Java入門講座①

はじめてのソフトウェア開発編

Java入門講座のねらい

- ~~Javaの文法をしっかり学ぶ~~
- ~~アルゴリズムの知識を身に付ける~~
- ~~オブジェクト指向について学ぶ~~

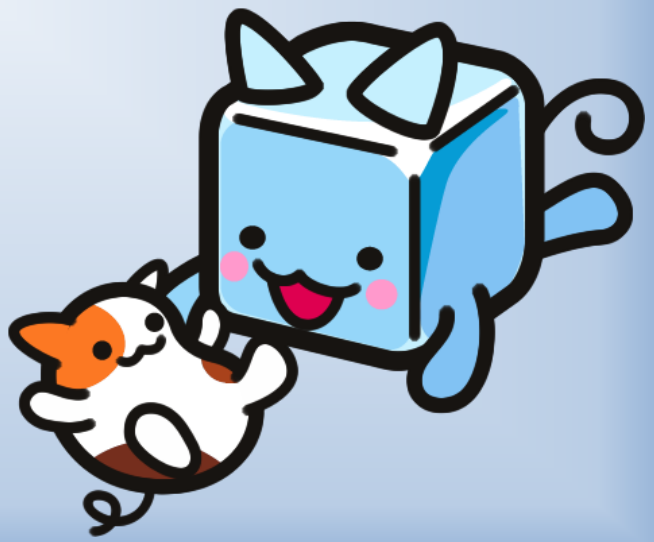
- プログラムが動くのを楽しんでもらう
- ソフトウェア開発を身近に感じる

初めに楽しさを知れば、学ぶのが楽しくなる！

Java実習の流れ

1. 開発環境のインストール
2. 基本的なプログラミング
3. ソフトウェアの作り方
4. Swingによるお絵かき
5. Threadによるアニメーション

1.　開発環境のインストール



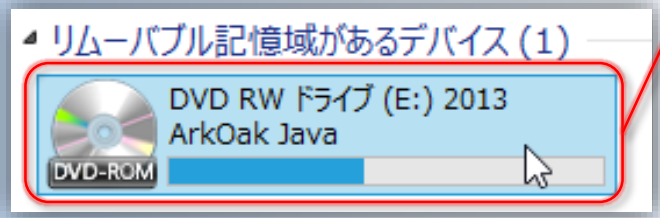
Javaの開発環境

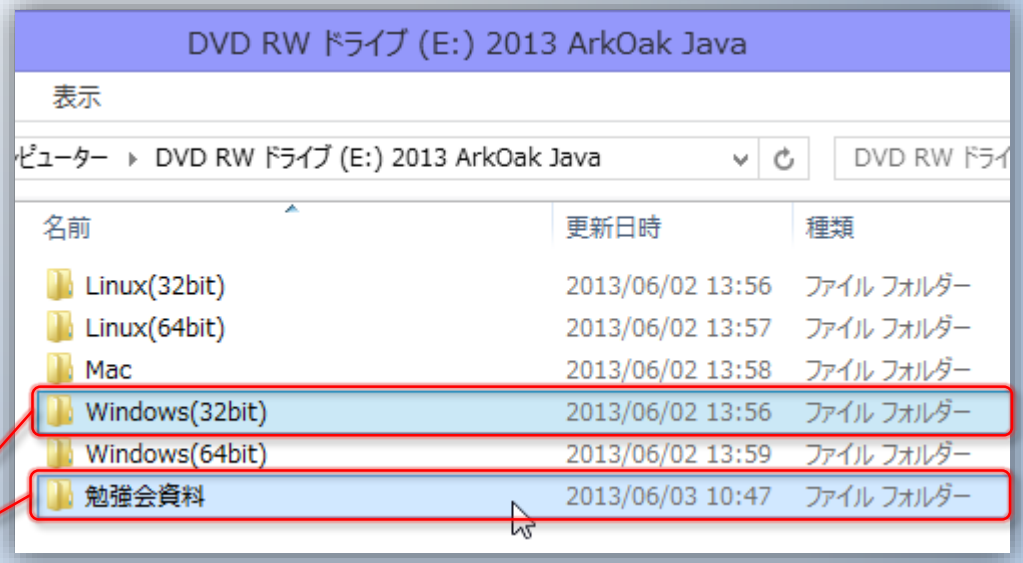
- JDK(Java SE Development Kit)
  - Javaでプログラミングする際に必要な開発キット
- NetBeans
  - 公式のJavaの統合開発環境(IDE)
  - 優れたGUIエディタを搭載
- Javaドキュメント
  - Javaの便利な取扱説明書

NetBeansによる開発は年々上昇中!

DVDからデータの取出し

DVDドライブが無い人は
USBフラッシュメモリもしくはネットから





PCの環境に合ったフォルダと
「勉強会資料」フォルダを選び、コピーする

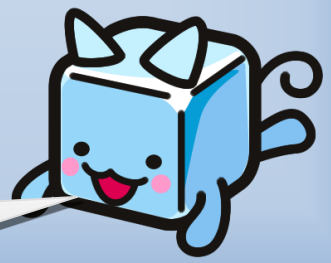
フォルダの中に2つのファイルが入ってるか確認しよう。
講習会参加者じゃない人は、公式サイトからダウンロード。

インストール手順

① 『jdk-7u21-nb-7_3-windows-xxx.exe』を実行して
JDKとNetBeansをインストール。

② 『jdk-6-doc-ja-ea.zip』を“C:¥Program Files¥Java”
等のフォルダに置いて、NetBeansにJavaDocを組み込む。

