

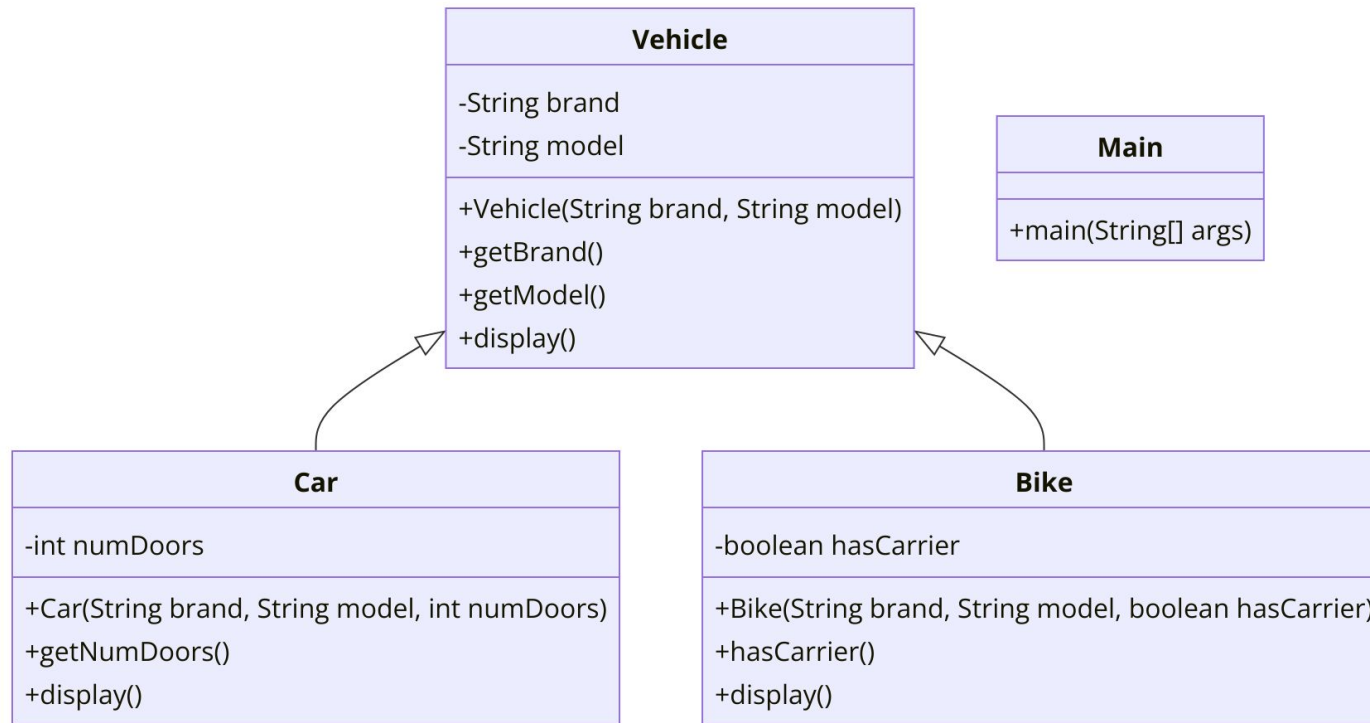
2024년 상반기 K-디지털 트레이닝

JAVA Inherit

[KB] IT's Your Life

Q1 - inherit

- 다음 클래스 다이어그램과 명세서를 보고 구현하시오.



The screenshot shows a Java IDE window titled "Run" with two tabs: "MovingTextAnimation" and "Main". The "Main" tab is active, displaying the output of a Java program. The output shows the results of creating a Car and a Bike object and calling their display methods.

```
Run
MovingTextAnimation x Main x

/Users/alicia/Library/Java/JavaVirtualMachines/corre
Brand: Toyota
Model: Camry
Number of Doors: 4
Brand: Yamaha
Model: MT-15
Has Carrier: true

Process finished with exit code 0
```

- **Vehicle 클래스**

- **필드**

- `brand (String)`: 차량의 브랜드를 나타냄.
`private` 접근 제한자.
 - `model (String)`: 차량의 모델을 나타냄. `private`
접근 제한자.

