第9章　アプレット

【学習内容とねらい】

本章では、Java 言語で作ったプログラムを Web ブラウザ上で動作させる方法を学習します。Java 言語には、これまで作成してきた Windows アプリケーションの他に、Web ブラウザ上で動作させる事のできる**アプレット**という形態があります。このアプレットを利用すれば、Web 上で Java プログラムを公開することもできます。

アプレットは Java 言語の普及当初は（Java 言語の機能の中で）最も注目された機能で、当時は「アプレットによって Web ページは変わる！」と大いにもてはやされたものです。現在では、その”熱狂”は醒めたものの、今でも Java アプレットが幅広く用いられていることに変わりはありません。特に、Web 上の学習教材の開発などには、今でも Java アプレットが多く活用されています。

さて、Web ブラウザ上で動作させるプログラムと言うと難しく聞こえるかも知れませんが、特殊な用途を除いては、これまでのアプリケーションの作成とほとんど変わることはありません。端的に言えば、これまでのフレームがアプレットに変わっただけです。ですから、本章を学習すれば、”簡単にアプレットを作成することができる”ということを体験できるはずです。上に述べたとおり、自作のアプレットを自分のホームページで公開する、と言うことも可能になります。興味のある人はチャレンジしてみて下さい。

＜第9章の構成＞

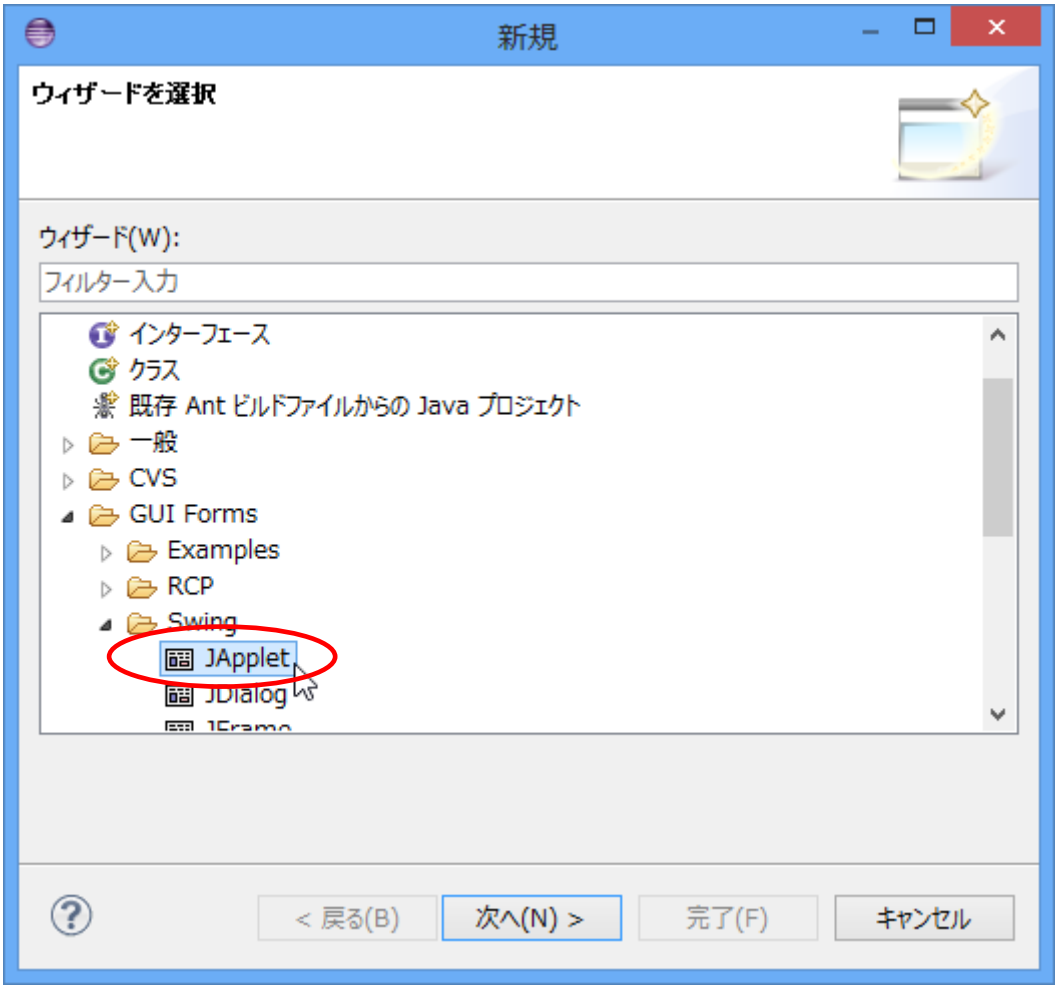
9-1　アプレットの作成の仕方
9-2　Web ブラウザ上でのアプレットの実行の仕方
9-3　アプレット作成の練習

