**Фреймворк Angular 9**

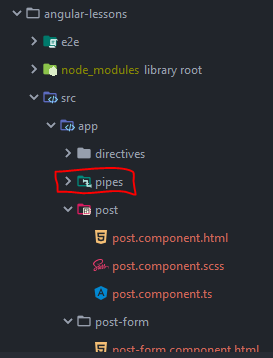
**Лабораторная работа № 4**

**Работа с пайпами**

**Задача. Создать пайп для поиска и фильтрации информации в постах.**

**Ход работы**

1. Создадим директорию в которой будем хранить файлы с пользовательскими пайпами и назавем ее pipes.

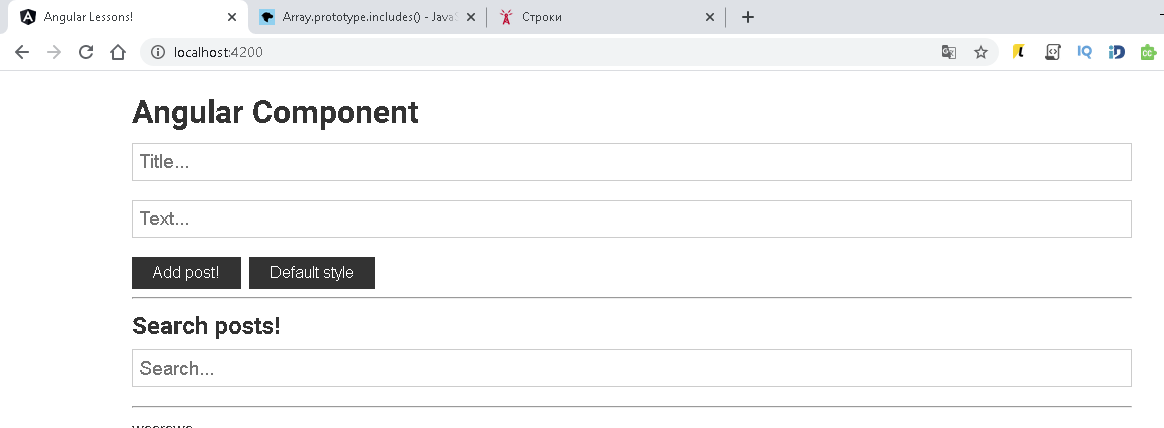


1. Используя Angular CLI создадим пайп. Для этого наберите в командной строке

**ng g p pipes/filter --skipTests**

1. Перед редактированием пайпа создадим поле в шаблоне для ввода информации в файле post-form.component.html.

<div>  
 <input type**="text"** class**="form-control"** #*myInputText* placeholder**="Title..."** [(ngModel)]**="**title**"** >  
 <input type**="text"** class**="form-control"** #*myInputTitle* placeholder**="Text..."** [(ngModel)]**="**text**"**>  
 <button class**="btn"** (click)**="**addPost()**"**>**Add post!**</button>  
<button class**="btn"** (click)**="**onLoadDefault()**"**>**Default style**</button>  
 <hr/>  
 <h2>**Search posts!**</h2>  
 <input type**="text"** class**="form-control"** placeholder**="Search..."**>  
</div>



1. Задача поставлена таким образом, что при вводе в поле любой информации посты автоматически фильтровались и показывались только те у которых название заголовков соответствуют введённому фильтру. Поэтому необходимо сделать обработку события нажатия на клавиатуру и перехват введённых символов. Для этого в файле post-form.component.html добавим event binding (keyup) и повесим на него метод onChangeSearch()

<div>  
 <input type**="text"** class**="form-control"** #*myInputText* placeholder**="Title..."** [(ngModel)]**="**title**"** >  
 <input type**="text"** class**="form-control"** #*myInputTitle* placeholder**="Text..."** [(ngModel)]**="**text**"**>  
 <button class**="btn"** (click)**="**addPost()**"**>**Add post!**</button>  
<button class**="btn"** (click)**="**onLoadDefault()**"**>**Default style**</button>  
 <hr/>  
 <h2>**Search posts!**</h2>  
 <input type**="text"** (keyup)**="**onChangeSearch(titleSearching)**"** class**="form-control"** placeholder**="Search..."** [(ngModel)]**="**titleSearching**"** >  
</div>

1. В файле post-form.component.ts опишем метод onChangeSearch() который должен передать главному компоненту изменения в фильтре

titleSearching = ''  
  
 @Output() titleSearch = new EventEmitter<string>()  
  
onChangeSearch(change: string) {  
 this.titleSearch.emit(change)  
}

1. В шаблоне главного компонента создадим прослушку метода titleSearch. Также запишем пайп для обработки информации в цикле. Для этого в файле app.component.html добавьте следующую запись

< <app-post-form  
(addPostUser)**="**updatePosts(*$event*)**"**(titleSearch)**="**onChangeSearch(*$event*)**"**>  
  
</app-post-form>  
<hr />  
<app-post *\*ngFor***="**let *p* of posts | filter: titleS**"** [post]**="***p***"** (postid)**="**removeIdPost(*$event*)**"** >

А в файле app.component.ts реализуем метод фильтрации постов onChangeSearch()

titleS=''  
onChangeSearch(event) {  
  
this.titleS=event  
}

В созданном файле filter.pipe.ts опишем логику пайпа, который обязательно должен состоять из метода transorm() принимающего в себя ряд параметров для преобразования данных

export class FilterPipe implements PipeTransform {  
 transform(post: Post[], titleSearch: string): Post[] {  
 if(!titleSearch.trim()) {  
 return post  
 } else {  
 return post.filter(item=>{  
 return item.title.toLowerCase().includes(titleSearch.toLowerCase())  
 })  
 }  
  
 }  
  
}

**!!!Самостоятельно.**  **Добавьте кнопки фильтрации информации по тексту поста и заголовку поста. Реализуйте логику фильтрации информации в зависимости от того какая кнопка нажата.**

