CT5106
SOFTWARE ENGINEERING II

1: Introduction

Lecture Topics (not fully pinned down yet)

- Technical Side
  - Web Application Development
    - Servlets
    - Java Server Pages
    - Java Server Faces
  - Enterprise Java Beans
    - Session Beans
    - Entity Beans
    - Message Driven Beans
  - Contexts and Dependency Injection
  - Persistence
    - JDBC
    - Java Persistence API
  - Web Services with JAX-RS
  - WebSocket

- Software Processes (DevOps) Side
  - Continuous Integration / Delivery / Deployment
  - Project Management

Module Format

- 1 assignment per week
- Exam at end of semester
- Marking: 50% exam + 50% CA

What is Java Enterprise

- The Java Platform, Enterprise Edition (Java EE) builds upon the Java SE platform and provides a set of technologies for developing and running portable, robust, scalable, reliable and secure server-side applications
- This means not just server-side web programming, but also business logic, object persistence, messaging, security,…
- It is built on the client-server model, and in particular we will be looking at implementing MVC applications

Jakarta = Java EE (Enterprise Edition)

- Java EE is a set of (interoperable) API specifications
- Was known as JEE and J2EE – currently known as Jakarta EE (handed over from Oracle to Eclipse Foundation, Java EE 8 became Jakarta 8)
  - Specifications created through a community approval process
- Java EE 8 still widely deployed, and application servers mainly support both Jakarta and Java EE

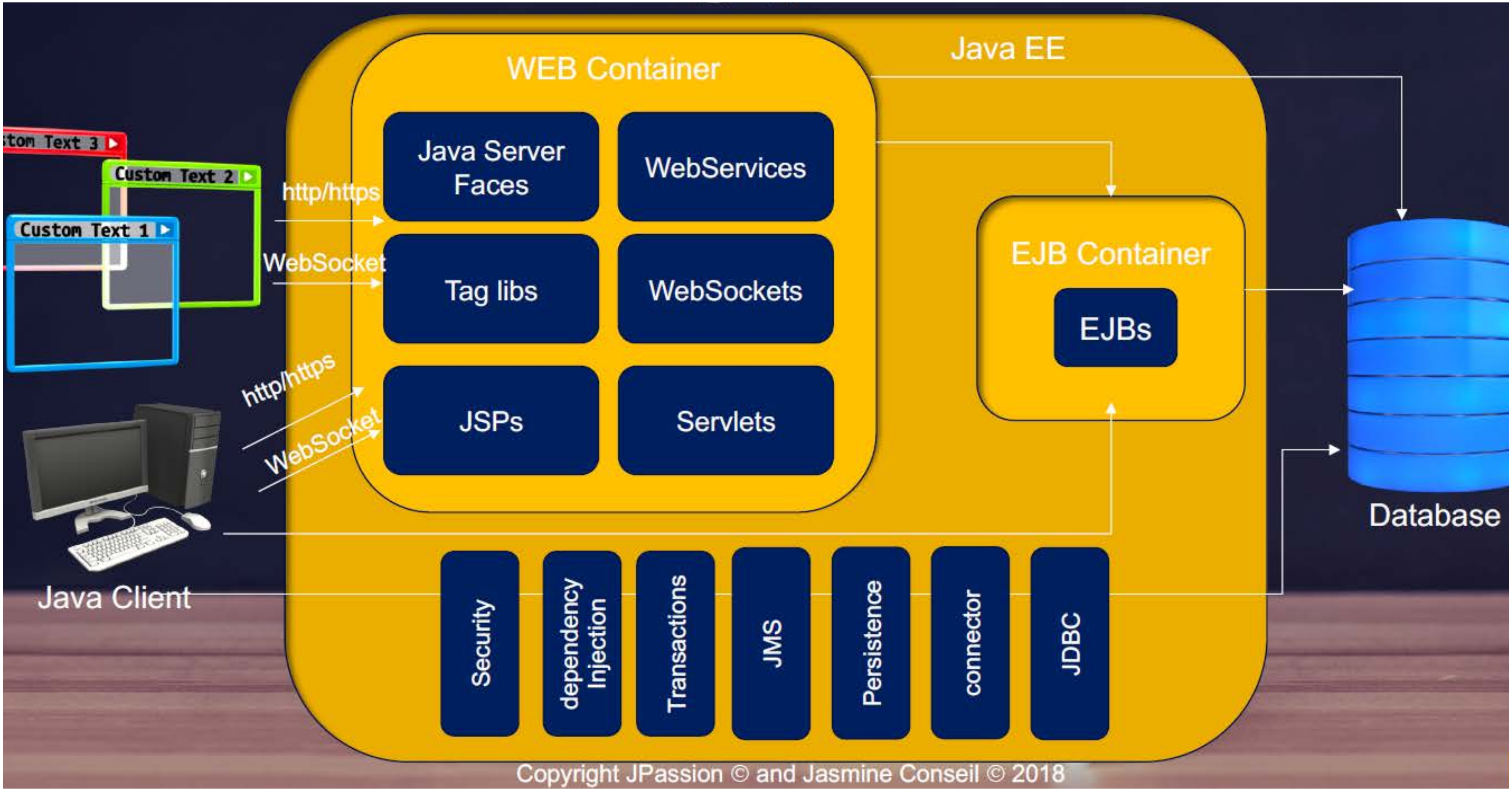
Some Important Java EE specifications

- Web Specifications
  - Servlet- defines how you can manage HTTP requests either in a synchronous or asynchronous way
  - WebSocket- this API provides a set of APIs to facilitate WebSocket connections
  - Java Server Faces- helps in building GUIs out of components.
  - (Unified) Expression Language- a simple language which was designed to facilitate web application developers.
- Web Service Specifications
  - Java API for RESTful Web Services
  - Java API for JSON Processing
  - Java API for JSON Binding- for binding or parsing a JSON file into Java classes.
  - Java Architecture for XML Binding- for binding of xml into Java objects.
  - Java API for XML Web Services- SOAP is an xml based protocol to access web services over http. This API allows you to create SOAP web services.

Some Important Java EE specifications *(continued)*

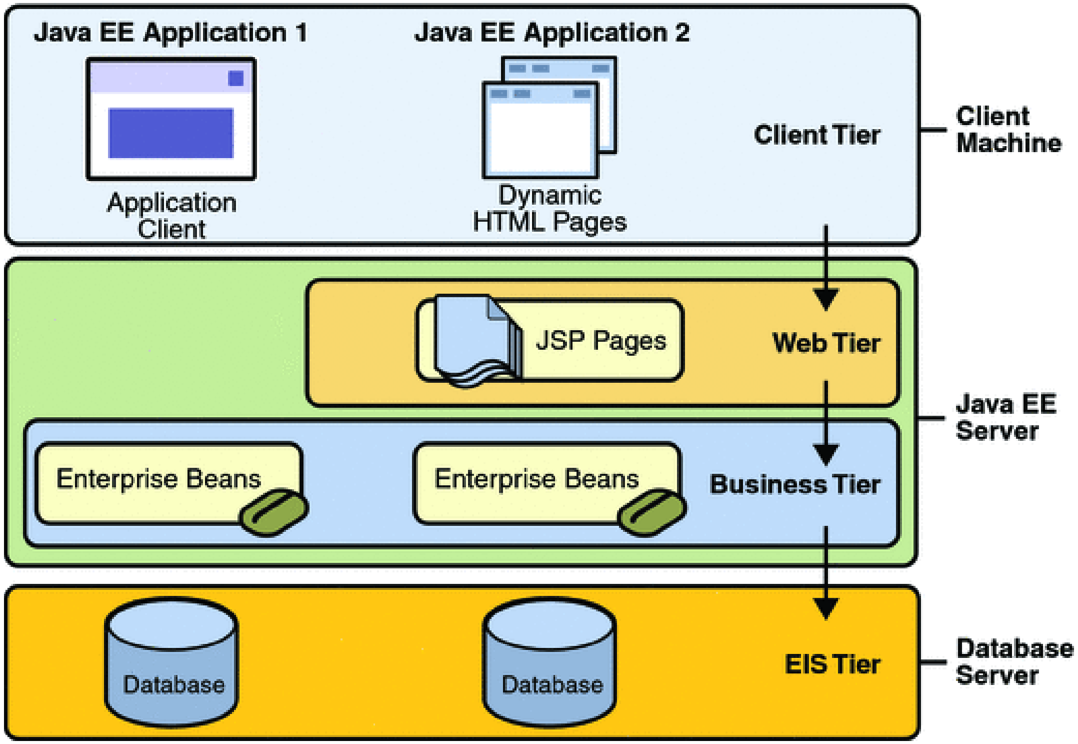
- Enterprise Specifications
  - Contexts and Dependency Injection
  - Enterprise JavaBean- APIs provided by object containers to provide transactions, remote procedure calls, and concurrency control.
  - Java Persistence API- object-relational mapping between relational database tables and Java classes.
  - Java Transaction API- It contains the interfaces and annotations to establish interaction between transaction support offered by Java EE
  - Java Message Service- It provides a common way to Java program to create, send and read enterprise messaging system's messages.
- Other Specifications of Java EE
  - Validation
  - Batch applications- It provides the means to run long running background tasks which involve a large volume of data and which need to be periodically executed.
  - Java EE Connector Architecture- This is a Java-based technological solution for connecting Java servers to Enterprise Information Systems

Overview


Java EE
WEB Container
Java Server Faces
WebServices
Tag libs
WebSockets
JSPs
Servlets
EJB Container
EJBs
Security
dependency Injection
Transactions
JMS
Persistence
connector
JDBC
tom Text 3
Custom Text 2
Custom Text 1
http/https
WebSocket
http/https
WebSocket
Java Client
Database
Copyright JPassion © and Jasmine Conseil © 2018

Java EE Server and Containers

- Java EE server: The runtime portion of a Java EE product. A Java EE server provides EJB and web containers.
- EJB container: Manages the execution of enterprise beans for Java EE applications. Enterprise beans and their container run on the Java EE server.
- Web container: Manages the execution of web pages, servlets, and some EJB components for Java EE applications. Web components and their container run on the Java EE server.
- Application client container: Manages the execution of application client components. Application clients and their container run on the client.



Application Servers

- Application Server – hosts your JEE application, running the components you develop within containers, which provide the interfaces to manage them, provide security, transaction management,…
- Reference implementation of Java EE server, Glassfish, also handed over to Eclipse
- Number of open source and commercial servers (certified Java EE *compliant*) which you can deploy Java EE applications on, e.g.
  - Glassfish, Liberty (IBM), WildFly (Red Hat), Jboss (Red Hat), WebLogic (Oracle), JEUS (TmaxSoft), InforSuite (Shandong), Tomee (Apache), Payara (Payara), Interstage (Fujitsu), NetWeaver (SAP)

Which application server?

- There are lot of commercial and open source application servers for Java EE
- Each will have their versions, supporting different versions of Java Enterprise spec, and sometimes not 100% of it
  - For example Apace Tomcat supports servlets and JSP (basic Java web components), whereas TomEE also supports back-end business logic components (enterprise java beans) and related specs

main Java EE Components

- Web applications: applications executed in a web container and respond to HTTP requests from web clients.
  - Made of servlets, servlet filters, JSP pages, and JSF (Java Server Faces).
  - Servlets also support web services (endpoints)
- Enterprise Java Beans: container-managed components for processing transactional business logic. They can be accessed locally and remotely through RMI (or HTTP for SOAP and RESTful web services).

Java EE Application

- In a Java EE application:
  - The model -- business layer functionality represented by JavaBeans or EJBs
  - The view -- the presentation layer functionality represented by JSPs or JSFs in a web application
  - The controller -- Servlet mediating between model and view
- Must accommodate input from various clients including HTTP requests from web clients

EJB

- Server-side components
- Encapsulate business logic
- Take care of transactions and security
- Used in building business layers to sit on top of the persistence layer and as an entry point for presentation-tier technologies such as JSP, JSF
- Can be built by annotating a POJO that will be deployed into a container

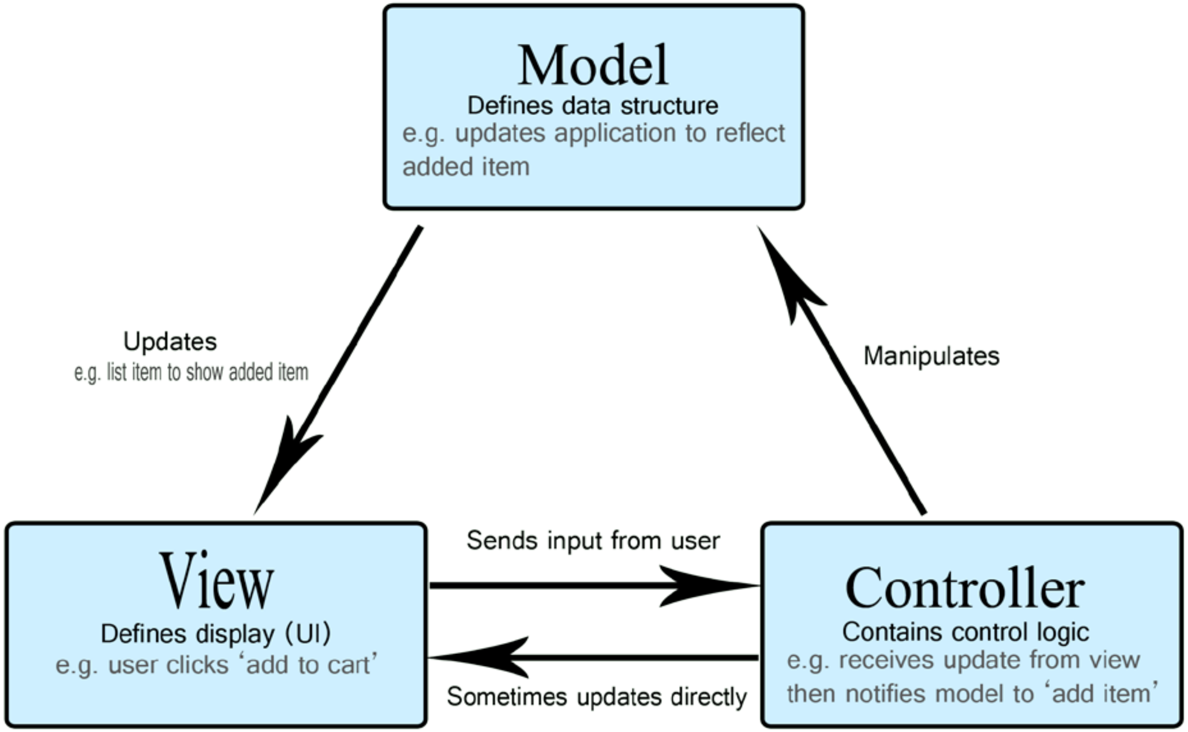
Type of EJBs

- Session beans and Message-driven Beans (MDBs)
- Session Beans are used to encapsulate high-level business logic and can be
  - Stateful: the state of the bean is maintained across multiple method calls. The "state" refers to the values of its instance variables. Because the client interacts with the bean, this state is often called the conversational state. Stateful session bean contains conversational state, which must be retained across method invocations for a single user
  - Stateless: contains no conversational state between invocations, and any instance can be used for any client
  - Singleton: A single session bean is shared between clients and supports concurrent access

Packaging Java EE Web Application

- A web application module contains:
  - servlets, JSPs, JSF pages, ejbs, and web services,
  - as well as HTML and XHTML pages, Cascading Style Sheets(CSS), JavaScripts, images, videos, and so on.
- All these artifacts are packaged in a jar file with a .war extension -- i.e., a war file, or Web Archive including
  - web deployment descriptors such as WEB-INF/web.xml
  - class files in WEB-INF/classes
  - any dependent jar files in WEB-INF/lib

MVC Architecture



Model

Defines data structure

e.g. updates application to reflect added item

Updates

e.g. list item to show added item

Manipulates

View

Defines display (UI)

e.g. user clicks 'add to cart'

Sends input from user

Sometimes updates directly

Controller

Contains control logic

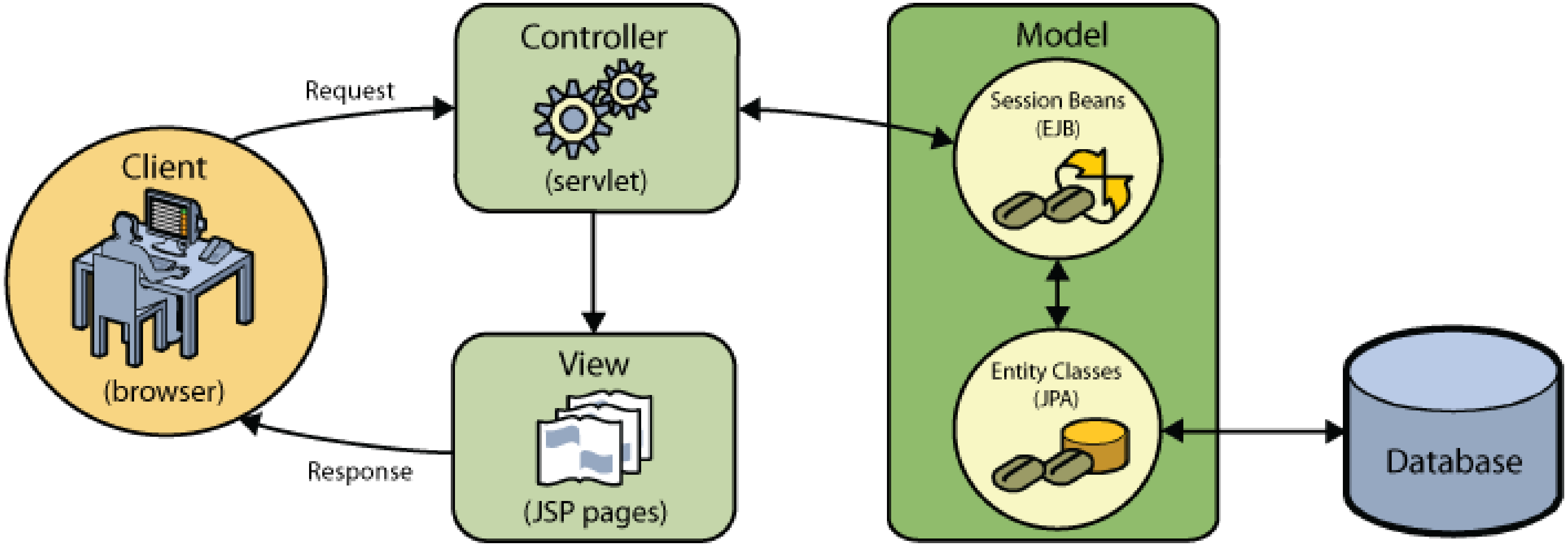
e.g. receives update from view then notifies model to 'add item'

MVC Architecture

- Model:
  - Represents the business data and any business logic that governs access to and modification of the data. The model notifies views when it changes and lets the view query the model about its state. It also lets the controller access application functionality encapsulated by the model.
- View:
  - The view renders the contents of a model. It gets data from the model and specifies how that data should be presented. It updates data presentation when the model changes. A view also forwards user input to a controller.
- Controller:
  - The controller defines application behavior. It dispatches user requests and selects views for presentation. It interprets user inputs and maps them into actions to be performed by the model. In a web application, user inputs are HTTP GET and POST requests. A controller selects the next view to display based on the user interactions and the outcome of the model operations.

