

Genesys Source Framework

eBook

Copyright © Genesys Source. All rights reserved. Printed in USA.

Genesys Source is a registered trademark of Genesys Source. Genesys Framework and Genesys Cloud Dev Environment are trademarks of Genesys Source.

Microsoft, Azure, HoloLens, Hyper-V, Visual Studio, Windows and Xamarin are registered trademarks of Microsoft Corporation.

All other product names and logos are trademarks and service marks of their respective companies.

This document is provided "as-is." Information in this document, including URL and other Internet website references, may change without notice. Genesys Source assumes no liability for damages incurred directly or indirectly from errors, omissions, or discrepancies between the product and this document.

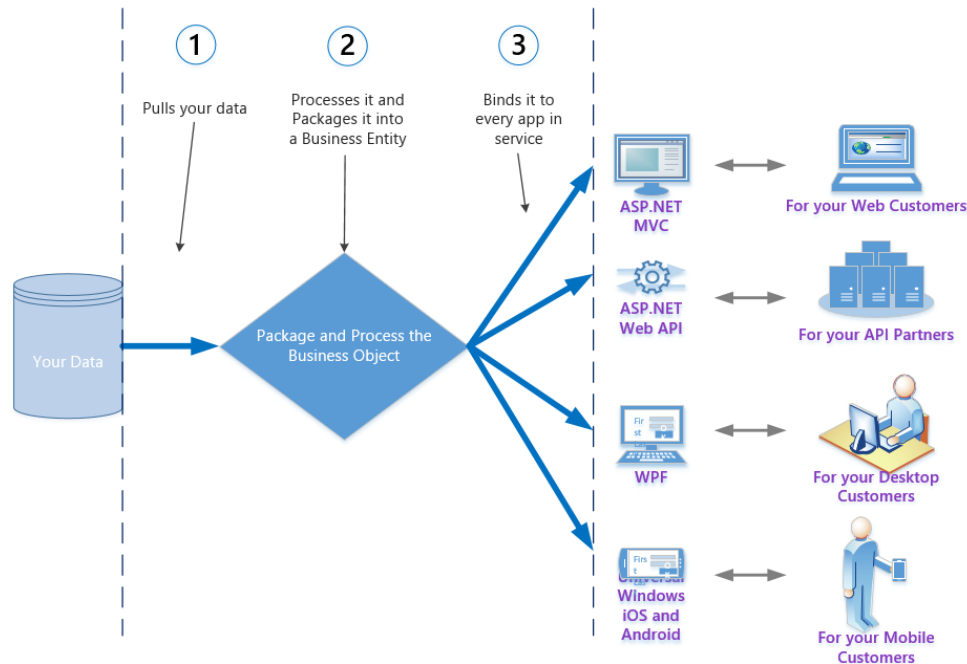
An attempt has been made to state all allowable values where applicable throughout this document. Any values or parameters used beyond those stated may have unpredictable results.

Contents

Genesys Framework: One Framework – Your Data – Any Platform.....	4
Pre-requisites	5
Installing the Genesys Framework.....	5
Uninstalling the Genesys Framework	7
What is in the Genesys Framework?	8
Running the Genesys Framework Tests [Framework.Test]	9
Debugging the Genesys Framework.....	10
Debugging Framework.WebApp (MVC) and Framework.WebServices (Web API)	10
Debugging Framework.UniversalApp (UWP) and Framework.DesktopApp (WPF)	11
The Database	11
Loose-couple your SQL Tables to Genesys Framework SQL Views/SPs	11
About Framework.Database (SSDT).....	12
About Framework.Database (SSDT).....	12
Re-wire Framework.Database SQL Views to connect to your SQL Tables	13
Add a new Field/Property to the Customer object.....	14
Publishing the Genesys Framework to IIS and SQL Server.....	16
Publishing Framework.Database (SSDT) to a SQL Server	16
Publishing Framework.WebApp (MVC) to an IIS Web Server	18
Publishing Framework.WebServices (Web API) to an IIS Web Server	19
Tech Aspects of the Genesys Framework.....	19
Database Connections in App_Data.....	19
Web Service Connections in App_Data.....	19
Data Access (EF Core) coupled to Database (SSDT) Views and Stored Procedures.....	19
Data Access project Repository and Data Objects.....	20
Can't cast? Copy with Fill<T> and FillRange<T>	20
Setup for 2-tier or Setup for n-tier.....	20
Why the Genesys Framework?	20
Why build reusable code?	20
Why code full-stack, cross-platform business objects?.....	20
Getting Help	21

Genesys Framework: One Framework – Your Data – Any Platform

The Genesys Framework is a C# framework that powers full-stack quick-start projects, enabling developers to code business objects that run cross-platform, on .NET Core and .NET Framework and in any application...without any code plumbing.



Tell me a little more about the Genesys Framework

What is it?

The *Genesys Framework* is a full-stack business object framework, exposing your data as C# objects from your SQL Database to any type of .NET app.

Your *Customer* object, for example, can exist as a:

- **Data object:** EF-enabled to pull data from your SQL Server via common repository-pattern methods like: *GetById()*, *GetAll()*, *Save()* and *Delete()*.
- **Domain object:** Enriched domain object containing domain behavior such as *IsEmployee*, *IsActive*, *HasRegistered*, *CurrentStatus*, etc.
- **View Model object:** Model for your screen Views that are thin, atomic and transportable. Can be extended to have view-specific properties such as a Gender select list, without altering your *CustomerModel* class.
- **Data-transfer object:** Expose your object as a DTO for data-transfer specific operations such as inter-web-services data sharing or returning your object through public API endpoints.

Why do I care?

Full-stack projects in seconds: Your entire C# stack solution is up in seconds, ready for you to add your business objects. From Visual Studio: *File -> New -> Project*, select *Genesys Source Quick-Start*, and your stack framework is runnable and ready to code.

Cross-platform by default: Enable your business-objects to run truly everywhere: In web apps - In any mobile app - In desktop native apps - As middle-tier domain service - As Public API web services - even in client-side TypeScript

	Full-stack by default: Other frameworks focus only on the UI, only in the web, only for data access. GF4 includes Database project (SSDT), Middle Tier (.NET Standard) and all major apps (MVC, Web API, UWP/iOS/Android, WPF.)
How do I get it?	<p>The Genesys Source Framework is available where you need it most.</p> <p>From Microsoft:</p> <ul style="list-style-type: none"> ➤ At Microsoft on http://marketplace.VisualStudio.com ➤ On Microsoft Azure at http://bit.ly/2zw5UzN ➤ In Visual Studio on Tools -> Extensions and Updates <p>From GitHub:</p> <ul style="list-style-type: none"> ➤ On GitHub at http://github.com/genesysource <p>From Genesys Source:</p> <ul style="list-style-type: none"> ➤ At Genesys Source at http://cloud.GenesysSource.com

1: Genesys Framework: One Framework – Your Data – Any Platform

Pre-requisites



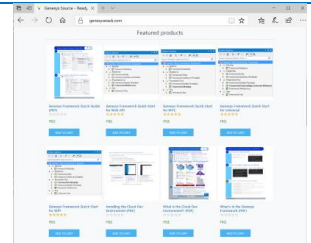

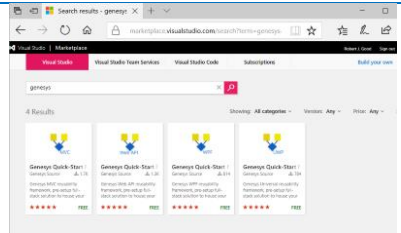

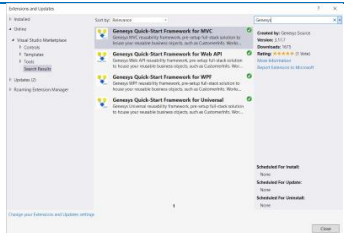
To get the most out of the Genesys Framework, the following skills are recommended:


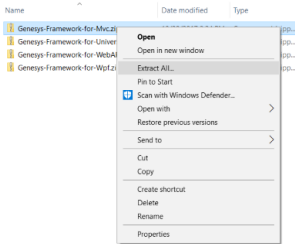

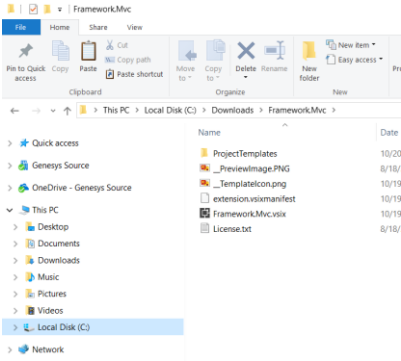

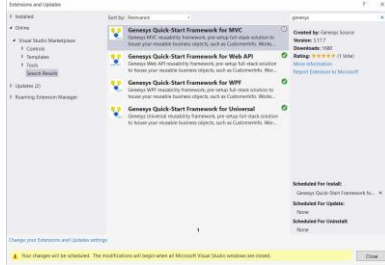

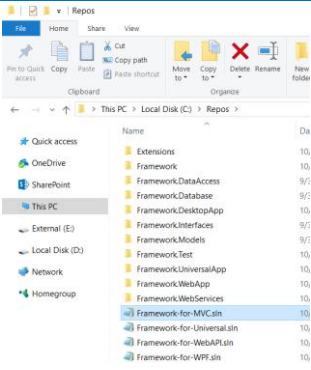

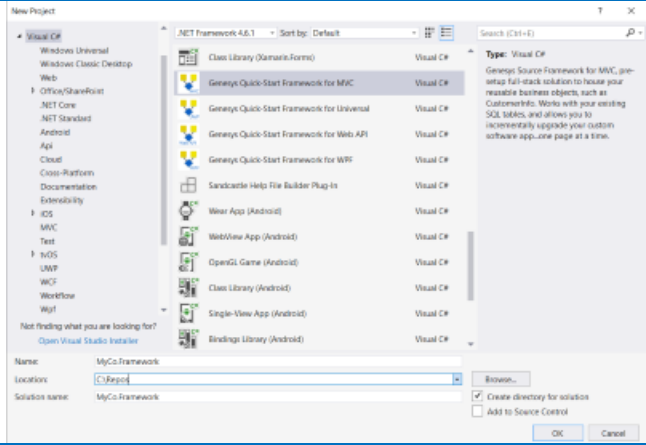

- ✓ Moderate C# and .NET, HTML and XAML
- ✓ Low/Moderate T-SQL and Database design
- ✓ Awareness of N-tier, MVC, MVVM and REST
- ✓ Visual Studio Community (or greater) from <https://www.visualstudio.com/downloads/>
- ✓ SQL Server Management Studio from <https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms>

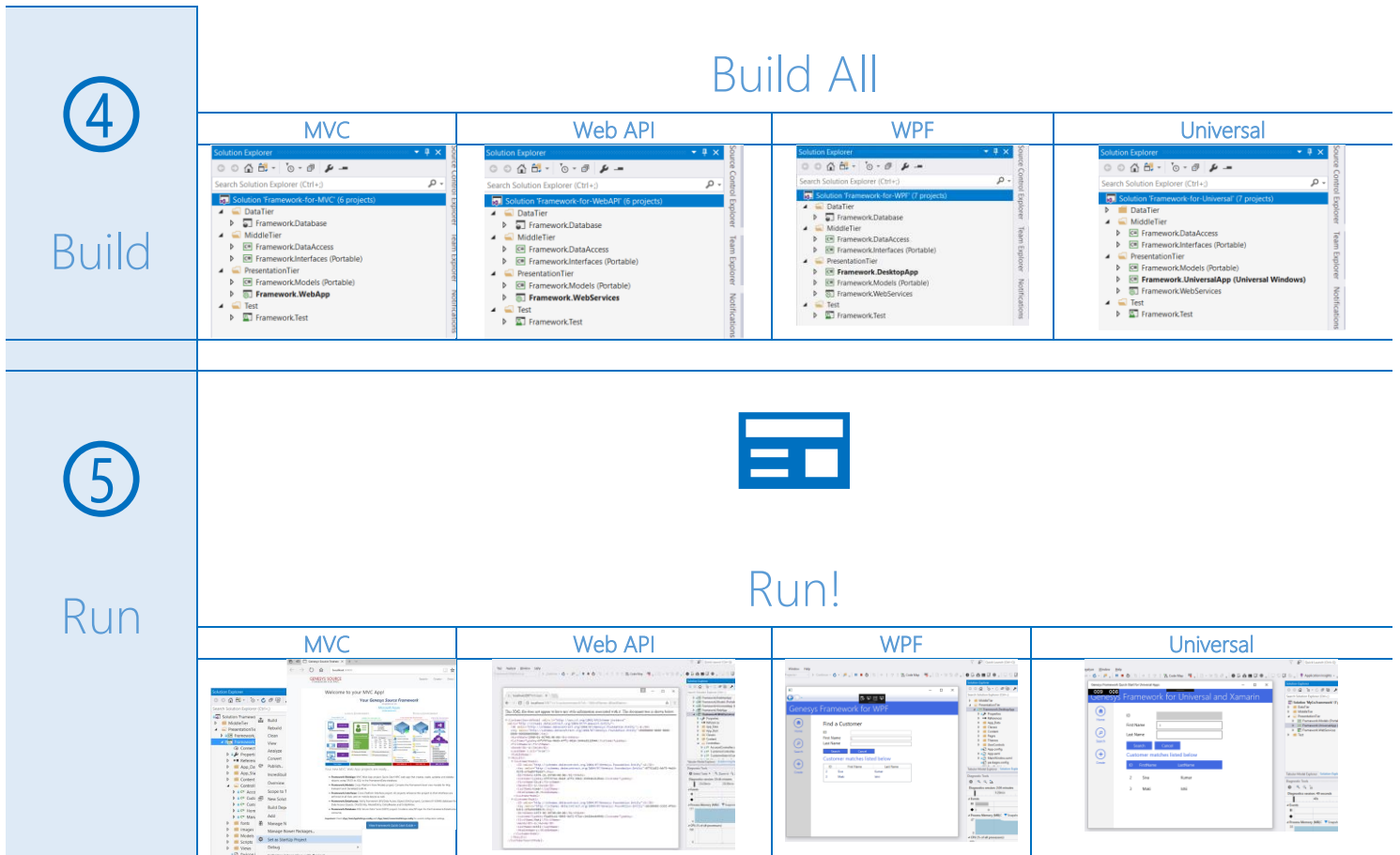
Installing the Genesys Framework

The Genesys Framework is free open-source on GitHub and available as a Zip download, a Vsix download or through Visual Studio Marketplace...installable with any method that you prefer most.

The easiest way to get the Genesys Framework is to download our Quick-Start projects for MVC, Web API, WPF and Universal. The Quick-Start projects are a small starter with the basics to get you running fast.

Get as Zip, Vsix or directly in Visual Studio			
 <p>Get</p>	 <p>Download Zip GenesysSource.com</p> 	 <p>Download Vsix VisualStudioMarketplace.com</p> 	 <p>Install in Visual Studio Tools -> Extensions and Updates</p> 
Framework for MVC	Download Framework-for-MVC.sln	Download Framework.MVC.vsix	Tools -> Extensions and Updates -> Search: GENESYS -> Download

Framework for Web API	Download Framework-for-WebAPI.sln	Download Framework.WebAPI.vsix	Tools -> Extensions and Updates -> Search: GENESYS -> Download
Framework for WPF	Download Framework-for-WPF.sln	Download Framework.WPF.vsix	Tools -> Extensions and Updates -> Search: GENESYS -> Download
Framework for Universal	Download Framework-for-Universal.sln	Download Framework.Universal.vsix	Tools -> Extensions and Updates -> Search: GENESYS -> Download
Framework for Core	Download Framework-for-Core.sln	Download Framework.Core.vsix	Tools -> Extensions and Updates -> Search: GENESYS -> Download
<div>②</div> <div>Install</div>	 <div>Extract Zip</div> 	 <div>Run Vsix</div> 	 <div>Close Visual Studio to Install</div> 
<div>③</div> <div>Open</div>	 <div>Open Solution File (.sln)</div> 	 <div>Create Solution</div> <div>File -> New Project -> C#</div> 	
			

