

자바 GUI 기초, 스윙(Swing)

한성대학교 컴퓨터공학부

신 성



자바의 GUI(Graphical User Interface) – 그동안에는 모두 콘솔-기반 프로그램만 작성

■ GUI 응용 프로그램

- GUI : 사용자가 편리하게 입출력 할 수 있도록 그래픽으로 화면을 구성하고,
마우스나 키보드로 입력 받을 수 있도록 지원하는 사용자 인터페이스

■ AWT와 Swing 패키지

- AWT : 자바가 처음 나왔을 때부터 배포된 GUI 패키지 **가끔**
- Swing : AWT 기술을 기반으로 작성된 자바 라이브러리 **자주 씀**

모든 AWT 기능 + 추가된 풍부하고 화려한 고급 컴포넌트

AWT 컴포넌트를 모두 스윙으로 재작성. AWT 컴포넌트 이름 앞에 J자를 덧붙임

(※ 그 외 JavaFx 패키지 등이 존재)

※ **편하게 GUI를 만들라고**

**미리 만들어 놓은 클래스 설명밖에 안되기 때문에
자바 기초만 튼튼히 하면 쉽게 이해할 수 있습니다.**

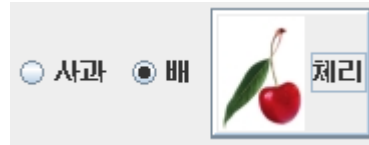
스윙 컴포넌트 예시



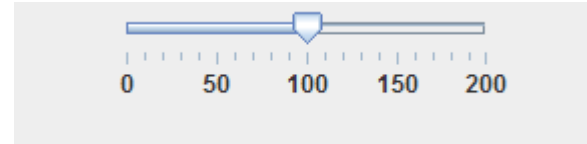
JButton



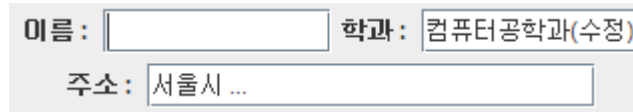
JCheckBox



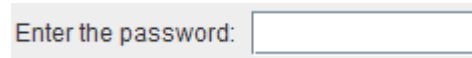
JRadioButton



JSlider



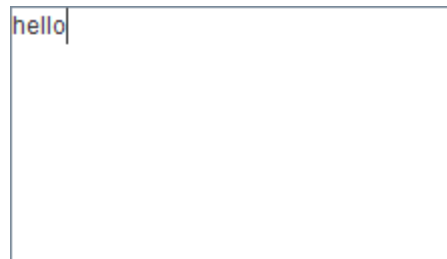
JTextField



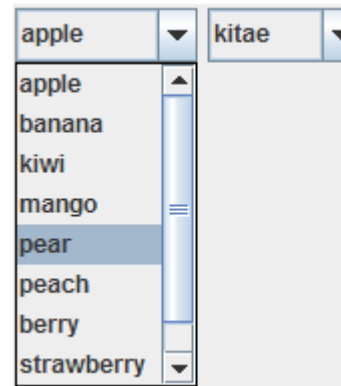
JPasswordField



JSpinner



JTextArea



JComboBox

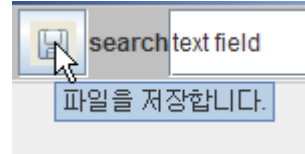


JList

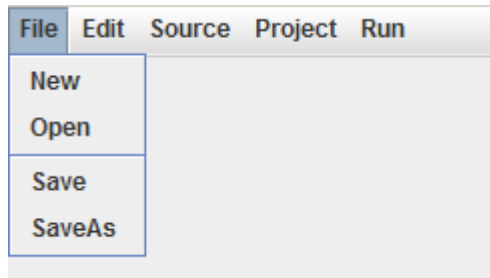
스윙 컴포넌트 예시



JProgressBar



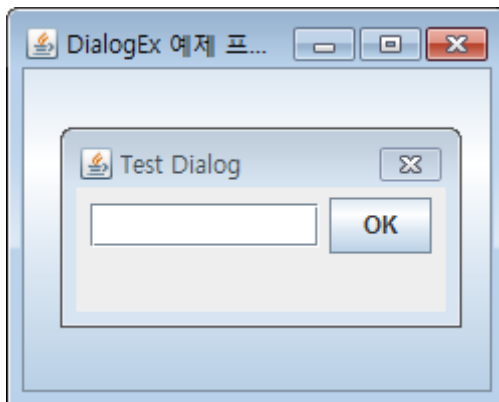
JToolTip



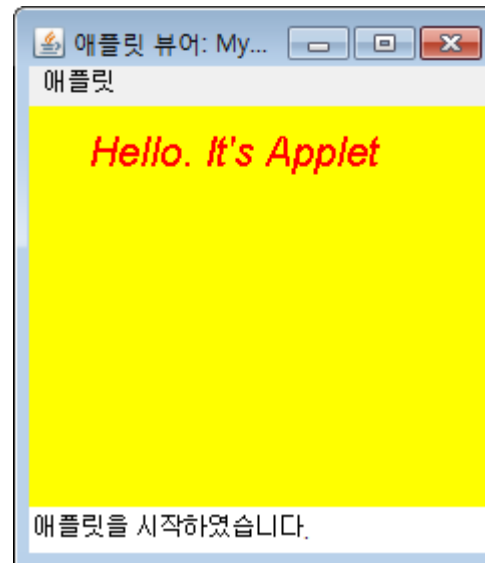
JMenu



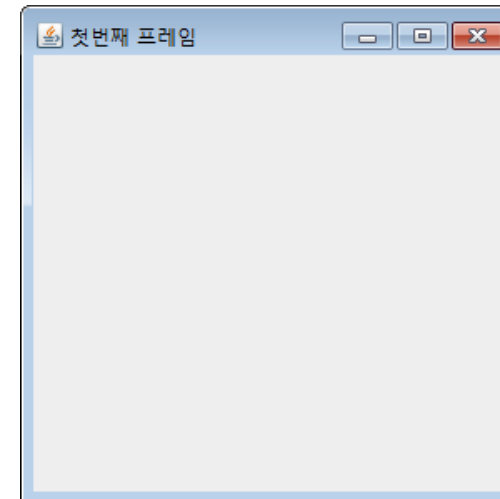
JScrollPane



JDialog



JApplet



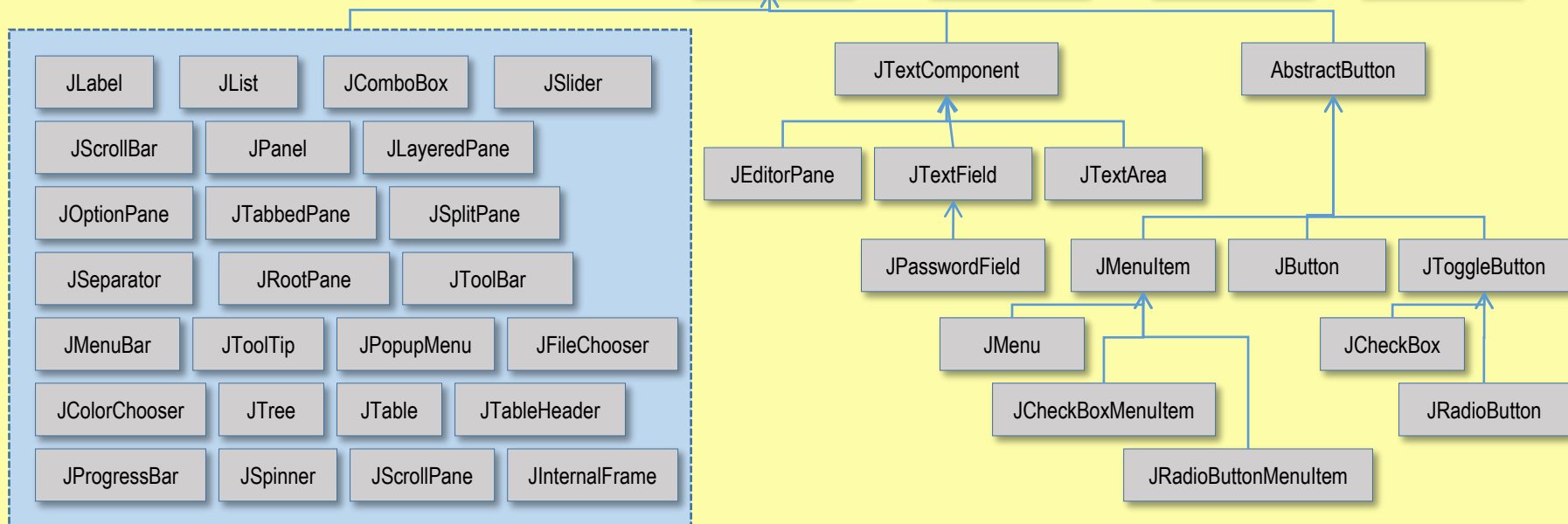
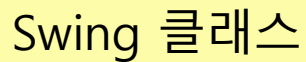
JFrame

