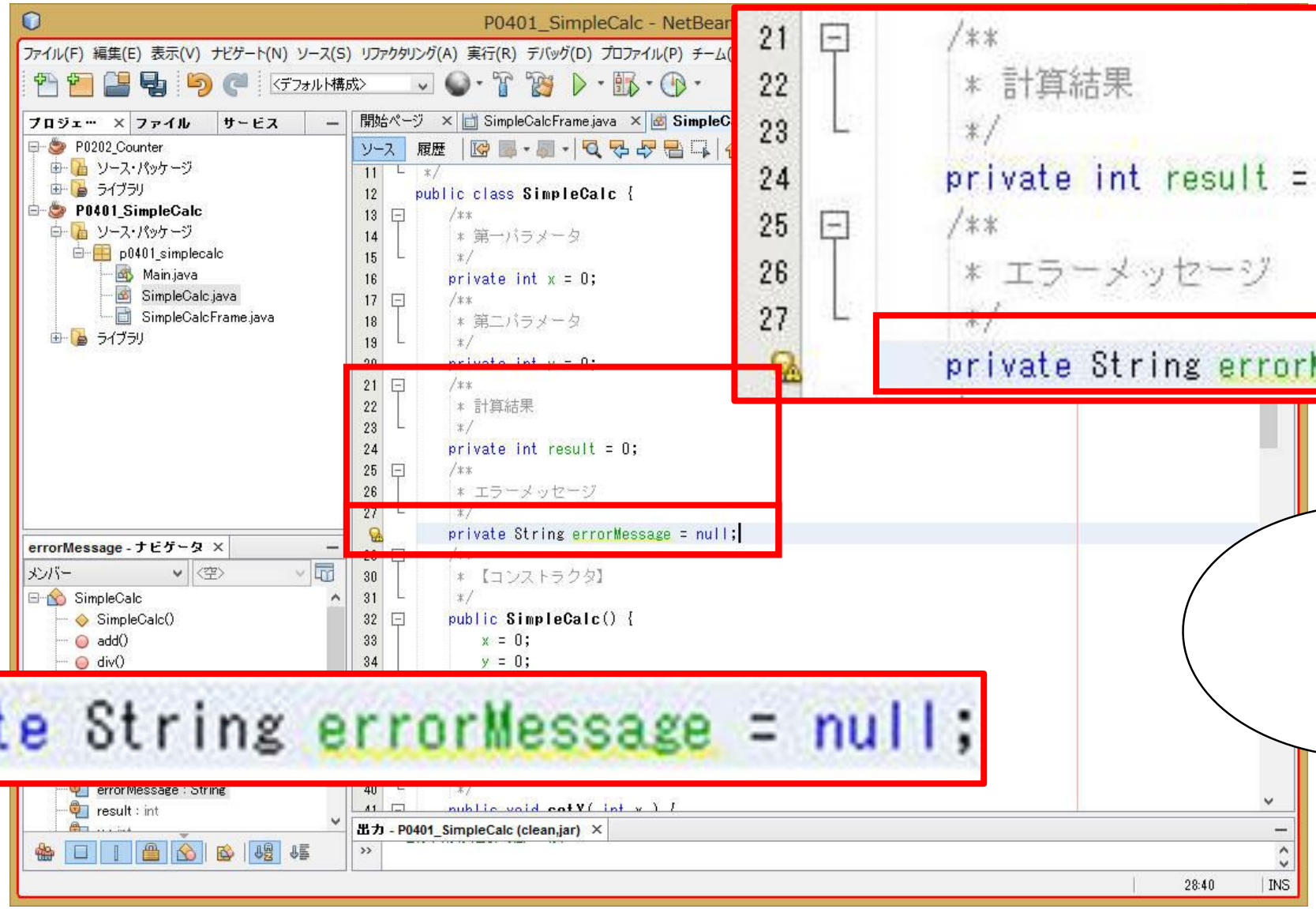
ソフトウェア工学実習
Software Engineering Practice
(第07回)

SEP07-007 【実習編】NetBeans-SimpleCalc

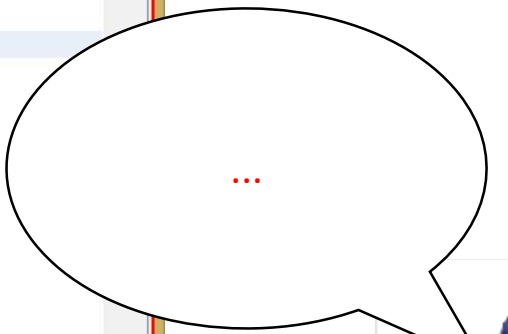
拡張(3):エラーメッセージを表示しよう

慶應義塾大学・理工学部・管理工学科
飯島 正
iijima@ae.keio.ac.jp

エラーメッセージ用の参照変数を追加

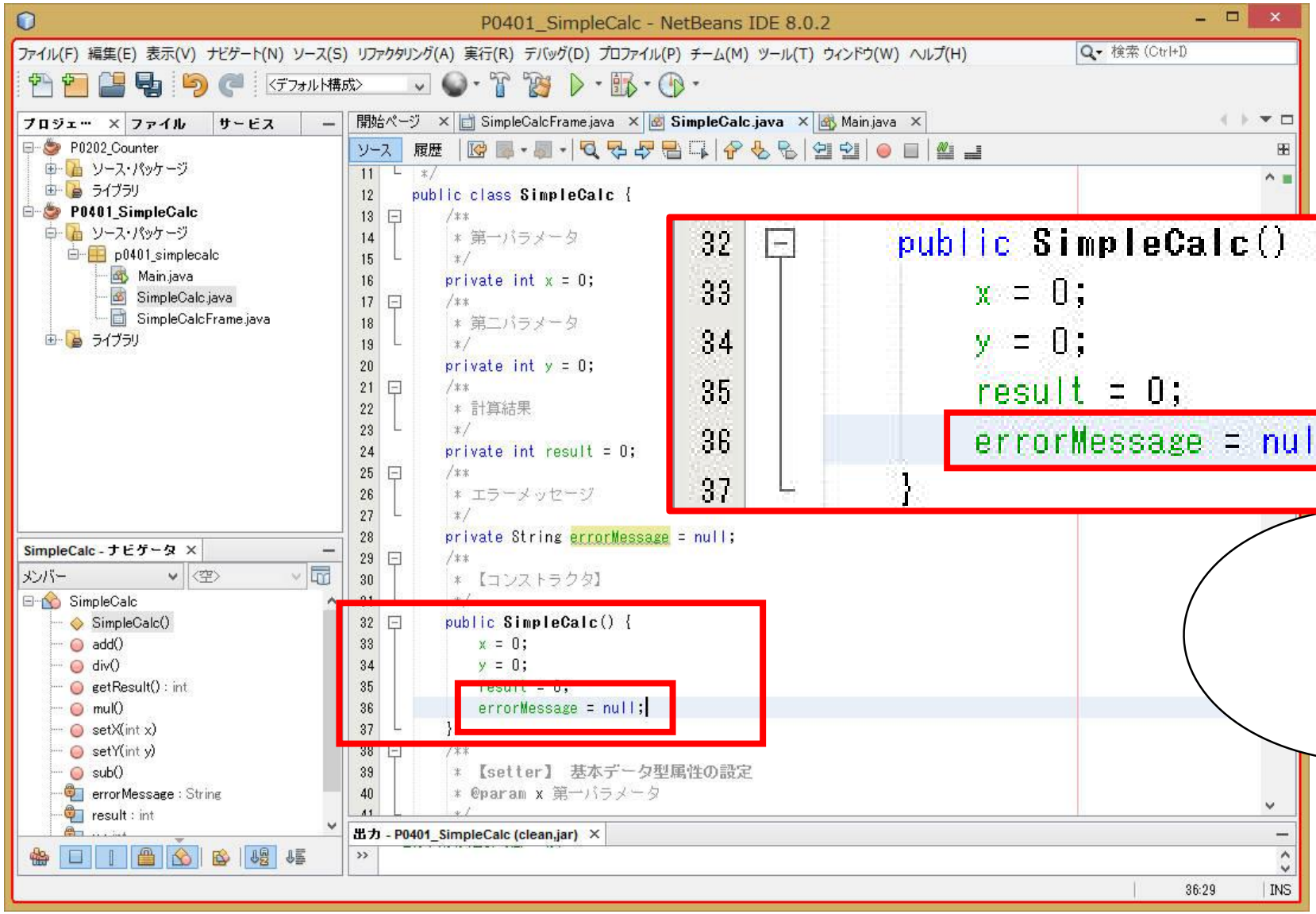



```
private String errorMessage = null;
```



iijima@ae.keio.ac.jp

コンストラクタで初期化（nullは正常終了も表す）

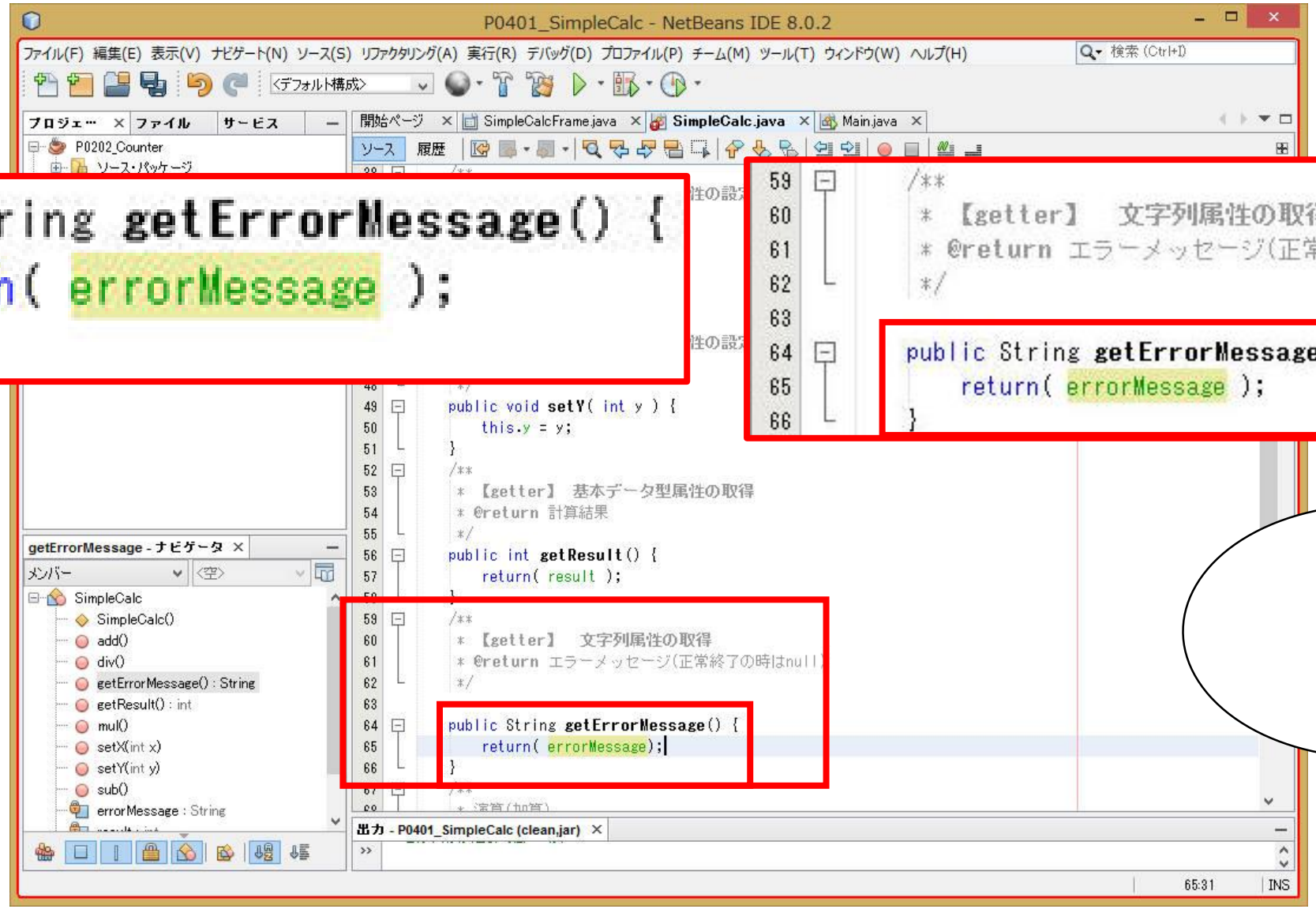


iijima@ae.keio.ac.jp

エラーメッセージのgetterを作成


```
public String getErrorMessage() {
    return( errorMessage );
}
```

```
59      /**
60       * 【getter】 文字列属性の取得
61       * @return エラーメッセージ(正常終了の時はnull)
62       */
63
64      public String getErrorMessage() {
65          return( errorMessage );
66      }
```



