**Фреймворк Angular 9**

**Лабораторная работа № 1**

**Тема: Работа с компонентами**

**Задача. Инициализировать проект на базе фреймворка Angular и создайть в нем два компонента отвечающих за форму добавления новых постов и отображения постов. Отобразите посты из массива объектов вида в шаблон основного компонента.**

**[{**

**id: numer,**

**titile: string,**

**text: string**

**},**

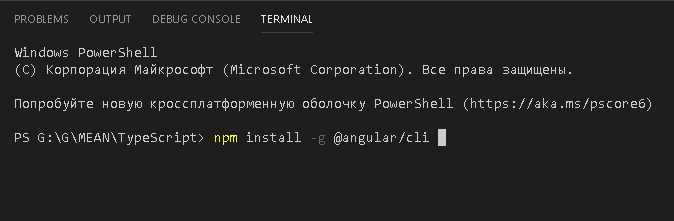
**…..**

**]**

**Ход работы**

1. Before you begin, make sure your development environment includes Node.js and an npm package manager Node.js (<https://nodejs.org/uk/>)
2. После установки Node.js  создайте папку для проекта и в терминале перейдите в нее. В терминале запустите команду

npm install -g @angular/cli



1. Инициализируем проект на фреймворке **Angular**

ng new my-app

Обязательно перейдите в эту папку в терминале

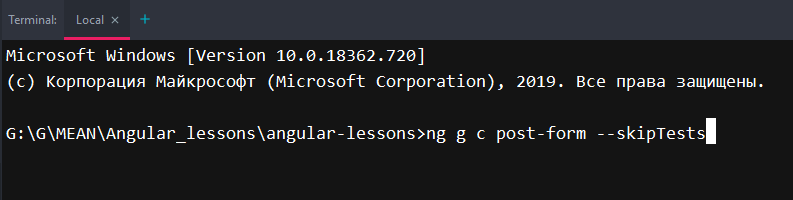
cd my-app

1. Создать два компонента в проекте **post-form, post**

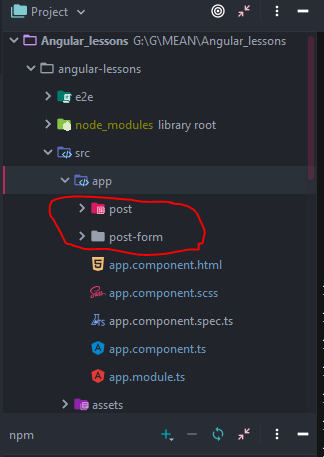
ng g c post --skipTests

ng g c post-form --skipTests

!!! Если возникают проблемы в процесе инициализации проекта и создания компонент, читайте гайд [**https://angular.io/guide/setup-local**](https://angular.io/guide/setup-local)



В итоге структура проекта у вас должна выглядеть следующим образом



Для более наглядной работы приложения создадим стили в файле styles.scss нашего приложения.

@import url('https://fonts.googleapis.com/css?family=Roboto');  
  
\* {  
 box-sizing: **border-box**;  
 margin: 0;  
 padding: 0;  
}  
  
body {  
 font-family: 'Roboto', **sans-serif**;  
 font-size: 1**rem**;  
 line-height: 1.6;  
 background-color: #fff;  
 color: #333;  
}  
  
.container {  
 max-width: 1000**px**;  
 margin: 0 **auto**;  
 padding-top: 1**rem**;  
}  
  
a {  
 text-decoration: **none**;  
}  
  
a:hover {  
 color: #666;  
}  
  
ul {  
 list-style: **none**;  
}  
  
img {  
 width: 100%;  
}  
  
.btn {  
 display: **inline-block**;  
 background: #333333;  
 color: #fff;  
 padding: 0.4**rem** 1.3**rem**;  
 font-size: 1**rem**;  
 border: **none**;  
 cursor: **pointer**;  
 margin-right: 0.5**rem**;  
 transition: **opacity** 0.2**s ease-in**;  
 outline: **none**;  
}  
  