- 하나의 클래스로
이걸 다 만드는 게
아니다
- JMenu
JToolBar

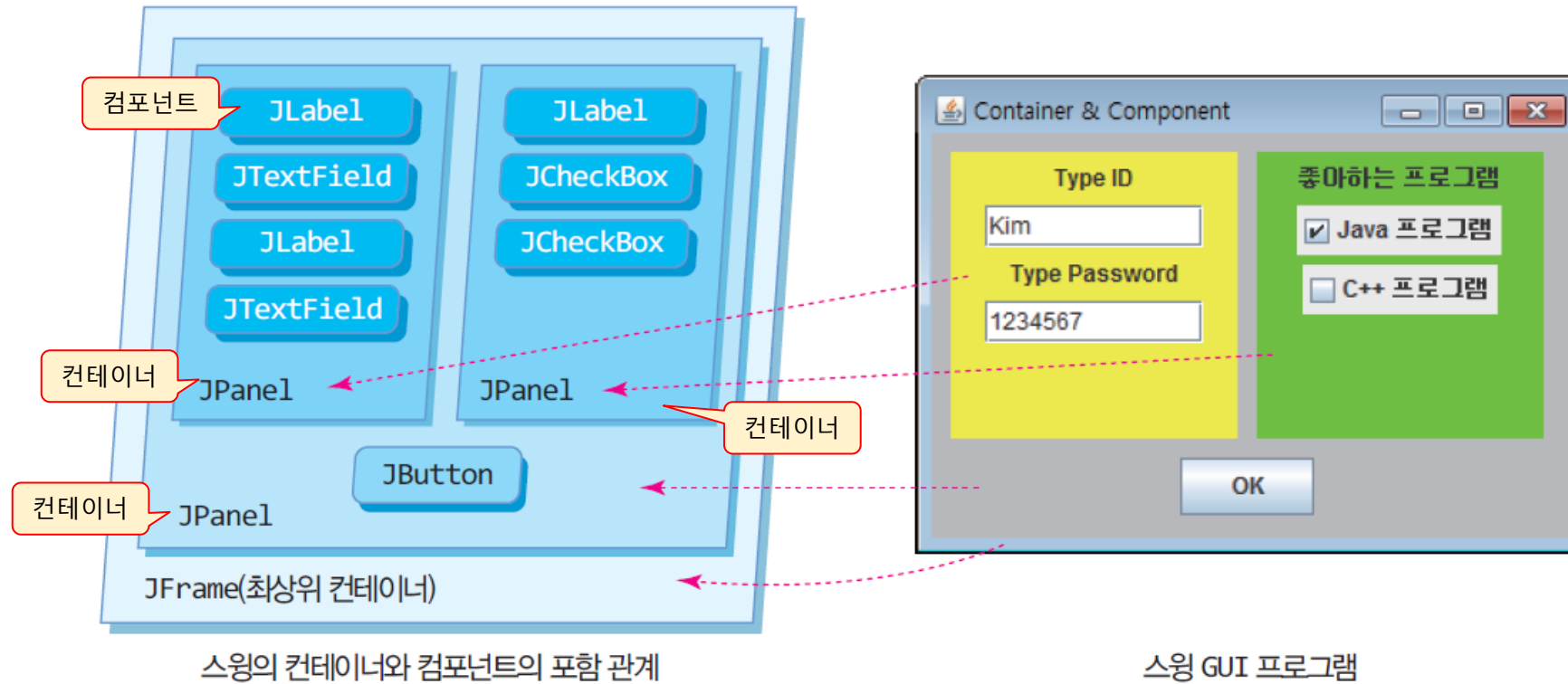
비트 만드는 클래스,
~ 만드는 클래스,
...으로 구성됨
전부 객체임



클래스의 문법



컨테이너와 컴포넌트의 포함 관계



최상위 컨테이너를 바닥에 깔고, 그 위에 컨테이너를 놓고,
다시 컴포넌트를 쌓아가는 방식, 즉 레고 블록을 쌓는 듯이 GUI 프로그램을 작성한다.

컨테이너 컴포넌트와 단순 컴포넌트 (보통 그냥 컨테이너와 컴포넌트라고도 부름)

■ 컨테이너 ➡ 컴포넌트를 넣을 수 있는 상자 같은 것 화면

- 다른 컴포넌트를 포함할 수 있는 GUI 컴포넌트
 - java.awt.Container를 상속받음
- 다른 컨테이너에 포함될 수 있음
 - AWT 컨테이너 : Panel, Frame, Applet, Dialog, Window
 - Swing 컨테이너 : JPanel JFrame, JApplet, JDialog, Jwindow (AWT 컴포넌트 이름 앞에 J자를 덧붙임)

- 최상위 컨테이너 ➡ 보통 다른 컨테이너 안에 포함될 수 없는 컨테이너를 의미

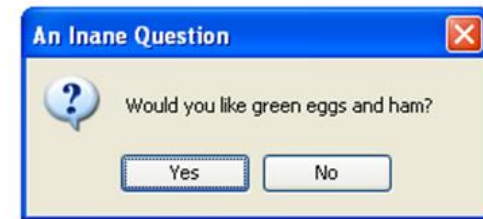
- 다른 컨테이너에 포함되지 않고도 화면에 출력되며 독립적으로 존재 가능한 컨테이너
 - 스스로 화면에 자신을 출력하는 컨테이너 : JFrame, JDialog, Japplet, 이 외에는 모두 다른 컨테이너에 부착해야 함(JPanel, JScrollPane 등)

■ 컴포넌트 버튼

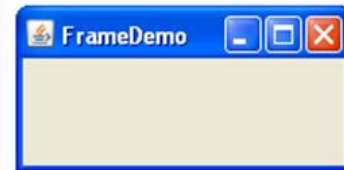
- 컨테이너에 포함되어야 화면에 출력될 수 있는 GUI 객체
- 다른 컴포넌트를 포함할 수 없는 순수 컴포넌트
