응용 애플리케이션 개발을 위한 자바 프로그래밍

AWT 패키지 활용하기

한국기술교육대학교
온라인평생교육원

학습내용

- AWT 이해하기
- AWT 컴포넌트 활용하기

학습목표

- AWT의 개념과 구조를 이해하고, AWT 패키지를 활용하여 GUI를 구성할 수 있다.
- AWT 컴포넌트에 관한 생성자 메소드와 메소드를 확인하고, 컴포넌트 별 프로그램을 작성할 수 있다.

AWT 이해하기

1 AWT(Abstract Window Toolkit)의 개요

⟨1⟩ AWT란?

1 GUI를 구현하기 위한 컴포넌트(Component)와 컨테이너(Container)들을 가진 패키지

컴포넌트	GUI 기능을 가지는 클래스
컨테이너	컴포넌트들을 배치할 수 있는 컴포넌트

1 AWT(Abstract Window Toolkit)의 개요

⟨1⟩ AWT란?

2 운영체제에 종속적이지 않음

운영체제에 따라 화면의 모양은 다르게 나타남

3 패키지 : java.awt

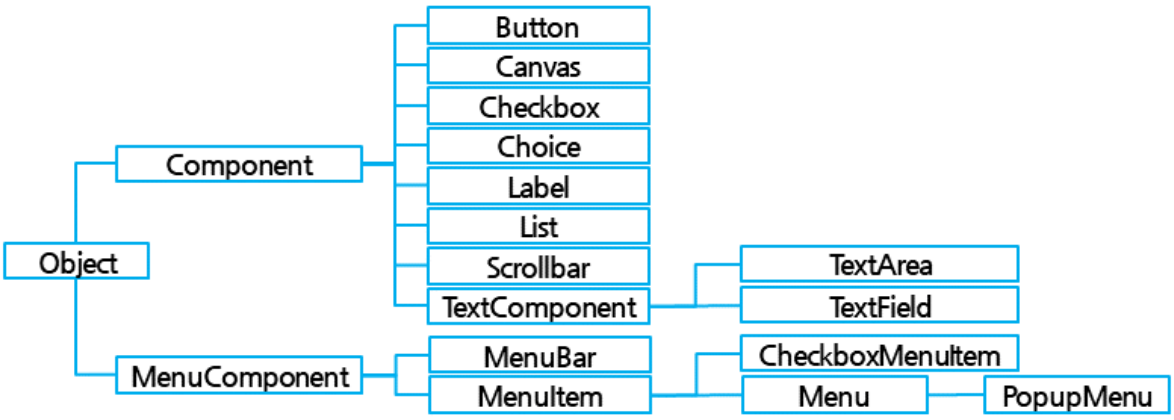
4 최상위 클래스 : Component

모든 컴포넌트는 Component 클래스의 상속

AWT 이해하기

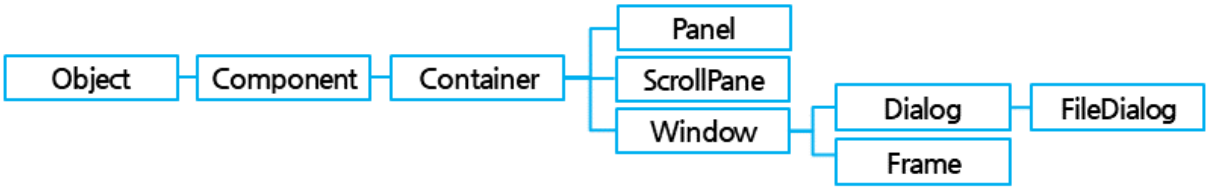
1 AWT(Abstract Window Toolkit)의 개요

〈2〉 AWT 컴포넌트의 구조



1 AWT(Abstract Window Toolkit)의 개요

〈3〉 AWT 컨테이너의 구조



AWT 이해하기

2 AWT 컴포넌트를 이용한 GUI 프로그램

① 프로그램 순서

1 java.awt 패키지 import

2 컨테이너 객체 생성

컨테이너를 단독으로 생성

컨테이너를 상속하여 생성

3 컨테이너 레이아웃 설정

레이아웃 설정 : 컴포넌트를 컨테이너에 배치하는 방법

2 AWT 컴포넌트를 이용한 GUI 프로그램

① 프로그램 순서

4 컴포넌트 생성

5 컴포넌트를 컨테이너에 추가

6 컨테이너와 컴포넌트에 이벤트 설정

이벤트 설정 : 프로그램과 사용자간의 인터페이스 방법 설정

7 실행

AWT 이해하기

2 AWT 컴포넌트를 이용한 GUI 프로그램

⟨2⟩ 컨테이너를 단독으로 생성하는 예제

```
import java.awt.*;

public class AwtFirst {
    public static void main(String[] args) {
        Frame f = new Frame("첫번째 AWT 프로그램");
        f.setLayout(new FlowLayout());

        Button b1 = new Button("확인");
        Button b2 = new Button("취소");

        f.add(b1);
        f.add(b2);

        f.setSize(300,200);
        f.setVisible(true);
    }
}
```

2 AWT 컴포넌트를 이용한 GUI 프로그램

⟨2⟩ 컨테이너를 단독으로 생성하는 예제

```
import java.awt.*;
public class AwtFirst {
    public static void main(String[] args) {
        Frame f = new Frame("첫번째 AWT 프로그램");
        f.setLayout(new FlowLayout());

        Button b1 = new Button("확인");
        Button b2 = new Button("취소");

        f.add(b1);
        f.add(b2);

        f.setSize(300,200);
        f.setVisible(true);
    }
}
```

← AWT 패키지를 import

AWT 이해하기

2 AWT 컴포넌트를 이용한 GUI 프로그램

⟨2⟩ 컨테이너를 단독으로 생성하는 예제

```
import java.awt.*;

public class AwtFirst {
    public static void main(String[] args) {
        Frame f = new Frame("첫번째 AWT 프로그램");
        f.setLayout(new FlowLayout());

        Button b1 = new Button("확인");
        Button b2 = new Button("취소");

        f.add(b1);
        f.add(b2);

        f.setSize(300,200);
        f.setVisible(true);
    }
}
```

컨테이너 중에
Frame 컴포넌트 단독 생성

2 AWT 컴포넌트를 이용한 GUI 프로그램

⟨2⟩ 컨테이너를 단독으로 생성하는 예제

```
import java.awt.*;

public class AwtFirst {
    public static void main(String[] args) {
        Frame f = new Frame("첫번째 AWT 프로그램");
        f.setLayout(new FlowLayout());

        Button b1 = new Button("확인");
        Button b2 = new Button("취소");

        f.add(b1);
        f.add(b2);

        f.setSize(300,200);
        f.setVisible(true);
    }
}
```

Frame 객체 f 에 레이아웃 설정

AWT 이해하기

2 AWT 컴포넌트를 이용한 GUI 프로그램

⟨2⟩ 컨테이너를 단독으로 생성하는 예제

```
import java.awt.*;

public class AwtFirst {
    public static void main(String[] args) {
        Frame f = new Frame("첫번째 AWT 프로그램");
        f.setLayout(new FlowLayout());

        Button b1 = new Button("확인");
        Button b2 = new Button("취소");

        f.add(b1);
        f.add(b2);

        f.setSize(300,200);
        f.setVisible(true);
    }
}
```

← Button 컴포넌트 2개 생성

2 AWT 컴포넌트를 이용한 GUI 프로그램

⟨2⟩ 컨테이너를 단독으로 생성하는 예제

```
import java.awt.*;

public class AwtFirst {
    public static void main(String[] args) {
        Frame f = new Frame("첫번째 AWT 프로그램");
        f.setLayout(new FlowLayout());

        Button b1 = new Button("확인");
        Button b2 = new Button("취소");

        f.add(b1);
        f.add(b2);

        f.setSize(300,200);
        f.setVisible(true);
    }
}
```

← Button 컴포넌트 2개를 컨테이너 객체 f 에 추가

AWT 이해하기

2 AWT 컴포넌트를 이용한 GUI 프로그램

⟨2⟩ 컨테이너를 단독으로 생성하는 예제

```
import java.awt.*;

public class AwtFirst {
    public static void main(String[] args) {
        Frame f = new Frame("첫번째 AWT 프로그램");
        f.setLayout(new FlowLayout());

        Button b1 = new Button("확인");
        Button b2 = new Button("취소");

        f.add(b1);
        f.add(b2);

        f.setSize(300,200);
        f.setVisible(true);
    }
}
```

객체 f 의 크기를 설정하고 화면에 표시

2 AWT 컴포넌트를 이용한 GUI 프로그램

⟨2⟩ 컨테이너를 단독으로 생성하는 예제

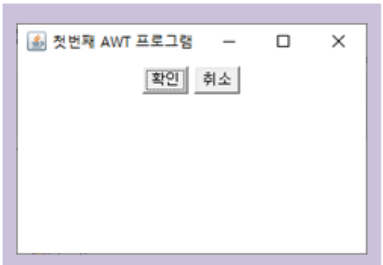
```
import java.awt.*;

public class AwtFirst {
    public static void main(String[] args) {
        Frame f = new Frame("첫번째 AWT 프로그램");
        f.setLayout(new FlowLayout());

        Button b1 = new Button("확인");
        Button b2 = new Button("취소");

        f.add(b1);
        f.add(b2);

        f.setSize(300,200);
        f.setVisible(true);
    }
}
```

실행결과

AWT 이해하기

2 AWT 컴포넌트를 이용한 GUI 프로그램

〈3〉 컨테이너를 상속하여 생성하는 예제

```
import java.awt.*;
public class AwtSecond extends Frame {
     AwtSecond() {
         super("두번째 AWT 프로그램");
         setLayout(new FlowLayout());
         Button b1 = new Button("확인");
         Button b2 = new Button("취소");
         add(b1);
         add(b2);
     }
     public static void main(String[] args) {
          AwtSecond as = new AwtSecond();
          as.setSize(300,200);
          as.setVisible(true);
     }
}
```

