



# **Hyperledger Sawtooth for Application Developers**

## **Module 6: Running the Simple Supply Application**

This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

# [MODULE 6] Running the Simple Supply Application

---

## Module 6: Running the Simple Supply Application

This module describes how to put everything together by running the Simple Supply application. Although the client web app is a very basic interface, this module explains how the log messages show important indicators of your application's behavior.

### Contents

- About the Sawtooth Simple Supply environment
- Starting Sawtooth Simple Supply with Docker
- Using the Curator web app
- Troubleshooting
- Shutting down the Sawtooth environment



## About the Sawtooth Simple Supply Environment

---

Module 6 > Running the Simple Supply Application > About the Sawtooth Simple Supply Environment

The test environment for Sawtooth Simple Supply is a single Sawtooth node with a validator and all application components. This environment uses Docker containers for each component.

### ! Prerequisites

This environment requires both [Docker Engine](#) and [Docker Compose](#). Make sure that the `docker` and `docker-compose` commands are available on your machine.

If you have not cloned the Simple Supply repository yet, see “Cloning the Simple Supply Repository” in the previous module.



## Starting Sawtooth Simple Supply with Docker

---

Module 6 > Running the Simple Supply Application > Starting Sawtooth Simple Supply with Docker

The Sawtooth Simple Supply repository has a `docker-compose.yaml` file that starts all the Sawtooth and application components.

To start Sawtooth and run the Simple Supply application:

1. Open a terminal window and navigate to the project's root directory (for example, `~/education-sawtooth-simple-supply`).

2. Run this command:

```
$ docker-compose up
```

This command downloads the Docker images for the Sawtooth environment, then starts all Sawtooth and Simple Supply components in separate containers.

★ Downloading the Docker images and building the Simple Supply application can take several minutes. When the log messages stop, the Sawtooth environment is ready.

3. Keep the terminal window open. You will notice many log messages from all Sawtooth and application components.

! If you want to stop the containers but keep the data, enter **CTRL-C** in this terminal window. Later, you can run `docker-compose up` to start the same containers again. To completely stop the Sawtooth environment, and remove all containers and data, see "Shutting Down the Sawtooth Environment", below.



## About the Docker Containers

The compose file defines the following containers. Note the HTTP endpoints for the Simple Supply client, the Simple Supply REST API, and the PostgreSQL Admin panel.

Component	Container Name	HTTP Endpoint	Description
Shell	simple-supply-shell		Shell for running Sawtooth commands
Client	curator-app	http://localhost:8040	Curator web app ( the client front end)
Simple Supply transaction processor	simple-supply-tp		Simple Supply business logic
Simple Supply REST API	simple-supply-rest-api	http://localhost:8000	Communication between client and transaction processor via HTTP/JSON
Validator	sawtooth-validator		Validation for blocks and transactions
Consensus engine	sawtooth-devmode-engine-rust-default		Dev mode consensus engine (for development use only)
Settings transaction processor	sawtooth-settings-tp		Built-in Sawtooth transaction processor for on-chain settings
Event subscriber	simple-supply-subscriber		Listens to Sawtooth events
PostgreSQL database	simple-supply-postgres		PostgreSQL Reporting database
PostgreSQL Adminer	simple-supply-adminer	http://localhost:8080	PostgreSQL admin tool for viewing data in the reporting database

The compose file also starts a Sawtooth REST API with the HTTP endpoint `http://localhost:8008`.



## Watching the Log Messages

When you first start the Sawtooth environment, the end of the log output shows startup messages from the Sawtooth and Simple Supply components.