- JButton, JLabel, Jcheckbox, Jlist, JTestField, Jscrollkbar 등



JApplet



JDialog



JFrame

스윙 GUI 프로그램 만들기

■ 스윙 프로그램 작성에 필요한 import문

- `import java.awt.*;` `//` 폰트 등 그래픽 처리를 위한 클래스들의 경로명
- `Import java.awt.event.*;` `//` 이벤트 처리에 필요한 기본 클래스들의 경로명
- `Import javax.swing.*;` `//` 스윙 컴포넌트 클래스들의 경로명
- `Import javax.swing.event.*;` `//` 스윙 이벤트 처리에 필요한 클래스들의 경로명

스윙 GUI 프로그램 만들기

- GUI 프로그램을 만드는 절차

- ☞ 먼저 컨테이너를 만들고, 그 안에 자신이 필요한 컴포넌트를 넣어서 작성

1. 컨테이너 생성

- 제일 먼저 최상위 컨테이너(JFrame)를 하나 생성

2. 컴포넌트 생성

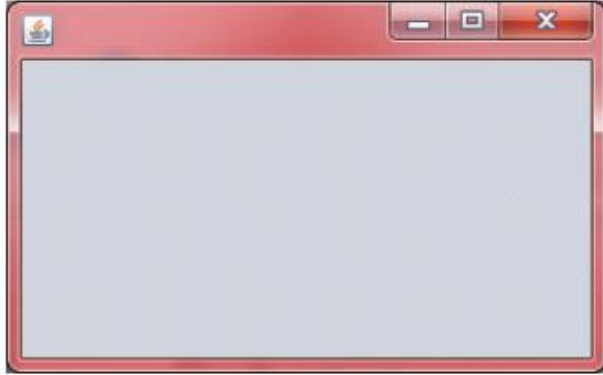
- 버튼 등 컨테이너에 넣어줄 컴포넌트 생성

3. 생성된 컴포넌트를 컨테이너에 추가(배치)

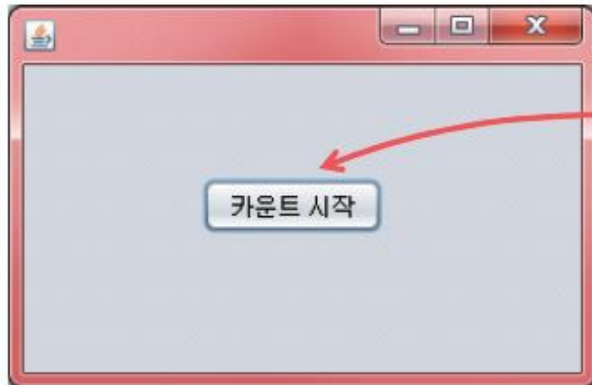
- 원하는 만큼 컴포넌트를 컨테이너에 추가

JDK 환경 변수 설정

컨테이너 생성



컴포넌트 생성 및 추가



Label	Button	Toggle Button
Check Box	Radio Button	Button Group
Combo Box	List	Text Field
Text Area	Scroll Bar	Slider
Progress Bar	Formatted Field	Password Field
Spinner	Separator	Text Pane
Editor Pane	Tree	Table

