インターネットプログラミング
教室2406
水曜日7限19:50-21:20

第8回
2013/11/13

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%2F2013InternetProg

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あっと im.dendai
- 研究室内線2844

講師

- 慶應義塾大学　卒
- 元東京電機大学　戸辺義人先生
  OSOITEProject参加
- 東京大学生産技術研究所　助教
- 2013.4月より未来科学部情報メディア学科
  准教授

- 詳しくはFacebook　岩井将行
- Twitter @masaiwai

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[atmark--].-gmail.com>

ソケット通信

プログラム同士の通信は

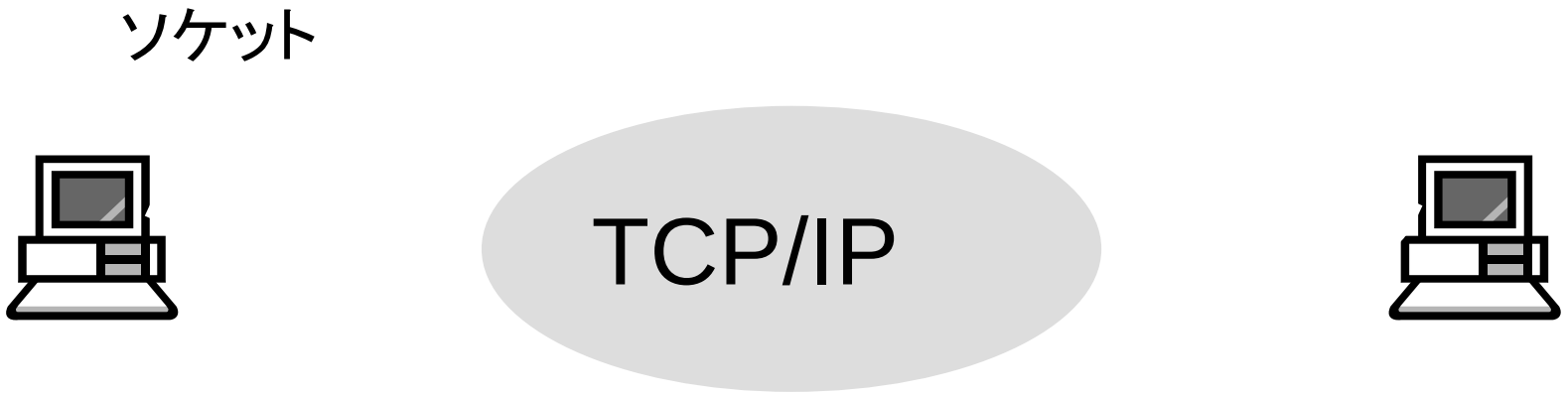
