Chapter 01: Introduction to Blazor WebAssembly

<table>
<thead>
<tr>
<th>Browser</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Edge</td>
<td>Current</td>
</tr>
<tr>
<td>Mozilla Firefox</td>
<td>Current</td>
</tr>
<tr>
<td>Google Chrome, including Android</td>
<td>Current</td>
</tr>
<tr>
<td>Safari, including iOS</td>
<td>Current</td>
</tr>
<tr>
<td>Opera, including Android</td>
<td>Current</td>
</tr>
<tr>
<td>Microsoft Internet Explorer</td>
<td>Not Supported</td>
</tr>
</tbody>
</table>
Command Prompt:

c:\Users\Toi>dotnet --version
5.0.102

c:\Users\Toi>
**SQL Server 2019**

**Express Edition**

Installation has completed successfully!

<table>
<thead>
<tr>
<th>INSTANCE NAME</th>
<th>CONNECTION STRING</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQLEXPRESS01</td>
<td>Server=localhost\SQLEXPRESS01;Database=master;Trusted_Connection=True</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SQL ADMINISTRATORS</th>
<th>SQL SERVER INSTALL LOG FOLDER</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOI-ACER,Toi</td>
<td>C:\Program Files\Microsoft SQL Server\150\Setup Bootstrap\Log\20210220</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FEATURES INSTALLED</th>
<th>INSTALLATION MEDIA FOLDER</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQLENGINE</td>
<td>C:\SQL2019\Express_ENU</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VERSION</th>
<th>INSTALLATION RESOURCES FOLDER</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.0.2000.5, RTM</td>
<td>C:\Program Files\Microsoft SQL Server\150\SSE\Resources</td>
</tr>
</tbody>
</table>

Connect Now  Customize  Install SSMS  Close
Chapter 02: Building Your First Blazor WebAssembly Application

Hello World!

Blazor WebAssembly is amazing!

Catch All
Route: error/type/3

<table>
<thead>
<tr>
<th>Constraint</th>
<th>.NET Type</th>
<th>Sample Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>System.Boolean</td>
<td>TRUE, false</td>
</tr>
<tr>
<td>datetime</td>
<td>System.DateTime</td>
<td>2020-12-31, 2020-12-31 5:26pm</td>
</tr>
<tr>
<td>decimal</td>
<td>System.Decimal</td>
<td>42.99, -1,000.01</td>
</tr>
<tr>
<td>double</td>
<td>System.Double</td>
<td>1.234, -1,001.01e8</td>
</tr>
<tr>
<td>float</td>
<td>System.Single</td>
<td>1.234, -1,001.01e8</td>
</tr>
<tr>
<td>guid</td>
<td>System.Guid</td>
<td>CD2C1638-1638-72D5-1638-DEADBEF1638, CD2C1638-1638-72D5-1638-DEADBEF1638</td>
</tr>
</tbody>
</table>
Blazor WebAssembly App
A project template for creating a Blazor app that runs on WebAssembly and is optionally hosted by an ASP.NET Core app. This template can be used for web apps with rich dynamic user interfaces (UIs).

Configure your new project

Project name
Demo

Location
E:\Blazor

Solution name
Demo

Place solution and project in the same directory
Additional information

Blazor WebAssembly App

Target Framework

.NET 5.0 (Current)

Authentication Type

None

Configure for HTTPS

ASP.NET Core hosted

Progressive Web Application

Back  Create

Demo

Home

Counter

Fetch data

Hello, world!

Welcome to your new app.

How is Blazor working for you? Please take our brief survey and tell us what you think.

Counter

Current count: 1

Click me
Weather forecast

This component demonstrates fetching data from the server.

<table>
<thead>
<tr>
<th>Date</th>
<th>Temp. (C)</th>
<th>Temp. (F)</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>5/6/2018</td>
<td>1</td>
<td>33</td>
<td>Freezing</td>
</tr>
<tr>
<td>5/7/2018</td>
<td>14</td>
<td>57</td>
<td>Bracing</td>
</tr>
<tr>
<td>5/9/2018</td>
<td>-13</td>
<td>9</td>
<td>Freezing</td>
</tr>
<tr>
<td>5/9/2018</td>
<td>-16</td>
<td>4</td>
<td>Balmy</td>
</tr>
<tr>
<td>5/10/2018</td>
<td>-2</td>
<td>29</td>
<td>Chilly</td>
</tr>
</tbody>
</table>
App component

MainLayout component

sidebar
top-row
main

NavbarMenu component

LayoutComponentBase.Body

<Counter Inc />

Increment

Counter
Current count: 1
Click me (1)

Counter
Current count: 5
Click me (5)

Demo

About

Sorry, there's nothing at this address.
Choose Template Type

This wizard will allow you to export a project or project item from the current solution to a template which future projects can then be based upon.

Which type of template would you like to create?

- **Project template**
  A project template will allow a user to create a new project based on your exported project. A user will be able to utilize your template from the New Project dialog box for client projects and from the New Website dialog box for websites.

- **Item template**
  An item template will allow a user to add your item to one of their existing projects. Your template will be available to the user from the Add New Item dialog box.

