

CSE2016

프로그램설계방법론

초기화 메소드와 필드 변수

Constructor Methods & Field Variables

도경구

한양대학교 ERICA 소프트웨어학부



클래스

class

선언

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

호출

```
new ClassName();
```

메소드

method

선언

```
public <type> methodName(<type> par_1, ..., <type> par_n) {  
    // code  
}
```

호출

```
methodName(arg_1, ..., arg_n);
```

메소드

method

공개 리턴 타입

parameter
0개 이상의 파라미터 나열

선언

```
public <type> methodName(<type> par_1, ..., <type> par_n) {  
    // code  
}
```

호출

methodName(arg_1, ..., arg_n);

개수 일치와
타입 부합

0개 이상의 인수 나열
argument

선언/호출 사례

선언

```
public double ctof(double c) {
    return (9.0 / 5.0) * c + 32;
}
```

```
public boolean checkAdding(int m, int n, int ans) {
    return m + n == ans;
}
```

```
public String createGreetings() {
    return "Hello, World!";
}
```

```
public void show(String message) {
    System.out.println(message);
}
```

호출

```
double f = ctof(24.0);
```

```
boolean pass =
checkAdding(39, 55, 84);
```

```
String g = createGreetings();
```

```
show("Welcome to Java!");
```

초기화 메소드

constructor method

객체 생성시
한번만 실행

선언

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

클래스 이름과 동일

호출

```
new ClassName(arg_1, ..., arg_n);
```

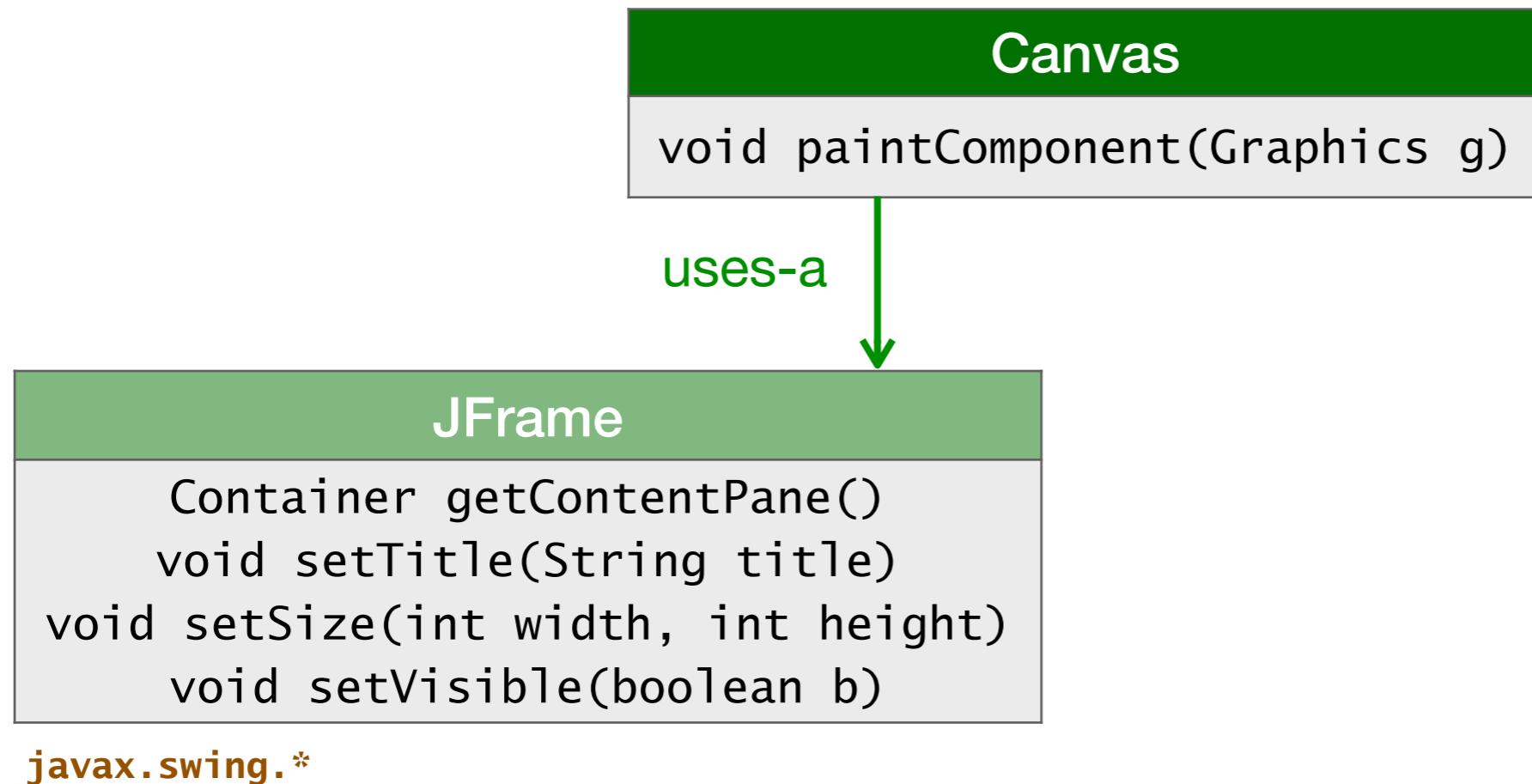
개수 일치와
타입 부합

Graphical Output

Java Swing package

`javax.swing.*`

View



`javax.swing.*`

빈 프레임 만들기

```
1 import javax.swing.*;
2
3 public class Canvas {
4
5     public Canvas() {
6         int width = 300;
7         int height = 200;
8         String title = "Canvas";
9         // 프레임 제작
10        JFrame frame = new JFrame();
11        frame.setTitle(title);
12        frame.setSize(width, height);
13        frame.setVisible(true);
14        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
15    }
16
17    // test code
18    public static void main(String[] args) {
19        new Canvas();
20    }
21
22 }
```

