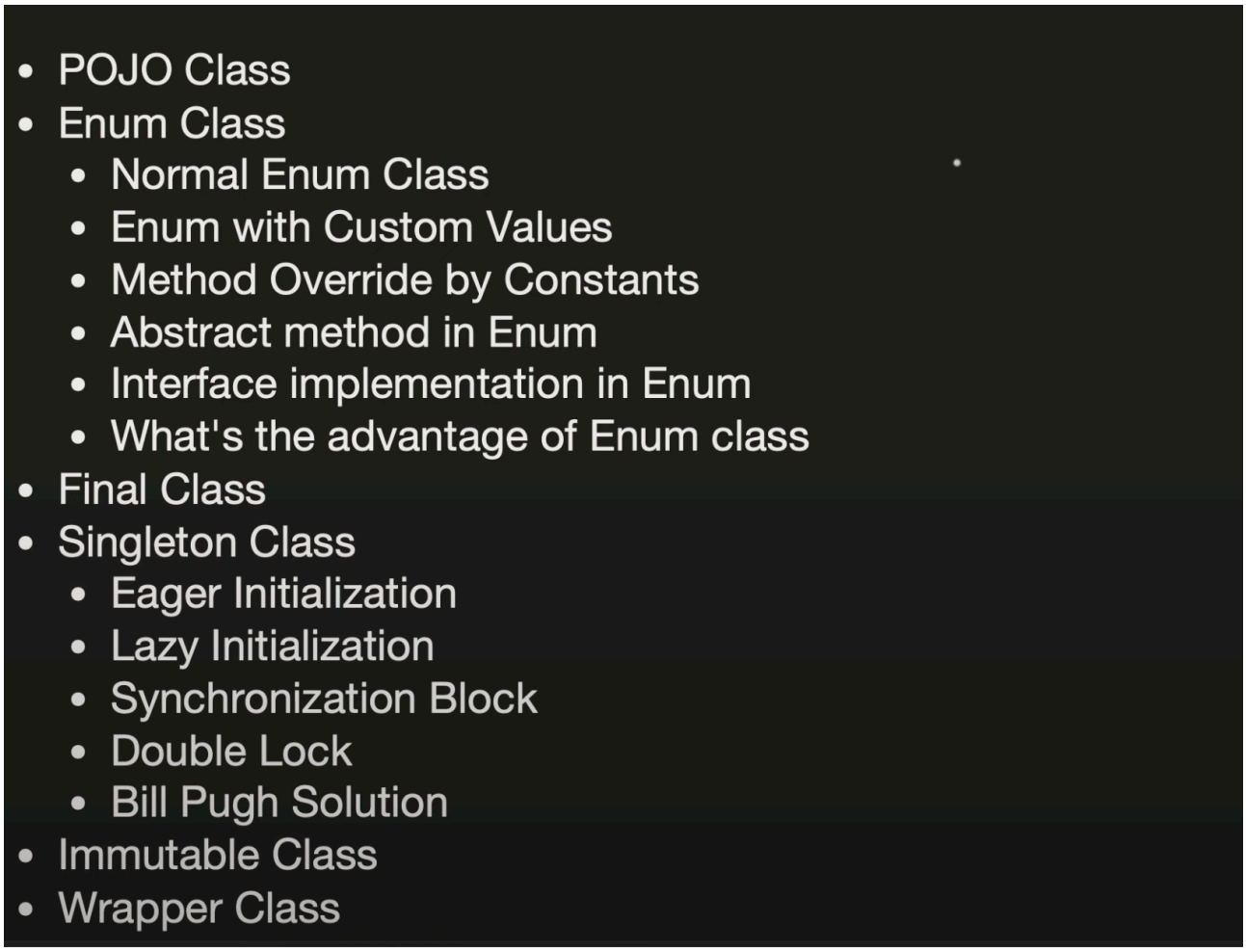
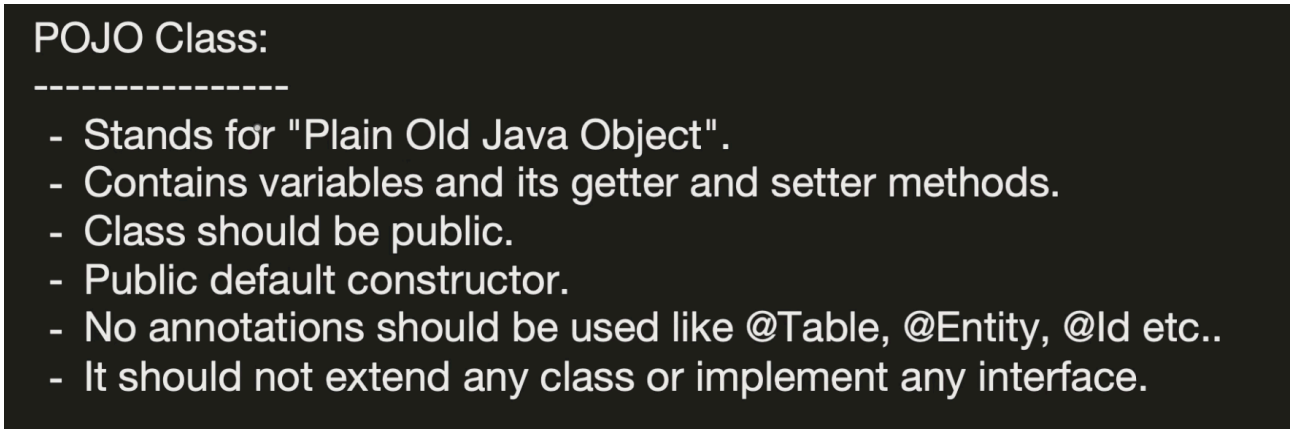
**"Concept && Coding" YT Video Notes**




```
 Public class student {
int name;
private int rollNumber;
protected String address;

Public int getName ()  {
return name;
}
Public void setName(int name) {
```

```
this.name=Name;
}
Public int getRollNumber () {
return rollNumber;
}
public void setRollNumber(int rollNumber) {
this.rollNumber= rollNumber;
}
public string getAddress() {
return address;
}
public void setAddress (String Address) {
this.address=address;
}
}
```

