**WPF MVVM project framework**

Common features:

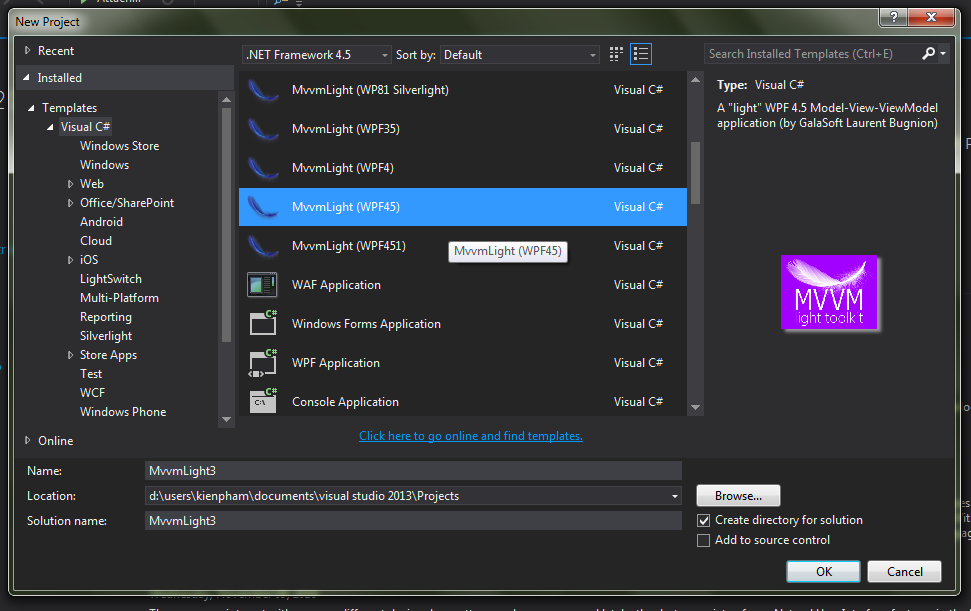
* ViewModelBase class (for the implementation of the INotifyPropertyChanged interface)
* RelayCommand like class to link UI command to ViewModel’s handlers
* Unit tests comes with the framework

[**Cinch**](http://sachabarber.net/?page_id=523)

* Author: [Sacha Barber](http://sachabarber.net/)
* Silverlight support: no
* Documentation: excellent, 6 articles published on CodeProject
* Hosting : [CodePlex](http://cinch.codeplex.com/)
* License:  Code Project Open License
* Features:
  + attached behaviors
  + validation suing IDataErrorInfo
  + support for IEditableObject
  + weak events creation and subscription
  + mediator messaging using weak events
  + IOC/DI support (using Unity)
  + services: event logger, message box, open save dialog, popup
  + threading helpers
  + support for menu items
  + closeable viewmodels
  + MVVM code generator

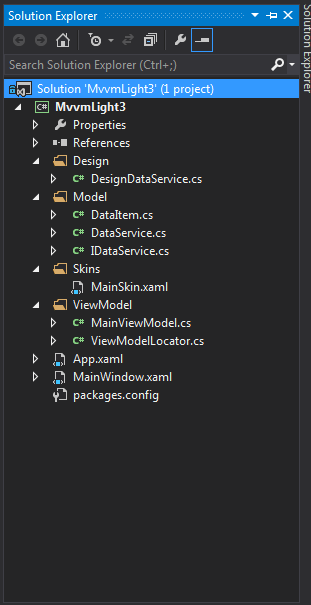
[**MVVM Light Toolkit**](http://www.galasoft.ch/mvvm/getstarted/)

* Author: [Laurent Bugnion](http://www.galasoft.ch/index.html)
* Silverlight support: yes
* Documentation: many articles available on Laurent’s blog + other developers as well
* Hosting: [CodePlex](http://mvvmlight.codeplex.com/)
* License: MIT license
* Features:
  + MSI installer
  + VS project and item template
  + VS code snippets
  + Messenger system for inter-viewmodel communication



Install MVVM Light ToolKit.

Create New Project -> MVVMLight (version NetFrameWork)

**Structure of Solution:**

+ Folder Design: configuration database connection.