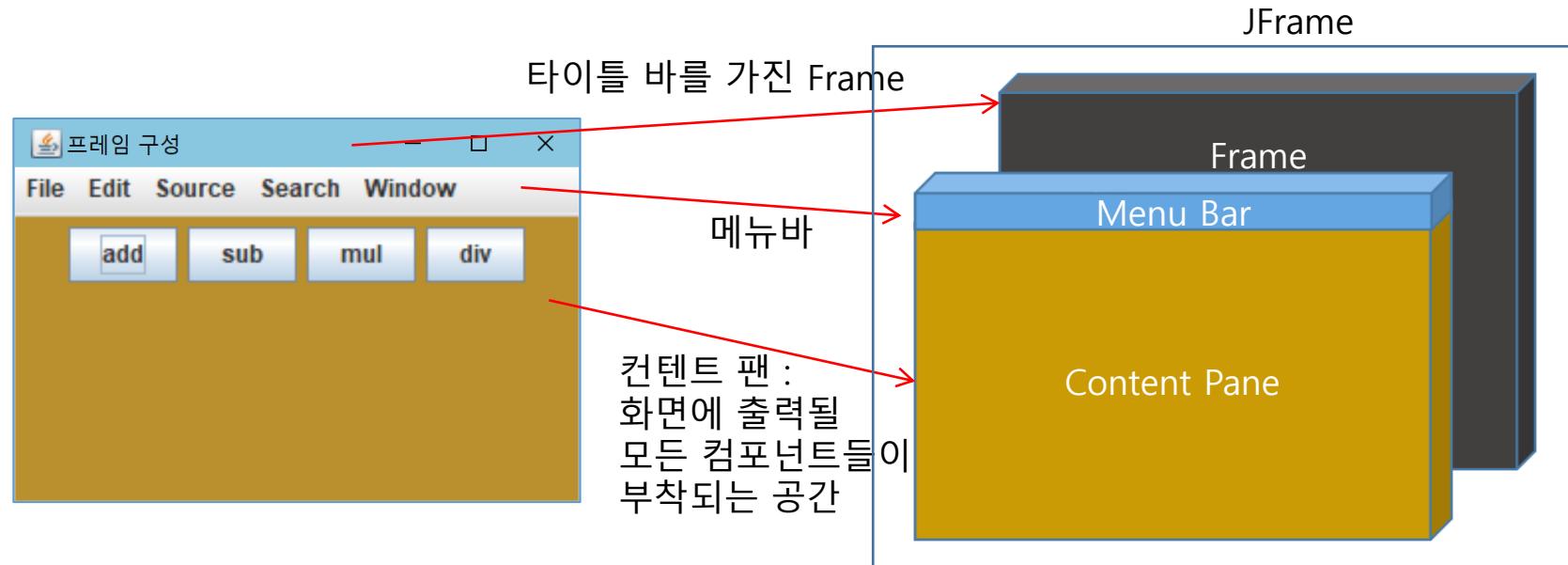
1. 프레임 만들기(JFrame 클래스 활용)

• 스윙 프레임 : 모든 스윙 컴포넌트를 담는 최상위 컨테이너

- JFrame 클래스 활용
- 컴포넌트들은 화면에 보이려면 스윙 프레임에 부착되어야 함
 - 프레임을 닫으면 프레임에 부착된 모든 컴포넌트가 보이지 않게 됨

• 스윙 프레임(JFrame) 기본 구성

- 프레임 - 스윙 프로그램의 기본 틀
- 메뉴바 - 메뉴들이 부착되는 공간
- 콘텐츠팬
 - 컴포넌트들이 부착되는 공간



1. 프레임 만들기(JFrame 클래스 활용)

방법 1. main() 안에서 JFrame 객체를 직접 생성

```
import javax.swing.*;  
  
public class GuiTest {  
    public static void main(String[] args) {  
  
        JFrame f = new JFrame("Frame Test");  
        f.setTitle("MyFrame");  
        f.setSize(300, 200);  
        f.setLocation(400, 200);  
        f.setVisible(true);  
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    }  
}
```

