

ネットワークプログラミング  
ネットワークプログラミング演習  
21001教室  
第5回  
2013/10/14

岩井将行

# 授業資料

- <http://www.cps.im.dendai.ac.jp>
- <http://www.cps.im.dendai.ac.jp/index.php?Classes%2F2013netpro>

# 講師紹介

- <http://cps.im.dendai.ac.jp/index.php?Members%2Fiwai>
- 岩井研究室
- <http://cps.im.dendai.ac.jp>
- 岩井研究室の研究分野
- <http://cps.im.dendai.ac.jp/index.php?Research%2FTopics>
- 連絡先 1号館11F 11107b
- iwaiあつと im.dendai

# TA・SA・副手

- 加藤 佳祐
- 鉄谷研究室
- 東京電機大学 未来科学部 情報メディア学科
- E-mail: Keisuke Kato <case.unl@gmail.com>

# 2限副手紹介

- 鉄谷研究室M2
  - － 多田 真之 tada先輩 2限
  - － 新井 駿 arashun先輩 2限

# 成績

- 出席
- 毎回課題
- 中間試験
- 最終試験 + 最終課題
  
- ★演習は演習最終発表会を加味

# 講義内容

|      |                              |
|------|------------------------------|
| 第1回  | Java理解度チェック                  |
| 第2回  | TCP/IPの復習                    |
| 第3回  | TCPサーバ                       |
| 第4回  | TCPクライアント/サーバ通信 チャットプログラム    |
| 第5回  | UDP通信                        |
| 第6回  | <b>中間試験（持ち込み不可 紙提出）</b>      |
| 第7回  | スレッド基礎 サーバのスレッド化 マルチスレッド     |
| 第8回  | デザインパターン<br>ファクトリメソッド シングルトン |
| 第9回  | ノンブロッキングI/O 最終課題に向けた目標設定     |
| 第10回 | マルチスレッド スレッドプール Twitter4J    |
| 第11回 | Twitter4J, Webクライアント         |
| 第12回 | WEBサーバ,プロジェクト設計              |
| 第13回 | プロジェクト実装                     |
| 第14回 | 予備                           |
| 第15回 | <b>学力考査（持ち込み可 プログラミング提出）</b> |

2013/10/14

# 授業予定日日程

- [http://www.soe.dendai.ac.jp/kyomu/portal/2013\\_schedule\\_t.pdf](http://www.soe.dendai.ac.jp/kyomu/portal/2013_schedule_t.pdf)
- (2)9/16 敬老の日【授業実施日】
- (3)9/23 秋分の日【授業実施日】
- (4)9/30
- (5)10/7
- **(6)10/14 体育の日【授業実施日】 中間試験**
- (7)10/21
- (8)10/28
- ~~休み 11/4 文化の日~~
- (9)11/11
- (10)11/18
- (11)11/25
- (12)12/2
- (13)12/9
- (14)12/16
- ~~休み 岩井出張休講 12/23~~
- ~~休み 1/13 成人の日~~
- **(15)1/20 学力考査**

2013/10/4 ※授業予定日に休講が有る場合は連絡します。



# 概要

- クライアント／サーバモデル、TCP/IPネットワークのアプリケーションプログラミングインタフェースの基本および、ネットワークアプリケーションを効率的に動作させるためのマルチスレッドプログラミングを講義する。この基本の後、チャット等の対話型アプリケーション、Twitter4J等のアプリケーション開発の実例を講義する。

# ゴール

- 通信ネットワークを利用したアプリケーションソフトウェアを、TCP/IP を意識したレベルで作成できる力を養成することを目標とする。

# 参考書

- 購入の必要はありません。
  - TCP/IPソケットプログラミング JAVA編
  - ISBN4-274-06520-0
  - オーム社
  - •[改訂第2版]JAVA言語プログラミングレッスン上
  - ISBN4-7973-3211-5
  - SoftBankCreative
  - •[改訂第2版]JAVA言語プログラミングレッスン下
  - ISBN4-7973-3212-3
  - SoftBankCreative

