Создание web приложений с помощью ASP.NET Core 1.0 и Angular 2

В этой лабораторной работе мы создадим Angular 2 приложение, состоящее из трёх страниц - главной, страницы списка клиентов и страницы редактирования клиента.

Код данной лабораторной работы можно найти по адресу:
https://github.com/spugachev/DevCon2016

Оглавление

Настройка среды

- Установите Visual Studio Code: https://code.visualstudio.com/
- Для OS X установите Home Brew: http://brew.sh/
  После установки выполните:
  ```
  brew update
  brew install openssl
  brew link --force openssl
  ```
- Установите .NET Core: https://www.microsoft.com/net/core
- Для OS X установите Mono[1]: http://www.mono-project.com/download/
- Установите Node.js. Для Windows обязательно обновите версию, установленную вместе с Visual Studio: https://nodejs.org/
- Установите компилятор TypeScript:
  ```
  npm install -g typescript
  npm install -g typings
  ```
- Установите Bower:
  ```
  Npm install -g bower
  ```
- Установите Yeoman:
  ```
  npm install -g yo
  ```
- Установите генератор для Yoman генератор для ASP.NET:
  ```
  npm install -g generator-aspnet
  ```

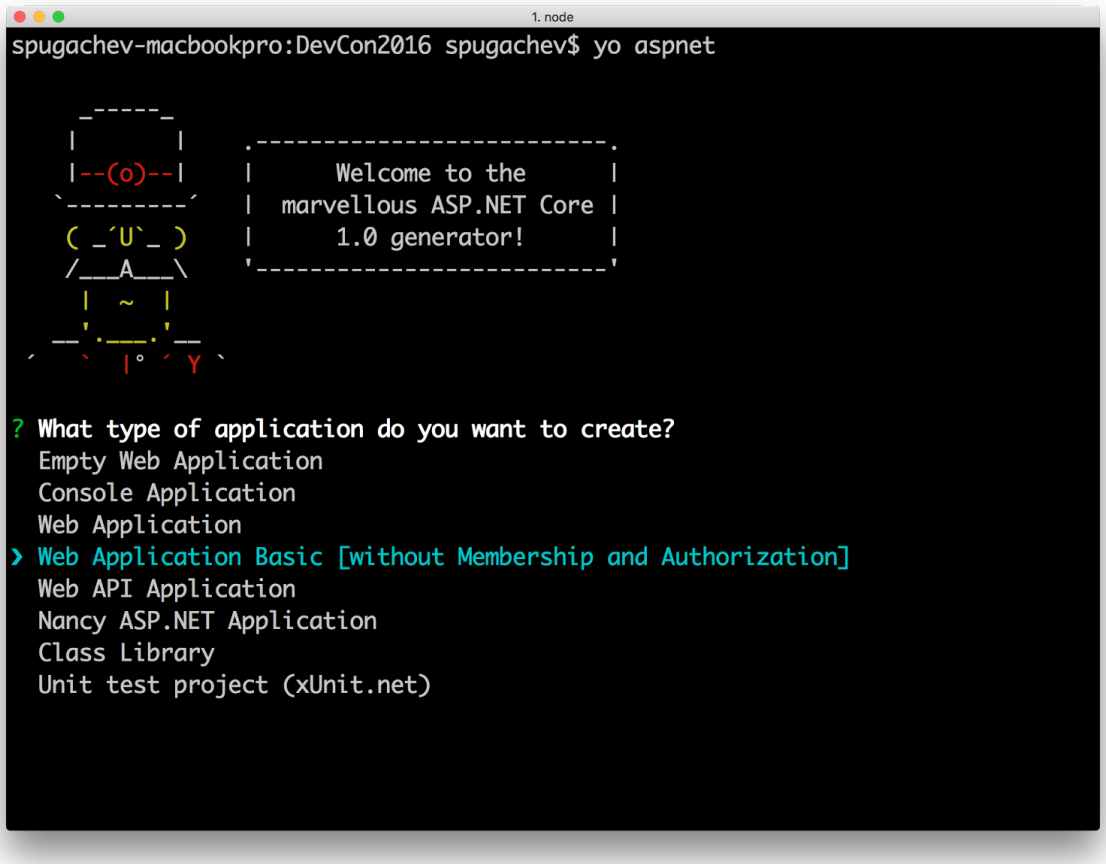
[1] `export MONO_MANAGED_WATCHER=disabled`

Создание проекта ASP.NET Core

Откройте консоль в том каталоге, где вы хотите разместить проект и введите следующую команду для генерации ASP.NET Core проекта:

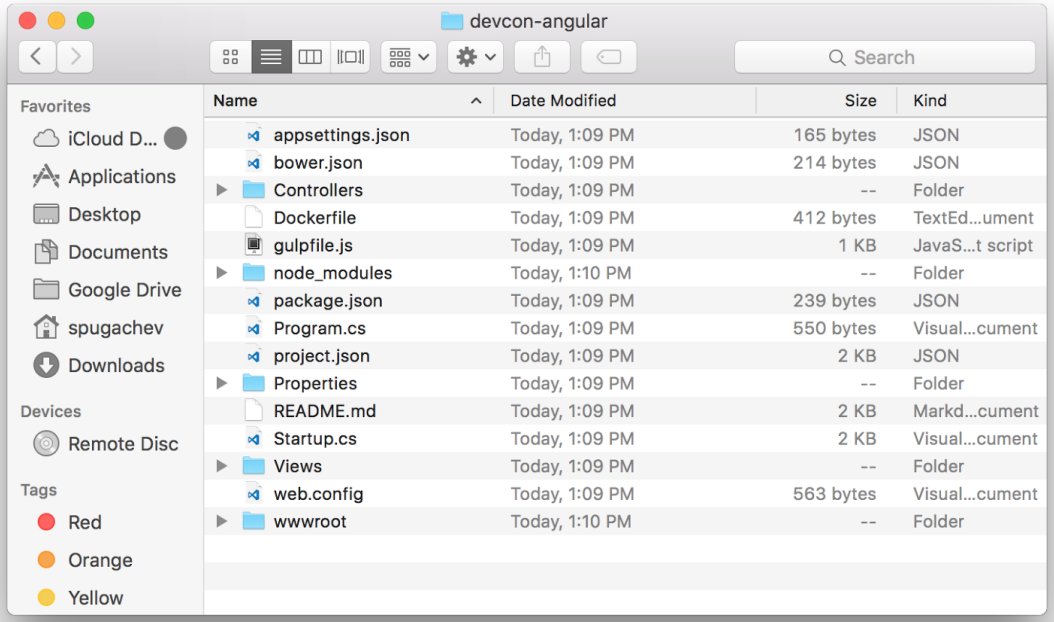
```
yo aspnet
```

Выберите тип проекта `Web Application Basic` , в качестве UI framework выбеерите Bootstrap, и назовите проект `devcon-angular`.