公式サイトから、最新版がダウンロードできる

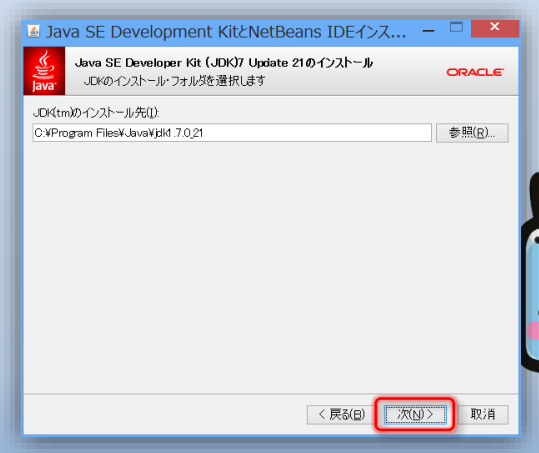


JDKとNetBeansのインストール①

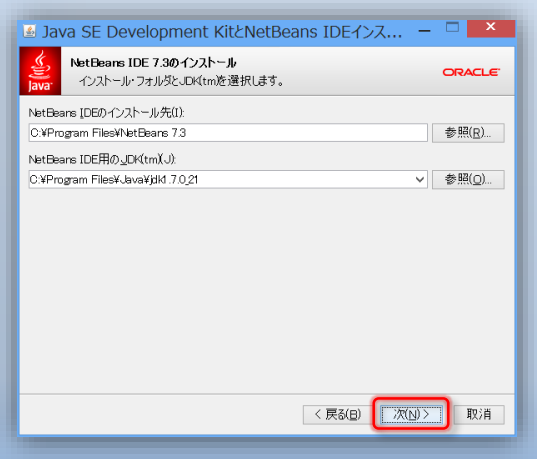
1. 〔次へ〕を押す

2. Junitをインストールにチェックを入れて、「次へ」を押す

3. 〔次へ〕を押す

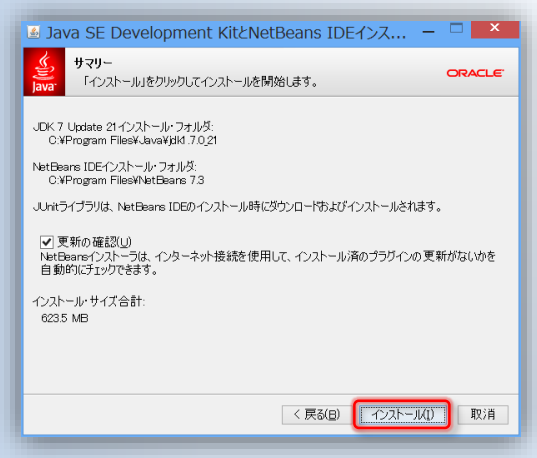


4. 〔次へ〕を押す

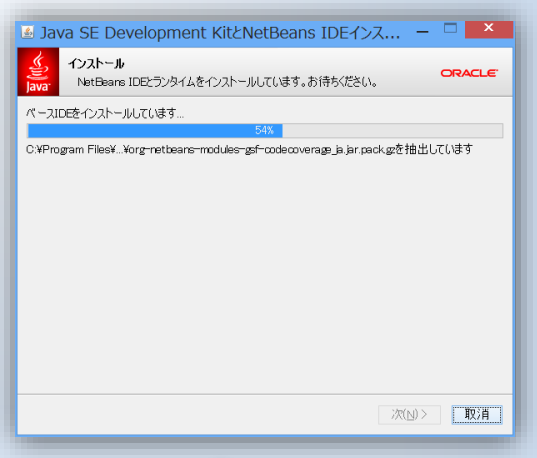


JDKとNetBeansのインストール②

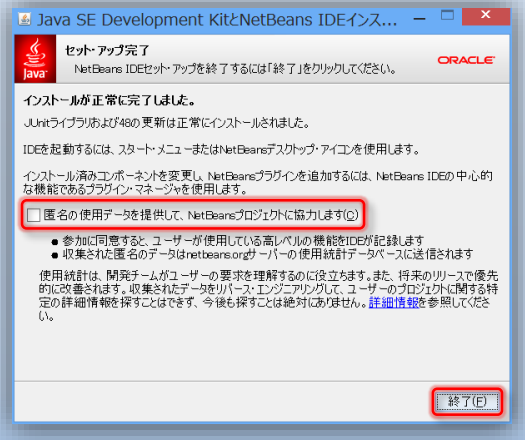
5. 〔インストール〕を押す

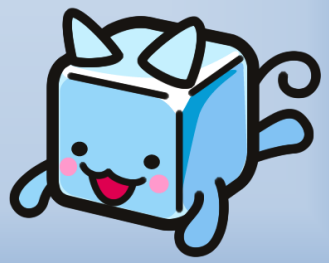


6. しばらく待つ



7. チェックを外して〔完了〕を押す

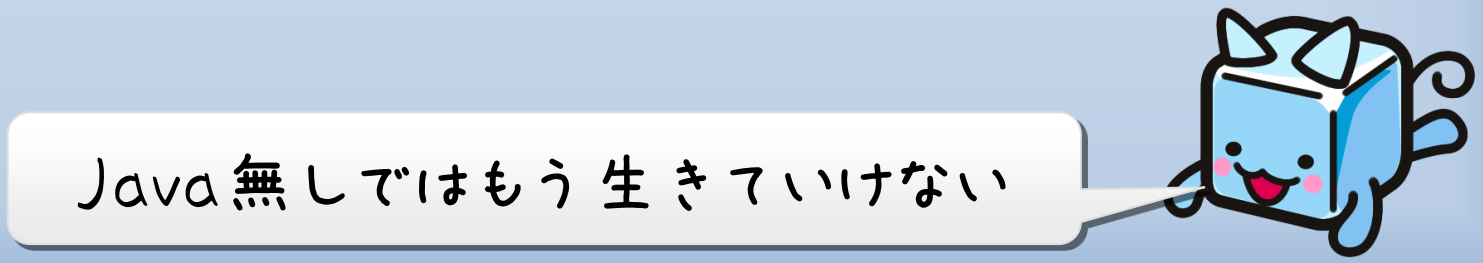




Javaとは

- プログラミング言語
- C言語から派生した言語(なので文法が似てる)
- ソフトウェア開発がしやすい

- マルチプラットフォーム(どんなPC上でも動く)
- ライブラリがものすごく豊富
- だいたい何でも作れる



C言語から引き継いだ文法

変数の型

- int型
- float型
- double型
- char型

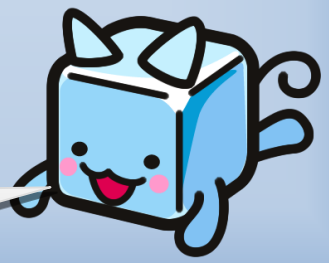
制御文

- if文
- switch文
- for文
- while文

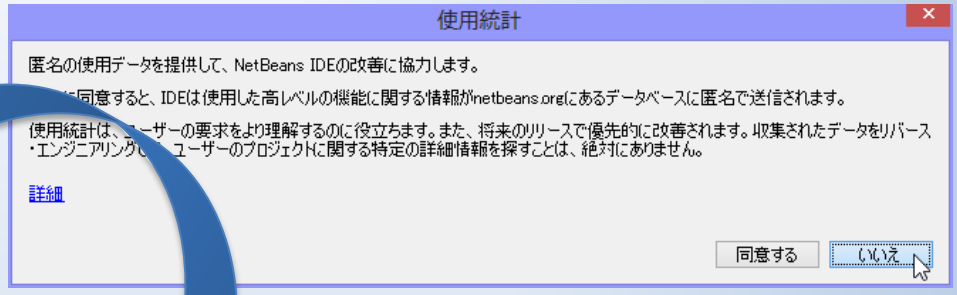
演算子

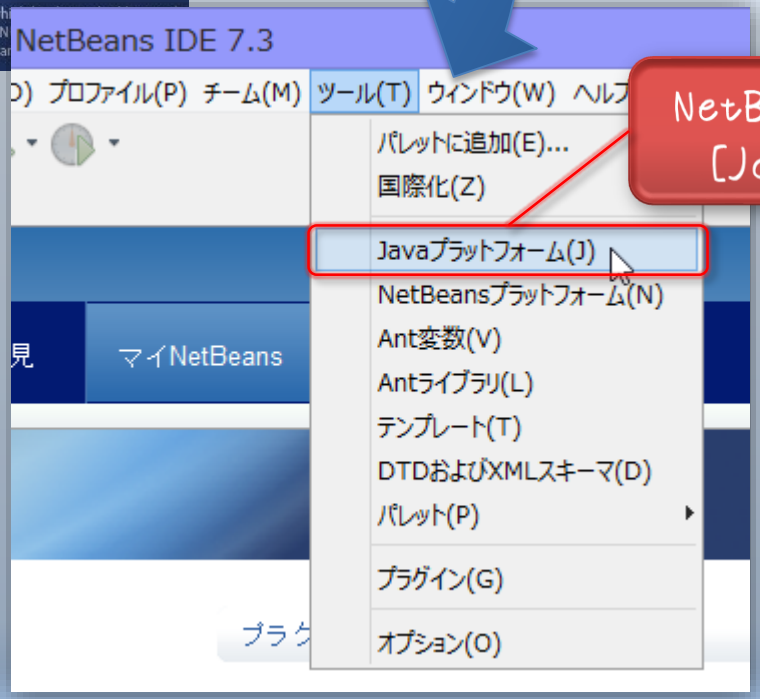
- 算術演算子
- 比較演算子
- 論理演算子

結構似た感じのソースコードになるよ

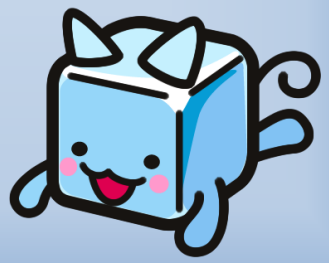


Javaドキュメントの組み込み





NetBeansを起動し、[ツール]から[Javaプラットフォーム]を選択



Javaドキュメントの組み込み

Javaプラットフォーム・マネージャ

「Javadoc」タブを使用して、JDKのAPIドキュメントをIDEに登録します。
「プラットフォームを追加」をクリックして、他のJavaプラットフォーム・バージョンを登録します。

プラットフォーム(P):
- Java SE
  - JDK 1.7(デフォルト)

プラットフォーム名: JDK 1.7(デフォ

プラットフォーム・フォルダ: C:¥Program Files¥Java¥jdk1.7.0_21

クラス | ソース | Javadoc

プラットフォームJavadoc:
http://download.oracle.com/javase/7/docs/api/

ZIP/フォルダの追加...
URLの追加...
除去
上へ移動
下へ移動

プラットフォームの追加... | 除去

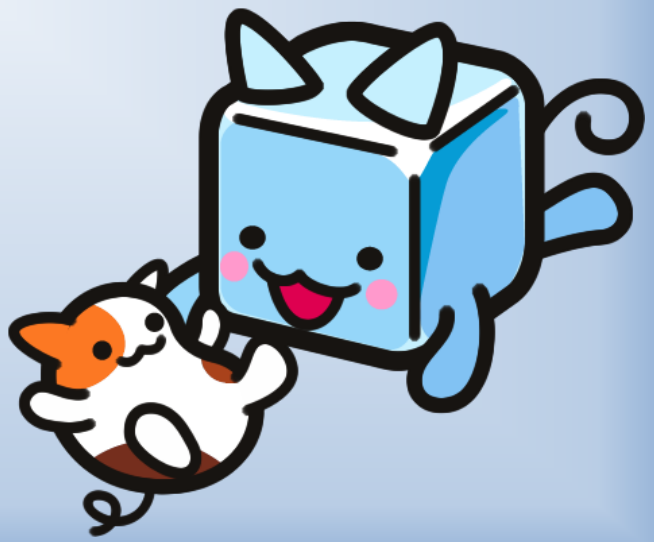
閉じる | ヘルプ(H)

選択して削除する

[jdk-6-doc-ja-ea.zip] を追加する

英語が得意な人はこの操作は必要なし

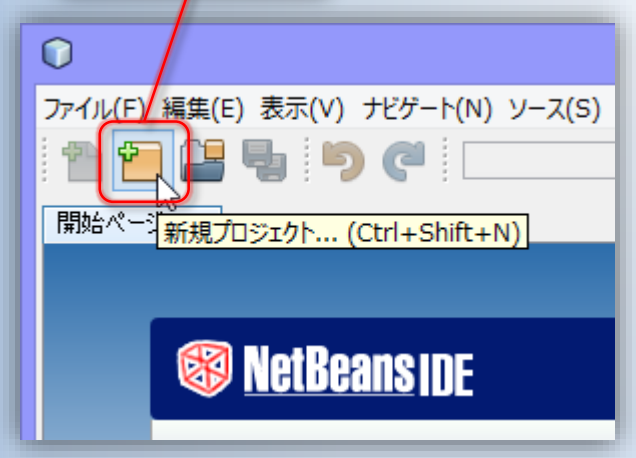
2.　基本的なプログラミング



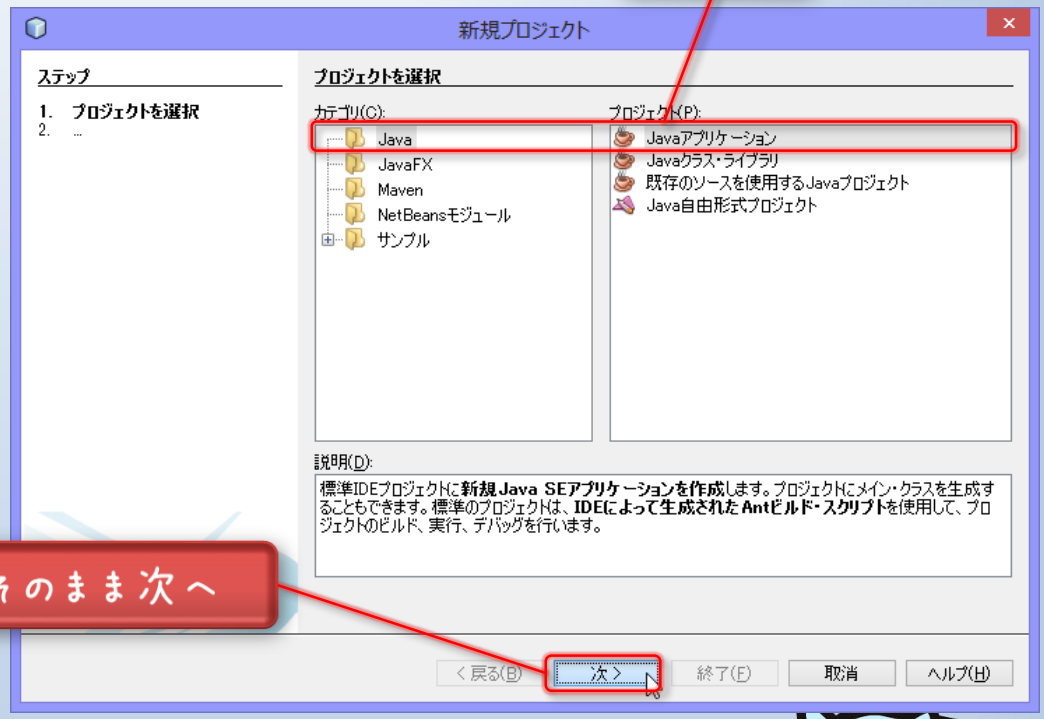
新規プロジェクト
「Lecture1」を作ろう

プロジェクトの作成

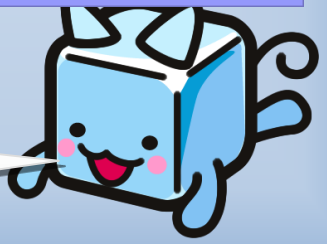
クリック



確認



そのまま次へ



〔ファイル〕→〔新規プロジェクト〕でもOK。

プロジェクトの作成

プロジェクトの名前をつける
(Lecture1)

新規 Javaアプリケーション

ステップ
1. プロジェクトを選択
2. 名前と場所

名前と場所

プロジェクト名(N): Lecture1

プロジェクトの場所(L): D:¥NetBeansProjects　参照(O)...

プロジェクト・フォルダ(D): D:¥NetBeansProjects¥Lecture1

☐ ライブラリの格納用に専用フォルダを使用(U)

ライブラリ・フォルダ(I):　参照(W)...

複数のユーザーとプロジェクトが同じコンパイル・ライブラリを共有できます(詳細はヘルプを参照)。

☑ メイン・クラスの作成(C) lecture1.Lecture1

確認

終了を押す

< 戻る(B)　次 >　終了(F)　取消　ヘルプ(H)

[プロジェクトの作成]をしないことには始まらない

プロジェクトの作成

プロジェクトウィンドウ

テキストエディタ

Lecture1 - NetBea

リファクタリング(A) 実行(R) デバッグ(D)

<デフォルト構成>

ファイル　サービス

- Lecture1
  - ソース・パッケージ
    - lecture1
      - Lecture1.java
  - ライブラリ

main - ナビゲータ

メンバー　<空>

- Lecture1
  - main(String[] args)

開始ページ　Lecture1.java

ソース　履歴

```
1  /*
2   * To change this template, choose Tools | Templates
3   * and open the template in the editor.
4   */
5  package lecture1;
6
7  /**
8   *
9   * @author ArkOak
10  */
11 public class Lecture1 {
12
13     /**
14      * @param args the command line arguments
15      */
16     public static void main(String[] args) {
17         // TODO code application logic here
18     }
19 }
20
```

Lecture1 > main >

17 | 44　INS

/*　・・・　*/で囲まれた部分と、//ではじまる行は
プログラムを読みやすくするためのコメント

プロジェクト作成のすすめ

画面左上に、時々こんなのを表示します。

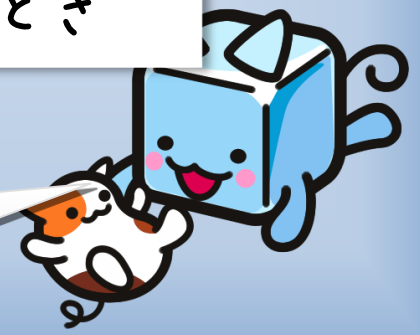
新規プロジェクト
「Lecture1」を作ろう

新しく作ったほうが
手っ取り早いとき

既存プロジェクト
「Lecture1」を使おう

作ったのを使ったほうが
手っ取り早いとき

あくまで目安



既存プロジェクト
「Lecture1」を使おう

JavaでHello world

[ツール]→[オプション]→[フォントと色]で、
文字のフォント・サイズ・色を編集できる

ここにソースコードを書いていく

```
ArkOak
10  */
11  public class Lecture1 {
12
13      /**
14       * @param args t
15       */
16      public static void main(String[] args) {
17          // TODO code application logic here
18      }
19  }
20
```

メインメソッドは必ず初めに実行される場所！
『System.out.println("Hello Netbeans!");』と書いてみよう

JavaでHello world

入力候補

```
print(Object obj)                void
print(String s)                  void
print(boolean b)                 void
print(char c)                    void
print(char[] s)                  void
print(double d)                  void
print(float f)                   void
print(int i)                     void
print(long l)                    void
printf(String format, Object... args)          PrintStream
printf(Locale l, String format, Object... args) PrintStream
println()                        void
println(Object x)                void
println(String x)                void
println(boolean x)               void
```

```
/**
 *
 * @author Ar
 */
public class Lec

  /**
   * @param
   */
  public static
    System.out.p
  }
}
```

java.io.PrintStream

public void **println**(String x)

String を出力して、行を終了します。このメソッドは、print(String) を呼び出してから println() を呼び出すのと同じように動作します。

パラメータ:
x – 出力される String 値

メソッドの説明

入力候補を出したいときは、「Ctrl」+［スペース］

JavaでHello world

ボタンを押すと実行

```
public class Lecture1 {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        System.out.println("Hello NetBeans!");
    }
}
```

```
run:
Hello NetBeans!
ビルド成功(合計時間: 0秒)
```

「Hello NetBeans!」と出力されたら成功！

データの入れ物「変数」

変数とは

〔データ型〕〔変数名〕;

の形で宣言される、数値や文字を入れておく入れ物です。

データの型	種類	サイズ	扱える範囲
short	整数	2バイト	−32786~32767
int	整数	4バイト	−2147483648~2147483647
long	整数	8バイト	−9223372036854775808~9223372036854775807
float	浮動小数点数	4バイト	少数以下約7桁までの値
double	浮動小数点数	8バイト	少数以下約14桁までの値
char	文字定数	2バイト	Unicode文字(一文字)
String	文字列	不定	大体どんだけでも
boolean	論理値	1バイト	true, false

既存プロジェクト
「Lecture1」を使おう

変数を使ってみよう

Lecture1 - NetBeans IDE 7.3

変数の宣言

値の代入

各値の表示

```
16      public static void main(String
17          int a;
18          double b;
19          String s;
20
21          a = 28;
22          b = 3.1415926;
23          s = "こんにちは NetBeans 7.3";
24
25          System.out.println("aの値は" + a);
26          System.out.println("bの値は" + b);
27          System.out.println("sの値は" + s);
28      }
29  }
```

出力 - Lecture1 (run)

```
aの値は28
bの値は3.1415926
sの値はこんにちは NetBeans 7.3
ビルド成功(合計時間: 0秒)
```

int a = 28; のように、宣言と代入は同時にできる

新規プロジェクト
「Lecture2」を作ろう

算術演算でお遊び

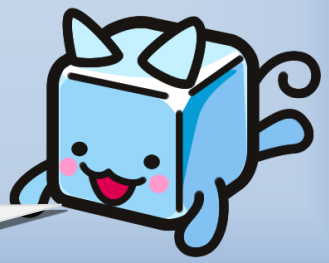
```
public static void main(String[] args) {

    int a = 16;
    int b = 4;

    System.out.println("a = " + a + ", b = " + b);
    System.out.println("a + b = " + (a + b));
    System.out.println("a - b = " + (a - b));
    System.out.println("a * b = " + (a * b));
    System.out.println("a / b = " + (a / b));
}
```

