::1
thread c::2
thread d::0
thread d::1
thread d::2
thread e::0
thread e::1
thread e::2
thread f::0
thread f::1
thread f::2
The END
C:\wpfang\sis>

isAlive

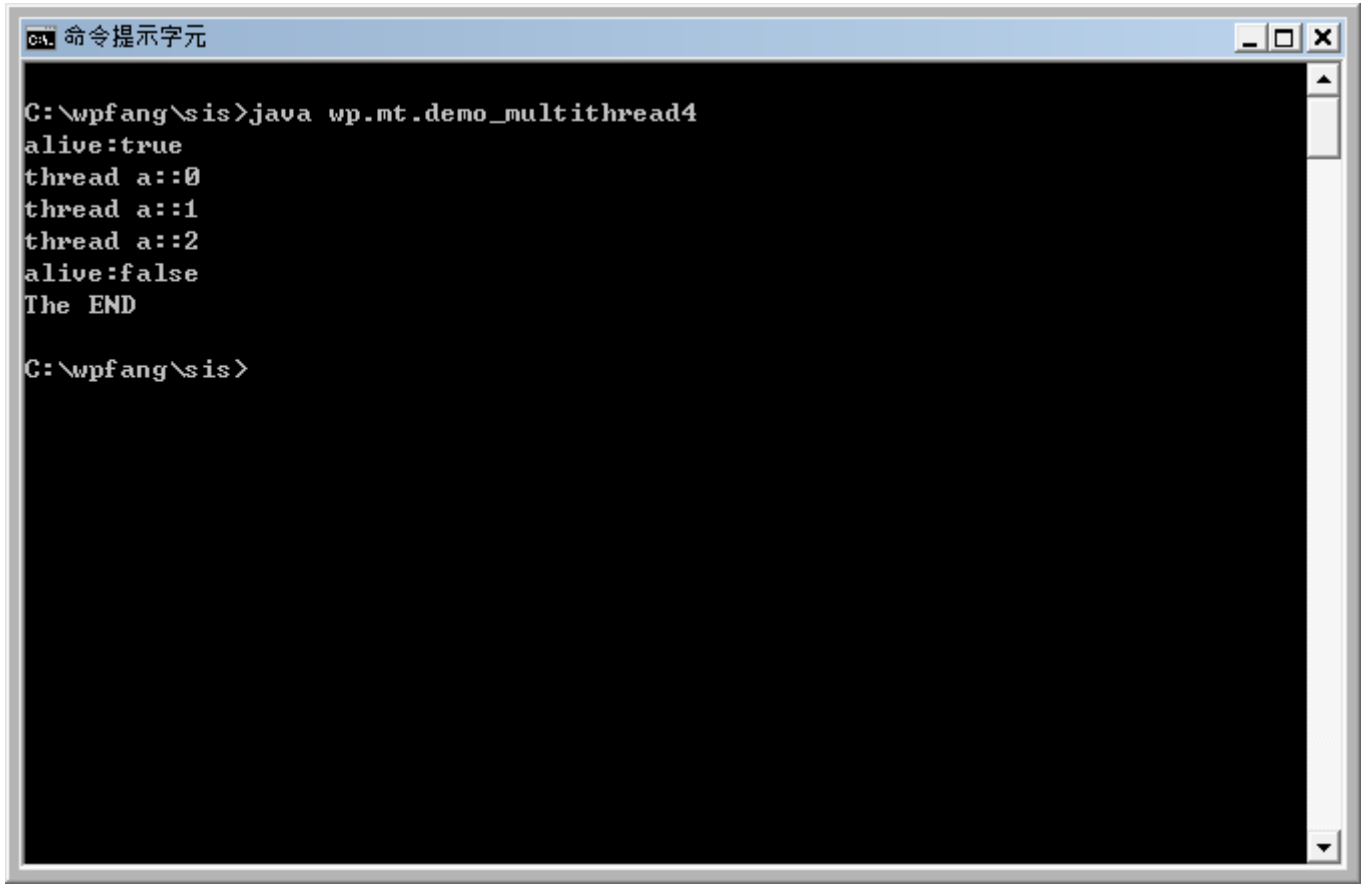
```
package wp.mt;
import java.lang.Thread;
import java.lang.*;
class MyThread extends Thread
{
   String name;
   Thread t;
   boolean flag=true;
   public int count=0;
   MyThread(String name,int priority)
   {
      this.name=name;
      setPriority(priority);
               start();
   }
   public void run()
   {

       for(int j=0;j<3;j++)
       {
        for(int i=0;i<100000;i++)
            t.yield();
        System.out.println(name+":"+j);
       }
   }

}
```

```
public class demo_multithread4
{
    public static void main(String [] args)
    {
          MyThread ta=new MyThread("thread a:",7);
      System.out.println("alive:"+ta.isAlive());
      try
       {
        Thread.sleep(100);
       }
       catch(InterruptedException e)
       {
           System.out.println(e.toString());
       }
      System.out.println("alive:"+ta.isAlive());

     System.out.println("The END");
    }
}
```


命令提示字元
C:\wpfang\sis>java wp.mt.demo_multithread4
alive:true
thread a::0
thread a::1
thread a::2
alive:false
The END
C:\wpfang\sis>

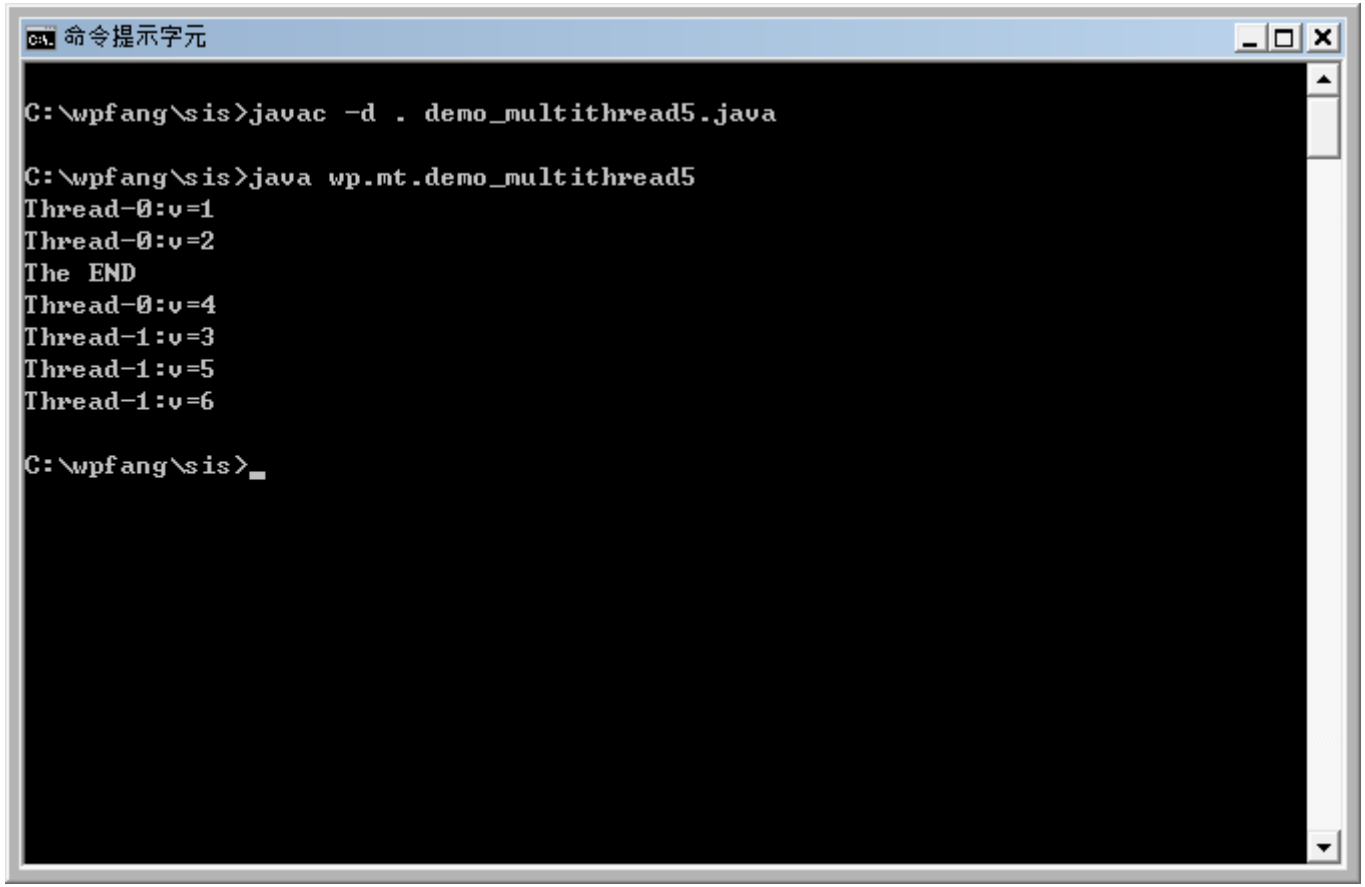
getName

```
package wp.mt;
import java.lang.Thread;

class MyThread implements Runnable
{
   private int v;
   public void run()
   {
     for(int i=0;i<3;i++)
      {
          v++;
         System.out.println(Thread.currentThread().getName()+":v="+v);
      }

   }

}
public class demo_multithread5
{
    public static void main(String [] args)
    {
              MyThread t=new MyThread();
     (new Thread(t)).start();
     (new Thread(t)).start();
     System.out.println("The END");
    }
}
```


命令提示字元
C:\wpfang\sis>javac -d . demo_multithread5.java
C:\wpfang\sis>java wp.mt.demo_multithread5
Thread-0:v=1
Thread-0:v=2
The END
Thread-0:v=4
Thread-1:v=3
Thread-1:v=5
Thread-1:v=6
C:\wpfang\sis>

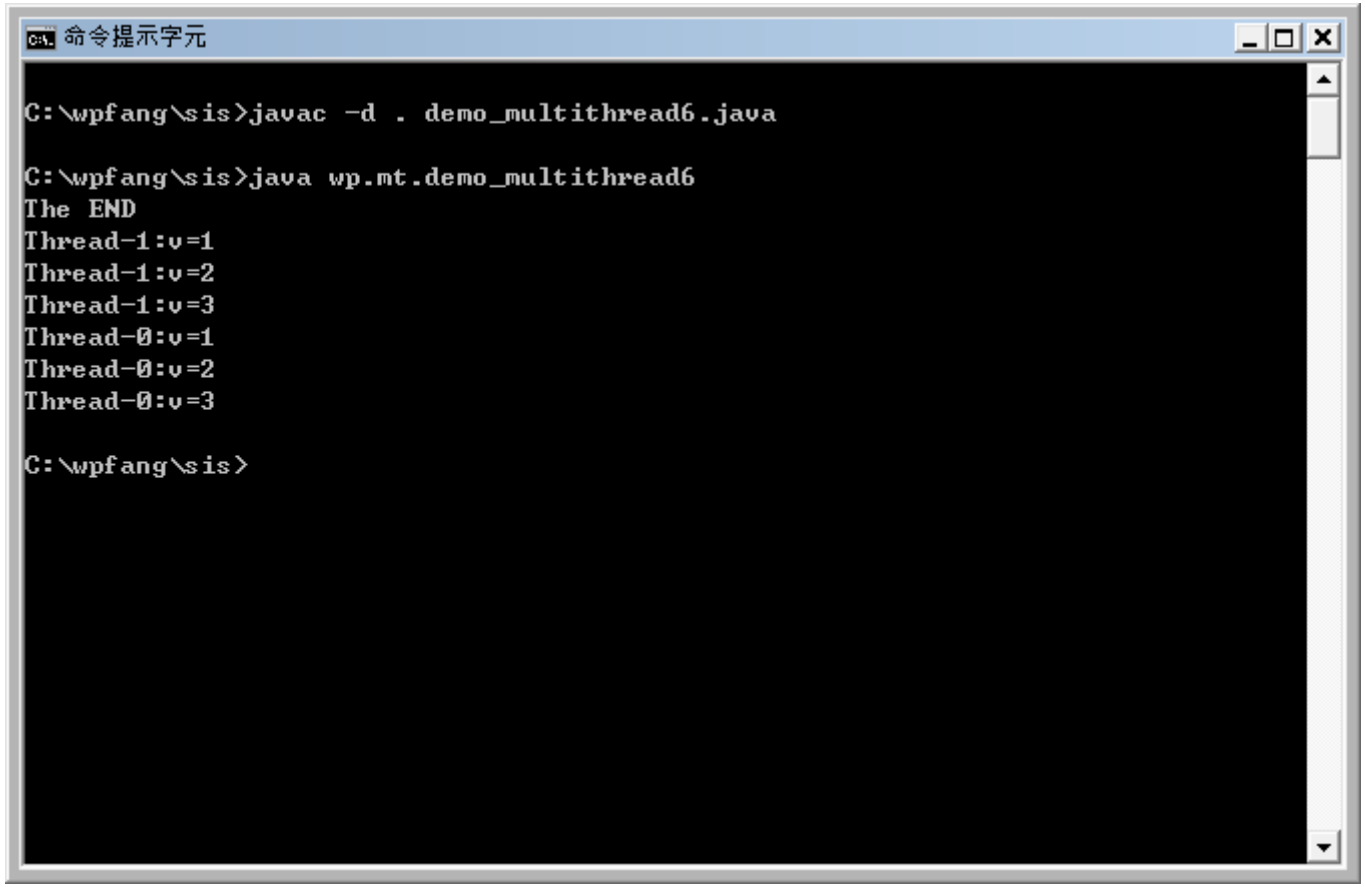
Share memory

```
package wp.mt;
import java.lang.Thread;

class MyThread implements Runnable
{
   private int v;
   public void run()
   {
     for(int i=0;i<3;i++)
      {
           v++;
          System.out.println(Thread.currentThread().getName()+":v="+v);
      }

   }

}
public class demo_multithread6
{
    public static void main(String [] args)
    {
                        MyThread ta=new MyThread();
                        MyThread tb=new MyThread();
      (new Thread(ta)).start();
      (new Thread(tb)).start();
     System.out.println("The END");
    }
}
```


命令提示字元
C:\wpfang\sis>javac -d . demo_multithread6.java
C:\wpfang\sis>java wp.mt.demo_multithread6
The END
Thread-1:v=1
Thread-1:v=2
Thread-1:v=3
Thread-0:v=1
Thread-0:v=2
Thread-0:v=3
C:\wpfang\sis>

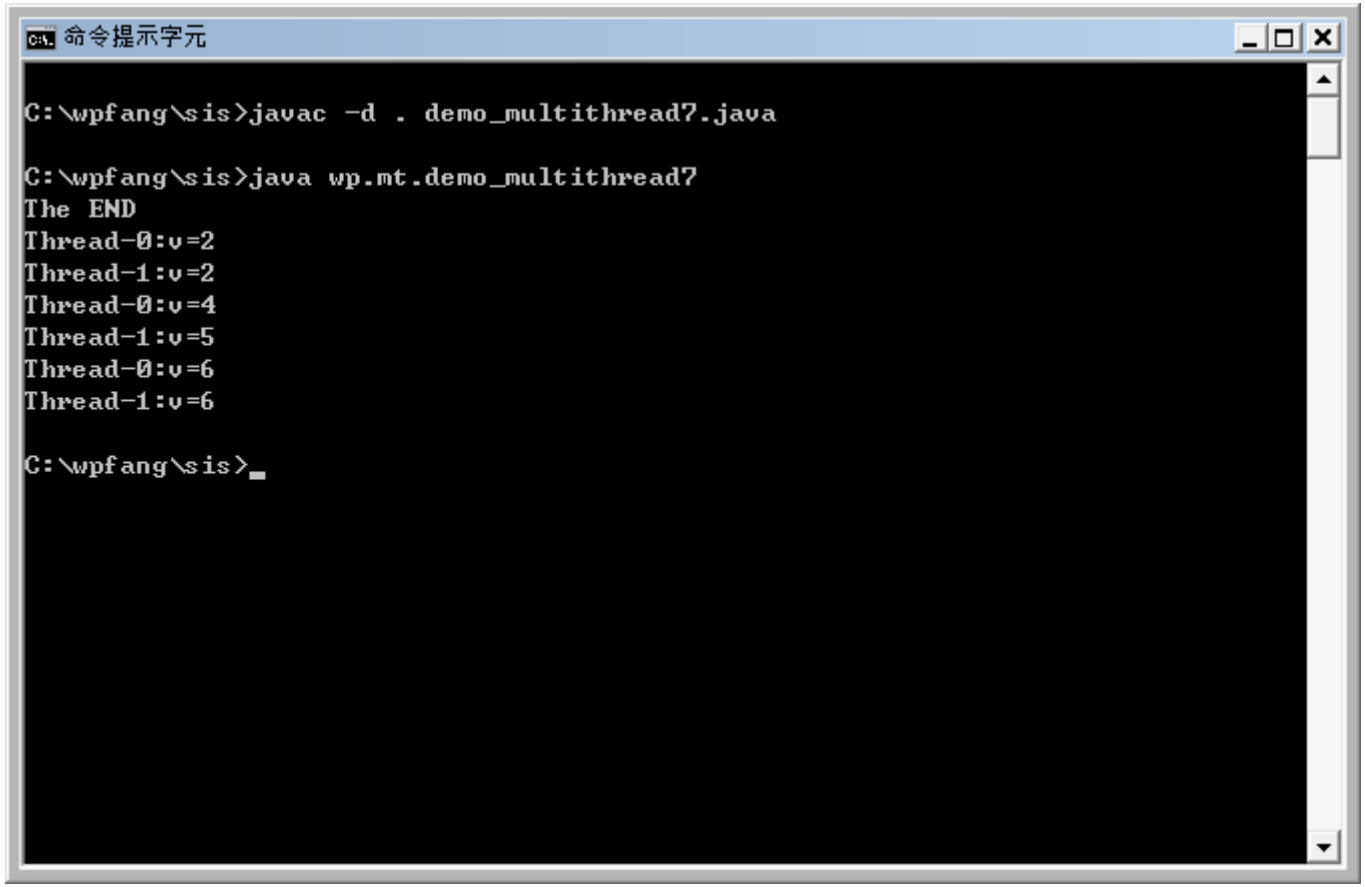
Thread.sleep

```
package wp.mt;
import java.lang.Thread;

class MyThread implements Runnable
{
   private int v;
   public void run()
   {
     for(int i=0;i<3;i++)
      {
          v++;
          try
          {
             Thread.sleep(1000);
          }
          catch(InterruptedException e){}
         System.out.println(Thread.currentThread().getName()+":v="+v);
      }

   }

}
public class demo_multithread7
{
    public static void main(String [] args)
   {
                              MyThread t=new MyThread();
      (new Thread(t)).start();
      (new Thread(t)).start();
     System.out.println("The END");
    }
}
```


命令提示字元
C:\wpfang\sis>javac -d . demo_multithread7.java
C:\wpfang\sis>java wp.mt.demo_multithread7
The END
Thread-0:v=2
Thread-1:v=2
Thread-0:v=4
Thread-1:v=5
Thread-0:v=6
Thread-1:v=6
C:\wpfang\sis>

synchronized(this)

```
package wp.mt;
import java.lang.Thread;

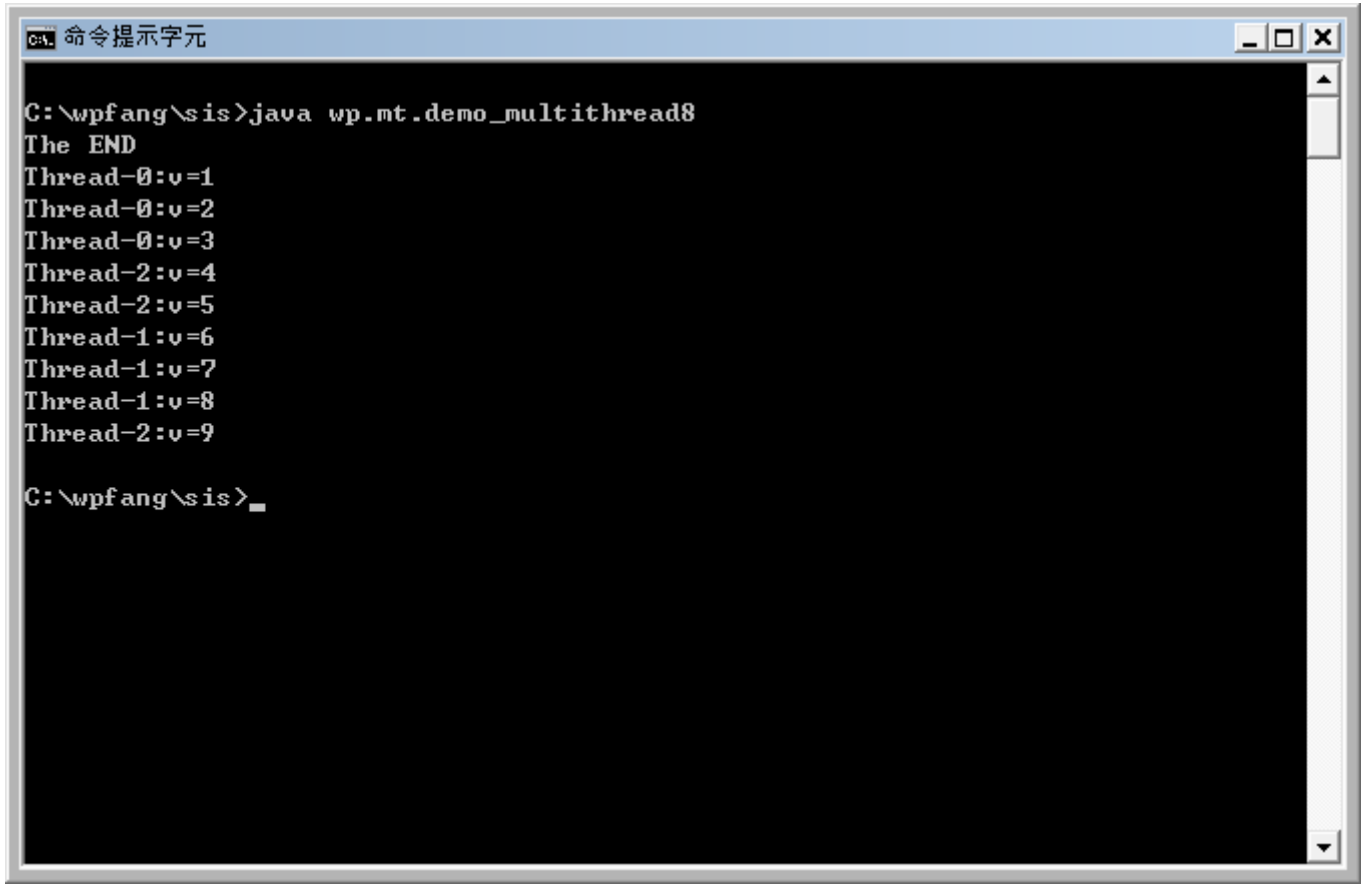
class MyThread implements Runnable
{
   private int v;
   public void run()
   {
     for(int i=0;i<3;i++)
     {
        synchronized(this)
        {
          v++;
          try
          {
             Thread.sleep(1000);
          }
          catch(InterruptedException e){}

System.out.println(Thread.currentThread().getName()+":v="+v);
        }
     }

   }

}
```

```
public class demo_multithread8
{
    public static void main(String []
args)
    {
            MyThread t=new
MyThread();
      (new Thread(t)).start();
      (new Thread(t)).start();
      (new Thread(t)).start();
     System.out.println("The
END");
    }
}
```


命令提示字元
C:\wpfang\sis>java wp.mt.demo_multithread8
The END
Thread-0:v=1
Thread-0:v=2
Thread-0:v=3
Thread-2:v=4
Thread-2:v=5
Thread-1:v=6
Thread-1:v=7
Thread-1:v=8
Thread-2:v=9
C:\wpfang\sis>

synchronized

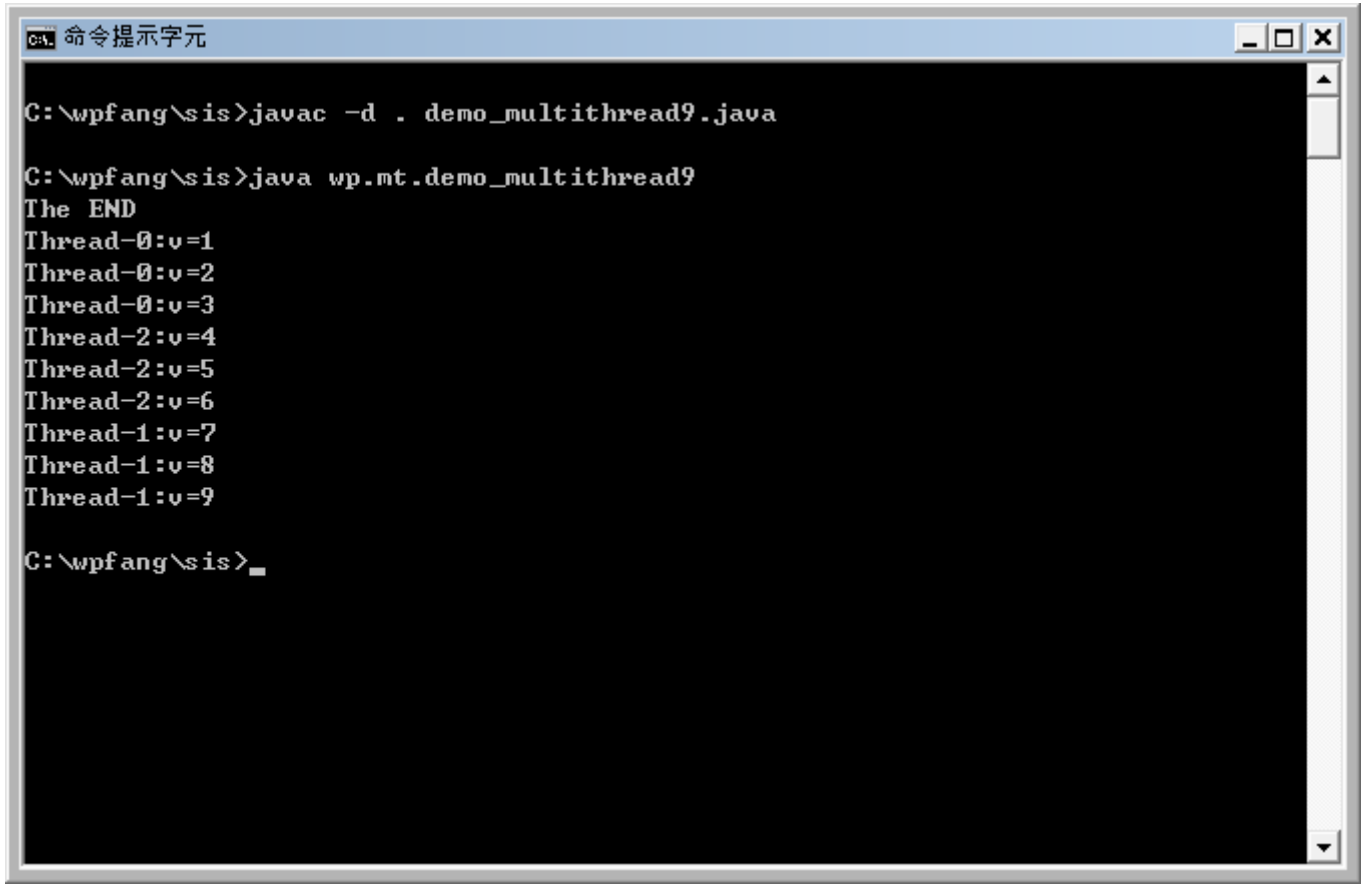
```
package wp.mt;
import java.lang.Thread;

class MyThread implements Runnable
{
   private int v;
   public void run()
   {
         go();
   }
   synchronized void go()
   {
     for(int i=0;i<3;i++)
      {
       // synchronized(this)
        {
          v++;
          try
          {
            Thread.sleep(1000);
          }
          catch(InterruptedException e){}
         System.out.println(Thread.currentThread().getName()+":v="+v);
       }
      }

   }

}
}
```

```
public class demo_multithread9
{
    public static void main(String [] args)
    {
        MyThread t=new MyThread();
        (new Thread(t)).start();
        (new Thread(t)).start();
        (new Thread(t)).start();
      System.out.println("The END");
    }
}
```


