BLAZOR + 認証

WRITE BY 松井 敏

VISUAL STUDIO 2019 LAUNCH AT KANSAI

2019-06-22

MYSELF

- 松井 敏 A.K.A 森理 麟 BELONG CODEER
- MY COMMUNITY : C#読書会

マンガ読んだ‼の紹介

- 「マンガ読んだ‼」はクリックするだけで簡単に読んだマンガを記録できるサービス
- 作者やシリーズ全部でまとめて1回で登録も簡単！
- マンガは集英社、講談社、小学館などから9万冊以上！毎日自動更新で、日々増加中！！

「マンガ読んだ!!」の紹介

- クイズ番組見てたら、マンガ1万冊読んだという人が出てきた。え？僕、それ以上余裕で読んでいると思った。普通ならその証明は出来ないけれど、このサイトなら証明できる！
- あと単純に他の人が読んだ1万冊の内容とか知ってみたいなーと。最終的には僕がそのクイズに出れば良いのか

アジェンダ

- BLAZOR
- IDENTITY
- BLAZOR × IDENTITY

アジェンダ

- BLAZOR
- IDENTITY
- BLAZOR × IDENTITY

BLAZORの特徴

- **BLAZOR**は**C#**の**SPA**フレームワーク
- **WEB ASSEMBLY**により**C#**でクライアントが書ける
- 現世代ブラウザなら動く。つまりスマホでも動く
- **ASSEMBLY**だが、速度面のアドバンテージはない

BLAZOR

- 知っている人が少なかったら簡易DEMOる

BLAZORの結論

- C#で作れる！

BLAZOR

- クライアントでJSを書かなくて良い、つまりクライアントでもVS+.NET+C#環境で書ける
- 使い慣れた言語だけでなくライブラリ、開発環境のパフォーマンスがそのまま引き出せる！
- 既存のコードもアルゴリズムはそのまま使える

BLAZOR

- JSをVSで書くことも可能だが、VSのポテンシャルが半分もだせない感じ
- 依存関係などJSならでは問題も多い。もちろん使わなければ避けられる。

BLAZOR

- ま、実際、**BLAZOR**でもバージョンとか困ることもある。これは実験的プロジェクトなのでしょうがない。それでもクライアントで**JS**を使うよりもマシな印象。

BLAZOR(SPA)の利点

- SPAフレームワークなので、COMPONETが使える
- オブジェクト指向的にパーツを組み合わせてクライアントが作れるので、使いまわしがきく

BLAZOR

- アルゴリズムはSHAREDで共有化できる。
- つまりクライアントからサーバまで全て同じ言語で開発出来る
- サーバーで書いたC#と同じ定義をクライアントJSで再度書くスイッチングコストがZERO

BLAZORの弱点

- CSHTMLはCSで書くことに比べるとパフォーマンスが落ちる（ASP.NET CORE 3 PREVIEW 5からCSHTML→RAZORに変更。使いやすくは…なってない）
- クライアントサイドでバリバリ書けないので、なるべく定義は共有側に流れていく。これは正しい実装に近づいているように感じる

まとめ

- **BLAZOR**は**C#**の**SPA**フレームワーク
- サーバーからクライアントまで全て使い慣れた **VS+.NET+C#**で作れてスイッチングコストがない！
- 作ったものが現世代ブラウザでちゃんと動く！

アジェンダ

- BLAZOR
- IDENTITY
- BLAZOR × IDENTITY

認証

- 認証って大事です。よね？

認証

- サイトを作って、その中でユーザーの情報を表示したい場合、認証は必須
- 作ったサイトを育てていく場合、ユーザー情報の表示は、かなり重要。例えば履歴とか。

認証

- 認証で扱う情報は個人情報。つまりセキュリティも重要。もし漏洩したらそれこそ金銭問題にまでつながる
- 例えばパスワード一つとっても、そのまま持つべきではない。漏洩対策としてハッシュに変換してもつ

