

Lecture Note – Supplementary Java and JSP

By Gabriel Fung, PhD
School of Information Technology and Electrical Engineering
The University Of Queensland

Java Style (1/3)

- We usually write Java programs in Package:
 - That means we are trying to create some folders to better maintain the files
 - Just like the files in your computer, you will NOT put ALL files in the whole computer in JUST ONE directory, right? You will have many many many different folders.
 - Similarly, we want to put different files in different directories.
 - For example, we have the following directory structure:
 - root
 - `infs3202`
 - » `practical04`
 - » `practical05`
 - » `practical06`
 - `myWeb`
 - » `database`
 - » `blog`

Java Style (2/3)

- Suppose we want to write a Java program called

`CalculateBean.java`

in the directory:

`root/infs3202/practical04`

- Then, we need to include the following statement in

`Calculate.java` (the first sentence):

```
package infs3202.practical04;
import java.util.*;
public class CalculateBean{
    :
    :
}
```

Java Style (3/3)

- Some common Java style that you should follow (not necessary, but all professional will write in this way)
 - All package name start with a lower case letter:
 - Good: `package practical;`
 - Bad: `package Practical;`
 - All class name start with an upper case letter:
 - Good: `public class CalculateBean{`
 - Bad: `public class calculateBean{`

Compile Java Program (1/3)

- In order to compile a Java program:
 - Download the latest JSDK:
 - <http://java.sun.com/javase/downloads/index.jsp>
 - Now compile your Java programs:
 - Click “Start > Run”
 - Type:
 - » cmd
 - Go to your directory that has the Java program
 - Suppose your Java program is in:
 - » H:\Tomcat\webapps\WEB-INF\classes\practical\Calculate.java
 - Then, type:
 - » cd H:\Tomcat\webapps\WEB-INF\classes\practical\
 - Compile the java program
 - Type the following after you go to your destination directory:
 - » javac CalculateBean.java
 - A file named `CalculateBean.class` should be appeared.

Compile Java Program (2/3)

- If you got the following error message:
 - 'javac' is not recognized as an internal or external command, operable program or batch file.
- To solve this problem:
 - **IF YOU HAVE ADMIN RIGHTS, then do the following only once in the same computer in its life time**
 - If Java is installed in C:\Program Files\Java\jdk1.6.0_04
 - There should be a directory called “bin”
 - Include this directory in your system path:
 - Go to “Control Panel”
 - Double click “System”
 - Go to “Advanced” panel
 - Click the button “Environment Variables”
 - Select “Path” in the “User variables for ...” box
 - Click the button “Edit”
 - Add this in the beginning (do not erase the existing stuff):
 - » C:\Program Files\Java\jdk1.6.0_04\bin;

Compile Java Program (3/3)

- **IF YOU DO NOT HAVE ADMIN RIGHTS, then do the following every time you start a new cmd window:**
 - If Java is installed in C:\Program Files\Java\jdk1.6.0_04
 - There should be a directory called “bin”
 - Then, in the command prompt window (i.e. the window that appeared when you type “cmd”), type:
 - `path=%path%C:\Program Files\Java\jdk1.6.0_04\bin;`

Some Useful Utilities (1/3)

- There are many utilities in Java that are very useful. I just listed some of them.
 - Covert a string to integer/double/float/boolean/byte/...

```
int i = Integer.parseInt(str);
double d = Double.parseDouble(str);
float f = Float.parseFloat(str);
:
:
```

- Reading the content of a file:

```
BufferedReader reader = new BufferedReader(new
    FileReader("xxx.txt"));
for(String str; (str=reader.readLine())!=null); ){
    // do sth e.g.:
    // System.out.println(str);
}
```

Some Useful Utilities (2/3)

– Tokenizer a string

```
String str = "Here are some texts."  
StringTokenizer strTok = new StringTokenizer(str);  
while(strTok.hasMoreTokens()) {  
    System.out.println(strTok.nextToken());  
}
```

```
// Output:
```

```
Here
```

```
are
```

```
some
```

```
texts.
```

Some Useful Utilities (3/3)

– Getting all information in a hashtable

```
Hashtable t = new Hashtable();
t.put("1", "hello");
t.put("2", "yes");
t.put("3", "no");
:
:
for (Map.Entry e : t.entrySet()) {
    System.out.print("key: " + e.getKey() + " ");
    System.out.println("value: " + e.getValue());
}
```

- A complete reference:

- <http://java.sun.com/javase/6/docs/api/>

Some Notes

- Always remember – Java is **static typing**.
 - This means – you need to define everything clearly!
 - You cannot mix-up the type of variables and their methods.
 - **The followings are all WRONG** (think about why. Ask me or the tutors if you are not sure):
 - ```
double d = 0;
if(d.equals("0")) {
 // do sth
}
```
    - ```
String d = "0";
if(d == 0) {
    // do sth
}
```
 - ```
String d = "0";
if(d == "0") {
 // do sth
}
```

# Where to Put the Bean Files in JSP?

- You should put all Java Bean in this directory (create one if you do not have it):
  - `Tomcat\webapps\YOUR_FOLDER\WEB-INF\classes`
- Note:
  - You must maintain the directory/package structure.
  - For example, the “root” directory in P.2 is equivalent to the directory “classes” in the Tomcat situation:
    - `Tomcat\webapps\YOUR_FOLDER\WEB-INF\classes`
      - `infs3202`
        - » `practical04`
        - » `Practical05`
        - » `:`

# Jar (1/2)

- Jar is a tool to zip all java class files. It is very useful.
  - Suppose we have a directory structure same as P.2:
    - Tomcat\webapps\YOUR\_FOLDER\WEB-INF\classes
      - infs3202
        - » practical04
        - » practical05
        - » practical06
  - Then, go to:
    - Tomcat\webapps\YOUR\_FOLDER\WEB-INF\classes
  - Type:
    - `jav -cvf infs3202.jar infs3202/`
  - And all files will be zipped into one single file only.
  - To unzip it:
    - `jav -xvf infs3202.jar`

# Jar (2/2)

- Why it is useful?
  - Because the size is much smaller! So we can more easy to bring all files to other platform!
- We can include just one .jar file in the following directory:  
`... \Tomcat\webapps\YOUR_FOLDER\WEB-INF\lib`
- And the result will be exactly the same as you put the files in multiple directory.
  - Same efficiency.
  - You will see a demo in Lecture 07.

# A Final Issue About JSP

- Sometimes, if you got the following error message:

- The value for the useBean class attribute XXXX is invalid

And you **guarantee** that all paths are correct, no typos, no bugs, etc. Then, you may try the following two things:

1. Stop Tomcat and Re-start it again.
2. Add the following sentence in your Java Bean:

```
package infs3202.practical04;
```

```
public class YourBean implements java.io.Serializable{
 :
 :
}
```

This sometimes **may** help.