「+」で変数や文字を繋げられる

出力 - Lecture1 (run)

```
run:
a = 16, b = 4
a + b = 20
a - b = 12
a * b = 64
a / b = 4
構築成功（合計時間: 0 秒）
```

C言語ではprintf("%d", a);とかでやってた



既存プロジェクト
「Lecture2」を使おう

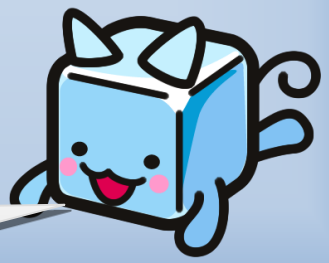
Mathクラスでお遊び

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

    double a = 2.0;
    double b = 5.0;

    System.out.println("a = " + a + ", b = " + b);
    System.out.println("a ^ b = " + Math.pow(a, b));
    System.out.println("sin(π/a) = " + Math.sin(Math.PI/a));
    System.out.println("cos(π/b) = " + Math.cos(Math.PI/b));
    System.out.println("log(a + b)= " + Math.log(a + b));

}
```

出力 - Lecture1 (run)

```
run:
a = 2.0, b = 5.0
a ^ b = 32.0
sin(π/a) = 1.0
cos(π/b) = 0.8090169943749475
log(a + b)= 1.9459101490553132
構築成功（合計時間: 0 秒）
```

『Math.』と打ち込んで候補を観察してみよう



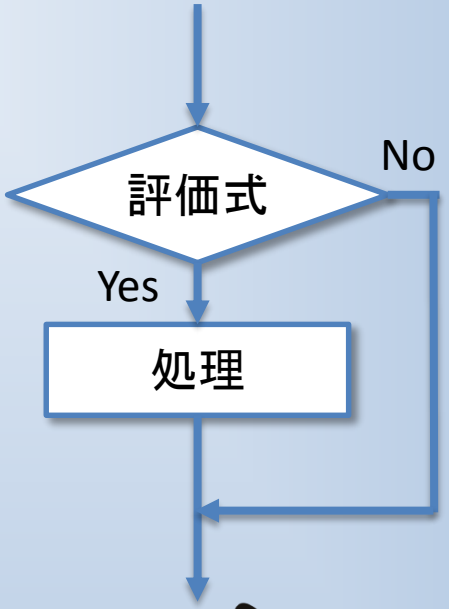
新規プロジェクト
「Lecture3」を作ろう

if文

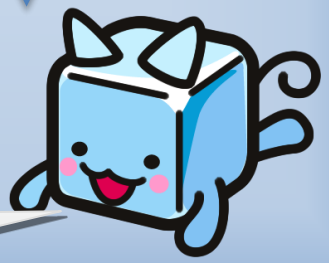
if（評価式）{ 処理 }

の形で記述され、評価式の結果が正しければ処理が実行されます。

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

    int a = 3, b = 5;

    if (a > b) {
        System.out.println("aはbよりも大きいよ");
    }
    if (a < b) {
        System.out.println("aはbよりも小さいよ");
    }
    if (a == b) {
        System.out.println("aとbは等しいよ");
    }
}
```



プログラムの分岐を作る、超重要な文法！



既存プロジェクト
「Lecture3」を使おう

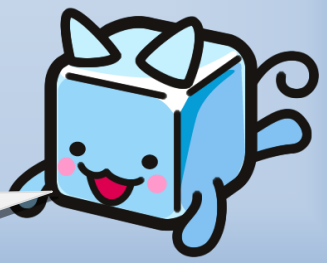
Boolean型でお遊び

『true』か『false』かどちらかの値（真偽値）のみを持つ変数

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

    int a = 3, b = 5;
    boolean big, small, equal;

    big   = (a > b);
    small = (a < b);
    equal = (a == b);

    System.out.println("big = " + big);
    System.out.println("small = " + small);
    System.out.println("equal = " + equal);
}
```

変数として
扱えるようになった

1byteしかメモリを使わない謙虚な型
後々とても重要になってくるよ

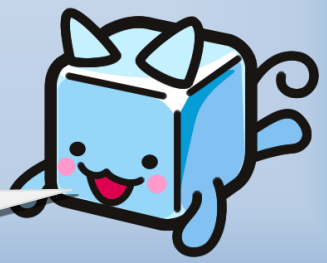


既存プロジェクト
「Lecture3」を使おう

Boolean型でお遊び

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

    int a = 3, b = 5;
    boolean big, small, equal;

    big   = a > b;
    small = a < b;
    equal = a == b;

    if (big) {
        System.out.println("aはbよりも大きいよ");
    }
    if (small) {
        System.out.println("aはbよりも小さいよ");
    }
    if (equal) {
        System.out.println("aとbは等しいよ");
    }
}
```

if文の中の比較結果は、trueかfalseで表現できる



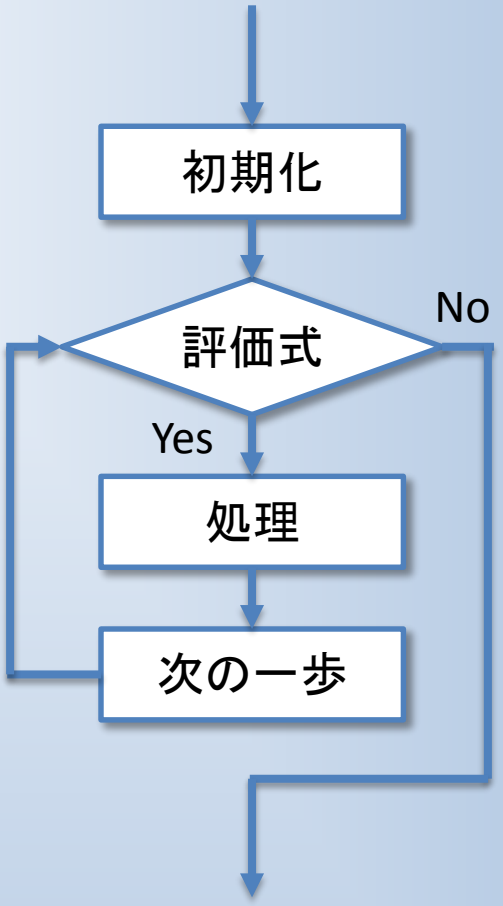
新規プロジェクト
「Lecture4」を作ろう

for文

for（初期化；評価式；次の一歩）{ 処理 }

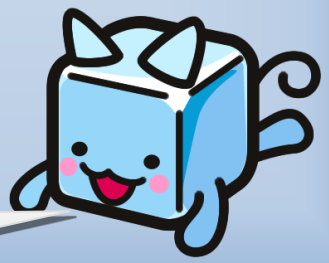
の形で記述され、次の流れで実行される。

1. 初期化を行う。
2. 評価式が正しければ、処理を実行する（違えば終了）。
3. 処理が終わったら、次の一歩を行う。
4. 手順2に戻る。




```
public static void main(String[] args) {
    int i;
    for(i = 0; i < 10; i++){
        System.out.println("i = " + i);
    }
}
```

iの値を1～10にしたければ、for(i = 1; i <= 10; i++)とか



既存プロジェクト
「Lecture4」を使おう

演習問題①

プログラムの仕様

- 1から500までの和を表示する

ヒント
結果用の変数 sum を用意して
for文で数字を加えていこう

実行結果

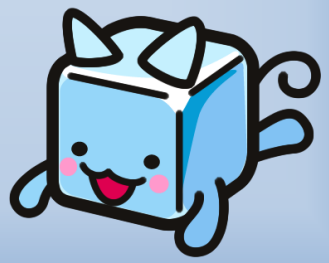
出力 - Lecture1 (run)

```
run:
1から500までの和は、125250
構築成功（合計時間: 0 秒）
```

解答例

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

    //1から500までの和を求めるプログラム
    int sum = 0;

    //forループで1から500の数字を、sumに加えていく
    for(int i = 1; i <= 500; i++){

    }

    System.out.println("1から500までの和は、" + sum);
}
```

ちら



新規プロジェクト
「Lecture5」を作ろう

演習問題②

プログラムの仕様

- 9×9の九九の表を表示する。
- きれいに揃えて出力する(任意)。

ヒント

print()メソッドを使うと改行されない。
for文2つとif文1つ使うとできそう。

実行結果

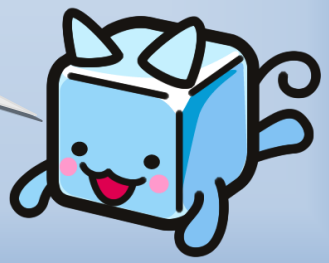
出力 - Lecture1 (run)

```
run:
 1  2  3  4  5  6  7  8  9
 2  4  6  8 10 12 14 16 18
 3  6  9 12 15 18 21 24 27
 4  8 12 16 20 24 28 32 36
 5 10 15 20 25 30 35 40 45
 6 12 18 24 30 36 42 48 54
 7 14 21 28 35 42 49 56 63
 8 16 24 32 40 48 56 64 72
 9 18 27 36 45 54 63 72 81
構築成功 (合計時間: 0 秒)
```

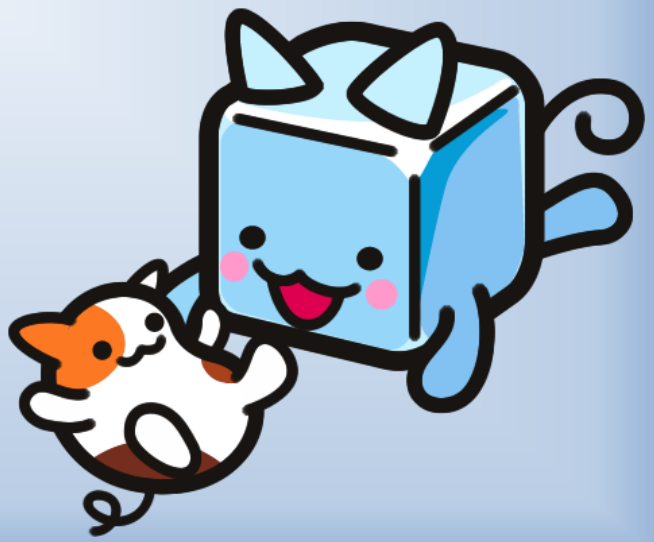
解答例