# 持ち物

- Laptop pc
  - Eclipse環境が整っていること
- PDF アクロバトリーダ
- LANでネットワークに接続できること。
- Macでもよい

# 日本語版eclipse

- <http://mergedoc.sourceforge.jp/>

The screenshot shows the MergeDoc Project website. The main content area is titled "Pleiades All in One 日本語ディストリビューション". Below the title, it specifies the version "Pleiades All in One 4.3.0.v20130626" and the base "Eclipse 4.3.0 Kepler for Windows ベース". There are three bullet points in Japanese describing the package contents and how to use it. Below the text is a table with columns for "Platform", "Ultimate", and "Java". The table has rows for "32bit" and "64bit", and sub-rows for "Full Edition" and "Standard Edition". Each cell in the table contains a "Download" button. The "Download" buttons for the "Java" column are circled in red. Below the table, there are links for "Eclipse 実行用 JRE 7", "開発対象用 JDK 6u45、7u25", "MinGW 32bit、64bit", "Tomcat 6.0.37、7.0.41", and "Python 2.7.5、3.3.2".

MergeDoc Project

Pleiades プラグイン日本語化プラグイン

JStyle 改行タブ表示プラグイン

MergeDoc / Javadoc 日本語化

フォーラム

チケット

プロジェクト Wiki

ブログ

Pleiades All in One 日本語ディストリビューション

**Pleiades All in One 4.3.0.v20130626**  
Eclipse 4.3.0 Kepler for Windows ベース

- 開発対象となる言語に合わせてパッケージをダウンロードしてください
- Full Edition には Eclipse 実行用の JRE や各言語の処理系が含まれていよく分からない場合は Full Edition を選んでください。

plugins、features ディレクトリーに格納されたプラグイン  
dropins ディレクトリーに格納されたプラグイン  
Eclipse 実行用の JRE や各言語のコンパイラー、ランタイムなどの処

|       |                  | Platform                 | Ultimate                 | Java                     |
|-------|------------------|--------------------------|--------------------------|--------------------------|
| 32bit | Full Edition     | <a href="#">Download</a> | <a href="#">Download</a> | <a href="#">Download</a> |
|       | Standard Edition | <a href="#">Download</a> | <a href="#">Download</a> | <a href="#">Download</a> |
| 64bit | Full Edition     | <a href="#">Download</a> | <a href="#">Download</a> | <a href="#">Download</a> |
|       | Standard Edition | <a href="#">Download</a> | <a href="#">Download</a> | <a href="#">Download</a> |

Eclipse 実行用 JRE 7

開発対象用 JDK 6u45、7u25

MinGW 32bit、64bit

Tomcat 6.0.37、7.0.41

Python 2.7.5、3.3.2

# ソケット通信

# プログラム同士の通信は

- ソケットを使ってデータの送受信
- ソケットを使った通信

ソケット通信

# ソケット

意味：「接続の端点」

コンピュータとTCP/IPを  
つなぐ出入り口

ソケット





# ソケット通信

- ソケットを使って通信を行うには  
2つのプログラムが必要

## クライアントプログラム

ソケットを用意して  
サーバに接続要求を行う

## サーバプログラム

ソケットを用意して接続要求を待つ

# ソケット通信の全体の流れ

クライアント

ソケット生成(socket)

サーバを探す  
(gethostbyname)

接続要求(connect)

データ送受信(send/rcv)

ソケットを閉じる(close)

サーバ

ソケット生成(socket)

接続の準備(bind)

接続待機(listen)

接続受信(accept)

データ送受信(send/rcv)

ソケットを閉じる(close)

識別情報

# InetAddress

- <http://docs.oracle.com/javase/jp/6/api/java/net/InetAddress.html>
- IPアドレスを扱うクラス
- java.net  
クラス InetAddress
- [java.lang.Object](#) java.net.InetAddress すべての  
の実装されたインタフェース:[Serializable](#)直系  
の既知のサブクラ  
ス:[Inet4Address](#), [Inet6Address](#)

# InetAddress

- [StringgetHostName\(\)](#)  
この IP アドレスに対応するホスト名を取得します。
- static [InetAddressgetByAddress\(String host, byte\[\] addr\)](#)  
指定されたホスト名および IP アドレスに基づいて InetAddress を作成します。
- static [InetAddressgetLocalHost\(\)](#)  
ローカルホストを返します。

