

Blazor Server

MUG.net – Nantes

Romain AVONDE – Expert .NET chez Capgemini

05/05/2020

Sommaire



PETIT HISTORIQUE



BLAZOR : UNE
NOUVELLE FAÇON DE
TRAVAILLER



DE BLAZORWASM À
BLAZOR SERVER



UNE FAÇON
DIFFÉRENTE DE
PENSER COMPOSANT



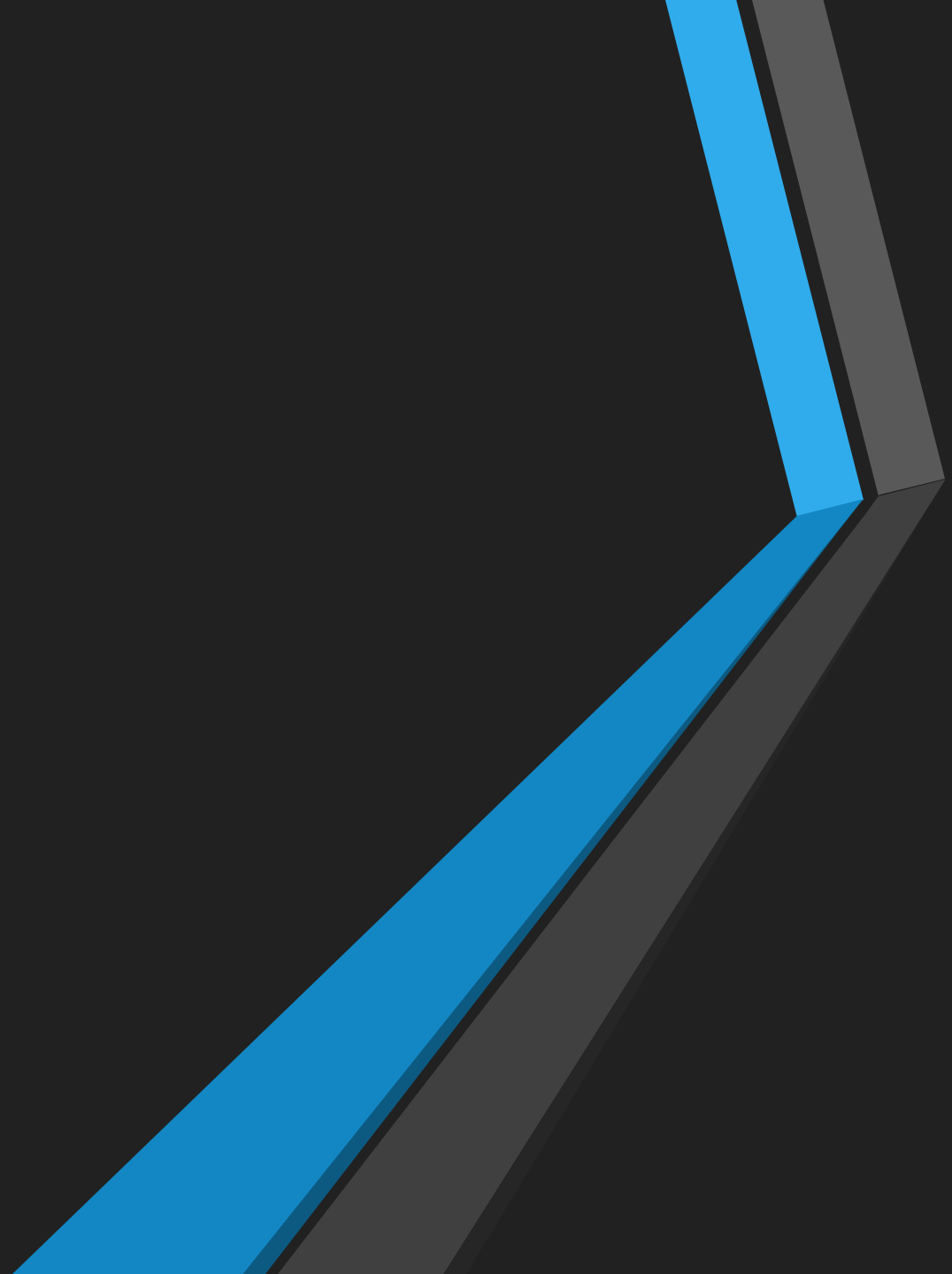
REX



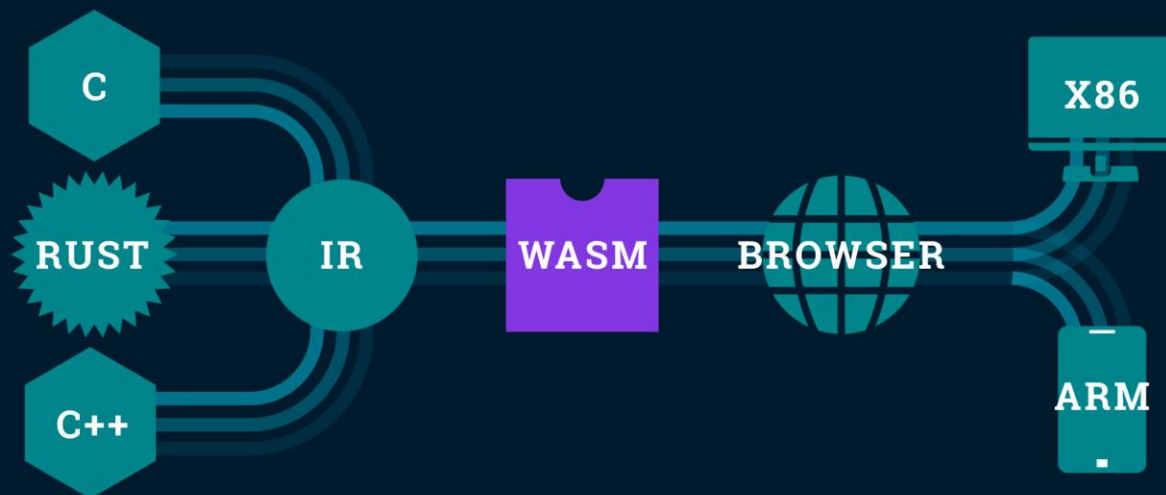
LA SUITE ?

Petit historique

Du framework au core



HOW DOES WEBASSEMBLY WORK?



Les WebAssembly ?