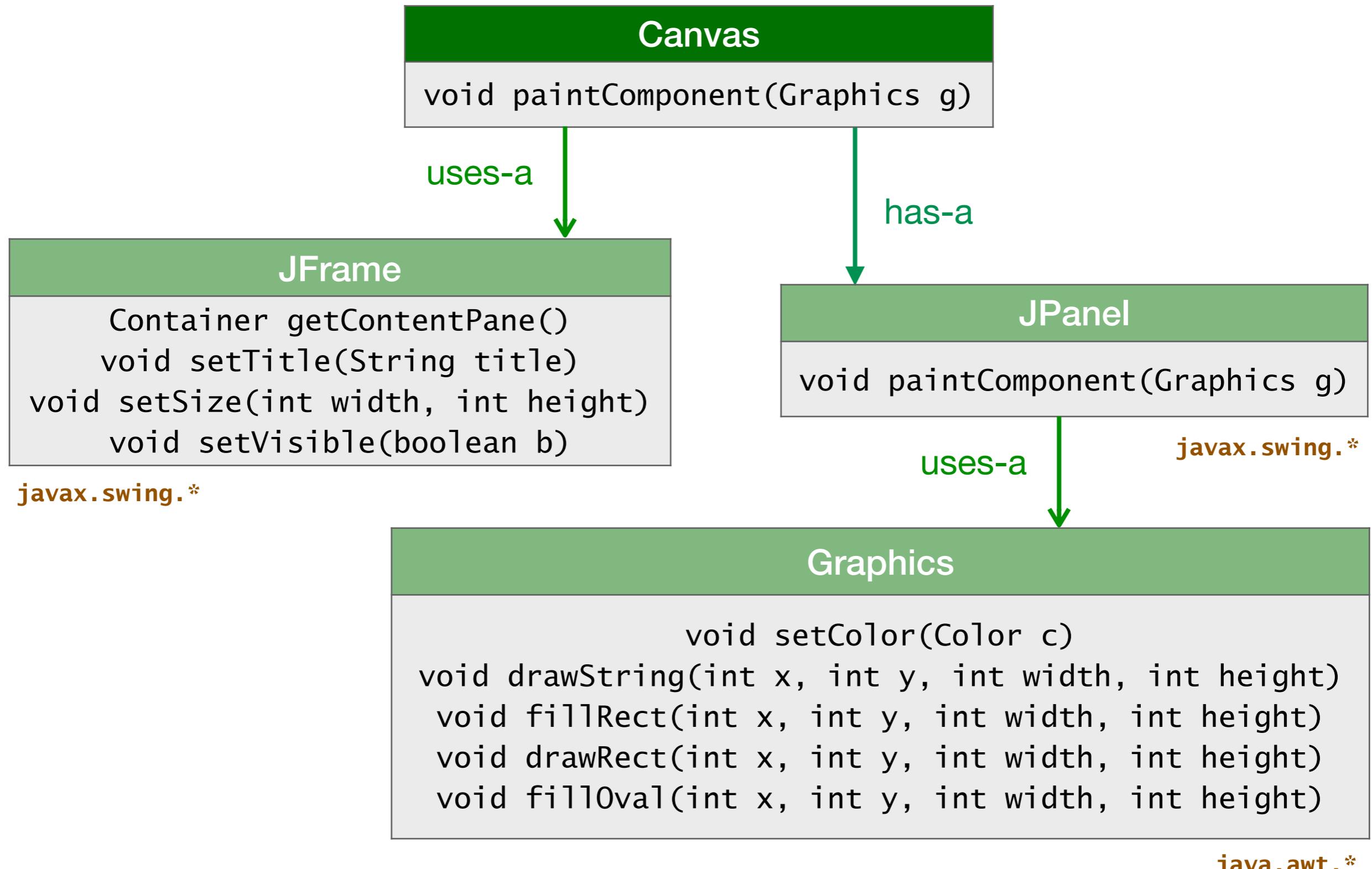


창을 닫아도 프로그램이 살아있어요!

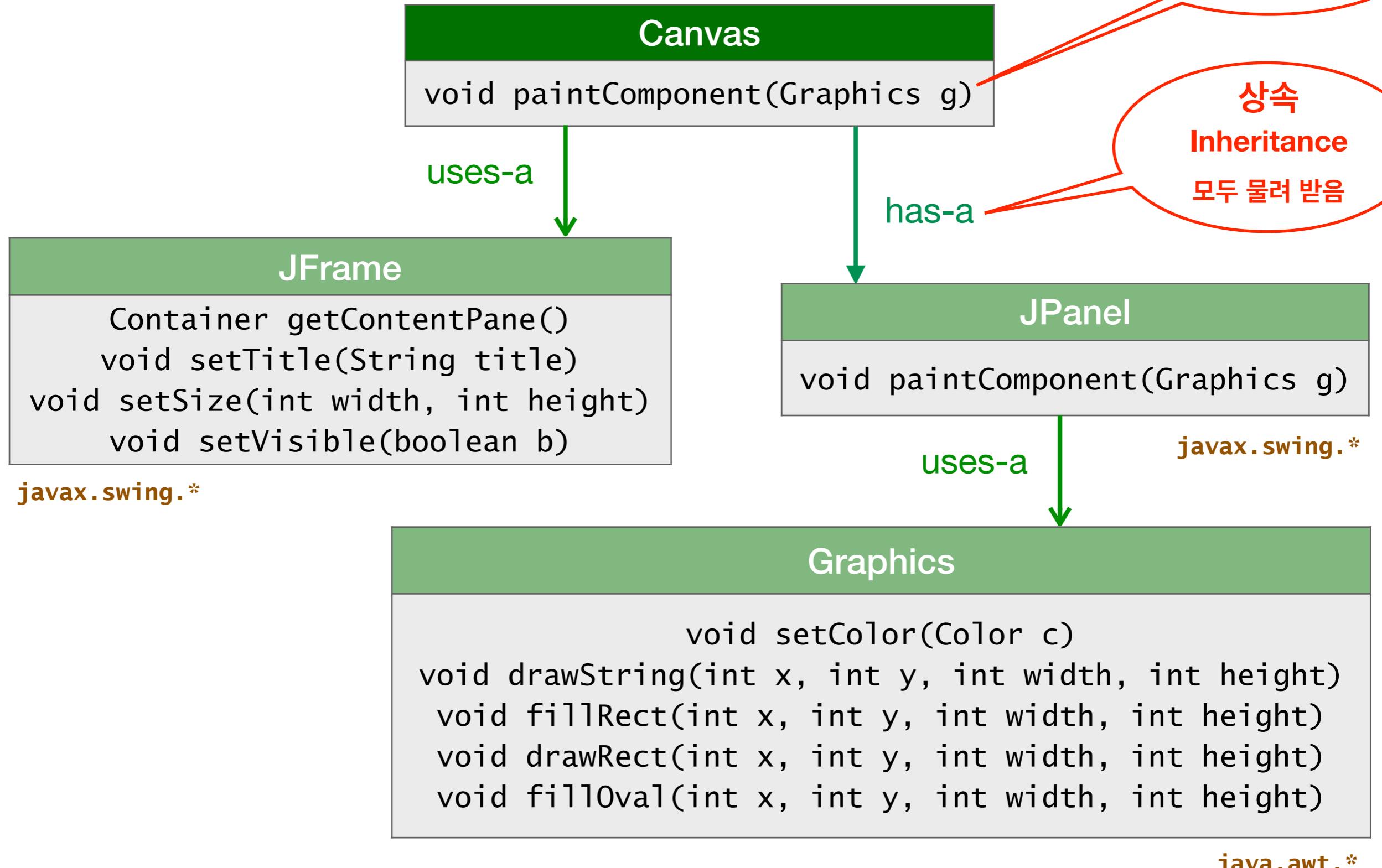
- 창을 만들때, main 메소드와는 별개의 쓰레드thread를 만들어 동시에 실행합니다.
- main 메소드 실행을 종료해도, 창을 만든 쓰레드는 종료하지 않습니다.
- 다음과 같이 프로그램에서 설정해두면, 창을 닫으면서 창을 만든 쓰레드도 함께 종료합니다.

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

View



View



창에 글쓰기

```

1① import javax.swing.*;
2 import java.awt.*;
3
4 public class Canvas extends JPanel {
5
6②     public Canvas() {
7         int width = 300;
8         int height = 200;
9         String title = "Canvas";
10
11        JFrame frame = new JFrame();
12        frame.setTitle(title);
13        frame.setSize(width, height);
14        frame.getContentPane().add(this);
15        frame.setVisible(true);
16        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
17    }
18
19③     public void paintComponent(Graphics g) {
20         g.setColor(Color.red);
21         g.drawString("Java!", 100, 60);
22     }
23
24     // test code
25④     public static void main(String[] args) {
26         new Canvas();
27     }
28
29 }

```



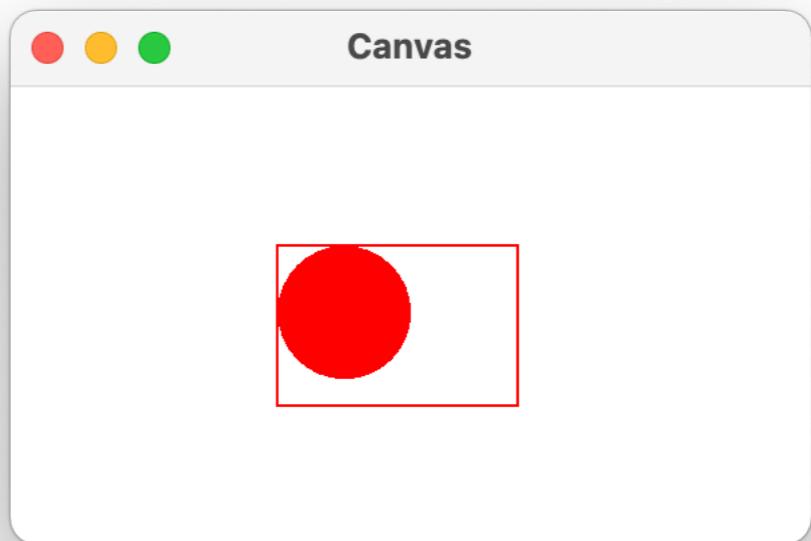
객체 자신을 가리키는 이름

창에 도형 그리기

```

1① import javax.swing.*;
2 import java.awt.*;
3
4 public class Canvas extends JPanel {
5
6②     public Canvas() {
7         int width = 300;
8         int height = 200;
9         String title = "Canvas";
10
11        JFrame frame = new JFrame();
12        frame.setTitle(title);
13        frame.setSize(width, height);
14        frame.getContentPane().add(this);
15        frame.setVisible(true);
16        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
17    }
18
19③     public void paintComponent(Graphics g) {
20         g.setColor(Color.white);
21         g.fillRect(0, 0, 300, 200);
22         g.setColor(Color.red);
23         g.drawRect(100, 60, 90, 60);
24         g.fillOval(100, 60, 50, 50);
25     }
26
27     // test code
28④     public static void main(String[] args) {
29         new Canvas();
30     }
31
32 }

