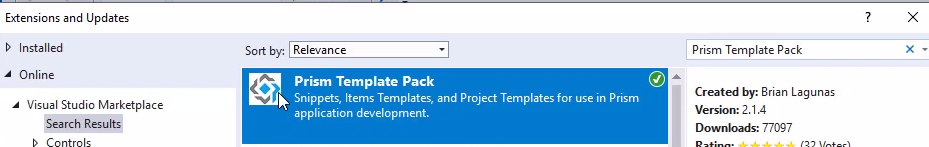
**Section01 Introduction to Prsim**

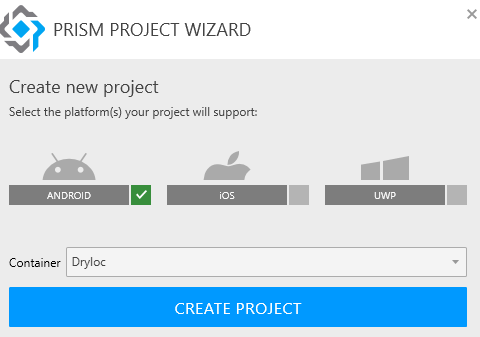
**Notes: -**

**1-we have to install Prism package on VS 2019**



**2-create new project of type prism blank app (Xamarin.Forms) > name it as MyFirstApp**

**3-We choose the android platform and container Dryloc**



**4-we see that in Prism it will create .netStandard project**

**<Project Sdk="Microsoft.NET.Sdk">**

**<PropertyGroup>**

**<TargetFramework>netstandard2.0</TargetFramework>**

**</PropertyGroup>**

**<ItemGroup>**

**<PackageReference Include="Xamarin.Forms" Version="4.8.0.1364" />**

**<PackageReference Include="Xamarin.Essentials.Interfaces" Version="1.5.3.2" />**

**<PackageReference Include="Prism.DryIoc.Forms" Version="8.0.0.1909" />**

**</ItemGroup>**

**</Project>**

**5-we see in the App.xaml that the xmlns:prism refer to the prismlirary.com**

**//we now are using <prism:PrismApplication> as container for all components we defined as application resources ,existing only on the App.xaml.**

<?xml version="1.0" encoding="utf-8" ?>

<prism:PrismApplication xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

xmlns:prism="http://prismlibrary.com"

x:Class="MyFirstApp.App">

<Application.Resources>

<!-- Application resource dictionary -->

</Application.Resources>

</prism:PrismApplication>

**6-on the Views folder > create new item > Prism ContentPage (Xamarin.Forms)**

**//we see that it automatically create the following things**

**//on the View Model >ViewAViewModel.cs**

**//on the Views > ViewA.xaml**

**//on the App.xaml.cs it will register the view model ViewAViewModel.cs with the ViewA.xaml**

**protected override void RegisterTypes(IContainerRegistry containerRegistry){**

**//we see the registration automatically happen**

**containerRegistry.RegisterForNavigation<ViewA, ViewAViewModel>();}**

**7-on the ViewAViewModel.cs we update the code as below by declare variable**

**public class ViewAViewModel : ViewModelBase{**

**public ViewAViewModel(INavigationService navigationService) : base(navigationService){Title = "This is View A Page";}}**

**8-on the ViewA.xaml we update the code to show the title variable passed in**

<?xml version="1.0" encoding="utf-8" ?>

<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

xmlns:prism="http://prismlibrary.com"

prism:ViewModelLocator.AutowireViewModel="True"

x:Class="MyFirstApp.Views.ViewA"

Title="{Binding Title}">

<StackLayout HorizontalOptions="CenterAndExpand" VerticalOptions="CenterAndExpand">

