



ООО «КВАНТУМ АРТ»

115184, Москва, Озерковский переулок, д. 12

тел. (495) 783-65-74

Модуль GraphQL для продукта QP8.CMS с поддержкой PostgreSQL

Руководство по установке (Linux)

Москва
2023

ИСТОРИЯ ИЗМЕНЕНИЙ

Версия	Дата	Автор	Описание
1.1	16.10.2023	Молькова М.Е.	Обновление структуры документа
1.0	05.07.2023	Григорьева М.А.	Первичное техническое описание

Оглавление

1. ОБОЗНАЧЕНИЯ	4
2. ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ	5
2.1. ОБЩИЕ ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ	5
2.2. ТЕРМИНЫ QP	6
3. ОБЩИЕ СВЕДЕНИЯ	7
3.1. СОСТАВ МОДУЛЯ	7
3.2. СИСТЕМНЫЕ ТРЕБОВАНИЯ	7
4. УСТАНОВКА QP.GRAPHQL (С ИСПОЛЬЗОВАНИЕМ DOCKER).....	8
5. УСТАНОВКА QP.GRAPHQL (С ИСПОЛЬЗОВАНИЕМ KUBERNETES)	9
6. УСТАНОВКА QP.GRAPHQL (БЕЗ ИСПОЛЬЗОВАНИЯ DOCKER)	10
6.1. При сборке из исходников	10
6.2. При использовании готовых бинарных файлов.....	10
6.3. ОБЩАЯ ЧАСТЬ	10
7. НАСТРОЙКА NGINX.....	12
7.1. NGINX С ИСПОЛЬЗОВАНИЕМ DOCKER	12
7.2. NGINX БЕЗ ИСПОЛЬЗОВАНИЯ DOCKER	12
8. СОЗДАНИЕ ПЛАГИНА QP	13

1. Обозначения

Обозначение	Описание	Пример использования
Технические данные	Используется для выделения различных технических данных в тексте: URL, названия свойств и методов, имена файлов и т. п.	ГПИ Системы доступен по URL <code>https://www.domainname.zone/</code>
Код	Пример кода.	<code>public DataTable Data { get; set; }</code>
Переменная	Используется для указания переменного значения.	Формат URL: Базовый URI/Псевдоним объекта
Требуется дополнения	TBD (to be determined). Указывает, что необходима доработка текста – проверка корректности утверждения, детализация, правка после внесения изменений в документ и т. п.	Система работает с одной БД.
Примечание:	Дополнительные данные справочного характера.	Используется при генерации классов LINQ to SQL.
Внимание:	Важные данные, которые требуется обязательно учитывать.	Опция поддерживается только ASP-сборкой в целях совместимости.

2. Термины и определения

2.1. Общие термины и определения

В таблице ниже приведено описание используемых терминов и определений.

Термин или определение	Описание
API	«Application Programming Interface» (интерфейс программирования приложений) – набор правил по использованию функциональных возможностей Системы, предоставляемый разработчикам для организации взаимодействия сторонних программных продуктов с Системой
QP8.CMS или QP8.CMS с поддержкой PostgreSQL (далее «QP»)	Программный продукт, обладающий широким спектром возможностей для разработки программной части Системы различной сложности
URL	«Uniform Resource Locator» – система унифицированных адресов электронных ресурсов, или единообразный определитель местонахождения ресурса
БД	База данных
ГПИ	Графический пользовательский интерфейс
HTTP	«HyperText Transfer Protocol» – протокол передачи гипертекста
Плагин	Независимо компилируемый программный модуль, динамически подключаемый к основной программе
Docker	Программная платформа для быстрой сборки, отладки и развертывания приложений с помощью контейнеров
Kubernetes	Инструмент оркестрации контейнеров с открытым исходным кодом, который позволяет беспрепятственно развертывать и масштабировать контейнерные приложения, управлять ими в различных производственных средах
nginx	Программное обеспечение с открытым исходным кодом, используемое в качестве почтового или обратного прокси-сервера
Мультитенантность	Архитектура программного обеспечения, в которой один экземпляр приложения обслуживает много клиентов (тенантов)

2.2. Термины QR

В таблице ниже приведены термины QR.

Термин или определение	Описание
GraphQL	Отдельный сервис для доступа к данным QR при разработке новых сайтов/продуктов, подключаемый к QR как плагин
Бэкенд	Копия QR; обладает ГПИ для работы с содержимым БД Системы
Контент	Раздел сайта, отвечающий за содержание пользовательской таблицы БД Системы и ее настройки
Поле	Атрибут контента. С использованием полей формируется структура данных для контента.
Сайт	Набор данных в бэкенде. Допускается создание нескольких сайтов. Содержимое каждого сайта определяется созданным в нём контентом.
Код клиента (англ. «Customer code»)	Уникальный параметр (тенант), определяющий БД, с которой взаимодействует бэкенд QR. Выбирается пользователем при входе в Систему.
Плагин QR	Набор дополнительных настроек, расширяющий стандартные сущности QR, такие как поле, контент, сайт без изменения приложения

3. Общие сведения

3.1. Состав модуля

Модуль GraphQL для QP8.CMS с поддержкой PostgreSQL состоит из:

- сервиса GraphQL API;
- настраиваемого плагина QP.