MVC- typical example

- Using the controller servlet to act as router / dispatcher for incoming HTTP requests
- Session Bean acts as a façade hiding the complexity of the JPA interface, while the JPA API does the object-relational mapping for us
- Java Server Pages for example display object data retrieved via session bean

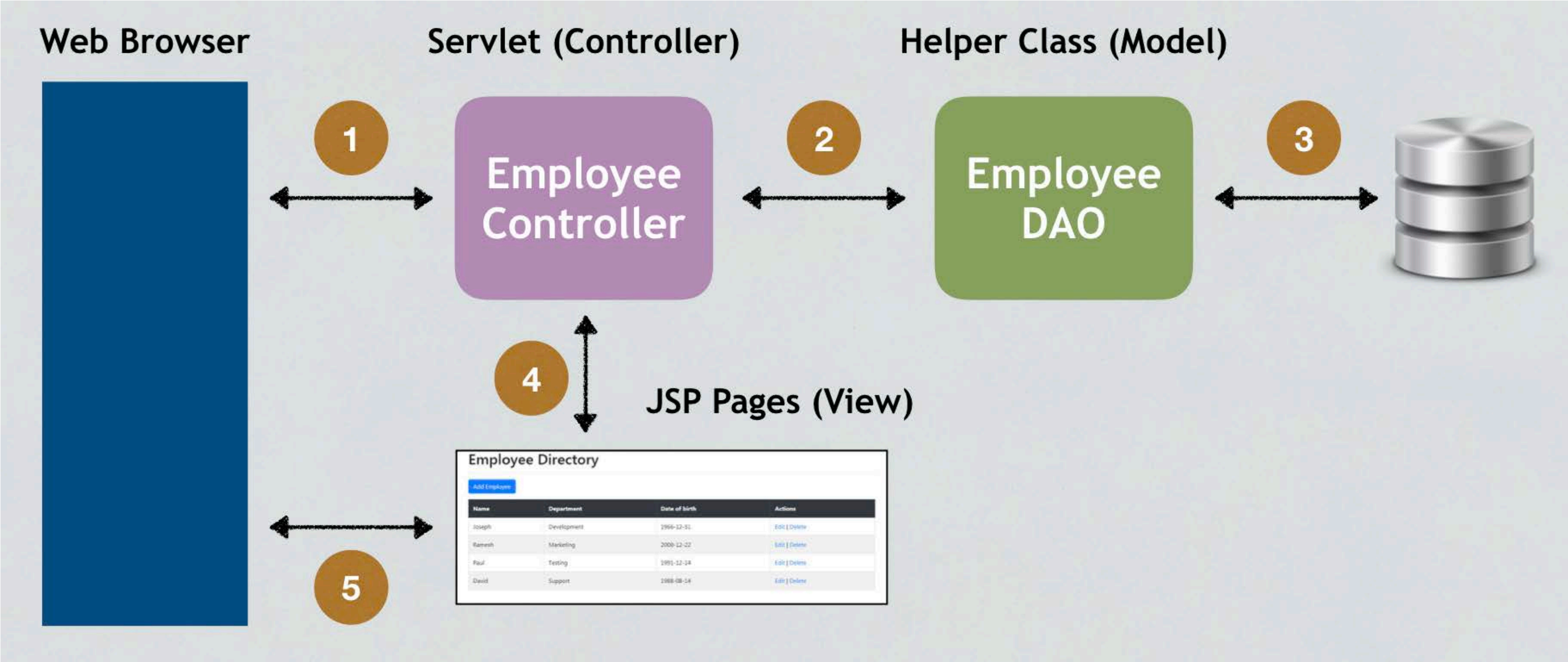


Web Application

- In Java EE platform, web components provide the dynamic extension capabilities for a web server.
  - Web components are either Java servlets, web pages, web service endpoints, JSP pages, or JSFs
- A lot of web servers (e.g. Tomcat) support the Java EE web specification, and there are plenty of hosting services which support them

Can start small

- Pretty easy to build a robust web application just using servlets, Java Server Pages and SQL
  - E.g. https://bushansirgur.in/creating-mvc-database-web-application-in-jsp-and-servlets-create-read-update-delete/



Name	Department	Date of birth	Actions
Joseph	Development	1996-12-31	Edit | Delete
Ramesh	Marketing	2000-12-22	Edit | Delete
Paul	Testing	1991-12-14	Edit | Delete
David	Support	1988-08-14	Edit | Delete

The View layer in Web Applications

- Display information according to client types
- Display result of business logic (Model)
- Not concerned with how the information was obtained, or from where (since that is the responsibility of Model)

Model layer in Web Application

- Models the data and behavior behind the business process
  - What it’s responsible for:
    - Performing DB queries
    - Calculating the business process
    - Processing orders
- Encapsulation of data and behavior which are independent of presentation

Controller in Web Application

- Serves as the logical connection between the user's interaction and the business services on the back end servers
- Responsible for making decisions among multiple presentations
  - e.g. User's language, locale or access level dictates a different presentation.
- A request enters the application through the control layer, which will decide how the request should be handled and what information should be returned

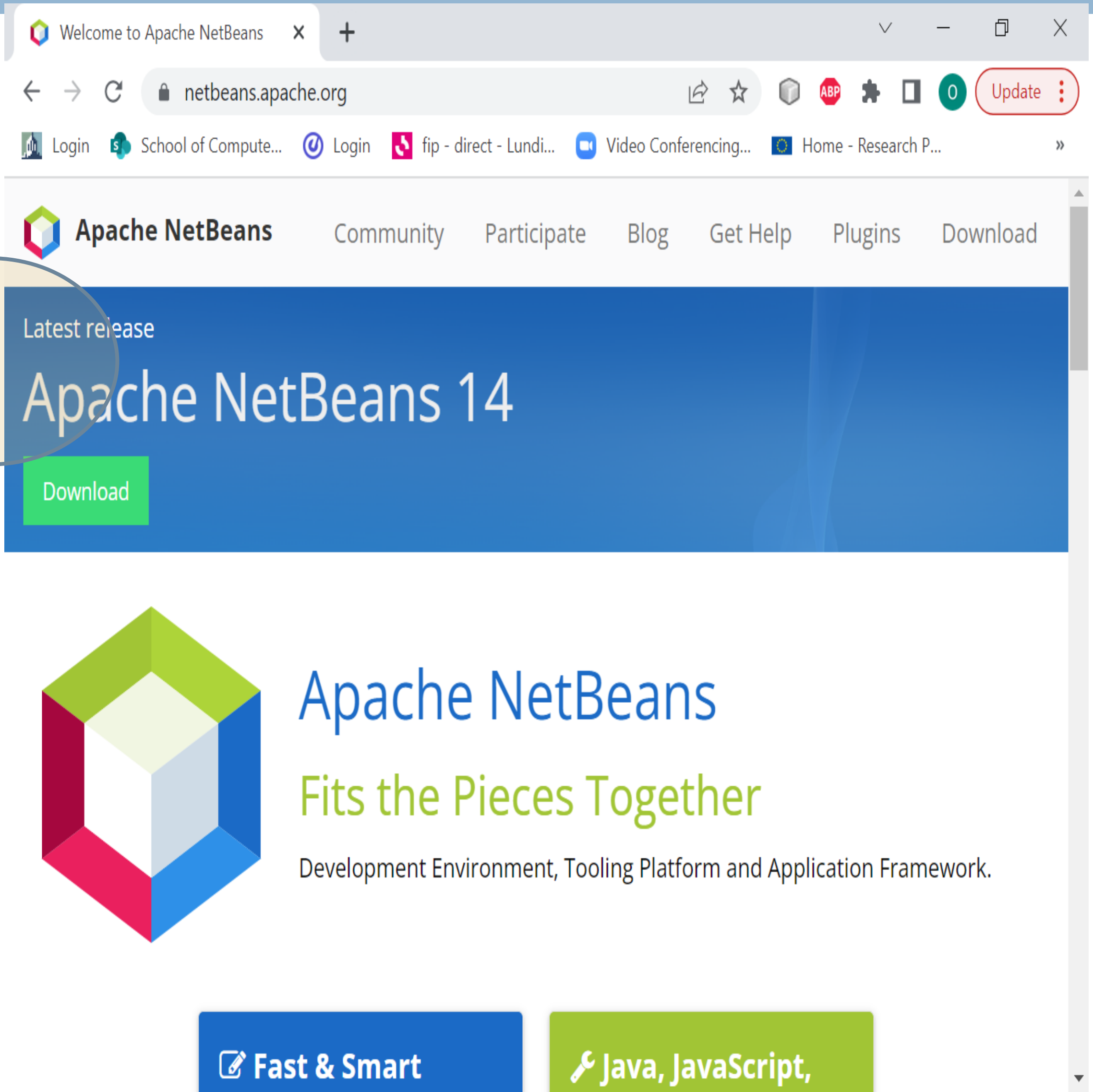
Getting Started

Download and install Apache NetBeans

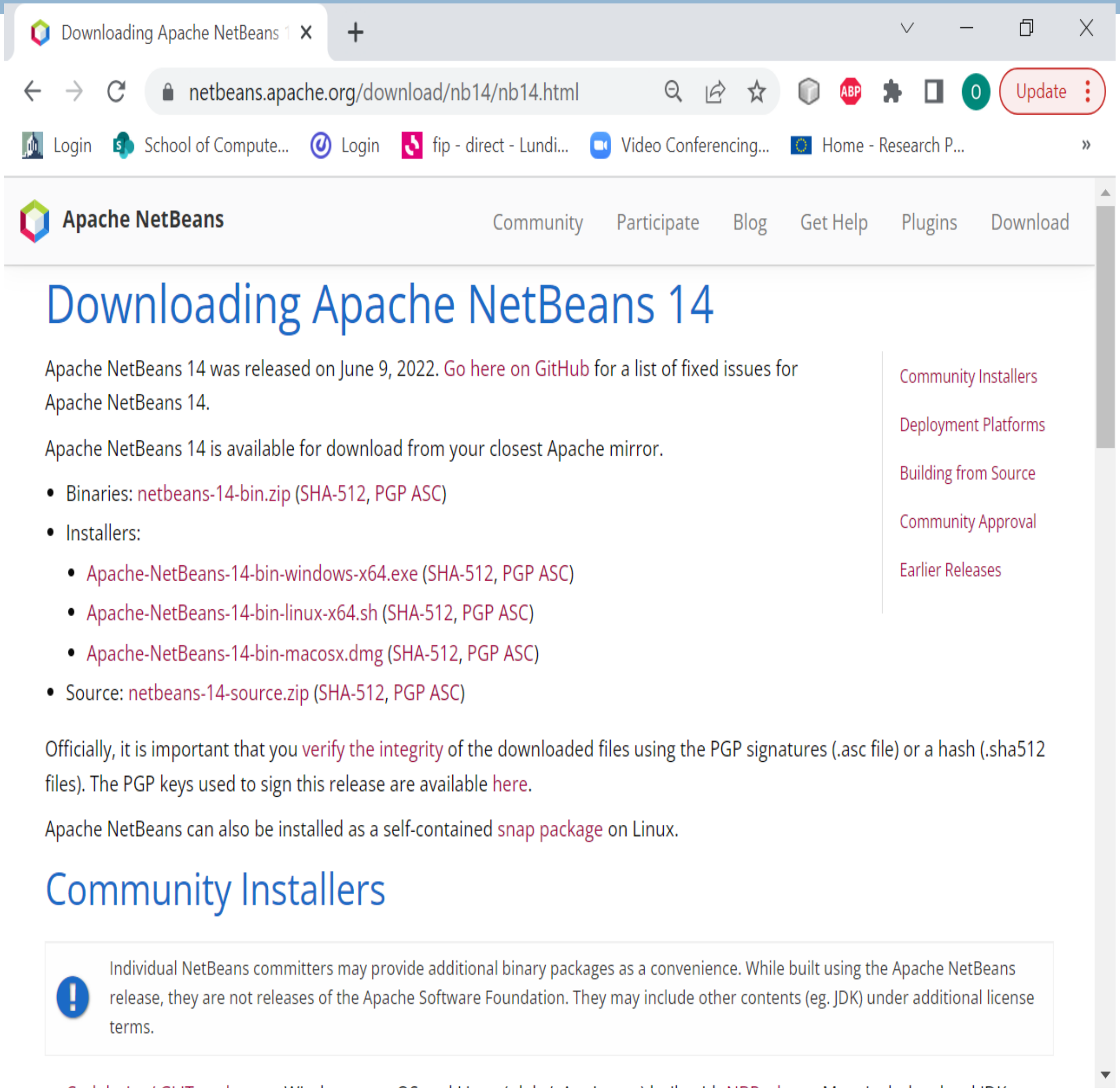
Install Application Server (Payara)

Create, Deploy and run test application

Download NetBeans



Choose relevant installer



Downloading Apache NetBeans 14

Apache NetBeans 14 was released on June 9, 2022. Go here on GitHub for a list of fixed issues for Apache NetBeans 14.

Apache NetBeans 14 is available for download from your closest Apache mirror.

- Binaries: netbeans-14-bin.zip (SHA-512, PGP ASC)
- Installers:
  - Apache-NetBeans-14-bin-windows-x64.exe (SHA-512, PGP ASC)
  - Apache-NetBeans-14-bin-linux-x64.sh (SHA-512, PGP ASC)
  - Apache-NetBeans-14-bin-macosx.dmg (SHA-512, PGP ASC)
- Source: netbeans-14-source.zip (SHA-512, PGP ASC)

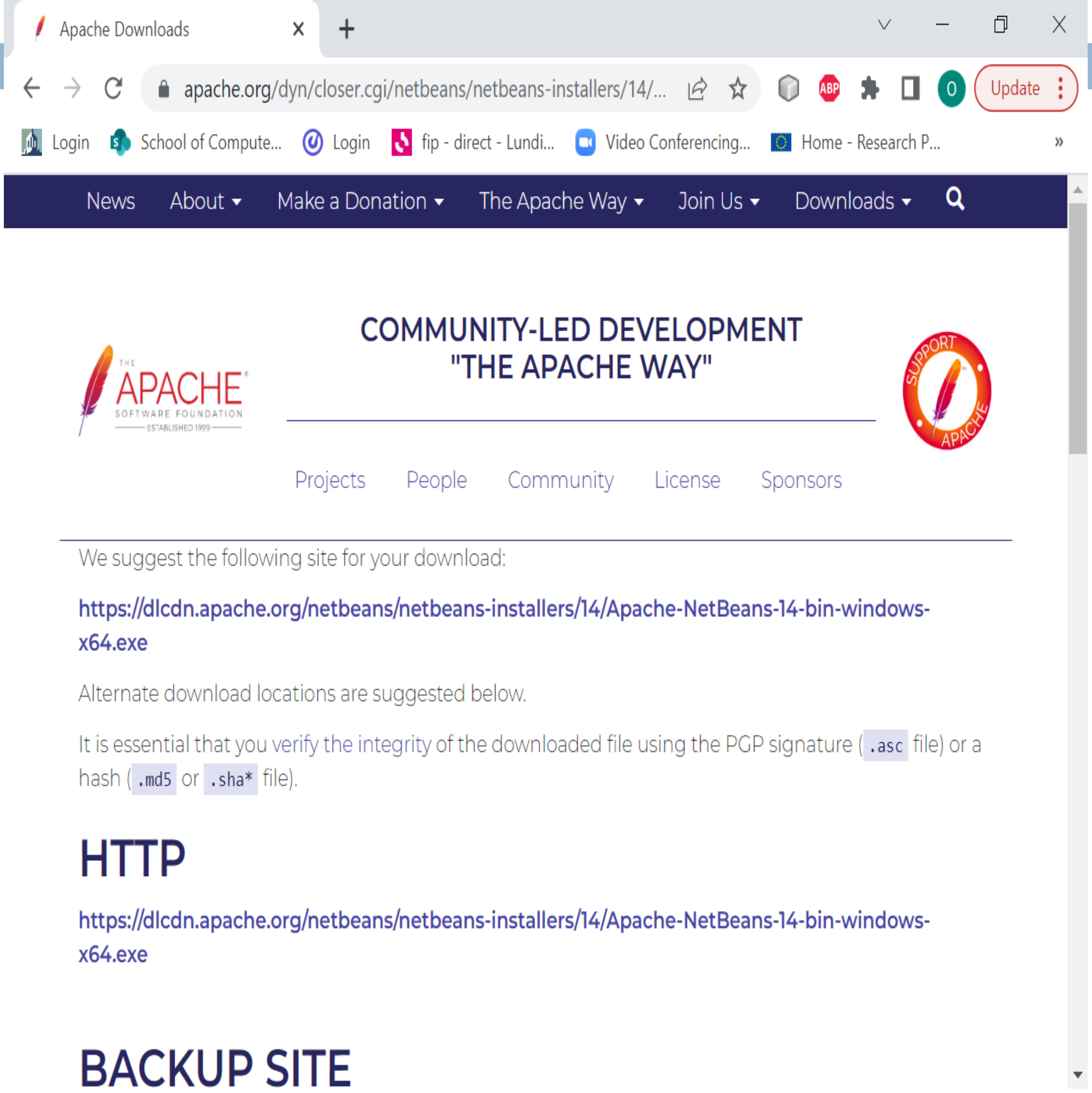
Officially, it is important that you verify the integrity of the downloaded files using the PGP signatures (.asc file) or a hash (.sha512 files). The PGP keys used to sign this release are available here.

