Integrating Angular with ASP.NET Core RESTful Services

Dan Wahlin
Dan Wahlin

https://blog.codewithdan.com

@DanWahlin
Agenda

• The Big Picture
• Creating a Web API Project
• Angular Services
• Injecting Services into Components
The Big Picture

HTML (Razor) -> ASP.NET Core

JSON -> Web API

Data Layer

Database
ASP.NET Core and Web API

- Cross Platform
- Middleware
- Controller
- Routing
- Dependency Injection
- Fast
ASP.NET Core Controllers

![Diagram showing the relationship between ASP.NET Core and controllers, with ActionResult for HTML and JSON]

- **ActionResult (HTML)**
- **ActionResult (JSON)**
Creating a Web API Project
Creating a Web API Project in Visual Studio
Creating a Web API Project on Mac

1. npm install -g yo generator-aspnet
2. yo aspnet
Creating a Controller

ASP.NET MVC and Web API controller classes both derive from `Controller`:

```csharp
[Route("api/[controller]")] public class CustomersController : Controller {

}
```
Defining Injectables

Configure dependency injection in Startup.cs

```csharp
public void ConfigureServices(IServiceCollection services)
{
    //Configure an injectable object
    services.AddScoped<ICustomersRepository, CustomersRepository>();
    ...
}
```

Instance created per request
Using Dependency Injection

Objects configured in Startup.cs ConfigureServices() can be injected:

```csharp
[Route("api/[controller]")]
public class CustomersServiceController : Controller
{
    ICustomersRepository _repo;

    public CustomersServiceController(ICustomersRepository repo) {
        _repo = repo;
    }
}

Instance injected at runtime
Adding an Action and Route

Web API actions can return a custom type or ActionResult

```csharp
[Route("api/[controller]")]
public class CustomersController : Controller
{
    ICustomersRepository _repo;

    [HttpGet("{id}\")]
    public async Task<ActionResult> Get(int id)
    {
        var customer = await _repo.GetCustomerAsync(id);
        if (customer == null) {
            return NotFound("No customer found!");
        }
        return Ok(customer);
    }
}
```

Async action
Angular Services
Angular - The Big Picture

App Module (@NgModule)

- Modules
- Routes
- Component
  - Template
  - Directives/Components
  - Component Code
  - Service
How Do You Integrate Angular Into an ASP.NET Core Project?

• Use an Angular/ASP.NET Core Seed Project

• Use the dotnet CLI:

```
dotnet new --install Microsoft.AspNetCore.SpaTemplates::*
```

• Use the Angular CLI

```
ng new [project_name] -sd wwwroot -dir .
```
Angular Services

Services are reusable classes that handle business rules, calculations, Ajax calls, etc.

```typescript
import { Injectable } from '@angular/core';
import { Http } from '@angular/http';

@Injectable()
export class DataService {
  constructor(private http: Http) { }
}
```
Angular Async Operations

• Services that perform asynchronous operations can use Promises or Observables

• Promise:
  • An operation that hasn't completed yet, but is expected in the future
  • Used with async/deferred operations
  • Can be hooked to a callback

• Observable
  • An object that can be “subscribed” to by other objects
  • Can return multiple values over time – an async data stream
  • Event based
# Observables versus Promises

<table>
<thead>
<tr>
<th>Promise</th>
<th>Observable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns a single value</td>
<td>Can return multiple values over time (think of an array into the future)</td>
</tr>
<tr>
<td>Cannot cancel</td>
<td>Can cancel</td>
</tr>
<tr>
<td></td>
<td>Supports standard array functions (map, filter, reduce, etc.)</td>
</tr>
</tbody>
</table>
Promises in Action

Component

- Function
- Promise
- Success callback

Service

GET function makes Ajax call

Diagram showing the flow of a component making an Ajax call and returning a promise to the component.
Observables in Action

1. Component
   - subscribe

2. subscribe callback

3. Service
   - 1 or more items returned

4. Server
Using Http to Call RESTful Services

RESTful services can be called using Http

data.service.ts

```typescript
import { Http } from '@angular/http';
import { Observable } from 'rxjs/Rx';
import 'rxjs/add/operator/map';
import 'rxjs/add/operator/catch';

@Injectable()
export class DataService {
  constructor(private http: Http) { }

  getCustomers() : Observable<ICustomer[]> {
    return this.http.get('api/customers')
      .map((response: Response) => response.json())
      .catch(this.handleError);
  }
}
```

Map response to customers
Http and Promises

Create a promise by calling toPromise()

```typescript
import { Http } from '@angular/http';
import 'rxjs/add/operator/toPromise';
import 'rxjs/add/operator/catch';

@Injectable()
export class DataService {
  constructor(private http: Http) { }

  getCustomers(): Promise<ICustomer[]> {
    return this.http.get('api/customers')
      .toPromise()
      .then((response: Response) => response.json())
      .catch(this.handleError);
  }
}
```
Injecting Services into Components
Angular - The Big Picture

App Module (@NgModule)

- Modules
- Routes
- Component
  - Template
  - Directives/Components
  - Component Code
  - Service
Angular - The Big Picture

App Module (@NgModule)

Component

Template

Component Code

Directives/Components

Service
Defining a Service Provider

- Services can be injected when a **provider** has been defined in a component or in an @NgModule

```typescript
import { DataService } from './shared/data.service';

@NgModule({
  imports: [ BrowserModule, HttpModule ],
  declarations: [ AppComponent, CustomersComponent ],
  providers: [ DataService ],
  bootstrap: [ AppComponent ]
})
export class AppModule { }
```
Injecting a Service into a Component

Services are "provided" to components

```typescript
customers.component.ts
import { DataService } from '../shared/data.service';
@Component({
...
})
export class CustomersComponent implements OnInit {
  customers: Customer[];
  constructor(private dataService: DataService) { }
  ngOnInit() {
    this.dataService.getCustomers()
      .subscribe((customers: Customer[]) => {
        this.customers = customers;
      });
  }
}
Angular and ASP.NET Core

https://github.com/DanWahlin/Angular-ASPNET-Core-Seed
https://github.com/DanWahlin/AspNetCorePostgreSQLDockerApp
https://github.com/DanWahlin/Angular-ASPNET-Core-CustomersService
https://github.com/DanWahlin/ASPNETCore-Sync-Async-EF

Angular Projects:

http://codewithdan.me/angular-10-projects
Thanks for Coming!

Dan Wahlin
@DanWahlin
Wahlin Consulting
Get the Content:

http://codewithdan.me/angular-aspnet-core
Angular, Node, TypeScript, JavaScript, C#, ASP.NET Core, Docker & more at your company or online!

https://codewithdan.com