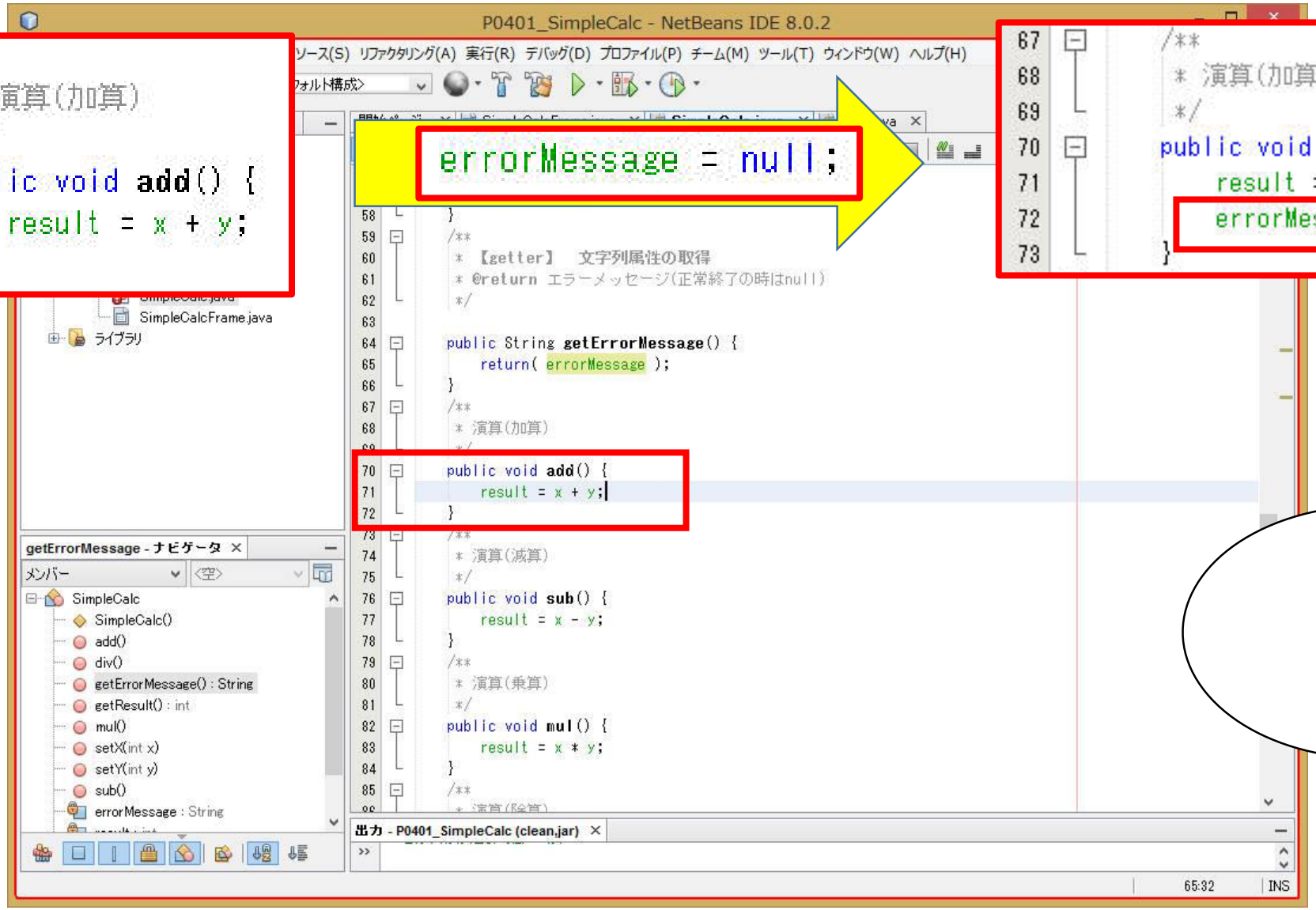
iijima@ae.keio.ac.jp

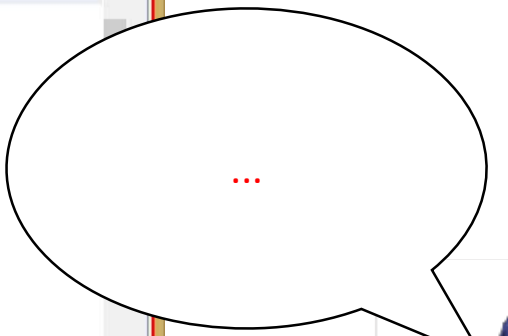
加算は正常終了するので，エラーメッセージにnull


```
67      /**
68       * 演算(加算)
69       */
70      public void add() {
71          result = x + y;
72      }
```

```
errorMessage = null;
```

```
67      /**
68       * 演算(加算)
69       */
70      public void add() {
71          result = x + y;
72          errorMessage = null;
73      }
```





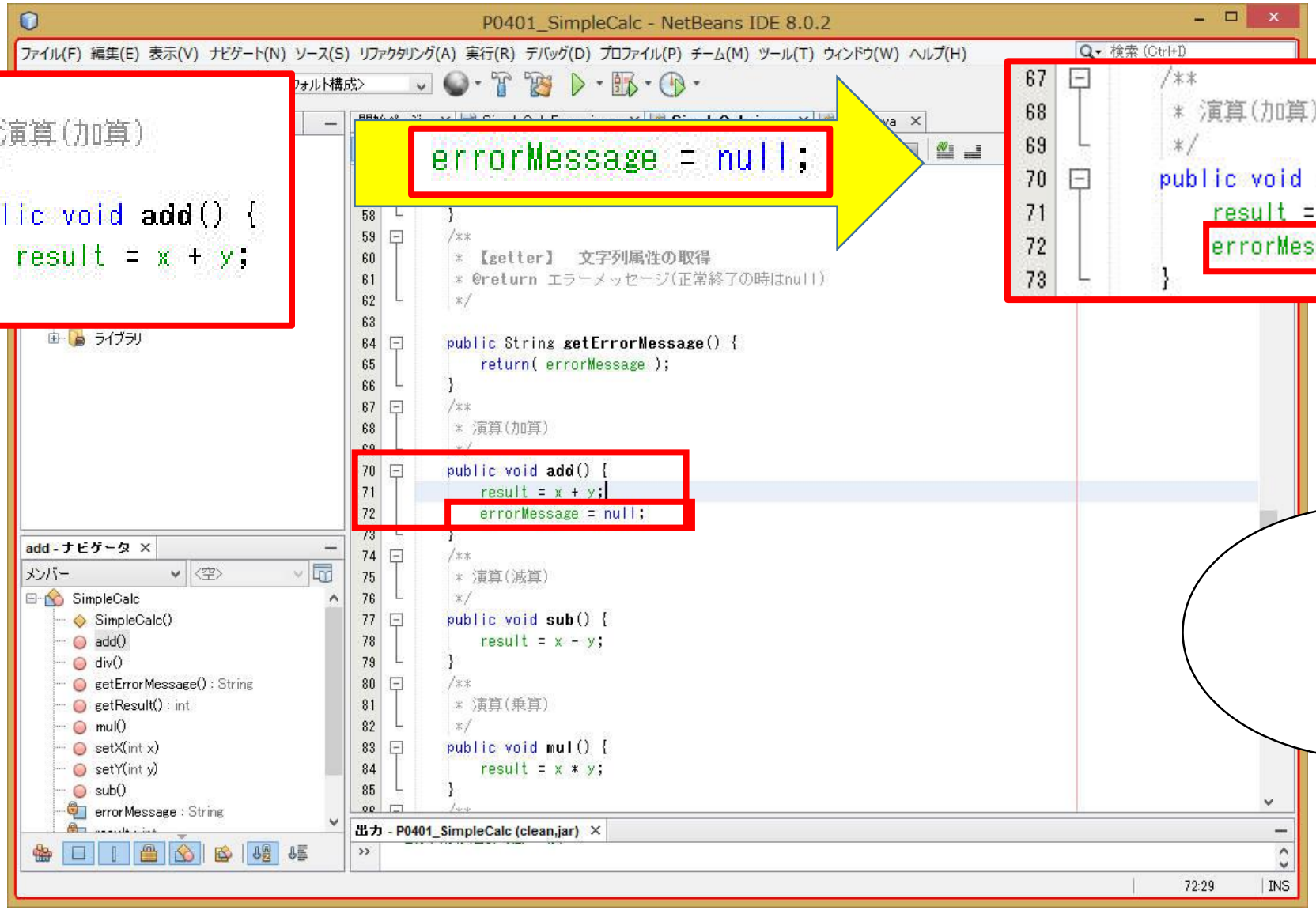
iijima@ae.keio.ac.jp

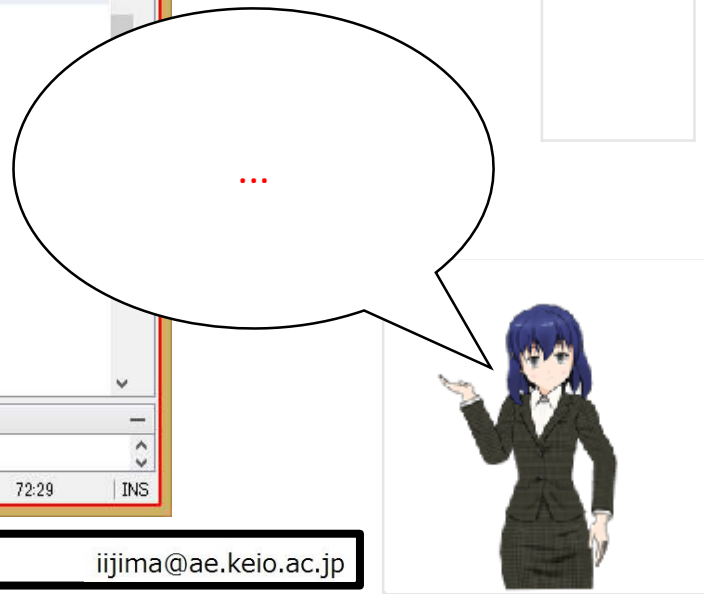
加算は正常終了するので，エラーメッセージにnull


```
67      /**
68       * 演算(加算)
69       */
70      public void add() {
71          result = x + y;
72      }
```

errorMessage = null;

```
67      /**
68       * 演算(加算)
69       */
70      public void add() {
71          result = x + y;
72          errorMessage = null;
73      }
```





