

プログラミングII

<http://bit.ly/Prog3i>

GUI (2)

後期 第11週

2019/12/11

本日は

AWTを利用してGUIアプリケーションの
イベントによる動作を作ってみます。

イベントの仕組み

【例】

ボタンを押したら「スティッキー（付箋）」を新規作成する

イベント

「New」ボタンを押す

Sticky

ミルクを買う

New

新規作成

イベントソース

「New」ボタンで
イベントが発生した

イベントリスナ

監視しているイベントが発生したら以下の処理をする

1. Stickyのフレームを作る
2. テキストフィールドの文字列を取得する
3. その文字列をラベルにセットする

イベントリスナの実装

リスナ (Listener) と呼ばれる インタフェース を使う

抽象メソッド (処理の定義がされていない) の集まり

イベントに関する主なリスナ

- ▶ ActionListener … ボタン、メニューの選択等
- ▶ WindowListener … ウィンドウを閉じる、最小化等
- ▶ MouseListener … マウスのクリック、ドラッグ等
- ▶ KeyListener … キーを押す等

【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

```
$ mygitclone 「自分のGitHubユーザ名」  
$ cd prog3i-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後期第11週のフォルダを作る
\$ cd prog3i-ユーザ名 (←既に移動しているなら不要)
\$ mkdir week211
\$ cd week211

「2_11_MyFrame2.java」を開きましょう

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

class MyFrame2 {
    Frame f1;
    Button b1;
    public MyFrame2() {
        f1 = new Frame("フレーム");

        b1 = new Button("Exit");
        f1.add(b1, BorderLayout.CENTER);
        f1.pack();
        f1.setVisible(true);
    }
    public static void main(String[] args) {
        MyFrame2 obj = new MyFrame2();
    }
}
```

「2_11_MyFrame2.java」 に次のクラスを追加しましょう

ActionListenerというインタフェースを実装する

```
class MyListener implements ActionListener {  
    public void actionPerformed(ActionEvent e) {  
        System.out.println("押されたボタン名: "  
                            + e.getActionCommand());  
        System.out.println("終了します。");  
        System.exit(0);  
    }  
}
```

ボタンのテキストを取得できる

アプリケーションを終了する

「ボタンを押す」というイベント発生時に呼び出されるメソッド

「2_11_MyFrame2.java」の赤線部分を追加しましょう

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

```
class MyFrame2 {
```

```
    Frame f1;
```

```
    Button b1;
```

```
    public MyFrame2() {
```

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

```
        b1 = new Button("Exit");
```

```
        f1.add(b1, BorderLayout.CENTER);
```

```
        b1.addActionListener(new MyListener());
```

```
        f1.pack();
```

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

```
    }
```

```
    public static void main(String[] args) {
```

```
        MyFrame2 obj = new MyFrame2();
```

```
    }
```

```
}
```

ボタンb1に、リスナとしてMyListener
のインスタンスを登録する



【練習11-1】

「2_11_MyFrame2.java」でボタンを押したら、アプリケーションが終了するのを確認しましょう。

「2_11_MyFrame3.java」 でリスナとソースをまとめた例

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

class MyFrame3 implements ActionListener {
    Frame f1;
    Button b1;
    public MyFrame3() {
        f1 = new Frame("フレーム");

        b1 = new Button("Exit");
        f1.add(b1, BorderLayout.CENTER);
        b1.addActionListener(this); // thisは自身を指すキーワード

        f1.pack();
        f1.setVisible(true);
    }
    public void actionPerformed(ActionEvent e) {
        System.out.println("押されたボタン名: " + e.getActionCommand());

        System.out.println("終了します。");

        System.exit(0);
    }
    public static void main(String[] args) {
        MyFrame3 obj = new MyFrame3();
    }
}
```

【練習11-2】

**「2_11_MyFrame3.java」でボタンを押したら、アプリケーションが終了するのを確認しましょう。
(動作は練習11-1と同じ)**

【課題11-1】

練習11-2の「Exit」を「Dark」に変更し、以下の処理をするように動作を付けて下さい。

▶ 「Dark」ボタンを押すと、
全てのボタンの文字色が白、背景色を黒にする

【課題の提出】

以下の流れで、作ったプログラムを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-prog3i/prog3i-\(ユーザ名\)](https://github.com/nit-ibaraki-prog3i/prog3i-(ユーザ名))

【課題11-2】

フレームのCENTERに、テキストフィールドを配置し、「Enter」キーを押すと、入力した文字を標準出力に表示するアプリケーションを作ってください。

- ▶ テキストフィールドに対して…
 - ▶ 「Enter」キーを押すとactionPerformedが実行される
 - ▶ テキストフィールドのgetTextを実行すると入力したテキストが取得できる

【課題の提出】

以下の流れで、作ったプログラムを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-prog3i/prog3i-\(ユーザー名\)](https://github.com/nit-ibaraki-prog3i/prog3i-(ユーザー名))

【課題12-3】

フレームに「Exit」と「Dark」を配置し、以下の処理をするように動作を付けて下さい。

- ▶ 「Dark」ボタンを押すと、
全てのボタンの文字色が白、背景色を黒にする
- ▶ 「Exit」ボタンを押すと、
アプリケーションが終了する
- ▶ 押したボタンがどれかを判別するには以下のように比較する
`if(e.getActionCommand().equals("Dark"))`

【課題の提出】

以下の流れで、作ったプログラムを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11-3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-prog3i/prog3i-\(ユーザ名\)](https://github.com/nit-ibaraki-prog3i/prog3i-(ユーザ名))