Tutorial 0 — Installing the VS Code Extension

Overview

This INTO-CPS tutorial will show you how to:

1. Install the `Cosimulation Studio` extension for Visual Studio Code
2. Install the `Maestro` COE (Co-simulation Orchestration Engine) w. Web API

Requirements

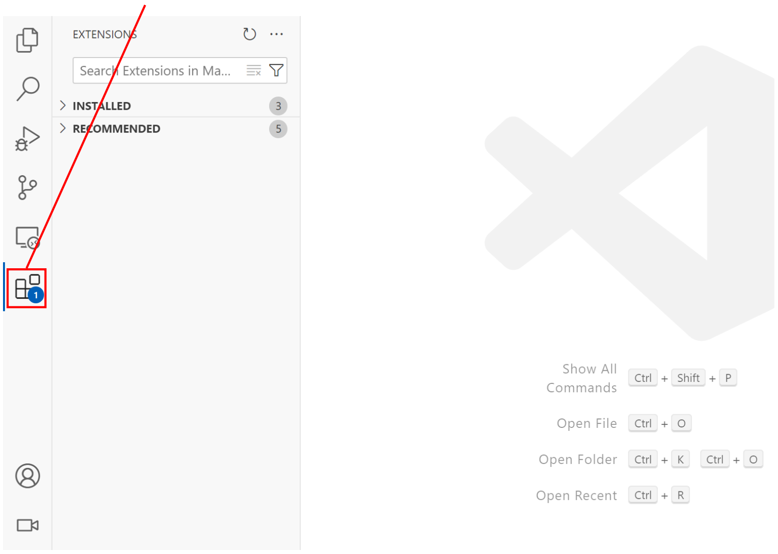
The tutorial assumes that you have the following pieces of software installed:

- Visual Studio Code. Can be installed from `https://code.visualstudio.com/`
- Java SE Runtime Environment 11, e.g. the latest version of the Temurin distribution from `https://adoptium.net/temurin/releases/?version=11`, making sure to select your corresponding operating system in the dropdown menu.

If any of these are missing from your system, it can make it difficult or impossible to follow certain steps of the tutorial, or some parts of the tool will not work, so ensure that the dependencies are properly installed before continuing. Unless otherwise stated, all the instructions in this tutorial are independent of whether you are using Linux, Windows or macOS.

1 Install the `Cosimulation Studio` extension for Visual Studio Code

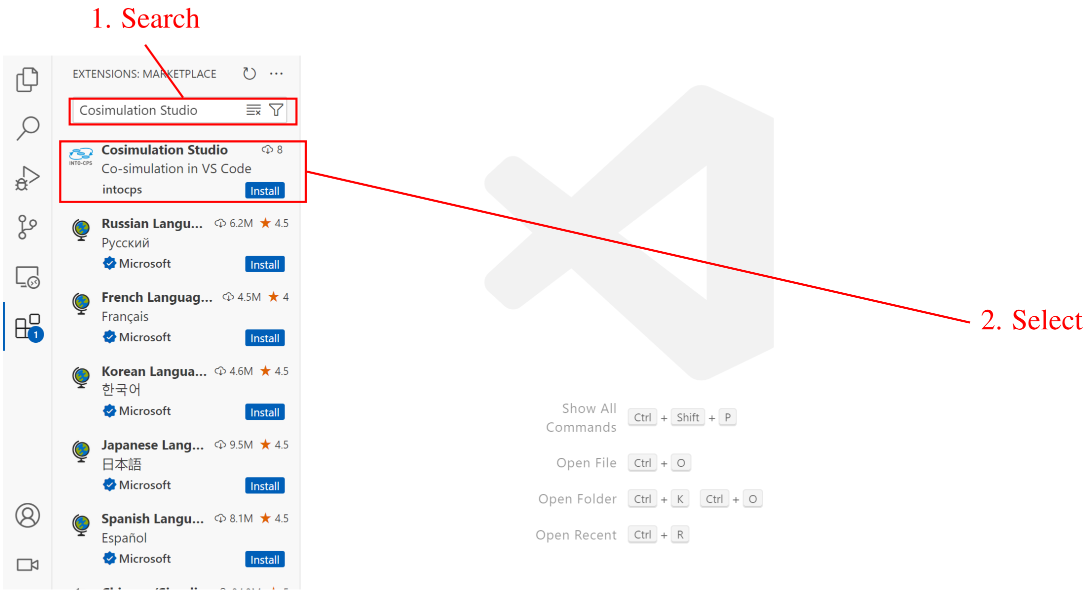
Step 1. To install the `Cosimulation Studio` extension, first open the `Extensions` view in Visual Studio Code.



