
kopius

kopius Academy

Webinar | Blazor: Straight to the meat

March 27th, 2024

kopius

Intro

kopius

Who am I?

Sebastián Vansteenkiste

- Software Developer with 10 years of experience (mostly .Net)
- 1-year tenure at Kopius
- Father of a two-year-old
- Kendo/laido Enthusiast (Japanese fencing)

kopius

What is Blazor?

kopius

Why use Blazor?

kopius

Blazor vs. *React* vs. *Angular* vs. *Vue*

Blazor

+ SPA friendly
+ Same component code for Wasm and Server projects
+ Leverages .Net ecosystem

- May have scalability issues
- Hot reload and debugging tend to fail
- Choice complexity

React.js

+ Easy to learn
+ Wide library of 3rd party components
+ SEO Friendly

- Tendency to abuse 3rd-party packages
- Tendency to over-fragment components
- Complicated state management

Vue.js

+ Lightweight and performance-oriented
+ Versatile
+ PWA friendly

- Excessive data mutations
- Excessive custom syntax

Angular

+ Time-tested and widely used

- Performance degradation with binding watchers
- Tendency to fall-back on jQuery

kopius

Blazor hosting models

WebAssembly (Wasm)

+ Runs entirely on client's browser
+ Works offline
+ Can be turned into PWA
+ Lighter Server resource usage

- Large initial download size (.Net assemblies)
- Incompatible with (very?) old Browsers
- Not SEO-friendly out of the box
- May be resource intensive on the client

Server

+ Code runs entirely on Server
+ Lighter download size (HTML+CSS+SignalR)
+ Older browser support
+ Lighter resource load on client

- Resource intensive for the Server (danger of multiple sessions per user)
- UI changes require roundtrips to Server
- Does not work Offline

"Hybrid" (MAUI)

+ Native application support

- Uses MAUI/Xamarin for compilation and must be published via platform's App Store

Selective (.Net 8)

+ Easily turned into PWA
+ "Best of both Wasm and Server"

- Only available since .Net 8
- "Brand new" so untested

kopius

Key *Blazor* Concepts / Architecture

Pages / Components

- HTML / Razor UI
  - Data binding
  - Form Validation
  - JS Interop
- C# code behind
  - Event Handlers
  - Obsers & Subscribers
- per-component CSS

