EventStoreDB From Scratch: Running Java in GitHub Codespaces

Overview

Welcome to the Java 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 From Scratch GitHub repo, and follow the instructions to run it locally
3. Build your own project

This document provides detailed instructions on launching GitHub Codespaces, starting an EnventStoreDB docker container, and running the sample JAVA code that writes and reads from EventStoreDB. *This is the recommended starting point for running Java code with EventStoreDB.*

Other clients in the **From Scratch** series include:
- Node
- .NET
- Python

Topics covered

1. Launching Codespaces
2. Running code in Codespaces

Launching Codespaces

GitHub created Codespaces to answer the *"It doesn't run on my machine"* problem faced by developers of all experience levels. CodeSpaces provides the IDE, the repo, and the environment so you can focus on running code.
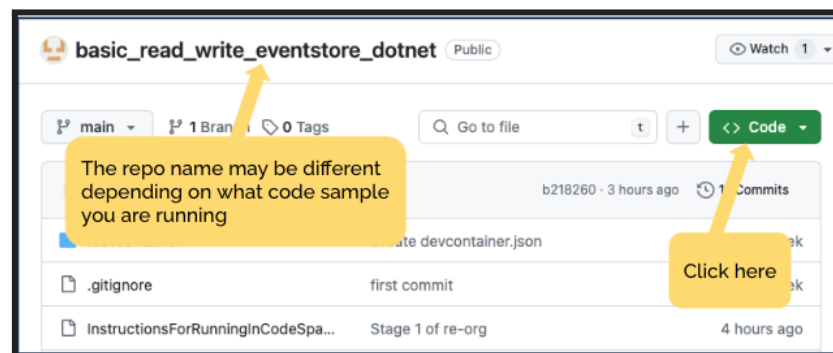
Here are the steps to launch the **From Scratch** repos in GitHub Codespaces.
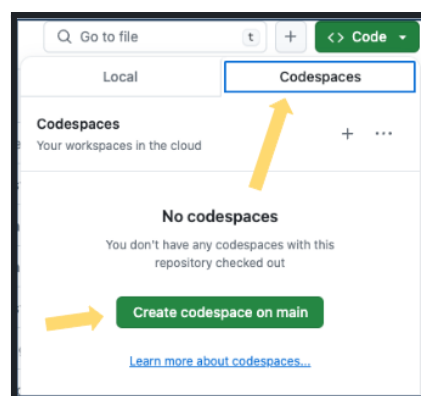
Requirements

1. A GitHub account
2. A browser
3. Access to the internet

Steps

1. Navigate to the Java **FromScratch** repository.

2. Click on the green "<> Code" button

3. You will see two tabs titled "Local" and "Codespaces." Select "Codespaces," then click the green button labeled "Create codespace on main."

4. Wait for your Codespace to launch. Depending on the container's configurations, this can take anywhere from a few seconds to a few minutes. While it launches you will see this image.


5. On the welcome page of VS Code, you will be given the option to make some formatting choices. VS Code is the default IDE in Codespaces for the **From Scratch** project. Choose your preferred theme or use the default theme by closing the "Welcome" tab.

Running the Code in Codespaces

To read and write code with EventstoreDB, you'll need a running cluster.

Using a docker container is a quick way to get started. More information is available here, https://developers.eventstore.com/server/v24.2/installation.html#run-with-docker

For the "From Scratch" project we provide a shell script that starts or restarts a docker container running EventStoreDB.
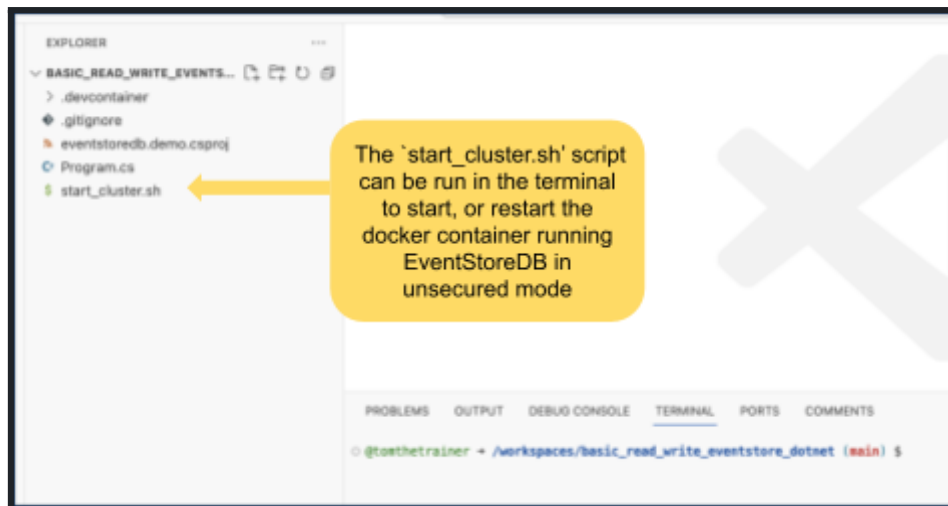
Some notes on the 'start_cluster.sh'