```

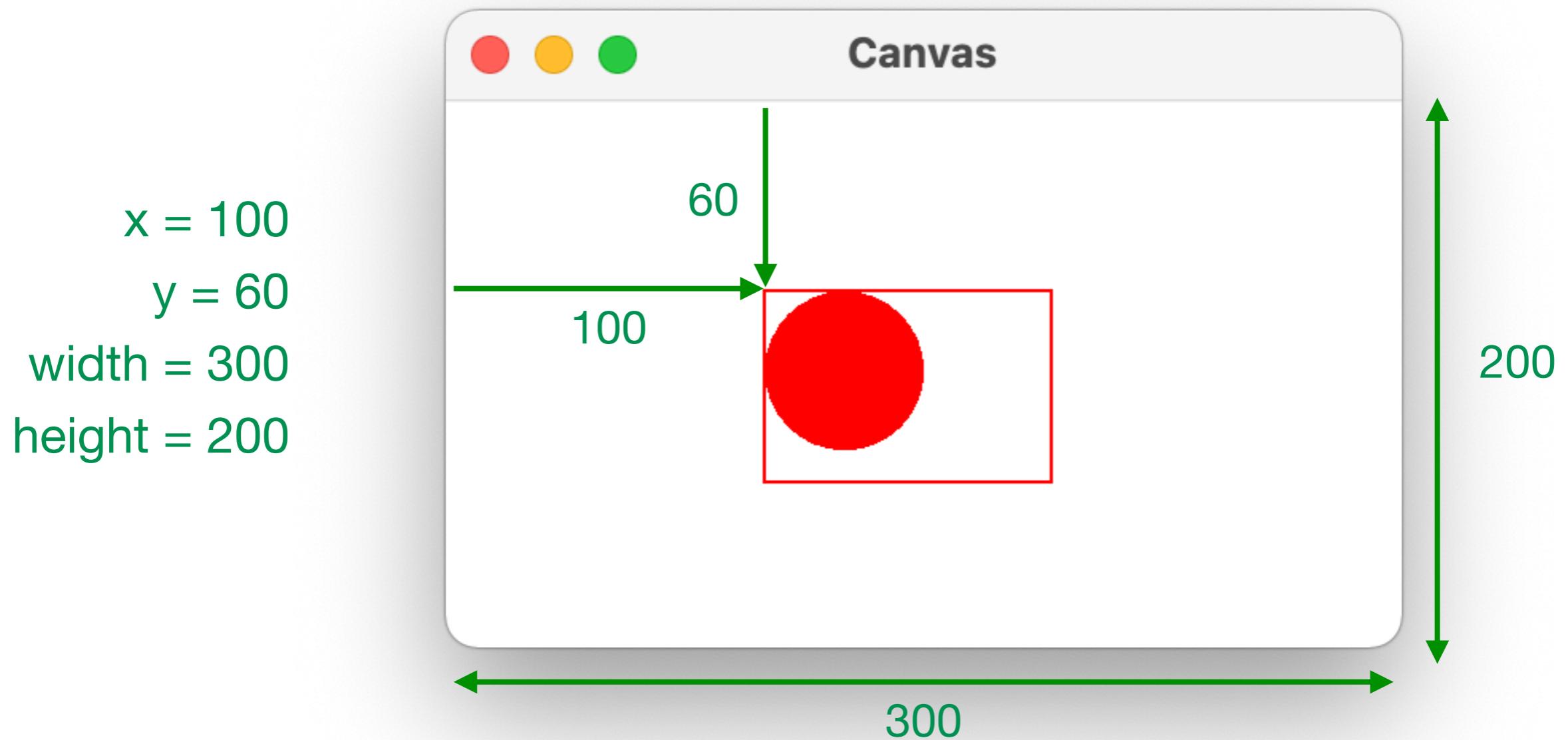


Graphics

그래픽스

위치 및 크기의 단위

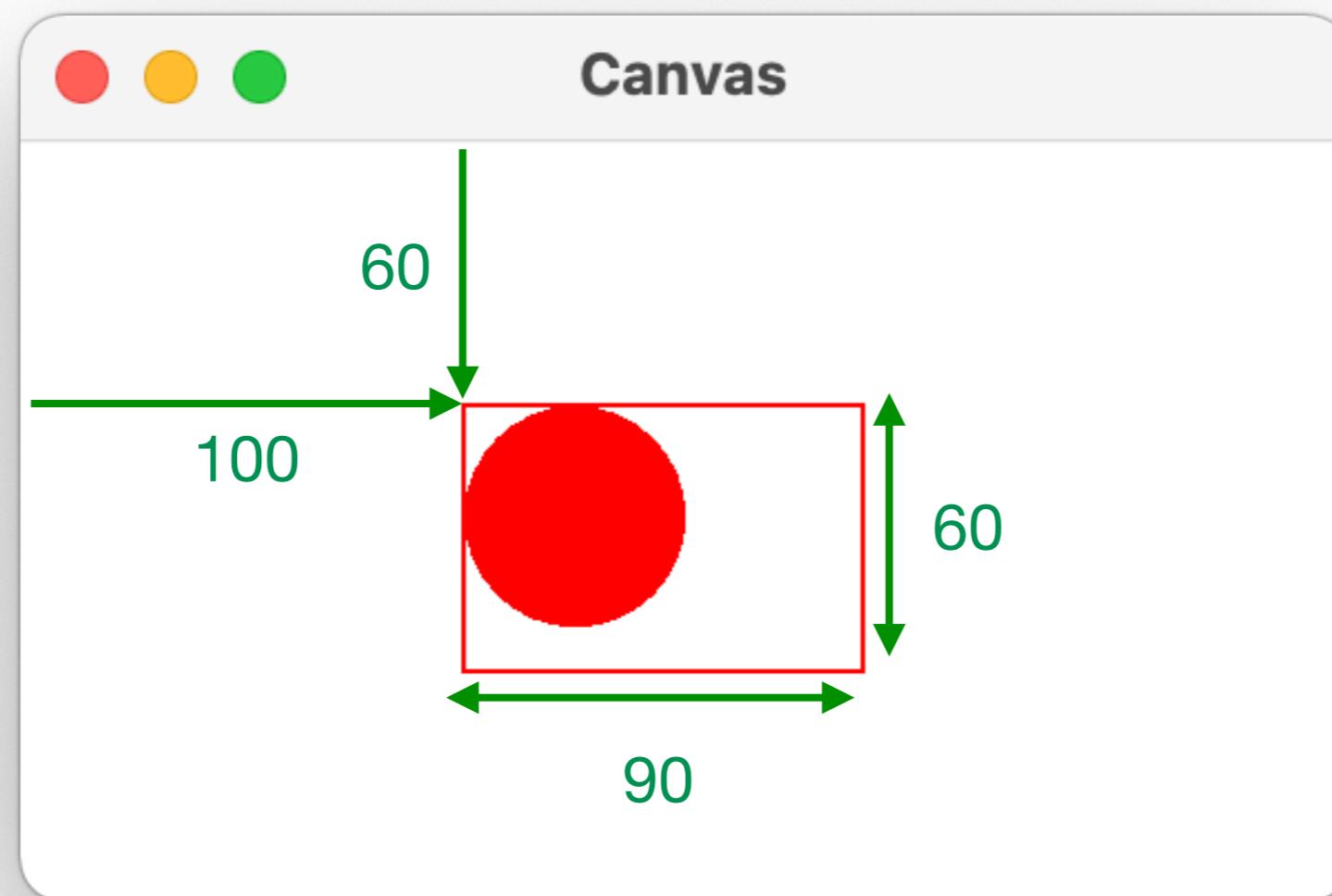
Pixels



사각형 그리기

```
g.drawRect(100, 60, 90, 60);
```

