



목차

- 윈도우 프레임
- 도형 그리기
- 색과 폰트
- 이미지 그리기
- 그리기 응용





윈도우 프레임 [1/2]

■ 윈도우 프레임

■ 제목표시줄을 갖는 윈도우를 의미

■ 생성 과정

- ① JFrame 객체 생성
- ② 프레임의 크기 설정
- ③ 프레임의 제목 설정
- ④ 기본 닫힘 연산 지정
- ⑤ 프레임이 보이도록 만들.





윈도우 프레임 [2/2]

■ 윈도우 프레임 예제

[예제 8.1 - EmptyFrameViewer.java]

```
import javax.swing.*;
public class EmptyFrameViewer {
    public static void main(String[] args) {
        JFrame frame = new JFrame();                // ①

        final int FRAME_WIDTH = 300;
        final int FRAME_HEIGHT = 400;

        frame.setSize(FRAME_WIDTH, FRAME_HEIGHT);    // ②
        frame.setTitle("An Empty Frame");            // ③
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // ④

        frame.setVisible(true);                      // ⑤
    }
}
```



도형 그리기 [1/4]

■ 그래픽 객체

- 자바의 그래픽 기능을 위해 제공되는 객체
- java.awt.Graphics 클래스의 객체
- 선, 사각형, 타원, 호, 다각형 등과 같은 도형을 그리는데 필요한 기본적인 메소드를 가지고 있음
- JComponent 클래스의 paintComponent() 메소드를 재정의하여 얻음
 - 화면을 다시 그려야 할 때, 자바에 의해 호출되는 메소드

```
class UserDefinedComponent extends JComponent {  
    public void paintComponent(Graphics g) {  
        // 그래픽 객체를 이용한 그리기 작업  
    }  
}
```



도형 그리기 [2/4]

■ 그리기 과정

- ① 윈도우 프레임을 만들
- ② 컴포넌트 클래스의 객체를 생성

```
UserDefinedComponent component = new UserDefinedComponent();
```

- ③ 컴포넌트 객체를 윈도우 프레임에 추가

```
frame.add(component);  
또는  
frame.getContentPane().add(component);
```

- ④ 윈도우 프레임을 보이도록 만들



도형 그리기 [3/4]

■ 사각형 그리기 예제

[예제 8.2 - DrawingSchemeViewer.java]

```
import java.awt.*;
import javax.swing.*;
class UserDefinedComponent extends JComponent {           // 그리기를 위한 클래스
    public void paintComponent(Graphics g) {
        g.drawRect(10, 20, 200, 150);
    }
}
public class DrawingSchemeViewer {                          // 보여주기 위한 클래스
    public static void main(String[] args) {
        JFrame frame = new JFrame();                       // ①

        frame.setSize(320, 240);
        frame.setTitle("사각형 그리기");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

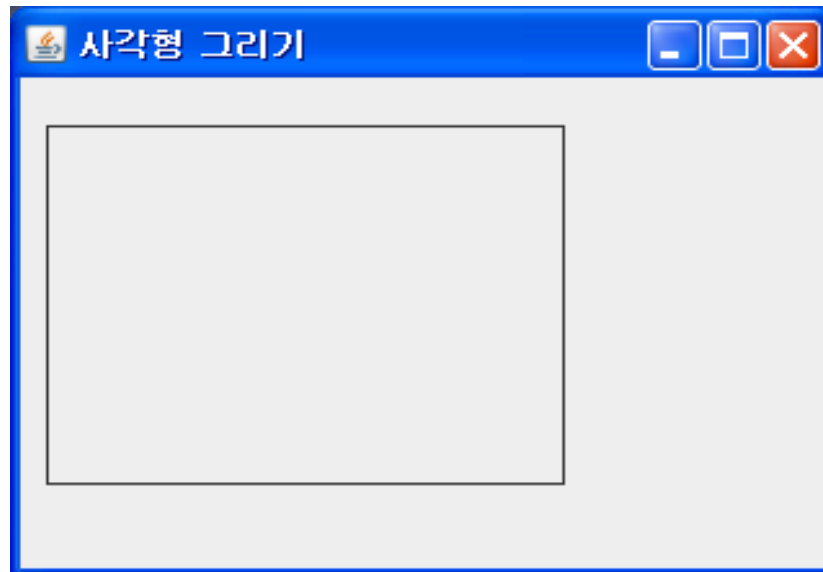
        UserDefinedComponent component = new UserDefinedComponent(); // ②
        frame.add(component);                                // ③

        frame.setVisible(true);                              // ④
    }
}
```