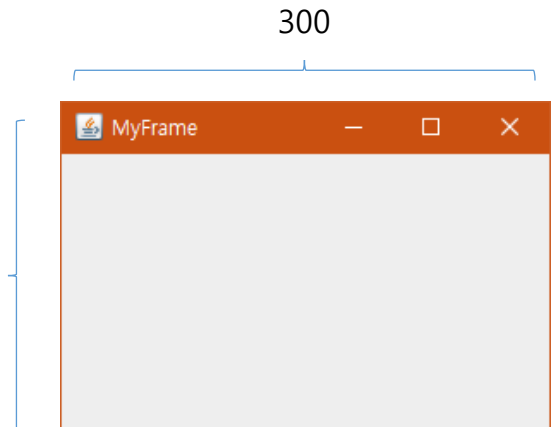
화면

프레임이 화면에 출력되도록 지정

17p

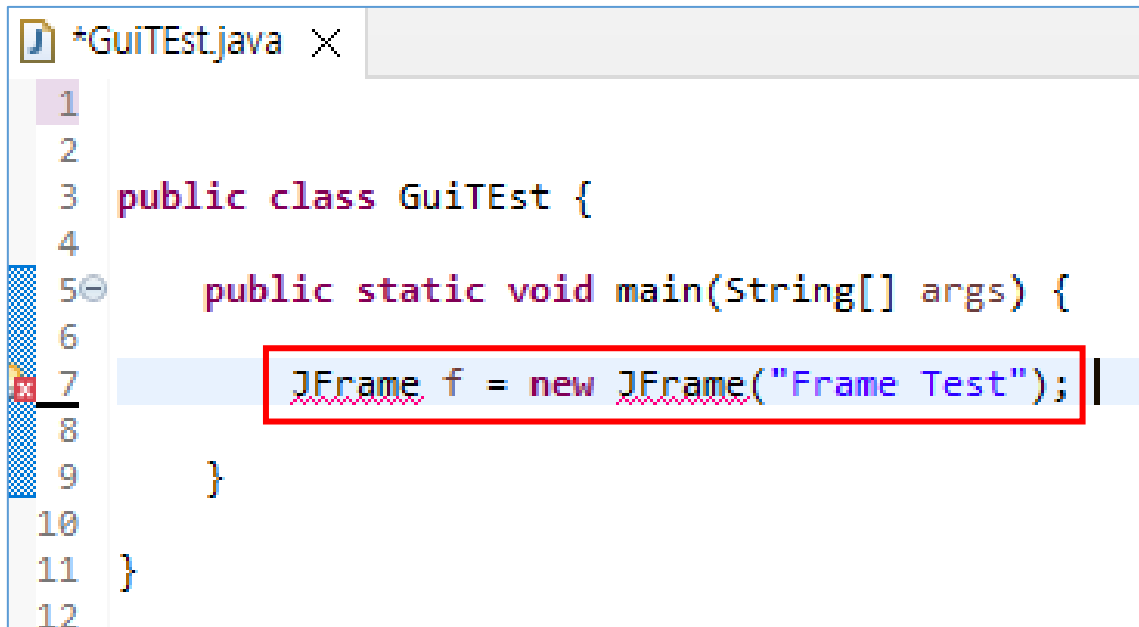
TSLVD

200



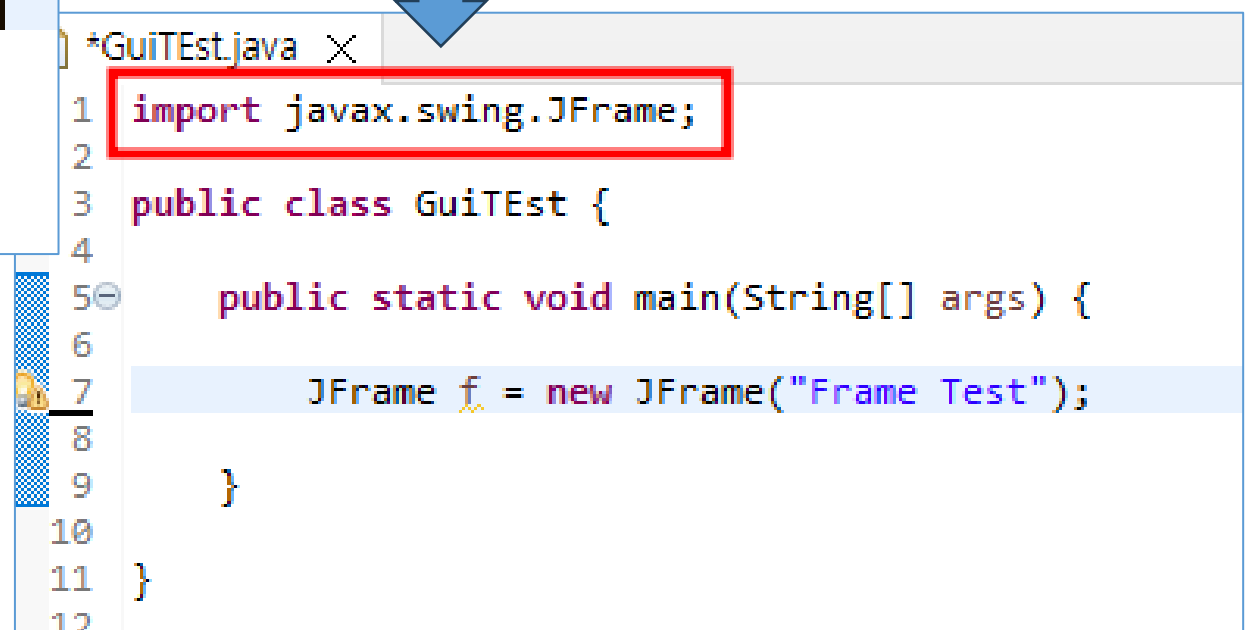
[참고] 이클립스에서 패키지 자동 импорт

- 단축키 : ctrl + shift + o



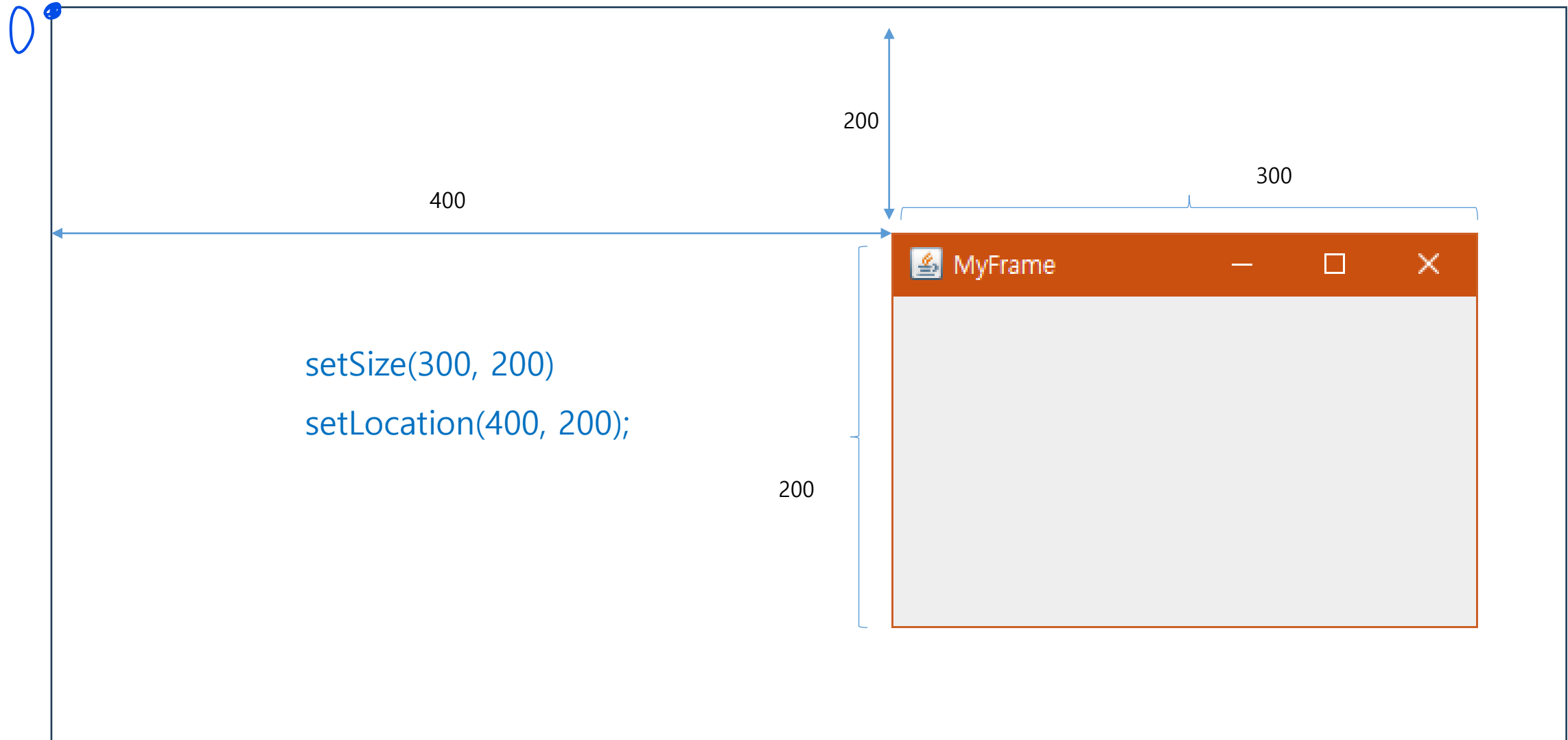
```
1
2
3 public class GuiTest {
4
5     public static void main(String[] args) {
6
7         JFrame f = new JFrame("Frame Test");
8
9     }
10
11 }
12
```

ctrl + shift + o



```
1 import javax.swing.JFrame;
2
3 public class GuiTest {
4
5     public static void main(String[] args) {
6
7         JFrame f = new JFrame("Frame Test");
8
9     }
10
11 }
12
```

setSize(), setLocation() 메소드



setBounds() 메소드

- 앞의 경우와 동일한 결과

setBounds(400, 200, 300, 200);

```
*GuiTest.java ×
1  import javax.swing.JFrame;
2
3  public class GuiTest {
4
5      public static void main(String[] args) {
6
7          JFrame f = new JFrame("Frame Test");
8          f.setTitle("MyFrame");
9          // f.setSize(300, 200);
10         // f.setLocation(400, 200);
11         f.setBounds(400, 200, 300, 200);
12         f.setVisible(true);
