

# プログラム設計

<http://bit.ly/design4d>

## GUI (1)

前期 第11週

2019/7/3

# 本日は

**AWT**を利用してGUIアプリケーションの  
**外観**を作ってみます。

# AWT

Javaに標準で用意されている

GUIの部品（GUIコンポーネント）のためのクラスライブラリ

## Java向けGUIの種類

- ▶ AWT → 機能がシンプルで軽量
- ▶ Swing → AWTよりも多機能だが重い
- ▶ SWT → 多機能・軽量だがOSに依存する

# 今回扱うGUIコンポーネント

- ▶ ウィンドウ（フレーム）
- ▶ ラベル
- ▶ ボタン
- ▶ テキストフィールド
- ▶ 色

# 【課題の準備】

演習室で作業する前に、以下のコマンドを  
入れるだけで準備が完了する

```
$ mygitclone4d 「自分のGitHubユーザ名」  
$ cd prog4d-(ユーザ名)  
$ ./myconf
```

※本体をシャットダウンするまでは、  
上記「mygitclone」と「myconf」の設定は有効です

# 【課題の準備】

以下の流れで、課題のプログラムを作るためのフォルダを準備しましょう。

1. 端末を起動して、以下のコマンドを実行して前期第10週のフォルダを作る  
\$ cd prog4d-(ユーザ名) (←既に移動しているなら不要)  
\$ mkdir week111  
\$ cd week111

※課題で作るファイル名は各自で決めて構いません。

# 今回扱うGUIコンポーネント

- ▶ ウィンドウ (フレーム)
- ▶ ラベル
- ▶ ボタン
- ▶ テキストフィールド
- ▶ 色

# 「1\_11\_MyFrame.java」を実行してみましょう

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

```
class MyFrame {  
    private Frame f1;  
    public MyFrame() {
```

ウィンドウ（フレーム）のインスタンス  
を作る

```
        f1 = new Frame("フレーム");
```

フレームの幅と高さを指定する

```
        f1.setSize(200, 100);
```

```
        f1.setVisible(true);
```

- true → フレーム表示
- false → フレーム非表示

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



# 赤枠部分を追加して「ラベル」を配置してみましよう

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

```
class MyFrame {  
    private Frame f1;  
    private Label l1;  
    public MyFrame() {  
        f1 = new Frame("フレーム");
```

ラベルl1（小文字のエルと数字の1）  
を作ってフレームに追加する

```
        l1 = new Label("ラベル1");  
        f1.add(l1, BorderLayout.NORTH);
```

```
        f1.setSize(200, 100);  
        f1.setVisible(true);
```

この代わりに、「f1.pack()」とすると、中に配置された部品大きさに合わせてf1の大きさが設定される

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

# レイアウトマネージャ

フレーム内の部品の配置（レイアウト）方法を管理している  
（レイアウトの種類については後程）

## 【先程の例】

```
f1.add(l1, BorderLayout.NORTH);
```

ラベルl1をフレームf1に追加する

レイアウトマネージャ：BorderLayout

- BorderLayout.NORTH → フレーム上に配置
- BorderLayout.SOUTH → フレーム下に配置
- BorderLayout.EAST → フレーム右に配置
- BorderLayout.WEST → フレーム左に配置