The 'start_cluster.sh' script is designed to either start, or in the case of an already running docker container, to restart the container. Restarting the container with `start_cluster.sh'` will delete any streams that you had written to the previous instance of the docker container. This design decision is intentional.

Please note that Codespaces are set to pause after a period of inactivity. When restarting an inactivated Codespace the start_cluster.sh script may fail to restart the Docker container. The most straightforward solution to this issue is terminating the Codespace and starting a new one.

Follow the steps below to start your cluster.

1. To launch a Docker container running EventStoreDB where the "FromScratch" code will write and read events, run the start_cluster.sh script. Type the following command into the terminal located at the bottom of your Codespace.

```
./start_cluster.sh
```

Once your Docker container has finished downloading, you may see a pop-up stating, "*Your application running on port 113 is available...*" Do not click "Open in Browser." Port :1113 is used for RPC calls and will not direct you to the WebUI (which you will do in the next step).
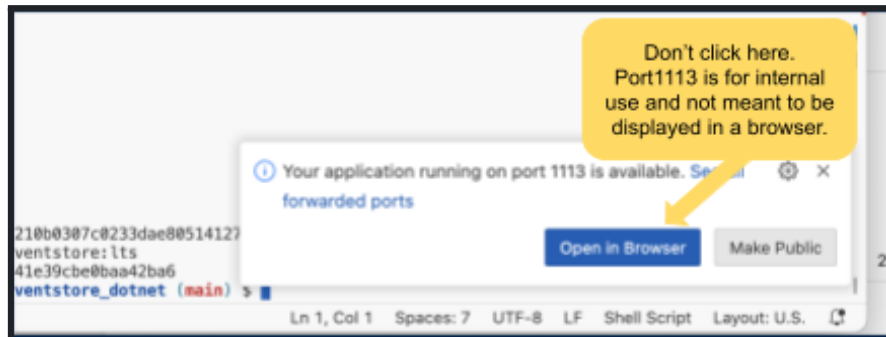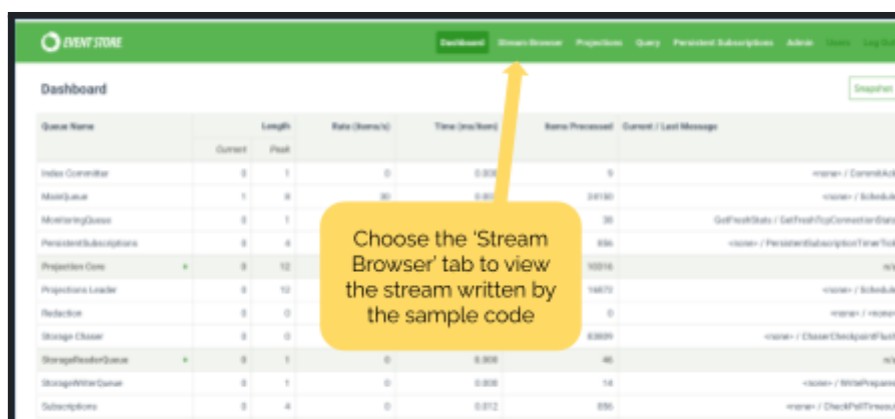

