

GeoServer 2.28 Setup

I want to download the latest version of GeoServer 2.28 from source and compile it as a WAR (to be deployed in any Servlet container). Can you advise on what Java version I would need and what is a possible(lightweight, if possible) servlet container to use? I am on Ubuntu 20.04.

Here's how to approach **downloading, building from source, and deploying GeoServer 2.28 as a WAR** on your Ubuntu 20.04 system — including the Java requirements and options for lightweight servlet containers.

✓ 1) Java Version Requirements

For **GeoServer 2.28.x**, the minimum required Java version is:

- **Java 17** — minimum supported and tested.
- **Java 21** — also supported and tested with GeoServer 2.28.x.
- **Java 11 and earlier are not supported for 2.28.x.** You should use **OpenJDK 17** at minimum. [GeoServer +1](#)

On Ubuntu 20.04, install OpenJDK 17:

```
bash

sudo apt update
sudo apt install openjdk-17-jdk
```

Verify:

```
bash

java -version
```

You should see something like:

```
nginx

openjdk version "17.x"
```

✓ 2) Downloading the Source Code

GeoServer's source is hosted on GitHub:

```
bash

git clone https://github.com/geoserver/geoserver.git
cd geoserver
# Checkout the 2.28 branch
git checkout 2.28.x
```

(or download the **2.28.x** source .zip from GitHub releases).

✓ 3) Building the WAR from Source

GeoServer uses **Apache Maven** as its build system.

Install Maven

```
bash

sudo apt install maven
```

Build

From the top of the source tree:

```
bash

mvn clean install -DskipTests
```

This will compile the project and build all modules — including the web application.

The `geoserver.war` will be generated in `web/app/target/` (or similar depending on the exact module structure).

If you want *all extensions*, you can add:

```
bash

mvn clean install -P allExtensions -DskipTests
```

This takes longer but includes optional modules. GeoServer

✓ 4) Choosing a Lightweight Servlet Container

GeoServer's WAR can be deployed to **any Servlet 4 container** (servlet API 4.x), but the community has historically tested these:

★ Recommended (most robust)

- **Apache Tomcat 9.x** — supported for GeoServer 2.x and tested with Java 17+. GeoServer

Lightweight Alternatives

If you want **lighter than Tomcat**, these work as well:

Container	Notes
Jetty 9.4.x	Lightweight, embeddable, supports Servlet API 3.1/4.0 required by GeoServer 2.28. <small>GeoServer</small>
Payara Micro	Lightweight GlassFish variant (heavier than Jetty/Tomcat, but still manageable).
Undertow	Ultra-lightweight (but requires extra config since GeoServer expects typical servlet deployment).

Important: GeoServer 2.x is *not* compatible with **Tomcat 10+** using Jakarta EE 9/Servlet API 5+ out of the box — stick to **Tomcat 9** or Jetty 9.x for that version. GeoServer

✂ Example: Deploy WAR to Jetty (lightweight)

1. Install Jetty:

```
bash  
  
sudo apt install jetty9
```

2. Copy the WAR:

```
bash
```

```
cp path/to/geoserver.war /var/lib/jetty9/webapps/geoserver.war
```

3. Start/restart Jetty:

```
bash
```

```
sudo systemctl restart jetty9
```

4. Access GeoServer in browser:

```
bash
```

```
http://localhost:8080/geoserver
```

(Replace `localhost` and port as needed.)

Tips

Java Options for GeoServer Build

GeoServer build might require specific JVM flags due to reflective access — this is usually handled by Maven, but if you encounter build errors you can add:

```
bash
```

```
export MAVEN_OPTS="- -add-opens=java.base/java.lang=ALL-UNNAMED"
```

but for most builds on OpenJDK 17 this isn't usually needed. GeoServer

Skip Tests for Faster Build

During development the tests can make builds slow. Use:

```
bash
```

```
mvn clean install -DskipTests
```

Summary

Task	Recommendation
Java version	OpenJDK 17 (minimum), Java 21 supported GeoServer

Task	Recommendation
Build tool	Maven (<code>mvn clean install</code>) <small>GeoServer</small>
WAR deployment containers	Tomcat 9.x (recommended) or Jetty 9.4 (lightweight) <small>GeoServer</small>
Ubuntu commands	<code>sudo apt install openjdk-17-jdk maven jetty9</code>

If you want, I can provide a **step-by-step script** that automates the build and deploy to Tomcat or Jetty on Ubuntu 20.04. Just let me know which container you prefer!

 Sources