明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

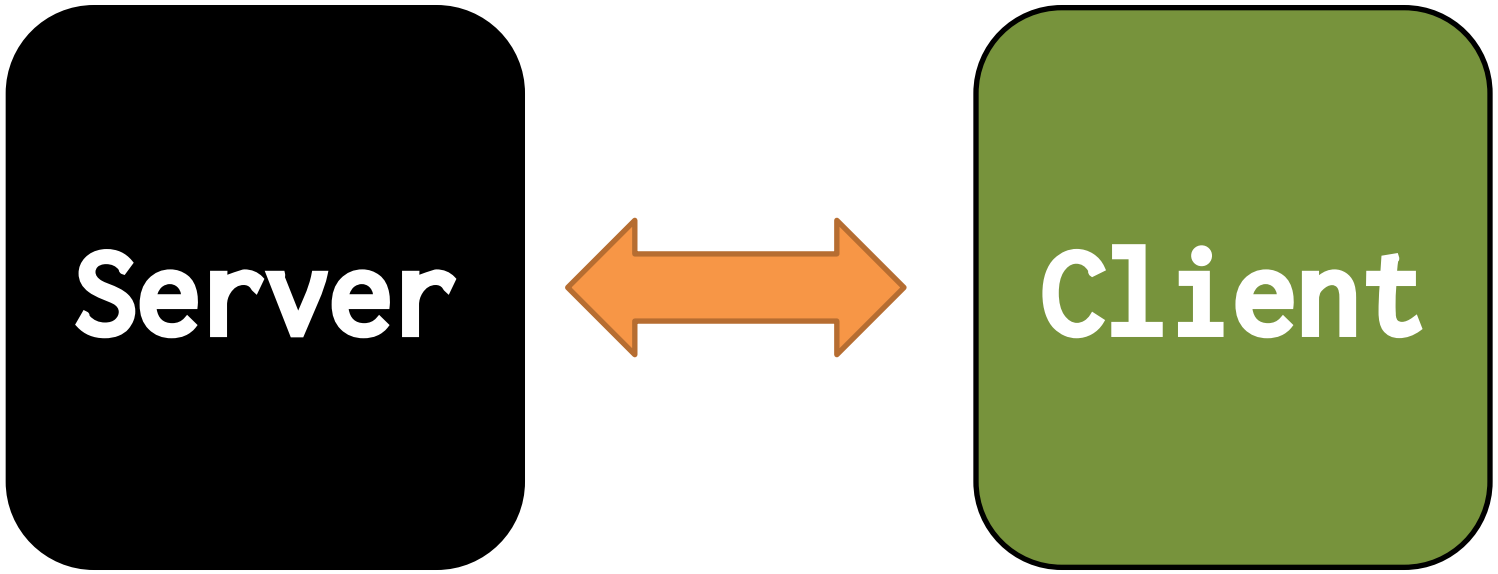
プログラミング演習2
ネットワーク通信

中村、小林、橋本、辻野

本日の内容

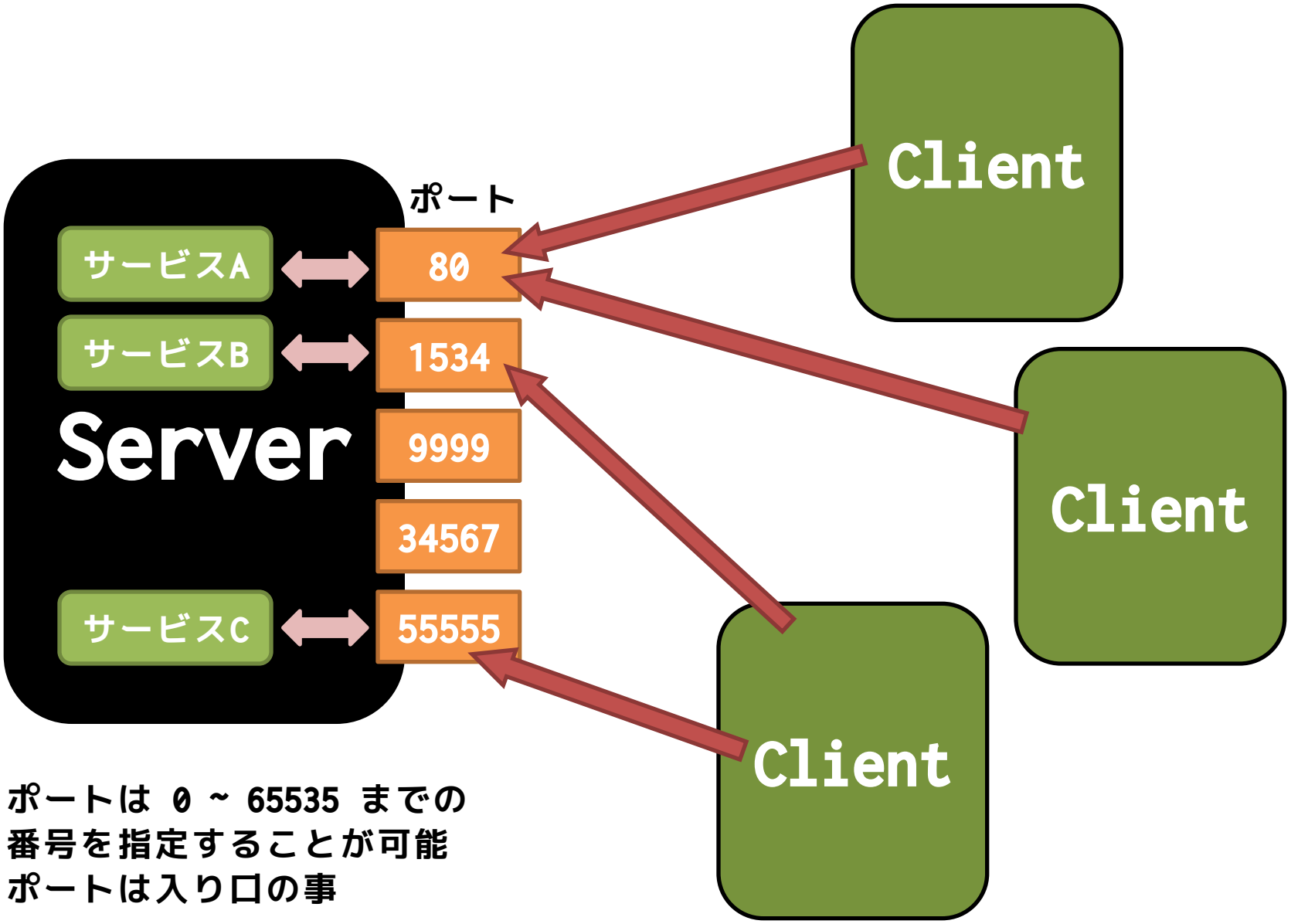
明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- ネットワーク通信で情報のやりとりをしよう



