In this tutorial, we will learn how to create a GitHub repository for a Java Maven project and push the local project files to GitHub using PowerShell.

This guide is useful for students, developers, and researchers who want to manage their compiler, Java, or Maven-based projects using Git and GitHub.

Prerequisites

Before starting, make sure you have:

1. Git installed on your system
2. A GitHub account
3. A local Java/Maven project folder
4. PowerShell or terminal access

Step 1: Open the Project Folder

First, open PowerShell and move to your project directory.

```
cd C:\Users\YourName\Downloads\YourProjectFolder
```

```
```

Example:

```
cd C:\Users\umrej\Downloads\promptsyntax_compiler
```

Step 2: Initialize Git

Run the following command:

```
git init
```

This creates a hidden `.git` folder and converts your project into a Git repository.

If you see a message like:

```
Reinitialized existing Git repository
```

it means Git was already initialized earlier. This is not a problem.

Step 3: Create a `.gitignore` File

For Java Maven projects, the `target/` folder should not be uploaded to GitHub because it contains generated build files.

Create a `.gitignore` file:

```
notepad .gitignore
```

Paste the following content:

```
# Maven
target/

# Java
*.class

# IDE
.idea/
.vscode/

# macOS
.DS_Store
```

Save and close Notepad.

Step 4: Add Files to Git

Now add all project files:

```
git add .
```

This stages all files for commit.

Step 5: Configure Git Username and Email

If this is your first time using Git, configure your name and email:

```
git config --global user.name "Your Name"
git config --global user.email "your_email@example.com"
```

Example:

```
git config --global user.name "Jayesh Umre"
git config --global user.email "umarejayesh@gmail.com"
```

Step 6: Commit the Project

Create your first commit:

```
git commit -m "Initial project commit"
```

Example output:

```
[main (root-commit) abc1234] Initial project commit
```

This means your first Git commit was created successfully.

Step 7: Rename Branch to Main

GitHub commonly uses `main` as the default branch name.

Run:

```
git branch -M main
```

Step 8: Create GitHub Repository

Go to GitHub and create a new repository.

Example repository name:

```
PromptSyntax
```

Copy the repository URL. It will look like this:

```
https://github.com/username/PromptSyntax.git
```

Step 9: Connect Local Project to GitHub

Run:

```
git remote add origin https://github.com/username/RepositoryName.git
```

Example:

```
git remote add origin https://github.com/umarejayesh/PromptSyntax.git
```

If you get this error:

```
remote origin already exists
```

then update the remote URL:

```
git remote set-url origin
https://github.com/username/RepositoryName.git
```

Step 10: Push Code to GitHub

Now push your local code:

```
git push -u origin main
```

```
```

If GitHub asks for authentication, complete the login in your browser.

Common Problem: Push Rejected

Sometimes you may see:

```
! [rejected] main -> main (fetch first)
```

This means the GitHub repository already contains some files, such as a README or license.

If the remote repository is new and does not contain important work, you can force push:

```
git push --force -u origin main
```

Use force push only when you are sure that the remote content is not important.

Common Problem: `target/` Still Appears in Git Status

If you see:

```
Untracked files:
    target/
```

then check your `.gitignore` file:

```
type .gitignore
```

Make sure it contains:

```
target/
*.class
.idea/
.vscode/
.DS_Store
```

If `target/` was already committed earlier, remove it from Git tracking:

```
git rm -r --cached target
git commit -m "Remove Maven target directory from tracking"
git push
```

Step 11: Verify Repository Status

Finally, run:

```
git status
```

Expected output:

```
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

This means your local project and GitHub repository are synchronized.

Final Result

You have successfully:

- Initialized Git

- Created `.gitignore`
- Added and committed files
- Connected the project to GitHub
- Pushed the project to GitHub
- Removed Maven build files from Git tracking

Complete Command Summary

```
cd C:\Users\YourName\Downloads\YourProjectFolder

git init

notepad .gitignore

git add .

git config --global user.name "Your Name"
git config --global user.email "your_email@example.com"

git commit -m "Initial project commit"

git branch -M main

git remote add origin https://github.com/username/RepositoryName.git

git push -u origin main
```

If remote already exists:

```
git remote set-url origin
https://github.com/username/RepositoryName.git
```

If push is rejected because the remote has existing files:

```
git push --force -u origin main
```

Conclusion

Git and GitHub are essential tools for managing software projects. For Java Maven projects, always remember to exclude the `target/` directory using `.gitignore`. This keeps your repository clean and professional.

Using GitHub also makes your project easier to share, collaborate on, and maintain over time.

Related posts:

1. Can Java have same name variable
2. Types of variables in Java programming
3. JAVA and its Support Systems
4. JAVA environment
5. JAVA program structure
6. Tokens

7. Java statements
8. Java virtual machine
9. C++ Versus JAVA
10. Constants and Variables in Java
11. Data types JAVA
12. Defining a class
13. Constructor in JAVA
14. Array in Java
15. Applet
16. Applets Vs Applications
17. Writing applets
18. Applets life cycle
19. Creating an Executable Applet
20. Graphics in Applet
21. Applet image display
22. Applet digital clock
23. Applet mouse event handling
24. JDBC
25. Execute an SQL Statement
26. Process the result
27. CLOSE THE DATABASE CONNECTION
28. File handling
29. Define a class to declare an integer array of size n and accept the elements into the array.
30. Define a class to declare an array of size 20 of the double datatype, accept the elements into the array and perform the following: Calculate and print the sum of all the elements.

31. Java program for String, to uppercase, to equal, length of string
32. Write a Java program for Buble sort.
33. Write a Java program String to uppercase and count words startig with ‘A’
34. How to set path in Java
35. Understanding public static void main (String args[]){ } in Java
36. Difference between static and non static methods in Java