

# 情報処理技法 (Javaプログラミング)2

第13回  
操作に対して処理が行われるGUI(2)  
人間科学科コミュニケーション専攻  
白銀 純子

Copyright (C) Aichi Shoinen, Tokyo Women's Christian University 2019. All rights reserved.

## 第13回の内容

- ボタンを押したときのウィンドウ操作
- 入力された情報を次のウィンドウに送るには?
- GUIでのファイル操作

Copyright (C) Aichi Shoinen, Tokyo Women's Christian University 2019. All rights reserved.

## 前回の復習問題の解答(1)

- 下図のウィンドウについて、レイアウトマネージャだけで部品を配置するとき、下記の点について考えて答えなさい。ただし、レイアウトマネージャは、授業で説明したBorderLayout・GridLayout・FlowLayoutのどれかとする。
- JFrameにどのレイアウトマネージャを設定するか
- どの部品をJPanelでグループ化し、そのJPanelにはどのレイアウトマネージャを設定するか  
※ 部品をBorderLayoutのJFrameまたはJPanelに配置するときには、その部品を東・西・南・北・中央のどの位置に配置するかを答えること



Copyright (C) Aichi Shoinen, Tokyo Women's Christian University 2019. All rights reserved.

## 前回の復習問題の解答(2)

解答例1:

- JPanelを3つ用意(panel1, panel2, panel3とする)
- JFrameにBorderLayoutを設定
  - ✓ JLabelをJFrameの北に配置
  - ✓ panel1をJFrameの中央に配置
  - ✓ panel3をJFrameの南に配置
- panel1にBorderLayoutを設定
  - JScrollPaneをpanel1の中央に配置
  - JListをJScrollPaneに貼り付け
  - panel2をpanel1の南に配置
- panel2にGridLayout(2, 3)またはFlowLayoutを設定
  - ✓ 5つのJRadioButtonをpanel2に配置
- panel3にFlowLayoutを設定
  - ✓ 2つのJButtonをpanel3に配置

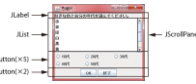


Copyright (C) Aichi Shoinen, Tokyo Women's Christian University 2019. All rights reserved.

## 前回の復習問題の解答(3)

解答例2:

- JPanelを3つ用意(panel1, panel2, panel3とする)
- JFrameにBorderLayoutを設定
  - ✓ panel1をJFrameの中央に配置
  - ✓ panel3をJFrameの南に配置
- panel1にBorderLayoutを設定
  - ✓ JLabelをpanel1の北に配置
  - ✓ JScrollPaneをpanel1の中央に配置
  - ✓ JListをJScrollPaneに貼り付け
  - ✓ panel2をpanel1の南に配置
- panel2にGridLayout(2, 3)またはFlowLayoutを設定
  - ✓ 5つのJRadioButtonをpanel2に配置
- panel3にFlowLayoutを設定
  - ✓ 2つのJButtonをpanel3に配置



Copyright (C) Aichi Shoinen, Tokyo Women's Christian University 2019. All rights reserved.

## 前回の復習

Copyright (C) Aichi Shoinen, Tokyo Women's Christian University 2019. All rights reserved.

## GUIプログラムのカタチ(処理つき)(1)

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

public class クラス名 extends JFrame implements リスナ名 {
    GUI部品の変数宣言
    public クラス名() { /* コンストラクタ */
        .....
        イベントが発生する部品の変数名 add リスナの名前(this);
        .....
    }
    public void リスナ名のメソッド名(イベント名 e) { /* リスナで決められたメソッド */
        .....
        イベントが発生したときの処理内容を書く領域
        .....
    }
    public static void main(String[] args) {
        new クラス名();
    }
}
```

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2016. All rights reserved.

## ActionListener

- もっともオーソドックスなリスナ
- ボタンをマウスの左ボタンで押すとき、メニューから選択するときのリスナ
- オーバーライドするメソッドは「actionPerformed(ActionEvent 引数名)」
  - 戻り値は「void」
  - 引数は「ActionEvent」
    - 主に、「ボタンを押す」という意味のイベント

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2016. All rights reserved.

## イベントが起こる部品が複数のとき?

- 「メソッドの引数名 getSource()」というメソッドで、どの部品でイベントが発生したかを知ることができる
- このメソッドを使ってイベントが発生した部品を受け取り、if文で処理内容を分岐させる