- ソケットを使ってデータの送受信
- ソケットを使った通信

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

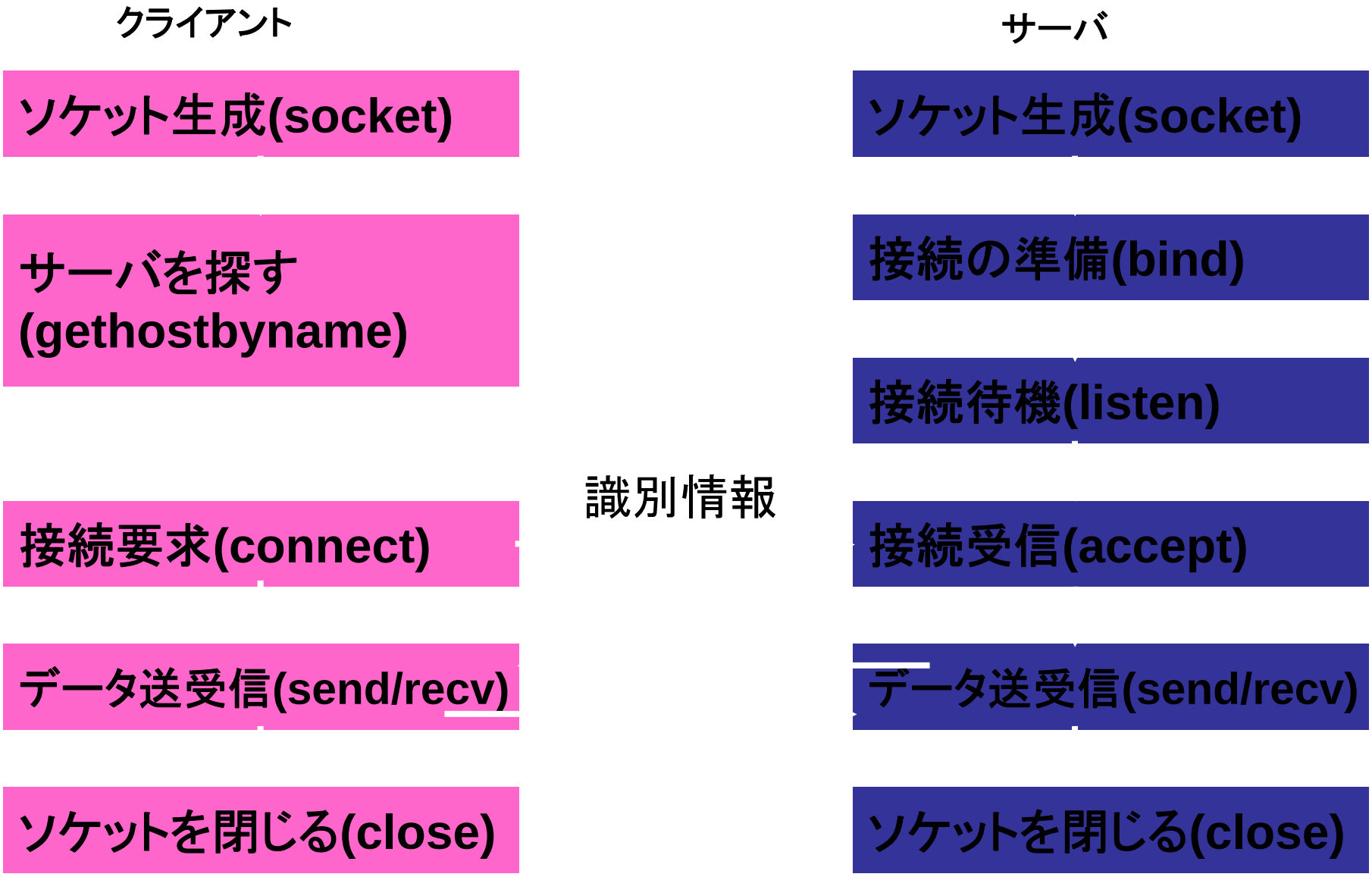
クライアントプログラム

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



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

- String**getHostName**()
  この IP アドレスに対応するホスト名を取得します。
- static InetAddress**getByAddress**(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 InetAddress**getLocalHost**()
  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