```
//九九の表を表示するプログラム
for (int i = 1; i <= 9; i++) {
    for (int j = 1; j <= 9; j++) {
        System.out.print((i * j) + " ");
    }
    System.out.println();
}
```

作ってみよう

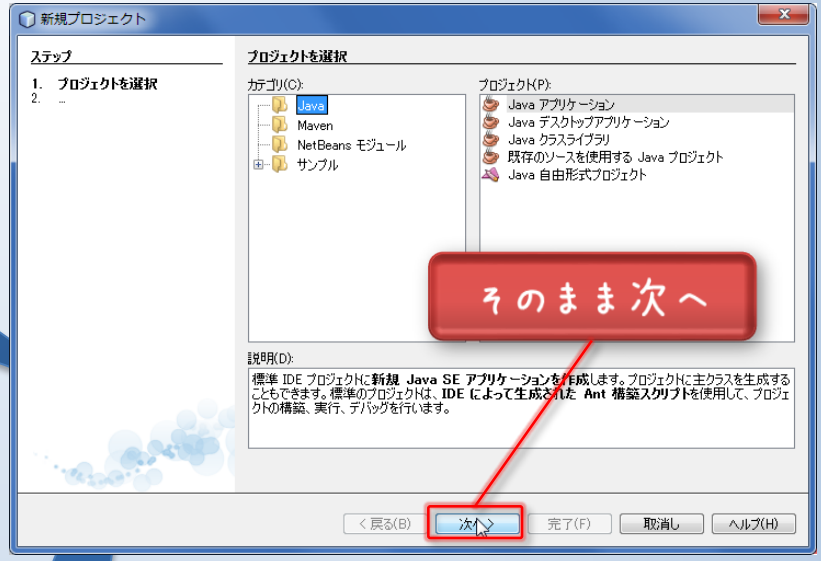


3. ソフトウェアの作り方

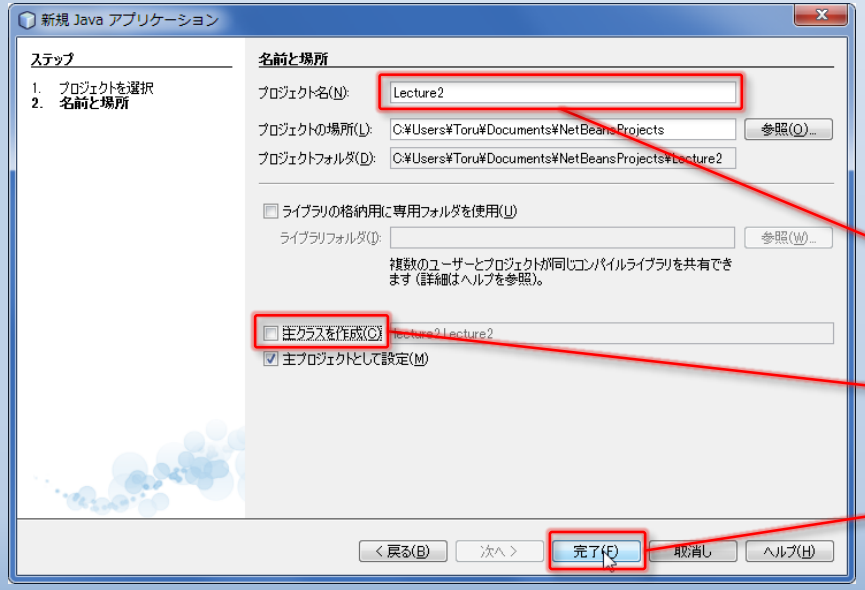


新規プロジェクト
「App1」を作ろう

プロジェクトの作成



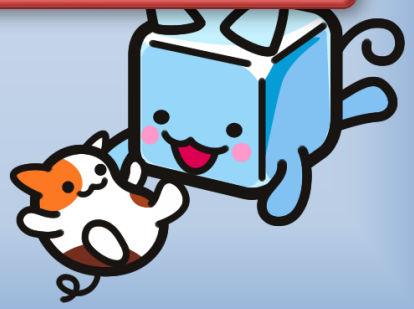
そのまま次へ



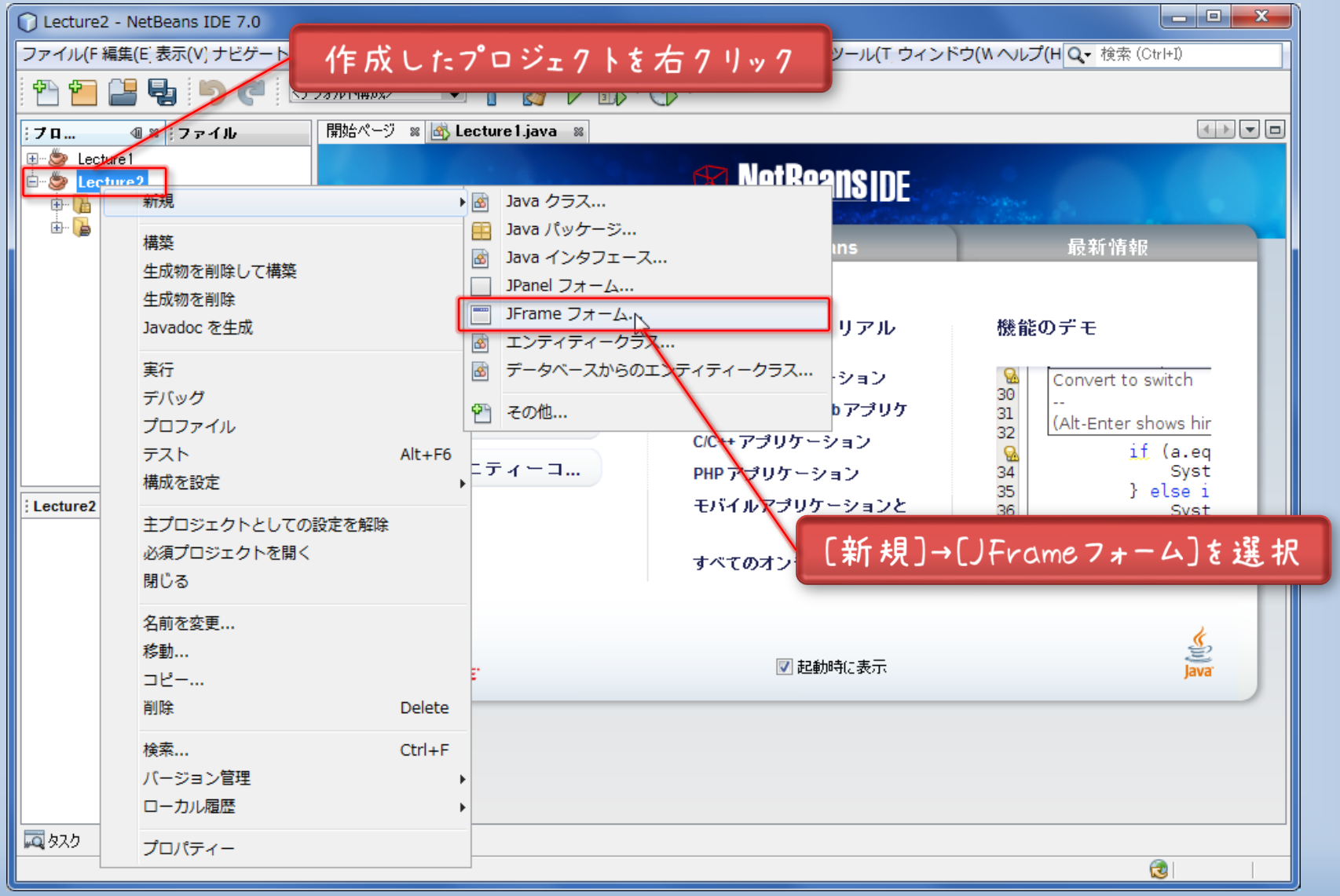
プロジェクトの名前をつける（App1）

チェックを外す

完了を押す



フォームの作成


Lecture2 - NetBeans IDE 7.0
作成したプロジェクトを右クリック
〔新規〕→〔JFrameフォーム〕を選択
Lecture1
Lecture2
新規
構築
生成物を削除して構築
生成物を削除
Javadoc を生成
実行
デバッグ
プロファイル
テスト Alt+F6
構成を設定
主プロジェクトとしての設定を解除
必須プロジェクトを開く
閉じる
名前を変更...
移動...
コピー...
削除 Delete
検索... Ctrl+F
バージョン管理
ローカル履歴
プロパティー
Java クラス...
Java パッケージ...
Java インタフェース...
JPanel フォーム...
JFrame フォーム...
エンティティークラス...
データベースからのエンティティークラス...
その他...
開始ページ
Lecture1.java
NetBeans IDE
最新情報
機能のデモ
起動時に表示

フォームの作成

クラス名を付ける(App1)

新規 JFrame フォーム

ステップ

1. ファイルの種類を選択
2. 名前と場所

名前と場所

クラス名(N): MainFrame

プロジェクト(P): Lecture2

場所(L): ソースパッケージ

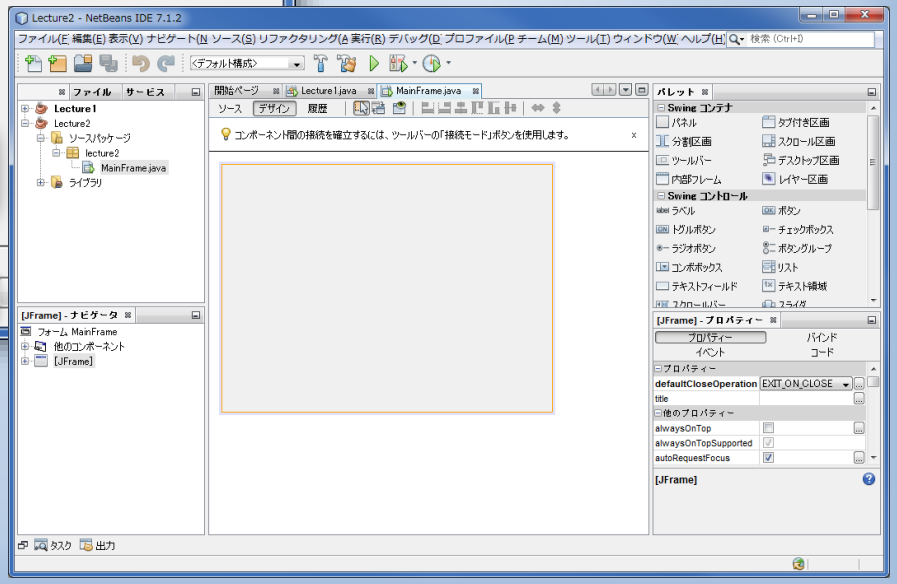
パッケージ(K): lecture2

作成されるファイル(C): ~oru¥Documents¥NetBeansProjects¥Lecture2¥src¥lecture2¥MainFrame.java

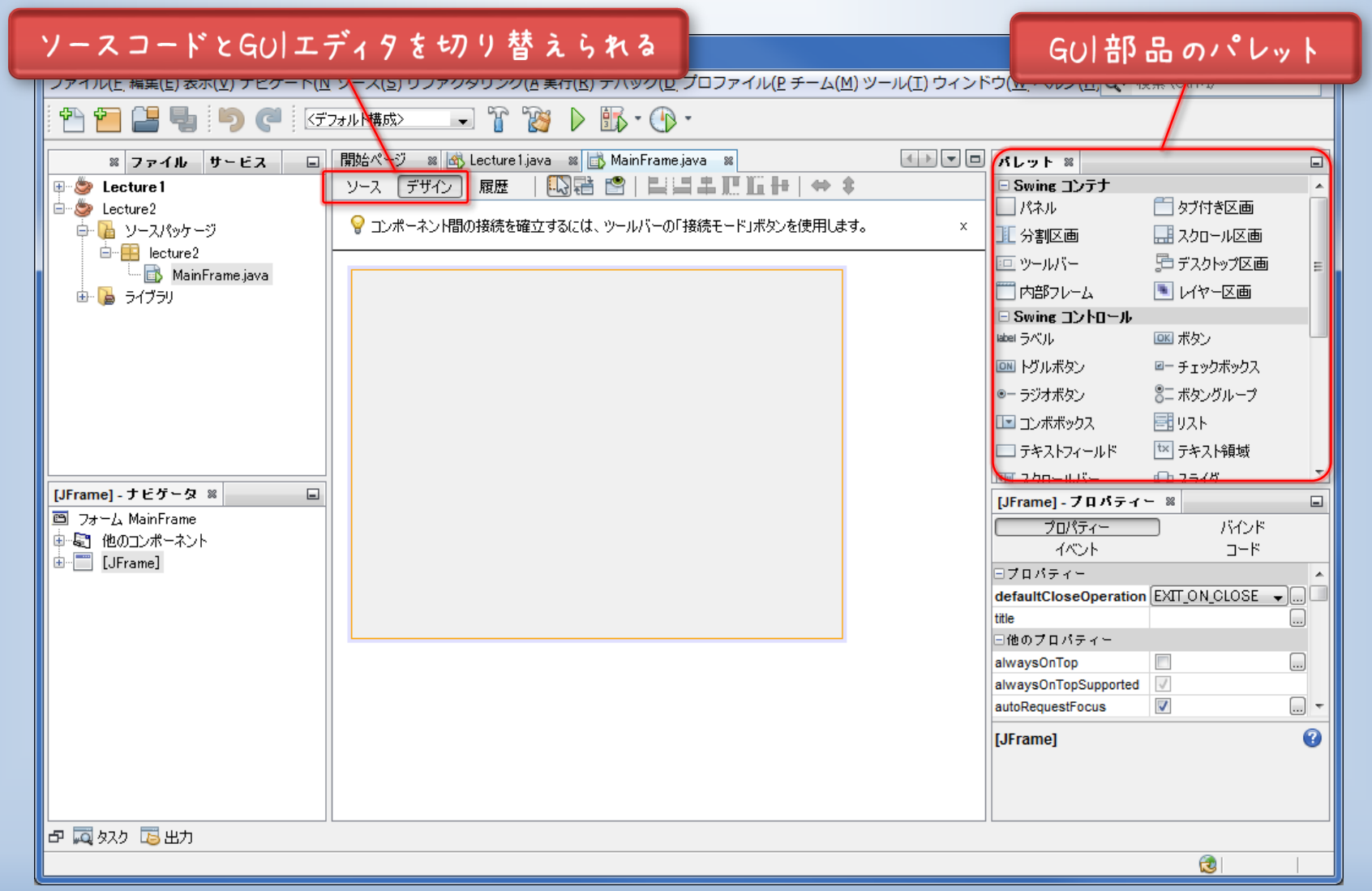
適当にパッケージ名も付ける(app1)

Jframeフォーム作成の完了

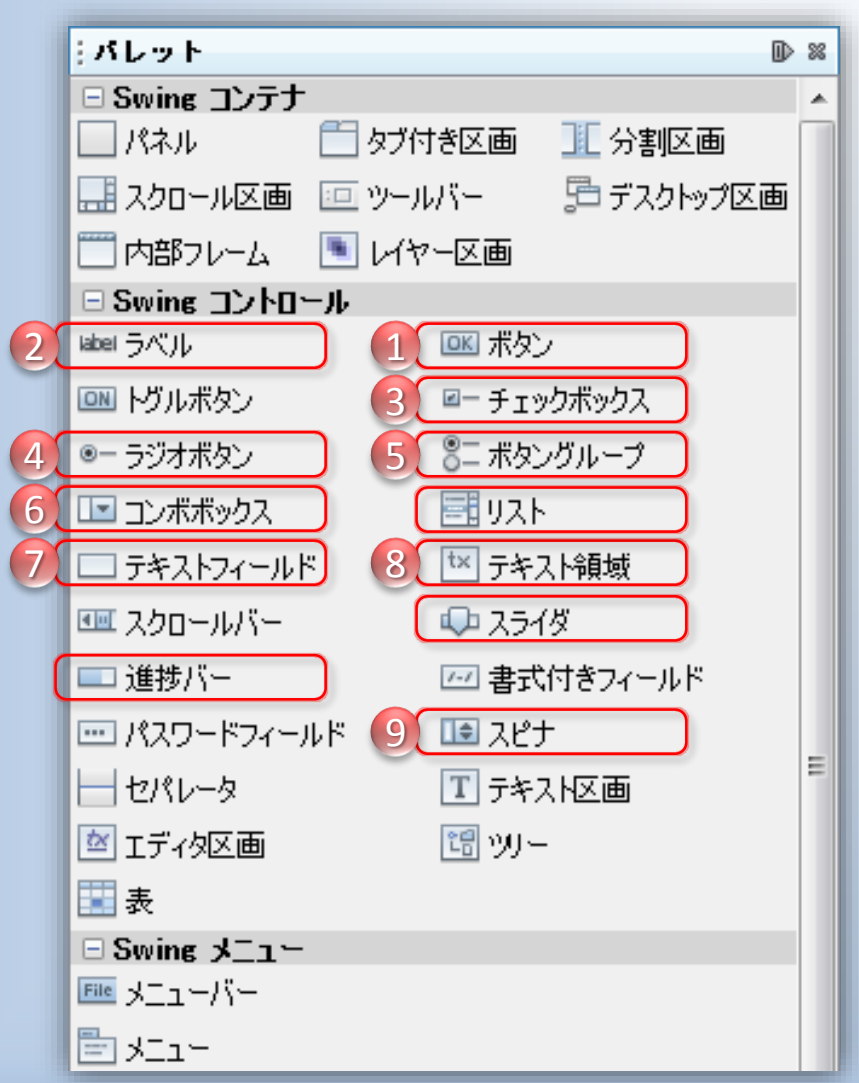
< 戻る(B) 次へ > 完了(F)



フォームの作成


ソースコードとGUIエディタを切り替えられる
GUI部品のパレット
ファイル
サービス
Lecture1
Lecture2
ソースパッケージ
lecture2
MainFrame.java
ライブラリ
開始ページ
Lecture1.java
MainFrame.java
ソース
デザイン
履歴
コンポーネント間の接続を確立するには、ツールバーの「接続モード」ボタンを使用します。
パレット
Swing コンテナ
パネル
タブ付き区画
分割区画
スクロール区画
ツールバー
デスクトップ区画
内部フレーム
レイヤー区画
Swing コントロール
ラベル
ボタン
トグルボタン
チェックボックス
ラジオボタン
ボタングループ
コンボボックス
リスト
テキストフィールド
テキスト領域
[JFrame] - ナビゲータ
フォーム MainFrame
他のコンポーネント
[JFrame]
[JFrame] - プロパティー
プロパティー
バインド
イベント
コード
プロパティー
defaultCloseOperation
EXIT_ON_CLOSE
title
他のプロパティー
alwaysOnTop
alwaysOnTopSupported
autoRequestFocus
[JFrame]
タスク
出力

よく使うGUI部品
(Swingコントロール)



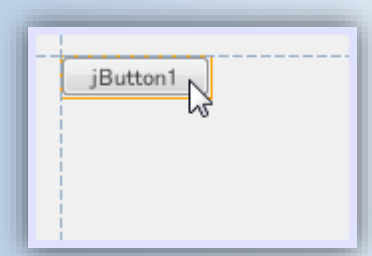
①ボタン 36ページ
②ラベル 38ページ
③チェックボックス 41ページ
④ラジオボタン 43ページ
⑤ボタングループ 44ページ
⑥コンボボックス 45ページ
⑦テキストフィールド 48ページ
⑧テキスト領域 49ページ
⑨スピナ 49ページ

既存プロジェクト
「App1」を使おう

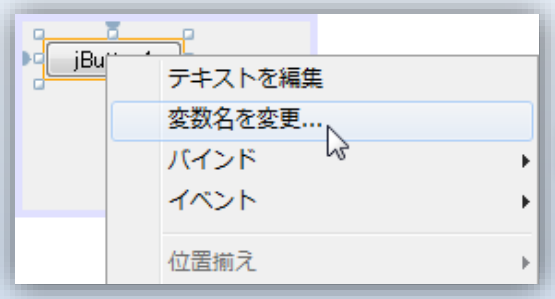
①ボタン

画面上に、機能を持った押しボタンを生成する。

1. ボタンの配置。

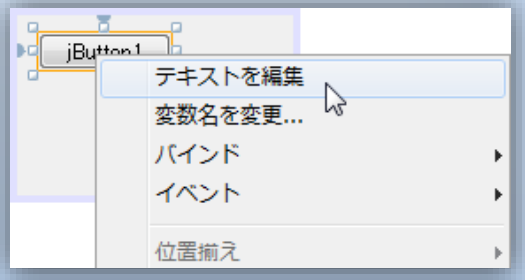


2. 右クリックして「変数名を変更」

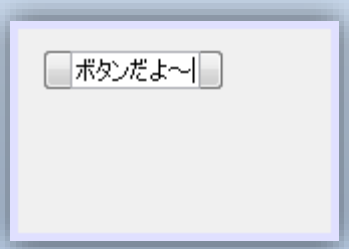


3. 新しい変数名を入力。

4. 右クリックして「テキストを編集」



5. テキストを入力。



6. ボタン準備完了！

名前の付け方について

コントトール変数名は、

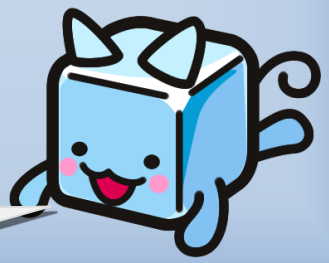
コントロール名 + 機能説明

といった感じでつける！

「例」

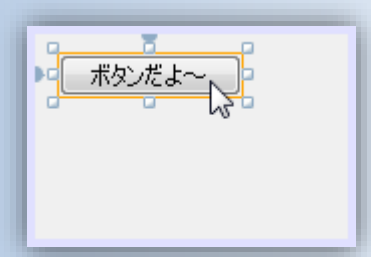
- 開始ボタン → btnStart
- アニメ開始ボタン → btnStartAnimation
- 終了ボタン → buttonEnd
- 描画ラベル → lblDraw
- 設定コンボボックス → cmbSetting
- 出力用テキスト領域 → taOutput

変数名の頭は小文字という習慣



①ボタン

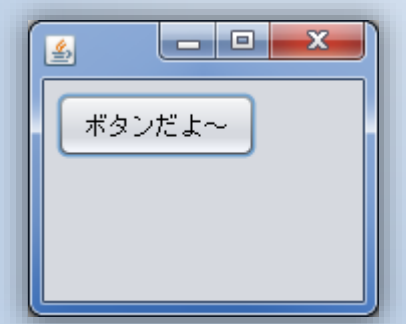
7. ボタンをダブルクリックすると、ソースコードに切り替わる。

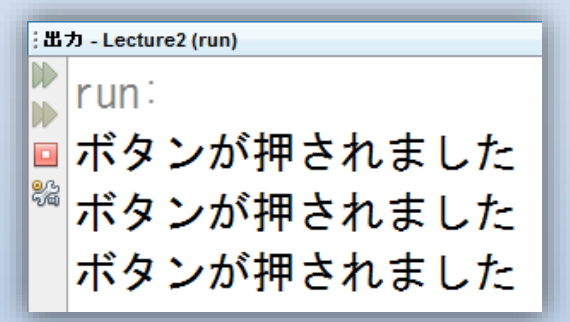


8. 押されたら文字を出力する命令を書いてみる

