

4

생성 메소드와 필드 변수

Constructor Methods & Field Variables



클래스

class

선언

```
public class ClassName {  
    // code  
}
```

호출

```
new ClassName();
```

ClassName 객체object 생성

메소드 method

선언

```
public <return_type> methodName(<type_1> par_1, ..., <type_n> par_n) {  
    // code  
}
```

호출

```
methodName(<exp_1>, ..., <exp_n>);
```

메소드 method

공개

리턴 타입

parameter

0개 이상의 파라미터 나열

선언

```
public <return_type> methodName(<type_1> par_1, ..., <type_n> par_n) {  
    // code  
}
```

개수 일치와
타입 부합

호출

```
methodName(<exp_1>, ..., <exp_n>);
```

0개 이상의 인수 나열
argument

생성 메소드

constructor method

선언

```
public class ClassName {  
    public ClassName(<type_1> par_1, ..., <type_n> par_n);  
    // code  
}
```

클래스 이름과 동일

개수 일치와
타입 부합

호출

```
new ClassName(<exp_1>, ..., <exp_n>);
```

객체 생성시
한번만 실행

사례 학습

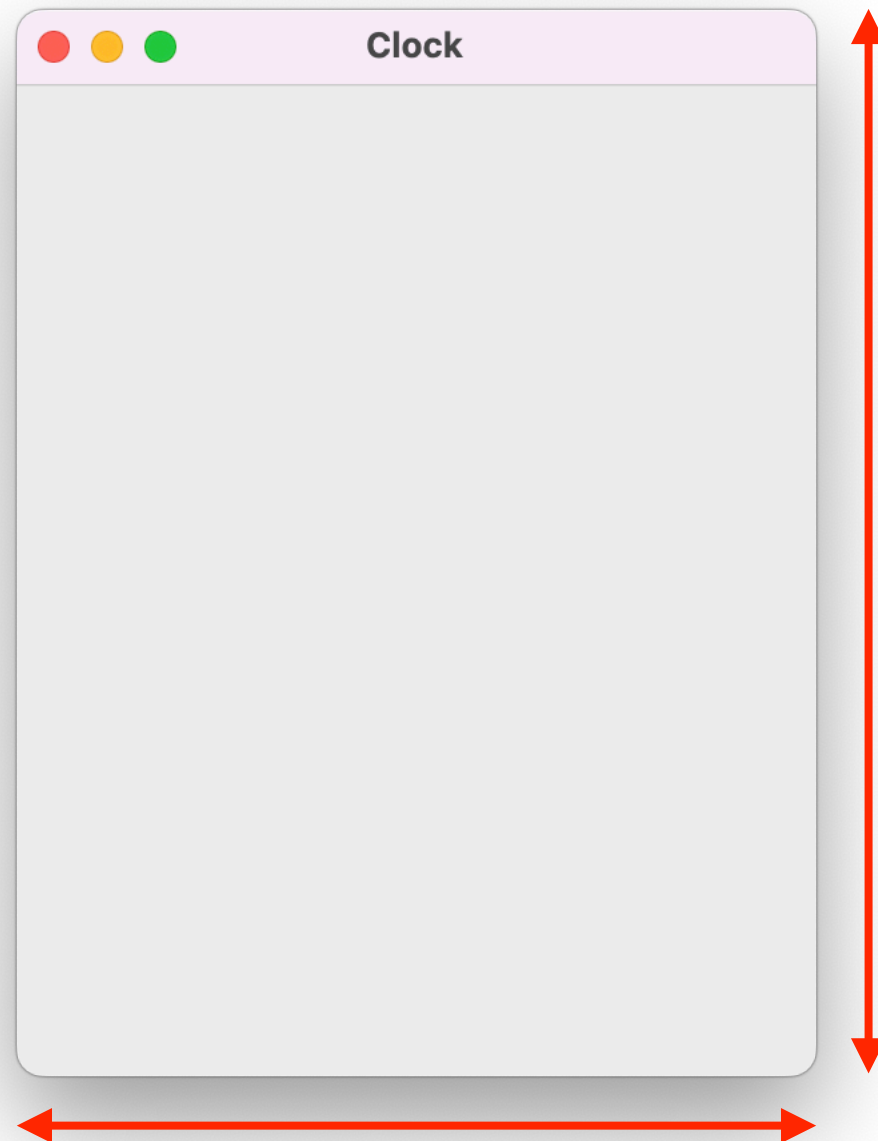
아날로그 시계 만들기

Graphical Output

Java Swing package

`javax.swing.*`

빈 프레임 만들기

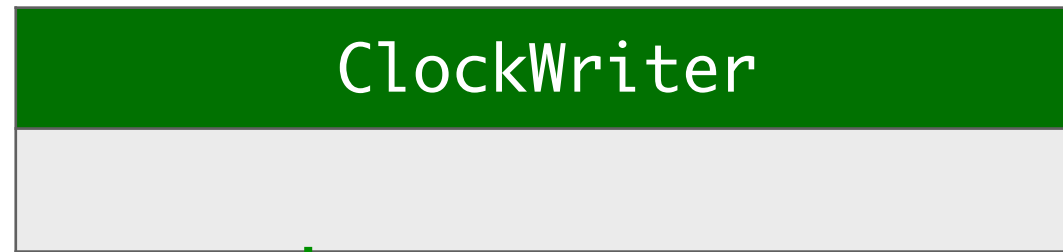


400

300