# InetAddress

- static [InetAddress](#)[getByName](#)([String](#) host)  
指定されたホスト名を持つホストの IP アドレスを取得します。
- boolean [isReachable](#)(int timeout)  
そのアドレスに到達可能かどうかをテストします
- [String](#)[toString](#)()  
この IP アドレスを String に変換します。

# 動かしてみよう。

- ChatClientLoop.java
- ChatServerLoop.java

# メソッドの引数

戻り値   メソッド名(型 変数名1,  
                  型 変数名2,  
                  型 変数名3,  
                  型 変数名4  
                  .....) {

メソッドの本体

}





# メソッド呼び出し

本来は、

```
g.drawString(XXXXXXXXXXXXXX);
```

のように、

```
オブジェクト.メソッド名(引数...);
```

と書く。

## メソッド呼び出し(2)

しかし、自分で定義したクラスの中のメソッドを呼び出すときは、  
オブジェクト。

なしに、  
メソッド名(引数...);  
でよい。

例:

```
drawBar(XXXXXXXXXX);
```

# methodとクラス

- Heikin.java と Kamoku.java
- Heikin と Kamoku クラスを作る
  - public class Heikin
  - class Kamoku
- Heikin クラス
  - Kamokuクラスのインスタンスとして、englishとmathを作る
  - english の name に "英語" を設定する
  - english の score に 80 を設定する
  - math も english と同様に (name→数学, score→70)
  - 英語と数学のscoreを読み出して、平均値を表示する
- Kamoku クラス
  - String name
  - setScore というメソッドを定義する。score に値を設定する。
  - getScore というメソッドを定義する。scoreを返す。

# 定数の宣言

C++/C では、#define 文を使用した。

(例)

```
#define WIDTH 80
```

Javaでは、final static で修飾する。

(例)

```
public final static int WIDTH = 80;  
public final static String school = "dendai";
```

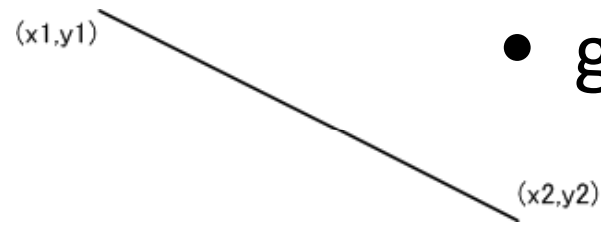
# Graphics

Graphics というクラスには、  
drawString, drawCircle 等の  
メソッドが定義されている。

Graphics クラスである g という  
オブジェクトに対して、  
g.drawString、  
g.drawCircle  
という形でメソッドを呼び出せる。

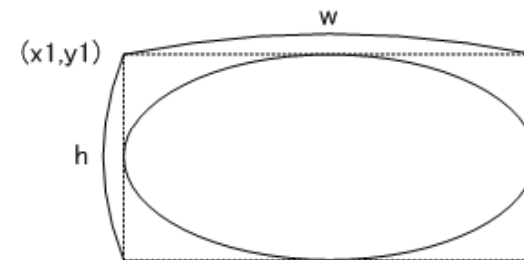
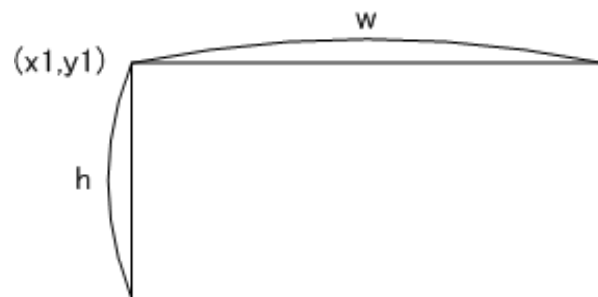
# Graphics $\mathcal{O}$ method