13         f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
14
15     }
16
17
18 }
```


스윙 응용프로그램의 종료

■ 응용프로그램 내에서 스스로 종료하는 방법

```
System.exit(0);
```

- 언제 어디서나 무조건 종료

■ 프레임의 오른쪽 상단의 종료버튼(X)이 클릭되면 어떤 일이 일어나는가?

- 프레임 종료, 프레임 윈도우를 닫음
 - 프레임이 화면에서 보이지 않게 됨
- 프레임이 보이지 않게 되지만 응용프로그램이 종료한 것 아님
 - 키보드나 마우스 입력을 받지 못함
 - 다시 **setVisible(true)**를 호출하면, 보이게 되고 이전 처럼 작동함

응용프로그램 종료 X



■ 프레임 종료 버튼이 클릭될 때, 프레임과 함께 프로그램을 종료시키는 방법

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

1. 프레임 만들기(JFrame 클래스 활용)

방법 2. JFrame 객체를 상속받는 MyFrame 객체 생성

```
import javax.swing.JFrame;

class GuiTest {
    public static void main(String [] args) {
        MyFrame f = new MyFrame();
    }
}

class MyFrame extends JFrame {
    public MyFrame() { 생성자
        setTitle("첫번째 프레임");
        setSize(300, 300);
        setLocation(400, 200);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

1. 프레임 만들기(JFrame 클래스 활용)

방법 3. MyFrame 클래스에 main() 메소드 추가

(앞 장의 2개 클래스를 1개로 합침)

※ main() 메소드는 어떤 클래스 안에서도 선언 가능, static으로 선언되므로 객체를 생성하지 않아도 얼마든지 외부에서 호출(실행) 가능, ※ 새로운 클래스를 작성했을 때도 테스트해 보고 싶으면 클래스 안에 main()을 만들면 됨

```
import javax.swing.*;