+ Model: Create class model + Create Service (DataService, IDataService) for use connection in Design folder.

+ Skins: Create views (VIEW in MVVM)

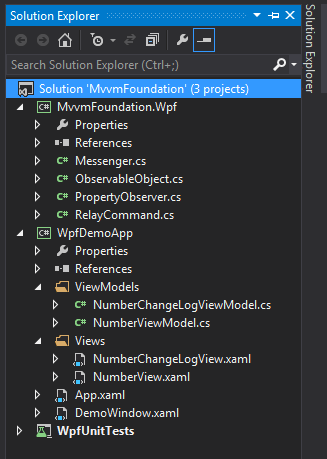
+ ViewModel: Creat class to control view in skins folder.

[**MVVM Foundation**](http://mvvmfoundation.codeplex.com/)

* Author: [Josh Smith](http://joshsmithonwpf.wordpress.com/)
* Silverlight support: no
* Documentation: articles about the Messenger implementation on Josh or Marlon Grech’s blog
* Hosting: [CodePlex](http://mvvmfoundation.codeplex.com/)
* License: MS-PL
* Features:
  + Messenger system for inter-viewmodel communication
  + PropertyChanged event monitor

Download from <https://mvvmfoundation.codeplex.com/> and use template to develop

**Structure of Solution**

Solution have 3 project: MvvmFoundation.Wpf, MvvmApp(WpfDemoApp), Mvvmtest(WpfUnitTests).

+ MvvmFoundation.Wpf : create Class of model (Messenger.cs) and the main procedures in handling WPF (ObservableObject.cs, PropertyObserver.cs), Command of event (ReplayCommand.cs).

+ MvvmApp: have ViewModel folder and Views folder.

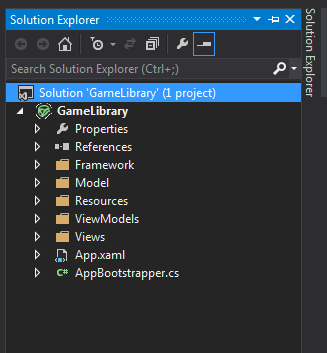
* + - Views: Interface screens. (NumberView.xaml,…)
    - ViewModel: Class control view in Views (NumberViewModel.cs).

[**Caliburn**](http://www.caliburnproject.org/)

* Author: [Rob Eisenberg](http://robeisenberg.com/Work.aspx)
* Silverlight support: yes
* Documentation: complete [online documentation](http://caliburn.codeplex.com/Wiki/View.aspx?title=Table%20Of%20Contents) available
* Hosting: CodePlex http://www.codeplex.com/caliburn
* License: MIT license
* Features:
  + commands are built on top of Actions and thus share many of the same features, including multiple input parameters, filters and automatic asynchronous execution
  + presenters that handles UI lifecycle issue such as handling activation, deactivation and shutdown semantics for various UI components
  + Caliburn applications are fully testable
  + various utilities such as a background task manager
  + supports various UI pattern (not MVVM only)
  + dependency injection container

Download from https://caliburm.codeplex.com/ and use template to develop

Structure of Solution:

Solution have: Framework folder, Model folder, Resource folder, ViewModels folder, Views folder.

+ FrameWork folder: have the serving class apart.

+ Model folder: declared the model.

+ Resource: contains extra screen sharing

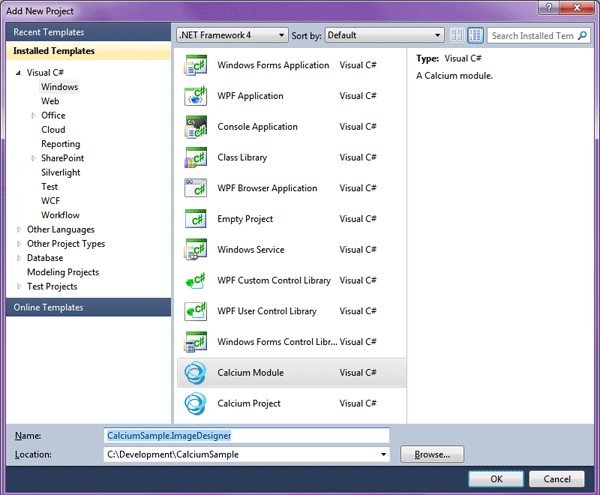
+ ViewModels: contain class control view in Views folder.