Внимание: Модуль GraphQL не является мультитенантным и устанавливается для конкретного *customer code*.

3.2. Системные требования

Для установки и использования модуля GraphQL требуется установка продукта QP8.CMS с поддержкой PostgreSQL.



4. Установка QP.GraphQL (с использованием Docker)

1. Скачать архив [graphql-config.tar](#) и распаковать его содержимое в `/etc/qp-graphql-config`.
2. Перейти в папку `/etc/qp-graphql/compose`.
3. В файле `.env` задать строку подключения к ранее развёрнутой базе демо-сайта в параметре `CONNECTION_STRING` (можно скопировать из конфигурационного файла QP).
4. В файле `docker-compose.yml` можно настроить ключ инстанса GraphQL в параметре `QpPluginSettings__InstanceKey` (по умолчанию он соответствует настройкам в БД демосайта)
5. Выполнить команду:

```
sudo docker-compose up -d
```


5. Установка QP.GraphQL (с использованием Kubernetes)

1. Скачать архив [graphql-config.tar](#) и распаковать его содержимое в `/etc/qp-graphql-config`.
2. Перейти в папку `/etc/qp-graphql-config/k8s`.
3. В файле `graphql.yml` можно настроить:
 - строку подключения к ранее развёрнутой базе демо-сайта в параметре `ConnectionStrings__QPConnectionString` (можно скопировать из конфигурационного файла QP);
 - ключ инстанса GraphQL в параметре `QpPluginSettings__InstanceKey` (по умолчанию он соответствует настройкам в БД демосайта).
4. Развернуть сервис командой:

```
kubectl apply -f graphql.yml
```

5. При необходимости настроить ингресс к `graphql-api` в файле `ing.yml`, задав свой DNS вместо `graphql-demosite-rus-api.test` и выполнив команду:

```
kubectl apply -f ing.yml
```

6. Установка QP.GraphQL (без использования Docker)

6.1. При сборке из исходников

1. Извлечь исходные файлы из [репозитория](#) на GitHub.
2. В папке проекта QP.GraphQL.App (где находится файл QP.GraphQL.App.csproj) выполнить команду на публикацию:

```
dotnet publish "QP.GraphQL.App.csproj" -c Release -o bin/release/publish/ -r linux-x64 --self-contained=false
```

3. В папке /home/qp создать подпапку QP.GraphQL и скопировать туда всё содержимое каталога bin/release/publish/, в который осуществлялась публикация. Следует убедиться, что у пользователя QP есть права на чтение и исполнение содержимого папки QP.GraphQL.

6.2. При использовании готовых бинарных файлов

1. Скачать архив [graphql-config.tar](#) и распаковать его содержимое в /etc/qp-graphql-config.
2. Скачать [архив](#), содержащий бинарные файлы GraphQL API.
3. Распаковать полученный архив в домашний каталог пользователя, созданного в рамках установки QP8.CMS (по умолчанию - /home/qp). Должна появиться папка QP.GraphQL.

6.3. Общая часть

1. Перейти в папку QP.GraphQL.
2. В файле appsettings.json в секции ConnectionStrings изменить значение параметра QpConnectionString на строку подключения к ранее развёрнутой базе демо-сайта в параметре (можно скопировать из конфигурационного файла QP).
3. В файле appsettings.json в секции QpPluginSettings можно изменить ключ инстанса GraphQL (по умолчанию он соответствует настройкам в БД демосайта).
4. В файле NLog.config найти internalLogFile и <variable name="logDirectory" value= и прописать там путь /var/log/qp-graphql/.

Внимание: для internalLogFile должен сохраниться конечный файл, т. е. правильно строка будет выглядеть так: /var/log/qp-graphql/internal-nlog.txt.

5. Создать на сервере директорию /var/log/qp-graphql/ и выдать QP-пользователю права на владение директорией.
6. Создать файл qp-graphql.service в папке /usr/lib/systemd/system/ и заполнить его следующим содержимым:

```
[Unit]
```

```
Description=QP GraphQL API
After=qp.service
StartLimitIntervalSec=0

[Service]
Type=simple
Restart=on-failure
RestartSec=5
User=qp
WorkingDirectory=/home/qp/QP.GraphQL/
ExecStart=/bin/dotnet QP.GraphQL.App.dll --urls http://*:6300

[Install]
WantedBy=multi-user.target
```

Внимание: если в конкретном дистрибутиве Linux отсутствует папка `/usr/lib/systemd/system/`, то все изменения следует вносить в папку `/lib/systemd/system/`.