```
JButton okBut, cancelBut;
.....
public void actionPerformed(ActionEvent e) {
    if (okBut == e.getSource()) { /* 「okBut」が押された場合 */
        .....
        okButが押されたときの処理を書く
    } else if (cancelBut == e.getSource()) {
        /* 「cancelBut」が押された場合 */
        .....
        cancelButが押されたときの処理を書く
    }
}
```

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2016. All rights reserved.

## 例

「東京女子大学」と標準出力に出力

「東女」ボタンにリスナ登録

「杉並区」と標準出力に出力

```
public class Sample extends JFrame implements ActionListener {
    JButton twcu, suginami;

    public Sample() { /* コンストラクタ */
        getContentPane().setLayout(null);

        twcu = new JButton("東女");
        twcu.setBounds(10, 10, 100, 25);
        twcu.addActionListener(this);
        getContentPane().add(twcu);
    }
}
```

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2016. All rights reserved.

## 例(続き1)

「杉並」ボタンにリスナ登録

```
suginami = new JButton("杉並");
suginami.setBounds(110, 10, 100, 25);
suginami.addActionListener(this);
getContentPane().add(suginami);

setTitle("サンプル");
setSize(280, 70);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setVisible(true);
} /* コンストラクタ終わり */
```

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2016. All rights reserved.

## 例(続き2)

「東女」ボタンの処理内容記述

「杉並」ボタンの処理内容記述

```
public void actionPerformed(ActionEvent e) {
    /* ボタンが押されたときの処理内容 */
    if (e.getSource() == twcu) { /* 「東女」が押されたときの処理 */
        System.out.println("東京女子大学");
    } else { /* 「杉並」が押されたときの処理 */
        System.out.println("杉並区");
    }
}

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

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2016. All rights reserved.

## ボタンを押したときのウィンドウ操作



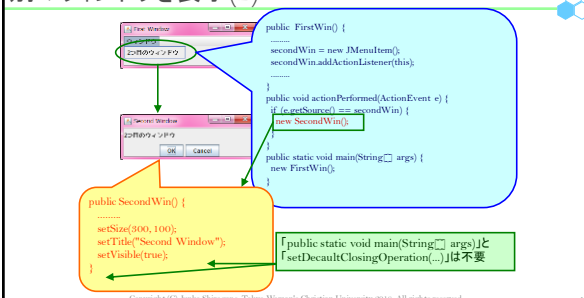
## ボタンを押したときのウィンドウ操作

- ウィンドウに関して、行われる主な処理としては...
  - 別のウィンドウを表示
  - 現在表示しているウィンドウを消去

## 別のウィンドウを表示(1)

1. 2つ目のウィンドウを表示するプログラムを書く
  - 原則として、1つのクラスに1つのウィンドウのプログラム
  - 2つ目以降のクラスには「`public static void main(String[] args)`」は不要
  - 2つ目以降のクラスには「`setDefaultClosingOperation`」も不要
    - 2つ目のウィンドウを閉じたときにソフトが終了すると困るから
2. 1つ目のウィンドウのリスナのメソッドの中に2つ目のウィンドウを表示する命令を書く
  - 「`new 2つ目のウィンドウのクラス名();`」でウィンドウを表示