+ Views folder: contain screen display.

[**Calcium**](http://calcium.codeplex.com/)

* Author: [Daniel Vaughan](http://danielvaughan.orpius.com/)
* Silverlight support: no
* Documentation: 2 very detailed articles on CodeProject ([part1](http://www.codeproject.com/KB/WPF/CalciumPart01.aspx) and [part2](http://www.codeproject.com/KB/WPF/CalciumPart02.aspx))
* Hosting: [CodePlex](http://calcium.codeplex.com/)
* License: use, copy, modify, and/or distribute and keep the copyright!
* Features:
  + Module Manager for enabling or disabling of modules at runtime
  + messaging services for interacting with the user from the client or server using the same API
  + Command Service to associate WPF ICommands with content interfaces that only become active when an active view or viewmodel implements the interface
  + Region Adapters for ToolBars and Menus
  + Client-server logging ready to work out-of-the-box
  + Includes modules, such as a Web Browser, Text Editor, Output Window, and many more
  + Tabbed interface with dirty file indication (reusable across modules)

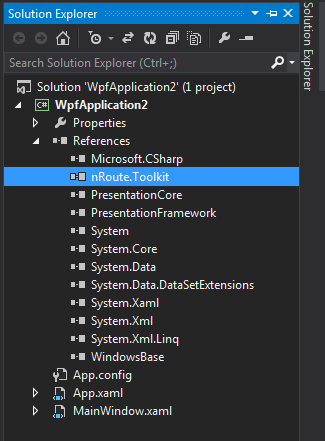
Lastest Version Calcium support for VS2010.



[**nRoute**](http://www.orktane.com/Blog/)

* Author: [Rishi](http://www.orktane.com/Blog/)
* Silverlight support: yes
* Documentation: many articles available on author’s blog (see CodePlex project home page for the links)
* Hosting: [CodePlex](http://nroute.codeplex.com/)
* License: MS-PL
* Features:
  + support Blend3 behaviors and triggers models
  + resource locator framework
  + view services: OpenFileDialog, ShowMessage…
  + uses attributes to map View and ViewModel together

Download nRoute from <https://nroute.codeplex.com/> to add to Project

code in .xaml :

<n:Application x:Class="nRoute\_MVVM\_Basics\_WPF.App"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:n="http://nRoute/schemas/2010/xaml"

StartupUri="MainWindow.xaml">

<n:Application.ApplicationLifetimeObjects>

<n:nRouteApplicationService />

</n:Application.ApplicationLifetimeObjects>

</n:Application>

Code in .xaml.cs

namespace nRoute\_MVVM\_Basics\_WPF

{

public partial class App : nRoute.ApplicationServices.Application

{

}

}

Code in ViewModel:

public class MainWindowViewModel : ViewModelBase

{ private DateTime? \_date; public DateTime? Date

{ get{return \_date; }

set{ if (\_date != value)

{ \_date = value;

NotifyPropertyChanged(() => Date); }}

public MainWindowViewModel()

