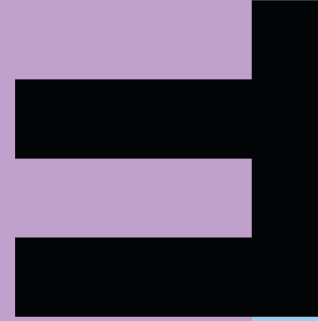


FHV

Vorarlberg University
of Applied Sciences



Application Integration and Security

Valmir Bekiri

Philipp Scambor

HTTP clients in .NET 8

Creating an API client with .NET 8

- If you want to consume an HTTP API then your .NET application becomes the client
- To send a HTTP request with `System.Net.Http.HttpClient` you need
 - To call the right method depending on the type of request you want to send (`PostAsync()`, `GetAsync()`, ...)
 - The request URI (string) containing query parameters if necessary
 - The request content (for POST, PUT, etc.)
 - Handle exceptions because network calls are always prone to failure
- Request content and response type depend on the type of API you aim to use
 - Can anything: XML, JSON, Raw text, etc.



Consuming a JSON API in .NET

- Use `System.Text.Json.JsonSerializer` to serialise to JSON or to parse JSON response bodies.
 - Serialise a class to a JSON string: `JsonSerializer.Serialize(someObject)`
 - Parse JSON: `JsonSerializer.Deserialize<T>(jsonString)`
 - Type T is returned and mirrors the JSON structure – if the JSON and the type do not match together deserialization fails!
- Tip: You can automatically generate a class from an example JSON with your IDE
 - Create a new empty class file
 - Visual Studio: Edit > Paste Special > Paste Json as classes
 - Rider: Edit > Paste > Paste Special: Json as classes



Consuming a JSON API in .NET

- Example on the right:
 - Sending a HTTP Post with JSON content
 - Parsing the response JSON into a class
 - Using async programming as network calls might take a while

```
1 using System.Text;
2 using System.Text.Json;
3 ..
4
5 class APIClient
6 {
7
8     public async Task<string?> callAPI()
9     {
10         var requestUri = "https://example.com/some/api";
11         var requestBody = new { some = "test" };
12         var jsonBody = JsonSerializer.Serialize(requestBody);
13         var httpContent = new StringContent(
14             jsonBody, Encoding.UTF8, "application/json");
15
16         try
17         {
18             var httpClient = new HttpClient();
19             var response = await httpClient.PostAsync(requestUri, httpContent);
20
21             // throws an exception if not success status code
22             response.EnsureSuccessStatusCode();
23
24             string jsonResponse = await response.Content.ReadAsStringAsync();
25             var responseObject = JsonSerializer
26                 .Deserialize<ExampleResponse>(jsonResponse);
27
28             return responseObject?.example;
29         }
30         catch (Exception ex)
31         {
32             // handle exception that may occur
33         }
34     }
35
36     private class ExampleResponse
37     {
38         public string example { get; set; }
39     }
40 }
41
```



That's it 😊
...for now