2 AWT 컴포넌트를 이용한 GUI 프로그램

〈3〉 컨테이너를 상속하여 생성하는 예제

```
import java.awt.*;
public class AwtSecond extends Frame {
     AwtSecond() {
         super("두번째 AWT 프로그램");
         setLayout(new FlowLayout());
         Button b1 = new Button("확인");
         Button b2 = new Button("취소");
         add(b1);
         add(b2);
     }
     public static void main(String[] args) {
          AwtSecond as = new AwtSecond();
          as.setSize(300,200);
          as.setVisible(true);
     }
}
```

Frame 컴포넌트를 상속

AWT 이해하기

2 AWT 컴포넌트를 이용한 GUI 프로그램

〈3〉 컨테이너를 상속하여 생성하는 예제

생성자 메소드

```
import java.awt.*;
public class AwtSecond extends Frame {
    AwtSecond() {
        super("두번째 AWT 프로그램");
        setLayout(new FlowLayout());
        Button b1 = new Button("확인");
        Button b2 = new Button("취소");
        add(b1);
        add(b2);
    }
    public static void main(String[] args) {
        AwtSecond as = new AwtSecond();
        as.setSize(300,200);
        as.setVisible(true);
    }
}
```

2 AWT 컴포넌트를 이용한 GUI 프로그램

〈3〉 컨테이너를 상속하여 생성하는 예제

Frame 컴포넌트의 생성자 메소드 호출

```
import java.awt.*;
public class AwtSecond extends Frame {
    AwtSecond() {
        super("두번째 AWT 프로그램");
        setLayout(new FlowLayout());
        Button b1 = new Button("확인");
        Button b2 = new Button("취소");
        add(b1);
        add(b2);
    }
    public static void main(String[] args) {
        AwtSecond as = new AwtSecond();
        as.setSize(300,200);
        as.setVisible(true);
    }
}
```

AWT 이해하기

2 AWT 컴포넌트를 이용한 GUI 프로그램

〈3〉 컨테이너를 상속하여 생성하는 예제

```
import java.awt.*;
public class AwtSecond extends Frame {
     AwtSecond() {
         super("두번째 AWT 프로그램");
         setLayout(new FlowLayout());
         Button b1 = new Button("확인");
         Button b2 = new Button("취소");
         add(b1);
         add(b2);
     }
     public static void main(String[] args) {
          AwtSecond as = new AwtSecond();
          as.setSize(300,200);
          as.setVisible(true);
     }
}
```

레이아웃 설정, 컴포넌트 생성, 컴포넌트 추가

2 AWT 컴포넌트를 이용한 GUI 프로그램

〈3〉 컨테이너를 상속하여 생성하는 예제

```
import java.awt.*;
public class AwtSecond extends Frame {
     AwtSecond() {
         super("두번째 AWT 프로그램");
         setLayout(new FlowLayout());
         Button b1 = new Button("확인");
         Button b2 = new Button("취소");
         add(b1);
         add(b2);
     }
     public static void main(String[] args) {
          AwtSecond as = new AwtSecond();
          as.setSize(300,200);
          as.setVisible(true);
     }
}
```

main 메소드

AWT 이해하기

2 AWT 컴포넌트를 이용한 GUI 프로그램

〈3〉 컨테이너를 상속하여 생성하는 예제

```
import java.awt.*;
public class AwtSecond extends Frame {
     AwtSecond() {
         super("두번째 AWT 프로그램");
         setLayout(new FlowLayout());
         Button b1 = new Button("확인");
         Button b2 = new Button("취소");
         add(b1);
         add(b2);
     }
     public static void main(String[] args) {
          AwtSecond as = new AwtSecond();
          as.setSize(300,200);
          as.setVisible(true);
     }
}
```

클래스의 객체를 생성

2 AWT 컴포넌트를 이용한 GUI 프로그램

〈3〉 컨테이너를 상속하여 생성하는 예제

```
import java.awt.*;
public class AwtSecond extends Frame {
     AwtSecond() {
         super("두번째 AWT 프로그램");
         setLayout(new FlowLayout());
         Button b1 = new Button("확인");
         Button b2 = new Button("취소");
         add(b1);
         add(b2);
     }
     public static void main(String[] args) {
          AwtSecond as = new AwtSecond();
          as.setSize(300,200);
          as.setVisible(true);
     }
}
```

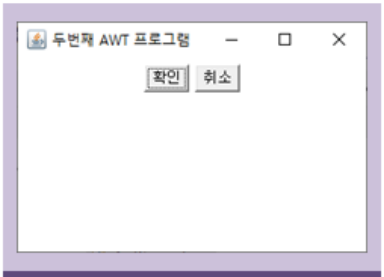
화면 크기 설정과 표시

AWT 이해하기

2 AWT 컴포넌트를 이용한 GUI 프로그램

〈3〉 컨테이너를 상속하여 생성하는 예제

```
import java.awt.*;
public class AwtSecond extends Frame {
     AwtSecond() {
         super("두번째 AWT 프로그램");
         setLayout(new FlowLayout());
         Button b1 = new Button("확인");
         Button b2 = new Button("취소");
         add(b1);
         add(b2);
     }
     public static void main(String[] args) {
          AwtSecond as = new AwtSecond();
          as.setSize(300,200);
          as.setVisible(true);
     }
}
```



실행결과

AWT 이해하기

AWT 패키지 활용하기 | AWT 패키지 이해하기

실습하기

AWT 패키지 활용하기 | AWT 패키지 이해하기

실습순서

1. AWT를 이용한 GUI 프로그램 실습
   1) 컨테이너를 단독으로 생성하는 예제 실습
   2) 컨테이너를 상속하여 생성하는 예제 실습

유의사항

- JDK와 이클립스를 설치한 후 실습이 가능함
- 본인이 원하는 작업 폴더를 미리 정해 놓은 다음 실습하기
- 작업 폴더는 C드라이브에 지정하기 보다는 D드라이브나 외장하드디스크를 활용하는 것을 추천함

※ 제공되는 실습 코드를 다운받아 실습해보시기 바랍니다.

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈1〉 Component 클래스의 주요 메소드

1 색상관련 메소드

메소드	설 명
Color getBackground( )	현재 설정된 배경색을 Color 객체로 반환
void setBackground(Color c)	배경색을 Color 객체로 지정
Color getForeground( )	현재 설정된 전경색을 Color 객체로 반환
void setForeground(Color c)	전경색을 Color 객체로 지정

1 Component 클래스 이해하기

〈1〉 Component 클래스 이해하기

1 색상관련 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setForeground(new Color(100,200,200));
b1.setBackground(new Color(200,100,200));

Color b1_f = b1.getForeground();
Color b1_b = b1.getBackground();

b2.setForeground(b1_b);
b2.setBackground(b1_f);
```

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨1⟩ Component 클래스 이해하기

1 색상관련 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setForeground(new Color(100,200,200));
b1.setBackground(new Color(200,100,200));

Color b1_f = b1.getForeground();
Color b1_b = b1.getBackground();

b2.setForeground(b1_b);
b2.setBackground(b1_f);
```

Button 객체 생성

1 Component 클래스 이해하기

⟨1⟩ Component 클래스 이해하기

1 색상관련 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setForeground(new Color(100,200,200));
b1.setBackground(new Color(200,100,200));

Color b1_f = b1.getForeground();
Color b1_b = b1.getBackground();

b2.setForeground(b1_b);
b2.setBackground(b1_f);
```

b1 객체의 전경색 지정

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

① Component 클래스 이해하기

1 색상관련 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setForeground(new Color(100,200,200));
b1.setBackground(new Color(200,100,200));

Color b1_f = b1.getForeground();
Color b1_b = b1.getBackground();

b2.setForeground(b1_b);
b2.setBackground(b1_f);
```

b1 객체의 배경색 지정

1 Component 클래스 이해하기

① Component 클래스 이해하기

1 색상관련 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setForeground(new Color(100,200,200));
b1.setBackground(new Color(200,100,200));

Color b1_f = b1.getForeground();
Color b1_b = b1.getBackground();

b2.setForeground(b1_b);
b2.setBackground(b1_f);
```

b1 객체의 전경색 가져오기

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

① Component 클래스 이해하기

1 색상관련 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setForeground(new Color(100,200,200));
b1.setBackground(new Color(200,100,200));

Color b1_f = b1.getForeground();
Color b1_b = b1.getBackground();

b2.setForeground(b1_b);
b2.setBackground(b1_f);
```

b1 객체의 배경색 가져오기

1 Component 클래스 이해하기

① Component 클래스 이해하기

1 색상관련 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setForeground(new Color(100,200,200));
b1.setBackground(new Color(200,100,200));

Color b1_f = b1.getForeground();
Color b1_b = b1.getBackground();

b2.setForeground(b1_b);
b2.setBackground(b1_f);
```

b2 객체의 전경색에 b1 객체의 배경색 지정하기

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

① Component 클래스 이해하기

1 색상관련 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setForeground(new Color(100,200,200));
b1.setBackground(new Color(200,100,200));

Color b1_f = b1.getForeground();
Color b1_b = b1.getBackground();

b2.setForeground(b1_b);
b2.setBackground(b1_f);
```

← b2 객체의 배경색에 b1 객체의 전경색 지정하기

1 Component 클래스 이해하기

① Component 클래스 이해하기

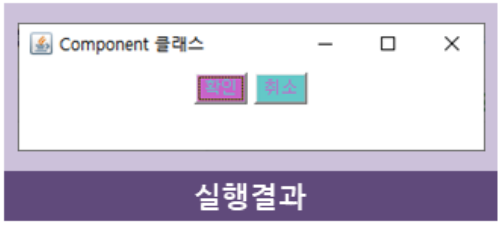
1 색상관련 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setForeground(new Color(100,200,200));
b1.setBackground(new Color(200,100,200));