命令提示字元
C:\wpfang\sis>javac -d . demo_multithread9.java
C:\wpfang\sis>java wp.mt.demo_multithread9
The END
Thread-0:v=1
Thread-0:v=2
Thread-0:v=3
Thread-2:v=4
Thread-2:v=5
Thread-2:v=6
Thread-1:v=7
Thread-1:v=8
Thread-1:v=9
C:\wpfang\sis>

DaemonThread

```
import java.io.*;
public class RunDaemonThread {
    public static void main(String[] args) {
        UserThread u = new UserThread();
        (new Thread(u)).start();
    }
}

class UserThread implements Runnable{

    UserThread(){

        DaemonThread d = new DaemonThread();

        Thread t = new Thread(d);

        t.setDaemon(true);

        t.start();

    }

    public void run(){

        for(int i=0; i<3; i++){

            try{

                Thread.sleep(1000);

            } catch (InterruptedException e){}

            System.out.println("User執行緒執行了 " + i + " 次");

        }

    }

}

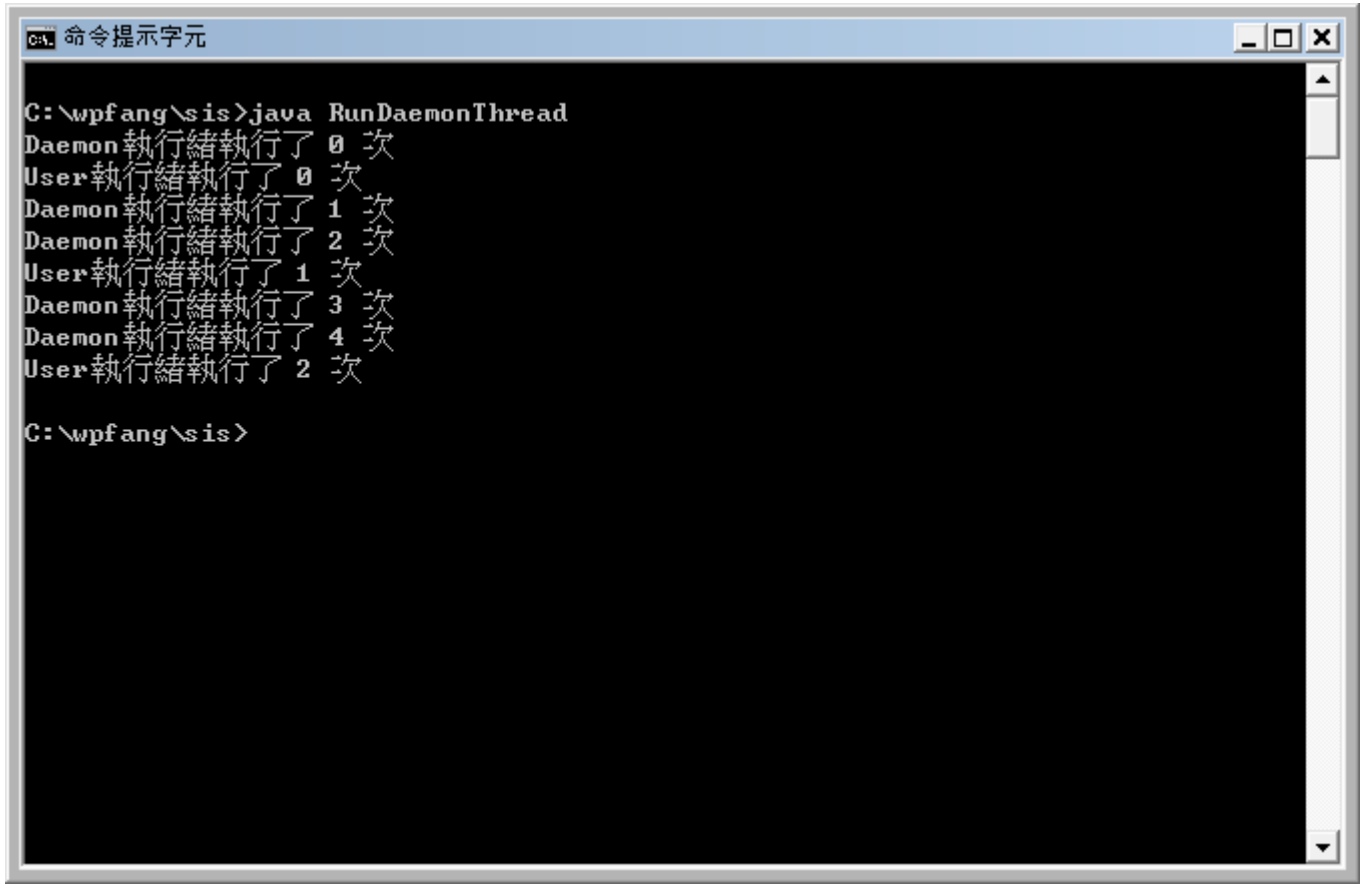
=
```

```
class DaemonThread implements Runnable{

    public void run(){
        int i=0;
        while(true){
            try{
                Thread.sleep(500);
            } catch (InterruptedException e){}
 System.out.println("Daemon執行緒執行了 " + i + " 次");
            i++;
        }
    }
}

```

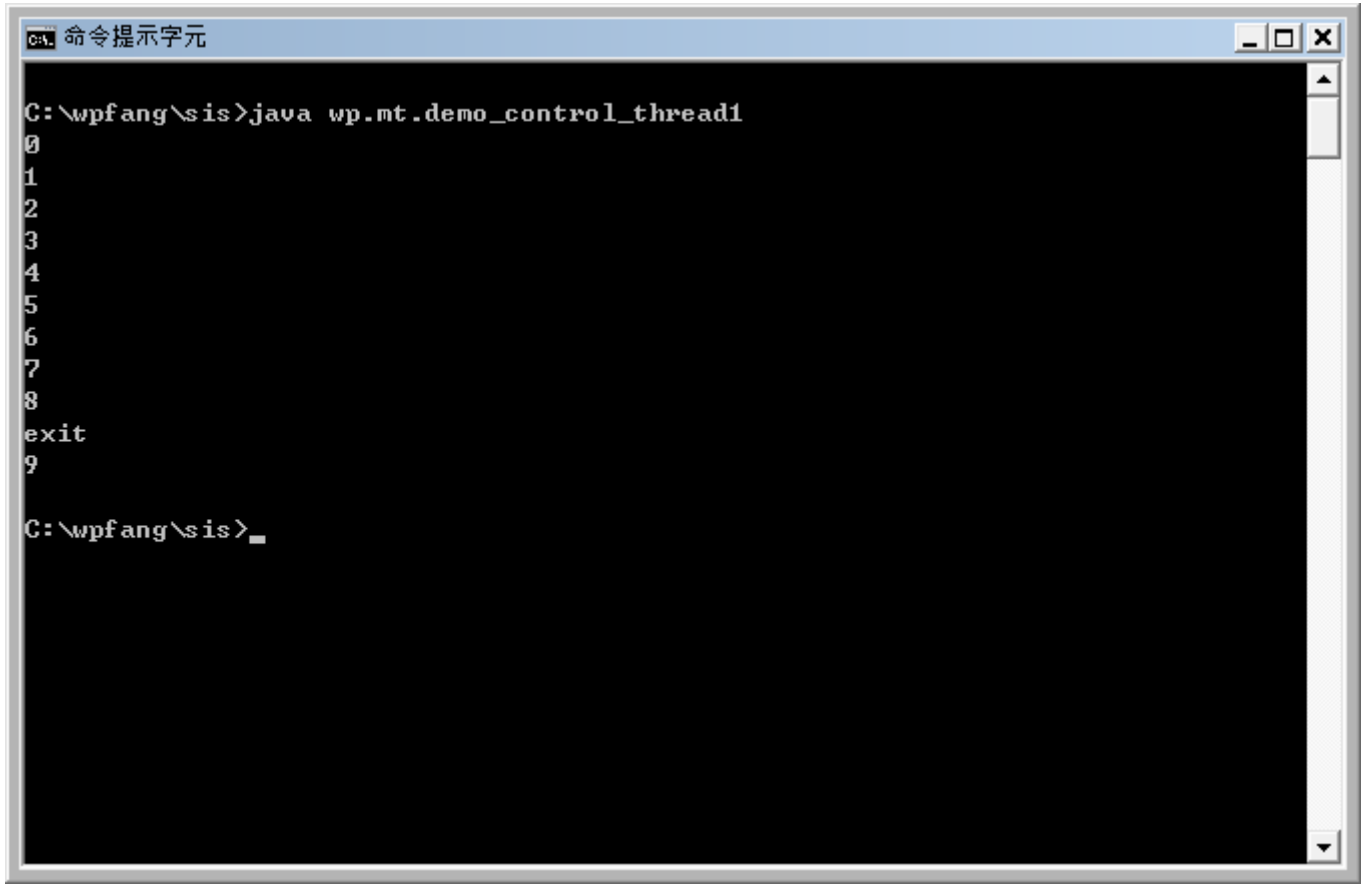
DaemonThread



Stop thread

```
package wp.mt;
class CounterThread extends Thread
{
  public boolean stopped = false;
  int count = 0;
  public void run()
 {
    while (!stopped)
   {
     try
    {
      sleep(1000);
     }
     catch (InterruptedException e) {     }
     System.out.println(count++);
   }
  }
}
```

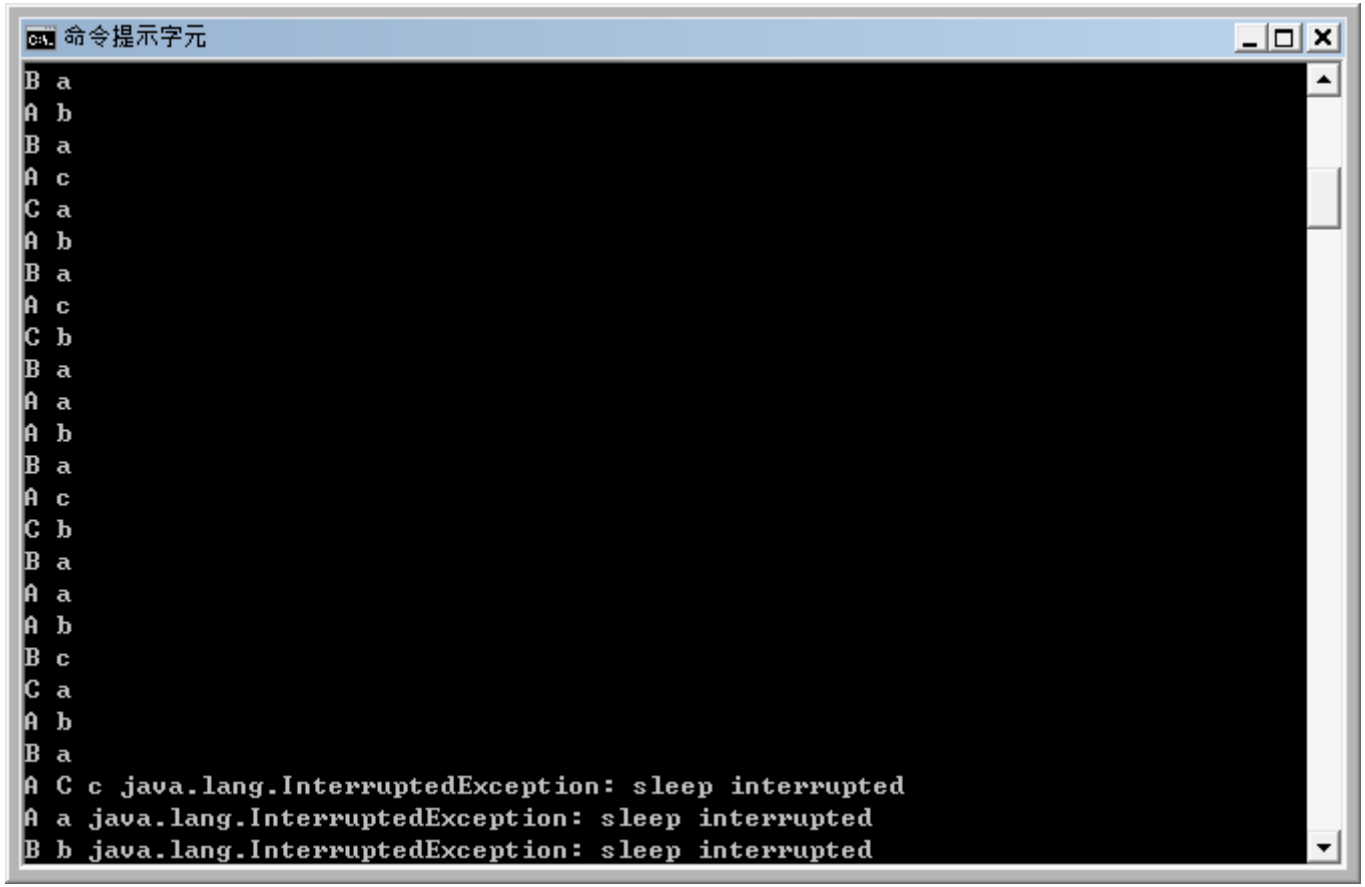
```
public class demo_control_thread1
 {
  public static void main(String[] args)
 {
    CounterThread thread = new
CounterThread();
    thread.start();
    try
  {
     Thread.sleep(10000);
    }
   catch (InterruptedException e) {   }
    thread.stopped = true;
    System.out.println("exit");
  }
}
```


命令提示字元
C:\wpfang\sis>java wp.mt.demo_control_thread1
0
1
2
3
4
5
6
7
8
exit
9
C:\wpfang\sis>

interrupt()

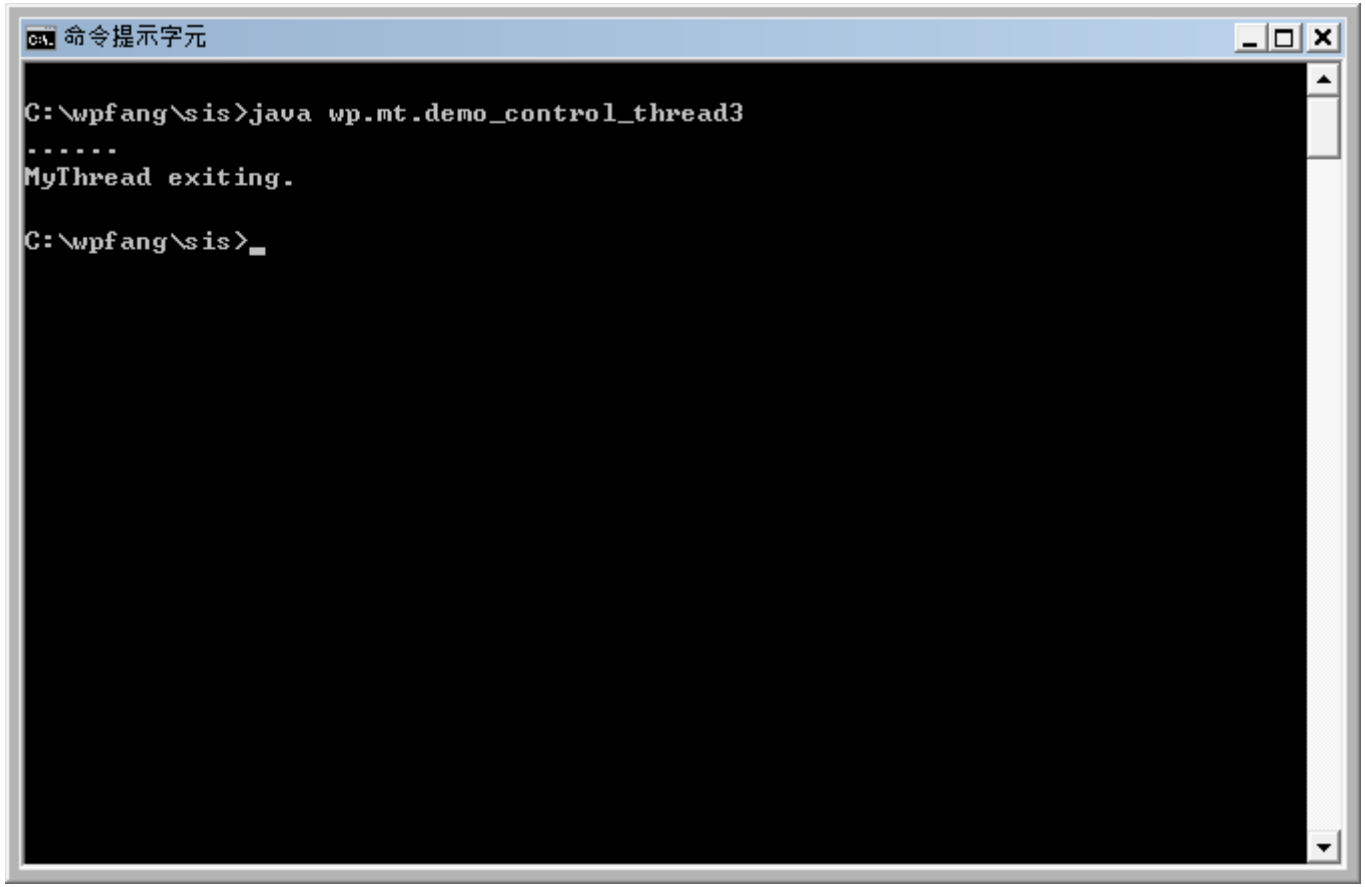
```
package wp.mt;
class TryThread extends Thread
{
 public TryThread(String firstName, String secondName, long delay)
{
   this.firstName = firstName;
   this.secondName = secondName;
   aWhile = delay;
   setDaemon(true);
 }
 public void run()
{
   try
   {
     while (true)
    {
      System.out.print(firstName);
      sleep(aWhile);
      System.out.print(secondName + "\n");
     }
    }
    catch (InterruptedException e)
   {
     System.out.println(firstName + secondName + e);
    }
 }
 private String firstName;
 private String secondName;
 private long aWhile;
}
```

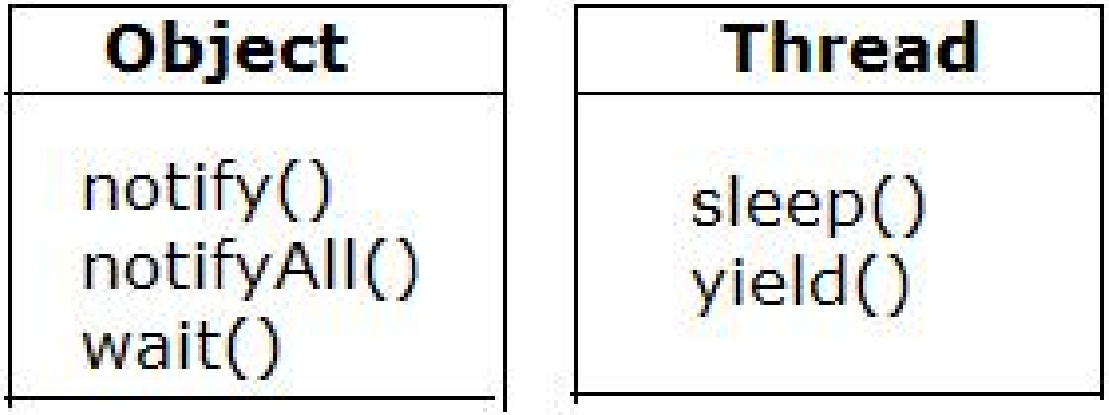
```
public class demo_control_thread2
{
  public static void main(String[] args)
{
   Thread first = new TryThread("A ", "a ", 200L);
   Thread second = new TryThread("B ", "b ",300L);
   Thread third = new TryThread("C ", "c ", 500L);
   first.start();
   second.start();
   third.start();
   try
   {
     Thread.sleep(3000);
     first.interrupt();
     second.interrupt();
     third.interrupt();
   }
   catch (Exception e)
   {
     System.out.println(e);
   }
  }
}
```


命令提示字元
B a
A b
B a
A c
C a
A b
B a
A c
C b
B a
A a
A b
B a
A c
C b
B a
A a
A b
B c
C a
A b
B a
A C c java.lang.InterruptedException: sleep interrupted
A a java.lang.InterruptedException: sleep interrupted
B b java.lang.InterruptedException: sleep interrupted

Suspend resume

```
package wp.mt;
class MyThread implements Runnable {
 Thread thrd;
 boolean suspended;
 boolean stopped;
 MyThread(String name) {
  thrd = new Thread(this, name);
  suspended = false;
  stopped = false;
  thrd.start();
 }
 public void run() {
  try {
   for (int i = 1; i < 10; i++) {
    System.out.print(".");
    Thread.sleep(50);
    synchronized (this) {
     while (suspended)
      wait();
     if (stopped)
      break;
    }
   }
  } catch (InterruptedException exc) {
   System.out.println(thrd.getName() + " interrupted.");
  }
  System.out.println("\n" + thrd.getName() + " exiting.");
 }
 synchronized void stop() {
  stopped = true;
  suspended = false;
  notify();
 }
 synchronized void suspend() {
  suspended = true;
 }
 synchronized void resume() {
  suspended = false;
  notify();
 }
}
public class demo_control_thread3 {
 public static void main(String args[]) throws Exception {
  MyThread mt = new MyThread("MyThread");
  Thread.sleep(100);
  mt.suspend();
  Thread.sleep(100);
  mt.resume();
  Thread.sleep(100);
  mt.suspend();
  Thread.sleep(100);
  mt.resume();
  Thread.sleep(100);
  mt.stop();
 }
}
```


命令提示字元
C:\wpfang\sis>java wp.mt.demo_control_thread3
.....
MyThread exiting.
C:\wpfang\sis>


Object
notify()
notifyAll()
wait()
Thread
sleep()
yield()

```
package wp.mt;
public class anonymous_class_thread
{
    public static void main(String [] args)
    {
        ( new Thread() {
            public void run() {  for(;;) System.out.println("thread running");    }
        }).start();
    }
}
```

Synchronized in Java

- synchronized methods
- synchronized blocks

- Synchronized methods are methods that are used to control access to an object.
- A thread only executes a synchronized method after it has acquired the lock for the method's object or class.

synchronized methods

```
package wp.mt;
public class SyncMethodsExample extends Thread {

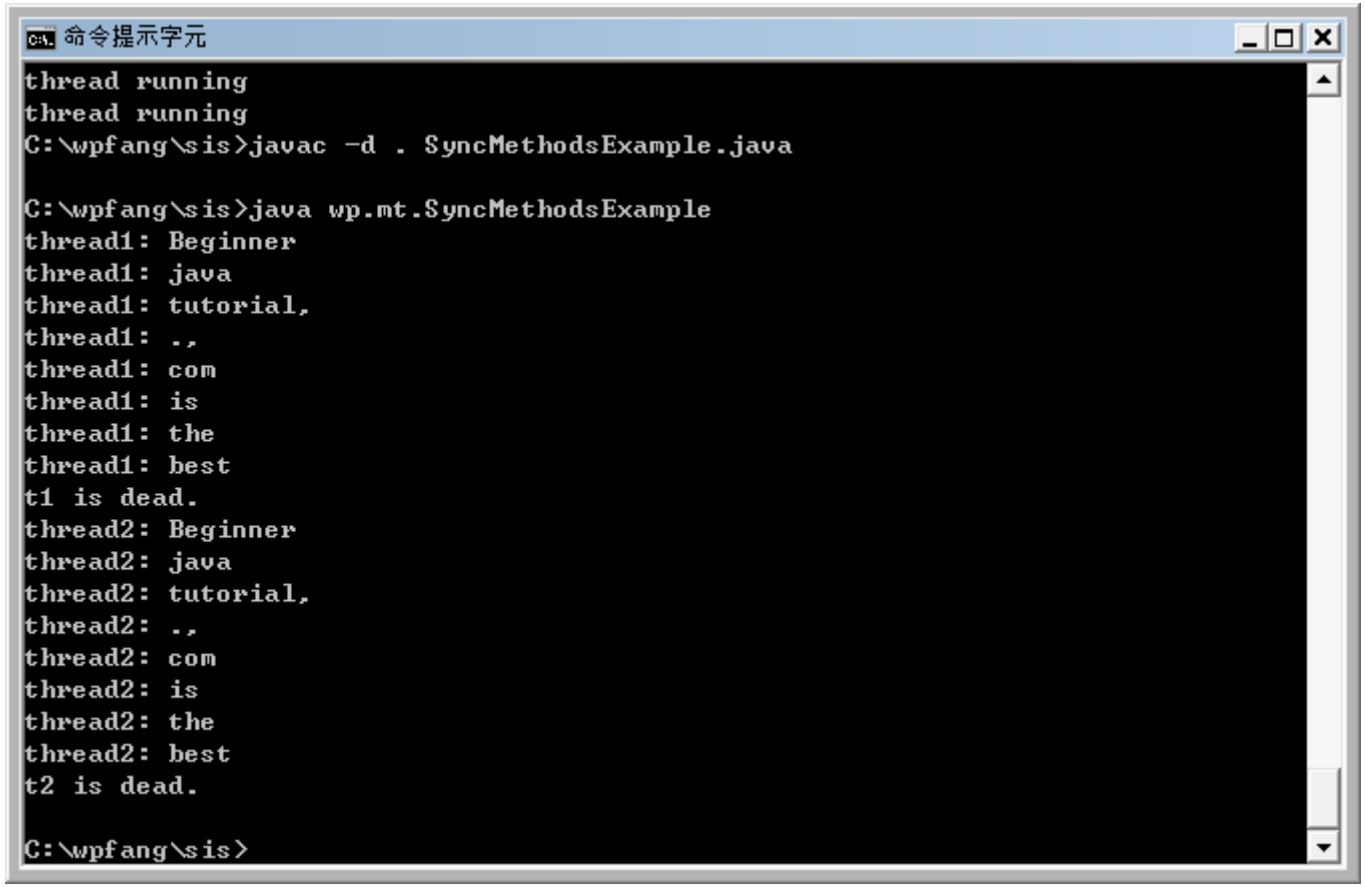
        static String[] msg = { "Beginner", "java", "tutorial,", ".,", "com",
                                        "is", "the", "best" };
        public SyncMethodsExample(String id) {
                super(id);
        }
        public static void main(String[] args) {
                SyncMethodsExample thread1 = new SyncMethodsExample("thread1: ");
                SyncMethodsExample thread2 = new SyncMethodsExample("thread2: ");
                thread1.start();
                thread2.start();
                boolean t1IsAlive = true;
                boolean t2IsAlive = true;
                do {
                        if (t1IsAlive && !thread1.isAlive()) {
                                t1IsAlive = false;
                                System.out.println("t1 is dead.");
                        }
                        if (t2IsAlive && !thread2.isAlive()) {
                                t2IsAlive = false;
                                System.out.println("t2 is dead.");
                        }
                } while (t1IsAlive || t2IsAlive);
        }
        void randomWait() {
                try {
                        Thread.currentThread().sleep((long) (3000 *

Math.random()));
                } catch (InterruptedException e) {
                        System.out.println("Interrupted!");
                }
        }
        public synchronized void run() {
                SynchronizedOutput.displayList(getName(), msg);
        }
}
```

synchronized methods

```
class SynchronizedOutput {
        public static synchronized void displayList(String name, String list[]) {
                for (int i = 0; i < list.length; i++) {
                        SyncMethodsExample t =

(SyncMethodsExample) Thread
                                                .currentThread();
                        t.randomWait();
                        System.out.println(name + list[i]);
                }
        }
}
```


命令提示字元
thread running
thread running
C:\wpfang\sis>javac -d . SyncMethodsExample.java
C:\wpfang\sis>java wp.mt.SyncMethodsExample
thread1: Beginner
thread1: java
thread1: tutorial,
thread1: .,
thread1: com
thread1: is
thread1: the
thread1: best
t1 is dead.
thread2: Beginner
thread2: java
thread2: tutorial,
thread2: .,
thread2: com
thread2: is
thread2: the
thread2: best
t2 is dead.
C:\wpfang\sis>

