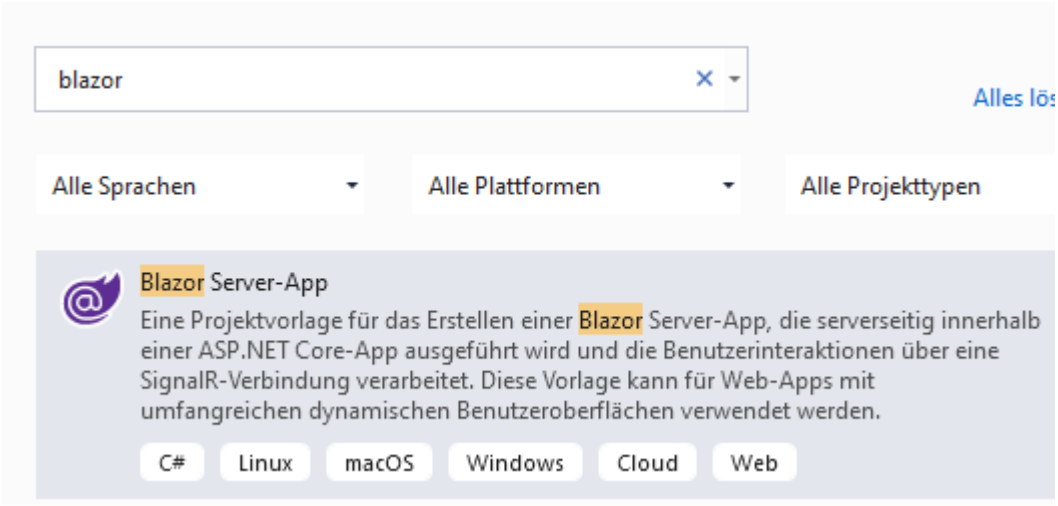
Lab Erzeugen einer Webassembly Blazor App

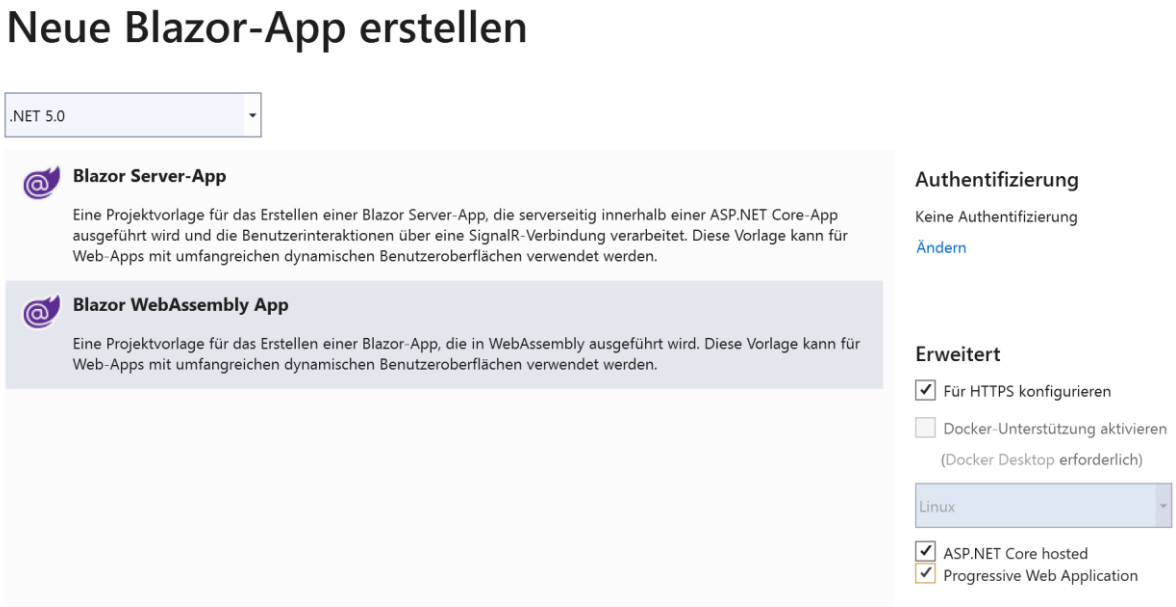
Starten Sie Visual Studio 2022

Erzeugen Sie ein neues Projekt vom Typ Blazor-App



Wählen Sie als Projektnamen BlazorLab1

Wählen Sie aus den Projekt Templates Blazor WebAssembly App und aktivieren sie die beiden Checkboxen „ASP.NET core hosted“ und Progressive Web Application.- Damit erhalten sie eine installierbare App



Starten Sie die Anwendung mit F5 (oder auch per Click auf IIS Express)