Color b1_f = b1.getForeground();
Color b1_b = b1.getBackground();

b2.setForeground(b1_b);
b2.setBackground(b1_f);
```

실행결과

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

1 Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

메소드	설 명
Dimension getSize()	컴포넌트의 크기를 Dimension 객체로 반환
void setSize(Dimension d)	컴포넌트의 크기를 Dimension 객체로 설정
void setSize(int w, int h)	컴포넌트의 크기를 폭과 넓이로 설정

1 Component 클래스 이해하기

1 Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

메소드	설 명
Point getLocation()	컴포넌트의 위치를 Point 객체로 반환
void setLocation(Dimension d)	컴포넌트의 위치를 Point 객체로 설정
void setLocation(int x, int y)	컴포넌트의 위치를 x, y 좌표값으로 설정

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

1 Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

```
setLayout(null);

Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setLocation(50, 50);
b1.setSize(50, 50);

Point b1_p = b1.getLocation();
Dimension b1_d = b1.getSize();

b2.setLocation(b1_p.x+50, b1_p.y+50);
b2.setSize(b1_d.width+25, b1_d.height+25);
```

1 Component 클래스 이해하기

1 Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

```
setLayout(null);
```

← 컴포넌트의 크기와 위치를 지정하기 위한 레이아웃 설정


```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setLocation(50, 50);
b1.setSize(50, 50);

Point b1_p = b1.getLocation();
Dimension b1_d = b1.getSize();

b2.setLocation(b1_p.x+50, b1_p.y+50);
b2.setSize(b1_d.width+25, b1_d.height+25);
```

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈1〉 Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

```
setLayout(null);

Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setLocation(50, 50);
b1.setSize(50, 50);

Point b1_p = b1.getLocation();
Dimension b1_d = b1.getSize();

b2.setLocation(b1_p.x+50, b1_p.y+50);
b2.setSize(b1_d.width+25, b1_d.height+25);
```

b1 객체의 위치를 지정

1 Component 클래스 이해하기

〈1〉 Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

```
setLayout(null);

Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setLocation(50, 50);
b1.setSize(50, 50);

Point b1_p = b1.getLocation();
Dimension b1_d = b1.getSize();

b2.setLocation(b1_p.x+50, b1_p.y+50);
b2.setSize(b1_d.width+25, b1_d.height+25);
```

b1 객체의 크기를 지정

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈1〉 Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

```
setLayout(null);

Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setLocation(50, 50);
b1.setSize(50, 50);

Point b1_p = b1.getLocation();
Dimension b1_d = b1.getSize();

b2.setLocation(b1_p.x+50, b1_p.y+50);
b2.setSize(b1_d.width+25, b1_d.height+25);
```

b1 객체의 위치를 Poinit 객체로 가져옴

1 Component 클래스 이해하기

〈1〉 Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

```
setLayout(null);

Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setLocation(50, 50);
b1.setSize(50, 50);

Point b1_p = b1.getLocation();
Dimension b1_d = b1.getSize();

b2.setLocation(b1_p.x+50, b1_p.y+50);
b2.setSize(b1_d.width+25, b1_d.height+25);
```

b1 객체의 크기를 Dimension 객체로 가져옴

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

① Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

```
setLayout(null);

Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setLocation(50, 50);
b1.setSize(50, 50);

Point b1_p = b1.getLocation();
Dimension b1_d = b1.getSize();

b2.setLocation(b1_p.x+50, b1_p.y+50);
b2.setSize(b1_d.width+25, b1_d.height+25);
```

b2 객체의 위치를 b1 객체 위치에서 50만큼 이동하여 지정

1 Component 클래스 이해하기

① Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

```
setLayout(null);

Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setLocation(50, 50);
b1.setSize(50, 50);

Point b1_p = b1.getLocation();
Dimension b1_d = b1.getSize();

b2.setLocation(b1_p.x+50, b1_p.y+50);
b2.setSize(b1_d.width+25, b1_d.height+25);
```

b2 객체의 크기를 b1 객체의 크기 보다 25만큼 크게 지정

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈1〉 Component 클래스의 주요 메소드

2 크기 및 위치와 관련된 메소드

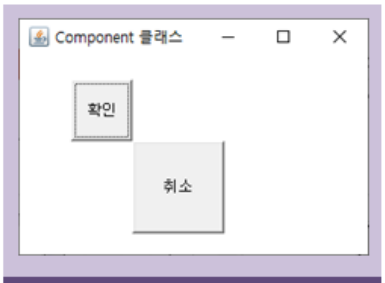
```
setLayout(null);

Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setLocation(50, 50);
b1.setSize(50, 50);

Point b1_p = b1.getLocation();
Dimension b1_d = b1.getSize();

b2.setLocation(b1_p.x+50, b1_p.y+50);
b2.setSize(b1_d.width+25, b1_d.height+25);
```



실행결과

1 Component 클래스 이해하기

〈1〉 Component 클래스의 주요 메소드

3 활성화 및 화면표시 여부와 관련된 메소드

메소드	설 명
boolean isEnabled()	컴포넌트의 활성화 여부를 반환
void setEnabled(boolean b)	컴포넌트의 활성화(true)/비활성화(false) 설정

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

1 Component 클래스의 주요 메소드

3 활성화 및 화면표시 여부와 관련된 메소드

메소드	설 명
boolean isVisible()	컴포넌트가 화면에 표시되어 있는지 여부를 반환
void setVisible(boolean b)	컴포넌트의 화면 표시(true)/화면 숨김(false) 설정

1 Component 클래스 이해하기

1 Component 클래스의 주요 메소드

3 활성화 및 화면표시 여부와 관련된 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setEnabled(true);
b2.setEnabled(true);
b1.setVisible(true);
b2.setVisible(true);

if( b1.isEnabled() ) b1.setVisible(false);

if( b2.isVisible() ) b2.setEnabled(false);
```

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨1⟩ Component 클래스의 주요 메소드

3 활성화 및 화면표시 여부와 관련된 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setEnabled(true);
b2.setEnabled(true);
b1.setVisible(true);
b2.setVisible(true);

if( b1.isEnabled() ) b1.setVisible(false);

if( b2.isVisible() ) b2.setEnabled(false);
```

b1, b2 객체를 활성화시킴

1 Component 클래스 이해하기

⟨1⟩ Component 클래스의 주요 메소드

3 활성화 및 화면표시 여부와 관련된 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setEnabled(true);
b2.setEnabled(true);
b1.setVisible(true);
b2.setVisible(true);

if( b1.isEnabled() ) b1.setVisible(false);

if( b2.isVisible() ) b2.setEnabled(false);
```

b1, b2 객체를 화면에 표시

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

1 Component 클래스의 주요 메소드

3 활성화 및 화면표시 여부와 관련된 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setEnabled(true);
b2.setEnabled(true);
b1.setVisible(true);
b2.setVisible(true);

if( b1.isEnabled() ) b1.setVisible(false);

if( b2.isVisible() ) b2.setEnabled(false);
```

b1 객체가 활성화된 상태이면
화면에 표시안함.

1 Component 클래스 이해하기

1 Component 클래스의 주요 메소드

3 활성화 및 화면표시 여부와 관련된 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setEnabled(true);
b2.setEnabled(true);
b1.setVisible(true);
b2.setVisible(true);

if( b1.isEnabled() ) b1.setVisible(false);

if( b2.isVisible() ) b2.setEnabled(false);
```

b2 객체가 화면에 표시된 상태라면 비활성화 함

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨1⟩ Component 클래스의 주요 메소드

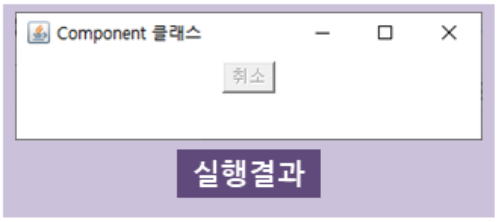
3 활성화 및 화면표시 여부와 관련된 메소드

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

b1.setEnabled(true);
b2.setEnabled(true);
b1.setVisible(true);
b2.setVisible(true);

if( b1.isEnabled() ) b1.setVisible(false);

if( b2.isVisible() ) b2.setEnabled(false);
```



1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

1 Button 컴포넌트

- 주요 메소드

생성자 메소드	설 명
Button(String label)	문자열을 지정하여 버튼 생성

메소드	설 명
String getLabel( )	버튼에 지정된 문자열을 반환
void setLabel(String label)	버튼에 문자열을 지정

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

1 Button 컴포넌트

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

String b1_s = b1.getLabel();
String b2_s = b2.getLabel();

b1.setLabel(b1_s+":"+b2_s);
b2.setLabel(b2_s+":"+b1_s);
```


1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

1 Button 컴포넌트


```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

String b1_s = b1.getLabel();
String b2_s = b2.getLabel();

b1.setLabel(b1_s+":"+b2_s);
b2.setLabel(b2_s+":"+b1_s);
```

← b1, b2 객체의 문자열을 가져옴

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

1 Button 컴포넌트

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

String b1_s = b1.getLabel();
String b2_s = b2.getLabel();

b1.setLabel(b1_s+":"+b2_s);
b2.setLabel(b2_s+":"+b1_s);
```

b1, b2 객체의 문자열을 조합해서 지정

1 Component 클래스 이해하기

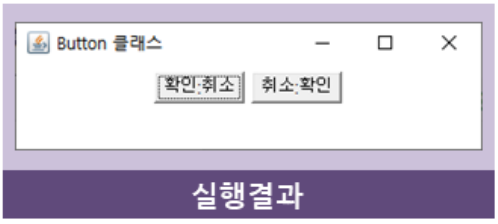
〈2〉 주요 컴포넌트

1 Button 컴포넌트

```
Button b1 = new Button("확인");
Button b2 = new Button("취소");

