URBAN EDGE

Advanced Java - 2

Creating our own Java Documentation

Java documentation is great!
It helps us get info about which method/class/Entity to use when.

We can create our own package's Documentation in Java.

Javadoc tool

javadoc command allows us to create documentation in HTML format for our own package.
Java provides tags for class or package to assist with the java doc generation

Tags for class or a package

1) @ author : Adds the author name
2) @ version : Adds the version
3) @ Since : To add when was this version written
4) @ See : Adds a see also heading with a link

Tags for methods

Javadoc provides following tags for methods :

1) @ param → for describing parameters of a method
2) @ return → for describing about the return value
3) @ throws → for describing exceptions thrown
4) @ deprecated → for describing deprecation status

Description can be added at the start of javadoc comment

URBAN EDGE

Annotations in Java

Used to provide extra information about a program
Annotations provides metadata to class/methods

Following are some common annotations built into Java

1. @Override → Used to mark Overridden elements in the child classes
2. @SuppressWarnings → Used to suppress the generated warnings by the Compiler
3. @deprecated → Used to mark deprecated methods
4. @FunctionalInterface → Used to ensure an interface is a functional interface

Lambda Expressions

Added in Java 8
Lambda expressions let us express instances of single method classes more compactly.
Anonymous classes are used to implement a base class without giving it a name.
For classes with a single method, even anonymous classes get slightly exessive & cumbersome

Java Generics

Introduced from JDK 5.0 onwards
Very similar to C++ Templates (but not the same)

If we write
ArrayList a = new ArrayList();

a.add(75)

int anum = a.get(0) → We cant do this

int anum = (int) a.get(0) → We will have to do this

Hence generics aim to reduce bugs & enhance type safety.

Note: Type parameter in java generics cannot be a primitive data type.

File Handling in Java

Reading from & writing to files is an important aspect of any programming language

We can use the File class in Java to create a file object

- CreateNewFile() method → Creates a file object
- For reading files we can use the same Scanner class and supply it a file object
- To delete a file in Java we can use File object's delete() method.