<Label Text="{Binding Title}" />

</StackLayout>

</ContentPage>

**Section02 Navigation Basics**

**Notes: -**

**1-you have to register all the views with the view models on the App.xaml.cs**

**2-you have to set the initial navigate to the startup view**

**3-with register built in page called NavigationPage it will provide** **funcitonaitliy like back button and so on.**

**Steps:-**

**1-We will create three pages MainPage and ViewA.xaml and ViewB.xaml as below**

**(right click on folder View Models > New > prism blank app (Xamarin forms))**

**<?xml version="1.0" encoding="utf-8" ?>**

**<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"**

**xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"**

**x:Class="PrismToutarial.Views.MainPage" Title="{Binding Title}">**

**<StackLayout HorizontalOptions="CenterAndExpand" VerticalOptions="CenterAndExpand">**

**<Label Text="Welcome to Xamarin Forms and Prism!" />**

**<Button Text="Navigate to ViewA" Command="{Binding NavigateCommand}" />**

**</StackLayout>**

**</ContentPage>**

**//on the MainPageViewModel**

**public class MainPageViewModel : ViewModelBase{**

**public MainPageViewModel(INavigationService navigationService**

**): base(navigationService){Title = "Main Page";}**

**private DelegateCommand \_navigateCommand;**

**public DelegateCommand NavigateCommand => \_navigateCommand ?? (**

**\_navigateCommand = new DelegateCommand(ExecuteNavigateCommand));**

**public async void ExecuteNavigateCommand(){**

**await NavigationService.NavigateAsync("ViewA");}}**

**on the ViewAPage.xaml**

**<?xml version="1.0" encoding="utf-8" ?>**

**<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"**

**xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"**

**xmlns:prism="http://prismlibrary.com"**

**prism:ViewModelLocator.AutowireViewModel="True"**

**x:Class="PrismToutarial.Views.ViewA"**

**Title="{Binding Title}">**

**<StackLayout HorizontalOptions="CenterAndExpand" VerticalOptions="CenterAndExpand">**

**<Label Text="Welcome to Xamarin Forms and Prism!" />**

**<Button Text="Navigate to ViewB" Command="{Binding NavigateCommand}" />**

**</StackLayout>**

**</ContentPage>**

**<?xml version="1.0" encoding="utf-8" ?>**

**<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"**

**xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"**

**xmlns:prism="http://prismlibrary.com"**

**prism:ViewModelLocator.AutowireViewModel="True"**

**x:Class="PrismToutarial.Views.ViewA" Title="{Binding Title}">**

**<StackLayout HorizontalOptions="CenterAndExpand" VerticalOptions="CenterAndExpand">**

**<Label Text="Welcome to Xamarin Forms and Prism!" />**

**<Button Text="Navigate to ViewB" Command="{Binding NavigateCommand}" />**

**</StackLayout>**

**</ContentPage>**

**On the ViewBPage.xaml**

**public class ViewBViewModel : ViewModelBase{**

**public ViewBViewModel(INavigationService navigationService) : base(navigationService){**

**Title = "View B Page";}**

**private DelegateCommand \_navigateCommand;**

**public DelegateCommand NavigateCommand => \_navigateCommand ?? (\_navigateCommand = new DelegateCommand(ExecuteNavigateCommand));**

**public async void ExecuteNavigateCommand(){**

**//it will navigate to the MainPage**

**await NavigationService.NavigateAsync("MainPage");}}**

**<?xml version="1.0" encoding="utf-8" ?>**

**<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"**

**xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"**

**xmlns:prism="http://prismlibrary.com"**

**prism:ViewModelLocator.AutowireViewModel="True"**

**x:Class="PrismToutarial.Views.ViewB"**

**Title="{Binding Title}">**

**<StackLayout HorizontalOptions="CenterAndExpand" VerticalOptions="CenterAndExpand">**

**<Label Text="Welcome to Xamarin Forms and Prism!" />**

**<Button Text="Navigate to MainPage" Command="{Binding NavigateCommand}" />**

**</StackLayout>**

**</ContentPage>**

**2-we see that on each create View model it will prepare all the configuration on the view model and view and registration on the app.xaml**