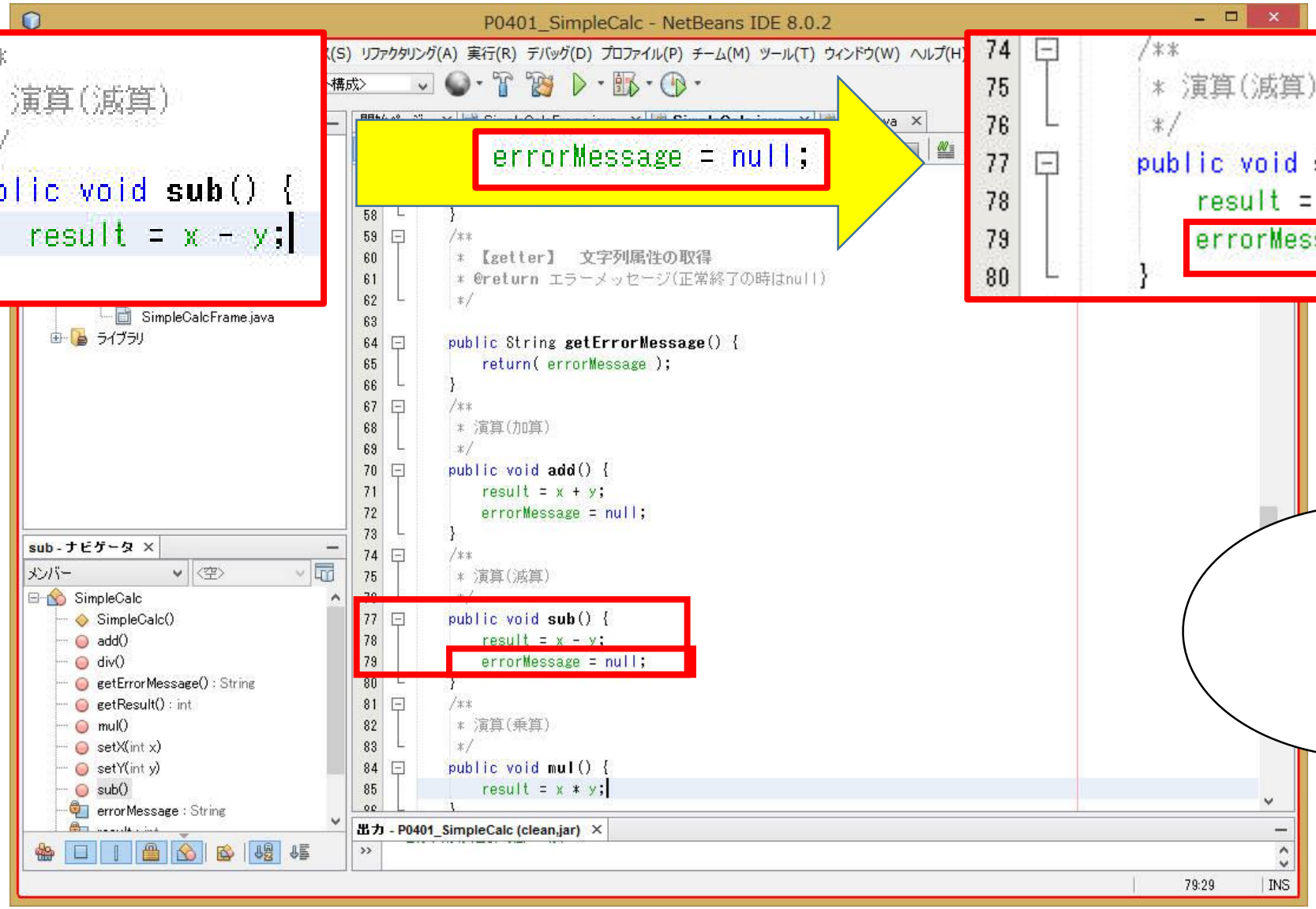
iijima@ae.keio.ac.jp

減算も同様


```
74      /**
75       * 演算(減算)
76       */
77      public void sub() {
78          result = x - y;
79      }
```

```
errorMessage = null;
```

```
74      /**
75       * 演算(減算)
76       */
77      public void sub() {
78          result = x - y;
79          errorMessage = null;
80      }
```



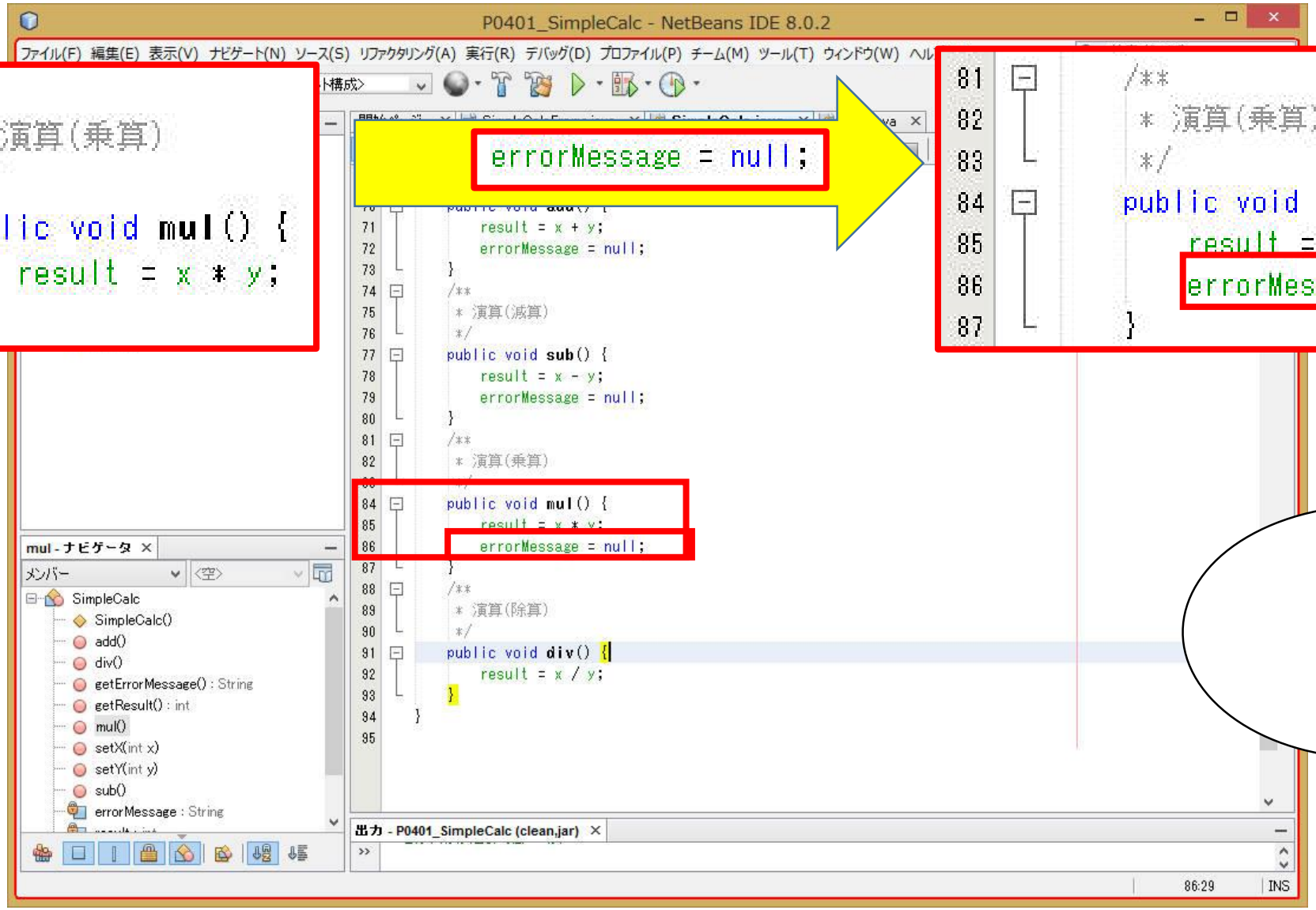
iijima@ae.keio.ac.jp

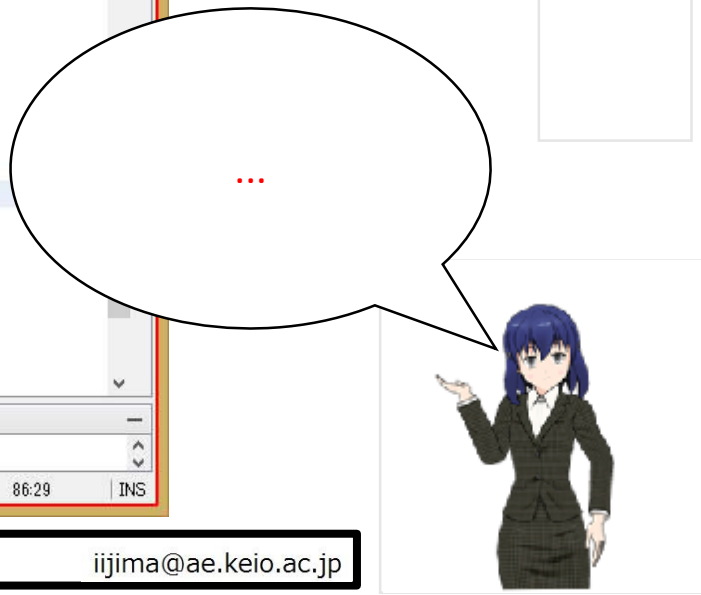
乗算も同様

```
80      /**
81       * 演算(乗算)
82       */
83      public void mul() {
84          result = x * y;
85      }
```

errorMessage = null;

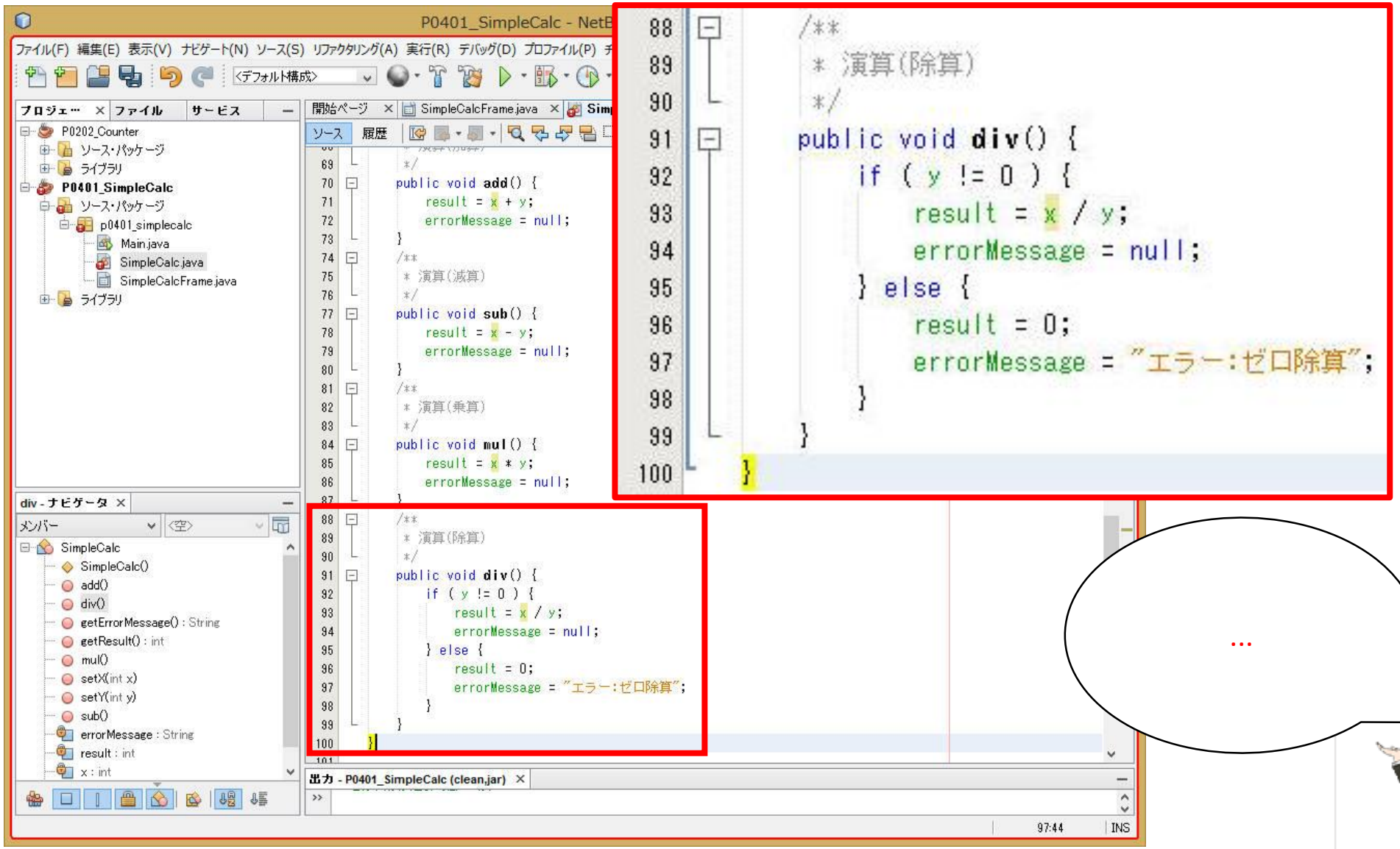
```
81      /**
82       * 演算(乗算)
83       */
84      public void mul() {
85          result = x * y;
86          errorMessage = null
87      }
```