9-1 アプレットの作成・実行

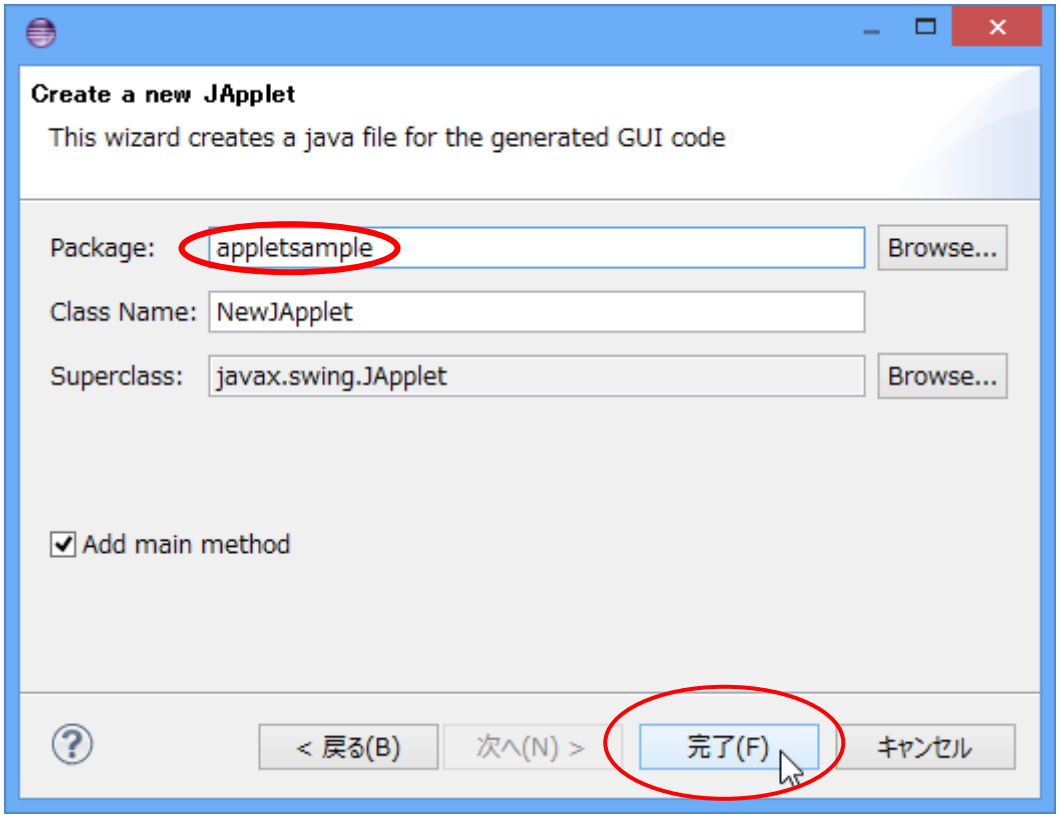
【練習課題】

Eclipse を用いてアプレットを作成する場合、その工程はこれまで学習した通常のアプリケーションの作成とほとんど変わりません。まず、簡単なサンプル（アプレット）を作ってみましょう。

いつも通り、Java プロジェクトを作成します。ここでは、プロジェクト名を「AppletSample」としました。そして、同プロジェクト内にアプレットを新規作成するために、[新規]→［その他］を選択し、下の様にテンプレート（ひな形）として「JApplet」を選択します。



［次へ］ボタンをクリックして現れる次の画面でパッケージ名を指定します。ここでは、これまで通り次の様にプロジェクト名を小文字で表し「appletsample」としました（Java プログラムでは、パッケージ名を小文字にするのが慣例です）。



パッケージ名指定後、［完了］ボタンをクリックすると設定が完了します。これで、アプレットのひな形（具体的には、NewJApplet.java というプログラム）が作成されました。

ここで「NewJApplet.java」のソースを見ると、次のようになっているはずです。

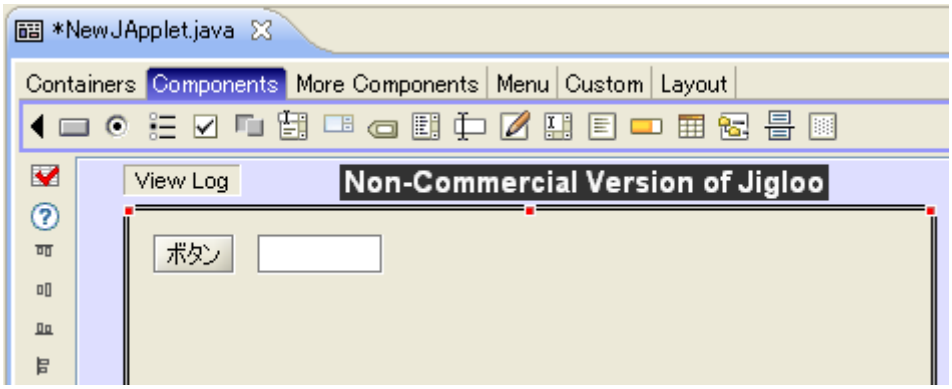
```
package appletsample;

import java.awt.Dimension;

import javax.swing.JApplet;    ①
import javax.swing.JComponent;
import javax.swing.JFrame;
import javax.swing.SwingUtilities;
                                                    ②
public class NewJApplet extends javax.swing.JApplet {
      /**
      * Auto-generated main method to display this
      * JApplet inside a new JFrame.
      */
      public static void main(String[] args) {
            . . .
      }
        . . .
```