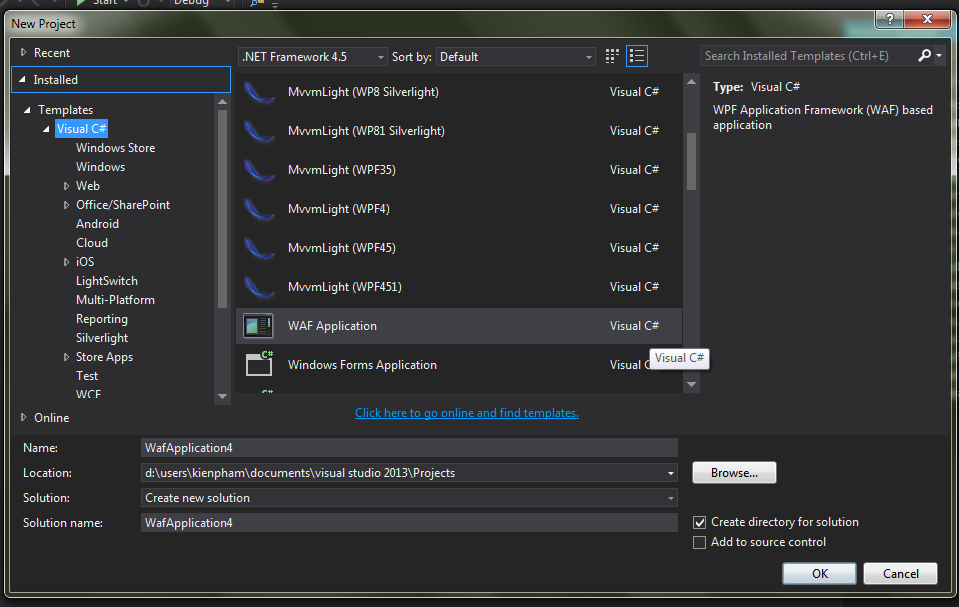
{Date = DateTime.UtcNow;}

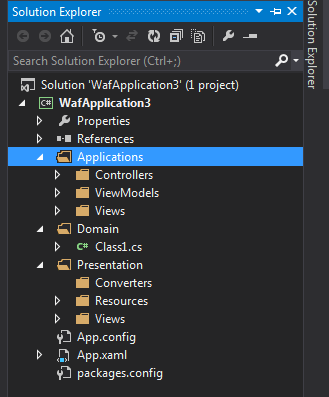
}

# [WAF](http://nitomvvm.codeplex.com/) ([WPF Application Framework](https://waf.codeplex.com/))

* Documentation: yes
* Hosting: https://waf.codeplex.com/
* Features: The WPF Application Framework (WAF) is a lightweight Framework that helps you to create well structured WPF Applications. It supports you in applying a Layered Architecture and the Model-View-ViewModel (aka MVVM, M-V-VM, PresentationModel) pattern.
* Intgrate WAF into your Project:
* Please use the following NuGet packages to integrate WAF into your Visual Studio Projects:
* PM> Install-Package [waf](http://www.nuget.org/packages/waf)
* PM> Install-Package [waf.testing](http://www.nuget.org/packages/waf.testing)
* Inclusive [SymbolSource](http://www.symbolsource.org/Public/Home/VisualStudio) support for debugging into the source code.

Create app with WAF



**Structure of Solution:**

Folder Application: contain Controllers, ViewModels, Views in App.

Folder Domain: contain Class Model

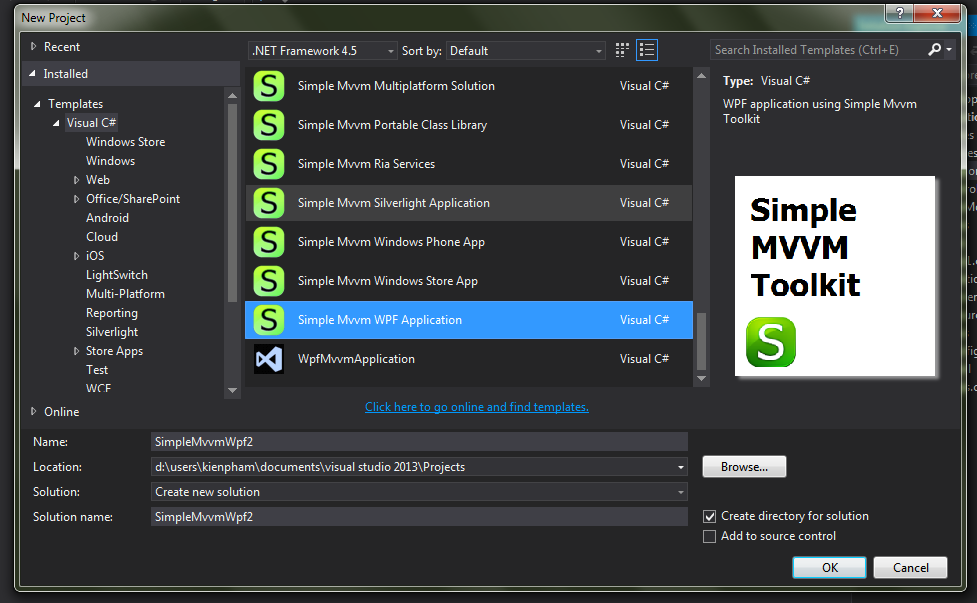
Folder Presentation: contain resource, extra display screen sharing,..

