WPF

Microsoft MVP

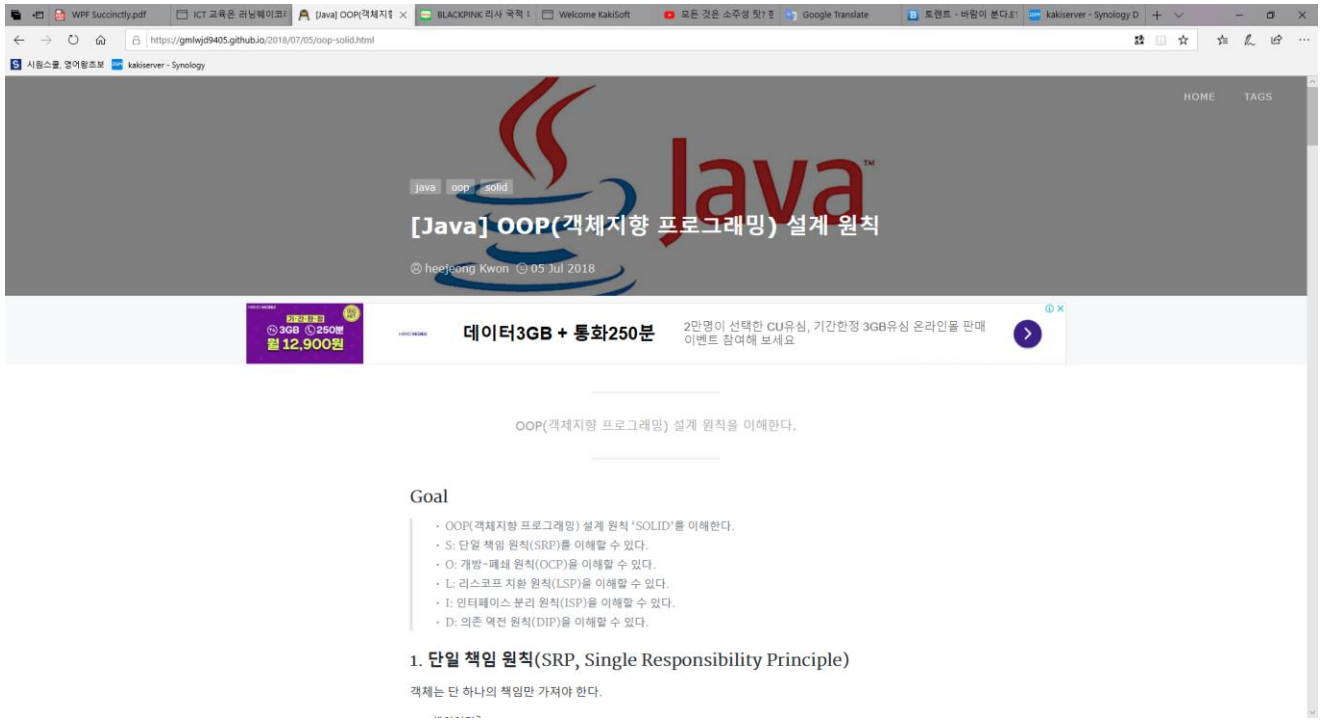
Connor Park

Day 3

WPF deep-dive 2

SOLID Principle

- Single Responsibility Principle
  - 단일 책임 원칙(SRP)
- Open Closed Principle
  - 개방-폐쇄의 원칙(OCP)
- Liskov Substitution Principle
  - 리스코프 치환 원칙(LSP)
- Interface Segregation Principle
  - 인터페이스 분리 원칙(ISP)
- Dependency Inversion Principle
  - 의존 역전 원칙(DIP)



List vs ObservableCollection

- IList<T>
- 메모리
- 성능

Id	Name	Age	Address	HasMarried
1	지수	23	주소1	False
2	제니	23	주소2	True
3	로제	21	주소3	False
4	리사	22	주소4	True

Id	Name	Age	Address	HasMarried
1	지수	23	주소1	False
2	제니	23	주소2	True
5	통키	10	주소5	False
3	로제	21	주소3	False
4	리사	22	주소4	True

Id	Name	Age	Address	HasMarried
2	제니	23	주소2	True
5	통키	10	주소5	False
3	로제	21	주소3	False
4	리사	22	주소4	True

