

Programmazione avanzata

Lezione 1

Installazione ambiente e prima applicazione

Installazione ambiente

L'ambiente di sviluppo per i laboratori è composto dai seguenti software:

- OpenJDK, un'implementazione open-source di Java Virtual Machine and Development Kit liberamente scaricabile e utilizzabile (a differenza della versione ufficiale di Oracle). La versione per diversi sistemi operativi, non solo per Windows, può essere scaricata qua:

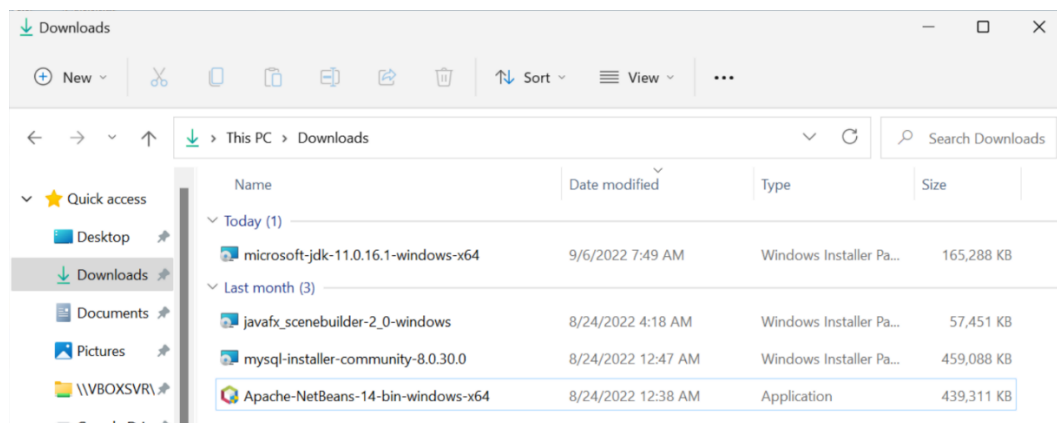
<https://docs.microsoft.com/en-us/java/openjdk/download>

Per i laboratori è stata scelta la versione 11, che offre piena compatibilità con tutti i framework/soluzioni considerate nel corso

OpenJDK 11.0.16.1 LTS | [See previous releases](#)

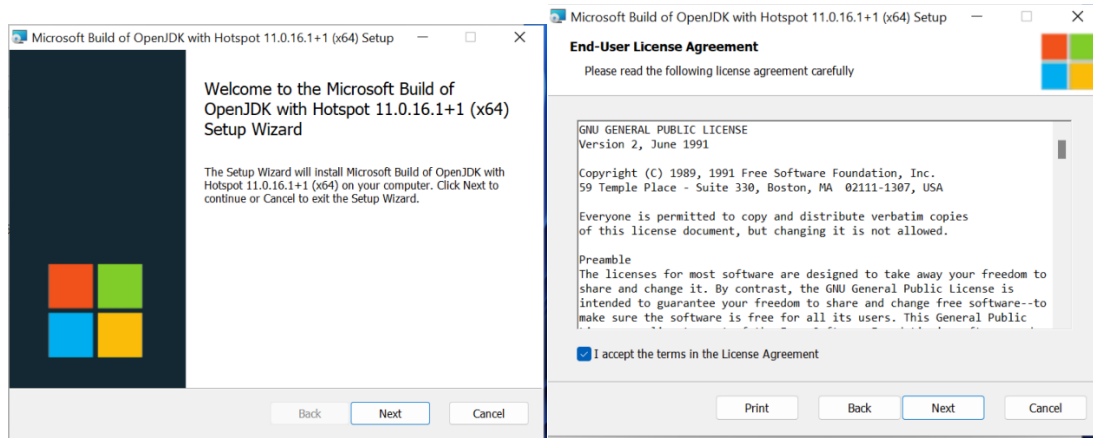
Platform	Architecture	Type	Download Link	Other Files
x64				
Linux	x64	tar.gz	microsoft-jdk-11.0.16.1-linux-x64.tar.gz	sha256 / sig
Alpine	x64 (musl)	tar.gz	microsoft-jdk-11.0.16.1-alpine-x64.tar.gz	sha256 / sig
macOS	x64	tar.gz	microsoft-jdk-11.0.16.1-macos-x64.tar.gz	sha256 / sig
macOS	x64	pkg	microsoft-jdk-11.0.16.1-macos-x64.pkg	sha256
Windows	x64	zip	microsoft-jdk-11.0.16.1-windows-x64.zip	sha256 / sig
Windows	x64	msi	microsoft-jdk-11.0.16.1-windows-x64.msi	sha256
AArch64				
Linux	AArch64 / ARM64	tar.gz	microsoft-jdk-11.0.16.1-linux-aarch64.tar.gz	sha256 / sig

- Apache NetBeans, un editor Java che integra tutte le funzionalità richieste per lo sviluppo di applicazioni di vario tipo.
- La suite MySQL, che include il database MySQL e il tool MySQL Workbench per l'interrogazione dei dati, in aggiunta a tutte le librerie JAVA necessarie per connettere i programmi al database
- JavaFX, un tool grafico integrato con NetBeans che permette la progettazione di interfacce grafiche.

