CSE2016

프로그램설계방법론

# MVC 아키텍처 설계 기초

## MVC Architecture Design Fundamentals

**도경구**

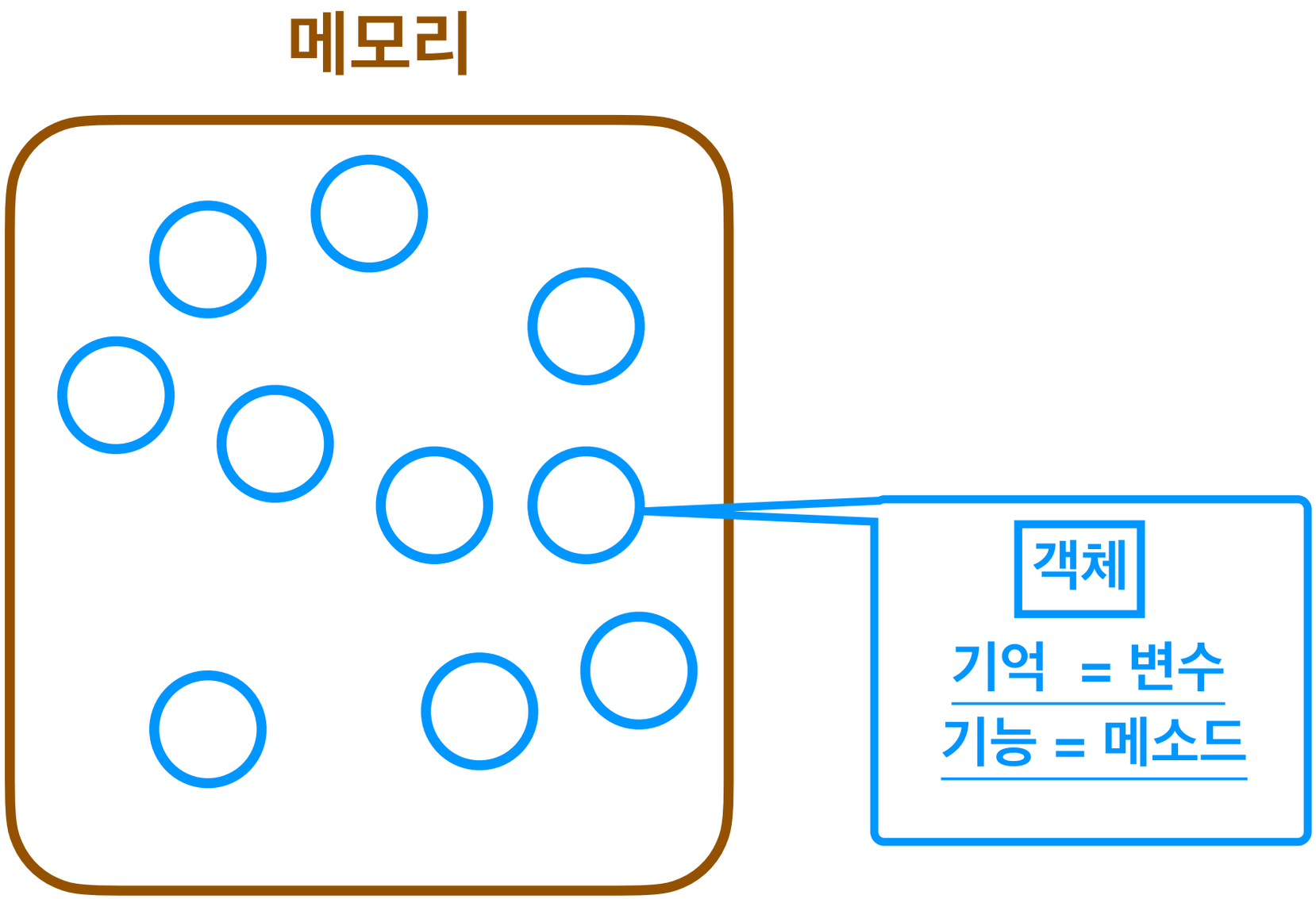
한양대학교 ERICA 소프트웨어학부

# 소프트웨어 아키텍처

- 건축은 설계와 공사를 분리
  - 건축가architect : 건물을 어떻게 지을지 전체적인 구성을 설계
  - 시공자builber : 건물 공사 담당
- 소프트웨어도 설계와 구현을 분리
  - 설계design : 소프트웨어 구조 설계
  - 구현implementation : 프로그램 작성

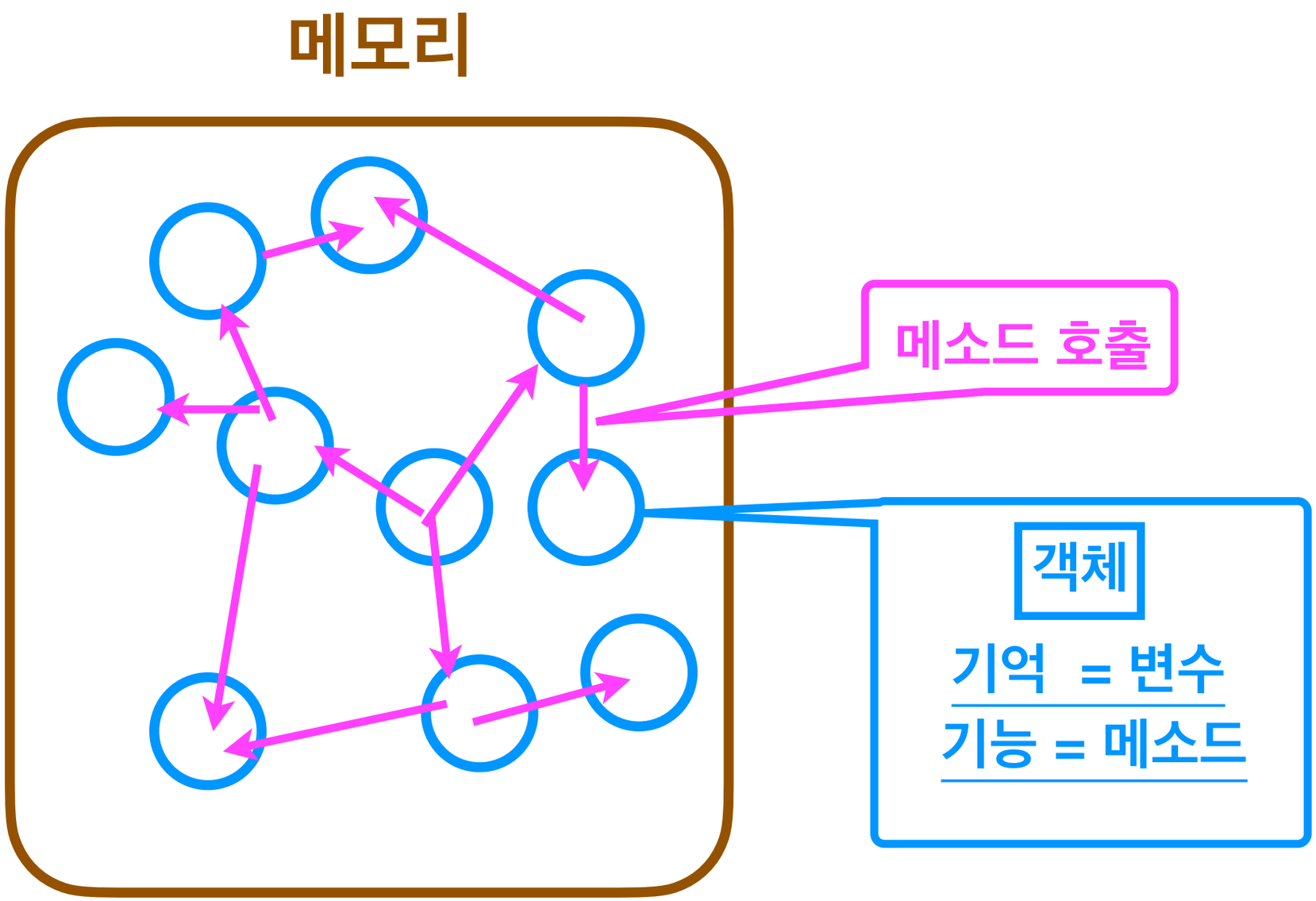
# 객체지향 프로그램 실행 원리

## Object-Oriented Program



# 객체지향 프로그램 실행 원리

## Object-Oriented Program



# 클래스의 종류

## Class

| 구분 | 역할 |
|---|---|
| Starter | 시동 걸기 |
| **M**odel | 핵심 계산을 하는 두뇌<br>선수 |
| **V**iew | 외부와 소통 창구<br>Input View / Output View |
| **C**ontroller | 매니저<br>지휘자 |