From which project would you like to create a template?

```
Demo
```

Select Template Options

**Template name:**
- `EmptyBlazorProject`

**Template description:**
- This template creates an empty Blazor WebAssembly project.

**Icon Image:**

**Preview Image:**

**Output location:**
- `C:\Users\YourName\Documents\Visual Studio 2019\My Exported Templates\EmptyBlazorProject.zip`

- ![Automatically import the template into Visual Studio](on)
- ![Display an explorer window on the output files folder](on)
**Blazor App**

Project templates for creating Blazor apps that run on the server in an ASP.NET Core app or in the browser on WebAssembly (wasm). These templates can be used to build web apps with rich dynamic user interfaces (UIs).

**Empty Blazor WebAssembly App**

This template creates an empty Blazor WebAssembly project.
Chapter 03: Building a Modal Dialog Using Templated Components

Alert

Today is Friday.

[OK]

Alert

Today is Saturday.

[OK]

Quick List [3]

Enter New Item:

[OK] [Cancel]

Empty Blazor WebAssembly App

This template creates an empty Blazor WebAssembly project.

C#  Web

Configure your new project

Empty Blazor WebAssembly App  Web

Project name

[ModalDialog]

Location

[ElBlazor]

Solution

Create new solution

Solution name

ModalDialog

[Place solution and project in the same directory]

Back  Create
Chapter 04: Building a Local Storage Service Using JavaScript

Interoperability (JS Interop)

Window Size: 1920 x 969

Get Window Size

Local Storage Service

Data: 

Save to Local Storage  Read from Local Storage
Configure your new project

Empty Blazor WebAssembly App

Project name
LocalStorage

Location
E:\Blazor

Solution
Create new solution

Solution name
LocalStorage

Place solution and project in the same directory

Back  Create
Chapter 05: Building a Weather App as a Progressive Web App (PWA)

Service Workers

- Offline
- Update on reload
- Bypass for network

https://localhost:44392/

Source: service-worker.js 2

Received 3/25/2021, 8:06:09 PM

Status:
- #400 activated and is running
- #500 waiting to activate

Clients: https://localhost:44392/ focus

Simple to Complex

- Offline Page
- Offline Copy with Offline Page
- Advanced Caching
- Offline Copy of Pages
- Cache-first Network
- Background Sync
Configure your new project

Empty Blazor WebAssembly App

Project name
WeatherForecast

Location
E:\Blazor

Solution
Create new solution

Solution name
WeatherForecast

Place solution and project in the same directory

Back  Create

localhost:44352 wants to Know your location

Allow  Block
You are offline.
### 5-Day Weather Forecast

<table>
<thead>
<tr>
<th>Monday, March 29, 2021</th>
<th>Tuesday, March 30, 2021</th>
<th>Wednesday, March 31, 2021</th>
<th>Thursday, April 1, 2021</th>
<th>Friday, April 2, 2021</th>
</tr>
</thead>
<tbody>
<tr>
<td>clear sky</td>
<td>light rain</td>
<td>light rain</td>
<td>clear sky</td>
<td>broken clouds</td>
</tr>
<tr>
<td>74°F / 51°F</td>
<td>71°F / 52°F</td>
<td>60°F / 47°F</td>
<td>64°F / 42°F</td>
<td>64°F / 49°F</td>
</tr>
</tbody>
</table>

**Recently added**

5-Day Weather Forecast

**Customize and control 5-Day Weather Forecast**

- **Remove "5-Day Weather Forecast"**?
  - Also clear data from Chrome (localhost:44352)
Chapter 06: Building a Shopping Cart Using Application State

<table>
<thead>
<tr>
<th>Products</th>
<th>Price</th>
<th>Add to Cart</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charger</td>
<td>$15</td>
<td></td>
</tr>
<tr>
<td>Ear Buds</td>
<td>$22</td>
<td></td>
</tr>
<tr>
<td>Key Chain</td>
<td>$1</td>
<td></td>
</tr>
<tr>
<td>Travel Mug</td>
<td>$8</td>
<td></td>
</tr>
<tr>
<td>T-Shirt</td>
<td>$20</td>
<td></td>
</tr>
</tbody>
</table>

**Your Order**

- Charger - $15
- Key Chain - $1
- Travel Mug - $8

**Total:** $24

Empty Blazor WebAssembly App

This template creates an empty Blazor WebAssembly project.
Chapter 07: Building a Kanban Board Using Events

Submit
This is a button

Submit
This is another button

Empty Blazor WebAssembly App
This template creates an empty Blazor WebAssembly project.
Configure your new project

Empty Blazor WebAssembly App  

Project name

KanbanBoard

Location

EA\Blazor - ...