위치 및 크기의 단위
Pixels

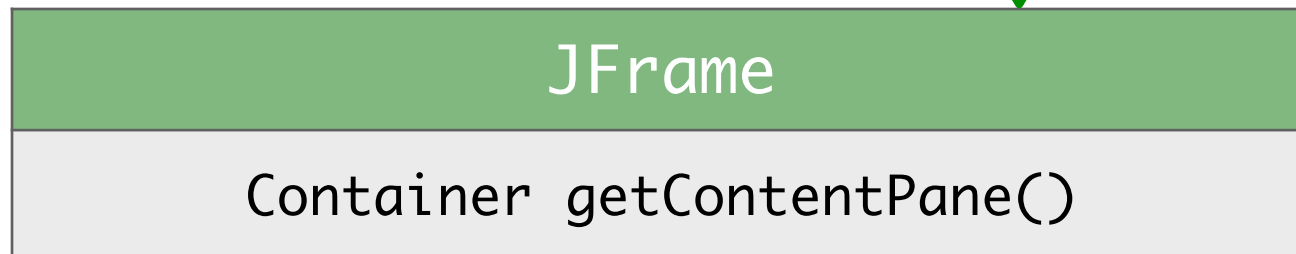
View



uses-a



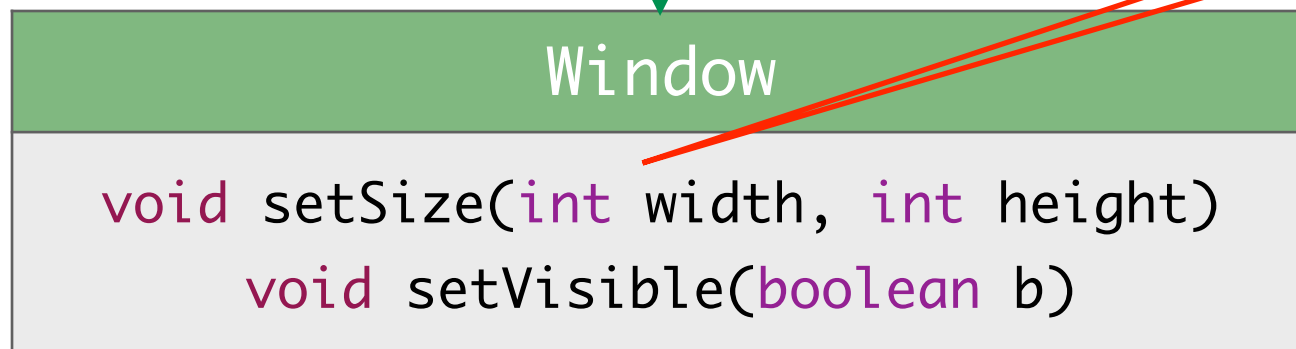
javax.swing



has-a



has-a



상속

Inheritance

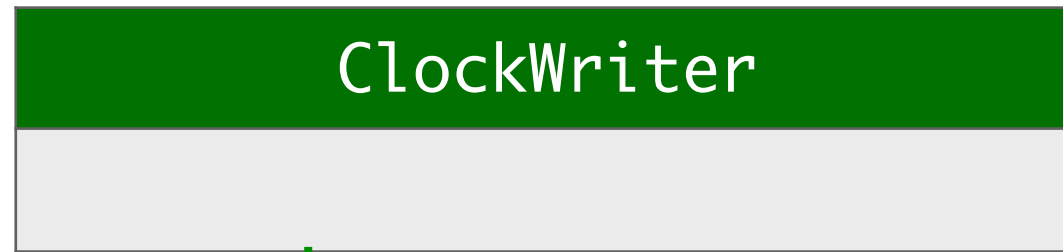
물려 받음

상속

Inheritance

물려 받음

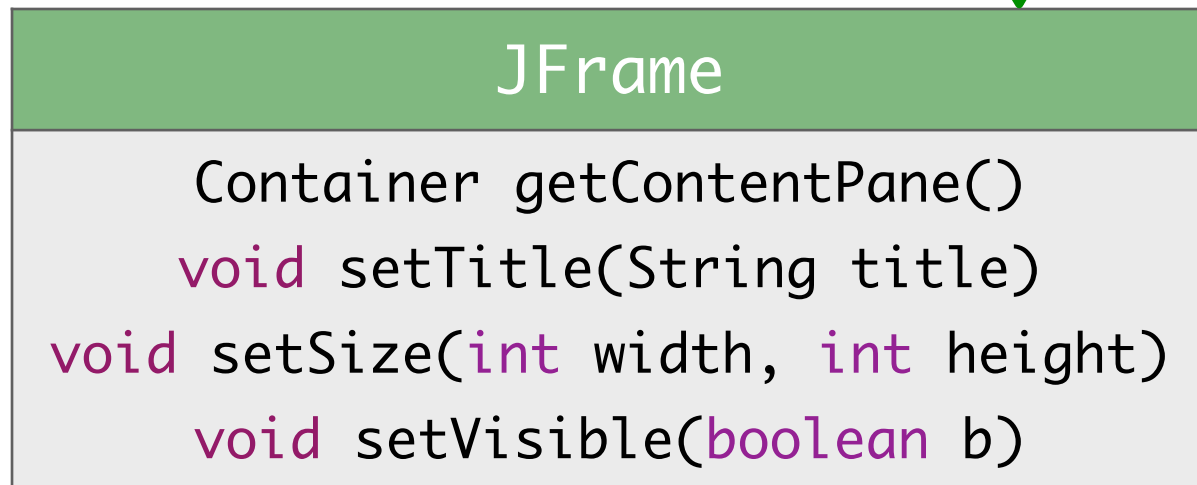
View



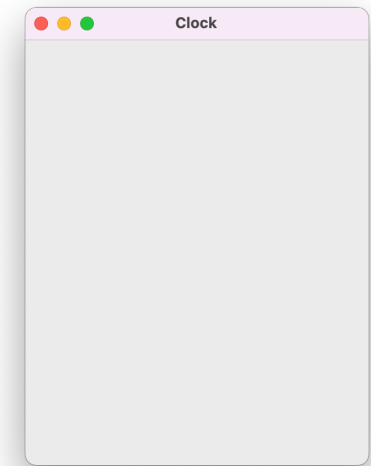
uses-a



javax.swing



빈 프레임 만들기



```
import javax.swing.*;

public class ClockWriter extends JPanel {

    public ClockWriter() {
        JFrame frame = new JFrame();
        frame.setTitle("Clock");
        frame.setSize(300, 400);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    }

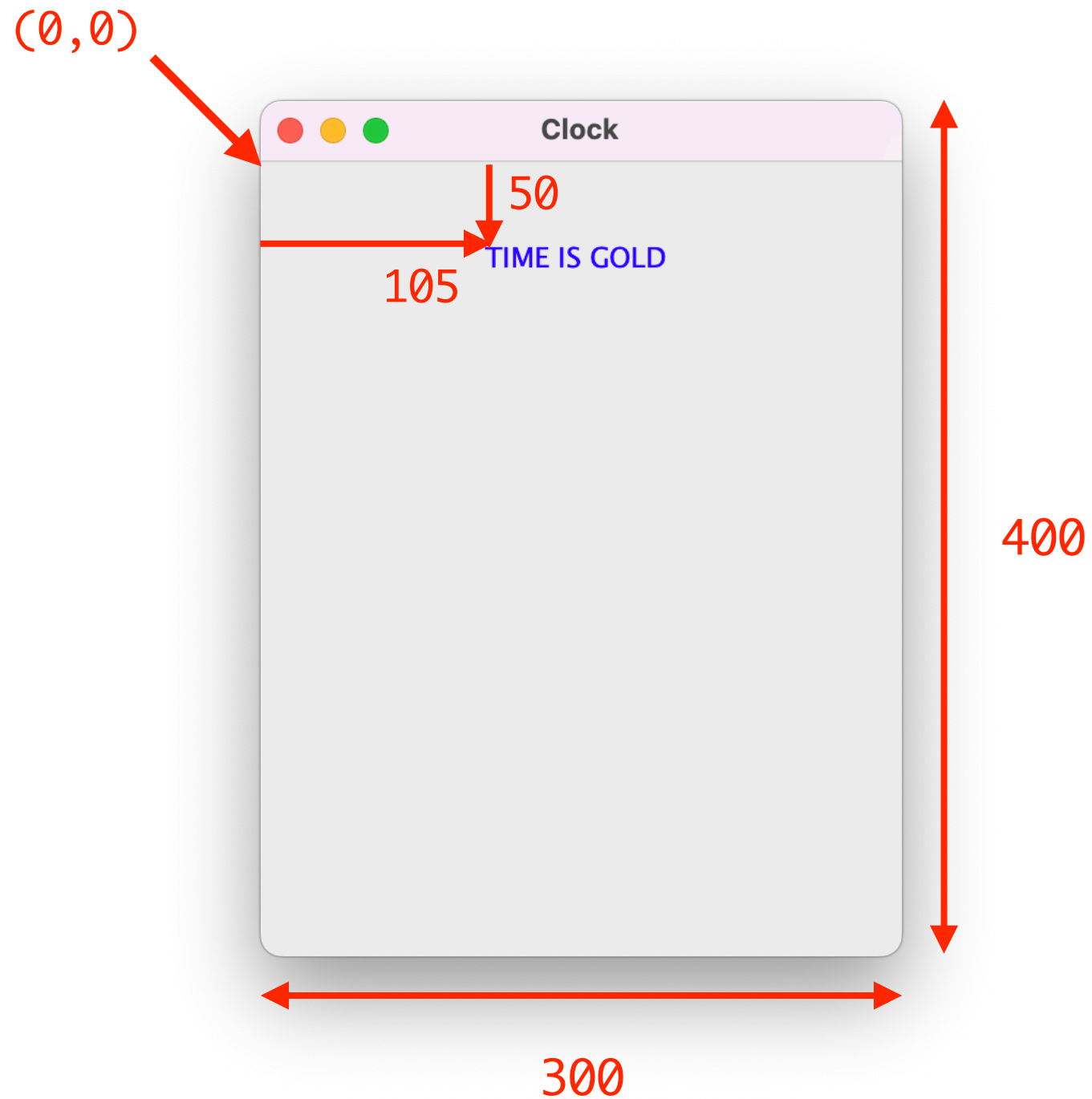
    // test code
    public static void main(String[] args) {
        new ClockWriter();
    }
}
```

창을 닫아도 프로그램이 살아있어요!

- 창을 만들때, `main` 메소드와는 별개의 쓰레드thread를 만들어 동시에 실행한다.
- `main` 메소드 실행을 종료해도, 창을 만든 쓰레드는 종료하지 않는다.
- 다음과 같이 프로그램에서 설정해두면, 창을 닫으면서 창을 만든 쓰레드도 함께 종료한다.

```
frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
```