Member 목록

Id	Name	Age	Address	Married
1	지수	23	주소1	미혼
2	제니	23	주소2	기혼
3	로제	21	주소3	미혼
4	리사	22	주소4	기혼

Member 상세

신규 삭제

Id TextBox

Name TextBox

Age TextBox

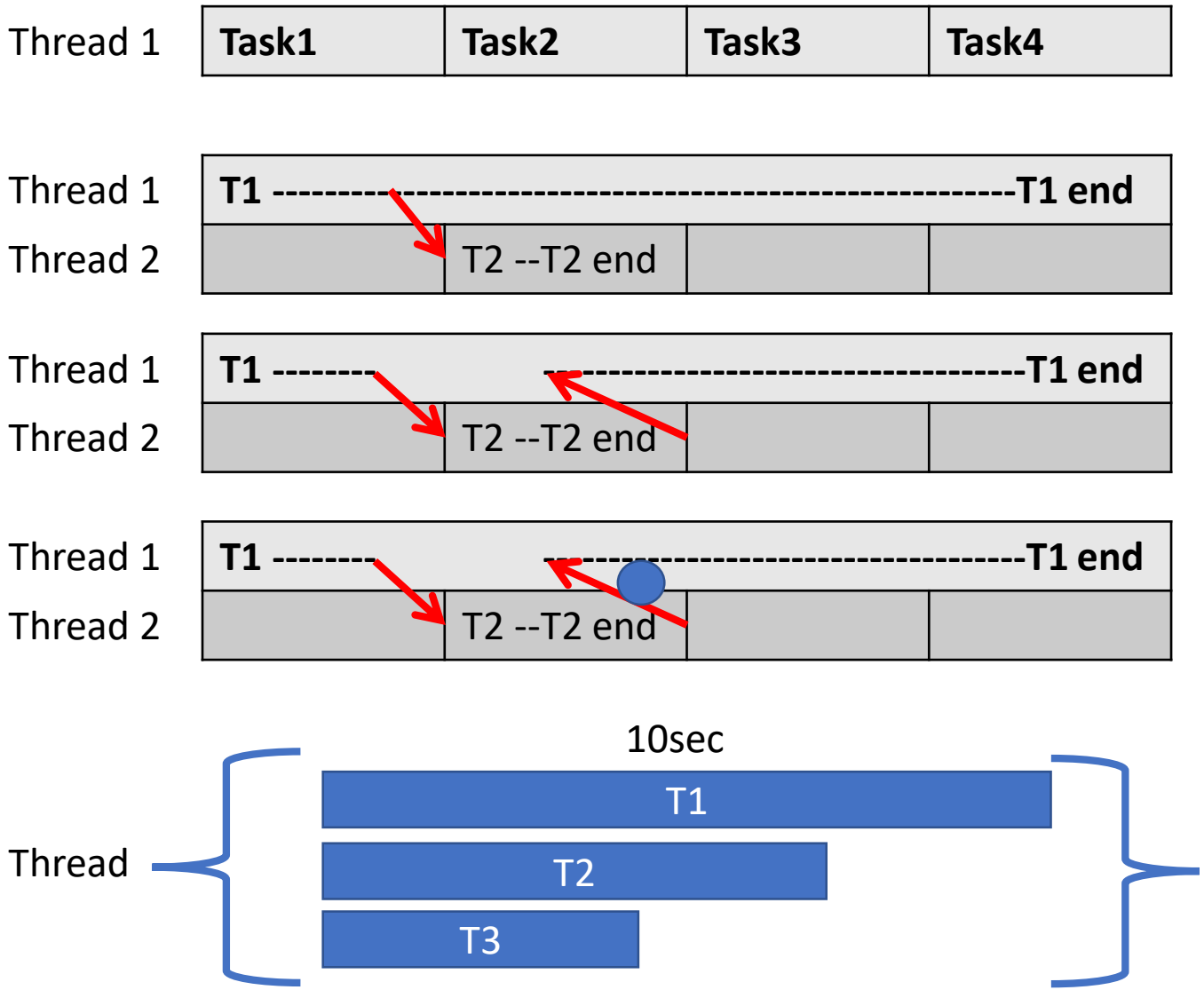
Address TextBox

Married CheckBox

- ObservableCollectionPage.xaml 추가
- ListView 추가
- Member 상세 구현
- 버튼 2개 추가
- People 연결
- Member 목록에서 선택 하면 Member 상세에 선택한 데이터 출력, 수정
- 신규 버튼 클릭시 상세 내용 클리어 및 Member 맨 하단에 한줄 추가
- 삭제 버튼 클릭시 해당 데이터 삭제

