

Agenda

- Packages
- Access Modifiers
- this reference
- Types of Methods
- Constructor Chaning
- ~~Array~~

Packages

- Packages makes Java code modular. It does better organization of the code.
- Package is a container that is used to group logically related classes, interfaces, enums, and other packages.
- Package helps developer:
 - To group functionally related types together.
 - To avoid naming clashing/collision/conflict/ambiguity in source code.
 - To control the access to types.
 - To make easier to lookup classes in Java source code/docs.
- Java has many built-in packages.
 - java.lang -> Integer, String , System
 - java.util -> Scanner
 - java.io -> PrintStream, Console
 - java.sql -> Connection, Statement
- To define a type inside package, it is mandatory write package declaration statement inside .java file.
- Package declaration statement must be first statement in .java file.
- Types inside package called as packaged types; while others (in default package) are unpackaged types.
- Any type can be member of single package only.
- It is standard practice to have multi-level packages (instead of single level). Typically package name is module name, dept/project name, website name in reverse order.
- When compiled, packages are created in form of directories (and sub-directories).

```
package com.sunbeaminfo.kdac
```

creating package using command line execution

- create a folder demo01
- create 2 sub directories src and bin
- write a .java file with the package p1.
- use below steps for compilation and execution

```
javac -d ../bin Program.java

export CLASSPATH=../bin

java p1.Program

// if without setting classpath we want to execute the java code use below
command
java -cp ../bin p1.Program
```

- for multiple files in multiple packages (Demo02)

```
javac -d ../bin Time.java
export CLASSPATH=../bin

//add import statement inside the Program.java file
javac -d ../bin Program.java

java p1.Program
```

- If the class is not kept public, the class won't be able to be accessed in other packages

Access Modifiers

- For class
 - 1. default
 - 2. public
- For class members
 - 1. private
 - only within the class directly
 - 2. default (package level private)
 - in same class directly
 - in all the classes in the same package on class object
 - 3. protected
 - in same class directly
 - in all the classes in the same package on class object
 - in subclasses directly
 - 4. public
 - are visible every where.

Difference between protected and default

- Default access restricts visibility to only classes within the same package. This allows you to encapsulate implementation details that are not intended to be accessed by classes outside the package.
- Protected access, on the other hand, allows access by subclasses (regardless of package) and by other classes within the same package.
- If you want to hide implementation details from all classes, including subclasses, default access provides stricter encapsulation.

this Reference

- "this" is implicit reference variable that is available in every non-static method of class which is used to store reference of current/calling instance
- Whenever any non-static method is called on any object, that object is internally passed to the method and internally collected in implicit "this"
- "this" is constant within method i.e. it cannot be assigned to another object or null within the method.
- Using "this" inside method (to access members) is optional.
- However, it is good practice for readability.
- In a few cases using "this" is necessary.

Types of Methods

1. constructor
2. setters
 - Used to set value of the field from outside the class.
 - It modifies state of the object.
3. getters
 - Used to get value of the field outside the class.
4. facilitators
 - Provides additional functionalities
 - Business logic methods

Constructor

- It is a special method of the class
- In Java fields have default values if uninitialized
- Primitive types default value is usually zero
- Reference type default value is null
- Constructor should initialize fields to the desired values.
- Types of Constructor
 - 1. Default/Parameterless Ctor
 - 2. Parameterized Ctor

Constructor Chaining

- Constructor chaining is executing a constructor of the class from another constructor (of the same class).
- Constructor chaining (if done) must be on the very first line of the constructor.

Lab Work

1. Do the classwork of packages using STS. If comfortable then only go for command line
2. Check for the visibility of access modifiers
3. class, object (Ctor, Setters, getters, facilitators)
4. Constructor chaining
5. Solve the assignment given
6. Revise the concept of Array from cpp
(1D Array of Pointers, Dynamic array)