

# Using Data from an API

---



**Daniel Villamizar**

Senior Cloud Solutions Architect - MVP

@danielvillamizara – <https://www.linkedin.com/in/danielvillamizara/>

# Module overview



**Working with an API**

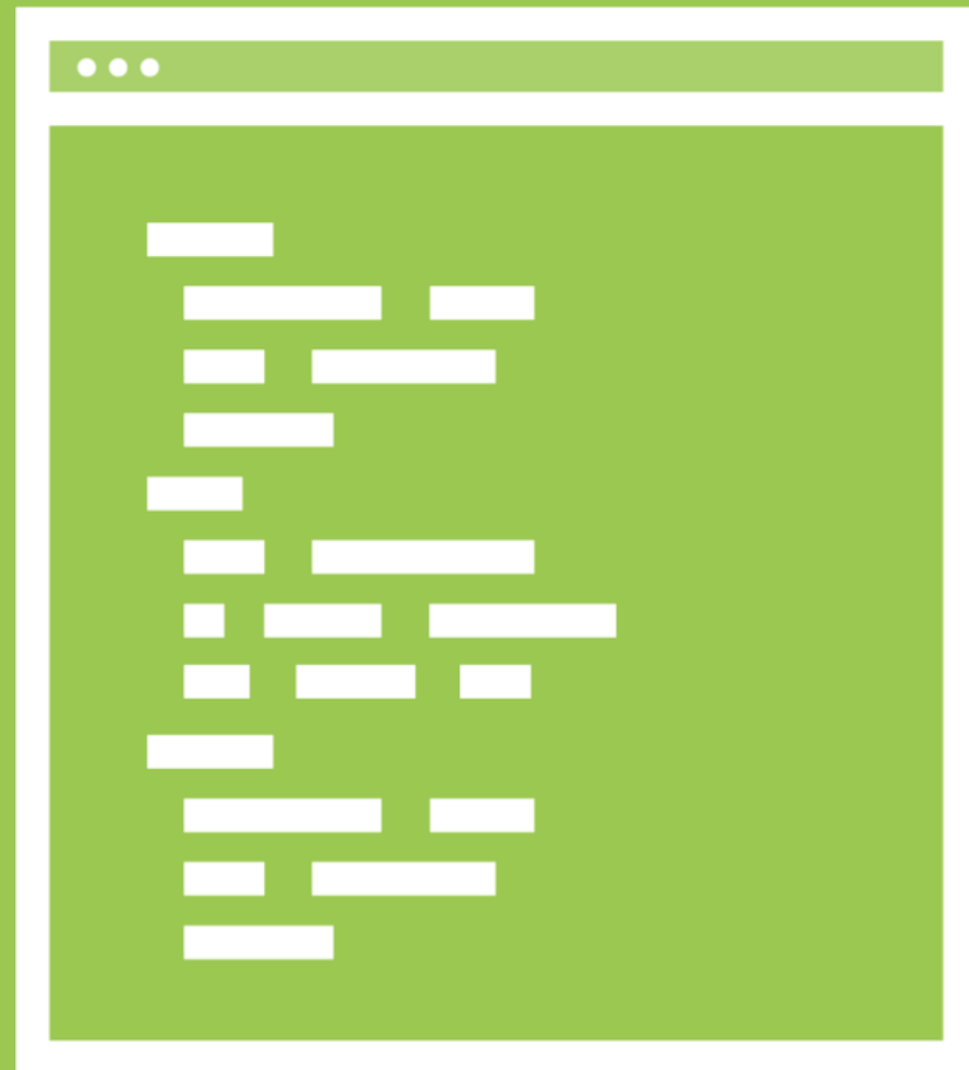
**Accessing API data using HttpClient**

**Managing the application state**

**Storing data locally**

# Working with an API

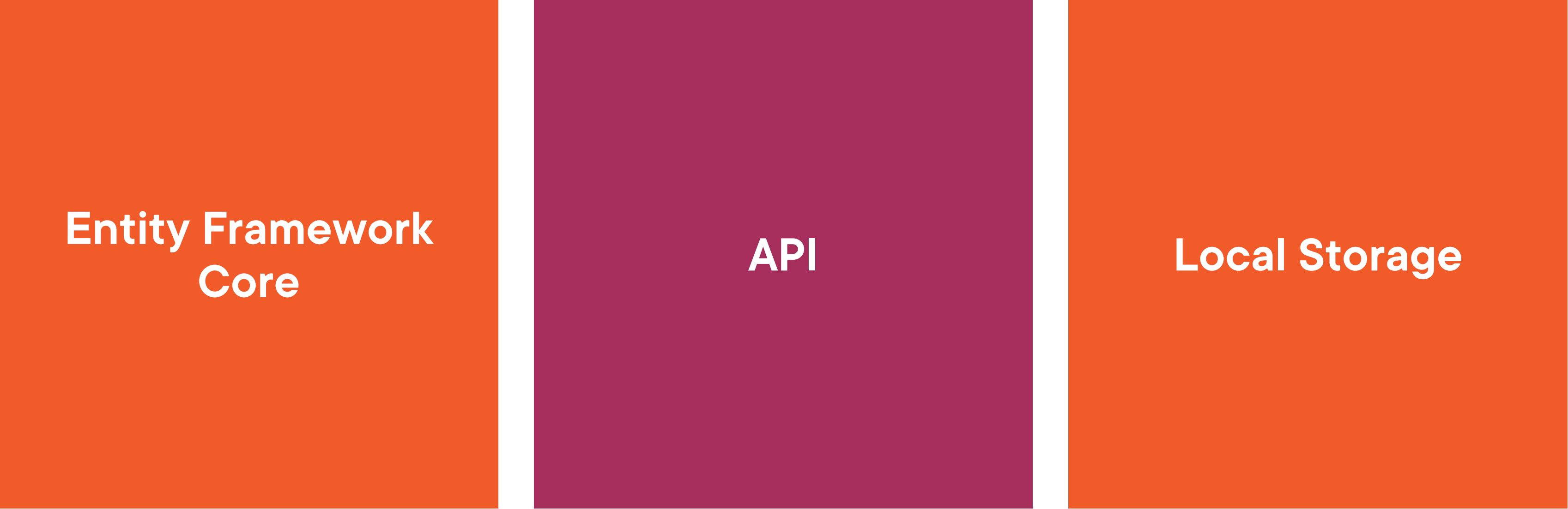
---



# Hard coded data

Almost every app will use “real” data

# Working with Data



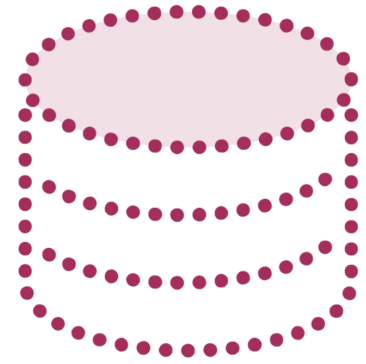
