1.

~ năm gần đây, nhìu a/c làm việc vs .net sẽ nghe tới cụm từ blazor.

Nay sẽ tìm hiểu blazor là j? tại sao Microsoft lại tâm huyết vs blazor như v

2.

Tìm hiểu công nghệ này, có j thú vị, hay, có lquan j tới blazor?

Tìm hiểu blazor, đặc tính

Gthieu hosting blazor -> van dung vào iu cầu cụ thể

Ưu nhược điểm mỗi loại

Demo, mổ xẻ proj. Component blazor sử dụng dễ dàng hay ko?

3. ko có

4.

JS và browser -> có đôi có cặp ko thể tách rời

… cho đến khi web assembly xuất hiện

Wasm có j dbiet khiến browser phải dóm ngó nó như thế

5.

Mô hình ptrien udung web đang sử dụng

Tách biệt client server

Client: dùng framwork library nổi tiếng, ví dụ….

Server: ngôn ngữ C#, java, php,…

-> tương tác = web api

-> mô hình ko có j cần phải bàn, có đội ngũ dev riêng biệt

Có BE, FE tách biệt -> thực tế, ai cũng muốn hướng tới full-stack

Ví dụ bản thân -> ko dễ dàng tiếp cận FE -> mất time để học -> mong muốn dùng chính ngôn ngữ BE cho FE -> may mắn gần đây ra đời webassembly

6.

Là 1 CNghe mới

Định dạng nhị phân chạy trên máy ảo (stack-based virtual machine)

stack-based virtual machine: chạy app = C# -> need CLR, java -> java virtual machine,…

webassembly -> biên dịch các BE language -> chuẩn chung chạy trên browser

Tóm lại -> code 1 ngôn ngữ BE -> có thể chạy dc trên browser như JS

Từ xưa đã có: flash adobe, Silverlight Microsoft -> Cnghe độc quyền -> cần phải học. Ng dùng cần cài extension -> bất tiện, rủi ro security

Wasm ko phải vậy -> ko phải extension. Chỉ cần có modern browser là use dc (laptop, mobile,…)

Wasm công bố 2015 -> 2017 đa số modern browser use dc -> 2019…

7.

2 định dạng

.wat: có thể hiểu dc

.wasm: browser sẽ sử dụng định dạng này

8.

-> gthieu trang web: https://mbebenita.github.io/WasmExplorer/

9.

Có thể viết được định dạng đó -> ko ai rảnh để làm như thế

Mỗi language có compiler riêng -> wasm -> chạy trên stackbase

Túm lại là concept sau wasm là:

Chạy precompiled code dc viết = 1 ngôn ngữ BE bất kì trên web

Arm -> mac or linux

10.

Vậy tại sao cần, có thể học them 1 FE language khác, như là JS đang thống trị, vua của ngôn ngữ web

Wasm -> sinh ra ko để thay thể js -> sinh ra để song hành -> chạy những tác vụ JS làm ko thực sự tốt

Thực sự cá nhân bản thân chưa code JS nhiều nhưng theo 1 số nguồn tham khảo tìm đc thì

JS thường thấy ko làm tốt nhất là các tác vụ realtime cùng 1 lúc, vdu như: image & video processing, 3D gaming, AR/VR,...

11.

=> Dùng với lí do sau

Efficient fast -> bởi vì là binary

Safe: memory-safe, sandboxed execution environment

12.

Show some games after this slides: https://www.webassemblygames.com/

https://blazorgames.net/

13.

Ít nhìu thấy tiềm năng wasm, nó tuyệt vời ntn

Microsoft thấy tiềm năng wasm -> tạo ra blazor gia nhập cuộc chơi vs wasm

Blazor là j, có lquan j tới wasm?

Thêm info, ở event ra mắt .net 6 (2021), Microsoft dành ra 3 section để nói về blazor -> cho thấy Microsoft tâm huyết ntn

14.

Thư viện phía client dung .NET trên nền tảng wasm -> ptrien ứng dụng SPA

Viết = C# -> cú pháp razor (làm việc vs MVC sẽ gặp nhiều)

Blazor: free and open source -> theo chiến lược ptrien Microsoft từ lúc ra mắt .netcore v1

15.

