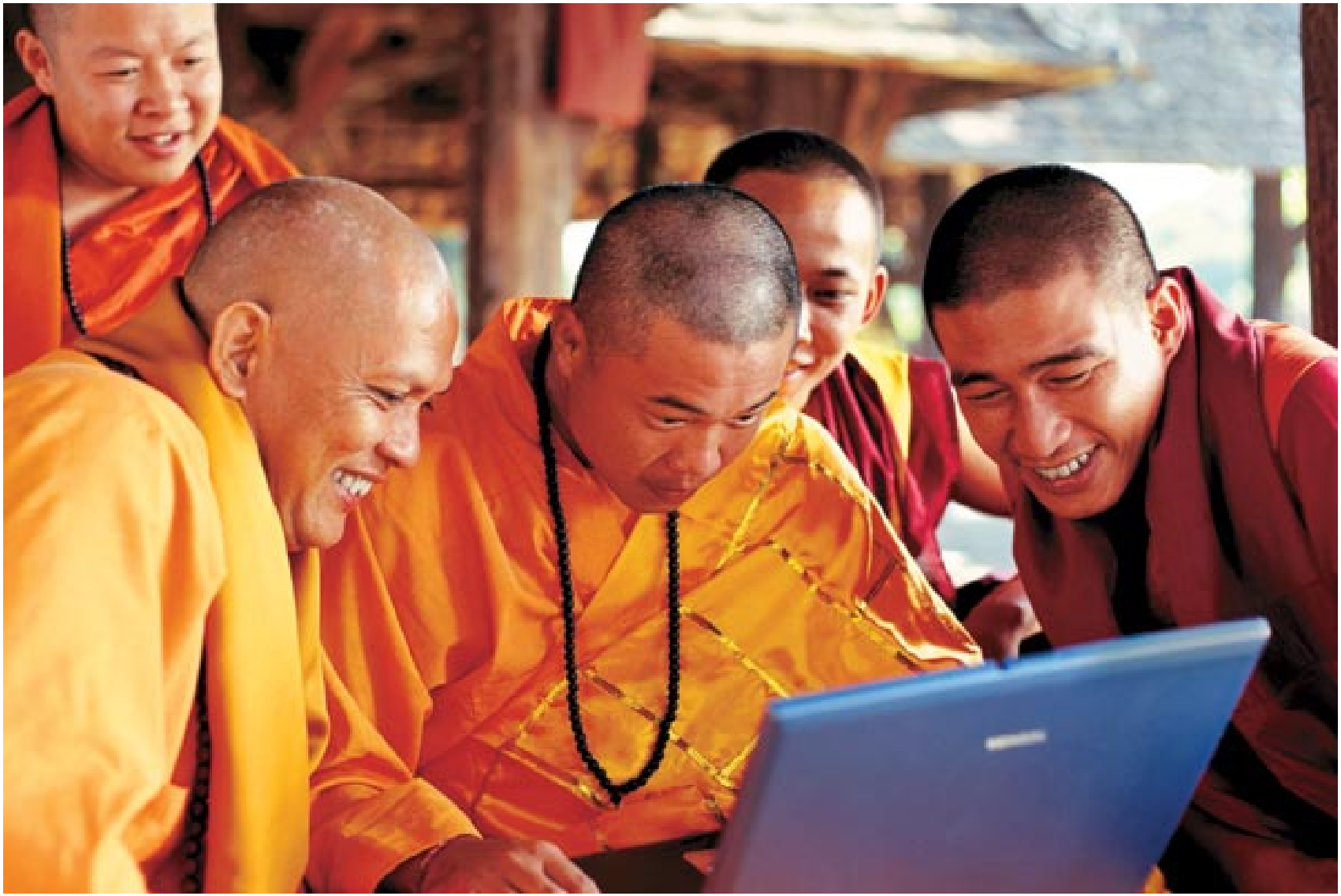
Разработка сетевых приложений на Java

Алексей Владыкин

1 декабря 2014

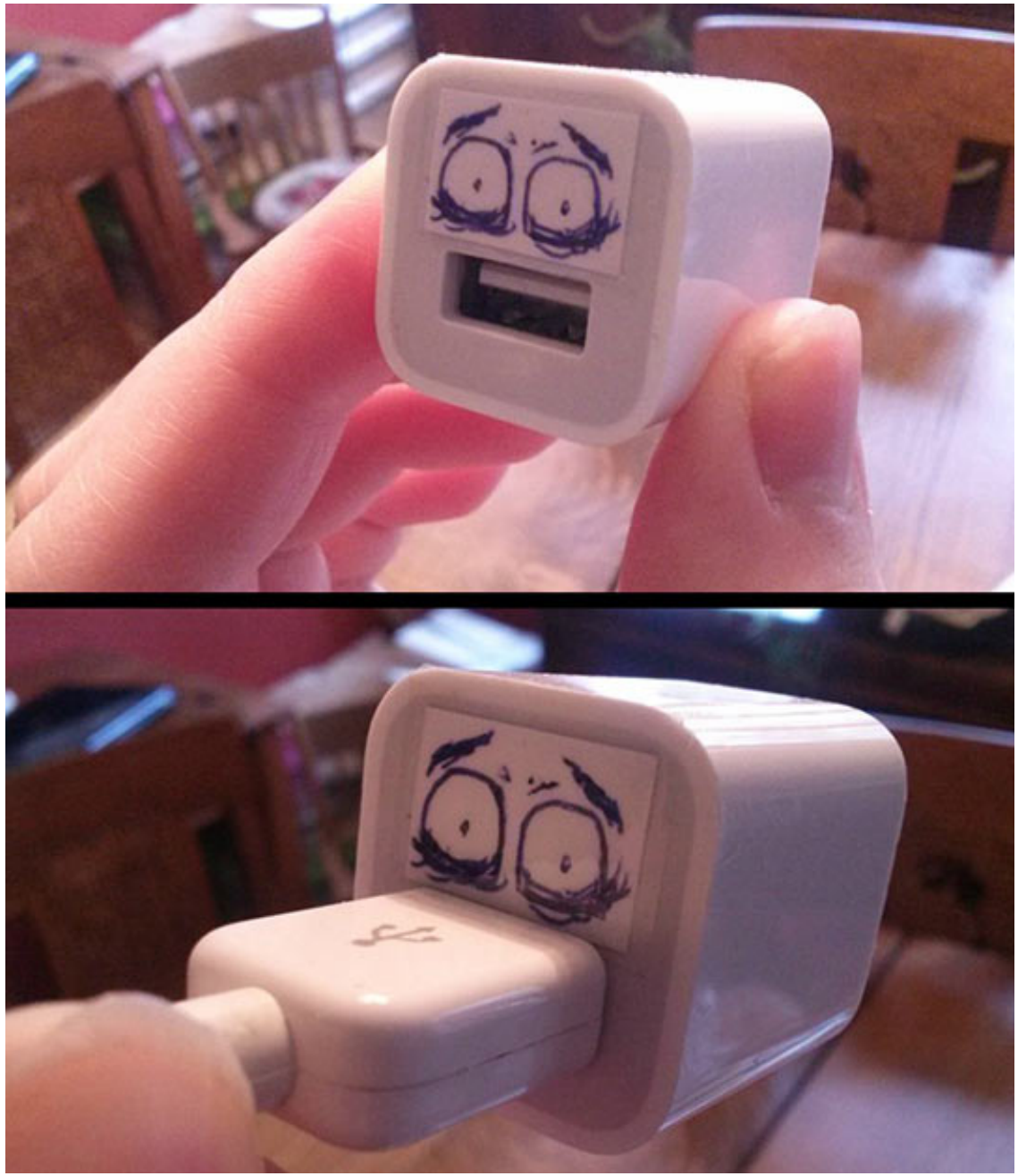


java.net.URI

- Uniform Resource Identifier, RFC 3986
- Примеры:
  - `mailto:java-net@java.sun.com`
  - `urn:isbn:096139210x`
  - `http://java.sun.com/j2se/1.3/`
  - `docs/guide/collections/designfaq.html#28`
- Синтаксические операции:
  - разбор на компоненты
  - `resolve`
  - `relativize`

java.net.URL

- Uniform Resource Locator, RFC 1738
- Примеры:
  - `http://java.sun.com/j2se/1.3/`
  - `file:/home/av/projects/`
- Поддерживает операции доступа: `openConnection()`, `openStream()`



- Низкоуровневый API для пересылки байтов по сети
- Поддерживаются протоколы TCP и UDP
- Поддерживается адресация IPv4 (`213.180.204.3`) и IPv6 (`2001:0db8:11a3:09d7:1f34:8a2e:07a0:765d`)

java.net.DatagramSocket (клиент)

```
try (DatagramSocket s = new DatagramSocket()) {
    DatagramPacket p = new DatagramPacket(
        buf, buf.length, remoteAddress);
    s.send(p);
}
```

java.net.DatagramSocket (сервер)

```
try (DatagramSocket s = new DatagramSocket(port)){
    byte[] buf = new byte[1024];
    DatagramPacket p = new DatagramPacket(
        buf, buf.length);
    s.receive(p);
}
```

java.net.Socket

- Клиентский сокет, устанавливает TCP соединение с сервером

```
Socket socket = new Socket("localhost", 11111);

OutputStream os = socket.getOutputStream();
os.write(requestBytes);
os.flush();

InputStream is = socket.getInputStream();
is.read(responseBytes);
```

java.net.ServerSocket

- Серверный сокет, ожидает подключений от клиентов