**Section03 Passing Navigation Parameters**

**Notes: -**

**1-prsim use HTTP URL based navigation scheme**

**//we define the NavigationParameters with set multiple parameters as below**

**public async void ExecuteNavigateCommand(){**

**//we can pass parameter through NavigationService as below**

**var p = new NavigationParameters("firstParam=First Params From MainPage&secondParam=Second Params From MainPage");**

**await NavigationService.NavigateAsync("ViewA", p);}**

**2-on the ViewModelBase we apply the following code as below**

**//we use the INavigationAware interface with implement the two method**

**OnNavigatedForm , OnNavigatedTo**

**public class ViewModelBase : BindableBase, IInitialize, INavigationAware, IDestructible ,IConfirmNavigationAsync{**

**protected INavigationService NavigationService { get; private set; }**

**protected IPageDialogService PageDialogService { get; private set; }**

**private string \_title;**

**public string Title{get { return \_title; }set { SetProperty(ref \_title, value); }}**

**private string \_firstParam;**

**public string FirstParam{get { return \_firstParam; }set { SetProperty(ref \_firstParam, value); }}**

**private string \_secondParam;**

**public string SecondParam{get { return \_firstParam; }set { SetProperty(ref \_secondParam, value); }}**

**public ViewModelBase(INavigationService navigationService){**

**NavigationService = navigationService;}**

**public virtual void Initialize(INavigationParameters parameters){}**

**//when thre request send from the current page**

**public virtual void OnNavigatedFrom(INavigationParameters parameters){**

**FirstParam = parameters["firstParam"] != null ? parameters["firstParam"].ToString() : string.Empty;**

**SecondParam = parameters["secondParam"] != null ? parameters["secondParam"].ToString() : string.Empty;}**

**//when the request recieved from the target page**

**public virtual void OnNavigatedTo(INavigationParameters parameters){**

**FirstParam = parameters["firstParam"] != null ? parameters["firstParam"].ToString() : string.Empty;**

**SecondParam = parameters["secondParam"] != null ? parameters["secondParam"].ToString() : string.Empty;}**

**public virtual void Destroy(){}}**

**Section04 Confirm or Preventing Navigation**

**Notes: -**

**1-Prism by default cannot prevent hardware back button or built in software back button, otherwise we can use IConfirmNavigationAsync interface to apply the preventing back to previous page and use show navigation alert to allow pass navigation or not**

**public class ViewModelBase : BindableBase, IInitialize, INavigationAware, IDestructible ,IConfirmNavigationAsync{**

**protected INavigationService NavigationService { get; private set; }**

**protected IPageDialogService PageDialogService { get; private set; }**

**private string \_title;**

**public string Title{get { return \_title; }set { SetProperty(ref \_title, value); }}**

**private string \_firstParam;**

**public string FirstParam{get { return \_firstParam; }set { SetProperty(ref \_firstParam, value); }}**

**private string \_secondParam;**

**public string SecondParam{get { return \_firstParam; }set { SetProperty(ref \_secondParam, value); }}**

**public ViewModelBase(INavigationService navigationService,IPageDialogService pageDialogService){**

**NavigationService = navigationService;**

**PageDialogService = pageDialogService;}**

**public virtual void Initialize(INavigationParameters parameters){}**

**//when thre request send from the current page**

**public virtual void OnNavigatedFrom(INavigationParameters parameters){**

**FirstParam = parameters["firstParam"] != null ? parameters["firstParam"].ToString() : string.Empty;**

**SecondParam = parameters["secondParam"] != null ? parameters["secondParam"].ToString() : string.Empty;}**

**//when the request recieved from the target page**

**public virtual void OnNavigatedTo(INavigationParameters parameters){**

**FirstParam = parameters["firstParam"] != null ? parameters["firstParam"].ToString() : string.Empty;**

**SecondParam = parameters["secondParam"] != null ? parameters["secondParam"].ToString() : string.Empty;}**

**public virtual void Destroy(){}**

**public Task<bool> CanNavigateAsync(INavigationParameters parameters){**

**return PageDialogService.DisplayAlertAsync("My Title", "Save ?", "Accept", "Cancel");}}**