.btn:hover {  
 opacity: 0.8;  
}  
  
.form-control {  
 display: **block**;  
 margin-top: 0.3**rem**;  
}  
  
.card {  
 padding: 1**rem**;  
 border: #ccc 1**px dotted**;  
 margin: 0.7**rem** 0;  
}  
  
input,  
select,  
textarea {  
 display: **block**;  
 width: 100%;  
 padding: 0.4**rem**;  
 font-size: 1.2**rem**;  
 border: 1**px solid** #ccc;  
 margin: 1.2**rem** 0;  
}  
  
hr {  
 margin: .5**rem** 0;  
}

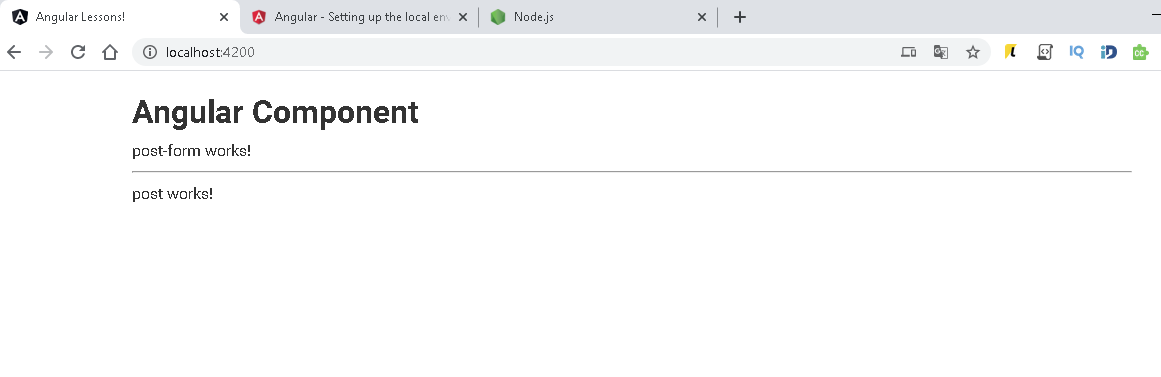
В главном шаблоне приложения app.component.ts зарегистрируем созданные компоненты и отделим их для наглядности вертикальной чертой.

<div class**="container"**>  
 <h1>**Angular Component**</h1>  
 <app-post-form></app-post-form>  
 <hr />  
 <app-post></app-post>  
  
  
</div>

Запустим наш проект и убедимся что он работает

ng run start

В браузере по адресу <http://localhost:4200/> будет отображена структура будущего приложения



1. Создадим форму для создания поста в шаблоне компонента PostFormComponent

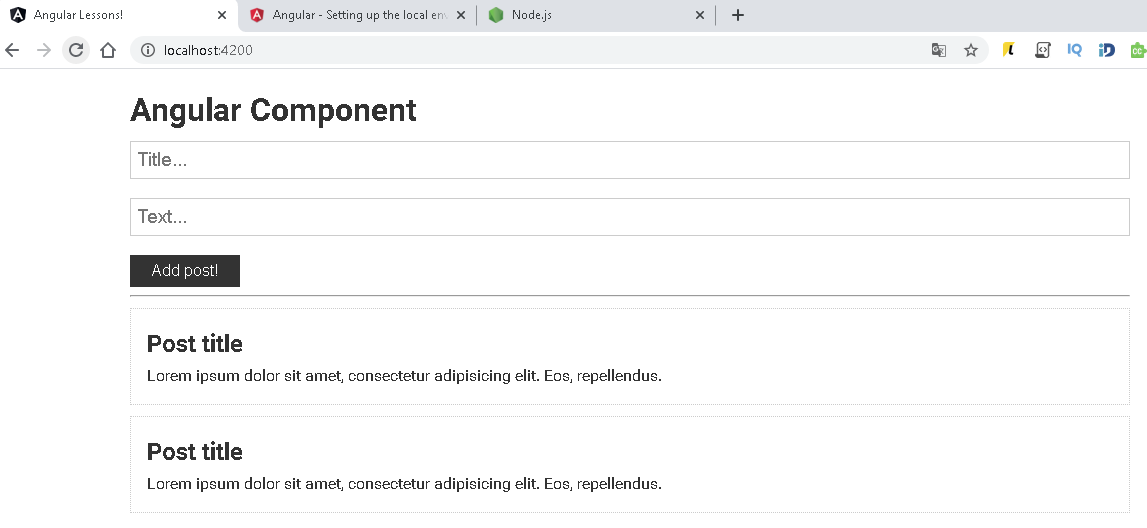
В файле post-form.component.html создайте два поля для ввода информации и кнопку для добавления поста.

<div>  
 <input type**="text"** class**="form-control"** placeholder**="Title..."**>  
 <input type**="text"** class**="form-control"** placeholder**="Text..."**>  
 <button class**="btn"**>**Add post!**</button>  
  
</div>

1. Создадим шаблон поста состоящий из заголовка и текста в файле form.component.html

<div class**="card"**>  
 <h2>**Post title**</h2>  
 <p>**Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eos, repellendus.**</p>  
</div>

В браузере приложение примет следующий вид.