## MVC (Model-View-Controller) Architecture

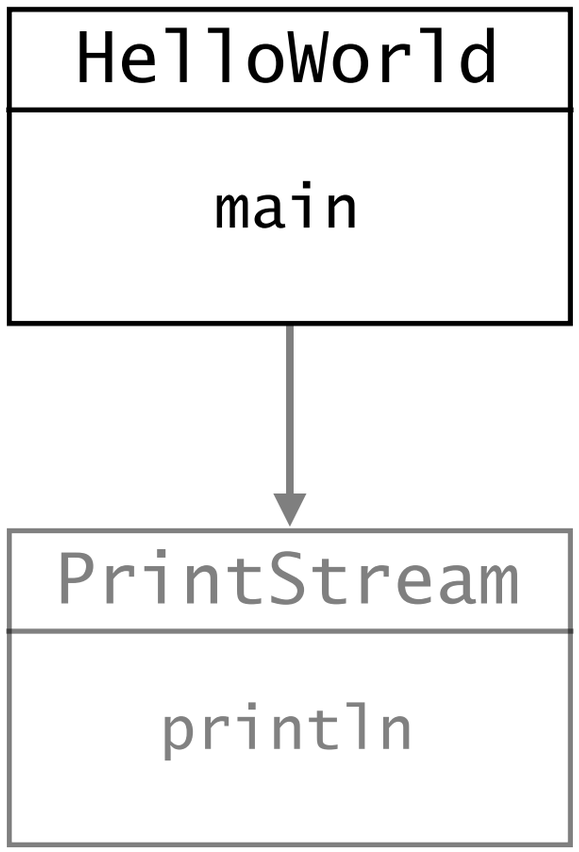
# 객체지향 설계
## Object-Oriented Program Design

- **설계**
  - 설계도 = 클래스 다이어그램 class diagram
  - 설계도 부품 = 클래스 class = 객체object를 만들기 위한 형틀
  - 클래스 별로 필요한 메소드 method 장착
- **구현**
  - 클래스 구현
  - 메소드 구현

## 설계

클래스 다이어그램
Class Diagram

**Controller** **Model**




**View**

---

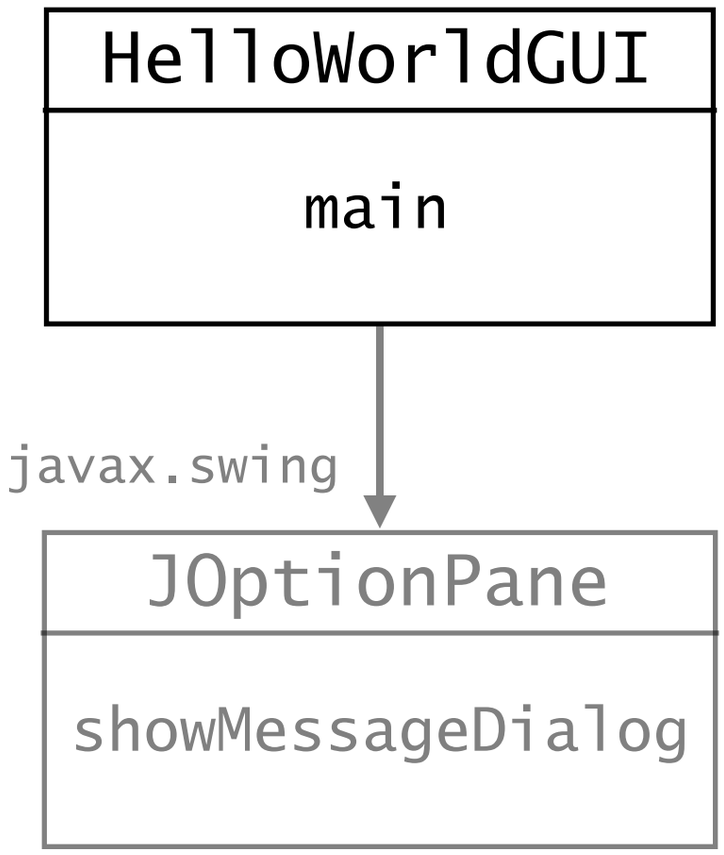
## 구현

애플리케이션
Application

```
1 public class HelloWorld {
2     public static void main(String[] args) {
3         System.out.println("Hello, World!");
4     }
5 }
```

## 설계

클래스 다이어그램
Class Diagram

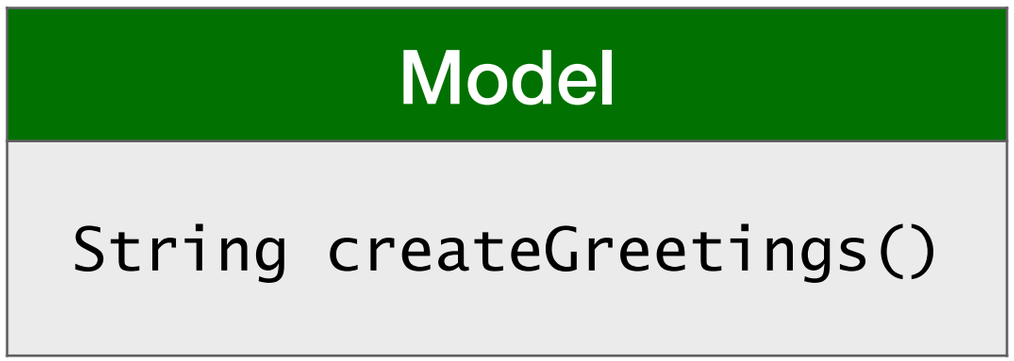



## 구현

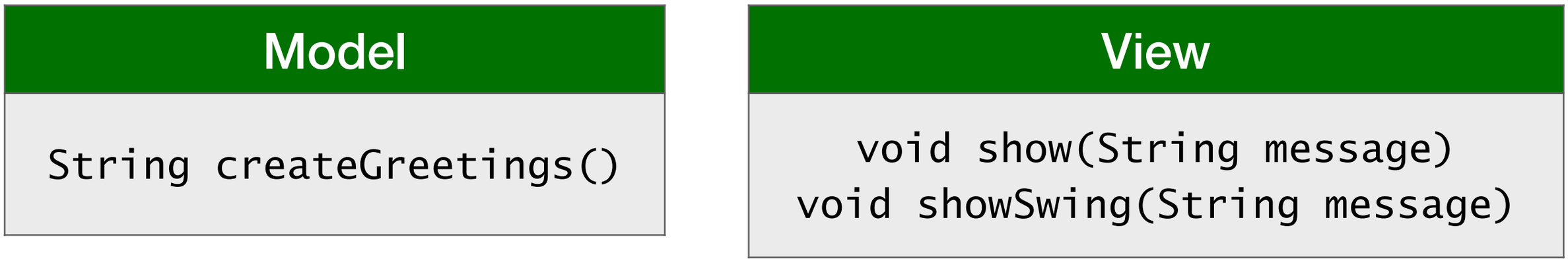
애플리케이션
Application

```
1 import javax.swing.*;
2
3 public class HelloWorldGUI {
4     public static void main(String[] args) {
5         JOptionPane.showMessageDialog(null, "Hello, World!");
6     }
7 }
```