通信の基本

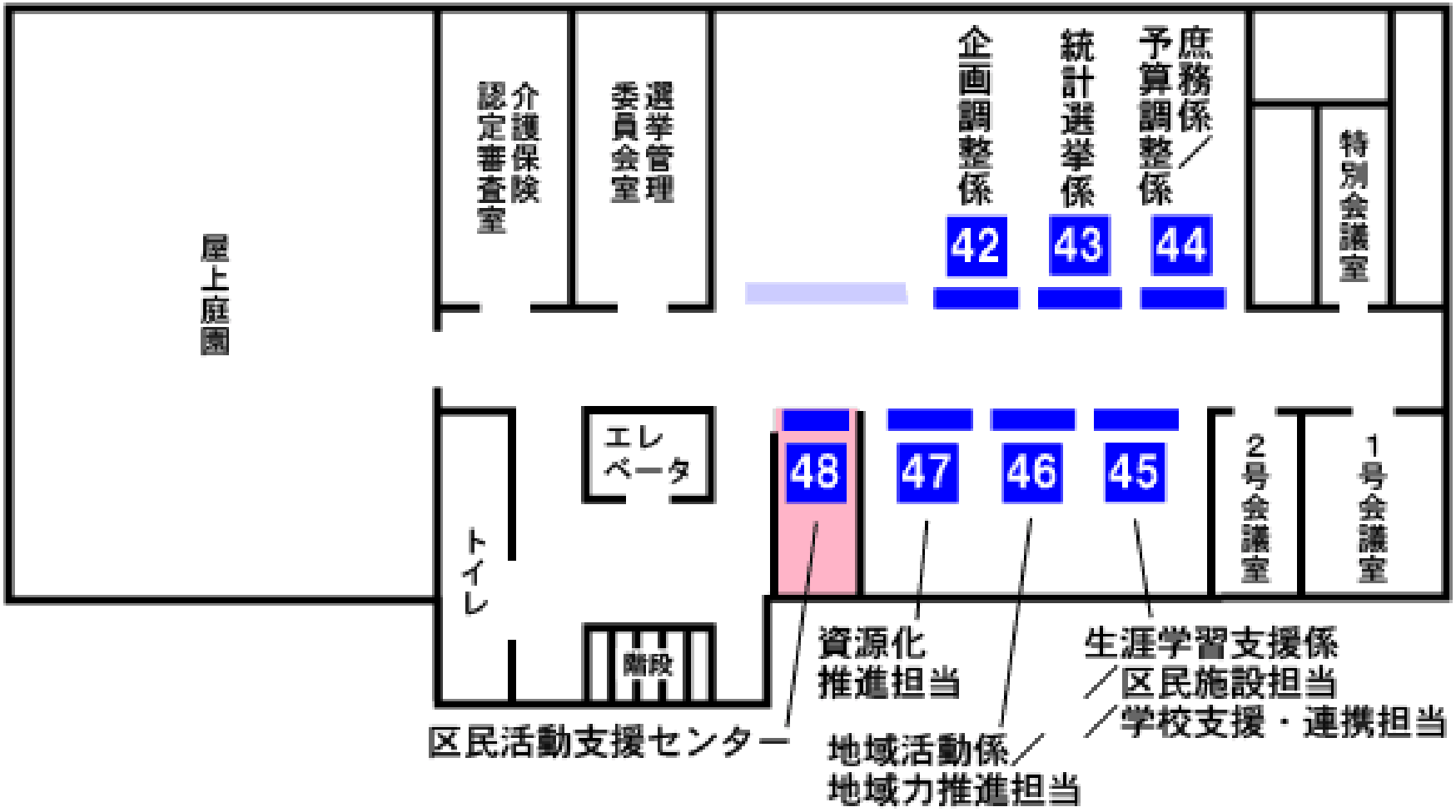
明治大学総合数理学部
先端メディアサイエンス学科
中村研究室



ポートは 0 ~ 65535 までの
番号を指定することが可能
ポートは入り口の事

窓口とサービスの対応関係

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室



http://www.city.yokohama.lg.jp/kohoku/kyoutuu/mado/mado-48.html

市役所で考えてみる

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

1. 役所の係の人が窓口を開けて準備する
2. 市民が役所（の住所）に行き、窓口に並ぶ
3. 窓口の人が最初に並んでいる市民を呼ぶ
4. 窓口で、役所の人と、市民が対話して何らかの処理を行う

サーバ・クライアント

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

1. サーバがポートを開けて準備する
2. クライアントがサーバのアドレスとポート番号を指定してサーバに接続要求を行う
3. サーバがクライアントに対して接続許可を出すことで、情報交換のための経路が確立される
4. サーバとクライアント間でやりとりが行われる

注意点

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 下記をプログラムに入れること

```
void stop(){
  myClient.stop();
}
```

```
void stop(){
  myServer.stop();
}
```

- draw() が無いと動作しません
- 問題が解決しない場合は再起動すること
  - ファイアウォールに関する問題が出てきたら、再起動しないとダメな模様

とりあえず書いてみよう

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

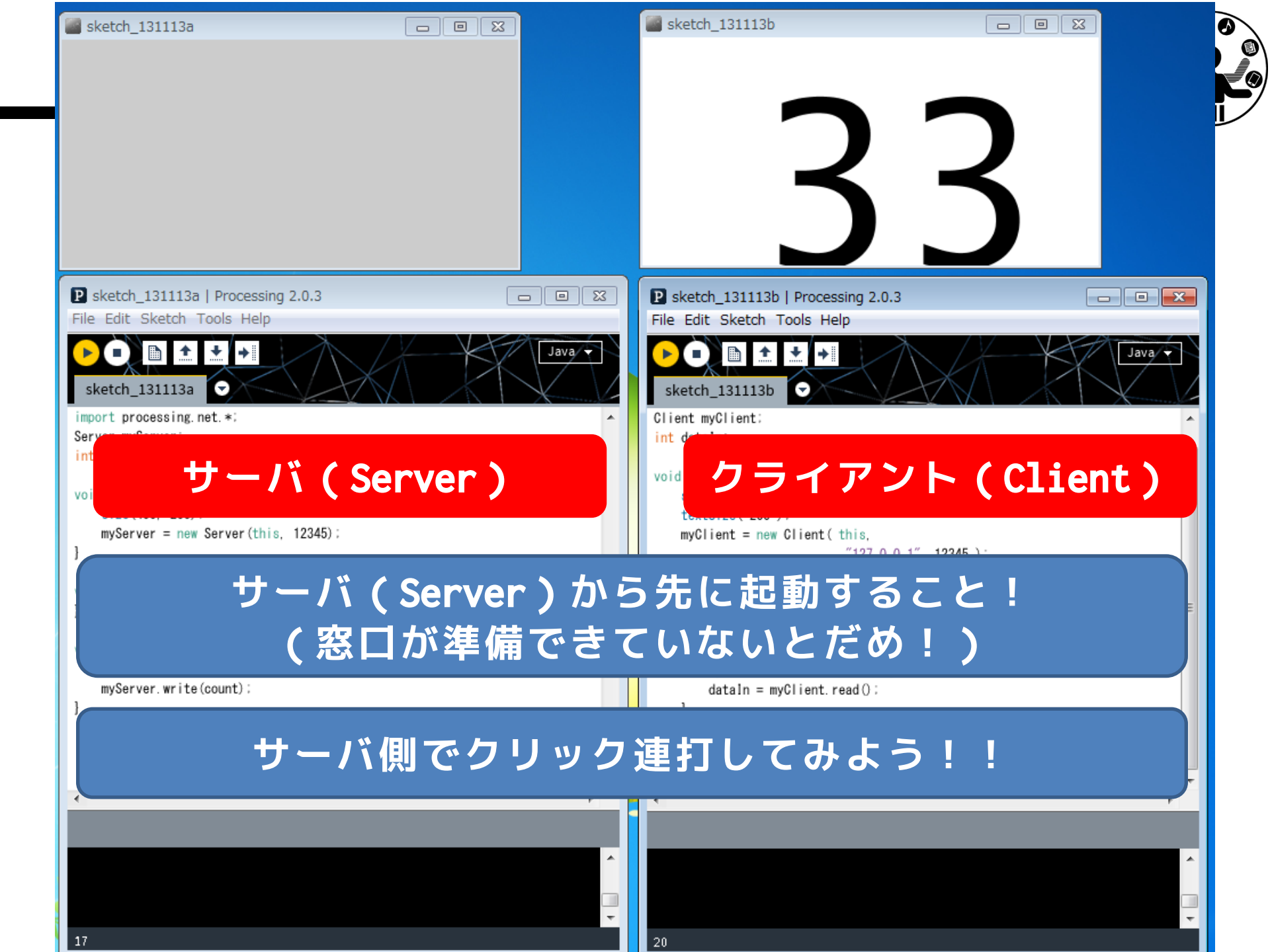
サーバ（Server1.pde）

```
import processing.net.*;
Server myServer
         = new Server(this, 12345);
int s_count = 0;

void setup() {
  size(400, 200);
}

void draw() {
}

void stop(){
  myServer.stop();
}

void mousePressed() {
  s_count++;
  myServer.write(s_count);
}
```

クライアント（Client1.pde）

```
import processing.net.*;
Client myClient = new Client(this,
                "127.0.0.1", 12345);
int c_count = 0;
void setup() {
  size(400, 200);
  textSize(200);
  fill(0);
}
void stop(){
  myClient.stop();
}

void draw() {
  background(255);
  if (myClient.available() > 0) {
    c_count = myClient.read();
  }
  text(c_count, 100, 200);
}
```

