Lab 1

This introductory lab is composed of **two main tasks**.
Your **first objective** is to run your first MapReduce application on the BigData@Polito cluster. For this goal, you must learn how to import a predefined project, compile the source code, produce a jar file, copy your application on a remote machine (the gateway) and submit your application on the BigData@Polito cluster.
The **second objective** is to write your first MapReduce application.

1. Compiling using the VSCode IDE

In this task, we will “compile” the source code of a simple Hadoop application and we will create a jar file with the class files of the project. The application is the MapReduce implementation of the **word count application**.

We assume you are using a **PC from the Lab to generate the *.jar* file** (and with the same instructions, you can also do it with your own PC).

Download from the course web page the zip file Lab1_BigData_with_libraries.zip, which contains the MapReduce-based implementation of the word count application and the example data folder example_data. The shared VSCode project contains the basic libraries that are needed to develop the application and create the class and jar files.

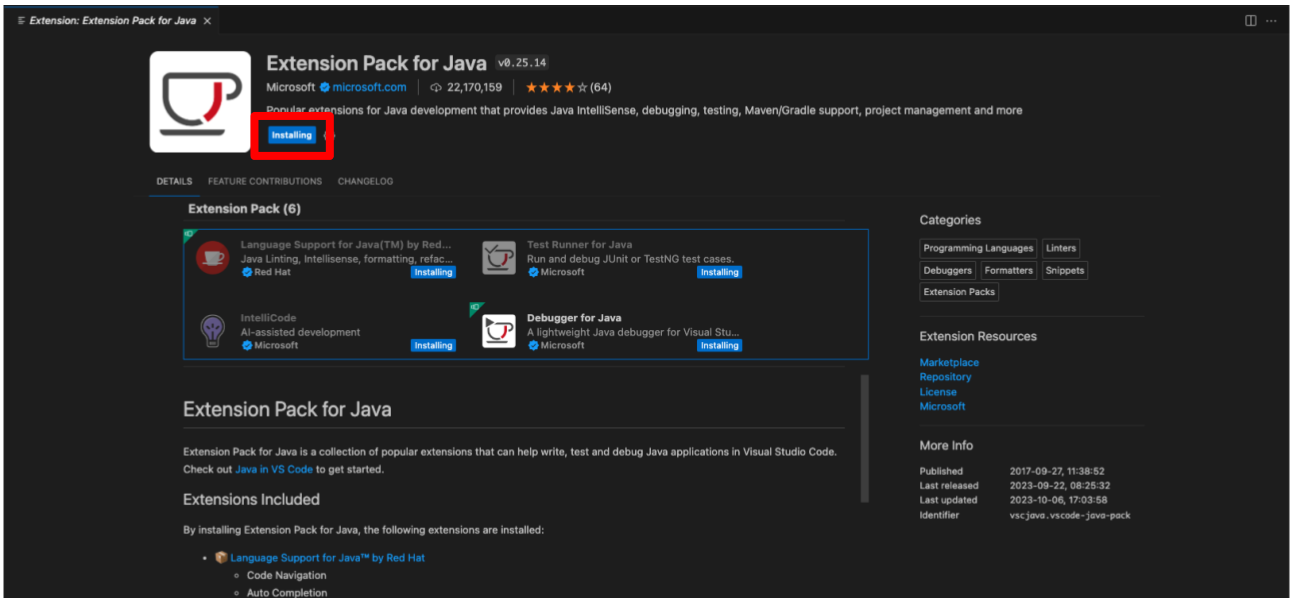
--- **Notice**
You **cannot** use this project to run the MapReduce application locally on the PC of the Lab/your own PC. For **local runs on your PC**, please download the Maven versions of the project (Windows, Mac/Linux) after following the guideline instructions (+extra for Windows users).

Decompress the zip file inside a local folder on the PC of the LAB or on your PC.
In the following, you will find the necessary steps to import the project in VSCode. If needed, we also provide a more complete guide here.

Steps:

1. Open VSCode (if not installed and you want to run the application on your own PC, download here)

2. Go under Extension tab and check whether “Extension Pack for Java” is installed
3. If not installed, you can click on the “Install” button in the page associated to the extension. Refer to the guide to also install JDK.