Synchronized blocks

- Once a thread has entered the code block after acquiring the lock on the specified object, no other thread will be able to execute the code block, or any other code requiring the same object lock, until the lock is relinquished.

```
package wp.mt;
public class SyncBlockExample extends Thread {

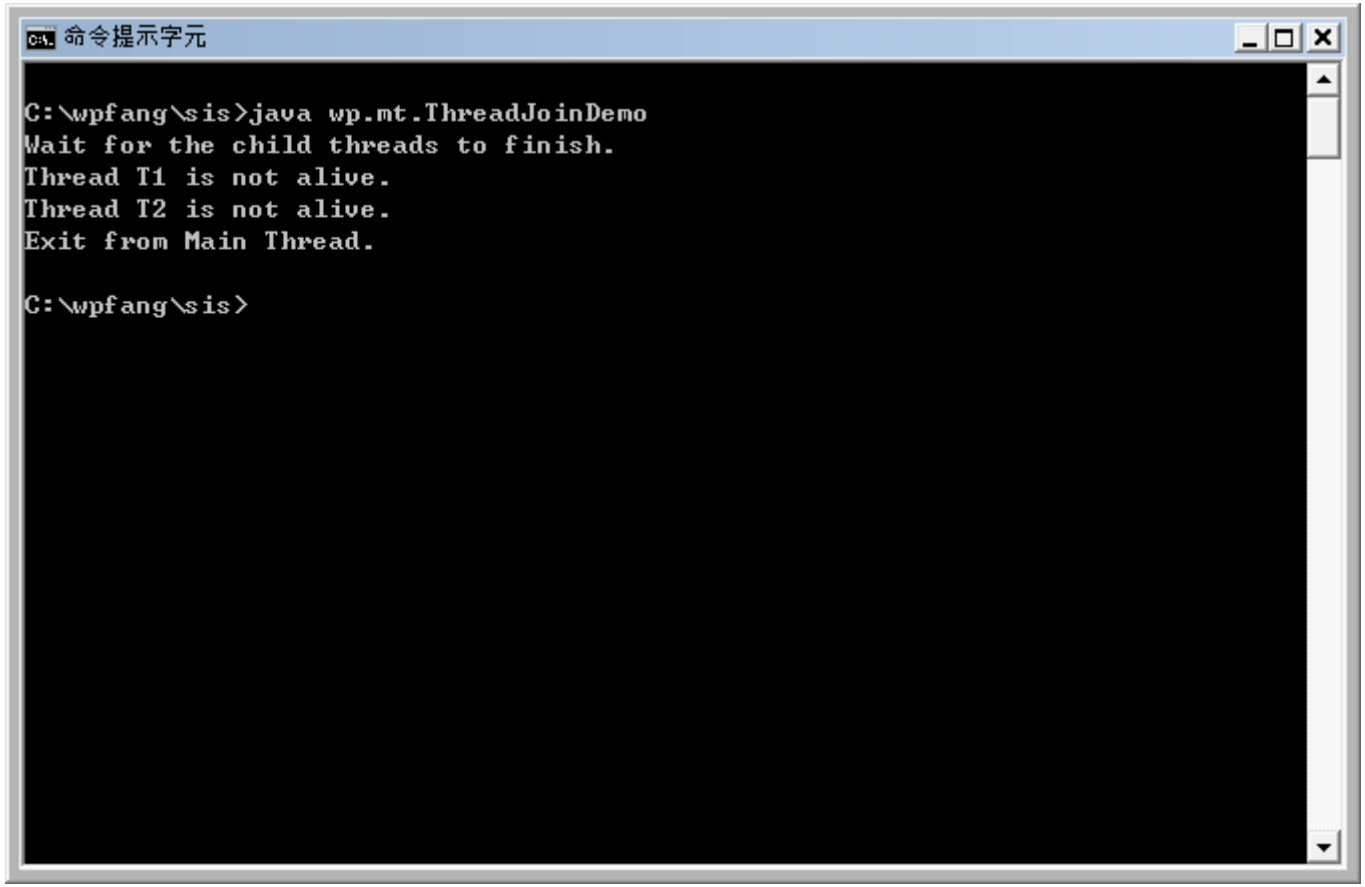
        static String[] msg = { "Beginner", "java", "tutorial,", ".,", "com“,"is", "the", "best" };
        public SyncBlockExample(String id) {
                super(id);
        }
        public static void main(String[] args) {
                SyncBlockExample thread1 = new SyncBlockExample("thread1: ");
                SyncBlockExample thread2 = new SyncBlockExample("thread2: ");
                thread1.start();
                thread2.start();
                boolean t1IsAlive = true;
                boolean t2IsAlive = true;
                do {
                        if (t1IsAlive && !thread1.isAlive()) {
                                t1IsAlive = false;
                                System.out.println("t1 is dead.");
                        }
                        if (t2IsAlive && !thread2.isAlive()) {
                                t2IsAlive = false;
                                System.out.println("t2 is dead.");
                        }
                } while (t1IsAlive || t2IsAlive);
        }
        void randomWait() {
                try {
                        Thread.currentThread().sleep((long) (3000 * Math.random()));
                } catch (InterruptedException e) {
                        System.out.println("Interrupted!");
                }
        }
        public void run() {
                synchronized (System.out) {
                        for (int i = 0; i < msg.length; i++) {
                                randomWait();
                                System.out.println(getName() + msg

[i]);
                        }
                }
        }
}
```

Join

```
package wp.mt;
public class ThreadJoinDemo {

		public static void main(String[] args) {
				Thread t1 = new Thread("T1");
				Thread t2 = new Thread("T2");
				try {
						System.out.println("Wait for the child threads to finish.");
						t1.join();
						if (!t1.isAlive())
								System.out.println("Thread T1 is not alive.");
						t2.join();
						if (!t2.isAlive())
								System.out.println("Thread T2 is not alive.");
				} catch (InterruptedException e) {
						System.out.println("Main Thread interrupted.");
				}
				System.out.println("Exit from Main Thread.");
		}
}
```


命令提示字元
C:\wpfang\sis>java wp.mt.ThreadJoinDemo
Wait for the child threads to finish.
Thread T1 is not alive.
Thread T2 is not alive.
Exit from Main Thread.
C:\wpfang\sis>

DATA STRUCTURE

Data Structure

- In computer science, a data structure is a data organization, management and storage format that enables efficient access and modification.

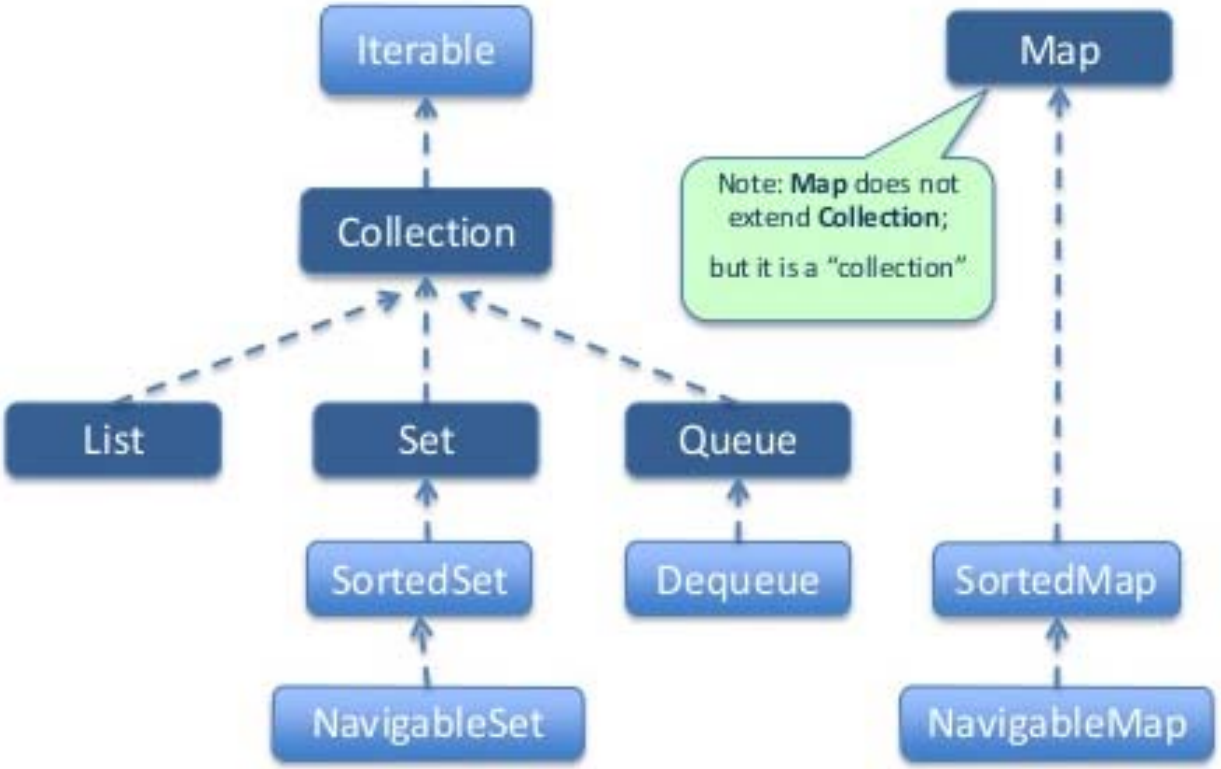
Data Structure

- Data types
  - Primitive types
  - Composite types or non-primitive type
  - Abstract data types
- Linear data structures
  - Arrays
  - Lists
- Trees
  - Binary trees
  - B-trees
  - Heaps
  - Tries
  - Multiway trees
  - Space-partitioning trees
  - Application-specific trees
- Hashes
- Graphs

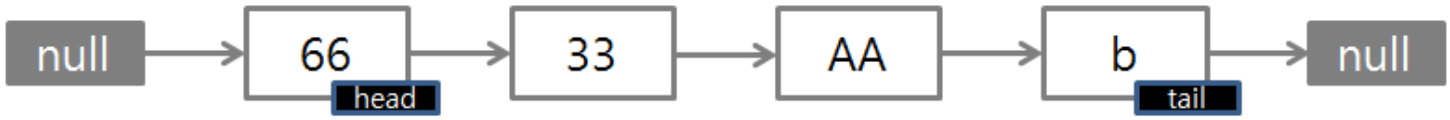
Data Structure

- Programming language
- Data structure
- Algorithm
- The Art of Programming

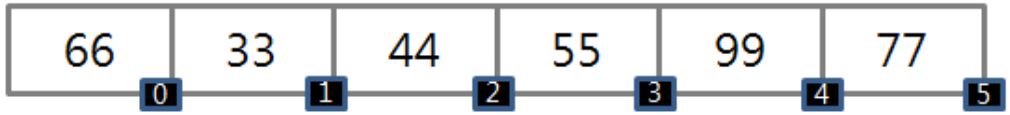
Data Structure



Linked list



Array List



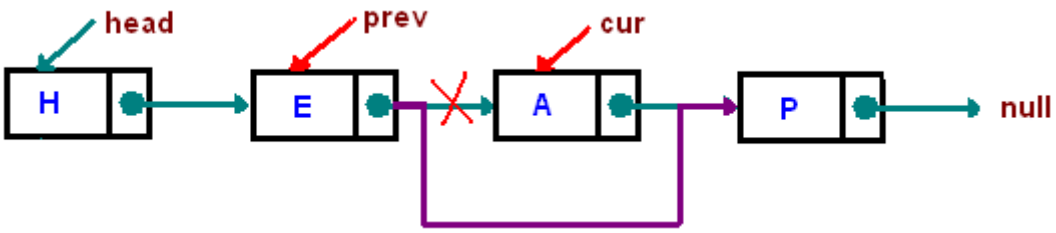
Linked-list

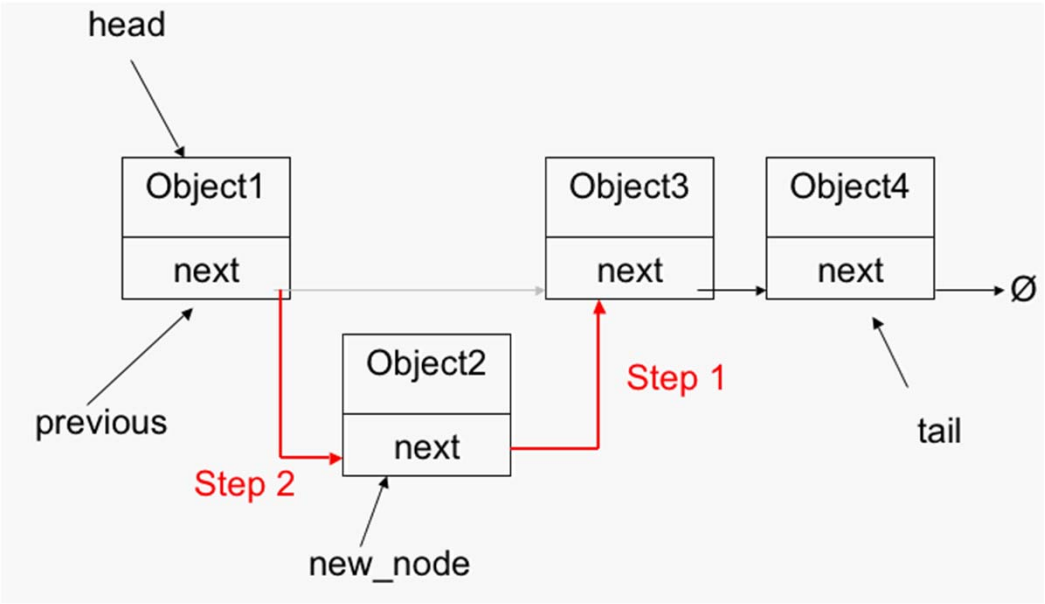
```
package demo_datastructure;
public class dolinkedlist {
public static void main(String[] args) {
Node n1=new Node(1);
Node n2=new Node(2);
Node n3=new Node(3);
Node n4=new Node(4);
Node n5=new Node(5);
Node n6=new Node(6);
n1.pNext=n2;
n2.pNext=n3;
n3.pNext=n4;
n4.pNext=n5;
n5.pNext=n6;

Node p=n1;
while(p!=null)
{
p.show();
p=p.pNext;
}
}
```

```
class Node
{
Node pNext;
int data;
Node(int d){
data=d;
pNext=null;
}
void show()
{
System.out.print("["+data+"
]");
}
}
```

Data Structure





Data Structure

- //----- insert ----
- p=**new Node(99);**
- p.pNext=n5.pNext;
- n5.pNext=p;

- p=n1;
- **while(p!=null)**
- {
- p.show();
- p=p.pNext;
- }

- //----- insert ----
- p=**new Node(99);**
- p.pNext=n5.pNext;
- n5.pNext=p;

- p=n1;
- **while(p!=null)**
- {
- p.show();
- p=p.pNext;
- }

Combination

- ABCD
  - A,B,C,D
  - AB,AC,AD,BC,BD,CD
  - ABC,ABD,ACD,BCD
  - ABCD

- ABCD
  - 0000
  - 0001 D
  - 0010 C
  - 0011 CD
  - 0100 B
  - 0101 B D
  - 0111 BCD
  - 1000 A
  - 1001 A D
  - 1010 A C
  - 1011 A CD
  - 1100 AB
  - 1101 AB D
  - 1111 ABCD

```
public class combination {
public static void main(String[] args) {
// TODO Auto-generated method stub
int i;
int j;
char []A={'A','B','C','D'};
for(i=0;i<16;i++)
{
for(j=0;j<4;j++)
if(((i>>j) & 1) ==1)
System.out.print(A[j]);
System.out.println();
}
}

}
```

Data Structure

- Enumeration
- BitSet
- Vector
- Stack
- Dictionary
- Hashtable
- Properties

Enumeration

```
import java.util.Vector;
import java.util.Enumeration;
public class EnumerationTester {
public static void main(String args[]) {
                Enumeration days;
                Vector dayNames = new Vector();
                dayNames.add("Sunday");
                dayNames.add("Monday");
                dayNames.add("Tuesday");
                 dayNames.add("Wednesday");
                dayNames.add("Thursday");
                dayNames.add("Friday");
                dayNames.add("Saturday");
                days = dayNames.elements();
                while (days.hasMoreElements()) {
                                System.out.println(days.nextElement());
                 }
   }
}
```

BitSet

```
import java.util.BitSet;
public class BitSetDemo {

  public static void main(String args[]) {
     BitSet bits1 = new BitSet(16);
     BitSet bits2 = new BitSet(16);
         // set some bits
     for(int i = 0; i < 16; i++) {
       if((i % 2) == 0) bits1.set(i);
       if((i % 5) != 0) bits2.set(i);
     }
      System.out.println("Initial pattern in bits1: ");
     System.out.println(bits1);
     System.out.println("\nInitial pattern in bits2: ");
     System.out.println(bits2);
     // AND bits
     bits2.and(bits1);
     System.out.println("\nbits2 AND bits1: ");
     System.out.println(bits2);
     // OR bits
     bits2.or(bits1);
     System.out.println("\nbits2 OR bits1: ");
     System.out.println(bits2);
     // XOR bits
     bits2.xor(bits1);
     System.out.println("\nbits2 XOR bits1: ");
     System.out.println(bits2);
   }
}
```

Vector

```
import java.util.*;
public class VectorDemo {

   public static void main(String args[]) {
     // initial size is 3, increment is 2
     Vector v = new Vector(3, 2);
     System.out.println("Initial size: " + v.size());
     System.out.println("Initial capacity: " + v.capacity());

     v.addElement(new Integer(1));
     v.addElement(new Integer(2));
     v.addElement(new Integer(3));
     v.addElement(new Integer(4));
     System.out.println("Capacity after four additions: " +
v.capacity());

     v.addElement(new Double(5.45));
     System.out.println("Current capacity: " + v.capacity());

     v.addElement(new Double(6.08));
     v.addElement(new Integer(7));
     System.out.println("Current capacity: " + v.capacity());
     v.addElement(new Float(9.4));
     v.addElement(new Integer(10));
     System.out.println("Current capacity: " + v.capacity());

     v.addElement(new Integer(11));
     v.addElement(new Integer(12));
     System.out.println("First element: " +
(Integer)v.firstElement());
     System.out.println("Last element: " +
(Integer)v.lastElement());

     if(v.contains(new Integer(3)))
       System.out.println("Vector contains 3.");

     // enumerate the elements in the vector.
     Enumeration vEnum = v.elements();
     System.out.println("\nElements in vector:");

     while(vEnum.hasMoreElements())
       System.out.print(vEnum.nextElement() + " ");
     System.out.println();
   }
}
```

Stack

```
import java.util.*;
public class StackDemo {

  static void showpush(Stack st, int a)
{
    st.push(new Integer(a));
    System.out.println("push(" + a +
")");
    System.out.println("stack: " + st);
  }

  static void showpop(Stack st) {
    System.out.print("pop -> ");
    Integer a = (Integer) st.pop();
    System.out.println(a);
    System.out.println("stack: " + st);
  }

  public static void main(String args[])
{
    Stack st = new Stack();
    System.out.println("stack: " + st);
    showpush(st, 42);
    showpush(st, 66);
    showpush(st, 99);
    showpop(st);
    showpop(st);
    showpop(st);
    try {
      showpop(st);
    }catch (EmptyStackException e) {
      System.out.println("empty
stack");
    }
  }
}
```

container

- Array
- ArrayList
- LinkedList
- HashMap
- set

ArrayList

```
import java.util.*;
public class ArrayListDemo {
		public static void main(String args[]) {
		Scanner scanner = new Scanner(System.in);
		List<String> list = new ArrayList<String>();
		System.out.println("輸入名稱(使用 quit 結束：)");
		while(true) {
				System.out.print("# ");
				String input = scanner.next();
				if(input.equals("quit")) break;
				list.add(input);
		}
		System.out.println("顯示輸入：");
		for(int i = 0 ; i < list.size() ; i++)
				System.out.print(list.get(i) + " ");
		System.out.println();
		for(String s : list)
				System.out.print(s + " ");
		System.out.println();
		Iterator iterator = list.iterator();
		while(iterator.hasNext())
				System.out.print(iterator.next() + " ");
		System.out.println();
}
}
```

Linked list

```
import java.util.LinkedList;
public class StringStack {
		private LinkedList<String> linkedList;
		public StringStack() {
				linkedList = new LinkedList<String>();
		}
		public void push(String name) {
				linkedList.addFirst(name);
		}
		public String top() {
				return linkedList.getFirst();
		}
		public String pop() {
				return linkedList.removeFirst();
		}
		public boolean isEmpty() {
				return linkedList.isEmpty();
		}
}
```

Map

```
import java.util.Map;
import java.util.HashMap;

public class HashMapDemo {
        public static void main(String args[]) {
        Map<String, String> map = new HashMap<String, String>();
        String key1 = "leon";
        String key2 = "godleon";
        map.put(key1, "leon 的資料");
        map.put(key2, "godleon 的資料");
        System.out.println(map.get(key1));
        System.out.println(map.get(key2));
}
}
```

HashMap

```
import java.util.Collection;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;

public class HashMapDemo2 {
public static void main(String args[]) {
    Map<String, String> map = new HashMap<String, String>();
    map.put("leon", "leon 的資料");
    map.put("godleon", "godleon 的資料");
   map.put("bill", "bill 的資料");
   // 最後由 Iterator 列出所有元素
   Collection collection = map.values();
   Iterator iterator = collection.iterator();
   while(iterator.hasNext())
            System.out.println(iterator.next());
     System.out.println();


    for(String s : map.values())
              System.out.println(s);
      System.out.println();
  }
}
```

TreeMap

- ```
  import java.util.Map;
  import java.util.TreeMap;

  public class TreeMapDemo {
  public static void main(String args[]) {
  Map<String, String> map = new TreeMap<String, String>();
  map.put("godleon", "godleon 的資料");
  map.put("leon", "leon 的資料");
  map.put("bill", "bill 的資料");

  for(String s : map.values())
  System.out.println(s);
  System.out.println();
  }
  }
  ```
  ```

Tree set

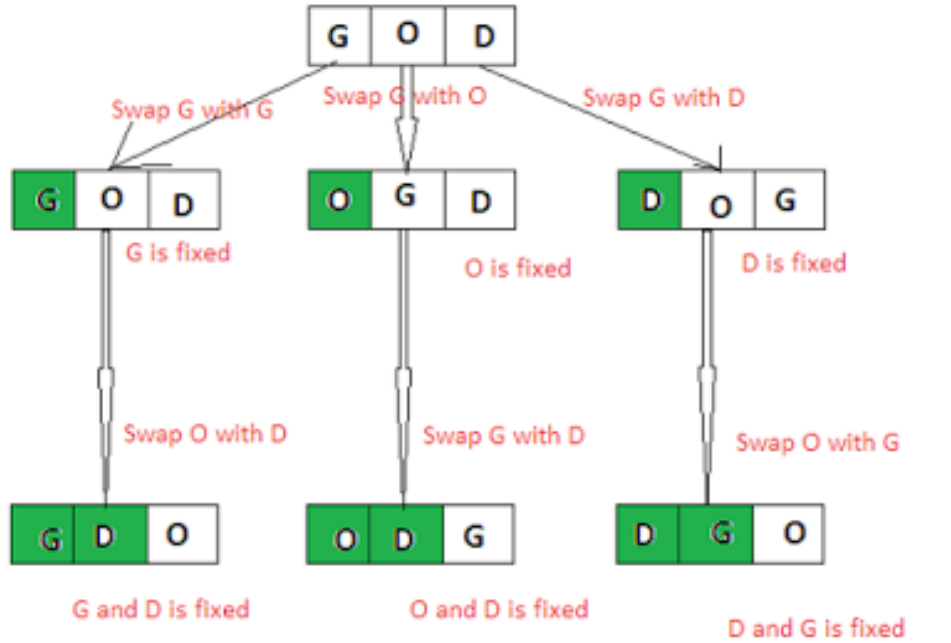
```
import java.util.Set;
import java.util.TreeSet;

public class TreeSetDemo {
public static void main(String args[]) {
Set<String> set = new TreeSet<String>();

set.add("godleon");
set.add("good");
set.add("bad");


for(String s : set)
        System.out.print(s + " ");
System.out.println();
}
}
```

Permutation



Now we will stop here as there can be no more swapping. All charecters are fixed except the last one.

Shows fixed elements

permutation

```
public class StringPermutations {
	public static void main(String args[]) {
		permutation("123");
	 }
	public static void permutation(String input){
		permutation("", input);
	 }
	private static void permutation(String perm, String word) {
		 if (word.isEmpty()) {
				 System.err.println(perm + word);
		}
		else {
				for (int i = 0; i &lt; word.length(); i++) {
					permutation(perm + word.charAt(i), word.substring(0, i)
						+ word.substring(i + 1, word.length()));
				}
		 }
	 }
}
```

REFLECTION

reflection

```
public class ClassDemo {
public static void main(String args[]) {
String name = "godleon";
Class stringClass = name.getClass();

System.out.println("類別名稱：" + stringClass.getName());
System.out.println("是否為介面：" + stringClass.isInterface());
System.out.println("是否為基本型態：" + stringClass.isPrimitive());
System.out.println("是否為陣列物件：" + stringClass.isArray());
System.out.println("父類別名稱：" + stringClass.getSuperclass().getName());
}
}
```

- Java 只有在真正要用到 class 的時候才會將其載入，而真正用到的時候是指以下情況：使用 class 生成 object 時
- 使用者指定要載入 class (利用 Class.forName() 或是 ClassLoader.loadClass())

動態載入類別

- 若要動態載入類別，可以使用 Class 類別中的 forName() method，此為 static method，因此可以直接透過 Class 類別使用；
- 若要取得類別的 instance，可使用 Class 類別中的 newInstance() method

動態載入類別

- forName()
- newInstance()

- Class c=Class.forName(str);
- List lst=(List) c.newInstance();

```
public class ClassForNameDemo {
public static void main(String[] args) {
try {
//forName的使用
Class c = Class.forName(args[0]);
System.out.println("getName : " + c.getName());
System.out.println("isInterface : " + c.isInterface());
System.out.println("isPrimitive : " + c.isPrimitive());
System.out.println("isArray : " + c.isArray());
System.out.println("SuperClass : " + c.getCanonicalName());

