Java

Message System

Александр Помосов

Reminder

ТЕХНОАТОМ

Отметьтесь на портале

Reminder

ТЕХНОАТОМ

Обновите репозиторий

Agenda

TEXHOATOM

- IO/NIO
- Serialization
- Reflection API
- Message System

Agenda

TEXHOATOM

- IO/NIO
- Serialization
- Reflection API
- Message System

IO

TEXHOATOM

http://docs.oracle.com/javase/tutorial/essential/io/

API for input and output to

- files
- network streams
- internal memory buffers
- …

IO API is **blocking**

IO. Byte streams

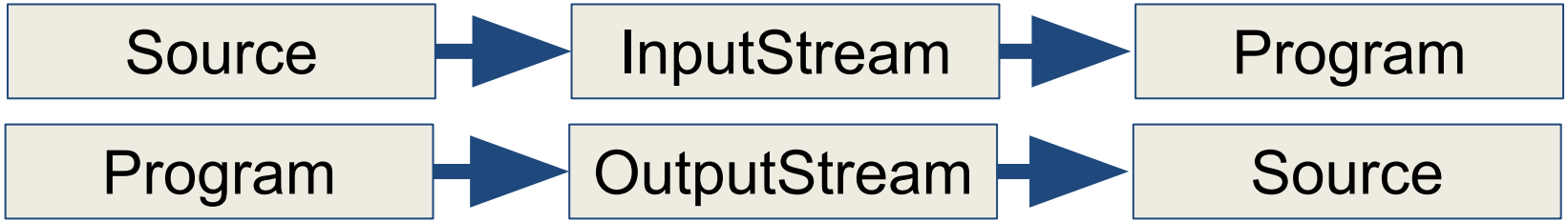
TEXHOATOM

- InputStream

  AudioInputStream, ByteArrayInputStream, FileInputStream, FilterInputStream, InputStream, ObjectInputStream, PipedInputStream, SequenceInputStream, StringBufferInputStream

- OutputStream

  ByteArrayOutputStream, FileOutputStream, FilterOutputStream, ObjectOutputStream, OutputStream, PipedOutputStream



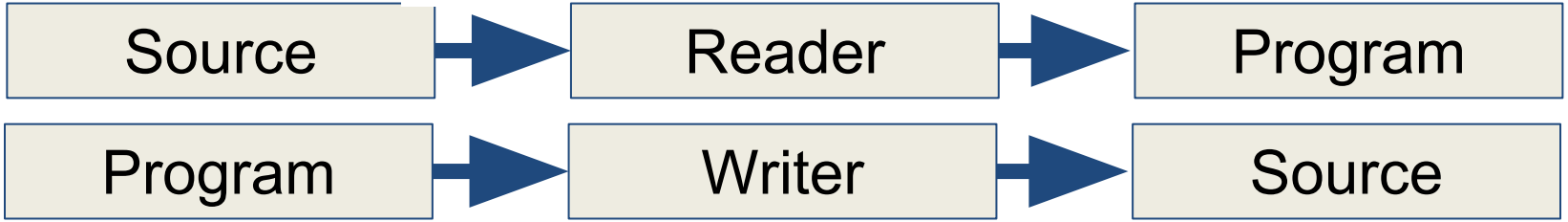
IO API is **blocking**

```
@see io.ByteStreams.java
@see System.out / System.err (PrintStream)
```

IO. Character streams

TEXHOATOM

- Reader — BufferedReader, CharArrayReader, FilterReader, InputStreamReader, PipedReader, StringReader
- Writer — BufferedWriter, CharArrayWriter, FilterWriter, OutputStreamWriter, PipedWriter, PrintWriter, StringWriter



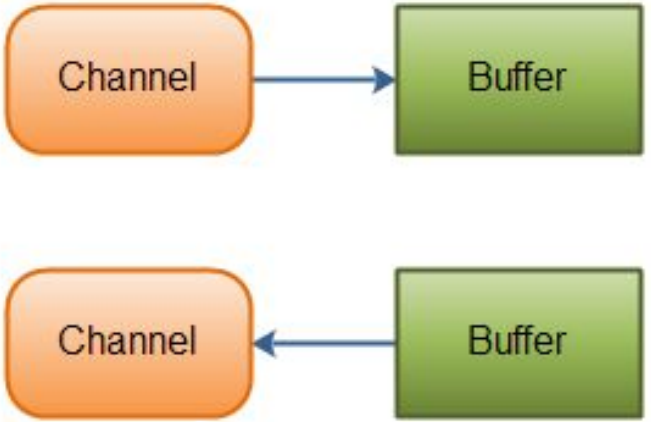
IO API is **blocking**

`@see io.CharacterStreams.java`

NIO

TEXHOATOM

- Channels
- Buffers



NIO API is **non-blocking**

For details @see http://tutorials.jenkov.com/java-nio/index.html

IO. File operations

TEXHOATOM

java.nio.file
contains modern file API

`@see nio.NIOFileAPI.java`

https://docs.oracle.com/javase/tutorial/essential/io/file.html

Agenda

TEXHOATOM

- IO/NIO
- Serialization
- Reflection API
- Message System

What is java serialization?

