Enums

&

Annotation

19th dec Enums and Annotation

20 December 2022 12:25

Enums → to define Group of Constants, Annotations used Enums internally.

```
//NORTH SOUTH EAST WEST
// SUN MON TUE WED THU FRI SAT
// PASS FAIL NR
// so on.......
enum Result1 {
    PASS, FAIL, NR; // public static final
    // fields --> instance var --> properties
    // methods
    // Constructor
}

// enum Compass
// {
// NORTH, SOUTH, EAST, WEST;
// }

// class Demo
// {
// //final int PASS=35;
// //PASS --> error
// }

// separte .class for every enum

public class LlaunchEnum {
    enum Gender {
        MALE, FEMALE, OTHERS;
    }
    public static void main(String[] args) {
    }
}
```

value of this is NR only.

Enums:

-> inside this we can create

1. instance var
2. methods
3. Constructor
4. Constant

-> Value of Enum Constant == Variable only.

```
enum Result {
    PASS, FAIL, NR; // static final
    int ResultId;
    // PASS --> public static final Result PASS=new Result();
    // FAIL --> static final Result FAIL =new Result();
    // NR ---> static final Result NR=new Result();

    Result() {
        System.out.println("Constructor is called");
    }

    void setResultId(int ResultId) {
        this.ResultId = ResultId;
    }

    int getResultId() {
        return ResultId;
    }
}
```

Internally

19th dec Annotation

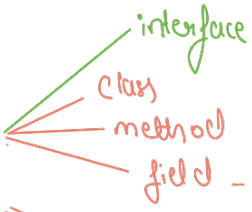
20 December 2022 14:04

with comments we can only give info to other developer but with Annotations we can give it to Compiler, JVM etc, also.

```
// Annotation  java 5 extra info giving to class or JVM or webserver etc.
//Annotation --> Annotation --> parent of all Annotations

//Annotation ---> Built in ==      98% use
//           ----> Custome (our Own) 2%

// Annotation ->
//1) class
//2) interface
//3) method
// 4) fields // instance var
//5 ) local variables
// 6) Consrtuctor
// 7) Parameters
// 8) enum

@FunctionalInterface -Annotation
interface Trial {
    int getNum();
    // void disp();
}
class JavaLearning {
    public void disp1() {
        System.out.println("Parent disp");
    }
}
class Focus extends JavaLearning {
    @Override
    public void disp1() {
        System.out.println("Focus is key");
    }
}
public class LaunchAnno {
    public static void main(String[] args) {
        Trial t = () -> {return 10;};
    }
}
```

Creating Own Annotation

1.Create interface with @
2.give @target:for which level you are making



3.give Retention till which level you want it


```
import java.lang.annotation.ElementType;

// There are 3 types of Annotation
// 1.Marker : nothing inside body
// 2.single :single method inside body
// 3.Multi value : more than one method inside body
```

How to Create Annotations:

```java
@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
@interface CricketPlayer {
    // @ --> its not interface but its Annotation being created
    String country() default "India";
    int runs();
}

@CricketPlayer(country = "US", runs = 2000)
class ViratKohli {
    private int innings;
    private String name;
    public int getInnings() {
        return innings;
    }
    public void setInnings(int innings) {
        this.innings = innings;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}

public class LaunchAnn2 {
    public static void main(String[] args) {
        ViratKohli vk = new ViratKohli();
        vk.setInnings(300);
        vk.setName("VK");

        System.out.println(vk.getInnings());//300
        System.out.println(vk.getName());//VK

        Class c = vk.getClass();
        Annotation an = c.getAnnotation(CricketPlayer.class);
        CricketPlayer cp = (CricketPlayer) an;//downCasting

        System.out.println(cp.runs());//2000
        System.out.println(cp.country());//US
    }
}
```

How to Access Annotation variable