Java Notes

DB (Oracle DB) -> Server Side (Java8, JDBC, Servlets, Hibernate, Spring) -> Client-Side (HTML, CSS, JavaScript, Angular)

"Full-Stack Java Developers" – What does this mean?

What is Java? An object-oriented programming language first released by Sun Microsystems in 1995. Java is fast, secure, reliable, free, and very widely used. The language is the core of the mentioned stack.

Packages: Java developers group related classes and other referenced types into packages. This language construct makes it easier to locate and use types, avoid naming conflicts between same-named types, and control access.

Java's API (Application Programming Interface) – Packages include:

• java.applet: Provides the classes necessary to create an applet.

• java.awt: Contains all the classes needed for creating UIs, and painting graphics and images.

• java.beans: Contains classes related to developing beans (components based on the JavaBeans architecture).

• java.io: Provides system input and output through data streams, serialization, and the file system.

• java.lang: Provides classes that are fundamental to the design of the Java programming language.

• java.math: Provides classes for performing arithmetic and various precision classes.

• java.net: Provides classes for implementing network applications.

• java.nio: Defines buffers, which are containers for data and provides an overview of other NIO (non-blocking I/O) packages.

• java.rmi: Remote Method Invocation package

• java.security: Classes and interfaces for the security framework.

• java.sql: Provides the API for accessing and processing data stored in a data source.

• java.text: Handles text, dates, numbers, and messages in a manner independent of natural languages.

• java.time: Main API for dates, times, instants, and durations.

• java.util: Contains the collections framework internationalizations and miscellaneous utility classes.

• javax: Extension to the Java API, java.* are the essentials.

In summary, packages are directories.

The import statement enables classes in other packages to be accessed without referring to the full path and file name.

```
import java.io.File
```

so that we can instantiate a file such as

```
File f = new File();
```

instead of

```
java.io.File f = new java.io.File();
```

Class structure:

```
package [package name];

import [external package(s)];

class [class name] {

      …

}
```

Other related concepts:

- java.lang
- static imports
- access modifiers
- naming conventions