- `g.drawLine(x1,y1,x2,y2);`



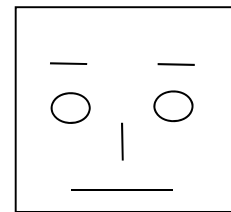
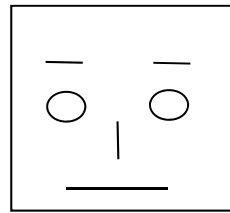
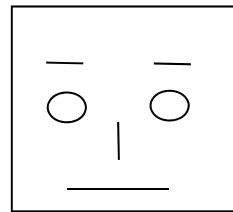
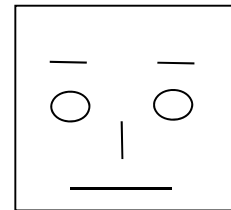
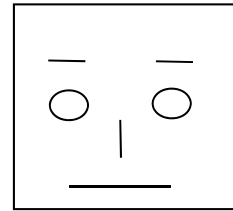
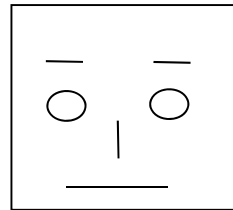
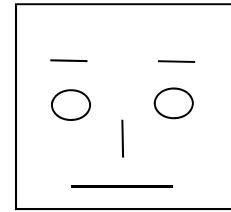
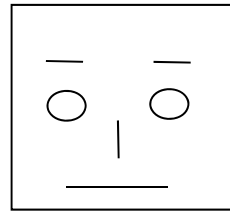
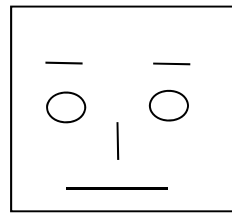
- `g.drawOval(x,y,w,h);`

- `g.drawRect(x,y,w,h);`



# 前回課題faceを沢山つくってみよう

## Face.javaを改造してFaces.java



# Objectの配列



# Objectの配列 : FaceObjKadaiAns.java

- `FaceObjAns[] fobjs = new FaceObjAns[9];`

```
for (int j = 0; j < 3; j++) {//行
    yStart = j * 220 + 50;
    for (int i = 0; i < 3; i++) {//列
        xStart = i * 220 + 40;
        fobjs[i + 3 * j] = new FaceObjAns(xStart, yStart);
    }
}
```

# 描画

```
// 9個数の顔を書く  
for (int i = 0; i < fobjs.length; i++) {  
    fobjs[i].makeFace(g);  
}
```

# FaceObjectのコンストラクタ

```
class FaceObjAns {  
    // コンストラクタ  
    int h;  
    int w;  
    int xStart = 0;  
    int yStart = 0;  
    public FaceObjAns(int x, int y) {  
        h = 200;  
        w = 200;  
        this.xStart = x;  
        this.yStart = y;  
    }  
    // 個々にメソッドを追加  
    public void makeFace(Graphics g) {  
        makeRim(g);  
        makeEyes(g, 20);  
        makeNose(g, 40);  
        makeMouth(g, 80);  
    }  
}
```

# Thread

**Thread,Runnable**  
**MovingBall**

# Threadの2種類の作りかた

- **1) implements Runnable**

Runnableを実装したクラスをThreadクラスのコンストラクタとして渡す。start()で開始。

```
CountTenRunnable ct = new CountTenRunnable();  
Thread th = new Thread(ct);  
th.start();
```

- **2) extends Thread**

Threadを拡張したクラスをnewしてstart()メソッドを呼び出す。

- `CountTest.java`
- `CountTenRunnable.java`
- `CountTesterTweThreads.java`

# MovingBall

```
MovingInnerFFrame f = new MovingInnerFFrame();  
Thread th = new Thread(f);  
th.start();
```

```
class MovingInnerFFrame extends Frame implements  
    Runnable {  
  
    public void run() {}  
}
```

# 第5回数演習課題

- MovingBallを改造し3色以上の複数のボールを表示せよ。
- 壁の跳ね返りを画面のサイズに修正せよ。
- コンソールに残り秒数カウントダウンしてゲームが終了するようにThread.sleep()メソッドを用いよ。
- なお終了時間は20秒とせよ。
- ヒント
  - Thread.sleep(20\*1000);