For example, the initial log output shows when the event subscriber (`simple-supply-subscriber`) starts and connects to the reporting database, creates tables, and then disconnects from the database.

```
simple-supply-subscriber | Initializing subscriber...
simple-supply-subscriber | Connecting to database
simple-supply-subscriber | Successfully connected to database
simple-supply-subscriber | Creating table: blocks
simple-supply-subscriber | Creating table: auth
simple-supply-subscriber | Creating table: records
simple-supply-subscriber | Creating table: record_locations
simple-supply-subscriber | Creating table: record_owners
simple-supply-subscriber | Creating table: agents
simple-supply-subscriber | Disconnecting from database
[...snip...]
simple-supply-subscriber | Starting subscriber...
simple-supply-subscriber | Connecting to database
simple-supply-subscriber | Successfully connected to database
simple-supply-subscriber | Connecting to validator: tcp://validator:4004
simple-supply-subscriber | Using selector: ZMQSelector
simple-supply-subscriber | Subscribing to state delta events
```

As you use the Curator web app, watch the log messages in this terminal window. These log messages will show interesting activity for Simple Supply components, including state changes to the blockchain.



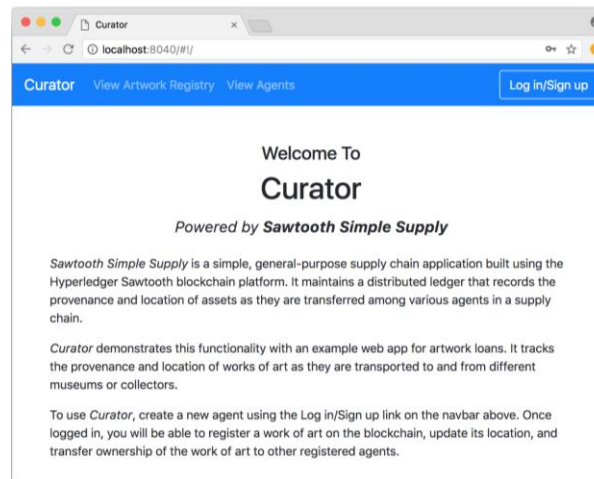
## Using the Curator Web App

Module 6 > Running the Simple Supply Application > Using the Curator Web App

In this procedure, you will use the Curator web app to record the transfer of a 14th-century hanging scroll, [Poem in Chinese about Sugar](#), from the Metropolitan Museum of Art in New York City to the J. Paul Getty museum in Los Angeles for a future exhibition, “The Art of Cooking”. You will create a Met museum agent and register the artwork, then create an agent for the Getty. Finally, you will update the scroll’s location and transfer ownership to the other agent to record how it makes its way across the country.

Although this Curator web app is extremely simple, it demonstrates the underlying functionality of the Simple Supply application and how its components work together to make changes on the blockchain.

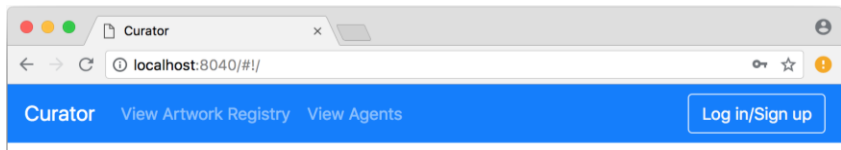
1. To get started, use a browser on your system to navigate to the Curator client’s endpoint, `http://localhost:8040`.



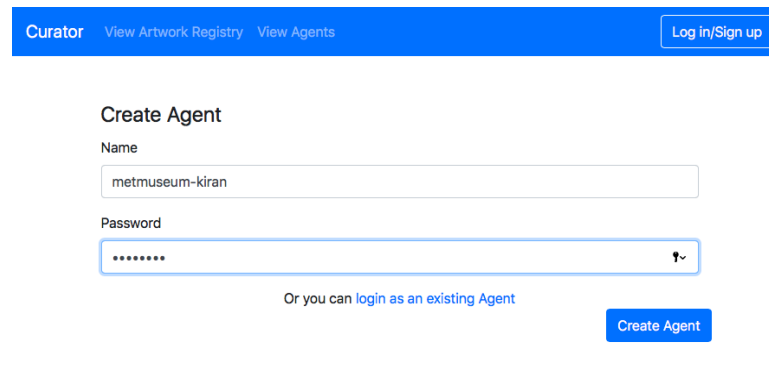
## Creating the First Agent

Module 6 > Running the Simple Supply Application > Creating the First Agent

2. To create the first agent (the artwork owner), click **Log in/Sign up**. On the next page, click **create a new agent**.

