Object-Oriented Programming in JAVA

Lecture 06

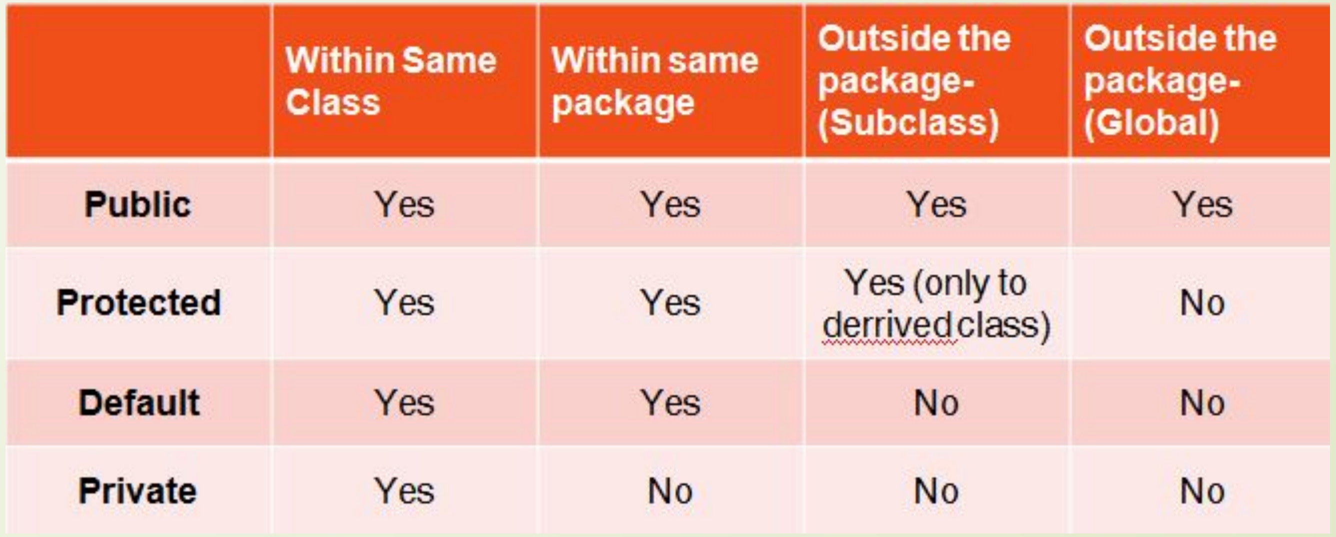
Encapsulation in Java

- What is Encapsulation?
  - **Definition**:
    - Encapsulation is the bundling of data (attributes) and methods (functions) that operate on the data into a single unit, or class.
  - **Purpose**:
    - Protects the integrity of the data.
    - Restricts direct access to some of the object's components.

Key Features of Encapsulation

 Access Modifiers:

- **private: Restricts access to the class itself.**
- **public: Accessible from other classes.**
- **protected: Accessible within the same package and subclasses.**
- **Default:**



	Within Same Class	Within same package	Outside the package-(Subclass)	Outside the package-(Global)
Public	Yes	Yes	Yes	Yes
Protected	Yes	Yes	Yes (only to derrived class)	No
Default	Yes	Yes	No	No
Private	Yes	No	No	No

Encapsulation Code Example

```
// Class definition
public class Person {
    // Private variables (attributes)
    private String name;
    private int age;

    // Public constructor
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    // Getter for name
    public String getName() {
        return name;
    }
```

Encapsulation Code Example

```
// Setter for name
public void setName(String name) {
    this.name = name;
}

// Getter for age
public int getAge() {
    return age;
}

// Setter for age
public void setAge(int age) {
    if (age > 0) {
        this.age = age;
    } else {
        System.out.println("Age must be positive.");
    }
}
}
```

```
// Usage
public class Main {
    public static void main(String[] args) {
        Person person = new Person("Alice", 30);
        person.setAge(25);
        System.out.println("Name: " +
person.getName() + ", Age: " + person.getAge());
    }
}
```

Benefits of Encapsulation:

- **Control**: Restrict how the data is accessed and modified.
- **Flexibility**: Easier to change and maintain code.
- **Improved Security**: Protects object integrity.

Understanding Encapsulation Through Real-Life Scenarios

Question 1: The Bank Account

Scenario:

Imagine a bank account where you can deposit and withdraw money.