class MyFrame extends JFrame {
    public MyFrame() {
        setTitle("첫번째 프레임");
        setSize(300, 300);
        setLocation(400, 200);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public static void main(String [] args) {
        MyFrame mf = new MyFrame();
    }
}
```

2~3. 컴포넌트 생성 후 프레임에 추가

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

class GuiTest {
    public static void main(String [] args) {
        JFrame mf = new JFrame();
        mf.setTitle("첫번째 프레임");

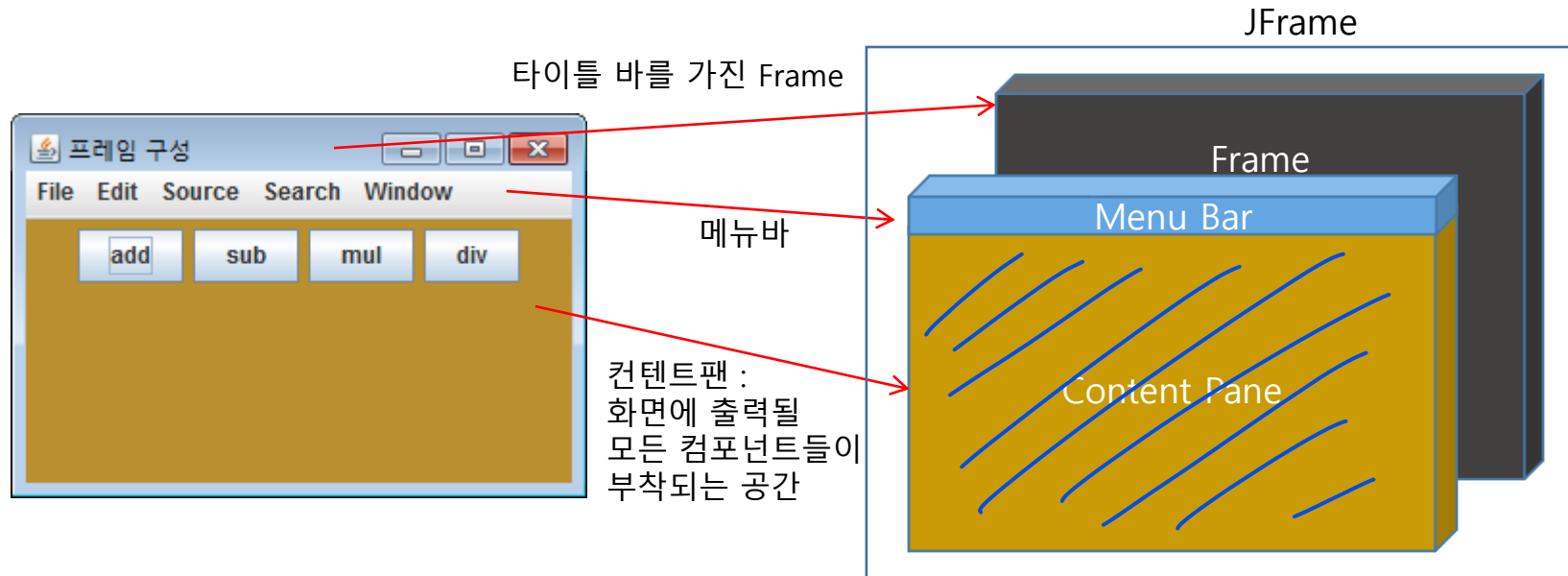
        JButton button = new JButton("버튼1"); 버튼 만들기
        mf.add(button); 버튼 추가

        mf.setSize(300, 300);
        mf.setVisible(true);
        mf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

컨텐츠팬

- 스윙 프레임 : 모든 스윙 컴포넌트를 담는 최상위 컨테이너
 - JFrame을 상속받아 구현
 - 컴포넌트들은 화면에 보이려면 스윙 프레임에 부착되어야 함
 - 프레임을 닫으면 프레임에 부착된 모든 컴포넌트가 보이지 않게 됨
- 스윙 프레임(JFrame) 기본 구성
 - 프레임 - 스윙 프로그램의 기본 틀
 - 메뉴바 - 메뉴들이 부착되는 공간
 - 컨텐츠팬 - GUI 컴포넌트들이 부착되는 공간

자
생
성



컨텐츠팬

- 타이틀 달기
 - `super()`나 `setTitle()` 이용

```
MyFrame() { // 생성자
    super("타이틀문자열");
}
```

```
MyFrame() { // 생성자
    setTitle("타이틀문자열");
}
```

- 컨텐츠팬에 컴포넌트 달기
 - 컨텐츠팬이란?
 - 스윙 컴포넌트들이 부착되는 공간
 - 컨텐츠팬 알아내기
 - 스윙 프레임에 붙은 디폴트 컨텐츠팬 알아내기

```
public class MyFrame extends JFrame {
    MyFrame() {
        ...
        // 프레임의 컨텐츠팬을 알아낸다.
        Container contentPane = getContentPane();
    }
    ...
}
```

```
// 버튼 컴포넌트 생성
JButton button = new JButton("Click");
contentPane.add(button); // 컨텐츠팬에 버튼 부착
```

- 컨텐츠팬에 컴포넌트 붙이기
- 컨텐츠팬 변경