//newInstance的使用
List lst = (List) c.newInstance();
for(int i = 0 ; i < 10 ; i++)
lst.add("element" + i);
for(Object o : lst.toArray())
System.out.println(o);
}
catch(Exception e) {
e.printStackTrace();
}
}
}
```

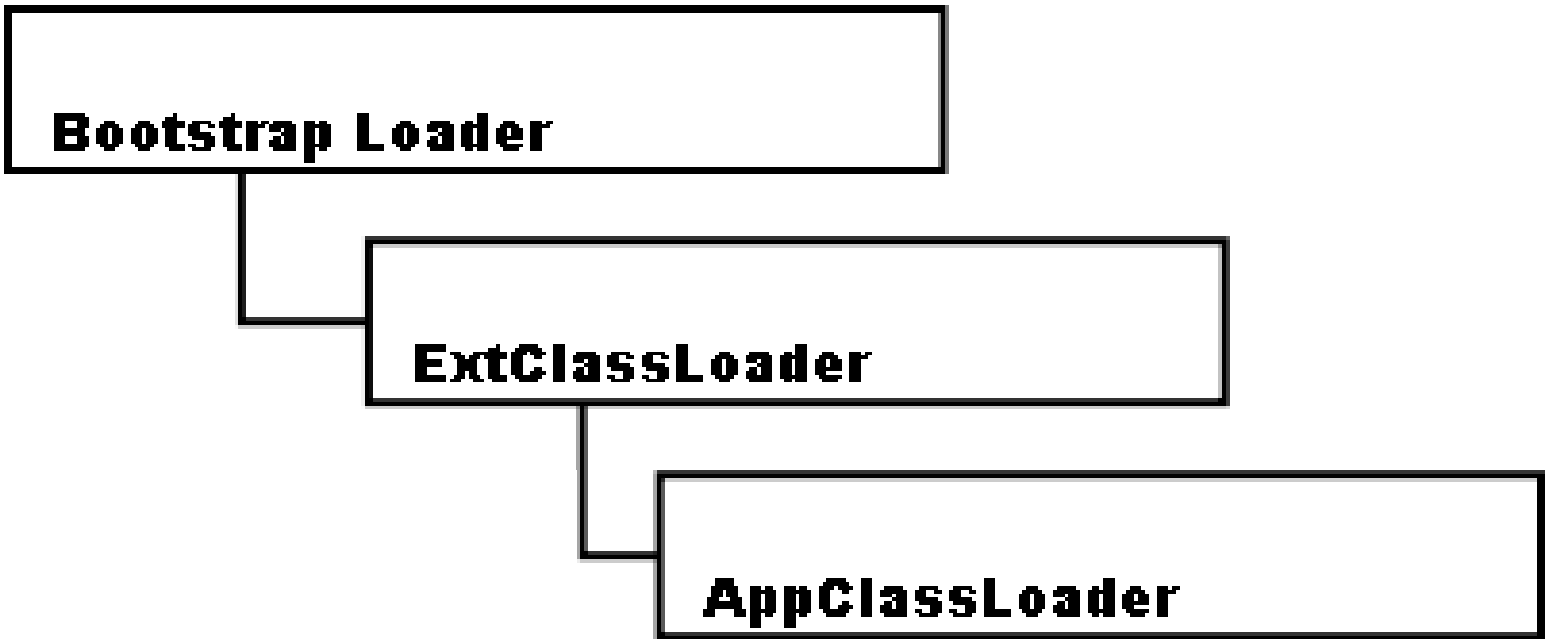
- getClassLoader()
- getConstructor()
- getFields()
- getMethods()
- getModifiers()
- getPackage()

- 當 Java 需要使用到 Class 時，才會將 Class 載入；而載入 Class 的工作是由 Class Loader(類別載入器) 所負責。

  每當 Java 程式啟動後，會有以下三種 Class Loader 進行載入 Class 的工作：
  Bootstrap Loader
  由 C++ 開發，會去搜尋 JRE 目錄($JAVA_HOME/jre/)中的 class 以及 lib 資料夾中的 *.jar 檔，檢查是否有指定的 class 需要載入。

- **ExtClassLoader**
  會去搜尋 JRE 目錄($JAVA_HOME/jre/)中的 lib/ext 資料夾，檢查其中的 class 與 *.jat 檔案，是否有指定的 class 需要載入。

- **AppClassLoader**
  搜尋 classpath 中是否有指定的 class 需要載入。


Bootstrap Loader
ExtClassLoader
AppClassLoader

Student.java

```
public class Student {
private String name;
private int score;
public Student() {
name = "N/A";
}
public Student(String name, int score) {
this.name = name;
this.score = score;
}
public void setName(String name) {
this.name = name;
}
public void setScore(int score) {
this.score = score;
}

public String getName() {
return name;
}
public int getScore() {
return score;
}

public void showData() {
System.out.println("Name : " + this.name);
System.out.println("Score : " + this.score);
}
}
```

```
newInstanceDemo.javaimport java.lang.reflect.Constructor;
import java.lang.reflect.InvocationTargetException;
import java.lang.reflect.Method;
public class newInstanceDemo {
public static void main(String[] args) {
Class c = null;
try {
c = Class.forName(args[0]);
/* (Begin)=============== 生成物件
===============(Begin) */

//指定Constructor所使用的參數型態
Class[] oParam = new Class[2];
oParam[0] = String.class;
oParam[1] = Integer.TYPE;

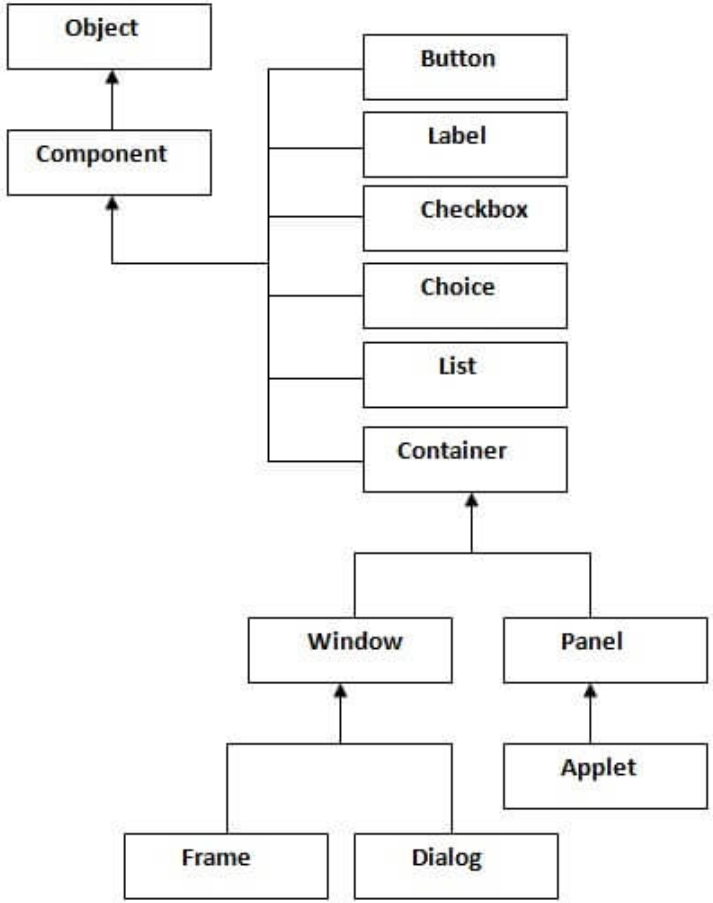
//產生Constructor
Constructor constructor = c.getConstructor(oParam);
//指定參數的內容
Object[] paramObjs = new Object[2];
paramObjs[0] = "godleon";
paramObjs[1] = new Integer(90);
//透過Constructor產生物件
Object obj = constructor.newInstance(paramObjs);
System.out.println(obj);
//指定Method所使用的參數類型
Class[] mParam1 = {String.class}; //只有一個參數
//產生Method(指定method名稱與參數)
Method setName = c.getMethod("setName", mParam1);
//指定參數內容
Object[] mParamObjs1 = {"godleon"};
//呼叫方法
setName.invoke(obj, mParamObjs1); //呼叫setName方法
//指定Method所使用的參數類型
Class[] mParam2 = {Integer.TYPE};
//產生Method(指定method名稱與參數)
Method setScore = c.getMethod("setScore", mParam2);
//指定參數內容
Object[] mParamObjs2 = {new Integer(90)};
//呼叫方法
setScore.invoke(obj, mParamObjs2); //呼叫setScore方法
//產生Method(指定method名稱與參數)
Method showData = c.getMethod("showData", null);
//呼叫方法
showData.invoke(obj, null); //呼叫showData方法
/* (End)=============== 呼叫方法 ===============(End) */
}
catch(Exception e) {
e.printStackTrace();
}
}
}
```

繪圖與常用介面

- Swing and awt

AWT 元件

- CheckboxGroup
- Component
  - Button
  - Canvas
  - Checkbox
  - Choice
  - Container
    - Panel
    - ScrollPane
    - Window
      - Dialog
        - » FileDialog
      - Frame
  - Label
  - List
  - Scrollbar
  - TextComponent
    - TextArea
    - TextField
- MenuComponent
  - MenuBar
  - MenuItem
    - Menu
      - PopupMenu
- MenuShortcut


Object
Component
Button
Label
Checkbox
Choice
List
Container
Window
Panel
Applet
Frame
Dialog

Show window

```
import java.awt.*;
class First2{
First2(){
Frame f=new Frame();
Button b=new Button("click me");
b.setBounds(30,50,80,30);
f.add(b);
f.setSize(300,300);
f.setLayout(null);
f.setVisible(true);
}
public static void main(String args[]){
First2 f=new First2();
}}
```

Show window with closing event

```
import java.awt.*;
import java.awt.event.*;
public class demoawt
{

   public static void main(String args[])
  {
        Frame f=new Frame();
        f.setSize(100,100);
        f.setLayout(null);

        f.addWindowListener(new WindowAdapter(){
                    public void  windowClosing(WindowEvent e)
                    {
System.exit(0);
                    }
        });
        f.setVisible(true);
  }
}
```

Window add component

```
import java.awt.*;
import java.awt.event.*;
class AEvent extends Frame implements ActionListener{
TextField tf;
AEvent(){
tf=new TextField();
tf.setBounds(60,50,170,20);
Button b=new Button("click me");
b.setBounds(100,120,80,30);
b.addActionListener(this);//passing current instance
add(b);add(tf);
setSize(300,300);
setLayout(null);
setVisible(true);
}
public void actionPerformed(ActionEvent e){
tf.setText("Welcome");
}
public static void main(String args[]){
new AEvent();
}
}
```

Using swing for convenient

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.JOptionPane;
public class demoawt
{

  public static void main(String args[])
 {
        Frame f=new Frame();
        f.setSize(100,100);
        f.setLayout(null);
        //--------------
        f.addWindowListener(new WindowAdapter(){
                public void  windowClosing(WindowEvent e)
                {
System.exit(0);
                }
        });
        //---------------
        Button b=new Button("Go");
        b.setBounds(10,10,50,30);
        b.addActionListener(new ActionListener()
          {
              public void actionPerformed(ActionEvent e)
                {
                                JOptionPane.showMessageDialog

(null,"test","test",JOptionPane.INFORMATION_MESSAGE);
                }
          }
        );
        f.add(b);
        f.setVisible(true);
 }
}
```

canvas

```
import java.awt.*;
public class CanvasExample
{
  public CanvasExample()
  {
    Frame f= new Frame("Canvas Example");
    f.add(new MyCanvas());
    f.setLayout(null);
    f.setSize(400, 400);
    f.setVisible(true);
  }
  public static void main(String args[])
  {
    new CanvasExample();
  }
}
class MyCanvas extends Canvas
{
        public MyCanvas() {
        setBackground (Color.GRAY);
        setSize(300, 200);
     }
  public void paint(Graphics g)
  {
    g.setColor(Color.red);
    g.fillOval(75, 75, 150, 75);
  }
}
```

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.JOptionPane;
public class demoawt
{
  demoawt()
 {
{
        Frame f=new Frame();
        f.setSize(500,500);
        f.setLayout(null);
        //--------------
        f.addWindowListener(new WindowAdapter(){
                public void  windowClosing(WindowEvent e)
                {
System.exit(0);
                }
        });
        //---------------
        Button b=new Button("Go");
        b.setBounds(10,10,100,50);
        b.addActionListener(new ActionListener()
          {
                public void actionPerformed(ActionEvent e)
                {
                                JOptionPane.showMessageDialog

(null,"test","test",JOptionPane.INFORMATION_MESSAGE);
                }
          }
        );
        f.add(b);
        f.add(new myCanvas());
        f.setVisible(true);
}
```

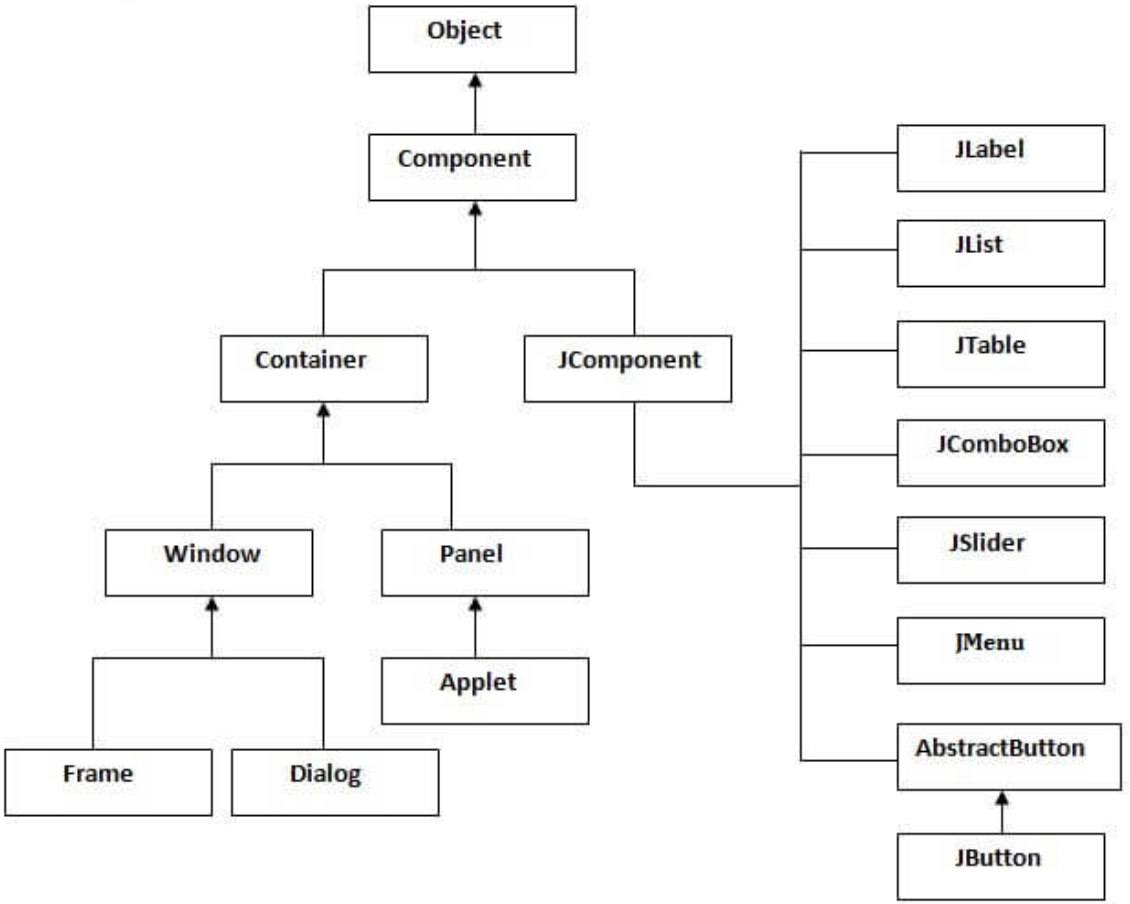
```
 }
  public static void main(String args[])
 {
        new demoawt();
 }
//=====================================
class myCanvas extends Canvas
{
        public myCanvas()
        {
        setBackground(Color.GRAY);
        setSize(300,300);
        }
        public void paint(Graphics g)
        {
                g.fillOval(100,100,150,150);
        }
}

}
```

Swing

- Java Swing tutorial is a part of Java Foundation Classes (JFC) that is used to create window-based applications.
- It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.

swing


Object
Component
Container
JComponent
Window
Panel
Frame
Dialog
Applet
JLabel
JList
JTable
JComboBox
JSlider
JMenu
AbstractButton
JButton

```
import javax.swing.*;
public class FirstSwingExample {
public static void main(String[] args) {
JFrame f=new JFrame

JButton b=new JButton("click");
b.setBounds(130,100,100, 40);

f.add(b);//adding button in JFrame

f.setSize(400,500);
f.setLayout(null);
f.setVisible(true);
}
}
```

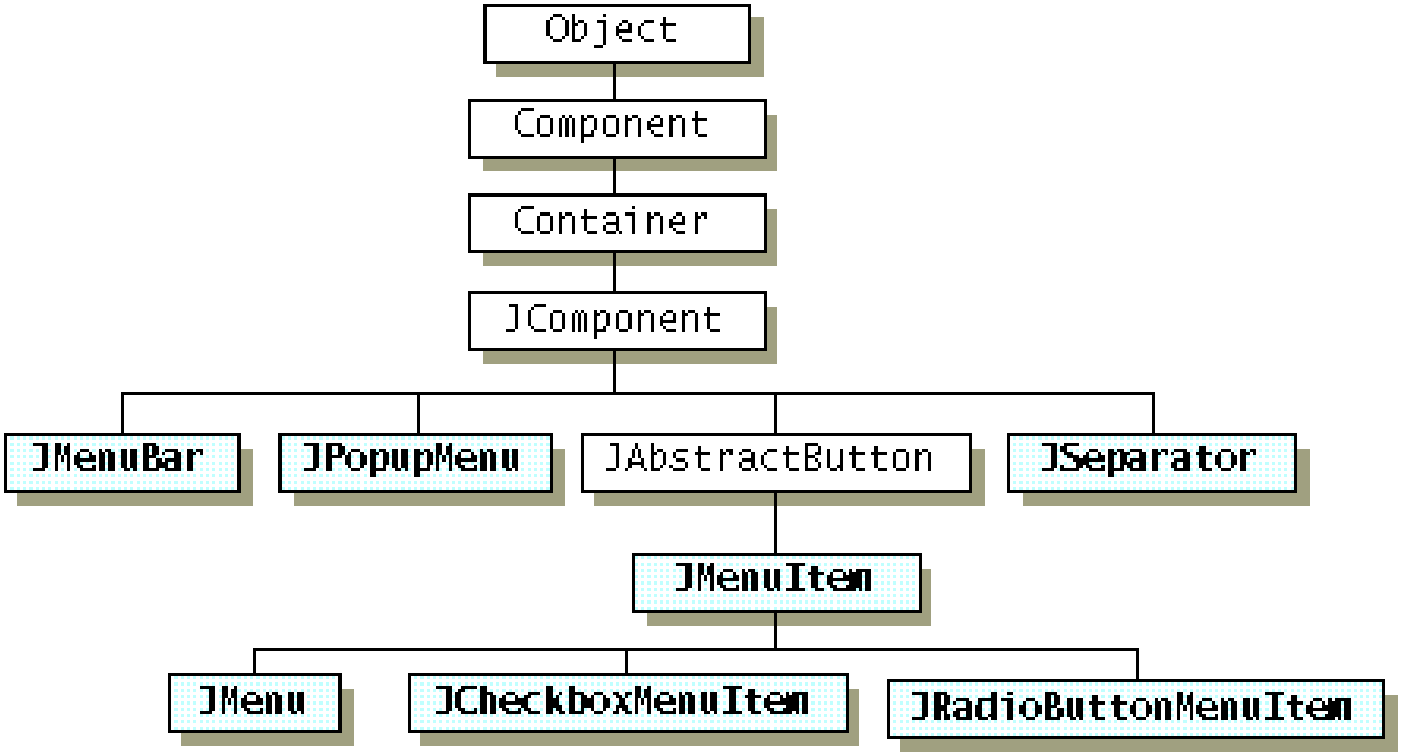
Difference between AWT and Swing

No.	Java AWT	Java Swing
1)	AWT components are platform-dependent.	Java swing components are platform-independent.
2)	AWT components are heavyweight.	Swing components are lightweight.
3)	AWT doesn't support pluggable look and feel.	Swing supports pluggable look and feel.
4)	AWT provides less components than Swing.	Swing provides more powerful components such as tables, lists, scrollpanes, colorchooser, tabbedpane etc.
5)	AWT doesn't follows MVC(Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view.	Swing follows MVC.

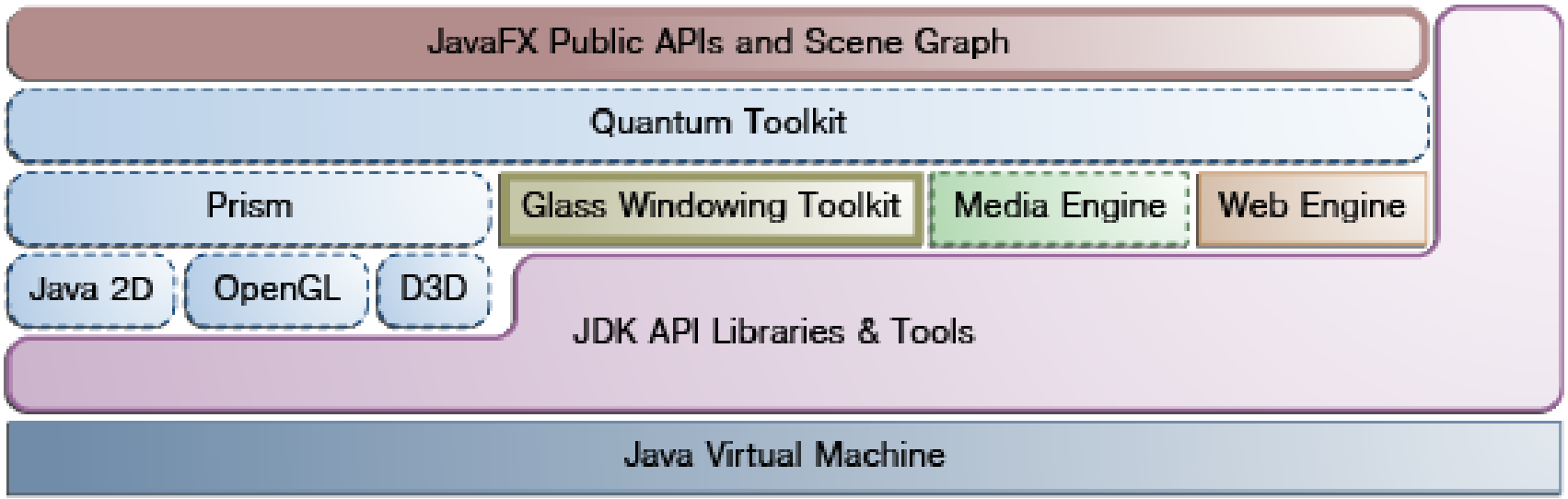
- Swing是一個基於Java的跨平台MVC框架。使用單執行緒模式。此框架還在代碼結構層和圖形彩現層之間插入了一個抽象層。

- 所有的元件都是從javax.swing.JComponent類繼承

- Java painter


Object
Component
Container
JComponent
JMenuBar
JPopupMenu
JAbstractButton
JSeparator
JMenuItem
JMenu
JCheckboxMenuItem
JRadioButtonMenuItem

JAVAFX


JavaFX Public APIs and Scene Graph
Quantum Toolkit
Prism
Glass Windowing Toolkit
Media Engine
Web Engine
Java 2D
OpenGL
D3D
JDK API Libraries & Tools
Java Virtual Machine

- https://openjfx.io/openjfx-docs/
- https://gluonhq.com/products/javafx/

- JavaFX is a software platform for creating and delivering desktop applications, as well as rich Internet applications (RIAs) that can run across a wide variety of devices.
- JavaFX is intended to replace Swing as the standard GUI library for Java SE, but both will be included for the foreseeable future
- JavaFX has support for desktop computers and web browsers on Microsoft Windows, Linux, and macOS.

- https://docs.oracle.com/javafx/2/api/javafx/scene/doc-files/cssref.html

Empty application

```
package demofx;

import javafx.application.Application;
import javafx.stage.Stage;

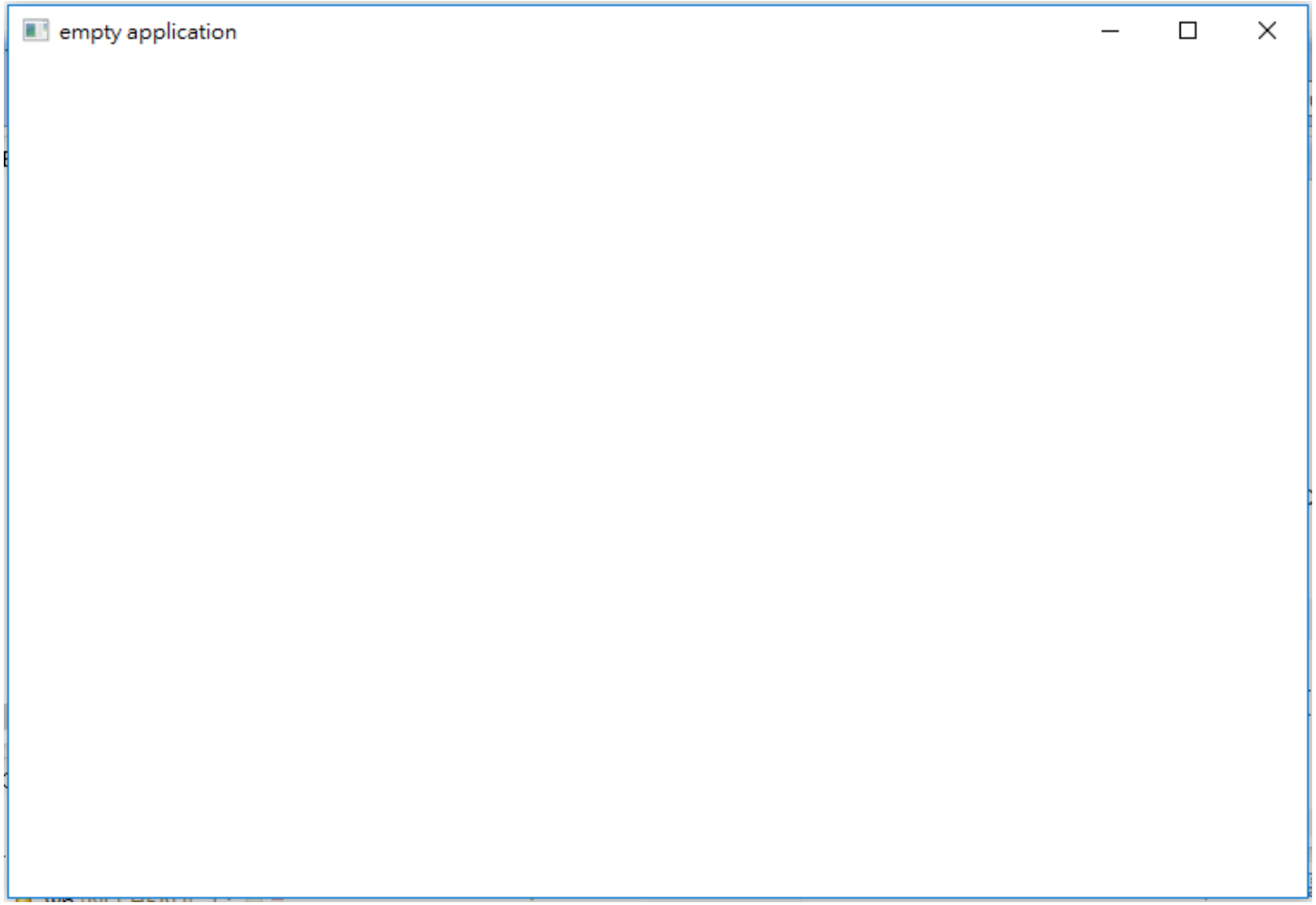
public class emptyFX extends Application {

	@Override
	public void start(Stage arg0) throws Exception {
		arg0.setTitle("empty application");
		arg0.show();
	}

	public static void main(String[] args) {
		launch(args);

	}

}
```


empty application

```
package demofx;

import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class demojavafx extends Application{

public static void main(String[] args) {
launch(args);
}
public void start(Stage arg0) throws Exception {

    final Button button = new Button();
    button.setText("Hello World");
    final StackPane root = new StackPane();
    root.getChildren().add(button);

    final Scene scene = new Scene(root, 300, 250);
    arg0.setTitle("Hello World!");
    arg0.setScene(scene);
    arg0.show();
}

}
```

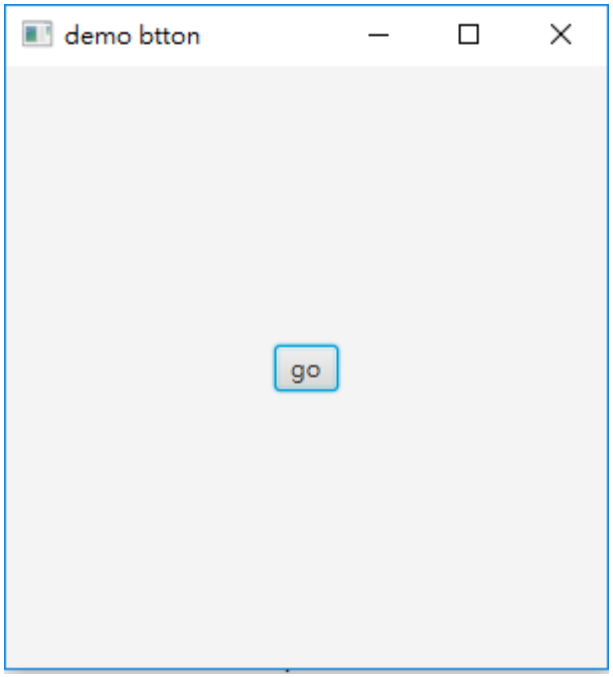
```
package demofx;
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;
import jdk.internal.org.objectweb.asm.Handle;

public class fxbutton extends Application {
  @Override
  public void start(Stage arg0) throws Exception {
        Button b=new Button();
        b.setText("go");
                b.setOnAction(new EventHandler<ActionEvent>() {