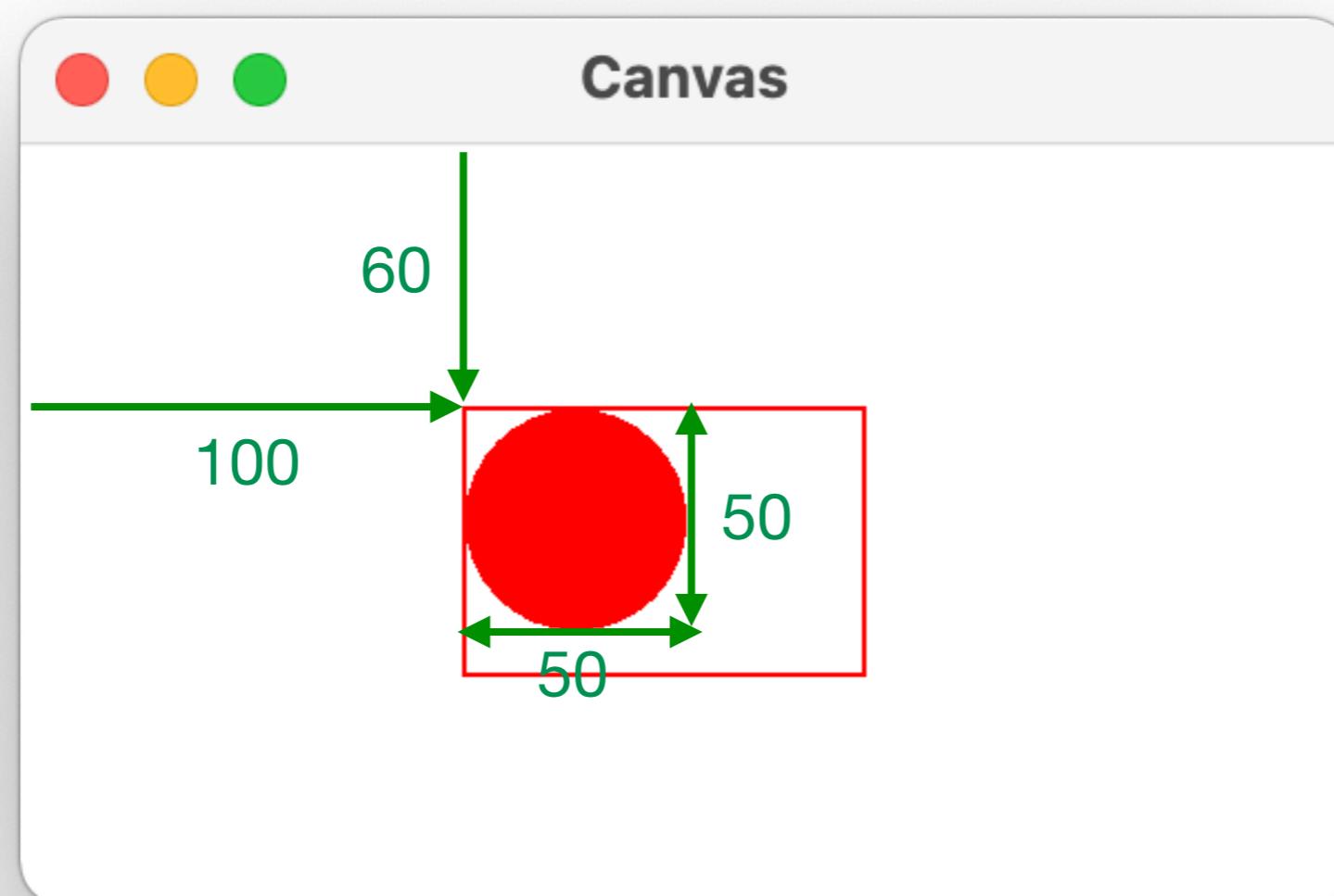
x y width height



타원 그리기

```
g.filloval(100, 60, 50, 50);
```

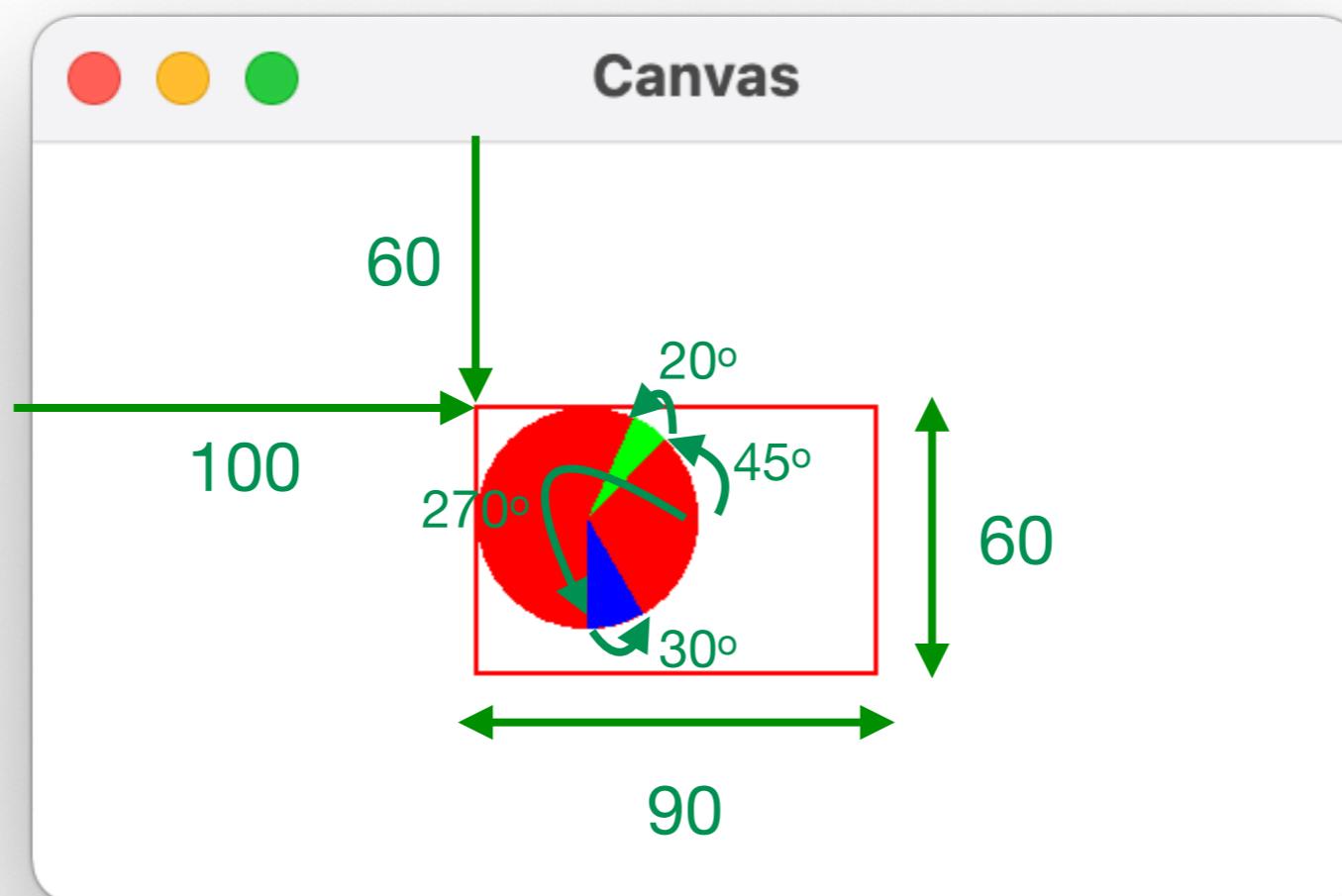
x y width height