```
class MyPanel extends JPanel {
    ... // JPanel을 상속받은 패널을 구현한다.
}
// frame의 컨텐츠팬을 MyPanel 객체로 변경
frame.setContentPane(new MyPanel());
```

컨텐츠팬

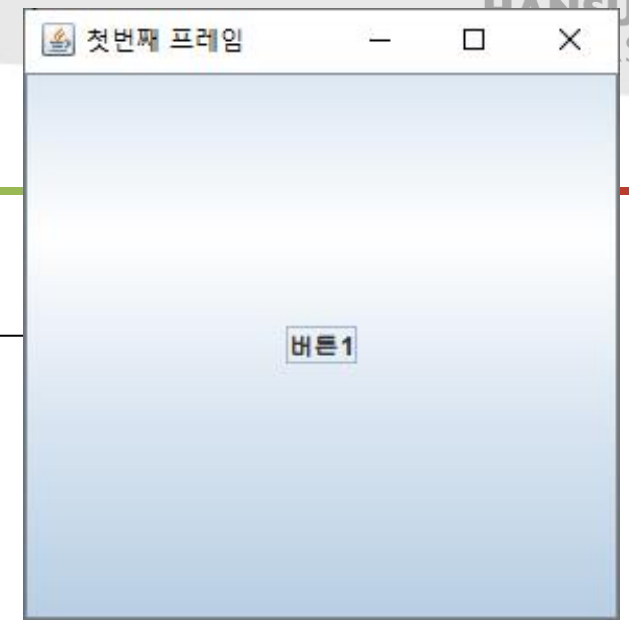
방법 1. main() 안에서 JFrame 객체를 직접 생성

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

class GuiTest {
    public static void main(String [] args) {
        JFrame mf = new JFrame(); // 프레임 만들기
        mf.setTitle("첫번째 프레임");

        JButton button = new JButton("버튼1"); // 컴포넌트 만들기
                                                // (버튼)
        Container <cont> = mf.getContentPane(); // 컨텐츠팬 알아내기
        <cont>.add(button); //컨텐츠팬에 버튼 부착

        mf.setSize(300, 300); // 화면에 출력
        mf.setVisible(true);
        mf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```



내부에서
일어나는 일이고
실제로 이런 식으로
코딩 할 필요없다

컨텐츠팬



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

class GuiTest {
    public static void main(String [] args) {
        MyFrame mf = new MyFrame();
    }
}

class MyFrame extends JFrame {
    public MyFrame() {
        setTitle("첫번째 프레임");

        JButton button = new JButton( "버튼1"); // 컴포넌트 만들기
                                                // (버튼)
        Container cont = getContentPane(); // 컨텐츠팬 알아내기
        cont.add(button); //컨텐츠팬에 버튼 부착

        setSize(300, 300); // 화면에 출력
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```


방법 3. MyFrame 클래스에 main() 메소드 추가 (앞 장의 2개 클래스를 1개로 합침)

컨텐츠팬

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

class MyFrame extends JFrame {
    public MyFrame() {
        setTitle("첫번째 프레임");

        JButton button = new JButton("버튼1"); // 컴포넌트 만들기
                                                // (버튼)
        Container cont = getContentPane(); // 컨텐츠팬 알아내기
        cont.add(button); //컨텐츠팬에 버튼 부착

        setSize(300, 300); // 화면에 출력
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
    public static void main(String [] args) {
        MyFrame mf = new MyFrame();
    }
}
```

컨텐츠팬에 대한 JDK 1.5 이후의 추가 사항

- JDK 1.5 이전

- 프레임의 컨텐츠팬을 알아내어 반드시 컨텐츠팬에 컴포넌트 부착

```
Container c = frame.getContentPane();  
c.add(new JButton("Click")); // 컨텐츠팬에 직접 컴포넌트 부착
```

- JDK 1.5 이후 추가된 사항

- 프레임에 컴포넌트를 부착하면 프레임이 대신 컨텐츠팬에 부착

```
frame.add(new JButton("Click"));  
// 프레임이 버튼 컴포넌트를 컨텐츠팬에 대신 부착
```

이전 이렇게 해라

정리

방법 1. main() 안에서 JFrame 객체를 직접 생성

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

class GuiTest {
    public static void main(String [] args) {
        JFrame mf = new JFrame();
        mf.setTitle("첫번째 프레임");

        JButton button = new JButton("버튼1");

        //Container cont = mf.getContentPane(); // 컨테트팬 알아내기
        //cont.add(button);
        mf.add(button); // 바로 프레임에 부착 가능

        mf.setSize(300, 300);
        mf.setVisible(true);
        mf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

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

```
class GuiTest {  
    public static void main(String [] args) {  
        MyFrame mf = new MyFrame();  
    }  
}
```

```
class MyFrame extends JFrame {  
    MyFrame() {  
        setTitle("첫번째 프레임");  
  
        JButton button = new JButton("버튼1");    // 컴포넌트 만들기  
                                                    // (버튼)  
        //Container cont = getContentPane(); // 컨테인트팬 알아내기  
        //cont.add(button); //컨테인트팬에 버튼 부착  
        add(button);  
  
        setSize(300, 300); // 화면에 출력  
        setVisible(true);  
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    }  
}
```

정리

(앞 장의 2개 클래스를 1개로 합침)

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

class MyFrame extends JFrame {
    public MyFrame() {
        setTitle("첫번째 프레임");

        JButton button = new JButton("버튼1");    // 컴포넌트 만들기
                                                // (버튼)
        //Container cont = getContentPane(); // 컨테이너판 알아내기
        //cont.add(button); //컨테이너판에 버튼 부착
        add(button);

        setSize(300, 300); // 화면에 출력
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public static void main(String [] args) {
        MyFrame mf = new MyFrame();
    }
}
```

참고 문헌

- 본 강의자료는 주로 강의계획서에 명시된 황기태 교수님의 '명품 JAVA PROGRAMMING(생능출판사)'을 기준으로 네트워크프로그래밍 과목에 맞게 내용을 추가해서 만든 교재임을 알려드립니다.
- 단, 일부 그림 등은 다른 참고 문헌(Power JAVA, 천인국, 인피니티박스)를 참고한 부분도 있습니다.

수업계획	
교과목 개요	네트워크프로그래밍 과목에서는 네트워크와 TCP/IP 기본 이론을 배우고 다양한 기기 간 데이터 교환을 할 수 있도록 소켓프로그래밍을 강의하며 환경하에서 데이터 교환을 위한 응용프로그램에서 표준으로 사용되는 프로그래밍 기법이다. 본 과목에서는 Java 프로그래밍 언어를 사용하도록 강의를 한다.
수업목표	본 강의의 목표는 4학년 1학기 캡스톤디자인에 필요한 네트워크 응용 기술을 미리 습득시키는 것이다. TCP/IP 소켓 통신을 이용한 네트워크 서버 환경을 이용한 시스템 개발 기술을 습득하게 된다. 학생들은 습득한 네트워크프로그래밍 기술을 활용하여 PC, 핸드폰, 태블릿간의 통신에
선수과목	데이터 통신, 객체지향언어 2(Java)
주교재	명품 JAVA PROGRAMMING(황기태 / 생능출판사)
부교재/참고문헌	"TCP/IP 소켓프로그래밍" 또는 "네트워크프로그래밍"으로 검색하면 많은 책들이 있습니다. 필요하면 스스로 참고서적을 준비할 수 있습니다.



 **T h a n k y o u**

TECHNOLOGY

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Velit ex
plicabo ipsum, labore sed tempora ratione asperiores des
quenerat bore sed tempora rati jgert one bore sed tem!