Synchronous vs Asynchronous

- 동기
- 비동기
  - Async void
  - Async Task
  - Async Task<T>
- 활용
  - Task.WhenAll()
- UI Freeze



Task1

T1 Start
T1 End
T2 Start
T2 End
T3 Start
T3 End

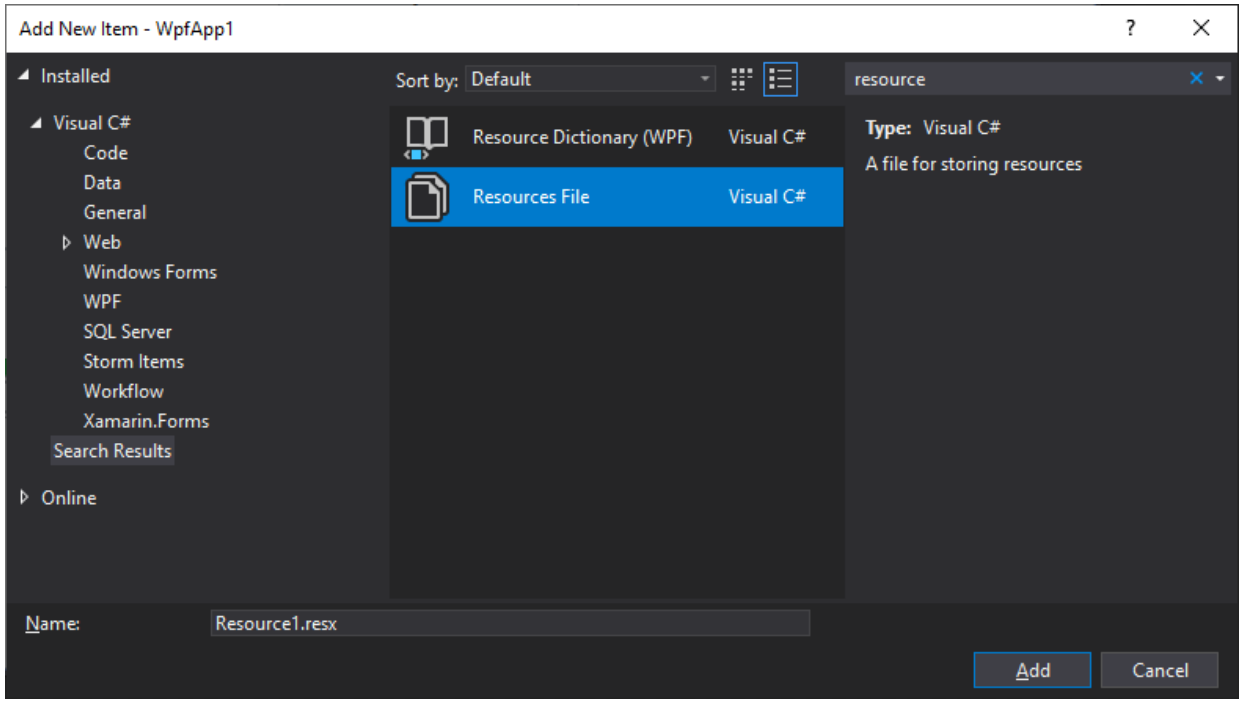
Task2

T1 Start
T2 Start
T3 Start
T2 End
T3 End
T1 End

- TaskPage.xaml 추가
- ListBox 추가
- Task 3개 생성
- Task.Delay()를 이용
- T1 : 10초
- T2 : 3초
- T3 : 5초
- 화면 처럼 결과가 나오게 만듭니다.