Apache NetBeans can also be installed as a self-contained snap package on Linux.

Community Installers

Individual NetBeans committers may provide additional binary packages as a convenience. While built using the Apache NetBeans release, they are not releases of the Apache Software Foundation. They may include other contents (eg. JDK) under additional license terms.

Choose relevant installer



COMMUNITY-LED DEVELOPMENT
"THE APACHE WAY"

Projects People Community License Sponsors

We suggest the following site for your download:

https://dlcdn.apache.org/netbeans/netbeans-installers/14/Apache-NetBeans-14-bin-windows-x64.exe

Alternate download locations are suggested below.

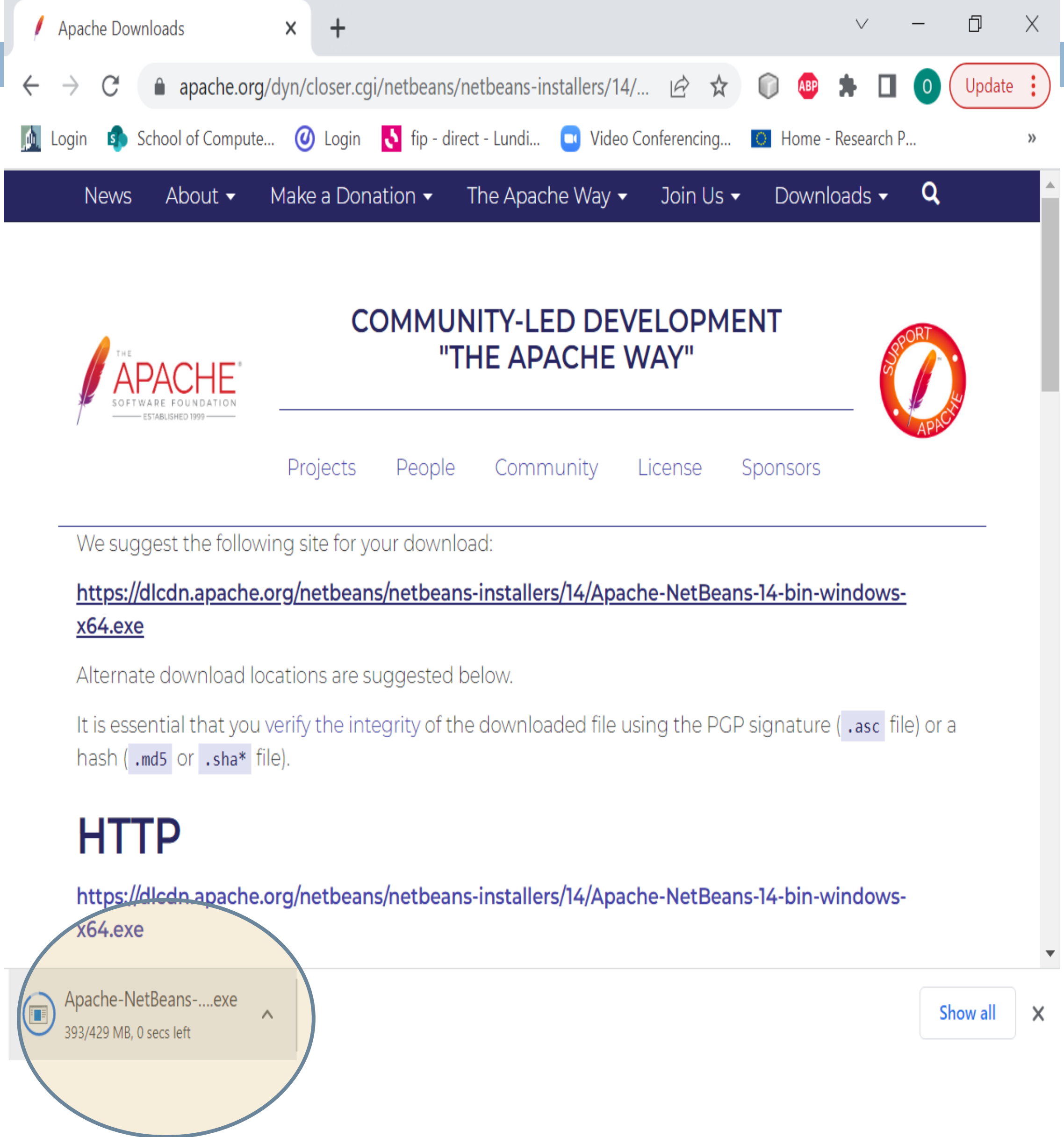
It is essential that you verify the integrity of the downloaded file using the PGP signature (`.asc` file) or a hash (`.md5` or `.sha*` file).

HTTP

https://dlcdn.apache.org/netbeans/netbeans-installers/14/Apache-NetBeans-14-bin-windows-x64.exe

BACKUP SITE

Run the installer



COMMUNITY-LED DEVELOPMENT "THE APACHE WAY"

We suggest the following site for your download:

https://dlcdn.apache.org/netbeans/netbeans-installers/14/Apache-NetBeans-14-bin-windows-x64.exe

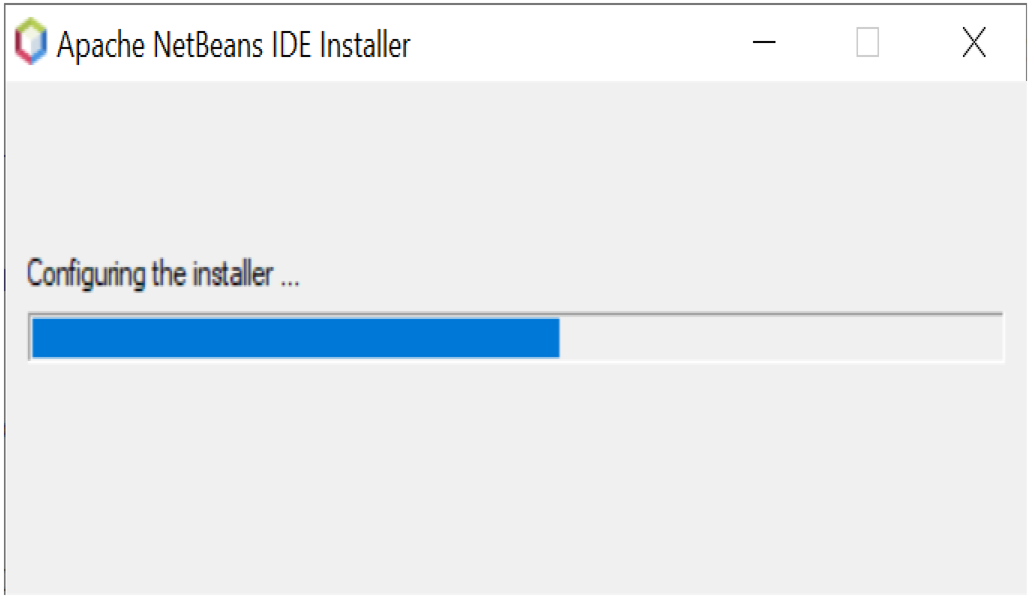
Alternate download locations are suggested below.

It is essential that you verify the integrity of the downloaded file using the PGP signature (`.asc` file) or a hash (`.md5` or `.sha*` file).

HTTP

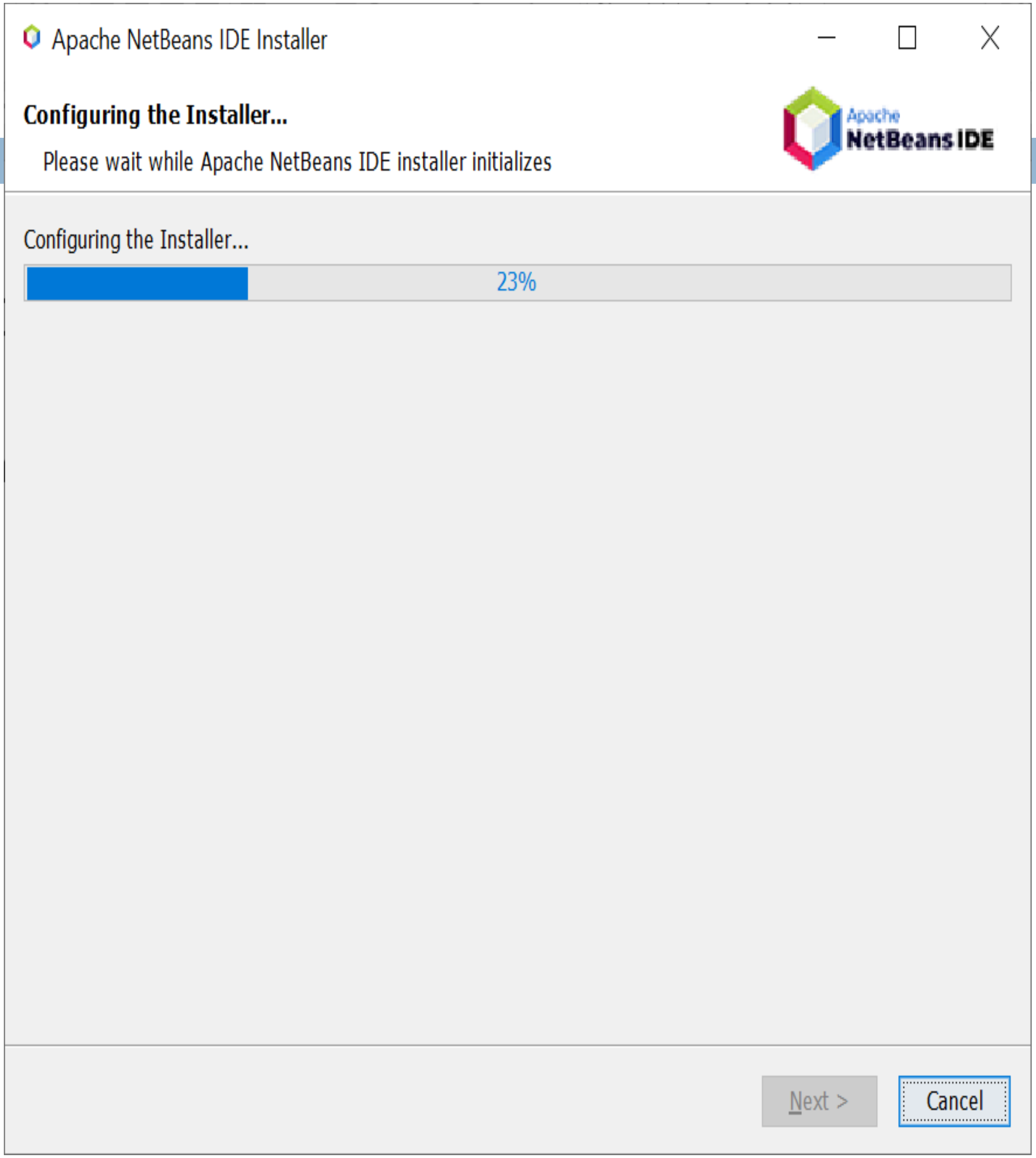
https://dlcdn.apache.org/netbeans/netbeans-installers/14/Apache-NetBeans-14-bin-windows-x64.exe

Running the installer



Apache NetBeans IDE Installer

Configuring the installer ...



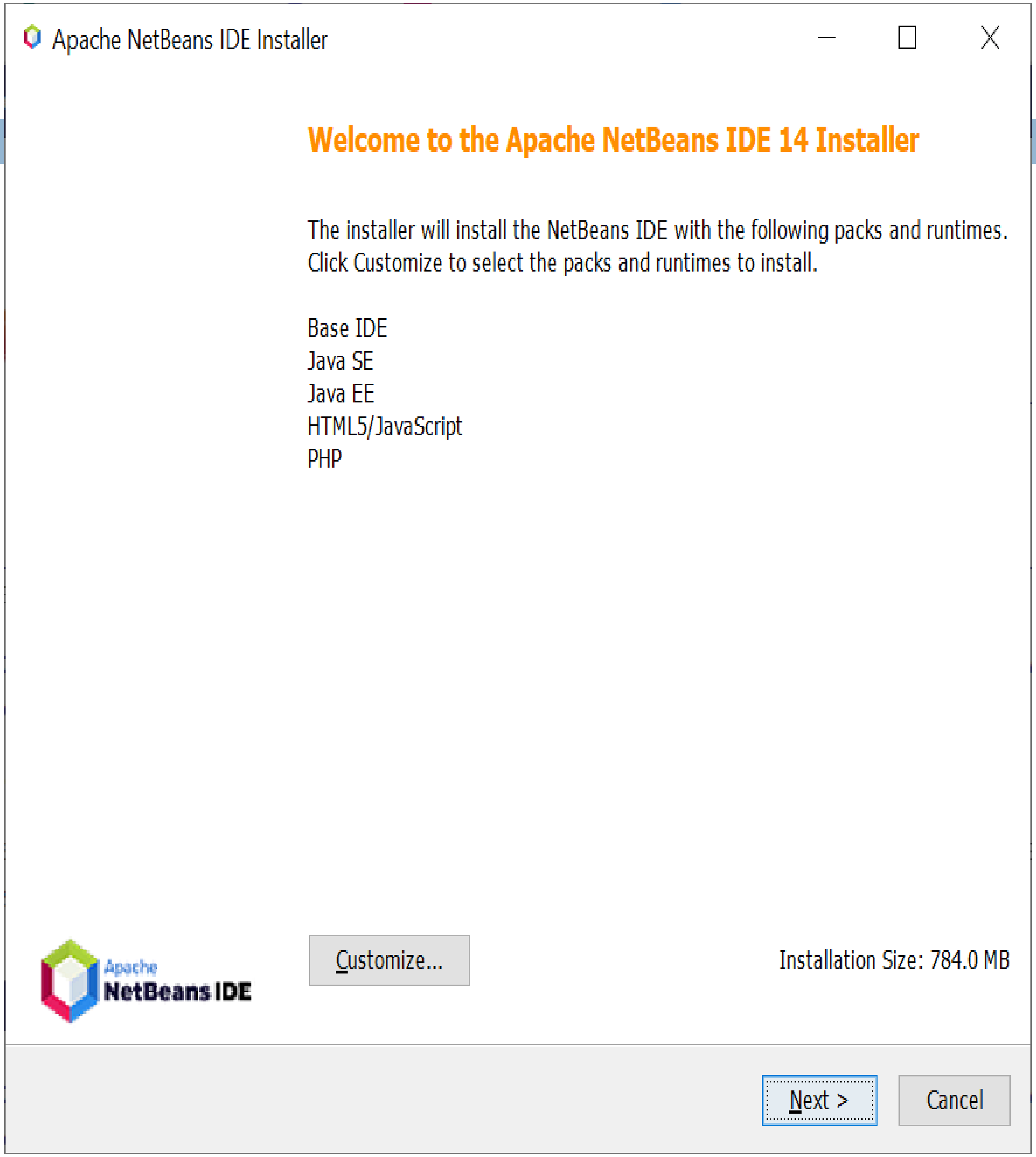
Apache NetBeans IDE Installer

Configuring the Installer...

Please wait while Apache NetBeans IDE installer initializes

Configuring the Installer...

