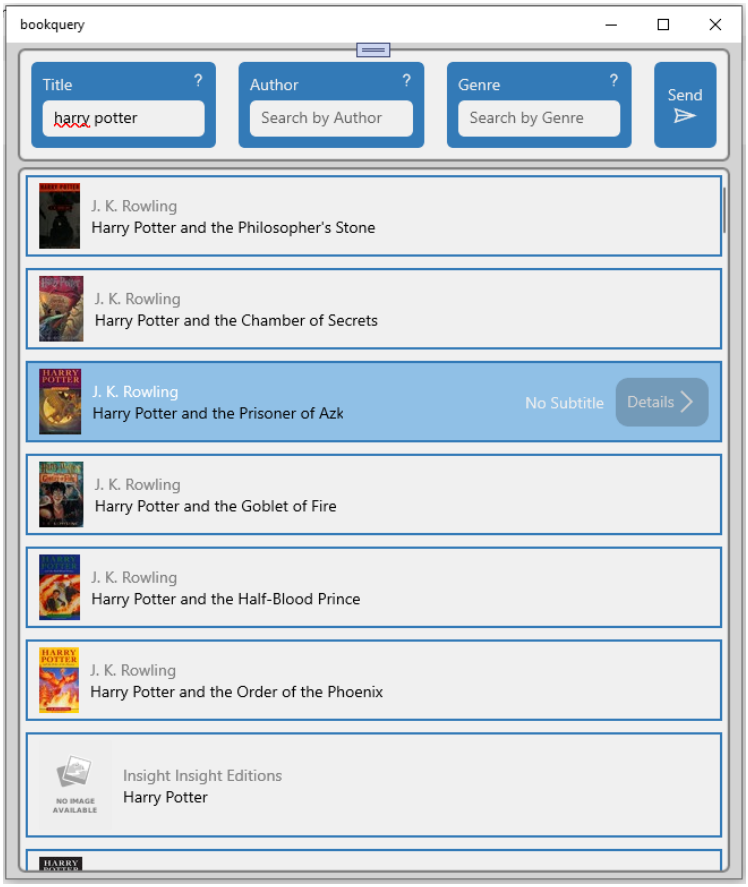
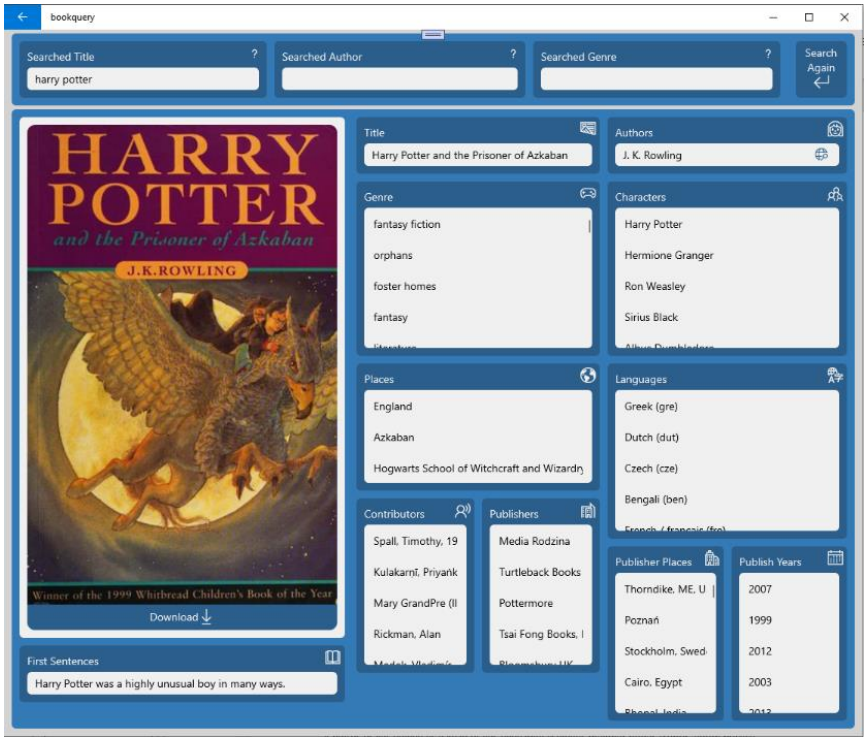
Könyvkereső kliens