TEXHOATOM

Way to **persist** java object (serialize)
from java program
and to **load** persisted java object (deserialize)
into java program

Why need java serialization?

TEXHOATOM

?

Default Serialization

TEXHOATOM

What we need for Serialization to work:

1) implement Serializable (marker interface)
2) add class version
   ```
   private static final long serialVersionUID = ...L;
   ```
3) put java object to ObjectOutputStream(OutputStream);
   that is we can immediately save it into File or send it via network e.t.c.
4) Deserialize via ObjectInputStream(InputStream);

```
@see serialization.SerializationDeserializationTest.java
```

Serialization is deep

TEXHOATOM

Serialization is **deep**

that is, every object, referenced from serialized will be serialized. So everything in reference hierarchy (if not **transient**) must be **Serializable**

Almost all common library classes are serializable (Strings, Numbers, Collection and Maps implementations)

Serialization customization

TEXHOATOM

1) **transient** - ignore this field during serialization and deserialization

2) Implement **Externalizable** instead of **Serializable**

```
public interface Externalizable {
  //custom serialization logic here
  void writeExternal(ObjectOutput out) throws IOException;
  //custom serialization logic here
  void readExternal(ObjectInput in) throws IOException, ClassNotFoundException;
}
```

3) Use something beyond java serialization

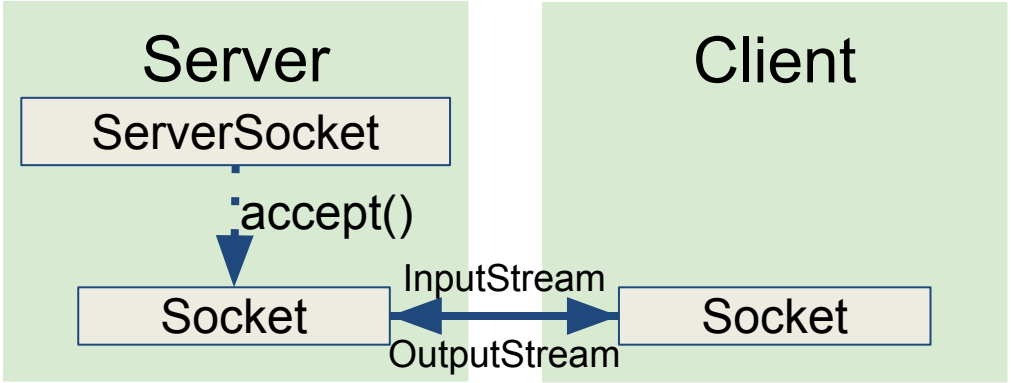
(store to custom json/xml/binary via library)

`@see serialization.SerializationDeserializationTest.java`

Practicum

TEXHOATOM

```
@see practicum
```



server accepts objects
of type Packet
Implement **client** for this server
and send Packet with
YOUR NAME AND SURNAME
as payload to my server

Agenda

TEXHOATOM

- IO/NIO
- Serialization
- Reflection API
- Message System

Reflection API

TEXHOATOM

Standard library API for accessing Type information at Runtime

- instanceof
- class Class<T> (and all the class contents)
- class ClassLoader

Official tutorial: https://docs.oracle.com/javase/tutorial/reflect/

Why need Reflection API?

TEXHOATOM

- Annotation processing
  (widely used inside frameworks)
- Class loading at runtime
- Introspection
  (for example for IDE or code generation toolchain)

```
@see ru.atom.reflection
```

Reflection Drawbacks

TEXHOATOM

- performance overhead
  reflection is actually fast, but it breaks some optimizations
  https://shipilev.net/blog/archive/reflection/
- security restrictions
  every reflective call goes through SecurityManager
  https://docs.oracle.com/javase/tutorial/essential/environment/security.html
- exposure of internals
  reflection breaks abstraction

One must use reflection Wisely!
(actually as part of specific design patterns)