- Date de 2016
- WebAssembly est un nouveau type de code
- Langage proche de l'assembleur.

```
;; hello_world.wat

(module

  ;; Import our myprint function
  (import "env" "jsprint" (func $jsprint (param i32)))

  ;; Define a single page memory of 64KB.
  (memory $0 1)

  ;; Store the Hello World (null terminated) string at byte offset 0
  (data (i32.const 0) "Hello World!\00")

  ;; Export the memory so it can be access in the host environment.
  (export "pagememory" (memory $0))

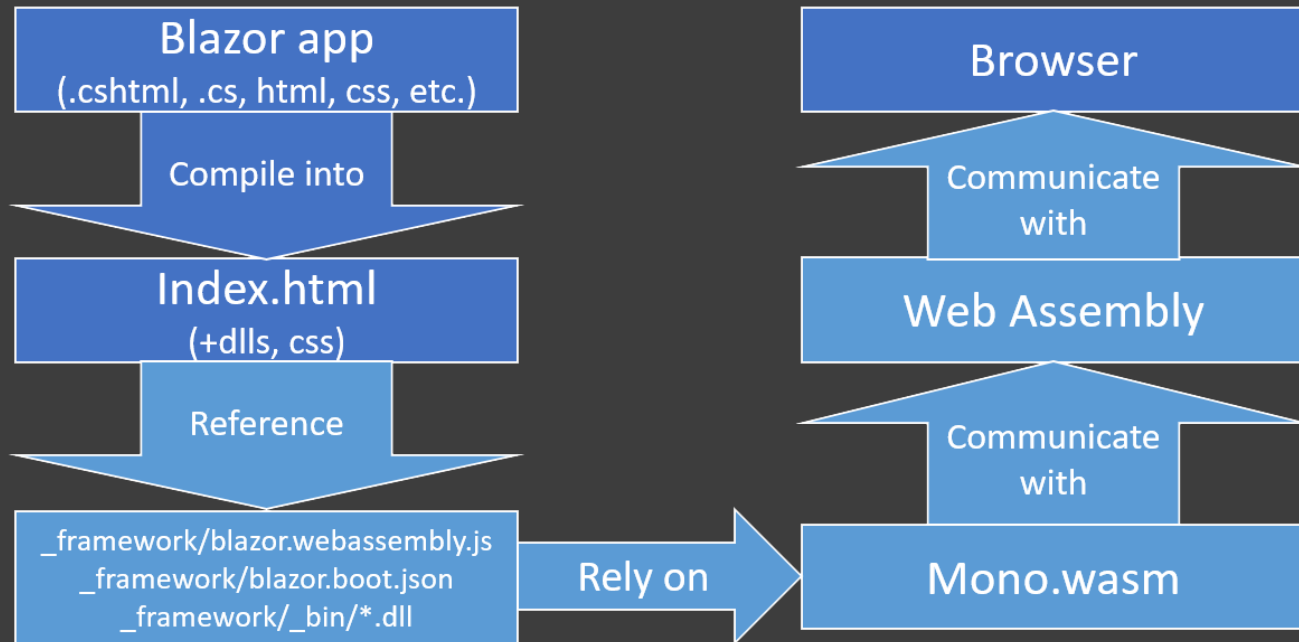
  ;; Define a function to be called from our host
  (func $helloworld
    (call $jsprint (i32.const 0))
  )

  ;; Export the wasmprint function for the host to call.
  (export "helloworld" (func $helloworld))
)
```

Exemple de code

Blazor : une nouvelle façon de travailler

Du web sans Javascript



Une nouvelle
implémentation
de Mono

- Développement par composant
- Le routage
- La simplicité d'implémentation de l'Asp.net Core

Des mécaniques repris
de framework reconnus
et de développement
« maison »