Services / Repositories

- Dependency Injection

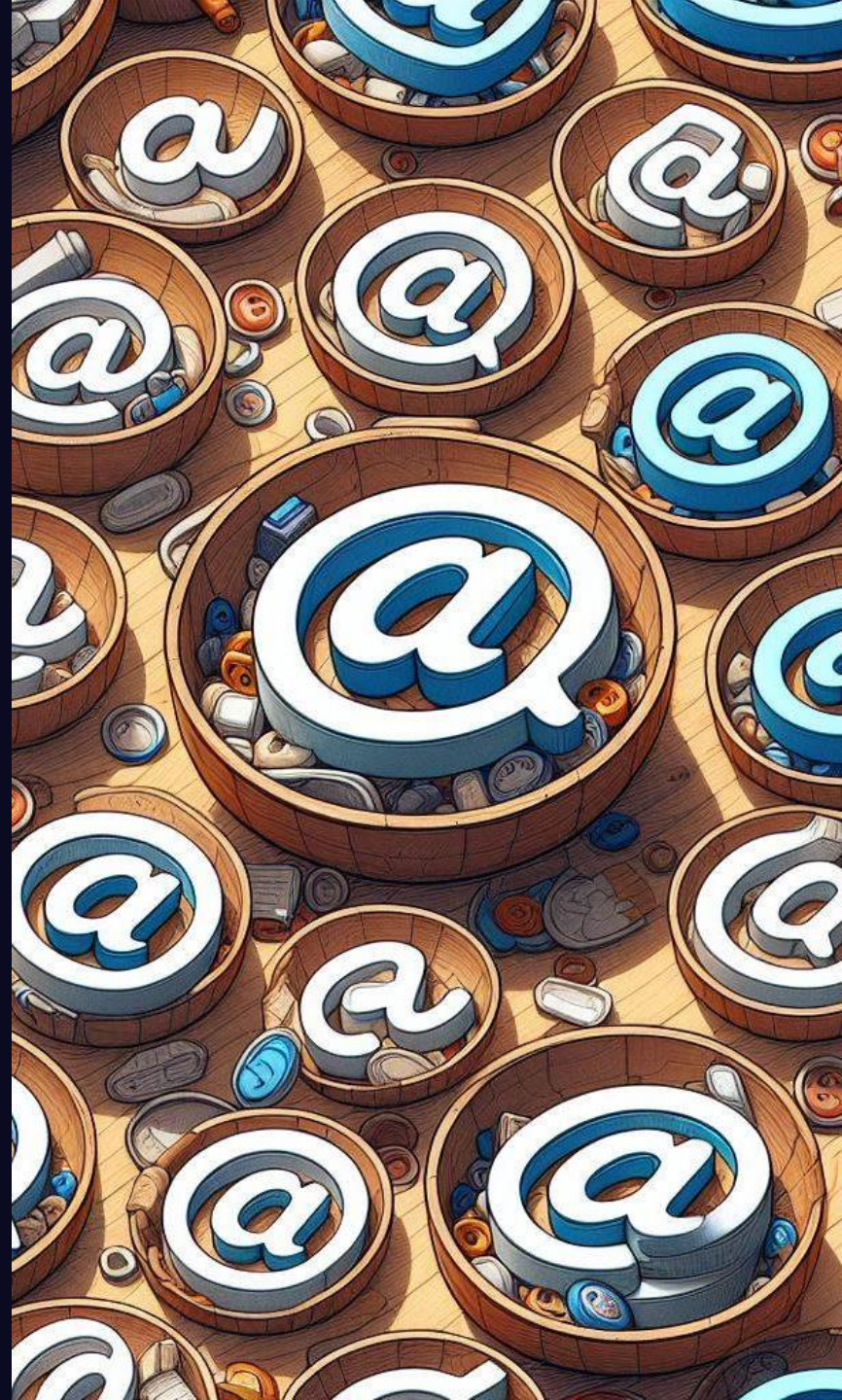
Unit Tests

- xUnit / nUnit / etc.
- Moq / etc.
- bUnit UI tests

kopius

Demo I – Mise en Place

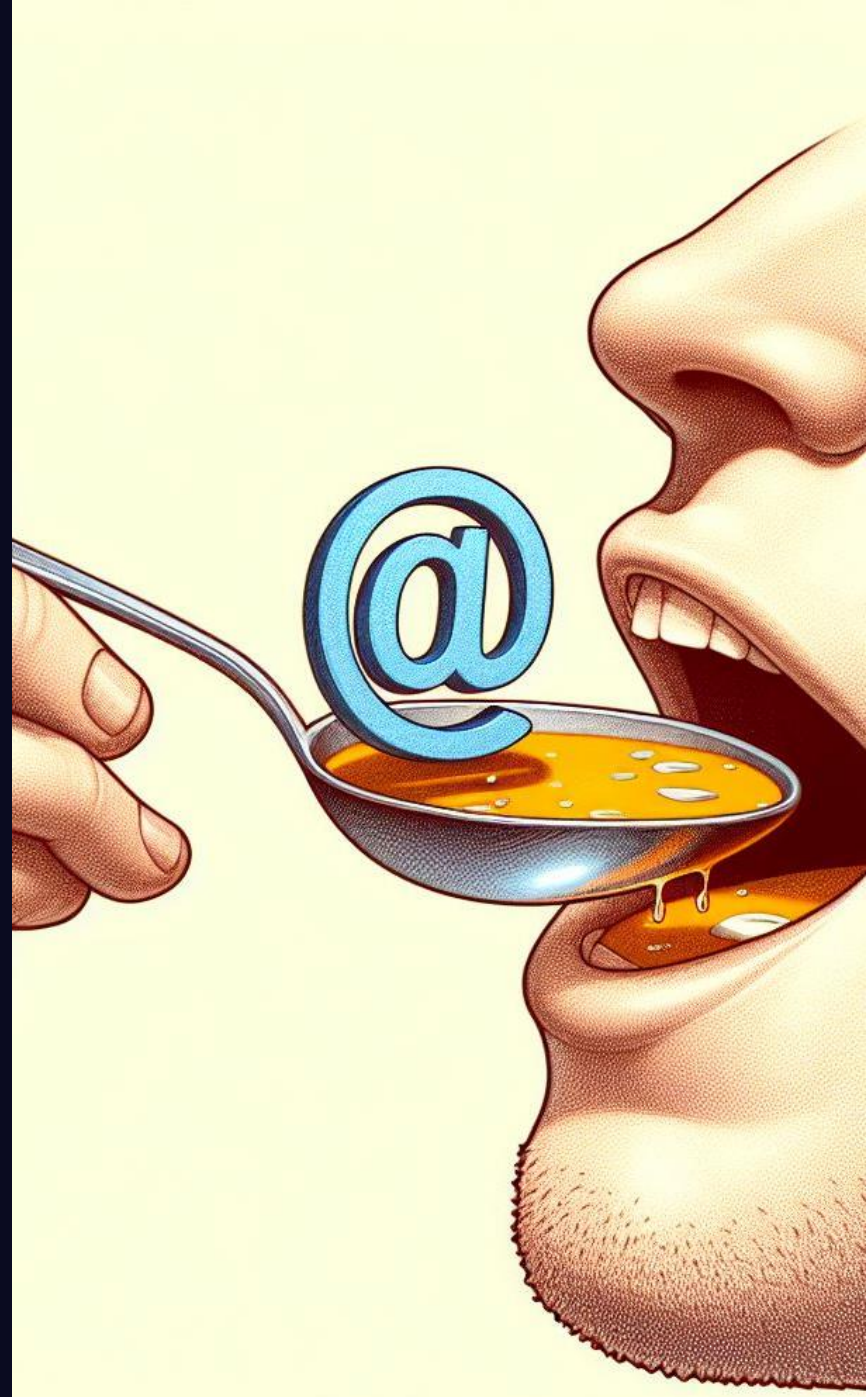
- Setting up new repo
- Create new project
- Wipeout default example
- Boilerplate set-up & covering MainLayout and Index with Unit Tests



kopius

Demo II – *Taste* Driven Development

- Implement Counter component
- Implement Counter list component



kopius

Demo III – Sautéeing

- Parameters & Data binding
- Event Handling
- Basic Shared-State Management

kopius

Demo IV - Plating

- Form validation

kopius

Demo V - Garnishing

- JS Interoperability

kopius

Q&A

Github repo: github.com/ZippoLag/MeatyBlazor/branches

Contact: svansteenkiste@kopiustech.com

kopius

Further Resources

- **Component Lifecycle docs** https://learn.microsoft.com/en-us/aspnet/core/blazor/components/lifecycle
- **Microsoft Blazor Learning Path** https://learn.microsoft.com/en-us/training/paths/build-web-apps-with-blazor
- **Blazor University** https://blazor-university.com (simple, practical guide to get started)
- **Blazor School** https://blazorschool.com (more detailed tutorial with emphasis on Server VS WebAssembly)
- **Blazor MAUI** https://learn.microsoft.com/en-us/dotnet/maui/user-interface/controls/blazorwebview
- **.Net 8 Full-Stack news** https://www.youtube.com/watch?v=0MBeYc9qcWY
- **Blazor PWA** https://learn.microsoft.com/en-us/aspnet/core/blazor/progressive-web-app
- **Blazor SignalR Chat example** https://learn.microsoft.com/en-us/aspnet/core/blazor/tutorials/signalr-blazor

Bonus: DALL-E's failed attempts at drawing people "tasting @s"