String b1_s = b1.getLabel();
String b2_s = b2.getLabel();

b1.setLabel(b1_s+":"+b2_s);
b2.setLabel(b2_s+":"+b1_s);
```

실행결과

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

2 Label 컴포넌트

- 문자열을 표시하는 컴포넌트

생성자 메소드	설 명
Label(String text)	문자열을 지정하여 레이블 생성
Label(String text, int alignment)	문자열과 정렬방법을 지정하여 레이블 생성

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

2 Label 컴포넌트

- 문자열을 표시하는 컴포넌트

메소드	설 명
String getText()	문자열을 반환
void setText(String text)	문자열을 지정
void setAlignment(int alignment)	문자열 정렬방법을 지정

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

2 Label 컴포넌트

- 문자열을 표시하는 컴포넌트

상수값(정렬방법)	설 명
static int CENTER	문자열을 중앙에 정렬하여 배치
static int LEFT	문자열을 왼쪽으로 정렬하여 배치(기본 배치)
static int RIGHT	문자열을 오른쪽으로 정렬하여 배치

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

2 Label 컴포넌트

```
Label la1 = new Label("첫번째 Label 입니다!");
Label la2 = new Label("두번째 Label 입니다!", Label.CENTER);

String text1 = la1.getText();
String text2 = la2.getText();

la1.setText(text2);
la2.setText(text1);

la1.setAlignment(Label.RIGHT);
la2.setAlignment(Label.LEFT);
```

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

2 Label 컴포넌트

문자열을 지정하여 Label 객체 생성
(기본 정렬 : LEFT)

```
Label la1 = new Label("첫번째 Label 입니다!");
Label la2 = new Label("두번째 Label 입니다!", Label.CENTER);

String text1 = la1.getText();
String text2 = la2.getText();

la1.setText(text2);
la2.setText(text1);

la1.setAlignment(Label.RIGHT);
la2.setAlignment(Label.LEFT);
```

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

2 Label 컴포넌트

문자열과 정렬방법을 CENTER로 설정

```
Label la1 = new Label("첫번째 Label 입니다!");
Label la2 = new Label("두번째 Label 입니다!", Label.CENTER);

String text1 = la1.getText();
String text2 = la2.getText();

la1.setText(text2);
la2.setText(text1);

la1.setAlignment(Label.RIGHT);
la2.setAlignment(Label.LEFT);
```

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

2 Label 컴포넌트

```
Label la1 = new Label("첫번째 Label 입니다!");
Label la2 = new Label("두번째 Label 입니다!", Label.CENTER);

String text1 = la1.getText();
String text2 = la2.getText();

la1.setText(text2);
la2.setText(text1);

la1.setAlignment(Label.RIGHT);
la2.setAlignment(Label.LEFT);
```

Label 객체의 문자열을 String 객체로 가져옴

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

2 Label 컴포넌트

```
Label la1 = new Label("첫번째 Label 입니다!");
Label la2 = new Label("두번째 Label 입니다!", Label.CENTER);

String text1 = la1.getText();
String text2 = la2.getText();

la1.setText(text2);
la2.setText(text1);

la1.setAlignment(Label.RIGHT);
la2.setAlignment(Label.LEFT);
```

2개의 문자열을 교환하여 지정

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

2 Label 컴포넌트

```
Label la1 = new Label("첫번째 Label 입니다!");
Label la2 = new Label("두번째 Label 입니다!", Label.CENTER);

String text1 = la1.getText();
String text2 = la2.getText();

la1.setText(text2);
la2.setText(text1);

la1.setAlignment(Label.RIGHT);
la2.setAlignment(Label.LEFT);
```

첫번째 Label 객체를 RIGHT 정렬로 지정

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

2 Label 컴포넌트

```
Label la1 = new Label("첫번째 Label 입니다!");
Label la2 = new Label("두번째 Label 입니다!", Label.CENTER);

String text1 = la1.getText();
String text2 = la2.getText();

la1.setText(text2);
la2.setText(text1);

la1.setAlignment(Label.RIGHT);
la2.setAlignment(Label.LEFT);
```

두번째 Label 객체를 LEFT 정렬로 지정

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

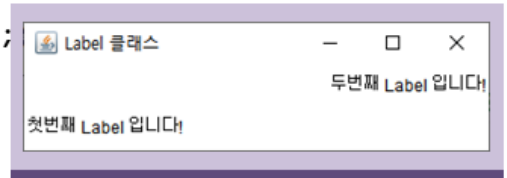
2 Label 컴포넌트

```
Label la1 = new Label("첫번째 Label 입니다!");
Label la2 = new Label("두번째 Label 입니다!", Label.CENTER);

String text1 = la1.getText();
String text2 = la2.getText();

la1.setText(text2);
la2.setText(text1);

la1.setAlignment(Label.RIGHT);
la2.setAlignment(Label.LEFT);
```



실행결과

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

3 Checkbox 컴포넌트

- 여러 선택 항목 중에서 한 개 이상을 선택할 수 있는 컴포넌트
- 여러 선택 항목 중에서 한 개만 선택하려면 그룹으로 묶어서 생성

생성자 메소드	설 명
Checkbox(String text)	문자열을 지정하여 레이블 생성
Checkbox(String text, boolean state)	• 문자열과 선택 여부를 지정하여 객체 생성 • state : true(선택함), false(선택하지 않음)
Checkbox(String text, Boolean state, CheckboxGroup group)	• 문자열, 선택 여부, 그룹을 지정하여 객체 생성 • 그룹 지정 : CheckboxGroup 객체를 생성하여 지정함)

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

3 Checkbox 컴포넌트

- 여러 선택 항목 중에서 한 개 이상을 선택할 수 있는 컴포넌트
- 여러 선택 항목 중에서 한 개만 선택하려면 그룹으로 묶어서 생성

메소드	설 명
boolean getState( )	선택 여부를 반환(선택됨 : true, 선택되어 있지 않음 : false)
void setState(boolean state)	선택 여부를 지정

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

3 Checkbox 컴포넌트

```
Checkbox c1 = new Checkbox("Melon “ , true);
Checkbox c2 = new Checkbox("Apple");
Checkbox c3 = new Checkbox("Banana");

if( c1.getState() ) c1.setState(false);

CheckboxGroup group = new CheckboxGroup();
Checkbox c4 = new Checkbox("Strawberry", true, group);
Checkbox c5 = new Checkbox("Mango", false, group);
Checkbox c6 = new Checkbox("Cherry“, false, group);
```

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

3 Checkbox 컴포넌트

선택된 상태로 객체 생성

```
Checkbox c1 = new Checkbox("Melon", true);
Checkbox c2 = new Checkbox("Apple");
Checkbox c3 = new Checkbox("Banana");

if( c1.getState() ) c1.setState(false);

CheckboxGroup group = new CheckboxGroup();
Checkbox c4 = new Checkbox("Strawberry", true, group);
Checkbox c5 = new Checkbox("Mango", false, group);
Checkbox c6 = new Checkbox("Cherry", false, group);
```

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

3 Checkbox 컴포넌트

```
Checkbox c1 = new Checkbox("Melon", true);
Checkbox c2 = new Checkbox("Apple");
Checkbox c3 = new Checkbox("Banana");

if( c1.getState() ) c1.setState(false);

CheckboxGroup group = new CheckboxGroup();
Checkbox c4 = new Checkbox("Strawberry", true, group);
Checkbox c5 = new Checkbox("Mango", false, group);
Checkbox c6 = new Checkbox("Cherry", false, group);
```

선택되지 않은 상태로 객체 생성

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

3 Checkbox 컴포넌트

```
Checkbox c1 = new Checkbox("Melon “ , true);
Checkbox c2 = new Checkbox("Apple");
Checkbox c3 = new Checkbox("Banana");

if( c1.getState() ) c1.setState(false);

CheckboxGroup group = new CheckboxGroup();
Checkbox c4 = new Checkbox("Strawberry", true, group);
Checkbox c5 = new Checkbox("Mango", false, group);
Checkbox c6 = new Checkbox("Cherry“, false, group);
```

첫번째 Checkbox 객체의 선택 여부를 반환 받아서 선택 해제를 함

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

3 Checkbox 컴포넌트

```
Checkbox c1 = new Checkbox("Melon “ , true);
Checkbox c2 = new Checkbox("Apple");
Checkbox c3 = new Checkbox("Banana");

if( c1.getState() ) c1.setState(false);

CheckboxGroup group = new CheckboxGroup();
Checkbox c4 = new Checkbox("Strawberry", true, group);
Checkbox c5 = new Checkbox("Mango", false, group);
Checkbox c6 = new Checkbox("Cherry“, false, group);
```

그룹으로 묶기 위해 객체 생성

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

3 Checkbox 컴포넌트

```
Checkbox c1 = new Checkbox("Melon “ , true);
Checkbox c2 = new Checkbox("Apple");
Checkbox c3 = new Checkbox("Banana");

if( c1.getState() ) c1.setState(false);

CheckboxGroup group = new CheckboxGroup();
Checkbox c4 = new Checkbox("Strawberry", true, group);
Checkbox c5 = new Checkbox("Mango", false, group);
Checkbox c6 = new Checkbox("Cherry“, false, group);
```

선택된 상태로 그룹으로 묶어서 생성

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

3 Checkbox 컴포넌트

```
Checkbox c1 = new Checkbox("Melon “ , true);
Checkbox c2 = new Checkbox("Apple");
Checkbox c3 = new Checkbox("Banana");

if( c1.getState() ) c1.setState(false);

CheckboxGroup group = new CheckboxGroup();
Checkbox c4 = new Checkbox("Strawberry", true, group);
Checkbox c5 = new Checkbox("Mango", false, group);
Checkbox c6 = new Checkbox("Cherry“, false, group);
```

선택되지 않은 상태로 그룹으로 묶어서 생성

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

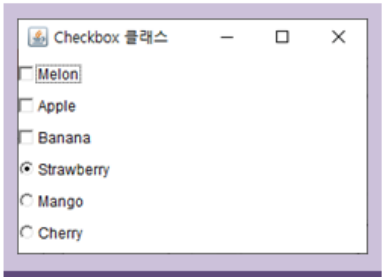
〈2〉 주요 컴포넌트

3 Checkbox 컴포넌트

