Android, iOS and Hybrid Applications

Mobile-Development

Overview

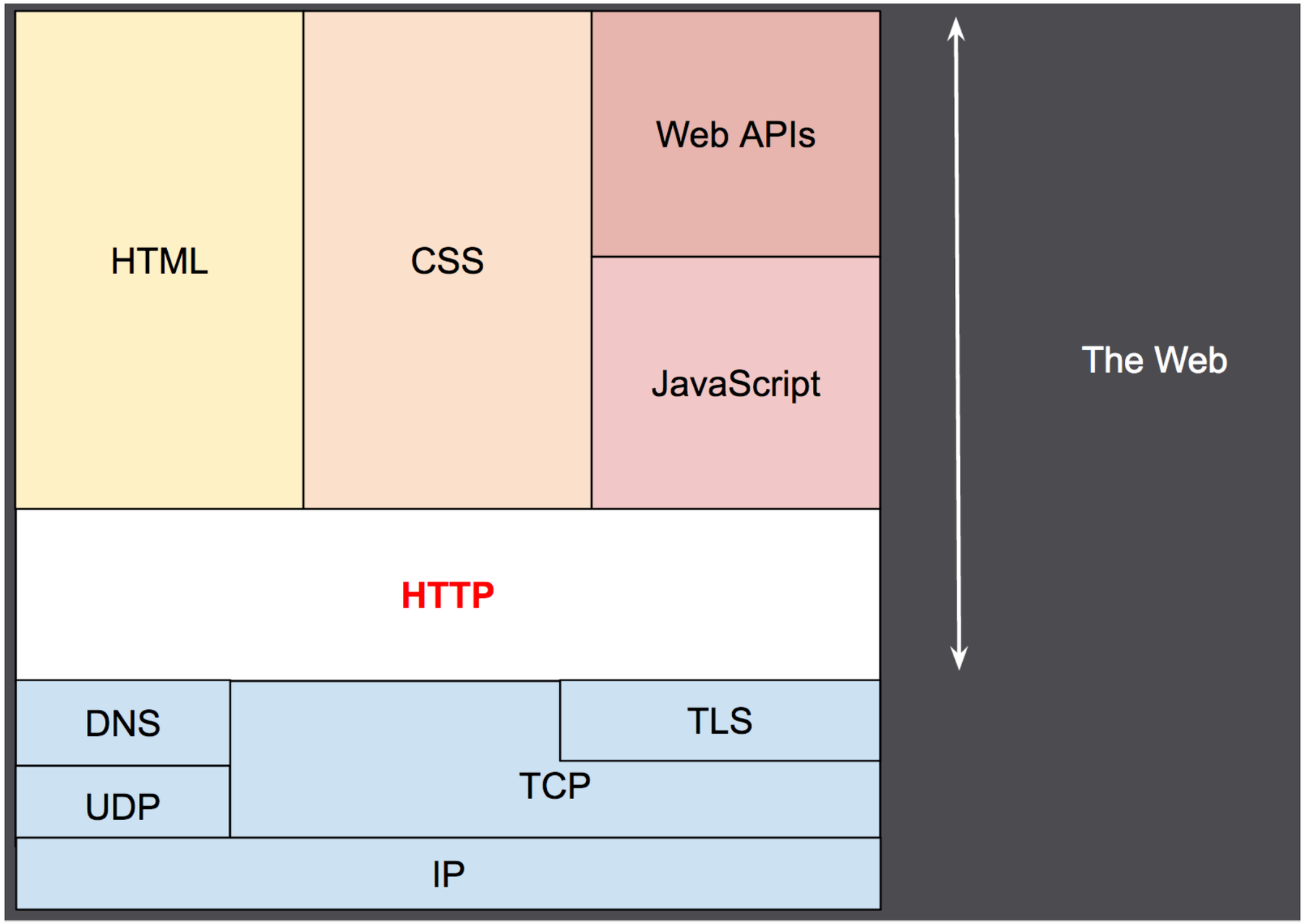
- Web-API with .NET Core
- Continue with your app
  - Biometrics
  - Notifications
  - Accessing the WebAPI