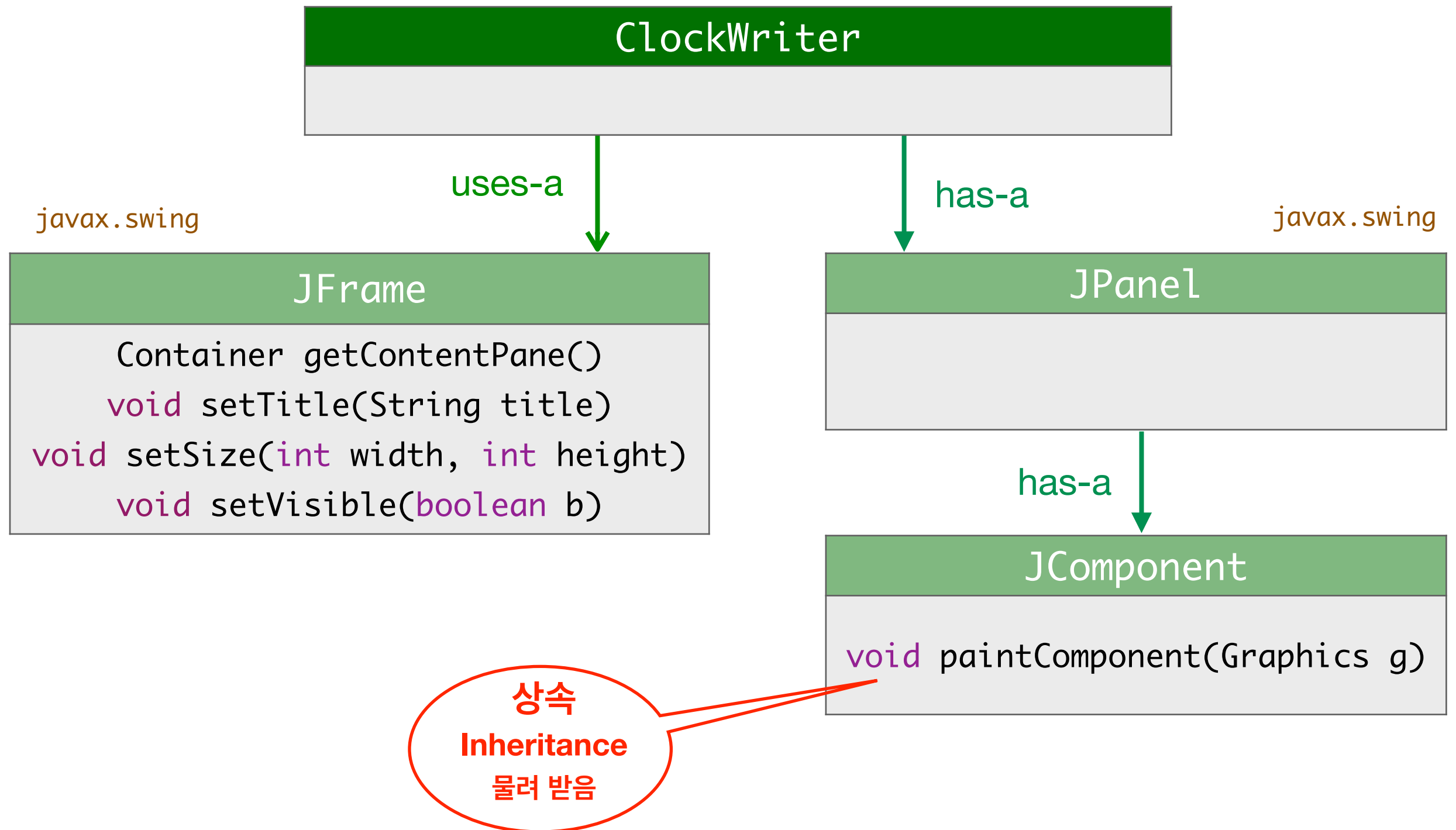
윈도우에 글쓰기



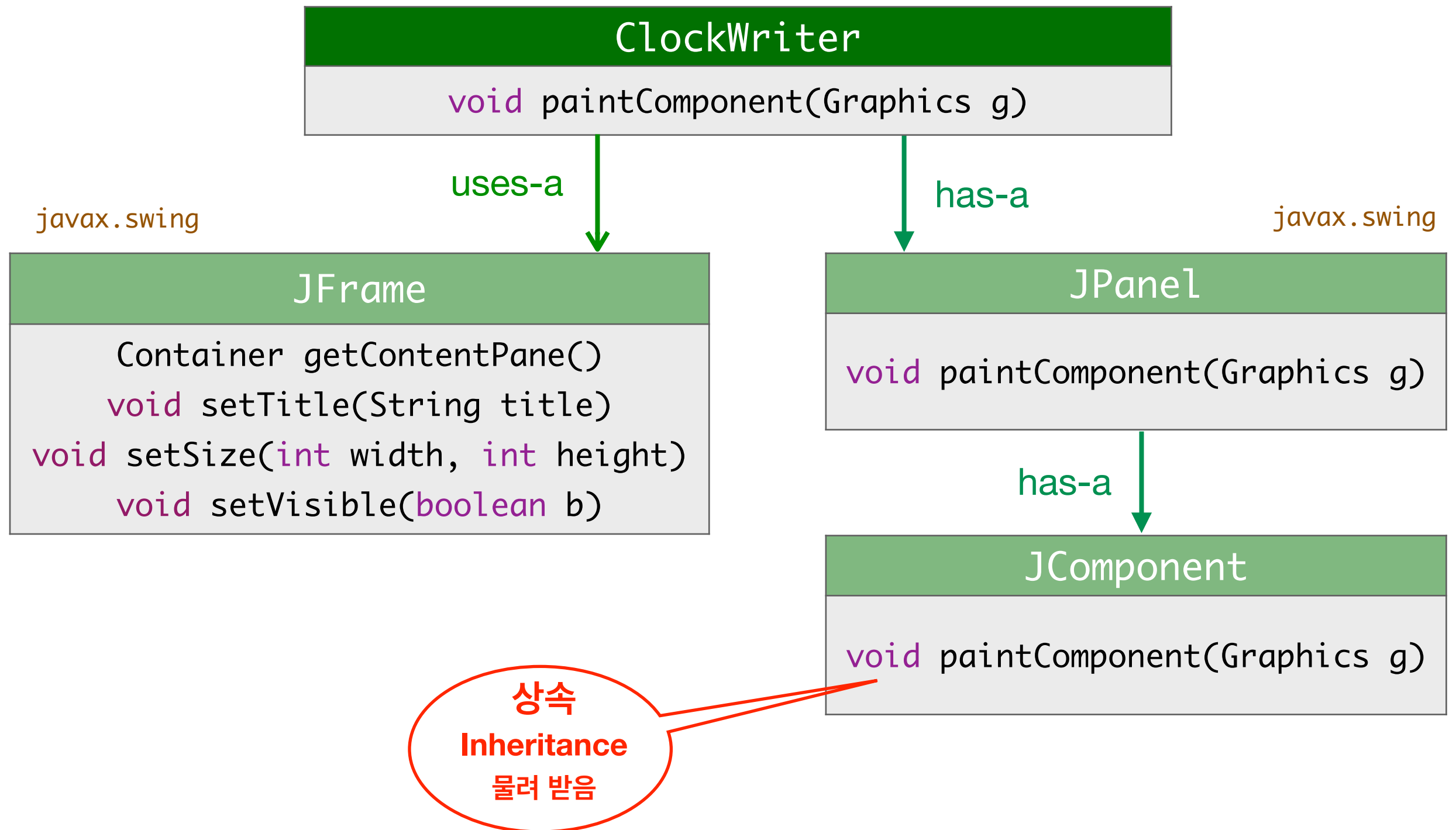
```
public void paintComponent(Graphics g) {  
    g.setColor(Color.BLUE);  
    g.drawString("TIME IS GOLD", 105, 50);  
}
```

