

«фреймворк Express»

HELLO WORLD

```
const express = require('express');
const app = express();

app.get('/', function (req, res) {
  res.send('Hello World!');
});

app.listen(3000, function () {
  console.log('Example app listening on port 3000!');
});
```





ОСНОВНЫЕ ЧАСТИ

URI / URL

Адрес который мы набираем в браузере
`https://google.com/`

HTTP METHOD

Последовательность из любых символов, указывающая на основную операцию над ресурсом `req.method`
`GET / POST / PUT / DELETE`

HTTP STATUS CODE

Код ответа сервера на запрошенную операцию
`200 OK, 404 Not Found`

HTTP-МЕТОДЫ GET/POST/PUT/DELETE

Тип HTTP-МЕТОДЫ GET/POST/PUT/DELETE



GET



PUT



POST



DELETE



- Запросы с использованием этого метода могут только извлекать данные.
- **POST** используется для отправки сущностей к определённому ресурсу. Часто вызывает изменение состояния или какие-то побочные эффекты на сервере.
- **PUT** заменяет все текущие представления ресурса данными запроса.
- **DELETE** удаляет указанный ресурс.
- **PATCH** используется для частичного изменения ресурса.

КОДЫ ОТВЕТОВ СЕРВЕРА

| Коды | Название | Описание |
|------|-----------------------|--|
| 200 | OK | все прошло так, как должно было |
| 206 | Partial Content | используется расширенными инструментами кэширования |
| 301 | Moved Permanently | запрашиваемый ресурс был перемещен |
| 302 | Found | расположение запрашиваемого ресурса временно изменено |
| 304 | Not Modified | запрашивает ресурс если он подвергался изменениям с момента последнего обновления кэша данного документа |
| 401 | Unauthorized | попытка авторизации была отклонена |
| 403 | Forbidden | доступ к ресурсу, к которому у пользователя нет доступа |
| 404 | Not Found | запрошенный ресурс не может быть найден |
| 500 | Internal Server Error | внутренняя ошибка сервера |

ОСНОВЫ МАРШРУТИЗАЦИИ EXPRESS.JS

Маршрутизация определяет, как приложение отвечает на клиентский запрос к конкретному адресу (конечной точке), которым является URI (или путь), и определенному методу запроса HTTP (GET, POST и т.д.)

Определение маршрута имеет следующую структуру
`app.METHOD(PATH, HANDLER)`

где:

`app` - это экземпляр `express`.

`METHOD` - метод запроса

`HTTP.PATH` - путь на сервере.

`HANDLER` - функция, выполняемая при сопоставлении маршрута.

ПУТИ МАРШРУТОВ

Пути маршрутов, в сочетании с методом запроса, определяют конкретные адреса, в которых могут быть созданы запросы. Пути маршрутов могут представлять собой:

Строки

```
app.get('/about', function (req,  
  res) {  res.send('about');  
});
```

Шаблоны строк

```
app.get('/ab?cd', function(req,  
  res) {  res.send('ab?cd');  
});
```

Регулярные выражения

```
app.get(/a/, function(req,  
  res) {  res.send('/a/');  
});
```

МЕТОДЫ ОТВЕТА СЕРВЕРА

- › `res.download()` - Приглашение загрузки файла.
- › `res.end()` - Завершение процесса ответа.
- › **`res.json()`** - Отправка ответа JSON.
- › `res.jsonp()` - Отправка ответа JSON с поддержкой JSONP.
- › `res.redirect()` - Перенаправление ответа.
- › **`res.render()`** - Вывод шаблона представления.
- › `res.send()` - Отправка ответа различных типов.
- › `res.sendFile` - Отправка файла в виде потока октетов.
- › `res.sendStatus()` - Установка кода состояния ответа и отправка представления в виде строки в качестве тела ответа.

ПРОМЕЖУТОЧНОЕ ПО

Middleware

Функции, имеющие доступ к объекту запроса (req), объекту ответа (res) и к следующей функции промежуточной обработки в цикле “запрос-ответ” приложения. Следующая функция промежуточной обработки, как правило, обозначается переменной next.

```
const express = require('express')
const app = express();
```

```
app.use(function (req, res, next) {
  console.log('Time:', Date.now());
  next();
});
```

```
app.get('/', function (req, res) {
  res.send('Hello World!')
})
```

```
app.listen(3000)
```


ПОЛУЧЕНИЕ ПАРАМЕТРОВ ЗАПРОСОВ



ЗАПРОС ПАРАМЕТРА СТРОКИ В МАРШРУТЕ

Получить значение переменной **name**

`http://localhost:3000/test?name=goit`

GET - запрос

Это свойство представляет собой объект , содержащий свойство для каждого запроса параметра строки в маршруте. Если нет строки запроса, это пустой объект, {}.

```
app.get('/test', function (req, res)
  { console.log(req.query.name);
    res.json(req.query.name);
  });
```

ИМЕНОВАННЫЕ ПАРАМЕТРЫ МАРШРУТИЗАЦИИ

Получить именованный параметр `goit`

Роутинг

`http://localhost:3000/test/goit`

Это свойство тоже является объектом. Например, если у вас есть маршрут `/user/:name`, то `:name` свойство «имя» доступно как `req.params.name`. Этот объект по умолчанию `{}`.

```
app.get('/test/:name', function (req, res) {  
  console.log(req.params.name);  
  res.json(req.params.name);  
})
```

POST ЗАПРОС

Получить параметра из тела запроса

```
const app = require('express')();
const bodyParser = require('body-parser');
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true
}));

app.post('/test', function (req,
  res)
  { console.log(req.body.name);
    res.json(req.body.name);
  })
```



ПРАКТИКА