23%

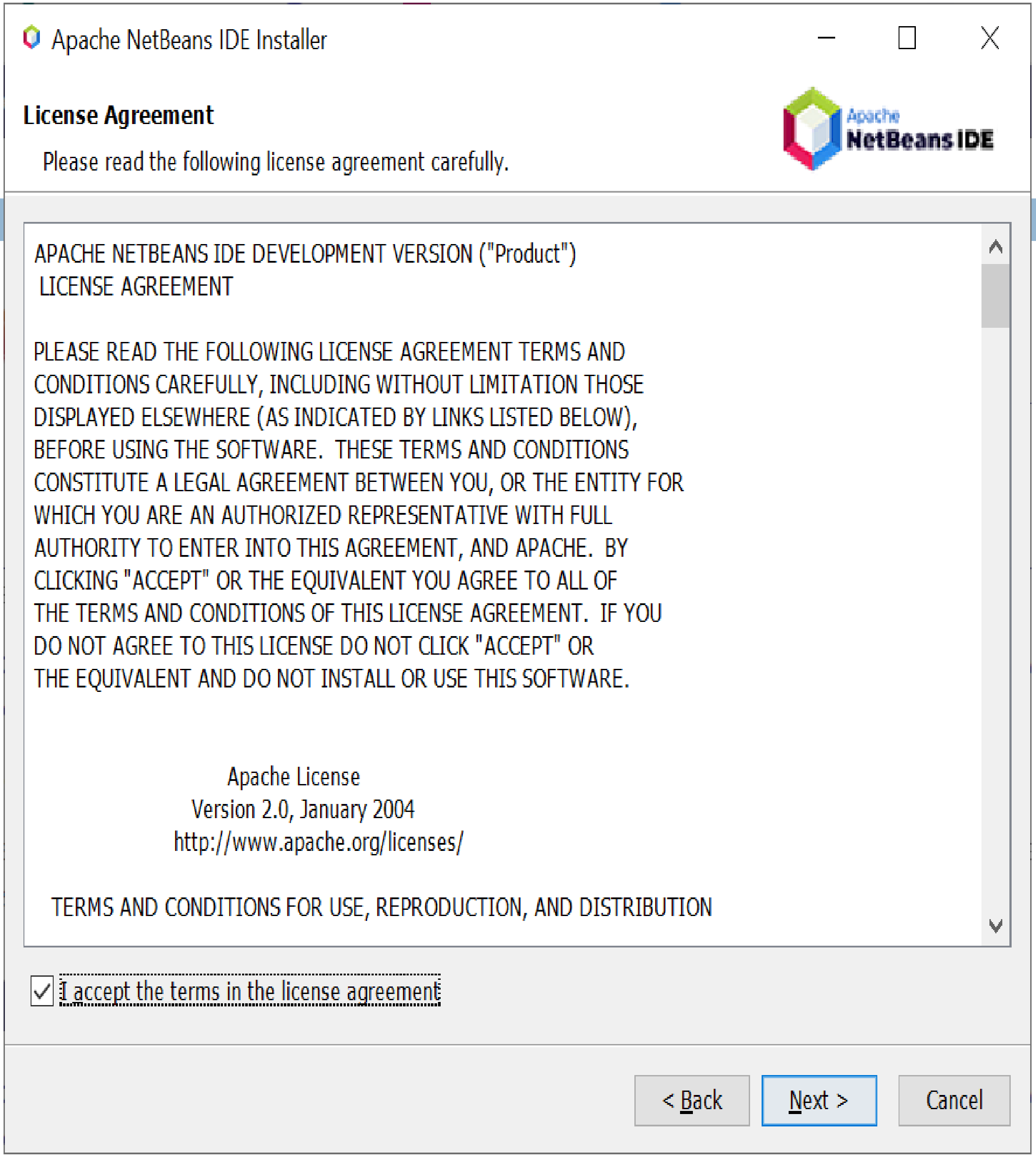


Welcome to the Apache NetBeans IDE 14 Installer

The installer will install the NetBeans IDE with the following packs and runtimes. Click Customize to select the packs and runtimes to install.

Base IDE
Java SE
Java EE
HTML5/JavaScript
PHP

Installation Size: 784.0 MB



License Agreement

Please read the following license agreement carefully.

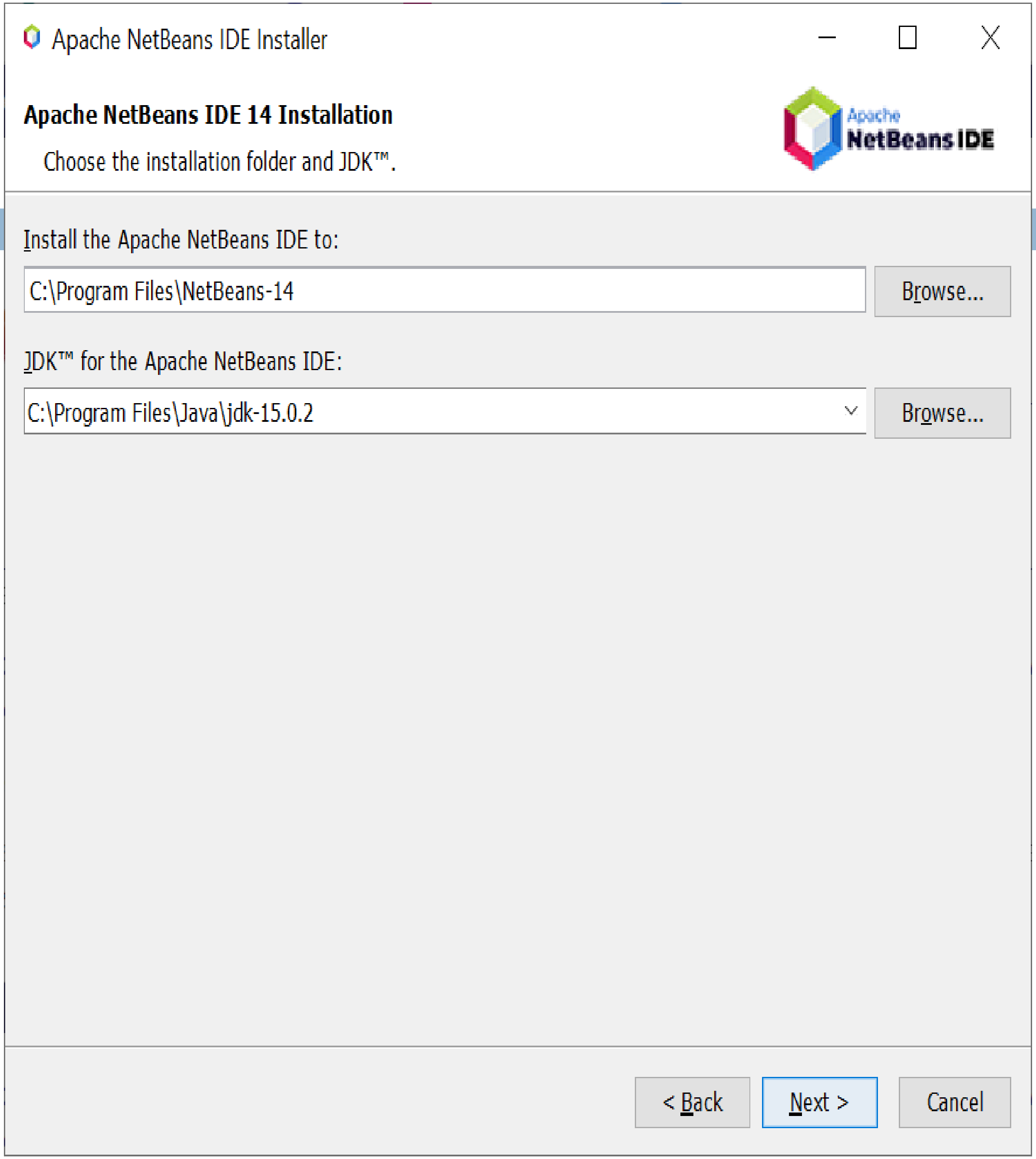
APACHE NETBEANS IDE DEVELOPMENT VERSION ("Product")
LICENSE AGREEMENT

PLEASE READ THE FOLLOWING LICENSE AGREEMENT TERMS AND CONDITIONS CAREFULLY, INCLUDING WITHOUT LIMITATION THOSE DISPLAYED ELSEWHERE (AS INDICATED BY LINKS LISTED BELOW), BEFORE USING THE SOFTWARE. THESE TERMS AND CONDITIONS CONSTITUTE A LEGAL AGREEMENT BETWEEN YOU, OR THE ENTITY FOR WHICH YOU ARE AN AUTHORIZED REPRESENTATIVE WITH FULL AUTHORITY TO ENTER INTO THIS AGREEMENT, AND APACHE. BY CLICKING "ACCEPT" OR THE EQUIVALENT YOU AGREE TO ALL OF THE TERMS AND CONDITIONS OF THIS LICENSE AGREEMENT. IF YOU DO NOT AGREE TO THIS LICENSE DO NOT CLICK "ACCEPT" OR THE EQUIVALENT AND DO NOT INSTALL OR USE THIS SOFTWARE.

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

☑ I accept the terms in the license agreement



Apache NetBeans IDE Installer

Apache NetBeans IDE 14 Installation

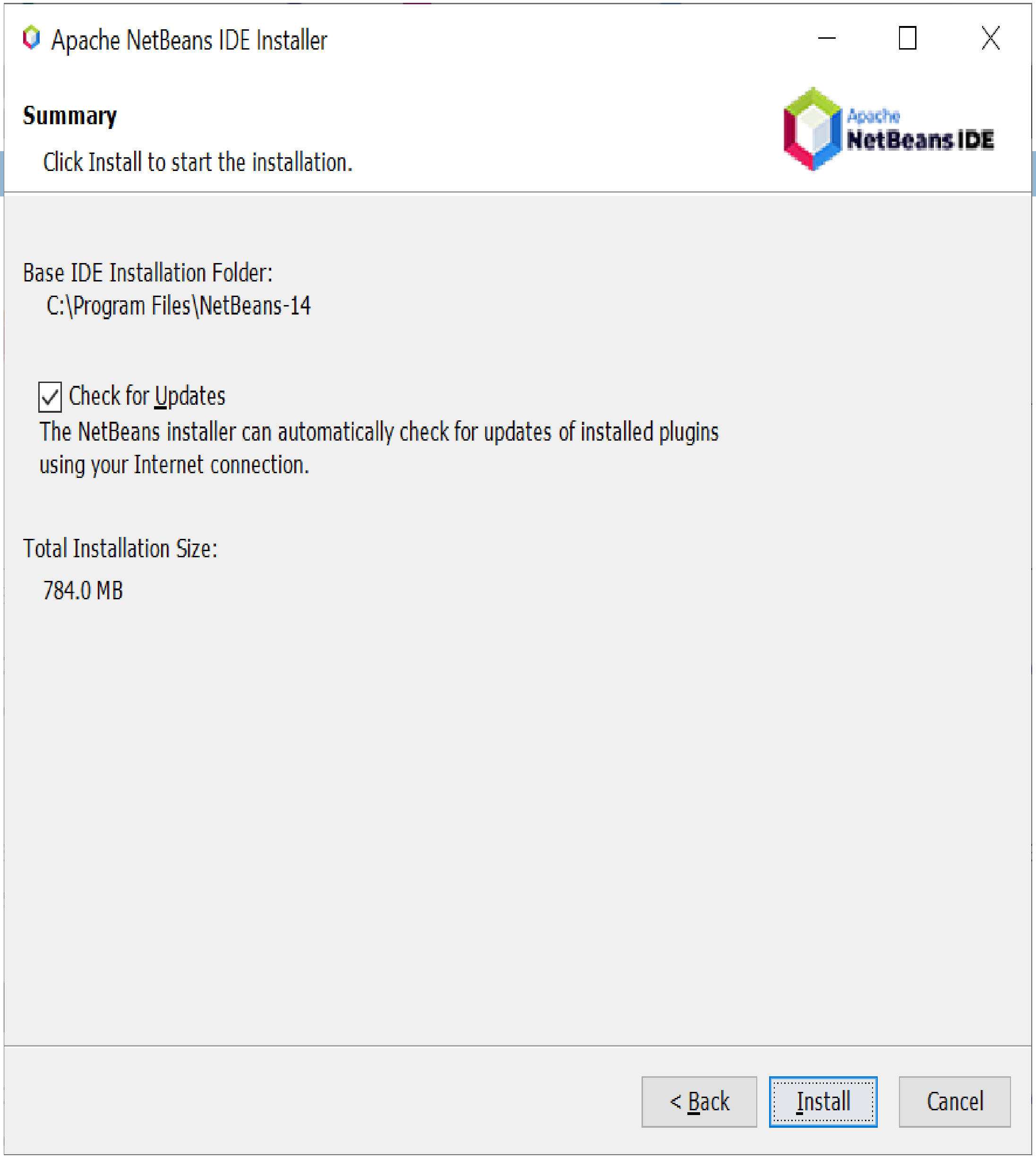
Choose the installation folder and JDK™.

Install the Apache NetBeans IDE to:

C:\Program Files\NetBeans-14

JDK™ for the Apache NetBeans IDE:

C:\Program Files\Java\jdk-15.0.2



Summary

Click Install to start the installation.

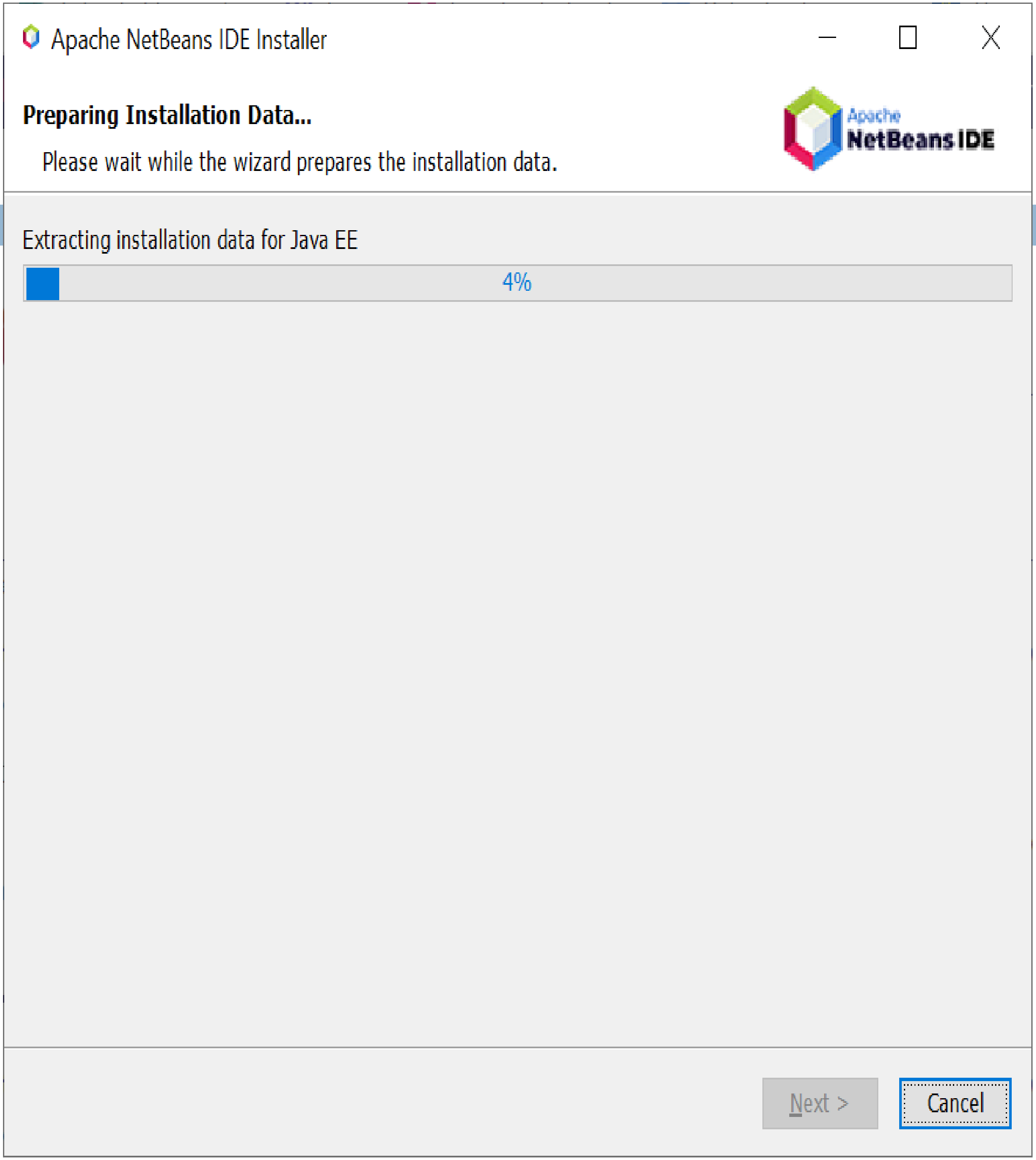
Base IDE Installation Folder:
C:\Program Files\NetBeans-14

☑ Check for Updates

The NetBeans installer can automatically check for updates of installed plugins using your Internet connection.

Total Installation Size:

784.0 MB



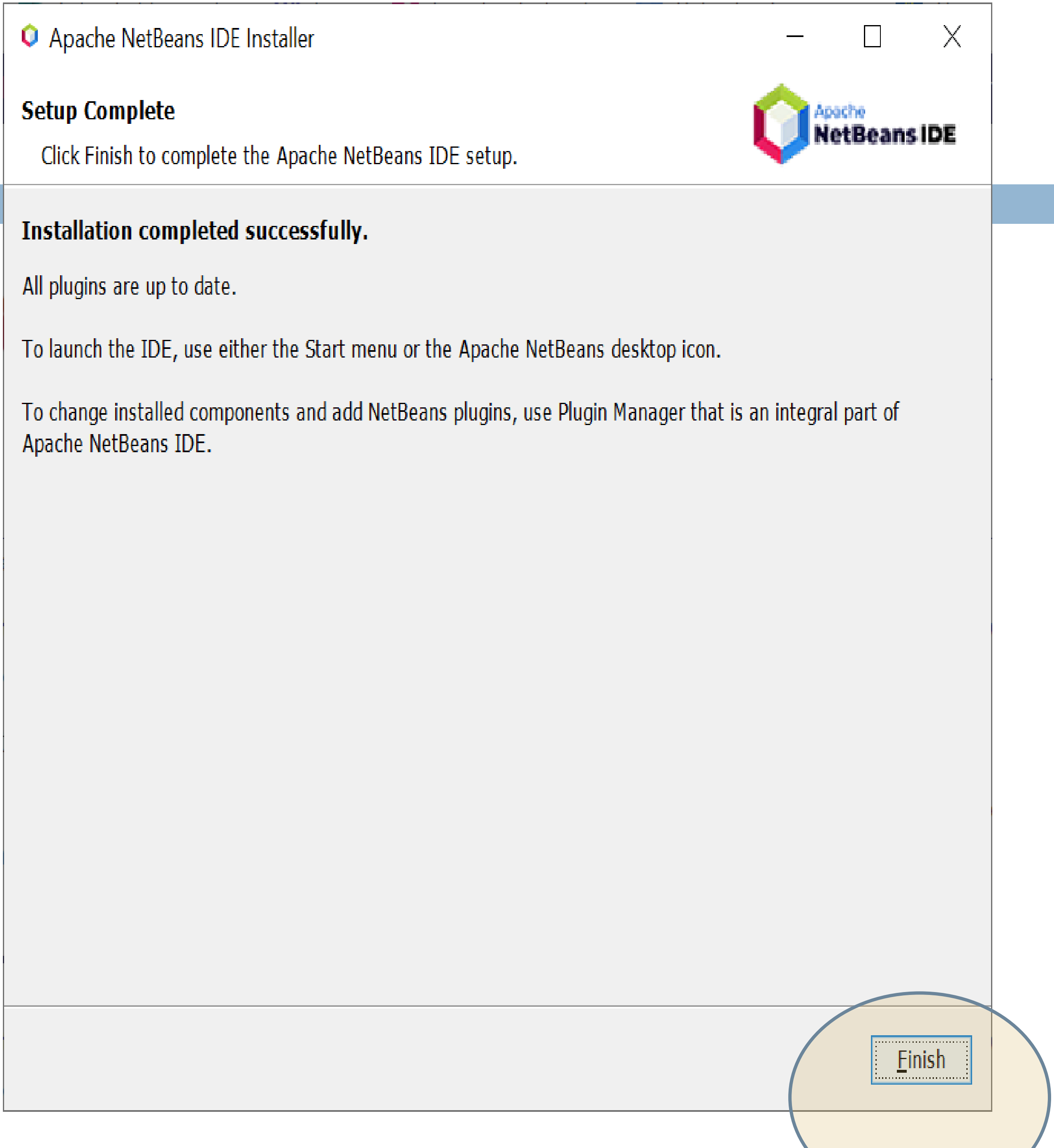
Apache NetBeans IDE Installer

Preparing Installation Data...

