PW SKILLS

Lecture

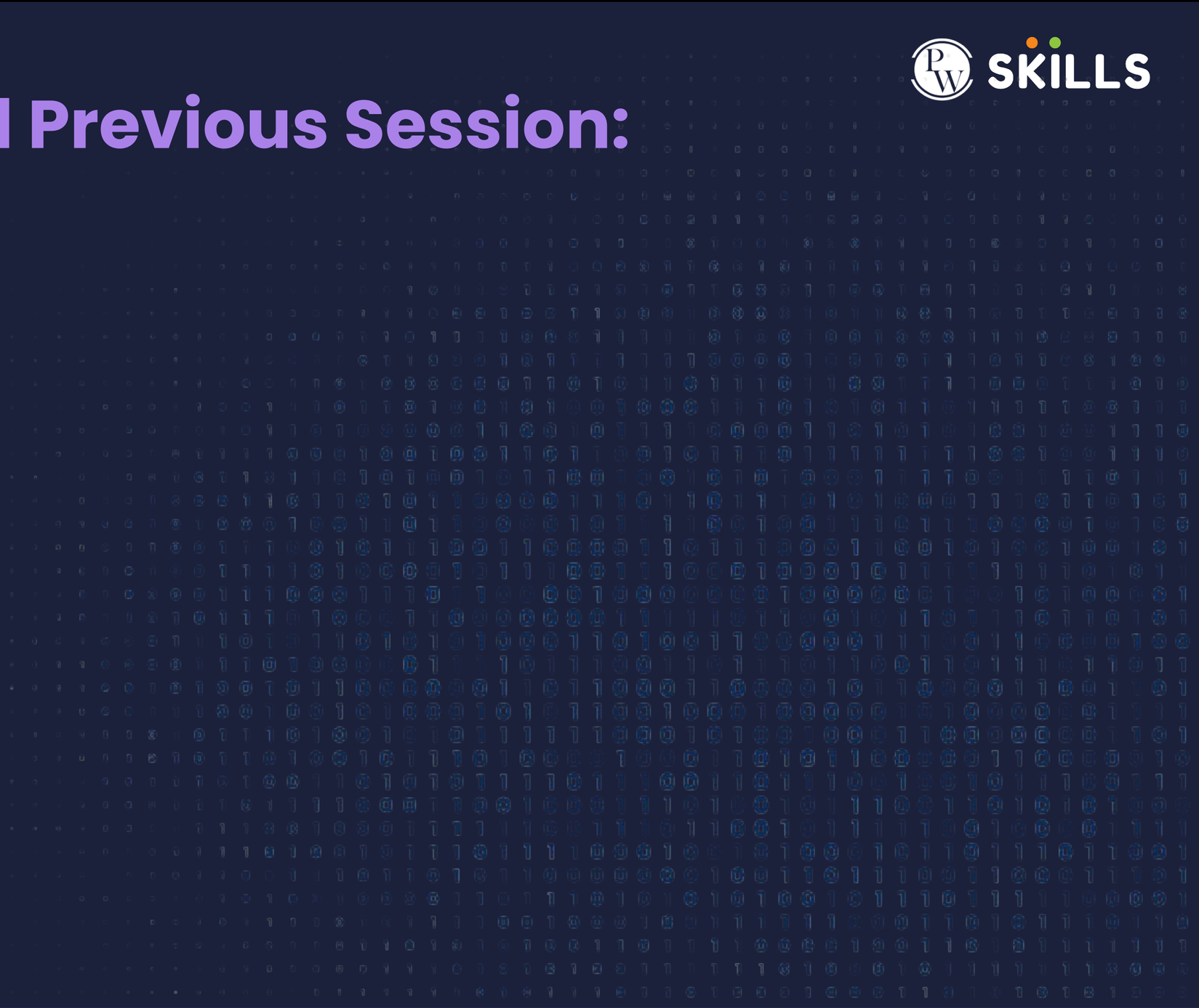
Polymorphism and Abstraction

List of Concepts Involved:

- What is polymorphism?
- How to achieve polymorphism
- Runtime vs Compile time polymorphism
- Abstract keyword and Abstraction
- Abstract class and Abstract method
- final class
- final variable
- final method

Topics covered Previous Session:

- Inheritance



What is polymorphism?

If one thing exists in more than one form then it is called Polymorphism. Polymorphism is a Greek word, where Poly means many and morphism means structures or forms.

1. Static Polymorphism
2. Dynamic Polymorphism

How to achieve polymorphism:

Polymorphism in Java can be achieved in two ways i.e., method overloading and method overriding.

Polymorphism in Java is mainly divided into two types.

1. **Static Polymorphism:**
   If polymorphism exists at compilation time then it is called Static Polymorphism.
   Ex: Overloading.

2. **Dynamic Polymorphism:**
   If the polymorphism exists at runtime then that polymorphism is called Dynamic Polymorphism.
   Ex: Overriding

Method Overriding:

The process of replacing existing method functionality with some new functionality is called Method Overriding.

- To perform Method Overriding, we must have inheritance relationship classes.
- In Java applications, we will override super class method with sub class method.
- If we want to override super class method with subclass method then both super class method and sub class method must have the same method prototype.

Runtime vs Compile time polymorphism

What are the differences between method overloading and method overriding?

method overloading

- The process of extending the existing method functionality with new functionality is called Method Overloading.
- In the case of method overloading, different method signatures must be provided to the methods

method overriding

- The process of replacing existing method functionality with new functionality is called Method Overriding.
- In the case of method overriding, the same method prototypes must be provided to the methods.
- With or without inheritance we can perform method overloading with inheritance only we can perform Method overriding

Abstract keyword and Abstraction:

- The abstract keyword is used to achieve abstraction in Java. It is a non-access modifier which is used to create abstract class and method.
- The role of an abstract class is to contain abstract methods. However, it may also contain non-abstract methods.
- The method which is declared with an abstract keyword and doesn't have any implementation is known as an abstract method.

Syntax:-

```
abstract class Employee
{
abstract void work();
}
```

Abstract Class and Abstract Methods

- In Java applications, if we declare any abstract class with abstract methods, then it is convention to implement all the abstract methods by taking sub classes.
- To access the abstract class members, we have to create an object for the subclass and we have to create a reference variable either for abstract class or for the subclass.
- If we create reference variables for abstract class then we are able to access only abstract class members, we are unable to access subclass own members.
- If we declare a reference variable for subclass then we are able to access both abstract class members and subclass members.

final class

- If a class is marked as final, then the class won't participate in inheritance,if we try to do so then it would result in "CompileTime Error".

  Eg: String, StringBuffer,Integer,Float,.....

final variable

- If a variable is marked as final, then those variables are treated as compile time constants and we should not change the value of those variables.

- If we try to change the value of those variables then it would result in “CompileTimeError”.

final method

- If a method is declared as final then those methods we can't override, if we try to do so it would result in "CompileTimeError".

Next Lecture

- Interface

PW SKILLS

THANK YOU