127.0.0.1 は自分という意味


sketch_131113a
sketch_131113b
33
sketch_131113a | Processing 2.0.3
File Edit Sketch Tools Help
Java
sketch_131113a
import processing.net.*;
myServer = new Server(this, 12345);
myServer.write(count);
17
sketch_131113b | Processing 2.0.3
File Edit Sketch Tools Help
Java
sketch_131113b
Client myClient;
myClient = new Client(this,
dataIn = myClient.read();
20
サーバ（Server）
クライアント（Client）
サーバ（Server）から先に起動すること！
（窓口が準備できていないとだめ！）
サーバ側でクリック連打してみよう！！

とはいえ

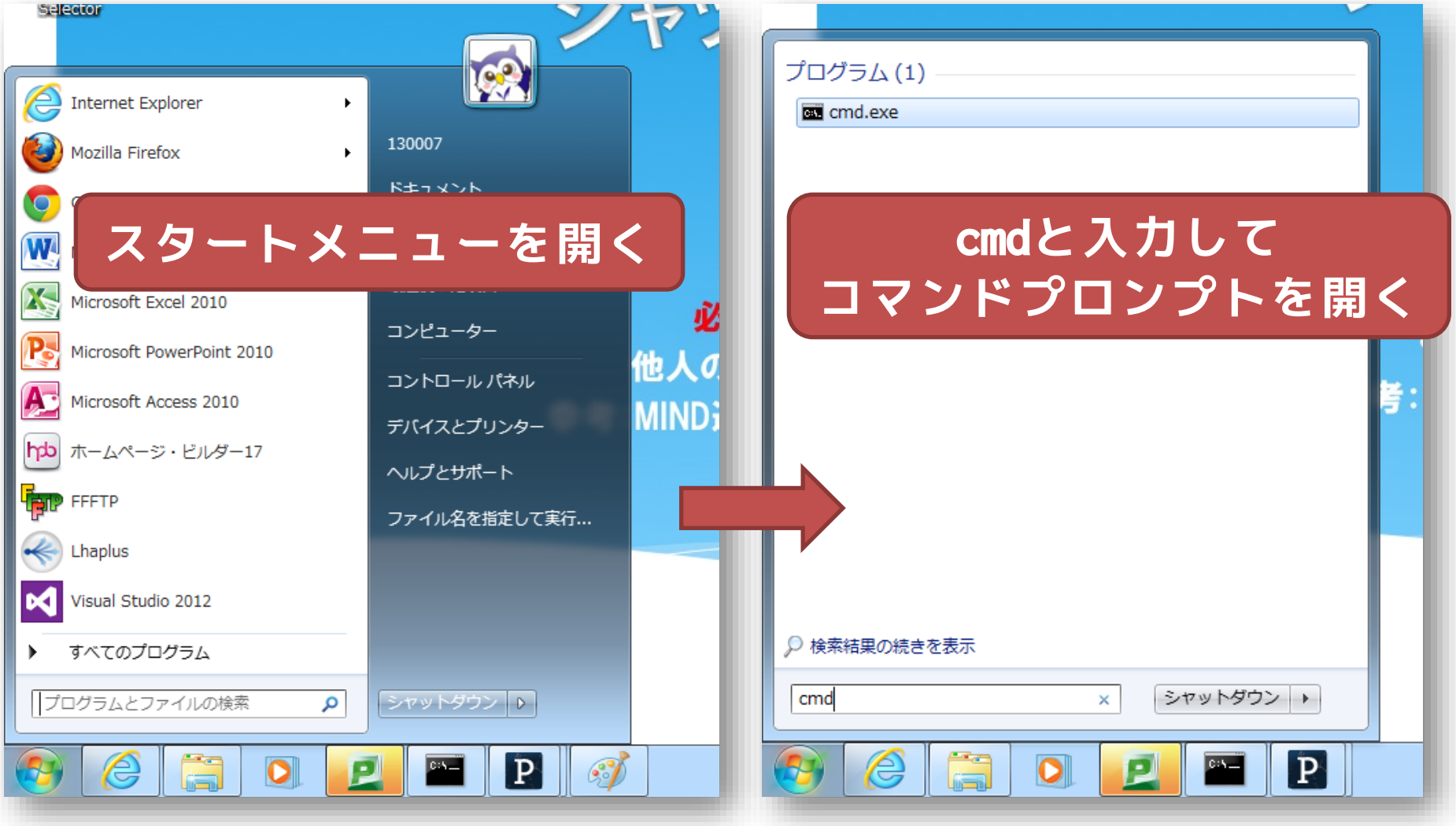
明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- なんだか、自分の中でServerとかClientとか言ってもよくわからない。

- 隣の人とペアを作って接続してみよう！
  - 隣の人がいない場合は、前後の余っている人同士でやってみる！
  - 一方はサーバ役、他方はクライアント役
    - 交互にやりましょう
  - まずは、隣の人のアドレス（住所）を聞く！
    - 実際の住所ではなく、ネットワーク上の住所

隣に接続してみよう

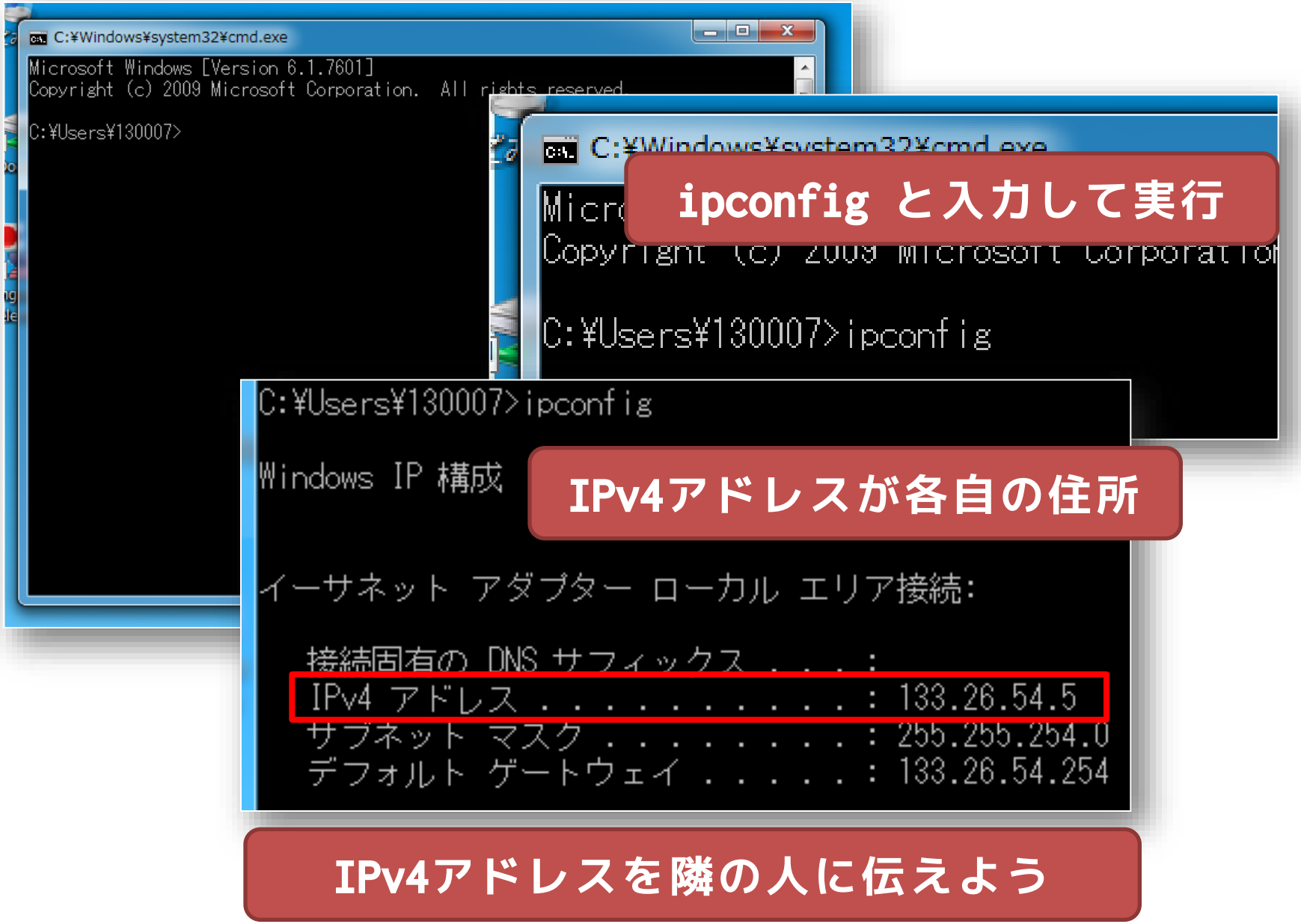
明治大学総合数理学部
先端メディアサイエンス学科
中村研究室



- ターミナルで ifconfig

隣に接続してみよう

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室



通信できるかチェック！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- コマンドプロンプトやTerminalでpingというコマンドを入力して確認しよう！

```
ping IPアドレス
```