Extension: Extension Pack for Java

Extension Pack for Java v0.25.14
Microsoft microsoft.com | 22,170,159 | ★★★★☆ (64)
Popular extensions for Java development that provides Java IntelliSense, debugging, testing, Maven/Gradle support, project management and more
Installing

DETAILS FEATURE CONTRIBUTIONS CHANGELOG

Extension Pack (6)

Language Support for Java(TM) by Red...
Java Linting, Intellisense, formatting, refac...
Red Hat — Installing

Test Runner for Java
Run and debug JUnit or TestNG test cases.
Microsoft — Installing

IntelliCode
AI-assisted development
Microsoft — Installing

Debugger for Java
A lightweight Java debugger for Visual Stu...
Microsoft — Installing

Extension Pack for Java

Extension Pack for Java is a collection of popular extensions that can help write, test and debug Java applications in Visual Studio Code. Check out Java in VS Code to get started.

Extensions Included

By installing Extension Pack for Java, the following extensions are installed:

- Language Support for Java™ by Red Hat
  - Code Navigation
  - Auto Completion

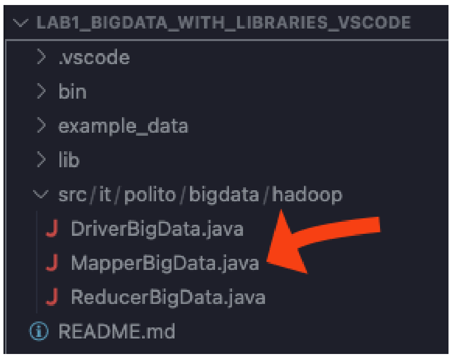
Categories
Programming Languages, Linters, Debuggers, Formatters, Snippets, Extension Packs

Extension Resources
Marketplace
Repository
License
Microsoft

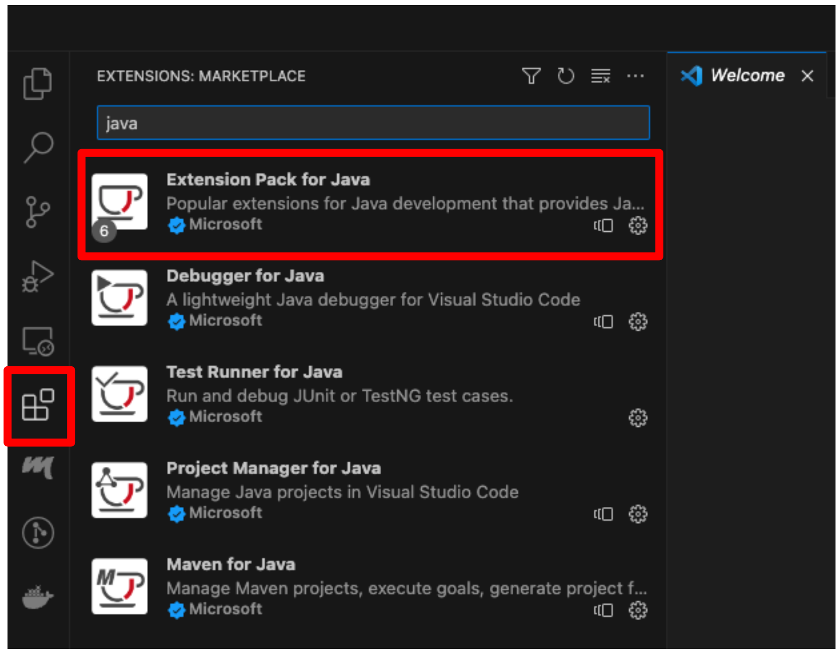
More Info

Published	2017-09-27, 11:38:52
Last released	2023-09-22, 08:25:32
Last updated	2023-10-06, 17:03:58
Identifier	vscjava.vscode-java-pack

4. Simply Open the folder containing the project under File -> “Open folder…” or “Open folder…” in the initial screen. After loading, you shall see a “Jave Projects” item in the lower-left part of VSCode. If you do not, single-click on one of the “.java” files under “src”.


```
LAB1_BIGDATA_WITH_LIBRARIES_VSCODE
  > .vscode
  > bin
  > example_data
  > lib
  v src / it / polito / bigdata / hadoop
      DriverBigData.java
      MapperBigData.java
      ReducerBigData.java
  README.md
```