iijima@ae.keio.ac.jp

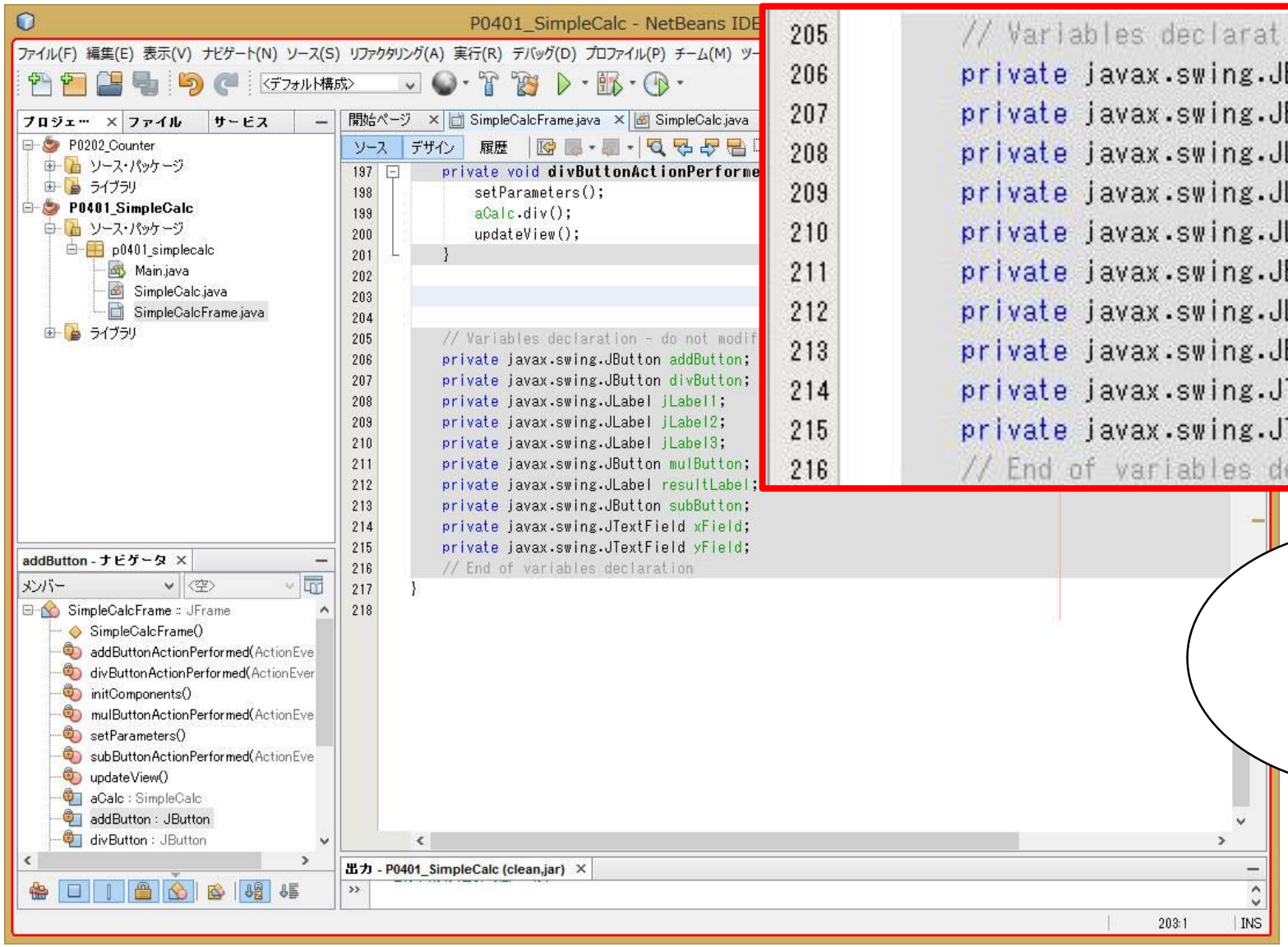
除算の場合は，「ゼロ除算エラー」があり得ます




```
88      /**
89       * 演算(除算)
90       */
91      public void div() {
92          if ( y != 0 ) {
93              result = x / y;
94              errorMessage = null;
95          } else {
96              result = 0;
97              errorMessage = "エラー:ゼロ除算";
98          }
99      }
100 }
```


iijima@ae.keio.ac.jp

GUI部品の参照変数を確認しておきます．



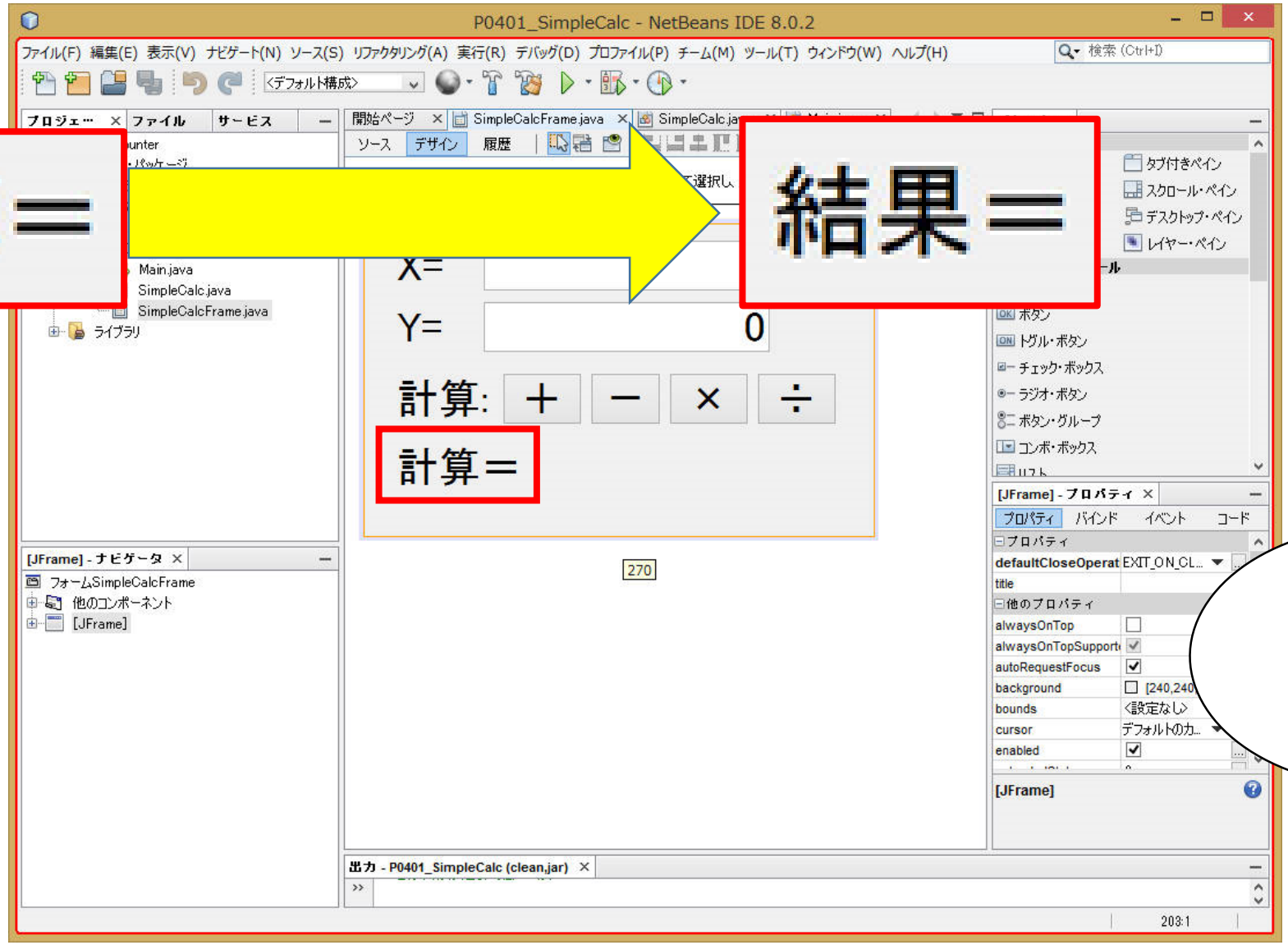

```
197     private void divButtonActionPerforme
198         setParameters();
199         aCalc.div();
200         updateView();
201     }
202
203
204
205     // Variables declaration - do not modif
206     private javax.swing.JButton addButton;
207     private javax.swing.JButton divButton;
208     private javax.swing.JLabel jLabel1;
209     private javax.swing.JLabel jLabel2;
210     private javax.swing.JLabel jLabel3;
211     private javax.swing.JButton mulButton;
212     private javax.swing.JLabel resultLabel;
213     private javax.swing.JButton subButton;
214     private javax.swing.JTextField xField;
215     private javax.swing.JTextField yField;
216     // End of variables declaration
217 }
218
```

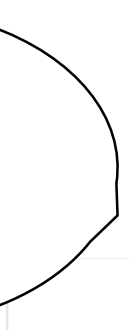
```
205     // Variables declaration - do not modify
206     private javax.swing.JButton addButton;
207     private javax.swing.JButton divButton;
208     private javax.swing.JLabel jLabel1;
209     private javax.swing.JLabel jLabel2;
210     private javax.swing.JLabel jLabel3;
211     private javax.swing.JButton mulButton;
212     private javax.swing.JLabel resultLabel;
213     private javax.swing.JButton subButton;
214     private javax.swing.JTextField xField;
215     private javax.swing.JTextField yField;
216     // End of variables declaration
```