도형 그리기 [4/4]

■ 사각형 그리기 실행 결과





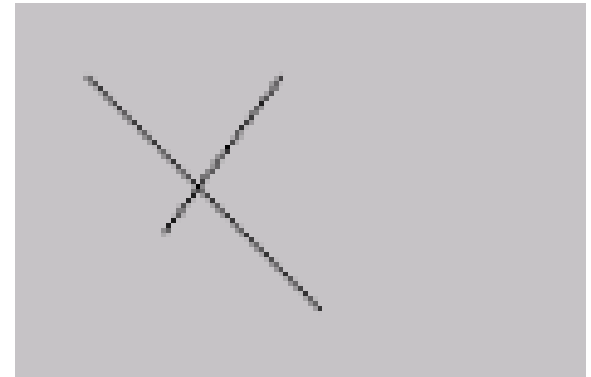
도형 그리기 : 선

■ 형식

```
void drawLine(startx, starty, endx, endy)
```

■ 예제

```
g.drawLine(20, 20, 80, 80);  
g.drawLine(70, 20, 40, 60);
```





도형 그리기 : 사각형 [1/3]

■ 종류

- 사각형, 둥근 모서리 사각형, 3차원 사각형
- 외각선만 그리는 drawRect() 메소드
- 채워진 사각형을 그리는 fillRect() 메소드

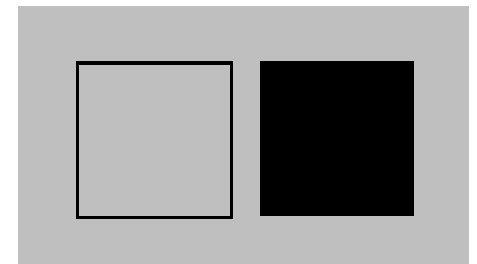
■ 일반 사각형 그리기

■ 형식

```
void drawRect(startx, starty, width, height)  
void fillRect(startx, starty, width, height)
```

■ 예제

```
g.drawRect(20, 20, 50, 50);  
g.fillRect(80, 20, 50, 50);
```





도형 그리기 : 사각형 [2/3]

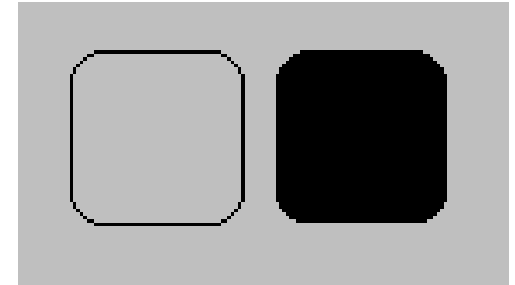
■ 둥근 모서리 사각형 그리기

■ 형식

```
void drawRoundRect(start_x, start_y, w, h, arc_w, arc_h)  
void fillRoundRect(start_x, start_y, w, h, arc_w, arc_h)
```

■ 예제

```
g.drawRoundRect(20, 20, 50, 50, 20, 20);  
g.fillRoundRect(80, 20, 50, 50, 20, 20);
```





도형 그리기 : 사각형 [3/3]

■ 3차원 사각형 그리기

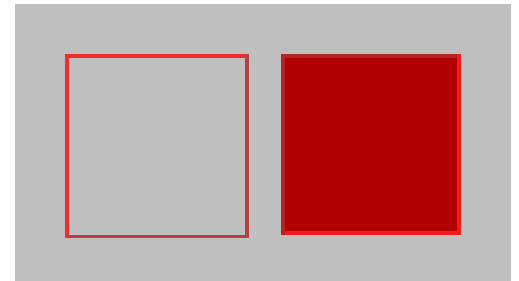
■ 형식

```
void draw3DRect(startx, starty, width, height, raised)  
void fill3DRect(startx, starty, width, height, raised)
```