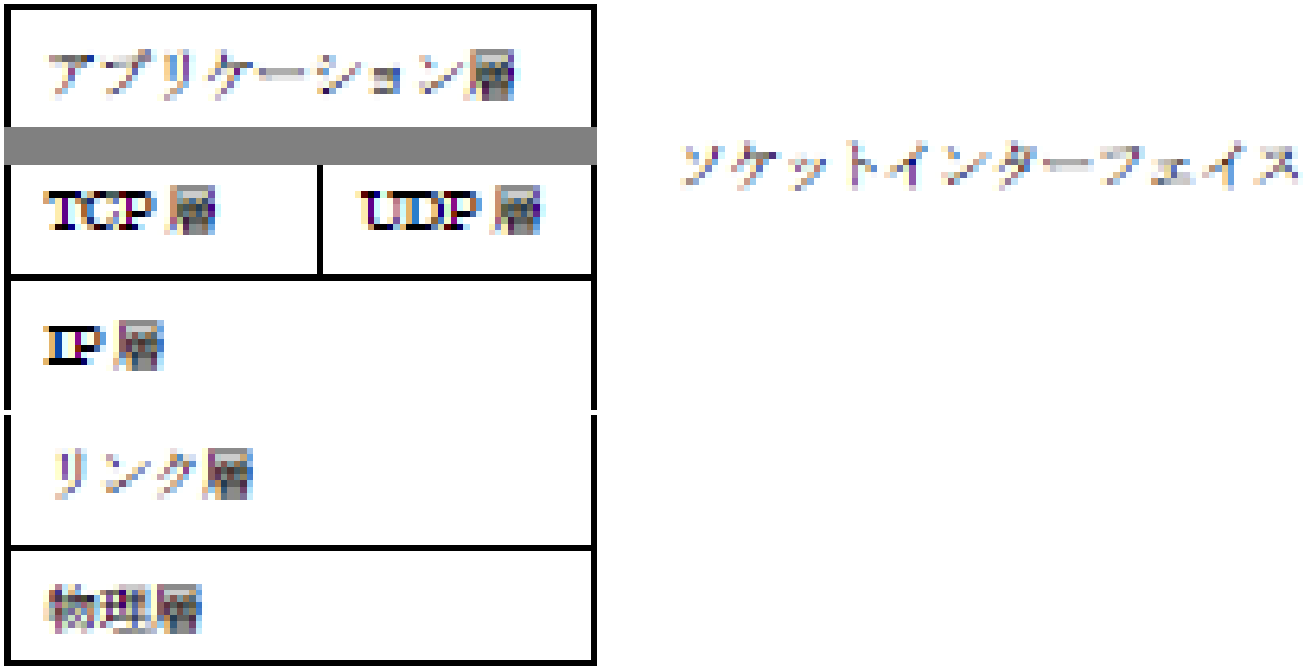
- String**toString**()
  この IP アドレスを String に変換します。

動かしてみよう。

- SwingAnimationBasic
- SwingAnimationFaceObj

- Swing　より美しいグラフィックスを提供 java2D
- ちらつき防止
- http://www.java2s.com/Tutorial/Java/0240__Swing/Catalog0240__Swing.htm

Socket interface



UDP通信

- UDP通信では接続という概念はない。
- ユーザは毎回データを送信し、また毎回自分のソケット、及び相手のソケットとアドレスを指定することになる。TCPでは最初に相手のソケットとの間の接続を行い、双方のソケット間の接続が確立されたら、互いの通信が可能になる。
- TCPではそのような制限はない。
- TCPでは接続が確立されたら2つのソケットはストリームのように振る舞う。TCPでは受信したデータの信頼性と順序が保障される。

DatagramPacket

- SocketやServerSocketでは、データのやりとりは入出力ストリームに抽象化されていたため、パケットという概念が見えてこなかった。
- DatagramPacketとDatagramSocketの場合はUDPパケットはUDPパケットとして抽象化されており、パケットにデータも送信先アドレスも含める必要がある。
- SocketやServerSocketとは別物

UDPServer

```
int serverPort = 5000;

DatagramSocket socket = new
    DatagramSocket(serverPort);

DatagramPacket receivePacket = new
    DatagramPacket(new byte[DMAX], DMAX);

socket.receive(receivePacket);


socket.close();
```

Udp Client

```
String servhostname="localhost";

InetAddress serverAddress =
    InetAddress.getByName(servhostname);
String message="hello UDP from yourname";
byte[] bytesToSend = message.getBytes();
int serverPort = 5000;
DatagramSocket socket = new DatagramSocket();
DatagramPacket sendPacket = new
    DatagramPacket(bytesToSend, bytesToSend.length,
    serverAddress, serverPort);
socket.send(sendPacket);
socket.close();
```

今週の課題[UDPの送受信をやってみよう]

- 堤出フォルダは第9回
- DendaiUDPClient1 DendaiUDPServ1を改造して、「送信したメッセージを真逆にして返してくる」
- 送信時はDendaiUDPServer5000番ポート
  返信時にはDendaiUDPClientの5001番ポートを指定すること