5. ***Question***: Have a look at the source files and the structure of the project. Where is the mapper? Where is the reducer?
6. Under the JAVA PROJECTS tab (bottom-left), check if the ***JRE System Library*** is set to **JAVASE-1.8**. If not, click on the three dots next to "JAVA PROJECTS" (More


EXTENSIONS: MARKETPLACE

java

Extension Pack for Java
Popular extensions for Java development that provides Ja...
Microsoft

Debugger for Java
A lightweight Java debugger for Visual Studio Code
Microsoft

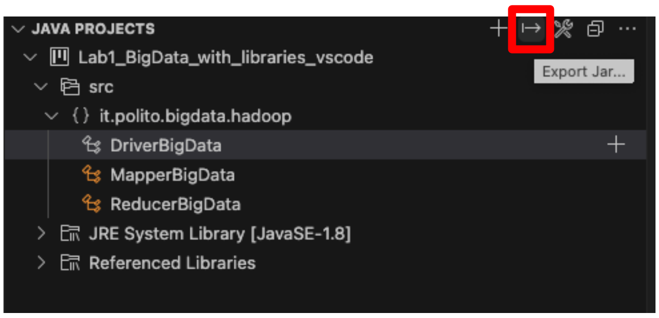
Test Runner for Java
Run and debug JUnit or TestNG test cases.
Microsoft

Project Manager for Java
Manage Java projects in Visual Studio Code
Microsoft

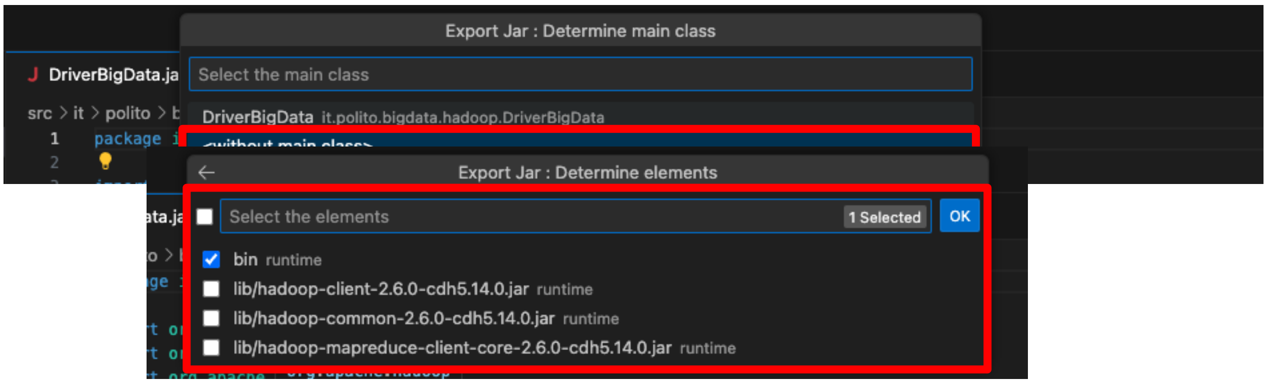
Maven for Java
Manage Maven projects, execute goals, generate project f...
Microsoft

Actions), select the "*Configure Java Runtime*" option, and, finally, set the "Java Version" field to **Java 8/1.8**.

7. To generate a ".jar" file, **use the "Export jar…" command**, as in the following steps.
    a. In the lower-left part of VSCode, in the Java Project menu, select the Export Jar button (right arrow symbol)



    b. In the upper area, in the center, VSCode, will ask you to choose the main class for your project. Select <without main class>