- 通信できるかどうかをチェックする先のIPアドレスが 133.26.54.5 の場合は下記

```
ping 133.26.54.5
```

- 自分に対しては

```
ping 127.0.0.1
```

通信できた？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

成功例（損失0%）： 通信できた！

```
C:\Users\nakamura>ping 127.0.0.1

127.0.0.1 に ping を送信しています 32 バイトのデータ:
127.0.0.1 からの応答: バイト数 =32 時間 <1ms TTL=128
127.0.0.1 からの応答: バイト数 =32 時間 <1ms TTL=128
127.0.0.1 からの応答: バイト数 =32 時間 <1ms TTL=128
127.0.0.1 からの応答: バイト数 =32 時間 <1ms TTL=128

127.0.0.1 の ping 統計:
    パケット数: 送信 = 4、受信 = 4、損失 = 0 (0% の損失)、
ラウンド トリップの概算時間 (ミリ秒):
    最小 = 0ms、最大 = 0ms、平均 = 0ms
```

失敗例（損失100%）： 通信できなかった！

```
C:\Users\nakamura>ping 133.26.54.5

133.26.54.5 に ping を送信しています 32 バイトのデータ:
要求がタイムアウトしました。
要求がタイムアウトしました。
要求がタイムアウトしました。
要求がタイムアウトしました。

133.26.54.5 の ping 統計:
    パケット数: 送信 = 4、受信 = 0、損失 = 4 (100% の損失)、
```

通信できたら書いてみよう

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
import processing.net.*;
Client myClient = new Client(this, "133.26.54.5", 12345);
int c_count = 0;

void setup() {
  size(400, 200);
  textSize(200);
  fill(0);
}

void stop(){
  myClient.stop();
}

void draw() {
  background(255);
  if (myClient.available() > 0) {
    c_count = myClient.read();
  }
  text(c_count, 100, 200);
}
```

Server役の人の
IPアドレスが133.26.54.5の場合

クライアント（Client）

準備と相互接続

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- サーバ・クライアント用のライブラリの準備

```
import processing.net.*;
```

- サーバの準備＋接続待ち

```
Server myServer = new Server(this,
                             接続待ちポート番号);
```

- クライアントの準備＋接続

```
Client myClient = new Client(this,
    "接続先アドレス", 接続先ポート番号);
```

メッセージの送信

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- サーバからクライアントへのメッセージの送信（整数または文字列を送信）

```
myServer.write(100);
myServer.write("hogehoge");
```

- クライアントからサーバへのメッセージの送信（整数または文字列を送信）

```
myClient.write(100);
myClient.write("hogehoge");
```

クライアントでのデータ受信

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- クライアントでのデータ受信（整数）

```
int recvInt = myClient.read();
```

- クライアントでのデータ受信（文字列）

```
String recvStr = myClient.readString();
```

- クライアントでのデータ受信量取得

```
int size = myClient.available();
```

サーバでのデータ受信

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- データ受信待ちしているクライアントの取得

```
Client nextClient = myServer.available();
```

- あるクライアントからのデータ受信（整数）

```
int recvInt = nextClient.read();
```

- あるクライアントからのデータ受信（文字列）

```
String recvStr = nextClient.readString();
```

通信終了

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- サーバからの切断

```
myClient.stop();
```

- 特定のクライアントの切断

```
myServer.disconnect(thisClient);
```

- サーバのサービスの終了

```
myServer.stop();
```

解説

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

133.26.54.5 さん

```
import processing.net.*;
Server myServer
      = new Server(this, 12345);
int s_count = 0;

void setup() {
  size(400, 200);
}

void draw() {
}

void stop(){
  myServer.stop();
}

void mousePressed() {
  s_count++;
  myServer.write(s_count);
}
```

12345番ポートで準備

クリック時にs_countを増やし
クライアントにs_countを送信！

133.26.54.6 さん

```
import processing.net.*;
Client myClient = new Client(this,
              "133.26.54.5", 12345);
int c_count = 0;