width height

View

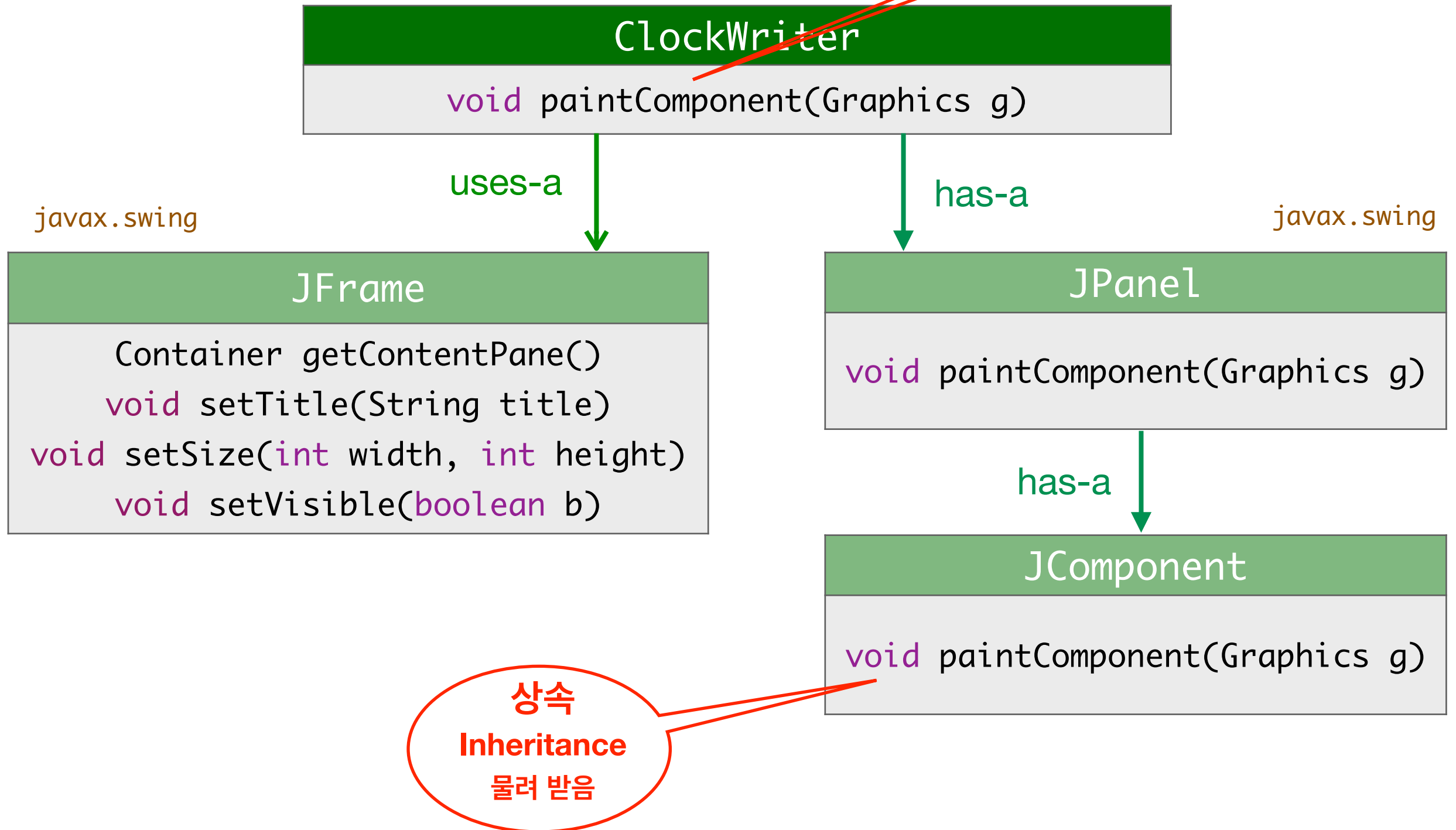


View



View

메소드 덮씌우기
method overriding



View

ClockWriter

`void paintComponent(Graphics g)`

uses-a

javax.swing

JFrame

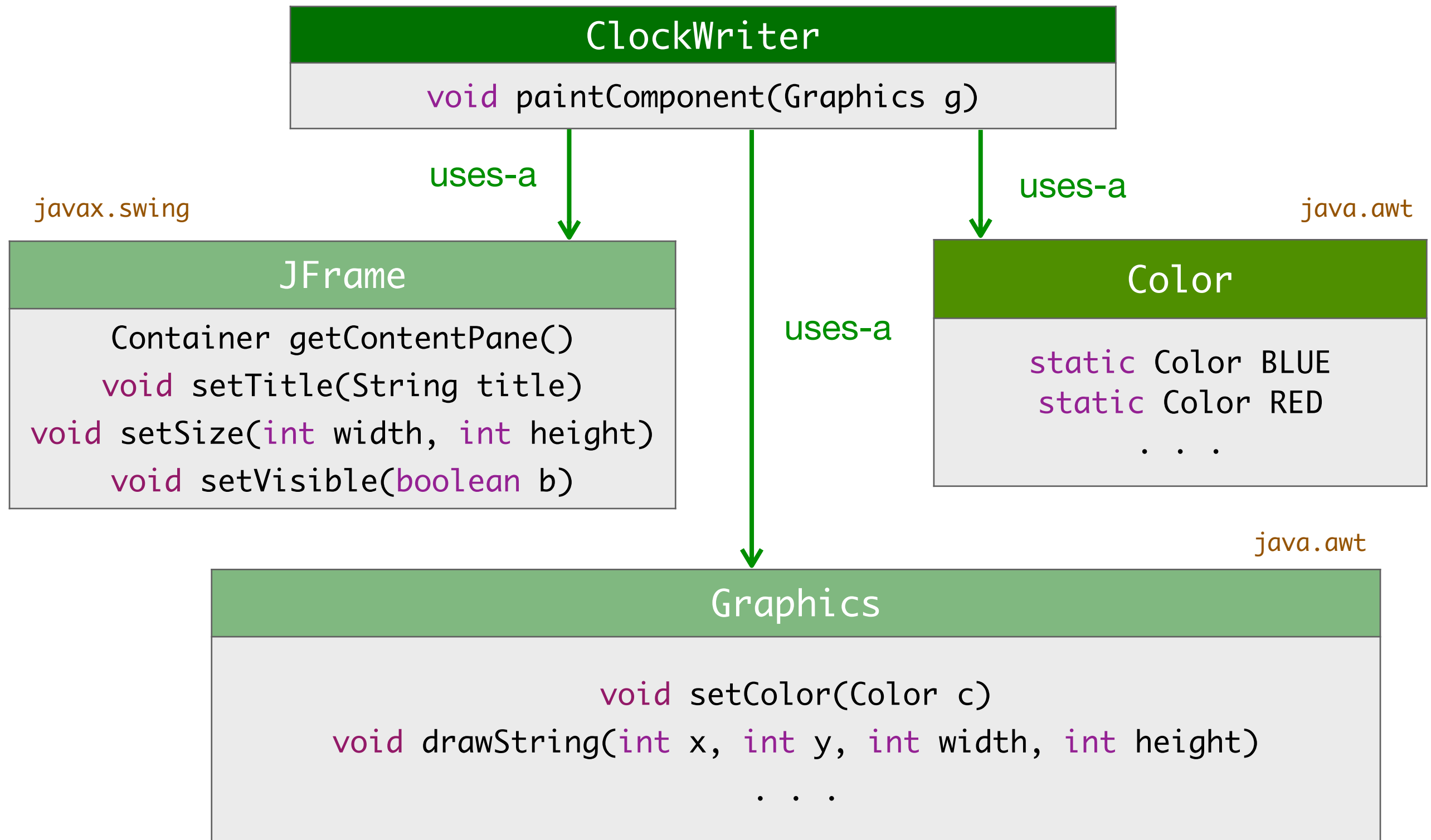
Container getContentPane()

`void setTitle(String title)`

`void setSize(int width, int height)`

`void setVisible(boolean b)`

View



창에 글쓰기

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

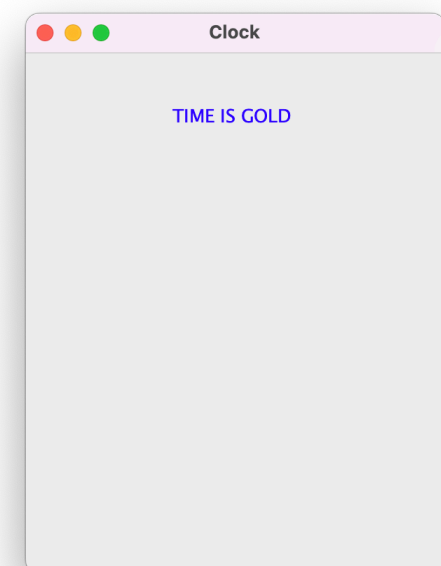
public class ClockWriter extends JPanel {

    public ClockWriter() {
        JFrame frame = new JFrame();
        frame.setTitle("Clock");
        frame.setSize(300, 400);
        frame.getContentPane().add(this);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    }

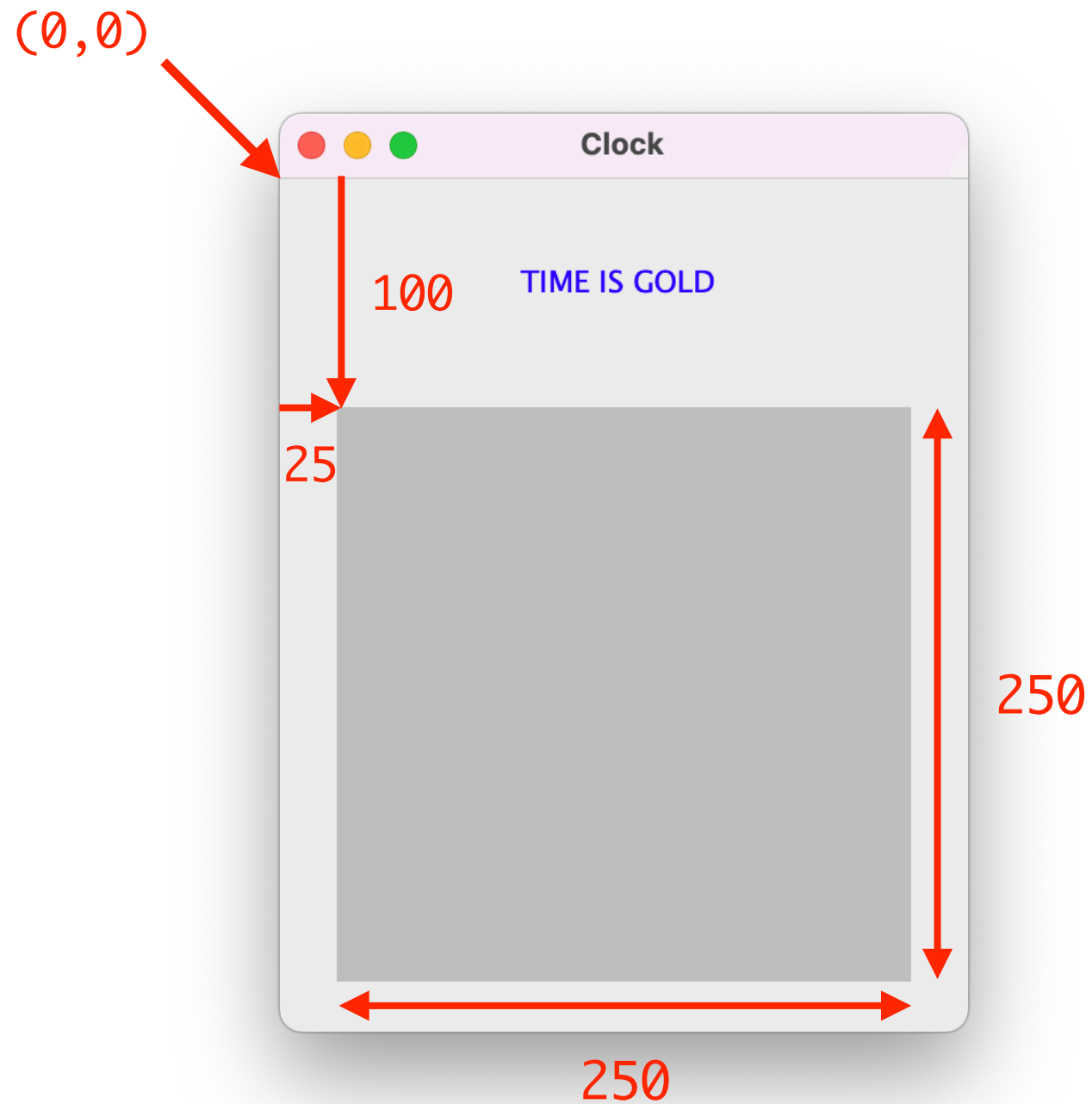
    public void paintComponent(Graphics g) {
        g.setColor(Color.BLUE);
        g.drawString("TIME IS GOLD", 105, 50);
    }