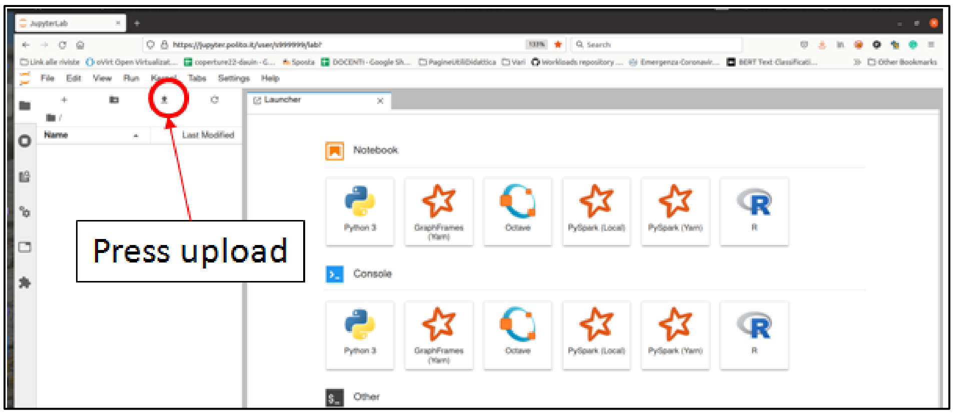

Export Jar : Determine main class
Select the main class
DriverBigData it.polito.bigdata.hadoop.DriverBigData
<without main class>
Export Jar : Determine elements
Select the elements
1 Selected
OK
bin runtime
lib/hadoop-client-2.6.0-cdh5.14.0.jar runtime
lib/hadoop-common-2.6.0-cdh5.14.0.jar runtime
lib/hadoop-mapreduce-client-core-2.6.0-cdh5.14.0.jar runtime

    c. Next, you will be prompted to add all the files that your jar file should contain. Only keep the "bin" folder (the other libraries are already present on our cluster – global exporting would take an unnecessary long time).
    d. You will find the jar file in the main folder of your project. You can upload such jar file directly on jupyter.polito.it cluster

2. Upload your application on BigData@Polito

The objective of this task is to upload your application on the cluster BigData@Polito.

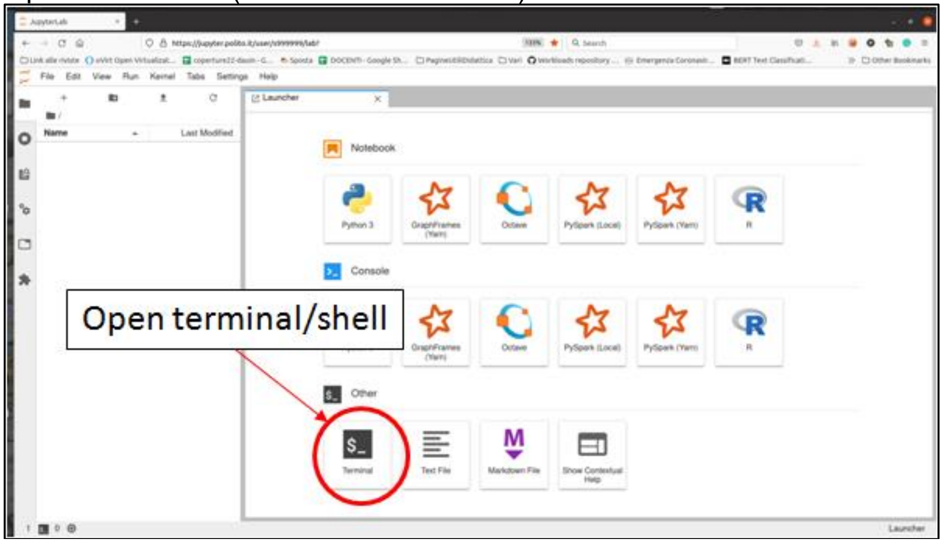
1. Connect to https://jupyter.polito.it and log in (**credentials**: check your email)
    - You are now inside the BigData@Polito cluster
    - Specifically, you are connected to a *gateway* machine.
2. Upload the jar file containing your application on the local file system of the gateway
    - **Note:** we recommend organizing your `home` directory
    - *e.g.*, create a folder *Lab_01*



3. Explore the BigData@Polito

Besides your **home** folder, our cluster also contains the datasets for this and the other labs.