Перейдите в консоле в папку вновь созданного проекта:

```
cd devcon-angular
```



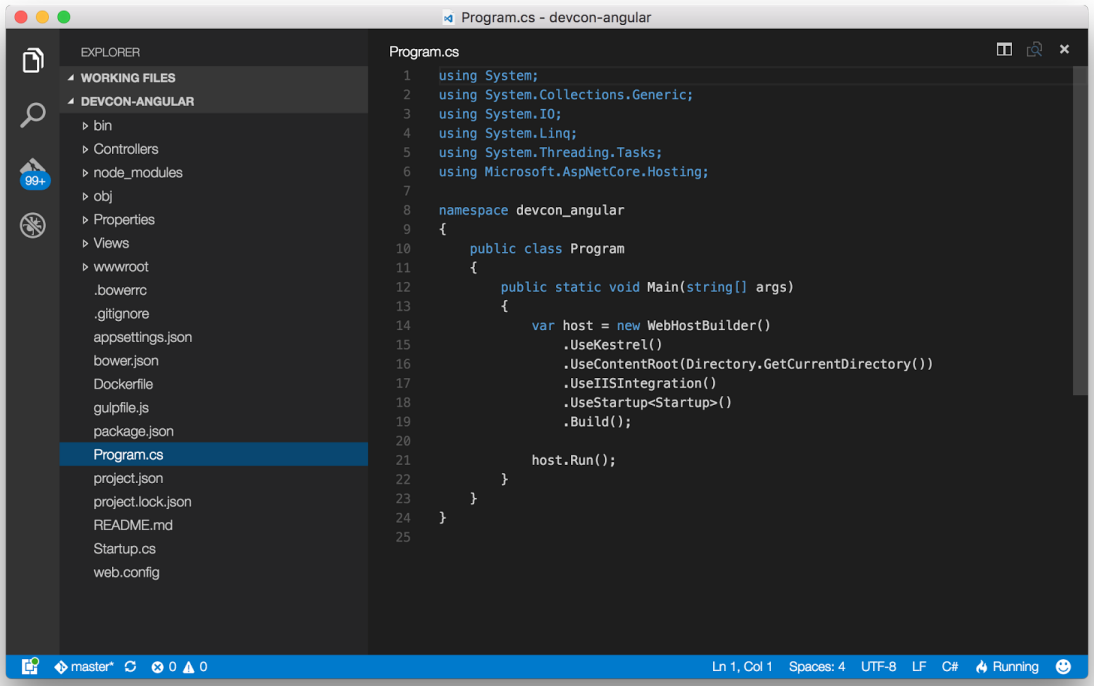
Выполните восстановление зависимостей и запустите проект.

```
dotnet restore
dotnet run
```

Или для запуска и компиляции TypeScript:

```
npm install -g concurrently
tsc && concurrently "tsc -w" "dotnet run"
```

Проверьте работу проекта в браузере и откройте папку с проектом в Visual Studio Code.

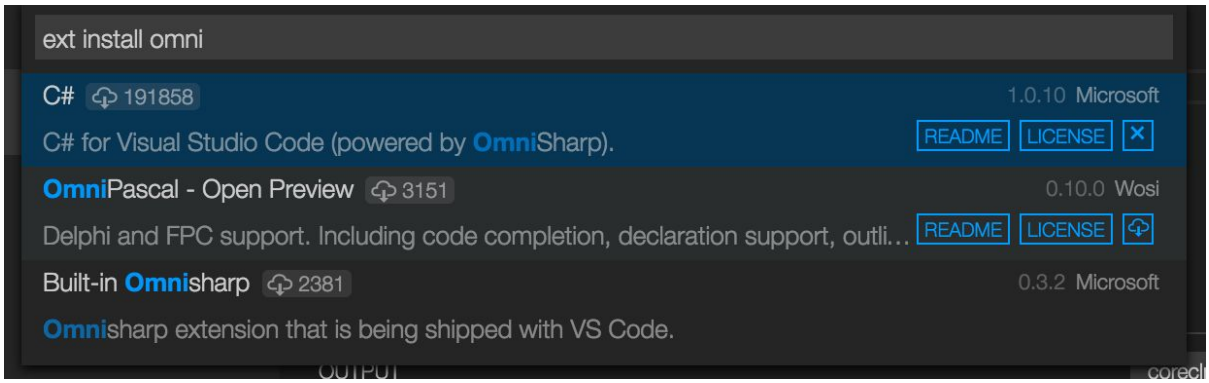


Перейдите в раздел отладки и запустите приложение. Но сначала необходимо проверить, что у вас установлено расширение для полной поддержки .NET и C#.