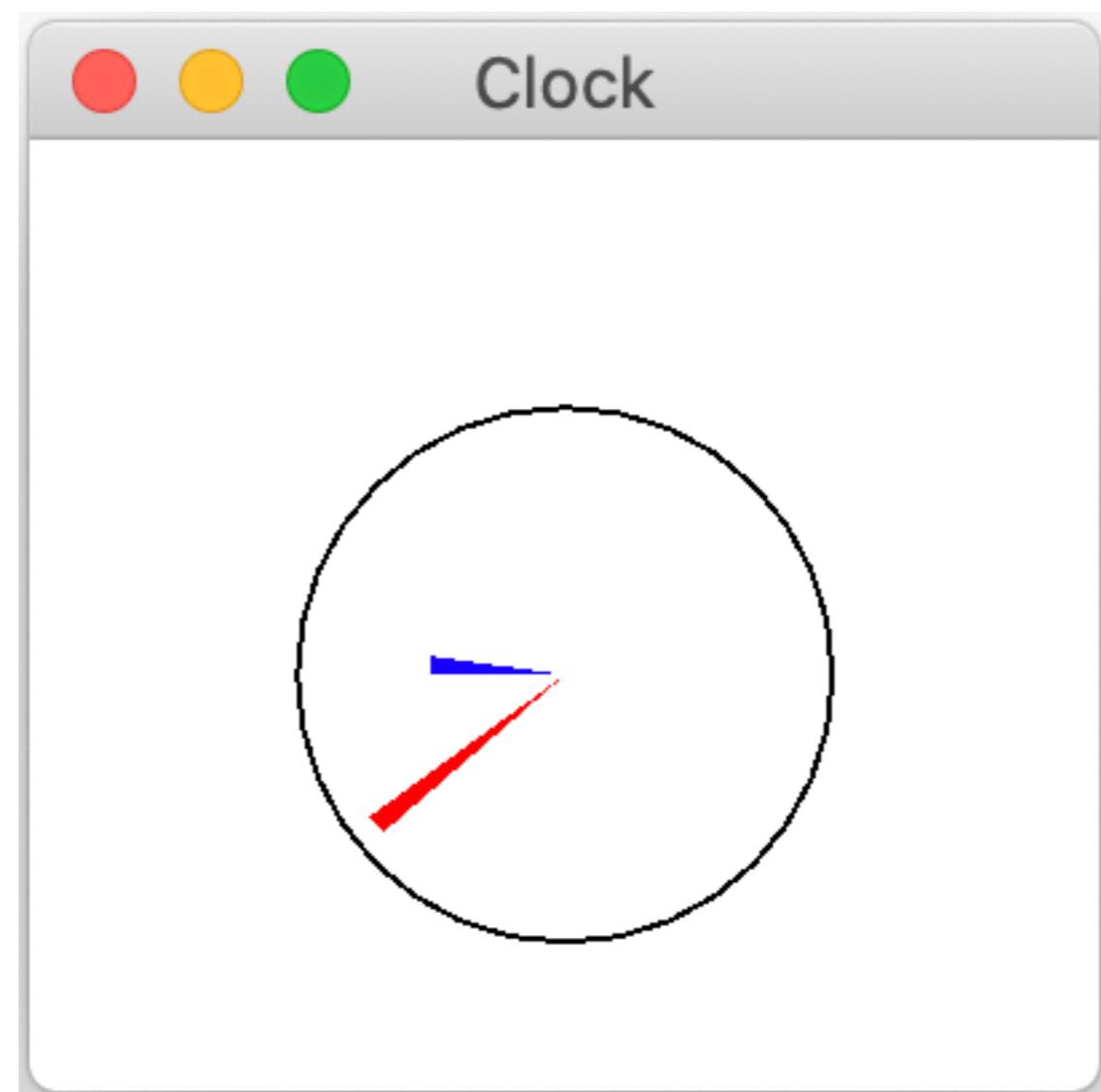
타원 그리기

```
g.setColor(Color.green);  
g.fillArc(100, 60, 50, 50, 45, 20);  
g.setColor(Color.blue);  
g.fillArc(100, 60, 50, 50, 270, 30);
```