void setup() {
  size(400, 200);
  textSize(200);
  fill(0);
}

void stop(){
  myClient.stop();
}

void draw() {
  background(255);
  if (myClient.available() > 0) {
    c_count = myClient.read();
  }
  text(c_count, 100, 200);
}
```

133.26.54.5さんの
12345番ポートに接続

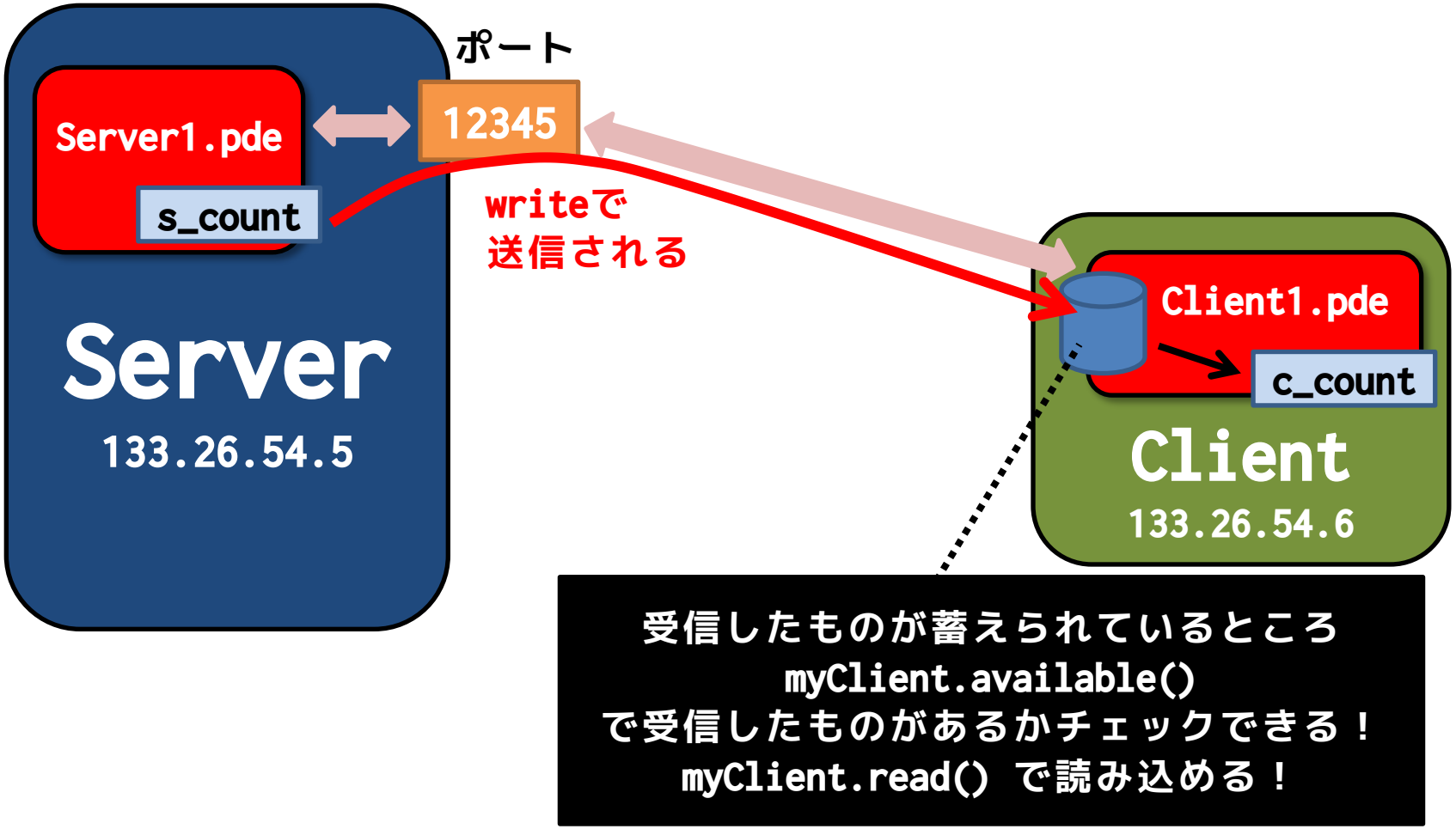
avialable() で受信
しているかチェック
0より大きければ受信

read() で値を受信して表示！

通信の基本

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

ポート12345を介して通信し、情報をやりとり



やってみよう！

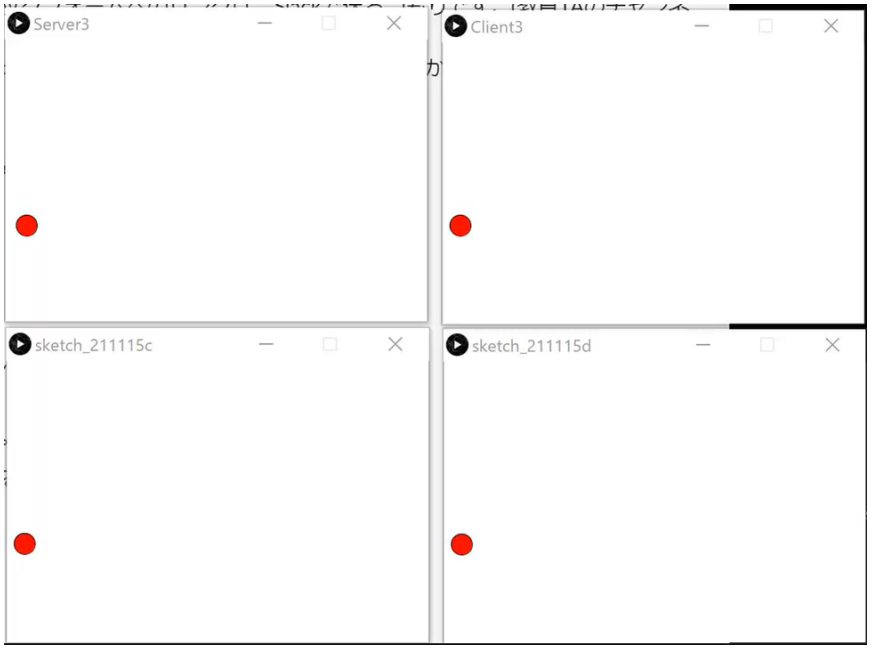
明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 相互に接続できた人は、1人をサーバにして他の人はみんなクライアントとして接続して、サーバのクリック状況がみんなのパソコンで見えるかどうかを確認してみよう！
- 相互接続できない場合は、127.0.0.1でクライアントを複数起動してみて、どうなるか試してみよう！

課題1：basic_SCSync

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

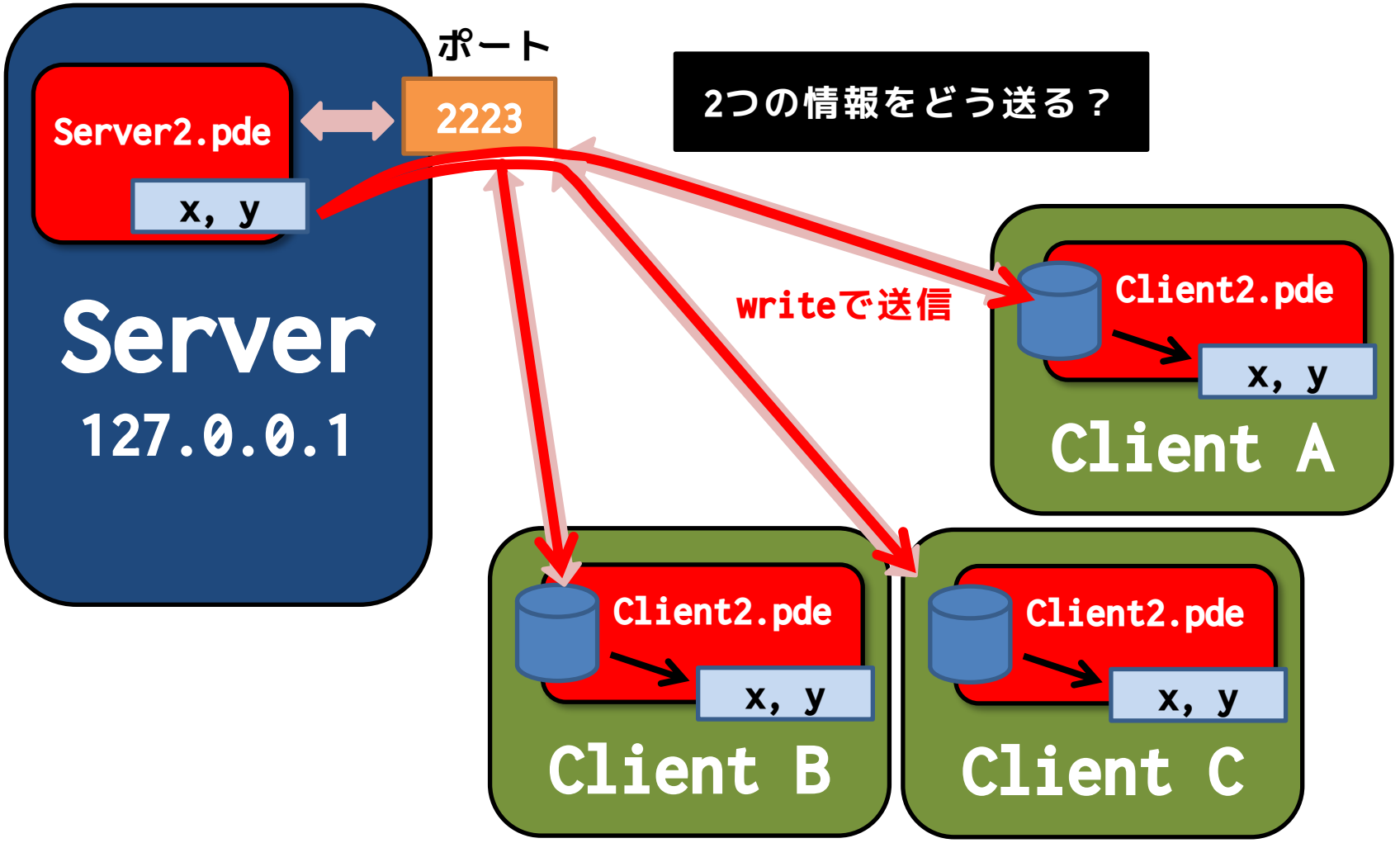
- サーバとクライアントで描画を同期させてみよう！（サーバで動いている円をクライアントに表示してみる！）
  - サーバ上の円の座標(x，y)をクライアントに送って、クライアント上で(x，y)に円を描画する



通信の基本

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

ポート2223を介して通信し、情報をやりとり



サーバの描画を転送！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

サーバ（Server2.pde）

```
import processing.net.*;
Server myServer = new Server(this, 2223);

int x = 0, y = 0;
int vx = 5, vy = 4;

void setup() {
  size(600, 400);
  fill(255, 0, 0);
}

void stop() {
  myServer.stop();
}

void draw() {
  background(255);
  x += vx;
  y += vy;
  x %= width;
  y %= height;
  background(255);
  circle(x, y, 30);
  myServer.write(                    );
}
```

クライアント（Client2.pde）

```
import processing.net.*;
Client myClient = new Client(this, "127.0.0.1", 2223);

void setup(){
  size(600, 400);
  fill(255, 0, 0);
  background(255);
}

void stop(){
  myClient.stop();
}

void draw(){
  background(255);

  if (myClient.available() > 0) {
    // 文字列として受信
    String msg = myClient.readString();

    // 描画してみよう！

  }
}
```

送受信の時に・・・

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- どのようなフォーマット（形式）でデータを送受信するかを考える
- クリック数は1つの数字だけだが、ある座標を送受信しようと思うと、2つの値が必要となるため困ったことに