Please wait while the wizard prepares the installation data.

Extracting installation data for Java EE

4%



Setup Complete

Click Finish to complete the Apache NetBeans IDE setup.

Installation completed successfully.

All plugins are up to date.

To launch the IDE, use either the Start menu or the Apache NetBeans desktop icon.

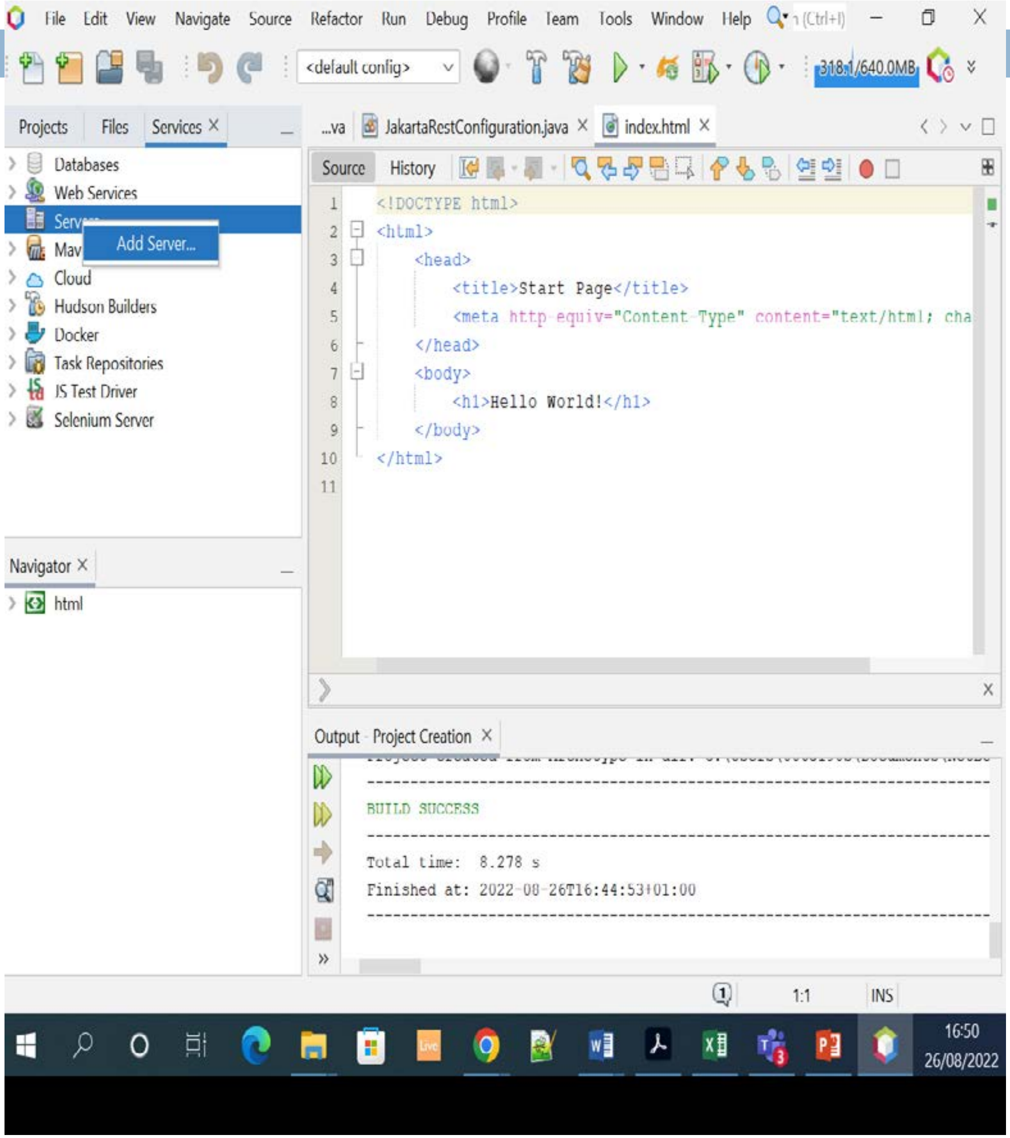
To change installed components and add NetBeans plugins, use Plugin Manager that is an integral part of Apache NetBeans IDE.

APACHE NetBeans

Apache NetBeans IDE 14

Done loading modules.

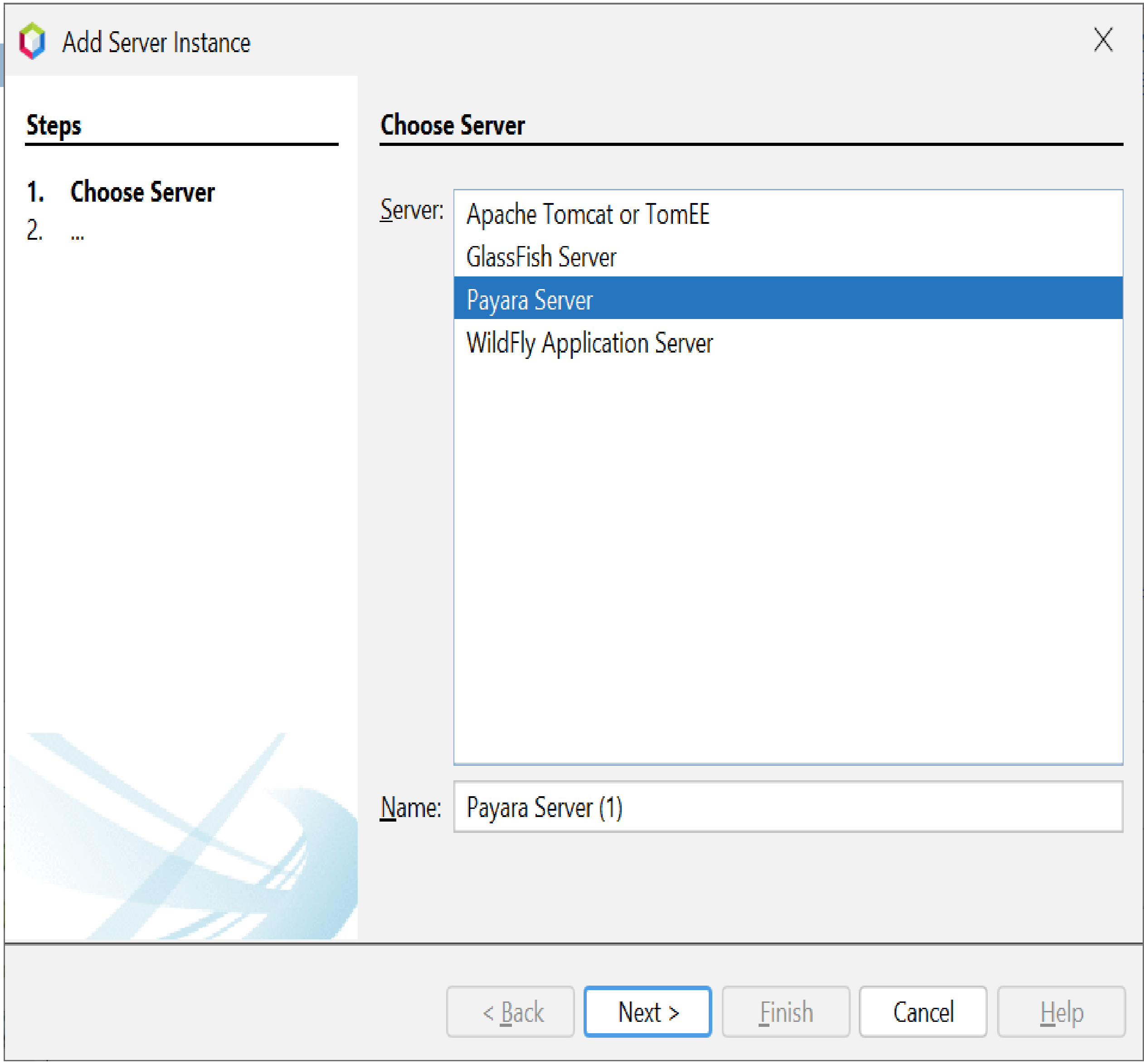
- Use ‘Services’ tab to add a new application server – this is where your application will be deployed



```
<!DOCTYPE html>
<html>
    <head>
        <title>Start Page</title>
        <meta http-equiv="Content-Type" content="text/html; cha
    </head>
    <body>
        <h1>Hello World!</h1>
    </body>
</html>
```

```
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  8.278 s
Finished at: 2022-08-26T16:44:53+01:00
------------------------------------------------------------------------
```

- Use the Payara server



Steps

1. Choose Server
2. ...

Choose Server

Server:

Apache Tomcat or TomEE

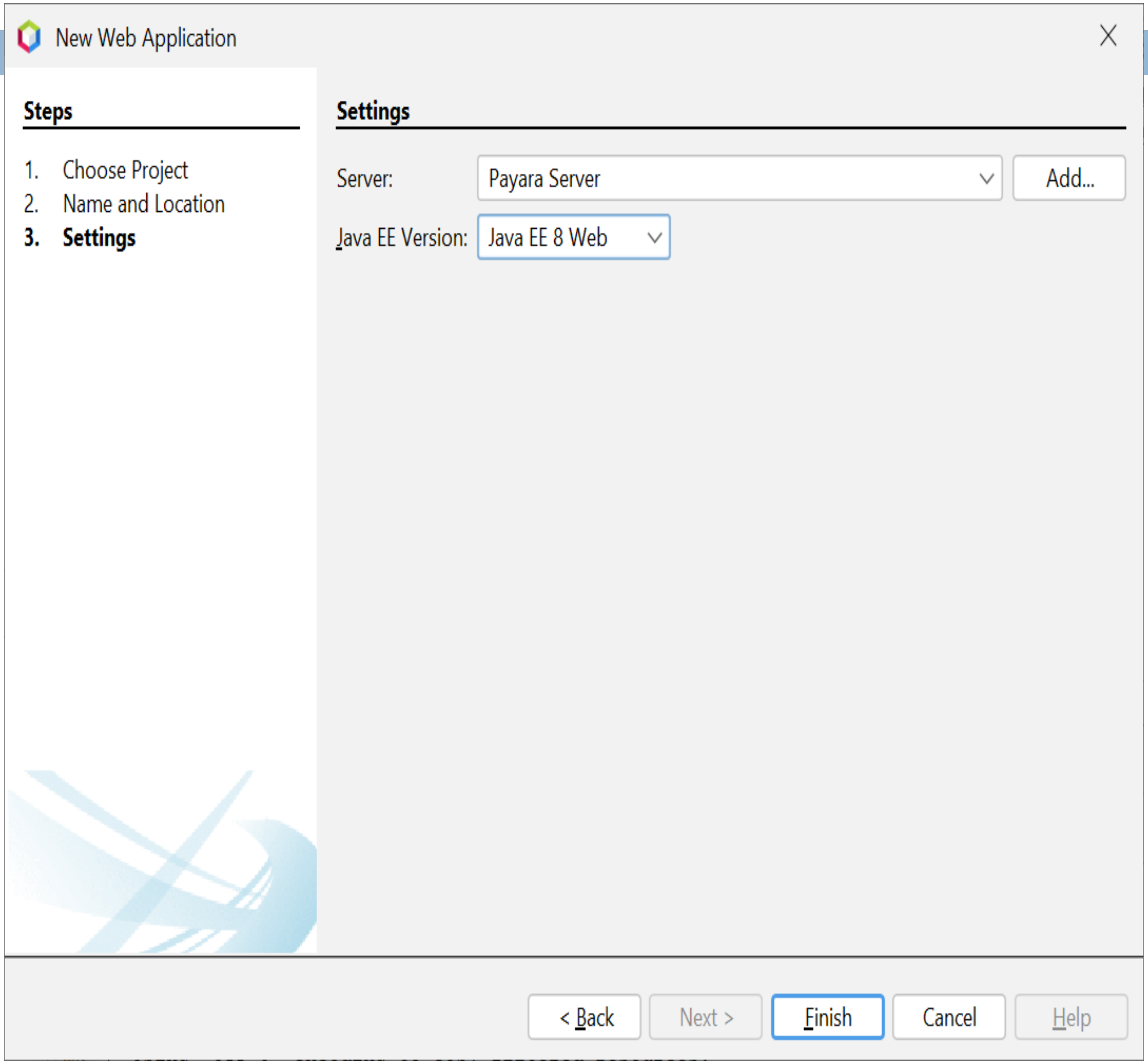
GlassFish Server

Payara Server

WildFly Application Server

Name: Payara Server (1)

- Java EE 8 version seems to work / be compatible with the JDK installed by the NetBeans installer



New Web Application

Steps

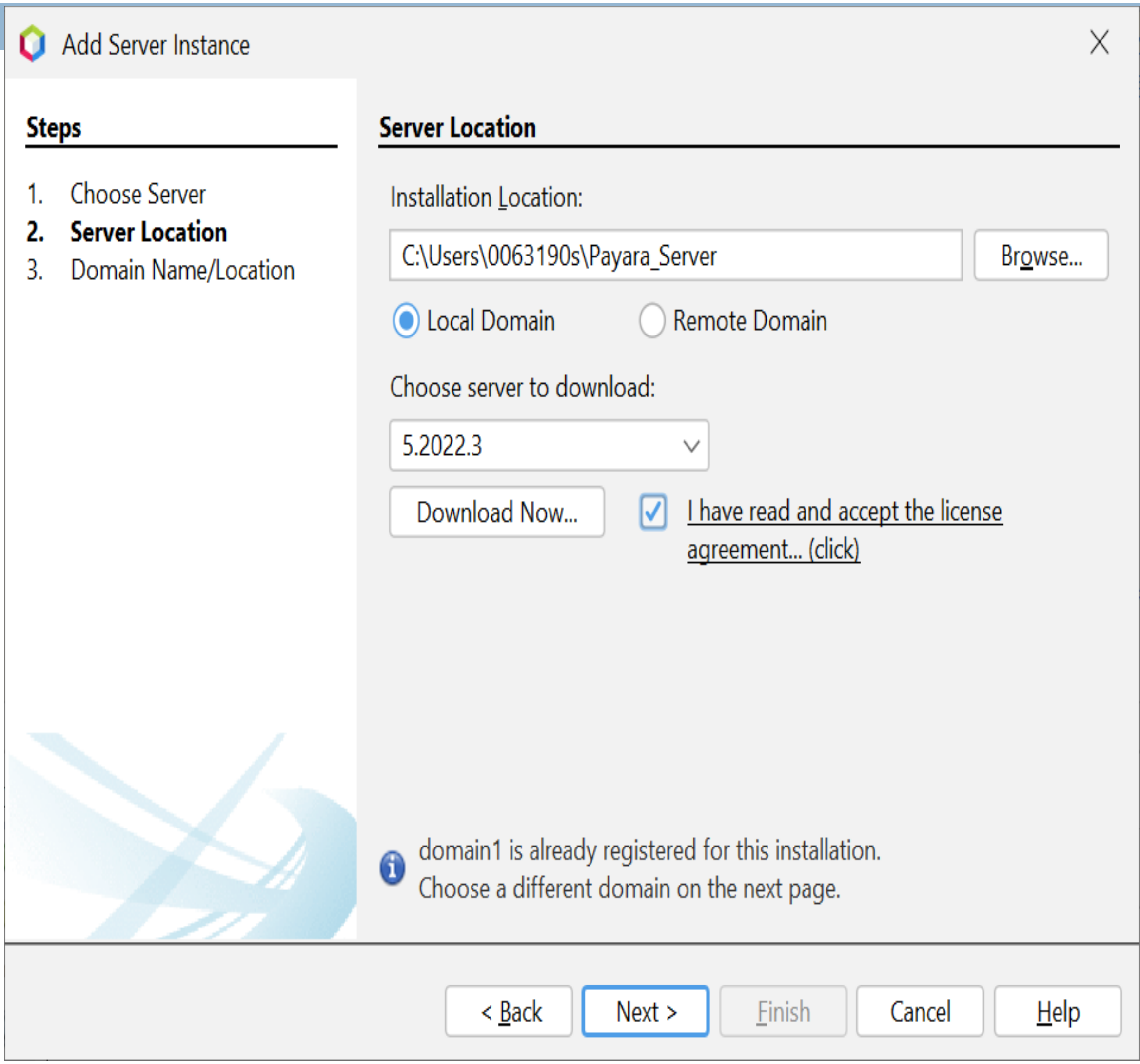
1. Choose Project
2. Name and Location
3. **Settings**