Access the EventStoreDB WebUI Stream Browser

1. Open the WebUI of EventStoreDB running in the docker container. EventStoreDB uses ports :1113 and :2113. Open the WebUI **'port :2113'** in a browser tab.
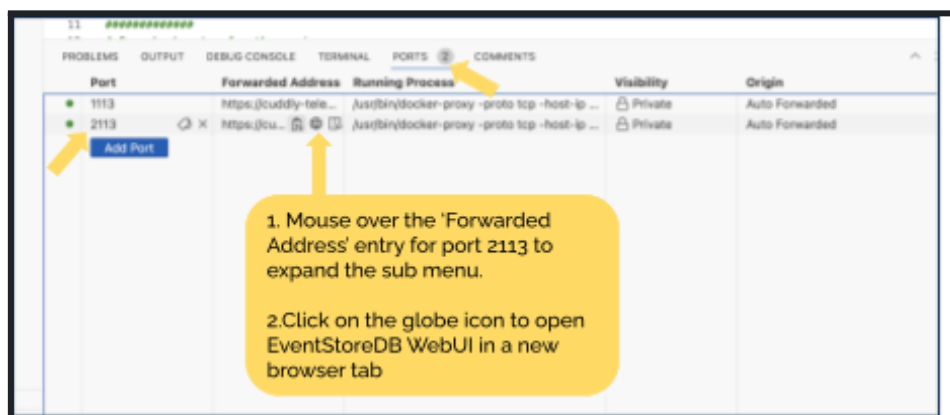
2. Select the Stream Browser tab from the EventstoreDB WebUI. After running the sample append code, the events written in the demo will be visible here.

Stream Browser view explained

The 'Stream Browser' tab provides an overview of recently created and changed streams. Clicking on a stream name shows details about the individual stream.

A system stream in EventStoreDB is a type of stream that is used for system-level operations and information. These streams are distinguished by a $ prefix. For example, the stream metadata for a stream named foo is $foo. System streams can contain metadata for other streams or system-level information.

For the purposes of this example you can ignore system streams. As you continue on your EventStoreDB journey, you will find them to be useful sources of information.
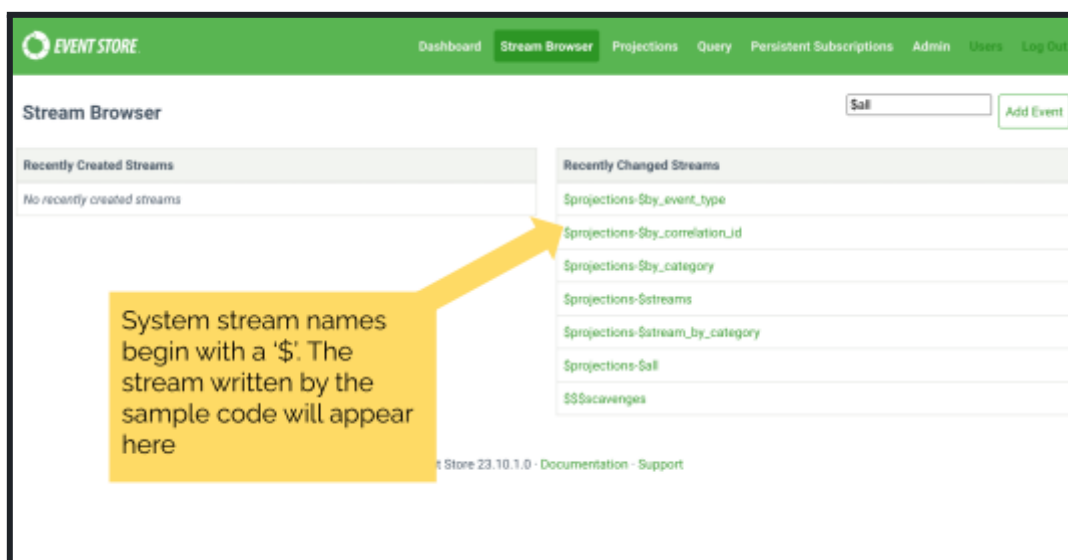
Congratulations! You have successfully started the EventStoreDB cluster and viewed the stream browser from the EventStoreDB WebUI.

Run the Java code sample

1. Verify you have a running EventStoreDB cluster by viewing the WebUI. If you need to start the cluster, run the following in a terminal window.

```
$ ./start_cluster.sh
```

2. Open the stream browser in the WebUI of the cluster.
3. In the VS Code Explorer tab on the left navigate to /src/main/java/eventstoredb_demo

You will see four Java files:

- App.java
- SampleRead.java
- SampleWrite.java
- TestEvent.java

Summary of classes in this project

App.java is a class created when "mvn archetype:generate…" was ran when the project was created. It was left in to provide a simple test. App.java prints "hello world" to the console to verify a basic working Java environment.