                        public void handle(ActionEvent arg0) {
                }
                });
        StackPane sp=new StackPane();
        sp.getChildren().add(b);
        Scene s=new Scene(sp,300,300);
        arg0.setScene(s);
        arg0.setTitle("demo btton");
        arg0.show();
    }

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        launch(args);
    }

}
```


demo btton
go

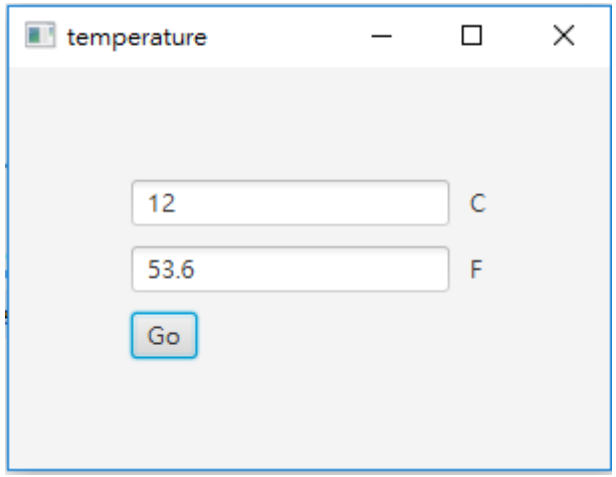
- GridPane

```
public class gridpanedemo extends Application {
        Label lC;
        Label lF;
        TextField C;
        TextField F;
        Button b;
        @Override
        public void start(Stage arg0) throws Exception {
        lC=new Label();
        lC.setText("C");
        lF=new Label();
        lF.setText("F");
        C=new TextField();
        F=new TextField();
        b=new Button();
        b.setText("Go");
        b.setOnAction(new EventHandler<ActionEvent>() {
        public void handle(ActionEvent arg0) {
        F.setText(String.valueOf(Double.parseDouble(C.getText()
)*9./5+32));
                                                }
                                        });

                        GridPane gp=new GridPane();
                        gp.setAlignment(Pos.CENTER);
                        gp.setHgap(10);
                        gp.setVgap(10);
                        gp.add(lC,1,0);
                        gp.add(C, 0, 0);
                        gp.add(lF,1,1);
                        gp.add(F, 0,1);
                        gp.add(b, 0, 2,2,1);
                        Scene s=new Scene(gp,300,200);
                        arg0.setScene(s);
                        arg0.setTitle("temperature");
                        arg0.show();
        }

        public static void main(String[] args) {
                        launch(args);
        }

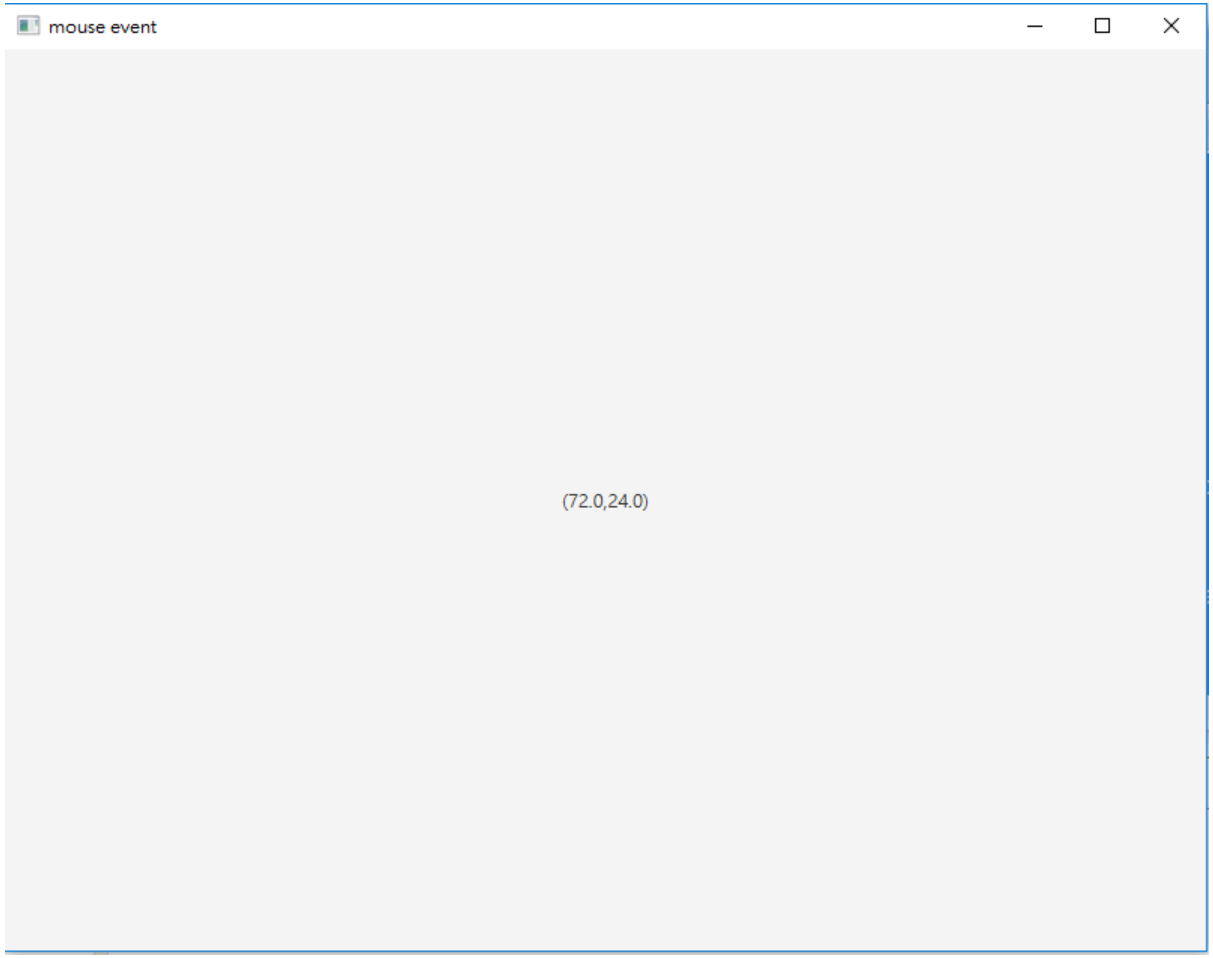
}
```


temperature
12
C
53.6
F
Go

```
public class FXmouse extends Application {
        Label l;
        @Override
        public void start(Stage arg0) throws Exception {
                StackPane sp=new StackPane();
                l=new Label();
                sp.getChildren().add(l);
                Scene s=new Scene(sp,800,600);
                s.setOnMouseMoved(new EventHandler<MouseEvent>() {
                public void handle(MouseEvent arg0) {
                                                l.setText("("+arg0.getX()+","+ arg0.getY()+")");

                                        }
                });
                arg0.setScene(s);
                arg0.setTitle("mouse event");
                arg0.show();
        }
        public static void main(String[] args) {
                launch(args);
        }

}
```


mouse event
(72.0,24.0)

Line

- **public class FXmouse extends Application {**
- Label l;
- @Override
- **public void start(Stage arg0) throws Exception {**
- Line line = **new Line();**
- line.setStartX(100.0);
- line.setStartY(50.0);
- line.setEndX(500.0);
- line.setEndY(50.0);
- Group root = **new Group(line);**
- StackPane sp=**new StackPane();**
- sp.getChildren().add(root);
- l=**new Label();**
- sp.getChildren().add(l);
- Scene s=**new Scene(sp,800,600);**
- s.setOnMouseMoved(**new EventHandler<MouseEvent>() {**
- **public void handle(MouseEvent arg0) {**
- l.setText("("+arg0.getX()+","+ arg0.getY()+")");
- }
- });
- arg0.setScene(s);
- arg0.setTitle("mouse event");
- arg0.show();
- }
- **public static void main(String[] args) {**
- *launch(args);*
- }
- }

canvas

```
package demofx;

import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;

public class fxcanvas extends Application {

	@Override
	public void start(Stage stage) throws Exception {
		Canvas canvas = new Canvas(400, 200);
		canvas.setWidth(400);
		canvas.setHeight(200);
		GraphicsContext gc = canvas.getGraphicsContext2D();

		gc.strokeText("Hello Canvas", 150, 100);
		gc.strokeLine(9, 9, 100,100);
		Pane root = new Pane();
		root.setStyle("-fx-padding: 10;" +
				"-fx-border-style: solid inside;" +
				"-fx-border-width: 2;" +
				"-fx-border-insets: 5;" +
				"-fx-border-radius: 5;" +
				"-fx-border-color: blue;");

		root.getChildren().add(canvas);
		Scene scene = new Scene(root);
		stage.setScene(scene);
		stage.setTitle("Creation of a Canvas");
		stage.show();

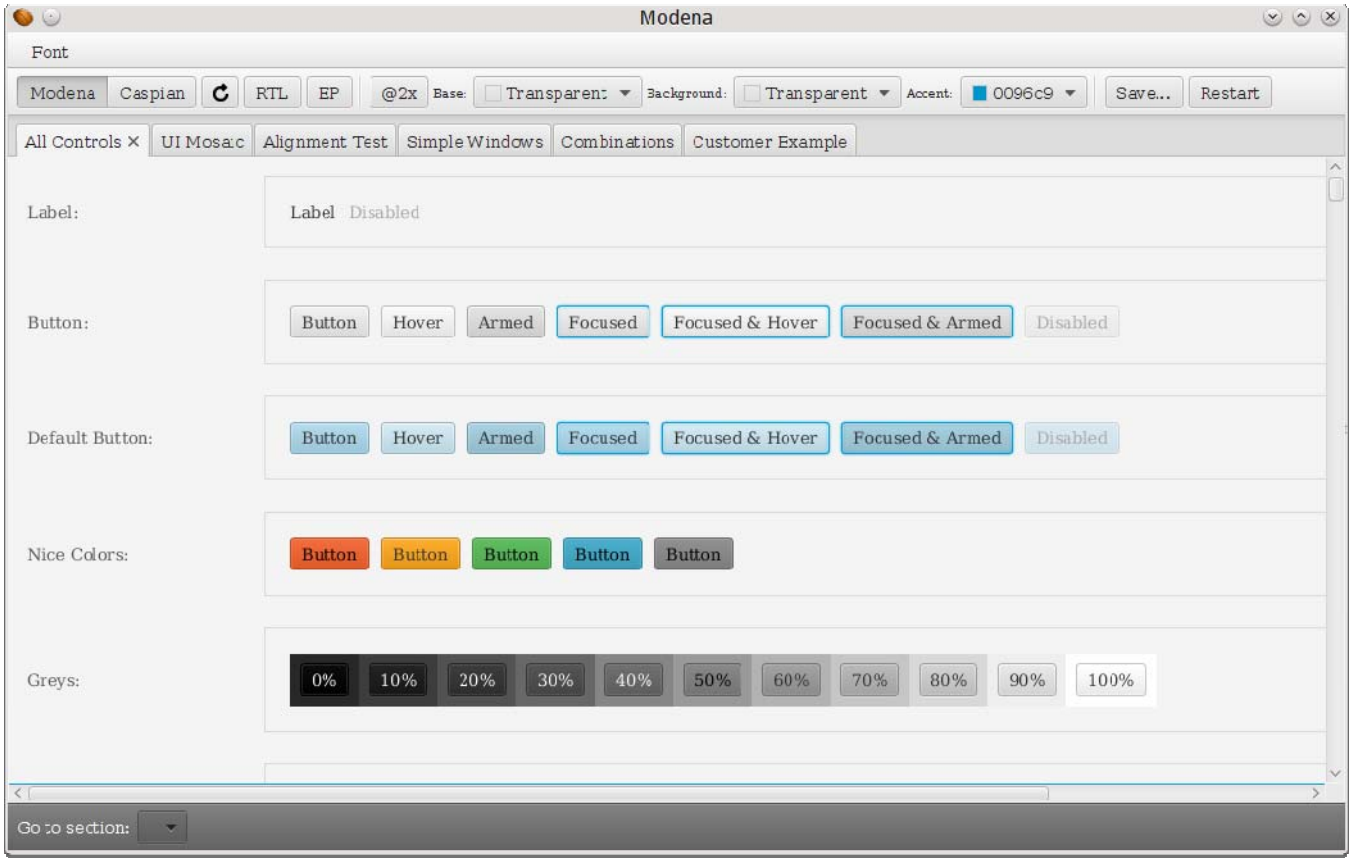
	}

	public static void main(String[] args) {
		launch(args);

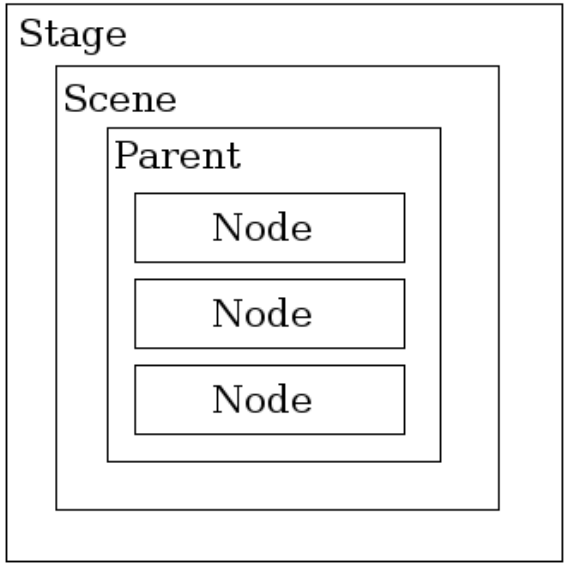
	}

}
```

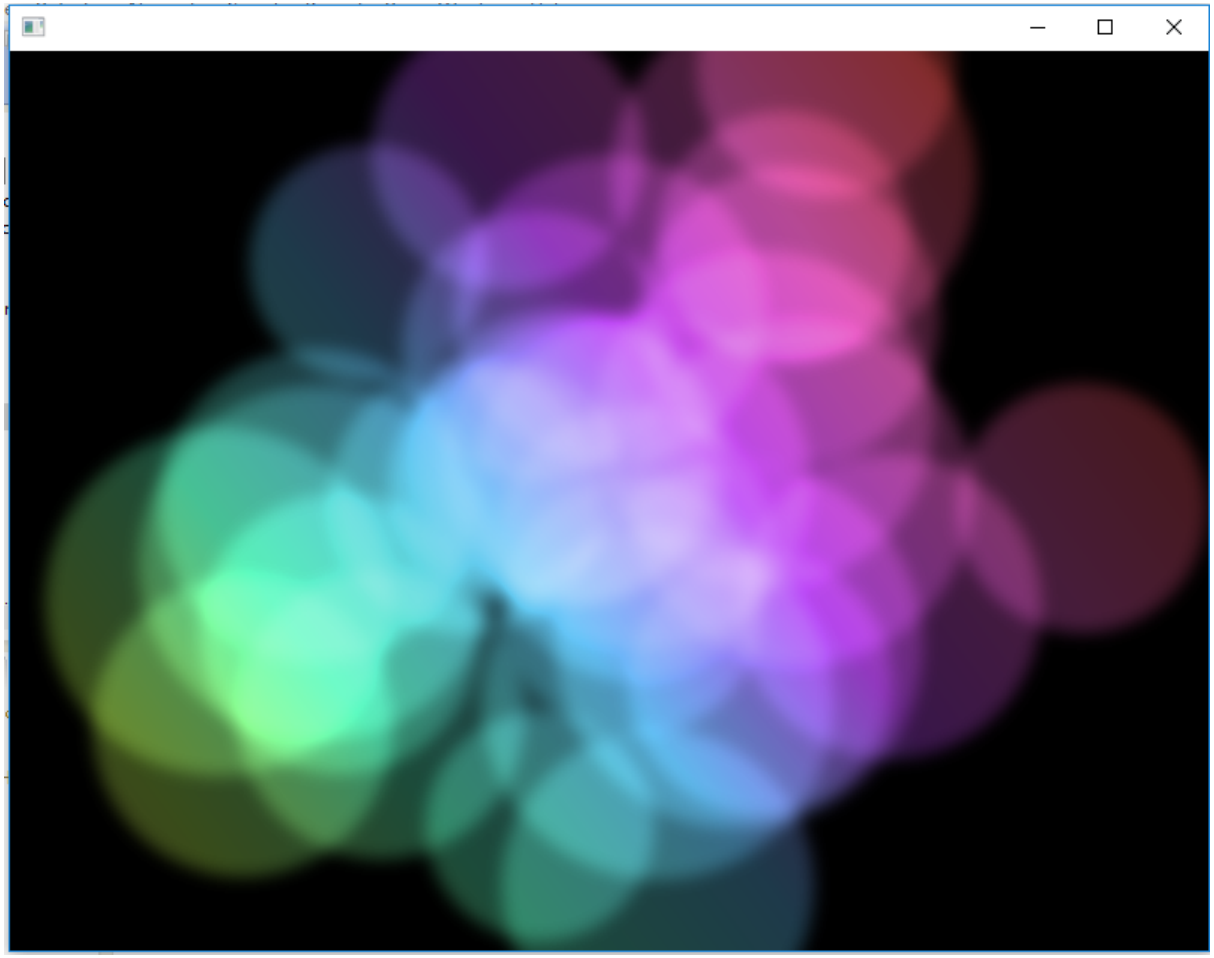
JavaFX


Modena
Font
Modena
Caspian
RTL
EP
@2x
Base:
Transparent
Background:
Transparent
Accent:
0096c9
Save...
Restart
All Controls
UI Mosaic
Alignment Test
Simple Windows
Combinations
Customer Example
Label:
Label
Disabled
Button:
Button
Hover
Armed
Focused
Focused & Hover
Focused & Armed
Disabled
Default Button:
Button
Hover
Armed
Focused
Focused & Hover
Focused & Armed
Disabled
Nice Colors:
Button
Button
Button
Button
Button
Greys:
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Go to section:

javaFX



- stackPane



```
public class fxanim extends Application {
public static final int width = 800;
   public static final int height = 600;
   public static final long duration = 30000;
   public static final int count = 30;
@Override
public void start(Stage primaryStage) throws Exception {
Group root = new Group();
Scene scene = new Scene(root, width, height, Color.BLACK);
primaryStage.setScene(scene);
Rectangle colors = new Rectangle(0, 0,
new LinearGradient(0f, 1f, 1f, 0f, true, CycleMethod.NO_CYCLE, new Stop[]{
   new Stop(0, Color.web("#f8bd55")),
   new Stop(0.14, Color.web("#c0fe56")),
   new Stop(0.28, Color.web("#5dfbc1")),
   new Stop(0.43, Color.web("#64c2f8")),
   new Stop(0.57, Color.web("#be4af7")),
   new Stop(0.71, Color.web("#ed5fc2")),
   new Stop(0.85, Color.web("#ef504c")),
   new Stop(1, Color.web("#f2660f")),}));

colors.widthProperty().bind(scene.widthProperty());
colors.heightProperty().bind(scene.heightProperty());

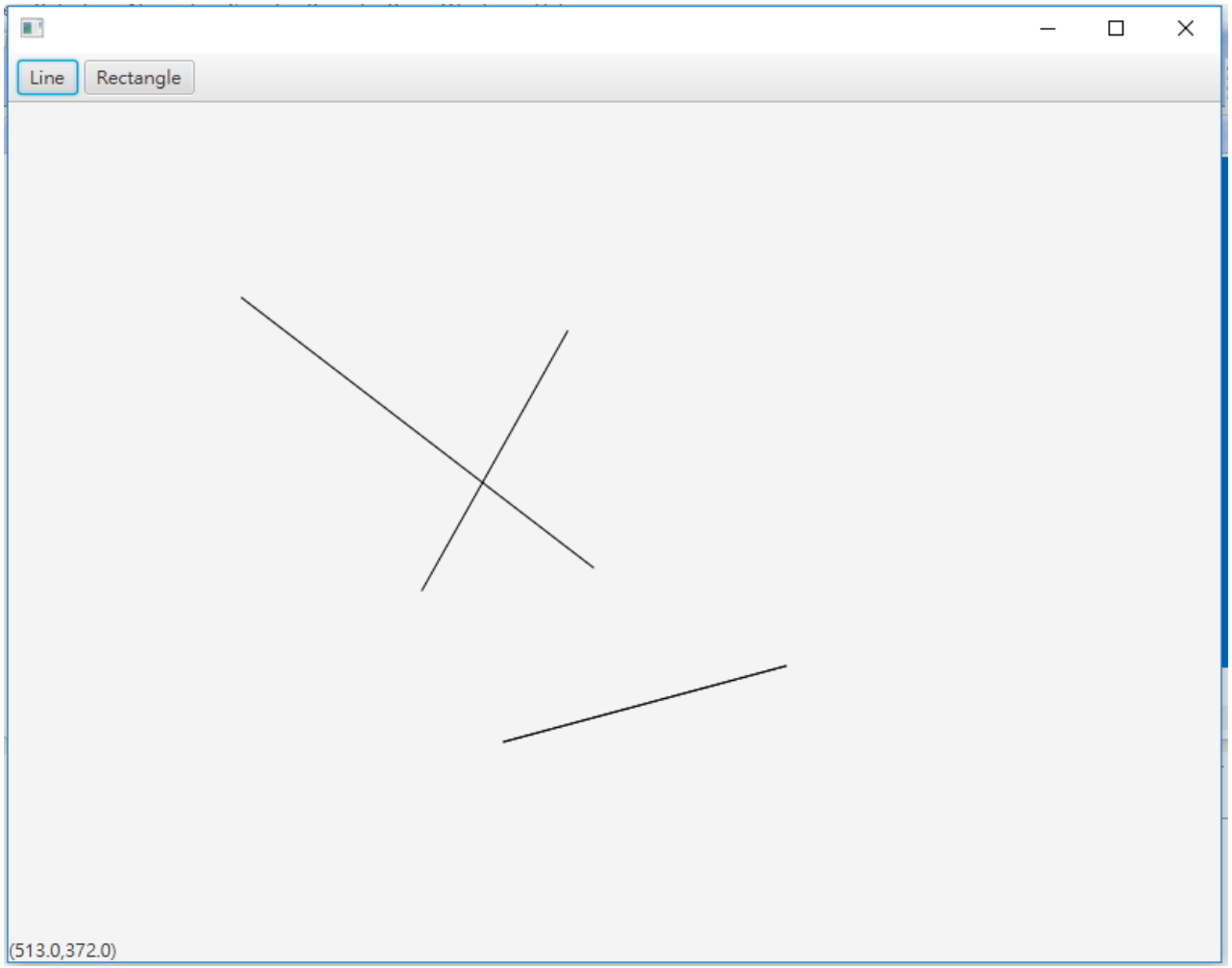
Group circles = new Group();
for (int i = 0; i < count; i++) {
   Circle circle = new Circle(Math.random() * 50 + 75, Color.web("white",
0.15));
   circle.setStrokeType(StrokeType.OUTSIDE);
   circle.setStrokeWidth(4);
   circles.setEffect(new BoxBlur(10, 10, 3));
   circles.getChildren().add(circle);
}
Timeline timeline = new Timeline();
for(Node circle : circles.getChildren())
{
KeyFrame start = new KeyFrame(Duration.ZERO, new
KeyValue(circle.translateXProperty(), Math.random() * scene.getWidth()),
new KeyValue(circle.translateYProperty(), Math.random() *
scene.getHeight()));
   KeyFrame end = new KeyFrame(new Duration(duration), new
KeyValue(circle.translateXProperty(), Math.random() * scene.getWidth()),
new KeyValue(circle.translateYProperty(), Math.random() *
scene.getHeight()));
   timeline.getKeyFrames().addAll(start, end);
}
timeline.play();


//colors.setBlendMode(BlendMode.SCREEN);
colors.setBlendMode(BlendMode.OVERLAY);

root.getChildren().add(circles);
root.getChildren().add(colors);

primaryStage.show();
}
public static void main(String[] args) {
launch(args);

}
}
```


Line
Rectangle
(513.0,372.0)

```
package fxpainter;

import com.sun.xml.internal.ws.resources.DispatchMessages;

import javafx.application.Application;
import javafx.beans.InvalidationListener;
import javafx.beans.Observable;
import javafx.event.Event;
import javafx.event.EventHandler;
import javafx.scene.Cursor;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.ToolBar;
import javafx.scene.input.MouseButton;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.HBox;
import javafx.scene.layout.Pane;
import javafx.scene.layout.StackPane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Line;
import javafx.stage.Stage;



public class painter_main extends Application {
public static final int width = 800;
   public static final int height = 600;

   BorderPane borderPane ;
   ToolBar toolbar ;
   HBox statusbar;
   Pane sp;
   Canvas canvas;
   Label coord;
   Button btnLine;

   Point p1=new Point(0,0);
   Point p2=new Point(0,0);
   Point p=new Point(0,0);
```

image

```
package imageprocessing;

import java.awt.Graphics2D;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;

import javax.imageio.ImageIO;

import sun.awt.image.PixelConverter.Rgba;
import javafx.application.Application;
import javafx.embed.swing.SwingFXUtils;
import javafx.event.Event;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.control.Button;
import javafx.scene.control.ToolBar;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.image.PixelFormat;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.GridPane;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.stage.FileChooser;
import javafx.stage.Stage;

public class imagetool extends Application {
final static int width=800;
final static int height=600;
BorderPane bp;
ToolBar tb;
Button btnread;
Button btngray;
GridPane gp;
ImageView im1;
Canvas canvas;
Stage mainstage;
String filename;
public void start(Stage stage) throws Exception {
mainstage=stage;
```

FILE

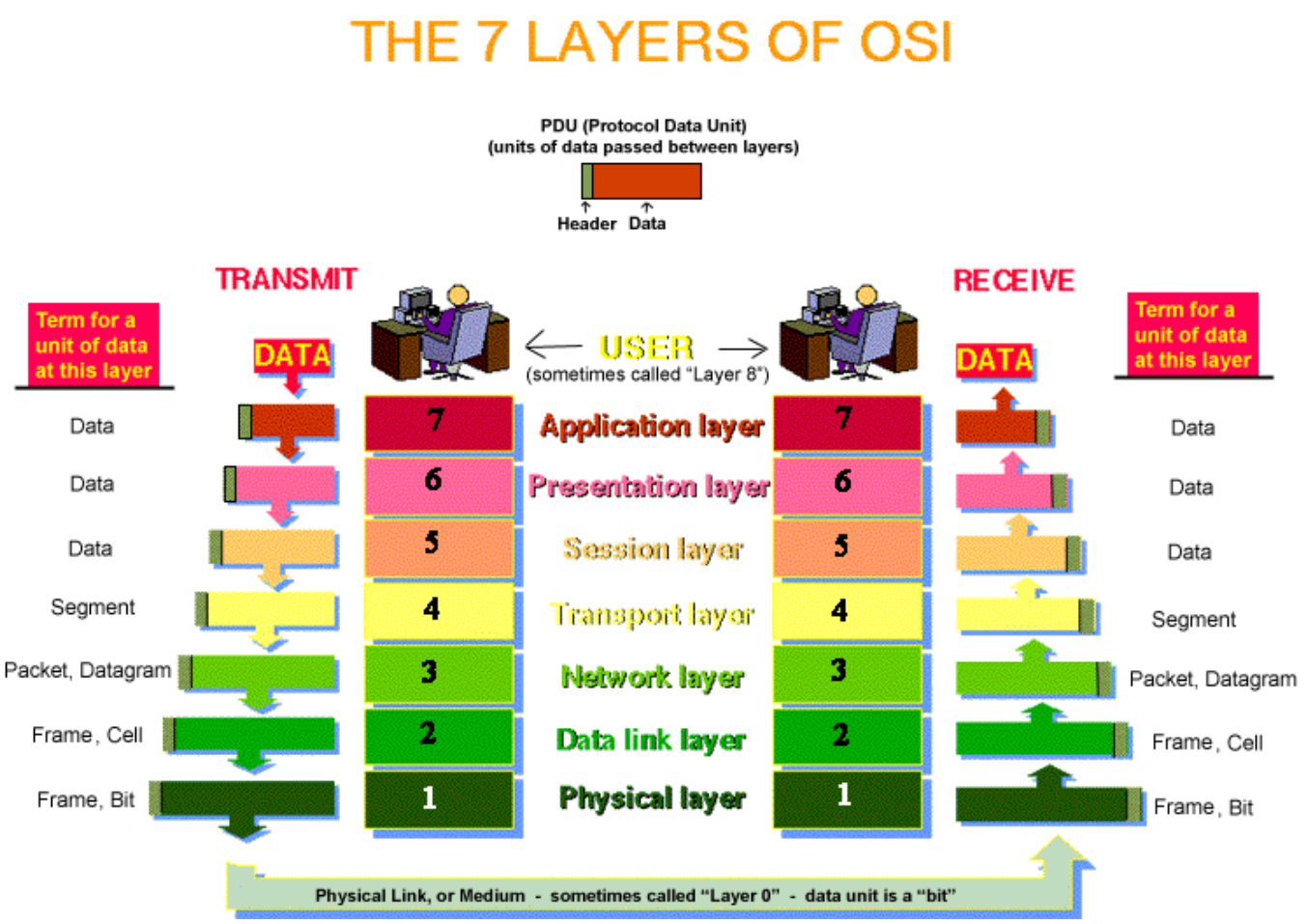
File