Settings

Server: Payara Server

Java EE Version: Java EE 8 Web

- These options work OK
- Creates an instance of the Payara server



Add Server Instance

Steps

1. Choose Server
2. **Server Location**
3. Domain Name/Location

Server Location

Installation Location: C:\Users\0063190s\Payara_Server

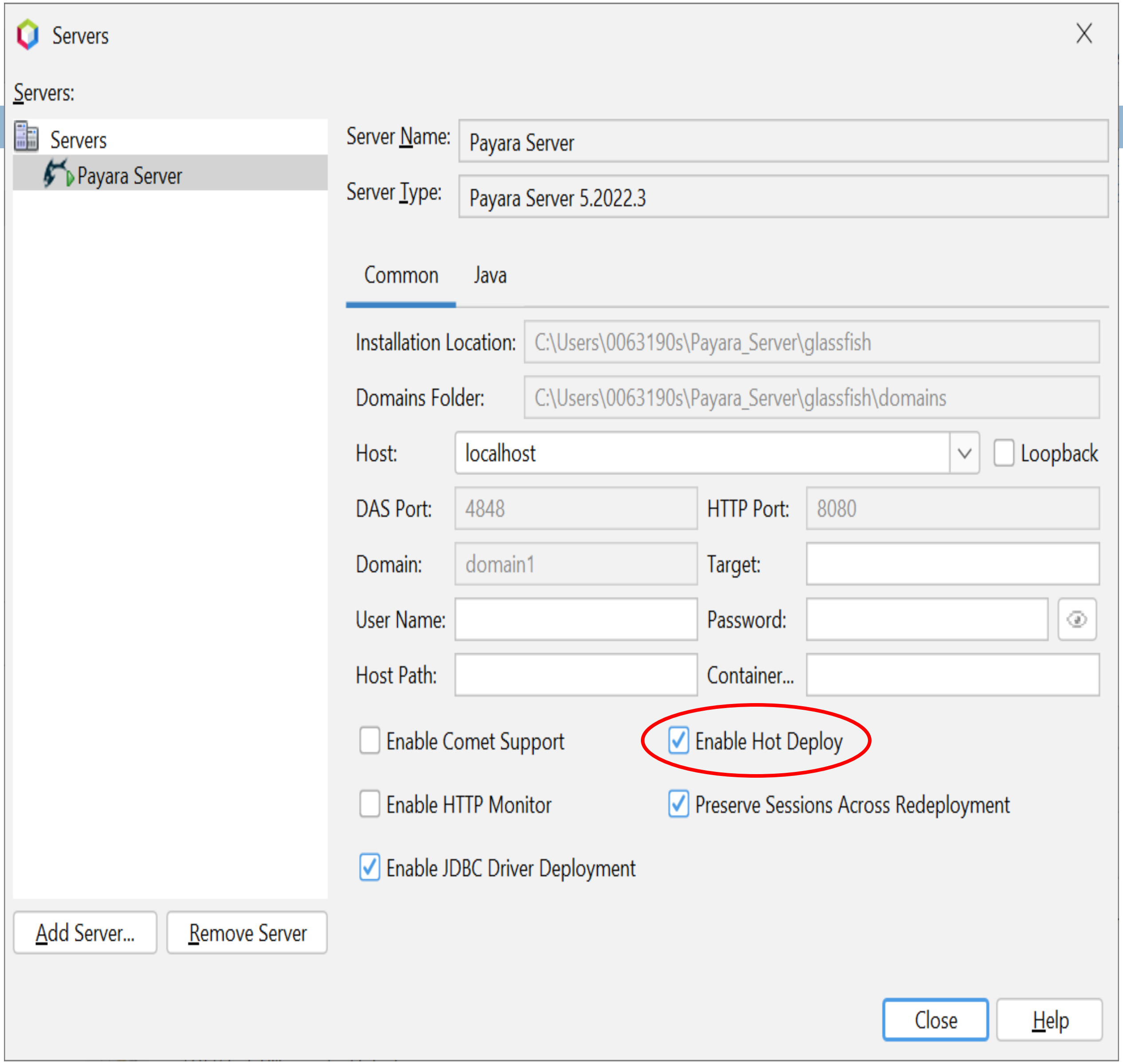
● Local Domain ○ Remote Domain

Choose server to download: 5.2022.3

☑ I have read and accept the license agreement... (click)

domain1 is already registered for this installation.
Choose a different domain on the next page.

- Leave username and password blank for now
- It will listen for HTTP requests on port 8080
- The admin console can be access via port 4848
- Enable Hot Deploy, so that the server doesn't have to be restarted every time you make a change to an application re-deploy it



Server Name: Payara Server

Server Type: Payara Server 5.2022.3

Installation Location: C:\Users\0063190s\Payara_Server\glassfish

Domains Folder: C:\Users\0063190s\Payara_Server\glassfish\domains

Host: localhost ☐ Loopback

DAS Port: 4848 HTTP Port: 8080

Domain: domain1 Target:

User Name: Password:

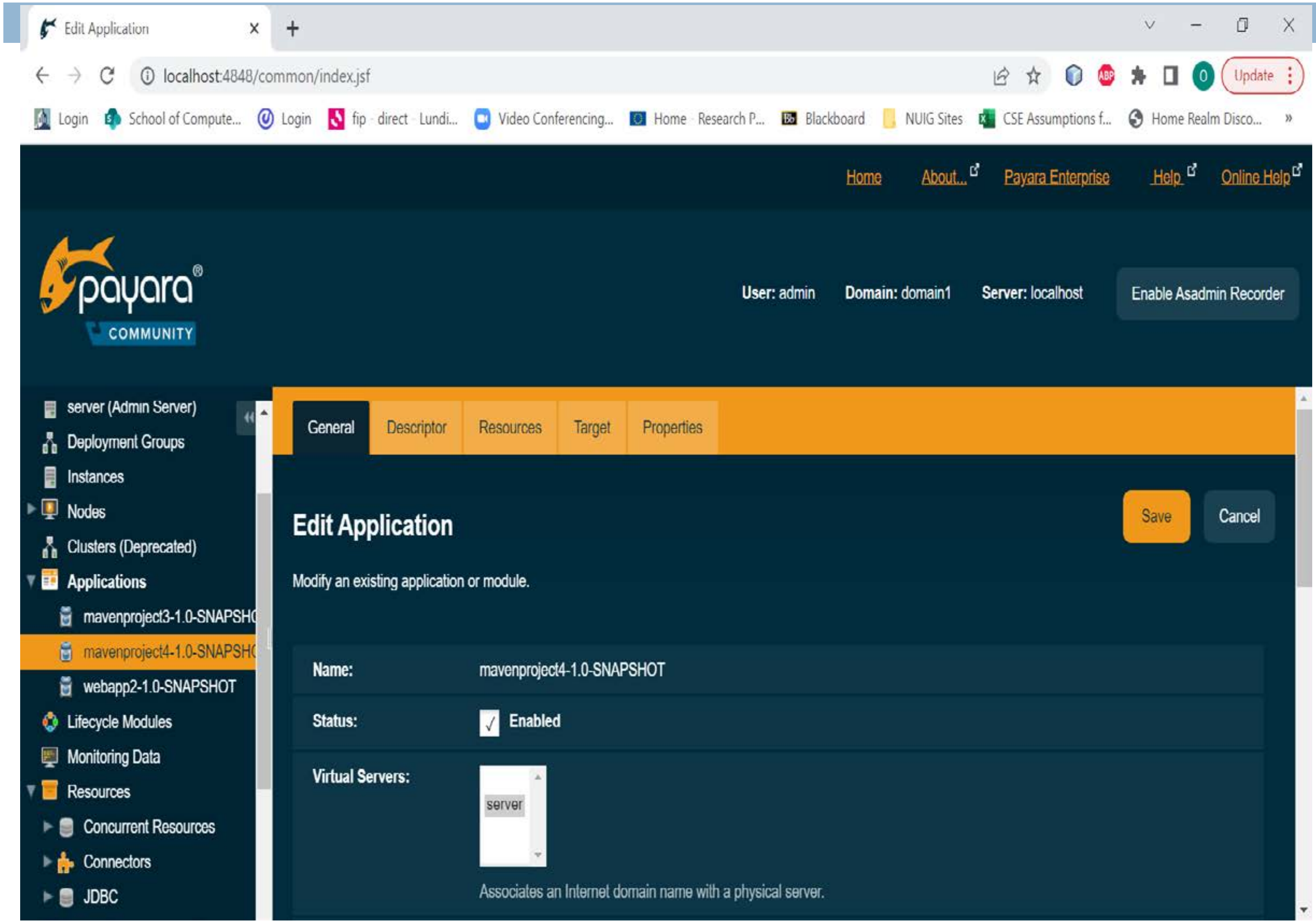
Host Path: Container...

☐ Enable Comet Support ☑ Enable Hot Deploy

☐ Enable HTTP Monitor ☑ Preserve Sessions Across Redeployment

☑ Enable JDBC Driver Deployment

Payara admin console (port 4848)



User: admin Domain: domain1 Server: localhost

Edit Application

Modify an existing application or module.

Name: mavenproject4-1.0-SNAPSHOT

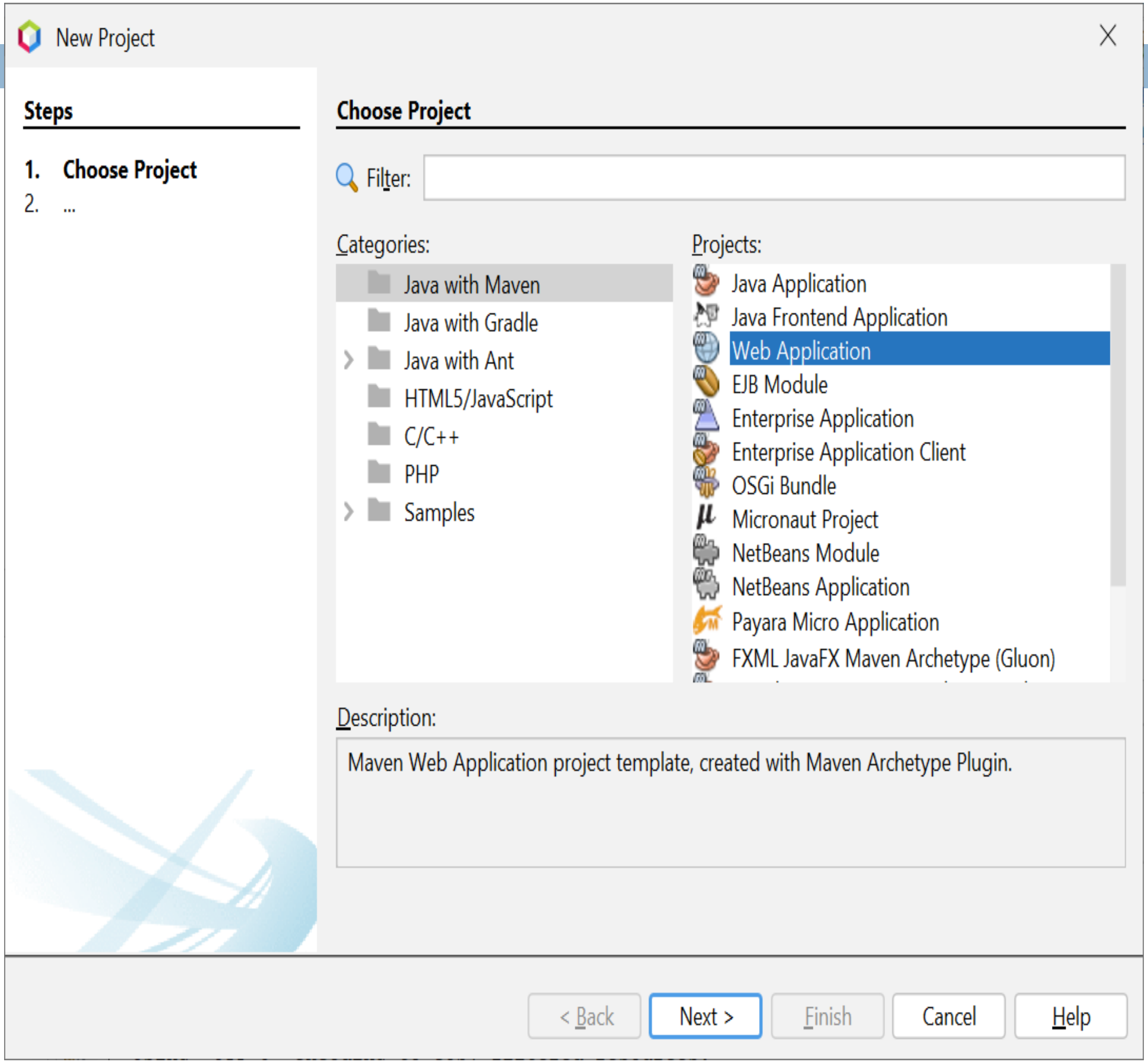
Status: ☑ Enabled

Virtual Servers: server

Associates an Internet domain name with a physical server.

New Project to test installation OK

- Create a new Web App, using Maven as the build tool



New Project

Steps

1. Choose Project
2. ...

Choose Project

Filter:

Categories:

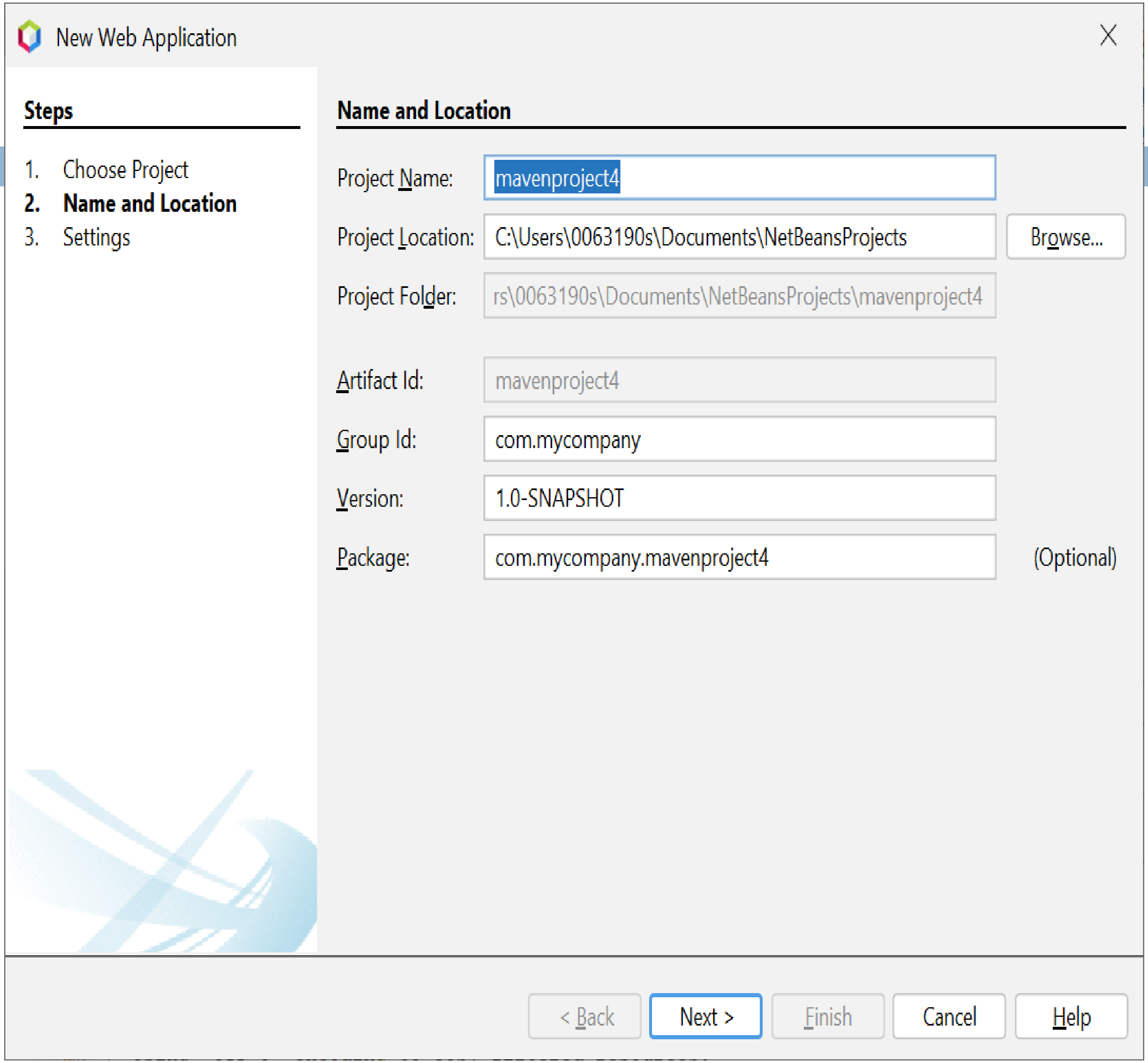
- Java with Maven
- Java with Gradle
- Java with Ant
- HTML5/JavaScript
- C/C++
- PHP
- Samples

Projects:

- Java Application
- Java Frontend Application
- Web Application
- EJB Module
- Enterprise Application
- Enterprise Application Client
- OSGi Bundle
- Micronaut Project
- NetBeans Module
- NetBeans Application
- Payara Micro Application
- FXML JavaFX Maven Archetype (Gluon)

Description:

Maven Web Application project template, created with Maven Archetype Plugin.