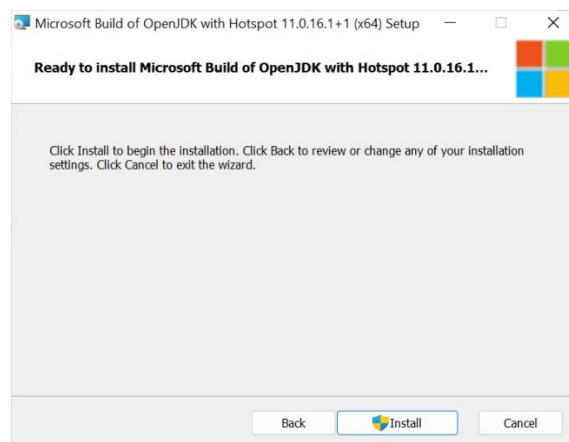
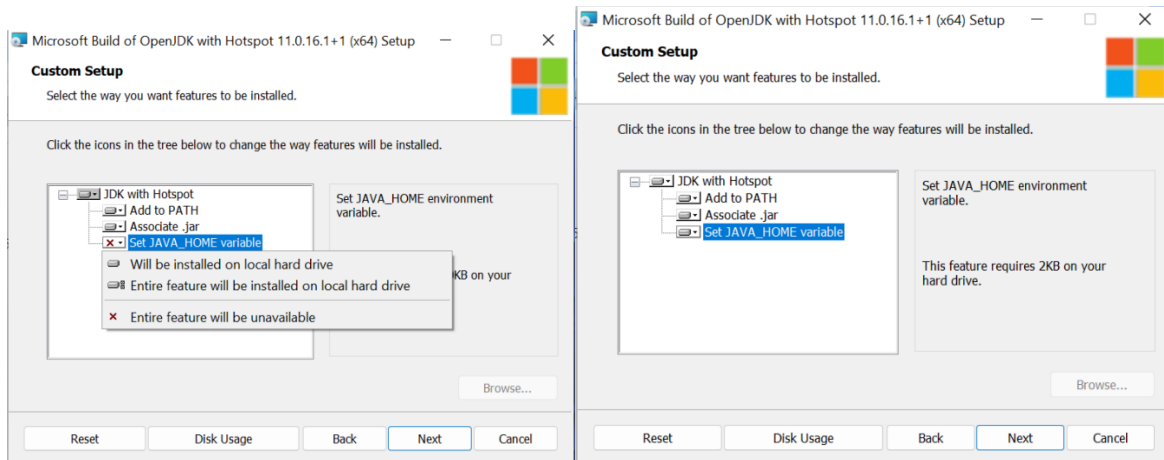


Installazione OpenJDK

L'installazione del Java Development Kit (che include anche il Runtime Environment) e' fatta seguendo i passaggi del programma di installazione.

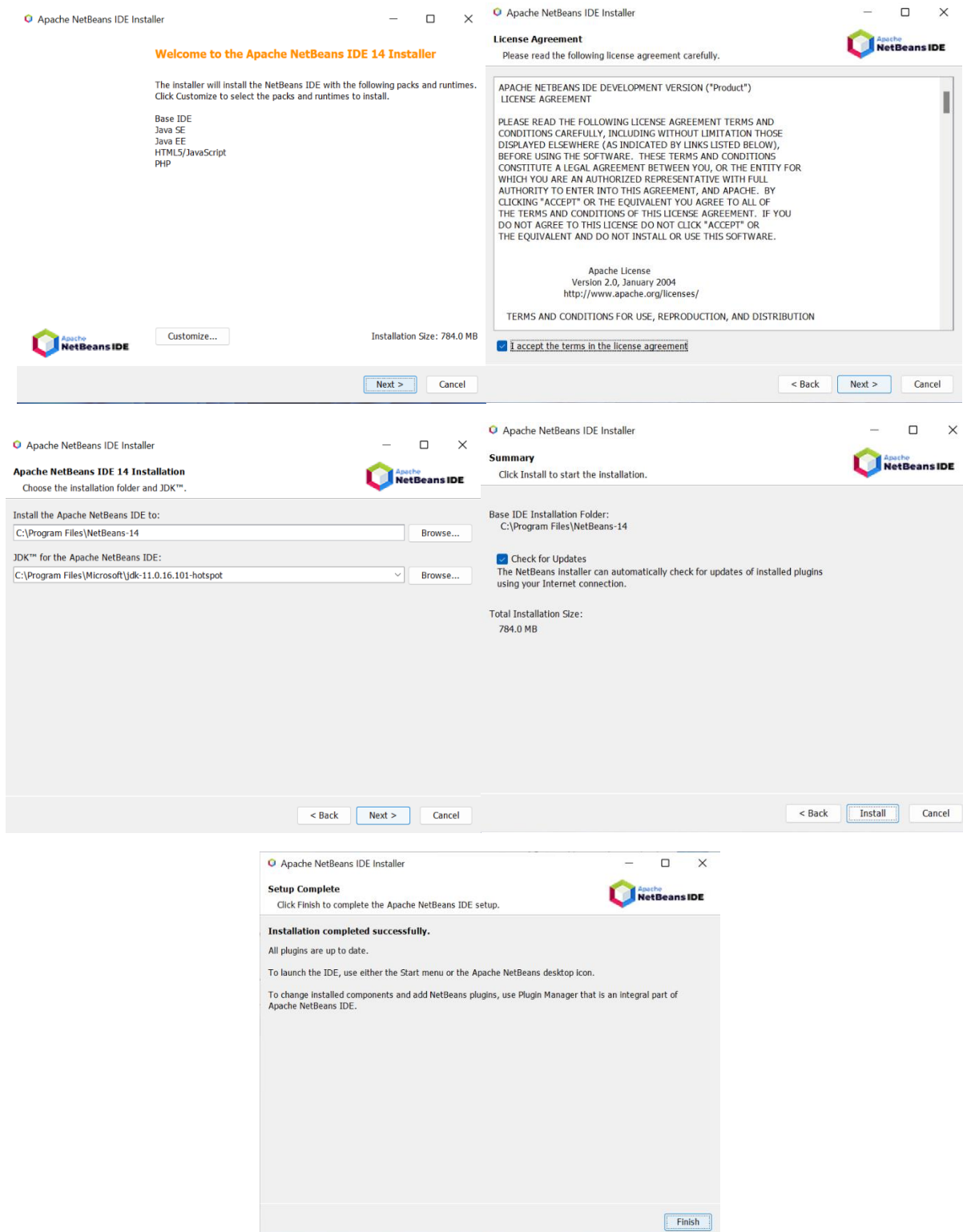


E' importante essere sicuri di aver spuntato l'opzione "Set JAVA_HOME variable" nella 3° schermata.



