

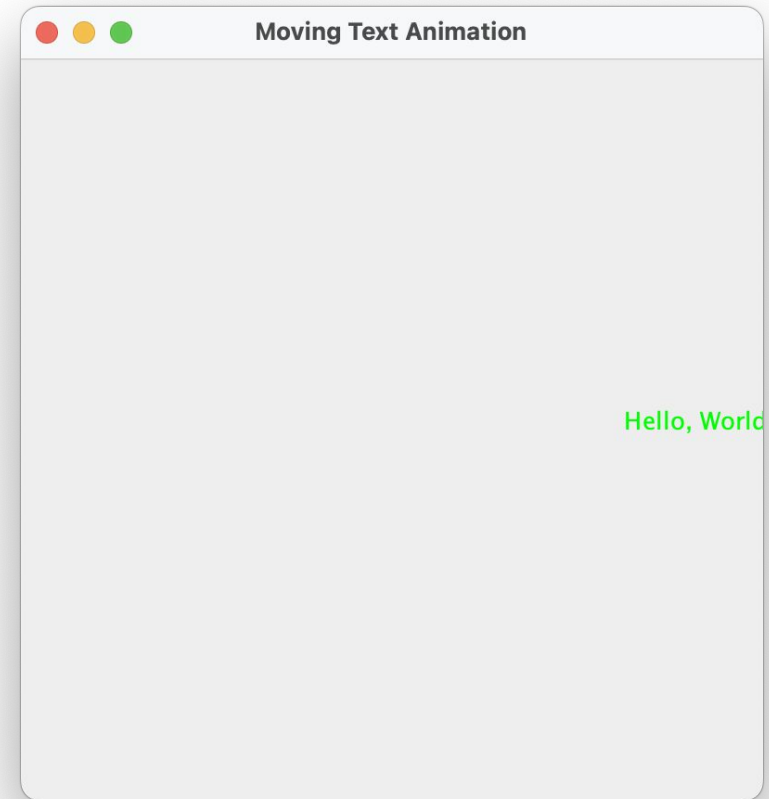
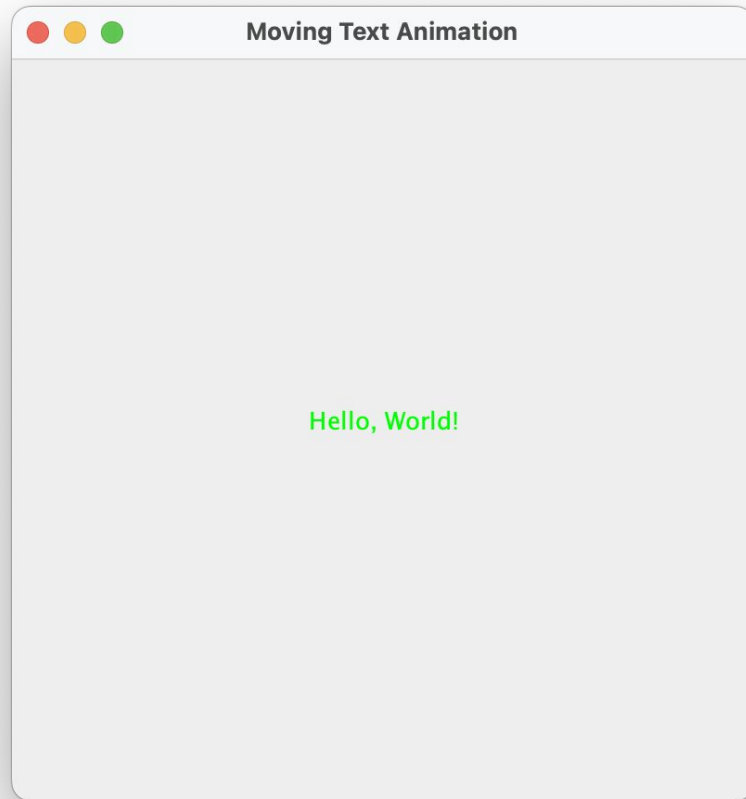
2024년 상반기 K-디지털 트레이닝