New Web Application

Steps

1. Choose Project
2. **Name and Location**
3. Settings

Name and Location

Project Name: mavenproject4

Project Location: C:\Users\0063190s\Documents\NetBeansProjects

Project Folder: rs\0063190s\Documents\NetBeansProjects\mavenproject4

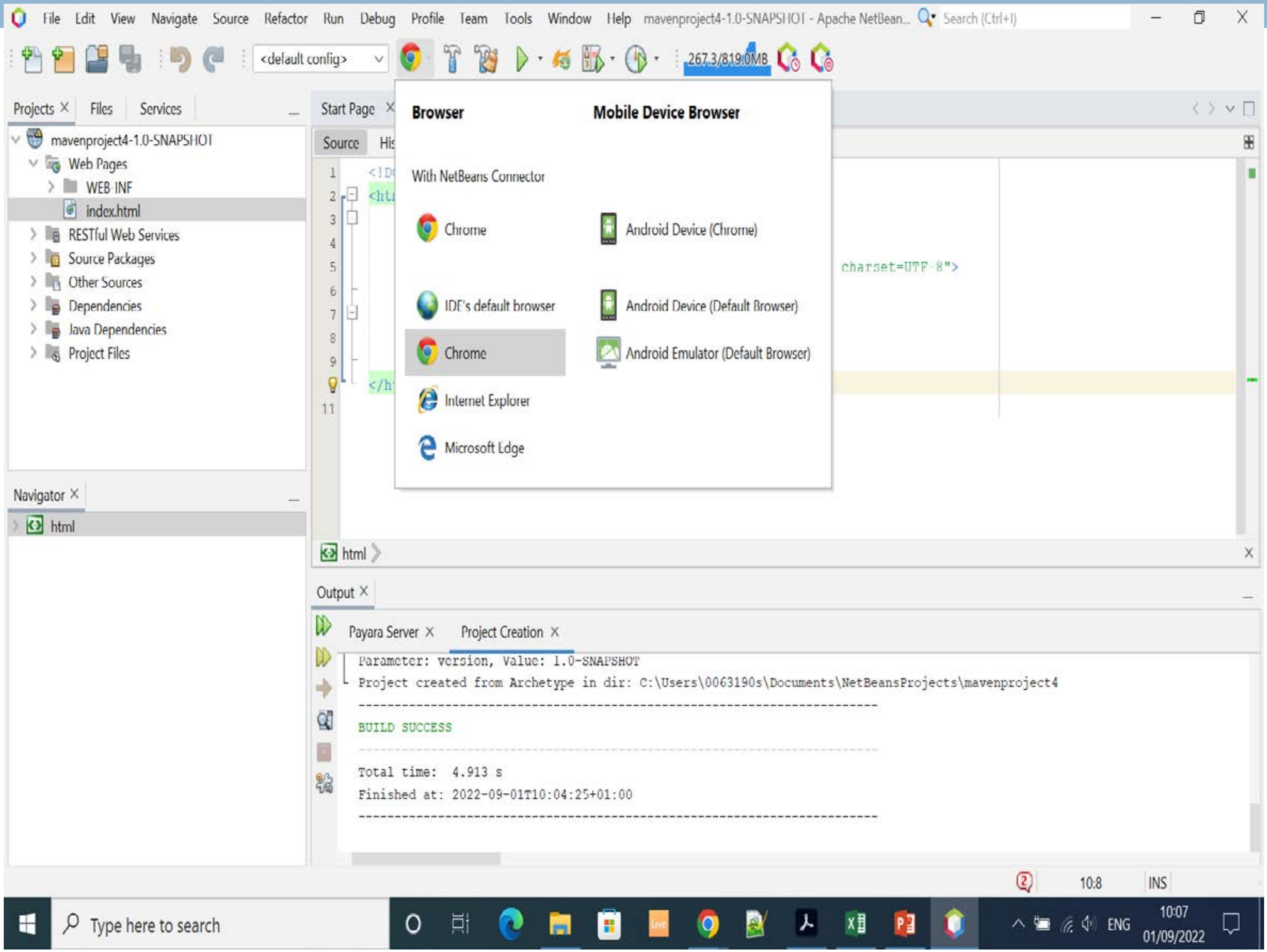
Artifact Id: mavenproject4

Group Id: com.mycompany

Version: 1.0-SNAPSHOT

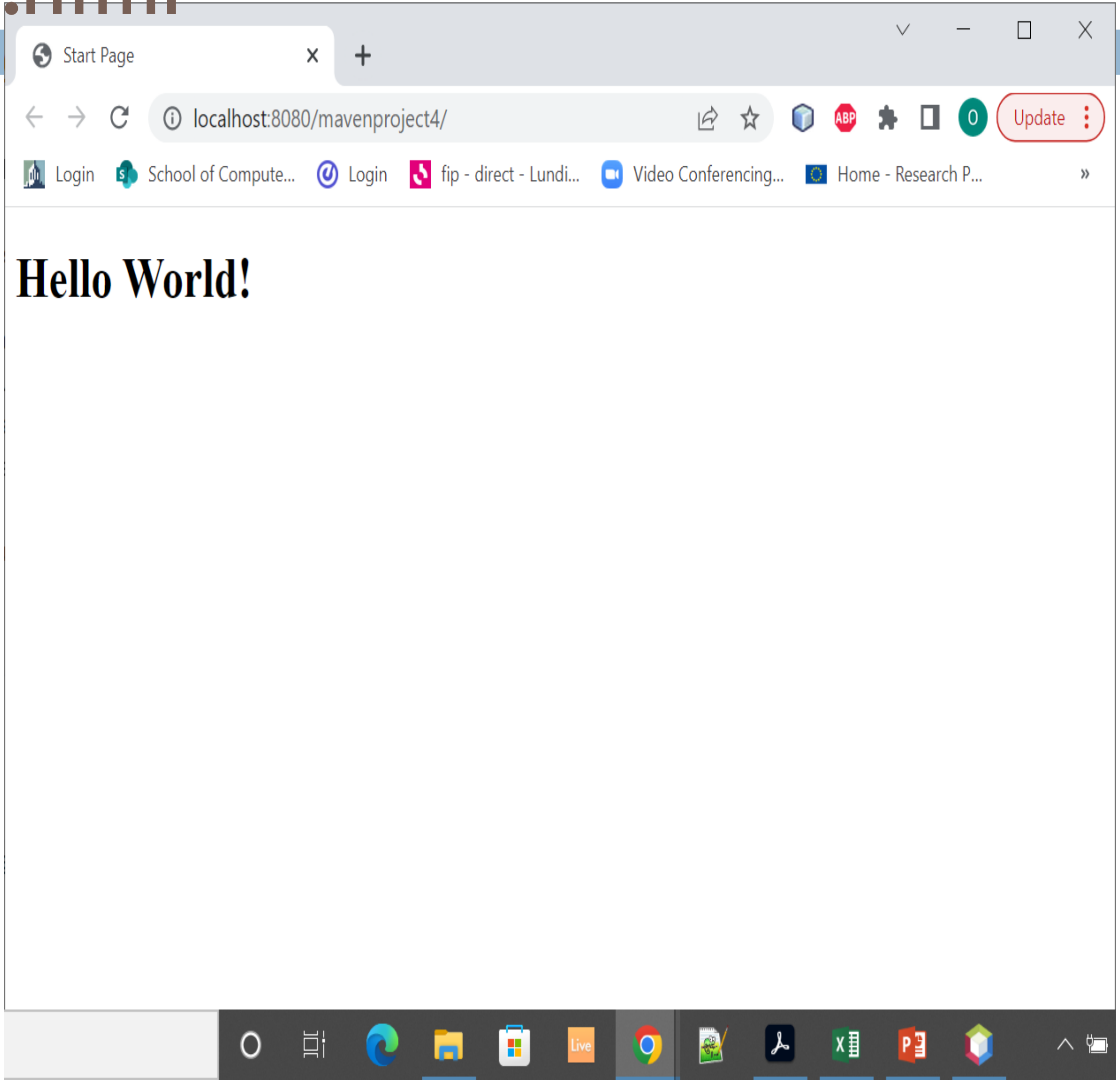
Package: com.mycompany.mavenproject4 (Optional)

Choose Chrome as the default browser to run this app with



```
Parameter: version, Value: 1.0-SNAPSHOT
Project created from Archetype in dir: C:\Users\0063190s\Documents\NetBeansProjects\mavenproject4
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  4.913 s
Finished at: 2022-09-01T10:04:25+01:00
------------------------------------------------------------------------
```

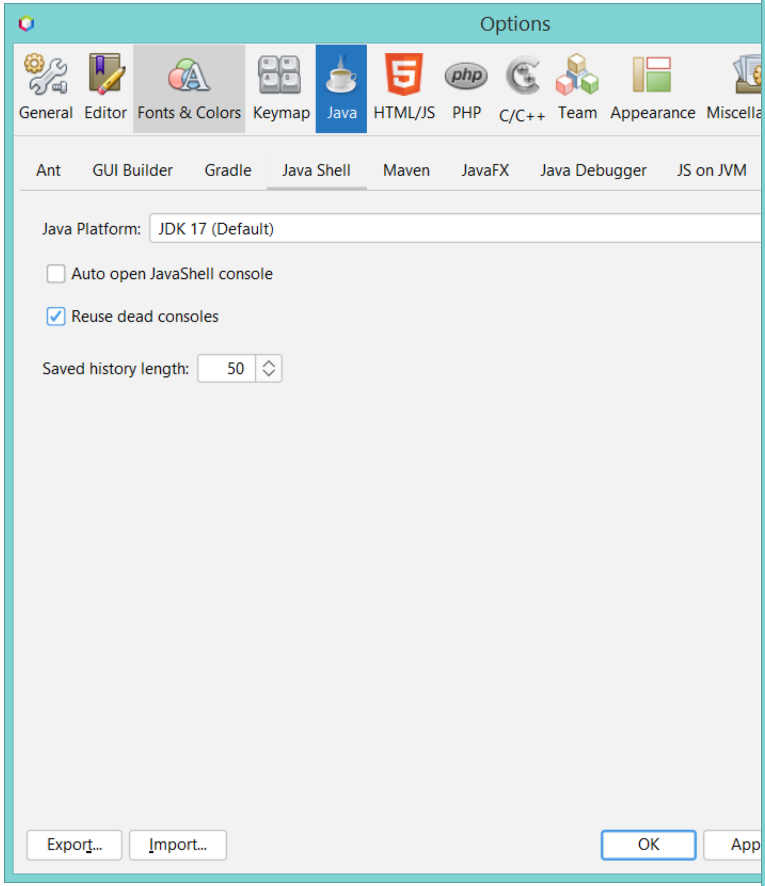
This is just the web server returning index.html



Hello World!

NetBeans 18 (latest version)

- Runs OK with JDK 17 for both Java Shell and Maven
- To check Java versions select Tools / Options



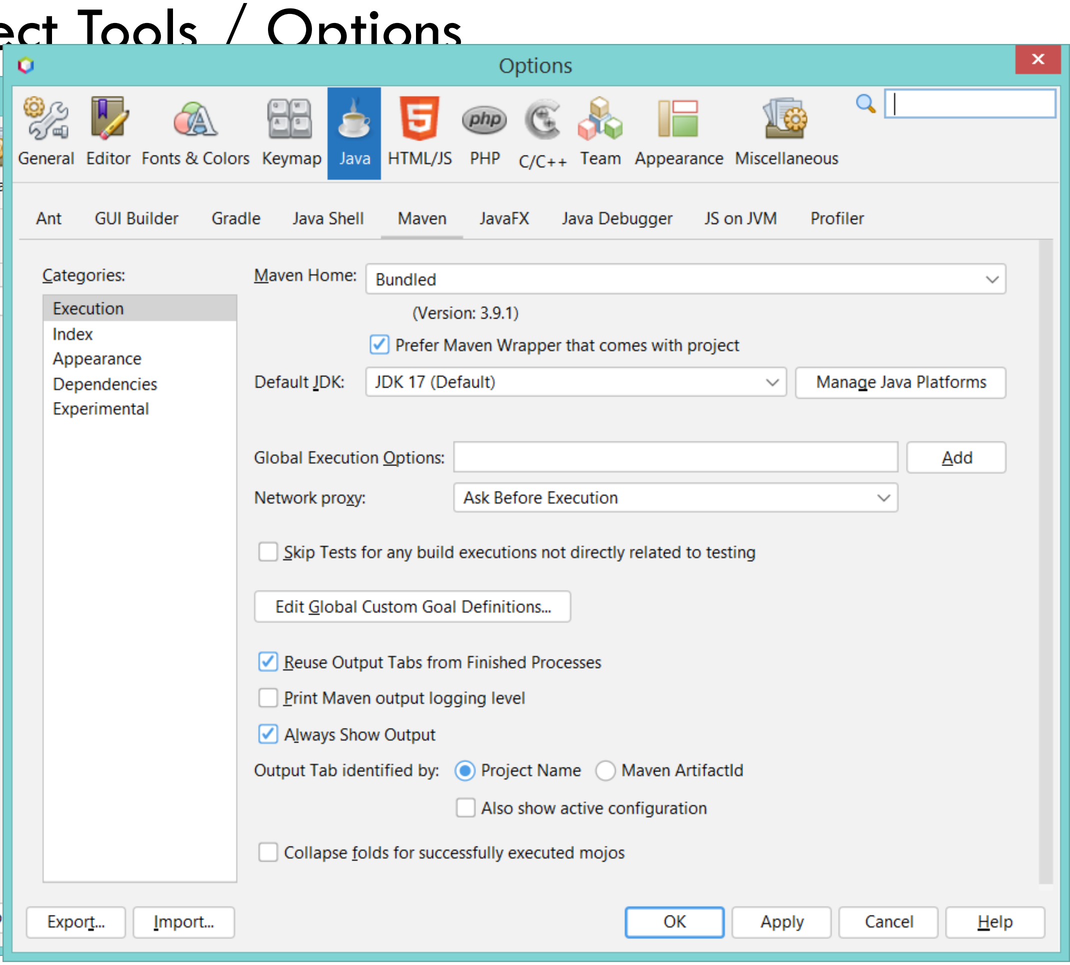
Options

Java Platform: JDK 17 (Default)

☐ Auto open JavaShell console

☑ Reuse dead consoles

Saved history length: 50



Options

Maven Home: Bundled

(Version: 3.9.1)

☑ Prefer Maven Wrapper that comes with project

Default JDK: JDK 17 (Default)

Global Execution Options:

Network proxy: Ask Before Execution

☐ Skip Tests for any build executions not directly related to testing

☑ Reuse Output Tabs from Finished Processes

☐ Print Maven output logging level

☑ Always Show Output

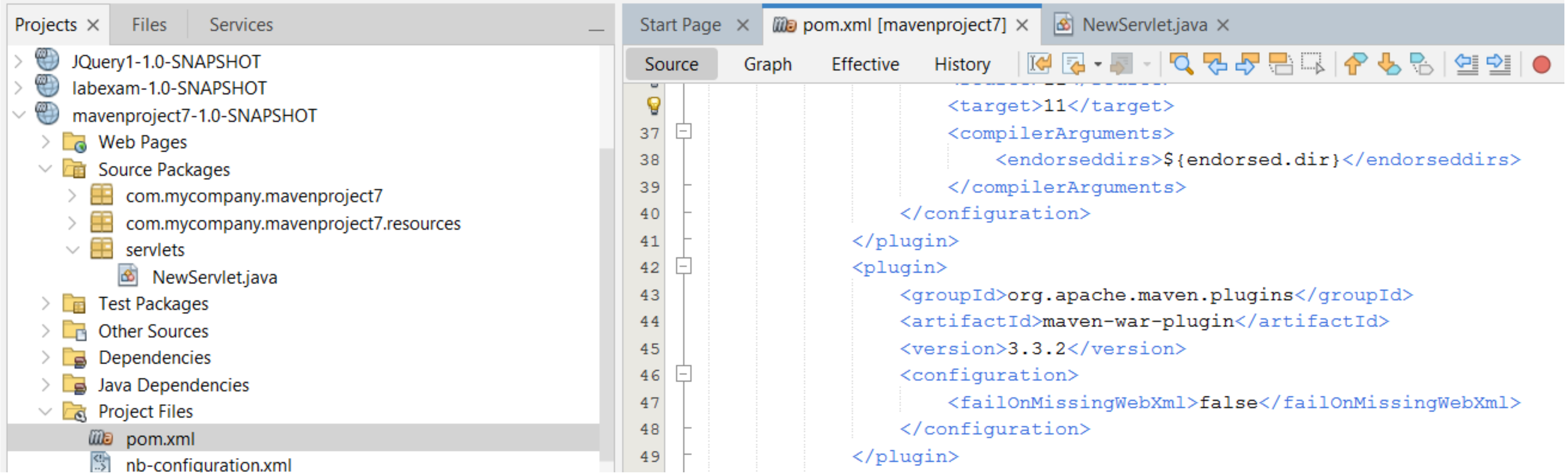
Output Tab identified by: ● Project Name ○ Maven ArtifactId