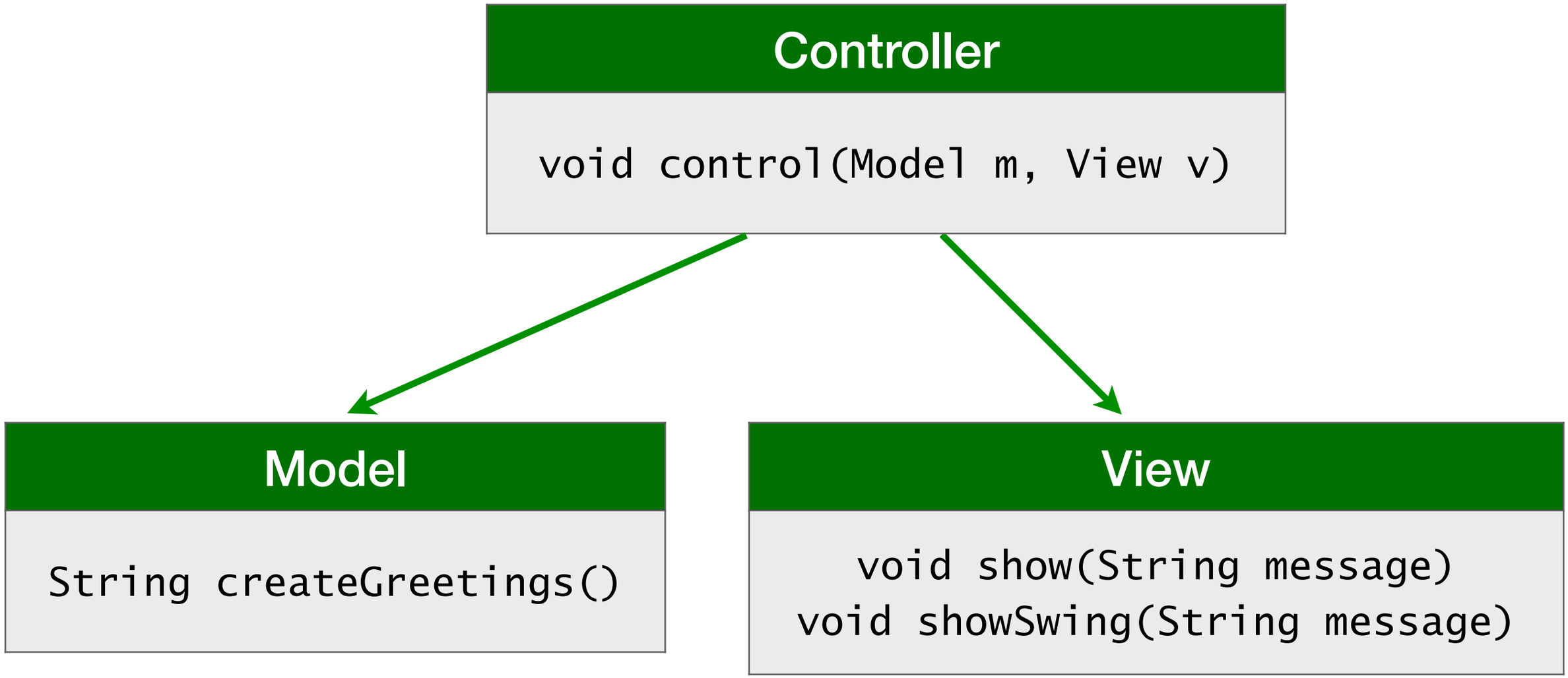
# 클래스 다이어그램 Class Diagram



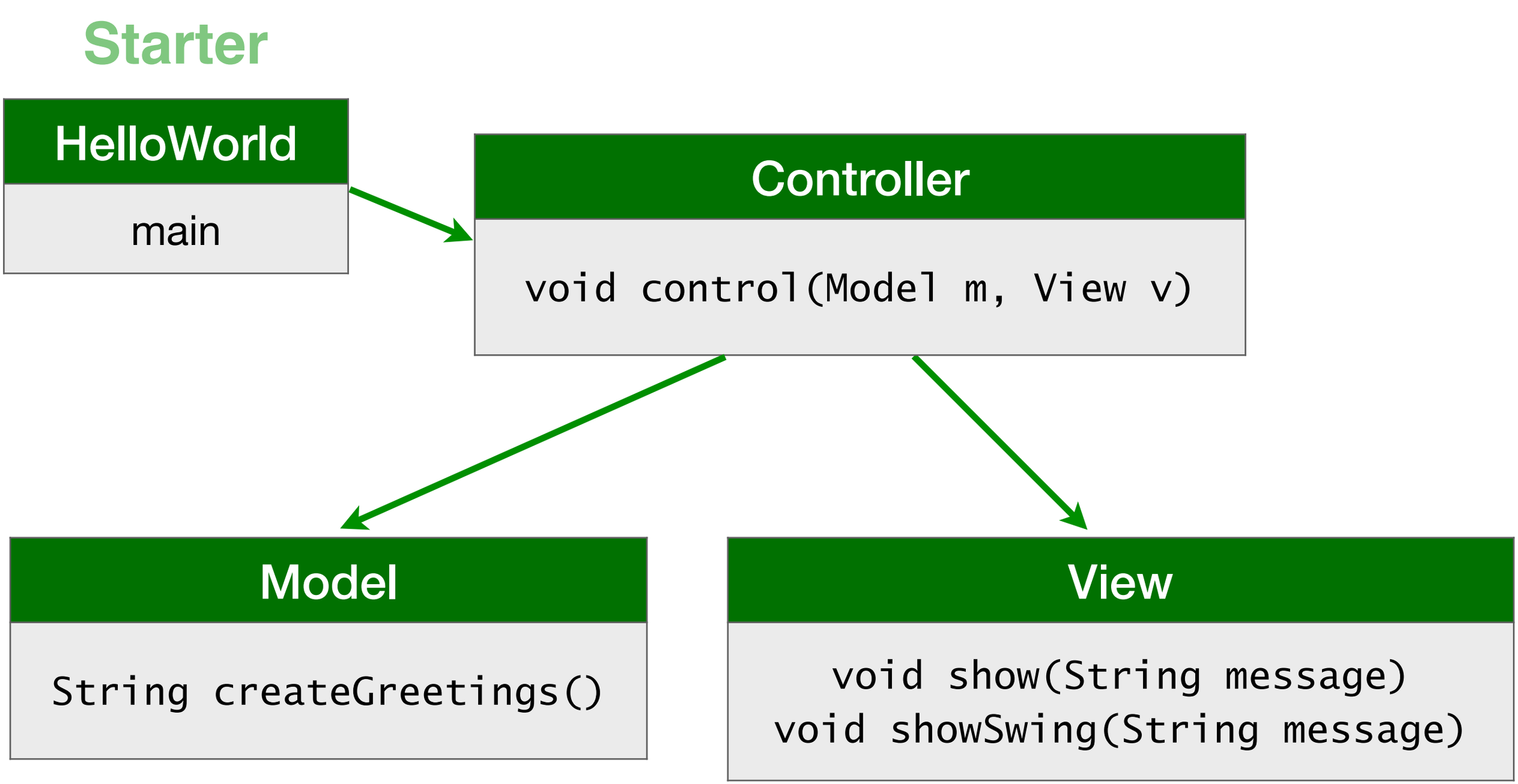
# 클래스 다이어그램 Class Diagram



# 클래스 다이어그램 Class Diagram

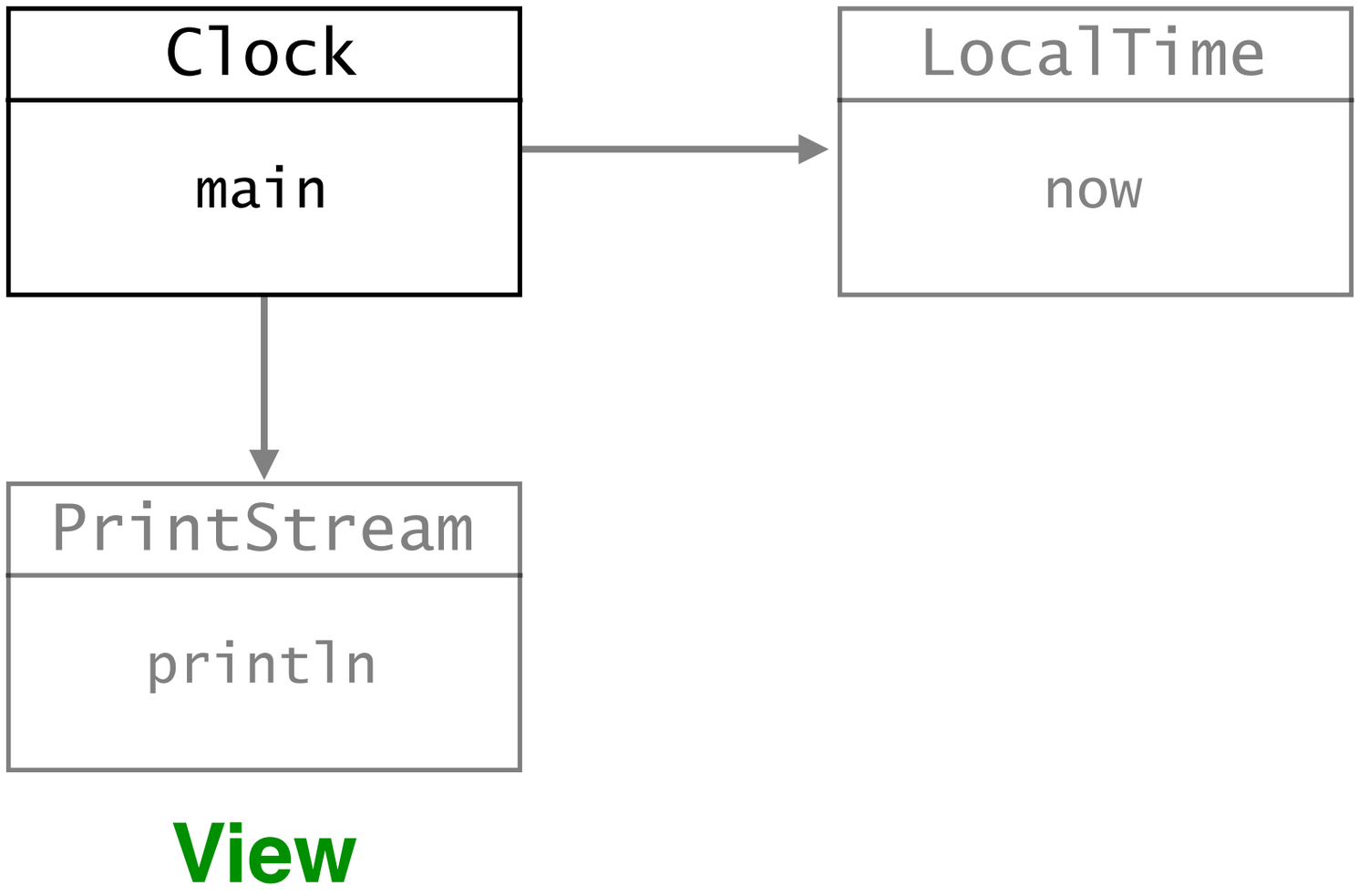


# 클래스 다이어그램 Class Diagram



# 설계

클래스 다이어그램