Kliensoldali technológiák házi feladat – Készítette: Markovics Gergely (ME7KKE)



Összefoglaló

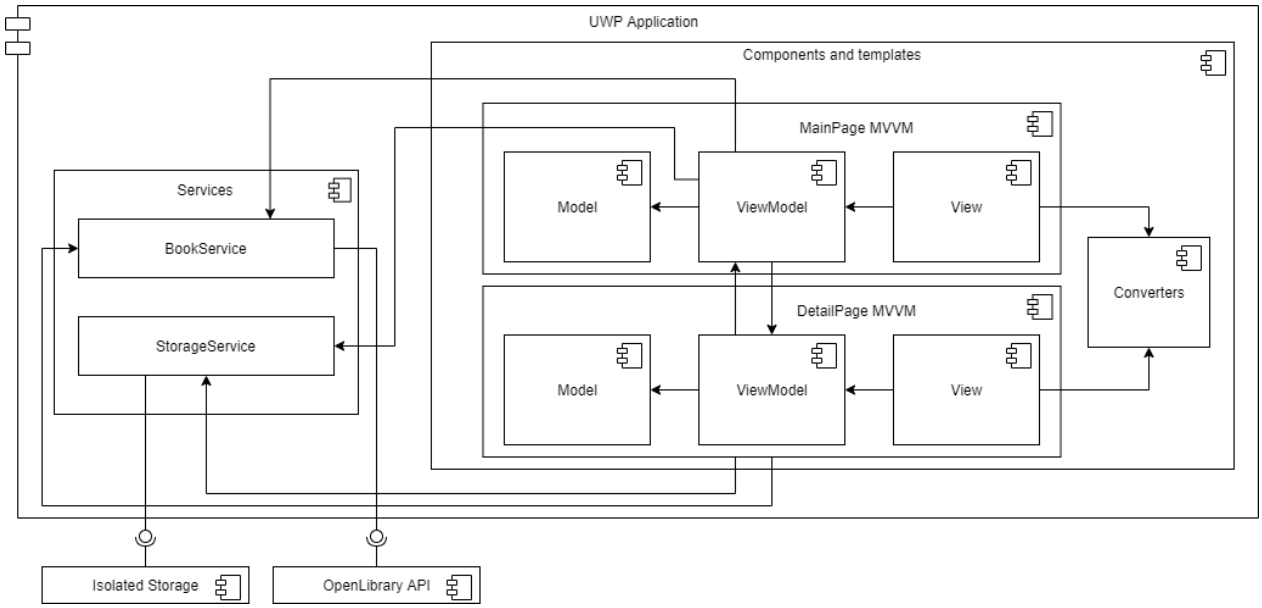
A könyvkereső kliens alkalmazás lehetővé teszi, hogy a felhasználó kulcsszavak megadásával keressen rá könyvekre. Az alkalmazás a kulcsszavak megadására kilistázza a találatokat, azoknak egy kisméretű borítóképét, szerzőjét és címét. Ha nincs megadva a könyvnek borítókép, akkor a wikipédia placeholder linken található kép jelenik meg. Kattintásra megjelenik a könyv alcíme is, vagy ha nincs „No Subtitle” felirat. Az alcím mellett megjelenik egy gomb is, ami az adott könyv részletes adatainak megjelenítésére szolgáló nézetre visz minket, és a szerző neve is fehérre változik szürkéről, a jobb olvashatóság kedvéért. 3 paraméterrel lehet keresni az alkalmazásban, cím, szerző és műfaj, mindhármat lehet egyszerre használni, de csak 3 karakter felett veszi figyelembe. A gombot is akkor lehet megnyomni, ha már legalább egyik 3 karakter felett van.



A kiválasztott elem melletti Details gombra kattintva lehet jutni a részletes nézetre, ahol a könyvnek többféle adata megjelenik.

