

JS JavaScript

Описание

Узел создает новый набор данных с заданным составом и параметрами полей и заполняет его данными в ходе выполнения кода JavaScript. Для заполнения выходного набора в коде возможно использование данных входных портов. Состав и параметры полей выходного набора могут задаваться как в мастере настройки узла, так и из кода JavaScript.

Вход

-  **Входной источник данных** — таблица данных, необязательный порт.
-  **Добавить еще один порт** — создает новый порт *Входной источник данных[N]*, где N – порядковый номер порта.
-  **Входные переменные** — переменные, необязательный порт.

Выход

-  **Выходной набор данных** — таблица данных.
-  **Добавить еще один порт** – создает новый порт *Выходной набор данных[N]*, где N – порядковый номер порта;

Мастер настройки

Последовательно выполняются следующие этапы настройки:

- Настройка столбцов выходных таблиц;
- JavaScript.

Настройка столбцов выходных таблиц

Страница мастера *Настройка столбцов выходных таблиц JavaScript* предназначена для создания столбцов в выходных портах узла. Если узел имеет несколько выходных портов, то столбцы для каждого из них можно задать в отдельных вкладках *OutputTables[0]*, *OutputTables[1]* и т.д. Если узел имеет только один выходной порт, то вкладки не отображаются и столбцы задаются для единственного выходного порта.

Столбцы выходных наборов можно задать как на данной странице мастера, так и динамически, в ходе исполнения кода JavaScript. При установке флага *Разрешить формировать выходные столбцы из кода* доступно динамическое создание, изменение и удаление выходных столбцов.

Примечание: Динамическое создание, изменение и удаление выходных столбцов возможно в коде до момента добавления первой записи в выходной набор, т.е. до первого вызова метода `Append()` объектов `OutputTable` и `OutputTables[N]`.

JavaScript

Страница *JavaScript* содержит редактор исполняемого узлом кода. По кнопке Предпросмотр... в отдельном окне выводится часть данных результирующих наборов и консоль отладки кода.

Примечание: При нажатии на кнопку *Предпросмотр...* активируются все входные порты. При успешном завершении активации открывается окно *Предпросмотра*.

В *Предпросмотре* количество выводимых данных ограничивается:

- Если в узле один выходной порт — отображаются первые 100 строк выходной таблицы.
- Если несколько портов — данные выводятся по вкладкам (`OutputTables[0]`, `OutputTables[1]` и т.д.) и в каждой вкладке отображается не менее 100 строк, если они были добавлены в коде.

Примечание: Количество отображаемых в *Предпросмотре* строк зависит от порядка заполнения выходных таблиц в коде. Вывод их результирующих строк во вкладки `OutputTables[N]` производится до тех пор, пока в каждой из них будет добавлено не менее 100 строк (или не будет достигнут конец таблицы). Например, сначала в коде заполняется набор данных первого порта в общем количестве 200 строк, затем второго в количестве 150 строк. В *Предпросмотре* в такой ситуации будет выведено во вкладке `OutputTables[0]` 200 строк, а во вкладке `OutputTables[1]` - 100 строк.

Поддерживается импорт внешних модулей JS.

Важно: Отсутствует поддержка `Atomics.xx` в JavaScript под Linux. Для реализации ожидания используется `setTimeout`.

См. также: Горячие клавиши редактора кода

Примечание: в узле *JavaScript* все числовые значения представляются в виде вещественного формата двойной точности, соответствующего стандарту IEEE 754-2008. При этом для целых чисел за пределами диапазона $[-(2^{53} - 1), 2^{53} - 1]$ происходит потеря точности.

Доступ из кода JavaScript к данным портов и другим встроенным объектам

Для доступа к данным портов и другим встроенным объектам в контексте выполнения JavaScript-кода предусмотрены следующие объекты:

- Входные наборы данных (`InputTables` , `InputTable`);
- Входные переменные (`InputVariables`);
- Выходные наборы данных (`OutputTables` , `OutputTable`);
- Необходимые перечисления (`DataType` , `DataKind` , `UsageType`).

Вышеуказанные объекты импортируются из встроенного модуля `"builtIn/Data"` . По умолчанию в текст исполняемого узлом кода добавляется строка импорта этих объектов.

В контексте выполнения JavaScript-кода доступны функции *Калькулятора*, которые импортируются из встроенного модуля `"builtIn/Calc"` .

Fetch API

В контексте выполнения JavaScript-кода доступны объекты интерфейса Fetch API, предназначенные для работы с HTTP-запросами и ответами. Перед использованием объекты должны быть импортированы из встроенного модуля `"builtIn/Fetch"` .

File Storage API

В контексте выполнения JavaScript-кода доступны объекты интерфейса File Storage API, предназначенные для работы с файловой системой. Перед использованием объекты должны быть импортированы из встроенного модуля `"builtIn/FS"` .

Обработка ошибок

При вызове окна предпросмотра или выполнении узла выводится сообщение об обнаруженных синтаксических ошибках и ошибках хода выполнения с указанием позиции кода, содержащего ошибку.

Важно: Неперехваченные ошибки в Promise записываются в лог сервера, при этом узел выполняется без сообщений об ошибке.

Статьи в разделе:

- Внешние модули
- Входные наборы
- Входные переменные
- Выходные наборы
- Глобальные функции
- Перечисления
- Функции Калькулятора
- Fetch API
- File Storage API
- Консоль отладки кода
- Описание API
- Горячие клавиши