- raised 값이 true이면, **볼록한** 3차원 사각형을 그림
- raised 값이 false이면, **오목한** 3차원 사각형을 그림

■ 예제

```
g.draw3DRect(20, 20, 50, 50, true);  
g.fill3DRect(80, 20, 50, 50, false);
```



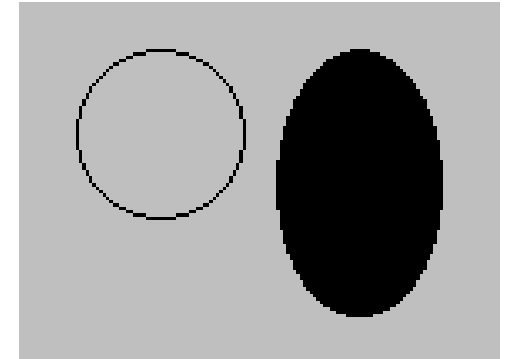


도형 그리기 : 원과 타원

■ 원과 타원 그리기

- drawOval() 메소드와 fillOval() 메소드
- 사각형을 그리는 매개변수와 같음
- 원을 그리기 위해서는 정사각형의 좌표 값을 줌
- 타원을 그리기 위해서는 직사각형의 좌표 값을 줌

```
g.drawOval(20, 20, 50, 50);  
g.fillOval(80, 20, 50, 80);
```



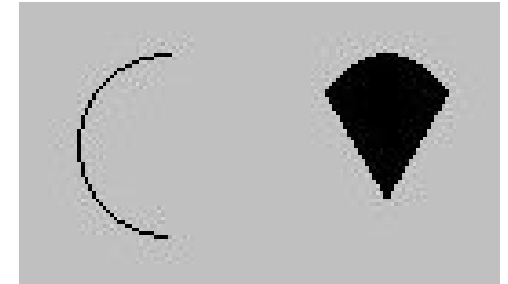


도형 그리기 : 호

■ 호 그리기

- drawArc() 메소드와 fillArc() 메소드
- 4개의 매개변수는 원이나 타원을 그리기 위한 매개변수 값과 같음
- 5번째 매개변수는 호를 그리기 위한 시작각
- 6번째 매개변수는 호의 중심각

```
g.drawArc(20, 20, 50, 50, 90, 180);  
g.fillArc(80, 20, 50, 80, 45, 90);
```





도형 그리기 : 다각형

■ 다각형 그리기

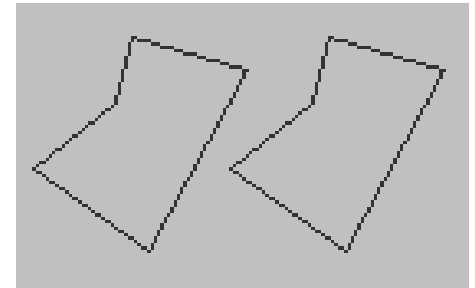
- 각 모서리 점들의 집합으로 표현
- 형식

```
void drawPolygon(Polygon p)  
void drawPolygon(int[] xPoints, int[] yPoints, int nPoints)
```

■ 예제

```
Polygon p = new Polygon();  
p.addPoint (30, 30);  
p.addPoint (5, 50);  
p.addPoint (40, 75);  
p.addPoint (70, 20);  
p.addPoint (35, 10);  
g.drawPolygon(p);
```

```
int[] xPnts = {90, 65, 100, 130, 95};  
int[] yPnts = {30, 50, 75, 20, 10};  
int nPnts = xPnts.length;  
g.drawPolygon(xPnts, yPnts, nPnts);
```





도형 그리기 : 텍스트

■ 텍스트 그리기

■ 형식

```
void drawString(String s, int x, int y)
```

■ 예제

```
g.drawString("Java Graphic", 10, 10);
```



Java Graphic



색과 폰트 [1/5]