# JAVA inherit

---

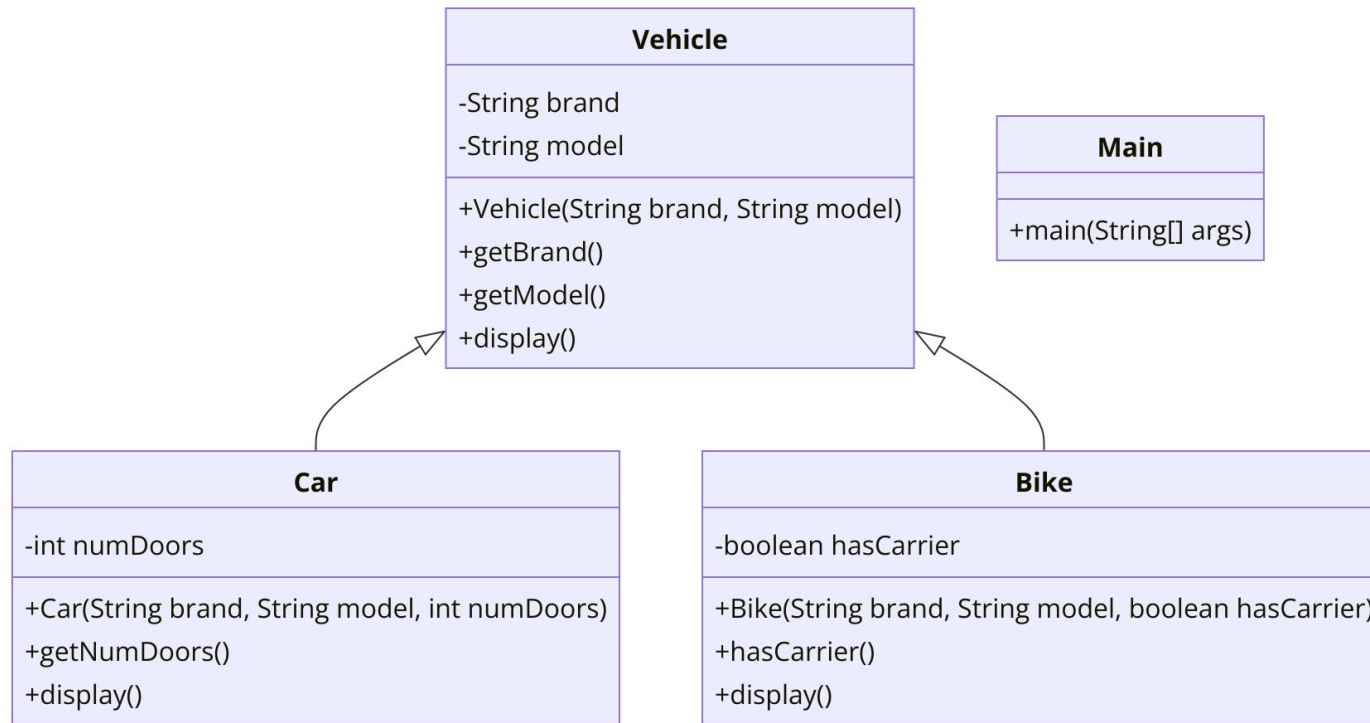
[KB] IT's Your Life

- 다음과 같이 움직이는 화면을 프로그래밍하시오.



