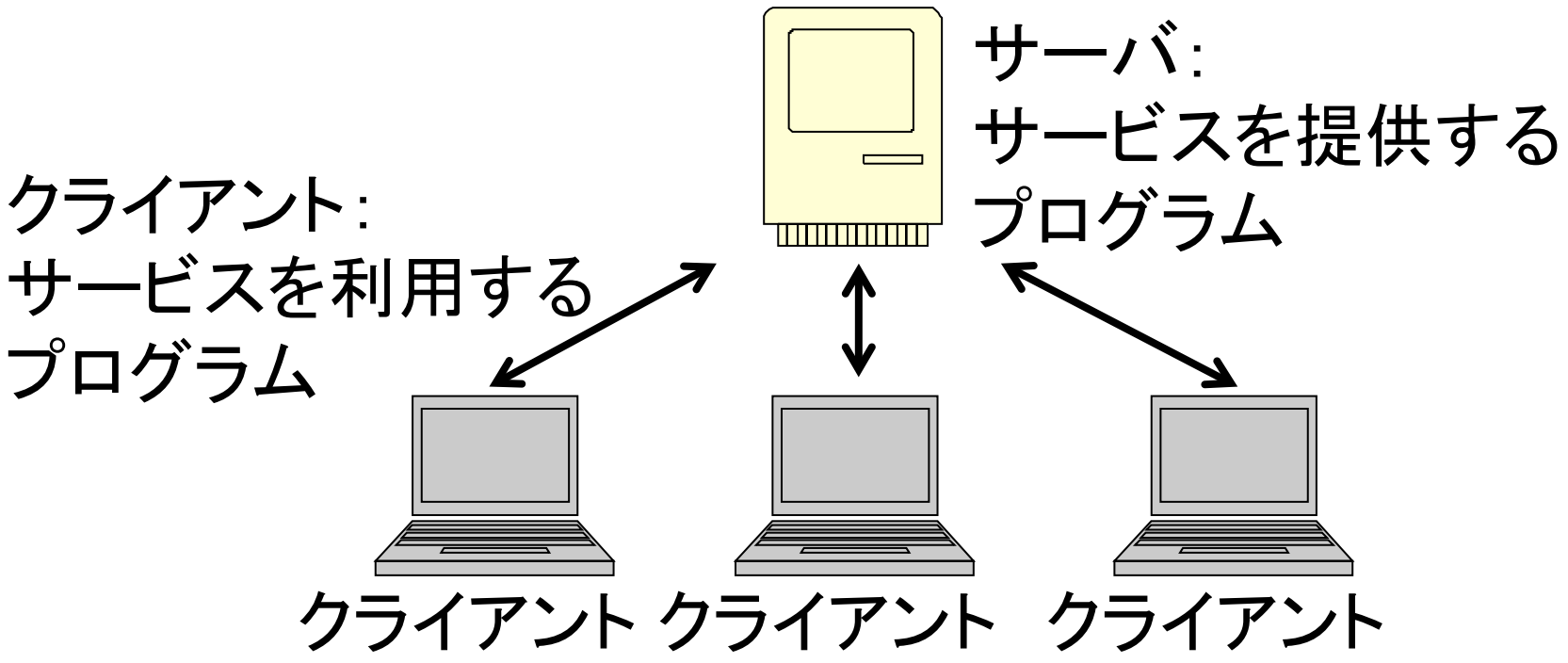
Processing で
通信するプログラムを作ろう

通信するプログラムの例

- ゲームのネットワーク対戦機能
- ネットワーク越しに一緒にお絵かきできる共有ホワイトボード

※Twitter, Facebook 等の SNS と連携するアプリは専用のAPIやライブラリを利用した方が良い

Server-Client モデル



- 今回は両方のプログラムを作ります

シンプルな例で、まずは写経から

Server

```
import processing.net.*;

Server server;

void setup(){
  server = new Server(this, 20000);
}

void draw(){
  Client c = server.available();
  if(c != null){
    String s = c.readString();
    println("server received: " + s);
    server.write(s);
  }
}
```