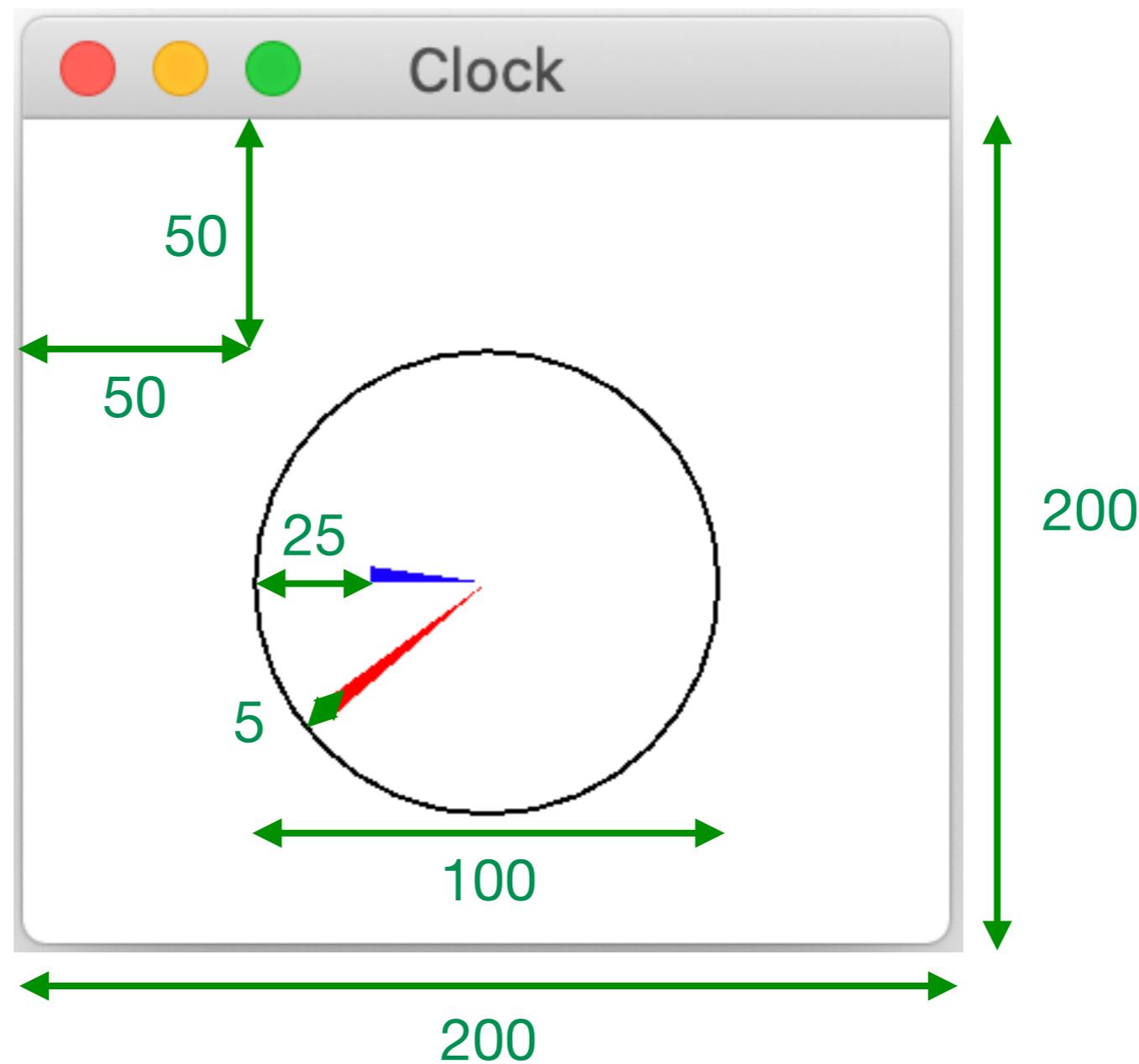
startAngle
arcAngle



아날로그 시계 만들기



크기와 위치 설계



ClockWriter.java

```
1① import java.awt.*;
2 import javax.swing.*;
3 import java.time.*;
4
5 public class ClockWriter extends JPanel {
6
7②     public ClockWriter() {
8         int width = 200;
9         // 프레임 생성
10        JFrame frame = new JFrame();
11        // 자신(패널)을 프레임에 끼우기
12        frame.getContentPane().add(this);
13        // 프레임 만들어 보여주기
14        frame.setTitle("Clock");
15        frame.setSize(width, width);
16        frame.setVisible(true);
17        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
18    }
19}
```

```
20⊕ public void paintComponent(Graphics g) {  
21     int width = 200;  
22     // 바탕은 흰색으로  
23     g.setColor(Color.white);  
24     g.fillRect(0, 0, width, width);  
25     // 현재 시간 + 시침, 분침 각도 계산  
26     LocalTime now = LocalTime.now();  
27     int minutes_angle = 90 - now.getMinute() * 6;  
28     int hours_angle = 90 - now.getHour() * 30;  
29     // 시계 크기 설정  
30     int x = 50;  
31     int y = 50;  
32     int diameter = 100;  
33     // 시계 판 그리기  
34     g.setColor(Color.black);  
35     g.drawOval(x, y, diameter, diameter);  
36     // 분침 그리기  
37     g.setColor(Color.red);  
38     g.fillArc(x+5, x+5, diameter-10, diameter-10, minutes_angle, 5);  
39     // 시침 그리기  
40     g.setColor(Color.blue);  
41     g.fillArc(x+25, x+25, diameter-50, diameter-50, hours_angle, -8);  
42 }  
43  
44⊕ public static void main(String[] args) {  
45     new ClockWriter();  
46 }  
47 }
```

필드 변수

Field Variables

= 상태 변수

Objects with State

객체

| | | |
|-----------------------|--------------------|--|
| 객체 Object | 메모리에 존재하는 실체 | 메모리 주소로 식별 |
| 필드 Field | 객체의 내부 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

클래스
내부 전용

ClockWriter.java

```

1① import java.awt.*;
2 import javax.swing.*;
3 import java.time.*;
4
5 public class ClockWriter extends JPanel {
6
7     private int width = 200;
8
9     public ClockWriter() {
10         // 프레임 생성
11         JFrame frame = new JFrame();
12         // 자신(패널)을 프레임에 끼우기
13         frame.getContentPane().add(this);
14         // 프레임 만들어 보여주기
15         frame.setTitle("Clock");
16         frame.setSize(width, width);
17         frame.setVisible(true);
18         frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
19     }
20 }
```

필드 변수
클래스 내부에서만
공유 및 수정 가능

지역 변수
local variable

```

21@  public void paintComponent(Graphics g) {
22
23      // 바탕은 흰색으로
24      g.setColor(Color.white);
25      g.fillRect(0, 0, width, width);
26      // 현재 시간 + 시침, 분침 각도 계산
27      LocalTime now = LocalTime.now();
28      int minutes_angle = 90 - now.getMinute() * 6;
29      int hours_angle = 90 - now.getHour() * 30;
30      // 시계 크기 설정
31      int x = 50;
32      int y = 50;
33      int diameter = 100;
34      // 시계 판 그리기
35      g.setColor(Color.black);
36      g.drawOval(x, y, diameter, diameter);
37      // 분침 그리기
38      g.setColor(Color.red);
39      g.fillArc(x+5, x+5, diameter-10, diameter-10, minutes_angle, 5);
40      // 시침 그리기
41      g.setColor(Color.blue);
42      g.fillArc(x+25, x+25, diameter-50, diameter-50, hours_angle, -8);
43  }
44
45@  public static void main(String[] args) {
46      new ClockWriter();
47  }
48 }

```

지역 변수
local variable

ClockWriter.java

```

1@ import java.awt.*;
2 import javax.swing.*;
3 import java.time.*;
4
5 public class ClockWriter extends JPanel {
6
7     private int width;
8
9@     public ClockWriter(int w) {
10        // 프레임 크기 기준값 설정
11        width = w;
12        // 프레임 생성
13        JFrame frame = new JFrame();
14        // 자신(패널)을 프레임에 끼우기
15        frame.getContentPane().add(this);
16        // 프레임 만들어 보여주기
17        frame.setTitle("Clock");
18        frame.setSize(width, width);
19        frame.setVisible(true);
20        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
21    }
22

```

필드 변수의 값
객체 생성시
인수로 제공

ClockWriter.java

```

1① import java.awt.*;
2 import javax.swing.*;
3 import java.time.*;
4
5 public class ClockWriter extends JPanel {
6
7     private final int WIDTH;
8
9     public ClockWriter(int w) {
10         // 프레임 크기 기준값 설정
11         WIDTH = w;
12         // 프레임 생성
13         JFrame frame = new JFrame();
14         // 자신(패널)을 프레임에 끼우기
15         frame.getContentPane().add(this);
16         // 프레임 만들어 보여주기
17         frame.setTitle("Clock");
18         frame.setSize(WIDTH, WIDTH);
19         frame.setVisible(true);
20         frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
21     }
22 }
```

final 변수

- 값을 한 번밖에 지정할 수 없음
- 값이 변하지 않으므로 상수(constant)라고 함
- 모두 대문자로 씀

```

23@  public void paintComponent(Graphics g) {
24
25      // 바탕은 흰색으로
26      g.setColor(Color.white);
27      g.fillRect(0, 0, WIDTH, WIDTH);
28      // 현재 시간 + 시침, 분침 각도 계산
29      LocalTime now = LocalTime.now();
30      int minutes_angle = 90 - now.getMinute() * 6;
31      int hours_angle = 90 - now.getHour() * 30;
32      // 시계 크기 설정
33      int x = WIDTH / 4;
34      int y = WIDTH / 4;
35      int diameter = WIDTH / 2;
36      // 시계 판 그리기
37      g.setColor(Color.black);
38      g.drawOval(x, y, diameter, diameter);
39      // 분침 그리기
40      g.setColor(Color.red);
41      g.fillArc(x+5, x+5, diameter-10, diameter-10, minutes_angle, 5);
42      // 시침 그리기
43      g.setColor(Color.blue);
44      g.fillArc(x+25, x+25, diameter-50, diameter-50, hours_angle, -8);
45  }
46
47@  public static void main(String[] args) {
48      new ClockWriter(500);
49  }
50 }

```

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 유효 범위 Scope

```
1① import java.awt.*;
2 import javax.swing.*;
3
4 public class FieldExample extends JPanel {
5
6     private int count;
7
8     public FieldExample() {
9         count = 0;
10        JFrame f = new JFrame();
11        f.getContentPane().add(this);
12        f.setSize(300, 200);
13        f.setVisible(true);
14    }
15
16     public void paintComponent(Graphics g) {
17         count = count + 1;
18         g.setColor(Color.black);
19         g.drawString(count + "번 그렸습니다", 25, 40);
20     }
21
22     public static void main(String[] args) {
23         new FieldExample();
24     }
25 }
```

The diagram illustrates the scope of the variable `count`. A green vertical arrow points upwards from its declaration at line 6 to its assignment in the constructor at line 10. Another green vertical arrow points downwards from its declaration to its use in the `paintComponent` method at line 19. A red double-headed horizontal arrow connects the variable `g` in the constructor at line 10 to the variable `g` in the `paintComponent` method at line 16.