A screenshot of the login/sign-up form. It features a 'Password' label above a text input field. Below the input field, there is a link that says 'Or you can create a new agent'. A blue 'Login' button is positioned at the bottom right of the form.

3. Enter a name and password, then click **Create Agent**. If the operation is successful, the client returns you to the Welcome page.

A screenshot of the 'Create Agent' form. The header is blue with the text 'Curator' and links 'View Artwork Registry' and 'View Agents'. A 'Log in/Sign up' button is in the top right. The main content area has the title 'Create Agent'. Below it are two input fields: 'Name' with the value 'metmuseum-kiran' and 'Password' with masked characters. A link 'Or you can login as an existing Agent' is below the password field. A blue 'Create Agent' button is at the bottom right.



## Creating the First Agent: Simple Supply Log Messages

Module 6 > Running the Simple Supply Application > Creating the First Agent: Simple Supply Log Messages

4. In the terminal window, the log messages show that the transaction is validated, then committed.

For example, the following message is displayed when a block-commit event is broadcasted after the block is committed.

```
sawtooth-validator | [2018-06-28 19:09:22.461 DEBUG broadcaster] Broadcasting events:
[event_type: "sawtooth/block-commit"]
```

Later in the output, other log messages show the REST API creating the `auth` entry in the reporting database.

```
simple-supply-rest-api | [2018-06-28 19:46:35.897 INFO cursor]
simple-supply-rest-api | INSERT INTO auth (
simple-supply-rest-api | public_key,
simple-supply-rest-api | encrypted_private_key,
simple-supply-rest-api | hashed_password
simple-supply-rest-api | )
simple-supply-rest-api | VALUES
('02d7213082f704b3221e4270c35e3e54b2428e94f1d59a1506395ff20e806a777b',
'943ec168d126ba990854a9c5456ff1e8776291e2b8ada44d8dfe7891771510b103a02b7167fc4ea640e41b9fe5f69c3d5cb65ce2ce8c1961ace0429893b064d6',
'243262243132246f6472587964566d57362e2e7432552f536a6c33562e747953554b34634f41704448432e6e51766467456a617439526f36714c6871');
```



## Viewing the Agent List

Module 6 > Running the Simple Supply Application > Viewing the Agent List

- Click **View Agents** to see the new agent.

Curator Register Artwork View Artwork Registry View Agents	
Name	Key
<a href="#">metmuseum-kiran</a>	02d7213082f704b3221e4270c35e3e54b2428e94f1d59a1506395ff20e806a777b

! Simple Supply identifies agents by public key, not user name, so you must enter a public key to log in and to transfer artwork to a different agent. If you forget a public key, go to this **View Agents** screen to see all agents' public keys.

- In the terminal window, you can see the log messages where Curator sends the request and the REST API returns the information.

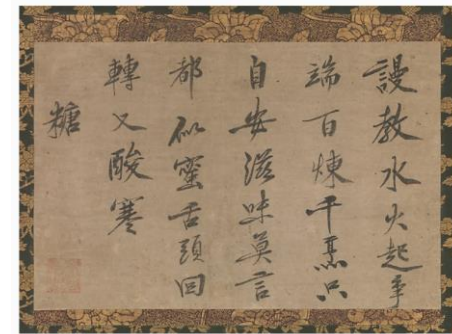
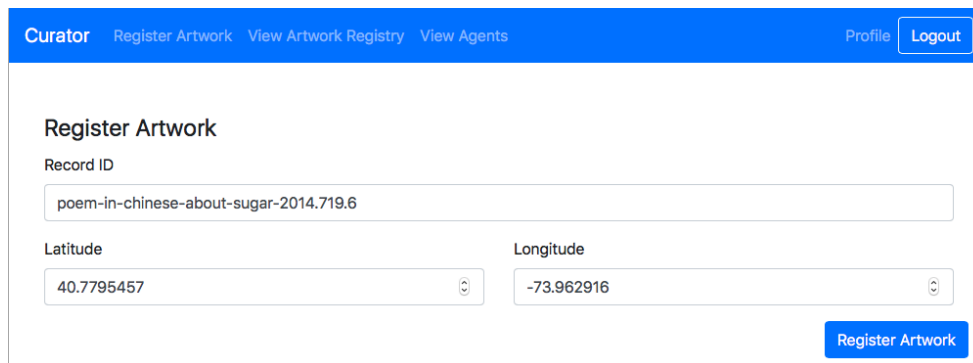
```
curator-app | 172.18.0.1 - - [28/Jun/2018:19:46:35 +0000] "POST /api/agents HTTP/1.1" 200
390
simple-supply-rest-api | [2018-06-28 19:46:38.617 INFO cursor]
simple-supply-rest-api | SELECT public_key, name, timestamp FROM agents
simple-supply-rest-api | WHERE (
simple-supply-rest-api | SELECT max(block_num) FROM blocks
simple-supply-rest-api | ) >= start_block_num
simple-supply-rest-api | AND (
simple-supply-rest-api | SELECT max(block_num) FROM blocks
simple-supply-rest-api | ) < end_block_num;
simple-supply-rest-api |
simple-supply-rest-api | [2018-06-28 19:46:38.617 INFO cursor] None
```



## Registering Artwork

Module 6 > Running the Simple Supply Application > Registering Artwork

- Next, register a new work of art. Click **Register Artwork**, then enter a unique ID and the artwork's location as decimal latitude and longitude values. (The Metropolitan Museum of Art is at latitude 40.7795457, longitude -73.962916.)



墨跡「糖」  
Poem in Chinese about Sugar

Artist:	Kokan Shiren (Japanese, 1278–1346)
Period:	Nanbokuchō period (1336–92)
Date:	14th century
Culture:	Japan
Medium:	Hanging scroll, ink on paper
Dimensions:	Image: 12 1/4 x 18 5/8 in. (311 x 473 cm) Overall with mounting: 47 x 24 in. (119.4 x 61 cm) Overall with knobs: 47 x 25 13/16 in. (119.4 x 65.6 cm)
Classification:	Calligraphy
Credit Line:	Gift of Sylvan Bernet and William Burto, in honor of Elizabeth and Neil Swinton, 2014
Accession Number:	2014.719.6

Simple Supply checks that the artwork ID is unique. As before, the log messages show the transaction being processed and the block being committed, then the REST API requests artwork information.



## Viewing Artwork Details

Module 6 > Running the Simple Supply Application > Viewing Artwork Details

- After you register new artwork, Curator displays the **Artwork Registry** screen. You can click the artwork ID to display the details.

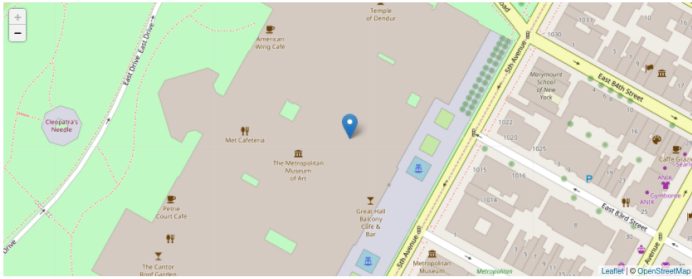
Curator Register Artwork View Artwork Registry View Agents Profile Logout