    // test code
    public static void main(String[] args) {
        new ClockWriter();
    }
}
```

객체 자신을 가리키는 이름



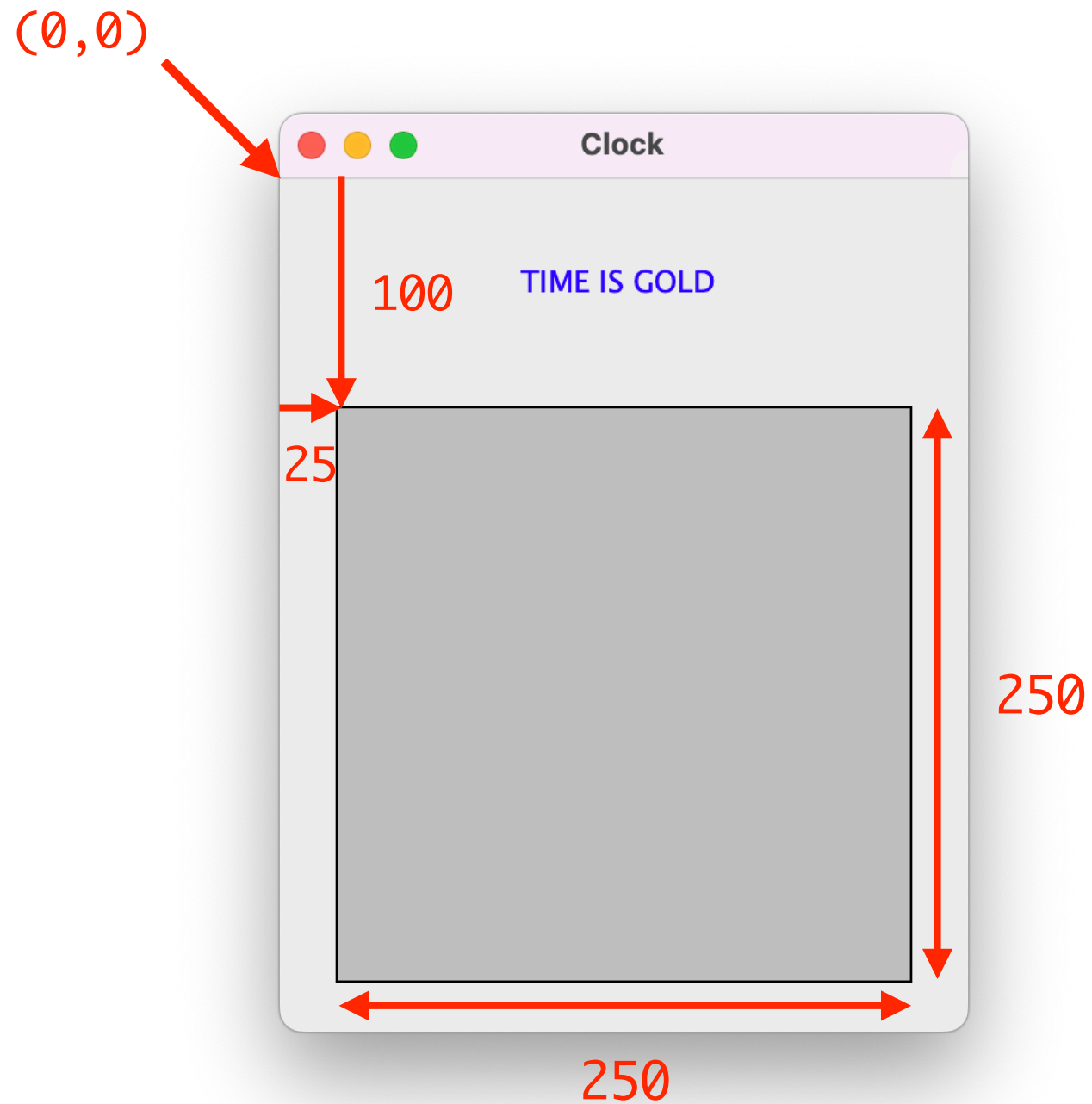
도형 그리기



```
public void paintComponent(Graphics g) {  
    g.setColor(Color.LIGHT_GRAY);  
    g.fillRect(25, 100, 250, 250);  
}
```

x y width height

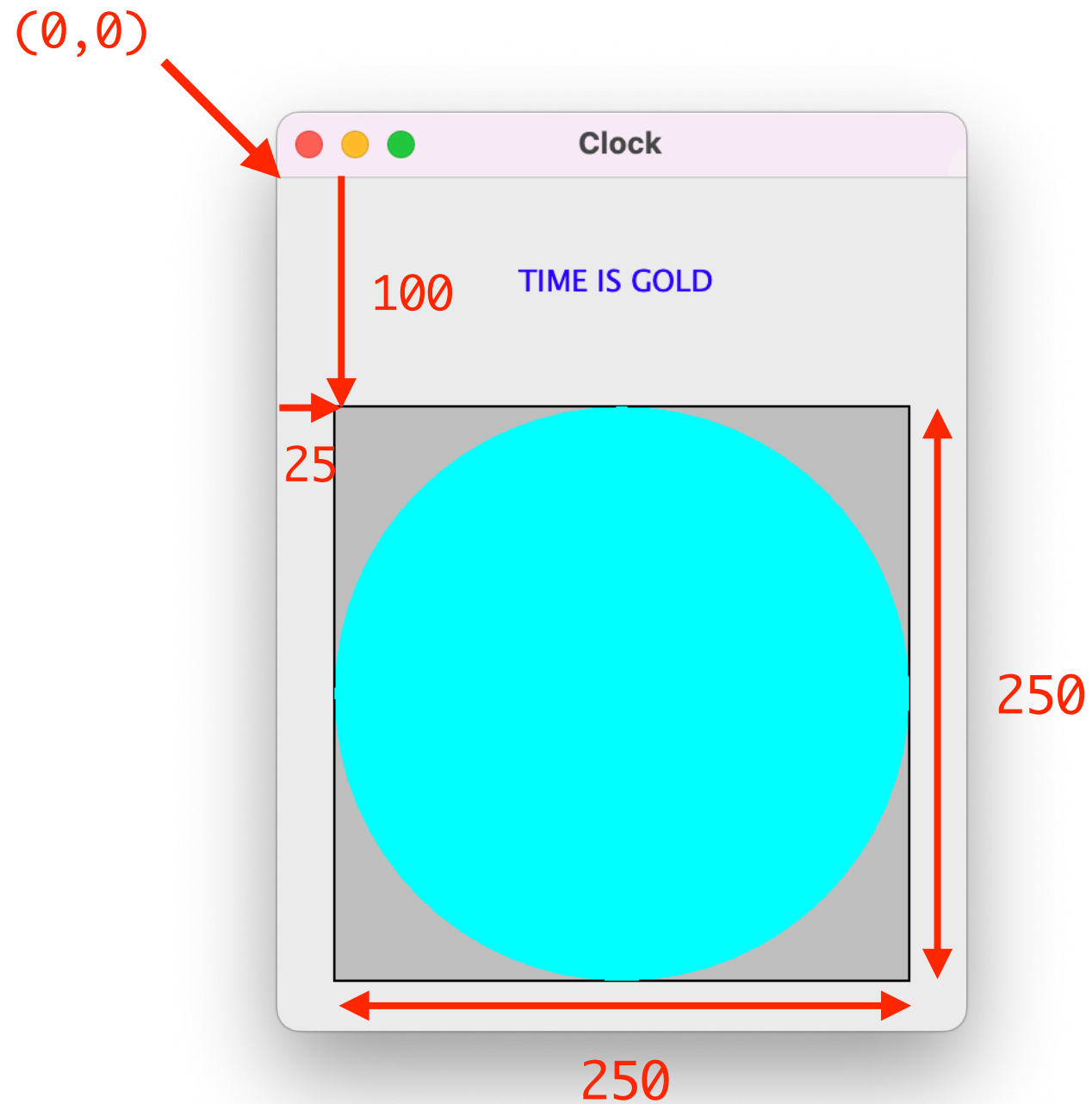
창에 도형 그리기



```
public void paintComponent(Graphics g) {  
    g.setColor(Color.BLACK);  
    g.drawRect(25, 100, 250, 250);  
}
```

x y width height

도형 그리기



