Detailed ESA-GODOT Conda Installation Guide for WSL and MacOS

Buyanbileg Amarsanaa

Student Researcher in Spacecraft Trajectory Analysis,
Mechanical Engineering Student, SDU-Galaxy,
University of Southern Denmark
alama24@student.sdu.dk

Mikkel Kielstrup Holst

SDU Faculty of Natural Science/ SDU-Galaxy

Tobias Cornelius Hinse

Astronomer/Astrophysicist, SDU-Galaxy,
Department of Physics, Chemistry and Pharmacy,
University of Southern Denmark
toch@cp3.sdu.dk

June 27, 2025

Denmark

Disclaimer

This document is for informational and educational purposes only. While efforts have been made to ensure accuracy, the author and contributors do not guarantee functionality across all system configurations. Users proceed at their own risk, and the author and contributors are not liable for issues, data loss, or damages. ESA-GODOT is developed by the European Space Agency (ESA). This guide is not official ESA documentation; refer to official ESA resources for updates and support. Ensure compatibility, security, and proper licensing when using ESA-GODOT.

Contents

ESA-GODOT Workshop: Conda Setup Homework

Deadline: Complete by the 3rd of July 2025

Prepare your **Conda environment** for the ESA-GODOT astrodynamics workshop which requires a Linux-based system (Windows: WSL2; MacOS: Unix) because it smoothens the workshop flow by adding more time to solely focus on the use of the ESA GODOT.

NOTE: **Acquire your personal token via https://gitlab.space-codev.org prior to the workshop.**

Please follow the condensed ESA-GODOT Conda Installation steps:

- **Windows OS**: Install WSL2 + Ubuntu 22.04 (Page 12-13), Miniconda (Page 14), `godotdev` environment with Python 3.10 (Pages 12-13) and Jupyter Notebook (Pages 19-20)
- **MacOS**: Install Miniconda for Intel/Apple Silicon (Page 16), `godotdev` environment with Python 3.10 (Pages 16-17) and Jupyter Notebook (Pages 19-20)

  Nota Bene: 5-10 GB free disk space; Windows 10/11 for WSL2; Intel or Apple Silicon MacOS

  Verify: Please run the following:

```
conda --version   # Expect: conda 24.9.2 or similar
conda activate godotdev
python --version # Expect: Python 3.10.X
jupyter notebook # Test random number script (Page 14)
```

Troubleshooting: Please see the related chapters (Pages 14-19)(e.g., `source~/miniconda3/bin/activate` if Conda fails). Email alama24@student.sdu.dk for help.

Bring your laptop with the setup ready. We will verify and proceed with ESA-GODOT installation.

Chapter 1

Obtainment of the ESA GODOT Personal Token

The `esa-godot` package is hosted on a Gitlab repository managed by the European Space Agency abbrievated by ESA at https://gitlab.space-codev.org/ requiring a personal access token for authentication during installation. This token acts as a secure key to access the package, and you are required to create an account and generate a token before proceeding with the installation on Linux via WSL or MacOS depending on the operating system.

Hence, follow these steps carefully to obtain your token, and handle with care to protect your account and project environment.

Important Security Notes:

- **Do Not Share Your Token**: You have to treat your personal access token like a password because sharing your personal token risks unauthorised access to your GitLab account and the `esa-godot` repository.
- **Save Your Token Immediately**: You can only view the token **once** upon creation. If lost, you will need to generate a new one, which may result in disrupting your setup.
- **Use a Secure Storage Method**: Copy the token to a secure location such as a password manager or a private text file to avoid any potential accidental loss.

The following sections introduce step-by-step instructions for obtaining the personal token in a systematic manner.

1.1 Step 1: Create a GitLab Account

- Navigate to GitLab: Open a web browser and go to https://gitlab.space-codev.org/. This is the official login page for the ESA-GODOT repository intendeded for the users outside the ESA framework.
- Sign Up: If you do not have an account, click the "**Register**" or "**Sign Up**" link (typically below the login fields).
  - Enter your email address, username and a strong password
  - Follow any verification steps (e.g., email confirmation) to activate your account
  - Expected Interface: Look for a registration form with fields for email, username and password, similar to standard web sign-up pages.