Architektúra

Az alkalmazást C# UWP keretrendszerrel valósítottam meg, ami az OpenLibrary API-ról szerzi a szükséges adatokat. A rendszer felépítése és az elemek főbb kapcsolódási pontjai a következők:



Komponensek

- **MainPage MVVM**: Fő kereső felületet megvalósító nézet és háttérlogika, itt lehet keresni könyveket a megadott paraméterekkel.
- **DetailPage MVVM**: Részletes felületet megvalósító nézet és háttérlogika, itt látható megadott könyvnek a részletes adatai.
- **Converters**: Nézet megjelenítését segítő változó átalakítók, például logikai értékhez szín rendelése.

Szolgáltatások

- **BookService**: OpenLibrary-val való kommunikáció megvalósítása, könyvfejléc, könyv részletes adatai, szerző adatai vagy nyelvek lekérdezése.
- **StorageService**: Fájlműveleteket megvalósító szolgáltatás, tehát kép lementése és Isolated Storage-ből való olvasás.

Fontosabb komponensek

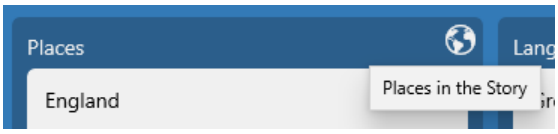
Az alkalmazás működése szempontjából két központi komponens a legfontosabb

MainPage MVVM

Központi felület, alkalmazás indítása is ide navigál. Indításkor azzal kezdi, hogy megpróbálja az Isolated Storage-ből olvasni, hogy benne van-e az utoljára lementett keresés paraméterei. Ha benne van, akkor rögtön indít egy keresést azokkal az értékekkel. Az Send gomb mögött egy delegateCommand van, ami érzékeli, hogy a bemenetekben van-e valahol legalább 3 hosszú bemenet. A felhasználót segítik a mezők jobb felső sarkában elhelyezett fonticonok is, amik alatt tooltipben megtalálható, hogy „At least 3 characters”. Ha megérkezett a könyvek listája, akkor a listaelemekre a felhasználó rá tud kattintani, amivel beállítja a model hozzá tartozó „isClicked” tagváltozó értékét, amire reagál a converterek segítségével a nézeten megtalálható gomb megjelenítése is, ha a könyvnek létezik ISBN kódja. A megjelent gombra kattintva átnavigál minket a részletes nézetre, átküldve a könyv egyik ISBN kódját, mivel az egyértelműen beazonosítja.

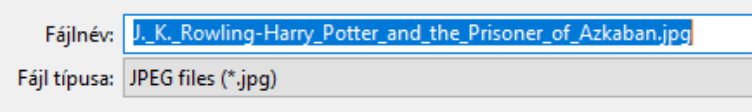
DetailPage MVVM

Ezen a felületen átnavigáláskor azzal kezdi, hogy fogadja a keresendő könyv isbn számát, és kozmetikumnak a keresett paraméterket, amiket megjelenít ugyanúgy a felső sávban. Ezután a szolgáltatás segítségével lekérdezi az isbn-hez tartozó könyvet, aminek a modelbe rakott adatait betölt a nézet megfelelő helyeire. Mivel a nyelvek csak rövidítéseket tartalmaznak, ezért utána még lekérdezi az API-tól azoknak a teljesen kiírt nevét is, és aztán azt betölti a Languages fül alá „teljes (rövidített)" alakban. Eredetileg a szerző linkjét se tartalmazza a lekérdezés, ezért azt is külön töltjük be, és ha van találat, akkor megjelenítjük a szerző neve mellett egy hozzátartozó ikon gombbban. Tooltip megmondja azt is, hogy mi az url címe, mielőtt rányomnánk.



Nagyrészt redundáns adatokkal, de a Tooltip segítségek minden ikon alatt megtalálható a jobb felső sarkokban, vagy a szerző neve alatt láthatjuk neki az alternatív neveit.

Mégegy fontos funkció, hogy a borítóképet le is lehet menteni a Download gombhoz kötött fájlkezelő szolgáltatás segítségével. A kép Jpeg lesz, és az alapértelmezett neve a szerző és cím párosból áll.



Persze, ha nincs borító, hanem az alapértelmezett kép van betöltve, akkor a gomb is le van tiltva.