■ 색

- java.awt.Color 패키지의 Color 클래스의 객체를 사용
- RGB(Red/Green/Blue) 색상 모델을 사용

```
Color magenta = new Color(1.0F, 0.0F, 1.0F);
```

- 색을 지정하기 위해서는 그래픽 객체의 setColor() 메소드를 사용
 - Color 클래스의 객체나 Color 클래스에서 지원하는 색 상수

```
g.setColor(Color.magenta);  
또는  
g.setColor(Color.MAGENTA);
```



색과 폰트 [2/5]

■ 폰트

- 글자의 폰트를 지정하기 위해서는 java.awt 패키지에서 제공하는 Font 클래스를 사용
- 생성자를 통해 폰트의 종류와 스타일, 그리고 폰트 크기를 지정

폰트의 종류	폰트의 스타일
Helvetica, TimesRoman, Courier, Dialog, DialogInput, ZapfDingbats, default	Font.PLAIN, Font.ITALIC, Font.BOLD, Font.BOLD+Font.ITALIC

■ 예제

```
Font f = new Font("TimesRoman", Font.BOLD+Font.ITALIC, 30);
```

- 그래픽 객체의 setFont() 메소드를 사용하여 폰트를 지정



색과 폰트 [3/5]

[예제 8.3 - FontAndColorViewer.java]

```
import java.awt.*;
import javax.swing.*;
class FontAndColorComponent extends JComponent {
    public void paintComponent(Graphics g) {
        Font fp = new Font("TimesRoman", Font.PLAIN, 17);
        Font fb = new Font("Helvetica", Font.BOLD, 17);
        Font fi = new Font("Courier", Font.ITALIC, 17);
        Font fbi = new Font("Dialog", Font.BOLD+Font.ITALIC, 17);

        g.setFont(fp);
        g.setColor(Color.BLACK);
        g.drawString("Font:TimesRoman, Style:Plain, Color:Black", 10, 22);
        g.setFont(fb);
        g.setColor(Color.RED);
        g.drawString("Font:Helvetica, Style:Bold, Color:Red", 10, 44);
        g.setFont(fi);
        g.setColor(Color.BLUE);
        g.drawString("Font:Courier, Style:Italic, Color:Blue", 10, 66);
        g.setFont(fbi);
        g.setColor(Color.GREEN);
        g.drawString("Font:Dialog, Style:Bold+Italic, Color:GREEN", 10, 88);
    }
}
```



색과 폰트 [4/5]

[예제 8.3 - FontAndColorViewer.java] (cont.)

```
public class FontAndColorViewer {  
    public static void main(String[] args) {  
        JFrame frame = new JFrame();  
  
        frame.setSize(480, 270);  
        frame.setTitle("FontAndColor");  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
  
        FontAndColorComponent component = new FontAndColorComponent();  
        frame.add(component);  
        frame.setVisible(true);  
    }  
}
```



색과 폰트 [5/5]

■ 실행 결과

```
Font:TimesRoman, Style:Plain, Color:Black
Font:Helvetica, Style:Bold, Color:Red
Font:Courier, Style:Italic, Color:Blue
Font:Dialog, Style:Bold+Italic, Color:Green
```



이미지 그리기 [1/5]

■ 이미지 읽기

- javax.imageio 패키지에 있는 ImageIO 클래스를 이용
- 형식

```
static BufferedImage read(File input)
static BufferedImage read(URL input)
```

■ 예제

```
try {
    Image img = ImageIO.read(new File("image.jpg"));

    // 이미지와 관련된 작업 수행.
} catch (IOException e) {
    // IOException에 대한 예외 처리.
}
```



이미지 그리기 [2/5]

■ 이미지 그리기

■ 형태

```
void drawImage(Image img, int x, int y, ImageObserver observer)
```

```
void drawImage(Image img, int x, int y, int w, int h, ImageObserver observer)
```

```
void drawImage(Image img, int dx1, int dy1, int dx2, int dy2,  
               int sx1, int sy1, int sx2, int sy2,  
               ImageObserver observer)
```