# 今回扱うGUIコンポーネント

▶ ウィンドウ（フレーム）

▶ ラベル

▶ ボタン

▶ テキストフィールド

▶ 色

# 赤枠部分を追加して、更に「ラベル」を配置してみましよう

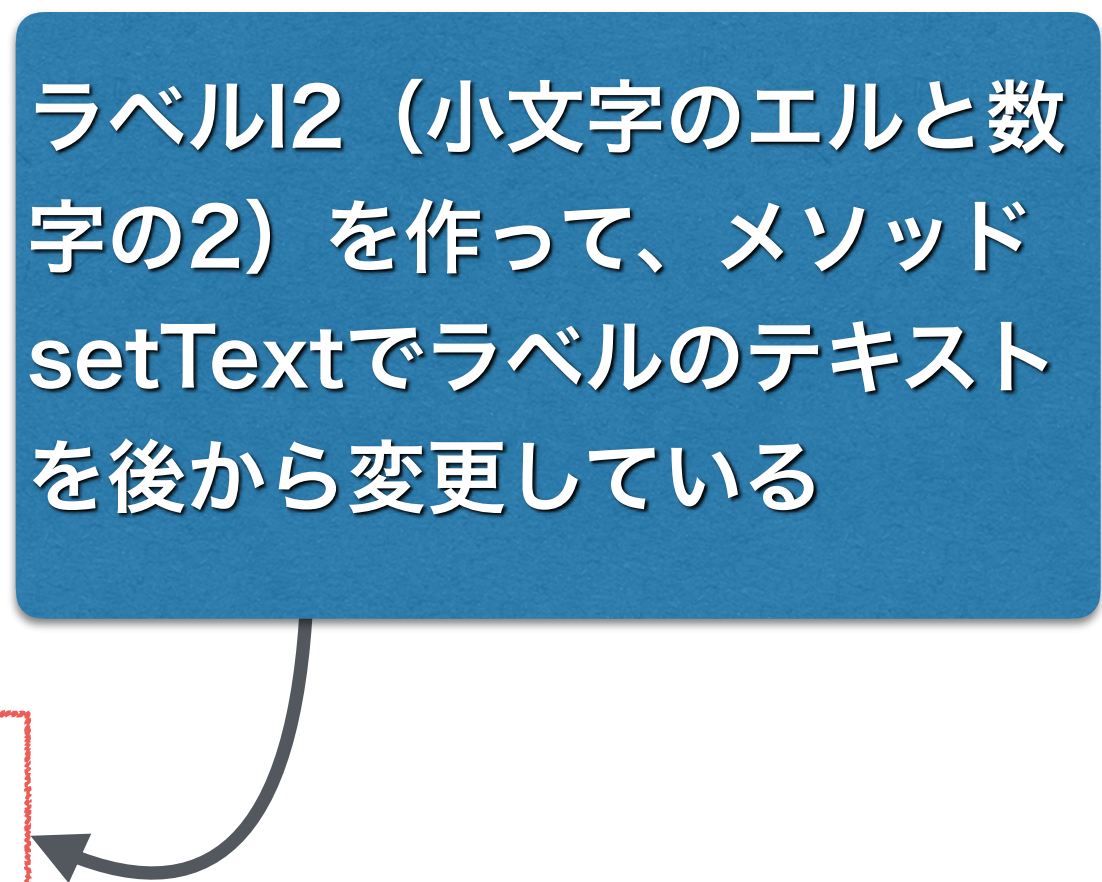
```
import java.awt.*;
import java.awt.event.*;

class MyFrame {
    private Frame f1;
    private Label l1, l2;
    public MyFrame() {
        f1 = new Frame("フレーム");

        l1 = new Label("ラベル1");
        f1.add(l1, BorderLayout.NORTH);
        l2 = new Label();
        f1.add(l2, BorderLayout.SOUTH);
        l2.setText("ラベル2");

        f1.setSize(200, 100);
        f1.setVisible(true);
    }
    public static void main(String[] args) {
        MyFrame obj = new MyFrame();
    }
}
```

ラベルl2（小文字のエルと数字の2）を作って、メソッドsetTextでラベルのテキストを後から変更している



# Labelの主なメソッド

▶ void setText(String text)

… ラベルに文字列textを**設定する**

【例】

```
String str = “ラベル2”;
```

```
l1.setText(str);
```

```
//ラベルl1にstrの文字列を表示する
```

▶ String getText()

… ラベルのテキストを**取得する**

【例】

```
String str = l1.getText();
```

```
//ラベルl1に表示されているテキストを取得して
```

```
//strに代入する
```