## Q2 - 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 program. The output shows the details of a Toyota Camry and a Yamaha MT-15 motorcycle.

```
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**

- 속성

- `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)**

- 속성

- `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)**

- 속성

- `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

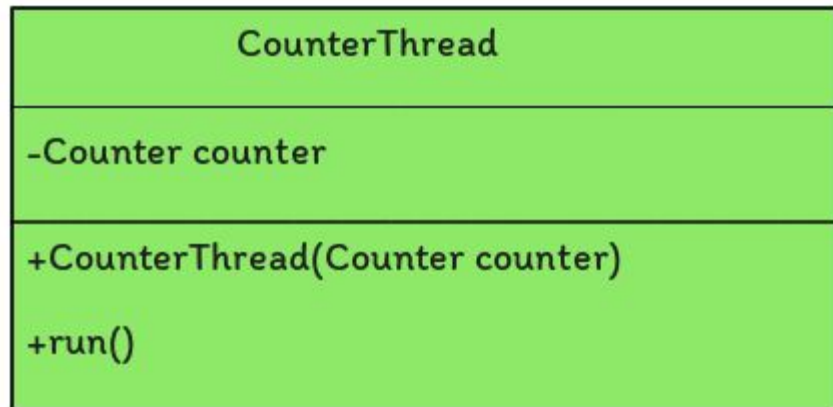
- `main(String[] args)`: 프로그램의 진입점인 메인 메서드. `Car`와 `Bike` 객체를 생성하고, 각 객체의 `display()` 메서드를 호출하여 정보를 출력함.

- 스레드를 3개 만들어 Counter변수를 공유하여 동시에 접근하도록 프로그래밍하시오.

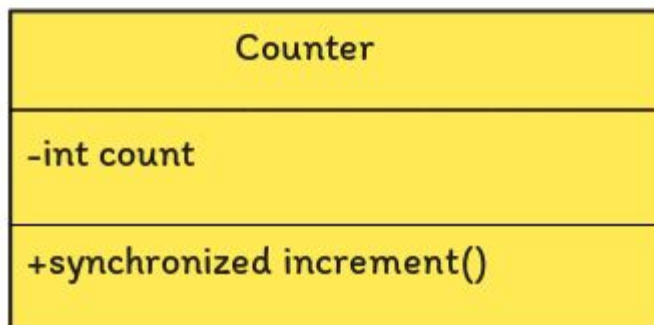
```
public class Counter {  
    private int count = 0;  
  
    public synchronized void increment() {  
        count++;  
        System.out.println(Thread.currentThread().getName() + " -  
Count: " + count);  
    }  
}
```



```
Run  
Run CircleAnimation x CounterThread x  
/Users/alicia/Library/Java/JavaVirtualMachines/corre  
Thread-0 - Count: 1  
Thread-2 - Count: 2  
Thread-1 - Count: 3  
Thread-1 - Count: 4  
Thread-0 - Count: 5  
Thread-2 - Count: 6  
Thread-2 - Count: 7  
Thread-0 - Count: 8  
Thread-1 - Count: 9  
Thread-2 - Count: 10  
Thread-1 - Count: 11  
Thread-0 - Count: 12  
Thread-2 - Count: 13  
Thread-1 - Count: 14  
Thread-0 - Count: 15  
Thread-2 - Count: 16  
Thread-1 - Count: 17  
Thread-0 - Count: 18  
Thread-2 - Count: 19  
Thread-1 - Count: 20  
Thread-0 - Count: 21  
Thread-2 - Count: 22  
Thread-0 - Count: 23  
Thread-1 - Count: 24  
Thread-2 - Count: 25  
Thread-0 - Count: 26  
Thread-1 - Count: 27  
Thread-2 - Count: 28  
Thread-1 - Count: 29  
Thread-0 - Count: 30  
  
