**Глава 21. Создание пользовательских функций Power Query**

Это продолжение перевода книги Кен Пульс и Мигель Эскобар. Язык М для Power Query. Главы не являются независимыми, поэтому рекомендую читать последовательно.

[Предыдущая глава](http://baguzin.ru/wp/?p=21338) [Содержание](http://baguzin.ru/wp/?p=20852#content) Следующая глава

Работа с пользовательской функцией, как правило, включает три этапа:

1. Один раз выполните сценарий обработки.
2. Преобразуйте сценарий в функцию.
3. Вызовите функцию из другого запроса.

Вернемся к сценарию из [главы 18](http://baguzin.ru/wp/?p=21280), где вы импортировали файл расписания 2015-03-14.txt. Там вы уже создали код для импорта и обработки. Теперь вы хотите применить ту же логику к подобным файлам. Однако вместо того, чтобы изменить *Источник*, вы хотите объединить все файлы (и любые другие, которые позже будут добавлены в папку). Для этого пригодится пользовательская функция.



Рис. 21.1. Ошибка из-за неверного пути к файлу

## Построение сценария

Откройте файл *Custom Functions.xlsx*. Он содержит код, идентичный тому, что был создан ранее, за одним исключением: путь к файлу указывает на папку *Source Files*. Когда вы перейдете к редактированию запроса, вы получаете сообщение об ошибке (рис. 21.1), потому что первый шаг сценария ссылается на расположение, отличное от того, куда вы поместили файлы примеров. Нажмите кнопку *Изменить параметры*. Появится окно импорта. Кликните *Обзор*, найдите и выберите файл 2015-03-14.txt.



Рис. 21.2. Окно импорта файла

Нажмите Ok. Вспомните главу 18: запрос загружает записи из текстового файла, извлекает имена сотрудников в новый столбец и изменяет тип данных:



Рис. 21.3. Финальный вид запроса

### Преобразование запроса в функцию

* Придумайте имя переменной, которая будет содержать данные, подлежащие изменению.
* Поместите в первую строку запроса код *variable\_name (имя\_переменной) =>*
* Найдите строки запроса, где указан путь к файлу, и замените его именем переменной.

Поскольку цель – преобразовать одноразовый сценарий в функцию, в которой вы сможете динамически обновлять путь к файлу, логично дать имя *filepath.* Или выберите иное имя, лишь бы оно не содержало пробелов или специальных символов.

Проводник Windows более «лоялен» к тому, какие символы могут содержаться в пути к папке и имени файла. Power Query более требователен. Например, нельзя использовать восклицательный знак в качестве первого символа.

Отредактируйте запрос, чтобы превратить его в функцию. В Excel пройдите по меню *Данные* –> *Запросы и подключения*. В области *Запросы и подключения* кликните правой кнопкой мыши на запросе *Timesheet* –> *Изменить*. В редакторе Power Query пройдите по меню *Главная* –> *Расширенный редактор*. Поместите курсор перед оператором *let*. Введите *(filepath)=>*

Нажмите Enter. Начало вашего кода должно выглядеть так:



Рис. 21.4. Переменная *filepath* в первой строке

Теперь просмотрите код, и замените путь к файлу именем переменной:

* Найдите путь к файлу в строке, которая начинается с *Source*
* Выделите путь к файлу без кавычек
* Нажмите Ctrl+C, чтобы скопировать путь в буфер для последующего использования
* Удалите кавычки, окружавшие путь к файлу
* Введите вместо пути к файлу имя переменной *filepath*



Рис. 21.4. Переменная *filepath* вместо пути к файлу

В расширенном редакторе нажмите *Готово*, и ваш запрос резко изменится:



Рис. 21.5. Куда подевалась таблица?

Теперь это функция, а не запрос. А функция просит указать переменную. В области ПРИМЕНЕННЫЕ ШАГИ вы потеряли все элементы. Не волнуйтесь: они все еще там. Просто, вы не можете их увидеть.

### Тестирование функции

Встаньте курсором в поле *Введите параметр*. Нажмите Ctrl+V, чтобы вставить из буфера ранее скопированный путь к файлу (без кавычек). Нажмите кнопку *Вызвать*. Запрос загрузит информацию из файла и выполняет все ранее записанные шаги.



Рис. 21.6. Данные, загруженные на основании динамического указания пути к файлу

В поле ПРИМЕНЕННЫЕ ШАГИ пока отражается только один шаг (1). А слева появилось поле *Запросы* (2). В нем показано, что вы вызвали функцию.