# 今回扱うGUIコンポーネント

- ▶ ウィンドウ（フレーム）
- ▶ ラベル
- ▶ ボタン
- ▶ テキストフィールド
- ▶ 色

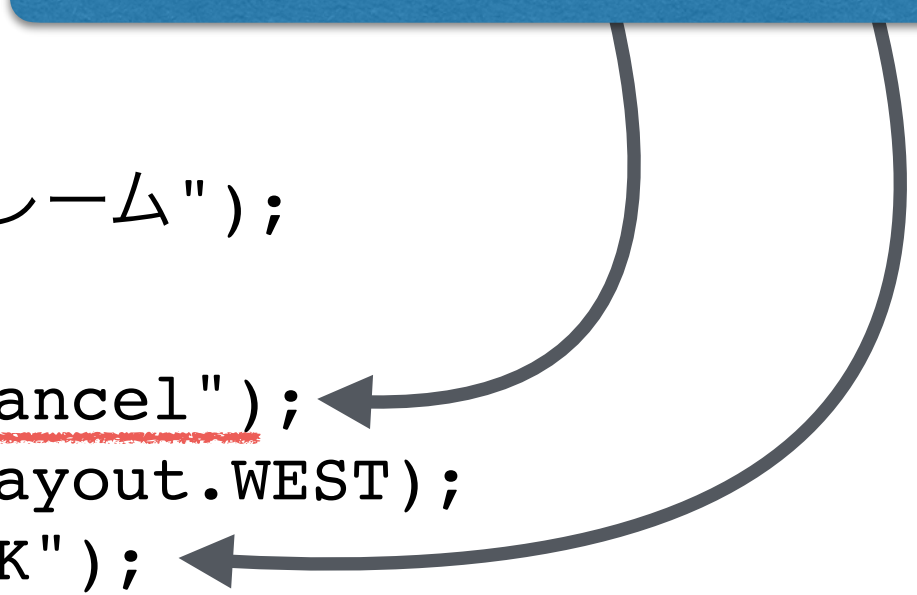


# 「1\_11\_MyButton.java」を実行してみましょう

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

```
class MyButton {  
    private Frame f1;  
    private Button b1, b2;  
    public MyButton() {  
        f1 = new Frame("フレーム");  
  
        b1 = new Button("Cancel");  
        f1.add(b1, BorderLayout.WEST);  
        b2 = new Button("OK");  
        f1.add(b2, BorderLayout.EAST);  
  
        f1.setSize(200, 100);  
        f1.setVisible(true);  
    }  
    public static void main(String[] args) {  
        MyButton obj = new MyButton();  
    }  
}
```

コンストラクタの引数に指定した文字列  
が表示されたボタンが作られる  
(メソッドsetTextで後から変更も可能)



# 今回扱うGUIコンポーネント

- ▶ ウィンドウ（フレーム）
- ▶ ラベル
- ▶ ボタン
- ▶ テキストフィールド
- ▶ 色



# 「1\_11\_MyTField.java」を実行してみましょう

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

```
class MyTField {  
    private Frame f1;  
    private TextField tf1, tf2, tf3;  
    public MyTField() {  
        f1 = new Frame("フレーム");  
  
        tf1 = new TextField();  
        f1.add(tf1, BorderLayout.NORTH);  
        tf2 = new TextField("入力して下さい");  
        f1.add(tf2, BorderLayout.CENTER);  
        tf3 = new TextField(50);  
        f1.add(tf3, BorderLayout.SOUTH);  
  
        f1.pack();  
        f1.setVisible(true);  
    }  
    public static void main(String[] args) {  
        MyTField obj = new MyTField();  
    }  
}
```

コンストラクタの引数に指定した文字列が  
入力されたテキストフィールドが作られる  
(メソッドsetTextで後から変更も可能)

コンストラクタの引数に  
整数を指定すると、その文  
字数の幅の長さになる

# TextFieldの主なメソッド

## ▶ void setText(String text)

… テキストフィールドに文字列textを**設定する**

【例】

```
String str = “テキストフィールド2”;
```

```
tf1.setText(str);
```

```
//テキストフィールドtf1にstrの文字列を表示する
```