7. Зарегистрировать сервис в автозапуск, выполнив команду:

```
systemctl enable qp-graphql.service
```

8. Запустить сервис, выполнив команду:

```
systemctl start qp-graphql.service
```

7. Настройка nginx

7.1. Nginx с использованием docker

Если nginx при установке QP8.CMS был запущен в docker, то следует:

- 1) выполнить команду:

```
docker-compose down
```

- 2) открыть на редактирование файл `nginx.conf`, предположительно расположенный по пути `/etc/qpconfig/nginx/nginx.conf`;
- 3) в конфигурационный файл добавить секции из файла `/etc/qp-graphql-config/nginx/graphql-nginx.conf`;
- 4) задать свой DNS вместо `graphql-demosite-rus-api.test`;
- 5) выполнить команду:

```
docker-compose up -d
```

7.2. Nginx без использования docker

Если nginx при установке QP8.CMS был запущен не в docker, то следует:

- 1) открыть на редактирование файл `nginx.conf`, предположительно расположенный по пути `/etc/nginx/nginx.conf`;
- 2) в конфигурационный файл добавить секции из файла `/home/qp/graphql-config/nginx/graphql-nginx.conf`;
- 3) задать свой DNS вместо `graphql-demosite-rus-api.test`;
- 4) проверить корректность конфигурации командой:

```
nginx -t
```

- 5) если всё корректно, то перечитать конфигурацию nginx командой:

```
nginx -s reload
```

8. Создание плагина QP

В БД демо-сайта, поставляемой вместе с продуктом QP8.WidgetPlatform, уже имеется созданный плагин (Рисунок 1).

[+ Add New QP Plugin](#)

ID	Name	Description	Service URL	Created	Modified	Last Modified By
5	GraphQL	GraphQL Headless API		7/2/2023 11:26:49 AM	7/2/2023 11:26:49 AM	admin

Рисунок 1 - Плагин демо-сайта

При необходимости создания нового плагина в контекстном меню следует выбрать «New QP Plugin» (Рисунок 2).

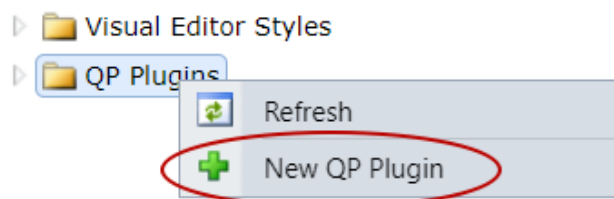


Рисунок 2 - Создание нового плагина

Основные параметры настройки следует выбирать в окне «Basic parameters». Система предлагает два способа его создания: через URL-код (<URL сервиса>/api/qpcontract) или через контракт (Рисунок 3).

Home Customer Code "qa..." - New QP Plugin X

Customer Code "qa_demosite_rus"

Save Refresh

Basic Parameters

Name:

Order:

Plugin creation mode: ☒ By Service Url ☐ By Contract

Service URL:

Рисунок 3 - Параметры создания плагина

Ниже представлен код контракта из демо-сайта:

```
{
  "code": "graphql",
  "description": "GraphQL Headless API",
}
```

```

"version": "0.5.1",
"instanceKey": "B3D1B88B-57F2-4933-BB79-1BF47B27F25A",
"allowMultipleInstances": true,
"fields": [
  {
    "id": 0,
    "name": "ApiKey",
    "description": "Ключ доступа для GraphQL API",
    "valueType": "String",
    "relationType": "Site",
    "sortOrder": 0
  },
  {
    "id": 0,
    "name": "MaxDepth",
    "description": "Максимальная глубина запроса",
    "valueType": "Numeric",
    "relationType": "Site",
    "sortOrder": 5
  },
  {
    "id": 0,
    "name": "MaxComplexity",
    "description": "Максимальный коэффициент сложности документа",
    "valueType": "Numeric",
    "relationType": "Site",
    "sortOrder": 10
  },
  {
    "id": 0,
    "name": "FieldImpact",
    "description": "Максимальное число объектов, возвращаемых каждым
полем",
    "valueType": "Numeric",
    "relationType": "Site",
    "sortOrder": 15
  },
  {
    "id": 0,
    "name": "MaxRecursionCount",
    "description": "Максимальное количество итераций для обхода узлов
дерева",
    "valueType": "Numeric",
    "relationType": "Site",
    "sortOrder": 20
  },
  {

```



```
        "id": 0,
        "name": "IsExposed",
        "description": "Доступность контента в GraphQL API",
        "valueType": "Bool",
        "relationType": "Content",
        "sortOrder": 0
    },
    {
        "id": 0,
        "name": "AliasSingular",
        "description": "Алиас в ед. числе в схеме GraphQL",
        "valueType": "String",
        "relationType": "Content",
        "sortOrder": 5
    },
    {
        "id": 0,
        "name": "AliasPlural",
        "description": "Алиас в множ. числе в схеме GraphQL",
        "valueType": "String",
        "relationType": "Content",
        "sortOrder": 10
    },
    {
        "id": 0,
        "name": "IsHidden",
        "description": "Недоступность поля в GraphQL API",
        "valueType": "Bool",
        "relationType": "ContentAttribute",
        "sortOrder": 0
    }
]
}
```