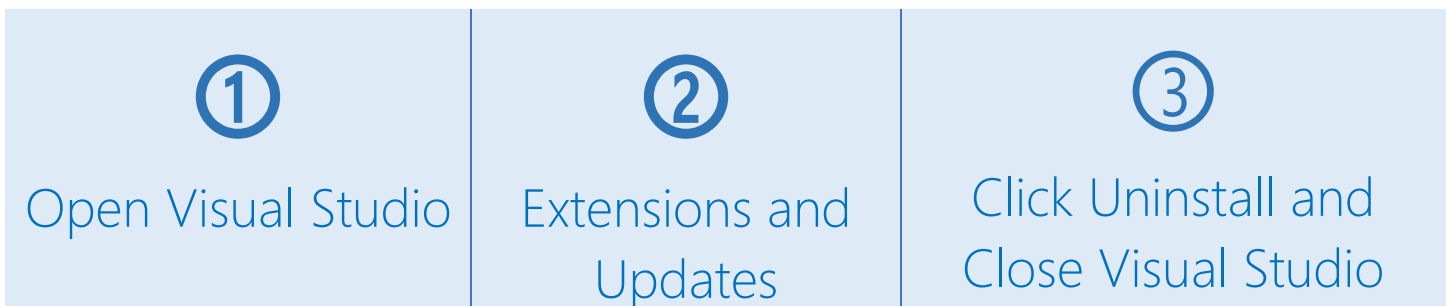


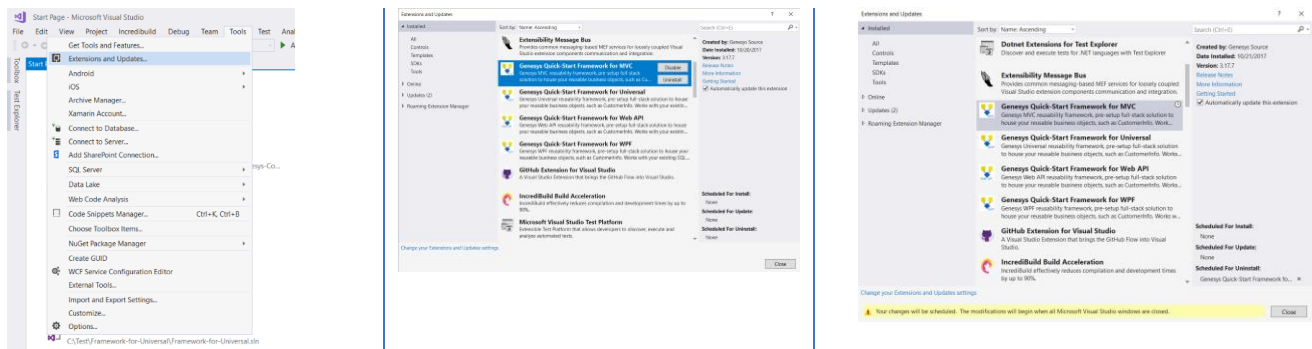
1: Installing Genesys Source Framework .NET Solutions

Uninstalling the Genesys Framework

If the Genesys Framework was downloaded via Zip file, no uninstallation is necessary.

For installs through the Visual Studio Marketplace or in Visual Studio, follow these steps to uninstall the Genesys Framework from your IDE:

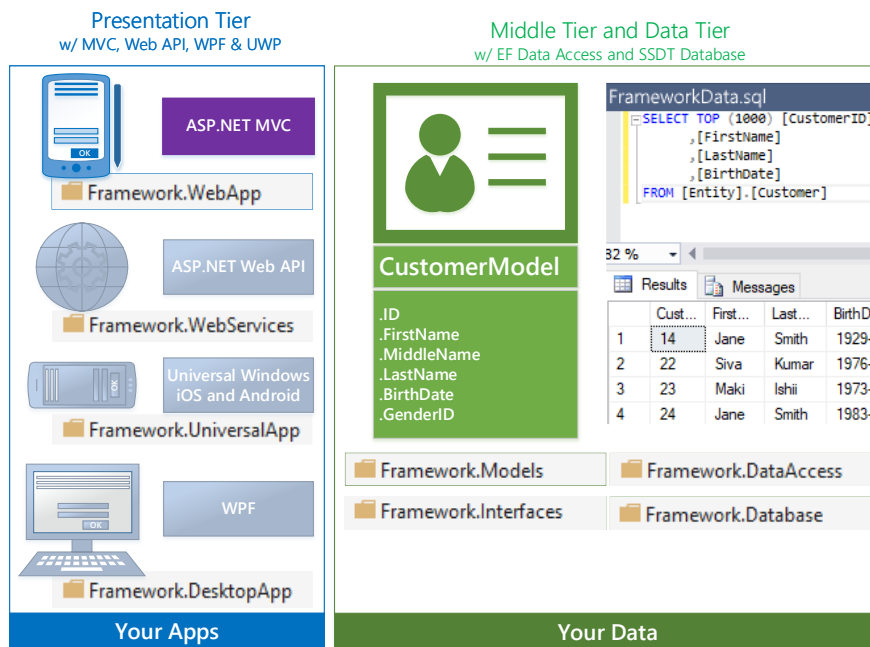




2: Uninstalling the Genesys Framework

What is in the Genesys Framework?

A Genesys Framework app includes full-stack projects for your application. From the database (SSDT), to data objects (EF), to models (.NET Core or Framework), exposed in any .NET application type such as MVC or UWP.



3: Genesys Framework Code and Runtime

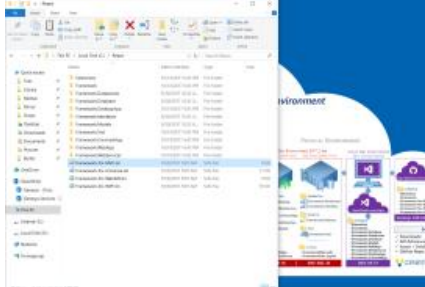
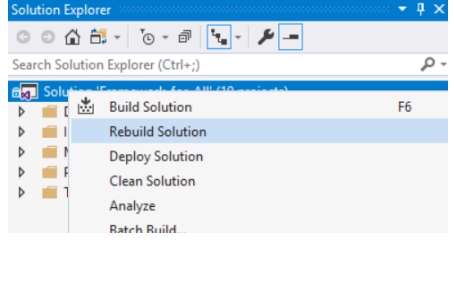
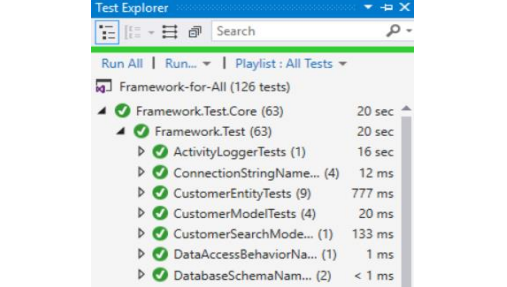
Visual Studio Project	
Framework.WebApp	<p>MVC Web App with all CRUD and Search operations for a Customer entity.</p> <p>Points of interest are:</p> <ul style="list-style-type: none"> ➤ \App_Data\ConnectionStrings.json – Database connection information ➤ \Views\Home\Index.cshtml – Home Page ➤ \Controllers\Customer\CustomerSearchController.cs – Processes all customer search requests
Framework.WebServices	<p>Web API web services with all CRUD and Search operations for a Customer entity.</p> <p>Points of interest are:</p> <ul style="list-style-type: none"> ➤ \App_Data\ConnectionStrings.json – Database connection information ➤ \Views\Home\Index.cshtml – Home Page ➤ \Controllers\Customer\CustomerSearchController.cs – Processes all customer search requests