- **생성자**

- `Vehicle(String brand, String model)`:
브랜드와 모델을 초기화하는 생성자.

- **메서드**

- `getBrand()`: 브랜드를 반환하는 `public` 메서드.
 - `getModel()`: 모델을 반환하는 `public` 메서드.
 - `display()`: 차량의 브랜드와 모델을 출력하는
`public` 메서드.

- **Car 클래스 (Vehicle 클래스 상속)**

- **필드**

- `numDoors (int)`: 자동차의 문 수를 나타냄.
`private` 접근 제한자.

- **생성자**

- `Car(String brand, String model, int
numDoors)`: 브랜드, 모델, 문 수를 초기화하는
생성자.

- **메서드**

- `getNumDoors()`: 문 수를 반환하는 `public`
메서드.
 - `display()`: `Vehicle` 클래스의 `display()`
메서드를 오버라이드하여 브랜드, 모델, 문
수를 출력하는 메서드.

- **Bike 클래스 (Vehicle 클래스 상속)**

- 필드

- `hasCarrier (boolean)`: 자전거의 캐리어 유무를 나타냄. `private` 접근 제한자.

- 생성자

- `Bike(String brand, String model, boolean hasCarrier)`: 브랜드, 모델, 캐리어 유무를 초기화하는 생성자.

- 메서드

- `hasCarrier()`: 캐리어 유무를 반환하는 `public` 메서드.
 - `display()`: `Vehicle` 클래스의 `display()` 메서드를 오버라이드하여 브랜드, 모델, 캐리어 유무를 출력하는 메서드.

- **Main 클래스**

- 메서드

- `main(String[] args)`: 프로그램의 진입점인 메인 메서드. `Car`와 `Bike` 객체를 생성하고, 각 객체의 `display()` 메서드를 호출하여 정보를 출력함.

- 다음 코드를 수정하여 프로그래밍하시오.(상속 이용)

```
class Smartphone { // 스마트폰
    private int batteryCapacity; // 배터리 용량
    private String color; // 색상
    private int cameraResolution; // 카메라 해상도

    public int getBatteryCapacity() { return batteryCapacity; }
    public void setBatteryCapacity(int capacity) { this.batteryCapacity = capacity; }
    public String getColor() { return color; }
    public void setColor(String color) { this.color = color; }
    public int getCameraResolution() { return cameraResolution; }
    public void setCameraResolution(int resolution) { this.cameraResolution = resolution; }
}

class Tablet { // 태블릿
    private int batteryCapacity; // 배터리 용량
    private String color; // 색상
    private boolean isConnected; // 연결 상태

    public int getBatteryCapacity() { return batteryCapacity; }
    public void setBatteryCapacity(int capacity) { this.batteryCapacity = capacity; }
    public String getColor() { return color; }
    public void setColor(String color) { this.color = color; }
    public boolean getIsConnected() { return isConnected; }
    public void setIsConnected(boolean status) { this.isConnected = status; }
}
```

```
class Laptop { // 노트북
    private int batteryCapacity; // 배터리 용량
    private String color; // 색상

    public int getBatteryCapacity() { return batteryCapacity; }
    public void setBatteryCapacity(int capacity) { this.batteryCapacity = capacity; }
    public String getColor() { return color; }
    public void setColor(String color) { this.color = color; }
    public void charge(int additionalCapacity) { this.batteryCapacity += additionalCapacity; }
}
```



```
Run

Run Main x

/Users/administrator/Library/Java/JavaVirtualMachines/corretto-17.0.10/Contents/Home/bin/java ...
Smartphone{cameraResolution=20, batteryCapacity=80} Move{batteryCapacity=80, color='red'}
Tablet{isConnected=false, batteryCapacity=70} Move{batteryCapacity=70, color='blue'}
Laptop{batteryCapacity=100} Move{batteryCapacity=100, color='silver'}

Process finished with exit code 0
```

수고하셨습니다!