Client

```
import processing.net.*;
Client client;

void setup(){
  client = new Client(this, "127.0.0.1", 20000);
}

void draw(){}

void clientEvent(Client c){
  String s = c.readString();
  if(s != null){
    println("client received: " + s);
  }
}

void mouseClicked(){
  String s = "(" + mouseX + "," + mouseY + ") was clicked";
  println(s);
  client.write(s);
}
```

実行してみよう

1. Server → Client の順で実行する
   - 注意: println の内容は1つのウィンドウにまとめて表示されます
2. 複数の Client で接続してみよう
   - 方法: File > Export application を利用する
3. 他のコンピュータに接続してみよう
   - 方法: “127.0.0.1” の部分を接続したいコンピュータのIPアドレスに変更してから実行する

解説 server (1/2)

```
import processing.net.*;

Server server;

void setup(){
  server = new Server(this, 20000);
}

void draw(){
  Client c = server.available();
  if(c != null){
    String s = c.readString();
    println("server received: " + s);
    server.write(s);
  }
}
```

通信するプログラムに必要
Sketch > Import Library > net
で追加できる

Client からの接続受付開始

ここの20000は「ポート番号」
Client と同じ番号を使う

解説 server (2/2)

```
import processing.net.*;

Server server;

void setup(){
  server = new Server(this, 20000);
}

void draw(){
  Client c = server.available();
  if(c != null){
    String s = c.readString();
    println("server received: " + s);
    server.write(s);
  }
}
```

いずれかの Client が
データを送ってきたとき、
その Client を取得できる

送ってきた文字列を取得

接続している Client 全員に
データを送る

解説 client

```
import processing.net.*;
Client client;

void setup(){
  client = new Client(this, "127.0.0.1", 20000);
}

void draw(){}

void clientEvent(Client c){
  String s = c.readString();
  if(s != null){
    println("client received: " + s);
  }
}

void mouseClicked(){
  String s = "(" + mouseX + "," + mouseY + ") was clicked";
  println(s);
  client.write(s);
}
```

Server に接続する
“127.0.0.1” は
接続したい先アドレス
20000はポート番号

Server からデータが
送られてくると実行される

Server にデータを送る

もう少し複雑な例

- 文字列以外のデータを送りたい
  - たとえば、数字・座標・等々
  1. 文字列に変換して送る
  2. 受け取った側がもとのデータに変換しなおす

- 受け取ったデータを蓄えておきたい
  - ArrayList 等に入れておく

Client ※一部省略されていることに注意

```
Client client;
ArrayList l;

void setup(){
  size(400,400);
  l = new ArrayList();
  client = new Client(this, "127.0.0.1", 20000);
}
```

Client つづき

```
void draw(){
  fadeToBlack();
  fill(255,0,0);
  for(int i = l.size() - 1; i >= 0; i--){
    Animation a = (Animation) l.get(i);
    a.display();
    if(a.step()){ l.remove(a); }
  }
}

void mouseClicked(){
  client.write(mouseX + " " + mouseY + '¥n');
}

void clientEvent(Client c){
  String s = client.readStringUntil('¥n');
  if(s != null){
    String[] ss = splitTokens(s);
    int x = int(ss[0]);
    int y = int(ss[1]);
    l.add(new Animation(x, y));
  }
}
```

送るとき
1. データを空白でつなぐ
2. 末尾に '\n' を付ける

受け取るとき
1. readStringUntil を使う
2. splitTokens で分割する
3. Int 等に変換する

Server ※draw 以外は省略されていることに注意

```
void draw() {
  Client c = server.available();
  if (c != null) {
    String s = c.readStringUntil('¥n');
    if (s != null) {
      println("server received: " + s);
      String[] ss = splitTokens(s);
      int x = int(ss[0]);
      int y = int(ss[1]);
      ellipse(x, y, 20, 20);
      server.write(s);
    }
  }
}
```

Server 側でも
ほぼ同じプログラムで
クリックされた場所を表示

受け取ったデータをそのまま
すべての Client に配信

課題

- 通信する何かおもしろいプログラムを作る

- 対戦ゲームでもいいし
- 蟻が別世界に旅立ってもいいし
- なんでもいい