ThriftUsageJava

Assuming you're using the definitions in "tutorial.thrift":

Importing the Classes

You may not need all of these.

```
import org.apache.thrift.TBase;
import org.apache.thrift.TException;
import org.apache.thrift.TSerializer;
import org.apache.thrift.TDeserializer;
import org.apache.thrift.protocol.TBinaryProtocol;
import org.apache.thrift.protocol.TProtocol;
import org.apache.thrift.protocol.TProtocolFactory;
import org.apache.thrift.protocol.TSimpleJSONProtocol;
import org.apache.thrift.transport.TIOStreamTransport;
import org.apache.thrift.transport.TTransport;

import tutorial.Work;
import tutorial.Operation;
import tutorial.Constants;
```

Creating Objects, Using Constants

```
Work work = new Work();
work.num1 = 7;
work.num2 = 9;
work.op = Operation.ADD;
```

Or, if you used the "beans" option to generate ThriftJavaBeans code:

```
Work work = new Work();
work.setNum1(7);
work.setNum2(9);
work.setOp(Operation.ADD);
```

Serializing to/from a Byte Array

```
TSerializer serializer = new TSerializer(new TBinaryProtocol.Factory());
byte[] bytes = serializer.serialize(work);

TDeserializer deserializer = new TDeserializer(new TBinaryProtocol.Factory());
Work moreWork = new Work();
deserializer.deserialize(moreWork, bytes);
```

Serializing to "Simple" JSON

```
TSerializer serializer = new TSerializer(new TSimpleJSONProtocol.Factory());
String json = serializer.toString(work);
```

The "Simple" JSON protocol produces output suitable for AJAX or scripting languages. It does not preserve Thrift's field tags and cannot be read back in by Thrift. Using the object created above, the "Simple" JSON string is:

```
{"num1":7,"num2":9,"op":1}
```