ネットワークアプリケーション

第4回 Javaによるネットワークプログラミング

石井 健太郎
（423研究室・オフィスアワー火3限）

スケジュール

9月27日　第1回「TCP/IPプロトコルスイート」
10月4日　第2回「Javaによるウィンドウプログラミング」
10月11日　第3回「ネットワークアプリケーションのプログラミングモデル」
10月18日　第4回「Javaによるネットワークプログラミング」
10月25日　第5回「Javaによるネットワークプログラミング」
11月8日　第6回「Javaによるネットワークプログラミング」
11月15日　第7回「Javaによるネットワークプログラミング」　最終課題（1）
11月17日　第8回「ウェブプログラミングについて」

スケジュール

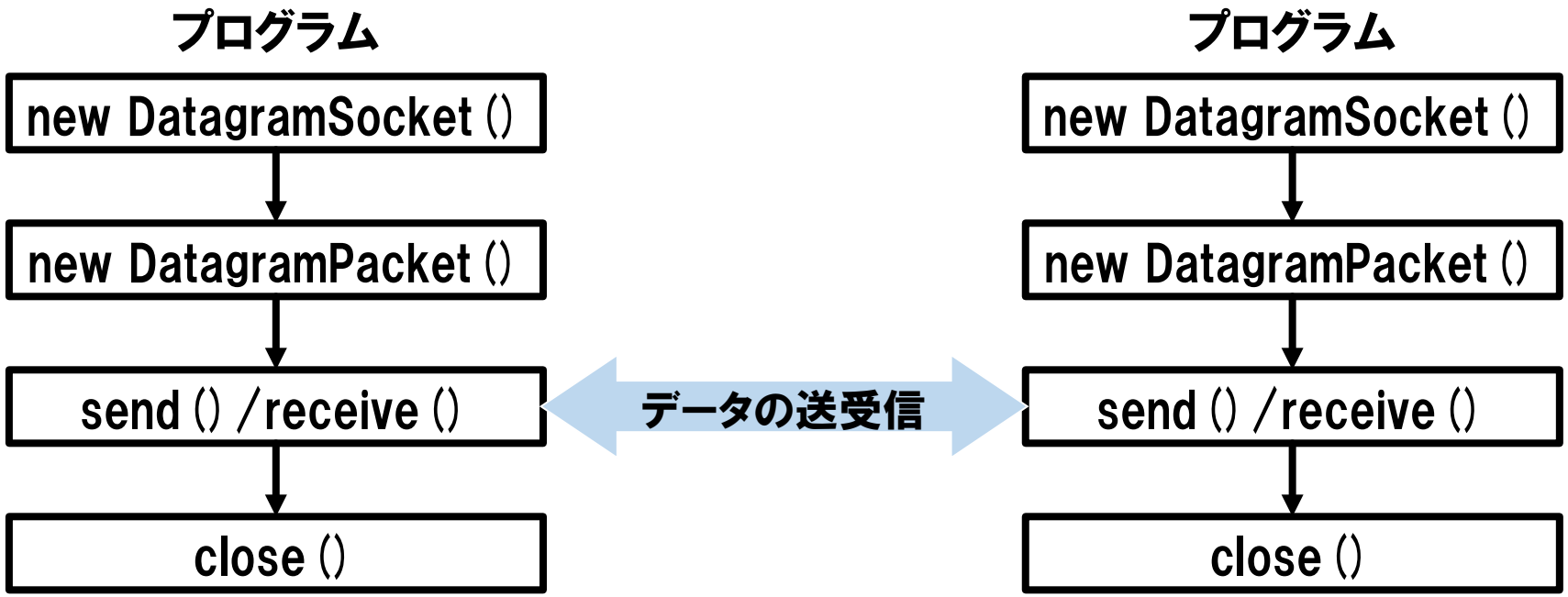
11月22日　第9回「JavaScriptによるクライアントサイドウェブプログラミング」
11月29日　第10回「JavaScriptによるクライアントサイドウェブプログラミング」
12月6日　第11回「JavaScriptによるクライアントサイドウェブプログラミング」
12月13日　第12回「JavaScriptによるクライアントサイドウェブプログラミング」
12月20日　第13回「JavaScriptによるクライアントサイドウェブプログラミング」
1月10日　第14回「JavaScriptによるクライアントサイド...」　**最終課題 (2)**
1月19日　第15回「まとめと演習」