- Sign In: Once you are registered, please return back to https://gitlab.space-codev.org/ to enter your credentials, and click **Sign In**.
  - Expected Output: You will be automatically redirected to your GitLab dashboard showing your profile or project overview.

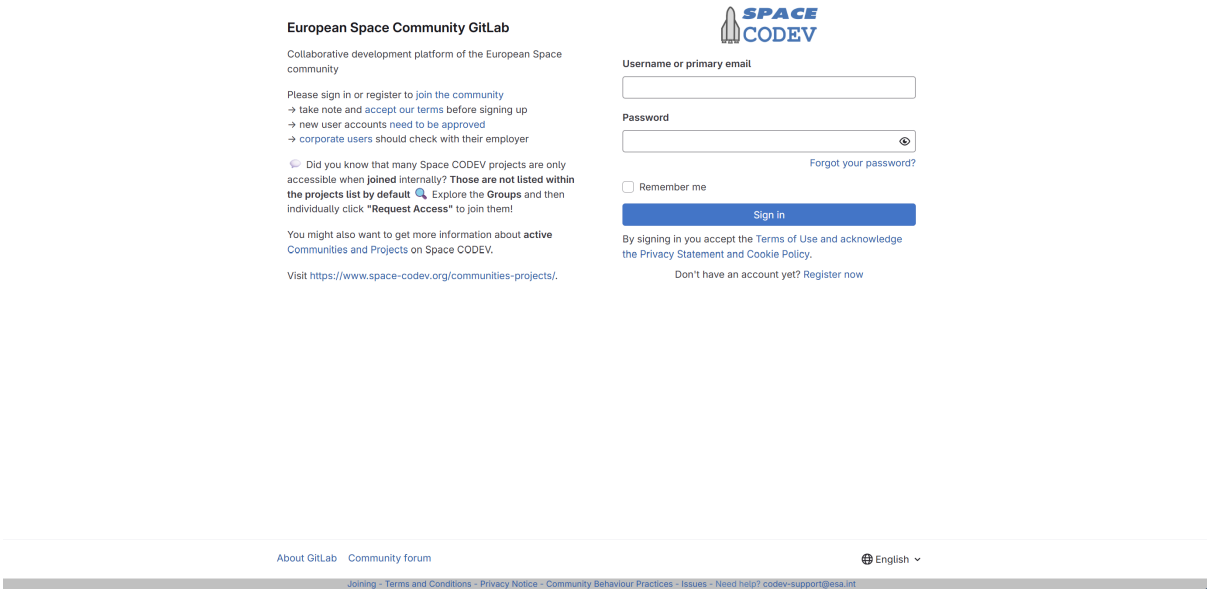

Figure 1.1: Entry of the SPACECODEV

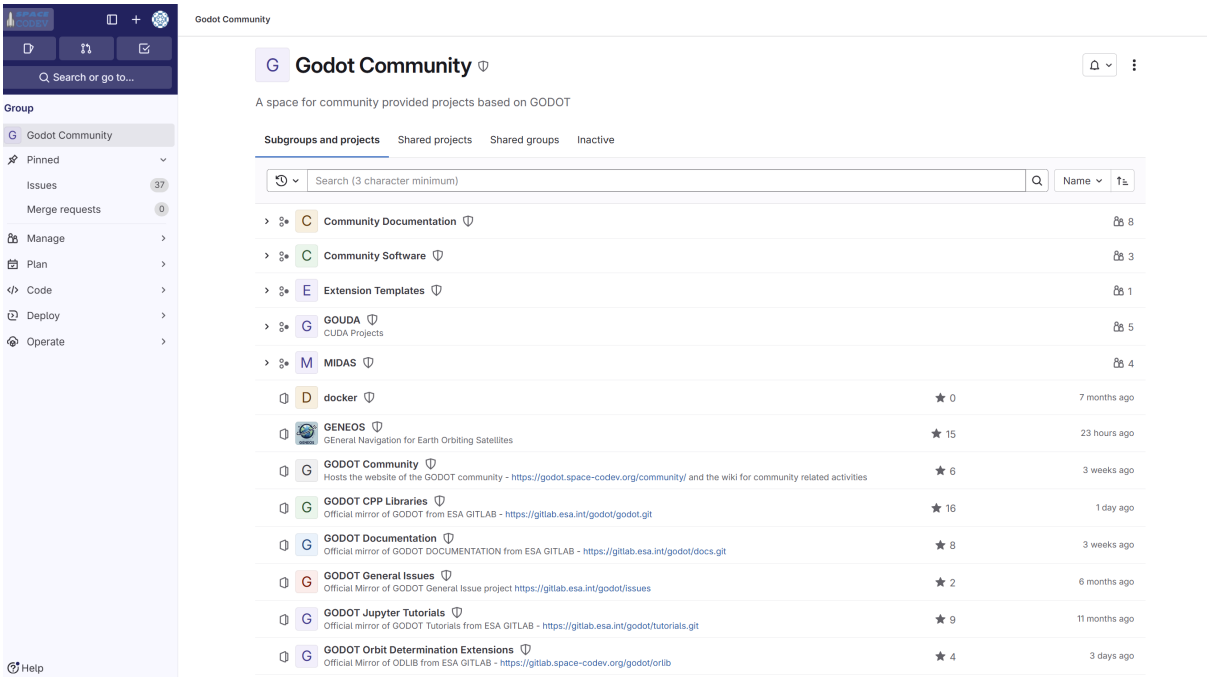

Figure 1.2: Expected Outcome after being redirected to ESA GODOT GitLab repository

1.2 Step 2: Navigate to User Settings

- Access Your Profile: After signing in as shown in Figure 1.2, locate your user profile in the top-right corner of the GitLab interface.
  - Click the **profile icon** or your username indicated by a red arrow in Figure 1.3 below.
  - Select "**Edit Profile**" or "**Settings**" from the dropdown menu.
  - Expected Interface: A dropdown menu with options like "Edit Profile", "Settings" or "Sign Out".