이미지 그리기 [3/5]

■ 이미지를 읽어오는 클래스

[예제 8. 4 - DrawImageViewer.java]

```
import java.awt.*;
import java.awt.image.BufferedImage;
import javax.swing.*;
import java.io.*;
import javax.imageio.ImageIO;

class ImageDrawingComponent extends JComponent {
    public void paintComponent(Graphics g) {
        try {
            Image img;
            img = ImageIO.read(new File("image.jpg"));
            g.drawImage(img, 0, 0, null);
        } catch (IOException e) { }
    }
}
```



이미지 그리기 [4/5]

■ 읽어온 이미지를 그리는 클래스

[예제 8. 4 - DrawImageViewer.java] (cont.)

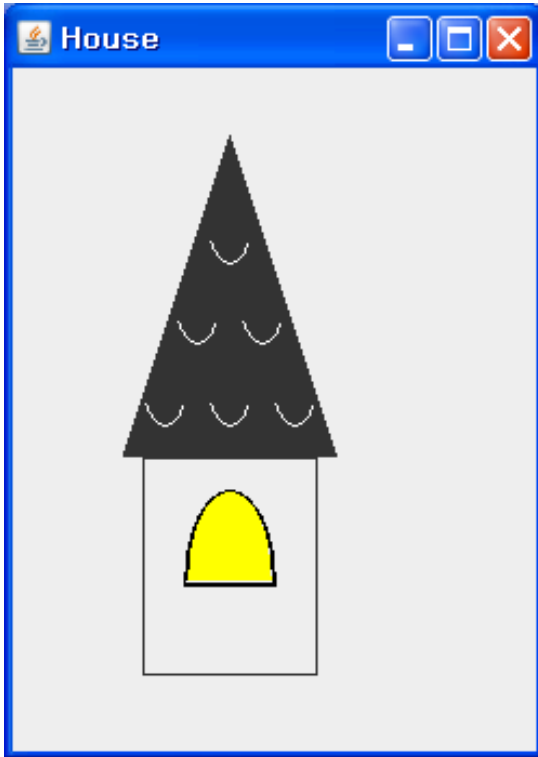
```
public class DrawImageViewer {  
    public static void main(String[] args) {  
        JFrame frame = new JFrame();  
        frame.setSize(320, 240);  
        frame.setTitle("DrawImageTest");  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
  
        ImageDrawingComponent component = new ImageDrawingComponent();  
        frame.add(component);  
        frame.setVisible(true);  
    }  
}
```




그리기 응용 [1/2]

■ 집 그리기

■ 예제 8.5: HouseViewer.java





그리기 응용 [2/2]

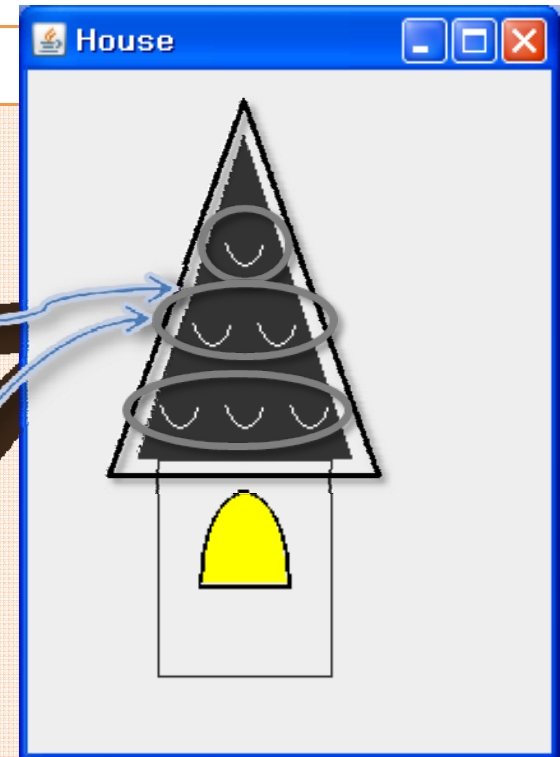
■ 집 그리기 예제 (주요 그리기 부분 코드)