OpenLibrary API

OpenLibrary felületéről elég sok route-on lehet sokféleképpen lekérdezni, de arra, hogy ezzel a 3 paraméterrel közösen tudjunk lekérdezni, az alap keresőjét megvalósító /search felületre volt szükség. Itt a title, subject, author és isbn query paramétereket használtuk, első hármat a listakeresőben, az utolsót a részletezőben. Mivel a lekérdezés a listát egy docs alváltozóban adja vissza, ezért bevezettem egy segédosztályt, amibe belerakja, majd abból másolom át egy önálló listába.

```
public class BookHeaderGroup
{
    public ObservableCollection<BookHeader> Docs { get; set; } =
        new ObservableCollection<BookHeader>();
}
```

Ugyanez a könyvek részletes modelljében.

A könyvek borítóképeit a könyv OLID azonosítója alapján találja meg az adott URL alapján. Ez alapján dönti el azt is, hogy ha nincs ilyen azonosító, akkor a placeholder képet tölti be a próbálkozás helyett.

```
/// <value>Cím, ami alatt a borítóképek találhatóak</value>
private readonly string coverBaseURL = "http://covers.openlibrary.org/b/olid/";
/// <value>Kis méretű kép kiterjesztése</value>
private readonly string coverSmallExtension = "-S.jpg";
/// <value>Nagy méretű kép kiterjesztése</value>
private readonly string coverLargeExtension = "-L.jpg";
```

A kétféle részletesség mellett még 2 módszerrel kérdezünk le a kiegészítő információk miatt.
Egyszer szükségünk van a nyelvek teljes neveire, amit a /languages route alatt tudunk lekérdezni, és beolvassuk egy segédosztályba, amit majd hozzáadunk a részletes könyv objektumunkhoz.

```
class Language
{
    public string Name { get; set; } = "";
    public string Code { get; set; } = "";
}
```

Másik kiegészítő információ a szerzőhöz tartozó link, amit a /authors route-on találunk meg, és ugyanúgy egy segédosztállyal kérdezünk le. Ha talált ilyen linket, akkor a szerző neve mellett megjelenik egy webes ikon, amire kattintva átnavigálhatunk a szerző weblapjára.

Storage

Kétféle fájlműveletet végzünk, egyik az Isolated Storage-be írás/olvasás, másik savepicker használatával borítókép lementése.

Borítókép lementésénél kap a szolgáltatás egy képet, és egy alapértelmezett fájlnevet, amit a könyv szerzőjéből és címéből alakít ki. A borítóképet betöltjük egy Bitmapbe, amit majd egy bitmapencoder belerak a kiválasztott fájlunkba. A savepicker alapértelmezett útvonala a „Letöltések” mappába vezet, úgy gondoltam az logikus, és megkapja alapértelmezett névnek a paraméterben megadott szöveget Jpeg formátumban.

```
//Mentsük alapértelmezetten a Letöltésekbe
FileSavePicker fileSavePicker = new FileSavePicker();
fileSavePicker.SuggestedStartLocation = PickerLocationId.Downloads;
fileSavePicker.FileTypeChoices.Add("JPEG files", new List<string>() { ".jpg" });
// Alapértelmezett neve legyen a fájlnak a könyv szerzője és címe
fileSavePicker.SuggestedFileName = filename;
```

Másik fájlművelet az Isolated Storage használata, ez igazából egy kozmetikum, sok jelentősége nincs a funkciónak. Az alkalmazás elindításakor lekérdezzük tőle, hogy van-e egy „SearchableStore.txt” nevű fájl benne, és ha van, akkor a 3 paramétert kiolvassuk belőle, amik „;” jellel vannak elválasztva egymástól. Ezután az alkalmazás futása közben, ha elindítunk egy keresést, akkor annak a paramétereivel felülírja, ami a fájlban van, hogy amikor újra elindítjuk akkor onnan kiolvassuk.

```
string resp = reader.ReadToEnd().Replace("\n", "").Replace("\r", "");
string[] splitted = resp.Split(';');
response.AddRange(splitted);
```

Források

A képek a https://www.diagrams.net/ segítségével készültek.