**Simple MVVM**

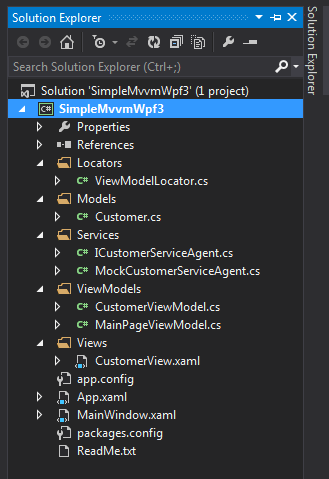
Download from: <https://simplemvvmtoolkit.codeplex.com/>

Features: Simple MVVM Toolkit makes it easier to develop WPF, Silverlight, Windows Store, Windows Phone, iOS and Android applications using the Model-View-ViewModeldesign pattern. The purpose of the toolkit is to provide a simple framework and set of tools for getting up to speed quickly with applications based on the MVVM design pattern. The emphasis is on **simplicity**, but it contains everything you need to implement MVVM for real-world line of business applications.

Create app:



Structure of Solution:

+ Folder Locators: contain connection method DB

+ Folder Models: contain class model

+ Folder Services: contain Service use connection method.

+ Folder ViewModels: contain class control view in Views folder

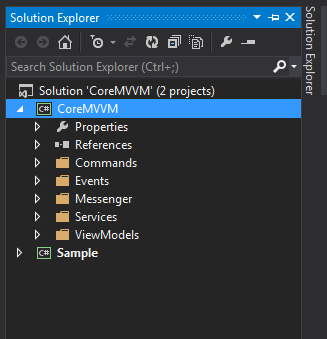
+ Folder Views: contain screen display .

[**Basic MVVM framework**](http://coremvvm.codeplex.com/)

* Author: [Lester Lobo](http://blogs.msdn.com/llobo/default.aspx)
* Silverlight support: no
* Documentation: sample application available with the library
* Hosting: [CodePlex](http://coremvvm.codeplex.com/)
* License: MS-PL
* Features:
  + delegating Commands\Keybinding
  + messaging between VM’s
  + handling events as commands with attached behaviors
  + handling dialogs (and more) as services
  + VS code snippets

Download from <https://coremvvm.codeplex.com/> to develop

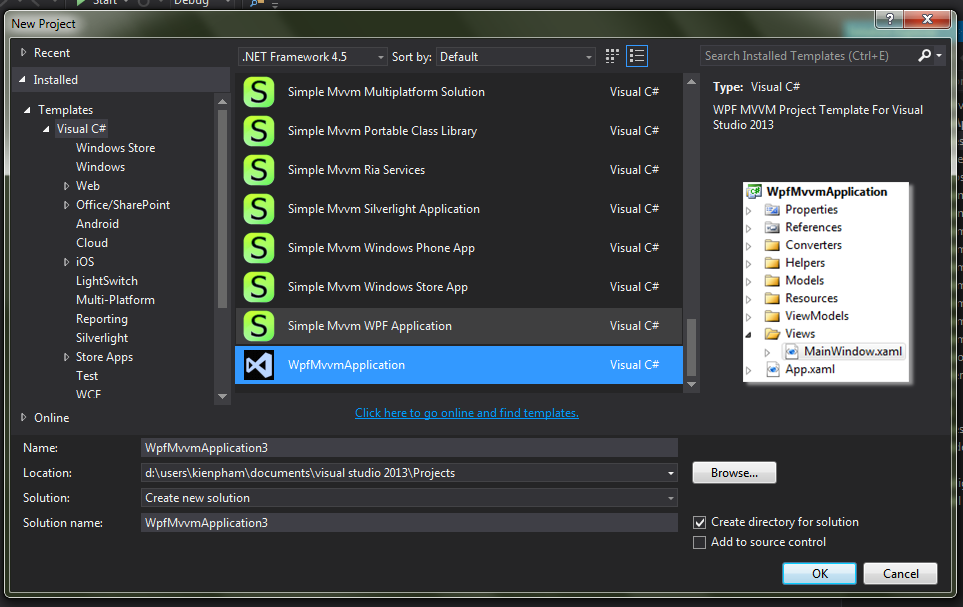
Library: CoreMVVM : have class template use to develop.

