1. **What is Debugging?**

**Debugging** is the routine process of locating and removing bugs, errors or abnormalities from programs. It’s a must have skill for any Java developer because it helps to find subtle bug that are not visible during code reviews or that only happens when a specific condition occurs.

1. **What are the default packages present in java?**

**Java compiler imports java.lang package internally by default**. It provides the fundamental classes that are necessary to design a basic [Java program](https://www.javatpoint.com/java-programs). The important classes are Object, which is the root of the class hierarchy, and Class, instances of which represent classes at run time.

1. **Can we have an else condition without if condition?**

No. You will get an error. error: 'else' without 'if'.

1. **Other than Boolean can we use anything in if condition?**

No, if (condition) statement; if is a Java reserved word **The condition must be a Boolean expression**. It must evaluate to either true or false. If the condition is true, the statement is executed.

1. **Why break is imp in switch case?**

The case statements in a switch statement are simply labels. When you switch on a value, the switch statement essentially does a go to the label with the matching value. This means that the break is necessary **to avoid passing through to the code under the next label**.

1. **Java Naming Convention**

Java naming convention is a rule to follow as you decide what to name your identifiers such as class, package, variable, constant, method, etc.

But it is not forced to follow. So, it is known as convention not rule. These conventions are suggested by several Java communities such as Sun Microsystems and Netscape.

All the classes, interfaces, packages, methods and fields of Java programming language are given according to the Java naming convention. If you fail to follow these conventions, it may generate confusion or erroneous code.

Advantage of Naming Conventions in Java

By using standard Java naming conventions, you make your code easier to read for yourself and other programmers. Readability of Java program is very important. It indicates that less time is spent to figure out what the code does.

Naming Conventions of the Different Identifiers

The following table shows the popular conventions used for the different identifiers.

|  |  |  |
| --- | --- | --- |
| **Identifiers Type** | **Naming Rules** | **Examples** |
| Class | It should start with the uppercase letter. It should be a noun such as Color, Button, System, Thread, etc. Use appropriate words, instead of acronyms. | public class **Employee** { //code snippet } |
| Interface | It should start with the uppercase letter. It should be an adjective such as Runnable, Remote, ActionListener. Use appropriate words, instead of acronyms. | interface **Printable** { //code snippet } |
| Method | It should start with lowercase letter. It should be a verb such as main(), print(), println(). If the name contains multiple words, start it with a lowercase letter followed by an uppercase letter such as actionPerformed(). | class Employee { // method void **draw()** {//code snippet} } |
| Variable | It should start with a lowercase letter such as id, name. It should not start with the special characters like & (ampersand), $ (dollar), \_ (underscore). If the name contains multiple words, start it with the lowercase letter followed by an uppercase letter such as firstName, lastName. Avoid using one-character variables such as x, y, z. | class Employee { // variable int **id**; //code snippet } |
| Package | It should be a lowercase letter such as java, lang. If the name contains multiple words, it should be separated by dots (.) such as java.util, java.lang. | //package package **com.javatpoint;** class Employee { //code snippet } |
| Constant | It should be in uppercase letters such as RED, YELLOW. If the name contains multiple words, it should be separated by an underscore(\_) such as MAX\_PRIORITY. It may contain digits but not as the first letter. | class Employee { //constant static final int **MIN\_AGE** = 18; //code snippet } |

**CamelCase in Java naming conventions**

Java follows camel-case syntax for naming the class, interface, method, and variable. If the name is combined with two words, the second word will start with uppercase letter always such as actionPerformed(), firstName, ActionEvent, ActionListener, etc.