The diagram consists of three rectangular boxes arranged horizontally. The leftmost box is orange and contains the text 'Entity Framework Core'. The middle box is a darker, maroon-like orange and contains the text 'API'. The rightmost box is orange and contains the text 'Local Storage'. All text is in a bold, white, sans-serif font.

**Entity Framework  
Core**

**API**

**Local Storage**

# Creating an API



**Uses “just” the data**

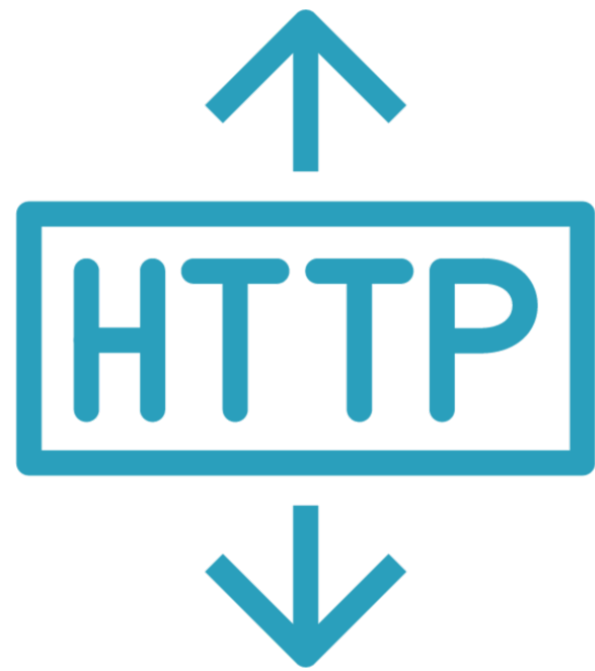
**{JSON}**

**JSON or XML**



**Open for many types of clients**

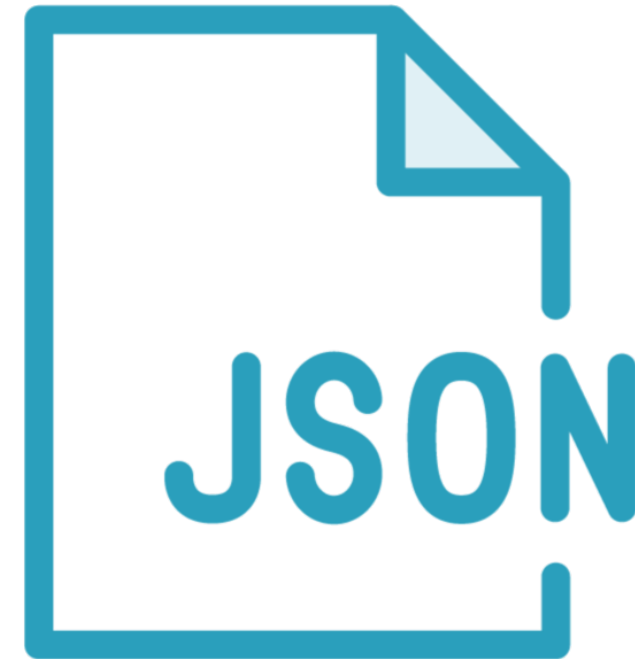
# Creating a RESTful API



HTTP request  
GET, POST,  
PUT...