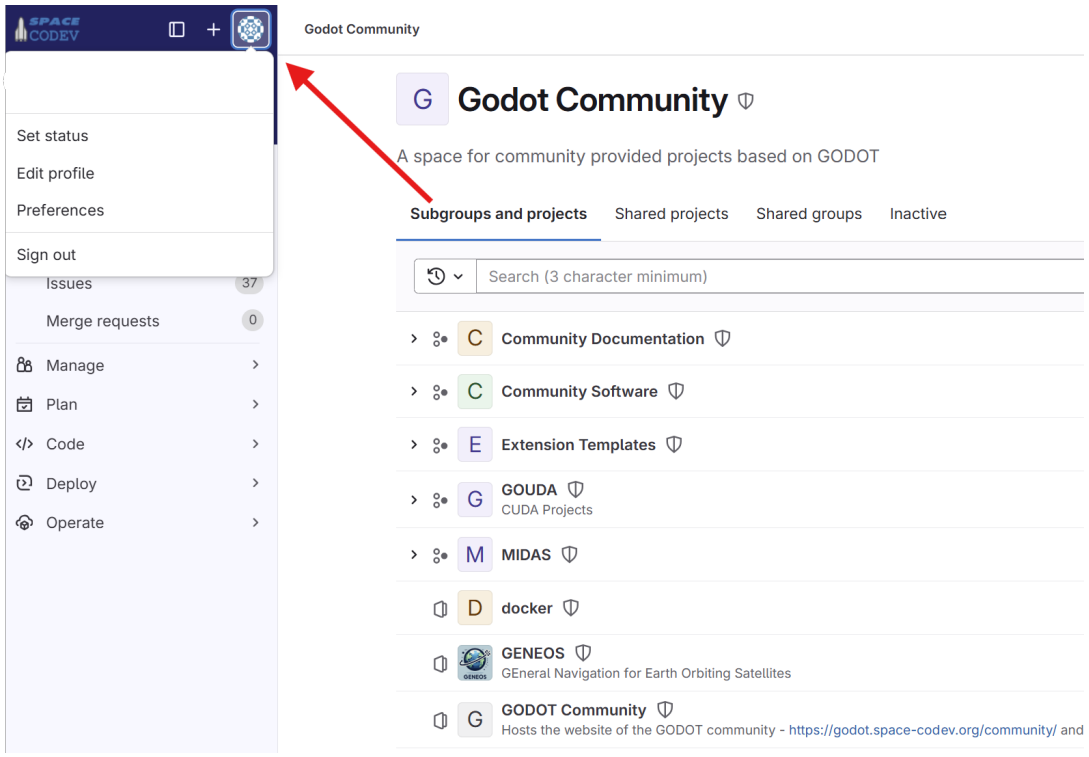

Figure 1.3: User setting interface on GitLab repository

- Go to Personal Access Tokens: In the user setting menu in the left hand side, find and click the "**Access Tokens**" or "**Personal Access Tokens**" as shown below in Figure 1.4
  - Expected Interface: A page titled "Personal Access Tokens" with a button or link to create a new token.

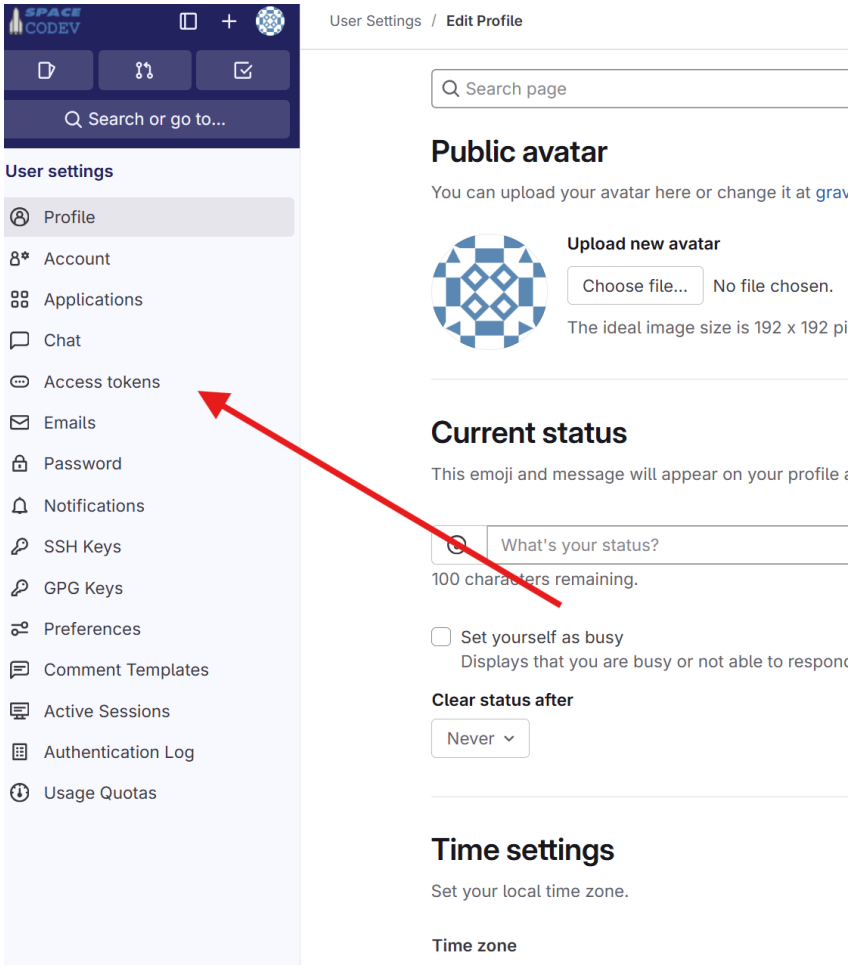

Figure 1.4: Access Token in the User Settings Panel

1.3 Step 3: Generate a Personal Access Token

After clicking the "**Access Tokens**" button, the following steps are:

- **Create a New Token:** On the Personal Access Tokens page, click the "**Add Personal Access Token**" or "**Generate New Token**" button.
  - **Token Name:** Enter a descriptive name such as `ESA-GODOT-Installation-2025 Workshop` to identify its purpose.
  - **Expiration Date (Optional):** Set an expiration date for added security (e.g., December 31, 2025, after the workshop). If left unset, the token remains valid until revoked.
  - **Scopes:** Select **all available scopes** (e.g., `api`, `read_repository`, `write_repository`) to ensure full access for installing `esa-godot`.

