Chapter - 6

Java ships with 100s of prebuilt classes.

ArrayList:

⇒Some of the methods in arraylist,

1. add(Object ele)
2. remove(Object ele)
3. contains(Object ele)
4. isEmpty()
5. indexOf(Object ele)
6. size()
7. get(int index)

⇒ It resizes dynamically
⇒ Cannot hold primitives
⇒ You can put primitives in an ArrayList, as long as it's wrapped in a primitive wrapper class. In Java 5.0, wrapping and unwrapping happens automatically.

Comparing Arrays with ArrayLists

Arrays	ArrayLists
1. Size to be known while creation.	1. No such requirement
2. To put an object, you must assign some index to it.	2. You can specify the index [add(anInt, anObject)] or in default case it simply gets appended .
3. Array Syntax is not used anywhere else in the Java	3. ArrayLists are plain old Java Objects, so they have no special syntax.

4. Methods cannot be called on Arrays, though they are objects. Only the instance variable length can be accessed.	4. It functions properly as a class.
5. Nothing like this here	5. They use something special that was added to java 5.0 - parameterized data types. ArrayList<String> Note: Initially they were simply the heterogeneous collection of objects.

Short Circuit Operators (&&, ||)

Why are they called Short Circuit Operators?
⇒ If JVM sees that the left side of && is false or left side of || is true, it doesn't bother to look for the other side.

Non Short Circuit Operators (&, |)

They act like && and ||, except that they force the JVM to check on both sides. Typically they are used for manipulating bits.

Java API (Java Library)

Classes are grouped into packages here.

You have to know the full name of the class if you want to use it in your code.

Eg: ArrayList is not the full name, it is java.util.ArrayList.

You have 2 options

A. Import:

import java.util.ArrayList;

- Does it make my class bigger?
  ⇒ An import is not the same as an include. It is simply the way you give Java the full name of a class.

B. Type:

Type the full name whenever you use it.

Why no import for String?

⇒ The java.lang package is sort of pre-imported for free. So all the classes in it need no import.

Purpose of Packages?

1. Helps in overall organisation
2. They give us a name scoping, that helps prevent collisions.
3. They provide a level of security, because you restrict the code so that only the other classes in the same package can access it.

How to play with the API?

2 things you want to know:

a. What classes are in the library?
b. Once you find a class, how do you know what it can do?

You can do this my following methods

1. Browse a book
2. Use the HTML API Docs

⇒ Java comes up with a fabulous set of online docs called API docs.

⇒ They are the best reference for getting more details about a class and its methods.

What does it mean when a package starts with javax?

- In the first and second versions of java, all the classes that shipped with java were considered as **Standard Library.**
- Then, the classes that weren't included in the Standard Library were considered as **Extensions**. And they came in 2 flavours: standard and non standard.
- Standard Extensions, by convention, all began with an 'x' appended to the regular java package starter.
- The mother of all standard extensions was the Swing Library.
- But Standard extensions got promoted to first-class, ships-with-java, standard-out-of-the-box library packages. But it was not possible to take away the 'x' from them.
- So now when you see a package in the library that begins with javax, you know it started its life as an extension and then got promoted.