＜解説＞

① 今の場合、アプレットの作成なので、（アプレット作成に必要なクラスが用意されている）javax.swing.JApplet パッケージをインポートしています。

② JFrame の代わりに「JApplet」が入っています。これは、アプレット作成の場合は、JFrame クラスではなく、JApplet クラス（のオブジェクト）の上に色々なコンポーネントを貼り付けてプログラムを作成する事を意味しています。逆に言えば、これまで学習したプログラムとアプレットプログラムの違いは、**フレームがアプレットに変わっただけ**、という事になります。

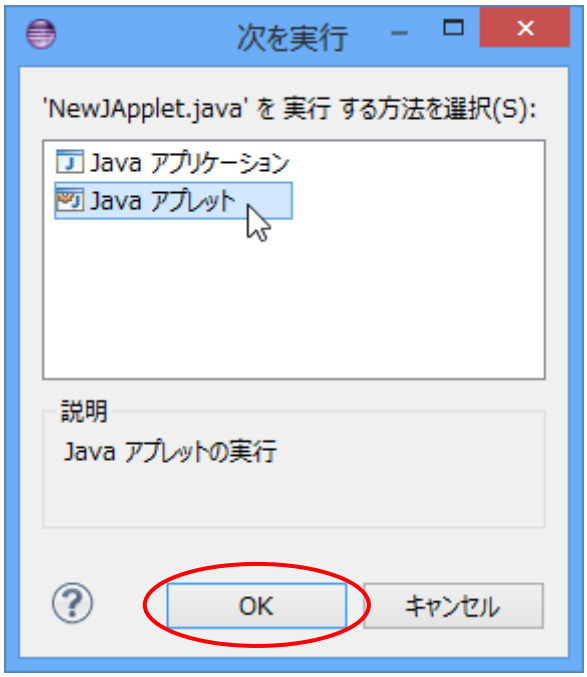
それでは、ボタンをクリックするとメッセージが表示される、という簡単なアプレットを作ってみましょう。アプレットへのコンポーネントの配置の仕方は、フレームの場合と全く同様です。次の様に［ボタン］コンポーネントとテキストフィールドコンポーネントを配置して下さい。



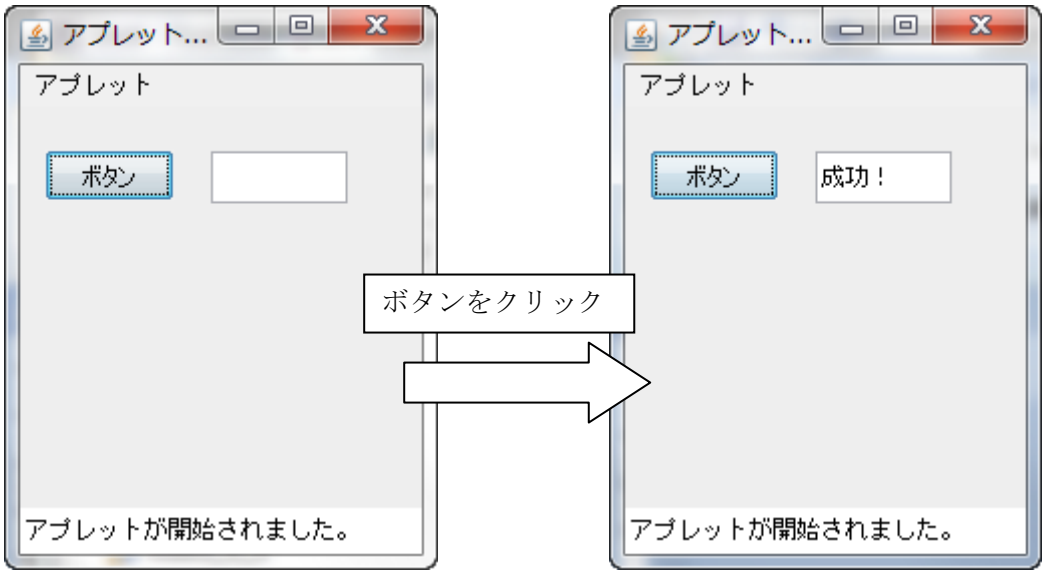
さらに、［ボタン］をクリックしたときに、テキストフィールドに「成功！」と表示されるようにしましょう。そこで、［ボタン］クリック時のイベントハンドラを次の様に記述します。

```
private void jButton1ActionPerformed(ActionEvent evt) {
        jTextField1.setText("成功！");
}
```

