

Forum:[Hack Forums](#)

Thread: [Beginner Networking - Sockets with Serialization \(Tutorial\)](#)

Aussie - June 8, 2019, 12:14 am

Very simple tutorial for beginners to learn how to send serialized classes/POJOs through sockets.

I hope this helps some beginners ?

Give me more ideas for tutorials, and I'll make them!

GenericSkid - June 8, 2019, 11:05 am

Minor detail, but I like the simple editing (Zooming in on relevant area)

Presentation

Anyways first I have a question, who is the target audience? As in, what level of skill are you looking to help out here? I get the thread title says beginner, but by just watching the video I wouldn't readily assume that.

For new users:General pacing is way too fast

Using keybinds (Control + Shift + O) can also confuse these people

You should have no issues of your own present (confusion moments, jdk8 vs jigsaw, exceptions, clicking buttons accidentally) while presenting a topic (If I do this I fix it and backtrack a bit, then continue on in the same recording. Editing can cut out the mistake)

For moderate users:Fine as is

For seasoned users:Too slow, obvious sections can be sped up / cut out and redubbed in editing

Simplistic implementation, you can make this more abstract and then dive into proper software engineering principles like architecture & maintainability

Content

Proper logging would be a bonus (Then using the logging framework to log the exception rather than letting it be raised all the way to the entry-point)

Client and server in this example have some common logic, a common parent can be made to keep things simple.

Using Serializable is simple, [but inefficient](#). Plus it makes code version-specific, so if you had an outdated packet structure being sent to a new server (With a modified packet class), it wouldn't be able to properly deserialize the incoming packet. The link before is a bit old, but gets the point across.

Future Ideas

Perhaps you can take a look at [what skills are in demand](#) and make tutorials for those. I've wanted to make tutorials for a while but haven't for a variety of reasons. If I were doing a series though I would cover them in like this (Moderate-Advanced target audience):

MavenUsing dependencies (Looking stuff up on maven central / google)

Using build plugins to generate jar files for you that include your dependencies instead of relying on an IDE's jar export feature

Unit TestingJUnit 5 as a maven dependencyWhy do we write tests?

Good practices while writing tests

Actually writing the tests, what the different annotations mean, etc.

Using a proper maven project structure so that you have the following:src/main/java - Used to write the client logic

src/test/java - Used to verify client logic

Using maven's surefire plugin to do automated testing when building your project to verify that things work before you generate your jar file

Test-driven development & other design philosophiesTDD: The design approach where you write test cases that outline how you want your code to work. It won't compile at first, but then after you write something that looks and feels pleasing you start filling in the holes until it works as you intended when writing the example code.

GitWhy is version control useful?

Examples:Cloning an existing project online and contributing

Making a new project and uploading it somewhere

Online services with Github (Continuous Integration to verify that your committed code compiles and passes unit tests)

Maintainability principlesWriting code that is easy to understand, why it is important, and why it should be more important than writing fast/efficient code.

Refactoring existing bad code into something that is maintainable.

The importance of writing documentation (Javadoc & inline comments)When and how to write comments... Don't write comments like "constructs the XYZ class". Write comments that explain the logic of a whole block of code in a TLDR way. Give examples of this in realistic samples.

Java core (Give small examples of what packages do. Keep the samples minimal)Regular expressions

Stream & functional programming

io vs nio

Concurrency (Bit dated, but using RMI as an example to work off of would be killing 2 birds with one stone)

The UI frameworks (Swing vs JavaFx)

Reflection & InvokeAnnotations

[Meta-programming concepts](#)

Java 8 vs Jigsaw & the future of Java

Aussie - June 9, 2019, 12:34 am

(Jun 8, 2019 11:05 AM)GenericSkid Wrote: [\[CITE\]](#) -snip-Thanks for the feedback! Yeah, the tutorial

is intended to help beginners who've ventured past the whole core Java concepts and are ready to investigate further into networking. The reason why I wanted to make this video was because there weren't too many videos on this (only one from my search results).

Yeah, I agree. It's kind of hard to estimate how fast you should pace the video depending on what audience is watching. I'll have to strike a balance.

I feel like for other content, I have to try and make stuff that there aren't a plethora of videos on. I want to do some container orchestration videos (Kubernetes) and maybe even some Spring/Hibernate tutorials because currently that section is dominated by people with very thick/hard to understand accents. (No offense to any Indians).

GermanShepherd - *June 13, 2019, 9:19 am*

thank you for this useful tutorial

5H4R4F2003 - *June 14, 2019, 1:08 pm*

Thank you very much for thr tutotial, it is very hopefuk

SilenceHacker - *July 15, 2019, 2:02 pm*

Thanks for the tutorial. It'll be helpful for me in the future.