Class Diagram



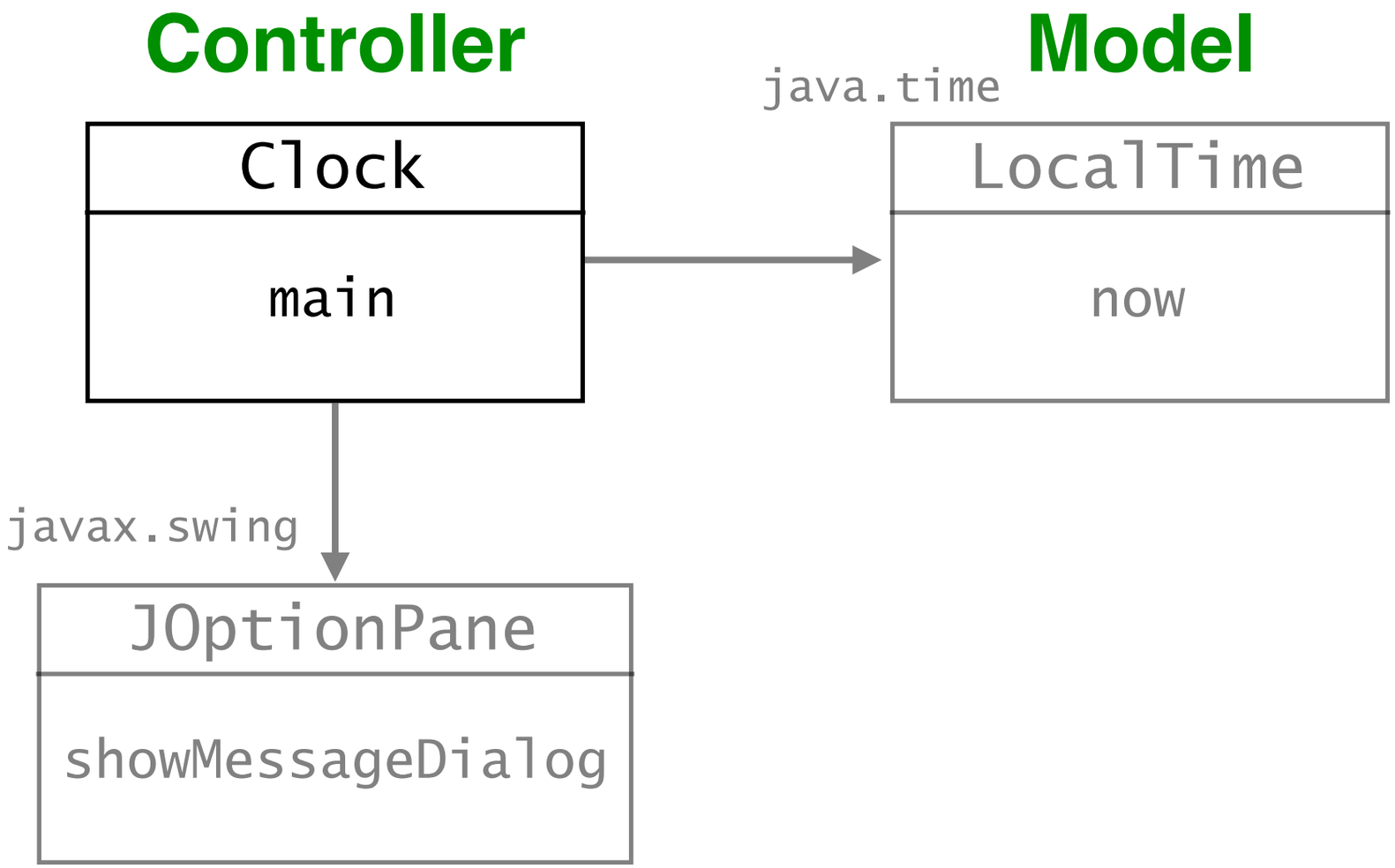

# 구현

애플리케이션
Application

```
1 import java.time.*;
2
3 public class Clock {
4     public static void main(String[] args) {
5         System.out.println(LocalTime.now());
6     }
7 }
```