```
public class file {
public static void main(String[] args) {
String tmp;
try {
    Scanner s=new Scanner(new
    File("C:/wpfang/teach/java/demo/demo_socket/src/demo_socket/file.java"));
    while(s.hasNextLine())
    {
        tmp=s.nextLine();
        System.out.println(tmp);
    }
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
}
}
```

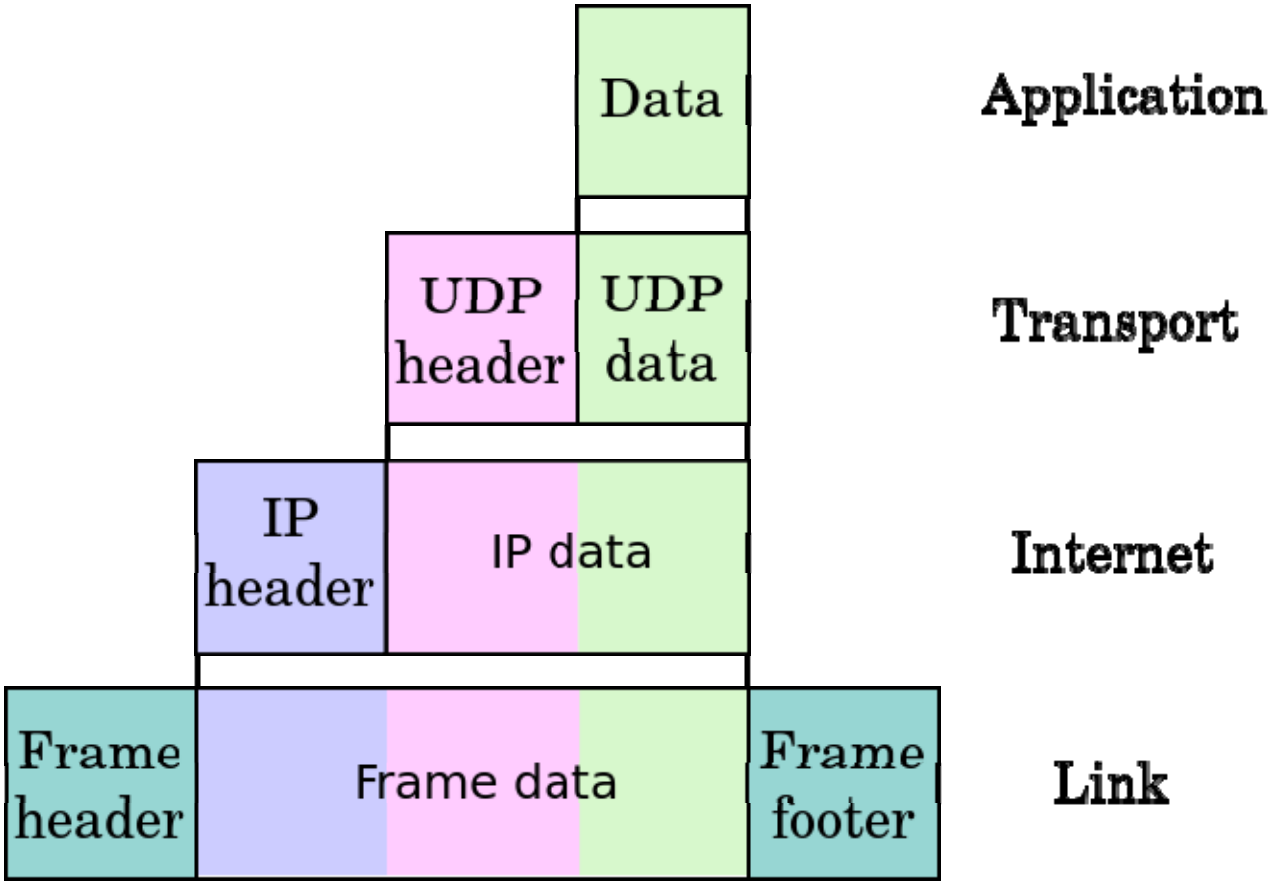
```
public class file_write {
public static void main(String[] args) {
     String str = "test";
    BufferedWriter writer;
     try {
            writer = new BufferedWriter(new FileWriter("test.txt"));
            writer.write(str);
             writer.close();}
     catch (IOException e) {
            e.printStackTrace();
     }
}
}
```

網路功能初探

OSI


THE 7 LAYERS OF OSI
PDU (Protocol Data Unit)
(units of data passed between layers)
Header
Data
TRANSMIT
RECEIVE
Term for a unit of data at this layer
DATA
USER
(sometimes called "Layer 8")
Data
Data
Data
Segment
Packet, Datagram
Frame, Cell
Frame, Bit
7 Application layer
6 Presentation layer
5 Session layer
4 Transport layer
3 Network layer
2 Data link layer
1 Physical layer
Physical Link, or Medium - sometimes called "Layer 0" - data unit is a "bit"

Internet


Data
Application
UDP header
UDP data
Transport
IP header
IP data
Internet
Frame header
Frame data
Frame footer
Link

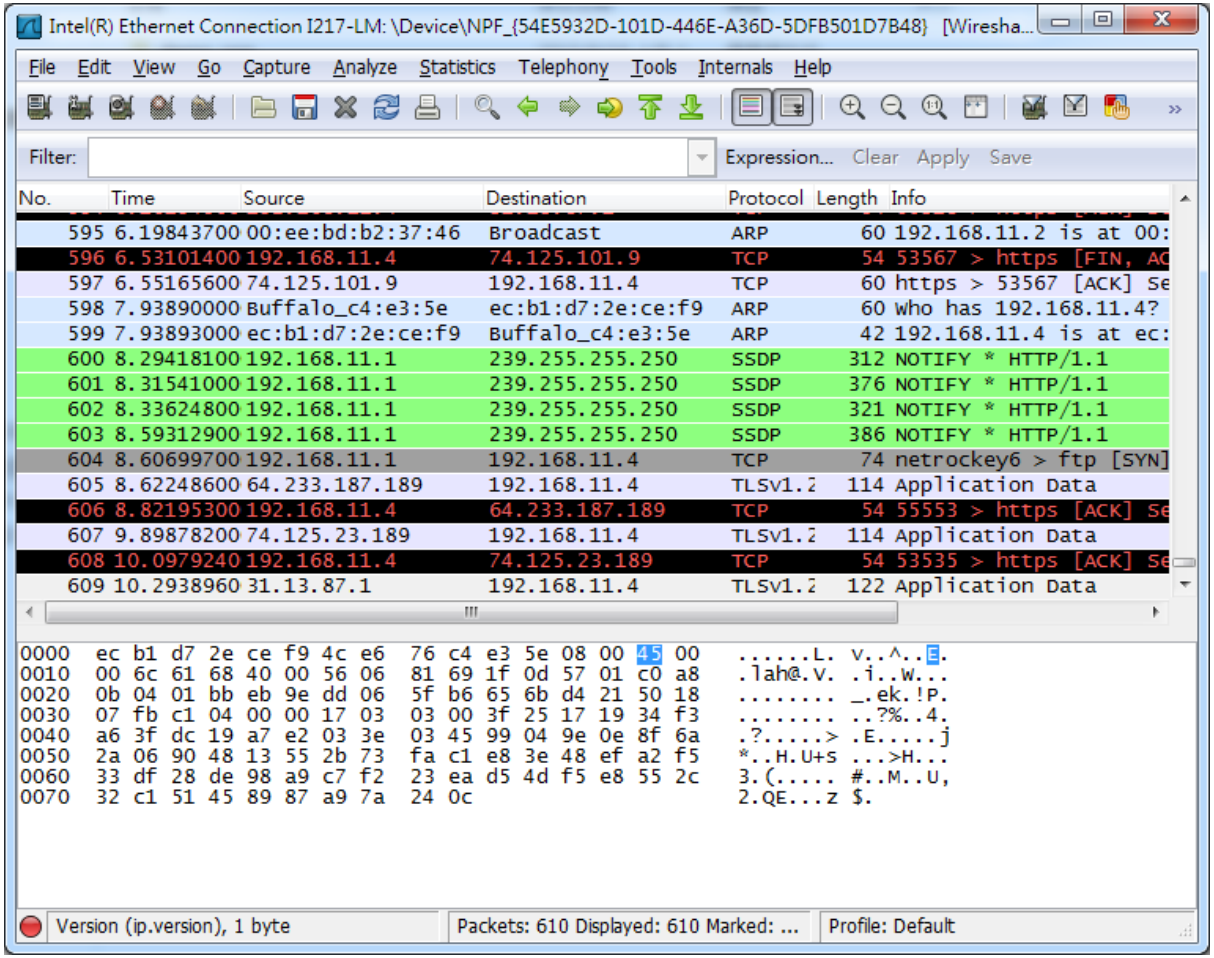
UDP

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source port																Destination port															
4	32	Length																Checksum															

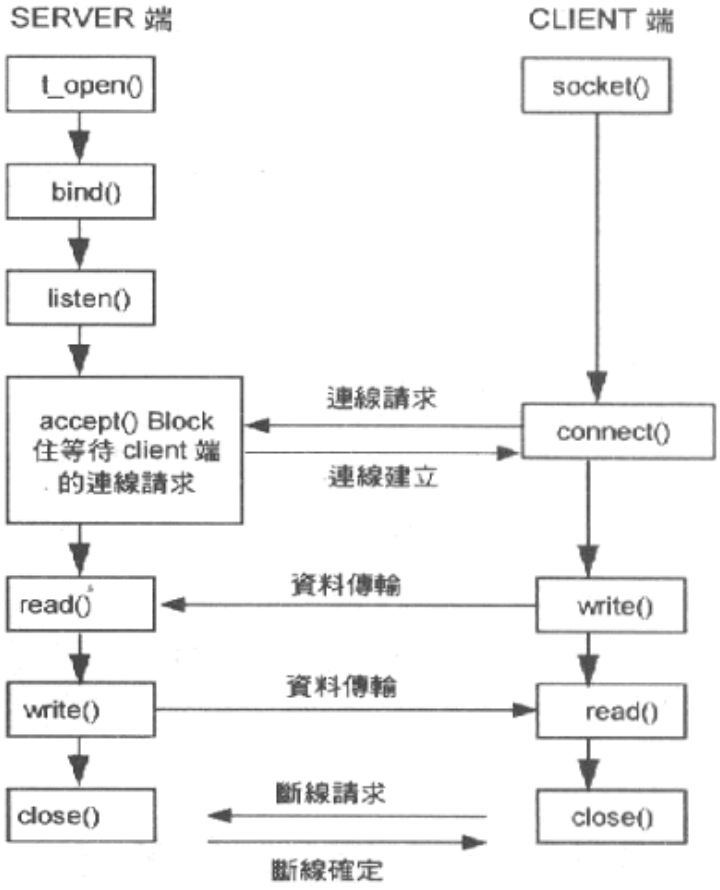
TCP

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset			Reserved **0 0 0**				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size															
16	128	Checksum																Urgent pointer (if URG set)															
20 ...	160 ...	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.) ...																															

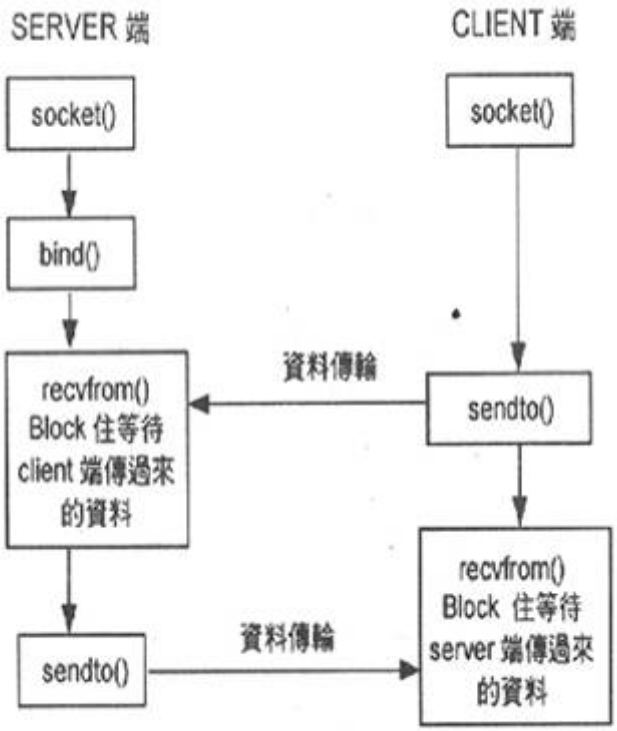
wiresherk


Intel(R) Ethernet Connection I217-LM: \Device\NPF_{54E5932D-101D-446E-A36D-5DFB501D7B48} [Wiresha...
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: Expression... Clear Apply Save
No. Time Source Destination Protocol Length Info
595 6.19843700 00:ee:bd:b2:37:46 Broadcast ARP 60 192.168.11.2 is at 00:
596 6.53101400 192.168.11.4 74.125.101.9 TCP 54 53567 > https [FIN, AC
597 6.55165600 74.125.101.9 192.168.11.4 TCP 60 https > 53567 [ACK] Se
598 7.93890000 Buffalo_c4:e3:5e ec:b1:d7:2e:ce:f9 ARP 60 Who has 192.168.11.4?
599 7.93893000 ec:b1:d7:2e:ce:f9 Buffalo_c4:e3:5e ARP 42 192.168.11.4 is at ec:
600 8.29418100 192.168.11.1 239.255.255.250 SSDP 312 NOTIFY * HTTP/1.1
601 8.31541000 192.168.11.1 239.255.255.250 SSDP 376 NOTIFY * HTTP/1.1
602 8.33624800 192.168.11.1 239.255.255.250 SSDP 321 NOTIFY * HTTP/1.1
603 8.59312900 192.168.11.1 239.255.255.250 SSDP 386 NOTIFY * HTTP/1.1
604 8.60699700 192.168.11.1 192.168.11.4 TCP 74 netrockey6 > ftp [SYN]
605 8.62248600 64.233.187.189 192.168.11.4 TLSv1.2 114 Application Data
606 8.82195300 192.168.11.4 64.233.187.189 TCP 54 55553 > https [ACK] Se
607 9.89878200 74.125.23.189 192.168.11.4 TLSv1.2 114 Application Data
608 10.0979240 192.168.11.4 74.125.23.189 TCP 54 53535 > https [ACK] Se
609 10.2938960 31.13.87.1 192.168.11.4 TLSv1.2 122 Application Data
0000 ec b1 d7 2e ce f9 4c e6 76 c4 e3 5e 08 00 45 00 ......L. v..^..E.
0010 00 6c 61 68 40 00 56 06 81 69 1f 0d 57 01 c0 a8 .lah@.V. .i..W...
0020 0b 04 01 bb eb 9e dd 06 5f b6 65 6b d4 21 50 18 .... _.ek.!P.
0030 07 fb c1 04 00 00 17 03 03 00 3f 25 17 19 34 f3 ....?%..4.
0040 a6 3f dc 19 a7 e2 03 3e 03 45 99 04 9e 0e 8f 6a .?.....> .E.....j
0050 2a 06 90 48 13 55 2b 73 fa c1 e8 3e 48 ef a2 f5 *..H.U+s ...>H...
0060 33 df 28 de 98 a9 c7 f2 23 ea d5 4d f5 e8 55 2c 3.(..... #..M..U,
0070 32 c1 51 45 89 87 a9 7a 24 0c 2.QE...z $.
Version (ip.version), 1 byte
Packets: 610 Displayed: 610 Marked: ...
Profile: Default

TCP


SERVER 端
CLIENT 端
t_open()
socket()
bind()
listen()
accept() Block
住等待 client 端
的連線請求
連線請求
連線建立
connect()
read()
資料傳輸
write()
write()
資料傳輸
read()
close()
斷線請求
close()
斷線確定

UDP


SERVER 端
CLIENT 端
socket()
socket()
bind()
recvfrom()
Block 住等待
client 端傳過來
的資料
資料傳輸
sendto()
recvfrom()
Block 住等待
server 端傳過來
的資料
sendto()
資料傳輸

Get http

```
public class geturl {
public static void main(String[] args) throws IOException {
URL url=new URL("http://140.138.146.85");
HttpURLConnection conn=(HttpURLConnection) url.openConnection();
conn.connect();
BufferedInputStream is=new BufferedInputStream(conn.getInputStream());
byte [] tmp = new byte[1024];
int len=0;
   final Charset UTF_8 = Charset.forName("UTF-8");
while ((len = is.read(tmp)) != -1) {

String value = new String(tmp, UTF_8);
System.out.print(value);
        }
is.close();
}
}
```

```
public class status {

public static void main(String[] args) {
try {
Socket s=new Socket("www.yzu.edu.tw",80);
System.out.println(s.getInetAddress());
} catch (UnknownHostException e) {
e.printStackTrace();
} catch (IOException e) {
e.printStackTrace();
}

}
}
```

```
public class socketserverdemo {
 public static void main(String args[]){
 mythread t=new mythread();
 t.start();
 }
}
class mythread extends Thread
{
ServerSocket ss;
mythread()
{
try {
ss=new ServerSocket(81);
} catch (IOException e) {
e.printStackTrace();
}

}

public void run() {
 while (true)
   {
    System.out.println("waiting...");
    try {
     Socket s=ss.accept();
     System.out.println(s.getLocalAddress());
     s.close();
    }
    catch (IOException e) {
    e.printStackTrace();
    }
   }
}
}
```

Server

```
import java.net.ServerSocket;
import java.net.Socket;

public class SocketServer extends java.lang.Thread {

  private boolean OutServer = false;
  private ServerSocket server;
  private final int ServerPort = 8765;// 要監控的port

  public SocketServer() {
    try {
      server = new ServerSocket(ServerPort);

    } catch (java.io.IOException e) {
      System.out.println("Socket啟動有問題 !");
      System.out.println("IOException :" + e.toString());
    }
  }

  public void run() {
    Socket socket;
    java.io.BufferedInputStream in;

    System.out.println("伺服器已啟動 !");
    while (!OutServer) {
      socket = null;
      try {
        synchronized (server) {
          socket = server.accept();
        }
        System.out.println("取得連線 : InetAddress = "
            + socket.getInetAddress());
        // TimeOut時間
        socket.setSoTimeout(15000);

        in = new
java.io.BufferedInputStream(socket.getInputStream());
        byte[] b = new byte[1024];
        String data = "";
        int length;
        while ((length = in.read(b)) > 0)// <=0的話就是結束
了
        {
          data += new String(b, 0, length);
        }

        System.out.println("我取得的值:" + data);
        in.close();
        in = null;
        socket.close();

      } catch (java.io.IOException e) {
        System.out.println("Socket連線有問題 !");
        System.out.println("IOException :" + e.toString());
      }

    }
  }

  public static void main(String args[]) {
    (new SocketServer()).start();
  }

}
```

Client

```
import java.net.InetSocketAddress;
import java.net.Socket;
import java.io.BufferedOutputStream;

public class SocketClient {
   private String address = "127.0.0.1";// 連線的ip
   private int port = 8765;// 連線的port

   public SocketClient() {

      Socket client = new Socket();
      InetSocketAddress isa = new InetSocketAddress(this.address, this.port);
      try {
         client.connect(isa, 10000);
         BufferedOutputStream out = new BufferedOutputStream(client
               .getOutputStream());
         // 送出字串
         out.write("Send From Client ".getBytes());
         out.flush();
         out.close();
         out = null;
         client.close();
         client = null;

      } catch (java.io.IOException e) {
         System.out.println("Socket連線有問題 !");
         System.out.println("IOException :" + e.toString());
      }
   }

   public static void main(String args[]) {
      new SocketClient();
   }
}
```

main

```
public class Main {

    public static void main(String args[])
    {
        if(args.length==0)
            System.out.println("請傳入參數server或client");
        if(args[0].equals("server"))
            (new SocketServer()).start();
        else
            new SocketClient();
    }
}
```

Server

```
public class SocketServer implements java.lang.Runnable {
 private int port;
 private java.net.ServerSocket ss;

 public SocketServer(int port) throws java.io.IOException {
  this.port = port;
  // 建立一個ServerSocket
  this.ss = new java.net.ServerSocket(port);
 }

 public void run() {
  java.net.Socket sk = null;
  while (true)// 永遠執行
  {
   // 等待連入
   System.out.println("waiting...");
   try {
    // 取得連線Socket
    sk = this.ss.accept();
    // 取得Client連線Address
    System.out.println(sk.getLocalAddress());
    sk.close();
   } catch (java.io.IOException e) {
    e.printStackTrace();
   }

  }
 }

 public static void main(String args[]) throws java.io.IOException {
  // runable要new一個Thread,再把runnable置入
  java.lang.Thread thread = new java.lang.Thread(new SocketServer(81));
  thread.start();
 }
}
```

http

- http://www.w3.org/Protocols/

get

```
public boolean doGet(String sURL, String cookie, String referer,
   String charset) {
 boolean doSuccess = false;
 BufferedReader in = null;
 try {
   URL url = new URL(sURL);
   HttpURLConnection URLConn = (HttpURLConnection) url .openConnection();
   URLConn .setRequestProperty( "User-agent",  "Mozilla/5.0 (Windows; U; Windows NT 6.0; zh-TW; rv:1.9.1.2) "
            + "Gecko/20090729 Firefox/3.5.2 GTB5 (.NET CLR 3.5.30729)");
   URLConn .setRequestProperty("Accept", "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8");
   URLConn.setRequestProperty("Accept-Language", "zh-tw,en-us;q=0.7,en;q=0.3");
   URLConn.setRequestProperty("Accept-Charse", "Big5,utf-8;q=0.7,*;q=0.7");
    if (cookie != null)    URLConn.setRequestProperty("Cookie", cookie);
   if (referer != null)    URLConn.setRequestProperty("Referer", referer);
   URLConn.setDoInput(true);
   URLConn.setDoOutput(true);
   URLConn.connect();
   URLConn.getOutputStream().flush();
   in = new BufferedReader(new InputStreamReader(URLConn .getInputStream(), charset));
    String line;
   while ((line = in.readLine()) != null) {
     System.out.println(line);
   }

 } catch (IOException e) {
   doSuccess = false;
   log.out.println(e);
   e.printStackTrace();
 } finally {
   if (in != null) {
     try {
       in.close();
     } catch (java.io.IOException ex) {
       logger.info(ex);
     }
              in = null;
   }
 }
  return doSuccess;
}
```

post

```
public boolean doPost(String sURL, String data, String cookie,
  String referer, String charset) {

 boolean doSuccess = false;
 java.io.BufferedWriter wr = null;
 try {
  URL url = new URL(sURL);
  HttpURLConnection URLConn = (HttpURLConnection) url .openConnection();
   URLConn.setDoOutput(true);
  URLConn.setDoInput(true);
  ((HttpURLConnection) URLConn).setRequestMethod("POST");
  URLConn.setUseCaches(false);
  URLConn.setAllowUserInteraction(true);
  HttpURLConnection.setFollowRedirects(true);
  URLConn.setInstanceFollowRedirects(true);
   URLConn.setRequestProperty( "User-agent",  "Mozilla/5.0 (Windows; U; Windows NT 6.0; zh-TW; rv:1.9.1.2) "
          + "Gecko/20090729 Firefox/3.5.2 GTB5 (.NET CLR 3.5.30729)");
  URLConn .setRequestProperty("Accept", "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8");
  URLConn.setRequestProperty("Accept-Language", "zh-tw,en-us;q=0.7,en;q=0.3");
  URLConn.setRequestProperty("Accept-Charse", "Big5,utf-8;q=0.7,*;q=0.7");
  if (cookie != null)       URLConn.setRequestProperty("Cookie", cookie);
  if (referer != null)    URLConn.setRequestProperty("Referer", referer);
   URLConn.setRequestProperty("Content-Type",  "application/x-www-form-urlencoded");
  URLConn.setRequestProperty("Content-Length", String.valueOf(data  .getBytes().length));
   java.io.DataOutputStream dos = new java.io.DataOutputStream(URLConn .getOutputStream());
  dos.writeBytes(data);
   java.io.BufferedReader rd = new java.io.BufferedReader(
     new java.io.InputStreamReader(URLConn.getInputStream(),
       charset));
  String line;
  while ((line = rd.readLine()) != null) {
   System.out.println(line);
  }
  rd.close();
 } catch (java.io.IOException e) {
  doSuccess = false;
  logger.info(e);
 } finally {
  if (wr != null) {
   try {
    wr.close();
 } catch (java.io.IOException e) {
  doSuccess = false;
  logger.info(e);
 } finally {
  if (wr != null) {
   try {
    wr.close();
   } catch (java.io.IOException ex) {
    logger.info(ex);
   }
   wr = null;
  }
 }