Configuration via reflection

TEXHOATOM

```
@see reflection.configuration_via_reflection
```

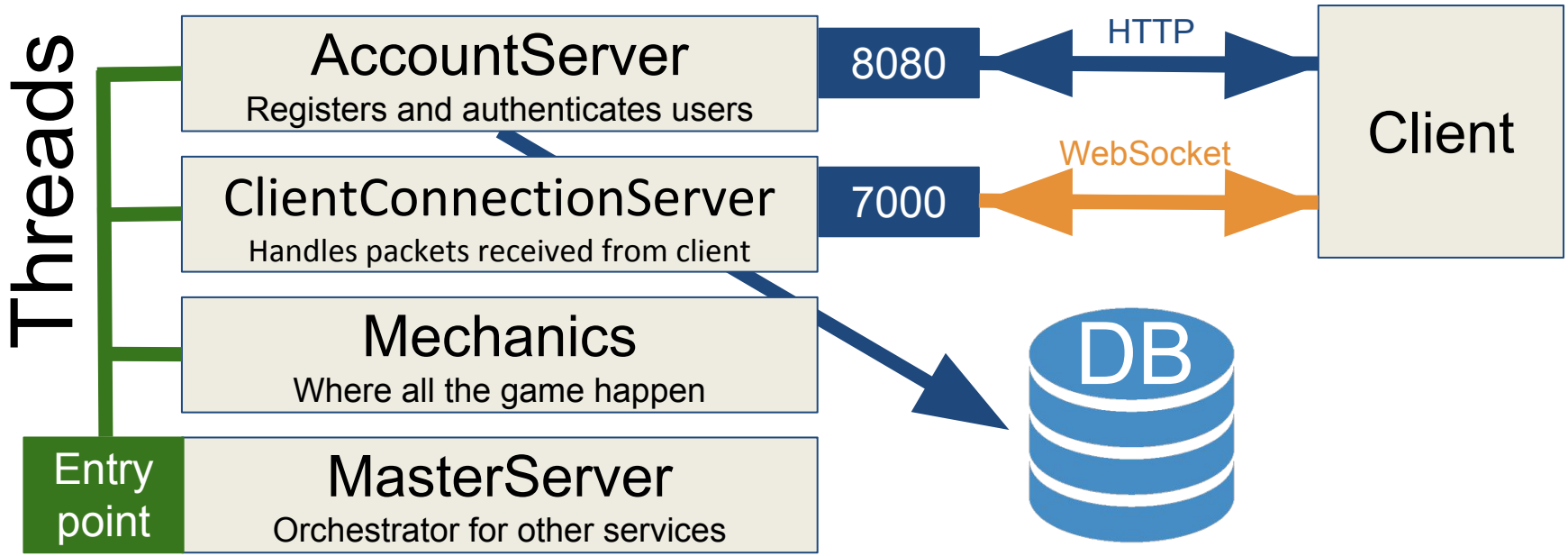
Agenda

TEXHOATOM

- IO/NIO
- Serialization
- Reflection API
- Message System

How do services interact?

TEXHOATOM



Communication problem

TEXHOATOM

We want services to do it’s job. No service must do the work for other service.

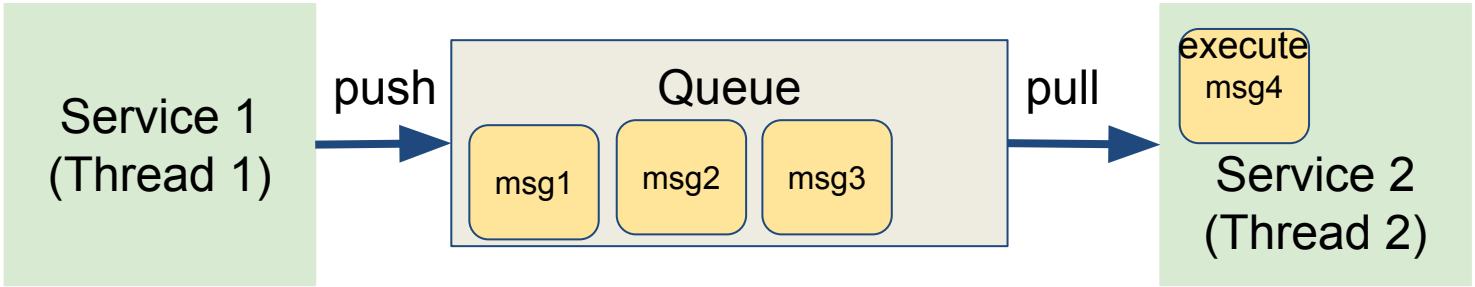
Example:
all mechanics actions must be executed in Mechanics Thread (Service), even if they are triggered from ClientConnectionService

(Why?)

Message System idea

TEXHOATOM

- Threads (Services) send messages to each other
- Messages contain code to be executed in other service
- Messages are stored within thread-safe queue
- In loop service executes messages from his queue



Message System

TEXHOATOM

```
@see zagar_server/message_system
```

ТЕХНОАТОМ

Спасибо за внимание!

Александр Помосов

alpieex@gmail.com