poem-in-chinese-about-sugar-2014.719.6

**Owner**  
02d7213082f704b3221e4270c35e3e54b2428e941d59a1506395ff20e806a777b

**Created**  
Thu Jun 28 2018 14:58:09 GMT-0500 (CDT)

**Updated**  
Thu Jun 28 2018 14:58:09 GMT-0500 (CDT)



**Update Location**

Latitude

Longitude

**Update Location**

**Transfer Ownership**

Public Key

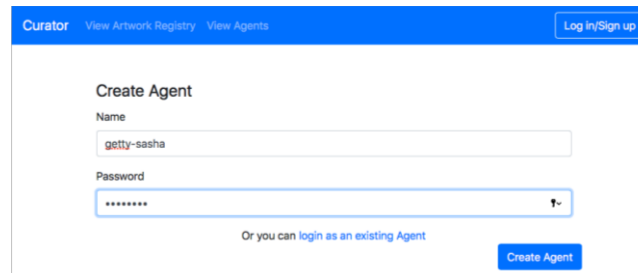
**Transfer Ownership**

Open #localhost:8040/# on this page in a new tab

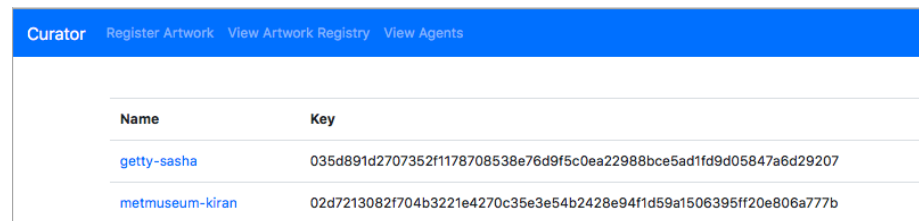


## Creating the Second Agent

9. Now create a second agent for the other museum:
  - a. Open another browser window in incognito or private mode so that you don't have to log out from the first Curator session. This is a simple way to pretend that you're a different person using a separate system.
  - b. Navigate to `http://localhost:8040`.
  - c. Click **Log in / Sign up**, then click **create a new agent** on the next page.
  - d. Enter a new agent name and password.



10. You are now logged in as the second agent. Click **View Agents** to see the updated list. In the terminal window, the log messages are similar to those for the first agent.



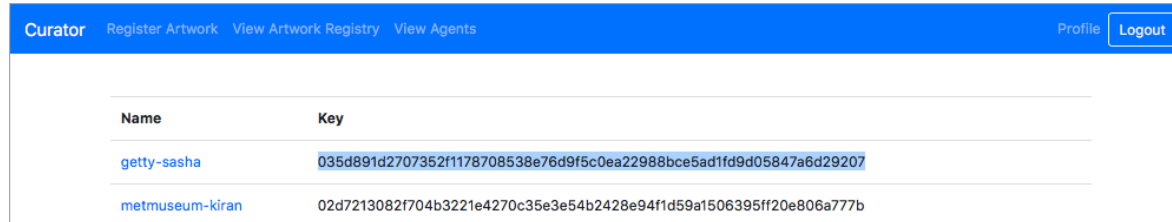
Name	Key
<a href="#">getty-sasha</a>	035d891d2707352f1178708538e76d9f5c0ea22988bce5ad1fd9d05847a6d29207
<a href="#">metmuseum-kiran</a>	02d7213082f704b3221e4270c35e3e54b2428e94f1d59a1506395ff20e806a777b