Resource

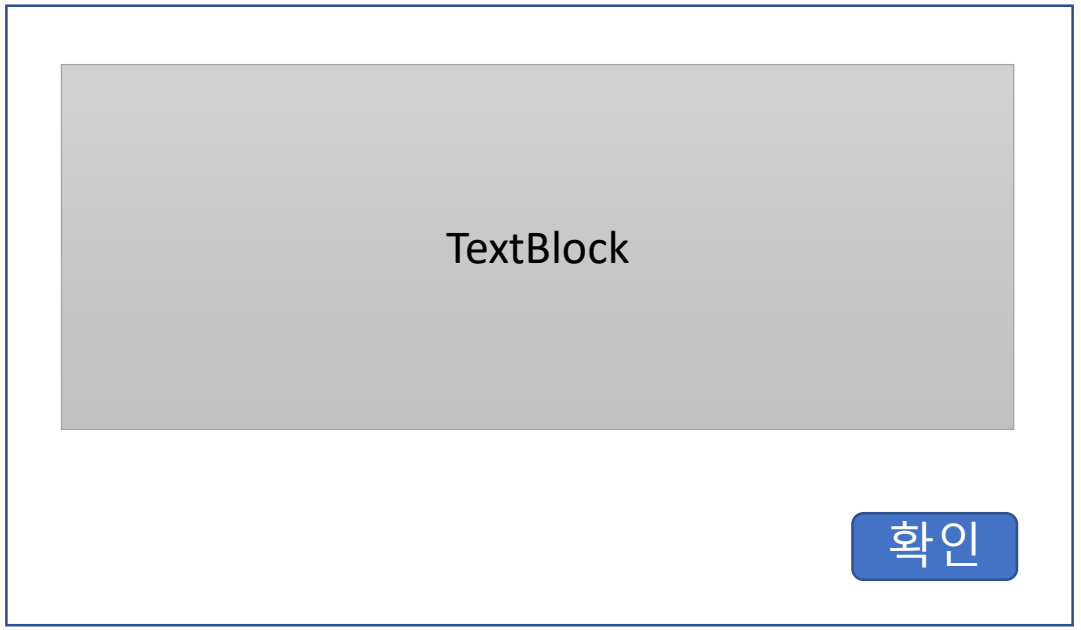
- 응용 프로그램의 여러 위치에서 재사용 가능한 객체
- 종류
  - Resource file
    - string
  - ResourceDictionary
    - Style, DataTemplate, …
- 사용
  - StaticResource
  - DynamicResource

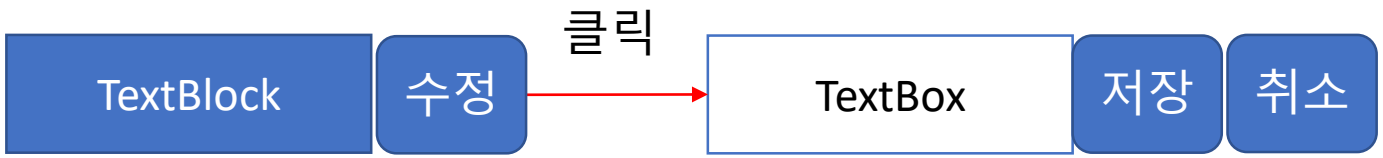



```
<SolidColorBrush x:Key="DefaultText" Color="Red"/>

 <TextBlock Text="hello textblcok"
                    Foreground="{StaticResource
DefaultText}"
                    Background="{DynamicResource
DefaultBackground}"/>

<SolidColorBrush x:Key="DefaultBackground"
Color="“Green"/>
```

UserControl vs CustomControl

- 사용자 컨트롤을 만듬
- UserControl
  - 여러개의 컨트롤을 추가해서 쉽게 만들 수 있음
  - Xaml과 Code가 함께있음
  - 메모리 점유율이 높고, 성능 저하 가능
- CustomControl
  - 메모리와 성능 최적화 가능
  - 최소한의 컨트롤을 이용해서 만들어야 해서 난이도 상승
  - Xaml과 Code가 분리 되어있음





TextBoxEx

- ConfirmUserControl.xaml 추가
- 타이틀과 메시지를 전달하면 출력
- 확인 버튼을 누르면 True, 취소 버튼을 누르면 False 값을 반환
- TextBoxEx Custom Control 추가
- TextBox에 포커스시에 자동으로 전체 선택 기능 추가
- 엔터키 입력시 Command를 실행할 수 있는 기능 추가