Resources with  
URLs

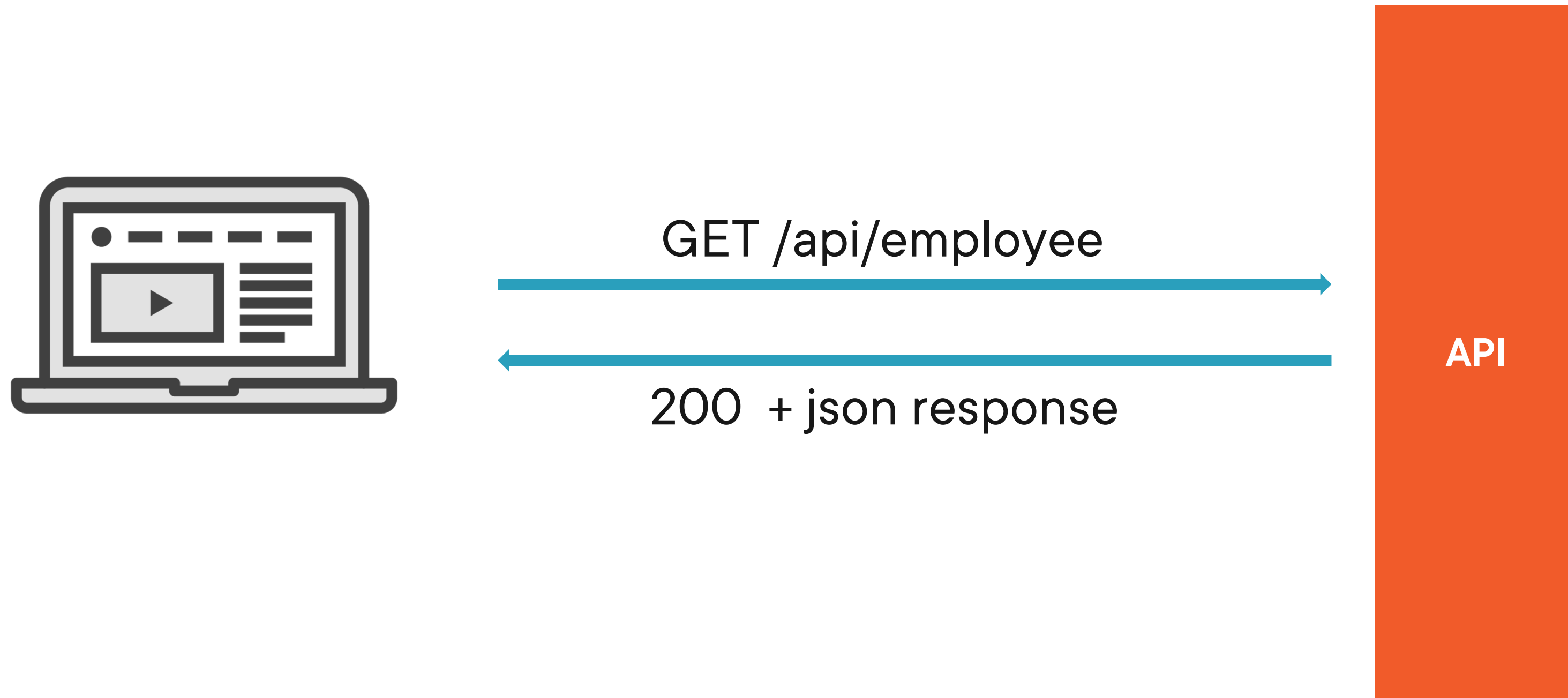


Responses in  
JSON



Status codes  
200, 404...

# Accessing a REST API





# HTTP Verbs

**GET**

**POST**

**PUT**

**DELETE**

# JSON Response

```
1  {
2
3      "employeeId": 1,
4      "firstName": "Bethany",
5      "lastName": "Smith",
6      "birthDate": "1979-01-16T00:00:00",
7      "email": "bethany@bethanyspieshop.com",
8      "street": "Grote Markt 1",
9      "zip": "1000",
10     "city": "Brussels",
11     "countryId": 1,
12     "country": null,
13     "phoneNumber": "324777888773",
14     "smoker": false,
15     "maritalStatus": 1,
16     "gender": 1,
17     "comment": "Lorem Ipsum",
18     "joinedDate": "2015-03-01T00:00:00",
19     "exitDate": null,
20     "jobCategoryId": 1,
21     "jobCategory": null,
22     "latitude": 50.8503,
23     "longitude": 4.3517,
24     "imageContent": null,
25     "imageName": null
26 },
27 {
28     "employeeId": 2,
29     "firstName": "aa",
30     "lastName": "aa",
31     "birthDate": "2022-07-24T10:29:07.143",
32     "email": "aa",
33     "street": "aa",
34     "zip": "a",
35     "city": "aa",
36     "countryId": 1,
37     "country": null,
38     "phoneNumber": "",
39     "smoker": true,
40     "maritalStatus": 0,
41     "gender": 0,
42     "comment": null,
43     "joinedDate": "2022-07-24T10:29:07.161",
44     "exitDate": null,
45     "jobCategoryId": 3,
46     "jobCategory": null,
47     "latitude": 2,
48     "longitude": 1,
49     "imageContent": null,
50     "imageName": null
51 },
```