Framework.UniversalApp	UWP Cross-Platform App with all CRUD and Search operations for a Customer entity. Points of interest are: <ul style="list-style-type: none"> ➤ \App_Data.ConnectionStrings.json – Database connection information ➤ \MainPage.xaml – Home Page ➤ \Pages\Customer\CustomerSearch.xaml – Processes all customer search requests
Framework.DesktopApp	WPF Desktop App with all CRUD and Search operations for a Customer entity. Points of interest are: <ul style="list-style-type: none"> ➤ \App_Data.ConnectionStrings.json – Database connection information ➤ \MainPage.xaml – Home Page ➤ \Pages\Customer\CustomerSearch.xaml – Processes all customer search requests
Framework.Models	Cross-platform PCL containing bindable screen models for MVC, WPF, UWP, WebForms, WinForms, Xamarin. Points of interest are: <ul style="list-style-type: none"> ➤ \Customer\CustomerModel.cs – View Model for Customer business object
Framework.Interop	Cross-platform PCL containing interfaces, to ensure all tiers share the same signature. Points of interest are: <ul style="list-style-type: none"> ➤ \App_Data.ConnectionStrings.json – Database connection information ➤ \Views\Home\Index.cshtml – Home Page ➤ \Controllers\Customer\CustomerSearchController.cs – Processes all customer search requests
Framework.DataAccess	Entity Framework data access objects, providing CRUD operations for Customer. Points of interest are: <ul style="list-style-type: none"> ➤ \Customer\CustomerInfo.cs – Data Access Object for Customer business object
Framework.Database	SSDT database containing all T-SQL for tables, views, stored procs, schemas, users. Points of interest are: <ul style="list-style-type: none"> ➤ \Tables\Customer\Customer.sql – Customer table ➤ \Views\CustomerCode\CustomerInfo.sql – View that connects table and code ➤ \Stored Procedures\CustomerCode\CustomerInsert.sql – Stored procedure that inserts to customer table

2: Genesys Framework .NET Projects

Running the Genesys Framework Tests [Framework.Test]

All products contain Framework.Test, an integration test project that tests your objects and support classes. To run all tests in the solution:

<p>①</p> <p>Open your Solution i.e. <i>Framework-for-MVC.sln</i></p>	<p>②</p> <p>Rebuild All on the Solution</p>	<p>③</p> <p>Click Run All in the Test Explorer Window</p>
		
<p>1. Open your solution, i.e. <i>Framework-for-MVC.sln</i> Visual Studio solution file - Default: C:\Source\Framework-for-MVC.sln</p>	<p>2. Right-click the solution and click <i>Rebuild Solution</i></p>	<p>3. Open the <i>Test Explorer</i> window - <i>Test</i> -> <i>Windows</i> -> <i>Test Explorer</i> 4. Click <i>Run All</i> to run all tests - All tests should execute successfully</p>

Hint for LocalDB: Check SSMS Server
(LocalDb)\MSSQLLocalDB if
FrameworkData_Primary.mdf is locked.

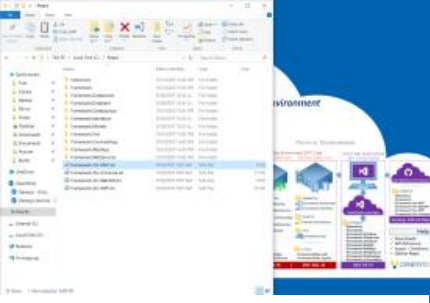
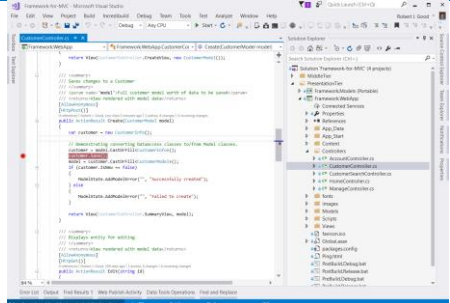
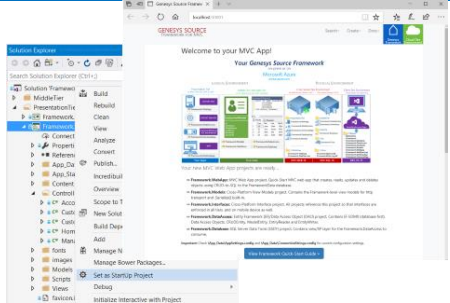
4: Running Framework.Test

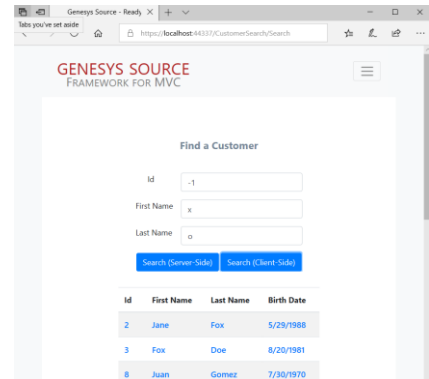
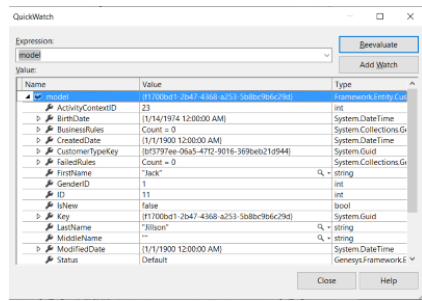
Debugging the Genesys Framework

The Genesys Framework contains App and Services projects that host your application. These Apps can be debugged using standard .NET debugging techniques in Visual Studio Community (or greater.)

Debugging Framework.WebApp (MVC) and Framework.WebServices (Web API)

To debug your Framework for MVC and Framework for Web API app, follow the procedures below:

<p>①</p> <p>Open the Solution <i>i.e. Framework-for-MVC.sln</i></p> 	<p>②</p> <p>Set Breakpoint in <i>CustomerSearchController.cs</i></p> 	<p>③</p> <p>Set as StartUp Project and Press F5 to Run</p> 
<p>5. Open the <i>Framework-for-MVC.sln</i> Visual Studio solution file - Default: C:\Source\Framework-for-MVC.sln</p> <p>6. Navigate to and open Framework.WebApp\Controllers\CustomerController</p>	<p>7. Set a breakpoint in the Search() method</p> <pre>public ActionResult Search(CustomerModel data) { var model = new CustomerSearchModel(); var searchResults = CustomerInfo.GetByAny(data).Take(25); }</pre>	<p>8. Right-click <i>Framework.WebApp</i> or <i>Framework.WebServices</i> project -> click <i>Set as StartUp Project</i></p> <p>9. Press F5 or ▶ to run - WebApp Url: http://localhost:30001/ - WebServices Url: http://localhost:30002/</p> <p>10. Home/Index.cshtml should display</p>
<p>④</p> <p>Enter First (x) and Last (o) -> Click Search</p>	<p>⑤</p> <p>Step-in to CustomerInfo.cs</p>	<p>⑥</p> <p>See Customer search data before returning to View</p>



11. Search from the home header
12. Enter a single letter (f) into First Name and a single letter (j) into Last Name
13. Click *Search* to hit breakpoint

14. Press F11 to step-in *CustomerInfo.cs*
`customer.Save();`
`returnValue = db.Save(this.ToEntity`

15. Pause execution on the following line and hit F9 to *Quick-Watch*
`var results = searchResults.ToList();`

5: Debugging Framework for MVC Apps

Debugging Framework.UniversalApp (UWP) and Framework.DesktopApp (WPF)

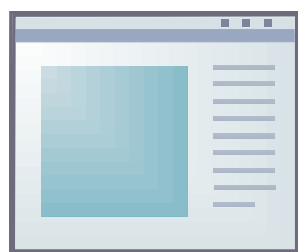
The Database

Loose-couple your SQL Tables to Genesys Framework SQL Views/SPs

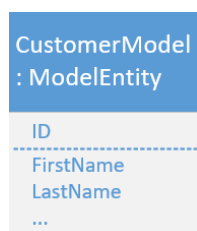
The Genesys Framework pulls data using Entity Framework, which can be tight-coupled directly to SQL Tables, or loose-coupled to SQL Views and Stored Procedures.