## Transferring the Artwork

### Module 6 > Running the Simple Supply Application > Creating the First Agent

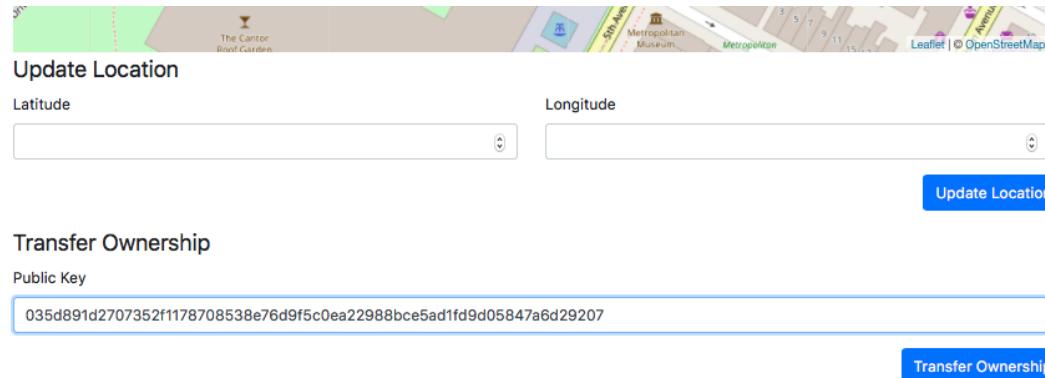
11. First, copy the recipient's public key. From the **View Agent** screen, copy the second agent's public key. (You can do this in either browser window.)



Name	Key
getty-sasha	035d891d2707352f1178708538e76d9f5c0ea22988bce5ad1fd9d05847a6d29207
metmuseum-kiran	02d7213082f704b3221e4270c35e3e54b2428e94f1d59a1506395ff20e806a777b

12. In the first agent's browser window, go to the artwork details page: Click **View Artwork Registry**, then click the artwork ID. On the details screen, scroll down to **Transfer Ownership** at the bottom of the page

13. In the **Public Key** field, paste in the the second agent's public key, then click **Transfer Ownership**.



Update Location

Latitude

Longitude

Update Location

Transfer Ownership

Public Key

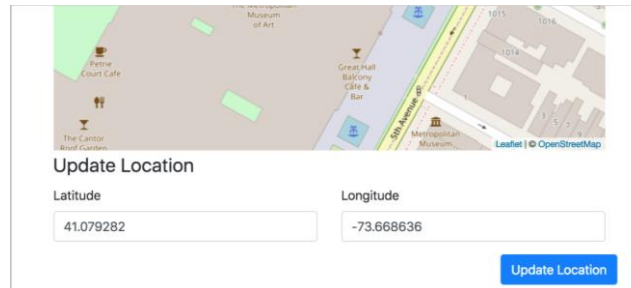
Transfer Ownership

- ★ Refresh the browser window to see the ownership change at the top of this window.



## Changing the Location

14. Finally, the second agent (who is now the owner of the artwork), can change the artwork's latitude and longitude to its new location.
  - a. In the second agent's browser window, go to the artwork details page: Click **View Artwork Registry**, then click the artwork ID.
  - b. On the details screen, scroll down to **Update location** at the bottom of the page.
  - c. Enter the new location. (The J. Paul Getty Museum is at latitude 34.0450085 and longitude -118.5650826.)

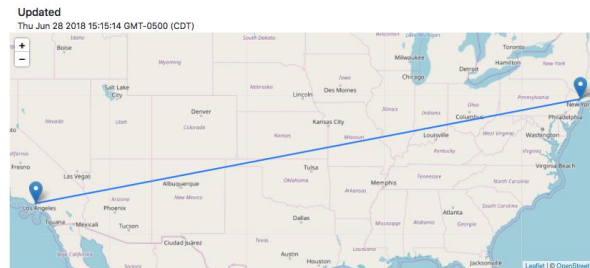


Update Location

Latitude: 41.079282 Longitude: -73.668636

Update Location

- d. Refresh the browser window to see the location change.