변수의 유효 범위 Scope

```

3① public static void main(String[] args) {
4    { int n = 2;
5
6        System.out.println(n);
7    }
8    double n = 3.14;
9
10   System.out.println(n);
11
12 }
13

```

type error →

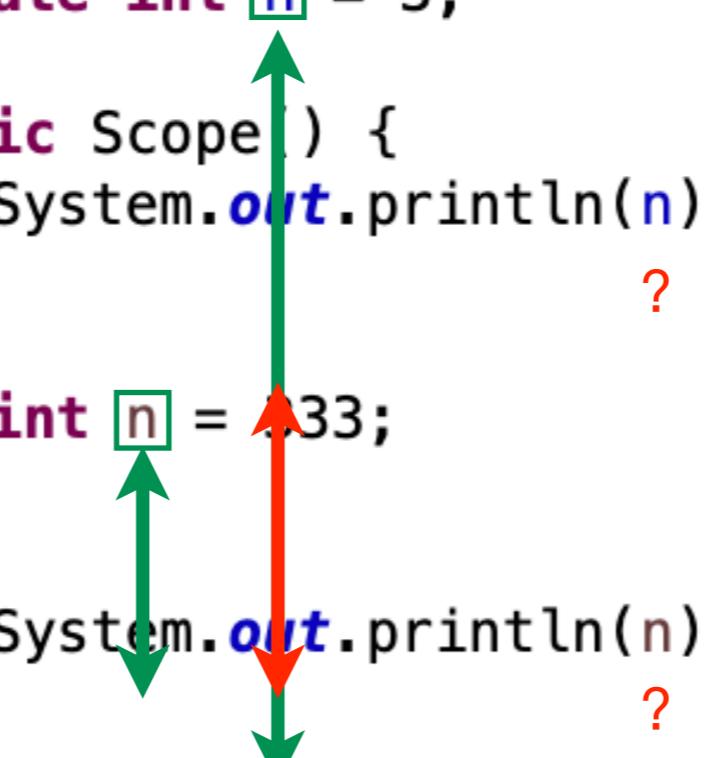
```

3① public static void main(String[] args) {
4    int n = 2;
5
6    System.out.println(n);
7    {
8        double n = 3.14;
9
10       System.out.println(n);
11
12    }
13

```

변수의 유효 범위 Scope

```
1 public class Scope {  
2  
3     private int n = 3;  
4  
5     public Scope() {  
6         System.out.println(n);  
7     }  
8  
9     int n = 33;  
10  
11    System.out.println(n);  
12  
13 }  
14  
15     public static void main(String[] args) {  
16         new Scope();  
17     }  
18 }
```



변수의 유효 범위 Scope

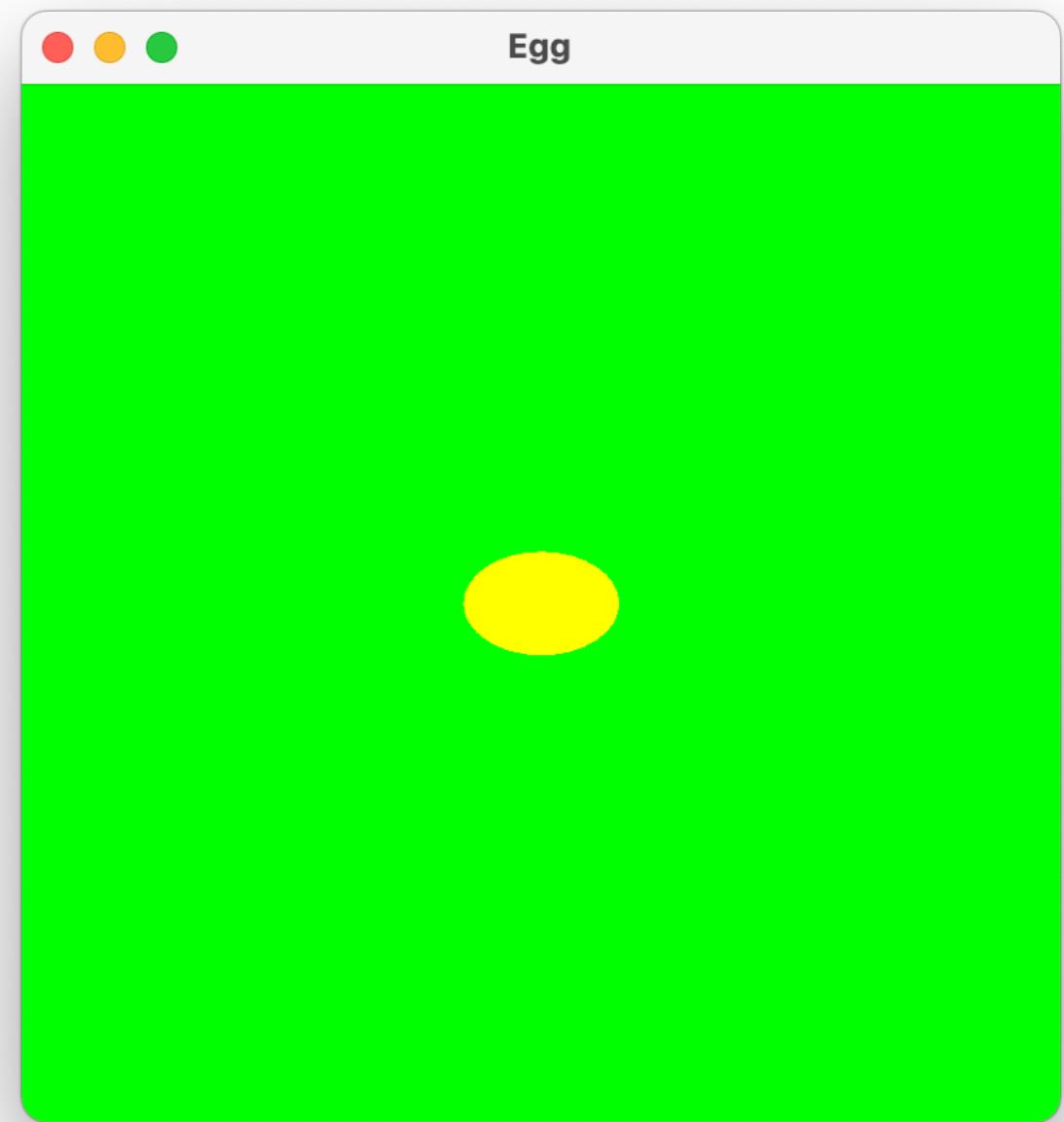
```
1 import java.awt.*;
2
3 public class Contrived {
4
5     private double d = 3.14;
6
7     ⊕ public Contrived() {
8         System.out.println(s);
9         System.out.println(d);
10        int d = 2;
11        System.out.println(d);
12        s = d + s;
13        System.out.println(s);
14    }
15
16    private String s = "X" + d;
17
18     ⊕ public void printComponent(Graphics g) {
19         System.out.println(d + " " + s);
20     }
21
22     ⊕ public static void main(String[] args) {
23         new Contrived();
24     }
25 }
```