```
myServer.write(x);
myServer.write(y);
```

```
myServer.write(x + "," + y);
```

送受信の時に...

ファイル入出力でもやったよね！

1. カンマ区切りで文字列として送信（他でもOK）
   - "x座標,y座標"　（例） 100,230
   - myServer.**write(x + "," + y);**
2. 文字列として受信
   - String recvStr = myClient.**readString();**
3. 受信した文字列（カンマ区切り）を、splitを利用してカンマで分割
   （"100,230" ➔ "100" と "230" に）
   - String[] data = **split(recvStr, ',');**
4. 分割された文字列を整数に戻す
   - int x = **int(data[0]);**
   - int y = **int(data[1]);**

クライアントに描画する

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- サーバで操作することで、クライアント上に描画してみる
  - サーバとクライアントで相互接続
  - サーバ上でクリックされた位置をクライアントに送信
  - クライアントで送信されてきた座標に円を描画

サーバから描画してみる

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

サーバ（Server3.pde）

```
// ライブラリを読み込む
import processing.net.*;
// サーバを2222ポートでスタート
Server myServer = new Server(this, 2222);

void setup() {
  size(600, 300);
  fill(0, 0, 255);
}

void mousePressed() {
  // クリック座標をカンマ区切りで送信
  myServer.write(mouseX + "," + mouseY);
}

void draw(){
}

void stop(){
  myServer.stop();
}
```

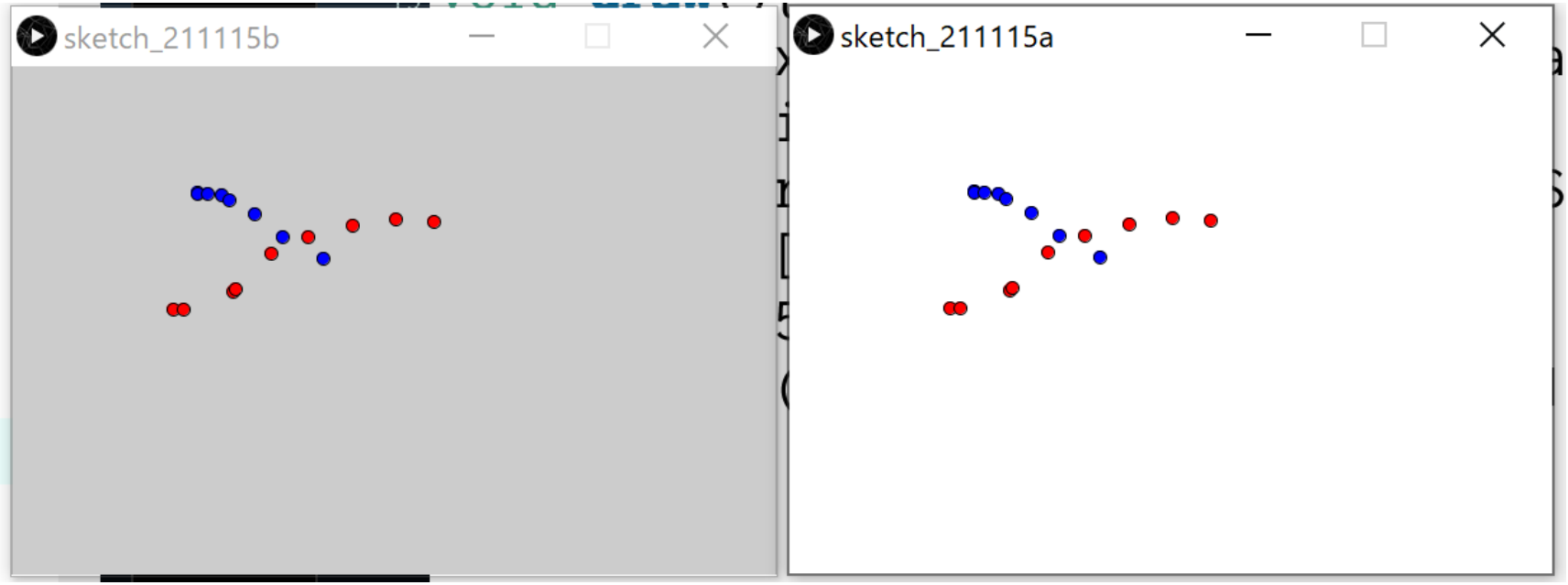
クライアント（Client3.pde）

```
// ライブラリを読み込む
import processing.net.*;
// X.X.X.X の 2222 番に接続
Client myClient = new Client(this, "X.X.X.X", 2222);

void setup() {
  size(600, 300);
  fill(255, 0, 0);
  background(255);
}

void stop(){
  myClient.stop();
}

void draw() {
  // 何か受信しているか確認
  if (myClient.available() > 0) {
    // 文字列として受信
    String msg = myClient.readString();
    // 文字列をカンマ区切りで分割し整数に戻す
    String[] data = split(msg, ',');
    circle(int(data[0]), int(data[1]), 10);
  }
}
```

クライアントに描画してみる

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 一方向だとなんだか面白く無い
- 双方向にしてみよう！
  - サーバでクリックするとクライアントに、クライアントでクリックするとサーバに！



双方向で描画してみる

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

サーバ（Server）

```
import processing.net.*;
Server myServer = new Server(this, 2222);

void setup() {
  size(400, 200);
  fill(0, 0, 255);
}

void mousePressed() {
  myServer.write(mouseX + "," + mouseY);
}

void stop(){
  myServer.stop();
}

void draw(){
  // データ送信しているクライアントの確認
  Client nextClient = myServer.available();
  // != null はクライアントがあるという意味
  if(nextClient != null){
     // readString() でデータ受信
     String recvStr = nextClient.readString();
     String[] data = split(recvStr, ',');
     circle(int(data[0]), int(data[1]), 10);
  }
}
```

クライアント（Client）

```
import processing.net.*;
Client myClient = new Client(this, "X.X.X.X", 2222);

void setup() {
  size(400, 200);
  fill(255, 0, 0);
  background(255);
}

void mousePressed(){
  myClient.write(mouseX + "," + mouseY);
}

void stop(){
   myClient.stop();
}

void draw() {
  if (myClient.available() > 0) {
    String recvStr = myClient.readString();
    String[] data = split(recvStr, ',');
    circle(int(data[0]), int(data[1]), 10);
  }
}
```

双方向で描画してみる v.2

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

サーバ（Server）

```
import processing.net.*;
Server myServer = new Server(this, 2222);

void setup(){
  size(400, 200);
  fill(0, 0, 255);
}

void mousePressed(){
  myServer.write(mouseX + "," + mouseY);
  fill(255, 0, 0);
  circle(mouseX, mouseY, 10);
}

void stop(){
  myServer.stop();
}

void draw(){
  Client nextClient = myServer.available();
  if(nextClient != null){
    String recvStr = nextClient.readString();
    String[] data = split(recvStr, ',');
    fill(0, 0, 255);
    circle(int(data[0]), int(data[1]), 10);
  }
}
```

クライアント（Client）

```
import processing.net.*;
Client myClient = new Client(this, "X.X.X.X", 2222);

void setup(){
  size(400, 200);
  fill(255, 0, 0);
  background(255);
}

void mousePressed(){
  myClient.write(mouseX + "," + mouseY);
  fill(0, 0, 255);
  circle(mouseX, mouseY, 10);
}

void stop(){
  myClient.stop();
}

void draw(){
  if(myClient.available() > 0){
    String recvStr = myClient.readString();
    String[] data = split(recvStr, ',');
    fill(255, 0, 0);
    circle(int(data[0]), int(data[1]), 10);
  }
}
```

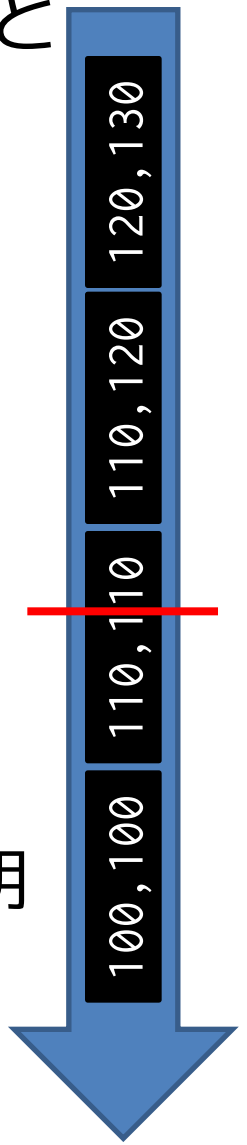
参考：フォーマット

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- データを送りまくるプログラムを作るとそのままではうまくいかなくなる
  - 例えば、下記のように座標情報を送信