    Nota Bene: The guide recommends selecting every scope to avoid permission errors during installation.

  - Click "**Create**" or "Generate" to generate the token.
  - **Expected Interface:** A form with fields for token name, expiration, scope checkboxes, followed by a confirmation button as shown in Figure 1.5.

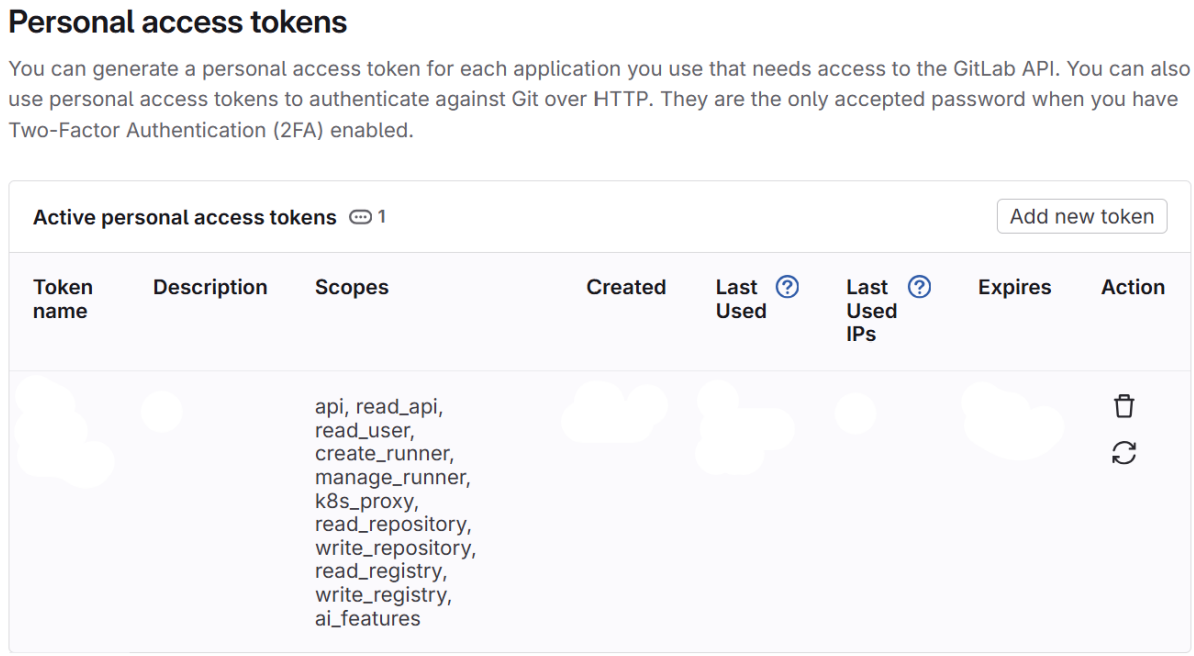

Figure 1.5: Personal Access Token with the checked scopes

- Copy the Token: Upon creating a person token, GitLab will display the token with a long string of characters such as:

```
glpat-123abc...
```

  - Immediately copy the token to a secure location (e.g., a password manager, Notepad or a private text file)
  - **Warning**: You can only view the token once. Closing the page or browser tab hides it permanently, requiring you to generate a new token.
  - Expected Output: A text box showing the token with a ”Copy” button or manual selection option.

1.4 Step 4: Securely Store the Token

- Save the Token: This is so important to paste the copied token into a secure and private location to prevent from any unauthorised access to your project environment.
  - **Recommended**: Use a password manager such as Microsoft Authenticator or a private and encrypted file on your computer.
  - **Avoid**: Please avoid saving the token in public cloud storage (e.g., Google Drive, OneDrive), email or shared documents as this highly risks exposure and unauthorised access to your project environment.
- Verify Access: Keep the token accessible for the next step (installation) but ensure it is not shared with others or left in an unsecured terminal

1.5 Troubleshoot Common Issues

- Lost Token: If you fail to copy the token before closing the page, please return to **Personal Access Tokens** in GitLab settings, revoke the old token if listed as shown in Figure 1.5 and generate a new one by repeating the aforementioned steps.
- Permission Errors: If installation fails due to insufficient scopes, ensure all checkboxes were selected during token selection. Regenerate the token if needed.
- Account Issues: If you cannot sign in or register, contact GitLab support via https://gitlab.space-codev.org/help or consult ESA's official GODOT documentation for alternative access methods.