```
public void paintComponent(Graphics g) {  
    g.setColor(Color.CYAN);  
    g.fillRect(25, 100, 250, 250);  
}
```

x y width height

도형 그리기

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

public class ClockWriter extends JPanel {

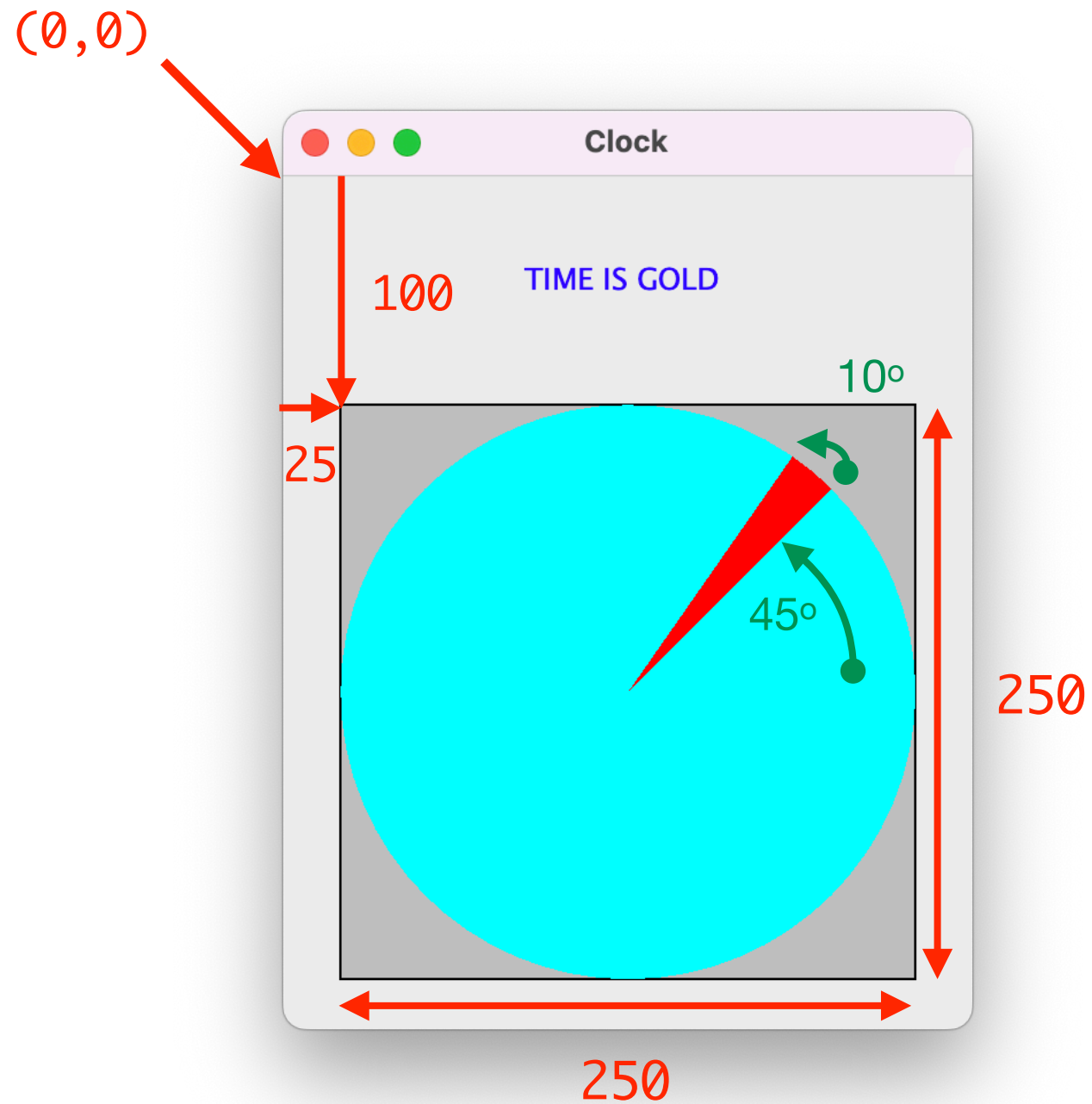
    public ClockWriter() {
        JFrame frame = new JFrame();
        frame.setTitle("Clock");
        frame.setSize(300, 400);
        frame.getContentPane().add(this);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    }

    public void paintComponent(Graphics g) {
        g.setColor(Color.BLUE);
        g.drawString("TIME IS GOLD", 105, 50);
        g.setColor(Color.LIGHT_GRAY);
        g.fillRect(25, 100, 250, 250);
        g.setColor(Color.BLACK);
        g.drawRect(25, 100, 250, 250);
        g.setColor(Color.CYAN);
        g.fillOval(25, 100, 250, 250);
    }

    // test code
    public static void main(String[] args) {
        new ClockWriter();
    }
}
```

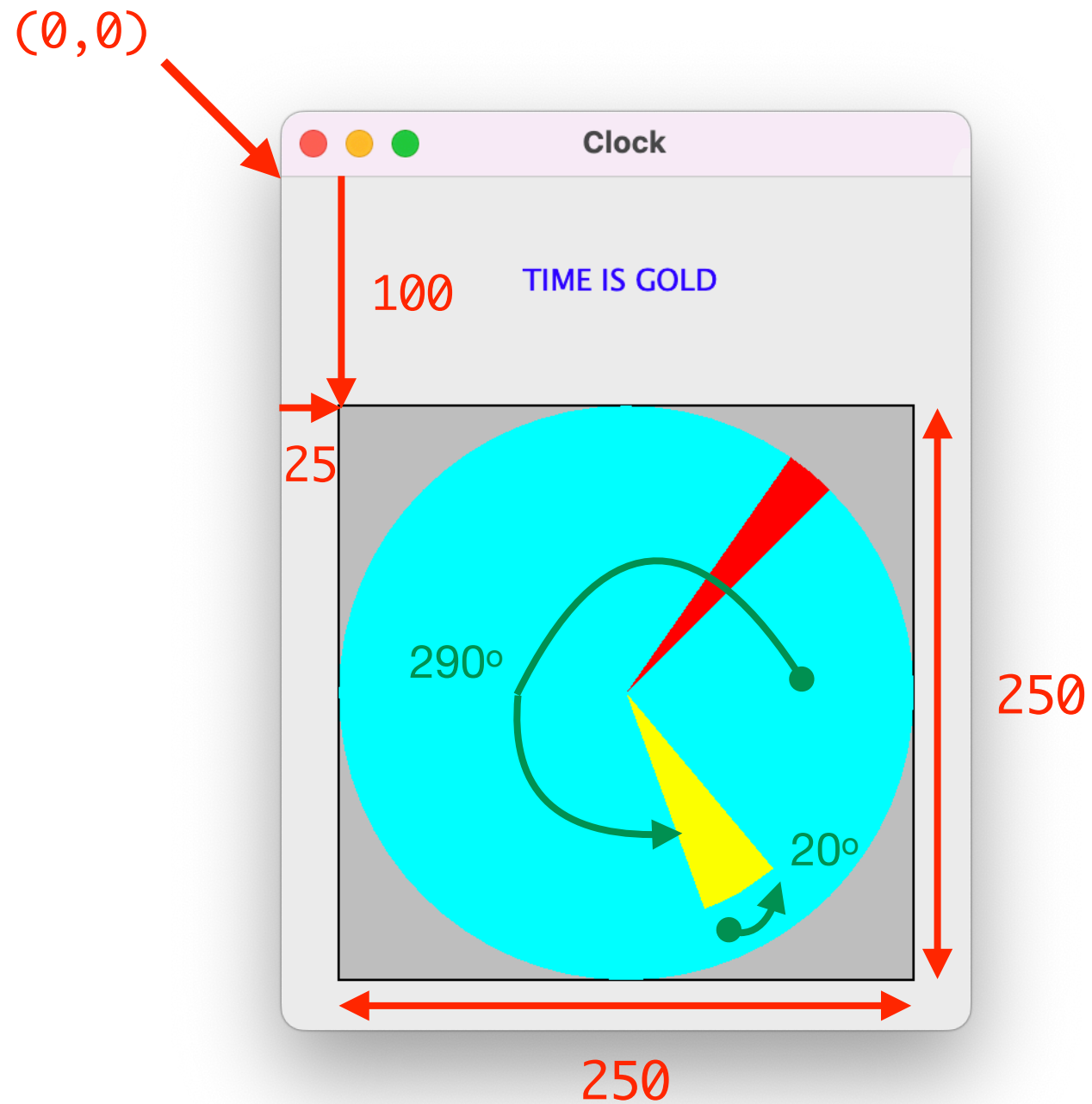


아크 그리기



```
public void paintComponent(Graphics g) {
    g.setColor(Color.RED);
    g.fillArc(25, 100, 250, 250, 45, 10);
}
```

아크 그리기



```
public void paintComponent(Graphics g) {  
    g.setColor(Color.YELLOW);  
    g.fillArc(50, 125, 200, 200, 290, 20);  
}
```

arcAngle

x y width height startAngle

사례 학습 : 아날로그 시계 그리기



짧은 아크를 시침, 긴 아크를 분침으로 하고,
현재 시간을 표시하는 아날로그 시계를 그리는 ClockWriter를 만들자.

시침, 분침, 각도 계산

```
int minutes_angle = 90 - now.getMinute() * 6;
```

```
int hours_angle = 90 - now.getHour() * 30;
```

```
import javax.swing.*;
import java.awt.*;
import java.time.*;

public class ClockWriter extends JPanel {

    public ClockWriter() {
        JFrame frame = new JFrame();
        frame.setTitle("Clock");
        frame.setSize(300, 400);
        frame.getContentPane().add(this);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    }

    public void paintComponent(Graphics g) {
        g.setColor(Color.BLUE);
        g.drawString("TIME IS GOLD", 105, 50);
        g.setColor(Color.LIGHT_GRAY);
        g.fillOval(25, 100, 250, 250);
        LocalDateTime now = LocalDateTime.now();
        int minutes_angle = 90 - now.getMinute() * 6;
        g.setColor(Color.RED);
        g.fillArc(25, 100, 250, 250, minutes_angle, 10);
        int hours_angle = 90 - now.getHour() * 30;
        g.setColor(Color.YELLOW);
        g.fillArc(50, 125, 200, 200, hours_angle, -20);
    }

    // test code
    public static void main(String[] args) {
        new ClockWriter();
    }
}
```

추가 요구사항

시계의 크기를 사용자가 선택하여 설정하게 하자.