認証

- 大事だけどぶっちゃけ、エンジニアとしては手間はかけたくない
- 最低限、登録とログイン画面だけあれば、よくね？と思わないでもない気がする今日この頃

認証の結論

- 認証を自分で作るならフレームワークを使う！

ASP.NET IDENTITY

- マンガ読んだ!!は最初、**ASP.NET MVC**で作成
- **ASP.NET MVC**は**ASP.NET IDENTITY**がある！

ASP.NET IDENTITY

- DEMO

ASP.NET IDENTITY

- さて、登録とログインだけあれば良いと言ったけれど、**VIEW**を見てみるとパッとみ結構ファイルが一杯ある

ASP.NET IDENTITY

- でも、実際使ってみると、これでもわりとミニマム。
- パスワード忘れ、パスワード忘れの確認、パスワードリセット、パスワードリセット確認、Eメール確認、外部認証、2段階認証など使用していくなら必須のものばかり

ASP.NET IDENTITY

- わりと重要なのがメール。メール機能は認証ではパスワード忘れにも使える。
- 登録時に、メルアドにURLを送って、そのURLをクリックして認証完了させるしくみもASP.NET IDENTITYにある。
- この仕組みは、ユーザーには一手間だが、生きているメールだけに限定出来るし、いたずらに沢山作られる対策にもなる
- 1個だけ残念だったのが、「メールを投げました。確認ください」という画面がない。ASP.NET IDENTITYで唯一自分で追加で作った

ASP.NET IDENTITY

- あと認証に関して言えばやっぱりSNS認証には対応したい。パスワード打たなくて良いのは魅力。
- これもASP.NET IDENTITYがやってくれる

ASP.NET IDENTITYの弱点

- 僕が以前作った時はVS2017だったが、機械翻訳であってもちゃんと日本語リソースがあった。
- VS2019で作った場合は日本語リソースがない。これはキツイ。。