Process finished with exit code 0
```



Uses



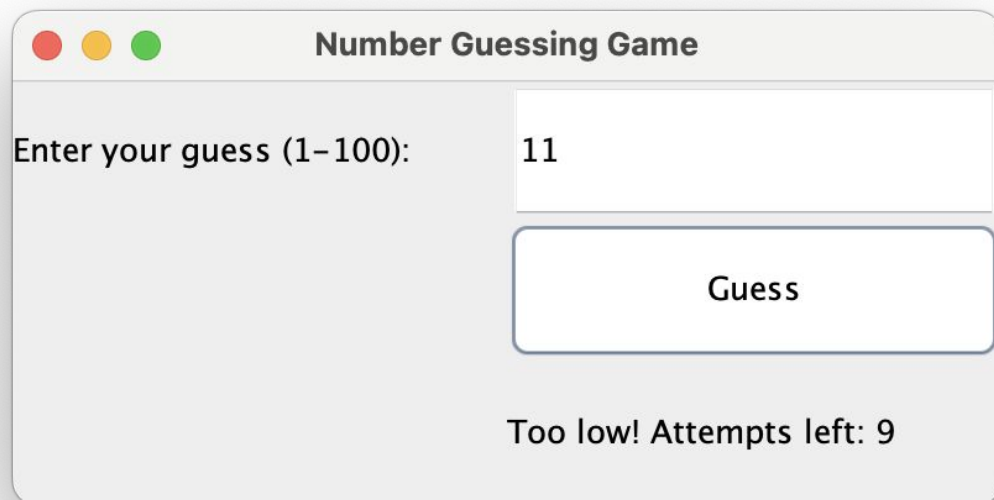
## ● Counter 클래스

- count 변수는 private 접근 제어자를 가지며 초기 값은 0
- increment 메서드는 동기화되어 있고 count를 증가시키고 현재 스레드 이름과 함께 출력함

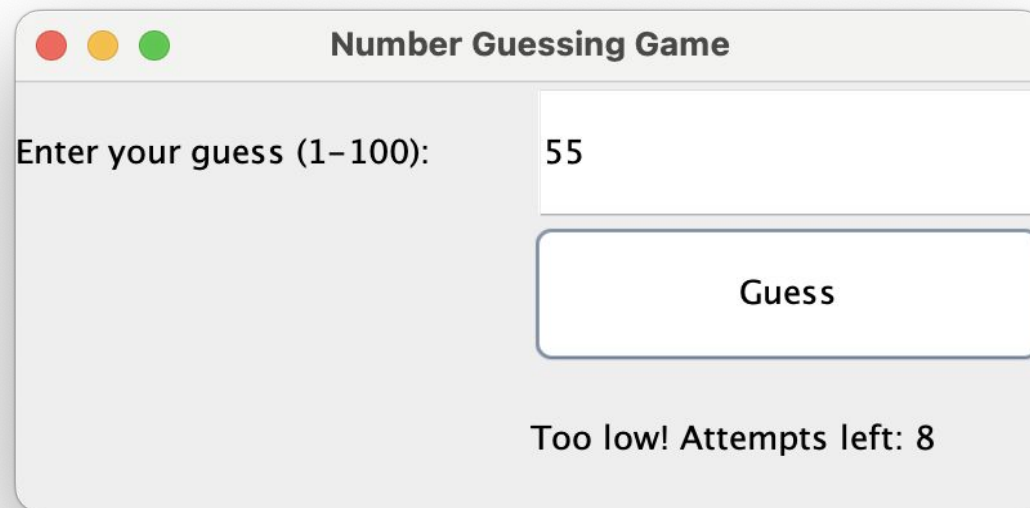
## ● CounterThread 클래스

- Counter 타입의 counter 변수를 가지고 있으며, 이는 생성자에서 초기화
- run 메서드는 10번의 루프 동안 counter의 increment 메서드를 호출하며 각 호출 사이에 0.1초 대기

- 다음 화면과 명세서를 보고 게임 프로그램을 구현하시오.

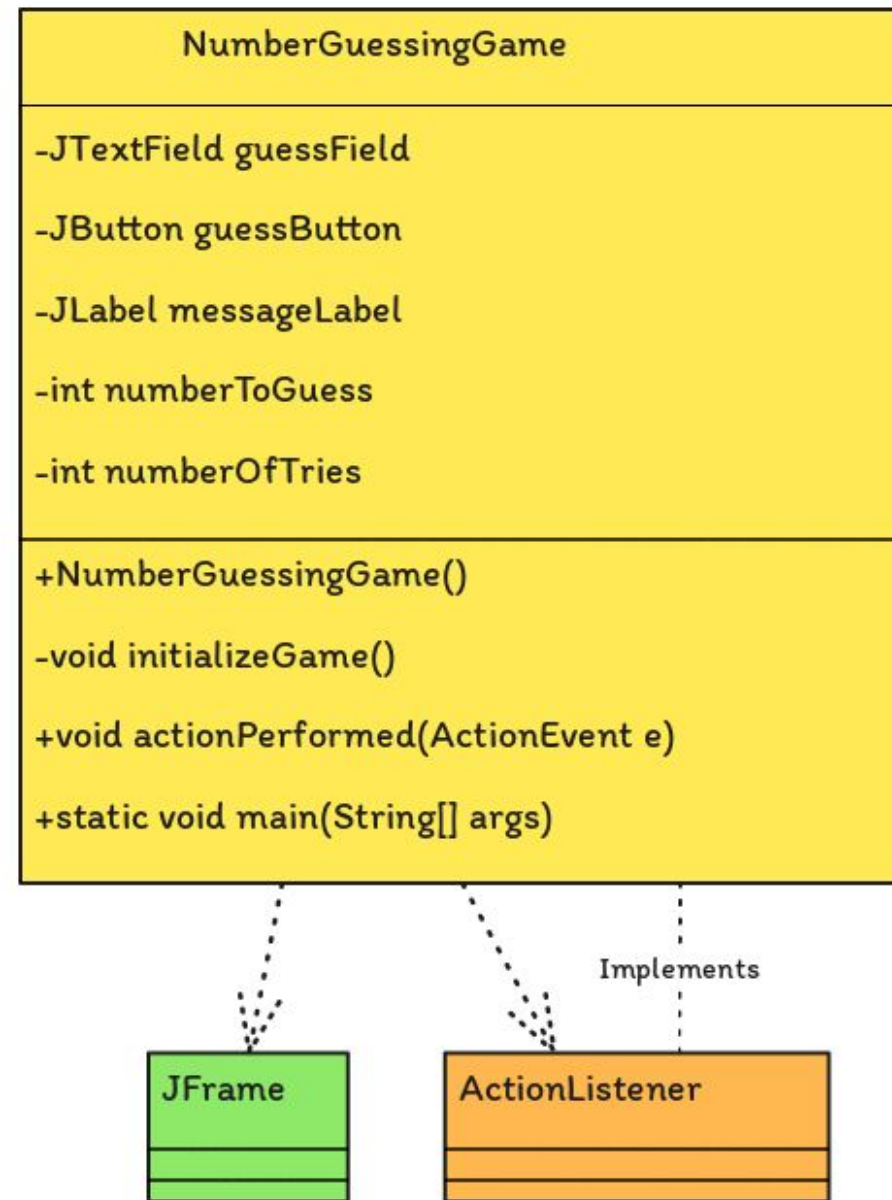
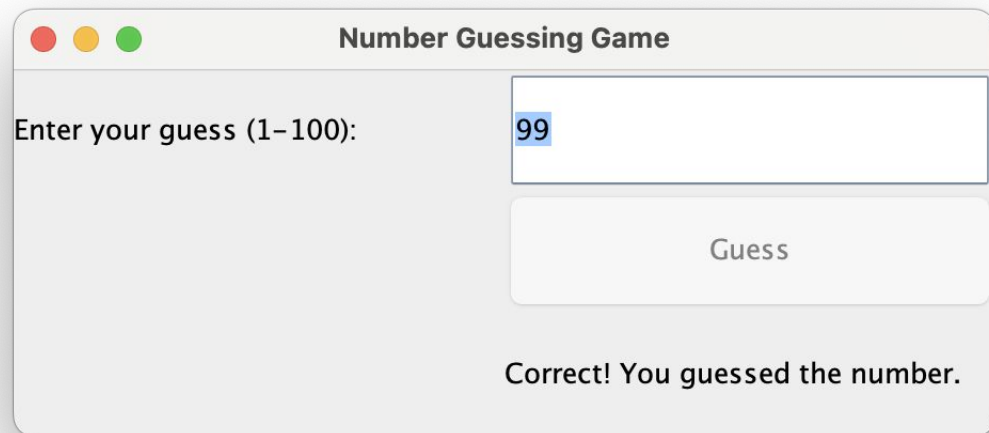


A screenshot of a 'Number Guessing Game' window. The title bar has three colored buttons (red, yellow, green) and the text 'Number Guessing Game'. The main area contains the text 'Enter your guess (1-100):' followed by a text input field containing the number '11'. Below the input field is a button labeled 'Guess'. At the bottom of the window, the text 'Too low! Attempts left: 9' is displayed.



A screenshot of a 'Number Guessing Game' window. The title bar has three colored buttons (red, yellow, green) and the text 'Number Guessing Game'. The main area contains the text 'Enter your guess (1-100):' followed by a text input field containing the number '55'. Below the input field is a button labeled 'Guess'. At the bottom of the window, the text 'Too low! Attempts left: 8' is displayed.





- **NumberGuessingGame** 클래스

- `TextField guessField`
- `Button guessButton`
- `Label messageLabel`
- `int numberToGuess`
- `int numberOfTries`
- `NumberGuessingGame()`: 생성자
- `initializeGame()`: 게임 초기화 메서드
- `actionPerformed(ActionEvent e)`: 이벤트 처리 메서드
- `static void main(String[] args)`: 메인 메서드

- **상속 및 구현 관계**

- `NumberGuessingGame` 클래스는 `JFrame`을 상속
- `NumberGuessingGame` 클래스는 `ActionListener` 인터페이스를 구현