```
Checkbox c1 = new Checkbox("Melon “ , true);
Checkbox c2 = new Checkbox("Apple");
Checkbox c3 = new Checkbox("Banana");

if( c1.getState() ) c1.setState(false);

CheckboxGroup group = new CheckboxGroup();
Checkbox c4 = new Checkbox("Strawberry", true, group);
Checkbox c5 = new Checkbox("Mango", false, group);
Checkbox c6 = new Checkbox("Cherry“, false, group);
```



실행결과

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

- 여러 선택 항목 중에서 한 개를 선택할 수 있는 풀다운 메뉴 형태의 컴포넌트

주요 메소드	설 명
void add(String item)	선택 항목을 추가
void remove(String item)	지정된 문자열 항목을 삭제
void remove(int position)	지정된 위치 값의 항목을 삭제 (위치 값은 첫 번째 항목이 0 부터)
void insert(String item, int position)	지정된 위치에 항목을 끼워 넣기

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

- 여러 선택 항목 중에서 한 개를 선택할 수 있는 풀다운 메뉴 형태의 컴포넌트

주요 메소드	설 명
String getSelectedItem()	선택된 항목의 문자열을 반환
int getSelectedIndex()	선택된 항목의 위치 값을 반환
void select (int position)	지정된 위치의 항목을 선택
void select (String item)	지정된 문자열로 지정된 항목을 선택

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

Choice 객체 생성

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

Choice 객체에 4개의 항목 추가

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

"Cherry" 항목 삭제

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

"Banana" 항목 삭제

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

"Banana" 항목을 두 번째 항목으로 끼워 넣기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

"Cherry" 항목을 두 번째 항목으로 끼워 넣기

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

세번째 항목을 선택

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

선택된 항목의 문자열을 반환

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

선택된 항목의 위치 값을 반환

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

4 Choice 컴포넌트

```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```

선택된 항목의 문자열과 위치 값을 출력
(출력결과는 “2:Banana”)

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

4 Choice 컴포넌트

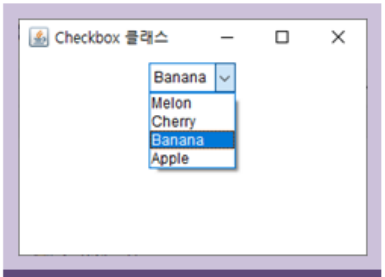
```
Choice c = new Choice();

c.add("Melon");
c.add("Apple");
c.add("Cherry");
c.add("Banana");

c.remove(2);
c.remove("Banana");

c.insert("Banana", 1);
c.insert("Cherry", 1);

c.select(2);
String select = c.getSelectedItem();
int position = c.getSelectedIndex();
System.out.println(position+":"+select);
```



실행결과

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

5 List 컴포넌트

- 선택 항목들이 스크롤되는 형태의 컴포넌트
- 한 개 또는 여러 개를 선택할 수 있음
- Choice 컴포넌트의 메소드들과 대부분 동일함

생성자 메소드	설 명
List(int rows)	지정된 표시행을 가지는 List 객체 생성 (한 개만 선택)
List(int rows, boolean multipleMode)	지정된 표시행을 가지는 List 객체 생성 (multipleMode : true 지정 시 여러 개 선택 가능)

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

5 List 컴포넌트

- 선택 항목들이 스크롤되는 형태의 컴포넌트
- 한 개 또는 여러 개를 선택할 수 있음
- Choice 컴포넌트의 메소드들과 대부분 동일함

메소드	설 명
void add(String item, int index)	선택 항목을 지정한 index 위치에 끼워 넣기
String[ ] getSelectedItems( )	선택된 항목들의 문자열 배열을 반환
int[ ] getSelectedIndexs( )	선택된 항목들의 위치 값들을 배열로 반환

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

5 List 컴포넌트

```
List c1 = new List(3);

c1.add("Melon");
c1.add("Apple");
c1.add("Banana");
c1.add("Cherry");
c1.add("Strawberry",2);

c1.select(1);
c1.select(2);

System.out.println(c1.getSelectedIndex()+":"+c1.getSelectedItem());
```

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

5 List 컴포넌트

```
List c1 = new List(3);

c1.add("Melon");
c1.add("Apple");
c1.add("Banana");
c1.add("Cherry");
c1.add("Strawberry",2);

c1.select(1);
c1.select(2);

System.out.println(c1.getSelectedIndex()+":"+c1.getSelectedItem());
```

표시행이 3줄인 List 객체 생성

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

5 List 컴포넌트

```
List c1 = new List(3);

c1.add("Melon");
c1.add("Apple");
c1.add("Banana");
c1.add("Cherry");
c1.add("Strawberry",2);

c1.select(1);
c1.select(2);

System.out.println(c1.getSelectedIndex()+":"+c1.getSelectedItem());
```

선택 항목 추가

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

5 List 컴포넌트

```
List c1 = new List(3);

c1.add("Melon");
c1.add("Apple");
c1.add("Banana");
c1.add("Cherry");
c1.add("Strawberry",2);

c1.select(1);
c1.select(2);

System.out.println(c1.getSelectedIndex()+":"+c1.getSelectedItem());
```

세 번째 위치에 항목 끼워 넣기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

5 List 컴포넌트

```
List c1 = new List(3);

c1.add("Melon");
c1.add("Apple");
c1.add("Banana");
c1.add("Cherry");
c1.add("Strawberry",2);

c1.select(1);
c1.select(2);

System.out.println(c1.getSelectedIndex()+":"+c1.getSelectedItem());
```

마지막에 선택한 항목만 유효함

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

5 List 컴포넌트

```
List c1 = new List(3);

c1.add("Melon");
c1.add("Apple");
c1.add("Banana");
c1.add("Cherry");
c1.add("Strawberry",2);

c1.select(1);
c1.select(2);

System.out.println(c1.getSelectedIndex()+":"+c1.getSelectedItem());
```

선택된 항목의 위치값과 문자열 출력
출력결과 : "2:Strawberry"

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

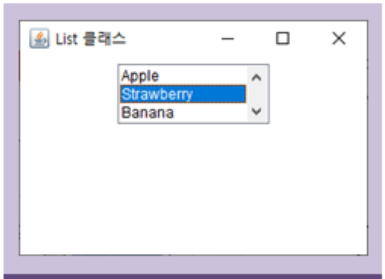
5 List 컴포넌트

```
List c1 = new List(3);

c1.add("Melon");
c1.add("Apple");
c1.add("Banana");
c1.add("Cherry");
c1.add("Strawberry",2);

c1.select(1);
c1.select(2);

System.out.println(c1.getSelectedIndex()+":"+c1.getSelectedItem());
```



실행결과

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

5 List 컴포넌트

```
List c2 = new List(3, true);

c2.add("Melon");
c2.add("Apple");
c2.add("Banana");
c2.add("Cherry");

c2.select(1);
c2.select(2);

for( int index : c2.getSelectedIndexes())
      System.out.println(index);

for( String item : c2.getSelectedItems())
      System.out.println(item);
```

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

5 List 컴포넌트

```
List c2 = new List(3, true);

c2.add("Melon");
c2.add("Apple");
c2.add("Banana");
c2.add("Cherry");

c2.select(1);
c2.select(2);

for( int index : c2.getSelectedIndexes())
      System.out.println(index);

for( String item : c2.getSelectedItems())
      System.out.println(item);
```

표시행이 3줄이고, 여러 개의 항목이 선택 가능하도록 객체를 생성함

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

5 List 컴포넌트

```
List c2 = new List(3, true);

c2.add("Melon");
c2.add("Apple");
c2.add("Banana");
c2.add("Cherry");

c2.select(1);
c2.select(2);

for( int index : c2.getSelectedIndexes())
      System.out.println(index);

for( String item : c2.getSelectedItems())
      System.out.println(item);
```

선택 항목 추가

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

5 List 컴포넌트

```
List c2 = new List(3, true);

c2.add("Melon");
c2.add("Apple");
c2.add("Banana");
c2.add("Cherry");

c2.select(1);
c2.select(2);

for( int index : c2.getSelectedIndexes())
      System.out.println(index);

for( String item : c2.getSelectedItems())
      System.out.println(item);
```

2개의 항목을 선택

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

5 List 컴포넌트

```
List c2 = new List(3, true);

c2.add("Melon");
c2.add("Apple");
c2.add("Banana");
c2.add("Cherry");

c2.select(1);
c2.select(2);

for( int index : c2.getSelectedIndexes())
      System.out.println(index);

for( String item : c2.getSelectedItems())
      System.out.println(item);
```

선택된 항목들의 위치 값들을 화면에 출력
출력결과 : 1 2

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

5 List 컴포넌트

```
List c2 = new List(3, true);

c2.add("Melon");
c2.add("Apple");
c2.add("Banana");
c2.add("Cherry");

c2.select(1);
c2.select(2);

for( int index : c2.getSelectedIndexes())
      System.out.println(index);

for( String item : c2.getSelectedItems())
      System.out.println(item);
```

선택된 항목들의 문자열들을 화면에 출력
출력결과 : “Apple” “Banana”

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

5 List 컴포넌트

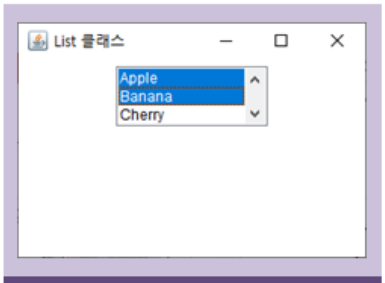
```
List c2 = new List(3, true);

c2.add("Melon");
c2.add("Apple");
c2.add("Banana");
c2.add("Cherry");

c2.select(1);
c2.select(2);

for( int index : c2.getSelectedIndexes())
      System.out.println(index);

for( String item : c2.getSelectedItems())
      System.out.println(item);
```



