SeleneNetwork Software

A Distributed Product & Sales Network

The SeleneNet software is an open source, permissionless, distributed product browser that uses the Theta blockchain where anyone can publish a digital product that can be purchased by anyone. The product seller maintains control of the product and the consumer receives the verifiable digital receipt that provides access to the product.

The SeleneNet software comes with example smart contracts that contain the interfaces needed so that the higher-level code can read the product. These samples provide a rich environment to describe what is being sold and how it might be accessed.

The software allows people that launch products to incentivize the sale of their products by offering commissions. All commissions are settled, via blockchain, at the moment of sale.

A key philosophical point is that the SeleneNet software works to remove the middleman from as many transactions as possible and provide ways to validate sellers and buyers in human friendly ways. Balances are held on contracts until withdrawn by the balance holder.

The Software Components

In order to provide a producer-to-consumer experience, the Selene Network is a collection of components that all work well together. The idea is to empower a digital product creator to make their product available via blockchain while at the same time providing a simple, yet rich avenue by which a consumer can buy and use the product.

To make this more interesting, one of the most important elements to selling products is raising awareness for the offerings. Thus the ‘network’ side of this is the ability of the product creator to incentivize sales. Because every product is different, product creators can choose how much of the sale’s price is commission. Only registered sales agents can compete for the commissions (SeleneNet NFT holders that anyone can buy and use).

Here is a more detailed breakdown outlining the rules of the different players.

The Product Creator

This is the person that has a digital product that they want to offer to the world (A ticket to an event, a membership, book or music download, etc.) and is known as the product creator.

The product creator selects from one of the open-source samples (that best describes their offering) and makes slight modifications to the sample so that it looks like and describes what they are providing. In the simplest example, it’s an image, description, price and ‘terms and conditions’.

The creator places the sample code on their server and writes the smart contract to the blockchain. The SeleneNet core code helps with this process.

In summary, the product creator is publishing a contract that describes the terms & conditions. Once published, the high-level SeleneNet software can interact with it.

The Consumer

The person that consumes the product visits a website that hosts the SeleneNet core code and views the offerings. This is generally the product creator's website, but it could be any SeleneNet install.

The consumer is shown a simple interface that shows the capabilities of the product. There is a common 'buy' button, or other buttons provided by the sample that allow for a common user experience with regards to getting to the product.

If the consumer buys, settlement is handled by the blockchain smart contract and the customer receives a receipt (NFT) in their wallet as a record of the transaction. Once the user holds the receipt, the code can validate their access to the product.

The SeleneNet core code

The SeleneNet core presents information to the visitor based on their permissions with regards to the product they are viewing. If the visitor doesn't hold a receipt of the product, they are shown a 'buy' button. If they already bought the product, they are shown actions that they can do with the product (like transfer it to someone else). Or, if it's a membership type product, the status of the membership is displayed. If the visitor is the product creator, other functionality is displayed. If the visitor is an agent in the network and they've earned a balance on the contract, that will be shown.

The core code is two downloadable open-source ZIP files that can be easily installed and configured on any server. Part of the SeleneNet philosophy is to make it simple for anyone to access their property held in any SeleneNet compatible contract, thus every install allows anyone to view and interact with any other compatible contract.

The Sales Agents and Web access providers

Because anyone can install the SeleneNet core code and provide access to blockchain products, sales agents may also install the code. This allows sales people to build out branded websites that provide access to retrieving any balance they may hold on products.

Expected Installation

The SeleneNet core code is split up into two different extractable ZIP files. One is the bulk of the code that provides the common experience. As updates find their way into the code, it is expected that the existing install can just be removed and the new install can replace it. Thus, it is not expected that anyone change code in the core.

To make the SeleneNet more customizable, the second file is provided that allows for install customizations. Thus, the installer can make customizations and expect them to work over installs.

The other key item about the SeleneNet core code is that it is a collection of script code which both runs on the server and the client machine. An effort has been made to keep the dependencies to as few as possible in order to keep the overall project manageable.

Overview of Install

The product was designed to be installed on a server that has three or four folders off the root. Those folders are ‘dsn’, ‘src’, ‘nfts’ and ‘private’. Permissions should be set so that anyone can read the first three, but private can only be accessed by code on the server. The private location is for digital products that should not be seen or accessed by anyone that is not approved for access.

The main directory is the ‘dsn’ folder. Copy the dsn.zip file to your server and then extract it in the root and it will create this directory. It is expected that you don’t make any changes to this core code. If you do, make sure your code makes it back to the master so that it can be propagated forward in a compatible way. A typical update to a new version involved deleting the old ‘dsn’ folder and extracting the new code which rebuilds that location.

The configuration information is stored in the ‘src’ folder. With the first install, you upload the src.zip file to your server and then extract in the root so that it creates the ‘src’ folder. This folder contains files that are used by the core to override the default values. For example, you might change the default gallery from the default to your custom showing off your projects rather than the default. It is not expected that src.zip file will change very often and that it maintains backward compatibility. Thus, a single extract should hold until there is a breaking change.

The currently released version of the SeleneNet core code is available on https://selenenet.org/download.

Example:

If your server is referenced like: https://myCoolSrv.ext, the SeleneNet install should be able to be referenced like http://myCoolSrv.ext/dsn.

Product Server Structure

For product creators, you’ll also create a ‘nfts’ folder off the root. Inside that folder, every new folder is a new product. Each sample demonstrates the file and folder structure that is needed for the sample.

Use a reasonable naming structure so if you create hundreds of products you’ll be able to find and manage them better.

Disclaimer

The SeleneNet core software is currently being developed, modified, and enhanced. Even though the intent is to maintain backward compatibility as described above, enhancements or new features my prevent full support as stated.

If you want changes made to the core code, get the SeleneNet creator to incorporate those changes or be prepared to maintain your fork. Keep in mind that when a change is better for everyone, and it aligns to the core philosophy, there is a high probability that it will be incorporated for everyone to enjoy.

And, software has bugs and there are dependencies. The SeleneNet creator has a vested interest in seeing the code continue to work, but if there are circumstances beyond what is

fixable could cause problems. Because of this, the code is opensource. If you find a bug, fix & report it.