 return doSuccess;
}
```

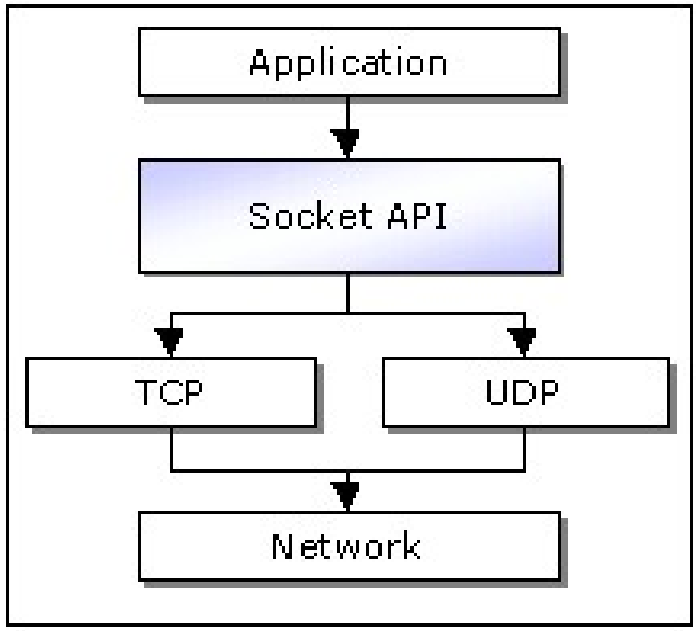
SOCKET PROGRAMMING

Socket Interface

- 一種應用程式介面(API) 介於應用程式與硬體之間，並提供標準的函式以符合不同的網路硬體規格
- 最早的Socket Interface於1980年代由加州柏克萊大學為支援UNIX作業系統上的TCP/IP應用所開發的Socket介面，稱為Berkeley Socket Interface，而其軟體則稱為Berkeley Software Distribution (BSD)

Berkeley Socket

- Berkeley Socket Interface是一組函式，介於網路應用程式與作業系統及網路硬體間，應用程式透過呼叫Socket Interface，發展具有TCP/IP網路功能之應用



Server端BSD API函式

- socket
  - 建立socket
- bind
  - 設定socket所使用的local端IP 位址及通訊埠
- listen
  - 設定socket等候(listen)Client端連結請求
- accept
  - 接受自Client端的連結請求並建立socket連結
- recv
  - 接收來自Client端所傳送的資料 (TCP)
- read
  - 接收來自Client端所傳送的資料 (TCP)

Server端BSD API函式 (cont.)

- recvfrom
  - 接收來自Client端所傳送的資料 (UDP)
- send
  - 傳送資料至Client端 (TCP)
- write
  - 傳送資料至Client端 (TCP)
- sendto
  - 傳送資料至Client端 (UDP)
- closesocket
  - 關閉通訊連結及socket，並且釋放系統資源
- shutdown
  - 關閉socket的傳送及接收功能

Client端BSD API函式

- socket
  - 建立socket
- connect
  - 建立與Server端socket連線
- recv
  - 接收來自Server端所傳送的資料 (TCP)
- read
  - 接收來自Server端所傳送的資料 (TCP)
- recvfrom
  - 接收來自Server端所傳送的資料 (UDP)

Client端BSD API函式 (cont.)

- send
  - 傳送資料至Server端 (TCP)
- write
  - 傳送資料至Server端 (TCP)
- sendto
  - 傳送資料至Server端 (UDP)
- closesocket
  - 關閉通訊連結及socket，並且釋放系統資源
- shutdown
  - 關閉socket的傳送及接收功能

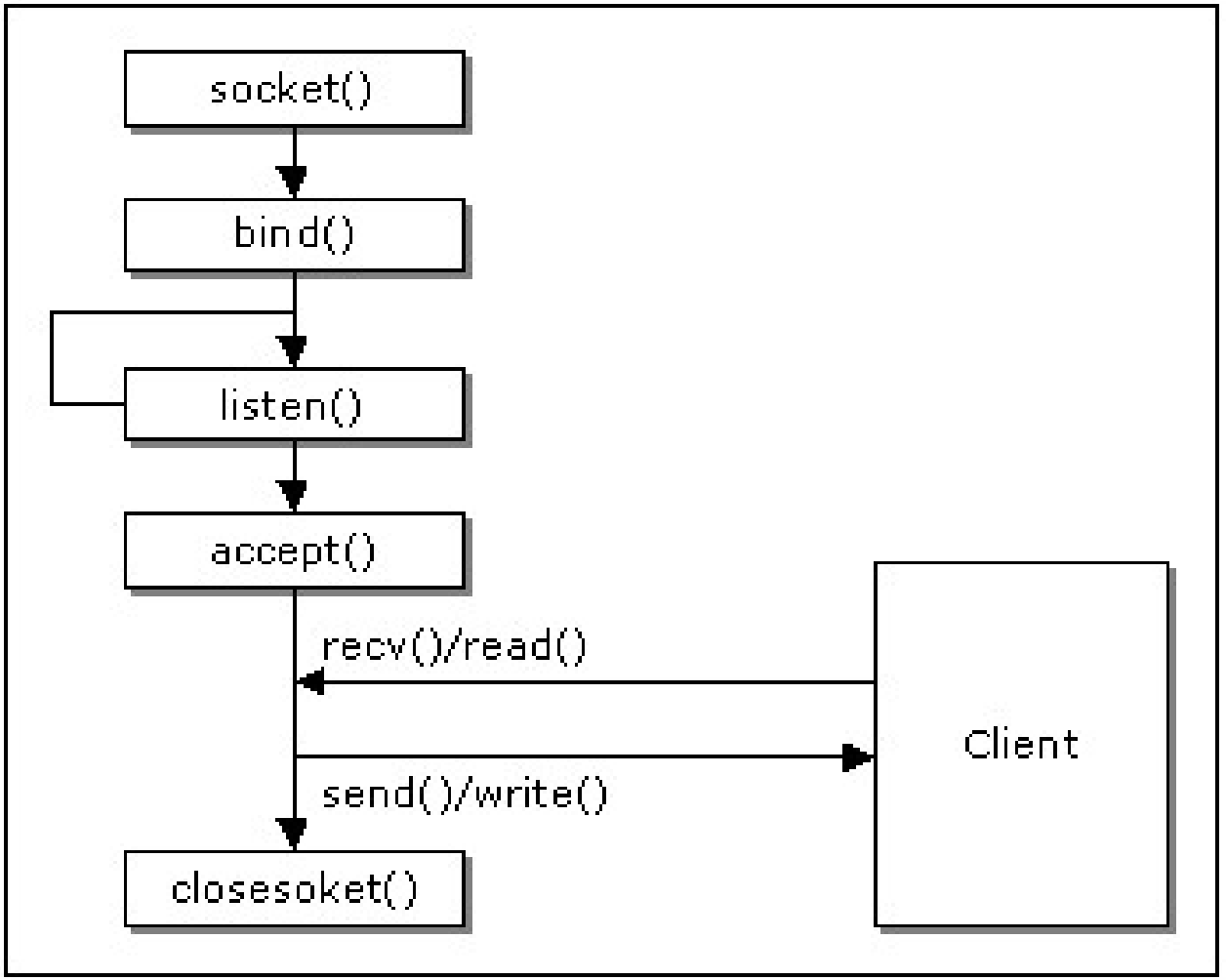
Berkeley Socket網路應用

- Berkeley Socket在網路應用程式開發上，大致有下列兩類
  - Stream Socket（Connection-Oriented Protocol）
  - Datagram Socket（Connectionless Protocol）

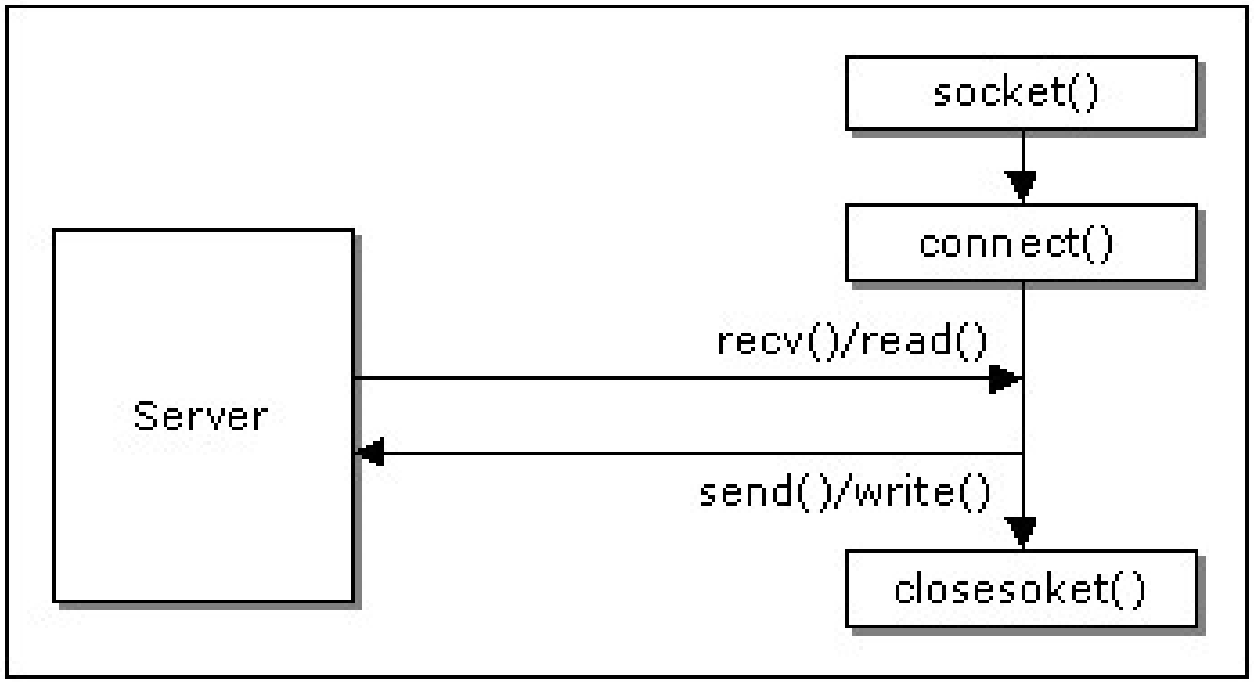
BSD Stream Socket

- 使用TCP傳送資料
- TCP 在傳遞資料之前，會先在主機間建立通訊連結，依據此通訊連結傳遞資料
- 可保證資料無誤送達，且到達順序與送出順序相同

Server端Stream Socket連線流程



Client端Stream Socket連線流程

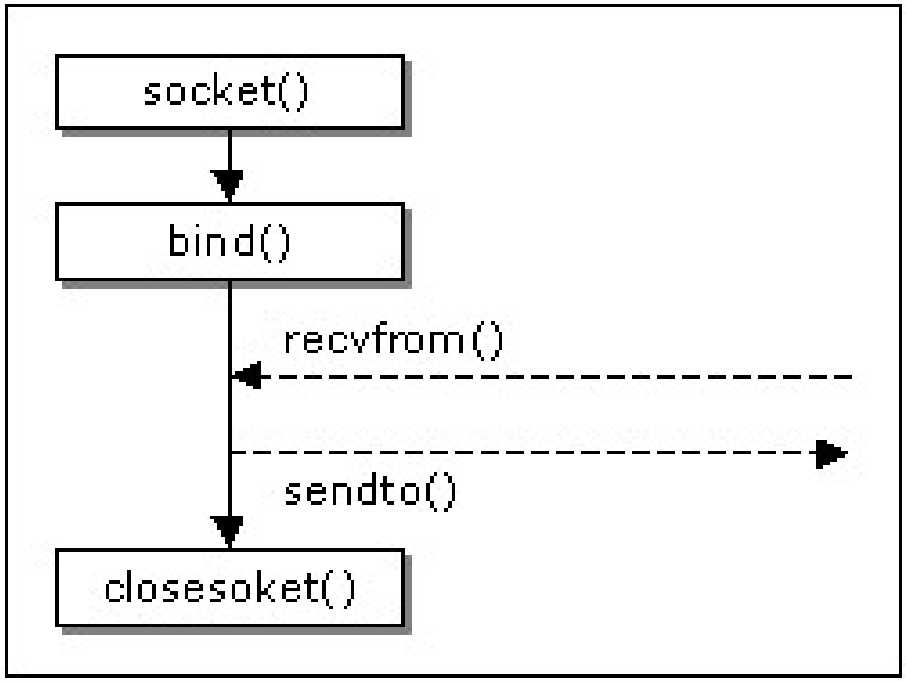


BSD Datagram Socket

- 使用UDP傳送資料
- 與TCP不同的是，TCP在傳遞資料之前須先建立通訊連結，但UDP則不需要，僅須設定電腦間的IP及使用的Port，則可互相傳遞訊息
- UDP不提供資料錯誤的偵測及資料重送等機制，因此並不確保資料能完整送達

Datagram Socket連線流程

- 由於不必建立雙方的連線，因此只需利用sendto及recvfrom傳送及接收資料
- 最後仍利用closesocket關閉Socket並釋放系統資源



Java Socket

- Java支援網路的API
  - java.net (Networking)
  - javax.net.ssl (Networking with Secure Socket Layer)
  - java.rmi (Remote Method Invocation)
  - javax.rmi (Remote Method Invocation for IIOP)

java.net

- 其API分為以下幾類
  - 處理URL（Uniform Resource Locator）
  - 處理主機名稱及IP Address
  - 處理UDP（User Datagram Protocol）通訊協定
  - 處理TCP（Transmission Control Protocol）通訊協定
  - 處理網路認證（Authentication）及權限（Permission）
  - 內容處理器（ContentHandler）

處理URL

- Interface
  - URLStreamHandlerFactory
- Class
  - java.net.HttpURLConnection
  - java.net.JarURLConnection
  - java.net.URL
  - java.net.URLClassLoader
  - java.net.URLConnection
  - java.net.URLDecoder
  - java.net.URLEncoder
  - java.net.URLStreamHandler
- Exception
  - java.net.MalformedURLException
  - java.net.UnknownServiceException

處理主機名稱及IP Address

- Class
  - java.net.InetAddress
- Method
  - public byte[] getAddress()
  - public static InetAddress[] getAllByName(String host) throws UnknownHostException
  - public static InetAddress getByName(String host) throws UnknownHostException
  - public String getHostAddress()
  - public String getHostName()
  - public static InetAddress getLocalHost() throws UnknownHostException
- Exception
  - java.net.UnknownHostException

處理UDP通訊協定

- Interface
  - DatagramSocketImplFactory
- Class
  - java.net.DatagramPacket
  - java.net.DatagramSocket
  - java.net.DatagramSocketImpl
  - java.net.MulticastSocket
- Exception
  - java.net.SocketException

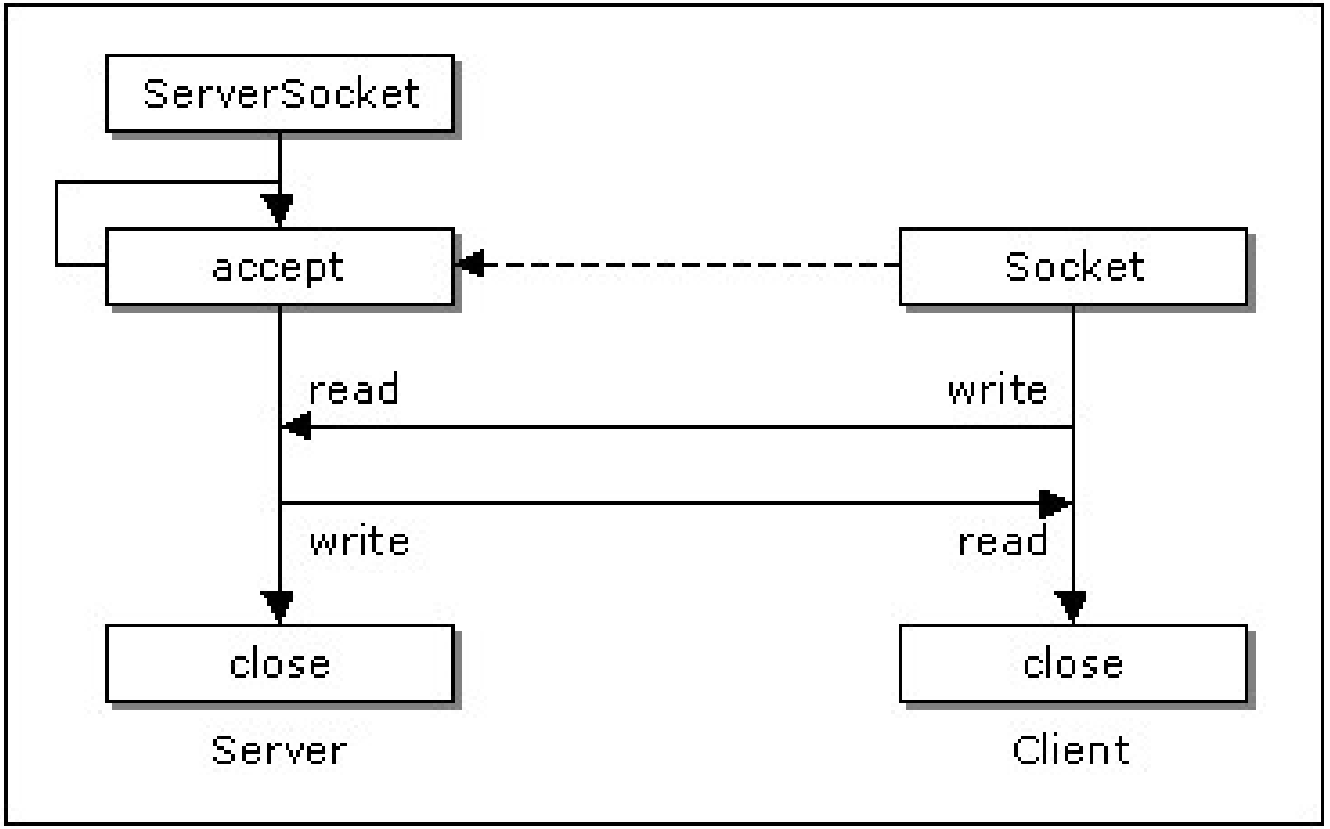
處理TCP通訊協定

- Interface
  - SocketImplFactory
  - SocketOptions
- Class
  - java.net.ServerSocket
  - java.net.Socket
  - java.net.SocketImpl
- Exception
  - java.net.BindException
  - java.net.ConnectException
  - java.net.NoRouteToHostException
  - java.net.ProtocolException
  - java.net.SocketException

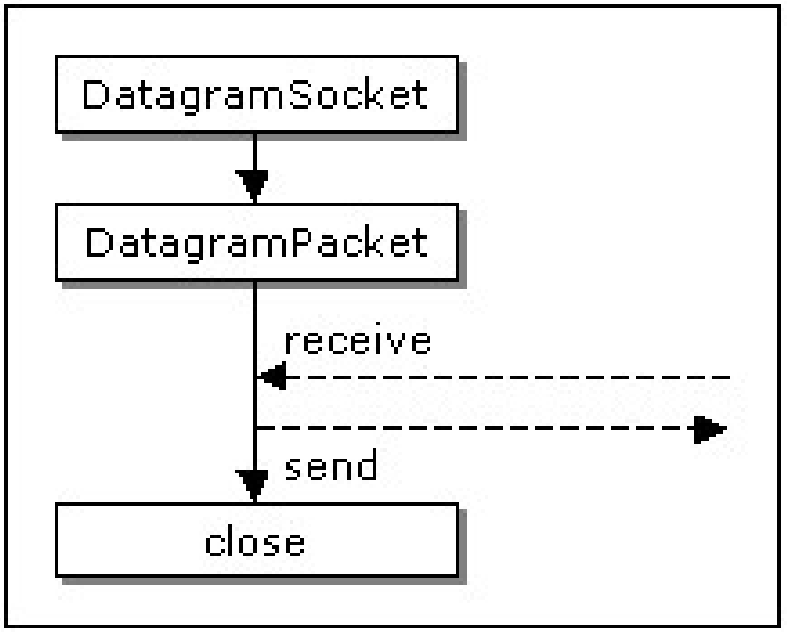
處理網路認證及權限

- Class
  - java.net.Authenticator
  - java.net.NetPermission
  - java.net.PasswordAuthentication
  - java.net.SocketPermission

Java Stream Socket流程



Java Datagram Socket流程



Download html

```
import java.io.*;
import java.net.*;
import java.util.*;

public class app
{
     public static void main(String[] args) throws Exception
   {
          String filename="index.txt";
     String result="";
     URL url = new URL("http://140.138.146.85/index.php");
     DataInputStream in = new DataInputStream(url.openStream());
      RandomAccessFile out = new RandomAccessFile(filename, "rw");
      try
      {
         byte data;
         while(true)
         {
         data = (byte)in.readByte();
         out.writeByte(data);
         System.out.print(new String(new byte[]{data}));

         }
      }
      catch(EOFException e) { }
      System.out.println("檔案下載成功.....");
      in.close();
      out.close();
   }

}
```

JSoup

```
package demo_jsoup;
import java.io.IOException;
import java.net.URL;
import java.util.ArrayList;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.select.Elements;
public class app {
public static void main(String[] args) throws IOException {
		URL url = new URL("http://140.138.146.85/courses/pd105b/wpclassroom/showall.php");
		Document xmlDoc = Jsoup.parse(url, 0);
		Elements node = xmlDoc.select("td");
		ArrayList list = new ArrayList();
		list.add(""+node.get(1).text());
		String res = String.join(" ",list);
		String parts[] = res.split("\\s");
		for (int i = 0; i < parts.length; i++)
				System.out.println(parts[i]);
		}
}
```

Server

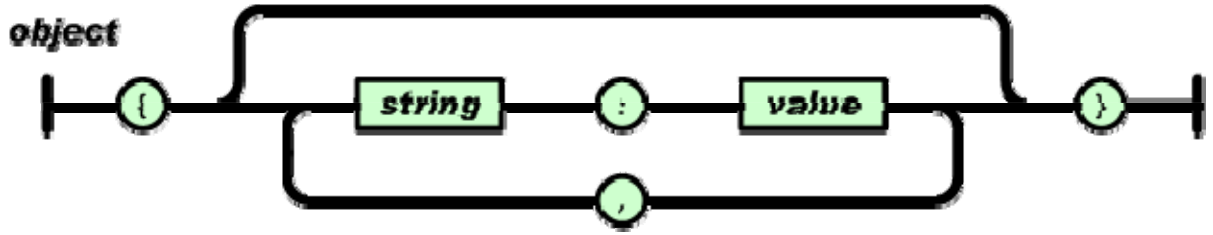
```
import java.io.*;
 import java.net.*;
public class UdpServer {
 int port;
public static void main(String args[]) throws Exception {
 UdpServer server = new UdpServer(5555);
server.run(); ◦
 }
public UdpServer(int pPort) { port = pPort;}
public void run() throws Exception {
                final int SIZE = 8192;
                byte buffer[] = new byte[SIZE];
                for (int count = 0; ; count++) {
                                 DatagramPacket packet = new DatagramPacket(buffer, buffer.length);
                                DatagramSocket socket = new DatagramSocket(port
                                Socket. socket.receive(packet);
                                String msg = new String(buffer, 0, packet.getLength());
                                System.out.println(count+" : receive = "+msg);  socket.close();
} } }
```

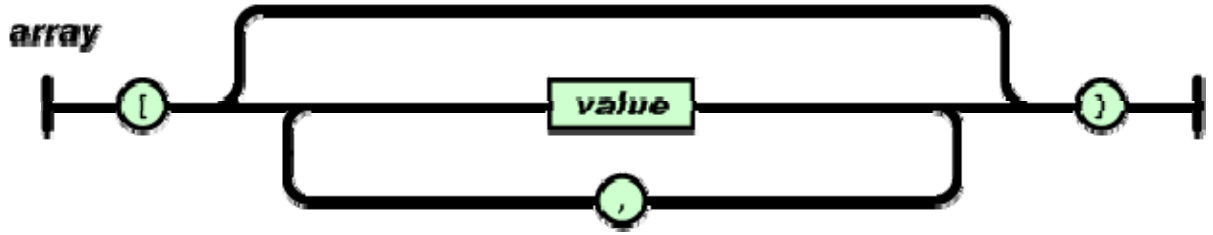
Client

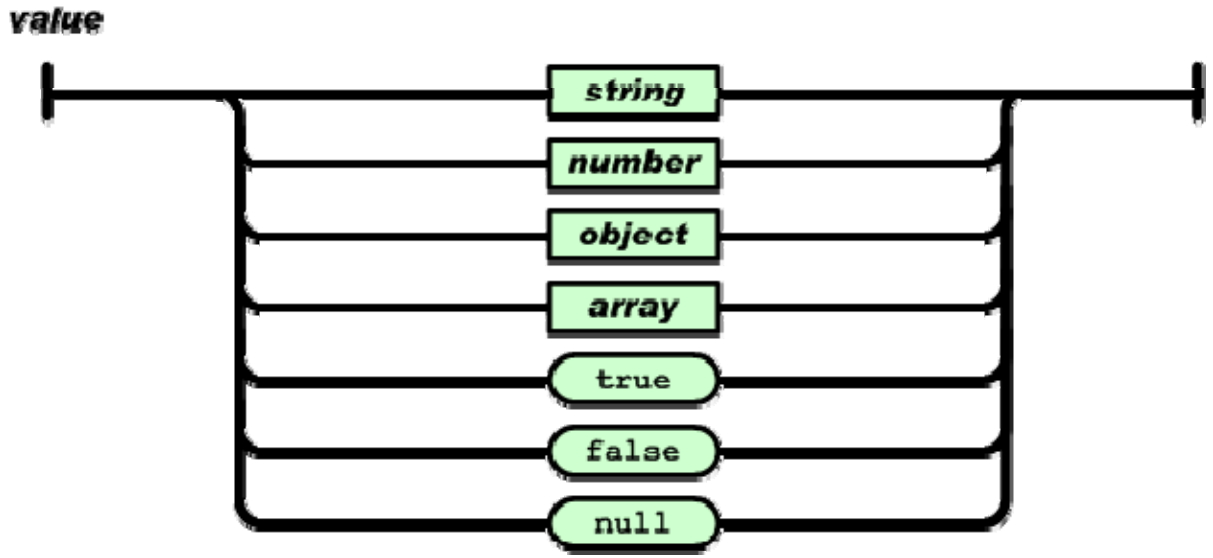
```
import java.io.*;
import java.net.*;
public class UdpClient extends Thread {
                        int port;
                        InetAddress server;
                        String msg;
                        public static void main(String args[]) throws Exception {
                                        for (int i=0; i<100; i++) {
                                                        UdpClient client = new UdpClient(args[0], 5555, "UdpClient : "+i+"th message");
                                                        client.run();
                                        }
                        }
 public UdpClient(String pServer, int pPort, String pMsg) throws Exception {
                        port = pPort;
                        server = InetAddress.getByName(pServer);
                        msg = pMsg;
}
public void run() {
                        try {
                                        byte buffer[] = msg.getBytes();
                                        DatagramPacket packet = new DatagramPacket(buffer, buffer.length, server, port);
                                        DatagramSocket socket = new DatagramSocket();
                                        socket.send(packet);
                                        socket.close();
                        }
catch (Exception e) {
                        e.printStackTrace(); }
}
}
```

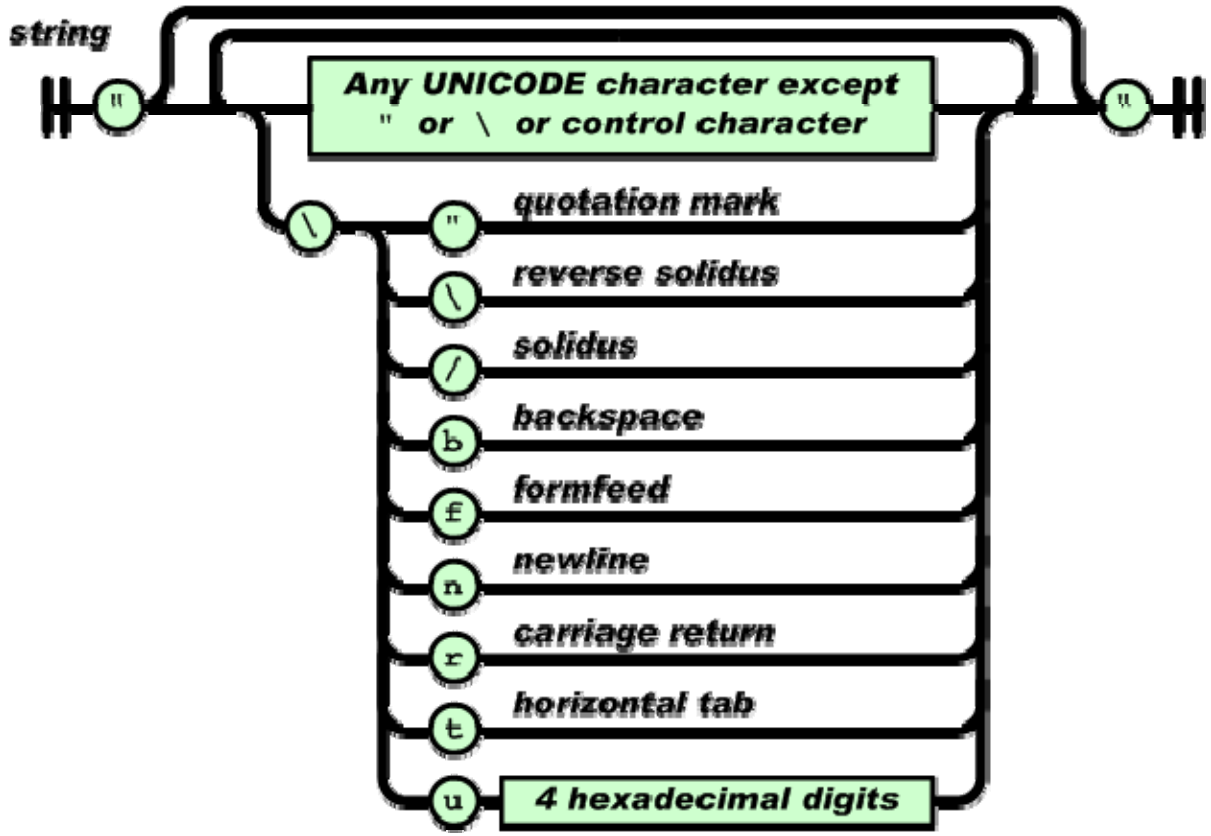
Json

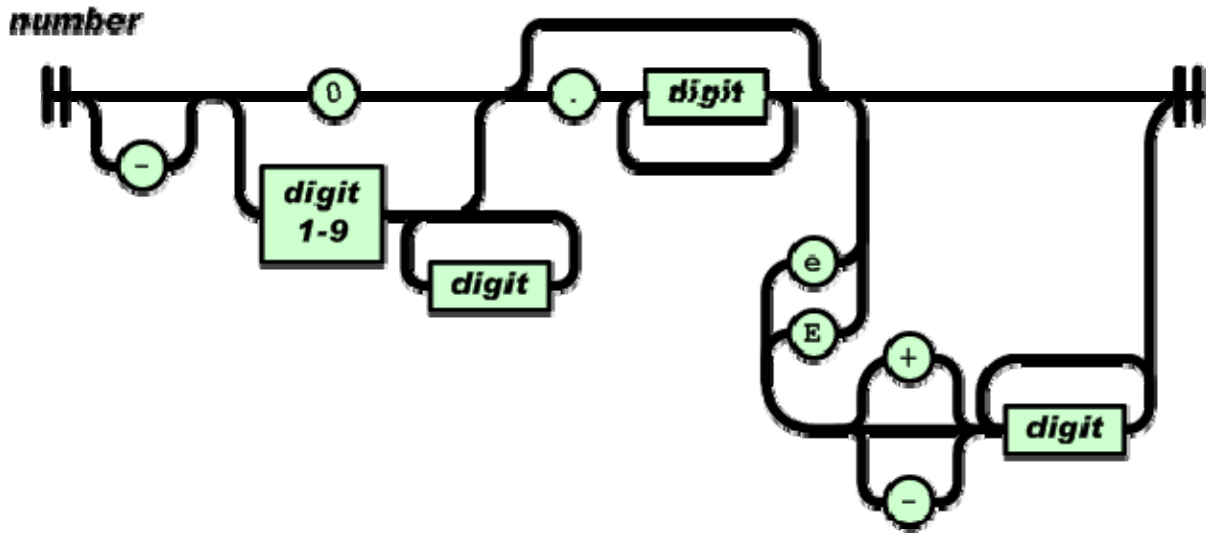
- JSON用於描述資料結構，有以下形式存在。
  - 物件（object）：一個物件以{開始，並以}結束。一個物件包含一系列非排序的名稱／值對，每個名稱／值對之間使用,分割。
  - 名稱／值（collection）：名稱和值之間使用：隔開，一般的形式是：


object
{
string
:
value
}
,


array
[
value
,
]


value
string
number
object
array
true
false
null


string
"
Any UNICODE character except
" or \ or control character
"
\
"
quotation mark
\
reverse solidus
/
solidus
b
backspace
f
formfeed
n
newline
r
carriage return
t
horizontal tab
u
4 hexadecimal digits


number
0
-
digit
1-9
digit
.
digit
e
E
+
-
digit

Json

- https://search.maven.org/#search%7Cgav%7C1%7Cg%3A%22org.json%22%20AND%20a%3A%22json%22

- http://data.coa.gov.tw/Service/OpenData/DataFileService.aspx?UnitId=069