TestEvent.java is required by the Jackson libraries which transform a Java object into a JSON object. SampleWrite uses the TestEvent class to format the JSON written to EventStoreDB.
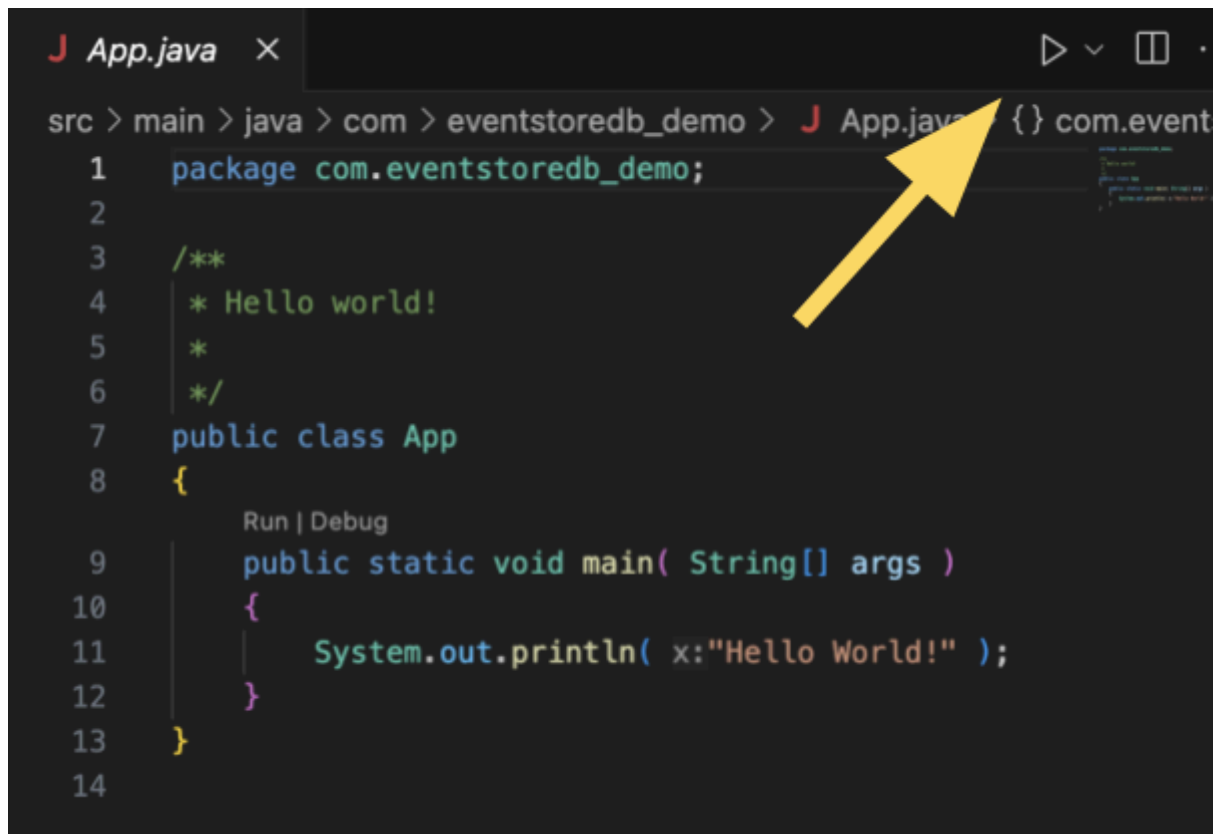
SampleWrite.java writes an event to EventStoreDB. It assumes EventStoreDB is running locally in the EventStoreDB Docker container. If it were run against Event Store Cloud or an alternate instance, you must modify the connection string to connect to that instance.

SampleRead.java reads the events back from EventStoreDB. Similar to the write, it assumes the EventStoreDB container runs locally in unsecured mode. The connection string must be modified to connect to another EventStoreDB instance, such as Event Store Cloud.

Run the code in the following order

Run App.java first.

- Click the *run arrow* top right. This will take a while as the first run builds the project.

Run the SampleWrite Class

Open the file and run it. You will see an event reflected in the EventStoreDB WebUI's Stream Browser. You will also get confirmation printed to the terminal

Run the SampleRead Class

SampleRead reads all events in the stream created by SampleWrite.

Run this code only after you have run SampleWrite. Otherwise, it will return an error.

Congratulations! After running SampleWrite.java followed by SampleRead.java you have succeeded in Writing and Reading events to and from EventstoreDB.

Next Steps

Now that you have successfully leveraged GitHub Codespaces to read and write events to and from EventStoreDB, we recommend you run the code locally. Please continue to the **From Scratch** Java instructions for running locally to continue your learning.

As you progress with your EventStoreDB skills, you can also 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