作成したら、アプリケーションの場合と同じように実行してみましょう。実行すると、次のような画面が現れるはずです。



ここで、「Java アプレット」を選択し、［OK］ボタンをクリックすると、下の様なアプレットビューアが起動し、実行結果を確認できます。

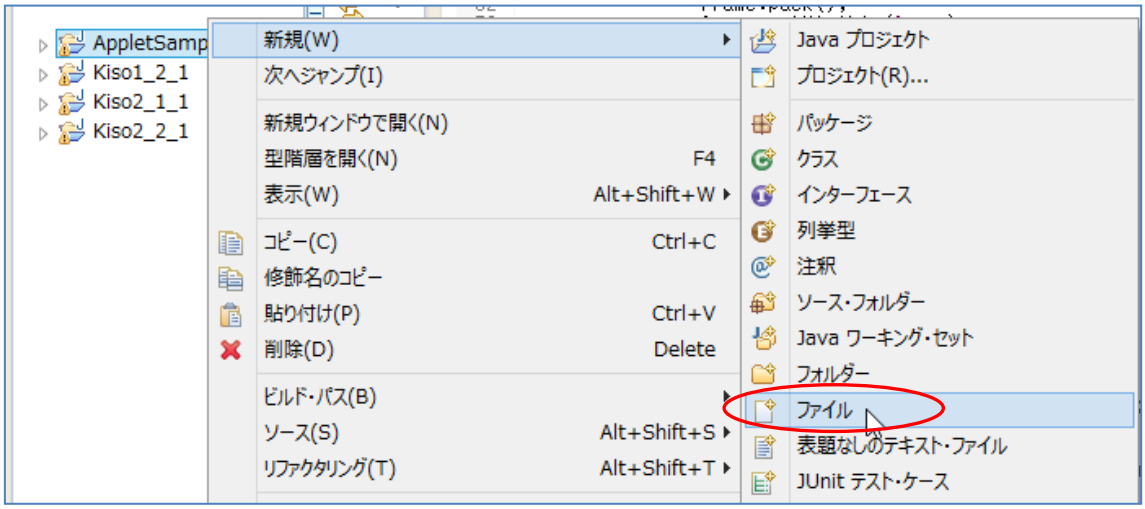


以上は、作成したプログラムの動作確認なのですが、これでは（通常の）アプリケーションの場合と変わらず、まだ Web ブラウザ上で動作させてはいません。そこで、次節でアプレットを Web ブラウザから呼び出す方法を確認しておきましょう。

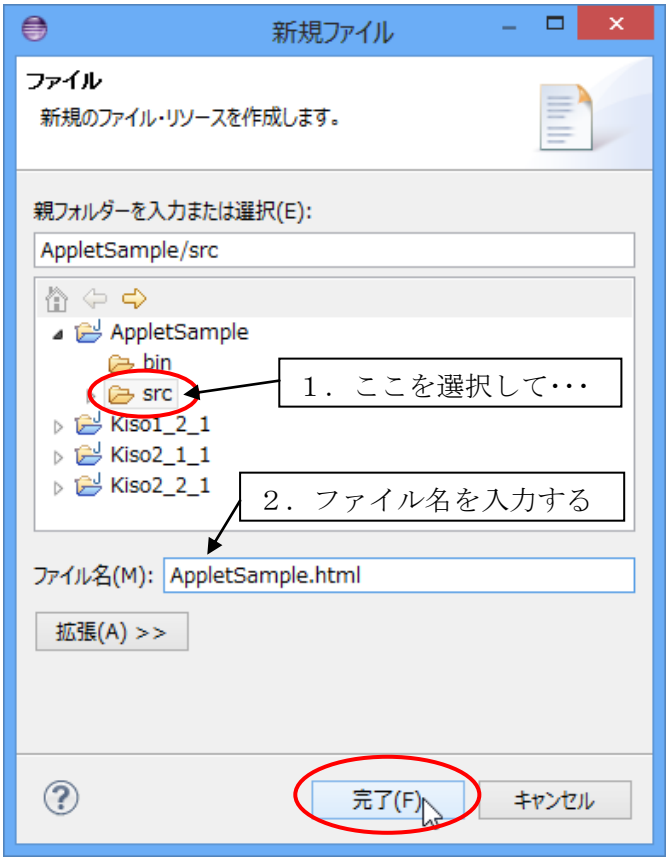
9-2 Web ブラウザ上でのアプレットの実行の仕方

アプレットは Web ブラウザ上で動作させる事ができます。というよりも、それが本来の用い方です。そのためには、アプレットを呼び出す HTML ファイルを作成する必要があります。次の要領で作成して下さい。