```
100,100
110,110
110,120
120,130
```

参考：ぎなた読み
この先生きのこるには
弁慶がなぎなたをもって
スク水揚げ
中国船さんご密漁



  - 受信する場合は、どのようになるかは不明

```
100,100110,1
10110,120120,130
```

参考：フォーマット

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- どこでデータが切れているのかという情報を付与する必要あり

[100,100]	100,100¥n
[110,110]	110,110¥n
[110,120]	110,120¥n
[120,130]	120,130¥n

¥n は改行

- 受信したものを適当に処理する

[100,100][110,1	100,100¥n110,1
10][110,120][120,130]	10¥n110,120¥n120,130¥n

- 受信したものをバッファに追加し、後処理する

送信と受信をしっかりと

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
import processing.net.*;
Server myServer = new Server(this, 2222);
String strBuffer;

void setup() {
  size(400, 200);
}
void stop(){
  myServer.stop();
}
void draw(){
  Client nextClient = myServer.available();
  if(nextClient != null ){
    String recvStr = nextClient.readString();
    strBuffer += recvStr;
  }
  if(strBuffer != null ){
    // 改行までを処理する
    int pos = strBuffer.indexOf("¥n");
    while(pos != -1 ){
      // 改行の手前までを取得する
      String strTarget = strBuffer.substring(0, pos);
      String[] m = split(strTarget, ",");
      // m[0] と m[1] で処理する
      // ¥n の1文字後から次のバッファに放り込む
      strBuffer = strBuffer.substring(pos+1);
      pos = strBuffer.indexOf("¥n");
    }
  }
}
```

複数のクライアントが
接続してきている場合は
nextClient.ip();
で、接続IPを取得し、
それを利用して処理しよう！

クライアントから命令！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- サーバ役とクライアント役、どちらかがどちらかになって下さい。また、IPアドレスをそれぞれサーバ役の人のにし、サーバを実行し、次にクライアントを実行してみよう！

サーバ（TrojanServer.pde）

明治大学総合数理学部
先端メディアサイエンス学科

```
import processing.net.*;
import java.awt.*;
import java.awt.event.*;
Server myServer = new Server(this, 3333);
Robot  robot;

void setup() {
  size(20, 20);
  try {
    robot = new Robot();
  } catch (Exception e ) {
  }
}

void stop(){
  myServer.stop();
}
```

3333番ポートで準備

```
void draw()
{
  Client nextClient = myServer.available();
  if (nextClient != null ) {
    String recvStr = nextClient.readString();
    String[] data = split(recvStr, ',');
    // [0]番目がmoveだったら
    if (data[0].equals("move" ) ) {
      // doMoveに[1]番目と[2]番目を送ろう！
      doMove(int(data[1] ), int(data[2] ));
    }
  }
}
```

最初のカンマの前が動作に関すること

```
void doClick(int x, int y ) {
  robot.mousePress(InputEvent.BUTTON1_MASK);
  delay(1);
  robot.mouseRelease(InputEvent.BUTTON1_MASK);
  delay(1);
}

void doMove(int x, int y ) {
  robot.mouseMove(x, y);
}
```

クライアント（TrojanClient.pde）

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
import processing.net.*;
Client myClient = new Client(this, "133.26.54.5", 3333);

void setup() {
  size(1200, 800);
}

void stop(){
  myClient.stop();
}
void draw() {
}

// マウスが移動した時
void mouseMoved() {
  // "move,100,80," のように送信
  myClient.write("move," + mouseX + "," + mouseY + ",");
}
// マウスが押された時
void mousePressed() {
  // "click,100,80," のように送信
  myClient.write("click," + mouseX + "," + mouseY + ",");
}
```

133.26.54.5さんの
3333番ポートに接続

write() で
メッセージを送信

TrojanServerClient

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- サーバのマウスを操作してみよう！
  - "move,x,y" でマウスの移動情報を送信
- トロイの木馬でやっているのは、これの更に高度なこと

演習： クリックを追加してみよう！

参考：文字入力

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- RobotのKeyPressを利用することで、相手のコンピュータに対して文字入力も指示できる！
- 興味がある人はやってみよう！！

- ライブラリのロード
  - `import java.awt.event.KeyEvent;`
- キープレス
  - `robot.KeyPress(KeyEvent.VK_キーの種類);`

参考：スクリーンキャプチャ

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- RobotのcreateScreenCaptureを利用することで、画面のスクリーンキャプチャを撮ることが可能
- つまり、相手のデスクトップを見ることが可能
  - リモートからコンピュータを操作する際には、こういった組み合わせでやっています

課題2: basic_SCJankenGuriko

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 配布している下記フォルダを開こう！
  - basic_SCJankenGuriko

- 誰かがサーバ役、それ以外の2名がクライアント役をやりましょう

やり取りを追ってみる

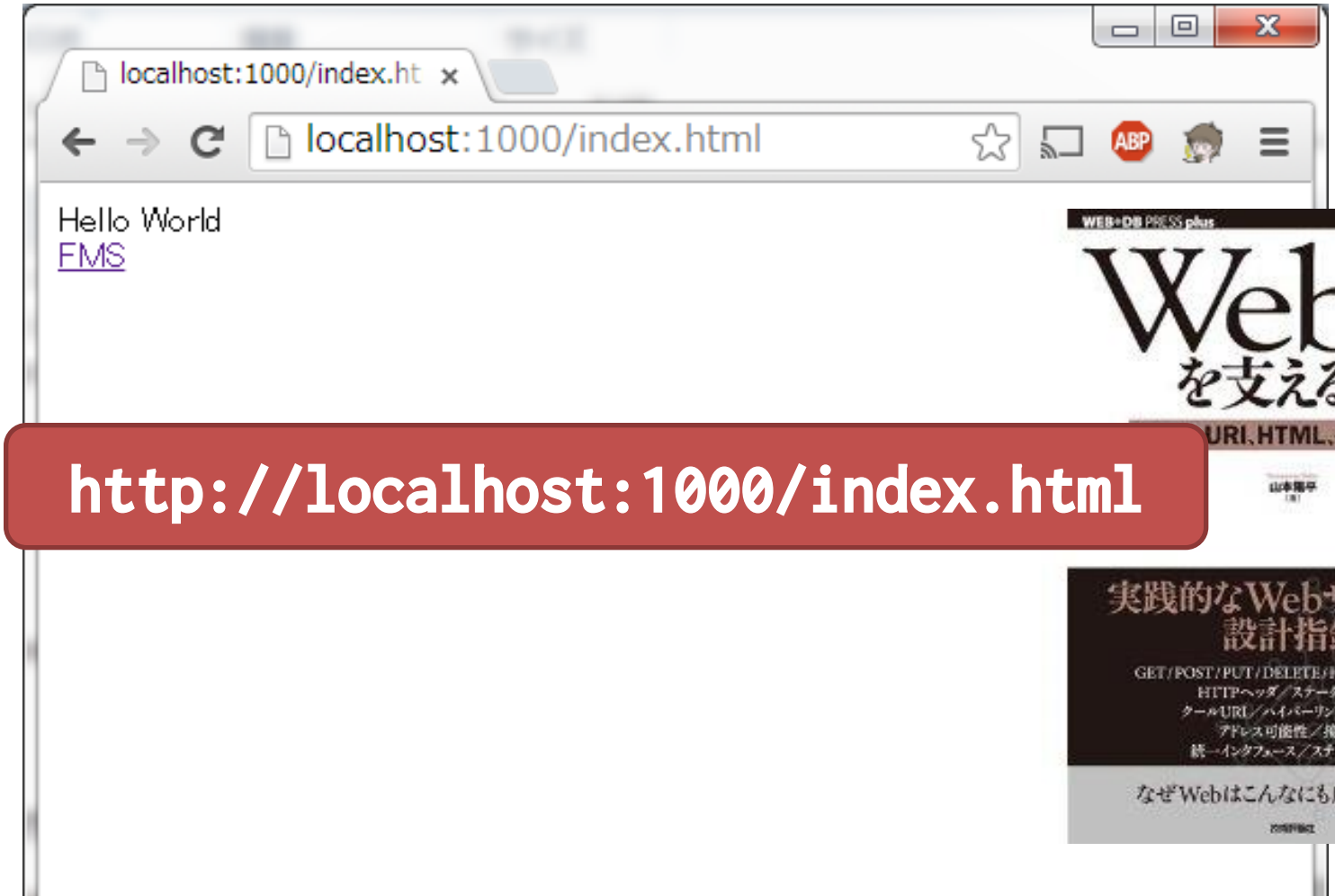
明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- サーバはクライアントが2つ揃うまで待つ
- クライアントは接続したらJOINを送る
- サーバはJOINを受けたら、ID1またはID2を送る。ID2まで来たら、ID1とID2にNextというメッセージを送る
- クライアントはNextというメッセージを受けたら、G/C/Pを送る
- サーバは両方からG/C/Pを受けたら、じゃんけんのジャッジをしてクライアントにWin/Loseを送る
- 決着がついていなかったらNextを送り、決着がついてたらWinner/Loserを送る

簡易ウェブサーバを作ろう

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Webサーバってどんな仕組みで成り立っているのだろうか？（Webを支える技術参照）



http://localhost:1000/index.html

簡易Webサーバを作ろう

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

SimpleWebServer.txt