[예제 8. 5 - HouseViewer.java]

```
class HouseDrawingComponent extends JComponent {  
    public void paintComponent(Graphics g) {
```

```
        // 단계 1: 지붕을 그린다.  
        int[] xPnts = {100, 50, 150, 100};  
        int[] yPnts = {30, 180, 180, 30};  
        int nPnts = xPnts.length;  
        g.fillPolygon(xPnts, yPnts, nPnts);
```

```
        // 지붕 위의 기와 모양을 그린다.  
        g.setColor(Color.WHITE);  
        g.drawArc(90, 50, 20, 40, 210, 120);  
        g.drawArc(75, 87, 20, 40, 210, 120);  
        g.drawArc(105, 87, 20, 40, 210, 120);  
        g.drawArc(60, 125, 20, 40, 210, 120);  
        g.drawArc(90, 125, 20, 40, 210, 120);  
        g.drawArc(120, 125, 20, 40, 210, 120);
```





그리기 응용 [2/2]

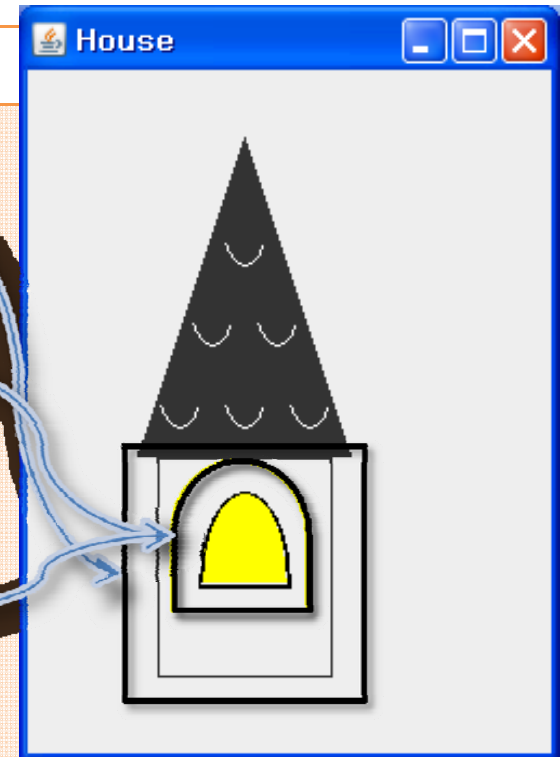
■ 집 그리기 예제 (주요 그리기 부분 코드)

[예제 8. 5 - HouseViewer.java (cont.)]

```
// 단계 2: 몸체를 그린다.  
g.setColor(Color.BLACK);  
g.drawRect(60, 180, 80, 100);
```

```
// 반원 모양의 창문을 그린다.  
g.setColor(Color.YELLOW);  
g.fillArc(80, 195, 40, 85, 0, 180);
```

```
// 노란색 창문 주위를 2개의 검은색 선으로 그린다.  
g.setColor(Color.BLACK);  
g.drawArc(80, 195, 40, 85, 0, 180);  
g.drawArc(79, 195, 42, 86, 0, 180);  
g.drawLine(79, 238, 120, 238);  
g.drawLine(79, 239, 120, 239);  
}
```

