

Gallery Install and Usage

The main purpose of the Gallery smart contract is to produce an implementation of the IMetadataV1 interface that can be used to provide pointers to files that describe what products should be shown to the user when they visit your Selene Network install.

This interface stores three elements on the blockchain that the Selene Network software uses in order to find and validate the information that it is presenting to the webpage visitor.

These three items are:

URL

A path to the location on the server where the gallery JSON file can be found.

Name

The name of the gallery JSON file.

Hash

A sha1 hash of the gallery JSON file.

All three elements are used to find, read and validate that the information provided though the contract is what was intended by the author.

Installing the Gallery Project

Every Selene Network ZIP file includes a version of the galleryv2.zip file. It can also be downloaded directly from <https://amorstyle.com/download>.

Location on Server

The default location for the Selene Network is a subdirectory off the main domain path like: <https://myserver.dom/dsn>.

The location for the gallery file should be something like:

<https://myserver.dom/src/galleryv2>.

Create a subdirectory on the server at the same location as 'dsn' and call it 'src'. In that folder, create another directory and call it 'galleryv2.' Use lower case letters.

Extract the galleryv2.zip file into this new gallery directory. When done correctly, you should be able to view the default galleryv2.jpg in the browser like:

<https://myserver.dom/src/galleryv2/images/galleryv2.jpg>.

Launching the Gallery Smart Contract

The Gallery smart contract is just like any other Selene Network compatible smart contract that can be launched using ThetaScan.io.

NOTE: Deploying a contract requires the use of Metamask, thus this document assumes that you will be launching the contract using a browser where Metamask is installed and

that you've already using an account that holds a Participant NFT (in the Selene Network).

Launch ThetaScan.io visit the "Smart Contract HQ" and click on "Deploy Contract." At this point, double check in your Metamask account that you have selected the account that you want to be the owner of the contract.

Deploy a Contract

ABI/ JSON Interface

Bytecode

Next

In the "ABI/JSON Interface" location, you need to put the GalleryV2 ABI there. That information can be found in the project directory of the galleryv2.zip file in a file called galleryv2_abi.json.

Likewise, in the "Bytecode" location, you need to put the contract bytecode which can be found in the project directory as galleryv2_bytecode.txt.

In both cases, just open the file in notepad, select all the content (ctrl-a) and then copy & paste it into the two locations. Once done, click the next button.

Setting up the Constructor

There are two addresses that need to be provided to the contract on launch as shown in the following image.

Deploy a Contract

Constructor Variables :

Estimated Transaction Fee:

Estimated Value in USD: \$

Transaction Hash:

Contract Address:

Estimate Gas

Deploy

_theTitled

This is the address of the account that will be considered the property rights holder for this contract. The current active account in Metamask will be assigned as the ‘owner’ during launch. It’s ok for them to be the same address.

_thePennyOracle

The address of the Penny Oracle can be found in the footer of any Selene Network install.

Once you have provided these two accounts, Estimate Gas and Deploy.

NOTE: Save the resulting address for that is the Gallery Address. This is the address that you will provide the Selene Network install.

Building a Gallery JSON file

All galleries are described by a JSON file. The layout is simple and its best to build upon an existing gallery JSON file.

You’ll find a sample gallery.json file in the same directory in the galleryv2.zip file as this document.

Open that file using Notepad because you’ll want to edit a few fields.

Here is an overview of some of the common data pairs. (note that case matters all entries start with lower case letters).

name

This will be the name displayed across the top of the gallery. Best to keep this short.

contracts

This is a collection of project smart contracts. Each one should be a Selene Network compatible smart contract. First one in list is displayed on the top of the page.

homeIcon

This is a fully qualified URL to an image that will be displayed in the upper left-hand corner of the gallery page. It is displayed as 49x48 and can be either a transparent PNG or JPG. When clicked, the user will be redirected to your main website.

image

This is a fully qualified URL to an image that will be displayed if this gallery contract is ever viewed as a project. It is not required.

showSimpleSwap

This should be set to either true or false (Boolean rather than quoted string). When set true, the clickable simple swap link will be shown under the gallery title.

limit

When resolving each product, the code makes a number of RPC calls to get configuration information from the blockchain. It is recommended that the number of projects displayed on each page is kept small. The current max is 12.

bio

This section is optional. If you keep this info, please edit it to reflect your own name and likeness.