…

iijima@ae.keio.ac.jp

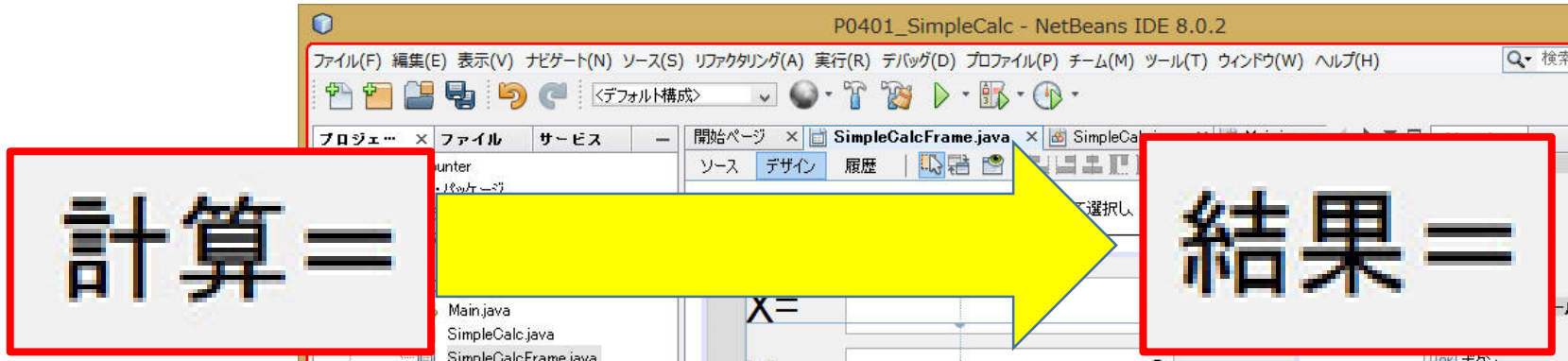
必要に応じてエラーメッセージを表示させるようにします

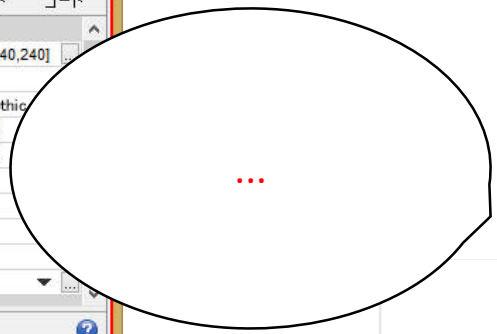




iijima@ae.keio.ac.jp

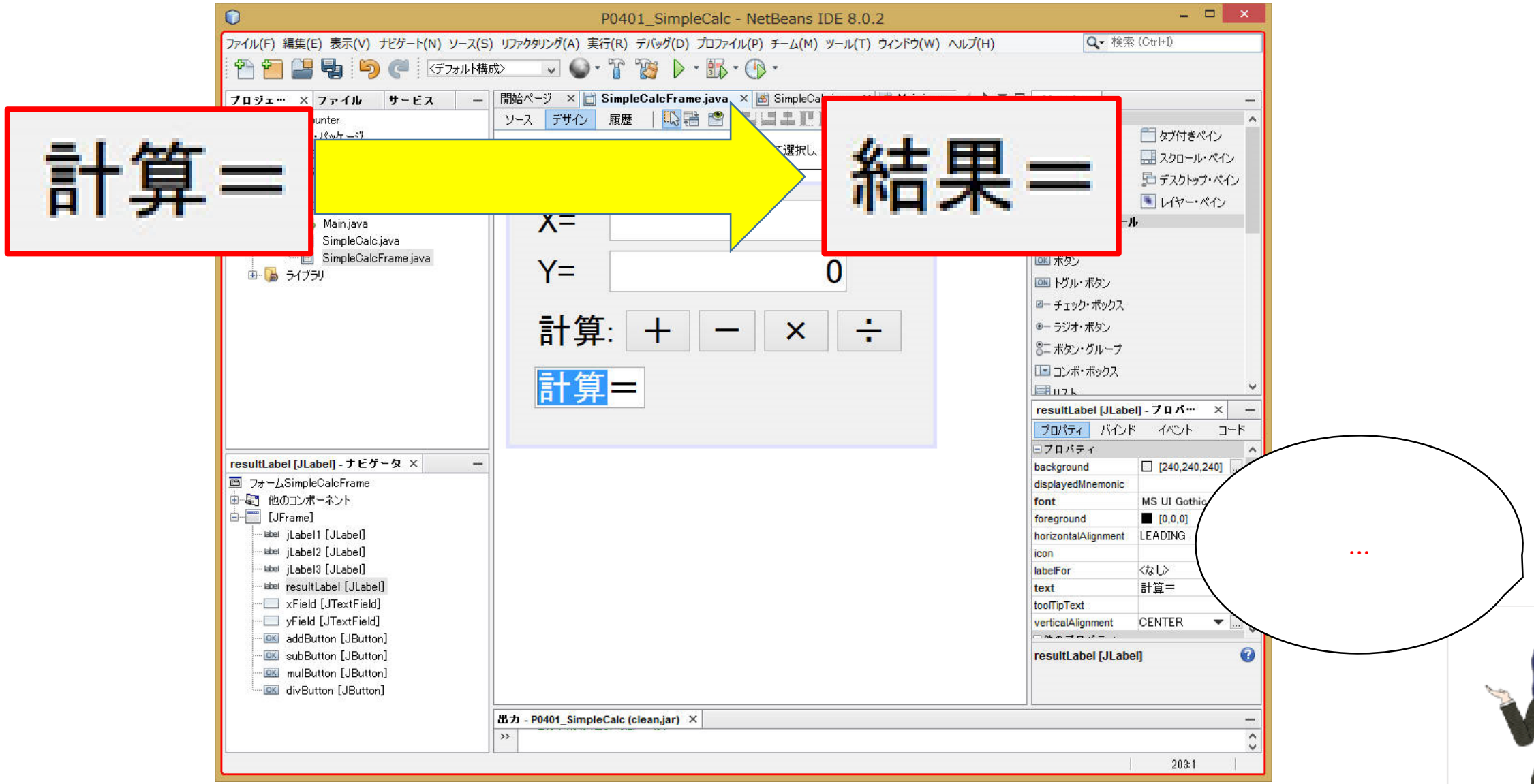
その前に一つ，ラベルを修正しましょう





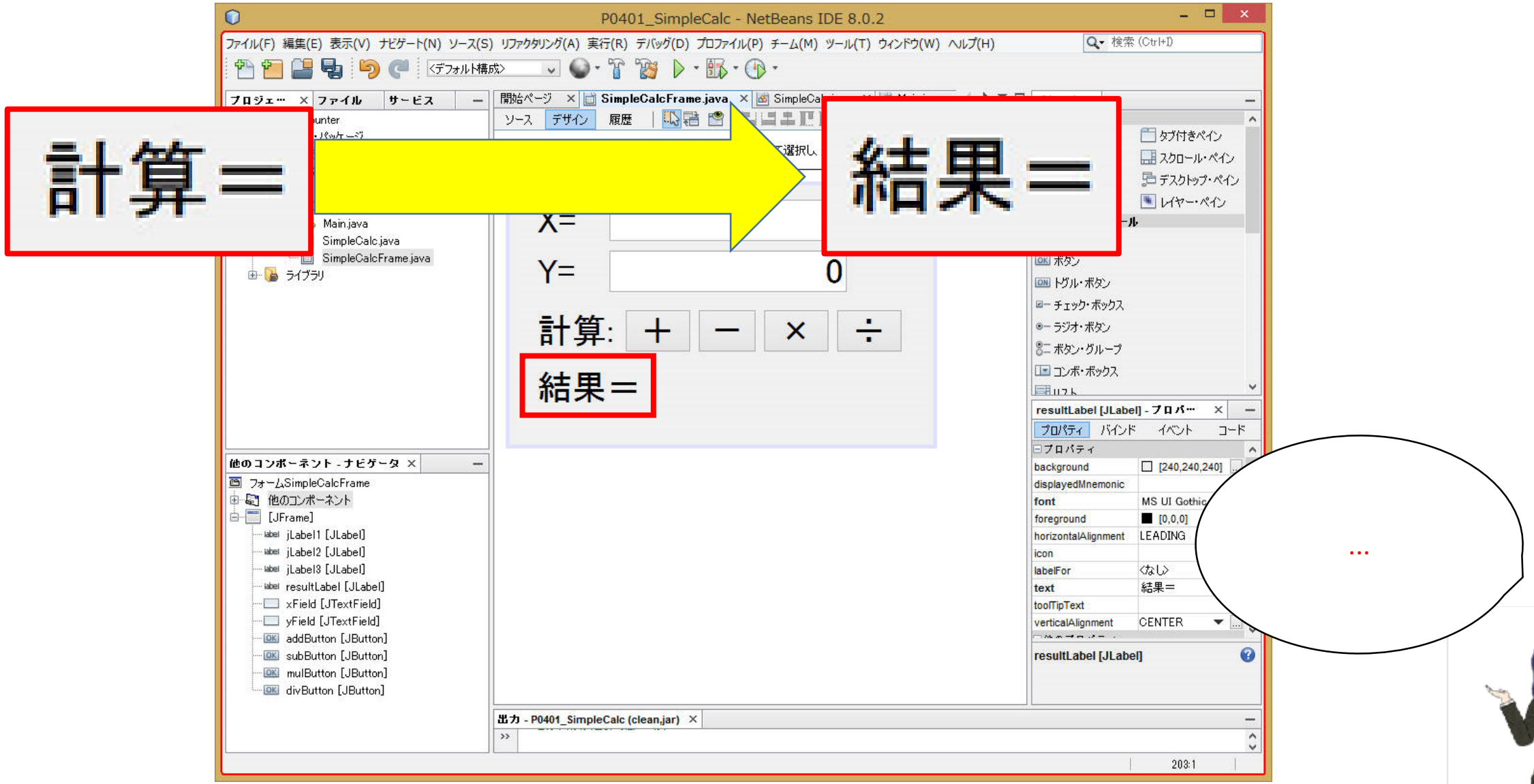
iijima@ae.keio.ac.jp

その前に一つ，ラベルを修正しましょう



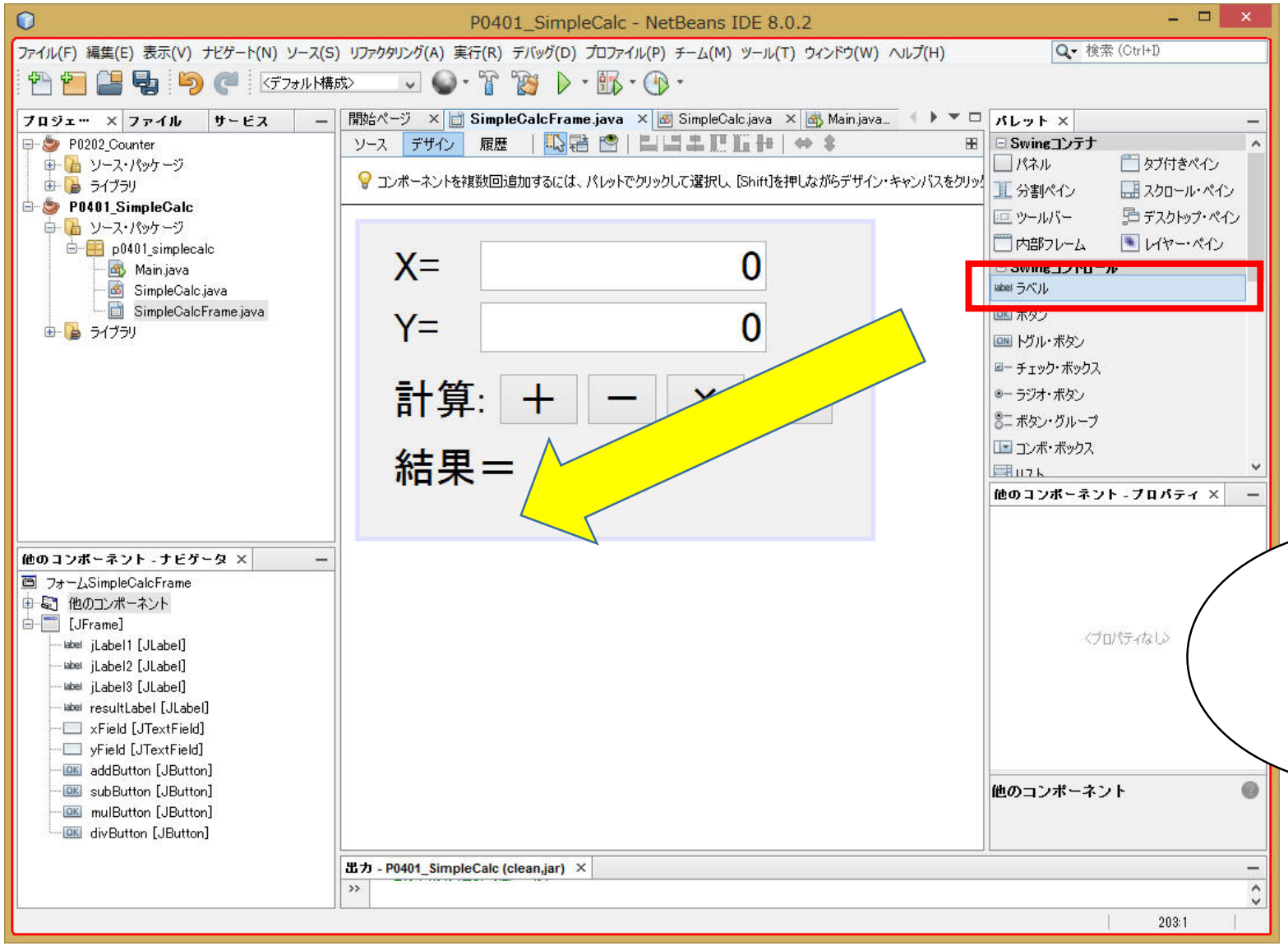
iijima@ae.keio.ac.jp

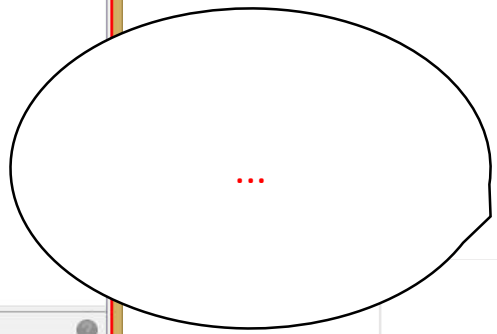
その前に一つ，ラベルを修正しましょう



iijima@ae.keio.ac.jp