```
ServerSocket server = new ServerSocket(11111);
Socket socket = server.accept();

InputStream is = socket.getInputStream();
is.read(requestBytes);

OutputStream os = socket.getOutputStream();
os.write(responseBytes);
os.flush();
```

Что такое NIO?

- Расширение `java.io`, добавлено в 1.4
- Уровень абстракции ближе к ОС
- Поддерживается неблокирующий ввод-вывод
- Более высокая производительность

Основные понятия NIO

- Buffer
  Аналог массива, но может представлять области памяти ОС
- Channel
  Аналог потока, поддерживает операции чтения/записи
- Selector
  Сервис для отслеживания событий в каналах

Пример NIO

```
try (FileChannel src =
        new FileInputStream(in).getChannel();
     FileChannel dst =
        new FileOutputStream(out).getChannel()) {

    src.transferTo(0, src.size(), dst);
}
```


Netty

- Удобная библиотека для разработки сетевых приложений
- Инкапсулирует низкоуровневую работу с сетью и потоками

Channel

- `io.netty.channel.Channel`
- Аналог `java.net.Socket`
- Представляет конкретное подключение
- Позволяет писать и читать данные (асинхронно)

ChannelPipeline

- `io.netty.channel.ChannelPipeline`
- Цепочка обработчиков ChannelHandler, связанная с каналом
- Inbound — направление из сети в программу,
  outbound — направление из программы в сеть

События в канале

- Inbound: messageReceived, exceptionCaught, channelOpen, channelClosed
- Outbound: write, connect, disconnect, close

ChannelFuture

- `io.netty.channel.ChannelFuture`
- Аналог `java.util.concurrent.Future`
- Представляет результат асинхронной операции