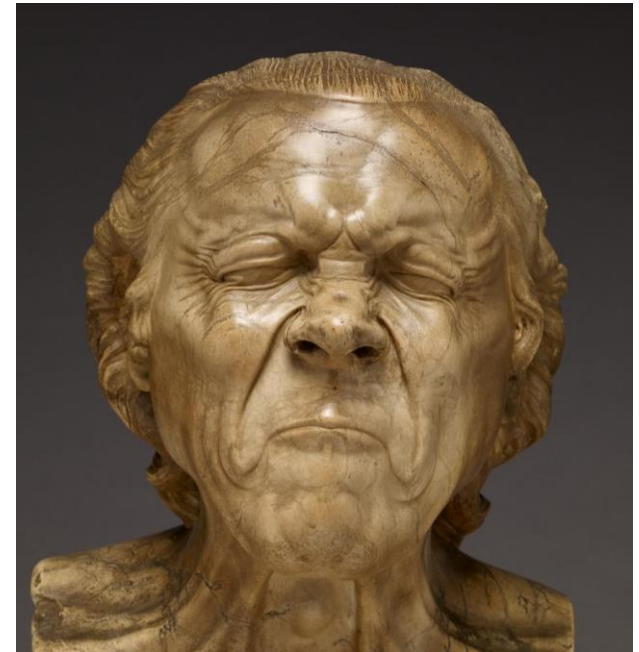
This section contains general tips for common problems with this procedure.

- Do you see unexpected Docker errors when you start up the Sawtooth environment? If so, be sure to start with a clean Docker environment. For more information, see “Shutting Down the Sawtooth Environment”, below.
- If there are problems with user information or event data in the reporting database, you can use the PostgreSQL Adminer to examine the data. The next topic shows how to connect and log in.
- Look at the Sawtooth logs to verify that the required components are running and communicating properly.

For example:

- Is the Simple Supply transaction processor running?
- Did the transaction processor succeed in registering with the validator?
- Are the expected blocks of transactions being committed to the blockchain?

The following topics describe how to view the Sawtooth logs.



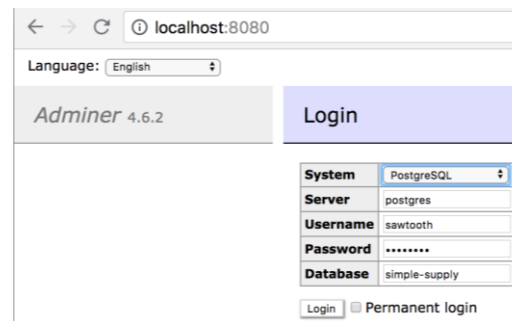
Franz Xaver Messerschmidt, The Vexed Man  
J. Paul Getty Museum  
Digital image courtesy of the Getty's Open Content Program



## Using the PostgreSQL Adminer with the Reporting Database

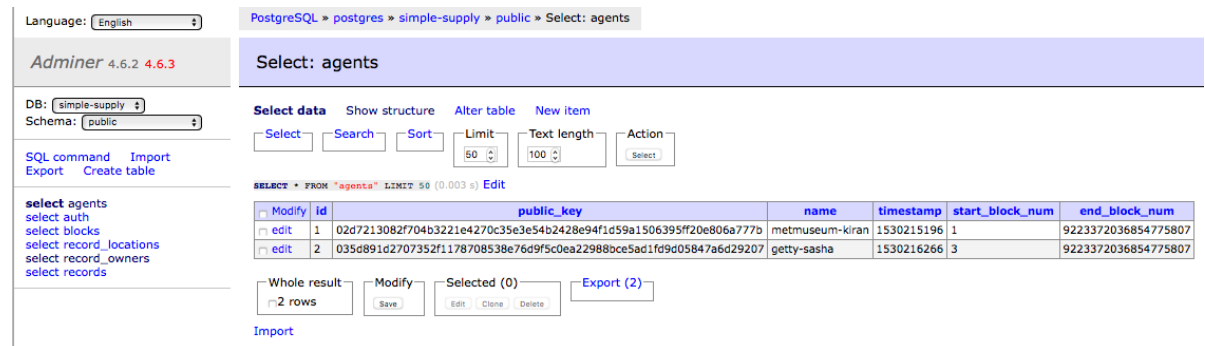
The compose file includes a container for PostgreSQL Adminer, which is a tool you can use to view data in the reporting database.