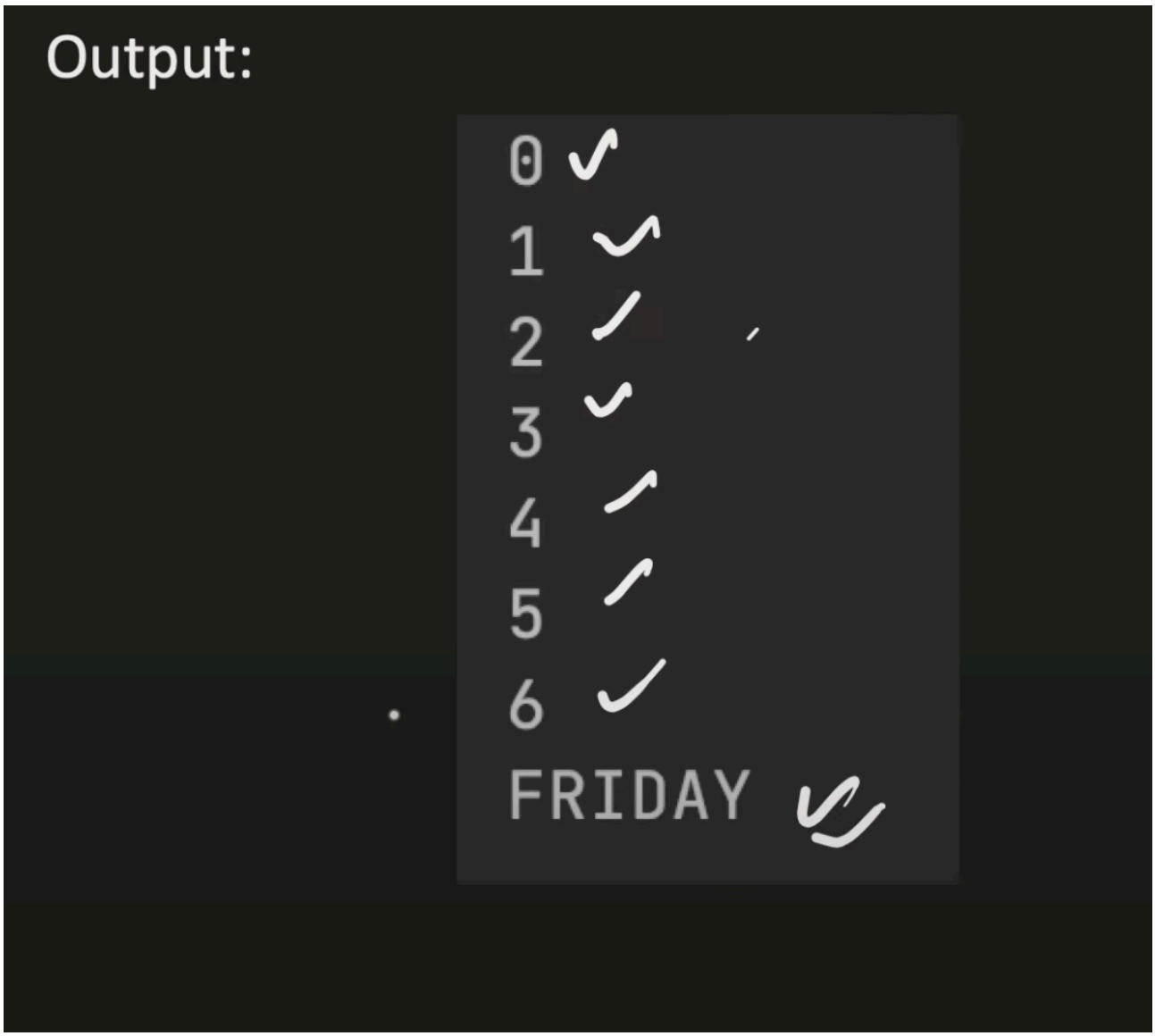
ENUM Class:
---

✓- It has a collection of CONSTANTS (variables which values can not be changed)
- Its CONSTANTS are static and final implicitly (we do not have to write it).
- It can not extend any class, as it internally extends java.lang.Enum class
- It can implement interfaces.
- It can have variables, constructor, methods.
- It can not be initiated (as its constructor will be private only, even you give default, in bytecode it make it private)
- No other class can extend Enum class
- It can have abstract method, and all the constant should implement that abstract method.

Normal Enum class:

```
public enum EnumSample{

    MONDAY,
    TUESDAY,
    WEDNESDAY,
    THURSDAY,
    FRIDAY,
    SATURDAY,
    SUNDAY;
}
```

```
public class Main {
    public static void main(String args[]) {

        /*Common functions which is used
        - values()
        - Ordinal()
        - valueOf()
        - name()
         */

        //1. usage of Values() and ordinal()
        for(EnumSample sample : EnumSample.values()){
            System.out.println(sample.ordinal());
        }


        //2. usage of ValueOf() and name()
        EnumSample enumVariable = EnumSample.valueOf( name: "FRIDAY");
        System.out.println(enumVariable.name());
    }
}
```




## Enum with custom values:

```
 public enum EnumSample{
MONDAY( 101 ,  "1st day of the week"),
TUESDAY(102 , "2nd day of the week"),
WEDNESDAY(103 , "3rd days of the week"),
THURSDAY(104 , "4th day of the week"),
FRIDAY(105 , "5th day of the week"),
SATURDAY(106 , "its 1st WeekOff"),
SUNDAY(107 , "its 2nd WeekOff");

private int val;
private String comment;
EnumSample(int val, String comment){
```

```
this.val  = val;
this.comment = comment;
}
public int getVal(){
return val;
}
public String getComment(){
return Comment;
}
public static EnumSample getEnumFromValue(int Value){
for(EnumSample sample : EnumSample.Values() ){
if(sample.val==value){
return sample;
}
}
return null;
}
}
```

```
public class Main {
    public static void main(String args[]) {

        EnumSample sampleVar = EnumSample.getEnumFromValue(107);
        System.out.println(sampleVar.getComment());

    }
}
```