Откройте панель вввода, нажав сочетание клавиш `Cmd + P` на Mac или `Ctrl + P` на Windows. Введите:

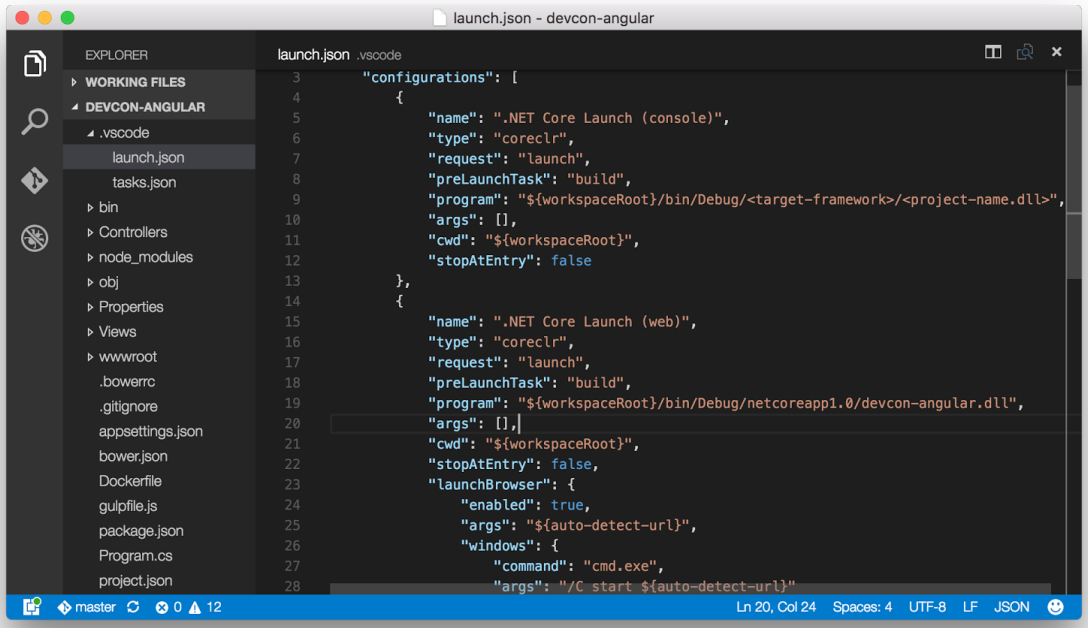
```
ext install omnisharp
```

Проверьте, что у вас установлены расширения C# и Omnisharp.



В созданном файле `launch.json` для `.NET Core Launch (web)` укажите библиотеку для отладки:

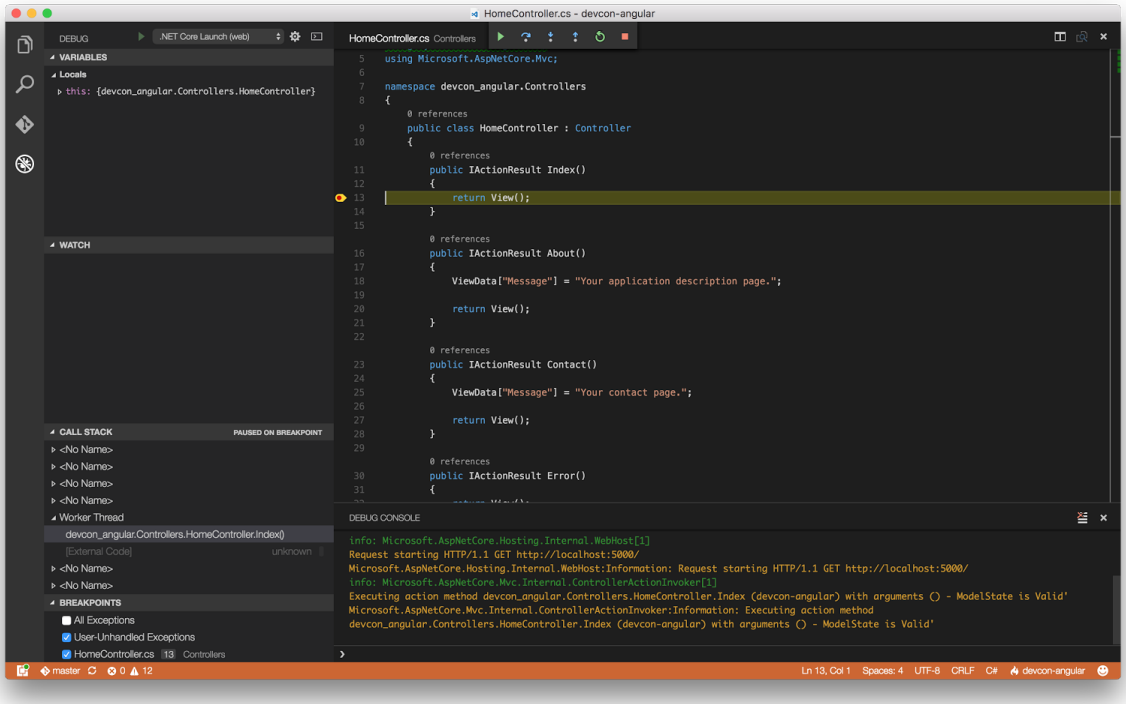
```
"program":
"${workspaceRoot}/bin/Debug/netcoreapp1.0/devcon-angular.dll"
```



В файл project.json добавьте строчку `"postcompile": ["tsc"]`:

```
"scripts": {
  ...
  "postcompile": ["tsc"]
}
```

Попробуйте отладку



Добавление Web API

В папке `Controllers` создайте новый файл `CustomersController.cs`

```
using System.Collections.Generic;
using Microsoft.AspNetCore.Mvc;

namespace DevCon.Controllers
{
    public class Customer{
        public int Id { get; set; }

        public string Name { get; set; }

        public int Age { get; set; }
    }


    [Route("api/[controller]")]
    public class CustomersController : Controller
    {
        [HttpGet]
        public IEnumerable<Customer> GetAll()
        {
            return new Customer[]{
                new Customer {Id = 1, Name = "Ivan Ivanov", Age = 20},
                new Customer {Id = 2, Name = "Petr Petrov", Age = 28},
                new Customer {Id = 3, Name = "Denis Denisov", Age = 14},
                new Customer {Id = 4, Name = "Ivan Ivanov", Age = 20},
```

```
                new Customer {Id = 5, Name = "Sergey Pugachev", Age = 31},
                new Customer {Id = 6, Name = "Stas Pavlov", Age = 17},
                new Customer {Id = 7, Name = "Mik Chernomordikov", Age = 99},
                new Customer {Id = 8, Name = "Ivan Ivanov", Age = 11},
                new Customer {Id = 9, Name = "Petr Ivanov", Age = 18},
                new Customer {Id = 10, Name = "Maks Sidorov", Age = 24},
            };
        }
    }
}
```