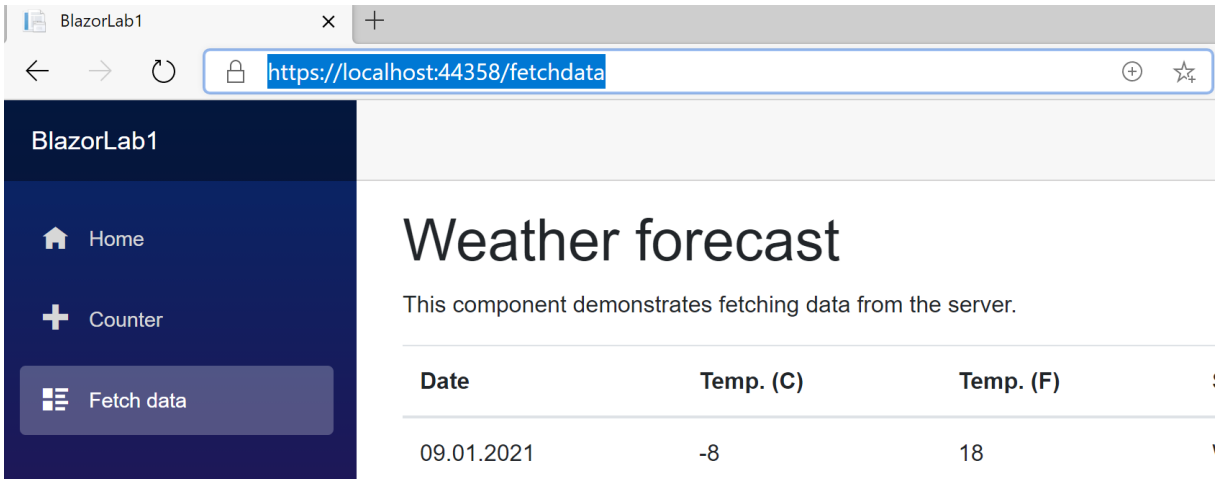
Der EDGE Browser startet automatisch

Bestätigen?

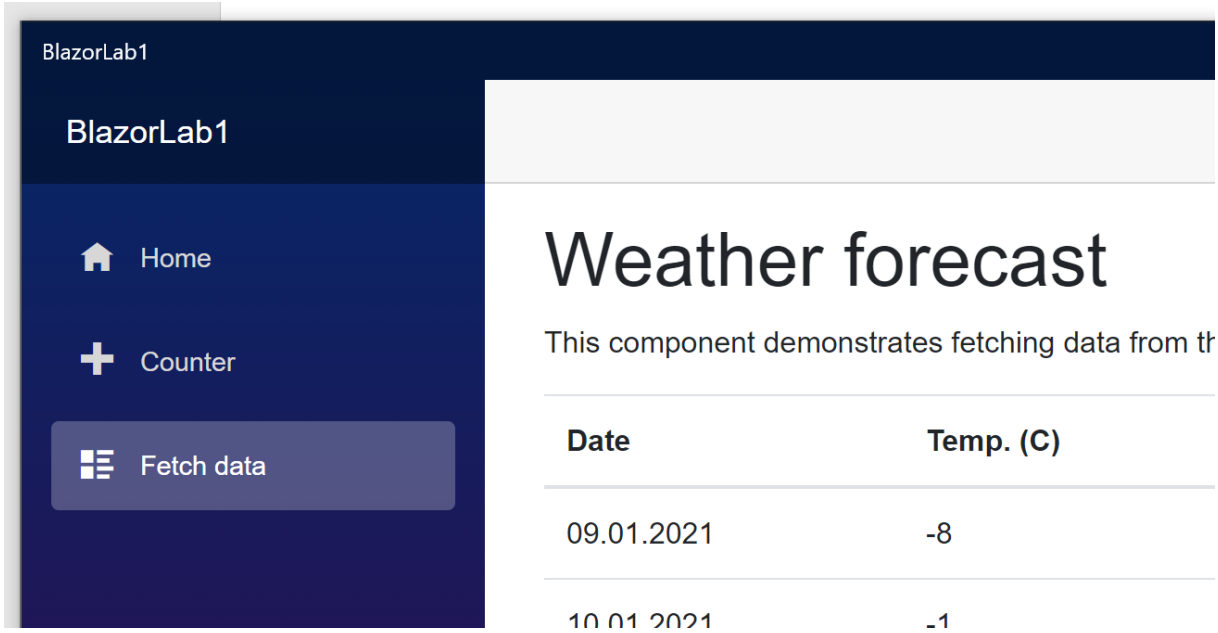
Rufen sie in der App die Funktion Fetchdata auf