실행결과

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

6 Scrollbar 컴포넌트

- 표시 항목이 많고, 화면이 작을 때 원하는 위치로 이동이 가능한 컴포넌트

생성자 메소드	설 명
Scrollbar(int orientation)	스크롤바의 방향을 지정하여 객체 생성 orientation : Scrollbar.HORIZONTAL (수평 스크롤바) Scrollbar.VERTICAL (수직 스크롤바)

AWT 컴포넌트 활용하기

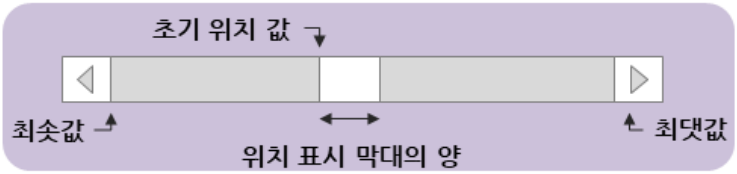
1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

6 Scrollbar 컴포넌트

- 표시 항목이 많고, 화면이 작을 때 원하는 위치로 이동이 가능한 컴포넌트

생성자 메소드	설 명
Scrollbar(int orientation, int value, int visible, int minimum, int maximum)	스크롤바의 다양한 값을 지정하여 객체 생성 • value : 스크롤바의 초기 위치 값 • visible : 스크롤바의 위치 표시 막대의 양 • minimun : 스크롤바의 최솟값 • maximum : 스크롤바의 최댓값



1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

6 Scrollbar 컴포넌트

생성자 메소드	설 명
int getValue()	스크롤바의 현재 값을 반환
void setValue(int newValue)	스크롤바의 현재 값을 새로운 값으로 지정

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

6 Scrollbar 컴포넌트

```
Scrollbar sb = new Scrollbar(Scrollbar.HORIZONTAL, 50, 0, 0, 100);
sb.setValue(80);
System.out.println(sb.getValue());
```


1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

6 Scrollbar 컴포넌트


```
Scrollbar sb = new Scrollbar(Scrollbar.HORIZONTAL, 50, 0, 0, 100);
sb.setValue(80);
System.out.println(sb.getValue());
```

수평 스크롤바 객체 생성
1. 스크롤바의 초기값 : 50
2. 위치 표시 막대 양 : 0(기본크기)
3. 최솟값 : 0
4. 최댓값 : 100

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

6 Scrollbar 컴포넌트

```
Scrollbar sb = new Scrollbar(Scrollbar.HORIZONTAL, 50, 0, 0, 100);
sb.setValue(80);
System.out.println(sb.getValue());
```

스크롤바의 값을 지정

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

6 Scrollbar 컴포넌트

```
Scrollbar sb = new Scrollbar(Scrollbar.HORIZONTAL, 50, 0, 0, 100);
sb.setValue(80);
System.out.println(sb.getValue());
```

스크롤바의 값을 화면에 출력 : 80

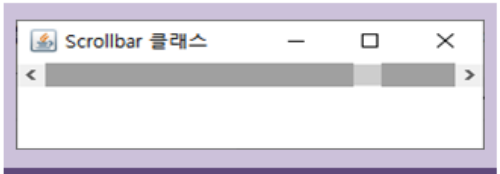
AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

6 Scrollbar 컴포넌트

```
Scrollbar sb = new Scrollbar(Scrollbar.HORIZONTAL, 50, 0, 0, 100);
sb.setValue(80);
System.out.println(sb.getValue());
```



실행결과

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

7 TextField 컴포넌트

- 한 줄짜리 문자열 입력이 가능한 컴포넌트

생성자 메소드	설 명
TextField()	기본 생성자 메소드
TextField(int columns)	열의 수를 지정하여 객체를 생성
TextField(String text)	초기 문자열이 지정된 객체를 생성
TextField(String text,int columns)	초기 문자열과 열의 수를 지정하여 객체를 생성

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

7 TextField 컴포넌트

- 한 줄짜리 문자열 입력이 가능한 컴포넌트

메소드	설 명
String getText( )	입력된 문자열을 반환
void setText(String t)	새로운 문자열을 지정
void setEchoChar(char c)	입력 문자에 대하여 화면에 표시할 문자(Echo 문자)를 지정 * 암호 입력 컴포넌트로 활용 가능

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

7 TextField 컴포넌트

```
TextField t1 = new TextField();
TextField t2 = new TextField(10);
TextField t3 = new TextField("이름 입력하기!");
TextField t4 = new TextField("abcdefg");
t4.setEchoChar('*');
t1.setText(t4.getText());
```

AWT 컴포넌트 활용하기

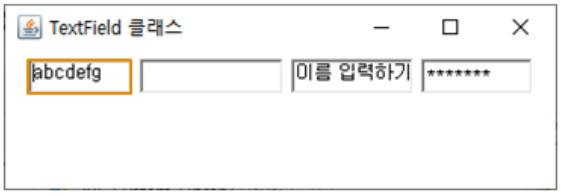
1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

7 TextField 컴포넌트

```
TextField t1 = new TextField();
TextField t2 = new TextField(10);
TextField t3 = new TextField("이름 입력하기!");
TextField t4 = new TextField("abcdefg");
t4.setEchoChar('*');
t1.setText(t4.getText());
```

기본 객체 생성



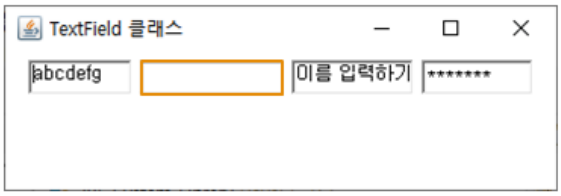
1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

7 TextField 컴포넌트

열의 수를 지정하여 객체 생성

```
TextField t1 = new TextField();
TextField t2 = new TextField(10);
TextField t3 = new TextField("이름 입력하기!");
TextField t4 = new TextField("abcdefg");
t4.setEchoChar('*');
t1.setText(t4.getText());
```



AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

7 TextField 컴포넌트

```
TextField t1 = new TextField();
TextField t2 = new TextField(10);
TextField t3 = new TextField("이름 입력하기!");
TextField t4 = new TextField("abcdefg");
t4.setEchoChar('*');
t1.setText(t4.getText());
```

초기 문자열을 지정하여 객체 생성

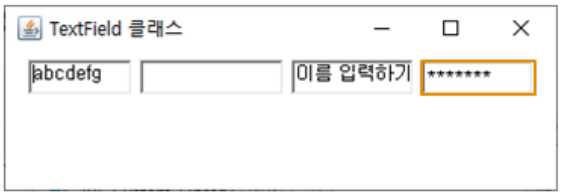
1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

7 TextField 컴포넌트

```
TextField t1 = new TextField();
TextField t2 = new TextField(10);
TextField t3 = new TextField("이름 입력하기!");
TextField t4 = new TextField("abcdefg");
t4.setEchoChar('*');
t1.setText(t4.getText());
```

초기 문자열과 Echo 문자를 지정하여 객체 생성



AWT 컴포넌트 활용하기

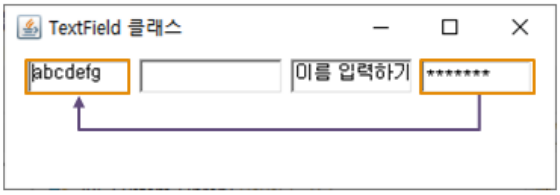
1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

7 TextField 컴포넌트

```
TextField t1 = new TextField();
TextField t2 = new TextField(10);
TextField t3 = new TextField("이름 입력하기!");
TextField t4 = new TextField("abcdefg");
t4.setEchoChar('*');
t1.setText(t4.getText());
```

t4 객체에 입력된 문자열을 가져와 t1 객체에 새로운 문자열로 지정



1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

8 TextArea 컴포넌트

- 여러 줄짜리 문자열 입력이 가능한 컴포넌트

생성자 메소드	설 명
TextArea( )	기본 생성자 메소드
TextArea(int rows, int columns)	행과 열의 수를 지정하여 객체를 생성
TextArea(String text)	초기 문자열이 지정된 객체를 생성

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

8 TextArea 컴포넌트

- 여러 줄짜리 문자열 입력이 가능한 컴포넌트

생성자 메소드	설 명
TextArea(String text, int rows, int columns)	초기 문자열, 행과 열의 수를 지정하여 객체를 생성
TextArea(String text, int rows, int columns, int scrollbars)	초기 문자열, 행과 열의 수, Scrollbar의 표시 방법을 지정하여 객체를 생성 scrollbars : TextArea.SCROLLBARS_BOTH TextArea.SCROLLBARS_HORIZONTAL_ONLY TextArea.SCROLLBARS_VERTICAL_ONLY TextArea.SCROLLBARS_NONE

1 Component 클래스 이해하기

⟨2⟩ 주요 컴포넌트

8 TextArea 컴포넌트

메소드	설 명
void append(String str)	마지막 줄에 문자열을 추가
void insert (String str, int pos)	지정된 위치에 문자열을 끼워 넣음
void replaceRange (String str, int start, int end)	start 부터 end-1 사이의 문자열을 바꾸어 넣음

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

8 TextArea 컴포넌트

```
TextArea t1 = new TextArea();
TextArea t2 = new TextArea("안녕하세요", 5, 10, TextArea.SCROLLBARS_BOTH);

t1.append("Java Application!!\n");
t1.append("Java Application!!\n");
t1.insert("JAVA\n", 19);
t1.replaceRange("JAVA",0,4);
```