# 설계

클래스 다이어그램
Class Diagram

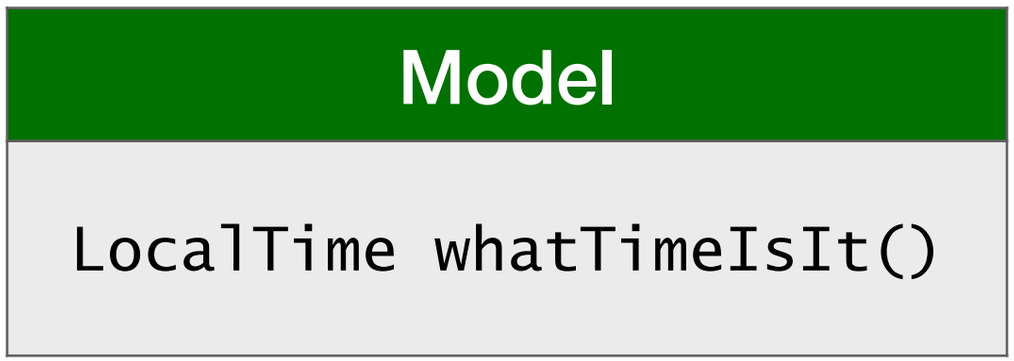



# 구현

애플리케이션
Application

```
1 import java.time.*;
2 import javax.swing.*;
3
4 public class Clock {
5     public static void main(String[] args) {
6         JOptionPane.showMessageDialog(null, LocalTime.now());
7     }
8 }
```

# 클래스 다이어그램 Class Diagram

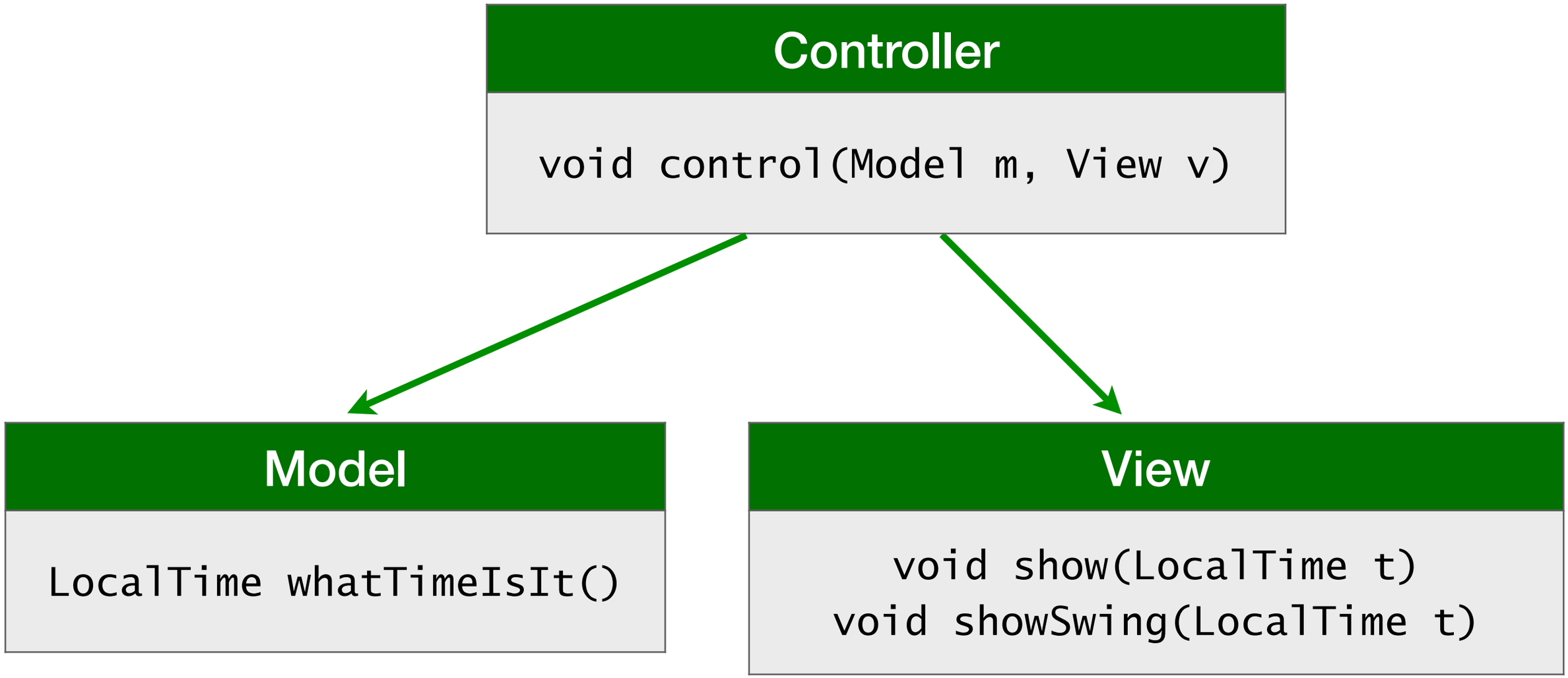


# 클래스 다이어그램 Class Diagram

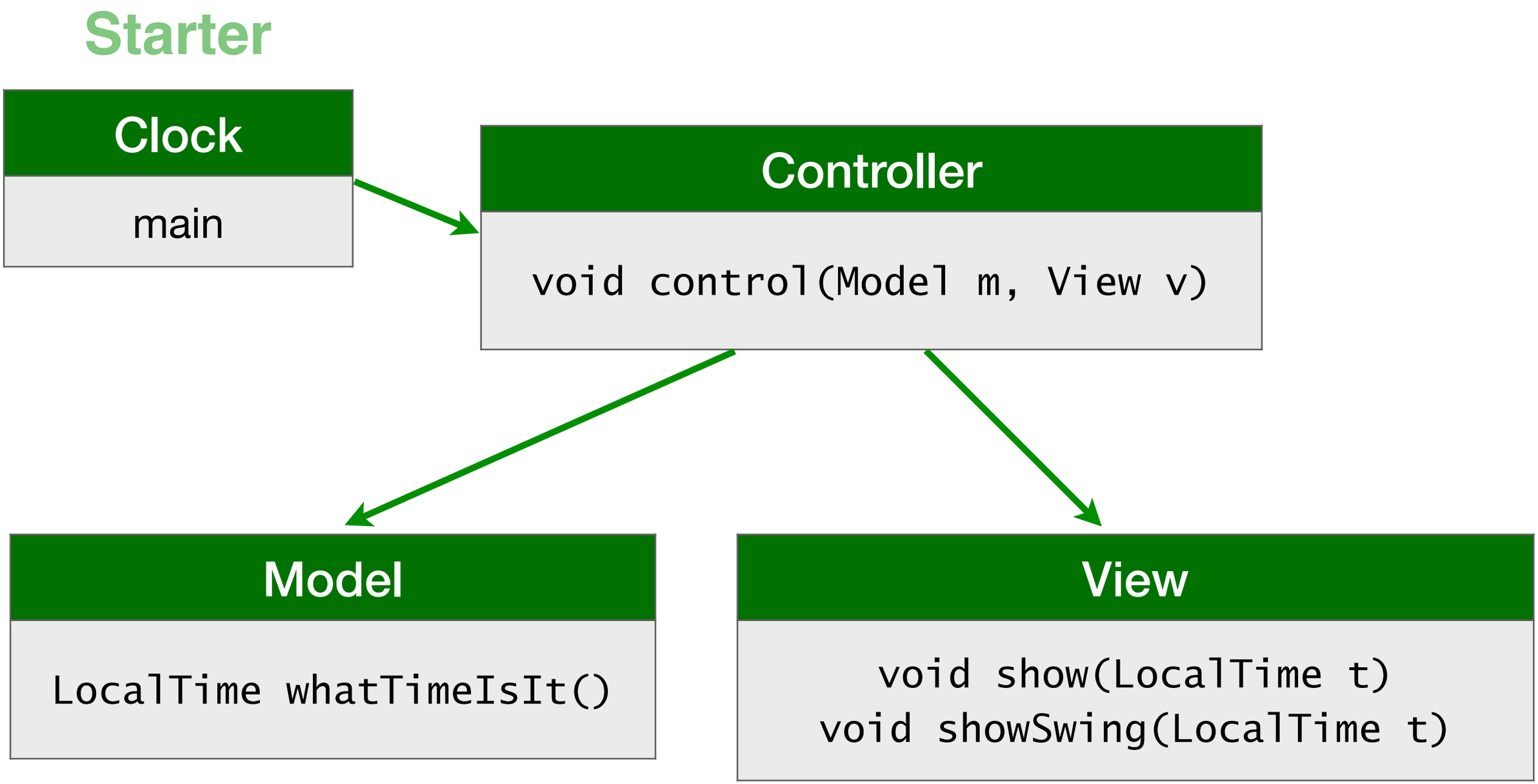
| Model |
| --- |
| LocalTime whatTimeIsIt() |