## 別のウィンドウを表示(2)



## 別のウィンドウを表示(3)

- プログラムのコンパイル:
 

```
javac 1つ目のファイル.java 2つ目のファイル ...
```

 で、関係するファイルをコンパイル可能
- プログラムの実行:
 

```
java mainを持つクラスのクラス名
```

 で実行
  - 「mainを持つクラスのクラス名」が、1つ目のウィンドウ

## 3つ目以降のウィンドウ

- 1つ目のウィンドウから2つ目のウィンドウを表示する場合と同様
- n番目のウィンドウからn+1番目のウィンドウを表示する場合
  - n+1番目のウィンドウを表示するプログラムを書く
  - n番目のウィンドウ内のリスナのメソッド内にn+1番目のウィンドウを表示する命令を書く
    - 「`new n+1番目のウィンドウのクラス名();`」でウィンドウを表示

## ウィンドウを消す処理

- 「setVisible(false)」でウィンドウが消える
- setVisible: JFrameクラスで定義されているメソッド
  - 「extends JFrame」でJFrameクラスを継承しているので、「オブジェクトの変数名.setVisible」という形でなく利用可能

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        // 「OK」ボタンが押されたときの処理
        setVisible(false);
    } else if (e.getSource() == cancelBut) {
        setVisible(false);
    }
}
```

ウィンドウを消す処理

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2019. All rights reserved.

## 入力された情報を次に送るには?

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2019. All rights reserved.

## 入力された情報を次に送る

- 例えば...入力した情報の確認をするとき



情報入力のウィンドウから情報確認のウィンドウへ、入力された情報を送る必要

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2019. All rights reserved.

## プログラムの書き方(1)

- 情報を受け取るウィンドウのコンストラクタに引数をつける

```
AddressConfirm.java
public AddressConfirm(String name, String address, String tel, String gender) {
    .....
    nameLabel = new JLabel();
    nameLabel.setText("名前:");
    .....
    nameField = new JTextField();
    nameField.setText(name);
    .....
    addressLabel = new JLabel();
    addressLabel.setText("住所:");
    .....
    addressField = new JTextField();
    addressField.setText(address);
    .....
}
```

入力された情報を受け取る引数  
 ➤ name: 名前の受け取り  
 ➤ address: 住所の受け取り  
 ➤ tel: 電話番号の受け取り  
 ➤ gender: 性別の受け取り

コンストラクタの引数(受け取った情報をラベルに表示)

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2019. All rights reserved.

## プログラムの書き方(2)

- 情報を入力するウィンドウから、確認するウィンドウを表示するときに、引数付きでウィンドウを作る

```
AddressInput.java
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        String name, address, tel, gender;
        name = nameField.getText();
        address = addressField.getText();
        tel = telField.getText();
        if (maleRadio.isSelected() == true) {
            gender = "男性";
        } else {
            gender = "女性";
        }
        new AddressConfirm(name, address, tel, gender);
    }
}
```

入力フィールドから入力された情報を受け取り、変数に代入

JRadioButtonは、どれが押されていれば変数に何を代入するかを記述

引数を使って確認用ウィンドウに情報を渡し

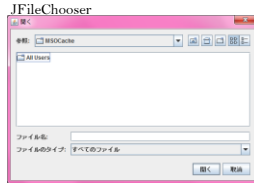
Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2019. All rights reserved.

## GUIでのファイル操作

Copyright (C) Tokyo Shoin, Tokyo Women's Christian University 2019. All rights reserved.

## GUIでのファイル操作

- ファイル操作専用のGUI部品 - JFileChooser
  - 読み込むファイルを決める
  - 情報を書き出す(保存する)ファイルを決める



Copyright (C) Atsuhiko Shimamura, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser

- これだけでファイル選択のウィンドウを表示する部品
  - JFrameに貼りつける必要はなし
- ウィンドウ表示の方法
  - ファイル読み込み: 「showOpenDialog(null)」というメソッドを利用
  - ファイル書き出し: 「showSaveDialog(null)」というメソッドを利用
- ウィンドウ表示のメソッドの戻り値が0の場合が、「OK」を押されたとき
- 選択されたファイルの受け取り: 「getSelectedFile()」メソッドで受け取り

Copyright (C) Atsuhiko Shimamura, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser～読み込み(1)～

- ファイルを読み込む場合

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        JFileChooser chooser = new JFileChooser();
        int code = chooser.showOpenDialog(null);
        if (code == 0) {
            File readFile = chooser.getSelectedFile();
            try {
                FileReader fr = new FileReader(readFile);
                BufferedReader br = new BufferedReader(fr);
                .....
            } catch (IOException ioe) {
            }
        }
    }
}
```