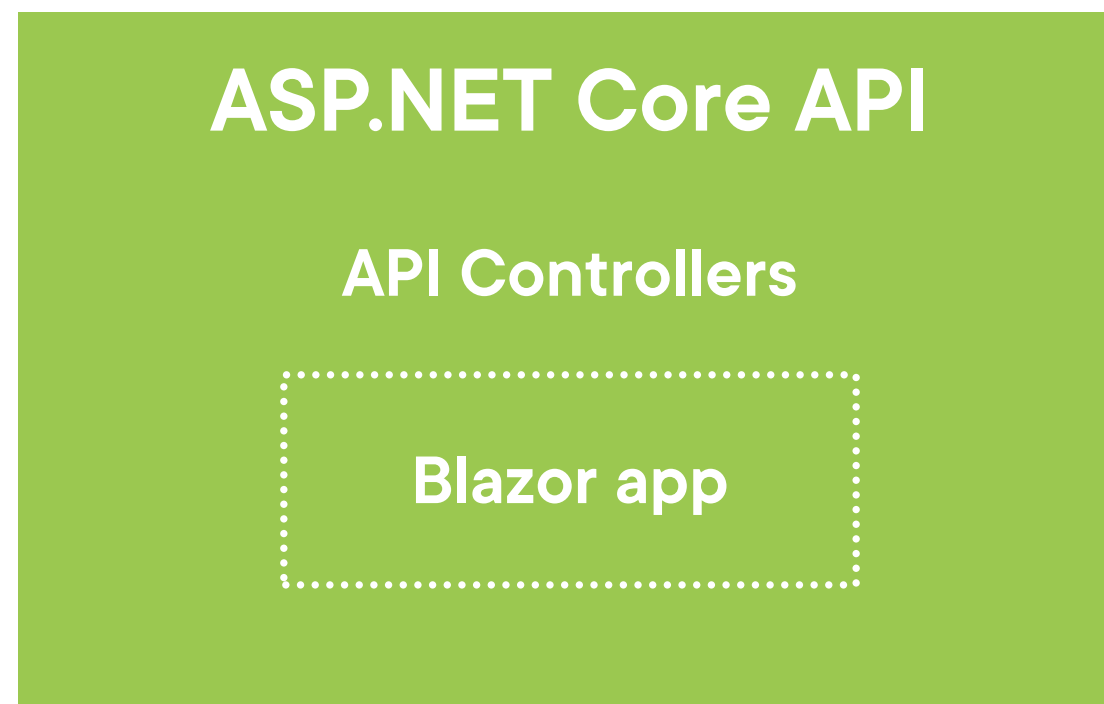
# Demo



## Exploring the API

- ASP.NET Core REST API

# Sidestep: ASP.NET Core Hosted



ASP.NET Core project exposes the API project and hosts the Blazor app too.