```
60    private void btnTestActionPerformed(java.awt.event.ActionEvent evt) {
61        System.out.println("ボタンが押されました");
62    }
```

9. 実行結果





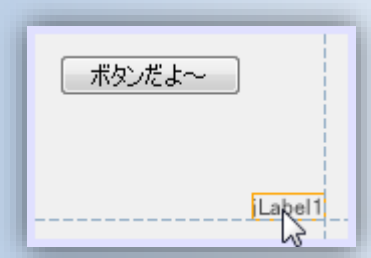
ボタンを押すと何かが起こる

既存プロジェクト
「App1」を使おう

②ラベル

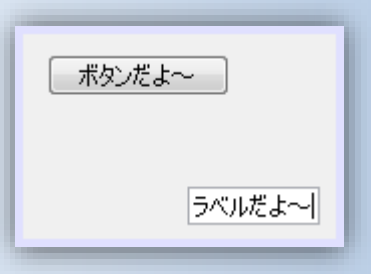
画面上にテキスト(ユーザが変更不可)を生成する。
他にも、描画領域の指定などに利用される。

1. ラベルコントロールの配置。



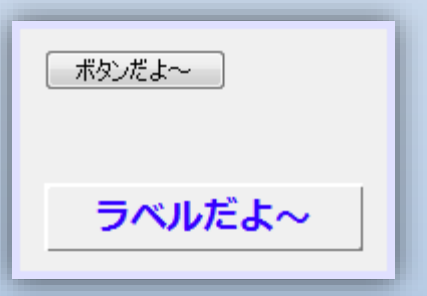
2. 変数名を変更する

3. テキストを変更する

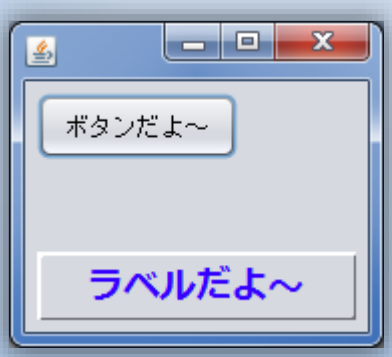


4. ラベル準備完了！

5. ラベルを[右クリック]→
[プロパティ]で色々変えてみる

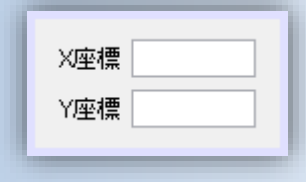


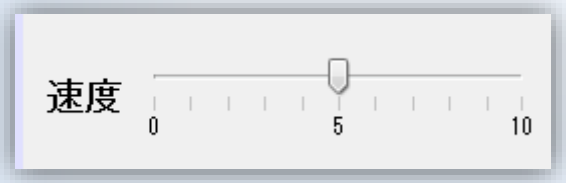
6. 実行結果

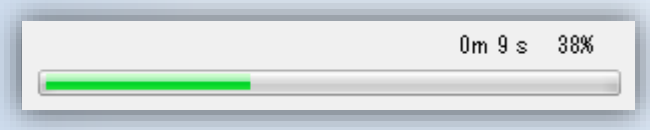


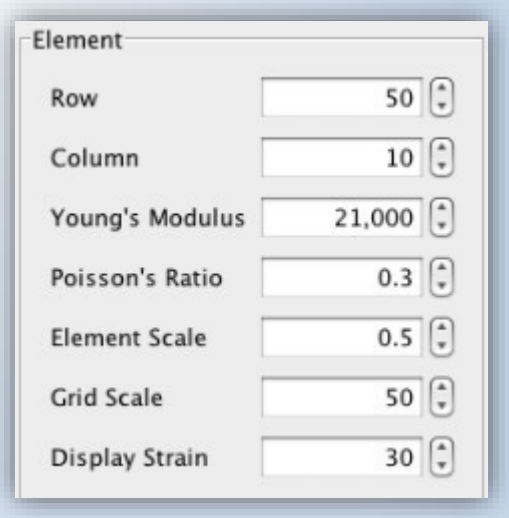
②ラベル

7. 実際のラベルの使用例。





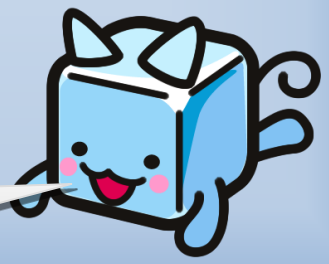




Points!

ラベルのテキストの変更は、
コントロール名.setText()
でおこなう！

ボタンを押した回数をラベルに表示させよう

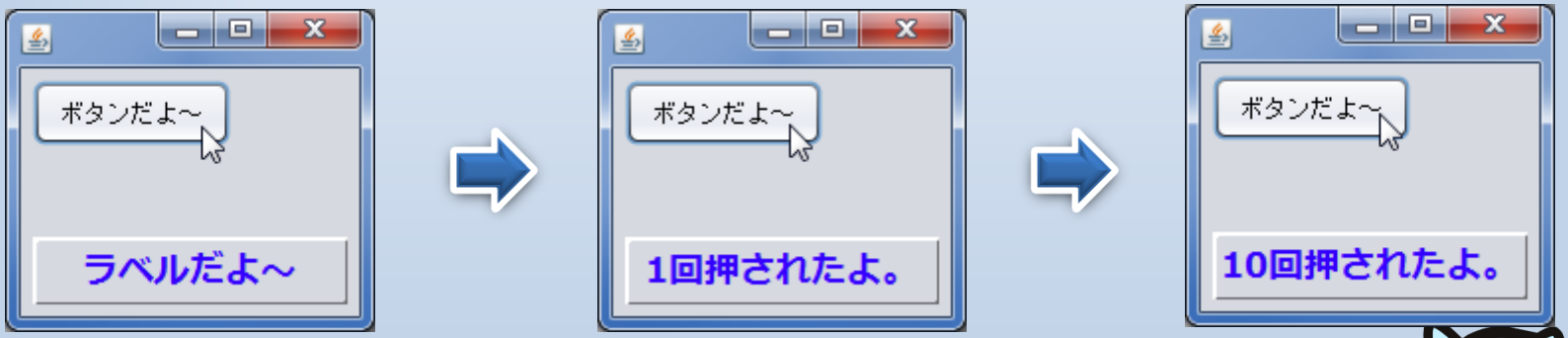


②ラベル

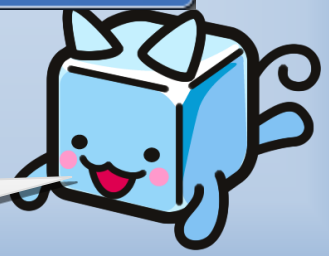
8. ボタンをダブルクリックして、次のようにソースコードを記述します。

