SPD UNIVERSITY

Node.js: Frameworks

Stanislav Vysotskyi,
FE Lead, Senior Software Engineer, SPD-Ukraine/Poynt

НЕ ЗАБУДЬ ВИМКНУТИ
~~КАМЕРУ~~ ТА МІКРОФОН

SPD UNIVERSITY

Agenda

- Node.js и фреймворки;
- Работа с Express.js;
- Работа с маршрутизацией;
- Работа с middleware.

SPD UNIVERSITY

Что такое Фреймворк?

Фреймворк - программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

Top NodeJS Frameworks

SPD UNIVERSITY

Что такое Express.js

Express.js, или просто Express, фреймворк web-приложений для Node.js, реализованный как свободное и открытое программное обеспечение под лицензией MIT. Он спроектирован для создания веб-приложений и API. Де-факто является стандартным каркасом для Node.js.

Что позволяет Express.js?

- Создавать MVC приложения;
- Создавать REST API.

SPD UNIVERSITY

Начало работы с Express.js

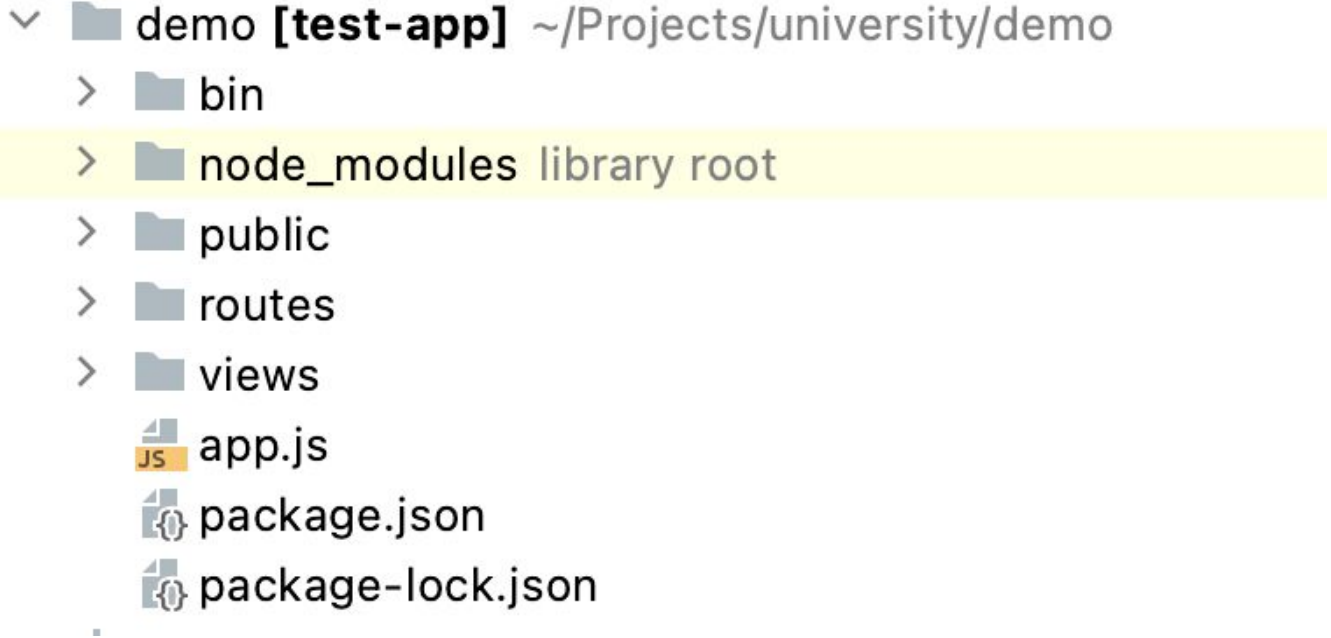
Приложение на Express.js можно сгенерировать с помощью express-generator CLI либо же сконфигурировать его вручную.

npm install -g express-generator

express --view=twig demo

SPD UNIVERSITY

Начало работы с Express.js



SPD UNIVERSITY

Что такое шаблонизатор?

Шаблонизатор (в web) — программное обеспечение, позволяющее использовать html-шаблоны для генерации конечных html-страниц. Основная цель использования шаблонизаторов — это отделение представления данных от исполняемого кода.

SPD UNIVERSITY

Какие шаблонизаторы поддерживает Express?