Demo



**Moving Blazor to ASP.NET Core Hosted**

# Accessing API Data Using HttpClient

---

# Interacting with the REST API

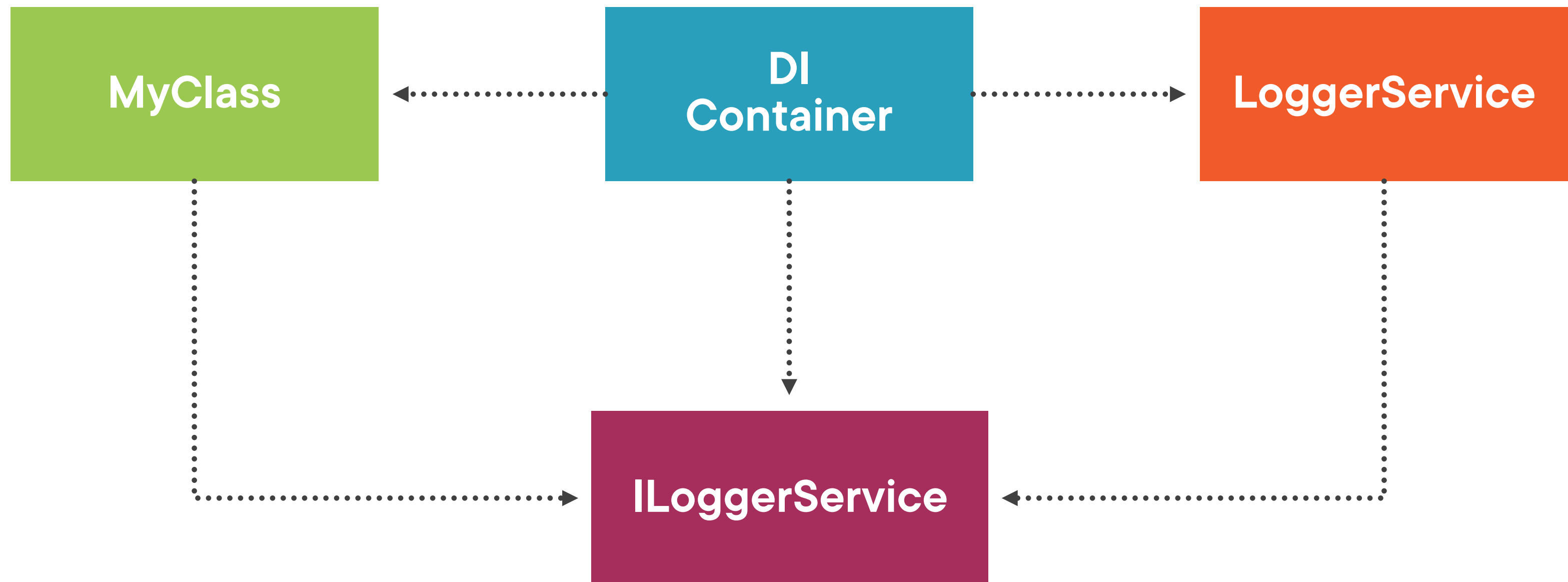


The diagram consists of two solid-colored rectangular boxes. The left box is maroon and contains the text 'HttpClient' in white. The right box is purple and contains the text 'IHttpClientFactory' in white. The boxes are positioned side-by-side, separated by a small gap.

**HttpClient**

**IHttpClientFactory**

# Sidestep: Dependency Injection (DI)





```
builder.Services.AddScoped(sp =>
    new HttpClient
    {
        BaseAddress = new Uri("http://<your-api-endpoint>")
    }
);
```

## Registering the HttpClient

```
[Inject]  
public HttpClient HttpClient { get; set; }
```

## Accessing the HttpClient in a Component

```
protected override async Task OnInitializedAsync()
{
    Employees = await HttpClient.GetFromJsonAsync<Employee[]>("api/employee");
}
```

## Working with the JSON Helper Methods

# Available Methods

**GetFromJsonAsync()**

**PostAsJsonAsync()**

**PutAsJsonAsync()**

**DeleteAsync()**



# IHttpClientFactory

Used to configure and create HttpClient instances in a central location

Support for named and typed HttpClient

Requires Microsoft.Extensions.Http package

```
builder.Services.AddHttpClient  
    <IEmployeeDataService, EmployeeDataService>  
        (client => client.BaseAddress = new Uri("https://localhost:44340/"));
```

## Registering in the Program

**Typed client is used here**

# Creating a Service Class

```
public class EmployeeDataService : IEmployeeDataService
{
    private readonly HttpClient _httpClient;

    public EmployeeDataService(HttpClient httpClient)
    {
        _httpClient = httpClient;
    }
}
```