1. Open a *Terminal* (from Launcher above)



   - By default, the terminal opens in your current directory
2. Change directory with
   `cd /share/students/bigdata/Dati/Lab1/example_data`
3. ***Question:*** What is the content of the folder? What is the size of each file?
   - **Hint**: use `ls -lh` to see the content of a folder in a human-readable format
4. Copy the path leading to `example_data`
   - **Hint**: use `pwd` to get the full path of the current directory

4. Install Hadoop

Before starting, we still have to install Hadoop in your cluster's profile.
P.S. don't worry. It's a one-time process :P

1. Go back to your `$HOME` folder typing `cd $HOME`
2. Download the Hadoop installation file:

```
wget 'https://dlcdn.apache.org/hadoop/common/hadoop-3.4.0/hadoop-
3.4.0.tar.gz'
```

3. Decompress the file: `tar -xvzf hadoop-3.4.0.tar.gz`
4. Lastly, create an alias for Hadoop (it will be simpler to run it later on):

```
echo "alias hadoop='$HOME/hadoop-3.4.0/bin/hadoop'" >> ~/.bashrc
source ~/.bashrc
```

5. Submit a job

Now we have everything we need to submit our sample application.

1. Open a new terminal.
   - Make sure that you are in the same folder where you loaded the *.jar* file
2. Now that you have a terminal on the gateway, launch a MapReduce job (i.e., run your application on the cluster) using the following command:

```
hadoop jar <your_jar.jar> \
it.polito.bigdata.hadoop.DriverBigData 2 <path_lab1_data> ex1_out
```

   where:
   - `<your_jar.jar>` is the JAR file containing your application (**change name**)
   - `<path_lab1_data>` is the input folder you copied in 3.4 (shall start with `/share/`).
   - "**`ex1_out`**" is the output folder.
3. ***Question:*** Check the number of mappers (i.e., the number of instances of the mapper class)
   - You can retrieve the information about the number of mappers (map tasks) and other statistics in the information showed on the terminal during the execution of your application on the cluster.
4. ***Question:*** What is inside **`ex1_out`**? How many files do you see? Why do you think it is the case?
5. ***Question:*** Try to re-run the same job. Does it succeed this time? If not, what is the problem?
6. ***Question:*** Remove *ex1_out* (`rm -r ./ex1_out`) and re-run now. Does it work?

7. ***Question:*** Run again the application and change the number of reducers (first parameter of the application, **set it to 1**). Analyze the content of the output folder. How many files do you see now?

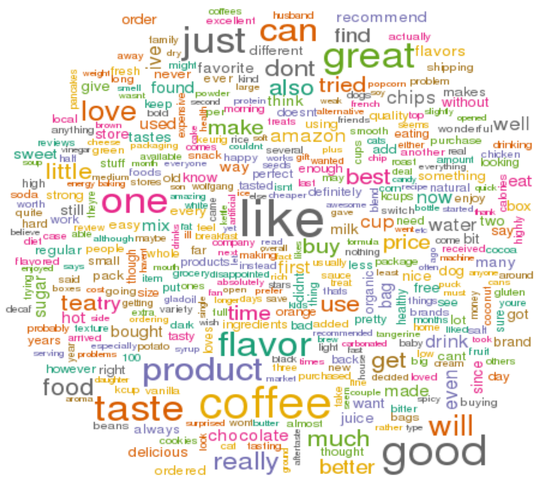
6. Test on a bigger file

Now you will execute your application on a larger. You can find such file at:

```
/share/students/bigdata/Dati/Lab1/finefoods_text.txt
```

It is a **large collection of Amazon reviews** in the food category. Each line of finefoods_text.txt contains one review.

1. Launch your application on the Amazon review file:
    a. Set the second parameter of your application to `/share/students/bigdata/Dati/Lab1/finefoods_text.txt`
2. ***Question:*** Analyze the results.
    a. Can you understand any interesting facts from your results?
    b. Do you see any space for improvements in your analysis?
3. ***Question:*** The following figure was done on a small sample of your data (10_000 reviews).
    a. Is it consistent with what you found on the complete dataset?
    b. Do you think a small sample is enough to represent the whole?



7. Bonus track

Definition: A word count can be seen as a special case of an *n-gram* count, where n is equal to 1. *n-grams*, in our context, are sets of contiguous words of length n.

For example, in the sentence “She sells seashells by the seashore“, 2-grams are: “She sells”, “sells seashells”, “seashells by”, “by the”, “the seashore”.