# Method Override by Constant:

```java
public enum EnumSample{

    MONDAY{

        @Override
        public void dummyMethod(){
            System.out.println("Monday dummy Method");
        }
    },
    TUESDAY,
    WEDNESDAY,
    THURSDAY,
    FRIDAY,
    SATURDAY,
    SUNDAY;

    public void dummyMethod(){
        System.out.println("default dummy Method");
    }
}
```

```java
public class Main {
    public static void main(String args[]) {

        EnumSample fridayEnumSample = EnumSample.FRIDAY;
        fridayEnumSample.dummyMethod();

        EnumSample mondayEnumSample = EnumSample.MONDAY;
        mondayEnumSample.dummyMethod();
    }
}
```

Output:

```
default dummy Method
Monday dummy Method
```

## Enum with Abstract method:

```
public enum EnumSample{

    MONDAY {
        public void dummyMethod(){
            System.out.println("in Monday");
        }
    },
    TUESDAY{
        public void dummyMethod(){
            System.out.println("in Tuesday");
        }
    },
    SUNDAY{
        public void dummyMethod(){
            System.out.println("in Sunday");
        }
    };

    public abstract void dummyMethod();
}
```

```java
public class Main {
    public static void main(String args[]) {

        EnumSample mondayEnumSample = EnumSample.MONDAY;
        mondayEnumSample.dummyMethod();

    }
}
```

Output:

```
in Monday
```

Enum implement interface:

```java
public interface MyInterface {

    public String toLowerCase();

}
```

```java
public enum EnumSample implements MyInterface{
    MONDAY,
    TUESDAY,
    WEDNESDAY,
    THURSDAY,
    FRIDAY,
    SATURDAY,
    SUNDAY;

    @Override
    public String toLowerCase() {
        return this.name().toLowerCase();
    }
}
```

```java
public class Main {
    public static void main(String args[]) {

        EnumSample mondayEnum = EnumSample.MONDAY;
        System.out.println(mondayEnum.toLowerCase());
    }
}
```

Output:

monday

What's the benefit of ENUM class when we can create constant through "static" and "final" keyword?

```java
public class WeekConstants {

    public static final int MONDAY = 0;
    public static final int TUESDAY = 1;
    public static final int WEDNESDAY = 2;
    public static final int THURSDAY = 3;
    public static final int FRIDAY = 4;
    public static final int SATURDAY = 5;
    public static final int SUNDAY = 6;

}
```

```java
public enum EnumSample{
    MONDAY,
    TUESDAY,
    WEDNESDAY,
    THURSDAY,
    FRIDAY,
    SATURDAY,
    SUNDAY;
}
```

```java
public class Main {
    public static void main(String args[]) {

        isWeekend( day: 2); //wednesday, so it will return false
        isWeekend( day: 6); //sunday, so it will return true
        isWeekend( day: 100); //this value is not expected, but still we are able to send this in parameter


        //better readability and full control on what value we can pass in parameter
        isWeekend(EnumSample.WEDNESDAY); // return false
        isWeekend(EnumSample.SUNDAY); // return true


    }

    public static boolean isWeekend(int day){

        if(WeekConstants.SATURDAY == day || WeekConstants.SUNDAY == day){
            return true;
        }
        return false;
    }

    public static boolean isWeekend(EnumSample day){

        if(EnumSample.SATURDAY == day || EnumSample.SUNDAY == day){
            return true;
        }
        return false;
    }
}
```

FINAL CLASS:

- Final class can not be inherited

```java
public final class TestClass {
}
```

```java
public class MyOtherClass extends TestClass{
}
```

Cannot inherit from final 'EnumDemo.TestClass'

Make 'TestClass' not final ⌥⇧⏎ More actions... ⌥⏎