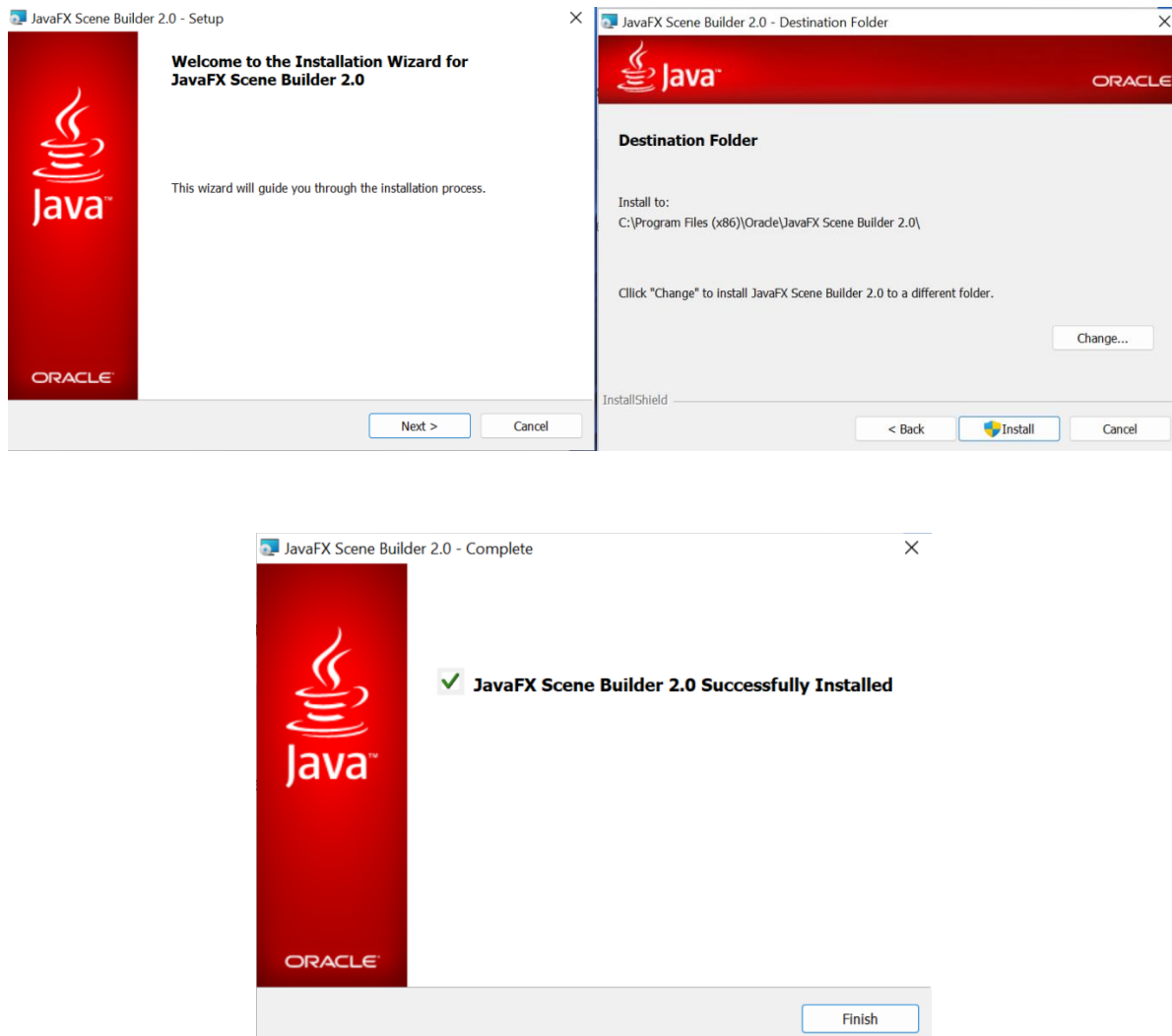
Installazione NetBeans 14

L'installazione dell'ambiente di sviluppo avviene in maniera analoga alla precedente installazione.