REST

- It's a "standard"
  - Stateless
  - Based on the HTTP-Verbs
    - GET, POST, PUT, DELETE
  - Keep it simple
  - Use the headers to transport information

https://de.wikipedia.org/wiki/Representational_State_Transfer

HTTP Overview



HTTP Protocol

- The top layer under JS/HTML/CSS
- Client - Server
- Request - Response
- HTTP Headers as a key concept for extensibility

https://www.w3.org/Protocols/
https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

HTTP Headers

- Can be unidirectional
  - Request & Response
- Request or Response only
- Key & Value
  - Content-Type : application/json

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers

HTTP Request Headers

- Authorization
  - Send authentication information
- Cache-Control
  - Control caching of a request
- Accept
  - Tell the server what sort of result we expect

HTTP Response Headers

- Content-Type
  - What type of content is sent
- Cache-Control
  - Manage caching of resources

EXAMPLE REQUEST

GET / HTTP/1.1

Host: developer.mozilla.org

Accept-Language: fr

HTTP/1.1 200 OK

Date: Sat, 09 Oct 2010 14:28:02 GMT

Server: Apache

Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT

Accept-Ranges: bytes

Content-Length: 29769

Content-Type: text/html

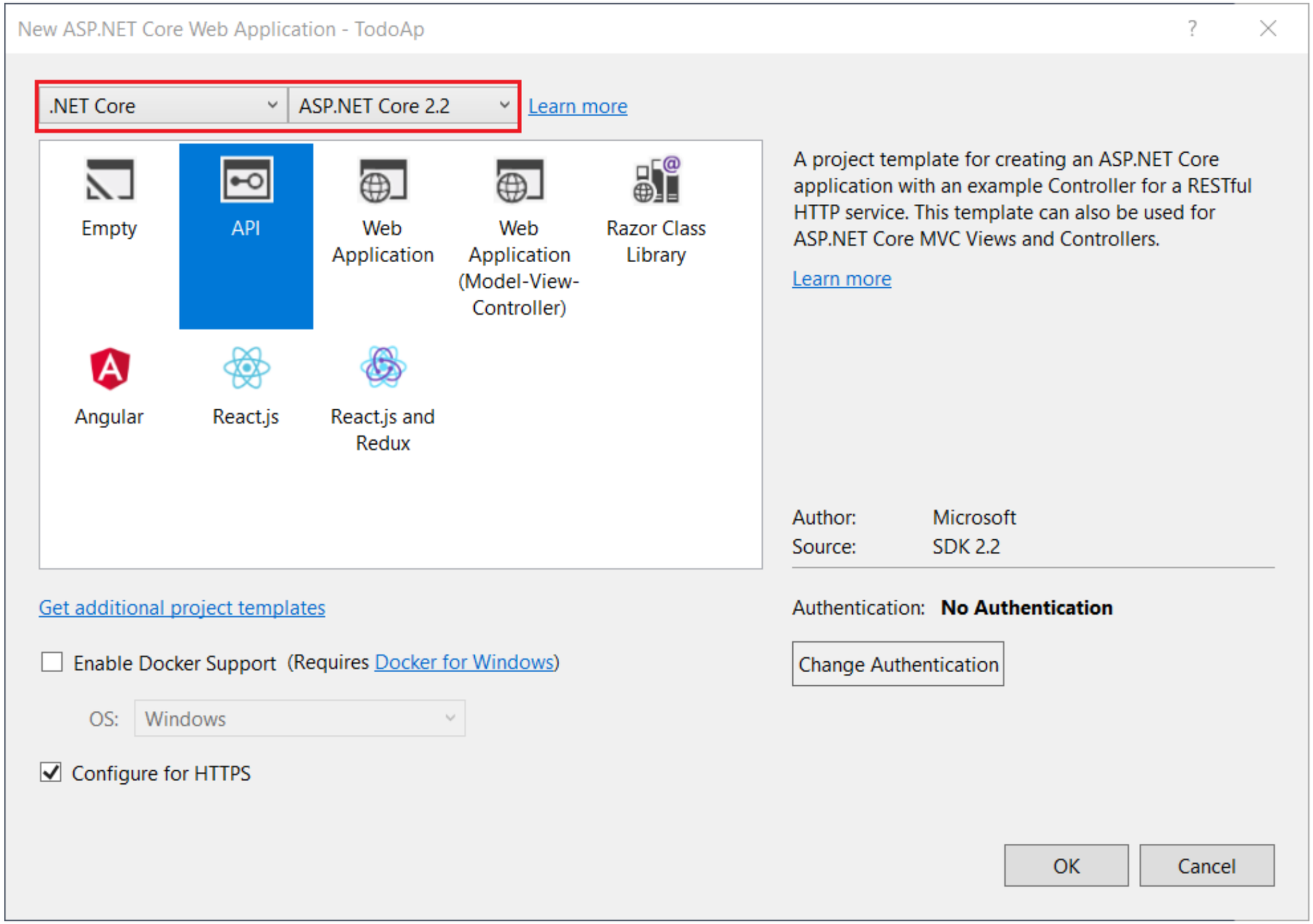
<!DOCTYPE html... (here comes the 29769 bytes of the requested web page)