## ▶ String getText()

… テキストフィールドに入力されているテキストを**取得する**

【例】

```
String str = tf1.getText();
```

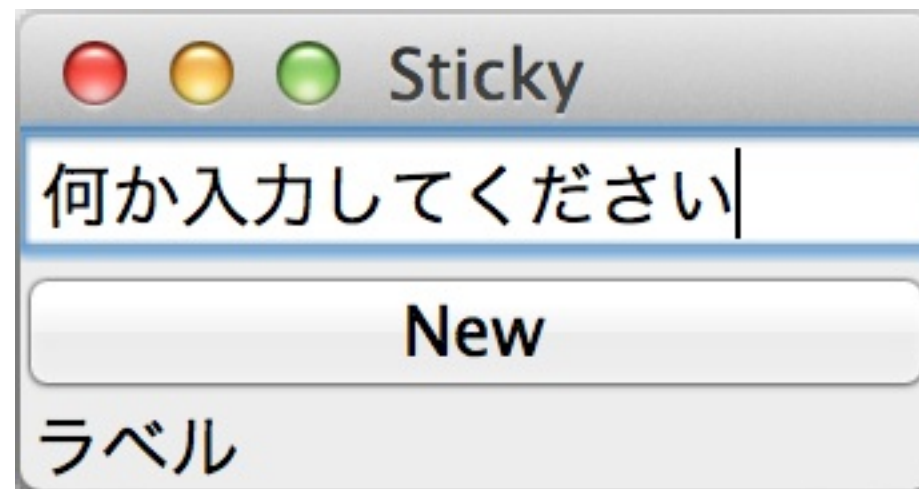
```
//テキストフィールドtf1に入力されている
```

```
//テキストを取得してstrに代入する
```

# 【課題11-1】

フレームの中に「テキストフィールド」「ボタン」「ラベル」が作られたウィンドウが表示されるアプリケーションを作ってみましょう。

## 完成図



# 【課題の提出】

以下の流れで、作ったCプログラムをGitHubにプッシュして、Webサイトで確認してみましょう。

1. 端末内で、以下のコマンドで課題を提出

```
$ git add -A
```

```
$ git commit -m “課題11-1提出”
```

```
$ git push origin master
```

2. 自分のリポジトリを開いて、提出したファイルがプッシュされているか確認する

[https://github.com/nit-ibaraki-prog4d-2019/prog4d-\(ユーザー名\)](https://github.com/nit-ibaraki-prog4d-2019/prog4d-(ユーザー名))

# 今回扱うGUIコンポーネント

- ▶ ウィンドウ（フレーム）
- ▶ ラベル
- ▶ ボタン
- ▶ テキストフィールド
- ▶ 色

# 色の種類

white, black, gray, blue, green, redなどいくつかの色は、  
クラスColorの変数としてあらかじめ定義されている

- ▶ Color.white → 白
- ▶ Color.black → 黒
- ▶ Color.blue → 青
- ▶ など

# 色の設定

ほとんどのGUIコンポーネントは、  
**文字色**と**背景色**を設定することができる

- ▶ 文字色を変更するメソッド → `setForeground`
- ▶ 背景色を変更するメソッド → `setBackground`

## 【例】

```
Label l = new Label("色付きラベル");  
l.setForeground(Color.white); //ラベルの文字色を白にする  
l.setBackground(Color.blue);  //ラベルの背景色を青にする
```

# 【課題11-2】

課題11-1で作成した「テキストフィールド」「ボタン」「ラベル」の文字色と背景色をそれぞれ好きな色に変更してみましょう。

色の種類は、以下のAPIドキュメントから、クラス「`java.awt.Color`」を検索して調べてみましょう。

<https://docs.oracle.com/javase/jp/11/docs/api/index.html>



# 【課題の提出】

以下の流れで、作ったCプログラムをGitHubにプッシュして、Webサイトで確認してみましょう。

1. 端末内で、以下のコマンドで課題を提出

```
$ git add -A
```

```
$ git commit -m “課題11-2提出”
```

```
$ git push origin master
```

2. 自分のリポジトリを開いて、提出したファイルがプッシュされているか確認する

[https://github.com/nit-ibaraki-prog4d-2019/prog4d-\(ユーザー名\)](https://github.com/nit-ibaraki-prog4d-2019/prog4d-(ユーザー名))