① パッケージエクスプローラから、作成したプロジェクト「AppletSample」を選択し右ボタンクリックします。そして、「新規」→「ファイル」と選択します。

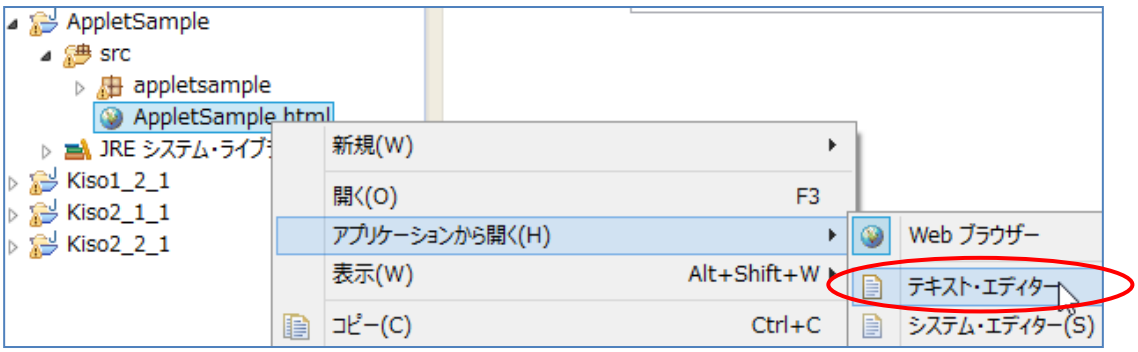


② 次の新規ファイル作成画面で、親フォルダ（ファイルを新規作成する場所）として、プロジェクト内の「src」フォルダを選択し、続いて（作成する HTML）ファイル名を指定します。ここでは、プロジェクトと同名としました。入力後、[完了] ボタンをクリックするとエディタ画面に戻ります。



③　すると、エディタ画面が次のように開かれています。何も表示されていませんが、これは（新規作成したファイルが html 文書であるので）Eclipse のエディタが Web ブラウザとして自動起動したためです。これでは編集できませんので、いったん、「AppletSample.html」タグを閉じます。

その後、下のように「テキストエディター」を用いて改めて開きます。これで、編集可能となりました。



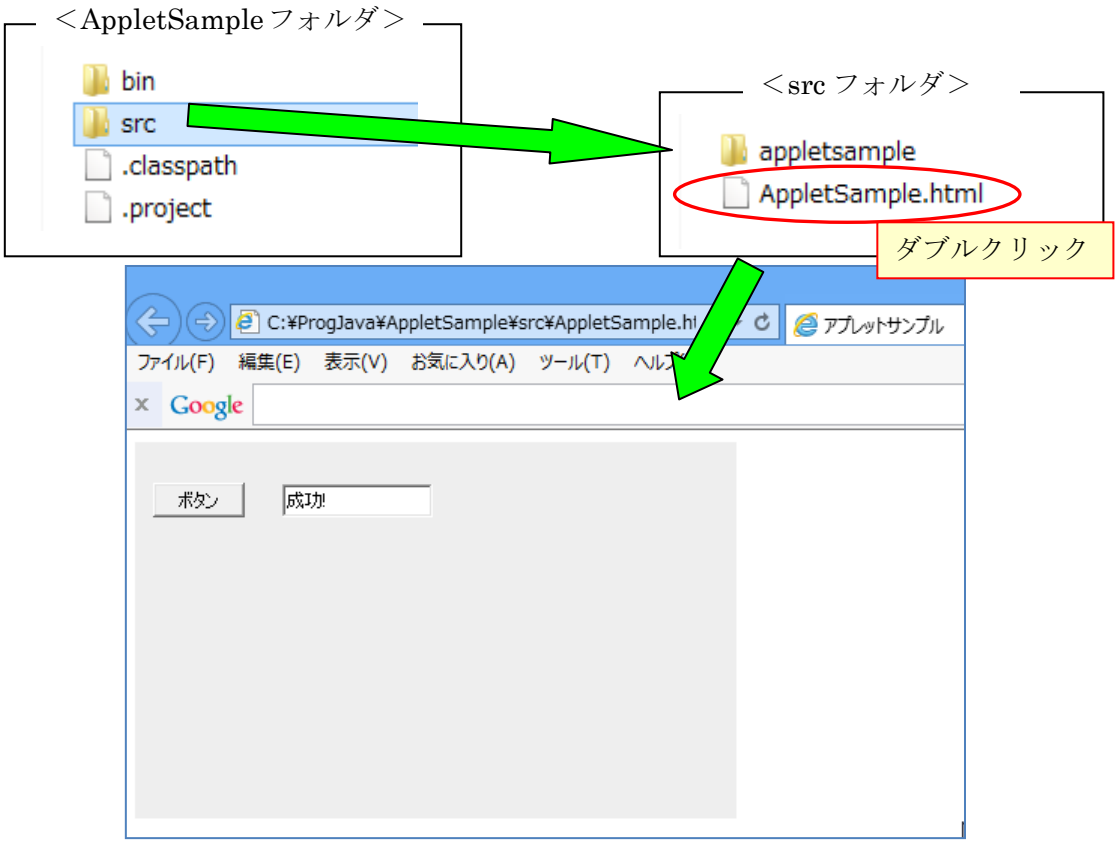
④　ファイルの中身（HTML 文書）は次のように記述します。

```
<html>
<head>
     <meta http-equiv="Content-Type" content="text/html;
                  charset=windows-31j">
  <title> アプレットサンプル </title>
</head>
<body>
  <applet
     codebase="..¥bin¥"
     code="appletsample.NewJApplet.class"
     width="400"
     height="250" >
  </applet>
</body>
</html>
```