Web-API

- Use it with .NET Core 2
- Create a new Project with the Web-API template
  - No Authentication
  - “API” template

https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-web-api?view=aspnetcore-2.2&tabs=visual-studio

WEB API


New ASP.NET Core Web Application - TodoAp
.NET Core
ASP.NET Core 2.2
Learn more
Empty
API
Web Application
Web Application (Model-View-Controller)
Razor Class Library
Angular
React.js
React.js and Redux
A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.
Learn more
Author: Microsoft
Source: SDK 2.2
Get additional project templates
Authentication: No Authentication
Enable Docker Support (Requires Docker for Windows)
Change Authentication
OS: Windows
Configure for HTTPS
OK
Cancel

Web-API

- Controllers
  - contain methods that are mapped to “routes”
  - control the behaviour via attributes
  - after project creation there’s already a “ValueController” around

Web-API

- Route attribute defines how the url looks
  - http://localhost:5001/api/values

```
[Route("api/[controller]")]
[ApiController]
public class ValuesController : ControllerBase
```

Web-API

- Use the attributes to control the method that is allowed

```
[HttpGet]
public ActionResult<IEnumerable<string>> Get()
{
    return new string[] {"value1", "value2"};
}

[HttpPost]
public void Post([FromBody] string value)
{
}

[HttpDelete("{id}")]
public void Delete(int id)
{
}
```

Web-API

- Use the Route attribute on methods
  - https://localhost:5002/api/values/shorter

```
[Route("shorter")]
[HttpGet]
public ActionResult<string> SomeOtherMethod()
{
    return "Return something...";
}
```

Web-API

- Return complex types - default is serialisation to JSON

```
[Route("shorter")]
[HttpGet]
public ActionResult<TodoModel> SomeOtherMethod()
{
    return new TodoModel()
    {
        Name = "Some Todo Name”,
        DueTime = DateTime.Now
    };
}

// https://localhost:5002/api/values/shorter

{
  "name": "Some Todo Name",
  "dueTime": "2019-06-15T20:38:17.007596+02:00"
}
```

Web-API

- Authentication and Authorisation
  - Done via the “Authorize” attribute

```
[Route("shorter")]
[HttpGet]
[Authorize]
public ActionResult<TodoModel> SomeOtherMethod()
{
    return new TodoModel()
    {
        Name = "Some Todo Name",
        DueTime = DateTime.Now
    };
}
```

Web-API

- To effectively implement authentication
  - you need an Identity Provider / Server
  - requests have to pass in that information
- We’re going to implement an example in the project week

HTTP-Client

- Implementation on C# can be used in Xamarin
- Reuse the same instance (IoC)

```
var client = new HttpClient();
var result = await client.GetAsync("http://localhost:5003/api/values");
```

JSON

- Use Newtonsoft to deserialise values
- Add the package to your project

```
JsonConvert.DeserializeObject<TodoItem>(result);
```

Web-API

- Setup you own project
- Download https://www.getpostman.com/ to create requests
- Run it with the app side by side and try to fetch some values