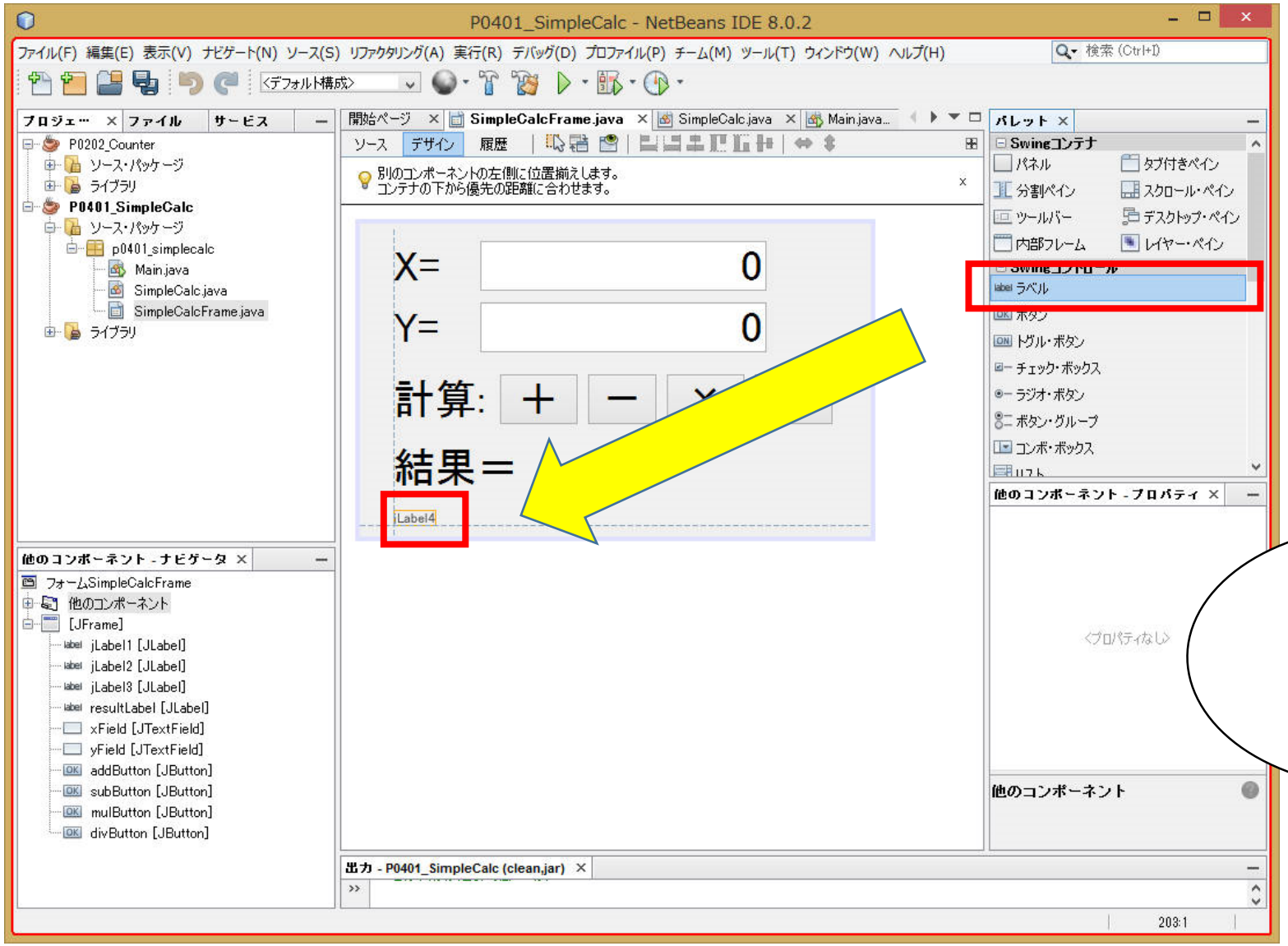
Swingコントロールパレットから ラベルをドラッグ&ドロップ

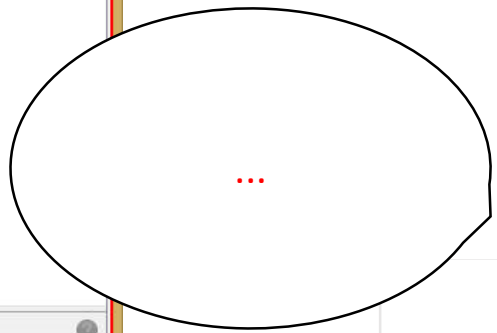




iijima@ae.keio.ac.jp

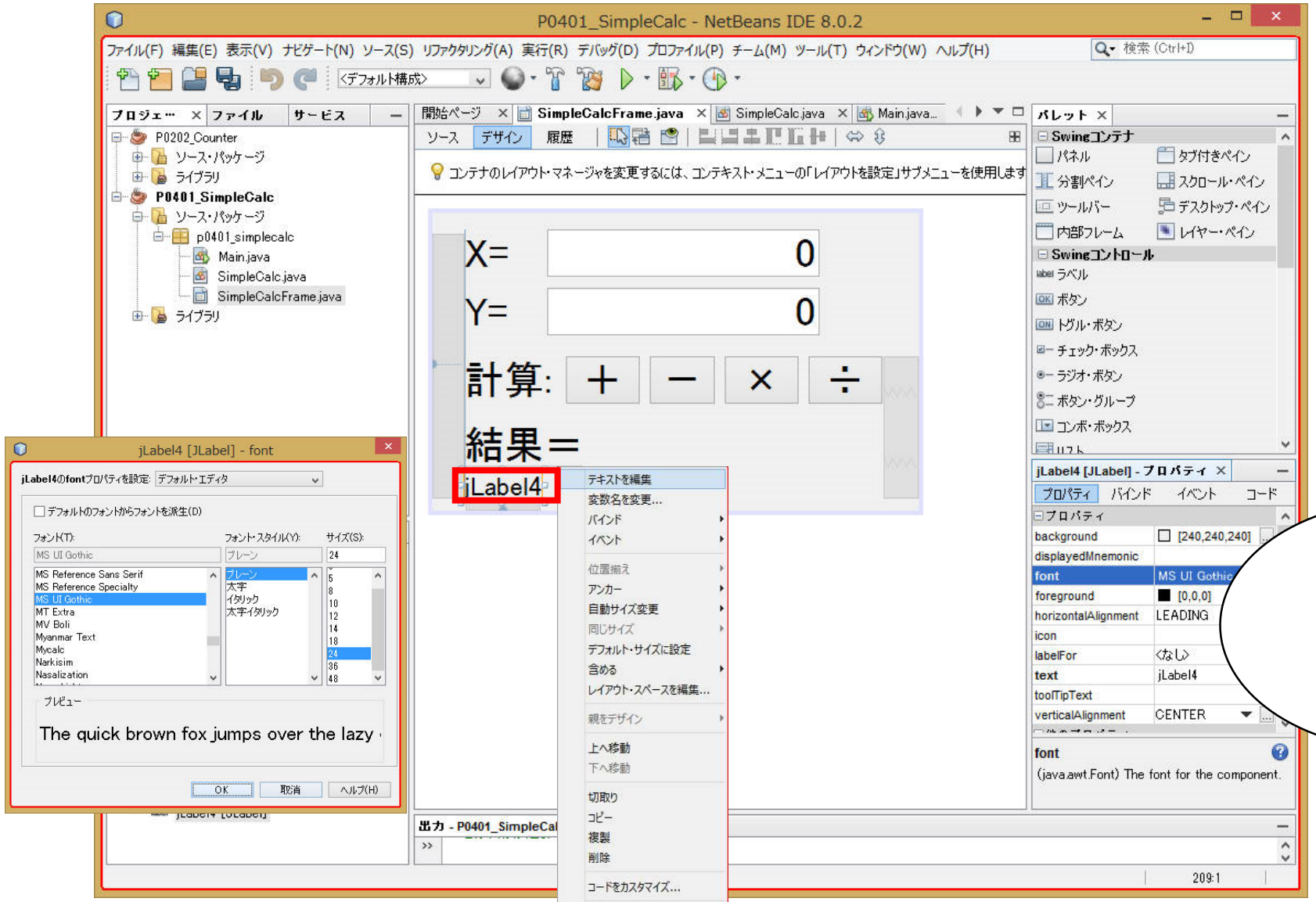
Swingコントロールパレットから ラベルをドラッグ&ドロップ





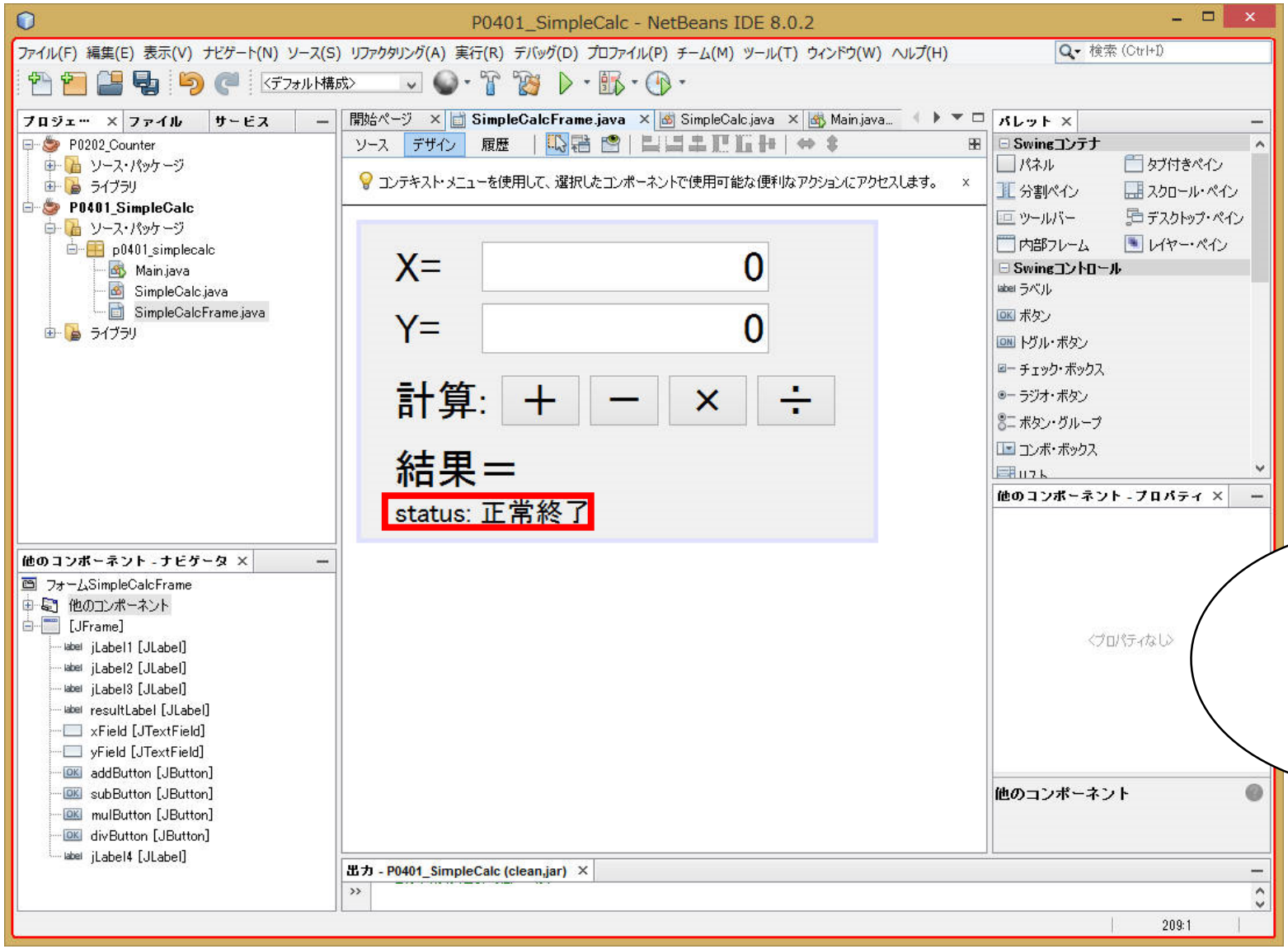
iijima@ae.keio.ac.jp

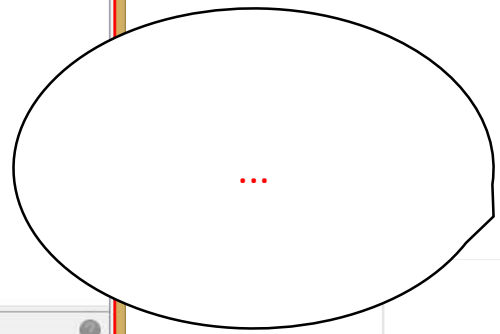
文字サイズを大きくして，文字列の変更



iijima@ae.keio.ac.jp

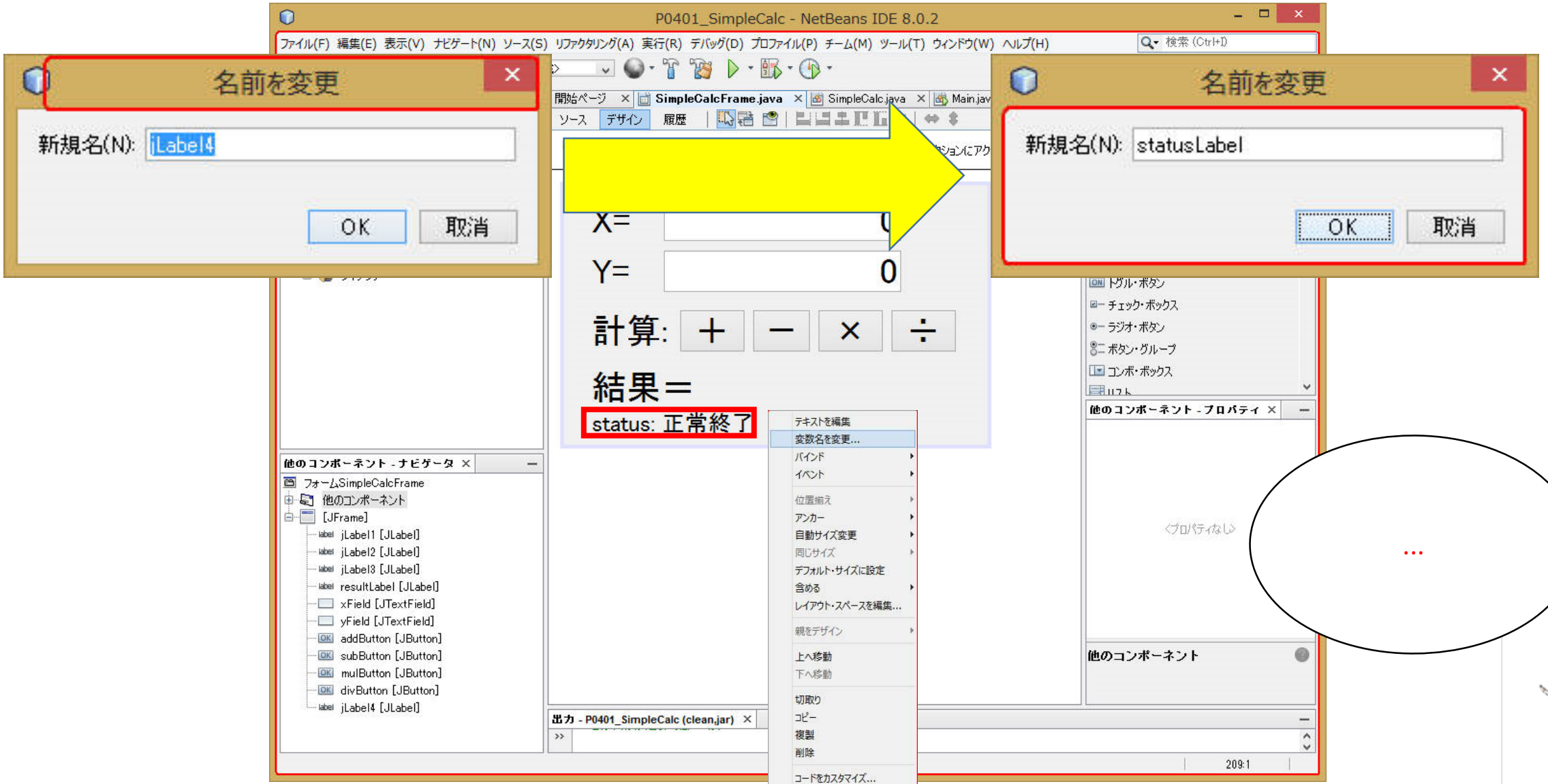
「Status:正常終了」としましょう





iijima@ae.keio.ac.jp