UDPによるデータの送受信

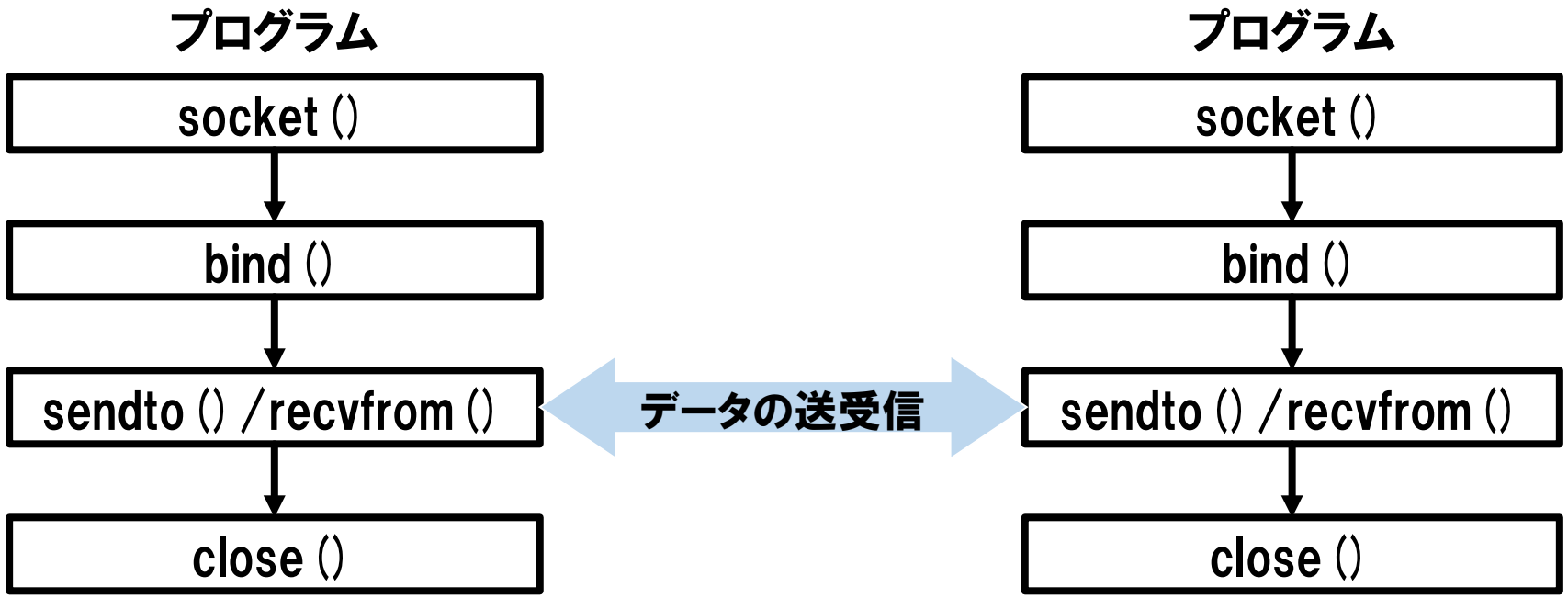
- コネクションレス（接続しない）
- データは byte型の配列 として送受信される
  - テキストデータのやりとりとバイナリデータのやりとりの方法は同じ

- DatagramSocketクラス と DatagramPacketクラス を用いる
  - cf. TCPは Socketクラス と ServerSocketクラス を用いる

UDPによるデータの送受信



（参考）UDPによるデータの送受信（C言語編）



UDPによるデータの送信

- テキストデータの送信

```
try {
    byte[] bytes = string.getBytes();
    DatagramPacket packet = new DatagramPacket(bytes, bytes.length, this.address);
    this.socket.send(packet);
} catch (SocketException se) {
    se.printStackTrace();
} catch (IOException ioe) {
    ioe.printStackTrace();
}
```

- バイナリデータの送信

```
try {
    byte[] bytes = ByteBuffer.allocate(4).putInt(i).array();
    DatagramPacket packet = new DatagramPacket(bytes, bytes.length, this.address);
    this.socket.send(packet);
} catch (SocketException se) {
    se.printStackTrace();
} catch (IOException ioe) {
    ioe.printStackTrace();
}
```

UDPによるデータの受信

- テキストデータの受信

```
byte[] buffer = new byte[1024];
DatagramPacket packet = new DatagramPacket(buffer, buffer.length);

try {
    this.socket.receive(packet);
} catch (IOException ioe) {
    ioe.printStackTrace();
    break;
}

byte[] data = packet.getData();
String string = new String(data);
System.out.println(string);
// this.application.setString(string);
```