まとめ

- 認証は大事
- 認証はフレームワークを使う！
- **ASP.NET MVC**はわりと手軽
- **VS2019**からは日本語リソースがなくなった。。

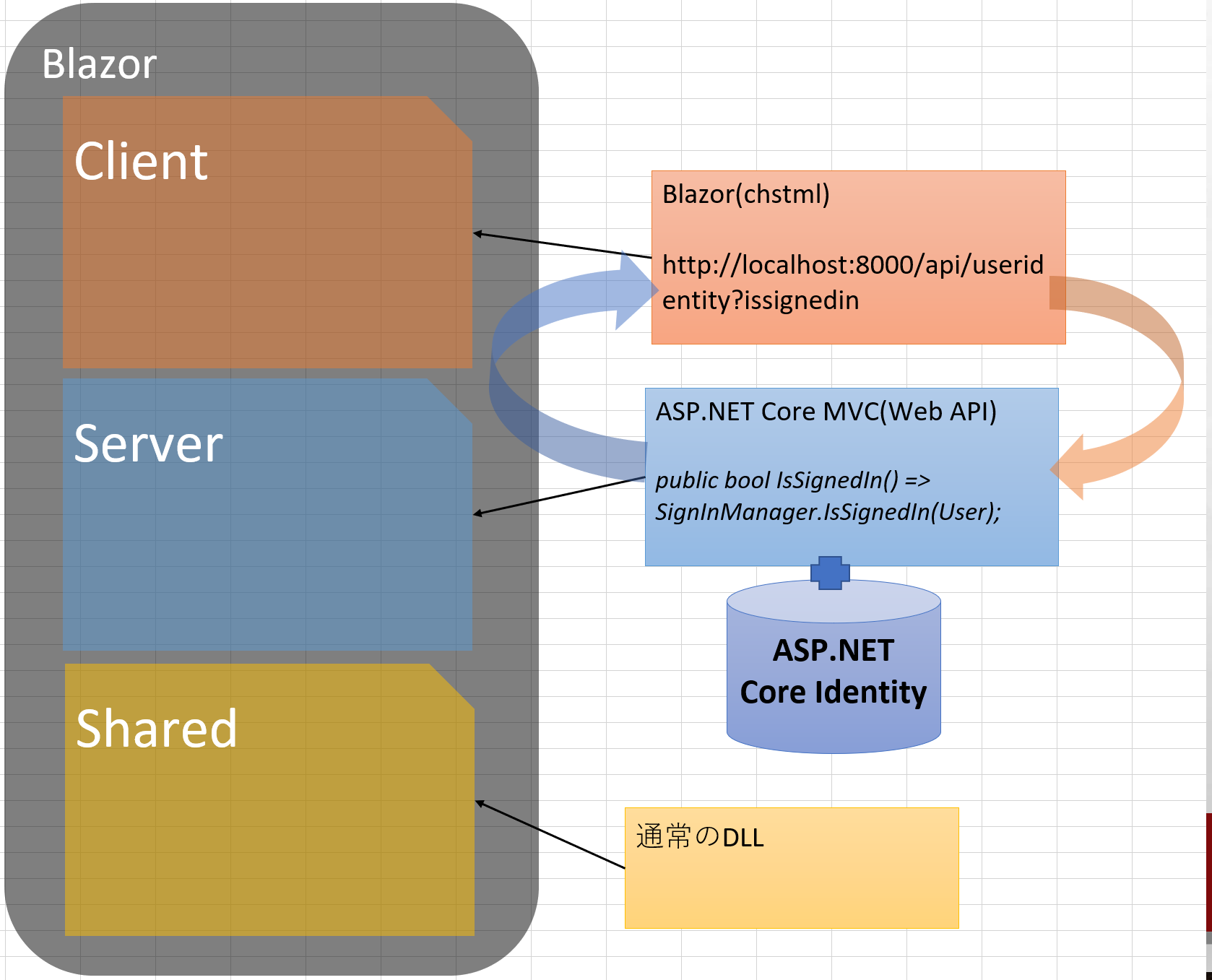
アジェンダ

- BLAZOR
- IDENTITY
- BLAZOR × IDENTITY

BLAZORの認証の作り方

- BLAZORのサーバーサイドはASP.NET CORE MVCのWEB API
- ASP.NET CORE MVCにもASP.NET CORE IDENTITY がある
- IDENTITYの結果をWEB APIに渡してBLAZORで書いてやればよい

MER


Blazor
Client
Server
Shared
Blazor(chstml)
http://localhost:8000/api/useridentity?issignedin
ASP.NET Core MVC(Web API)
public bool IsSignedIn() =>
SignInManager.IsSignedIn(User);
ASP.NET
Core Identity
通常のDLL

ASP.NET CORE IDENTITY

- **ASP.NET IDENTITY → ASP.NET CORE IDENTITY**
- **MSDN**に細かく書いてある。最近の**MSDN**は何とか読めるレベルの変な日本語の記事が増えた
- まあ、読めなくはない

ASP.NET CORE IDENTITY

- 基本的にはCOREで作って、以前の設定に合わせる部分を合わせるだけ。実際STARTUP.CSをいじるぐらいしかしてない？例えばデフォルトだとパスワードに記号が入るので、それをやめるとか
- あとDBのカラムの増減があるのでそれを対応すれば動く

ASP.NET CORE IDENTITY

- COREはバージョンがポイント。ASP.NET CORE 2.0の場合、ASP.NET IDENTITYとASP.NET CORE IDENTITYに大差ない
- 2.1以降は大差ある

ASP.NET CORE IDENTITY

- 2.1でプロジェクト作ると面食らう
- IEDNTITYのソースが全部なくなっている
- でも動いている。

```
public void ConfigureServices(IServiceCollection services)
{
    services.Configure<CookiePolicyOptions>(options =>
    {
        // This lambda determines whether user consent for non-essential cookies
        options.CheckConsentNeeded = context => true;
        options.MinimumSameSitePolicy = SameSiteMode.None;
    });

    services.AddDbContext<ApplicationDbContext>(options =>
        options.UseSqlServer(
            Configuration.GetConnectionString("DefaultConnection")));
    services.AddDefaultIdentity<IdentityUser>()
        .AddEntityFrameworkStores<ApplicationDbContext>();

    services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version_2_1);
}
```

ASP.NET CORE IDENTITY

- ただ、デフォルトでやっていくのは、特に日本ではほぼ無理
- だって画面全部英語だし。リソース分かれてないので、作るしか選択肢ない気が

BLAZOR ASP.NET CORE

- ローカライズ対応。**ASP.NET CORE**は一応ヘルパー的な仕組みがある
- あるが、やっぱり面倒。結局リソース用意しないと駄目。まあ頑張ってやった

ASP.NET CORE IDENTITY

- CONTROLLERとVIEWも大幅に変わった。
- 全てCSHTMLの中でCSのロジックが書けるようになった
- 新規に作るなら2.1以降の方が良いが、移植となると2.0で作った方が良いかも