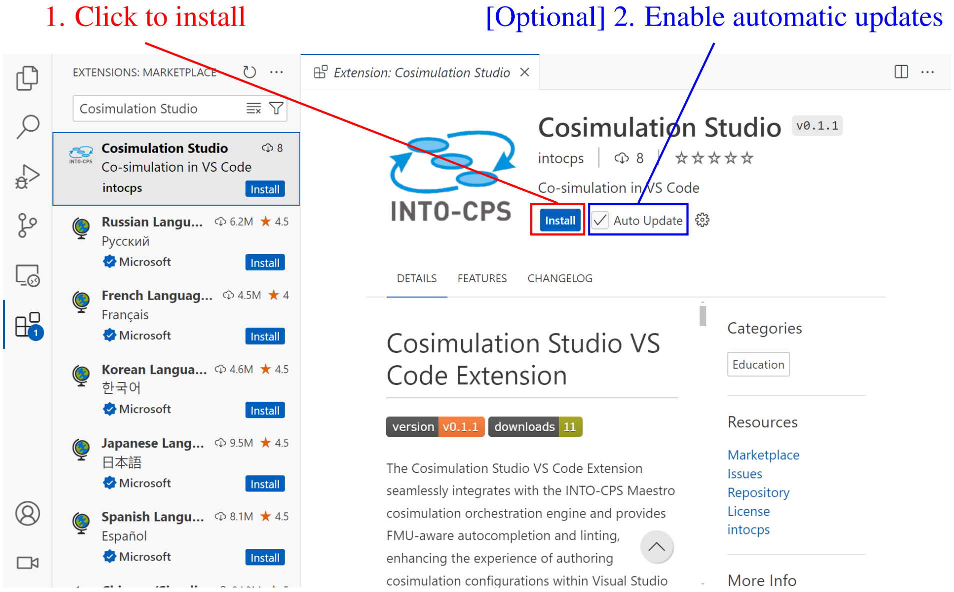
Open Extensions view

EXTENSIONS
Search Extensions in Ma...
INSTALLED 3
RECOMMENDED 5
1
Show All Commands Ctrl + Shift + P
Open File Ctrl + O
Open Folder Ctrl + K Ctrl + O
Open Recent Ctrl + R

Step 2. Then search for “Cosimulation Studio” and select the extension published by “intocps”.



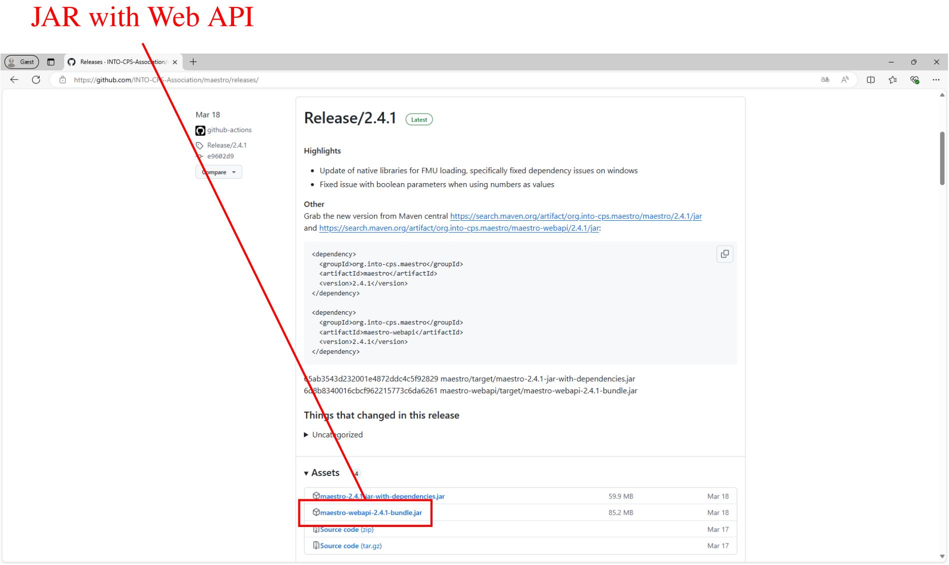
Step 3. Now install the extension. Optionally enable automatic updates to always get the most recent version of the extension, when new releases are published.



The Cosimulation Studio VS Code Extension seamlessly integrates with the INTO-CPS Maestro cosimulation orchestration engine and provides FMU-aware autocompletion and linting, enhancing the experience of authoring cosimulation configurations within Visual Studio

2 Install the Maestro COE (Co-simulation Orchestration Engine) w. Web API

Step 4. Find the latest release of `Maestro` on the GitHub Releases page at `https://github.com/INTO-CPS-Association/maestro/releases/`, and download the file named `maestro-webapi-<latest-version>-bundle.jar`.