1. In a browser, go to <http://localhost:8080>.
2. Use the following settings:
  - **System:** PostgreSQL
  - **Server:** postgres
  - **Username:** sawtooth
  - **Password:** sawtooth
  - **Database:** simple-supply



3. After you log in, you see the schema page for the reporting database.
4. Use the menu on the left to select agents, blocks, record owners, or other information in the reporting database.

See [adminer.org](https://adminer.org) or [adminer | Docker Documentation](#) for information on using Adminer.



	id	public_key	name	timestamp	start_block_num	end_block_num
edit	1	02d72130827704b3221e4270c35e3e54b2428e94fd59a1506395ff20e806a777b	metmuseum-kiran	1530215196	1	9223372036854775807
edit	2	035d891d2707352f1178708538e76d9f5c0ea22988bce5ad1fd9d05847a6d29207	getty-sasha	1530216266	3	9223372036854775807



## Examining the Sawtooth Logs

The Sawtooth logs often contain useful information for troubleshooting problems with an application or the Sawtooth environment.

- You can view the log files for any a Docker container with the command `docker logs {CONTAINER}`. For example, use the following command to see the log for the Simple Supply transaction processor:  

```
$ docker logs simple-supply-tp
```

See the docker compose file or the topic “About the Docker Containers” for the list of container names.

- You can also connect to a container to look at the log files for that component. By default, Sawtooth log files are stored in the directory `/var/log/sawtooth`. Each Sawtooth component has both a debug log and an error log. For example, the validator has these log files:

```
/var/log/sawtooth/validator-debug.log  
/var/log/sawtooth/validator-error.log
```

For more information, see [Examine Sawtooth Logs](#) and [Log Configuration](#) in the Sawtooth documentation.

Even if the problem isn't obvious, the [Sawtooth community](#) might be able to help. Try to collect all relevant log information before asking for help.



## Shutting Down the Sawtooth Environment

To temporarily stop the Docker containers but keep the Sawtooth environment and all data, enter **CTRL-C** in the same terminal window where you started the Sawtooth environment. You can run **docker-compose up** to start the same containers again.

When you're done with this Sawtooth environment, or if you want to clear everything and start again, you must do a clean shutdown.

1. First, enter **CTRL-C** in the same terminal window where you started the Sawtooth environment, then wait for all containers to stop.

```
^CGracefully stopping... (press Ctrl+C again to force)
Stopping curator-app ... done
Stopping simple-supply-subscriber ... done
Stopping simple-supply-tp ... done
Stopping simple-supply-rest-api ... done
Stopping sawtooth-settings-tp ... done
Stopping sawtooth-rest-api ... done
Stopping sawtooth-devmode-engine-rust-default ... done
Stopping simple-supply-postgres ... done
Stopping sawtooth-validator ... done
Stopping simple-supply-shell ... done
Stopping simple-supply-adminer ... done
```

2. Next, run the following command to remove the Sawtooth environment (containers and all data):

```
$ docker-compose down
Removing curator-app ... done
Removing simple-supply-subscriber ... done
[...snip...]
Removing simple-supply-adminer ... done
Removing network education-sawtooth-simple-supply_default
```



## Module 6: Summary

---

### Module 6 > Summary

Now that you have gone through the Sawtooth Simple Supply code and have successfully run the application, it's time to celebrate!



Paestan Red-Figure Fish Plate  
J. Paul Getty Museum  
Digital image courtesy of the Getty's Open Content Program

Of course, Sawtooth Simple Supply isn't as complex as an actual proof-of-concept or production application. This course has explained the basics of application development for Hyperledger Sawtooth, but you'll probably want to go on to develop a fully featured application. The last module in this course lists helpful resources for advanced application development.