| View |
| --- |
| void show(LocalTime t)<br>void showSwing(LocalTime t) |

# 클래스 다이어그램 Class Diagram

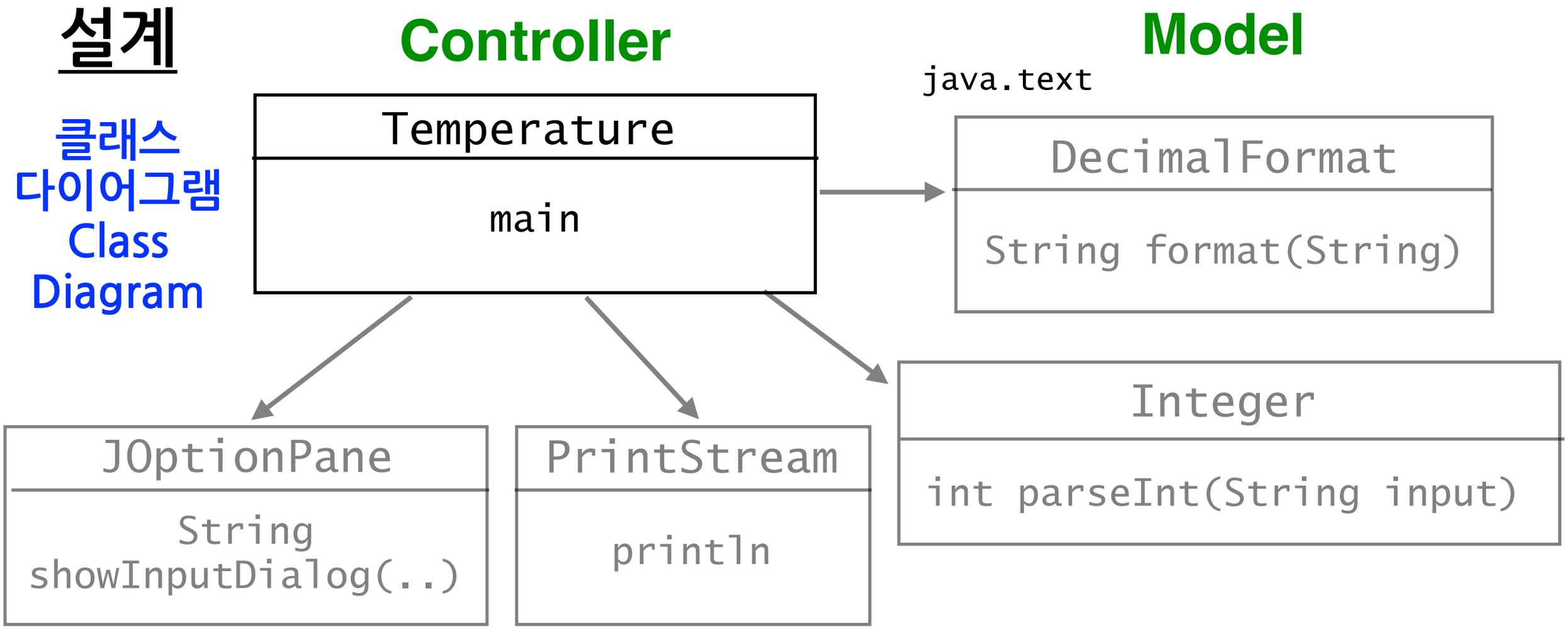


# 클래스 다이어그램 Class Diagram



# 설계

클래스 다이어그램
Class Diagram




# View

# 구현

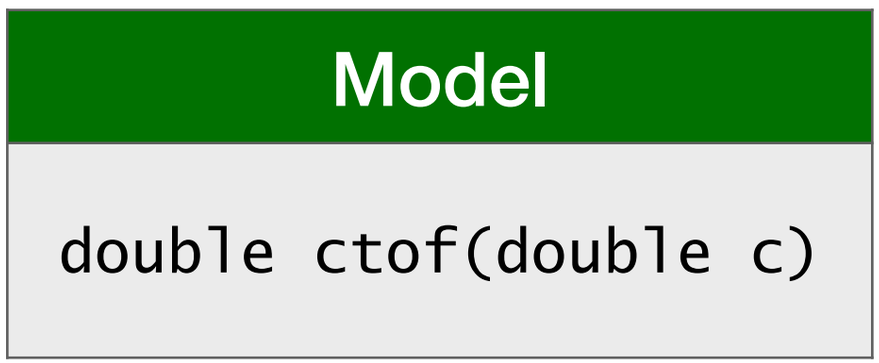
애플리케이션
Application

```java
import java.text.*;
import javax.swing.*;

public class CelsiusToFahrenheit {

    public static void main(String[] args) {
        String message = "섭씨 온도를 정수로 입력해주세요.";
        String input = JOptionPane.showInputDialog(message);
        int c = Integer.parseInt(input);
        double f = (9.0 / 5.0) * c + 32;
        DecimalFormat formatter = new DecimalFormat("0.0");
        System.out.print("섭씨 " + c + "도는 ");
        System.out.println("화씨로 " + formatter.format(f) + "도 입니다.");
    }
}
```