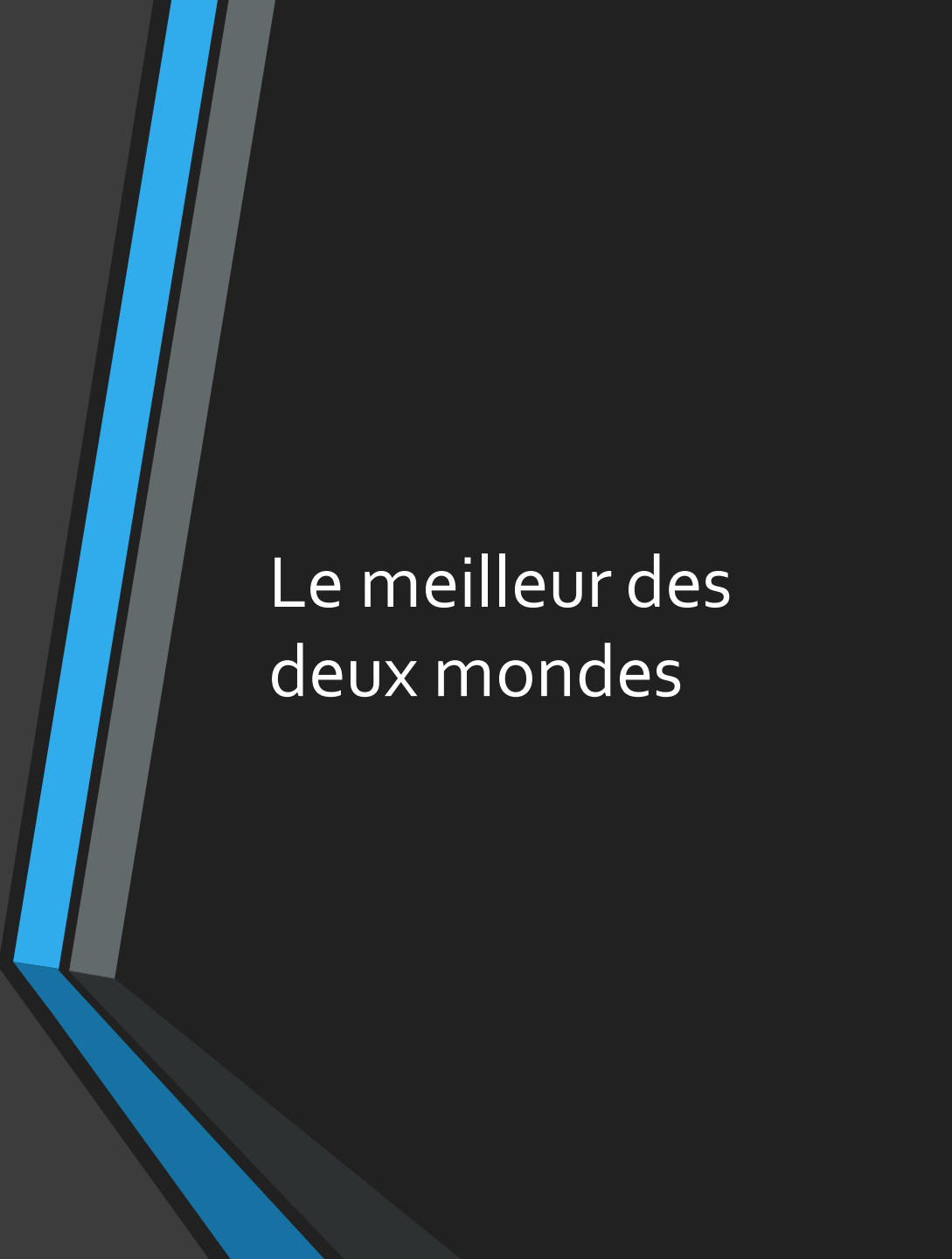
De Blazor WASM à Blazor Server

Une idée qui paraît temporaire mais qui a un avenir...

