Agenda

```
Switch Case
This reference
Constructor
Constructor Chaning
Static Field
Static method
Package
Access Modifiers
Static import
```

Switch Case (Demo01)

Steps to create the project in STS

```
1. Switch Workspace every single day (day01)
2. Change the Perspective
3. Create a new Project (Demo01,Demo02,Demo03....)
4. Create a class inside src called as Program (Follow Naming conventions)
5. Every single project should have only 1 main().
```

This reference (Demo02)

```
this reference is passed internally to all the non staic methods of the
class
this reference stores the address of current calling object
use of this is optional but it is recommended as per industry standard
***this reference is used to identity  the difference between class fields
and local variables.
```

Constructor (Demo03)

```
- Special method of the class
    1. Name of ctor is same as that of class name
    2. It does not return type
    3. It gets automatically called for every object you create
```

Types of Constructor (Demo04)

```
1. Parameterless Constructor
2. Parameterized Constructor
3. Deafult Constructor -> Parameterless Ctor

- If developer provides his own ctor inside the class then the default ctor
gets replaced
- Generally in the parameterized ctor the name of parameters should be same
as that of class fields that we need to initialize.
```

Constructor Chaning (Demo05)

```
- If we want to call exixisting Constructor form the current constructor
that is called then we should use constructor chaning
- For this we have to use this statement
- this statement should be the first statement  inside the constructor.
```

Static field (Demo 06)

```
- If we want to share the fields of a class between multiple objects of
that class , then we should make such fields as static
- Static fields get space on to method area.
- Staic fields should be initialized inside static initializer block
- They are shared between multiple objects
- If one object changes the value of such field than the changes will be
observed between all the objects
```

Static method (Demo07)

```
- If you want to call any methods of the class without creating the class
object then make such methods as static
- static methods are designed to call on classname
- static methods do not get this reference
- We cannot access non static fields inside static methods
- we can only access static fields inside static methods
- Inside static methods we can call only static menthods of the same class,
we cannot call nonstatic methods directly.
```

package (Demo08)

```
- it is a container that holds related classes together
- the package should have package name in small case
- Generally the package names are kept on company's domain name
- (domain name)sunbeaminfo.com -> com.sunbeaminfo(package name)
```

Access Modifiers (Demo10)

```
1. private
2. protected
3. package level private (default)
4. public

In the same package (p1)
All the members with any access modifier
    1. Are accessible within that class
    2. Except Private all are accessible in other class

In different package (p1 & p2)
All the members with any access modifier
    1. Are accessible within that class
    2. Only public are accessible in other class in different package
```

Name of public class and the name of .java file should be same

All public classes should be defined in its own .java file

Static import (Demo 11)

```
If you want to use any static members in a differnt package and every time
when you use, you don't want to use the classname and . operator then use
import static.
```

Lab Flow

```
switch
this
Constructor
Static
Package
Access modifier
```