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

8 TextArea 컴포넌트

```
TextArea t1 = new TextArea();
TextArea t2 = new TextArea("안녕하세요", 5, 10, TextArea.SCROLLBARS_BOTH);

t1.append("Java Application!!\n");
t1.append("Java Application!!\n");
t1.insert("JAVA\n", 19);
t1.replaceRange("JAVA",0,4);
```

← 기본 생성자 메소드로 객체 생성

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

8 TextArea 컴포넌트

```
TextArea t1 = new TextArea();
TextArea t2 = new TextArea("안녕하세요", 5, 10, TextArea.SCROLLBARS_BOTH);

t1.append("Java Application!!\n");
t1.append("Java Application!!\n");
t1.insert("JAVA\n", 19);
t1.replaceRange("JAVA",0,4);
```

1. 초기문자열 : "안녕하세요"
2. 행의 수 : 5
3. 열의 수 : 10
4. 스크롤 표시 방법 : 수평과 수직 스크롤 모두 표시

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

8 TextArea 컴포넌트

```
TextArea t1 = new TextArea();
TextArea t2 = new TextArea("안녕하세요", 5, 10, TextArea.SCROLLBARS_BOTH);

t1.append("Java Application!!\n");
t1.append("Java Application!!\n");
t1.insert("JAVA\n", 19);
t1.replaceRange("JAVA",0,4);
```

마지막 줄에 문자열 2번 추가

AWT 컴포넌트 활용하기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

8 TextArea 컴포넌트

```
TextArea t1 = new TextArea();
TextArea t2 = new TextArea("안녕하세요", 5, 10, TextArea.SCROLLBARS_BOTH);

t1.append("Java Application!!\n");
t1.append("Java Application!!\n");
t1.insert("JAVA\n", 19);
t1.replaceRange("JAVA",0,4);
```

19번째 위치에 문자열 "JAVA"를 끼워 넣기

1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

8 TextArea 컴포넌트

```
TextArea t1 = new TextArea();
TextArea t2 = new TextArea("안녕하세요", 5, 10, TextArea.SCROLLBARS_BOTH);

t1.append("Java Application!!\n");
t1.append("Java Application!!\n");
t1.insert("JAVA\n", 19);
t1.replaceRange("JAVA",0,4);
```

위치 0 부터 3까지 "JAVA" 문자열로 교체하기

AWT 컴포넌트 활용하기

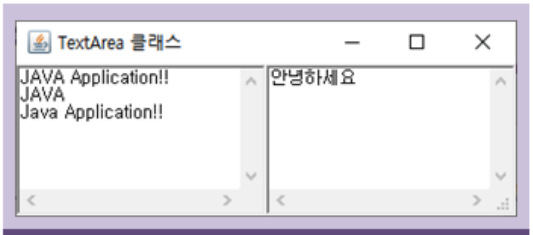
1 Component 클래스 이해하기

〈2〉 주요 컴포넌트

8 TextArea 컴포넌트

```
TextArea t1 = new TextArea();
TextArea t2 = new TextArea("안녕하세요", 5, 10, TextArea.SCROLLBARS_BOTH);

t1.append("Java Application!!\n");
t1.append("Java Application!!\n");
t1.insert("JAVA\n", 19);
t1.replaceRange("JAVA",0,4);
```



실행결과

AWT 컴포넌트 활용하기

AWT 패키지 활용하기 | AWT 컴포넌트 활용하기

실습하기

AWT 패키지 활용하기 | AWT 컴포넌트 활용하기

실습순서

2. Component 클래스 관련 실습
   1) Component 기본 실습
   2) Button, Label, Checkbox, Choice, List, Scrollbar, TextField, TextArea

유의사항

- JDK와 이클립스를 설치한 후 실습이 가능함
- 본인이 원하는 작업 폴더를 미리 정해 놓은 다음 실습하기
- 작업 폴더는 C드라이브에 지정하기 보다는 D드라이브나 외장하드디스크를 활용하는 것을 추천함

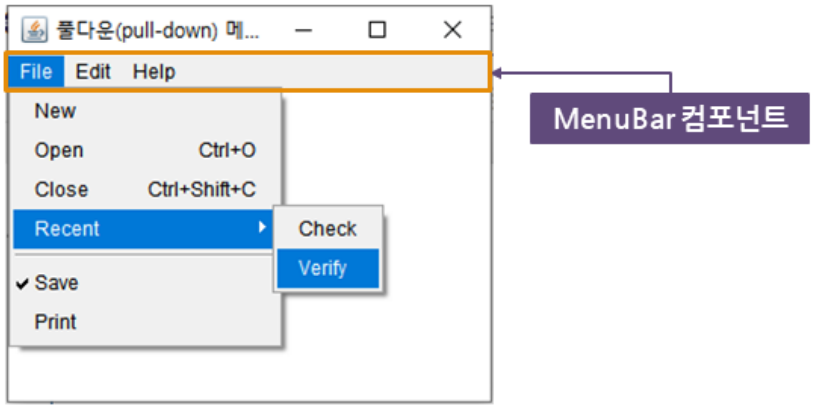
※ 제공되는 실습 코드를 다운받아 실습해보시기 바랍니다.

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

⟨1⟩ 풀다운(pull-down) 메뉴

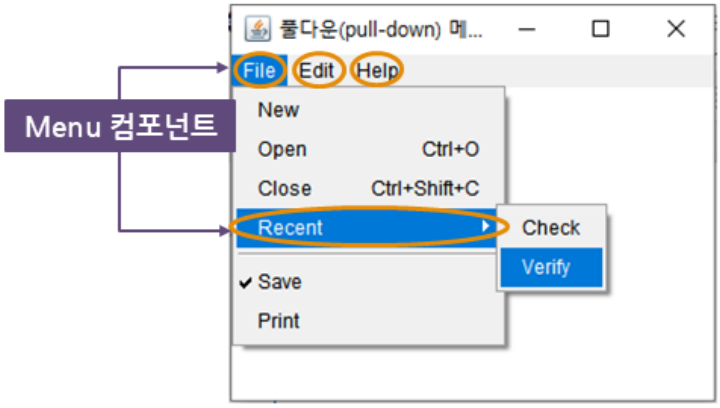
1 풀다운 메뉴 구성 컴포넌트



2 MenuComponent 클래스 이해하기

⟨1⟩ 풀다운(pull-down) 메뉴

1 풀다운 메뉴 구성 컴포넌트

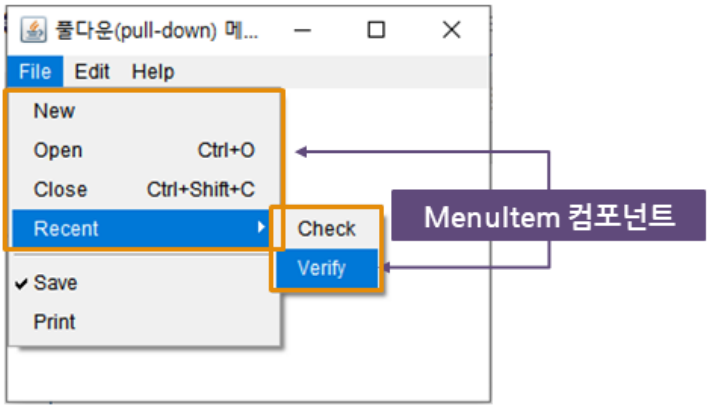


AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

⟨1⟩ 풀다운(pull-down) 메뉴

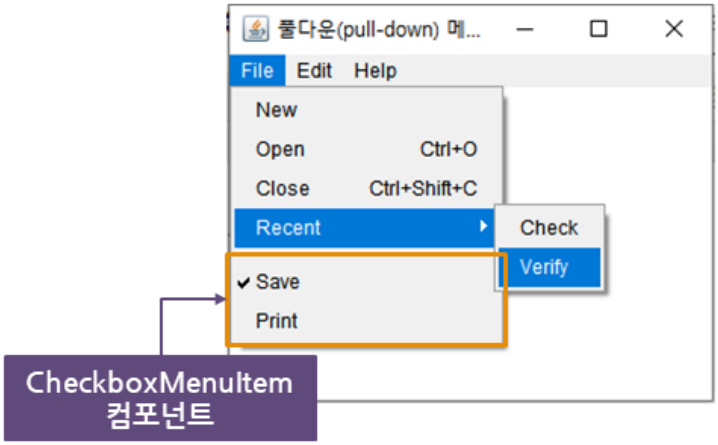
1 풀다운 메뉴 구성 컴포넌트



2 MenuComponent 클래스 이해하기

⟨1⟩ 풀다운(pull-down) 메뉴

1 풀다운 메뉴 구성 컴포넌트



AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

1 풀다운(pull-down) 메뉴

2 풀다운 메뉴 프로그램 순서

```
MenuBar mb = new MenuBar();

Menu file = new Menu( " File " );
mb.add(file);

MenuItem open = new MenuItem( " Open " );
CheckboxMenuItem  save = new CheckboxMenuItem( " Save " );
file.add(open);
file.add(save);

setMenuBar( mb );
```


2 MenuComponent 클래스 이해하기

1 풀다운(pull-down) 메뉴

2 풀다운 메뉴 프로그램 순서


```
MenuBar mb = new MenuBar();
```

1. MenuBar 객체 생성


```
Menu file = new Menu( " File " );
mb.add(file);

MenuItem open = new MenuItem( " Open " );
CheckboxMenuItem  save = new CheckboxMenuItem( " Save " );
file.add(open);
file.add(save);

setMenuBar( mb );
```

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

1 풀다운(pull-down) 메뉴

2 풀다운 메뉴 프로그램 순서

```
MenuBar mb = new MenuBar();

Menu file = new Menu( " File " );
mb.add(file);

MenuItem open = new MenuItem( " Open " );
CheckboxMenuItem  save = new CheckboxMenuItem( " Save " );
file.add(open);
file.add(save);

setMenuBar( mb );
```

