EventStoreDB Python Environment Installation

Overview

Welcome to the Python example of Event Store's **From Scratch** series. This series allows you to quickly overcome the common challenges of setting up and configuring a new development environment, and focus on advancing your EventStoreDB skills.

The **From Scratch** series provides working code examples for basic reads and writes to EventStoreDB, a tested environment to run the code, and instructions that clearly describe the steps required to run the code successfully.

Each **From Scratch** repository provides the following:

- A working Github Codespaces environment
- Instructions on running EventStoreDB locally
- Instructions to set up a similar project on your own

We recommend you progress through the **From Scratch** projects in the following order:

1. Run the code in Codespaces
2. Clone the For Scratch GitHub repo, and follow the instructions to run it locally
3. Build your own project

This document provides detailed instructions for **setting up your own Python project with EventStoreDB.** ***This is the recommended third stage in Event Store's From Scratch Python series.***

Other clients in the **From Scratch** series include:

- Node
- Java
- .NET

Topics covered

1. Setting up a Python project
2. Adding EventStoreDB dependencies to the Python project
3. Start a local Docker container running EventStoreDB
4. Configure the directory for GitHub
5. Create a GitHub repository

This is intended as a baseline working example of an EventStoreDB Python project. Your Python projects may be significantly more complex.

1. Setting up a Python project

Installing Python

To execute Python code, you must have a Python interpreter installed. One way to verify if Python is installed is to run the following command in a terminal window.

```
python –version
```

Note that in some environments it might be necessary to run “python3” instead of “python.” In this case, use the following command.

```
python3 --version
```

If you need to install Python, please reference: https://www.python.org/downloads/

Create a directory named “EventStoreDB_project.”

```
mkdir EventStoreDB_project
```

Navigate to your new directory.

```
cd EventStoreDB_project
```

Using Virtual Environments for Python

Modifying your global Python environment for every project will quickly become extremely difficult to manage. Additionally, the system tools in most Linux environments depend on the global Python environment, and changing the global Python version can cause serious problems.

One way to deal with this is with virtual environments, or “venvs.” Similar tools, such as Conda and Miniconda, exist, but “venv” is used in this project.

The following is an example of creating and using a venv in an environment where the system Python is executed with the command `python3`, rather than simply `python`.

```
python -m venv .EventStoreDB
```

Activate the “venv” by executing the following command.

```
source .EventStoreDB/bin/activate
```

Your prompt will change to `(.EventStoreDB) $` to reflect that you are inside a “venv.”

2. Add EventStoreDB dependencies to the Python project

Pip is a Python tool used to add dependencies. Install esdbclient libraries with pip.

```
pip install esdbclient
```

Saving and loading dependencies

You can save the dependencies you installed into your "venv" by writing them into a requirements.txt file. The GitHub repo includes the requirements.txt file. This allows you to run `python sample_append.py` in VS code, as the IDE parses the requirements.txt and adds the dependencies listed in that file.

Loading dependencies from requirements.txt.

To generate your requirements.txt run the following command.

```
pip freeze > requirements.txt
```

To load your dependencies run the following command.

```
pip install -r requirements.txt
```

3. Start a local Docker container running EventStoreDB

If you have yet to do so, install Docker: https://docs.docker.com/engine/install/.

Run the following command to initiate an unsecured single instance EventStoreDB cluster locally.

```
docker run -d --name esdb-node -it -p 2113:2113 -p 1113:1113 \
eventstore/eventstore:lts --insecure --run-projections=All \
--enable-external-tcp --enable-atom-pub-over-http
```

4. Configuring the directory as a git repository

Once you have built your code and configured your "venv," configure the directory as a local git repository.

To start, save your dependencies into a requirements.txt file as described above.

```
pip freeze > requirements.txt
```

Install git and run the following command to create a .git folder where git stores version management information.

```
git init
```

Some content should not be shared to git. For example, the "venv" folder created when a "venv" is created in this directory should not be shared. To exclude that local directory from being shared to GitHub, add an entry to the .gitignore file. To learn more about this process, please see the GitHub documentation.

The "venv" was created in this project with the following command.

```
python -m venv .EventStoreDB
```

So the .gitignore will have an entry as follows:

```
 .EventStoreDB/%
```

5. Create a repo on GitHub and push your directory as the first commit

After creating the repo, GitHub presents a page with instructions. Follow the instructions titled "**...or push an existing repository from the command line**." You will see two steps below that are not included in GitHub. Ensure you follow the steps below.

Run the following commands to link and push your local Git repo to your GitHub repo. Remember to replace `<YOUR REPO NAME>` with the name of your repository.

```
git remote add https://github.com/<YOUR REPO NAME>.git
```

```
git branch -M main
```

```
git add -A
```

```
git commit -m "first commit"
```

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

Congratulations! You now have a working Python project that includes a basic write and read using EventStoreDB. This can be the foundation for building more advanced and complete projects.

Next Steps

Now that you have successfully created a Python project, you have completed the **From Scratch** Python lessons. Please feel free to venture into another **From Scratch** series. Event Store offers similar content for .NET, Java, and Node.js.

To continue your learning, you can find additional examples in the following repo:

https://github.com/EventStore/samples

In particular, we recommend the Quickstart examples here:

https://github.com/EventStore/samples/tree/main/Quickstart