```
73      int i = 0;
74      private void btnTestActionPerformed(java.awt.event.ActionEvent evt) {
75          //ボタンが押された回数をカウントし、ラベルに表示する処理
76          i++;
77          lblTest.setText( i + "回押されたよ。");
78      }
```

9. 実行結果



なんとも言えないカウンタのできあがり。

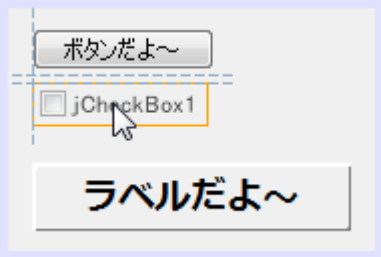


既存プロジェクト
「App1」を使おう

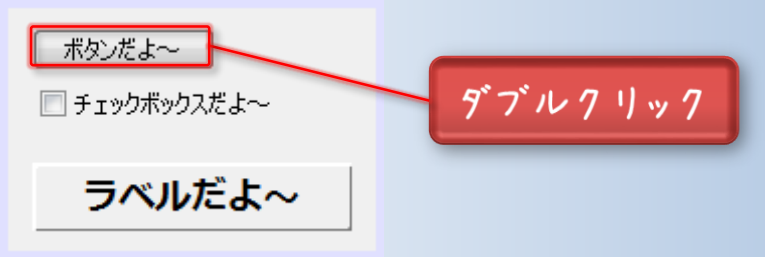
③チェックボックス

画面上に選択・選択解除できる項目を生成するGUI部品。

1. チェックボックスコントロールの配置。



2. 変数名とテキストを変更。
（今回の変数名は"checkTest"）



3. ボタンの処理を記述するところに移動します。

```
74    private void btnTestActionPerformed(java.awt.event.ActionEvent evt) {
75        // TODO add your handling code here:
76    }
```

Points!

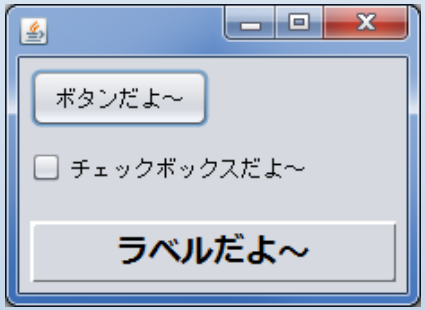
チェックボックスが押されているかどうかは
コントロール名.isSelected()
で判定する！

③チェックボックス

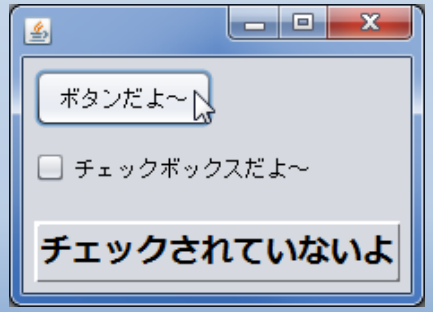
4. 次のようにソースコードを記述します。

```
79    private void btnTestActionPerformed(java.awt.event.ActionEvent evt) {
80        //チェックボックスが押されているかどうかを、ラベルに表示する処理
81        if (checkTest.isSelected()) {
82            lblTest.setText("チェックされているよ");
83        } else {
84            lblTest.setText("チェックされていないよ");
85        }
86
87    }
```

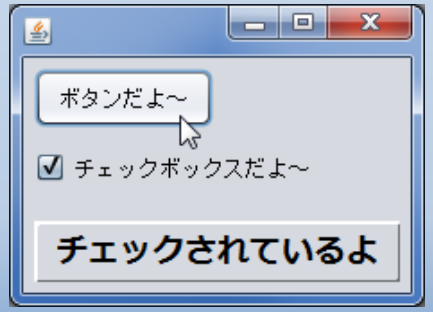
5. 実行してみましょう。



そのままボタンを押した場合



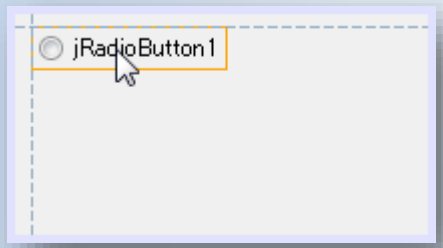
チェックしてボタンを押した場合



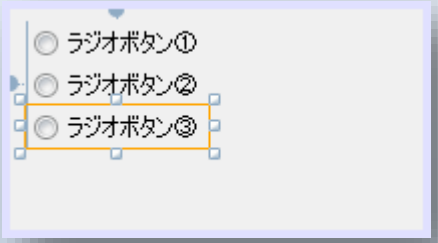
④ラジオボタン

画面上に選択・選択解除できる項目を生成する。
ButtonGroupと組み合わせることで、1つのみを選択できるボタングループを作成できる。

1. ラジオボタンを配置。



2. 一個じゃ意味があまり無いので、
三個くらい配置します。



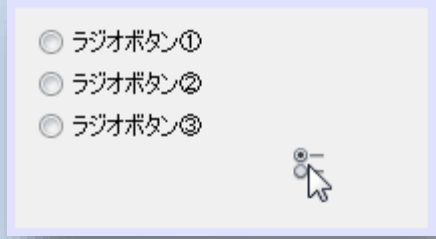
Points!

ラジオボタンが押されているかどうかも
コントロール名.isSelected()
で判定する！

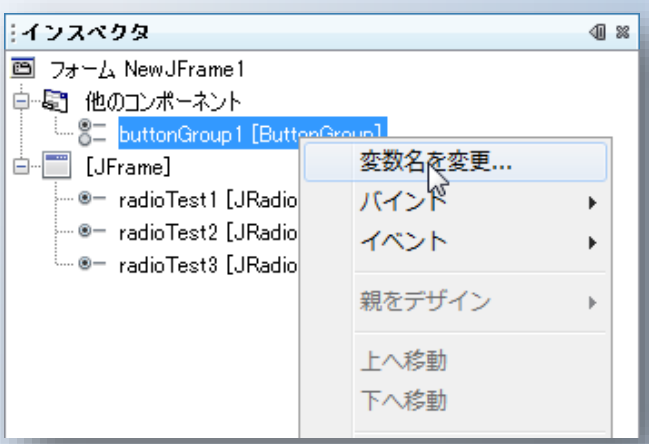
このままではチェックボックス3個と同じ。

⑤ボタングループ

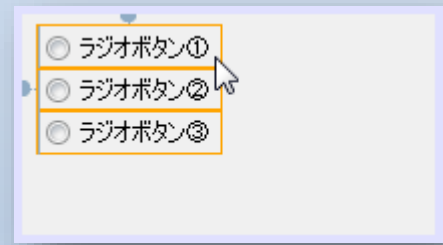
1. ボタングループを適当に配置します。
（デザインビュー上では見えません）

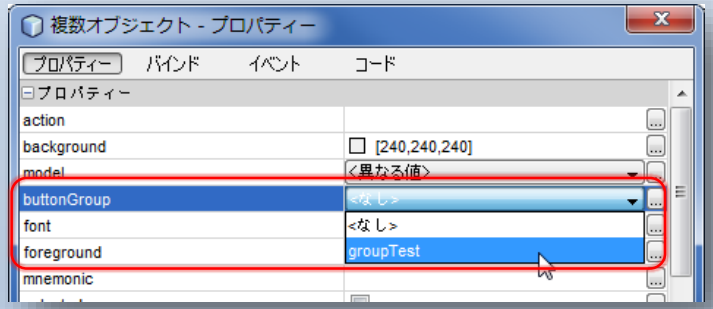


2. インスペクタ画面で変数名を変更できます。
（今回は変数名を"groupTest"に変更）



3. ラジオボタンを選択し、
〔右クリック〕→〔プロパティ〕→〔buttonGroup〕
で"groupTest"を選択。





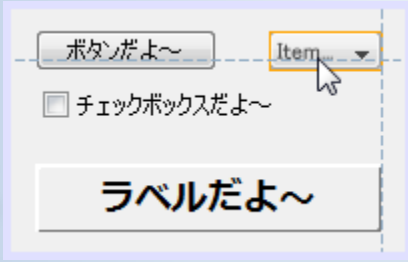
グループ化されはボタンは、複数選択できなくなる。

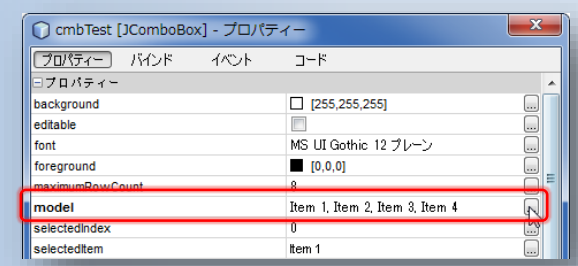
既存プロジェクト
「App1」を使おう

⑥コンボボックス

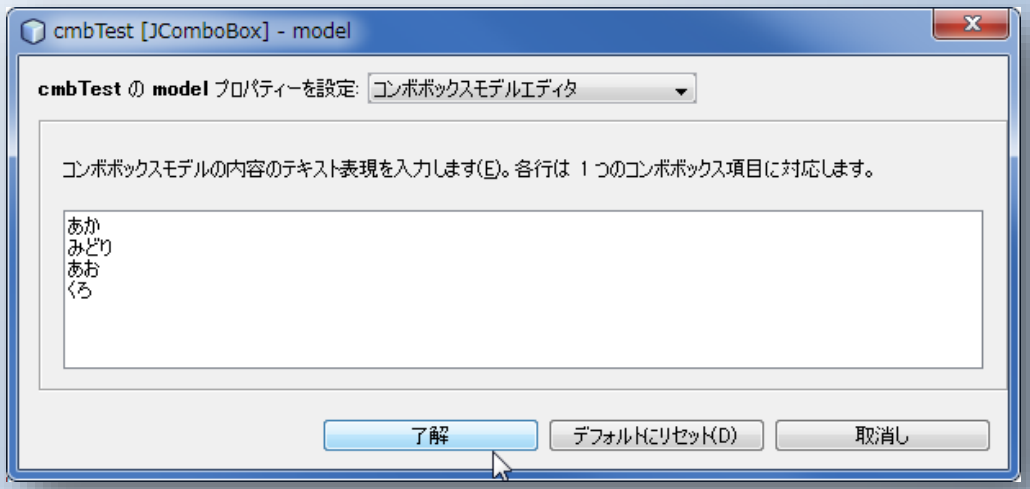
ボタンとリストを組み合わせた選択画面を生成する。

1. コンボボックスを配置。
（変数名は"cmbTest"）





2. [右クリック]→[プロパティ]→[model]を選択し、
リストに加える項目を記述します。



Points!

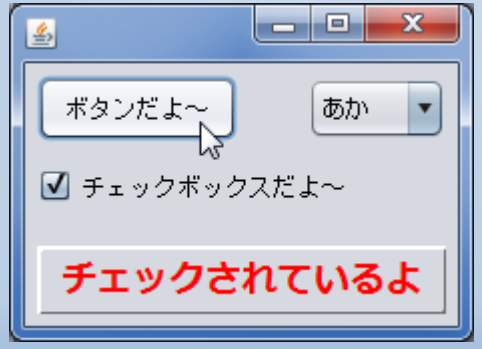
コンボボックスのどれが選択されているかは
コントロール名.getSelectedIndex()
で判定する！

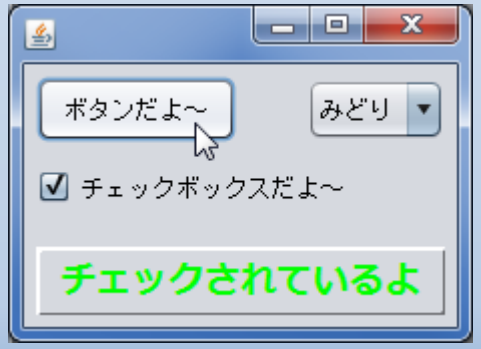
⑥コンボボックス

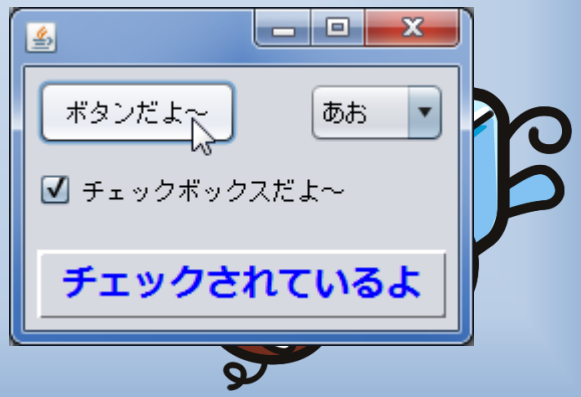
3. ボタンをダブルクリックし、次のソースコードを記述します。