- バイナリデータの受信

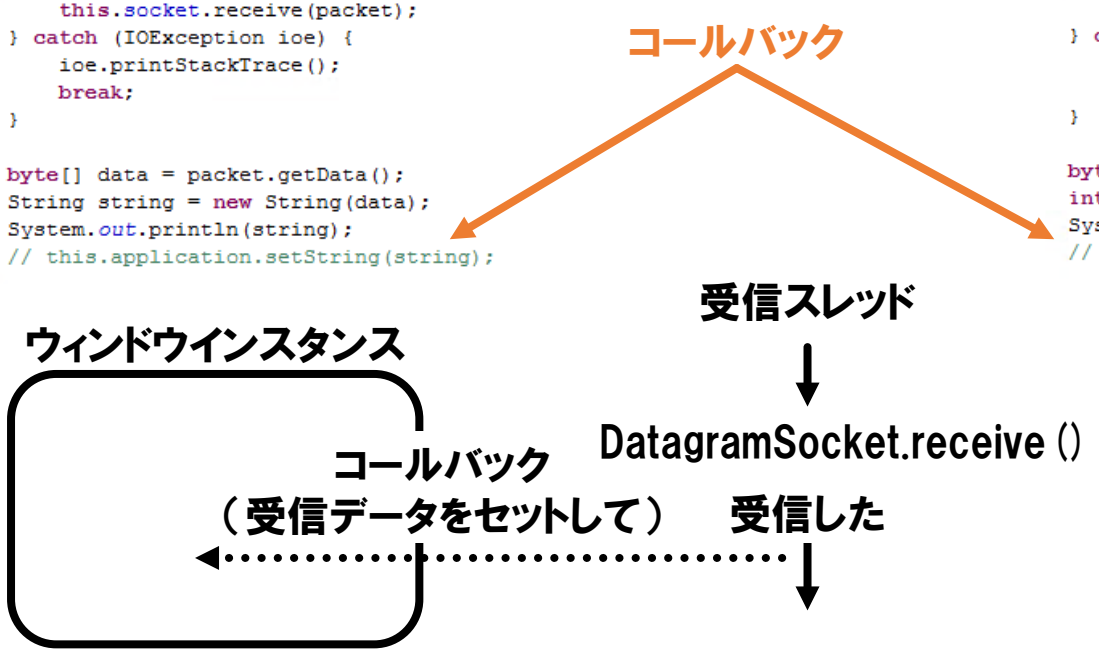
```
byte[] buffer = new byte[1024];
DatagramPacket packet = new DatagramPacket(buffer, buffer.length);

try {
    this.socket.receive(packet);
} catch (IOException ioe) {
    ioe.printStackTrace();
    break;
}

byte[] data = packet.getData();
int i = ByteBuffer.wrap(data).getInt();
System.out.println(i);
// this.application.setInt(i);
```

受信したときの処理

- 受信したときの処理を
  - 受信スレッドで処理できるときはそのまま処理する（エコーバックするなど）
  - ウィンドウに反映させたいときはコールバックインスタンスを用いる

```
byte[] buffer = new byte[1024];
DatagramPacket packet = new DatagramPacket(buffer, buffer.length);

try {
    this.socket.receive(packet);
} catch (IOException ioe) {
    ioe.printStackTrace();
    break;
}

byte[] data = packet.getData();
String string = new String(data);
System.out.println(string);
// this.application.setString(string);
```

```
byte[] buffer = new byte[1024];
DatagramPacket packet = new DatagramPacket(buffer, buffer.length);

try {
    this.socket.receive(packet);
} catch (IOException ioe) {
    ioe.printStackTrace();
    break;
}

byte[] data = packet.getData();
int i = ByteBuffer.wrap(data).getInt();
System.out.println(i);
// this.application.setInt(i);
```



演習

- UDPによるデータの送受信を用いて，ウィンドウアプリケーションを作成する
  - 基本課題
    - マウスのクリック位置を相手がたプログラムに送信して，そのデータを文字で表示させる
    - 受信したデータに応じた位置に図形を描画させるなど，
      見た目におもしろいプログラムに仕上げよう
  - 発展課題
    - クリック位置だけじゃなく，いろんなデータを送受信して，おもしろいアプリケーションを考えよう
      - 例えば，ドラッグすると相手がたに線が引ける協調お絵かきソフトとするなど

- プログラム（UDP*.java）の内容を，1つのテキストファイルにコピーして，
  次々回の講義開始時までに，提出フォルダ（Xドライブ）にファイルで提出
  - X:¥IN科専門¥石井講師¥ネットワークアプリケーション¥第4回
  - ファイル名は「<学生番号>.txt」とする（ハイフンなし）　例: N14999.txt