変数名を，statusLabelに変更します

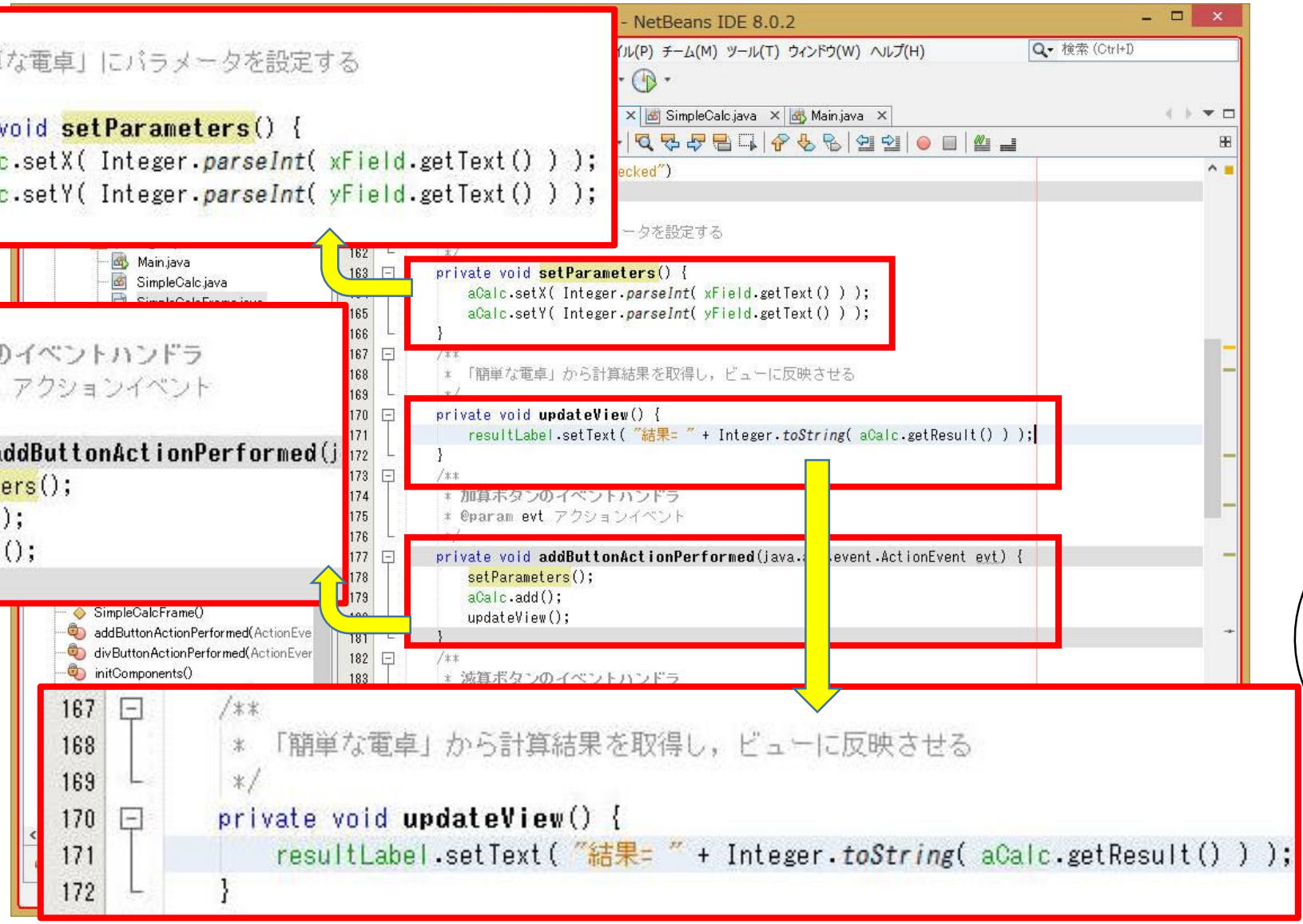


iijima@ae.keio.ac.jp

演算時のイベントハンドラのメソッド


```java
/**
 * 「簡単な電卓」にパラメータを設定する
 */
private void setParameters() {
    aCalc.setX( Integer.parseInt( xField.getText() ) );
    aCalc.setY( Integer.parseInt( yField.getText() ) );
}
```

```java
/**
 * 加算ボタンのイベントハンドラ
 * @param evt アクションイベント
 */
private void addButtonActionPerformed(j
    setParameters();
    aCalc.add();
    updateView();
}
```