```
package demo_json;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.Reader;
import java.net.URL;
import java.nio.charset.Charset;
import org.json.JSONException;
import org.json.JSONObject;
public class app {
 private static String readAll(Reader rd) throws IOException {
   StringBuilder sb = new StringBuilder();
   int cp;
   while ((cp = rd.read()) != -1) {
    sb.append((char) cp);
   }
   return sb.toString();
 }
 public static JSONObject readJsonFromUrl(String url) throws IOException, JSONException {
  InputStream is = new URL(url).openStream();
  try {
   BufferedReader rd = new BufferedReader(new InputStreamReader(is, Charset.forName("UTF-8")));
   String jsonText = readAll(rd);
   JSONObject json = new JSONObject(jsonText.substring(jsonText.indexOf('{')));
   return json;
  } finally {
   is.close();
  }
 }
 public static void main(String[] args) throws IOException, JSONException {
  JSONObject json = readJsonFromUrl("http://data.coa.gov.tw/Service/OpenData/DataFileService.aspx?UnitId=069&ndctype=JSON&ndcnid=6034");
  System.out.println(json.toString());
  System.out.println(json.get("屆數"));
 }
}
```

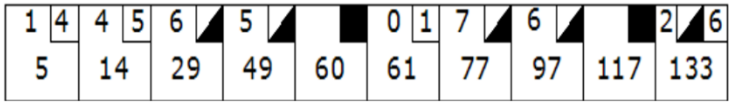
單元測試

JUnit

- 撰寫測試案例請依據下列的步驟：
- 定義一個TestCase的子類別。
- 覆寫setUp()方法以初始化測試中的一個或多個物件。
- 覆寫tearDown()方法以釋放測試中的一個或多個物件。
- 定義一個或多個公開的testXXX()方法；這些方法檢驗這些測試中的物件並且評估期望的結果。
- 定義一個靜態的suite()工廠方法；這個工廠方法構建一個TestSuite其中包含TestCase的所有testXXX()方法。
- 隨意的定義一個main()方法以批次的方式執行TestCase。
-

BOWLING GAME KATA

Scoring Bowling.



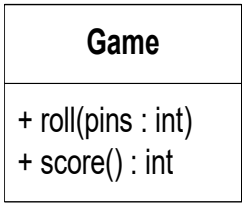
The game consists of 10 frames as shown above. In each frame the player two opportunities to knock down 10 pins. The score for the frame is the total number of pins knocked down, plus bonuses for strikes and spares.

A spare is when the player knocks down all 10 pins in two tries. The bonus that frame is the number of pins knocked down by the next roll. So in frame above, the score is 10 (the total number knocked down) plus a bonus of 5 (the number of pins knocked down on the next roll.)

A strike is when the player knocks down all 10 pins on his first try. The bonus for that frame is the value of the next two balls rolled.

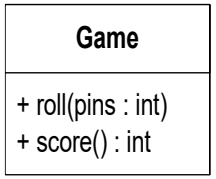
In the tenth frame a player who rolls a spare or strike is allowed to roll the extra balls to complete the frame. However no more than three balls can be rolled in the tenth frame.

The Requirements.



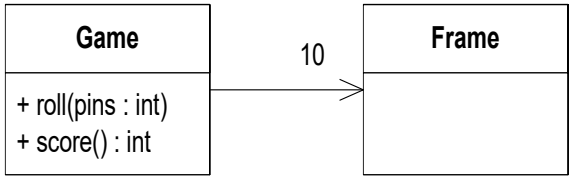
- Write a class named “Game” that has two methods
  - roll(pins : int) is called each time the player rolls a ball. The argument is the number of pins knocked down.
  - score() : int is called only at the very end of the game. It returns the total score for that game.

A quick design session



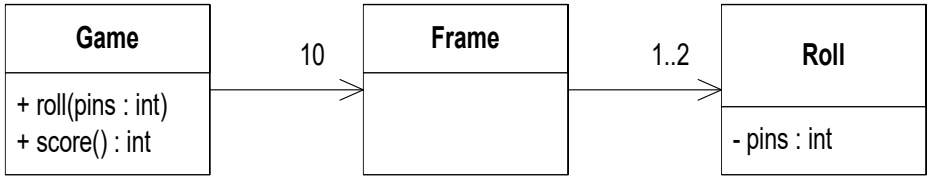
Clearly we need the Game class.

A quick design session



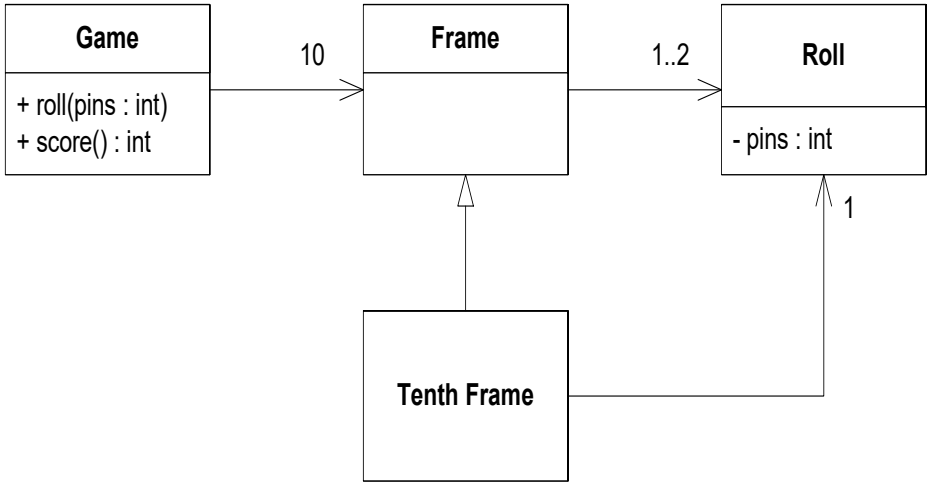
A game has 10 frames.

A quick design session



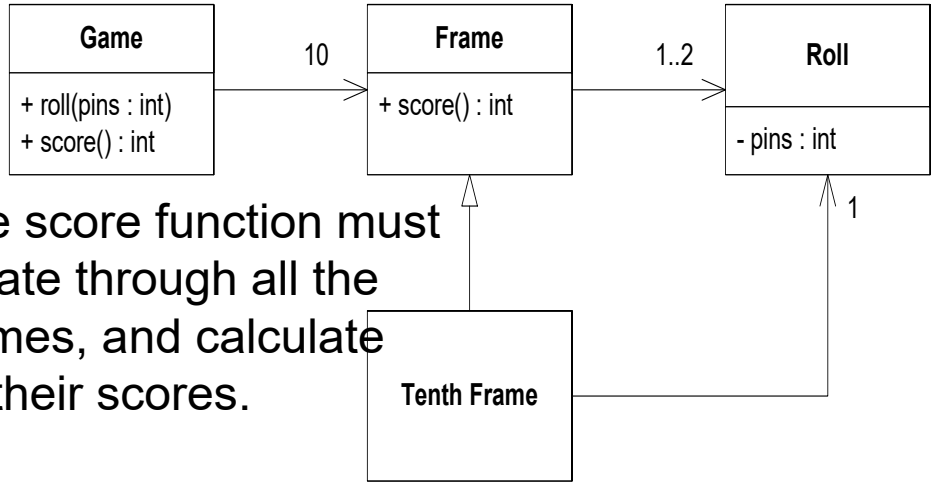
A frame has 1 or two rolls.

A quick design session



The tenth frame has two or three rolls.
It is different from all the other frames.

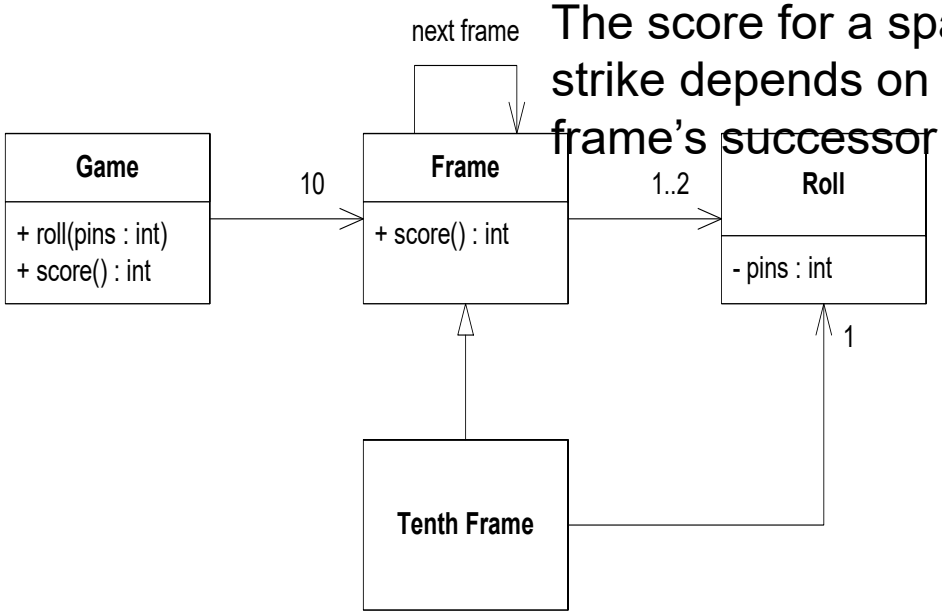
A quick design session



The score function must iterate through all the frames, and calculate all their scores.

A quick design session

The score for a spare or a strike depends on the frame's successor



Begin.

- Create a project named BowlingGame
- Create a unit test named BowlingGameTest

```
import junit.framework.TestCase;

public class BowlingGameTest extends TestCase {
}
```

Begin.

- Create a project named BowlingGame
- Create a unit test named BowlingGameTest

```
import junit.framework.TestCase;

public class BowlingGameTest extends TestCase {
}
```

Execute this program and verify that you get the following error:

```
No tests found in BowlingGameTest
```

The first test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
  }
}
```

The first test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
  }
}
```

```
public class Game {
}
```

The first test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
  }
}
```

```
public class Game {
}
```

The first test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
    for (int i=0; i<20; i++)
      g.roll(0);
  }
}
```

```
public class Game {
}
```

The first test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
    for (int i=0; i<20; i++)
      g.roll(0);
  }
}
```

```
public class Game {
  public void roll(int pins) {
  }
}
```

The first test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
    for (int i=0; i<20; i++)
      g.roll(0);
    assertEquals(0, g.score());
  }
}
```

```
public class Game {
  public void roll(int pins) {
  }
}
```

The first test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
    for (int i=0; i<20; i++)
      g.roll(0);
    assertEquals(0, g.score());
  }
}
```

```
public class Game {
  public void roll(int pins) {
  }

  public int score() {
    return -1;
  }
}
```

```
expected:<0> but was:<-1>
```

The first test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
    for (int i=0; i<20; i++)
      g.roll(0);
    assertEquals(0, g.score());
  }
}
```

```
public class Game {
  public void roll(int pins) {
  }

  public int score() {
    return 0;
  }
}
```

The Second test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
    for (int i = 0; i < 20; i++)
      g.roll(0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    Game g = new Game();
    for (int i = 0; i < 20; i++)
      g.roll(1);
    assertEquals(20, g.score());
  }
}
```

```
public class Game {
  public void roll(int pins) {
  }

  public int score() {
    return 0;
  }
}
```

- Game creation is duplicated
- roll loop is duplicated

The Second test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
    for (int i = 0; i < 20; i++)
      g.roll(0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    Game g = new Game();
    for (int i = 0; i < 20; i++)
      g.roll(1);
    assertEquals(20, g.score());
  }
}
```

```
public class Game {
  public void roll(int pins) {
  }

  public int score() {
    return 0;
  }
}
```

- Game creation is duplicated
- roll loop is duplicated

The Second test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  public void testGutterGame() throws
Exception {
    Game g = new Game();
    for (int i = 0; i < 20; i++)
      g.roll(0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    Game g = new Game();
    for (int i = 0; i < 20; i++)
      g.roll(1);
    assertEquals(20, g.
  }
}
```

```
expected:<20> but was:<0>
```

```
public class Game {
  public void roll(int pins) {
  }

  public int score() {
    return 0;
  }
}
```

- roll loop is duplicated

The Second test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  public void testGutterGame() throws
Exception {
    for (int i = 0; i < 20; i++)
      g.roll(0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    for (int i = 0; i <
      g.roll(1);
    assertEquals(20, g.score());
  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

- roll loop is duplicated

The Second test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  public void testGutterGame() throws
Exception {
    int n = 20;
    int pins = 0;
    for (int i = 0; i < n; i++) {
      g.roll(pins);
    }
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    for (int i = 0; i <
      g.roll(1);
    assertEquals(20, g.score());
  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

- roll loop is duplicated

The Second test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  public void testGutterGame() throws
Exception {
    int n = 20;
    int pins = 0;
    rollMany(n, pins);
    assertEquals(0, g.score());
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testAllOnes() throws
Exception {
    for (int i = 0; i <
      g.roll(1);
    assertEquals(20, g.score());
  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

- roll loop is duplicated

The Second test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testAllOnes() throws
Exception {
    for (int i = 0; i <
      g.roll(1);
    assertEquals(20, g.score());
  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

- roll loop is duplicated

The Second test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testAllOn
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

The Second test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOn
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

```
expected:<16> but was:<13>
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

tempted to use flag to remember previous roll. So design must be wrong.

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

roll() calculates score, but name does not imply that.

score() does not calculate score, but name implies that it does.

Design is wrong.
Responsibilities are misplaced.

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

//  public void testOneSpare() throws
Exception {
//    g.roll(5);
//    g.roll(5); // spa
//    g.roll(3);
//    rollMany(17,0);
//    assertEquals(16,g.score());
//  }
}
```

```
public class Game {
  private int score = 0;

  public void roll(int pins) {
    score += pins;
  }

  public int score() {
    return score;
  }
}
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

//  public void testOneSpare() throws
Exception {
//    g.roll(5);
//    g.roll(5); // spa
//    g.roll(3);
//    rollMany(17,0);
//    assertEquals(16,g.score());
//  }
}
```

```
public class Game {
  private int score = 0;
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    score += pins;
    rolls[currentRoll++] = pins;
  }

  public int score() {
    return score;
  }
}
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

//  public void testOneSpare() throws
Exception {
//    g.roll(5);
//    g.roll(5); // spa
//    g.roll(3);
//    rollMany(17,0);
//    assertEquals(16,g.score());
//  }
}
```

```
public class Game {
  private int score = 0;
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    score += pins;
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    for (int i = 0; i < rolls.length; i++)
      score += rolls[i];
    return score;
  }
}
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

//  public void testOneSpare() throws
Exception {
//    g.roll(5);
//    g.roll(5); // spa
//    g.roll(3);
//    rollMany(17,0);
//    assertEquals(16,g.score());
//  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    for (int i = 0; i < rolls.length; i++)
      score += rolls[i];
    return score;
  }
}
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    for (int i = 0; i < rolls.length; i++)
      score += rolls[i];
    return score;
  }
}
```

```
expected:<16> but was:<13>
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    for (int i = 0; i < rolls.length; i++)
{
      if (rolls[i] + rolls[i+1] == 10) //
spare
        score += ...
      score += rolls[i];
    }
    return score;
  }
}
```

This isn't going to work because i might not refer to the first ball of the frame.

Design is still wrong.

Need to walk through array two balls (one frame) at a time.

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

//  public void testOneSpare() throws
Exception {
//    g.roll(5);
//    g.roll(5); // spa
//    g.roll(3);
//    rollMany(17,0);
//    assertEquals(16,g.score());
//  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    for (int i = 0; i < rolls.length; i++)
      score += rolls[i];
    return score;
  }
}
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

//  public void testOneSpare() throws
Exception {
//    g.roll(5);
//    g.roll(5); // spa
//    g.roll(3);
//    rollMany(17,0);
//    assertEquals(16,g.score());
//  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int i = 0;
    for (int frame = 0; frame < 10;
frame++)  {
      score += rolls[i] + rolls[i+1];
      i += 2;
    }
    return score;
  }
}
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int i = 0;
    for (int frame = 0; frame < 10;
frame++)  {
      score += rolls[i] + rolls[i+1];
      i += 2;
    }
    return score;
  }
}
```

```
expected:<16> but was:<13>
```

- ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int i = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (rolls[i] + rolls[i + 1] == 10)
// spare
      {
        score += 10 + rolls[i + 2];
        i += 2;
      } else {
        score += rolls[i] + rolls[i + 1];
        i += 2;
      }
    }
    return score;
  }
}
```

-ugly comment in test.
-ugly comment in conditional.
-i is a bad name for this variable

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int i = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (rolls[i] + rolls[i + 1] == 10)
// spare
      {
        score += 10 + rolls[i + 2];
        i += 2;
      } else {
        score += rolls[i] + rolls[i + 1];
        i += 2;
      }
    }
    return score;
  }
}
```

-ugly comment in test.
-ugly comment in conditional.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int frameIndex = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (rolls[frameIndex] +
          rolls[frameIndex + 1] == 10) //
spare
      {
        score += 10 + rolls[frameIndex +
2];
        frameIndex += 2;
      } else {
        score += rolls[frameIndex] +
                 rolls[frameIndex + 1];
        frameIndex += 2;
      }
    }
    return score;
  }
}
```

-ugly comment in test.

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    g.roll(5);
    g.roll(5); // spare
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int frameIndex = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (isSpare(frameIndex))
      {
        score += 10 + rolls[frameIndex +
2];
        frameIndex += 2;
      } else {
        score += rolls[frameIndex] +
                 rolls[frameIndex + 1];
        frameIndex += 2;
      }
    }
    return score;
  }

  private boolean isSpare(int frameIndex)
{
    return rolls[frameIndex] +
           rolls[frameIndex + 1] == 10;
```

The Third test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  private Game g;

  protected void setUp() throws Exception
{
    g = new Game();
  }

  private void rollMany(int n, int pins)
{
    for (int i = 0; i < n; i++)
      g.roll(pins);
  }

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    rollSpare();
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }

  private void rollSpare() {
    g.roll(5);
    g.roll(5);
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int frameIndex = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (isSpare(frameIndex))
      {
        score += 10 + rolls[frameIndex +
2];
        frameIndex += 2;
      } else {
        score += rolls[frameIndex] +
                 rolls[frameIndex + 1];
        frameIndex += 2;
      }
    }
    return score;
  }

  private boolean isSpare(int frameIndex)
{
    return rolls[frameIndex] +
           rolls[frameIndex + 1] == 10;
```

- ugly comment in testOneStrike.

The Fourth test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  ...

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    rollSpare();
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }

  public void testOneStrike() throws
Exception {
    g.roll(10); // strike
    g.roll(3);
    g.roll(4);
    rollMany(16, 0);
    assertEquals(24, g.score());
  }

  private void rollSpar
    g.roll(5);
    g.roll(5);
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int frameIndex = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (isSpare(frameIndex))
      {
        score += 10 + rolls[frameIndex +
2];
        frameIndex += 2;
      } else {
        score += rolls[frameIndex] +
                 rolls[frameIndex + 1];
        frameIndex += 2;
      }
    }
    return score;
  }

  private boolean isSpare(int frameIndex)
{
    return rolls[frameIndex] +
           rolls[frameIndex + 1] == 10;
  }
}
```

```
expected:<24> but was:<17>
```

-ugly comment in testOneStrike.
-ugly comment in conditional.
-ugly expressions.

The Fourth test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  ...

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    rollSpare();
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }

  public void testOneStrike() throws
Exception {
    g.roll(10); // strike
    g.roll(3);
    g.roll(4);
    rollMany(16, 0);
    assertEquals(24, g.score());
  }

  private void rollSpare() {
    g.roll(5);
    g.roll(5);
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int frameIndex = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (rolls[frameIndex] == 10) //
strike
      {
        score += 10 +
                 rolls[frameIndex+1] +
                 rolls[frameIndex+2];
        frameIndex++;
      }
      else if (isSpare(frameIndex))
      {
        score += 10 + rolls[frameIndex +
2];
        frameIndex += 2;
      } else {
        score += rolls[frameIndex] +
                 rolls[frameIndex + 1];
        frameIndex += 2;
      }
    }
    return score;
  }

  private boolean isSpare(int frameIndex)
{
                                  Index] +
                                  Index + 1] == 10;
  }
```

-ugly comment in testOneStrike.
-ugly comment in conditional.

The Fourth test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  ...

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    rollSpare();
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }

  public void testOneStrike() throws
Exception {
    g.roll(10); // strike
    g.roll(3);
    g.roll(4);
    rollMany(16, 0);
    assertEquals(24, g.score());
  }

  private void rollSpare() {
    g.roll(5);
    g.roll(5);
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int frameIndex = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (rolls[frameIndex] == 10) //
strike
      {
        score += 10 +
strikeBonus(frameIndex);
        frameIndex++;
      } else if (isSpare(frameIndex)) {
        score += 10 +
spareBonus(frameIndex);
        frameIndex += 2;
      } else {
        score +=
sumOfBallsInFrame(frameIndex);
        frameIndex += 2;
      }
    }
    return score;
  }

  private int sumOfBallsInFrame(int
frameIndex) {
    return
rolls[frameIndex]+rolls[frameIndex+1];
  }

  private int spareBonus(int frameIndex)
{
    return rolls[frameIndex + 2];
  }

                              nus(int frameIndex)
    return
```

-ugly comment in testOneStrike.

The Fourth test.

```
import junit.framework.TestCase;

public class BowlingGameTest extends
TestCase {
  ...

  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    rollSpare();
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }

  public void testOneStrike() throws
Exception {
    g.roll(10); // strike
    g.roll(3);
    g.roll(4);
    rollMany(16, 0);
    assertEquals(24, g.score());
  }

  private void rollSpare() {
    g.roll(5);
    g.roll(5);
  }
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int frameIndex = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (isStrike(frameIndex)) {
        score += 10 +
strikeBonus(frameIndex);
        frameIndex++;
      } else if (isSpare(frameIndex)) {
        score += 10 +
spareBonus(frameIndex);
        frameIndex += 2;
      } else {
        score +=
sumOfBallsInFrame(frameIndex);
        frameIndex += 2;
      }
    }
    return score;
  }

  private boolean isStrike(int frameIndex)
{
    return rolls[frameIndex] == 10;
  }

  private int sumOfBallsInFrame(int
frameIndex) {
    return rolls[frameIndex] +
rolls[frameIndex+1];
  }

  private int spareBonus(int frameIndex)
{
    return rolls[frameIndex+2];
  }
```

The Fourth test.

```
  ...
  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    rollSpare();
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }

  public void testOneStrike() throws
Exception {
    rollStrike();
    g.roll(3);
    g.roll(4);
    rollMany(16, 0);
    assertEquals(24, g.score());
  }

  private void rollStrike() {
    g.roll(10);
  }

  private void rollSpare() {
    g.roll(5);
    g.roll(5);
  }
}
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int frameIndex = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (isStrike(frameIndex)) {
        score += 10 +
strikeBonus(frameIndex);
        frameIndex++;
      } else if (isSpare(frameIndex)) {
        score += 10 +
spareBonus(frameIndex);
        frameIndex += 2;
      } else {
        score +=
sumOfBallsInFrame(frameIndex);
        frameIndex += 2;
      }
    }
    return score;
  }

  private boolean isStrike(int frameIndex)
{
    return rolls[frameIndex] == 10;
  }

  private int sumOfBallsInFrame(int
frameIndex) {
    return rolls[frameIndex] +
rolls[frameIndex+1];
  }

  private int spareBonus(int frameIndex)
{
    return rolls[frameIndex+2];
  }
```

The Fifth test.

```
  ...
  public void testGutterGame() throws
Exception {
    rollMany(20, 0);
    assertEquals(0, g.score());
  }

  public void testAllOnes() throws
Exception {
    rollMany(20,1);
    assertEquals(20, g.score());
  }

  public void testOneSpare() throws
Exception {
    rollSpare();
    g.roll(3);
    rollMany(17,0);
    assertEquals(16,g.score());
  }

  public void testOneStrike() throws
Exception {
    rollStrike();
    g.roll(3);
    g.roll(4);
    rollMany(16, 0);
    assertEquals(24, g.score());
  }

  public void testPerfectGame() throws
Exception {
    rollMany(12,10);
    assertEquals(300, g.score());
  }

  private void rollStrike() {
    g.roll(10);
  }

  private void rollSpare() {
```

```
public class Game {
  private int rolls[] = new int[21];
  private int currentRoll = 0;

  public void roll(int pins) {
    rolls[currentRoll++] = pins;
  }

  public int score() {
    int score = 0;
    int frameIndex = 0;
    for (int frame = 0; frame < 10;
frame++) {
      if (isStrike(frameIndex)) {
        score += 10 +
strikeBonus(frameIndex);
        frameIndex++;
      } else if (isSpare(frameIndex)) {
        score += 10 +
spareBonus(frameIndex);
        frameIndex += 2;
      } else {
        score +=
sumOfBallsInFrame(frameIndex);
        frameIndex += 2;
      }
    }
    return score;
  }

  private boolean isStrike(int frameIndex)
{
    return rolls[frameIndex] == 10;
  }

  private int sumOfBallsInFrame(int
frameIndex) {
    return rolls[frameIndex] +
rolls[frameIndex+1];
  }

  private int spareBonus(int frameIndex)
{
    return rolls[frameIndex+2];
  }
```