SMART CONTRACTS LAB

Liam Kane

Functional Platform

SCL Oracle API Guide

1. Randomorg_api

This API from random.org allows receivers to order random integers. The required queries are structured as follows: **“<# of random numbers> <lower bound> <upper bound>”**. The advantage of randomorg_api compared to random_api is that random.org offer **true randomness** as opposed to pseudo-randomness by making use of atmospheric noise (Haahr, n.d.).

1.1 Order a single random integer

- query: “1 <lower bound> <upper bound>”
- see result in “get_single_random_number” / see result in “getData”

1.2 Order multiple random integers

- query example: “4 <lower bound> <upper bound>”
- call “multiple_random_numbers” → see result in “get_multiple_random_numbers”

2. CSV_api

This API was created in-house. It allows senders to upload their findings into their SCA folder as CSV files. Receivers can then query these CSV files and receive the requested data via the oracle. The required queries are structured as follows: **“<CSV file> <index bool> <row> (<column>)”**. Note that “/docker_sca” should always be used as the root directory of the CSV file. The “index bool” should be provided by the sender and tells the API whether the first column of the CSV file is reserved for the index and thus should be ignored. The “column” argument must only be given if one certain datapoint, in other words one specific combination of rows and columns, is ordered.

2.1 Order a single CSV file datapoint

2.1.1 Order a single integer from a CSV file

- query example: “/docker_sca/sender/data_reader/example_csv_file.csv true 1 2”
- see result in getData

2.1.2 Order a single non-integer from a CSV file

- query example: “/docker_sca/sender/data_reader/example_csv_file.csv true 4 1”

- call “single_csv_datapoint” → see result in “get_single_csv_datapoint”

2.2 Order a CSV file row

- query example: “/docker_sca/sender/data_reader/example_csv_file.csv true 3”
- call multiple_csv_datapoints → see result in get_multiple_csv_datapoints

Remarks:

- Both randomorg_api and CSV_api require helper files. Make sure that “multiple_numbers_helper.sol” and “words_helper.sol” are imported into the **same directory** as your smart contract (you can use SCL_Testmodul.sol as a test smart contract). Make sure to include the following import statements in your smart contract:
  - import "./SCL_informed.sol";
  - import "./multiple_numbers_helper.sol";
  - import "./words_helper.sol";

- SCL_Testmodul.sol provides the correct implementation of helper function to make special datapoints such as multiple random numbers or certain csv datapoints visible. Without these, the receiver cannot see the final ordered datapoint, just an interim integer that requires further manipulation. These helper functions are listed above. After calling the helper functions, the final datapoints are stored for the receiver to use as they wish. Additionally, getter functions are provided within SCL_Testmodul.sol, which make these final datapoints visible, which is useful particularly for RemixIDE.

- Data ordered over the oracle can only be of type int88, which only allows for integers with 26 digits or less. Therefore, the number of random integers or the length of the CSV file datapoint / CSV row, is limited. Only very short CSV file rows are meant to be ordered, otherwise the order will throw an error. The 26 digits include the necessary metadata, which means that the effective length of the ordered CSV data must be shorter than 26, unless the CSV datapoint is a single integer, in which case no metadata is required. This can still be very useful for very many cases, however a useful future endeavour would be to modify the oracle to allow for longer integer ordering, or even alternative data types, which would forego the need to transform strings into integers and back again, reducing the amount of metadata.

- Although this guide instructs the user to manually call certain functions such as “multiple_csv_datapoints” in RemixIDE, this needn’t be so cumbersome in reality, as each receiver could automate these function calls however they wish.

Bibliography

Haahr, M. (n.d.). *RANDOM.ORG - True Random Number Service*. RANDOM.ORG. https://random.org/