Testing your gallery.json file

The best way to double check your file is to use the tools functionality provided through the Selene Network to validate your gallery file before using it.

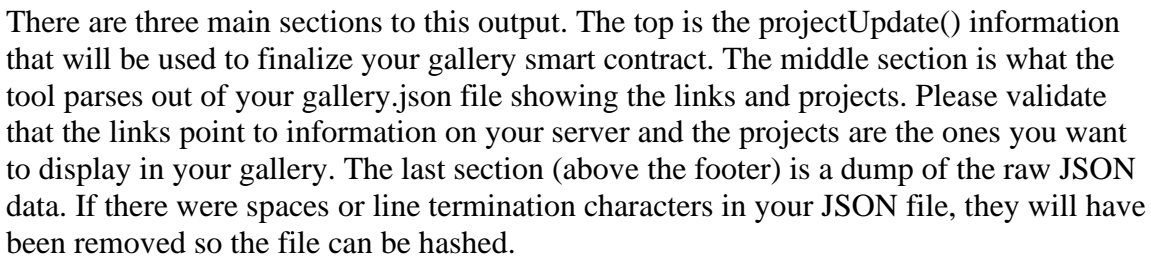
Copy your version of the file to your server: <https://myserver.dom/src>.

You should be able to view your file like: <https://myserver.dom/src/gallery.json>.

To make sure the Selene Network sees your JSON file correctly, view it using the tools like:

<https://myserver.dom/dsn/tools/?file=https://myserver.dom/src/gallery.json>

It should look something like the following.



Once your happy with what you are offering with your gallery file, you are going to need to write that information to your Gallery smart contract so it can be picked up by any Selene Network install.

To do this, once again, you're going to use ThetaScan.io's 'Smart Contract HQ'. Visit that location in your Metamask enhanced browser using the same account that launched the Gallery smart contract. Click on 'interact with.' You'll get something like:

Interact with a Smart Contract

Contract Address

ABI/ JSON Interface

Next

Place your Gallery smart contract address (that you saved from above) into the ‘Contract Address’ location.

Find the Gallery ABI in the ZIP file’s project directory as galleryv2_abi.json. Open that with notepad, select, copy and paste into this “ABI/JSON Interface” location.

After you click Next, find the projectUpdate function and you’ get something like:

Interact with a Smart Contract

projectUpdate

_URI

_Project

_hash

Write

All three of the parameters here can be found in the projectUpdate section from the tools page that reviewed your gallery file. Fill each parameter with the appropriate tools page info (top section) and then press the write button.

Once done, you can confirm it was written correctly by viewing what the projectDataCurrent function returns. Just select this function in the drop down list and click the read button.

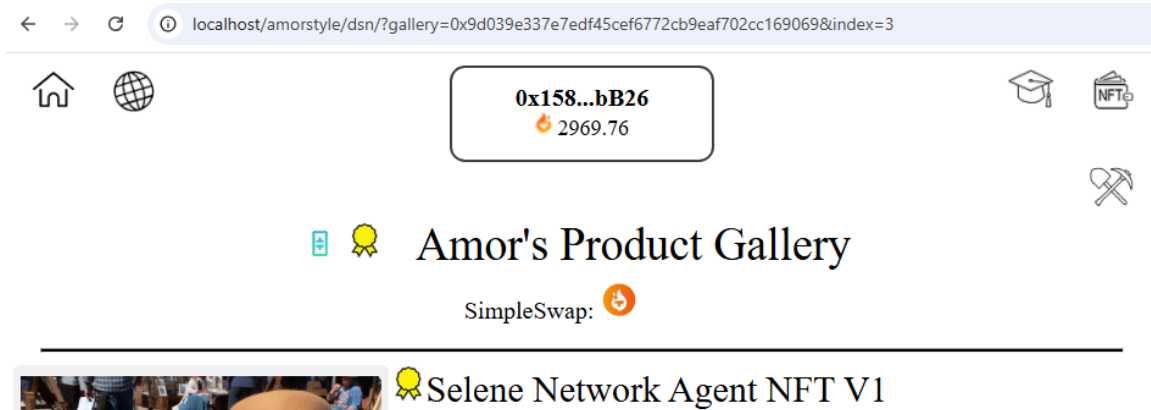
Special note

Note that you can create and write multiple gallery files to the gallery contract. Each time a gallery JSON file is recorded, it gets the next available index into the contract. You can use the projectData function and give it the index (starting at zero) to the specific file that you're interested in accessing.