BLAZOR認証

- BLAZORはVS2017ではASP.NET CORE2.1、VS2019ではASP.NET CORE 3.0でしか作れなかった。
- 2.2は少なくとも非対応。で、
- ASP.NET CORE 2.1 BLAZORに認証画面を乗せようとするとタグヘルパーがうまく動いてくれなかった

BLAZOR ASP.NET CORE

- BLAZORのASP.NET CORE 3.0で作ったBLAZORのプロジェクトとASP.NET CORE 2.0 MVCで作った認証ありのプロジェクトを切り貼りして構築した感じ。
- 因みにちゃんと追ってないけど、ASP.NET CORE 2.1の認証を持ってくるのは何度かやったがうまくいかなかった

BLAZOR ASP.NET CORE

Blazor/Identity	2.0	2.1	2.2	3.0	
2.0	×	×	×	×	←Blazorが非対応
2.1	×	×	×	×	←Blazorタグヘルパーが動かない
2.2	×	×	×	×	←Blazorが非対応
3.0	○	×	×	×	
		→AddIdentity指定しているとBlazorの画面に行かない			

Blazor

Client

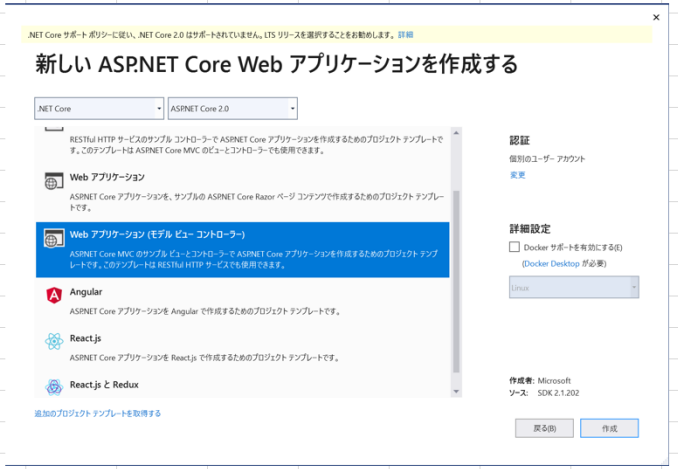
Server

Shared

Blazor(chstml)

ASP.NET Core 3.0 MVC(Web API)

通常のDLL



ASP.NET Core 2.0 MVC

ASP.NET Identity

ASP.NET CORE 3.0

- .NET CORE 3.0は今日現在まだPREVIEWがついている
- 今回動かすのにこのバージョンしかないという必然性があったので使った
- PREVIEWを使うのは何かと面倒。おすすめはしない。正式版まで待てるなら待った方が良いと思う
- 例えばエラー自体も洗練されてなく時間がかかることもある

ASP.NET CORE 3.0

- でも進化はとても感じる。実際2.1から使っている身としては安定してきているのは凄く感じる
- でもPREVIEW4で拡張子まで変わるとか、結構ドラスティックに変わる
- PREVIEW6入れたらコンパイル通らず、色々見れなかったのでアンインストールしたら、それ以降結構おかしい。。。。

BLAZOR認証

- **BLAZOR**側にも対応が必要
- **ASP.NET WEBAPI**でログイン情報を取得できるように
- 複数画面で共有して値を使いたい場合は**CASCADINGVALUE**を使う

BLAZOR認証

- **BLAZOR**の画面遷移で普通に作るとサーバー側の**URL**パラメータを打っても画面遷移してくれない

BLAZOR認証　THANKS TO JSAKAMOTO

- **A**タグの**TARGET**に_**TOP**と打てば画面遷移してくれる。

- <A CLASS="NAV-LINK" HREF="ACCOUNT/LOGIN" TARGET="_TOP">ログイン</A>

まとめ

- BLAZORで認証を乗せたかったらASP.NET CORE IDENTITYを使う
- ASP.NET CORE 2.0 IDENTITY + ASP.NET CORE 3.0 BLAZORの組み合わせだけ動作実績あり
- BLAZOR側でも幾つか対応が必要

最後に

- **WEB**開発には**BLAZOR**を使って**C#**で書こう**!**
- 認証にはフレームワークを使おう**!**
- マンガの記録には**マンガ読んだ‼**を使おう**‼**
- マンガの記録には**マンガ読んだ‼**を使おう**‼**