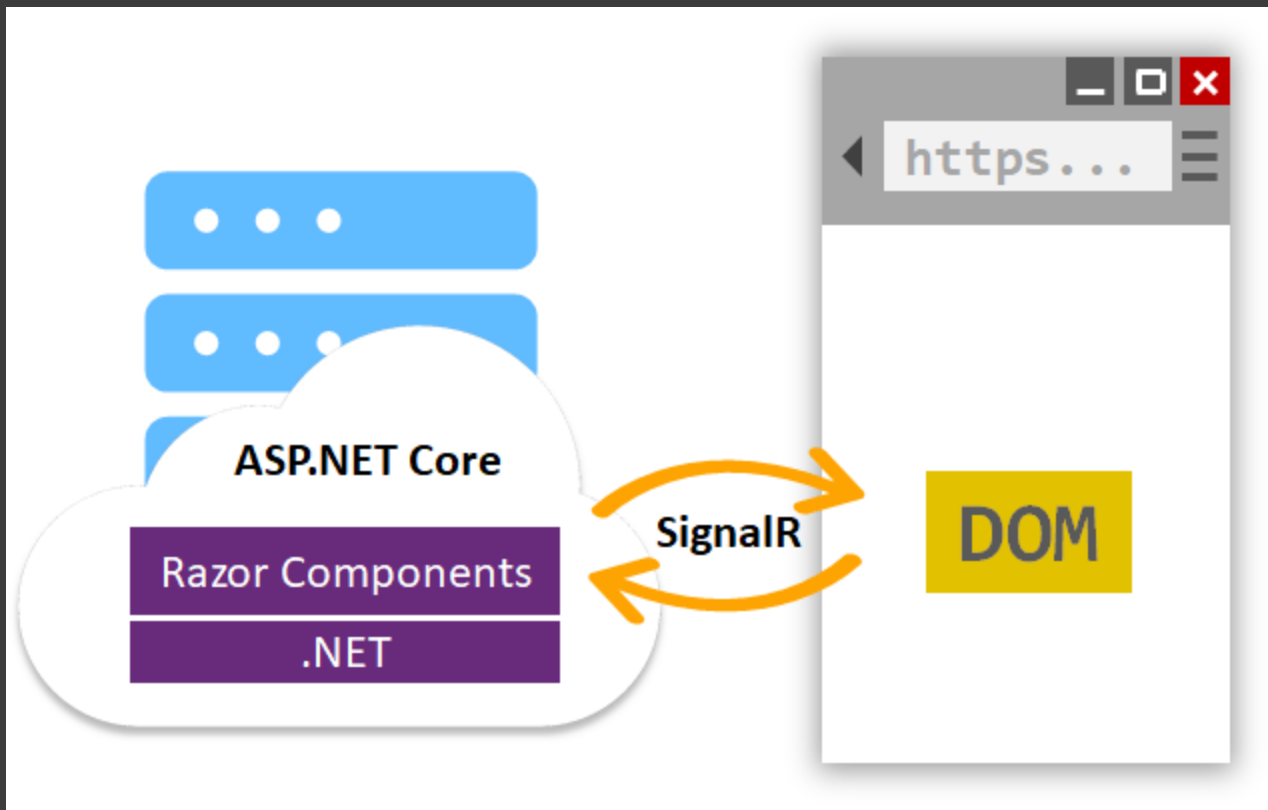
- Nécessité de chargé mono et « toute » la CLR au premier chargement.
- La taille des dll chargés.
- Toute les api javascript ne sont pas implémentées en Blazor, donc nécessité d'interagir avec du Javascript.
- Les WebAssembly toujours en développement et donc pas stable.

Les limitations de Blazor WASM

- Le modèle de développement par composants séduit.
- Frustration des limites des WebAssembly (pour l'instant...).
- Pourquoi ne pas utiliser les composants en les exécutant côté serveur ?