Out of the box, Genesys Framework:

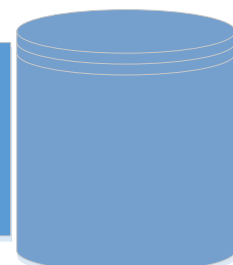
- Connects to the *FrameworkData* database
- Selects data from SQL Views, i.e. *FrameworkData.CustomerCode.CustomerInfo*
- Insert, update and delete through SQL Stored Procedures, i.e. *FrameworkData.CustomerCode.CustomerInsert*
- *Framework.DataAccess* project contains Repository and Data objects for the data, i.e. *Customer\CustomerInfo.cs*
- In the App project, data is exposed as View Model objects, i.e. *Customer\CustomerModel.cs*



Your Site, Service or App
with Customer Pages



Customer View



Your Database with
Customer Data

6: Data passing through Framework objects

About Framework.Database (SSDT)

The *Framework.Database* is a SQL Server project built on SQL Server Data Tools (SSDT). This project is responsible for:

1. Holding T-SQL for tables, schemas, indexes, constraints, users and roles
2. Holding and running the PreDeployment and PostDeployment scripts
3. DB Compare the *Framework.Database* project to the *FrameworkData* database
4. Publish the *Framework.Database* project to the *FrameworkData* database

Once deployed, you test the *FrameworkData* database as any other SQL Server database. Select from the Customer tables and *CustomerCode* views. Insert, update and delete from the *CustomerCode* stored procedures.

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'FrameworkData' database is expanded, showing 'Database Diagrams', 'Tables', 'System Tables', 'FileTables', 'External Tables', 'Activity.Activity', 'Activity.ExceptionLog', 'Entity.Customer', and 'Entity.CustomerType'. The 'Tables' folder is expanded, showing 'Customer'. In the center, a T-SQL query is displayed:
`-- Customer`
`--`
`Use FrameworkData`
`Select Top 100 C.CustomerID, C.FirstName, C.MiddleName, C.LastName, C.BirthDate, C.CreatedDate, C.ModifiedDate,`
`CT.CustomerTypeID, ct.CustomerTypeName`
`From FrameworkData.Entity.Customer C`
`Left Join FrameworkData.Entity.CustomerType CT On C.CustomerTypeID = CT.CustomerTypeID`
Below the query, the 'Results' tab shows a table with 10 columns: CustomerID, FirstName, MiddleName, LastName, BirthDate, CreatedDate, ModifiedDate, CustomerTypeID, and CustomerTypeName. The table contains 3 rows of data.

	CustomerID	FirstName	MiddleName	LastName	BirthDate	CreatedDate	ModifiedDate	CustomerTypeID	CustomerTypeName
1	1	John	M	Smith	1968-05-20 00:00:00.000	2016-10-30 16:14:06.237	2016-10-30 16:14:06.237	2	Premium Customers
2	2	Siva	N	Kumar	1976-11-15 00:00:00.000	2016-10-30 16:14:06.237	2016-10-30 16:14:06.237	3	Standard Customers
3	3	Maki	L	Ishii	1973-06-30 00:00:00.000	2016-10-30 16:14:06.237	2016-10-30 16:14:06.237	4	Potential Customers

7: Selecting from the Customer table

About Framework.Database (SSDT)

The *Framework.Database* is a SQL Server project built on SQL Server Data Tools (SSDT). This project is responsible for:

1. Holding T-SQL for tables, schemas, indexes, constraints, users and roles
2. Holding and running the PreDeployment and PostDeployment scripts
3. DB Compare the *Framework.Database* project to the *FrameworkData* database
4. Publish the *Framework.Database* project to the *FrameworkData* database

Once deployed, you test the *FrameworkData* database as any other SQL Server database. Select from the Customer tables and *CustomerCode* views. Insert, update and delete from the *CustomerCode* stored procedures.

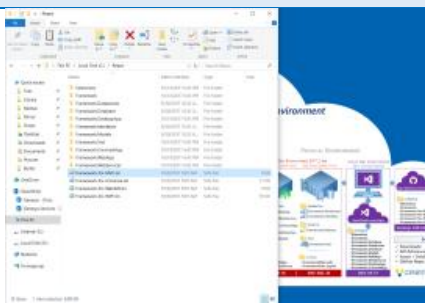
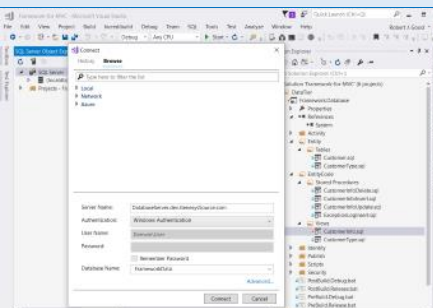
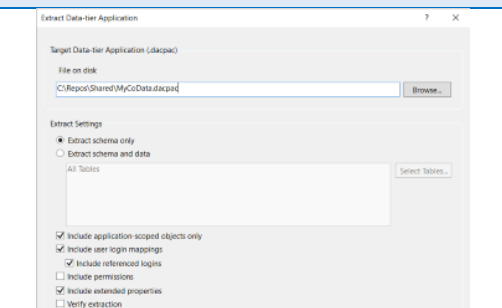
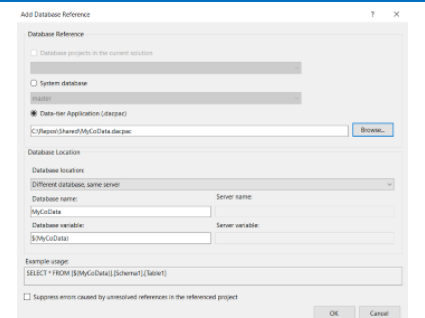
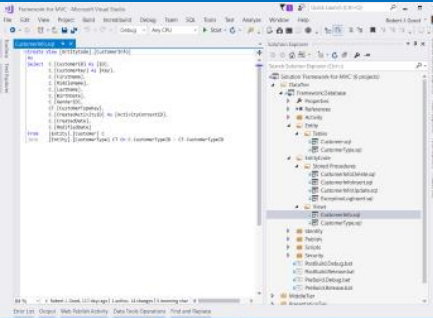
The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'FrameworkData' database is expanded, showing 'Database Diagrams', 'Tables', 'System Tables', 'FileTables', 'External Tables', 'Activity.Activity', 'Activity.ExceptionLog', 'Entity.Customer', and 'Entity.CustomerType'. The 'Tables' folder is expanded, showing 'Customer'. In the center, a T-SQL query is displayed:
`-- Customer`
`--`
`Use FrameworkData`
`Select Top 100 C.CustomerID, C.FirstName, C.MiddleName, C.LastName, C.BirthDate, C.CreatedDate, C.ModifiedDate,`
`CT.CustomerTypeID, ct.CustomerTypeName`
`From FrameworkData.Entity.Customer C`
`Left Join FrameworkData.Entity.CustomerType CT On C.CustomerTypeID = CT.CustomerTypeID`
Below the query, the 'Results' tab shows a table with 10 columns: CustomerID, FirstName, MiddleName, LastName, BirthDate, CreatedDate, ModifiedDate, CustomerTypeID, and CustomerTypeName. The table contains 3 rows of data.

	CustomerID	FirstName	MiddleName	LastName	BirthDate	CreatedDate	ModifiedDate	CustomerTypeID	CustomerTypeName
1	1	John	M	Smith	1968-05-20 00:00:00.000	2016-10-30 16:14:06.237	2016-10-30 16:14:06.237	2	Premium Customers
2	2	Siva	N	Kumar	1976-11-15 00:00:00.000	2016-10-30 16:14:06.237	2016-10-30 16:14:06.237	3	Standard Customers
3	3	Maki	L	Ishii	1973-06-30 00:00:00.000	2016-10-30 16:14:06.237	2016-10-30 16:14:06.237	4	Potential Customers

Re-wire Framework.Database SQL Views to connect to your SQL Tables

This procedure guides you through the process of re-wiring *Framework.Database.CustomerCode.CustomerInfo* view to pull data from your Person table. This is an example of a one-to-one swap of the FrameworkData.Customer table, with any table of yours that contains person data.

Important Tip: Keep the field names the same (use AS keyword) and column type the same. No code changes will be necessary. The existing Framework projects will work as if pulling from the FrameworkData.Customer table.