# 클래스 다이어그램 Class Diagram

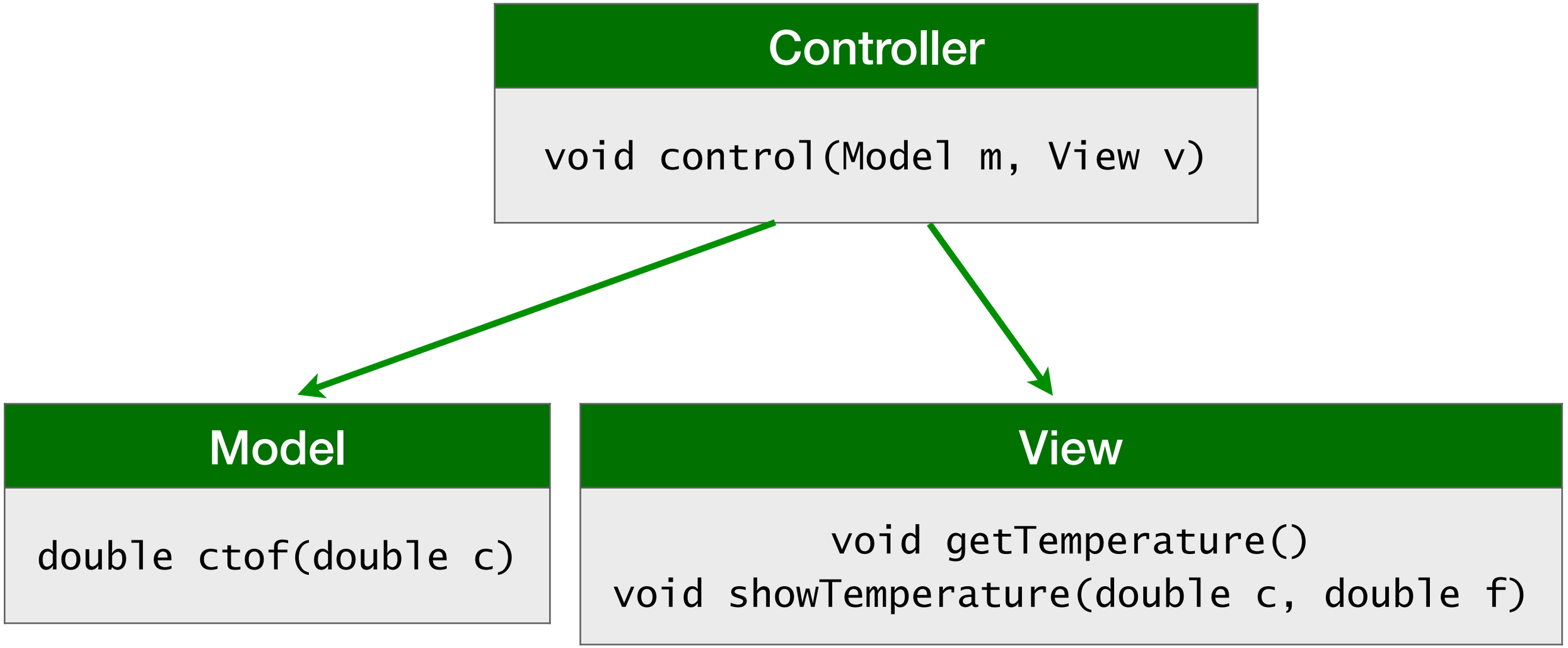


# 클래스 다이어그램 Class Diagram

| Model |
| --- |
| double ctof(double c) |

| View |
| --- |
| void getTemperature()<br>void showTemperature(double c, double f) |