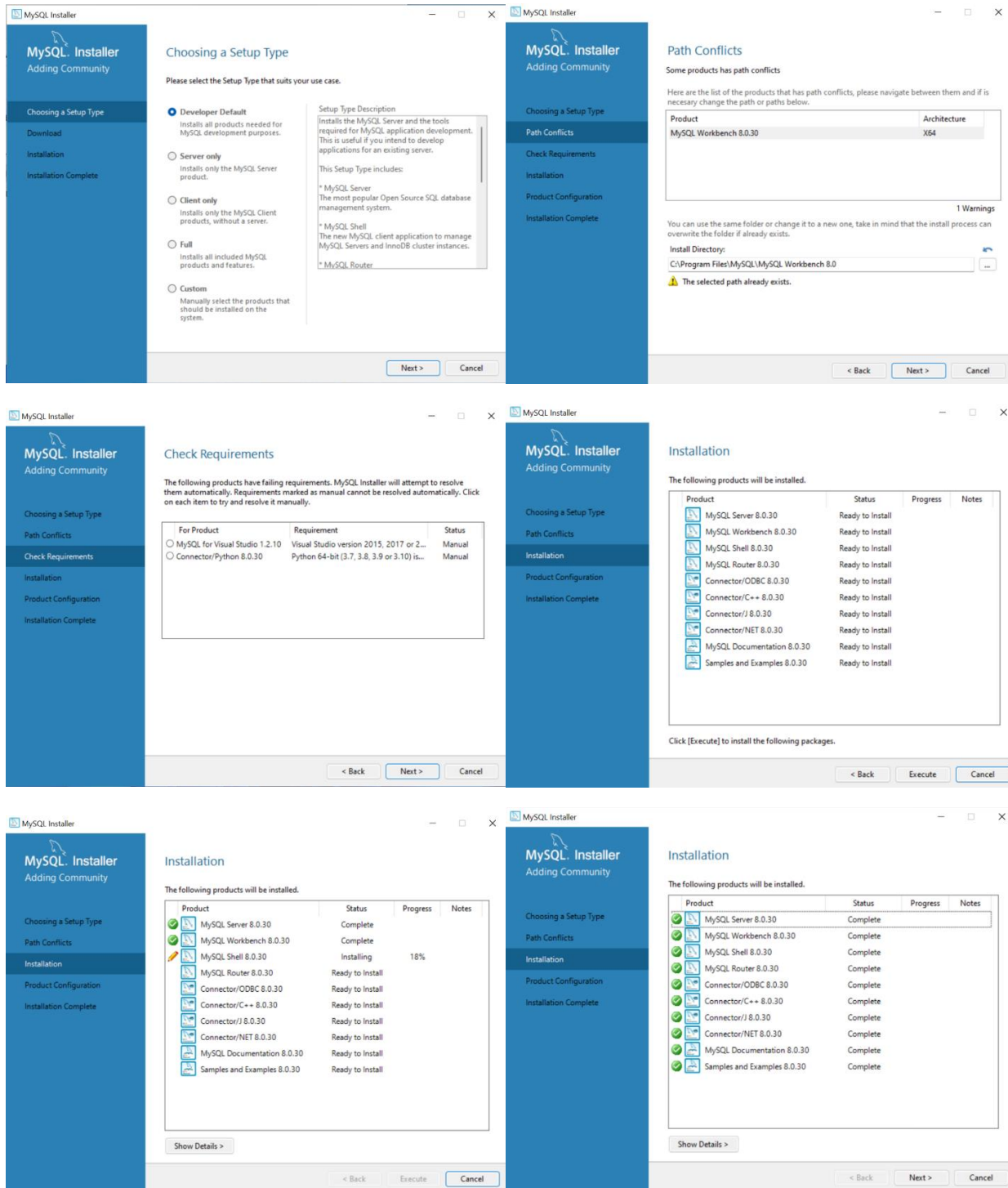
Installazione di JavaFX Builder

L'installazione di JavaFX Scene Builder installa un tool che permette la composizione delle interfacce grafiche. Il completo utilizzo integrato in netbeans richiede un ulteriore passaggio di configurazione (che vedremo).

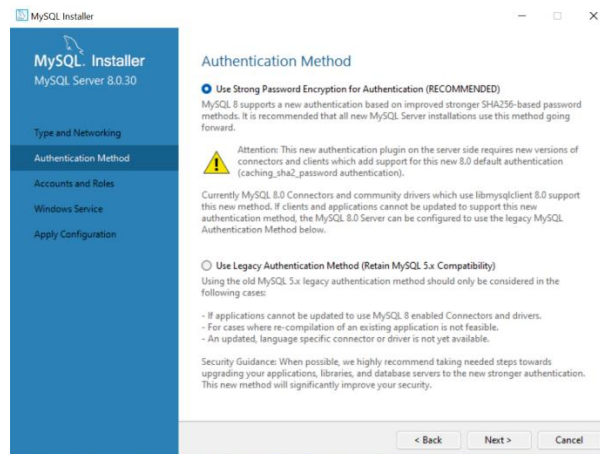
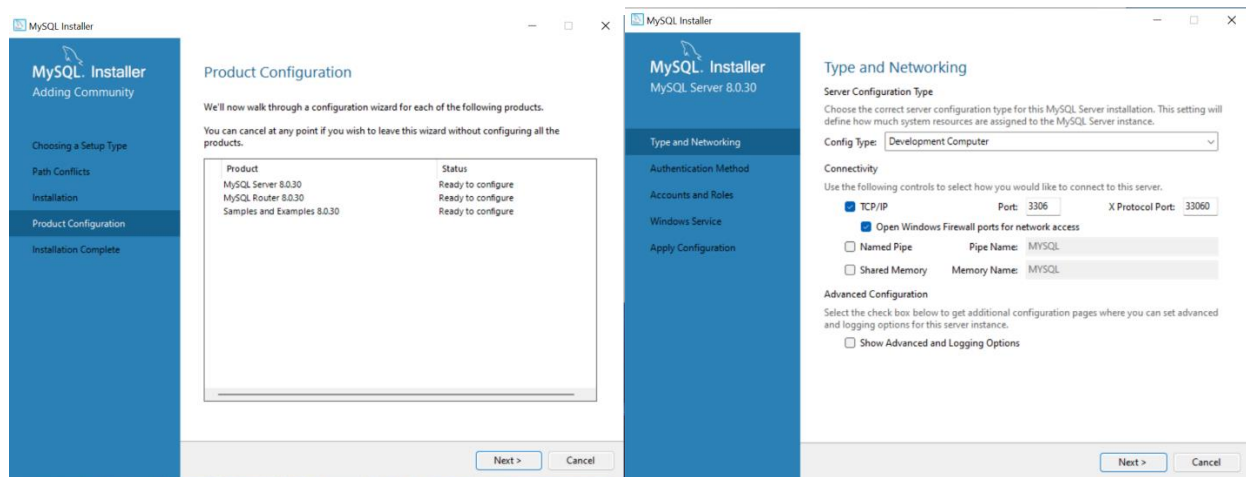


Installazione di MySQL

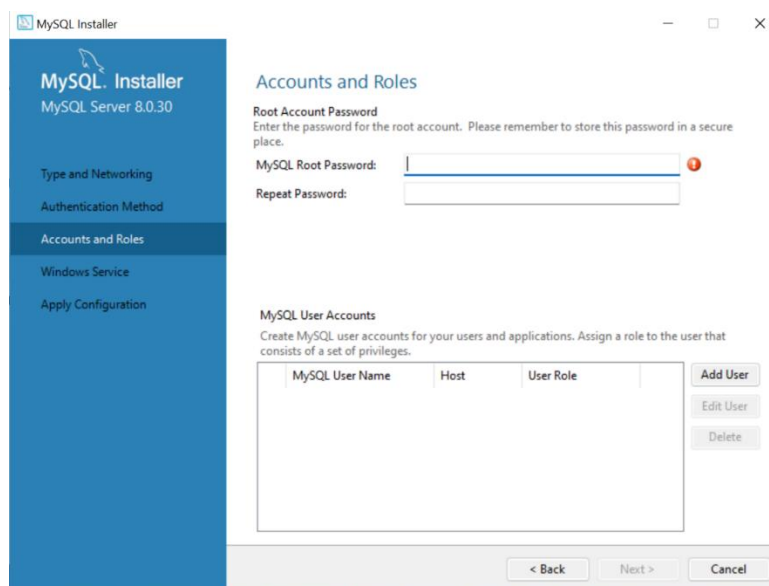
MySQL è composto da un servizio (il DB manager), un tool, MySQL Workbench, che può essere utilizzato dallo sviluppatore per interagire con il database e le librerie necessarie per interagire con il sistema. L'installazione è automatizzata e avviene attraverso le seguenti fasi:

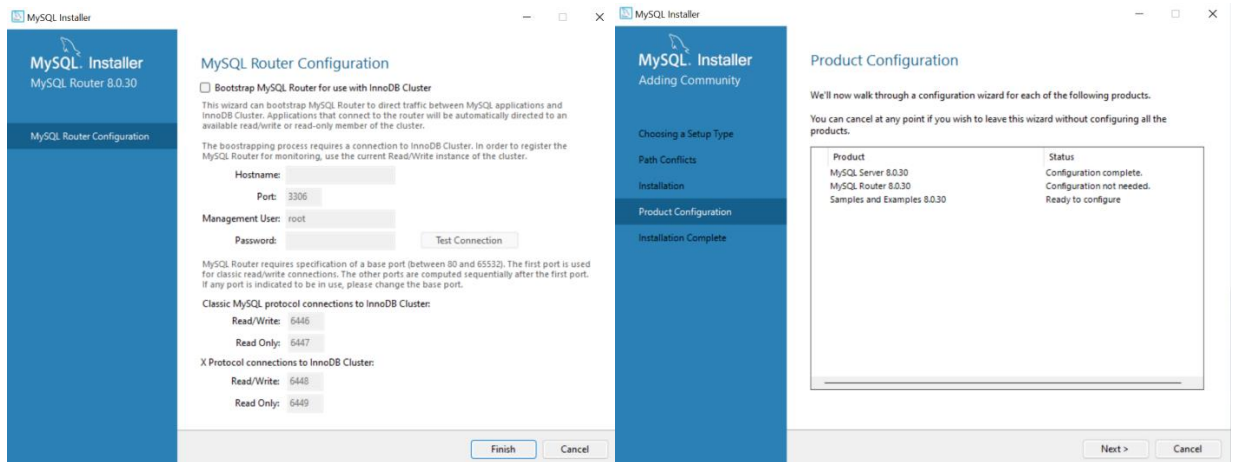
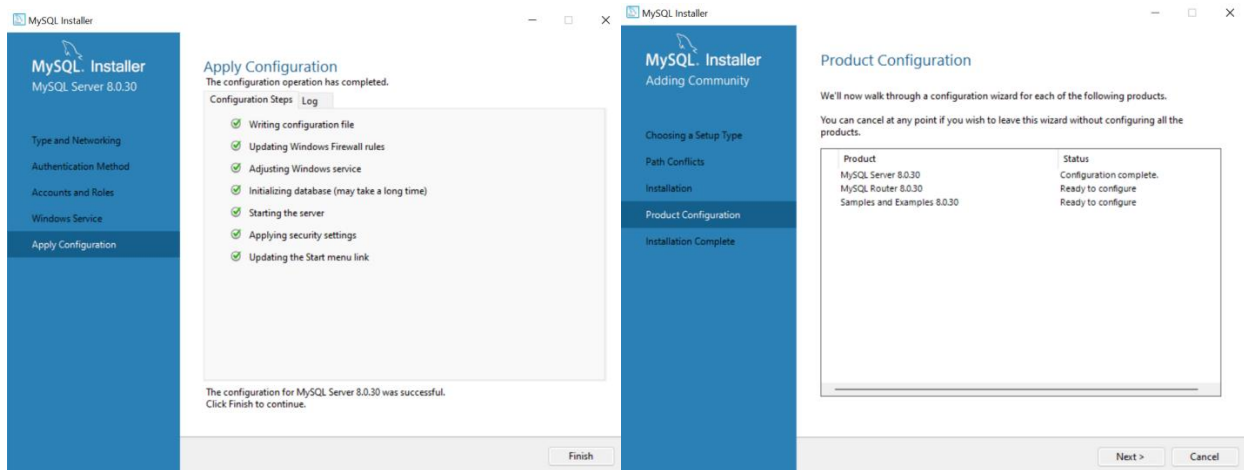
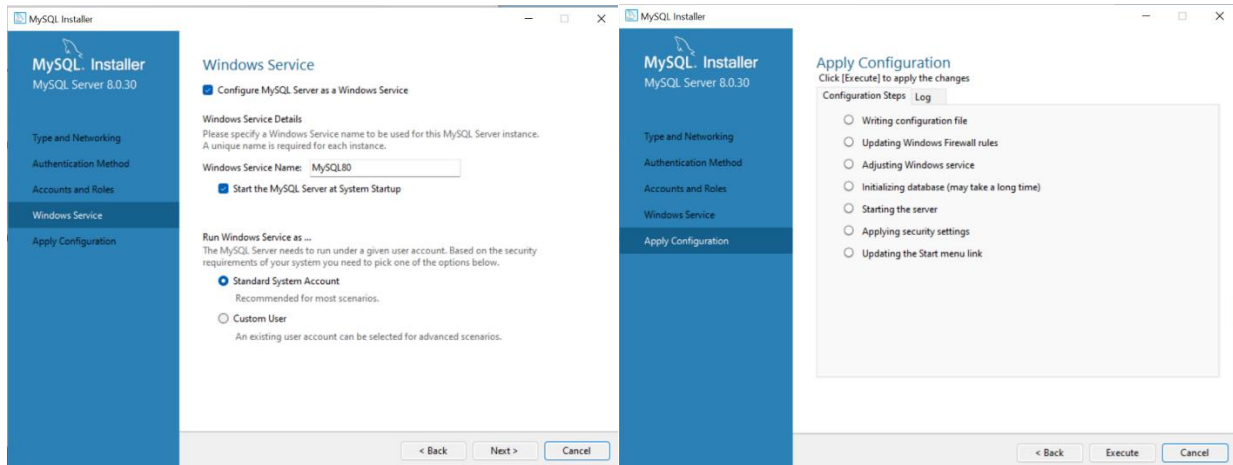


Finita l'installazione automatizzata si procede alla configurazione delle componenti:

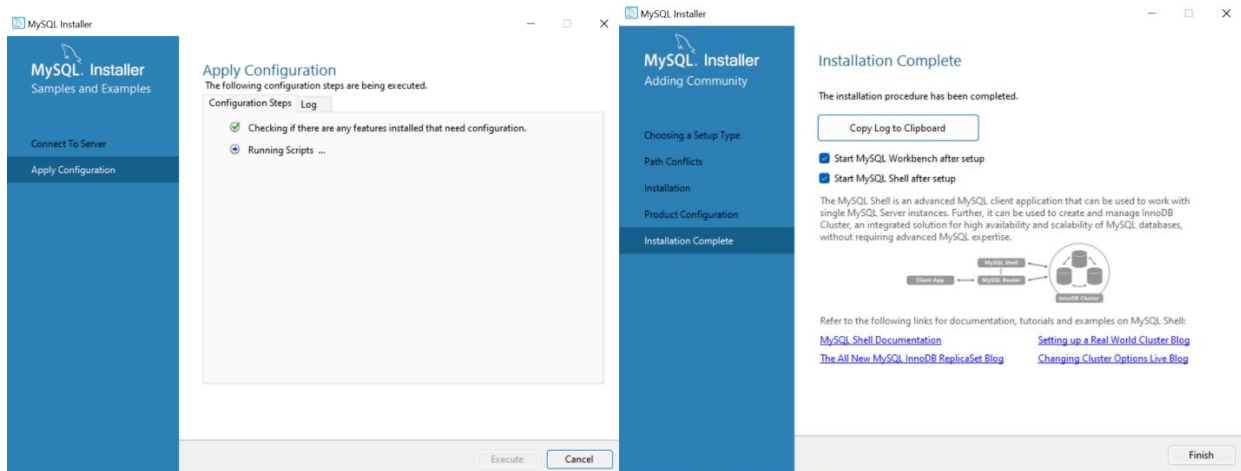
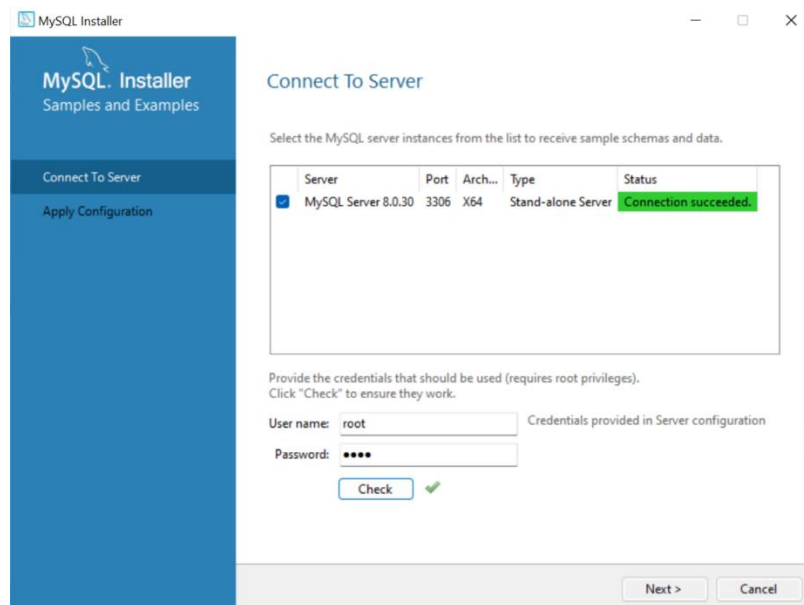


Passaggio importante e' la configurazione della password di 'root', utente amministratore del database.



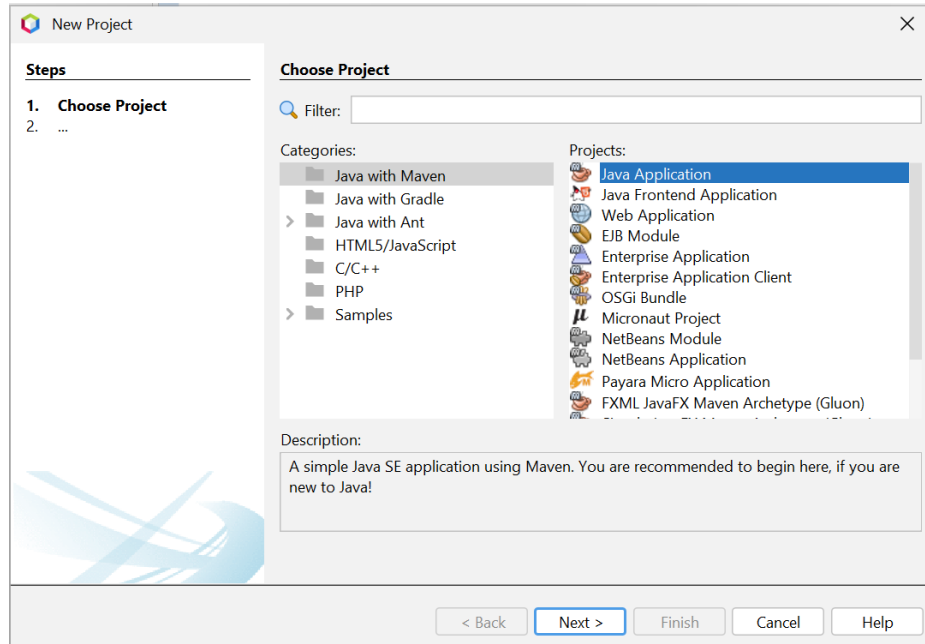


Alla fine dell'installazione si può fare un test di connettività prima di procedere alla finalizzazione.