<p style="text-align: center; font-size: 2em;">①</p> <p style="text-align: center;">Open your Solution <i>i.e. Framework-for-MVC.sln</i></p> 	<p style="text-align: center; font-size: 2em;">②</p> <p style="text-align: center;">Connect to your database in SQL Object Explorer</p> 	<p style="text-align: center; font-size: 2em;">③</p> <p style="text-align: center;">Extract your database schema to a .dacpac file</p> 
<p>1. Open the <i>Framework-for-MVC.sln</i> Visual Studio solution file - Default: C:\Source\Framework-for-MVC.sln</p>	<p>2. Click View -> SQL Server Object Explorer 3. Enter connection info to your database 4. Click Connect to add the connection</p>	<p>5. In SQL Server Object Explorer, right-click your database -> click <i>Extract Data-tier Application</i> 6. Select: <i>Extract Schema Only</i> 7. Enter file-on-disk as: C:\Source\Shared\MyCoData.dacpac 8. Click OK to extract your schema to .dacpac</p>
<p style="text-align: center; font-size: 2em;">④</p> <p style="text-align: center;">Add a Database Reference to your .dacpac</p> 	<p style="text-align: center; font-size: 2em;">⑤</p> <p style="text-align: center;">Open View Views\CustomerCode\CustomerInfo.o.sql</p> 	<p style="text-align: center; font-size: 2em;">⑥</p> <p style="text-align: center;">Replace the SELECT with T-SQL that pulls data from your table</p>

For example...

If your table is: [MyCoData].[dbo].[Cust]
With fields: Cust_ID, F_Name, L_Name, B_Date

Change the SELECT to your [Cust] table...

```
Create View [CustomerCode].[CustomerInfo] As
Select
  C.[Cust_ID] As [ID],
  C.[F_Name] As [FirstName],
  C.[L_Name] As [LastName],
  C.[B_Date] As [BirthDate],
  ... (Alias Missing Fields Here) ...
From [MyCoData].[dbo].[Cust] C
```

9. In Solution Explorer, right-click your Framework.Database\Views\CustomerCode\CustomerInfo.sql

10. Navigate to and open Customer view: Framework.Database\Views\CustomerCode\CustomerInfo.sql

11. In CustomerInfo.cs, change the SELECT statement to pull data from your database
Note: Databases must be in same SQL instance

⑦

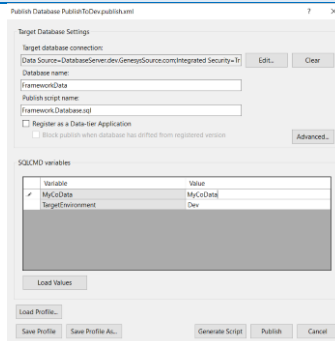
Alias any missing fields with Default Values

Alias missing fields for [Cust] example...

```
' As [MiddleName],  
-1 As [GenderID],  
1 As [ActivityContextID],  
'00000000-0000-0000-0000-000000000000' As  
[Key],  
'00000000-0000-0000-0000-000000000000' As  
[CustomerTypeKey],  
'01/01/1900' As [CreateDate],  
'01/01/1900' As [ModifiedDate]
```

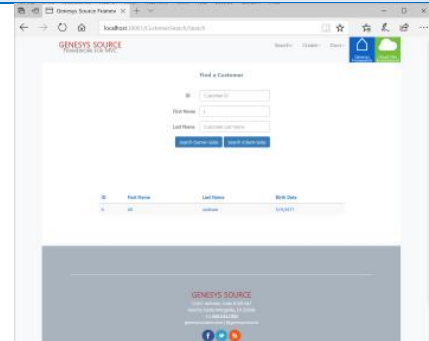
⑧

Publish FrameworkData to SQL Server



⑨

Run Framework.WebApp to pull your customer data



12. Alias all fields that do not have an equivalent in your customer data:

- Integer: -1
- String: ''
- Date: '01/01/1900'
- Guid: '00000000-0000-0000-0000-000000000000'

13. Open SSDT publish screen:

Framework.Database\Publish\PublishToDev.publish.xml

- Ensure *Target database connection* is correct
- Ensure MyCoData is set to the name of your database

14. Click *Generate Script* and review

15. Click *Publish* to push changes to SQL

16. Ensure connection string is correct:

Framework.WebApp\App_Data\ConnectionStrings.json

17. Right-click *Framework.WebApp* -> click *Set as Startup Project*

18. Press F5 or ▶ to run

- Should run this Url: <http://localhost:30001/>
Search screen & customer object now pulls your data

9: Pulling your data through the CustomerInfo object

Add a new Field/Property to the Customer object

This procedure walks you through the process of adding, changing or deleting an entity field. Including the column in the database, the data access object, the model and a MVC View.

①

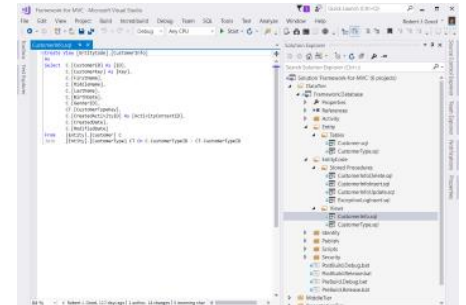
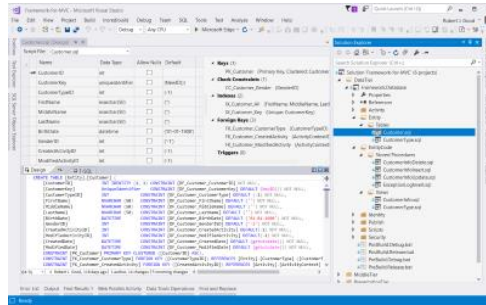
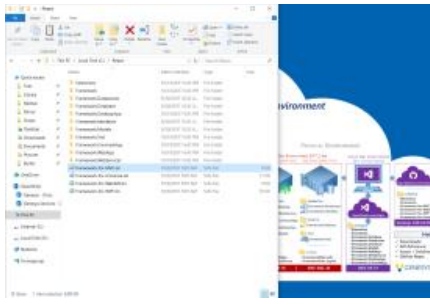
Open
Framework-for-MVC.sln

②

Open Table & Add Column
Tables\Customer\Customer.sql

③

Open View & Add Column
Views\CustomerCode\CustomerInfo.s
ql



1. Open the *Framework-for-MVC.sln* Visual Studio solution file
- Default: C:\Source\Framework-for-MVC.sln
2. Navigate to and open Customer table:
Framework.Database\Tables
\CustomerCode\CustomerInfo.sql
3. Add a new field: NickName
`[NickName] NVARCHAR (50) CONSTRAINT [DF_Customer_NickName] DEFAULT (') NOT NULL,`
4. Navigate to and open Customer view:
Framework.Database\Views
\CustomerCode\CustomerInfo.sql
5. Add the NickName field:
`C. [NickName],`

④

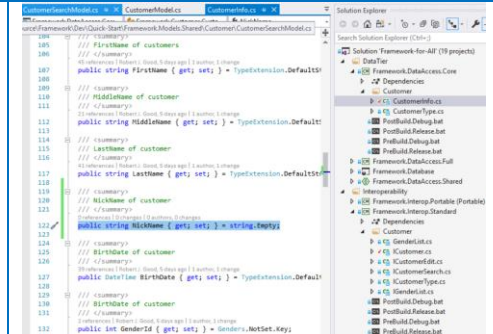
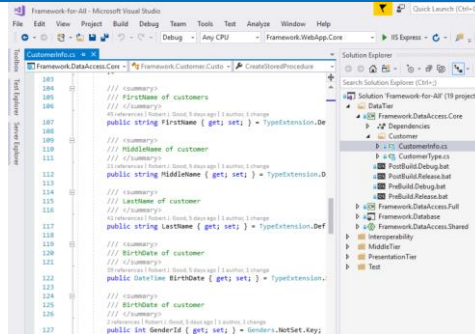
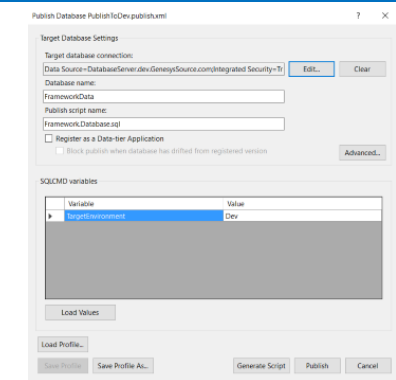
Publish *FrameworkData* to SQL Server