MvvmLight

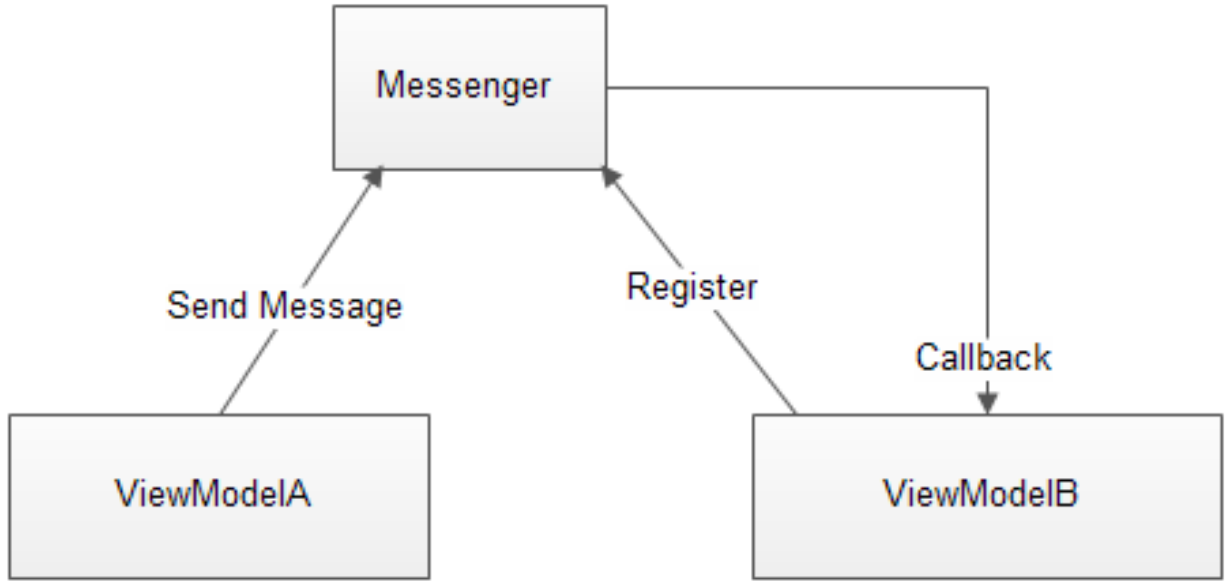
Navigation

- INavigationService
- Frame

```
public class SimpleNavigationService : INavigationService
    {
        private readonly Frame _frame;

        public SimpleNavigationService(Frame frame)
        {
            _frame = frame;
            _frame.Navigated += _frame_Navigated;
        }

        private void _frame_Navigated(object sender,
System.Windows.Navigation.NavigationEventArgs e)
        {
        }

        public string CurrentPageKey { get; private set; }

        public void GoBack()
        {
            if (_frame.CanGoBack == false) return;
            _frame.GoBack();
        }

        public void NavigateTo(string pageKey)
        {
            NavigateTo(pageKey, null);
        }

        public void NavigateTo(string pageKey, object parameter)
        {
            CurrentPageKey = pageKey;
            _frame.Navigate(new Uri($"/Views/{pageKey}.xaml", UriKind.Relative),
parameter);
        }
    }
```