Le meilleur des deux mondes



La naissance
de Blazor
Server

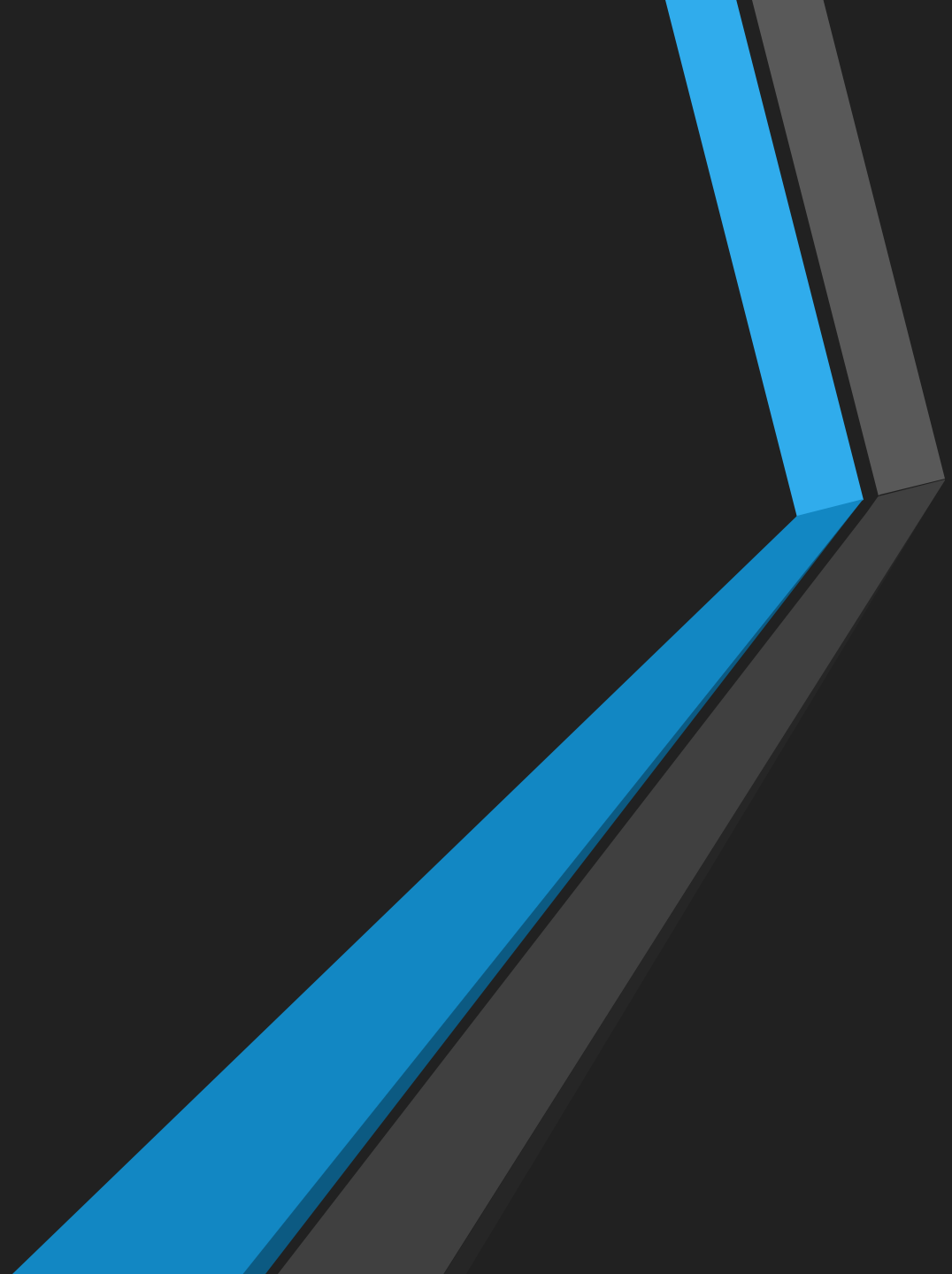
The background features a dark grey to black gradient. On the right side, there is a complex geometric shape composed of several overlapping, parallel lines. These lines are primarily in shades of blue and grey, creating a sense of depth and movement. The lines appear to be part of a larger, partially visible structure that extends towards the top right corner of the frame.

Une nouvelle façon de
penser composants.



Démo

REX



- La réactivité de l'interface
- Le routage
- La simplicité d'implémentation de l'Asp.net Core
- Le principe des composants
- La puissance de calcul du back pour les différentes actions clientes.