# 클래스 다이어그램 Class Diagram



# 클래스 다이어그램 Class Diagram

## Starter

**Temperature**

main

**Controller**

```
void control(Model m, View v)
```

**Model**

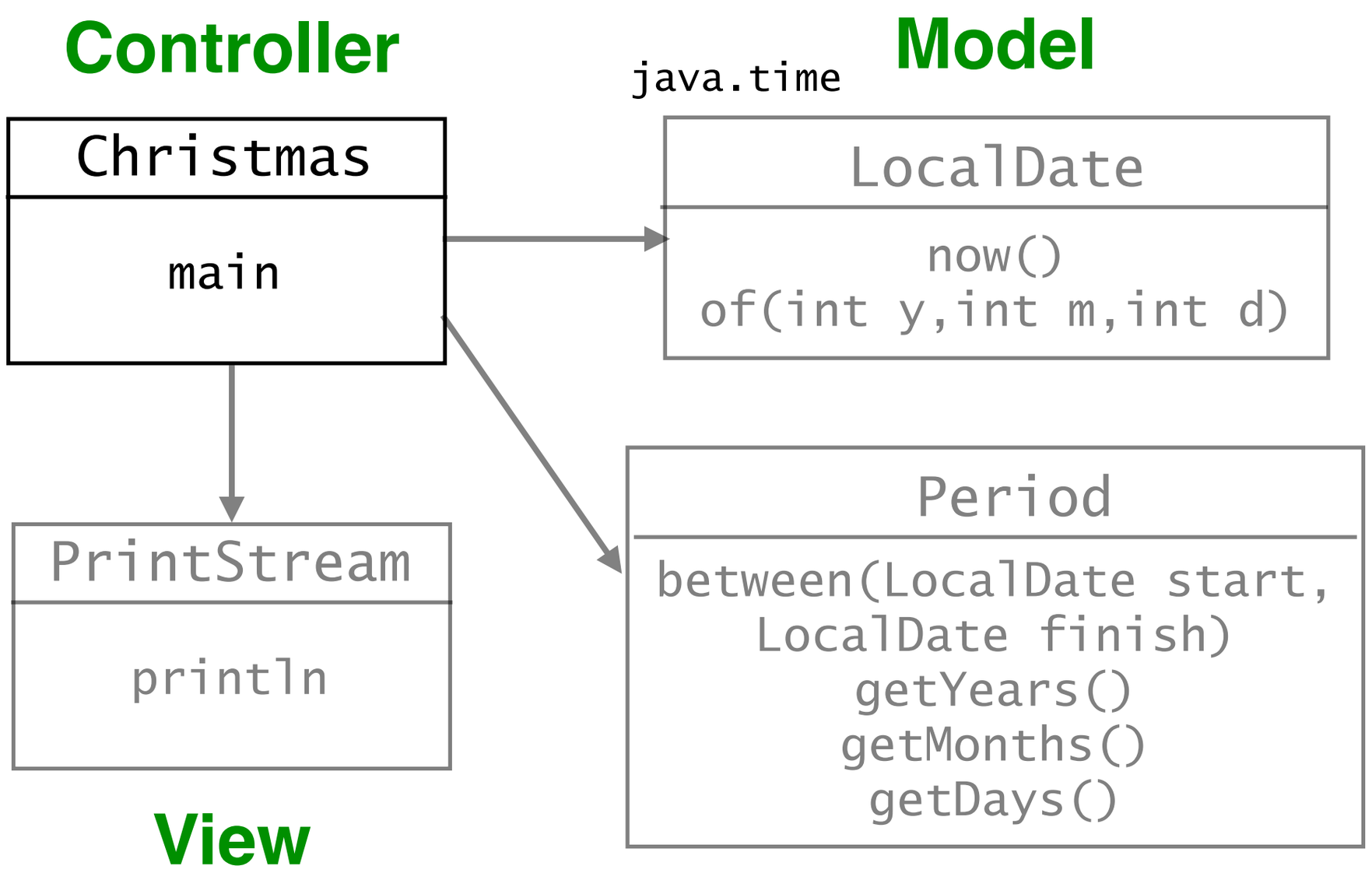
```
double ctof(double c)
```

**View**

```
void getTemperature()
void showTemperature(double c, double f)
```

# 설계

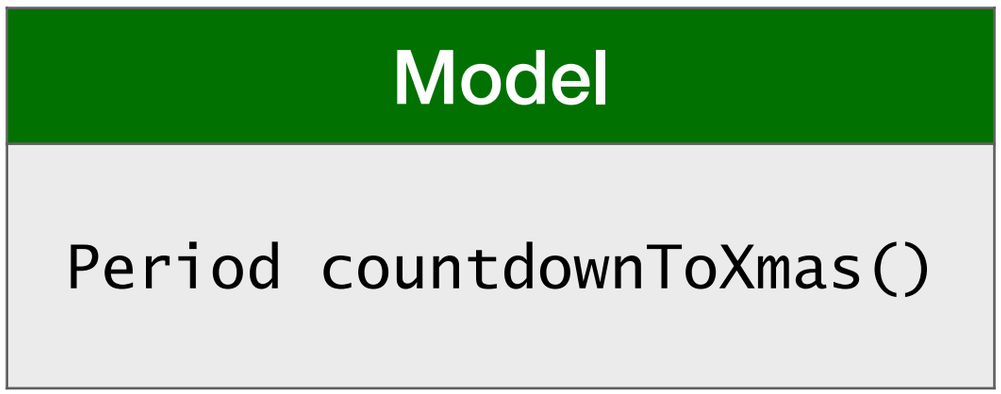
클래스
다이어그램
Class
Diagram




# 구현

애플리케이션
Application

# 클래스 다이어그램 Class Diagram

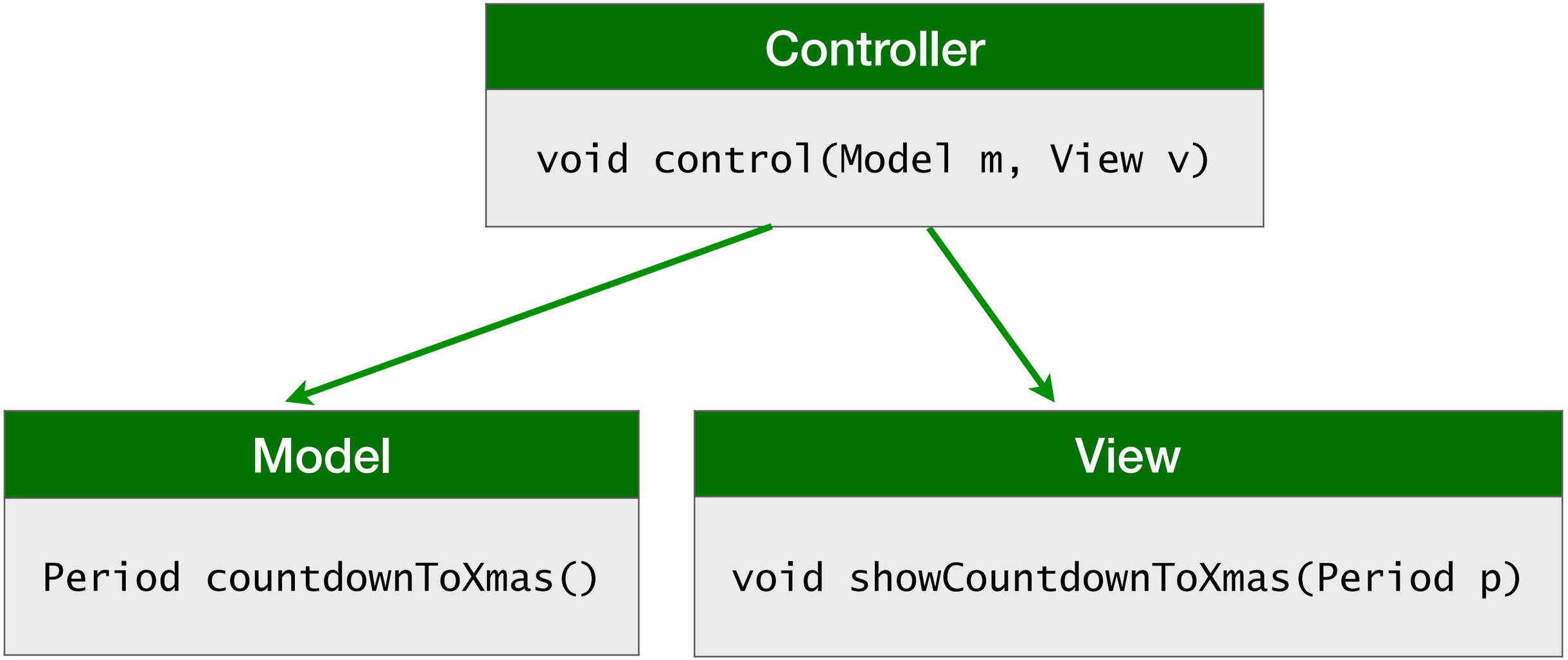


# 클래스 다이어그램 Class Diagram

| Model |
| --- |
| `Period countdownToXmas()` |

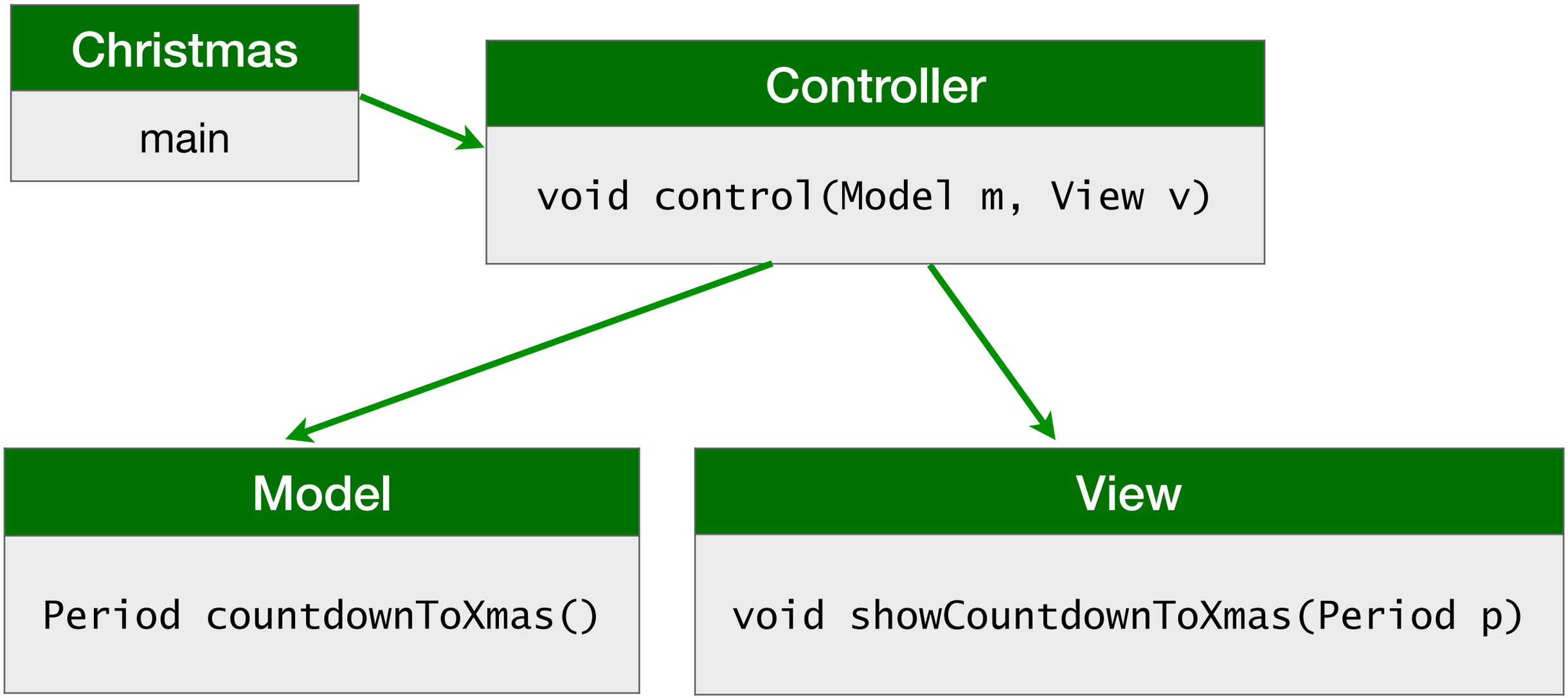
| View |
| --- |
| `void showCountdownToXmas(Period p)` |

# 클래스 다이어그램 Class Diagram



# 클래스 다이어그램 Class Diagram

Starter



# 소프트웨어 아키텍트

- 능력있는 프로그래머(소프트웨어 아키텍트)가 되려면?
    - 소프트웨어를 창작하는 기본기와 기법을 배우고,
    - 실제 사례를 가지고 설계해보고, 구현해보고, 분석해보고,
    - 연습하는데 부단히 시간과 노력을 투자