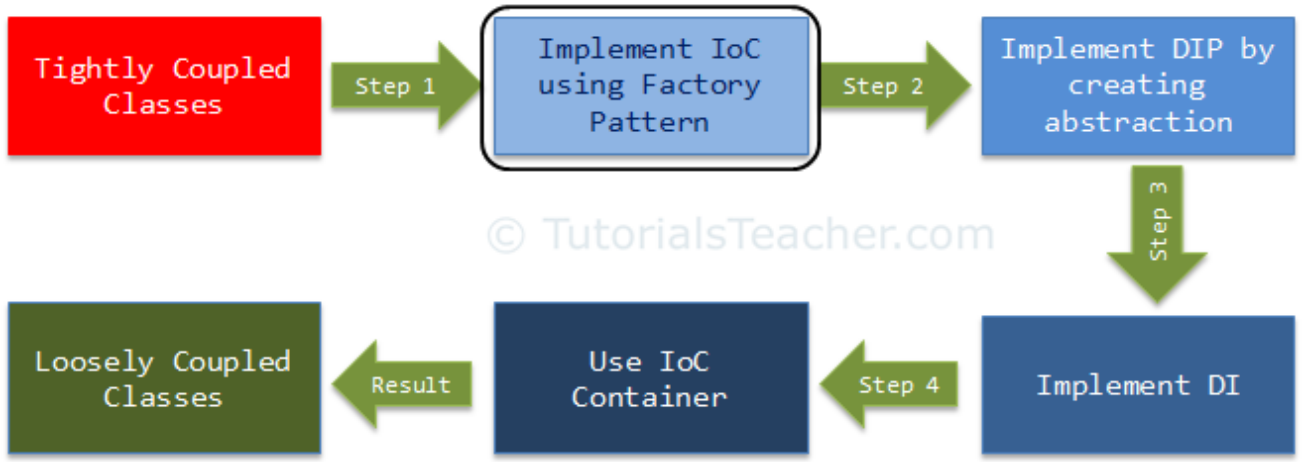
MessengerInstance

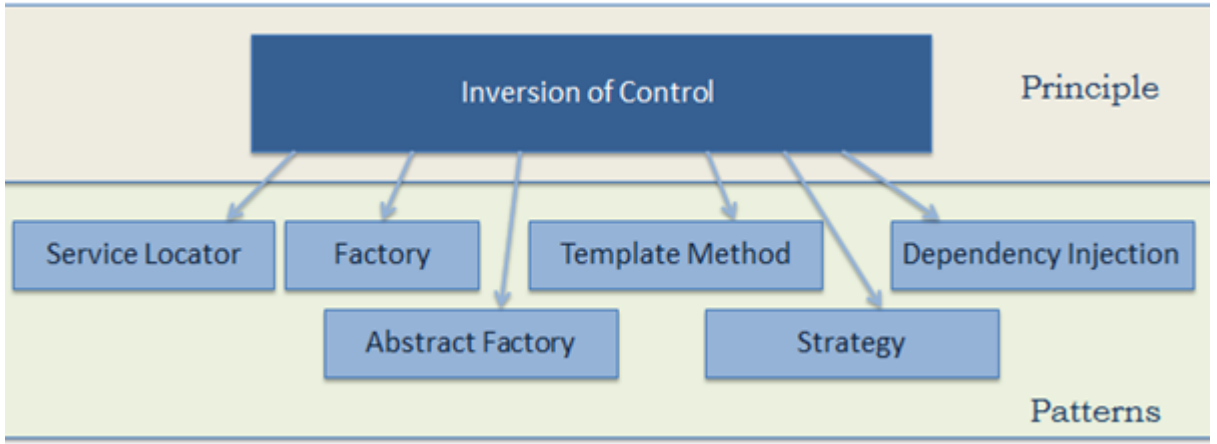
- 주로 뷰모델간에 메시지를 전송
- Default
- Reset
- Register<TMessage>()
- Unregister<TMessage>()
- Send<TMessage>()
- Cleanup()



SimpleIoc

- 싱글톤 방식의 단점
  - Public static 속성을 사용해야지만 생성
  - 시작과 동시에 생성 불가
  - 인스턴스 삭제 불가
  - 특정 상황에서 메모리 누수 발생
- IoC
  - 인스턴스를 생성, 유지 행위가 클래스 내부에서 이루어지지 않고, 컨터이너에 위임
  - Dependency Injection, Dependency Inversion의 근간
  - 관리와 사용을 분리





https://www.tutorialsteacher.com/ioc/inversion-of-control

.NET Standard 2.0

.NET Standard 2.0 overview

- PCL의 상위 버전
- 프레임워크와 독립적인 통일된 라이브러리 구현
- .NET Framework 4.6.1 수준의 API 지원

.NET Standard	1.0	1.1	1.2	1.3	1.4	1.5	1.6	2.0
.NET Core	1.0	1.0	1.0	1.0	1.0	1.0	1.0	2.0
.NET Framework [1]	4.5	4.5	4.5.1	4.6	4.6.1	4.6.1 [2]	4.6.1 [2]	4.6.1 [2]
Mono	4.6	4.6	4.6	4.6	4.6	4.6	4.6	5.4
Xamarin.iOS	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.14
Xamarin.Mac	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.8
Xamarin.Android	7.0	7.0	7.0	7.0	7.0	7.0	7.0	8.0
Universal Windows Platform	10.0	10.0	10.0	10.0	10.0	10.0.16299	10.0.16299	10.0.16299
Unity	2018.1	2018.1	2018.1	2018.1	2018.1	2018.1	2018.1	2018.1