필드 변수

Field Variables

= 상태 변수

객체

객체 Object	메모리에 존재하는 실체	메모리 주소로 식별
필드 Field	객체의 상태를 나타내는 정보	필드 변수에 저장하여 내부 공유 및 수정 (객체를 생성할 때 초기값을 정함*)
메소드 Method	객체가 수행 가능한 기능	메시지 호출을 받으면 메소드 실행

* 변수의 타입에 따라, 0, false, null로 초기값이 자동으로 매겨짐

필드 변수 선언

클래스 내부 전용

```
import javax.swing.*;  
import java.awt.*;  
import java.time.*;
```

```
public class ClockWriter extends JPanel {
```

```
    private final int SIZE;
```

필드 변수 선언
클래스 내부에서만
공유 및 수정 가능

```
    public ClockWriter(int size) {  
        SIZE = size;  
        JFrame frame = new JFrame();  
        frame.setTitle("Clock");  
        frame.setSize(SIZE+50, SIZE+150);  
        frame.getContentPane().add(this);  
        frame.setVisible(true);  
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);  
    }
```

```
    public void paintComponent(Graphics g) {  
        g.setColor(Color.BLUE);  
        g.drawString("TIME IS GOLD", 105, 50);  
        g.setColor(Color.LIGHT_GRAY);  
        g.fillOval(25, 100, SIZE, SIZE);  
        LocalDateTime now = LocalDateTime.now();  
        int minutes_angle = 90 - now.getMinute() * 6;  
        int minute_hand = SIZE;  
        g.setColor(Color.RED);  
        g.fillArc(25, 100, minute_hand, minute_hand, minutes_angle, 10);  
        int hours_angle = 90 - now.getHour() * 30;  
        int hour_hand = SIZE - 50;  
        g.setColor(Color.YELLOW);  
        g.fillArc(50, 125, hour_hand, hour_hand, hours_angle, -20);  
    }
```

```
    // test code  
    public static void main(String[] args) {  
        new ClockWriter(250);  
    }
```

```
}
```

final 변수

```
import javax.swing.*;
import java.awt.*;
import java.time.*;

public class ClockWriter extends JPanel {

    private final int SIZE;

    public ClockWriter(int size) {
        SIZE = size;
        JFrame frame = new JFrame();
        frame.setTitle("Clock");
        frame.setSize(SIZE+50, SIZE+150);
        frame.getContentPane().add(this);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    }

    public void paintComponent(Graphics g) {
        g.setColor(Color.BLUE);
        g.drawString("TIME IS GOLD", 105, 50);
        g.setColor(Color.LIGHT_GRAY);
        g.fillOval(25, 100, SIZE, SIZE);
        LocalDateTime now = LocalDateTime.now();
        int minutes_angle = 90 - now.getMinute() * 6;
        int minute_hand = SIZE;
        g.setColor(Color.RED);
        g.fillArc(25, 100, minute_hand, minute_hand, minutes_angle, 10);
        int hours_angle = 90 - now.getHour() * 30;
        int hour_hand = SIZE - 50;
        g.setColor(Color.YELLOW);
        g.fillArc(50, 125, hour_hand, hour_hand, hours_angle, -20);
    }

    // test code
    public static void main(String[] args) {
        new ClockWriter(250);
    }
}
```

final 변수

- 값 지정 후 변경 불가
- 이름은 모두 대문자로 씀

필드 변수 값 설정

```
import javax.swing.*;
import java.awt.*;
import java.time.*;

public class ClockWriter extends JPanel {

    private final int SIZE;

    public ClockWriter(int size) {
        SIZE = size;
        JFrame frame = new JFrame();
        frame.setTitle("Clock");
        frame.setSize(SIZE+50, SIZE+150);
        frame.getContentPane().add(this);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    }

    public void paintComponent(Graphics g) {
        g.setColor(Color.BLUE);
        g.drawString("TIME IS GOLD", 105, 50);
        g.setColor(Color.LIGHT_GRAY);
        g.fillOval(25, 100, SIZE, SIZE);
        LocalDateTime now = LocalDateTime.now();
        int minutes_angle = 90 - now.getMinute() * 6;
        int minute_hand = SIZE;
        g.setColor(Color.RED);
        g.fillArc(25, 100, minute_hand, minute_hand, minutes_angle, 10);
        int hours_angle = 90 - now.getHour() * 30;
        int hour_hand = SIZE - 50;
        g.setColor(Color.YELLOW);
        g.fillArc(50, 125, hour_hand, hour_hand, hours_angle, -20);
    }

    // test code
    public static void main(String[] args) {
        new ClockWriter(250);
    }
}
```



필드 변수의 값
객체 생성시
인수로 제공한 값을
전달받아
필드 변수값 설정

지역 변수 vs 필드 변수

	지역 변수	필드 변수
탄생	선언시	객체 생성시
소멸	메소드 (블록) 실행 종료시	객체 소멸시
개수	메소드 호출 횟수 만큼	객체 생성 개수 만큼
초기화	수동	자동
유효 범위 scope	선언 이후부터 소속 블록 끝까지	객체 내부 전체 + 한정자(public 등)에 따라 객체 외부 접근 가능

변수의 유효범위 scope

필드변수
field variable

지역 변수
local variable

```
import javax.swing.*;
import java.awt.*;
import java.time.*;

public class ClockWriter extends JPanel {

    private final int SIZE;

    public ClockWriter(int size) {
        SIZE = size;
        JFrame frame = new JFrame();
        frame.setTitle("Clock");
        frame.setSize(SIZE+50, SIZE+150);
        frame.getContentPane().add(this);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    }

    public void paintComponent(Graphics g) {
        g.setColor(Color.BLUE);
        g.drawString("TIME IS GOLD", 100, 50);
        g.setColor(Color.LIGHT_GRAY);
        g.fillOval(25, 100, SIZE, SIZE);
        LocalDateTime now = LocalDateTime.now();
        int minutes_angle = 90 - now.getMinute() * 6;
        int minute_hand = SIZE;
        g.setColor(Color.RED);
        g.fillArc(25, 100, minute_hand, minute_hand, minutes_angle, 10);
        int hours_angle = 90 - now.getHour() * 30;
        int hour_hand = SIZE - 50;
        g.setColor(Color.YELLOW);
        g.fillArc(50, 125, hour_hand, hour_hand, hours_angle, -20);
    }

    // test code
    public static void main(String[] args) {
        new ClockWriter(250);
    }
}
```


변수의 유효 범위

Scope

```
public class Scope {  
    public static void main(String[] args) {  
        {  
            int n = 2;  
            System.out.println(n);  
        }  
        double n = 3.14;  
        System.out.println(n);  
    }  
}
```

같은 이름
중복 선언
오류

```
public class Scope {  
    public static void main(String[] args) {  
        int n = 2;  
        System.out.println(n);  
        {  
            double n = 3.14;  
            System.out.println(n);  
        }  
    }  
}
```



변수의 유효 범위

Scope

```
public class Scope {  
    private int n = 3;  
  
    public Scope() {  
        System.out.println(n);  
        int n = 333;  
        System.out.println(n);  
    }  
  
    public static void main(String[] args) {  
        new Scope();  
    }  
}
```

변수의 유효 범위

Scope

```
import java.awt.*;

public class Scope {

    private double d = 3.14;

    public Scope() {
        System.out.println(s);
        System.out.println(d);
        int d = 2;
        System.out.println(d);
        s = d + s;
        System.out.println(s);
    }

    private String s = "X" + d;

    public void printComponent(Graphics g) {
        System.out.println(d + " " + s);
    }

    public static void main(String[] args) {
        new Scope();
    }
}
```