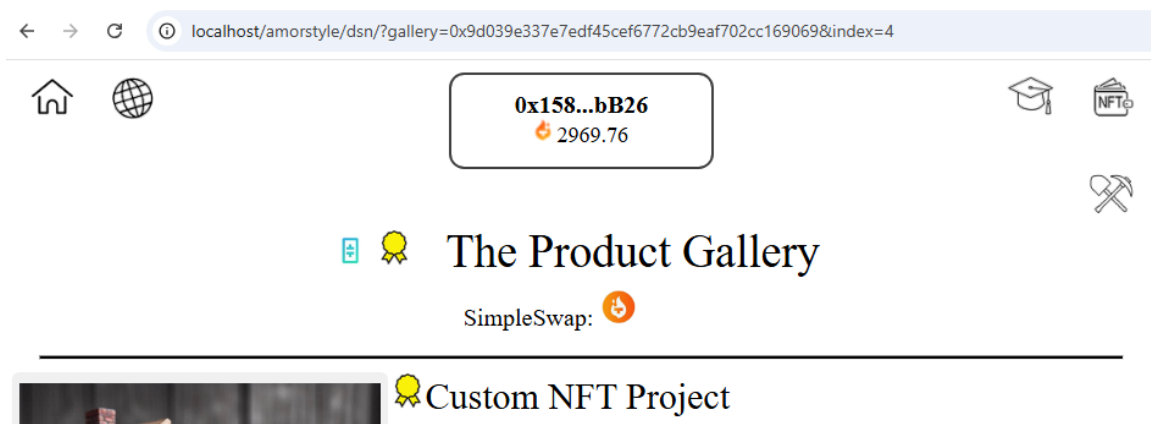
For instance, let's say there are three artists that all use the same website (thus the same Selene Network install), each artist creates a file, say: alice.json, bob.json and cathy.json. If they are each written in alphabetical order to the contract, index 0 will point to alice.json, index1 to bob.json and index 2 to cathy.json.

The default for the Selene Network is to reference the last updated entry, but this can be overwritten by supplying '&index=number' on the URL.

Like this,



And this.



Referencing the Gallery Contract

To set a default gallery, you'll need to edit the `dsnreadme.php` file that is located in the root of the Selene Network install. Find the 'GalleryAddr' entry and set it to your own Gallery contract.

Note that when the Selene Network loads and references your gallery, it will get the most recent version. In other words, the last registered update.

Linking to Custom Gallery

To link to custom galleries, or to use indexing, you'll want to use the URL to specify what you want.

Adding '?Gallery=your_contract' will instruct the code to replace the default gallery with your custom one.

If you want to retrieve a specific indexed version stored in the gallery, add '&index=number' on the URL.

Using the two items together allows you to index any Selene Network compatible Gallery smart contract that is published to the blockchain.

Viewing someone else's default gallery contract

It may be helpful to view how other people use their gallery contracts.

To assist you with this process, the Selene Network will return to you JSON information about the default gallery used on any install. To get that information, the REST API endpoint 'api/gallery' like:

<https://amorstyle.com/dsn/api/gallery>

If you have a JSON content viewer installed in your browser, you'll get a nice formatted display of all the galleries recorded on that contract.

Disclaimer

As with any software, there could be bugs that change or prevent the intended usage. This Selene Network code is provided to you 'as is' to use. If you encounter a bug, please fix it and, if possible, help others resolve the same issue. Also note that this code relies on other 3rd party projects that may, or may not stop working at any time or change rendering what is authored here obsolete. Understand the code and use at your own risk.