Les plus

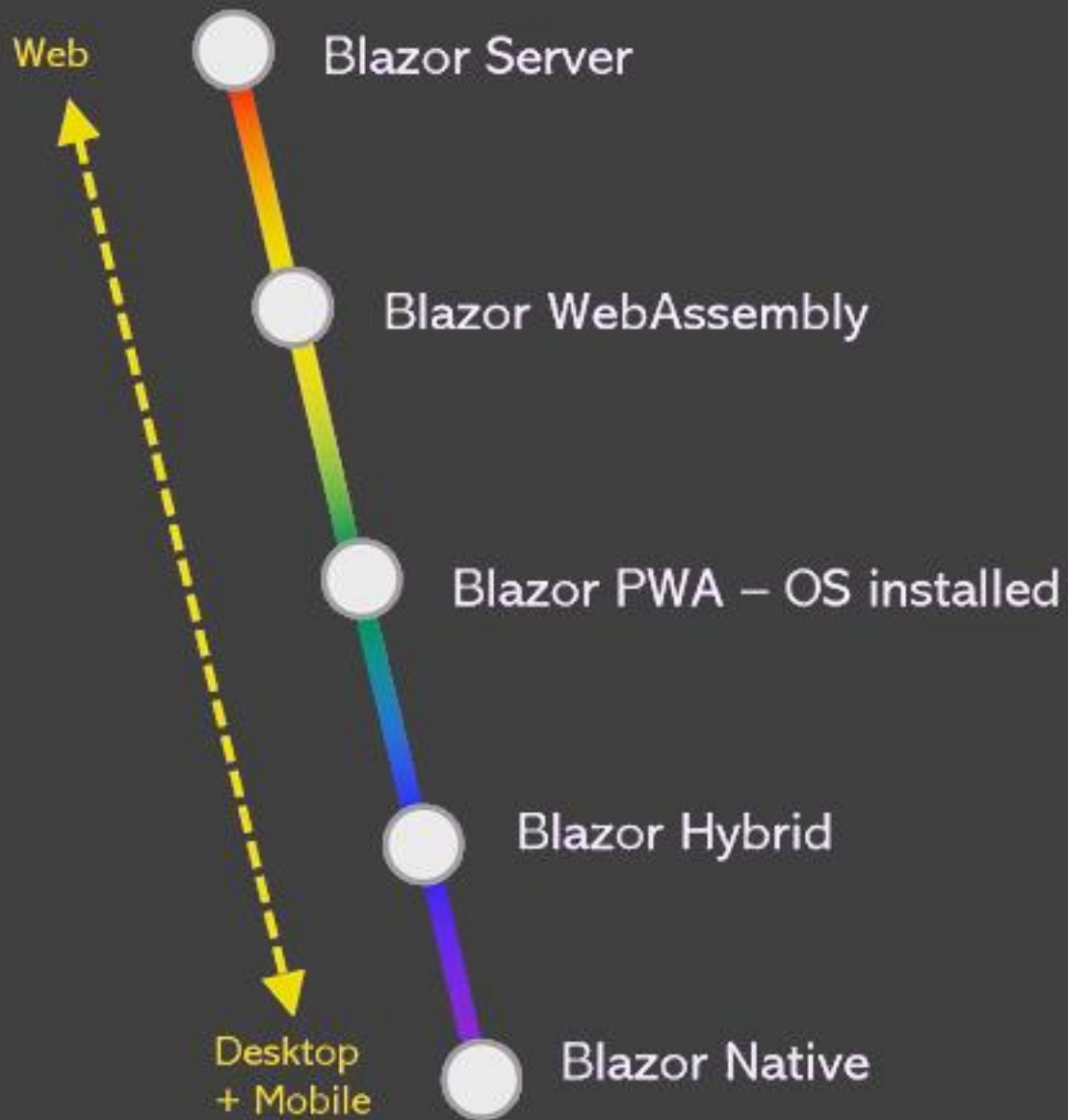
- Communication par WS : Quid du nombre de connexion en // ?
- Technologie nouvelle :
 - Difficulté d'avoir des retours sur les bonnes pratiques
 - Peu de documentation sur les forums,

Les limites



Et la suite ?

Du Blazor partout !!



Web app
Every interaction handled on server
Prerendered HTML (optional)

Web app with client-side execution
Loaded from web server
Can work offline via Service Worker

Appears as native app (own window)
Works offline or online

Native .NET renders to Electron / WebView
Appears as native app (own window)
Works offline or online

Same programming model, but
rendering non-HTML UI

Nouveautés

- Création d'exécutables par défaut
- Outils dotnet locaux
- Suppression de la dépendance avec Newtonsoft.Json
 - (Ecriture d'une version 2x plus rapide)
- Plages et Index
 - `var slice = a[i1..i2];`
 - `Index i1 = 3; // number 3 from beginning`
 - `Index i2 = ^4; // number 4 from end`
- Flux asynchrones
- TLS 1.3 et OpenSSL 1.1.1 sous Linux
- SerialPort pour Linux
- Compilation hiérarchisée par défaut