Chapter 2

Overview of the Conda Installation for the ESA GODOT Software Toolkit

ESA-GODOT requires a Linux-based environment for operation due to the complexity. On Windows, this is achieved using Windows Subsystem for Linux (WSL), enabling a lightweight Ubuntu 22.04 environment. On MacOS, a Unix-based system, Conda is used directly to manage isolated Python environments. This guide details the setup of WSL on Windows or Conda on MacOS, creating a Conda environment for ESA-GODOT, with explanations of where to start, what actions to take, and why each step is necessary. It excludes ESA-GODOT package installation, focusing on establishing the environment. Further installation process is to be presented and discussed during the workshop.

ESA-GODOT is an astrodynamics library for analysis and operations of space missions. GODOT library was developed by the European Space Operations Centre under the framework of European Space Agency in Darmstadt, Germany. This library allows users to choose between C++ and Python as the language of the mission analysis [1].

Chapter 3

Installation on Windows OS via WSL with Conda

3.1 Advantages and Disadvantages of using Conda

It is important to understand how important using Conda is as a benchmark to download GODOT as it offers both benefits and drawbacks depending on the use of the development environment [3, 2].

3.1.1 Advantages of Using Conda for GODOT on Linux

Utilising Conda gives the following benefits as follows:

- Isolated Environment: This is crucial when preventing any potential conflicts with other Python packages or system dependencies because when downloading GODOT through the pip command, it is automatically downloaded with the necessary dependencies.

```
conda create -n godot_env python=3.10
```

  and then use the following:

```
pip install esa-godot ...
```

  inside the conda environment.

- Hybrid Dependency Management: Conda can install non-Python dependencies (e.g., scipy.integrate) more cleanly.

```
conda install numpy scipy
pip install esa-godot...
```

- Cross-Platform Development: If changing in devices between Windows, Mac and Linux or collaborating with other people, Conda helps ensure reproducible environments.

3.1.2 Disadvantages of using Conda for GODOT on Linux

- Misuse: This is crucial to understand that using `pip` and `conda` is completely different. It can be seen as undermining the ”only Conda” workflow
- Larger Environments: Conda environments can be heavier and take more disk space, especially if many packages are installed.

- Channel Conflicts: Mixing `conda` and `pip` can sometimes lead to dependency version conflicts but it is very rate if Conda packages **are installed first**.

3.2 Where to Start

Begin on a Windows computer, as ESA-GODOT requires a Linux environment. WSL2 provides a performant Linux distribution (Ubuntu 22.04) integrated with Windows. Administrative access and an internet connection are required to proceed this step.

3.3 Step 1: Configure WSL2

What to Do:

1. Open PowerShell as Administrator:
   - Press `Win + X` and select "Terminal (Admin)" or search for "PowerShell" in the Start menu, right-click, and choose "Run as administrator."
2. If prompted by User Account Control (UAC), click "Yes" to allow changes.
3. Install WSL with Ubuntu 22.04:

```
wsl --install -d Ubuntu-22.04
```

4. If WSL is installed, update it:

```
wsl --update
```

5. Restart the computer.
6. Verify WSL version:

```
wsl --list --verbose
```

   Expected output:

```
NAME                   STATE           VERSION
* Ubuntu-22.04         Running         2
```

7. If version is 1, upgrade to WSL2:

```
wsl --set-version Ubuntu-22.04 2
```

8. Update WSL kernel:

```
wsl --update
```

9. Launch WSL:

```
wsl
```

10. Set up a Linux user account:
    - Enter a username and password when prompted. Confirm the password twice. Store credentials securely.