**!!!Самостоятельно добавьте второй и третий пост в главном шаблоне.**

1. Создадим интерфейс поста в файле app.component.ts, для описания объекта (Это считается хорошей практикой описания объектов в TS). Сразу же его экспортируем

export interface Post {  
title: string  
text: string  
id?: number  
}

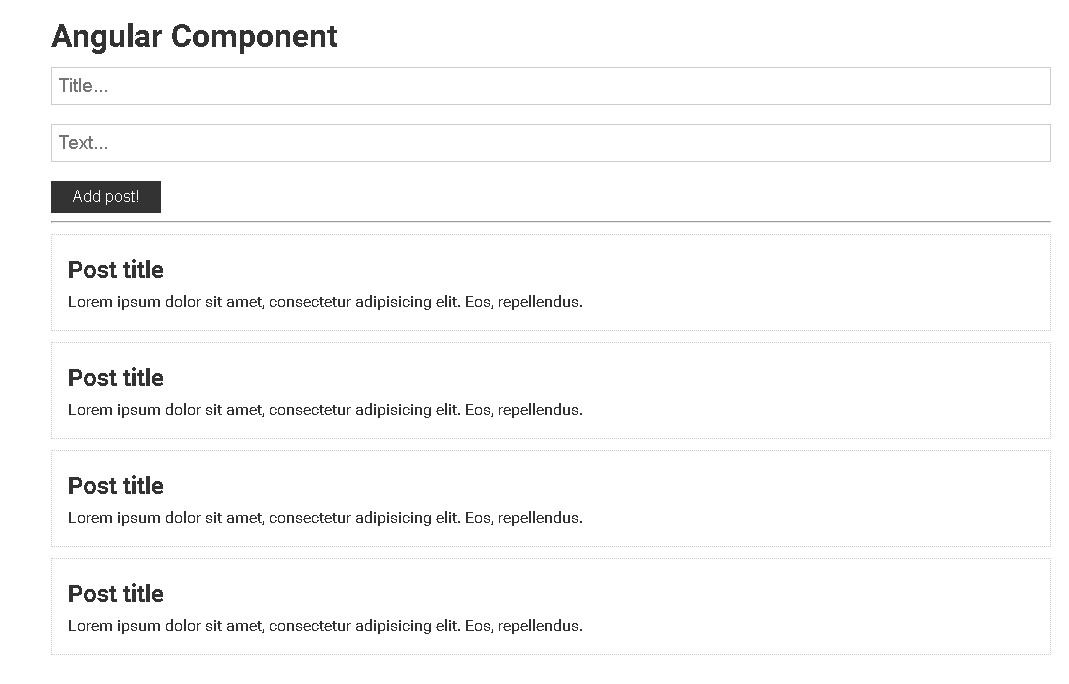
В этом же файле app.component.ts в классе заведем переменную posts которая будет содержать в себе все посты.

export class AppComponent {  
posts: Post[] = [  
 {title:"React", text: "JavaScript-библиотека для создания пользовательских интерфейсов", id: 1},  
 {title:"Angular", text: "Angular is an app-design framework and development platform for creating efficient and sophisticated single-page apps.", id: 2},  
 {title:"Vue", text: "Vue (pronounced /vjuː/, like view) is a progressive framework for building user interfaces", id: 3},  
 {title:"Node.js", text: "Node.js® — це JavaScript–оточення побудоване на JavaScript–рушієві Chrome V8.", id: 4}  
]  
}

1. Перейдем в шаблон app.component.html и добавим директиву \*ngFor для компонента постов.

<div class**="container"**>  
 <h1>**Angular Component**</h1>  
 <app-post-form></app-post-form>  
 <hr />  
 <app-post *\*ngFor***="**let *p* of posts**"**></app-post>  
  
</div>

В результате браузер должен показать правильное количество постов.



1. Для передачи данных в другой компонент нужно воспользоваться декоратором @Input()

Поэтому в файл post.component.ts передадим параметр post и укажем декоратор @Input()

export class PostComponent implements OnInit {  
@Input() post: Post  
 constructor() { }  
  
 ngOnInit() {  
 }  
  
}

Для того чтобы в дальнейшем передавать параметры мы должны забандить этот атрибут в файле app.component.html

<div class**="container"**>  
 <h1>**Angular Component**</h1>  
 <app-post-form></app-post-form>  
 <hr />  
 <app-post *\*ngFor***="**let *p* of posts**"** [post]**="***p***"**></app-post>  
  
</div>

После чего post будет содержаться в себе объекты постов из массива posts

Далее в шаблоне post.component.html нужно интерполировать значения свойств нашего объекта post

<div class**="card"**>  
 <h2>{{post.title}}</h2>  
 <p> **#**{{post.id}} **-** {{post.text}} </p>  
</div>

В результате информация из объекта будет выводиться в браузер:

