Syntax Basics

Java syntax is the set of rules that define how a Java program is written and interpreted by the compiler.

- **Class Names:** By convention, class names should start with an **uppercase letter.** If several words are used to form the name of the class, each inner word's first letter should be in Upper Case (e.g., **MyFirstJavaClass**).
- **PascalCase or Upper CamelCase.**

- **Method Names:** All method names should start with a **lowercase letter**. If several words are used to form the name of the method, then each inner word's first letter should be in Upper Case (e.g., **myMethodName()**).

camelCase

- **File Name:** The name of the program file should exactly **match the class name** with the **.java** extension. For example, if the class name is **MyFirstJavaClass**, the file should be saved as **MyFirstJavaClass.java**.

- **Main Method Entry Point:** Every Java application must have a **main()** method as the entry point. The signature of this method looks like: **public static void main(String[] args)**

Naming Conventions for Identifiers:

Identifiers are the names given to classes, methods, variables, and other entities in Java code to identify them. Some conventions include:

- **Variables:** Start with a **lowercase letter** and use **camelCase** for compound names (e.g., **myVariable**, **studentAge**).
- **Constants:** Typically, all uppercase **using underscore** to separate words (e.g., **MAX_HEIGHT**, **TOTAL_COUNT**).
-
```
public class Main {
  // Constant declaration using all uppercase letters and underscores
  public static final int MAX_HEIGHT = 100;
  public static final int TOTAL_COUNT = 50;

  public static void main(String[] args) {
    // Variable declaration using camelCase
    int studentAge = 20;
    String myVariable = "Hello, World";

    System.out.println("Maximum height allowed: " + MAX_HEIGHT);
```

```
        System.out.println("Total count: " + TOTAL_COUNT);
        System.out.println("Student age: " + studentAge);
        System.out.println(myVariable);
    }
}
```

Commenting Your Code:

Comments in Java are notes that explain the code but are not executed. They're crucial for maintaining code, making it more readable and understandable. Java supports single-line and multi-line comments.

- **Single-Line Comments:** Start with `//` and continue till the end of the line. For example: **`// This is a single-line comment`**
- **Multi-Line Comments:** Start with `/*` and end with `*/`. They can span multiple lines. For example:
-

```
public class Main {
    public static void main(String[] args) {
        // This is a single-line comment explaining the next line of code
        System.out.println("Hello, world!");

        /*
         * This is a multi-line comment spanning over
         * multiple lines, used to provide more detailed
         * explanations or to comment out blocks of code.
         * Below is a print statement that will print
         * a simple message to the console.
         */
        System.out.println("Multi-line comments are useful for longer descriptions.");
    }
}
```

Question:

1. How do you declare a static variable in Java, and why might you declare one outside the main method?
2. How do you comment in XML files? For e.g pom.xml

//Please write the answer in comment section, if you know.