2. Menu 객체 생성

2 MenuComponent 클래스 이해하기

1 풀다운(pull-down) 메뉴

2 풀다운 메뉴 프로그램 순서

```
MenuBar mb = new MenuBar();

Menu file = new Menu( " File " );
mb.add(file);

MenuItem open = new MenuItem( " Open " );
CheckboxMenuItem  save = new CheckboxMenuItem( " Save " );
file.add(open);
file.add(save);

setMenuBar( mb );
```

3. Menu 객체를 MenuBar 객체에 추가

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

1 풀다운(pull-down) 메뉴

2 풀다운 메뉴 프로그램 순서

```
MenuBar mb = new MenuBar();

Menu file = new Menu( " File " );
mb.add(file);

MenuItem open = new MenuItem( " Open " );
CheckboxMenuItem  save = new CheckboxMenuItem( " Save " );
file.add(open);
file.add(save);

setMenuBar( mb );
```

4. MenuItem 또는 CheckboxMenuItem 객체 생성

2 MenuComponent 클래스 이해하기

1 풀다운(pull-down) 메뉴

2 풀다운 메뉴 프로그램 순서

```
MenuBar mb = new MenuBar();

Menu file = new Menu( " File " );
mb.add(file);

MenuItem open = new MenuItem( " Open " );
CheckboxMenuItem  save = new CheckboxMenuItem( " Save " );
file.add(open);
file.add(save);

setMenuBar( mb );
```

5. MenuItem 또는 CheckboxMenuItem 객체를 Menu 객체에 추가

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

⟨1⟩ 풀다운(pull-down) 메뉴

2 풀다운 메뉴 프로그램 순서

```
MenuBar mb = new MenuBar();

Menu file = new Menu( " File " );
mb.add(file);

MenuItem open = new MenuItem( " Open " );
CheckboxMenuItem  save = new CheckboxMenuItem( " Save " );
file.add(open);
file.add(save);

setMenuBar( mb );
```

← 6. MenuBar 객체를 Frame 객체에 추가

2 MenuComponent 클래스 이해하기

⟨1⟩ 풀다운(pull-down) 메뉴

3 MenuBar 컴포넌트

생성자 메소드	설 명
MenuBar()	기본 생성자 메소드

메소드	설 명
void add(Menu m)	Menu 객체를 MenuBar 객체에 추가

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

〈1〉 풀다운(pull-down) 메뉴

3 MenuBar 컴포넌트

*MenuBar 객체를 Frame에 추가하는 메소드

Frame 클래스의 메소드	설 명
void setMenuBar(MenuBar mb)	MenuBar 객체를 Frame 객체에 추가

2 MenuComponent 클래스 이해하기

〈1〉 풀다운(pull-down) 메뉴

4 Menu 컴포넌트

생성자 메소드	설 명
Menu(String label)	메뉴의 문자열을 지정하여 객체 생성
Menu(String label, boolean tearOff)	메뉴의 문자열을 지정하여 객체 생성 tearOff : true 지정시 분리되는 메뉴가 됨

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

〈1〉 풀다운(pull-down) 메뉴

4 Menu 컴포넌트

생성자 메소드	설 명
MenuItem add(MenuItem mi)	MenuItem 객체를 Menu 객체에 추가
void addSeparator()	메뉴의 구분선을 추가

2 MenuComponent 클래스 이해하기

〈1〉 풀다운(pull-down) 메뉴

5 MenuItem 컴포넌트

생성자 메소드	설 명
MenuItem(String label)	메뉴 항목의 문자열을 지정하여 객체 생성
MenuItem(String label, MenuShortcut s)	메뉴 항목의 문자열을 지정하여 객체 생성 • s : 메뉴 항목의 단축키를 MenuShortcut 객체로 지정

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

① 풀다운(pull-down) 메뉴

5 MenuItem 컴포넌트

생성자 메소드	설 명
boolean isEnabled()	MenuItem 객체가 활성되어 있는지 여부를 반환
void setEnabled(boolean b)	MenuItem 객체의 활성화 여부를 지정 • b : true 지정시 활성화, false 지정시 비활성화

2 MenuComponent 클래스 이해하기

① 풀다운(pull-down) 메뉴

6 MenuShortcut 클래스

생성자 메소드	설 명
MenuShortcut(int key)	키 값을 지정하여 MenuShortcut 객체를 생성 단축키 : Ctrl+key
MenuShortcut(int key, boolean useShiftModifier)	키 값과 Shift 키 사용여부를 지정 useShiftModifirer - true를 지정하면 Shift 키 사용 단축키 : Ctrl+Shift+key - false를 지정하면 Shift 키 사용 안함 단축키 : Ctrl+key

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

〈1〉 풀다운(pull-down) 메뉴

6 MenuShortcut클래스

```
MenuItem file = new MenuItem( " File " , new MenuShortcut(KeyEvent.VK_A, true));
```

MenuShortcut 객체 생성

2 MenuComponent 클래스 이해하기

〈1〉 풀다운(pull-down) 메뉴

6 MenuShortcut클래스

```
MenuItem file = new MenuItem( " File " , new MenuShortcut(KeyEvent.VK_A, true));
```

단축키 값 : 'A'
KeyEvent 클래스의 상수 값을 이용

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

⟨1⟩ 풀다운(pull-down) 메뉴

6 MenuShortcut 클래스

```
MenuItem file = new MenuItem( " File " , new MenuShortcut(KeyEvent.VK_A, true));
```

Shit키 사용 지정(Ctrl+Shift+A)

2 MenuComponent 클래스 이해하기

⟨1⟩ 풀다운(pull-down) 메뉴

7 CheckboxMenuItem 컴포넌트

생성자 메소드	설 명
CheckboxMenuItem(String label)	체크 박스 메뉴 항목의 문자열을 지정하여 객체 생성
CheckboxMenuItem(String label, boolean state)	체크 박스 메뉴 항목의 문자열을 지정하여 객체 생성 • state : true를 지정하면 체크 박스가 선택됨 false를 지정하면 체크 박스가 비선택됨

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

⟨1⟩ 풀다운(pull-down) 메뉴

7 CheckboxMenuItem 컴포넌트

메소드	설 명
boolean getState( )	CheckboxMenuItem 객체가 선택되어 있는지 여부를 반환
void setState(boolean b)	CheckboxMenuItem 객체의 선택여부를 지정 • b : true 지정시 선택, false 지정시 비선택

2 MenuComponent 클래스 이해하기

⟨2⟩ 팝업(pop-up) 메뉴

1 팝업 메뉴란?

마우스 오른쪽 버튼 클릭 시 나타나는 메뉴

AWT 컴포넌트 활용하기

2 MenuComponent 클래스 이해하기

〈2〉 팝업(pop-up) 메뉴

2 팝업 메뉴 프로그램 순서

- PopupMenu 객체 생성
- MenuItem 객체 생성
- MenuItem 객체를 PopupMenu 객체에 추가
- PopupMenu 객체를 표시하고 싶은 컴포넌트에 추가
- show() 메소드로 팝업 메뉴를 화면에 표시 (마우스 오른쪽 버튼 클릭시)

2 MenuComponent 클래스 이해하기

〈2〉 팝업(pop-up) 메뉴

3 PopupMenu 컴포넌트

생성자 메소드	설 명
PopupMenu()	기본 생성자 메소드

메소드	설 명
void show(Component origin, int x, int y)	PopupMenu 객체를 화면에 표시함 • origin : PopupMenu 객체가 추가된 컴포넌트를 지정 • x, y : 팝업 메뉴가 표시될 좌표값

AWT 컴포넌트 활용하기

AWT 패키지 활용하기 | AWT 컴포넌트 활용하기

실습하기

AWT 패키지 활용하기 | AWT 컴포넌트 활용하기

실습순서

3. MenuComponent 클래스 관련 실습
   1) 풀다운 메뉴 실습
   2) 팝업 메뉴 실습

유의사항

- JDK와 이클립스를 설치 완료된 후에 실습이 가능
- 본인이 원하는 작업 폴더를 미리 정해 놓은 다음 실습하기
- 작업 폴더는 C드라이브에 지정하기 보다는 D드라이브나 외장하드디스크를 활용하기를 추천

※ 제공되는 실습 코드를 다운받아 실습해보시기 바랍니다.

응용문제

AWT 패키지 활용하기 응용문제

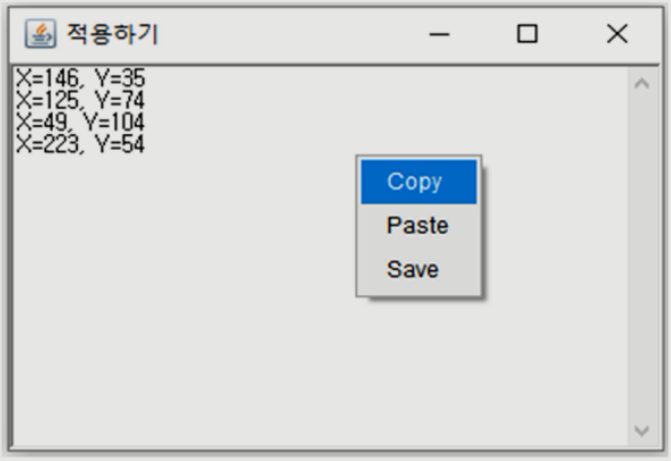
다음 실행화면과 조건에 맞게 프로그램을 작성하시오.

조건

1. Frame 클래스를 상속하시오.
2. TextArea 객체를 화면에 배치하시오.
3. TextArea 객체에서 팝업 메뉴를 출력하시오.
4. 팝업 메뉴가 화면에 표시될 경우 좌표값을 TextArea에 표시하시오.
5. 팝업 메뉴 : Copy, Paste, Save

클래스명 : AWTLast

제공되는 실습 소스코드를 다운받아 실습해보시기 바랍니다.

AWT 패키지 활용하기 응용문제

실행화면