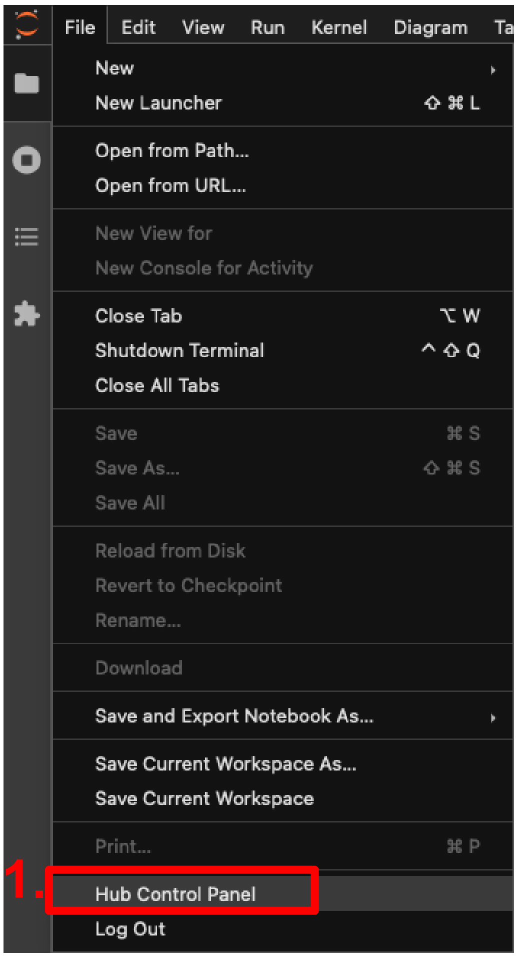
Question: Modify your word count *.java* program to count 2-grams frequencies. Consider each line of text as a separate document, as in the Amazon reviews file (so, do not count as contiguous words on separate lines).

1. What is the time/space complexity of your program?
2. How long would you expect it to run, compared to the simple word count?
3. Try to run it on the toy text and on the Amazon reviews.

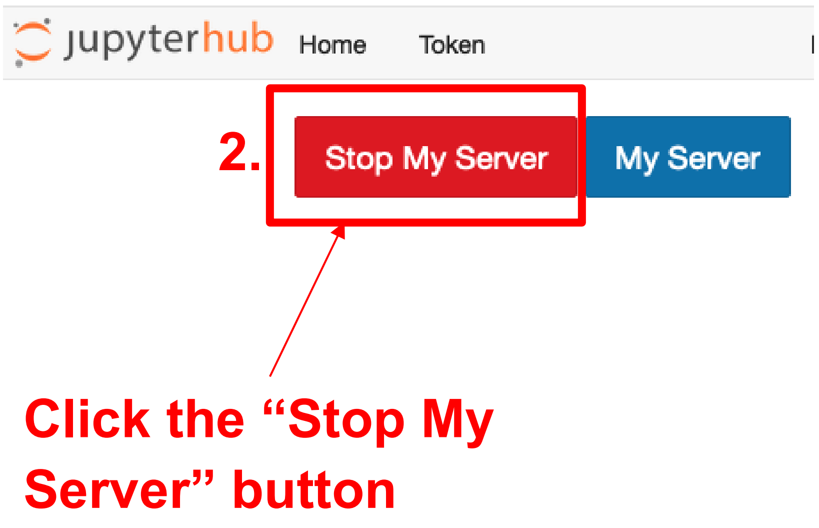
⚠ ⚠ ⚠ Shut down JupyterHub container ⚠ ⚠ ⚠

As soon as you complete all the tasks and activities on JupyterHub environment, please remember to shut down the container to let all your colleagues in all the sessions connect on JupyterHub and do all the lab activities.

1. Go into File -> Hub Control Panel menu
2. A new browser tab opens with the "Stop My Server" button. Click on it and wait till it disappears.



File
Edit
View
Run
Kernel
Diagram
Ta
New
New Launcher ⇧ ⌘ L
Open from Path...
Open from URL...
New View for
New Console for Activity
Close Tab ⌥ W
Shutdown Terminal ^ ⇧ Q
Close All Tabs
Save ⌘ S
Save As... ⇧ ⌘ S
Save All
Reload from Disk
Revert to Checkpoint
Rename...
Download
Save and Export Notebook As...
Save Current Workspace As...
Save Current Workspace
Print... ⌘ P
1.
Hub Control Panel
Log Out



jupyterhub
Home
Token
2.
Stop My Server
My Server
Click the "Stop My Server" button