실습



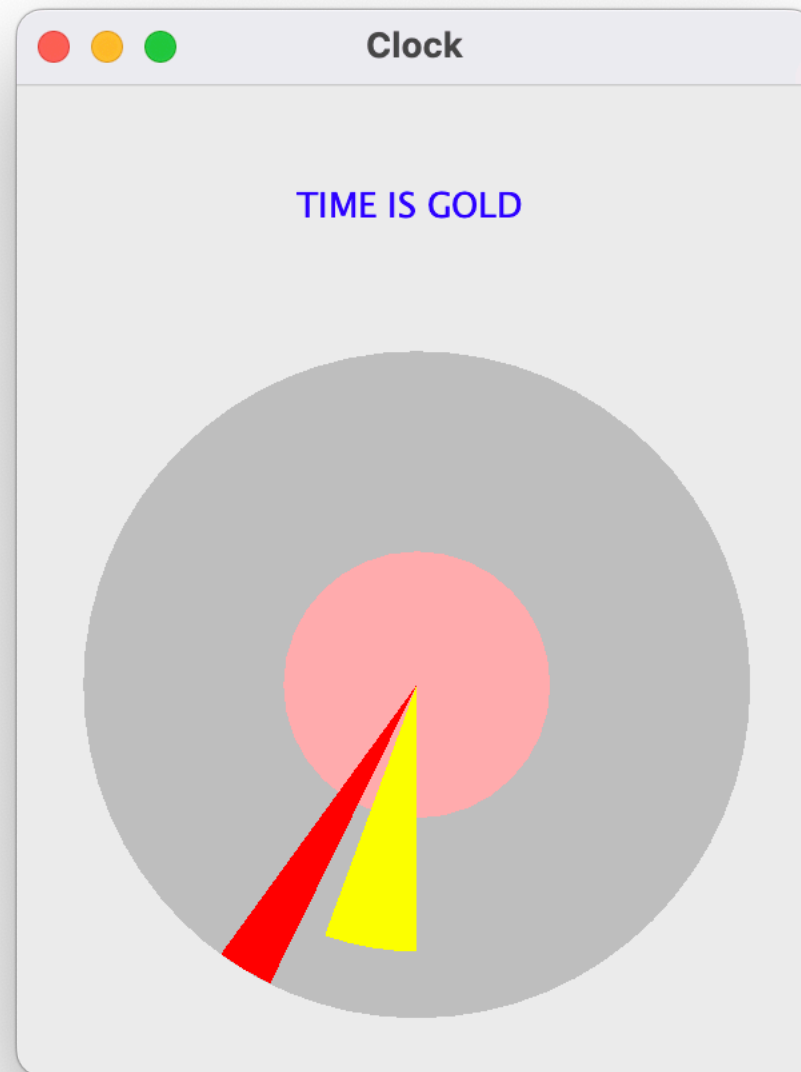
**창을 최소화 했다가 다시 띄울때마다 핑크색 원이 일정 비율로 커짐
전체를 채운 후에는 다시 처음으로 돌아가 같은 패턴을 반복**

실습



**창을 최소화 했다가 다시 띄울때마다 핑크색 원이 일정 비율로 커짐
전체를 채운 후에는 다시 처음으로 돌아가 같은 패턴을 반복**

실습



**창을 최소화 했다가 다시 띄울때마다 핑크색 원이 일정 비율로 커짐
전체를 채운 후에는 다시 처음으로 돌아가 같은 패턴을 반복**

실습



**창을 최소화 했다가 다시 띄울때마다 핑크색 원이 일정 비율로 커짐
전체를 채운 후에는 다시 처음으로 돌아가 같은 패턴을 반복**

실습



**창을 최소화 했다가 다시 띄울때마다 핑크색 원이 일정 비율로 커짐
전체를 채운 후에는 다시 처음으로 돌아가 같은 패턴을 반복**

실습

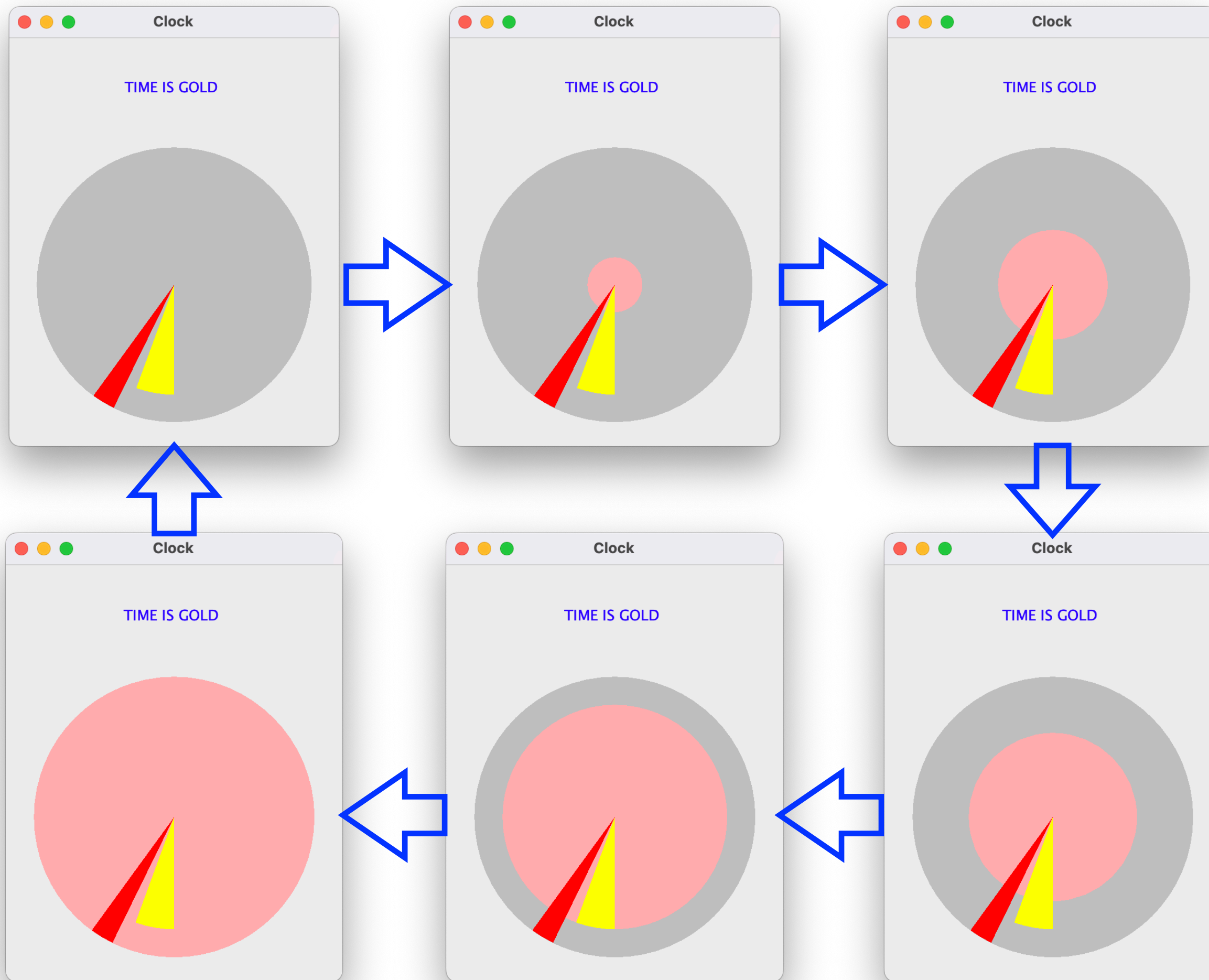


**창을 최소화 했다가 다시 띄울때마다 핑크색 원이 일정 비율로 커짐
전체를 채운 후에는 다시 처음으로 돌아가 같은 패턴을 반복**

실습



**창을 최소화 했다가 다시 띄울때마다 핑크색 원이 일정 비율로 커짐
전체를 채운 후에는 다시 처음으로 돌아가 같은 패턴을 반복**



```
import javax.swing.*;
import java.awt.*;
import java.time.*;

public class ClockWriter extends JPanel {

    private final int SIZE;
    private final int MARGIN;
    private int diameter;

    public ClockWriter(int size, int rate) {
        SIZE = size;
        MARGIN = size / rate;
        JFrame frame = new JFrame();
        frame.setTitle("Clock");
        frame.setSize(SIZE+50, SIZE+150);
        frame.getContentPane().add(this);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    }

    public void paintComponent(Graphics g) {
        g.setColor(Color.BLUE);
        g.drawString("TIME IS GOLD", 105, 50);
        g.setColor(Color.LIGHT_GRAY);
        g.fillOval(25, 100, SIZE, SIZE);
        g.setColor(Color.PINK);
        int base = (SIZE - diameter) / 2;
        g.fillOval(25+base, 100+base, diameter, diameter);
        diameter = (diameter >= SIZE) ? 0 : diameter + MARGIN;
        LocalTime now = LocalTime.now();
        int minutes_angle = 90 - now.getMinute() * 6;
        int minute hand = SIZE;
```

```
JFrame frame = new JFrame();
frame.setTitle("Clock");
frame.setSize(SIZE+50, SIZE+150);
frame.getContentPane().add(this);
frame.setVisible(true);
frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
}
```

```
public void paintComponent(Graphics g) {
    g.setColor(Color.BLUE);
    g.drawString("TIME IS GOLD", 105, 50);
    g.setColor(Color.LIGHT_GRAY);
    g.fillOval(25, 100, SIZE, SIZE);
    g.setColor(Color.PINK);
    int base = (SIZE - diameter) / 2;
    g.fillOval(25+base, 100+base, diameter, diameter);
    diameter = (diameter >= SIZE) ? 0 : diameter + MARGIN;
    LocalDateTime now = LocalDateTime.now();
    int minutes_angle = 90 - now.getMinute() * 6;
    int minute_hand = SIZE;
    g.setColor(Color.RED);
    g.fillArc(25, 100, minute_hand, minute_hand, minutes_angle, 10);
    int hours_angle = 90 - now.getHour() * 30;
    int hour_hand = SIZE - 50;
    g.setColor(Color.YELLOW);
    g.fillArc(50, 125, hour_hand, hour_hand, hours_angle, -20);
}
```

```
// test code
```

```
public static void main(String[] args) {
    new ClockWriter(250, 5);
}
```

```
}
```

숙제

아날로그 시계 (기능 추가)

1. 수업 시간에 공부한 아날로그 시계에 초침을 추가하자.

2. 시계에 눈금을 추가한다.

- 디자인은 자유
- 도형을 사용하여 눈금을 표시해도 좋고, 숫자 (1~12)를 넣어도 좋다.