Solution

Create new solution -

Solution name

KanbanBoard

Place solution and project in the same directory

Back  Create

Task  enter new task  Add Task
Chapter 08: Building a Task Manager Using ASP.NET Web API

Additional information

Blazor WebAssembly App

Target Framework

- .NET 5.0 (Current)

Authentication Type

- None
- Configure for HTTPS
- ASP.NET Core hosted
- Progressive Web Application

<table>
<thead>
<tr>
<th>JSON Helper Method</th>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetFromJsonAsync</td>
<td>GET</td>
<td>Read</td>
</tr>
<tr>
<td>PostAsJsonAsync</td>
<td>POST</td>
<td>Create</td>
</tr>
<tr>
<td>PutAsJsonAsync</td>
<td>PUT</td>
<td>Update</td>
</tr>
<tr>
<td>HttpClient.DeleteAsync</td>
<td>DELETE</td>
<td>Delete</td>
</tr>
</tbody>
</table>

TaskManager

Enter Task

- Enter Task
- □ Read Email
- ✅ Buy Milk

Home
**Blazor WebAssembly App**

A project template for creating a **Blazor** app that runs on WebAssembly and is optionally hosted by an ASP.NET Core app. This template can be used for web apps with rich dynamic user interfaces (UIs).

Here's an example of configuring a new project within the Solution Explorer.

**Solution Explorer**

- **Solution ‘TaskManager’ (3 of 3 projects)**
  - TaskManager.Client
  - TaskManager.Server
  - TaskManager.Shared

**Add New Scaffolded Item**

- **API Controller - Empty**
- **API Controller with read/write actions**
- **API Controller with actions, using Entity Framework**

Click here to go online and find more scaffolding extensions.
Add Data Context

New data context type: TaskManager.Server.Data.TaskManagerServerContext

Add API Controller with actions, using Entity Framework

Model class: TaskItem (TaskManager.Shared)
Data context class: TaskManager.Server.Data.TaskManagerServerContext
Controller name: TaskItemsController

SQL Server Object Explorer

Add SQL Server EXPRESS2019

Databases
Security
Server Objects
Projects - TaskManager

TaskManager
Tables
- System Tables
- File Tables
- External Tables
- dbo._EFMigrationsHistory
dbo.Taskitem
<table>
<thead>
<tr>
<th>TaskItemId</th>
<th>TaskName</th>
<th>IsComplete</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Buy Milk</td>
<td>False</td>
</tr>
<tr>
<td>2</td>
<td>Read Email</td>
<td>False</td>
</tr>
<tr>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
</tbody>
</table>

```json
[{
  "taskId": 1,
  "taskName": "Buy Milk",
  "isComplete": false
},
{
  "taskId": 2,
  "taskName": "Read Email",
  "isComplete": false
}]
```

- Buy Milk
- Read Email

- Buy Milk
Chapter 09: Building an Expense Tracker Using the EditForm Component

<table>
<thead>
<tr>
<th>Input Component</th>
<th>HTML Rendered</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputCheckbox</td>
<td>&lt;input type=&quot;checkbox&quot; &gt;</td>
</tr>
<tr>
<td>InputDate&lt;TValue&gt;</td>
<td>&lt;input type=&quot;date&quot; &gt;</td>
</tr>
<tr>
<td>InputFile</td>
<td>&lt;input type=&quot;file&quot; &gt;</td>
</tr>
<tr>
<td>InputNumber&lt;TValue&gt;</td>
<td>&lt;input type=&quot;number&quot; &gt;</td>
</tr>
<tr>
<td>InputRadio</td>
<td>&lt;input type=&quot;radio&quot; &gt;</td>
</tr>
<tr>
<td>InputRadioGroup</td>
<td>&lt;input type=&quot;radio&quot; &gt;</td>
</tr>
<tr>
<td>InputSelect&lt;TValue&gt;</td>
<td>&lt;select&gt;</td>
</tr>
<tr>
<td>InputText</td>
<td>&lt;input&gt;</td>
</tr>
<tr>
<td>InputTextArea</td>
<td>&lt;textarea&gt;</td>
</tr>
</tbody>
</table>

Add/EditExpense

- The Date field is required.
- The Amount field is required.

ValidationSummary

Date

mm/dd/yyyy

The Date field is required.

Amount

The Amount field is required.
Add/EditExpense

Date
mm/dd/yyyy

Vendor

Description

Type

Amount

Paid

Save

Blazor WebAssembly App
A project template for creating a Blazor app that runs on WebAssembly and is optionally hosted by an ASP.NET Core app. This template can be used for web apps with rich dynamic user interfaces (UIs).
Configure your new project

Blazor WebAssembly App  C#  Linux  macOS  Windows  Cloud  Web

Project name

ExpenseTracker

Location

E:\Blazor

Solution name

ExpenseTracker

Add Data Context

New data context type: ExpenseTracker.Server.DataExpenseTrackerServerContext

Add  Cancel

Add API Controller with actions, using Entity Framework

Model class: ExpenseType (ExpenseTracker.Shared)


Controller name: ExpenseTypesController

Add  Cancel

[{"id":1,"type":"Airfare"},{"id":2,"type":"Lodging"},{"id":3,"type":"Heal"},{"id":4,"type":"Other"}]
ExpenseTracker

Expenses
None Found

ExpenseTracker

Add/EditExpense

Date
mm/dd/yyyy

Vendor

Description

Type

Amount

Paid

Save

- The Date field is required.
- The Vendor field is required.
- The Expense Type field is required.
- The Amount field is required.