УДК 004.432

Студ. Д.В. Сивчик, Д.А. Бабич
Науч. рук. асс. А.Н. Щербакова,
(кафедра информатики и веб-дизайна, БГТУ)

РАБОТА С ФРЕЙМВОРКОМ NEST.JS

Nest.js – это масштабируемая серверная среда JavaScript, созданная на основе TypeScript, которая сохраняет совместимость с JavaScript, что делает его эффективным инструментом для создания надежных серверных приложений. Он имеет модульную архитектуру, которая обеспечивает структурный шаблон проектирования для разработки Node.js [1]. Для начала работы с фреймворком Nets.js необходимо установить Nest CLI, используя команду

```
:npminstall -g @nestjs/cli.
```

Для создания и запуска проекта необходимо выполнить следующие команды:

```
nest new my-nest-project
cd my-nest-project
npm runstart:dev
```

Автор фреймворка был вдохновлен идеями Angular, и NestJS получился ну очень похожим на Angular, особенно в ранних версиях.

Рассмотри основные компоненты приложения на Nest.

Controllers. Слой контроллеров отвечает за обработку входящих запросов и возврат ответа клиенту.

Providers. Почти все является Providers – Service, Repository, Factory, Helper и т.д. Они могут быть внедрены в контроллеры и другие провайдеры. Если сказать языком Angular – то это все `@Injectables.

Modules. Модуль – это класс с декоратором Module(). Декоратор Module() предоставляет метаданные, которые Nest использует для организации структуры приложения. Каждое приложение Nest имеет как минимум один модуль, корневой модуль. Корневой модуль – это место, где Nest начинает упорядочивать дерево приложений. Фактически, корневой модуль может быть единственным модулем в вашем приложении, особенно когда приложение маленькое, но это не имеет смысла. В большинстве случаев у вас будет несколько модулей, каждый из которых имеет тесно связанный набор возможностей. В Nest модули по умолчанию являются синглетонами, поэтому можно без труда делить один и тот же экземпляр компонента между двумя и более модулями.

Middlewares. Middlewares – это функция, которая вызывается перед обработчиком роута. Они имеют доступ к request и response. По сути они являются такими же, как и в express.

Pipes. Pipe должен реализовывать интерфейс PipeTransform. Pipe преобразует входные данные в желаемый результат. Кроме того, это может сойти за валидацию, так как им же возможно генерировать исключение, если данные неверны. Например:

```
@Post()
@UsePipes(new ValidationPipe(createCatSchema))
async create(@Body() createCatDto: CreateCatDto) {
this.catsService.create(createCatDto);
}
```

NestJS относительно новое решение в области бэкенд-разработки, с большим набором функций для быстрого построения и развертывания корпоративных сервисов, которые отвечают требованиям современных клиентов приложений и придерживаются принципов SOLID и приложения двенадцати факторов. NestJS – это express на стероидах [2].

ЛИТЕРАТУРА

1. Community [Электронный ресурс]. – Режим доступа: https://www.digitalocean.com/community/tutorials/how-to-build-a-blog-with-nest-js-mongodb-and-vue-js. – Дата доступа: 17.04.2021.

2. DocumentationNestJS [Электронный ресурс]. – Режим доступа: https://docs.nestjs.com/. – Дата доступа: 17.04.2021.

УДК 004.9

Студ. Е.А. Шуляк
Науч. рук. доц. О.А. Новосельская
(кафедра информатики и веб-дизайна, БГТУ)

ЛЕВЕЛ ДИЗАЙН В КОМПЬЮТЕРНЫХ ИГРАХ

Левел дизайн – это дисциплина разработки игр, связанная с созданием этапов игры. В отличие от дизайна среды, в котором основное внимание уделяется композиции фона и декораций, левел дизайн включает в себя возможности игрока, игровую механику, создание уровней, локаций, миссий, препятствий и обнаруживаемых элементов, которые создают положительный пользовательский опыт.

Игровые уровни как концепция являются пережитком ранних игр, в которых игроку приходилось переходить с одной стороны уровня на другую, избегая или преодолевая угрозы. Уровни будут служить инструментом для отметки прогресса и постепенного увеличения сложности. Они также были необходимы из-за технологических