Why Have Code Conventions

Code conventions are important to programmers for a number of reasons:

• Help yourself with analyzing questions, writing logical code
• Faster and easier debug, minimize code complexity
• Other programmers can understand your code, it's a universal language, makes you professional

How to Write Universal Language

Code Comments
• Comments are used to explain the code, the compiler will skip comments
• Single-line comments start with two forward slashes (//).

```java
// This is a comment
System.out.println("Hello World");
```

• Multi-line comments start with /* and ends with */. Any text between /* and */ will be ignored by Java.

```java
/* The code below will print the words Hello World
to the screen, and it is amazing */
System.out.println("Hello World");
```

Use of Indentation
• Do not write long line. Maximumly 69 characters (you don't need to count one by one).
• Bad example

```java
public static void execute(String command) throws SyntacticException, SemanticException {
    commandHandler.reinitialise();
    if (command == null ||command.trim().length() == 0) {
        System.out.println() throw new SyntacticException(
            "Empty command", commandHandler.getIdentifier());
    }
    // more code...
}
```

• Good example

```java
public static void execute(String command)
    throws SyntacticException, SemanticException
{
    commandHandler.reinitialise();
    if (command == null ||
        command.trim().length() == 0)
    {
        System.out.println()
        throw new SyntacticException(
            "Empty command",
            commandHandler.getIdentifier());
    }
    // more code...
}
```

Avoid Commenting on Obvious Things
• Don't restate code

```java
// Print Hello World
System.out.println("Hello World");
```

Proper and Consistent Scheme for Naming
• Variable names should make sense, usually lower case
• If the variable name includes more than 1 word, capitalize the first letter of the words except the first one. Some basic example can be

```java
int count;
int pastScore;
String username;
String password;
String groupName;
```

Deep nesting structure should be avoided
• If you have some structure like this, you probably need to rethink about the conditions and simplify the logic.

```java
if(){
    if(){
        if(){

        }
        else if()
        {

        }
        else{

        }

    }
    else{

    }
}
```