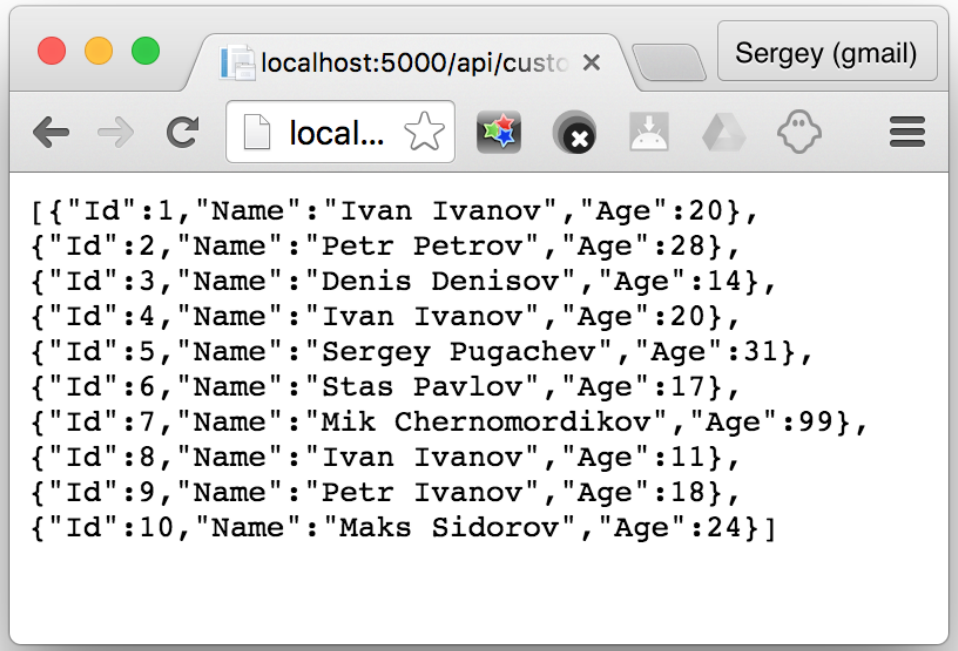
Измените `Startup.cs`, заменив `app.UseStaticFiles()` на:

```
app.UseStaticFiles(new StaticFileOptions()
{
    OnPrepareResponse = (context) =>
    {
        // Disable caching of all static files.
        context.Context.Response.Headers["Cache-Control"] = "no-cache, no-store";
        context.Context.Response.Headers["Pragma"] = "no-cache";
        context.Context.Response.Headers["Expires"] = "-1";
    }
});
```

А также измените `Routing` в файле `Startup.cs`:

```
app.UseMvc(routes =>
{
     routes.MapRoute(
         name: "default",
         template: "{*url}",
         defaults: new { controller = "Home", action = "Index" });
});
```

Перейдите в браузере на http://localhost:5000/api/customers и проверьте, что данные о клиентах корректно выдаются.



Настройка TypeScript и Angular 2

Установка зависимостей

Создайте в папке `wwwroot` файл `package.json` и добавьте в него зависимости зависимости

```
{
  "name": "DevCon",
  "version": "0.0.0",
  "dependencies": {
    "@angular/common":  "2.0.0-rc.1",
    "@angular/compiler":  "2.0.0-rc.1",
    "@angular/core":  "2.0.0-rc.1",
    "@angular/http":  "2.0.0-rc.1",
    "@angular/platform-browser":  "2.0.0-rc.1",
    "@angular/platform-browser-dynamic":  "2.0.0-rc.1",
    "@angular/router":  "2.0.0-rc.1",
    "@angular/router-deprecated":  "2.0.0-rc.1",
    "@angular/upgrade":  "2.0.0-rc.1",
```

```
    "systemjs": "0.19.27",
    "es6-shim": "^0.35.0",
    "reflect-metadata": "^0.1.3",
    "rxjs": "5.0.0-beta.6",
    "zone.js": "^0.6.12",

    "angular2-in-memory-web-api": "0.0.7"
  }
}
```

Перейдите в папку и выполните:

```
npm update
```

Конфигурация TypeScript

Создайте в корне проекта файл `tsconfig.json` со следующим содержимым

```
{
  "compilerOptions": {
    "target": "es5",
    "module": "commonjs",
    "moduleResolution": "node",
    "sourceMap": true,
    "emitDecoratorMetadata": true,
    "experimentalDecorators": true,
    "removeComments": false,
    "noImplicitAny": false
  },
  "exclude": [
    "node_modules",
    "wwwroot/node_modules",
    "wwwroot/typings/main",
    "wwwroot/typings/main.d.ts"
  ]
}
```

Создайте в папке `wwwroot` файл `typings.json` со следующим содержимым:

```
{
  "ambientDependencies": {
    "es6-shim": "registry:dt/es6-shim#0.31.2+20160317120654",
    "jasmine": "registry:dt/jasmine#2.2.0+20160412134438"
  }
}
```

Выполните:

```
typings install
```

Конфигурация System.js

Создайте в папке `wwwroot` файл `systemjs.config.js` со следующим содержимым

```
(function(global) {

  // map tells the System loader where to look for things
  var map = {
    'app':                        'app', // 'dist',
    'rxjs':                       'node_modules/rxjs',
    'angular2-in-memory-web-api': 'node_modules/angular2-in-memory-web-api',
    '@angular':                   'node_modules/@angular'
  };

  // packages tells the System loader how to load when no filename and/or no
extension
  var packages = {
    'app':                        { main: 'main.js',  defaultExtension: 'js' },
    'rxjs':                       { defaultExtension: 'js' },
    'angular2-in-memory-web-api': { defaultExtension: 'js' },
  };

  var packageNames = [
    '@angular/common',
    '@angular/compiler',
    '@angular/core',
    '@angular/http',
    '@angular/platform-browser',
    '@angular/platform-browser-dynamic',
    '@angular/router',
    '@angular/router-deprecated',
    '@angular/testing',
    '@angular/upgrade',
  ];

  // add package entries for angular packages in the form '@angular/common':
  // { main: 'index.js', defaultExtension: 'js' }
  packageNames.forEach(function(pkgName) {
    packages[pkgName] = { main: 'index.js', defaultExtension: 'js' };
  });

  var config = {
    map: map,
    packages: packages
  }

  // filterSystemConfig - index.html's chance
  // to modify config before we register it.
  if (global.filterSystemConfig) { global.filterSystemConfig(config); }
```

```
  System.config(config);

})(this);
```