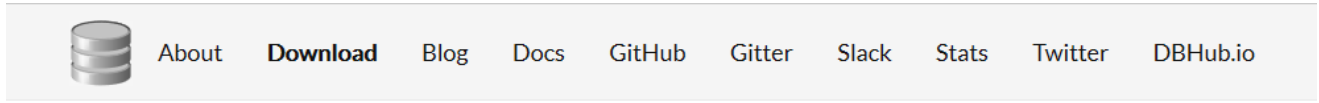
Database

SQLite overview

- 로컬에서 실행, 작고, 빠르며, 독립된 실행 환경
- SQL 문을 이용해서 조작 가능
- 오픈 소스 C 라이브러리
- 크로스 플랫폼 지원
- 시스템간의 콘텐츠를 임시로 저장하는 컨테이너로 활용
- DBMS Version 3.28.0 – 2019-04-16
- NuGet package로 제공
- Sqlite-net, Sqlite-net-pcl 패키지 사용

SQLite tool

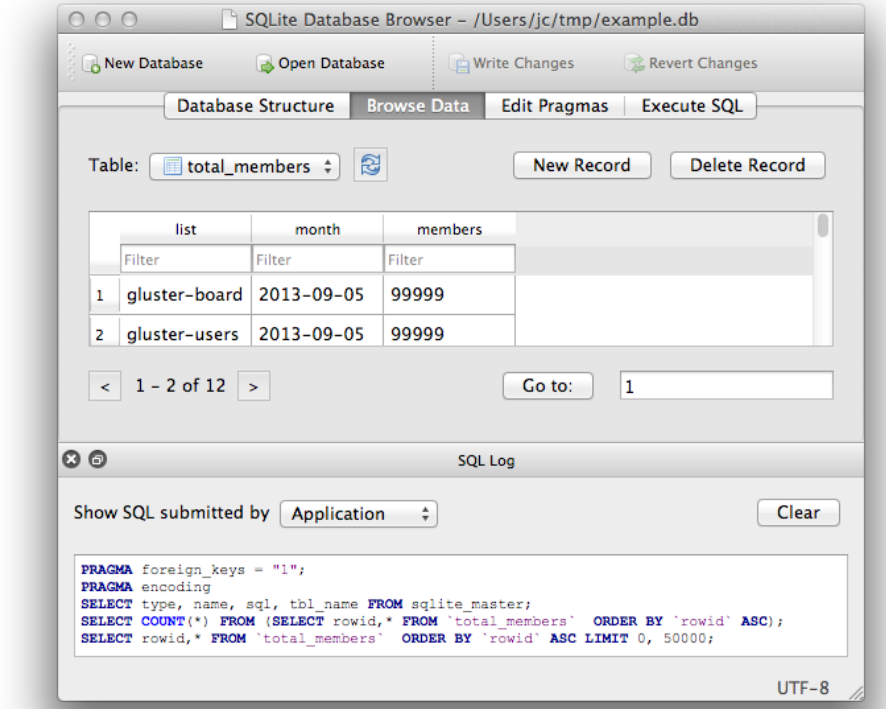
https://sqlitebrowser.org/



DB Browser for SQLite

The Official home of the DB Browser for SQLite

Screenshot



Person

```
using SQLite;


public class Person
{
    [PrimaryKey, AutoIncrement]
    public int Id { get; set; } = -1;

    public string Name { get; set; }

    public int Age { get; set; }

    public string Address { get; set; }

    public bool HasMarried { get; set; }
}
```

Member 목록

Get Add

Id	Name	Age	Address	Married
1	지수	23	주소1	미혼
2	제니	23	주소2	기혼
3	로제	21	주소3	미혼
4	리사	22	주소4	기혼

Member 상세

Save Delete

Id

TextBlock

Name

TextBox

Age

TextBox

Address

TextBox

Married

CheckBox

- SqlitePage.xaml 추가
- 해당 화면을 Sqlite database에 CRUD하도록 만듬
- 삭제 전에 확인 창을 출력해서 Yes이면 삭제, No이면 그냥 종료
- Add 버튼 클릭시 Member 상세에 입력 받을 수 있도록 만듬

Day 3 정리