- ejs;
- handlebars;
- pug;
- hogan.js;
- jade;
- twig.

SPD UNIVERSITY

Где используются шаблонизаторы?

- Рендеринг страницы;
- Рендеринг определенного файла.

SPD UNIVERSITY

Работа с шаблонами?

```
// подключение шаблонизатора
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'twig');

{#Как выглядит код шаблона#}
{% extends 'layout.twig' %}

{% block body %}
  <h1>{{title}}</h1>
  <p>Welcome to {{title}}</p>
{% endblock %}
```

SPD UNIVERSITY

Работа с шаблонами?

```
// Рендеринг страницы и отправка пользователю
res.render( view: 'index',  options: {
  title: 'Express'
});
```

```
// Рендеринг шаблона и отправка пользователю
res.render( view: 'index',  options: { title: 'Express'},  callback: (error : Error , output : string ) => {
  res.send( body: {
    data: output
  });
});
```

SPD UNIVERSITY

Что такое маршрутизация?

Маршрутизация определяет, как приложение отвечает на клиентский запрос к конкретному адресу (конечной точке), которым является URI, и определенному методу запроса HTTP (GET, POST и т.д.). Каждый маршрут может иметь одну или несколько функций обработки, которые выполняются при сопоставлении маршрута.

SPD UNIVERSITY

Как работать с роутингом?

```
import express from 'express';

import usersRouter from './routes/users.mjs';

const app = express();

app.use('/users', usersRouter);

export default app;
```

SPD UNIVERSITY

Как работать с роутингом?

```
import express from 'express';

const router = express.Router();

router.get( path: '/:id', handlers: (request : Request<P, ResBody, ReqBody, ReqQuery, Locals> , response : Response<ResBody, Locals> ) => {
  response.send( body: {
    id: request.params.id
  });
});

router.put( path: '/:id/profile/:profileId', handlers: (request : Request<P, ResBody, ReqBody, ReqQuery, Locals> , response : Response<ResBody, Locals> ) => {
  response.send( body: {
    id: request.body.id
  });
});

export default router;
```

SPD UNIVERSITY

Params, Query, Body, Cookies, Headers...

```
router.put( path: '/:id/profile/:profileId', handlers: (request : Request<P, ResBody, ReqBody, ReqQuery, Locals> ,
  const params = request.params;
  const body = request.body;
  const queryParams = request.query;
  const cookies = request.cookies;
  const headers = request.headers;

  response.send( body: {
    params,
    body,
    queryParams,
    cookies,
    headers
  });
});
```

SPD UNIVERSITY

Как отправить статус-код?

```
router.put( path: '/:id/profile/:profileId', handlers: (request : Request<P, ResBody, ReqBody, ReqQuery, Locals> ,
  response
    .status( code: 404)
    .end();
});
```

SPD UNIVERSITY

Что такое middleware?

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

SPD UNIVERSITY

Основные задачи middleware в Express

1. Выполнение промежуточного кода;
2. Изменение данные до того, как они попадут в обработчик endpoint’a.
3. Завершение запроса.

SPD UNIVERSITY

Как объявить глобальный middleware

```
const removeBodyIdFromRequest = function (request, response, next) {
  delete request.body['id'];
  next()
};

app.use(removeBodyIdFromRequest);
```

SPD UNIVERSITY

Как объявить локальный middleware

```
const removeBodyIdFromRequest = function (request, response, next) {
  delete request.body['id'];
  console.log(request.headers);
  next()
};

router.put('/:id/profile/:profileId', removeBodyIdFromRequest, (request : Request<ParamsDictionary, any, any, ParsedQs, Record<string, any>> ,
  response.send( body: {
    id: request.body.id
  })
});
```

SPD UNIVERSITY

Документация по Express.js:

http://expressjs.com/en/starter/hello-world.html


Q&A
Отвечаем на вопросы, заданные во время встречи.
Помни о просьбе выключить
микрофон и видео

Практическое задание

- Сгенерировать приложение Express.js
- Сделать endpoint **POST /login**, который позволит пользователю авторизоваться в системе (используя зашитые в код логин и пароль). После авторизации программа должна сгенерировать и отправить пользователю токен.
- Сделать endpoint **GET /details**, который будет доступен только для пользователей, которые уже авторизированные (у которых есть токен, который находится в header запросе). Данный endpoint должен отдать пользователю какую-то информацию (что либо) в формате JSON.