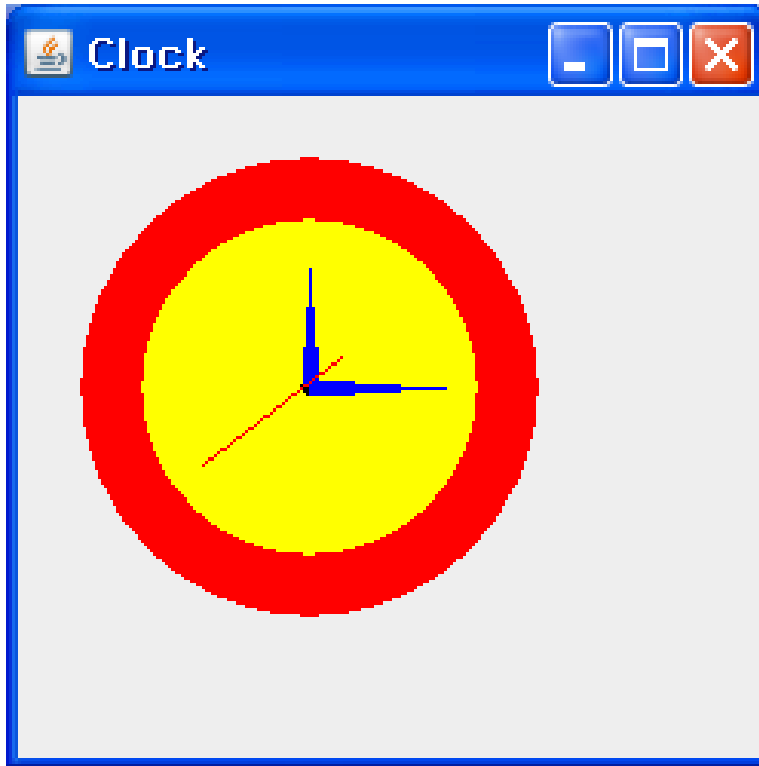




그리기 응용 [2/2]

■ 시계 그리기

■ 예제 8.6: ClockViewer.java





그리기 응용 [2/2]

■ 시계 그리기 예제 (주요 그리기 부분 코드)

[예제 8. 6 - ClockViewer.java]

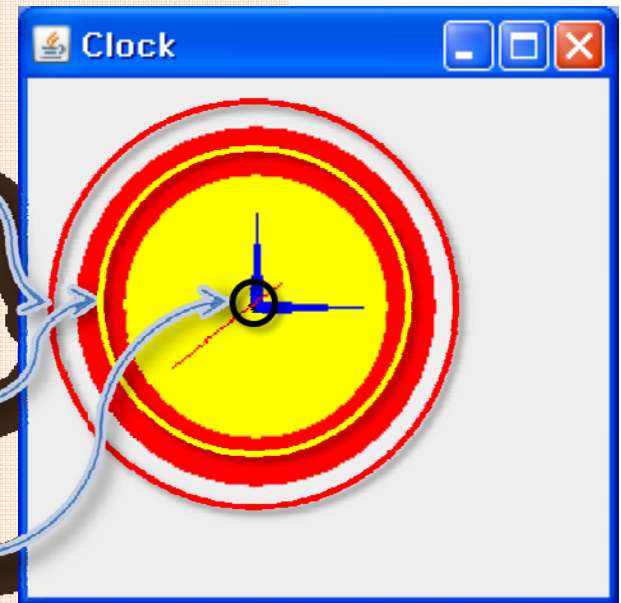
```
class ClockDrawingComponent extends JComponent {  
    public void paintComponent(Graphics g) {
```

```
        Polygon mp = new Polygon();  
        Polygon mp = new Polygon();
```

```
        // 시계 바깥 원  
        g.setColor(Color.RED);  
        g.fillOval(20, 20, 150, 150);
```

```
        // 시계 안쪽 원  
        g.setColor(Color.YELLOW);  
        g.fillOval(40, 40, 110, 110);
```

```
        // 중심점  
        g.setColor(Color.BLACK);  
        g.fillOval(92, 92, 6, 6);
```





그리기 응용 [2/2]

■ 시계 그리기 예제 (주요 그리기 부분 코드)

[예제 8. 6 - ClockViewer.java] (cont.)

```
// 시침
hp.addPoint(92, 95);
hp.addPoint(95, 55);
hp.addPoint(98, 95);
g.setColor(Color.BLUE);
g.fillPolygon(hp);

// 분침
mp.addPoint(95, 92);
mp.addPoint(140, 95);
mp.addPoint(95, 98);
g.fillPolygon(mp);

// 초침
g.setColor(Color.RED);
g.drawLine(105, 85, 60, 120);
}
```