# Demo



**Registering the HttpClient**

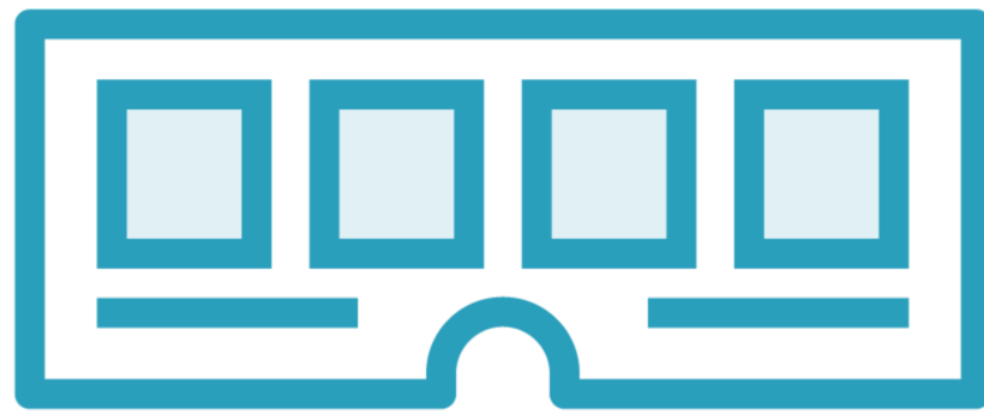
**Creating a data service**

**Updating the pages to use data from the API**



# Managing the Application State

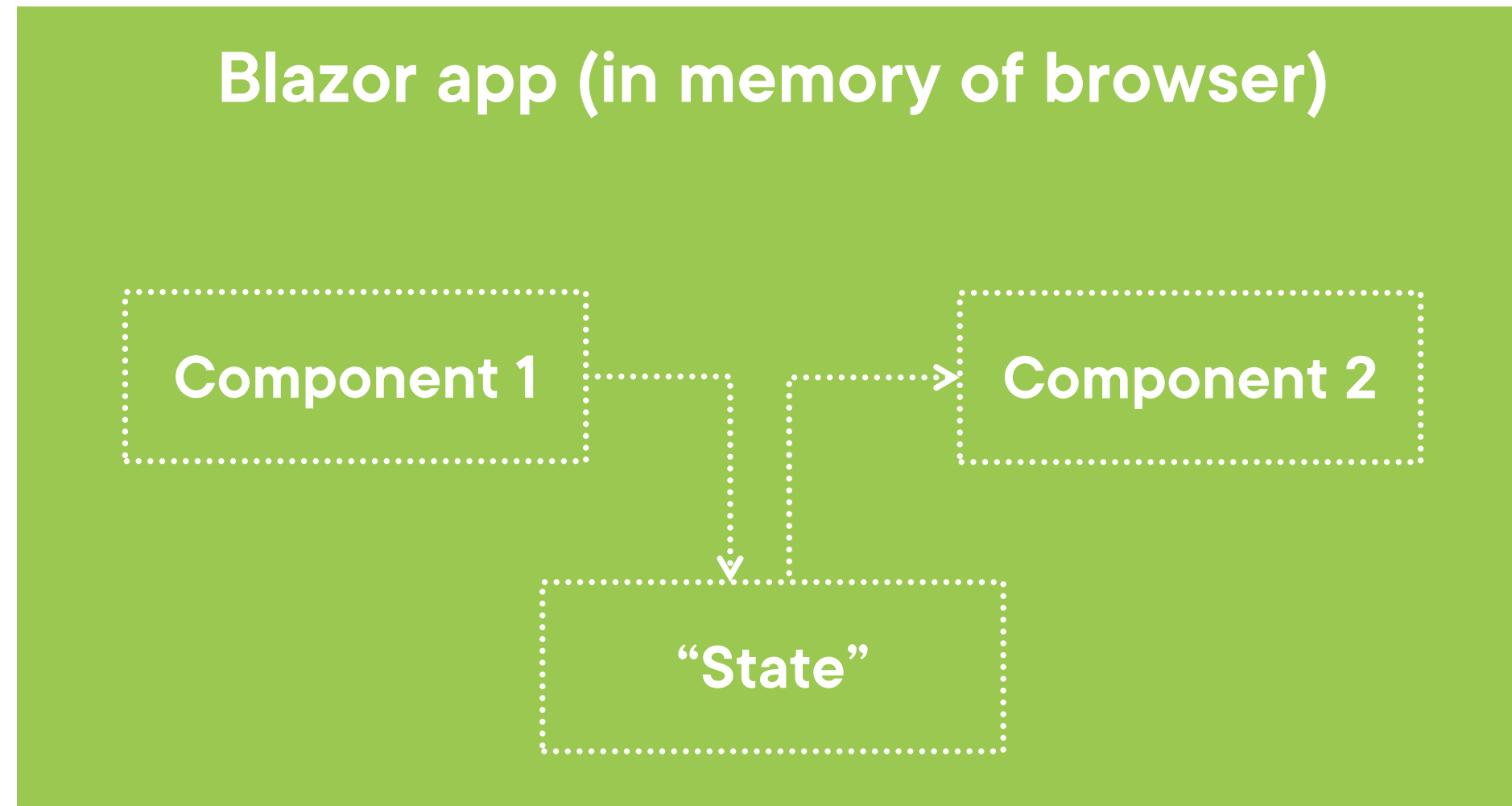
---



## Application state

- Blazor WASM is an application that's running in the memory of the browser
- By default, each component is an island that's recreated every time

# Application State



```
public class ApplicationState
{
    public int NumberOfMessages { get; set; } = 0;
}
```

## Creating an Application State Class

```
builder.Services.AddScoped<ApplicationState>();
```

Adding an Instance to the DI Container

```
[Inject]
public AppState? AppState { get; set; }

int a = AppState.NumberOfMessages;
```

## Accessing the Application State from Components



# Application State

This type of state is in-memory and will be removed when the application restarts!