Такой вызов функции хорош для тестирования, но не удобен при вызове функции из другого места. Теперь, когда вы протестировали функцию, удалите этот шаг: кликните правой кнопкой мыши на *Вызванная функция* –> *Удалить*. Переименуйте запрос *fnGetTimesheet. Главная* –> *Закрыть и загрузить*. Обратите внимание, что таблица, которая находилась в Excel на листе *Timesheet*, исчезнет. Это связано с тем, что запрос был изменен на функцию. А функции могут быть созданы только в режиме подключения соединения:



Рис. 21.7. Запрос теперь является функцией

### Вызов функции

Теперь вам нужно вызвать эту функцию из другого запроса. Поскольку конечной целью является консолидация всех файлов из папки, можно запустить новый запрос для импорта всех расписаний сразу. В Excel пройдите по меню *Данные* –> *Получить данные* –> *Из файла* –> *Из папки.* Найдите и выберите папку *Source Files*. Нажмите Ok. В окне предварительного просмотра нажмите *Преобразовать данные*. Выберите столбец *Folder Path* –> удерживая нажатой Ctrl, выберите столбец *Name* (именно в таком порядке) –> Щелкните правой кнопкой мыши –> *Удалить другие столбцы*. Перейдите на вкладку *Добавление столбца* –> *Настраиваемый столбец*. Введите формулу:

=fnGetTimesheet([Folder Path]&[Name])

Нажмите Ok. Хитрость в том, чтобы использовать только что определенную функцию, а полный путь к файлу получить путем конкатенации столбцов пути и имени – [Folder Path]&[Name]. Затем функция выполнит все шаги, и вернет таблицу в пользовательский столбец. Эта таблица представляет собой обработанный функцией набор данных:



Рис. 21.8. Предварительный просмотр таблицы

Теперь вы можете удалить столбцы *Folder Path* и *Name* и кликнуть на кнопке *Развернуть* (1 на рис. 21.9), тем самым консолидируя содержимое файлов в одной таблице. Отключите параметр *Использовать исходное имя столбца как префикс* (2), снимите галочки с двух последних столбцов. (3). Нажмите Ok.



Рис. 21.9. Настройка параметров команды *Развернуть*

Переименуйте запрос *Timesheets*. К сожалению, определение типов данных, выполненное в исходном запросе, легшем в основу функции, не сохраняется. Это связано с тем, что тип данных будет поддерживаться при импорте только если во всех объединяемых файлах аналогичные столбцы имеют одинаковый тип. Так что вам нужно еще раз установить типы данных для всех столбцов: *Expenses, Misc Hrs, OT Hrs, Reg Hrs* –> *Десятичное число*, *Work Date* –> *Дата*, *Out* –> *Время*, *Employee* оставьте тип *Текст*. Загрузите запрос на лист Excel: *Главная* –> *Закрыть и загрузить* –> *На существующий рабочий лист*. Для красоты можете удалить первоначальный шаг изменения типов данных. Это 4-й шаг кода, начинающийся #"Changed Type".

### Отладка пользовательских функций

Одна из проблем использования пользовательских функций заключается в том, что вы теряете возможность проходить по шагам сценария. Это делает их отладку сложной задачей. Хотя это и не идеально, можно преобразовать функцию обратно в запрос. В таком временном состоянии теряются возможности использования функции в других запросах (поскольку это уже не функция).

Закомментируйте строку, которая превращает запрос в функцию. Присвойте переменной значение. Добавьте новый шаг после строки let:

*Имя\_переменной = присвоить\_значение,*

Строка должна заканчиваться запятой. Откройте функцию *fnGetTimesheet* в редакторе Power Query –> *Главная* –> *Расширенный редактор*. Внесите два изменения:

//(filepath)=>

filepath = "C:\*yourfilepath*\Source Files\2015-03-14.формат txt",

Помните про запятую в конце строки:



Рис. 21.10. Функция преобразована обратно в запрос

При нажатии *Готово* можно перейти к пошаговому выполнению запроса, и скорректировать сценарий при необходимости. Пока ваша функция находится в режиме отладки, все последующие запросы, которые ссылаются на нее, не будут работать.

Чтобы превратить запрос обратно в функцию, вам снова нужно отредактировать код M:

* Удалите символы // из первой строки.
* Добавьте символы // перед строкой, которая в данный момент объявляет переменную *filepath* (третья строка на рис. 21.10).

Нажмите *Готово*. После этого, ваша функция возобновит свой обычный метод работы, и все запросы, использующие эту функцию, смогут использовать ее снова.