Làm mn nhớ đến Silverlight (C# on browser) -> có time cực kì ptrien -> NHƯNG -> cài đặt them plugin, sự ptrien chóng mặt của JS và HTML5 -> Silverlight khai tử

Blazor ra mắt -> Mn lầm tưởng Microsoft muốn hồi sinh Silverlight -> thay vỏ -> NHƯNG ko -> ko cần plugin, chỉ cần modern browser

Share libraries between client and server -> tkiem chi phí

16.

Tóm lại -> C# cho cả FE và BE

17.

Blazor Server ra mắt cùng .netcore 3.0

Blazor Web Assembly ra mắt cùng .netcore 3.1

LTS for Blazor -> chính thức hỗ trợ cùng .net 6

-> mang vào dự án thực tế ko cần lo lắng -> có Microsoft support

18.

ko có

19.

Blazor server: thực thi hết ở server -> thông qua SignalR -> tương tự bên client, update UI,… -> gửi SignalR cho server

SignalR: kthuat của Microsoft, hdong trên cơ chế websocket

Blazor server ko sử dụng kthuat web assembly

Blazor webassembly: kế thừa toàn bộ ưu điểm của webassembly (tốc độ, khả năng tương thích JS,…)

Khác biệt lớn nhất

Blazor Web assembly download toàn bộ application về browser

Blazor server: download 1 phần, cần thì download tiếp -> chỉ hoạt động khi có connection giữa client và server

-> hiểu rõ hơn -> qua slide tiếp theo về ưu nhược điểm

20.

ko có

21.

Good performance

Progressive Web App (PWA) : website nhìn ko khác j 1 ứng dụng thông thường

Khi ng dung mở PWA ở trình duyệt -> thông báo cài đặt ở dthoai -> cài xong -> icon hiển thị -> user dung như 1 app -> PWA hỗ trợ offline

Old browser cannot be used -> IE ko hỗ trợ, còn lại support hết

Need times to developer tools for devs -> vì do mới ptrien gần đây

22.

ko có

23.

Server-side rendering -> for SEO

Can be used in older browser -> web assembly ko làm dc

API private -> ưu điểm chung của server side rendering

SignalR trên mọi event -> luôn có độ trễ dù là nhỏ nhất

Túm lại

Performance < SEO -> dung blazor server -> có serverside rendering

Ưu tiên ng dung -> dung blazor wasm

24.

ko có

25.

ko có

26.

Đã đi qua lý thuyết và 2 kiểu hosting model

-> sang phần demo để hiểu rõ hơn

-> ưu tiên demo wasm

dotnet new blazorwasm –h -> help

dotnet new blazorwasm -> default template

dotnet watch run

Bật F12 lên -> show số lượng MB cần load và show DLL (trong cache storage)

-> load nhiều -> NHƯNG -> chỉ là lần đầu tiên -> cache lại rồi, DLL là cái rất ít khi thay đổi, nếu có them mới thì chỉ download 1 xíu thôi, mấy tram KB -> rất là nhanh

Bật solution lên -> có thể dung VS Code or VS

-> mở file program.cs

-> có rootcomponent -> nó sẽ hiển thị cái j -> có index.html -> id = app -> hiển thị cái này

-> shared: component dung chung

-> có file _import.razor -> import library

-> wwwroot: như là 1 web app bth thôi, lưu static html, css style

Về cơ bản, blazor webassembly là tập hợp các file có đuôi razor

-> pages: file có đuôi .razor, style riêng từng page -> nói sơ sơ về component (@page, các tag, @code: logic component, có thể tách ra file .CS khác =cách dung partial class)

Coding:

1. Gọi component trong page

2.Tạo 1 partial class cho code lúc nãy (partial class: tách nhỏ để dễ qly code hơn)

3. tạo 1 file JS trong folder scripts -> viết hàm alert()…

-Gọi script trong index

-Add @using Microsoft.JSInterop in _imports.razor

-Call JS in razor page

27.

ko có

28.

ko có