# Demo



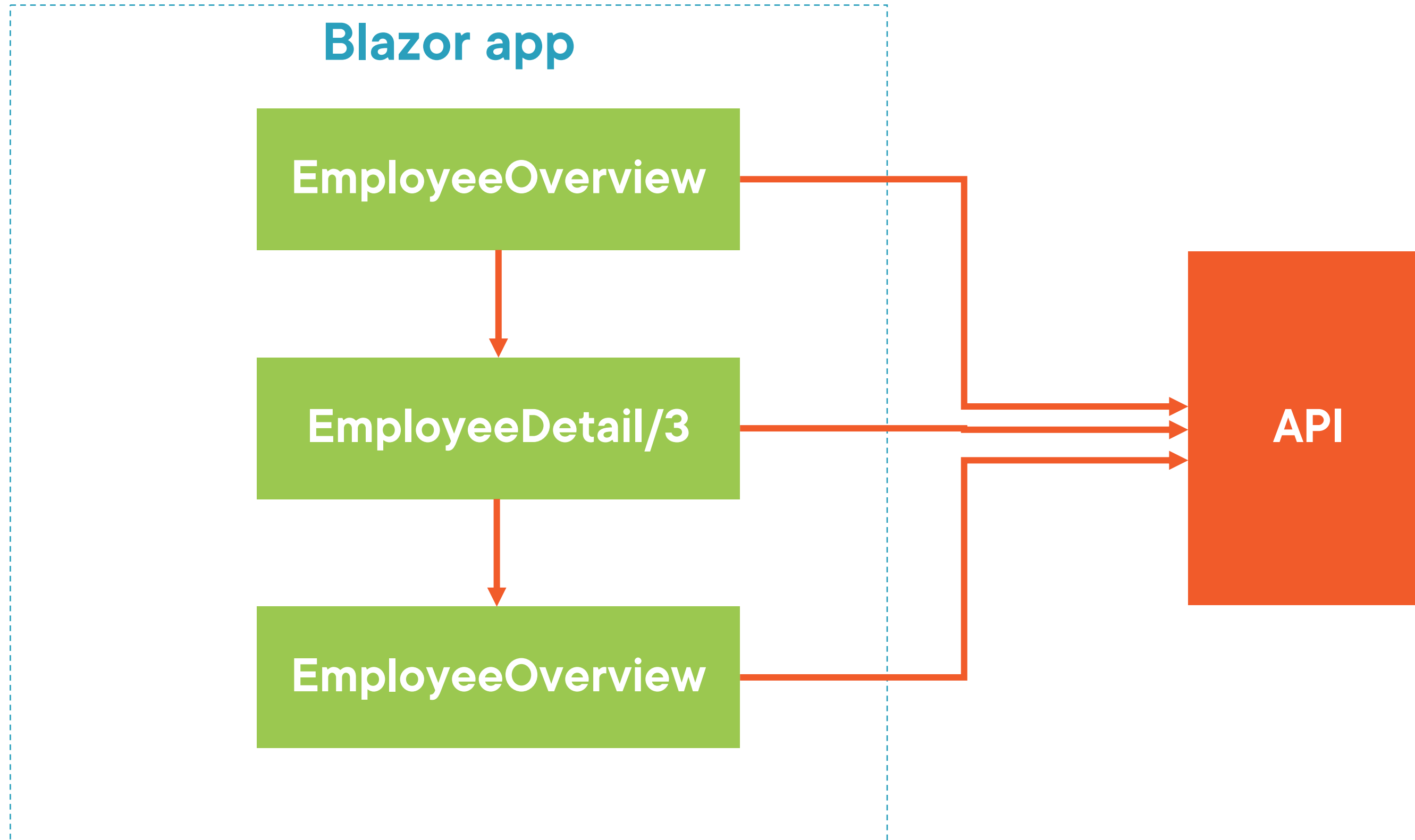
## **Adding application state**



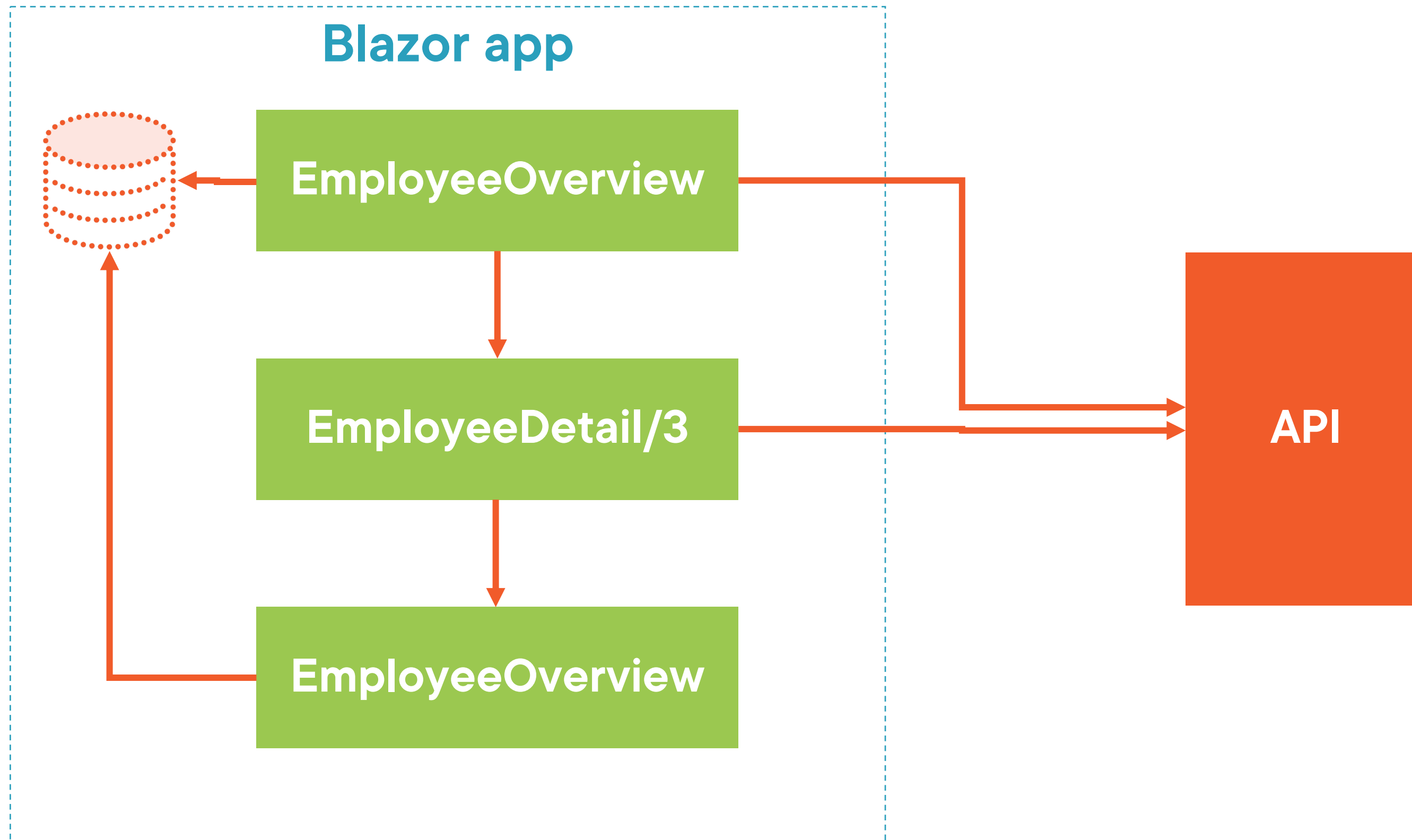
# Storing Data Locally

---

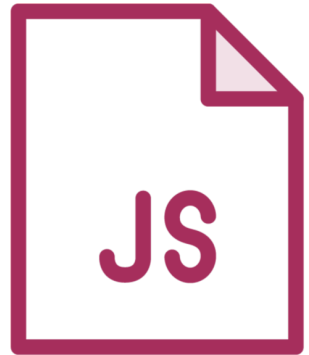
# Unneeded API Calls



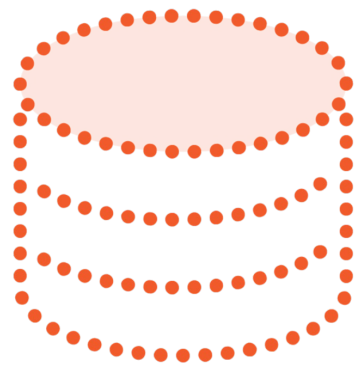
# Adding Local Storage



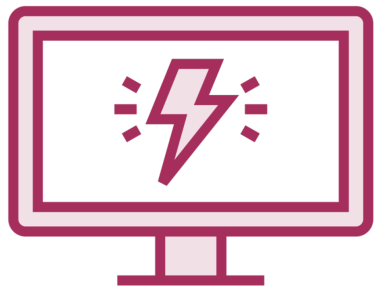
# Storing Data Locally



**Made possible through the browser, accessible using JavaScript**



**SessionStorage or LocalStorage**



**Possible to use from Blazor WASM too**

# Using Blazored LocalStorage

Blazored / LocalStoragePublic

Sponsor

<> Code

Issues11

Pull requests1

Discussions

Actions

Projects

Wiki

Security

Insights

main2 branches14 tags

Go to file

Add file

Code

chrissainty Removes support for NetStandard (#176)16a6597 on Mar 28209 commits

.github	Removes support for NetStandard (#176)	4 months ago
samples	Removes support for NetStandard (#176)	4 months ago
src	Removes support for NetStandard (#176)	4 months ago
tests/Blazored.LocalStorage.Tests	Upgrade to .NET 6 (#157)	8 months ago
.editorconfig	Added editconfig	15 months ago
.gitignore	Separates test extensions into it's own assembly (Fixes #149) (#150)	12 months ago
Blazored.LocalStorage.sln	Added ability to register local storage services as singleton (#175)	4 months ago
Directory.Build.props	Upgrade to .NET 6 (#157)	8 months ago
LICENSE	Initial commit	4 years ago
README.md	Added ability to register local storage services as singleton (#175)	4 months ago

☰ README.md

nuget v4.2.0

downloads 2M

Build & Test Main passing

Blazored LocalStorage

```
@inject Blazored.LocalStorage.ILocalStorageService localStorage  
  
var firstName = await localStorage.GetItemAsync<string>("EmployeeFirstName");
```

Using ILocalStorageService

# Available APIs

**SetItem()  
SetItemAsync()**

**GetItem()  
GetItemAsync()**

**ContainKey()  
ContainKeyAsync()**

**RemoveItem()  
RemoveItemAsync()**

# Demo



**Adding the Blazored.LocalStorage package**

**Extending the service with local storage support**



## Summary



**APIs offer us a way to access remote data**

**Use HttpClient and IHttpClientFactory to access remote API**

**Storing data locally will reduce load on API**



**Up next:**  
Creating a form