JAR with Web API

Guest
Releases · INTO-CPS-Association/
https://github.com/INTO-CPS-Association/maestro/releases/
Mar 18
github-actions
Release/2.4.1
e9602d9
Compare

Release/2.4.1 Latest

Highlights

- Update of native libraries for FMU loading, specifically fixed dependency issues on windows
- Fixed issue with boolean parameters when using numbers as values

Other

Grab the new version from Maven central https://search.maven.org/artifact/org.into-cps.maestro/maestro/2.4.1/jar and https://search.maven.org/artifact/org.into-cps.maestro/maestro-webapi/2.4.1/jar:

```
<dependency>
  <groupId>org.into-cps.maestro</groupId>
  <artifactId>maestro</artifactId>
  <version>2.4.1</version>
</dependency>

<dependency>
  <groupId>org.into-cps.maestro</groupId>
  <artifactId>maestro-webapi</artifactId>
  <version>2.4.1</version>
</dependency>
```

65ab3543d232001e4872ddc4c5f92829 maestro/target/maestro-2.4.1-jar-with-dependencies.jar
6d8b8340016cbcf962215773c6da6261 maestro-webapi/target/maestro-webapi-2.4.1-bundle.jar

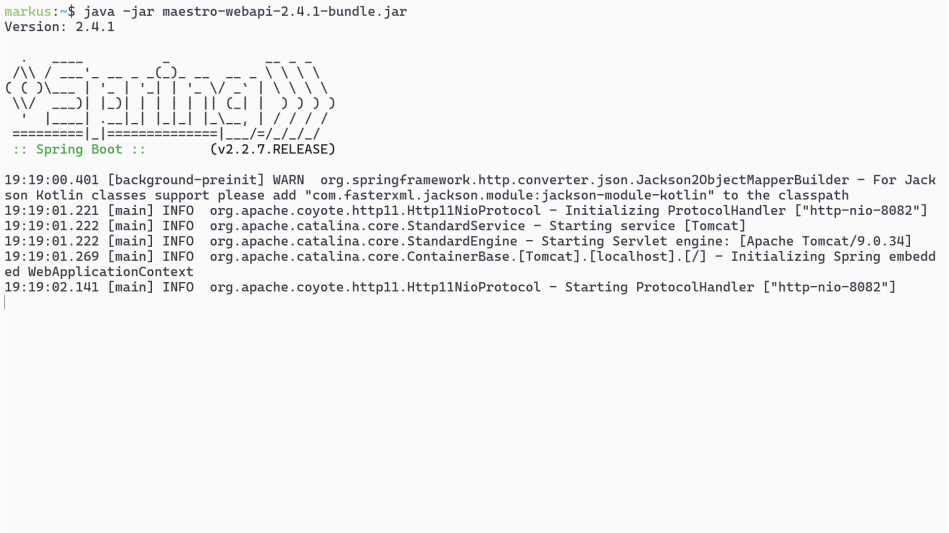
Things that changed in this release

▸ Uncategorized

▾ Assets 4

maestro-2.4.1-jar-with-dependencies.jar	59.9 MB	Mar 18
maestro-webapi-2.4.1-bundle.jar	85.2 MB	Mar 18
Source code (zip)		Mar 17
Source code (tar.gz)		Mar 17

Step 5. To verify that your installed version of Java works with the downloaded JAR, run `java -jar maestro-webapi-<latest-version>-bundle.jar` in a terminal. You should see something similar to the image below:



```
markus:~$ java -jar maestro-webapi-2.4.1-bundle.jar
Version: 2.4.1

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.2.7.RELEASE)

19:19:00.401 [background-preinit] WARN  org.springframework.http.converter.json.Jackson2ObjectMapperBuilder - For Jack
son Kotlin classes support please add "com.fasterxml.jackson.module:jackson-module-kotlin" to the classpath
19:19:01.221 [main] INFO  org.apache.coyote.http11.Http11NioProtocol - Initializing ProtocolHandler ["http-nio-8082"]
19:19:01.222 [main] INFO  org.apache.catalina.core.StandardService - Starting service [Tomcat]
19:19:01.222 [main] INFO  org.apache.catalina.core.StandardEngine - Starting Servlet engine: [Apache Tomcat/9.0.34]
19:19:01.269 [main] INFO  org.apache.catalina.core.ContainerBase.[Tomcat].[localhost].[/] - Initializing Spring embedd
ed WebApplicationContext
19:19:02.141 [main] INFO  org.apache.coyote.http11.Http11NioProtocol - Starting ProtocolHandler ["http-nio-8082"]
```

Step 6. As a final check, with `Maestro` running in the background, open `http://localhost:8082/ping` in your browser. You should see an almost blank page with only the text “OK” to indicate that `Maestro` is running properly.

Congratulations!

You have now successfully installed the `Cosimulation Studio` extension for Visual Studio Code and `Maestro`. You can now move on to the next tutorial that will describe how to set up and run your first simulation.