⑤

Open *CustomerInfo.cs* in *Framework.DataAccess*

⑥

Add NickName to *CustomerInfo*



6. Open SSDT publish screen:
Framework.Database\Publish
\PublishToDev.publish.xml
- Ensure *Target database connection* is correct
7. Click *Generate Script* and review
8. Click *Publish* to push changes to SQL
9. Open *CustomerInfo.cs*
- *Framework.DataAccess\Customer*
10. Add NickName by Copy/Paste the following property:
`public string NickName { get; set; } = string.Empty;`

⑦

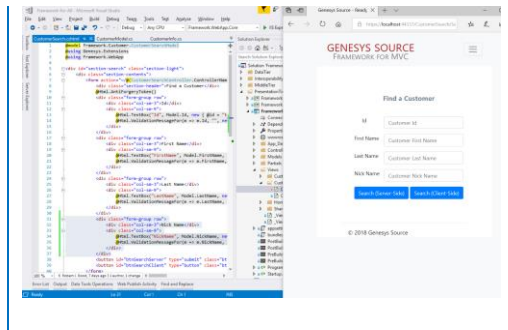
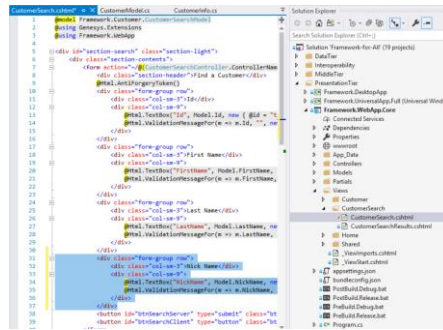
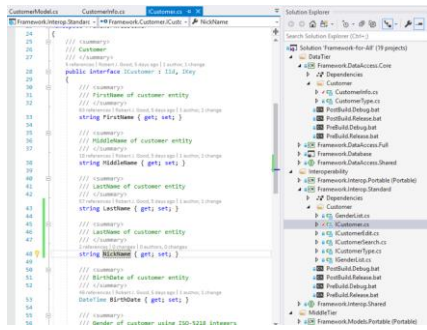
Add NickName to ICustomer & Models


⑧

Add NickName to *Framework.WebApp Search*

⑨

Run!



11. Open Framework.Interfaces\Customer\ICustomer.cs
12. Add NickName property as a string - Notice all classes that implement ICustomer throw an error requiring ICustomer.NickName
13. Add NickName to all dependent models (CustomerModel and CustomerSearchModel)
14. Open Framework.WebApp\Views\CustomerSearch\CustomerSearchResults.cshtml
15. Add Nick Name to table header and body
16. Double-check the connection string, to make sure it is pointing to the proper database
17. Right-click Framework.WebApp project - > click Set as StartUp Project
18. Press F5 or  to run

Publishing the Genesys Framework to IIS and SQL Server

For the Genesys Framework to function in your dev or production environments, you minimally need:

1. Framework.Database project published to a SQL Server or SQL Express
2. At least one Presentation Tier project, such as Framework.WebApp, published to an IIS Server

Publishing Framework.Database (SSDT) to a SQL Server

This procedure describes publishing the Framework.Database SSDT project to your SQL Server. The Framework.Database project holds all database objects for the FrameworkData database. Including tables, schemas, logins, users, views, stored procedures, etc.

①

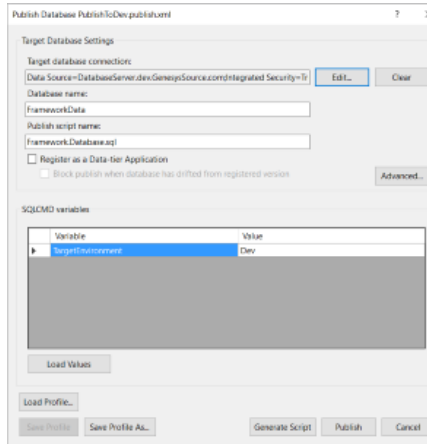
Build

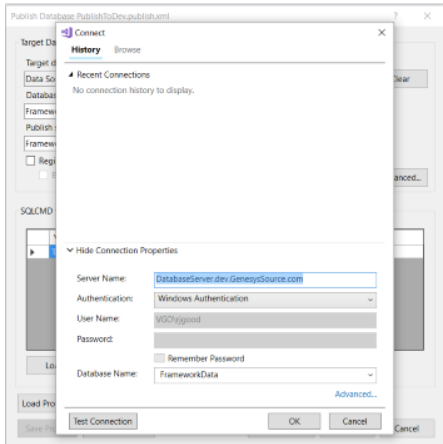
②

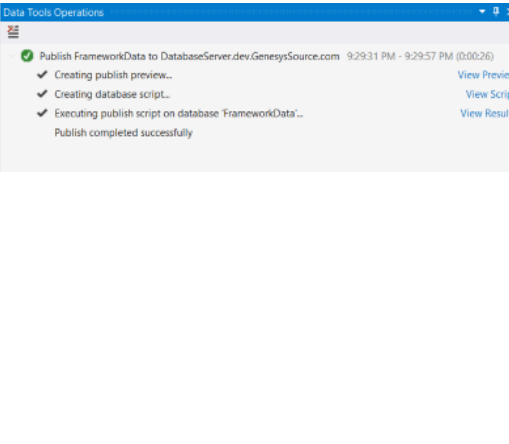
Set SQL Server

③

Publish





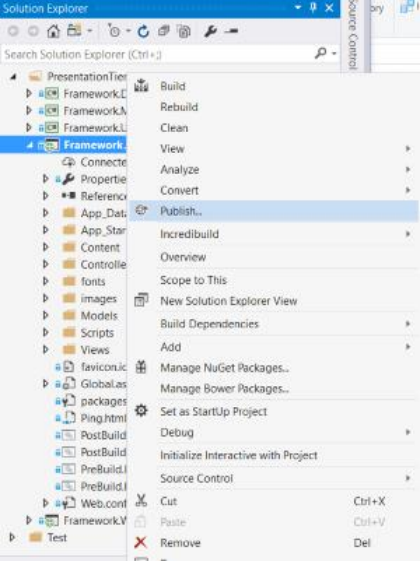
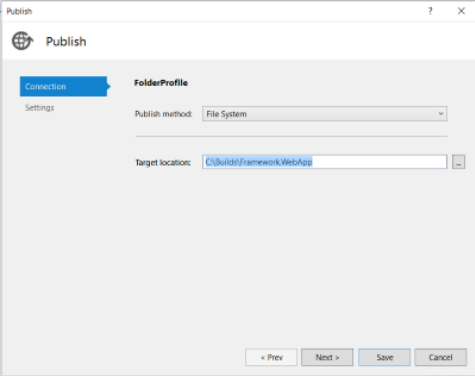
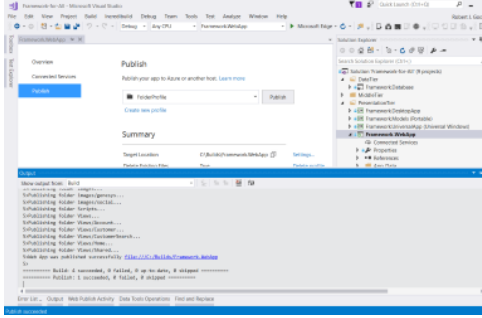


1. Open the framework solution you wish to publish, for example: C:\Repos\Framework-for-Mvc.sln	4. Click <i>Edit</i> button	8. Click <i>Generate Script</i> to see the change script (no database changes will be applied.)
2. Build <i>Framework.Database</i> project to ensure no errors	5. Change <i>Server Name</i> field to be the name of your SQL Server (i.e. Dev-Sql-16)	9. Click <i>Publish</i> to apply changes to the <i>FrameworkData</i> database
3. Open the Dev publish file \Publish\PublishToDev.publish.xml	6. Click <i>OK</i>	Your <i>Framework.Database</i> project is now in sync with your SQL Server
	7. Click the <i>Save Profile</i> button to save your changes	