A. **Which of the following represents encapsulation in this scenario?**

1. **Public access to the account balance.**
2. **Private variables for account balance and methods for deposit and withdrawal.**
3. **Sharing your PIN code with everyone.**

Answer: 2. Private variables for account balance and methods for deposit and withdrawal.

Bank Account

```
public class BankAccount {

    private double balance; // Private variable

    public BankAccount(double initialBalance) {
        this.balance = initialBalance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
        }
    }

    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
        }
    }

    public double getBalance() {
        return balance; // Controlled access to balance
    }
}
```

```
// Usage
public class Main {
    public static void main(String[] args) {
        BankAccount account = new BankAccount(1000.0);

        account.deposit(500);

        account.withdraw(200);

        System.out.println("Account Balance: $" + account.getBalance());
    }
}
```

Real Time Scenarios - Encapsulation

Question 2: The Remote Control

Scenario:

You have a remote control for your TV.

A. How does the remote control illustrate encapsulation?

1. You can see the internal wiring.
2. You have buttons to change channels and adjust volume, but you don’t need to know how it works internally.
3. The remote control has no buttons.

Answer: 2. You have buttons to change channels and adjust volume, but you don’t need to know how it works internally.

The Remote Control

```java
public class RemoteControl {

    private boolean isOn; // Private variable

    public void power() {
        isOn = !isOn; // Toggle power
    }

    public void changeChannel(int channel) {
        if (isOn) {
            System.out.println("Changing to channel: " + channel);
        } else {
            System.out.println("Remote is off.");
        }
    }

    public void adjustVolume(int level) {
        if (isOn) {
            System.out.println("Setting volume to: " + level);
        } else {
            System.out.println("Remote is off.");
        }
    }
}
```

```java
// Usage
public class Main {
    public static void main(String[] args) {
        RemoteControl remote = new RemoteControl();
        remote.power();
        remote.changeChannel(5);
        remote.adjustVolume(10);
    }
}
```

Real Time Scenarios - Encapsulation

Question 3: The Smartphone App

Scenario: **You have a banking app on your smartphone.**

A. Which of these features reflects encapsulation?

1. **The app shows all your transaction details and allows anyone to modify them.**
2. **The app requires a password for access, and you can perform transactions through secure methods without revealing your balance directly.**
3. **The app crashes frequently.**

Answer: 2. The app requires a password for access, and you can perform transactions through secure methods without revealing your balance directly.

The Smartphone App

```
public class BankingApp {
   private double balance; // Private variable
   private String password; // Private variable

   public BankingApp(double initialBalance, String password) {
      this.balance = initialBalance;
      this.password = password;
   }

   public boolean login(String password) {
      return this.password.equals(password); // Validate password
   }

   public void deposit(double amount) {
      if (amount > 0) {
         balance += amount;
      }
   }

   public double getBalance(String password) {
      if (login(password)) {
         return balance; // Controlled access to balance
      } else {
         System.out.println("Invalid password.");
         return -1;
      }
   }
}
```

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

      BankingApp app = new BankingApp(1500.0,
"securePassword");
      app.deposit(300);
      System.out.println("Account Balance: $" +
app.getBalance("securePassword"));

   }
}
```

Real Time Scenarios - Encapsulation

Question 4: The Library System

Scenario: In a library, books are categorized and managed by a system.

A. How does the library system exemplify encapsulation?

1. Everyone can change the book details directly.
2. The library staff has specific access to add or remove books while patrons can only check them out.
3. Books are left lying around without organization.

Answer: 2. **The library staff has specific access to add or remove books while patrons can only check them out.**

The Smartphone App

```
import java.util.HashMap;

public class Library {
    private HashMap<String, Boolean> books; // Private variable for books

    public Library() {
        books = new HashMap<>();
    }

    public void addBook(String title) {
        books.put(title, true); // Book available
    }

    public boolean checkOutBook(String title) {
        if (books.containsKey(title) && books.get(title)) {
            books.put(title, false); // Mark book as checked out
            return true;
        }
        return false; // Book not available
    }

    public void returnBook(String title) {
        if (books.containsKey(title)) {
            books.put(title, true); // Mark book as available
        }
    }
}
```

```
// Usage
public class Main {
    public static void main(String[] args) {
        Library library = new Library();
        library.addBook("The Great Gatsby");

        if (library.checkOutBook("The Great Gatsby")) {
            System.out.println("Checked out 'The Great Gatsby'");
        } else {
            System.out.println("'The Great Gatsby' is not available.");
        }

        library.returnBook("The Great Gatsby");
    }
}
```