Web ページのタイトル

アプレット呼び出しタグ

⑤　記述が完了したら、作成した html ファイル「AppletSample.html」を保管します。その後、ワークスペース内（指定通りであれば C:¥ProgJava）の「AppletSample」フォルダを開いてください。その中の「src」フォルダ内にある「AppletSample.html」をダブルクリックすると、ブラウザが起動し、下のようにアプレットが Web ブラウザに表示されます。



もし、ここで、下の画面が現れアプレットが表示されない場合は、［OK］ボタンをクリックしてください。そして次項の「アプレットが実行されない場合」に進んでください。

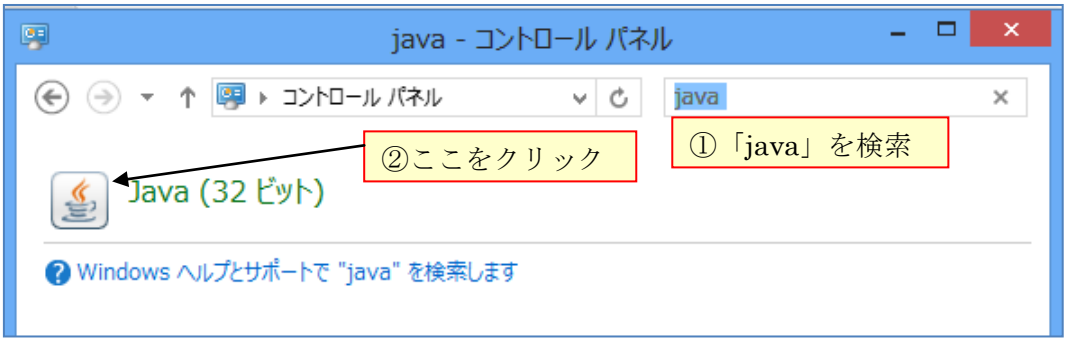
アプレットが実行されない場合

- 前ページの画面は、アプレットが Java 環境のセキュリティによってブロックされたことを意味します。これは、Web 上で不正な処理を行うプログラムが動作しないよう、Java 環境のセキュリティレベルが上げられたため起きたものです。
- ここでは、とりあえずアプレットの実行を確認するために、一時的に Java のセキュリティレベルを下げましょう。以下の手順にしたがってください。

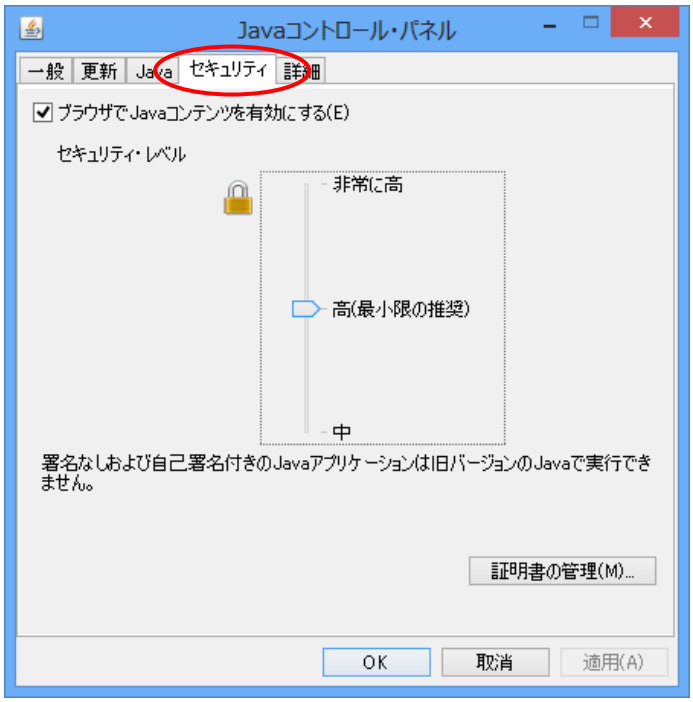
＜Java セキュリティレベルの変更の仕方＞

① コントロールパネルを開いてください。

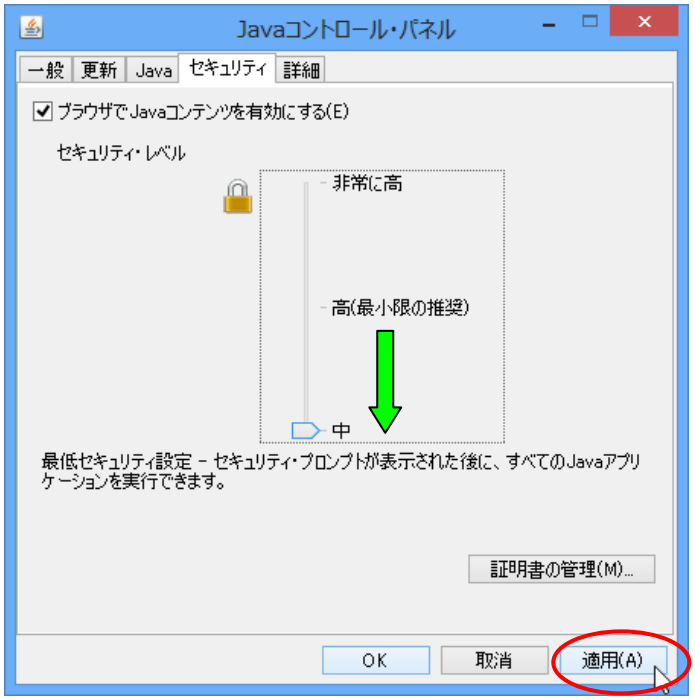
② その検索窓から「java」を検索してください。そうして現れた Java アイコンをクリックします。