☐ Also show active configuration

☐ Collapse folds for successfully executed mojos

However

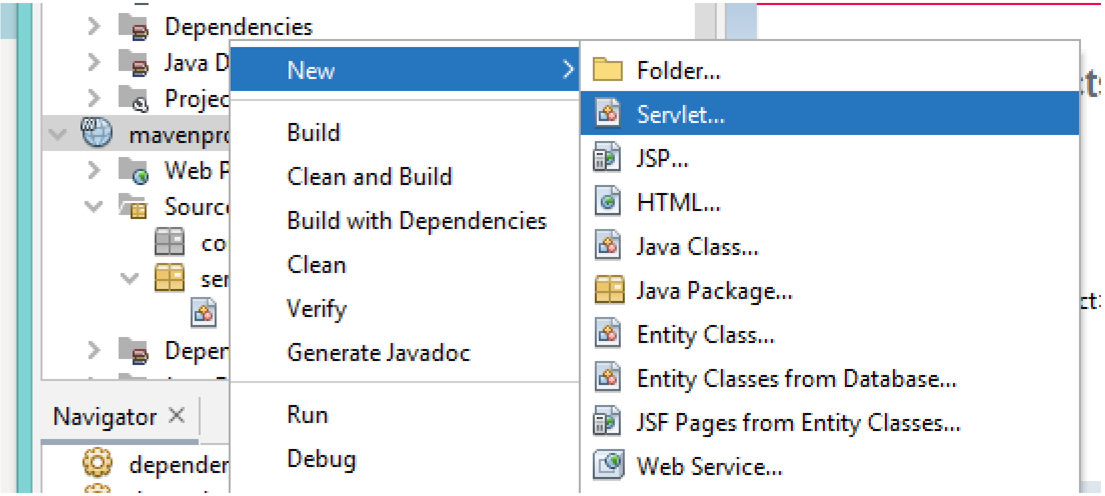
- the pom.xml (Maven build file) seems to be using an old version of a plugin used to package the application into a .war file
- So changed to a later version (3.3.2) and it works OK



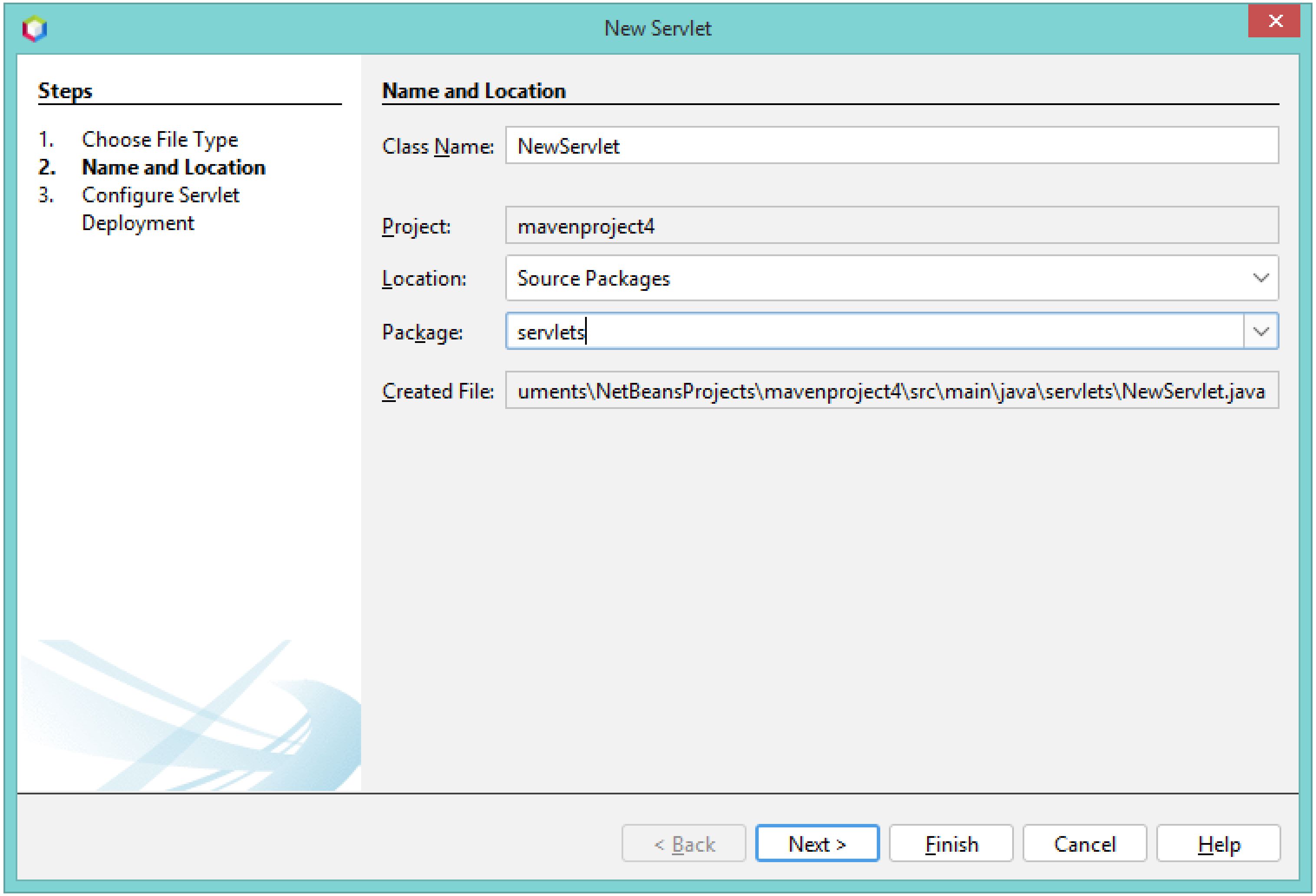
```
        <target>11</target>
        <compilerArguments>
            <endorseddirs>${endorsed.dir}</endorseddirs>
        </compilerArguments>
    </configuration>
</plugin>
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-war-plugin</artifactId>
    <version>3.3.2</version>
    <configuration>
        <failOnMissingWebXml>false</failOnMissingWebXml>
    </configuration>
</plugin>
```

Now add a servlet

- Right click on project and select New/ Servlet



Specify package ‘servlets’ and give the servlet a name



Steps

1. Choose File Type
2. **Name and Location**
3. Configure Servlet Deployment

Name and Location

Class Name: NewServlet

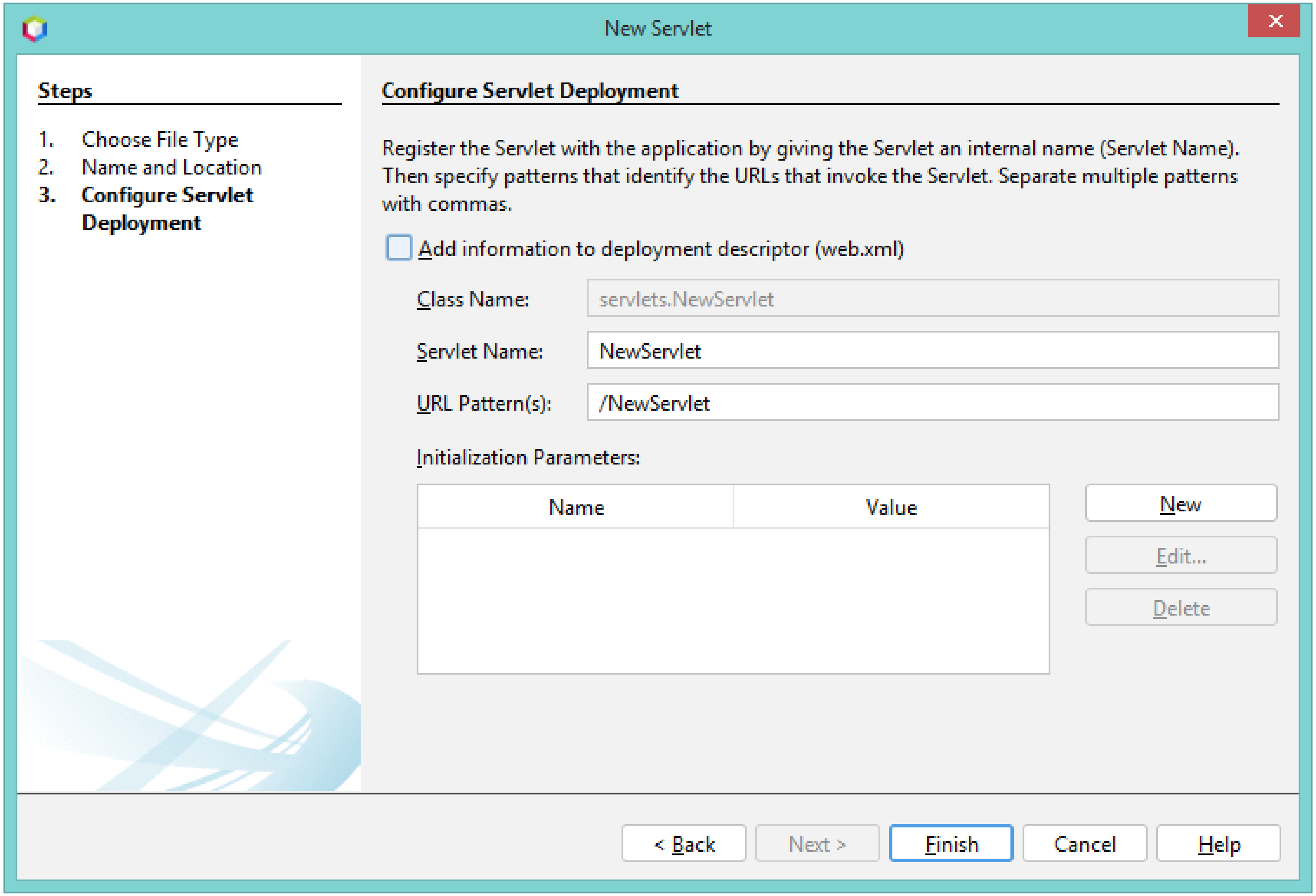
Project: mavenproject4

Location: Source Packages

Package: servlets

Created File: uments\NetBeansProjects\mavenproject4\src\main\java\servlets\NewServlet.java

No need to select anything here - Finish



Steps

1. Choose File Type
2. Name and Location
3. **Configure Servlet Deployment**

Configure Servlet Deployment

Register the Servlet with the application by giving the Servlet an internal name (Servlet Name). Then specify patterns that identify the URLs that invoke the Servlet. Separate multiple patterns with commas.

□ Add information to deployment descriptor (web.xml)

Class Name: servlets.NewServlet

Servlet Name: NewServlet

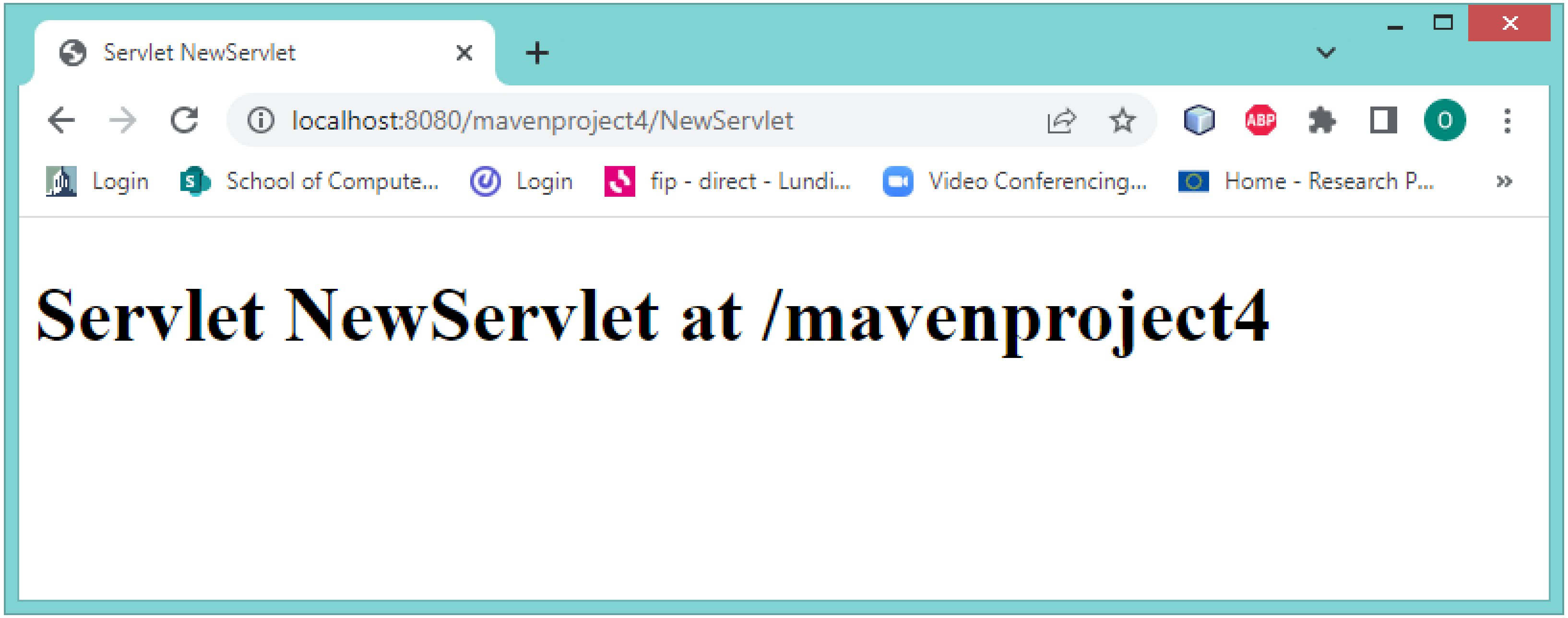
URL Pattern(s): /NewServlet

Initialization Parameters:

Name	Value

Run the application again and call servlet using the URL specified

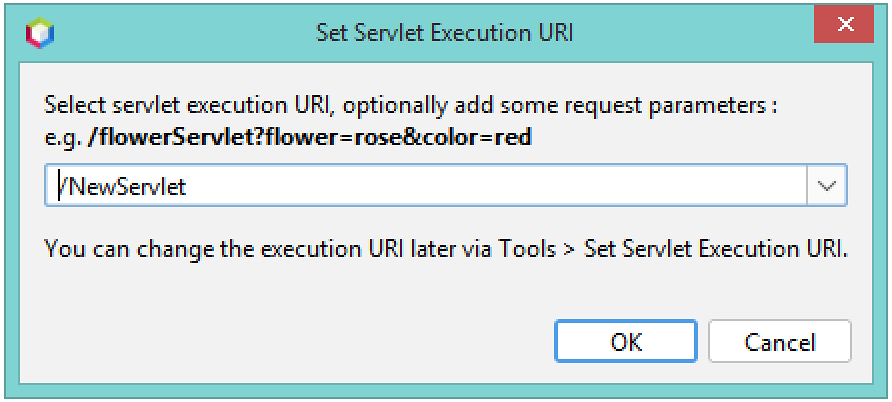
- See servlet code:
  - @WebServlet(name = "NewServlet", urlPatterns = {"/NewServlet"})



Servlet NewServlet at /mavenproject4

Can also use “Run File”

- Right click in code in servlet and select *Run File*
- Gives you the option to specify request parameters in the URL (e.g. for testing purposes)



Select servlet execution URI, optionally add some request parameters :
e.g. **/flowerServlet?flower=rose&color=red**

/NewServlet

You can change the execution URI later via Tools > Set Servlet Execution URI.

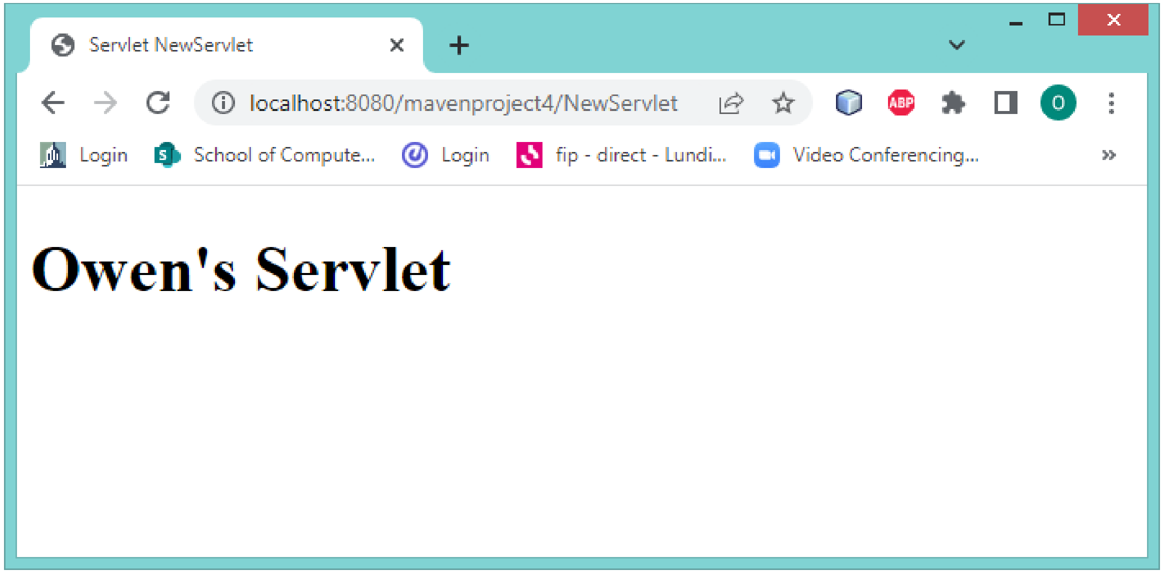
Core method being used in servlet

- Servlets process HTTP requests – the default code created in NetBeans redirects both GET and POST requests to the method ***processRequest***

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
 throws ServletException, IOException  {
    response.setContentType("text/html;charset=UTF-8");
    try ( PrintWriter out = response.getWriter()) {
    /* TODO output your page here. You may use following sample code. */
      out.println("<!DOCTYPE html>");
      out.println("<html>");
      out.println("<head>");
      out.println("<title>Servlet NewServlet</title>");
      out.println("</head>");
      out.println("<body>");
      out.println("<h1>Servlet NewServlet at " + request.getContextPath() + "</h1>");
      out.println("</body>");
      out.println("</html>");
 } }
```

What to do this week

- Install NetBeans
- Create Web App
- Create servlet which returns your name, e.g.



Owen's Servlet