JAVA

Balaji Dinakaran
balajidinakaran1@gmail.com
https://www.linkedin.com/in/balaji-dinakaran/

CORE JAVA

- Encapsulation
- Inheritance and Super Keyword
- Polymorphism
- Final
- Abstract
- Interface

Encapsulation

- Encapsulation is one of the four fundamental OOP concepts.
- Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit.
- In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods.
- Steps to achieve encapsulation:
  1) Declare the variables of a class as private.
  2) Provide public setter and getter methods to modify and view the variables values.
- The fields of a class can be made read-only or write-only.

Inheritance

- 1. Inheritance can be defined as the process where child succeeds the properties of the parent.
- 2. Generally it is "like father like son" which means son is having the qualities of father.
- 3. Inheritance in java has parent or super class and child or sub or derived class that inherits or acquire properties of parent class.
- 4. After inheriting child class will reuse the same and not create.

Super Keyword

- Child class default or parameterized constructors implicitly call parent class default constructor
- If parent class is having only parameterized constructors then, from child class constructor explicitly invoke parent class constructor using super() keyword
- Super keyword should be called first in the constructor

Static Polymorphism

- Two or more method in a java class can have same name if their argument list are different
- Argument lists could differ in
  1. Number of arguments
  2. Datatypes of arguments
  3. Sequence of datatypes of arguments
- This feature is called method overloading or Compile time binding or early binding.

Dynamic Polymorphism

- It is called Run time polymorphism as it is exhibited at run time.
- Call to the method is resolved during run time.
- Method to be invoked is based on the object being referred.
- Method override in java means that method can be defined with the same name and signature in the base class as well as derived class.
- When a method is defined in the base class and again redefined in derived class then method in base class can be considered as overridden method by the method in derived class.

Final

- Final keyword in java is used to restrict the user from performing changes.
- Final keyword can be used in many context
- Final keyword can be applied in
  1. Variable
  2. Method
  3. classes

Usage of Final Keyword

- 1. Prevent variable from having value change
- 2. Prevent method overriding
- 3. Prevent inheritance of classes

Final Variable

- A final variable once being initialized, cannot have it's value changed again, as it will be constant.
- If the variable is primitive datatype or an object reference, it permits a read-only access to it.
- By convention should be named in UPPERCASE
- A final variable that is not initialized at the time of declaration is called as blank final variable
- A blank final variable should be initialized only within a constructor.

Final method and Final class

- Final method:
  - A final method cannot be overridden by subclasses
  - This is to prevent a child class from redefining a method from its parent class

- Final class:

1. A class declared as final cannot be inherited by subclasses
2. This can foster security and efficiency benefits.
3. Many of the java standard library classes are final, such as java.lang.System and java.lang.String
4. All methods in final class are final by default

Abstraction

- Abstraction is the quality of dealing with ideas rather than events.
- Abstraction is a process of hiding the implementation details from the user, only the functionality will be provided to the user.
- In Java, abstraction is achieved using Abstract classes and interfaces.

Abstract Class

- A class which contains the abstract keyword in its declaration is known as abstract class.
- Abstract classes may or may not contain abstract methods, i.e., methods without body (public void get();)
- But, if a class has at least one abstract method, then the class must be declared abstract.
- If a class is declared abstract, it cannot be instantiated.
- To use an abstract class, you have to inherit it from another class, provide implementations to the abstract methods in it.
- If you inherit an abstract class, you have to provide implementations to all the abstract methods in it.

Interface

- It is similar to class. It is a collection of abstract methods.
- Writing an interface is similar to writing a class. But a class describes the attributes and behaviors of an object. And an interface contains behaviors that a class implements.
- You cannot instantiate an interface.
- An interface does not contain any constructors.
- All of the methods in an interface are abstract.
- An interface is not extended by a class; it is implemented by a class.
- An interface can extend multiple interfaces.

Regular Expression

- Java provides the java.util.regex package for pattern matching with regular expressions.
- A regular expression is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern.
- Pattern Class – A Pattern object is a compiled representation of a regular expression.

Note: You must first invoke one of its public static **compile()** methods.

- Matcher Class – A Matcher object is the engine that interprets the pattern and performs match operations against an input string.

Password Regx Explanation

- Pattern: ^(?=.*[0-9])(?=.*[a-z])(?=.*[A-Z])(?=.*[@#$%^&+=])(?=\S+$).{8,}$

```
^                 # start-of-string
(?=.*[0-9])       # a digit must occur at least once
(?=.*[a-z])       # a lower case letter must occur at least once
(?=.*[A-Z])       # an upper case letter must occur at least once
(?=.*[@#$%^&+=])  # a special character must occur at least once
(?=\S+$)          # no whitespace allowed in the entire string
.{8,}             # anything, at least eight places though
$                 # end-of-string
```

Datetime in Java

- Java provides the Date class available in java.util package, this class encapsulates the current date and time.
- SimpleDateFormat is a concrete class for formatting and parsing dates in a locale-sensitive manner.
- SimpleDateFormat allows you to start by choosing any user-defined patterns for date-time formatting.

“

Thank you for joining the session.

”

- Contact Details: balajidinakaran1@gmail.com
- https://www.linkedin.com/in/balaji-dinakaran/