Copyright (C) Atsuhiko Shimamura, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser～読み込み(2)～

- ファイルを読み込む場合

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        JFileChooser chooser = new JFileChooser();
        int code = chooser.showOpenDialog(null);
        if (code == 0) {
            File readFile = chooser.getSelectedFile();
            try {
                FileReader fr = new FileReader(readFile);
                BufferedReader br = new BufferedReader(fr);
                .....
            } catch (IOException ioe) {
            }
        }
    }
}
```

JFileChooserのオブジェクトを作り、「showOpenDialog(null)」メソッドでウィンドウを表示

Copyright (C) Atsuhiko Shimamura, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser～読み込み(3)～

- ファイルを読み込む場合

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        JFileChooser chooser = new JFileChooser();
        int code = chooser.showOpenDialog(null);
        if (code == 0) {
            File readFile = chooser.getSelectedFile();
            try {
                FileReader fr = new FileReader(readFile);
                BufferedReader br = new BufferedReader(fr);
                .....
            } catch (IOException ioe) {
            }
        }
    }
}
```

「showOpenDialog(null)」メソッドの戻り値が「0」のときが、「開く」ボタンを押された場合

Copyright (C) Atsuhiko Shimamura, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser～読み込み(4)～

- ファイルを読み込む場合

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        JFileChooser chooser = new JFileChooser();
        int code = chooser.showOpenDialog(null);
        if (code == 0) {
            File readFile = chooser.getSelectedFile();
            try {
                FileReader fr = new FileReader(readFile);
                BufferedReader br = new BufferedReader(fr);
                .....
            } catch (IOException ioe) {
            }
        }
    }
}
```

JFileChooserで選択されたファイルを受け取り、Fileクラスの「readFile」変数に代入

「File」クラス: Java でファイルを扱うために用意されているクラス

Copyright (C) Atsuhiko Shimamura, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser～読み込み(5)～

### ● ファイルを読み込む場合

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        JFileChooser chooser = new JFileChooser();
        int code = chooser.showSaveDialog(null);
        if (code == 0) {
            File writeFile = chooser.getSelectedFile();
            try {
                FileReader fr = new FileReader(writeFile);
                BufferedReader br = new BufferedReader(fr);
                .....
            } catch (IOException ioe) {
            }
        }
    }
}
```

選択されたファイルの内容を読む処理(FileReaderクラスの  
コンストラクタの引数は、Fileクラスのオブジェクトまたは  
String型のファイル名)

Copyright (C) Akihiro Shimizu, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser～書き出し(1)～

### ● ファイルに書き込む場合

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        JFileChooser chooser = new JFileChooser();
        int code = chooser.showSaveDialog(null);
        if (code == 0) {
            File writeFile = chooser.getSelectedFile();
            try {
                FileWriter fw = new FileWriter(writeFile);
                PrintWriter pw = new PrintWriter(fw);
                .....
            } catch (IOException ioe) {
            }
        }
    }
}
```

Copyright (C) Akihiro Shimizu, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser～書き出し(2)～

### ● ファイルに書き込む場合

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        JFileChooser chooser = new JFileChooser();
        int code = chooser.showSaveDialog(null);
        if (code == 0) {
            File writeFile = chooser.getSelectedFile();
            try {
                FileWriter fw = new FileWriter(writeFile);
                PrintWriter pw = new PrintWriter(fw);
                .....
            } catch (IOException ioe) {
            }
        }
    }
}
```