10: Publishing Framework.Database (SSDT) to SQL Server

Publishing Framework.WebApp (MVC) to an IIS Web Server

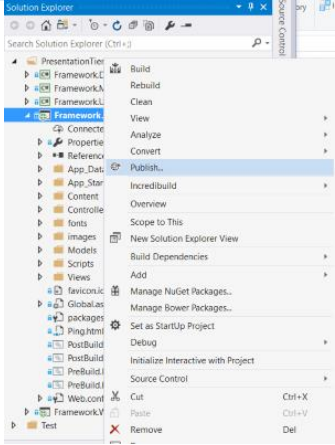
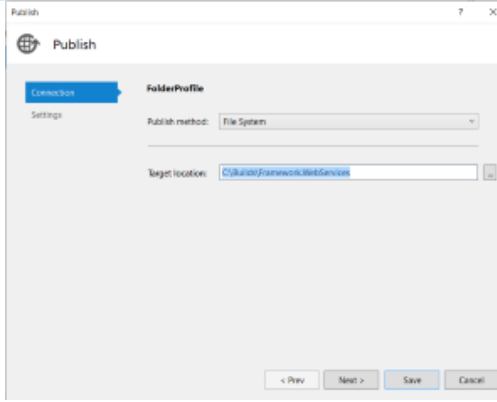
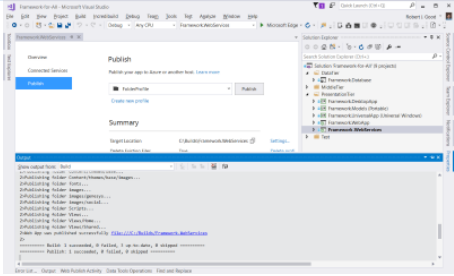
This procedure outlines how to publish the *Framework.WebApp* ASP.NET MVC project from Visual Studio to the IIS Web Server.

①	②	③
Build	Set Target Path	Publish
		
<ol style="list-style-type: none">1. Open the MVC solution you wish to publish, for example: <code>C:\Repos\Framework-for-Mvc.sln</code>2. Build the solution to ensure no errors3. Right-click the project and click <i>Publish</i>	<ol style="list-style-type: none">4. Click the <i>Settings...</i> link in the <i>Publish</i> window5. Change Target Location to be the path to your web project (default is local drive), for example: <code>\\Dev-Web-16\WebSites\Framework.WebApp</code>6. Click <i>Save</i>	<ol style="list-style-type: none">7. Click <i>Publish</i> to publish the project to your development web server <p>The MVC project has now been published to your IIS Web Server.</p>

11: Publishing Framework.WebApp (MVC) to IIS Web Server

Publishing Framework.WebServices (Web API) to an IIS Web Server

This procedure outlines how to publish the *Framework.WebServices* ASP.NET Web API project from Visual Studio to the IIS Web Server.

① Build	② Set Target Path	③ Publish
		
<ol style="list-style-type: none">1. Open the MVC solution you wish to publish, for example: <code>C:\Repos\Framework-for-Mvc.sln</code>2. Build the solution to ensure no errors3. Right-click the project and click <i>Publish</i>	<ol style="list-style-type: none">4. Click the <i>Settings...</i> link in the <i>Publish</i> window5. Change Target Location to be the path to your web project (default is local drive), for example: <code>\\Dev-Web-16\WebSites\Framework.WebServices</code>6. Click <i>Save</i>	<ol style="list-style-type: none">7. Click <i>Publish</i> to publish the project to your development web server <p>The Web API project has now been published to your IIS Web Server.</p>

12: Publishing Framework.WebServices (Web API) to IIS Web Server

Tech Aspects of the Genesys Framework

The Genesys Framework is a .NET Framework and Core stack containing C#, EF, SSDT, T-SQL, MVC, Web API, WPF, UWP, JavaScript, CSS and HTML. This section aims to explain some key tech aspects, to enable you to run and alter the projects with minimal learning curve.

Database Connections in App_Data

HTML Code

Web Service Connections in App_Data

HTML Code

Data Access (EF Core) coupled to Database (SSDT) Views and Stored Procedures

HTML Code

Data Access project Repository and Data Objects

[HTML Code](#)

Can't cast? Copy with Fill<T> and FillRange<T>

[HTML Code](#)

Setup for 2-tier or Setup for n-tier

[HTML Code](#)

Why the Genesys Framework?

The Genesys Framework was built out of frustration with the Copy-paste Anti-pattern in our daily software engineering lives. Boomerang bugs, bloated classes, inconsistent coding standards made development slow and tedious. Most software engineers know of good practices, some have even built reusable stacks...but inevitably the project would not be approved or completed.

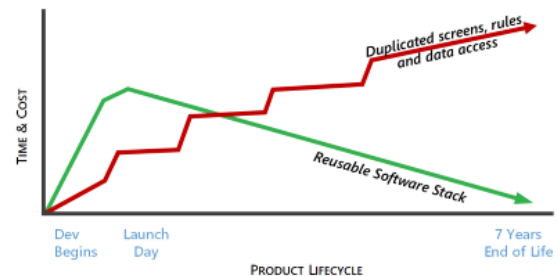
We set out to make code reuse fast, easy and with a minimal learning curve.

Why build reusable code?

Code reuse is an important theme in many of today's accepted software practices, such as N-tier and Object-oriented programming (OOP.)

Typically, reusable software stacks and services have low technical debt and are cheaper to maintain over time. Reusable code "settles" over time and costs decrease. Your return on investment (RoI) is greater with reusable software stacks

Conversely, the code duplication method tends to cost more over time, with high technical debt in the form of maintenance time and costs spiking per each duplicated item. Your costs go up over time, until the software is rewritten or retired.



The Genesys Framework offers n-tier, reusable business objects, with a low learning curve. Reusability without the cost of doing it yourself, and without the uncertainty of an untested new code base.

Why code full-stack, cross-platform business objects?

Microsoft .NET classes have a unique characteristic...they can run almost anywhere on any popular platform and run in any software tier. This allows a .NET entity class, like a Customer entity, to enjoy a 100% strongly-typed stack and consistency in properties and validation rules...in web sites, web services, native apps, CLR stored procedures and in class libraries.

With cross-platform full-stack entity objects, spelling errors and type errors show immediately as a compile error...in a stored procedure, in a data access C# file, in a MVC controller...everywhere that entity is used. Typing is maintained through the stack:

[Any SQL Data -> Framework.Database -> Framework.DataAccess -> Framework.Models -> Any .NET App](#)





Genesys Framework takes advantage of run-anywhere to enable any business object to run in Web, Services, Desktop and Mobile.

Take the Customer entity as an example:

- *CustomerInfo.cs*: Heavy Data Access Object (DAO) based on Entity Framework database-first. Supports CRUD-to-SQL methods of *Create()*, *Read(Expression)*, *Update()*, *Delete()*.
- *CustomerModel.cs*: Lightweight screen and transport models. This class is cross-platform and runs in MVC, Web API, UWP, WPF, Xamarin iOS, Xamarin Android, CLR Stored Procedures.
- *CrudViewModel<CustomerModel>*: MVVM ViewModel with CRUD-to-Services methods such as *CreateAsync()*, *ReadAsync(Expression)*, *UpdateAsync()* and *DeleteAsync()*.
- *customer.Serialize()*: JSON string is returned from any class that inherits *CrudEntity* or *ModelEntity*. This JSON can be controller generated and used by client-side web applications.

Getting Help

Have a question? Have a problem? Contact us anytime...

Contact Genesys Source	Help and Guidance	On Social
GENESYSSOURCE 22431 Antonio, Suite B160-843 Rancho Santa Margarita, CA 92688 +1 949.544.1900 www.genesyssource.com help@genesyssource.com	[Docs] docs.genesyssource.com [FAQs] Frequently Asked Questions [Q+A] Question and Answer [+/-] Report an Issue [Zip] Download Genesys Framework [Azure] Try Cloud Dev Environment Free	 http://twitter.com/GenesysSource  http://facebook.com/GenesysSource  http://GenesysSource.com/blog  http://GenesysSource.com/news/rss/1