Sie erhalten eine Liste von Wetterdaten im Browser

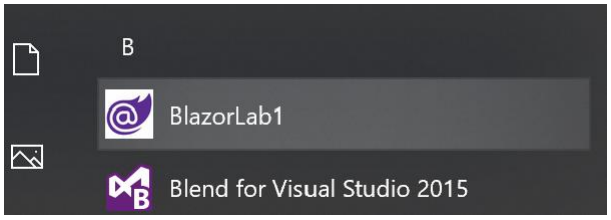
Drücken Sie das Plus Zeichen in der Url Eingabe (rechts) und installieren sie die App



Diese sieht nun wie folgt aus und hat keinen sichtbaren Browser



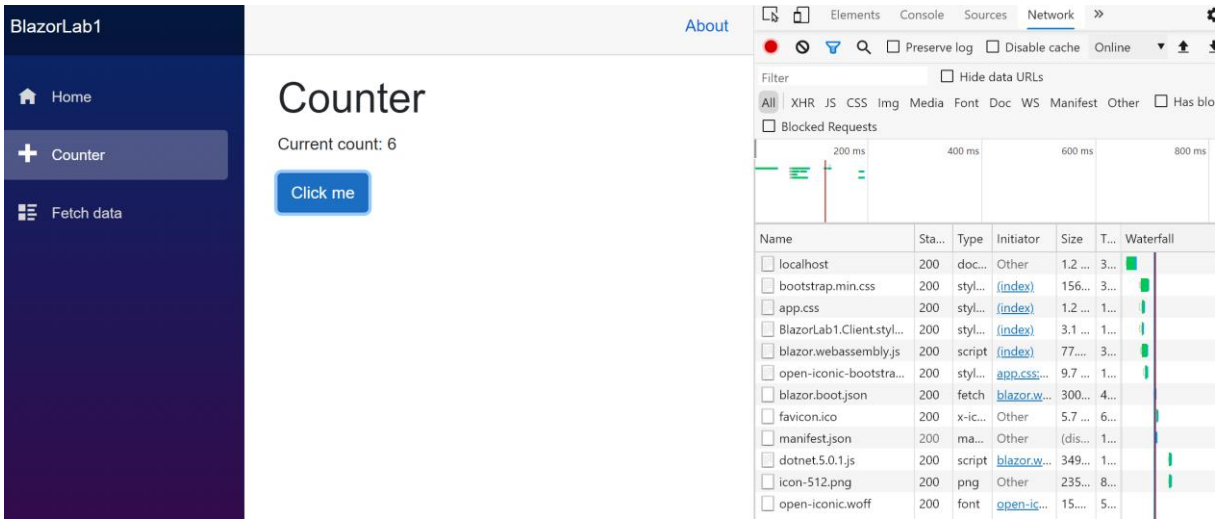
Öffnen Sie das Windows Menü und starten sie über den neu (automatisch erstellen Eintrag) die App BlazorLab1



Schließen Sie Visual Studio und versuchen die App noch einmal zu starten. Dies schlägt fehl.

Öffnen sie das Visual Studio Projekt wieder und starten es mit F5 im debugging Modus.

Drücken Sie im Edge Browser F12 um die Browser Tools zu öffnen und wechseln in den Netzwerk Reiter. Drücken sie SHIFT F5 um einen Reload zu erzwingen. Führen Sie in der Blazor App die Funktion Counter aus und drücken mehrfach auf den click me button

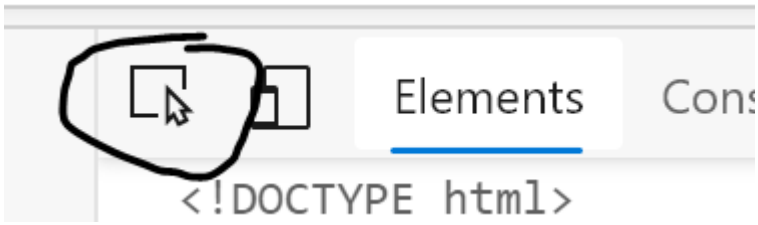


Sie beobachten

- Die .NET Assemblies werden nicht neu geladen
- Die HTML typischen Element werden neu geladen
- Es wird kein zusätzlicher Netzwerk Verkehr erzeugt, die Logik läuft im Browser

Wechseln sie den Reiter Elements und untersuchen die HTML Struktur der Browser Seite.

Mit der Funktion Inspect lassen sich einzelne DOM Elemente fokussieren




```
  <h1>Counter</h1>
▼ <p>
    "Current count: "
    "6"
  </p>
  <!--!-->
  <button class="btn btn-primary">Click me</button> == $0
</div>
```

Sie beobachten

- Reiner HTML Code

- Keine Code Logik sichtbar