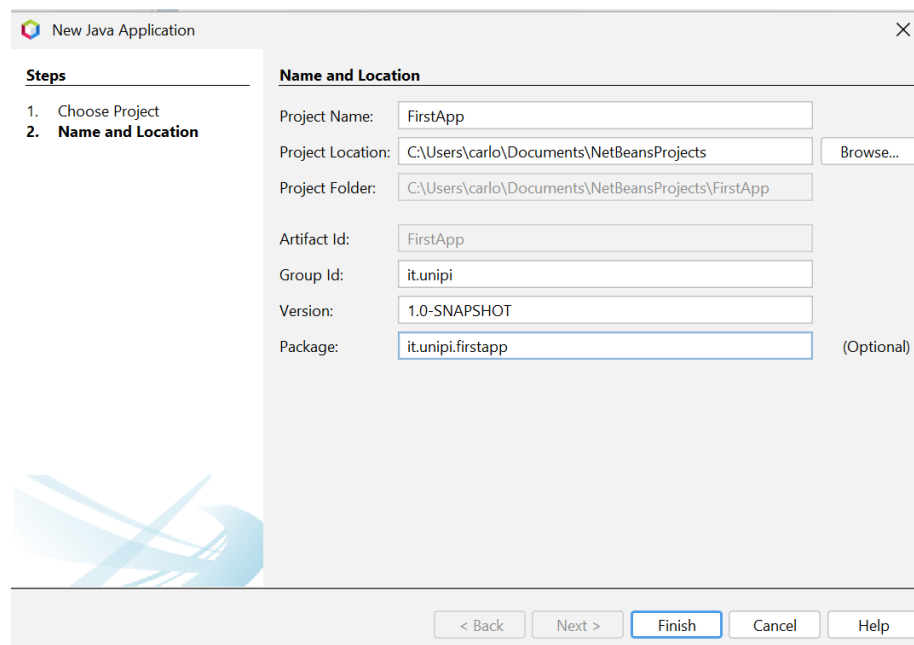


Creare la prima applicazione

La prima applicazione con uno scheletro vuoto pronto per lo sviluppo può essere creata tramite l'interfaccia di NetBeans New->Project nella quale si può selezionare la creazione di una semplice applicazione java nella categoria Java with Maven (sarà chiaro tra poco cosa è Maven).



Specificare il nome e il nome del package da creare.

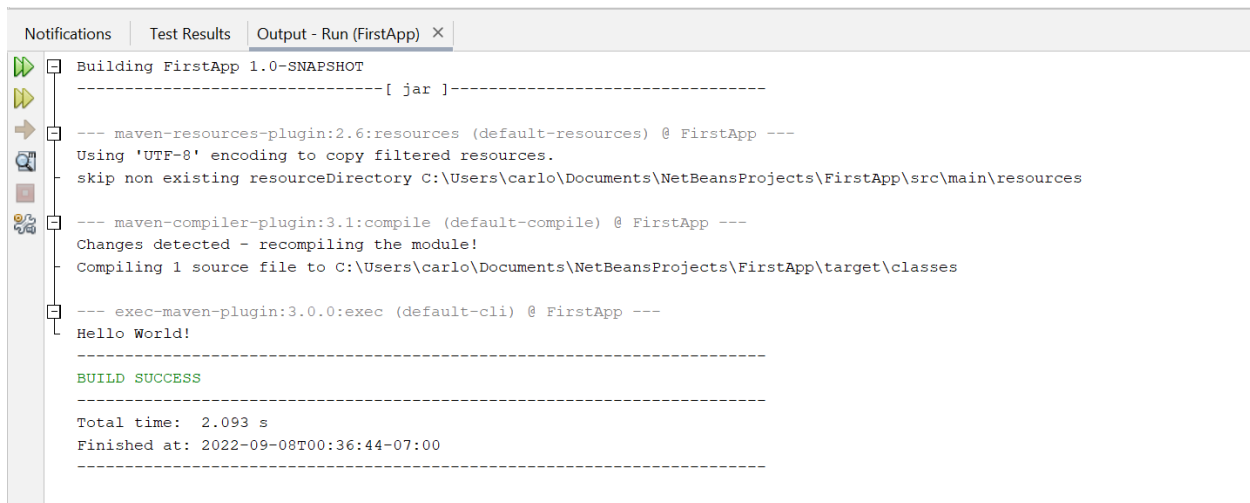


Il sistema crea una scheletro della app con un codice base che include un main con una sola istruzione all'interno pronto per essere eseguito.

```
package it.unipi.firstapp;

/**
 *
 * @author carlo
 */
public class FirstApp {

    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

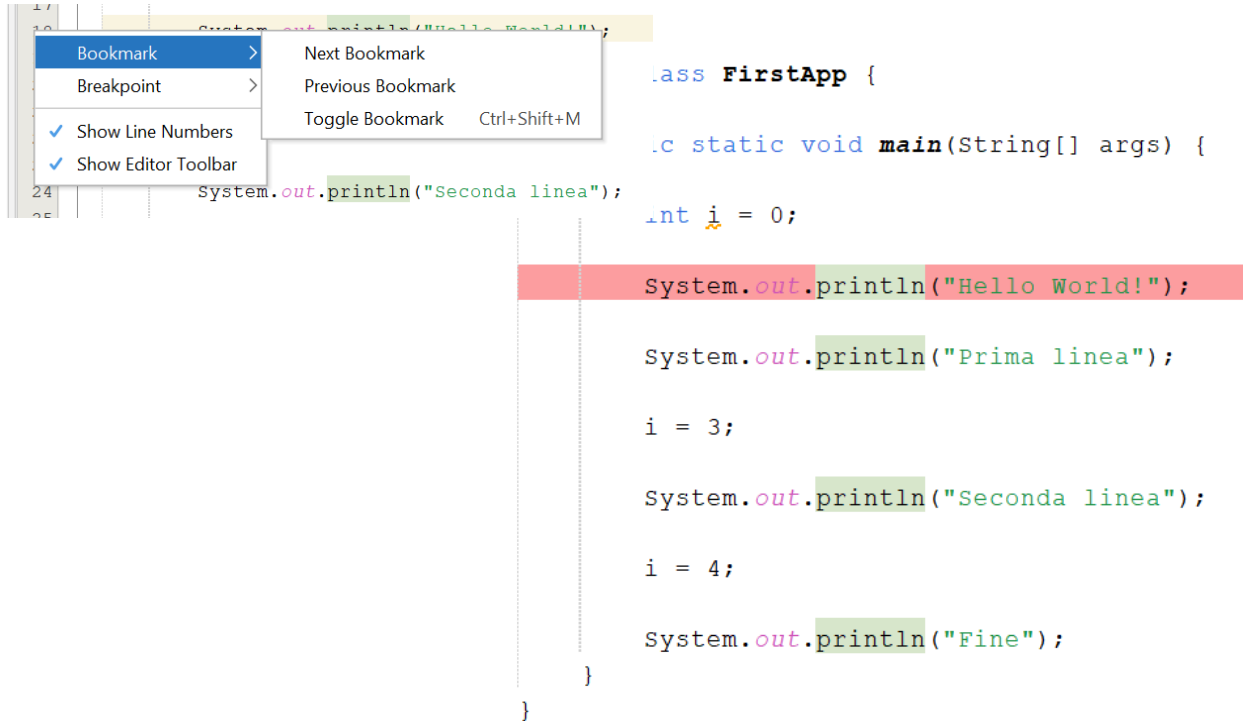


The screenshot shows the 'Output - Run (FirstApp)' window in NetBeans. The output log details the build process, including resource copying, compilation, and execution. The final output is 'Hello World!' and 'BUILD SUCCESS'.