```
// ライブラリを読み込む
import processing.net.*;
// サーバを1000ポートでスタート
Server myServer = new Server(this, 1000);

void setup() {}
void stop() {
  myServer.stop();
}

void draw(){
  Client nextClient = myServer.available();
  if(nextClient !=null){
    println(nextClient.readString());
    nextClient.write("HTTP/1.1 200 OK¥n");
    nextClient.write("Server: myServer¥n");
    nextClient.write("Content-type: text/html¥n");
    nextClient.write("¥n");
    nextClient.write("Hello World");
    myServer.disconnect(nextClient);
  }
}
```

writeでHTMLのヘッダと
Hello Worldを返す！

最後にdisconnectで切断

簡易Webサーバを作ってみよう

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

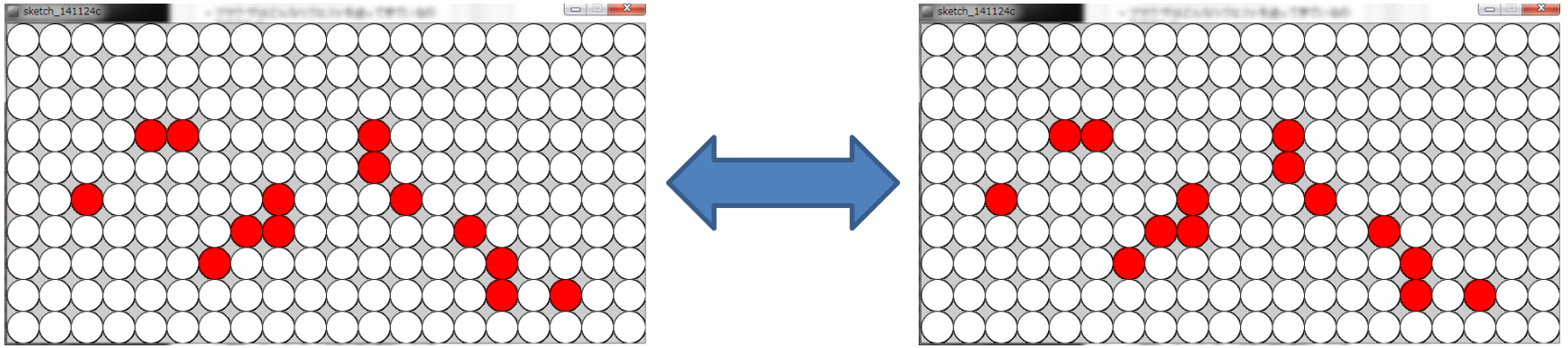
- 簡易Webサーバへのアクセス方法
  - ブラウザのURLに http://localhost:1000/ と入力してみよう！
  - 隣の人のサーバにアクセスしてみよう
    （localhostをIPに変更するだけ）

- リクエストに応じて処理を変えてみよう
  - http://localhost:1000/hello.html
  - http://localhost:1000/fms.html
- ブラウザはどんなリクエストを送ってきているのだろうか？

繋がる電光掲示板を作ろう

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 20x10の電光掲示板を作成し、サーバまたはクライアントの操作で、サーバとクライアントが同期して色が変わるようにするにはどうするか？

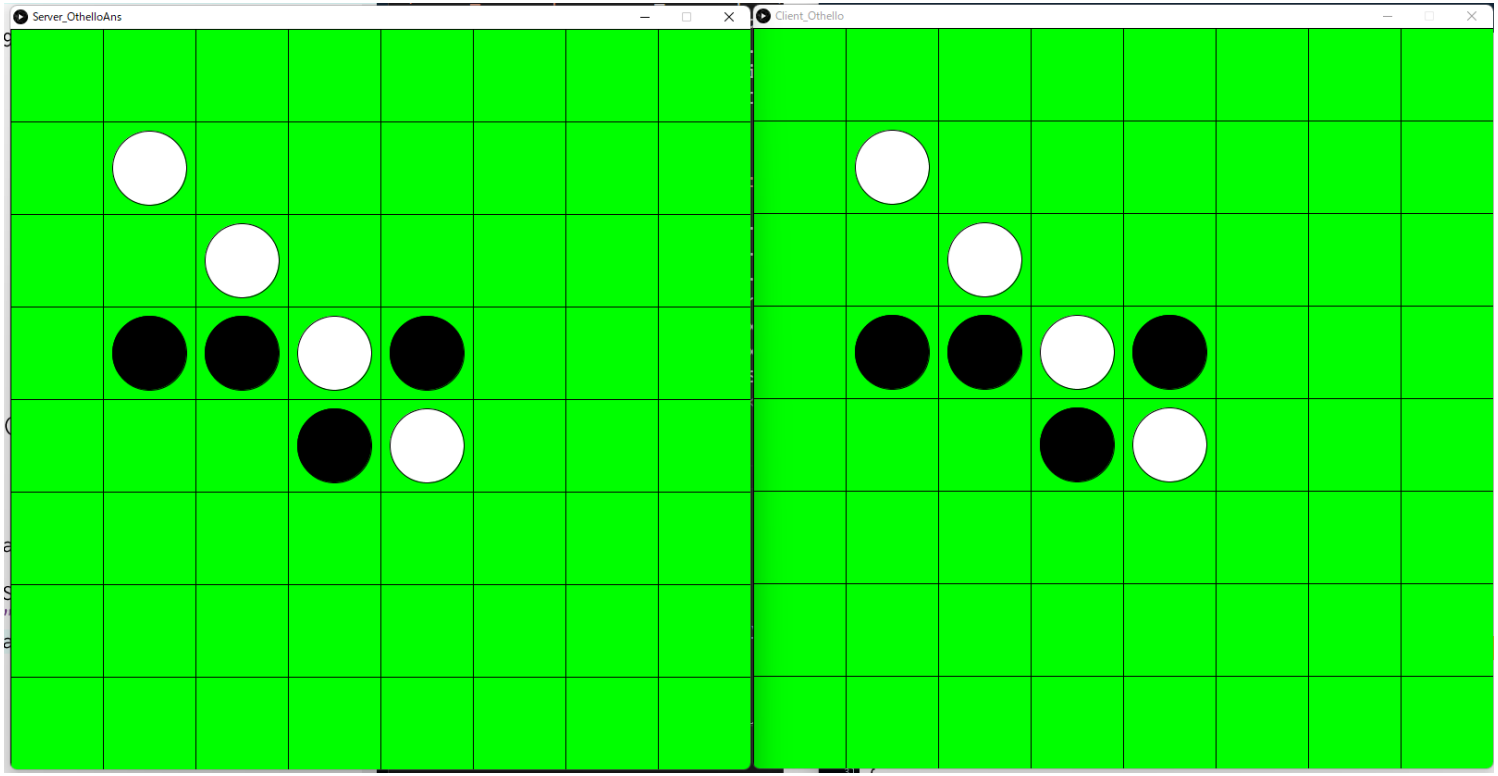


座標データを送り合えば良い？

課題3: オセロを作ろう

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- basic_SCOthelloを開いて、オンラインでオセロっぽいことをできるようにしてみよう



宿題 hw_ServerClientApp

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- サーバ・クライアントで通信を行う何らかのアプリを作れ
- 作るものは何でもOKです！

- 例
  - マルバツゲーム
  - オセロ
  - チャットアプリ