Why:

- ESA-GODOT requires Linux; WSL2 offers a lightweight, integrated solution.
- Administrative access is needed for system modifications.
- Ubuntu 22.04 is a stable, ESA-GODOT-compatible distribution.
- WSL2 provides better performance than WSL1.
- The user account secures the Linux environment.

3.4 Step 2: Install Miniconda

Where to Start: In the WSL Ubuntu 22.04 terminal, accessed via `wsl`.

What to Do:

1. Download Miniconda:

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

2. Run the installer:

```
bash Miniconda3-latest-Linux-x86_64.sh
```

3. Follow on-screen instructions:
   - Agree to the license (`yes`).
   - Accept the default path (`/miniconda3`) or specify another.
   - Initialize Conda for shell scripts (recommended).
4. Activate Conda:

```
source ~/.bashrc
```

Why:

- Miniconda provides a minimal Conda and Python installation for managing ESA-GODOT dependencies.
- Installing in WSL ensures Linux compatibility.
- Activation enables Conda commands.

3.5 Step 3: Create a Conda Environment

Where to Start: In the WSL Ubuntu terminal, with Conda activated.

What to Do:

1. Add conda-forge channel:

```
conda config --add channels conda-forge
```

2. Create a new environment:

```
conda create --name godotdev
```

3. Activate the environment:

```
conda activate godotdev
```

4. Install Python 3.10:

```
conda install python=3.10
```

5. Verify Python version:

```
python --version
```

   Expected output: `Python 3.10.X`.

Why:

- Conda-forge provides ESA-GODOT-compatible packages.
- The `godotdev` environment isolates dependencies.
- Python 3.10 is within ESA-GODOT's supported range (3.8–3.11).

3.6 Step 4: Make the Conda Environment Permanent

Where to Start: In the WSL Ubuntu terminal, within `godotdev`.

What to Do:

1. Auto-activate `godotdev`:

```
echo 'conda activate godotdev' >> ~/.bashrc
```

2. Apply changes:

```
source ~/.bashrc
```

Why:

- Auto-activation streamlines workflow by loading `godotdev` on WSL startup.

Chapter 4

Installation on MacOS with Conda

4.1 Where to Start

Begin in the MacOS Terminal (`Applications > Utilities > Terminal`). MacOS is Unix-based, so Conda is used directly for environment management.

4.2 Step 1: Install Miniconda

What to Do:

1. Open Terminal.
2. Download Miniconda (Intel Macs):

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-MacOSX-x86_64.sh
```

   For Apple Silicon (M1/M2):

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-MacOSX-arm64.sh
```

3. Run the installer:

```
bash Miniconda3-latest-MacOSX-x86_64.sh
```

   Or, for Apple Silicon:

```
bash Miniconda3-latest-MacOSX-arm64.sh
```

4. Follow on-screen instructions:
   - Agree to the license (`yes`).
   - Accept the default path (`/miniconda3`).
   - Initialize Conda (recommended).
5. Activate Conda:

```
source ~/.bashrc
```

   If `/.bashrc` fails, use:

```
source ~/miniconda3/bin/activate
```

Why:

- Miniconda manages ESA-GODOT dependencies on MacOS.
- The correct installer ensures hardware compatibility.
- Activation enables Conda commands.

4.3 Step 2: Create a Conda Environment

What to Do:

1. Add conda-forge channel:

```
conda config --add channels conda-forge
```

2. Create a new environment:

```
conda create --name godotdev
```

3. Activate the environment:

```
conda activate godotdev
```

4. Install Python 3.10:

```
conda install python=3.10
```

5. Verify Python version:

```
python --version
```

   Expected output: `Python 3.10.X`.

Why:

- Conda-forge ensures package availability.
- The `godotdev` environment prevents conflicts.
- Python 3.10 is ESA-GODOT-compatible.

4.4 Step 3: Make the Conda Environment Permanent

What to Do:

1. Auto-activate `godotdev`:

```
echo 'conda activate godotdev' >> ~/.zshrc
```

   For Bash (older MacOS):

```
echo 'conda activate godotdev' >> ~/.bashrc
```

2. Apply changes:

```
source ~/.zshrc
```

Or, for Bash:

```
source ~/.bashrc
```

Why:

- Auto-activation simplifies environment access.
- MacOS uses Zsh (post-Catalina) or Bash, requiring the correct configuration file.