JFileChooserのオブジェクトを作り、  
「showSaveDialog(null)」メソッドでウィンドウを表示

Copyright (C) Akihiro Shimizu, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser～書き出し(3)～

### ● ファイルに書き込む場合

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        JFileChooser chooser = new JFileChooser();
        int code = chooser.showSaveDialog(null);
        if (code == 0) {
            File writeFile = chooser.getSelectedFile();
            try {
                FileWriter fw = new FileWriter(writeFile);
                PrintWriter pw = new PrintWriter(fw);
                .....
            } catch (IOException ioe) {
            }
        }
    }
}
```

「showSaveDialog(null)」メソッドの戻り値が  
「0」のときに「開く」ボタンを押された場合

Copyright (C) Akihiro Shimizu, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser～書き出し(4)～

### ● ファイルに書き込む場合

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        JFileChooser chooser = new JFileChooser();
        int code = chooser.showSaveDialog(null);
        if (code == 0) {
            File writeFile = chooser.getSelectedFile();
            try {
                FileWriter fw = new FileWriter(writeFile);
                PrintWriter pw = new PrintWriter(fw);
                .....
            } catch (IOException ioe) {
            }
        }
    }
}
```

JFileChooserで選択されたファイルを受け取り、  
Fileクラスの「writeFile」変数に代入

「File」クラス: Javaでファイルを扱うために用意されているクラス

Copyright (C) Akihiro Shimizu, Tokyo Women's Christian University 2016. All rights reserved.

## JFileChooser～書き出し(5)～

### ● ファイルに書き込む場合

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == okBut) {
        JFileChooser chooser = new JFileChooser();
        int code = chooser.showSaveDialog(null);
        if (code == 0) {
            File writeFile = chooser.getSelectedFile();
            try {
                FileWriter fw = new FileWriter(writeFile);
                PrintWriter pw = new PrintWriter(fw);
                .....
            } catch (IOException ioe) {
            }
        }
    }
}
```

選択されたファイルの内容を読む処理(FileReaderクラスのコンストラクタの  
引数は、FileクラスのオブジェクトまたはString型のファイル名)

Copyright (C) Akihiro Shimizu, Tokyo Women's Christian University 2016. All rights reserved.

## やってみよう!(1)

- 「情報入力」ウィンドウに入力した情報を「情報確認」ウィンドウに表示するプログラム



- 「情報入力」ウィンドウに情報を入力し、「OK」を押したら、ファイル選択ウィンドウからファイルを選択し、そのファイルに入力された情報を書き込むプログラム

- 「東京子, 東京都杉並区..., 03-3333-4444, 女性」という形で書き込み

Copyright (C) Akiyo Shimizu, Tokyo Women's Christian University 2019. All rights reserved.

## やってみよう!(2)

- 下記の処理をするプログラム

1. ファイルを1つ選択
  - ファイルの内容は下図のようなもの
2. 1. で選択したファイルの内容を読み込み、1行1行の内容をウィンドウに表示

ファイルの内容の例

```
k17s1001, 東京子, 東京都出身
k17y2030, 善福寺花子, 千葉県出身
k17z3050, 吉祥寺祥子, 埼玉県出身
```

学生番号と氏名、出身地を「,」で区切って

1人1行で表したもの

ファイルの行数は最大100行

※どのようなウィンドウの構成にすれば良いかも自分で考えること

Copyright (C) Akiyo Shimizu, Tokyo Women's Christian University 2019. All rights reserved.

## 補講と期末試験のお知らせ

- 期末試験: 1月29日(火) 2限 9301教室

- 時間: 90分
- 持ち込み: 全て可
- 内容: 後期の講義内容すべて
  - 用語の意味の選択・説明
  - 概念に関する説明
  - 実技

Copyright (C) Akiyo Shimizu, Tokyo Women's Christian University 2019. All rights reserved.