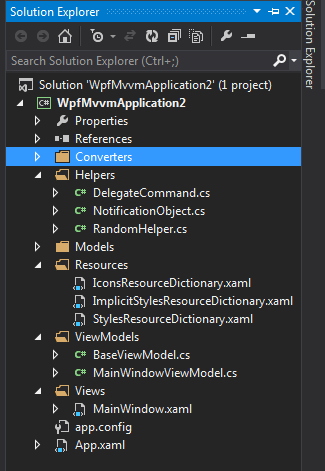


**WPF MVVM project template**

* Author: Ofir Shemesh
* Documentation: sample application available with the library
* Hosting: <https://visualstudiogallery.msdn.microsoft.com/970005b8-ee15-4295-9960-375e6ea1276c>
* Features:
  + - This extension will add a project template to create a WPF client application with MVVM pattern project structure.
    - the new project have an MVVM folders and classes structure with some randomly generated data and a simple view.

Create app:



**Structure of Solution:**

+ Folder Helpers: contain Class support to MVVM application

+ Folder Models: contain Class Model

+ Folder Resource: contain extra screen display sharing

+ Folder ViewModels: contain Class control view in Views folder.

+ Folder Views: contain screen display. (.xaml)

**Prims**

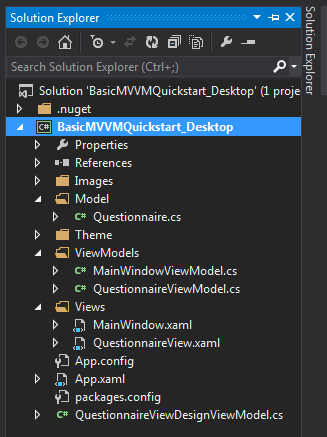
Download from: <https://www.microsoft.com/en-us/download/details.aspx?id=42537>

The Prism.Mvvm portable class library targets .NET 4.0 or higher, Windows Phone 8, and Windows Store apps (Windows 8) and higher. Prism.Mvvm includes the following:

* DelegateCommand
* CompositeCommand
* BindableBase
* ErrorsContainer
* PropertySupport
* ViewModelLocationProvider

Install and develop

**Structure of Solution**

+ Model: contain class model

+ ViewModels: contain class control view in Views folder

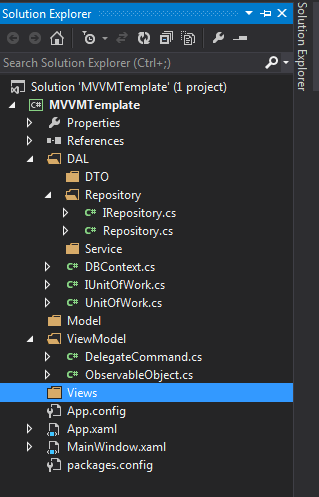
+ Views: contain screen display.

**MVVM Built Craft**

* Features:
  + - Built time consuming and many code line.
    - Highly customizable built by craft
    - Is the basic model

Create Solution:

**Structure of Solution:**

Model: contain class Model

ViewMode: Class control view in Views folder. Contain (DelegateCommand and ObservableObject). Inside:

-DelegateCommand: Call method ICommand to handling event in Views.

-ObservableObject: use interface INotifyPropertyChanged to perform data updates change.

Views: contain screen display in App

DAL: - DTO: data tranfer object

- Repository: 2 class (Repository and IRepository)

- Service: contain Using connection DB method to load data from DB.

- Contain UnitOfWork and IUnitOfWork to contain method how to handle the data of the Entity Framework

- Contain DBContext: setting connection to DB