Chapter 5

Installation of Jupyter Notebook

This chapter introduces the installation of the Jupyter Notebook which is a web-based interactive computing platform that allows the users to create and share documents containing live code, equations, visualisations and narrative texts. It supports over 40 programming languages, including Python for the workshop software-package language.

5.1 Advantages of Jupyter Notebook

1. Interactive Development: The users can write code, run it and see the output immediately which is a high advantage for testing and debugging.
2. Rich Media Support:
   - Especially for the mission analysis with the use of `esa-godot`, Jupyter Notebook supports visualisation libraries such as `matplotlib` and directly within the notebook.
   - Following this it allows embedding of images, videos, LaTeX equations and HTML.
3. Documentation Friendly: This supports Markdown cells which is highly beneficial for sharing research results and help document your workflow or analysis right alongside your code.
4. Great for Data Science and Prototyping: It is ideal for exploratory data analysis (ESA) including the use of `esa-godot` as it is possible to visualise and manipulate data step by step, accordingly.
5. Shareability: Notebooks can be exported to HTML, PDF or shared via GitHub or NBViewer if `pandoc` package is installed alongside the installation of Jupyter Notebook.

5.2 Disadvantages of Jupyter Notebook

1. Hidden State Issues: This is important to understand that on Jupyter Notebook, execution order is not always linear, cells can be run out of order, which can lead to misleading results or errors. However, it is fixable by using the command of `Run All Cells` if needed.
2. Not Ideal for Large Projects: This is important to understand that sometimes large projects like the full mission analysis could potentially introduce poor structure for large codebases due to the lack of modularisation and maintainability compared to traditional script-based development.

3. Performance Overhead: It is also important to note that as said in the first disadvantage point, hidden state issues, long-running notebooks reinforced by the incoherent execution order or very large datasets can slow down the notebook execution which could further lead to crash the kernel.
4. Security Risks: Notebooks can execute arbitary code and might contain malicious content, especially when obtained from untrusted sources.
5. Reproducibility Concerns: Hidden state and mutable data can make it hard to reproduce results unless all cells are re-run in order from the top. However, it is important to note that some of esa-godot specific files are sensitive to the re-run command as they are assumed to be extracted once.

5.3 Step 1: Install and Configure Jupyter Notebook

What to Do:

1. Ensure `godotdev` is active:

```
conda activate godotdev
```

2. Install Jupyter Notebook:

```
conda install jupyter
```

   After approving the installation process by clicking `Y` with the user's password, the user is expected to have Jupyter Notebook installed.

3. Launch Jupyter Notebook to test:

```
jupyter notebook
```

   - A browser open at the following:

```
http://localhost:8888 (or similar)
```

   - Otherwise, the user is expected to copy the given URL from the terminal. The user gets such a link as follows:

```
http://localhost:8888/?token= ...
```

4. Create a test notebook:
   - Click **`New > Python 3 (ipykernel)`**
   - Run a test cell:

```
import random
print("Random number between 1 and 10:", random.
    randint(1,10))
```

Listing 5.1: Random library for the Jupyter Notebook test

Expected Outcome:

```
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
```

Chapter 6

Next Steps

The Conda environment is now ready for ESA-GODOT installation and configuration (e.g., `universe.yaml`), and further explanation and hand-out instructions for the configuration and installation of the package is to be presented at the workshop.

Bibliography

[1] ESA Flight Dynamics. *GODOT Python (godotpy): Python interface for the ESA/ESOC GODOT astrodynamics library*. https://godot.io.esa.int/godotpy/. Documentation for version 1.11.0 (latest release as of June 2025). 2025. (Visited on 06/11/2025).

[2] Jon Krohn. *Conda vs. Pip: What's the Difference?* https://codesolid.com/conda-vs-pip/. Accessed: 2025-06-11. 2024.

[3] Jannis Leidel. *12 Reasons to Choose Conda*. https://www.anaconda.com/blog/12-reasons-to-choose-conda. Blog post. Sept. 2023. (Visited on 06/11/2025).