Создание каркаса Angular приложения

Добавьте в `Views/Shared/_Layout.html` в раздел `HEAD`

```
<base href="/">

<script src="node_modules/es6-shim/es6-shim.min.js"></script>
<script src="node_modules/zone.js/dist/zone.js"></script>
<script src="node_modules/reflect-metadata/Reflect.js"></script>
<script src="node_modules/systemjs/dist/system.src.js"></script>

<script src="systemjs.config.js"></script>
<script>
System.import('app').catch(function(err){ console.error(err);  });
</script>
```

Уберите всё лишнее из `BODY`, оставив только

```
<body>
    @RenderBody()

    <environment names="Development">
        <script src="~/lib/jquery/dist/jquery.js"></script>
        <script src="~/lib/bootstrap/dist/js/bootstrap.js"></script>
        <script src="~/js/site.js" asp-append-version="true"></script>
    </environment>
    <environment names="Staging,Production">
        <script src="https://ajax.aspnetcdn.com/ajax/jquery/jquery-2.2.3.min.js"
                asp-fallback-src="~/lib/jquery/dist/jquery.min.js"
                asp-fallback-test="window.jQuery">
        </script>
        <script src="https://ajax.aspnetcdn.com/ajax/bootstrap/3.3.6/bootstrap.min.js"
                asp-fallback-src="~/lib/bootstrap/dist/js/bootstrap.min.js"
                asp-fallback-test="window.jQuery && window.jQuery.fn &&
window.jQuery.fn.modal">
        </script>
        <script src="~/js/site.min.js" asp-append-version="true"></script>
    </environment>

    @RenderSection("scripts", required: false)
</body>
```

Содержимое `Views/Home/Index` замените на

```
@{
```

```
    ViewData["Title"] = "Home Page";
}

 <my-app>Loading...</my-app>
```

Создание компонента приложения

Создайте в папке `wwwroot` папку `app`. В папке app создайте две подпапки: `components` и `services`.

Создайте файл `main.ts` в папке `app`:

```
import {bootstrap} from '@angular/platform-browser-dynamic';
import {ROUTER_PROVIDERS} from '@angular/router';
import {AppComponent} from
'./components/application/app.component'

bootstrap(AppComponent, [
    ROUTER_PROVIDERS
]);
```

В папке `components` создайте компонент приложения, создав папку `application` и файлы `app.component.ts` и `application.html`:

```
import {Component} from '@angular/core'
import {Routes, ROUTER_DIRECTIVES} from '@angular/router'
import {Http, HTTP_PROVIDERS} from '@angular/http'
import {NavbarComponent} from '../navbar/navbar.component'

@Component({
    selector: 'my-app',
    templateUrl: 'app/components/application/application.html',
    directives: [
        ROUTER_DIRECTIVES,
        NavbarComponent
    ],
    providers: [
        HTTP_PROVIDERS
    ]
})
export class AppComponent{

}
-------------------------------------------------------
<app-navbar></app-navbar>
```

```
<div class="container" role="main">
    <router-outlet></router-outlet>

    <footer>
        <p>&copy; 2016 Company, Inc.</p>
    </footer>

</div>
```

Создание панели навигации

Создайте компонент `navbar` (`navbar.component.ts` и `navbar.html`) в папке `app/components/navbar`:

```
import {Component,Input,Output,EventEmitter} from '@angular/core';
import {ROUTER_DIRECTIVES} from '@angular/router';

@Component({
    selector: 'app-navbar',
    directives: [ROUTER_DIRECTIVES],
    templateUrl: 'app/components/navbar/navbar.html'
})
export class NavbarComponent{

}
```

```
<nav class="navbar navbar-inverse navbar-fixed-top">
 <div class="container">
   <div class="navbar-header">
     <button type="button" class="navbar-toggle collapsed"
      data-toggle="collapse" data-target="#navbar"
      aria-expanded="false" aria-controls="navbar">
       <span class="sr-only">Toggle navigation</span>
       <span class="icon-bar"></span>
       <span class="icon-bar"></span>
       <span class="icon-bar"></span>
     </button>
     <a class="navbar-brand" href="#">DevCon</a>
   </div>
   <div id="navbar" class="navbar-collapse collapse">
     <ul class="nav navbar-nav">
```

```
      <li><a [routerLink]="['/']">Home</a></li>
      <li><a [routerLink]="['/customers']">Customers</a></li>
    </ul>
  </div>
 </div>
</nav>
```

Добавление навигации

Создайте компоненты `home`, `customers` и `customer`.
Компонент `Home`:

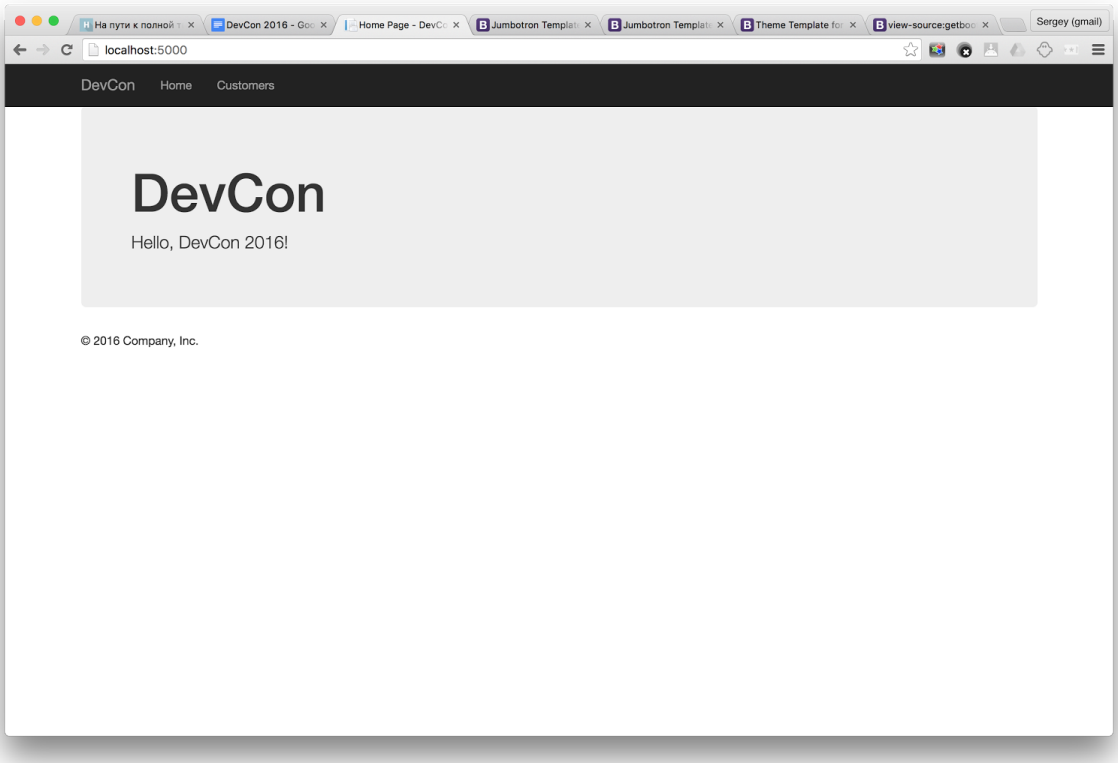
```
import {Component} from '@angular/core'

@Component({
    selector: 'app-home-page',
    templateUrl: 'app/components/home/home.html'
})
export class HomeComponent{

}
```

```
<div class="jumbotron">
    <h1>DevCon</h1>
    <p>Hello, DevCon 2016!</p>
</div>
```

В компонент `AppComponent` добавьте декоратор `Routes`:

```
import {HomeComponent} from '../home/home.component'
```

```
@Routes([
    {path: '/', component: HomeComponent}
])
export class AppComponent{
```



Создание сервиса доступа к данным

В папке `services` создайте подпапку `customers`, а в ней `customers.service.ts`:

```
import {Injectable} from '@angular/core';
import {Http, Response} from '@angular/http'
import {Observable} from 'rxjs/Rx';
import 'rxjs/add/operator/map';

export class Customer{
    Id: number;
    Name: string;
    Age: number;
}

@Injectable()
export class CustomersService{
    constructor(private _http: Http){

    }

    getCustomers(){
        return this._http.get('/api/customers').map(
       (resp: Response)=>
```

```
        <Customer[]>resp.json())
        .do(data => console.log(data))
        .catch(this.handleError);
    }

    private handleError(error: Response){
        console.error(error);

        return Observable.throw(
        error.json().error || 'server error');
    }
}
```

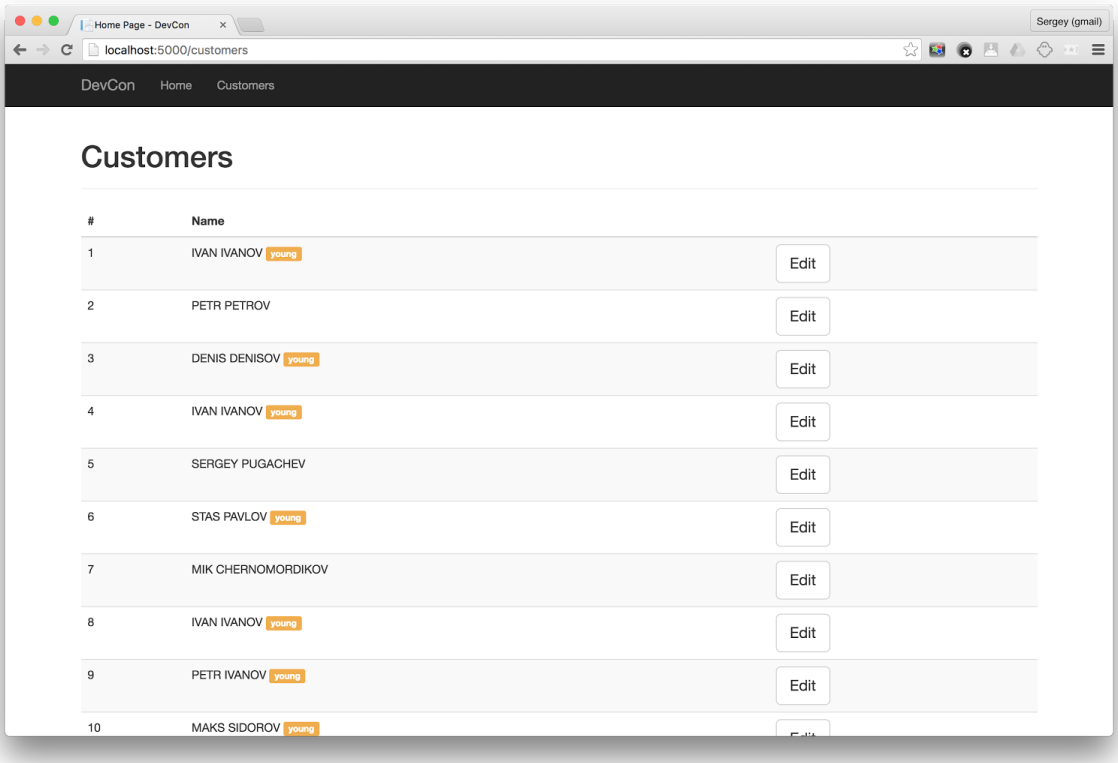
Подключите сервис к компоненту приложения:

```
import {CustomersService} from
'../../services/customers/customers.service'

----------------------------------------------------

providers: [
     HTTP_PROVIDERS,
     CustomersService
]
```