```
Notifications | Test Results | Output - Run (FirstApp) X
Building FirstApp 1.0-SNAPSHOT
-----[ jar ]-----
--- maven-resources-plugin:2.6:resources (default-resources) @ FirstApp ---
Using 'UTF-8' encoding to copy filtered resources.
skip non existing resourceDirectory C:\Users\carlo\Documents\NetBeansProjects\FirstApp\src\main\resources
--- maven-compiler-plugin:3.1:compile (default-compile) @ FirstApp ---
Changes detected - recompiling the module!
Compiling 1 source file to C:\Users\carlo\Documents\NetBeansProjects\FirstApp\target\classes
--- exec-maven-plugin:3.0.0:exec (default-cli) @ FirstApp ---
Hello World!
-----
BUILD SUCCESS
-----
Total time: 2.093 s
Finished at: 2022-09-08T00:36:44-07:00
-----
```

Debug dell'applicazione

NetBeans ha integrata la gestione del debug dell'applicazione, per inserire un breakpoint prima dell'esecuzione basta cliccare con il tasto destro sulla barra a sinistra con i numeri delle righe:



Cosa è Maven

Maven è uno strumento di gestione di progetti SW basati su Java. Altro software con funzionalità simili e' Gradle.

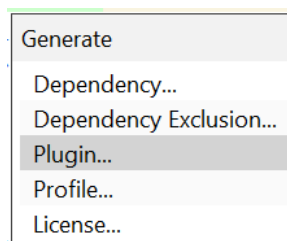
La particolarità di tools come Maven e Gradle è quella di poter gestire la compilazione di un progetto, in tutte le sue fasi, dalla gestione delle dipendenze all'esecuzione dei test per individuare eventuali errori.

Il sistema gestisce il flusso di compilazione seguendo le istruzioni riportate in un file 'pom.xml', creato nel nostro caso automaticamente e che gestisce tutti i passaggi:

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <project xmlns="http://maven.apache.org/POM/4.0.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
   4.0.0.xsd">
3.     <modelVersion>4.0.0</modelVersion>
4.     <groupId>it.unipi.firstapp</groupId>
5.     <artifactId>FirstApp</artifactId>
6.     <version>1.0-SNAPSHOT</version>
7.     <packaging>jar</packaging>
8.     <dependencies>
9.         <dependency>
10.             <groupId>org.apache.logging.log4j</groupId>
11.             <artifactId>log4j-core</artifactId>
12.             <version>2.18.0</version>
13.         </dependency>
14.     </dependencies>
15.     <properties>
16.         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
17.         <maven.compiler.source>11</maven.compiler.source>
18.         <maven.compiler.target>11</maven.compiler.target>
19.         <exec.mainClass>it.unipi.firstapp.FirstApp</exec.mainClass>
20.     </properties>
21. </project>
```

Aggiungere una dipendenza

Una dipendenza può essere aggiunta utilizzando tramite NetBeans. Dopo aver aperto il file 'pom.xml' si può usare il menu aperto tramite il tasto destro per poi selezionare "Dependency"



A questo punto dalla nuova interfaccia si possono riportare i dati della dipendenza. Le dipendenze sono gestite da Maven attraverso dei repository pubblici che contengono i binari delle librerie che possono essere scaricati e importati automaticamente da Maven al momento della prima compilazione.

Ogni pacchetto è individuato da un Group ID e da un Artifact ID che sono specificati dallo sviluppatore al momento della creazione della libreria e come è specificato nella app che state sviluppando.

Supponiamo di voler creare una app in grado di recuperare delle informazioni da un sito web attraverso la libreria HTTPClient che implementa il protocollo http (esempio <https://mkyong.com/java/apache-httpclient-examples/>). Il primo passo è di importare la libreria come dipendenza:

La libreria con le versioni disponibili possono essere consultate tramite il sito:

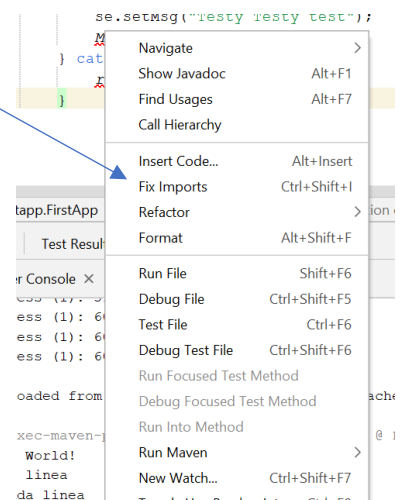
<https://mvnrepository.com/>

Una volta inserita la dipendenza si può cominciare ad usare la libreria per recuperare dei dati da un sito internet¹:

```
1.      CloseableHttpClient httpClient = HttpClients.createDefault();
2.
3.      HttpGet request = new HttpGet("https://www.google.com");
4.
5.      CloseableHttpResponse response = httpClient.execute(request);
6.
7.      // Get HttpResponse Status
8.      System.out.println(response.getProtocolVersion());           // HTTP/1.1
9.      System.out.println(response.getStatusLine().getStatusCode()); // 200
10.     System.out.println(response.getStatusLine().getReasonPhrase()); // OK
11.     System.out.println(response.getStatusLine().toString());       // HTTP/1.1 200 OK
12.
13.     HttpEntity entity = response.getEntity();
14.     if (entity != null) {
15.         // return it as a String
16.         String result = EntityUtils.toString(entity);
17.         System.out.println(result);
18.     }
19.
```

Questo codice recupera una pagina web (riga 7), google.com in questo caso, e ne stampa il contenuto a video (riga 20), stampando anche i dati della risposta web (righe 12-15).

La libreria per essere utilizzata correttamente ha bisogno di un import che può essere inserito automaticamente da NetBeans attraverso il menu “Fix Import”.



¹ Il codice per il recupero di una pagina web potrebbe lanciare delle eccezioni (le vedrete nelle lezioni successive nel dettaglio). Tale codice, quindi, richiede una dichiarazione esplicita di lancio di eccezione al momento della dichiarazione della funzione:

```
public static void main(String[] args) throws Exception {
```