실습

자라는 알 Growing Egg

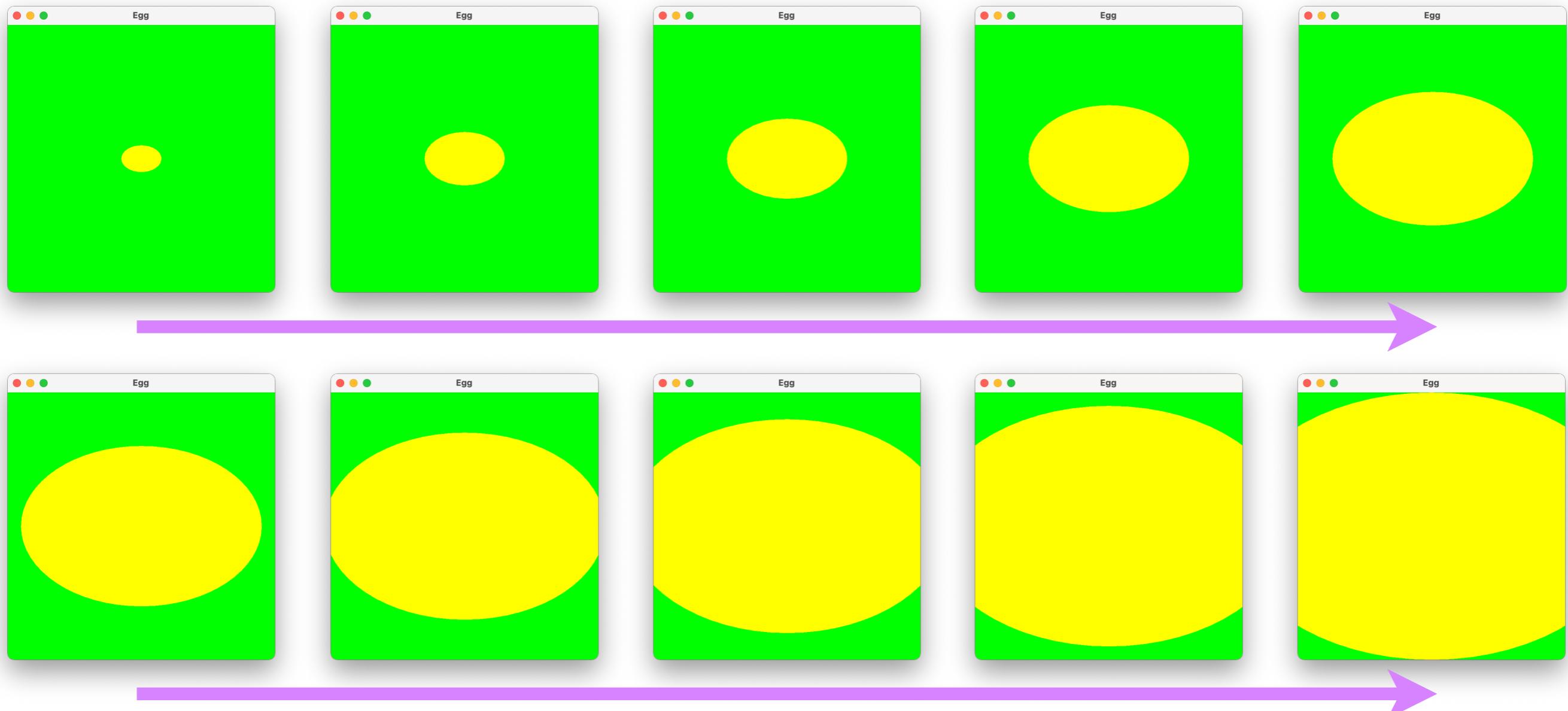
1 단계

- pixel 단위로 400x400크기의 창에 다음과 같아 녹색(Color.green) 바탕에 노란색(Color.yellow)의 알 모양 타원을 정 가운데 보여주는 창을 만드는 클래스 GrowingEgg를 만들자.
- 알의 크기는 너비가 60, 높이가 40 pixel 이다.



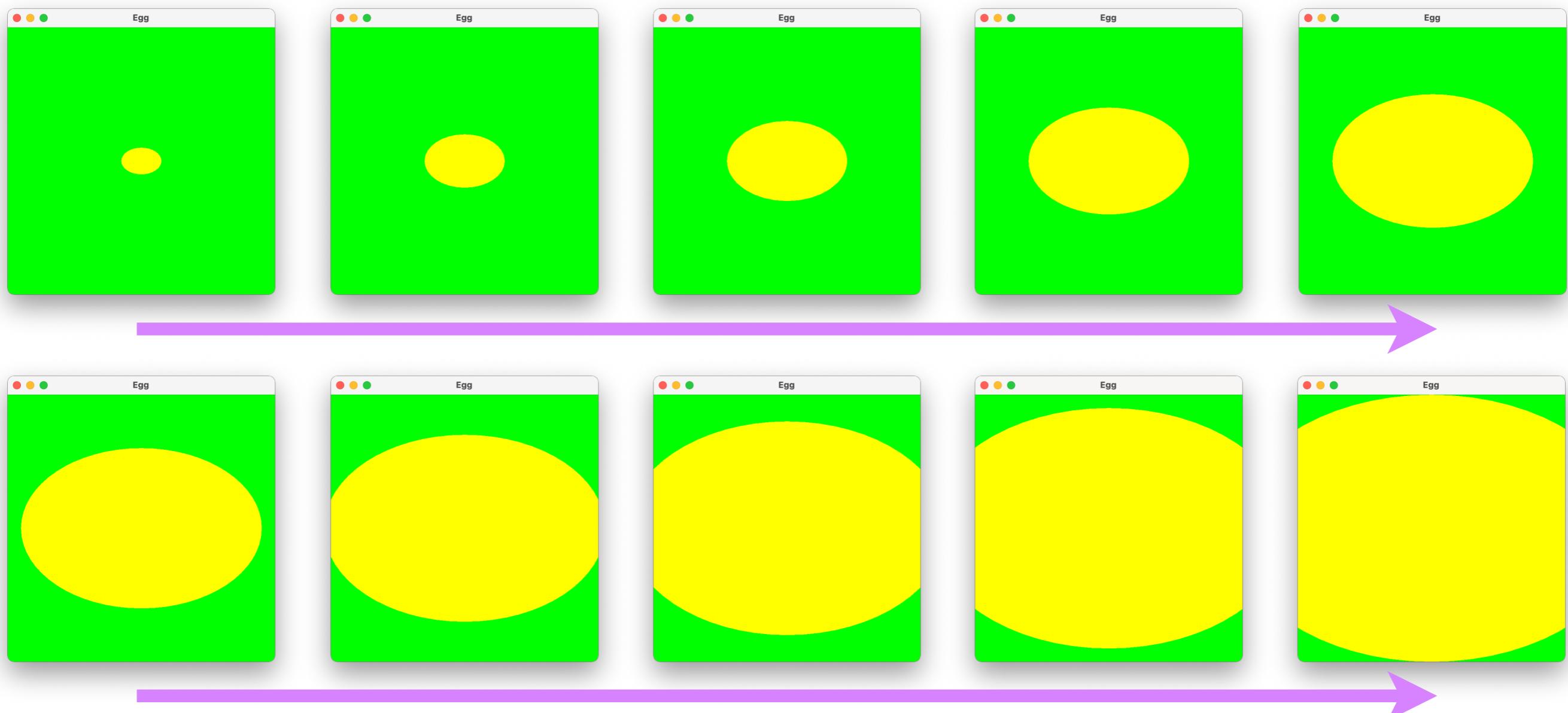
2 단계

- 창을 최소화하고 다시 활성화하면, `paintComponent` 메소드가 저절로 다시 실행되면서 창을 다시 그린다.
- 창을 활성화할 때마다 알이 아래와 같이 너비 60, 높이 40 pixel 씩 자라도록 `GrowingEgg`을 수정하자.
- 알의 중심은 항상 일치해야 한다.



3 단계

- 창의 크기를 400x400으로 고정하는 대신, GrowingEgg(400)과 같이 창의 크기를 사용자가 정할 수 있게 애플리케이션을 개선하자.
- 알의 크기와 자라는 비율은 창의 크기와 비례해야 한다.



숙제

아날로그 시계 (1 단계)

1. 수업 시간에 공부한 아날로그 시계에 다음 요구사항에 맞추어 초침을 추가하자.

- 초침은 검정색(Color.black)으로 한다.
- 초침의 원둘레로 부터의 거리는 10 픽셀, 두께 각도는 3도로 한다.
- 시침, 분침, 초침은 항상 시계 창을 여는 시점의 정확한 현재 시각을 가리켜야 한다.

2. 시계에 눈금을 추가한다.

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