③ すると、次のような「Java コントロールパネル」が現れるので、ここで、「セキュリティ」タブを選択します。



④　そして、セキュリティレベルを「中」に下げ、［適用］ボタンをクリックします。



⑤　この状態のままで、p.244 の⑤の処理を行って下さい。途中で「このアプリケーションを実行しますか。」という確認画面が出てきますが、ここで「･･･実行します」を選択して［実行］ボタンをクリックすると、アプレットが実行されるはずです。

⑥　動作の確認が終わったら、Java のセキュリティレベルを「高」に戻しておいてください。

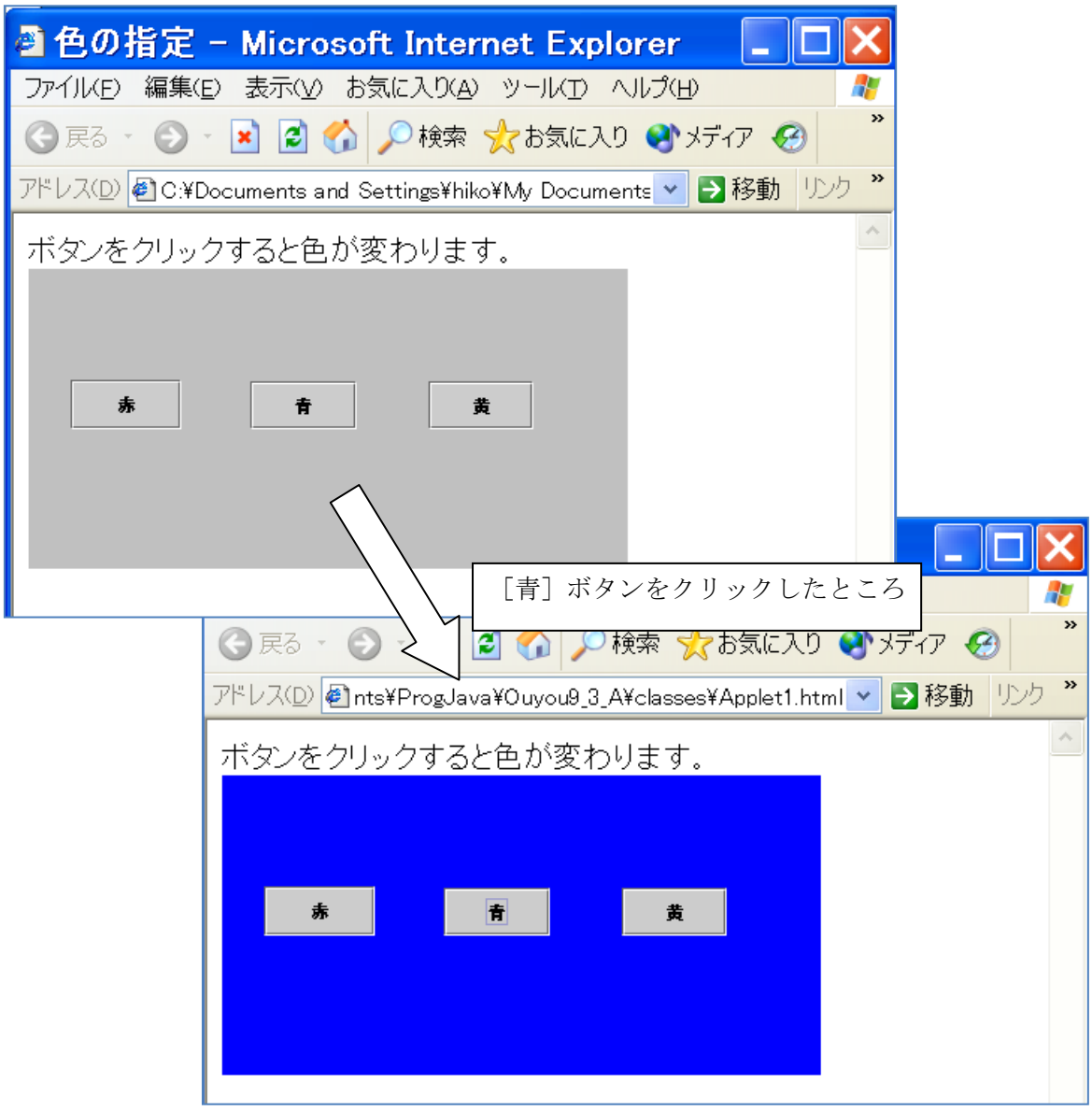
Java のセキュリティレベルを下げずにアプレットを動作可能とするためには、アプレットに制作者の電子署名をつける必要があります。その方法は付録 C を参照してください。