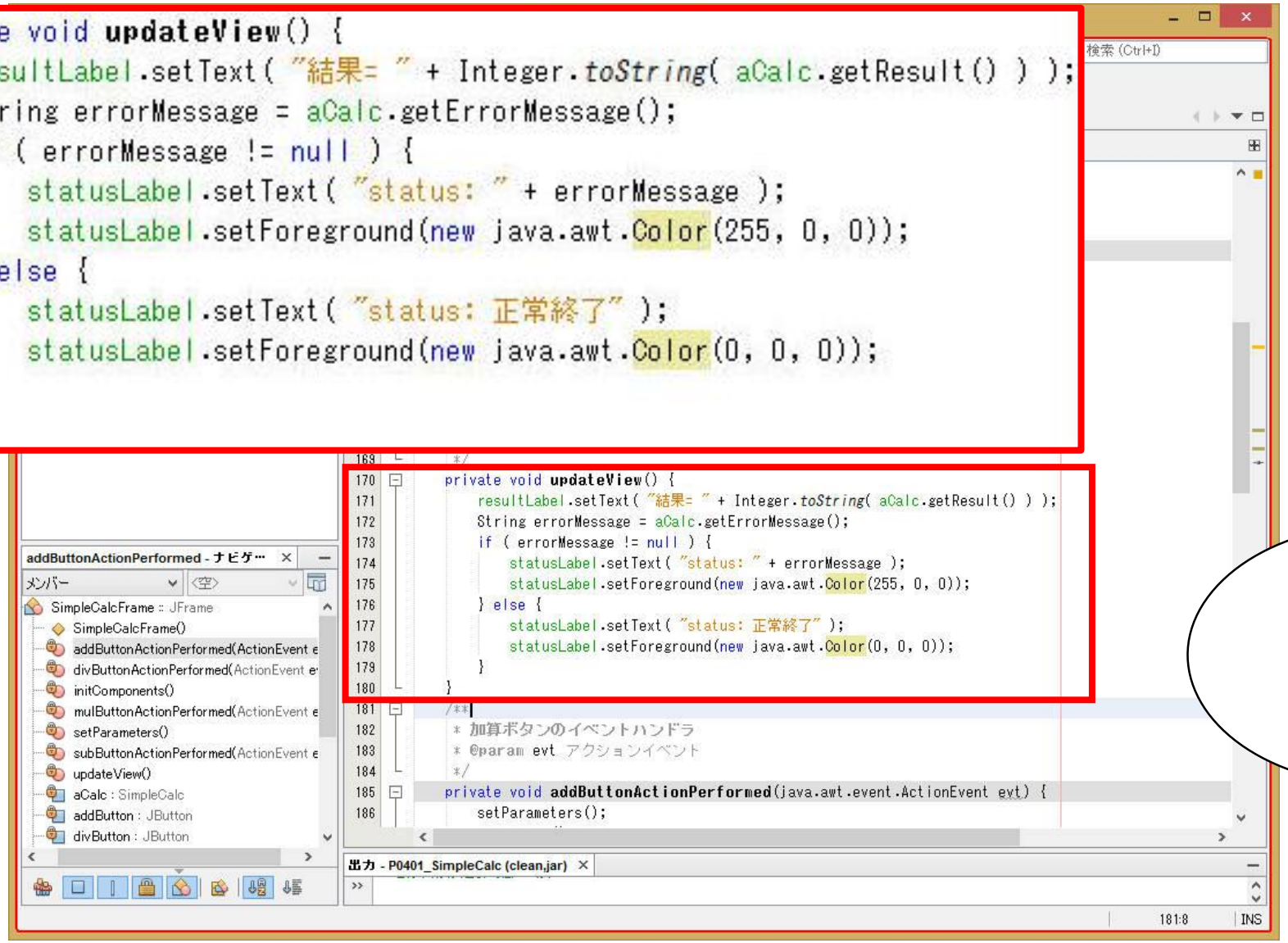



```java
/**
 * 「簡単な電卓」から計算結果を取得し，ビューに反映させる
 */
private void updateView() {
    resultLabel.setText( "結果= " + Integer.toString( aCalc.getResult() ) );
}
```

iijima@ae.keio.ac.jp

updateView() メソッドに エラーメッセージ表示機能を追加

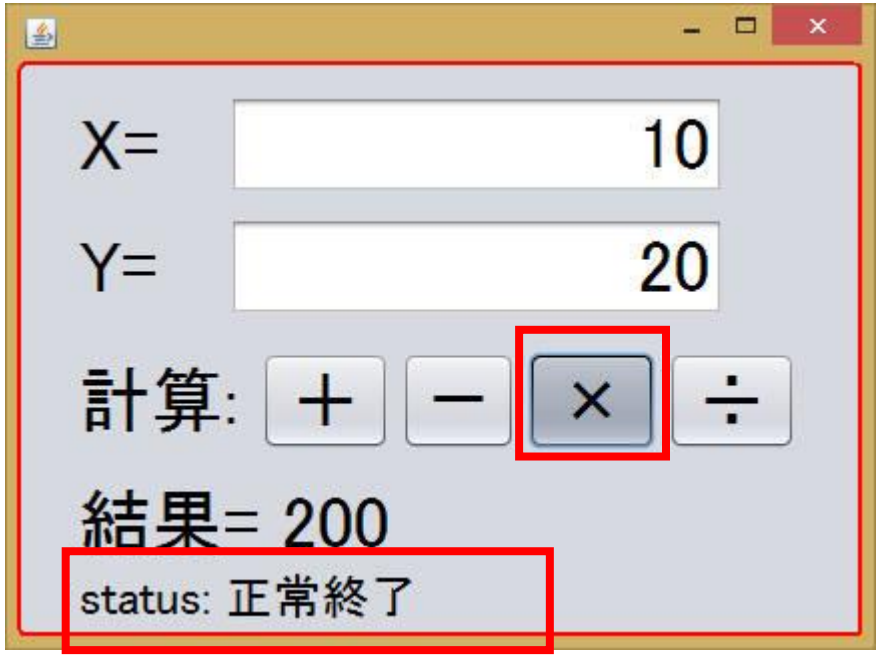

```
170    private void updateView() {
171        resultLabel.setText( "結果= " + Integer.toString( aCalc.getResult() ) );
172        String errorMessage = aCalc.getErrorMessage();
173        if ( errorMessage != null ) {
174            statusLabel.setText( "status: " + errorMessage );
175            statusLabel.setForeground(new java.awt.Color(255, 0, 0));
176        } else {
177            statusLabel.setText( "status: 正常終了" );
178            statusLabel.setForeground(new java.awt.Color(0, 0, 0));
179        }
180    }
```



…

iijima@ae.keio.ac.jp

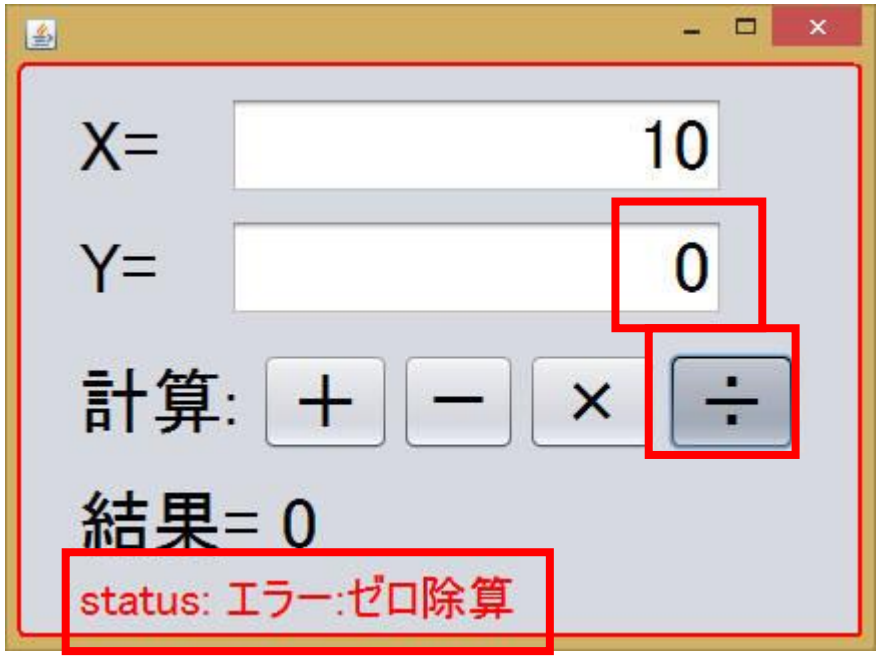
乗算は正常終了



…

iijima@ae.keio.ac.jp

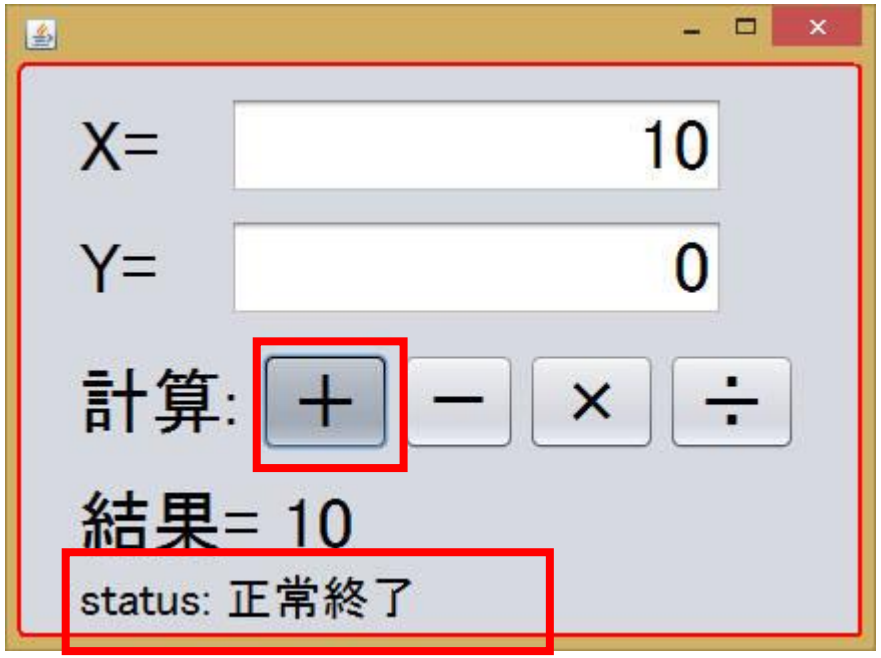
ゼロ除算は，エラー表示

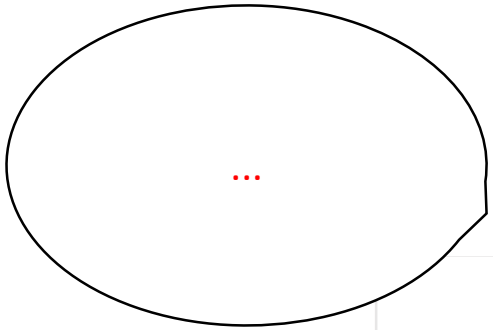


…

iijima@ae.keio.ac.jp

続けて，正常な計算をしますと…





iijima@ae.keio.ac.jp