Страница Customers

```
import {Component, OnInit} from '@angular/core';
import {ROUTER_DIRECTIVES} from '@angular/router'
import {CustomersService, Customer} from
'../../services/customers/customers.service'
import {Observable} from 'rxjs/Rx'

@Component({
    selector: 'app-customers-page',
    directives: [ROUTER_DIRECTIVES],
    templateUrl: 'app/components/customers/customers.html'
})
export class CustomersComponent implements OnInit{
    customers: Observable<Customer>;

    constructor(private _customerService: CustomersService){

    }
```

```
    ngOnInit(){
        this.customers = this._customerService.getCustomers();
    }
}

-------------------------------------------------------

<div class="page-header">
    <h1>Customers</h1>
</div>

<table class="table table-striped">
  <thead>
    <tr>
      <th>#</th>
      <th>Name</th>
      <th></th>
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let customer of customers | async">
      <td>{{customer.Id}}</td>
      <td>{{customer.Name | uppercase}}
        <span class="label label-warning"
        *ngIf="customer.Age <= 25">young</span></td>
        <td><a class="btn btn-lg btn-default"
        [routerLink]="['/customer', customer.Id]">Edit</a>
      </td>
    </tr>
  </tbody>
</table>
```

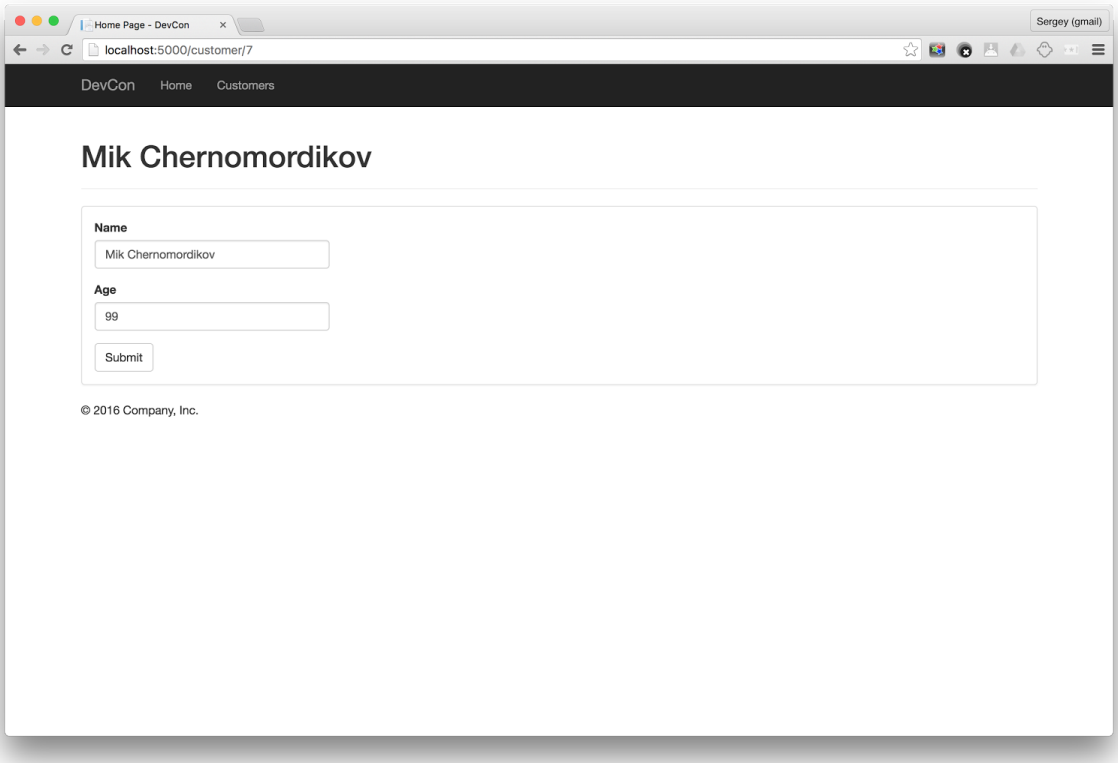


Страница Customer

```
import {Component} from '@angular/core'
import {Router, RouteSegment, OnActivate} from '@angular/router'
import {CustomersService, Customer} from
'../../services/customers/customers.service'

@Component({
    selector: 'app-customer-page',
    templateUrl: 'app/components/customer/customer.html'
})
export class CustomerComponent implements OnActivate{
    customer: Customer = {Id: 0, Name: '', Age: 0};

    constructor(
        private _router: Router,
        private _customerService: CustomersService
        ){}

    routerOnActivate(segment: RouteSegment):void{
        let id = +segment.getParam('id');

        this._customerService.getCustomers().subscribe(
       (customers:Customer[]) =>{
```

```
            let filteredCustomers = customers.filter(
            c => c.Id == id);

            if(filteredCustomers.length > 0){
                this.customer = filteredCustomers[0];
            }
        });
    }
}
```

```
<div class="page-header">
  <h1>{{customer.Name}}</h1>
</div>


<div class="panel panel-default">
  <div class="panel-body">
    <form>
        <div class="form-group">
            <label for="customerName">Name</label>
            <input type="text" class="form-control"
            id="customerName" required
            [(ngModel)]="customer.Name">
        </div>
        <div class="form-group">
            <label for="customerAge">Age</label>
            <input type="number" class="form-control"
            id="customerAge" required
            [(ngModel)]="customer.Age">
        </div>
        <button type="submit"
         class="btn btn-default">Submit</button>
    </form>
  </div>
</div>
```



Настроим Routing в компоненте приложения:

```
import {CustomersComponent} from
'../customers/customers.component'
import {CustomerComponent} from '../customer/customer.component'

------------------------------------------------------

@Routes([
    {path: '/', component: HomeComponent},
    {path: '/customers', component: CustomersComponent},
    {path: '/customer/:id', component: CustomerComponent}
])
```

Валидация формы[2]

Добавим компоненту Customer поддержку валидации:

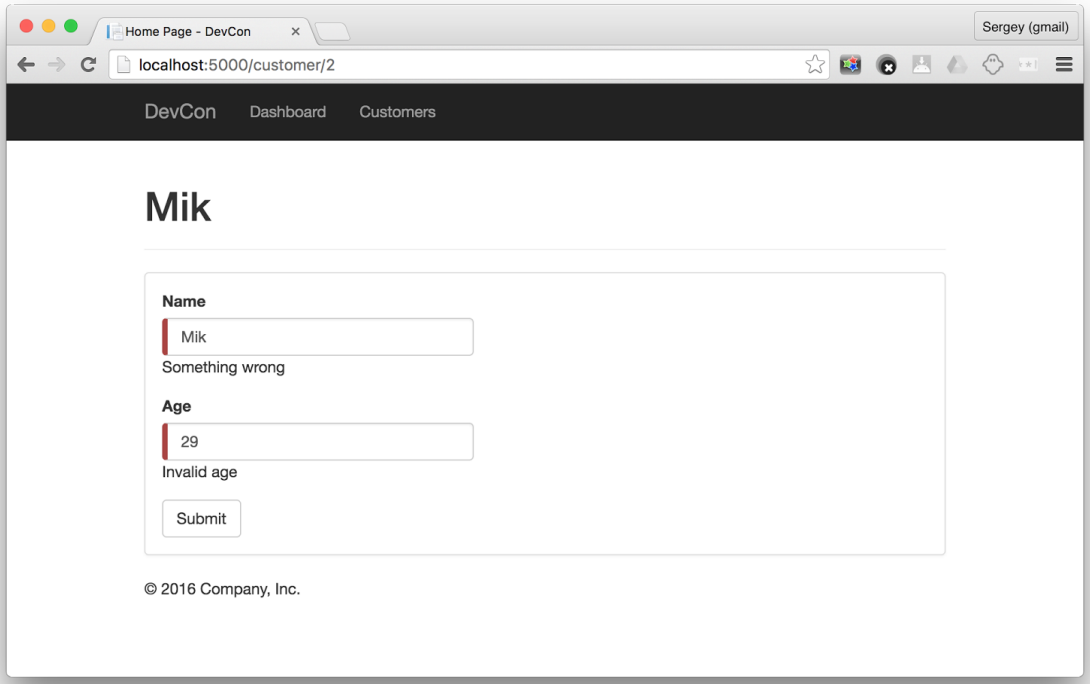
```
import {Component} from '@angular/core'
import {FORM_DIRECTIVES, FormBuilder, NgForm, Control, ControlGroup,
Validators} from '@angular/common'
```

[2] http://blog.ng-book.com/the-ultimate-guide-to-forms-in-angular-2/

```
import {Router, RouteSegment, OnActivate} from '@angular/router'
import {CustomersService, Customer} from
'../../services/customers/customers.service'

@Component({
    selector: 'app-customer-page',
    templateUrl: 'app/components/customer/customer.html',
    directives: [FORM_DIRECTIVES]
})
export class CustomerComponent implements OnActivate{
    customer: Customer = {Id: 0, Name: '', Age: 0};
    customerForm: ControlGroup;

    constructor(
        private _router: Router,
        private _customerService: CustomersService,
        private _formBuilder: FormBuilder
    ){

       this.customerForm = _formBuilder.group({
        name: ['', Validators.compose(
       [Validators.required, Validators.minLength(4)])],
        age: ['', Validators.compose(
        [Validators.required, this.ageValidator])]
       });
    }

    onSubmit(form: any): void {
        console.log(this.customerForm.valid);
        console.log('Submitted value:', form);
    }

    ageValidator(control: Control): { [s: string]: boolean } {
        let val = +control.value;
        if(val % 2 != 0) return {invalidAge: true};
    }

    routerOnActivate(segment: RouteSegment): void{
        let id = +segment.getParam('id');

        this._customerService.getCustomers().subscribe(
            (customers:Customer[]) =>{
            let filteredCustomers = customers.filter(
             c => c.Id == id);
            if(filteredCustomers.length > 0){
                this.customer = filteredCustomers[0];
```

```
            }
        });
    }
}

-----------------------------------------------------

<style type="text/css">
.ng-valid[required] {
  border-left: 5px solid #42A948; /* green */
}

.ng-invalid {
  border-left: 5px solid #a94442; /* red */
}
</style>

<div class="page-header">
  <h1>{{customer.Name}}</h1>
</div>

<!--
<form #f="ngForm" (ngSubmit)="onSubmit(f.value)">
-->
<div class="panel panel-default">
  <div class="panel-body">
    <form [ngFormModel]="customerForm"
          (ngSubmit)="onSubmit(customerForm.value)">
        <div class="form-group">
            <label for="customerName">Name</label>
            <input type="text" class="form-control"
            id="customerName"
            required [(ngModel)]="customer.Name"
            [ngFormControl]="customerForm.controls.name">
            <div *ngIf="!customerForm.controls.name.valid">
              Something wrong
            </div>
        </div>
        <div class="form-group">
            <label for="customerAge">Age</label>
            <input type="number" class="form-control"
            id="customerAge"
            required [(ngModel)]="customer.Age"
            [ngFormControl]="customerForm.controls.age">
            <div
*ngIf="customerForm.controls.age.hasError('invalidAge')">
```

```
                Invalid age
              </div>
          </div>
          <button type="submit"
          class="btn btn-default">Submit</button>
      </form>
    </div>
</div>
```



Свойства и события

Добавьте в компонент Navbar свойство `HomeTitle` и событие `LinkMouseOver`:

```
export class NavbarComponent{
    @Input() HomeTitle: string;
    @Output() LinkMouseOver: EventEmitter<any> = new
    EventEmitter<any>();

    fireEvent(evt){
        this.LinkMouseOver.emit(evt);
    }
}
```

```
<ul class="nav navbar-nav">
<li><a [routerLink]="['/']"
(mouseover)="fireEvent($event)">{{HomeTitle}}</a></li>
<li><a [routerLink]="['/customers']"
(mouseover)="fireEvent($event)">Customers</a></li>
</ul>
```

Задайте знаыение свойства и подпишитесь на событие в application.html:

```
<app-navbar HomeTitle="Dashboard"
(LinkMouseOver)="onEvent($event)"></app-navbar>

-----------------------------------------------------
export class AppComponent{
    onEvent(evt){
        console.log('onEvent');
        console.dir(evt);
    }
}
```

Заключение

Поздравляем, вы создали полноценное Angular 2 приложение с поддержкой навигации, сервисов доступа к данным, и формаой с валидацией.