```
//コンボボックスに応じて、ラベルの色を変更する処理
int index = cmbTest.getSelectedIndex();

if (index == 0) {
    lblTest.setForeground(Color.red);
} else if (index == 1) {
    lblTest.setForeground(Color.green);
} else if (index == 2) {
    lblTest.setForeground(Color.blue);
} else {
    lblTest.setForeground(Color.black);
}
```

4. 実行してみましょう。





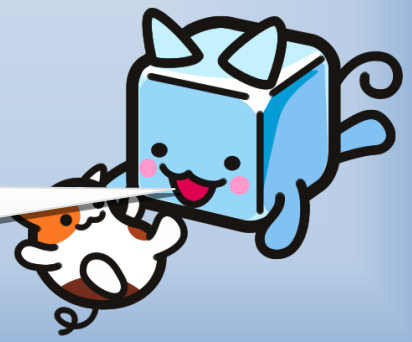


⑥コンボボックス

5. おまけ

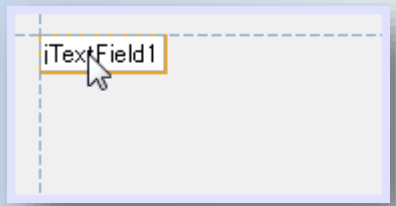
```
//コンボボックスに応じて、ラベルの色を変更する処理
int index = cmbTest.getSelectedIndex();

switch(index) {
    case 0:
        lblTest.setForeground(Color.red);
        break;
    case 1:
        lblTest.setForeground(Color.green);
        break;
    case 2:
        lblTest.setForeground(Color.blue);
        break;
    default:
        lblTest.setForeground(Color.black);
}
```

switch文を使ったほうが、ちょっとかっこいい



⑦テキストフィールド

1行のテキスト編集領域を生成する。



Points!

テキストフィールドからテキストの取得は、

コントロール名.getText()

テキストフィールドへテキストの代入は、

コントロール名.setText()

でおこなう！

Points!

取得したString型変数をInt型やDouble型として扱いたい場合は、

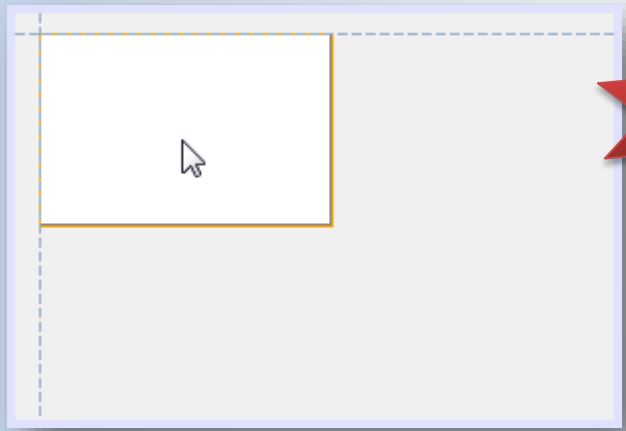
Integer.parseInt(コントロール名.getText())

Double.parseDouble(コントロール名.getText())

といった具合にキャスト(型変換)する！

⑧テキスト領域

複数行のテキスト編集領域を生成する。



Points!

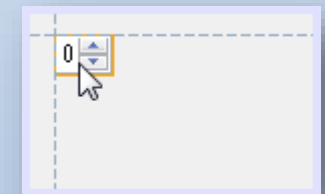
テキストエリアへのテキストの出力は、

コントロール名.append()

でおこなう！

⑨スピナ

要素間を自由に移動できる、1行の入力フィールドを生成する。

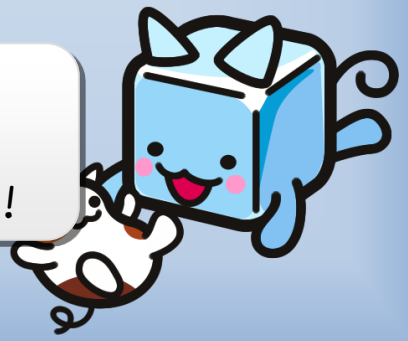


Points!

スピナからの値の取得は、

コントロール名.getValue()

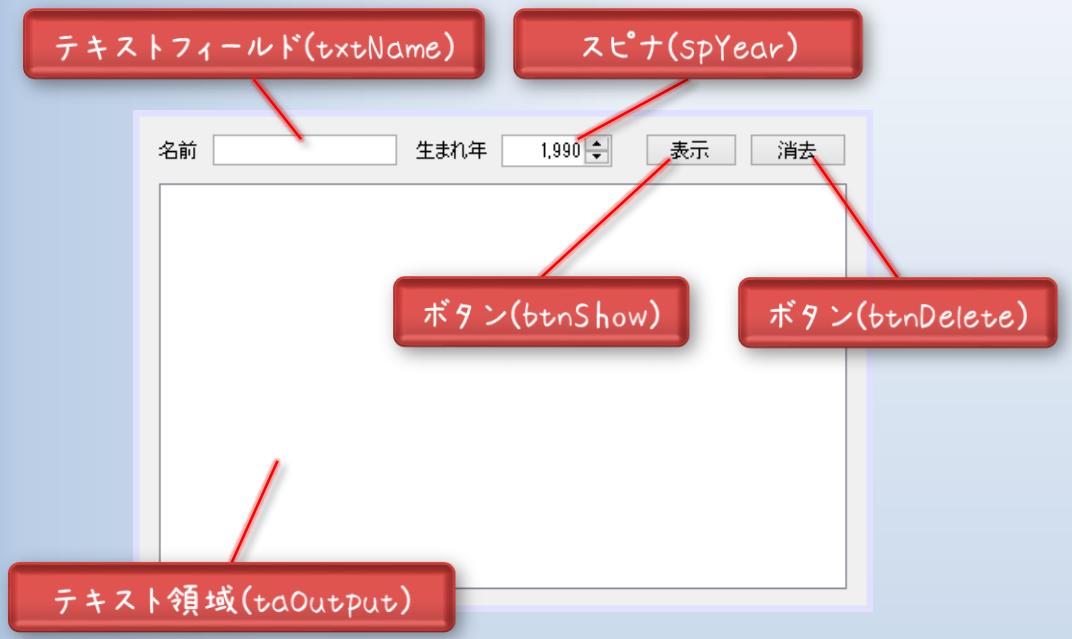
でおこなう！



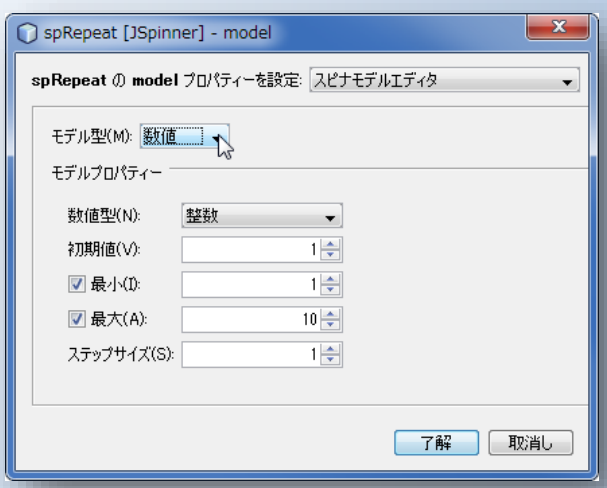
新規プロジェクト
「App2」を作ろう

⑨スピナ

1. 次のようにSwingコントロールを配置しよう。



2. スピナを[右クリック]→[プロパティ]→[model]から、スピナモデルの詳細を設定できます。



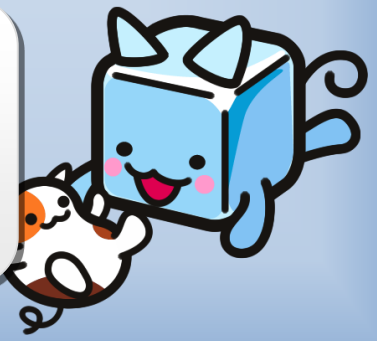
Points!

取得した値はObject型のため、int型やdouble型として扱いたい場合は、

(Integer)(コントロール名.getValue())

(Double)(コントロール名.getValue())

といった具合にキャスト(型変換)する!

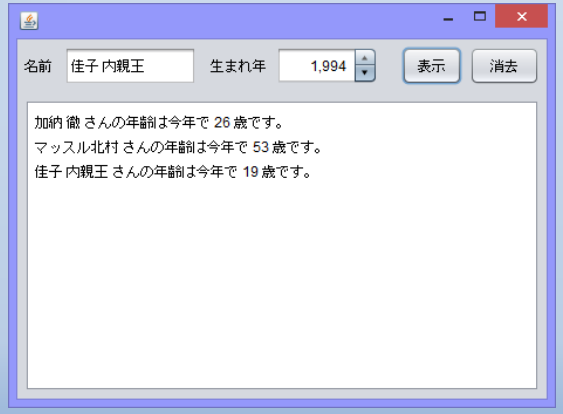


⑨スピナ

3. ボタンをダブルクリックし、次のソースコードを記述します。

```
105    private void btnShowActionPerformed(java.awt.event.ActionEvent evt) {
106        String s;
107        int old;
108
109        s = txtName.getText();
110        old = 2013 - (Integer)spYear.getValue();
111
112        taOutput.append(s + "さんの年齢は今年で" + old + "歳です。¥n");
113    }
114
115    private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
116        taOutput.setText("");
117    }
```

4. 入力した文字と、スピナの数字を出力するプログラム。



スピナやスライダは
ユーザに親切なインタフェース

演習問題③

プログラムの仕様

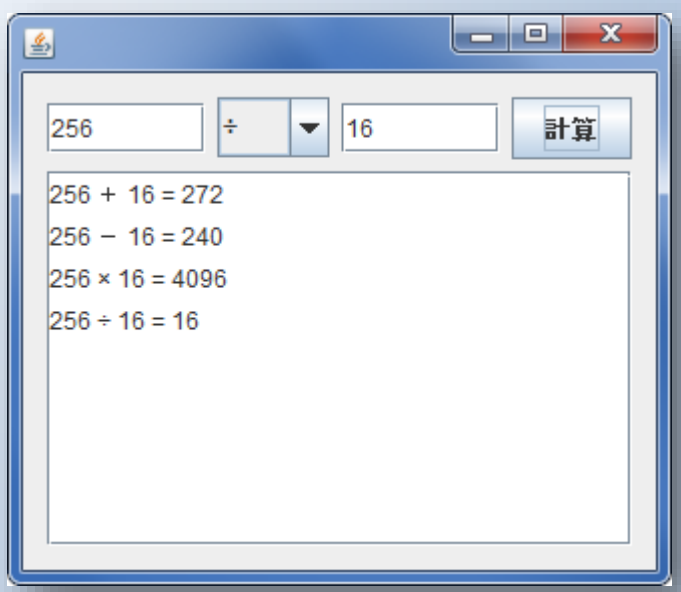
2つの数の四則演算を行うプログラム。

- 2つの数はテキストフィールド(スピナ)で入力
- 「+」「−」「×」「÷」をコンボボックスで選択
- 計算ボタンでテキストエリアに結果を出力

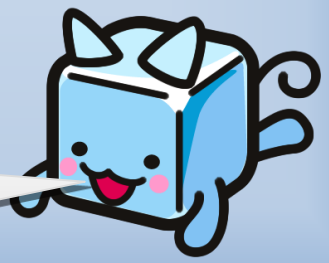
ヒント

なし

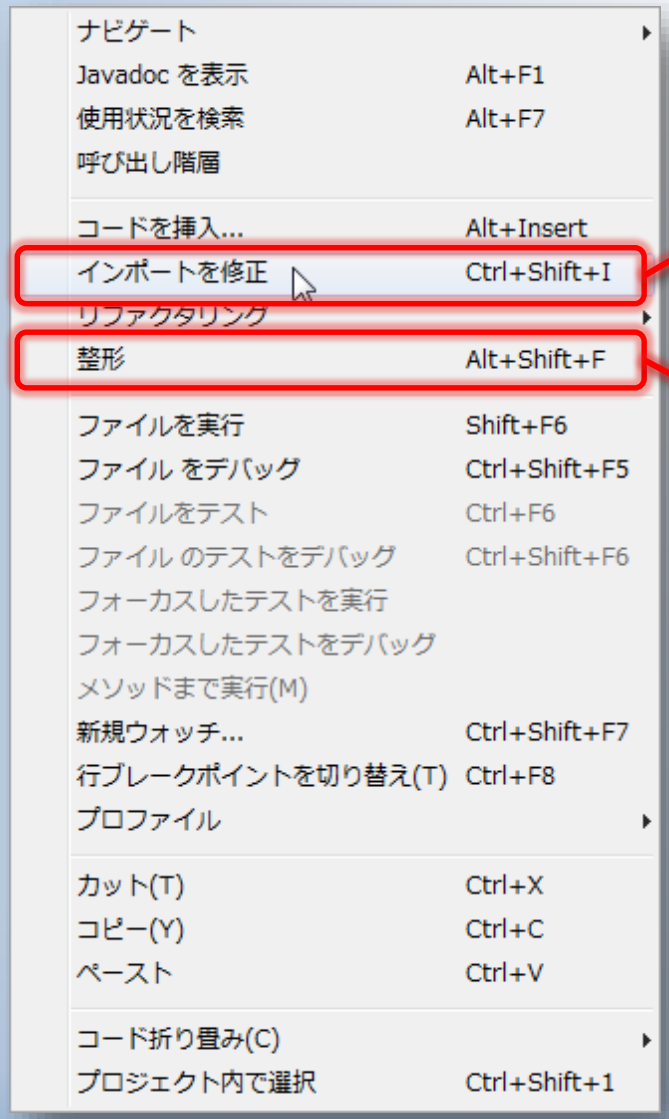
実行例



小数を含めた計算や、
0割りなどのエラー処理も考えてみよう



NetBeansの便利な機能



自動的に必要なライブラリ等をインポートしてくれる機能

インデント(字下げ)を自動的に調整してくれる機能

エディタ上のどこかで右クリックすると出てくる

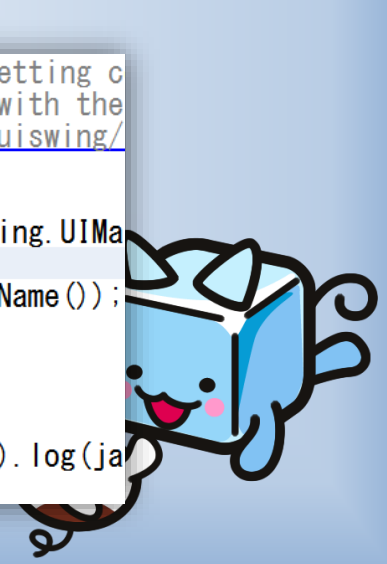
Look and Feelについて

```
97      /**
98       * @param args the command line arguments
99       */
100     public static void main(String args[]) {
101         /* Set the Nimbus look and feel */
102         Look and feel setting code (optional)
123
124         /* Create and display the form */
125         java.awt.EventQueue.invokeLater(new Runnable() {
126
            public void run() {
128             new Drawing().setVisible(true);
129         }
130         });
131     }
```

ここに隠れているソースコードを展開する

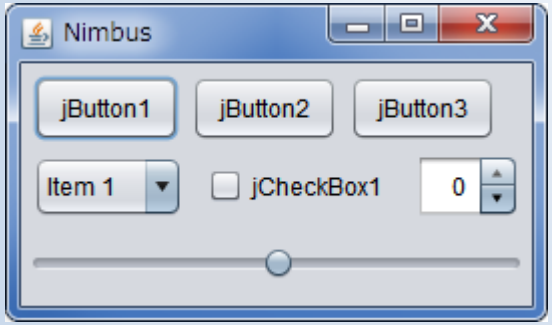
ここの文字を変更する

```
102         //<editor-fold defau                                  feel setting c
103         /* If Nimbus (introduced in Java SE 6) is not available, stay with the
104          * For details see http://download.oracle.com/javase/tutorial/uiswing/
105          */
106         try {
107             for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIMa
                    if ("Nimbus".equals(info.getName())) {
109                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
110                     break;
111                 }
112             }
113         } catch (ClassNotFoundException ex) {
114             java.util.logging.Logger.getLogger(Drawing.class.getName()).log(ja
115         } catch (InstantiationException ex) {
```

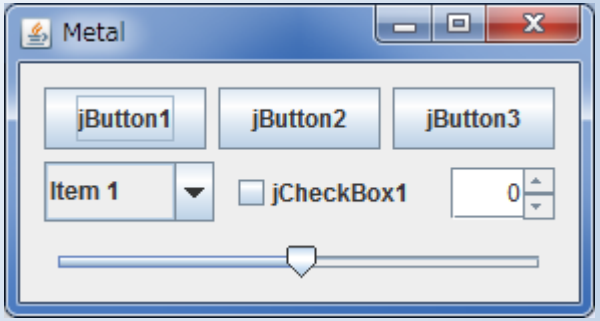


Look and Feelについて

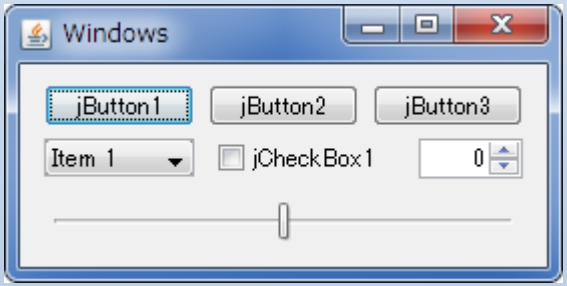
Nimbus



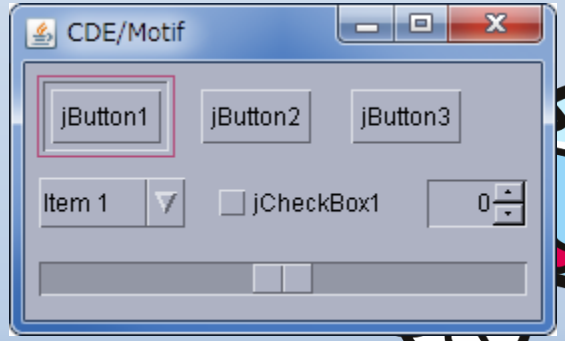
Metal



Windows



CDE/Motif



次回のJava実習（もしやるなら‥）

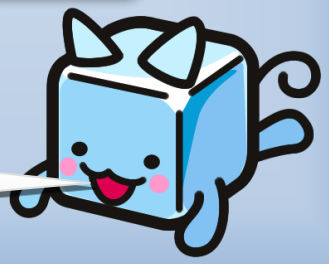
1. Swingによるお絵かき
2. Threadによるアニメーション
3. マウス処理の実装
4. ファイルの入出力処理

Ark Oak の宣伝(おまけ)

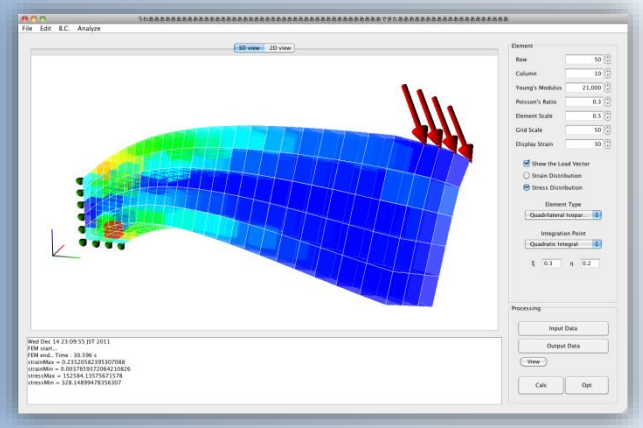
大学向けソフトウェア開発ベンチャー

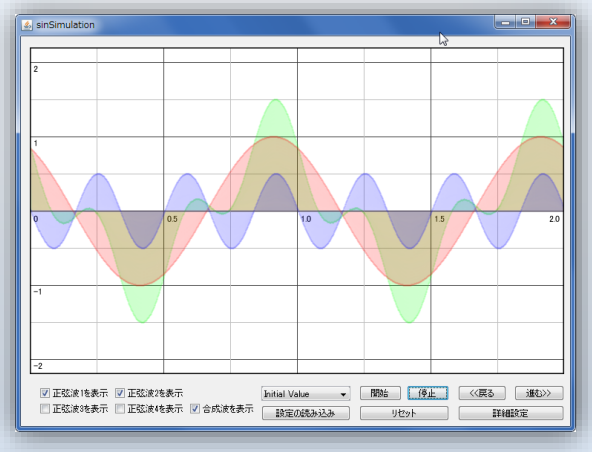
顧客	活動内容
大学教員	プログラミングの勉強
研究職員	教育支援ソフトウェアの開発
大学生	研究支援ソフトウェアの開発

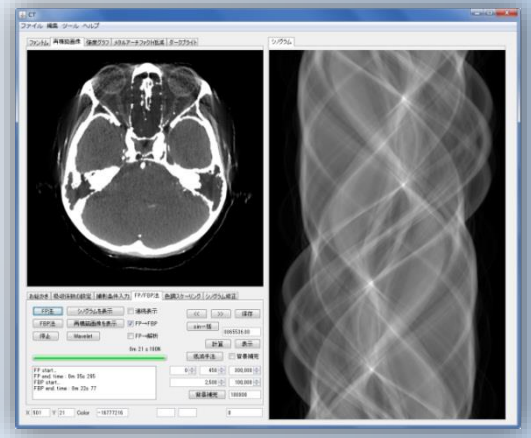
大学向けソフトウェアの需要はものすごく高い

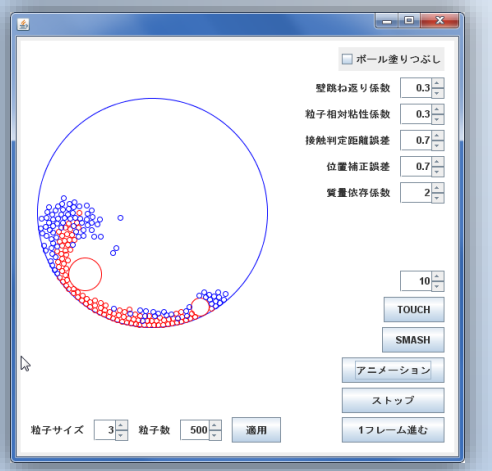


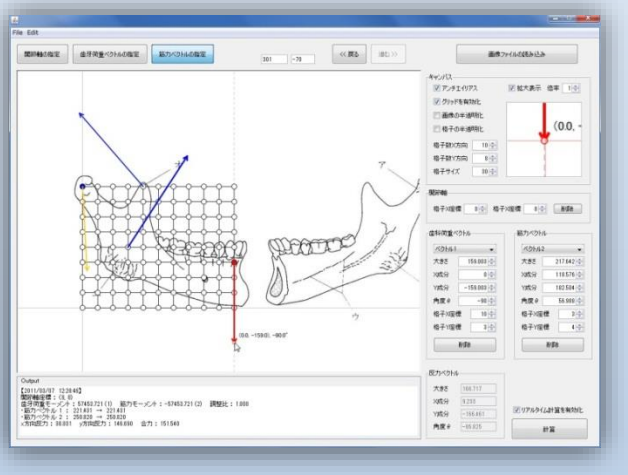
これまでに開発したソフトウェア

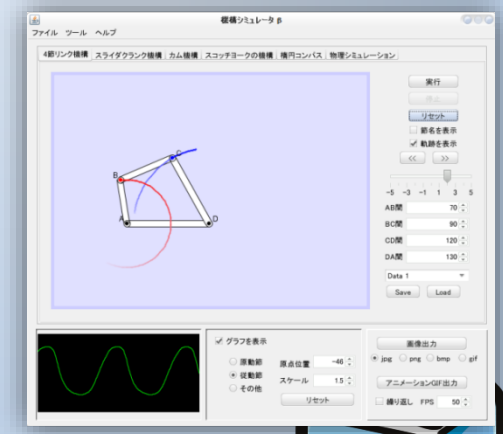








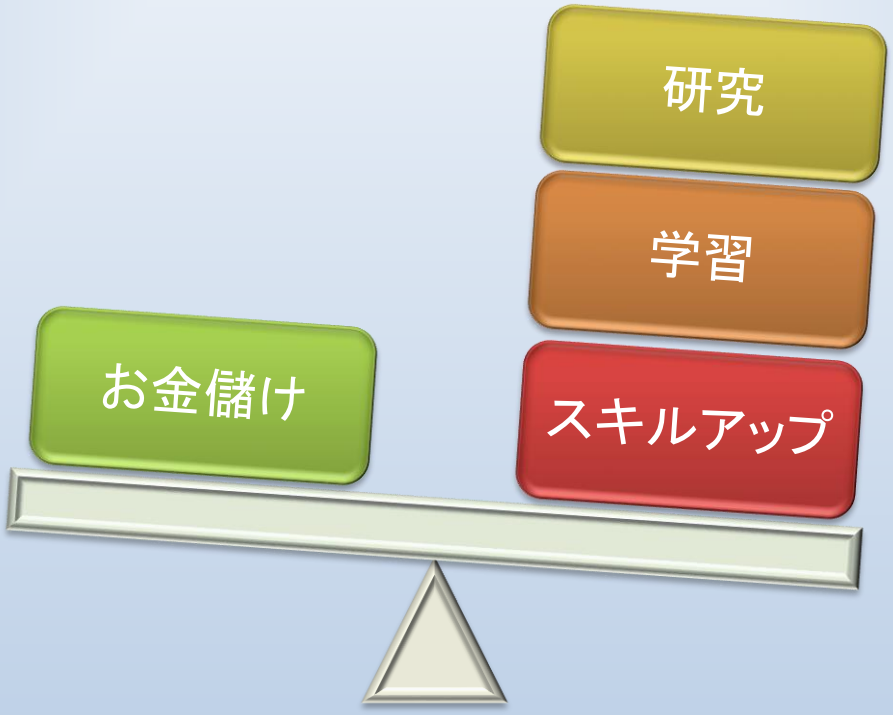




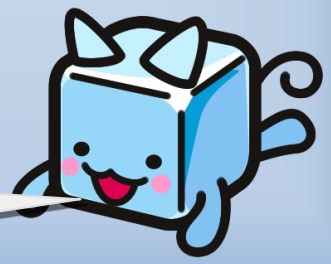
色んなソフト作ってます

活動方針

ソフトウェア開発について学ぶことで、
研究や就職で役に立つ能力の養成が目的。



あくまでお金儲けは二の次



募集中

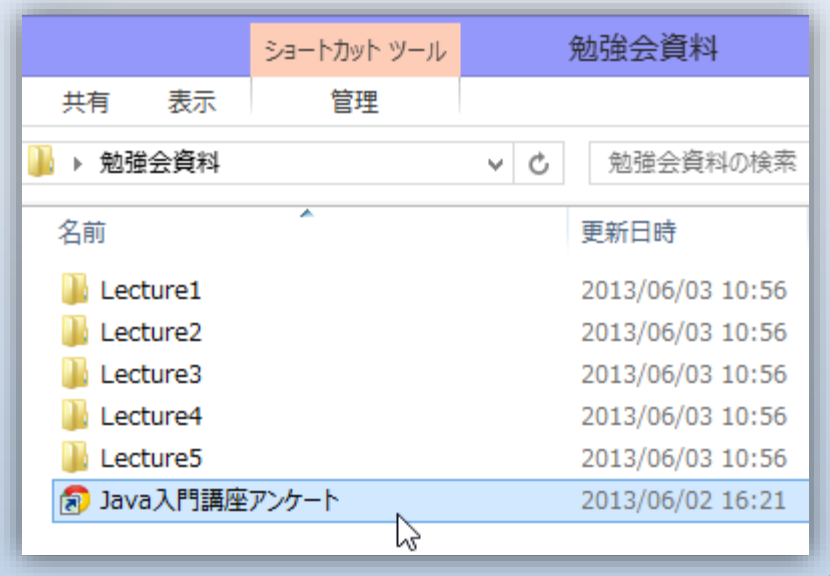
- 研究室で作って欲しいソフトがある人
- スキルアップしたい人
- ソフト作ってお金儲けしたい人

toru@arkoak.com

お気軽にこちら(ArkOak代表:加納)まで

お疲れ様でした。

アンケートにご協力下さい。



おわり

製作：ArkOak代表
信州大学大学院　加納徹