Suplement do wykładów z IPM - studia zaoczne, prowadzący Paweł Fiderek.

Tworzenie i startowanie wątku w aplikacjach Windows Store:

```
IAsyncAction asyncAction = Windows.System.Threading.ThreadPool.RunAsync((workItem) => nazwaMetody());
```

Kroki:

Inicjalizujemy obiekt typu IAsyncAction;

Wywołujemy metodę RunAsync z klasy ThreadPool;

W parametrze wywołania "rzutujemy" nazwę metody którą za pomocą wyrażenia lambda na typ workItem;

Nadawanie praw modyfikacji kontrolki przez wątek inny niż ten w którym została stworzona:

```
await Dispatcher.RunAsync(CoreDispatcherPriority.Normal, () => this.nazwaKontrolki.Metoda());
```

Należy pamiętać że metoda w której wywoływany jest powyższy kod powinna być udekorowane słówkiem kluczowym async!

SqLite w aplikacjach Windows Store:

Kolejność instalacji narzędzi:

-Pobieramy dedykowaną bibliotekę: Tools -> Extensions and Updates -> Online -> find sqlite for Windows Runtime

W projekcie dodajemy 2 referencje:

1: SqLite for Windows Runtime;

2: VC++ Runtime Package;

UWAGA należy pamiętać o braku wsparcia dla kompilacji Any CPU!

Doinstalowujemy paczkę sql.net: Tools -> Library Package Manager -> Manage NuGet Packages for Solution -> sql.net

Co robimy w kodzie:

Krok 1:

Tworzymy klasę do obsługi bazy danych:

Definiujemy pola w tabeli:

```
[SQLite.PrimaryKey, AutoIncrement]
        public int Id { get; set; }
        [MaxLength(30)]
        public string Name { get; set; }
        [MaxLength(30)]
        public string Surname { get; set; }
        [MaxLength(30)]
        public string PlaceOfBirth { get; set; }
        [MaxLength(6)]
        public string Sex { get; set; }
        [MaxLength(3)]
        public int Age { get; set; }
```

Krok 2 - Tworzymy bazę i ustawiamy do niej połączenie:

```
  public async void CreateDatabase()
        {
            bool ifExist = false;
            try
            {
                var dbFile = await Windows.Storage.
                    ApplicationData.Current.LocalFolder.
                    GetFileAsync("bazaUzytkownikow");
                ifExist = true;
            }
            catch (Exception)
            {
                ifExist = false;
            }

            if (!ifExist)
            {
                SQLiteAsyncConnection conn = new
SQLiteAsyncConnection("bazaUzytkownikow");//nie koniecznie
                await conn.CreateTableAsync<UserFile>();
            }
            else
            {
                SQLiteAsyncConnection conn = new
SQLiteAsyncConnection("bazaUzytkownikow"); //nie koniecznie
            }

        }
```

Krok 3 - definiujemy metody współpracujące z bazą danych, np. dodawanie użytkownika lub szukanie użytkownika po wartości pola w tabeli:

```
  public async void addUser(string name, string surname, string placeOfBirth, string
sex, int age)
        {
            SQLiteAsyncConnection conn = new
SQLiteAsyncConnection("bazaUzytkownikow");

            UserFile user = new UserFile
            {
                Name = name,
                Surname = surname,
                PlaceOfBirth = placeOfBirth,
                Sex=sex,
                Age=age
            };

            await conn.InsertAsync(user);
        }
```

```
        public async void searchUser(string condition, string value)
        {
            int val;
            var result = new List<UserFile>();
            SQLiteAsyncConnection conn = new
SQLiteAsyncConnection("bazaUzytkownikow");
            if (condition == "Age")
            {
                val = Int32.Parse(value);
                var query = conn.Table<UserFile>().Where(x => x.Age == val);
                result = await query.ToListAsync();
            }
            else
            {
                if (condition == "Name")
                {
                    var query = conn.Table<UserFile>().Where(x => x.Name == value);
                    result = await query.ToListAsync();
                }
                else
                {
                    if (condition == "Surname")
                    {
                        var query = conn.Table<UserFile>().Where(x => x.Surname ==
value);
                        result = await query.ToListAsync();
                    }
                    else
                    {
                        if (condition == "PlaceOfBirth")
                        {
                            var query = conn.Table<UserFile>().Where(x =>
x.PlaceOfBirth == value);
                            result = await query.ToListAsync();
                        }
                        else
                        {
                            if (condition == "PlaceOfBirth")
                            {
                                var query = conn.Table<UserFile>().Where(x => x.Sex ==
value);
                                result = await query.ToListAsync();
                            }
                            else
                            {
                                val = Int32.Parse(value);
                                var query = conn.Table<UserFile>().Where(x => x.Id ==
val);
                                result = await query.ToListAsync();
                            }
                        }
                    }
                }
            }
```