9-3　アプレット作成の練習

前節までに学習した通り、アプレットを利用したプログラムは、フレームがアプレットに変わった事をのぞけば、その作成方法はこれまでのアプリケーションとほとんど同じです。ですから、これまで作ったプログラムをアプレットに変更することは難なくできるはずです。実際に、２題ほど課題プログラムでアプレットを作ってみましょう。

【応用課題 9-3-A】

【基礎課題 3-3-3】で作った、「ボタンをクリックすればフレームの色が変わる」プログラムをアプレットとして作り、下のようにブラウザ上で動作させて下さい。



ヒント

フレームの色を指定する場合には、次のように contentPane コンポーネントの background プロパティを変えました（下は赤色に指定した場合）。→ p.62 参照

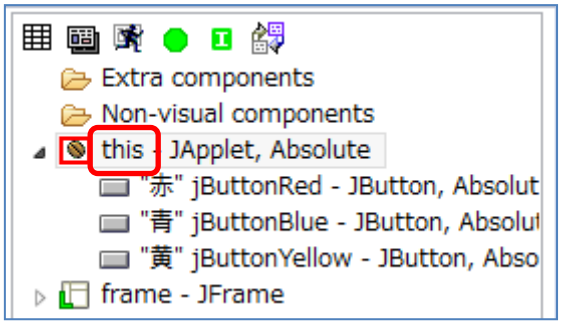
```
getContentPane().setBackground(Color.red);
```

一方、アプレットの場合はアプレットの background プロパティを直接指定できます。ところが、よく考えてみると（今記述しようとしている）ボタンクリックのイベントハンドラは、アプレットクラス NewJApplet の中のメソッドです。ですから、そのクラスに属するメソッドの中から、

```
（NewJApplet クラスのオブジェクト）.setBackground(Color.red);
```

と NewJApplet クラスから生成されたオブジェクト名を指定することはできません。なぜなら、クラス定義時点ではオブジェクト名は未定だからです。では、どうすればよいのでしょうか？

ここで、エディタ画面を GUI Editor 画面に切り替え、アプレット自身が選択された状態にしてみて下さい。そして「アウトライン」ビューを見ると、下のようにその名前（name）は **this** となっていることが分かるでしょう。



このように **this** 変数は、今定義しているクラス内で自分自身を指す場合に用いられます。（より正確に言うと当該クラスのオブジェクトを指します）。ですから、例えば［赤］ボタンのイベントハンドラは

```
void jButtonRedActionPerformed(ActionEvent evt) {
    this.setBackground(Color.red);
}
```

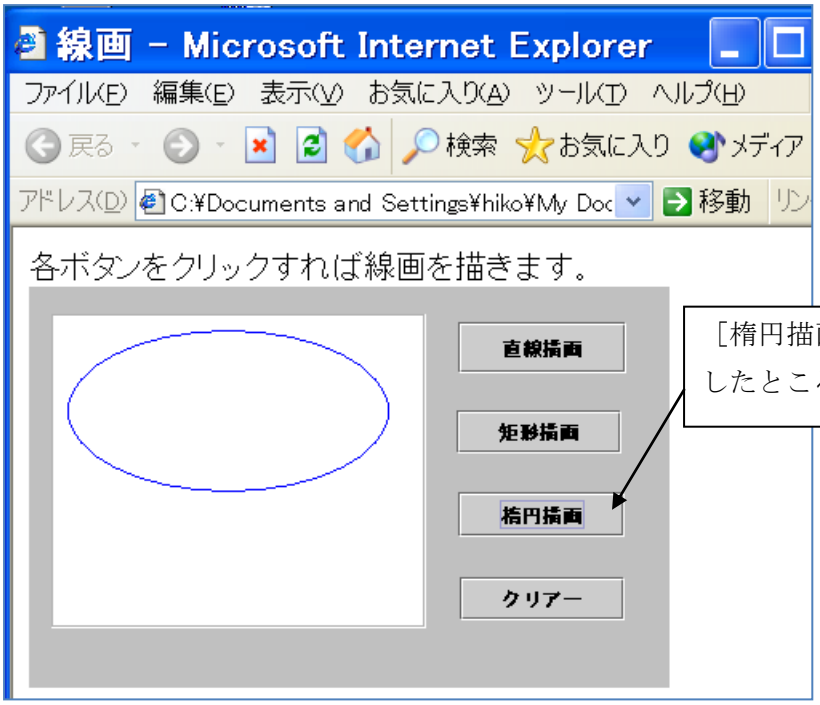
と記述すれば良いのです。その他のボタンのイベントハンドラについても同様です。なお、この **this** に対して、継承元のクラス（つまりスーパークラス）を指す変数が 7-4 節（p.194

～195）で学習した **super** です。対にして覚えておくと良いでしょう。

【応用課題 9-3-B】

今度は、8-3 節の【練習課題】で作成したプログラムを、下のようにアプレットとして作成して下さい。

アプレットをブラウザで開いたところ



［楕円描画］ボタンをクリックしたところ