SIF 3.0 Framework (.NET)

Version

**Author:** Rafidzal Rafiq,

**Revision:** 1.00 (draft)

**Published:** May 2016

Copyright © 2016,

Table of Contents

1. Introduction 3

1.1. Summary 3

1.2. Development environment pre-requisites 3

1.3. Third-party libraries 3

1.3.1. Web API 3

1.3.2. NHibernate 4

1.3.3. SQLite (core) 4

1.3.4. AutoMapper 4

1.3.5. log4net 4

1.4. Framework limitations 5

2. Installation and configuration 5

2.1. Download the code 5

2.1.1. Code directory 5

2.1.2. Data directory 6

2.1.3. Documentation directory 6

2.1.4. Scripts directory 6

2.1.5. SharedLibs directory 6

2.2. Load into Visual Studio 6

2.3. Configuration files 6

3. Concepts and terminology 7

3.1. Direct Environment 7

3.2. Service Consumer 7

3.3. Object Service Provider 7

3.4. Environment 7

3.5. Environment Provider 7

4. Solutions, Projects and Namespaces 7

4.1. Sif3Framework Solution 8

4.1.1. Sif.Framework Project 8

4.1.2. Sif.Framework.EnvironmentProvider Project 9

4.1.3. Sif.Framework.Tests Project 9

4.2. Sif3FrameworkDemo Solution 9

4.2.1. Sif.Framework.Demo.Au.Consumer Project 9

4.2.2. Sif.Framework.Demo.Au.Provider Project 9

4.2.3. Sif.Framework.Demo.Setup Project 9

4.2.4. Sif.Framework.Demo.Us.Consumer Project 9

4.2.5. Sif.Framework.Demo.Us.Provider Project 9

4.3. Sif3Specification Solution 9

5. Using this framework 9

6. Framework database 10

# Introduction

## Summary

The SIF 3.0 Framework for .NET is a basic development framework intended to assist in the creation of SIF compliant applications. It provides a simple, and intuitive, means of implementing providers and consumers of SIF data without the need to have read the SIF 3.0 specification. It encapsulates much of the functionality of the SIF 3.0 specification so that a developer can focus on the logic associated with data storage and retrieval.

As the SIF 3.0 Framework is based upon RESTful web services, some knowledge of ASP.NET Web API is required to fully understand how a provider of SIF data is implemented. Microsoft provides a comprehensive guide of Web API at <http://www.asp.net/web-api>. As Web API services, providers need to be deployed to IIS.

Consumers of SIF data can simply be implemented as executables without the need for deployment to IIS.

The SIF 3.0 Framework is an initiative driven by the National Schools Interoperability Program (NSIP) and implemented by Systemic Pty Ltd. It is an open-source framework that is available under the Apache License, Version 2.0 (<http://www.apache.org/licenses/LICENSE-2.0)> and is available free of charge with no licence purchase necessary.

## Development environment pre-requisites

The SIF 3.0 Framework was developed in C# using Microsoft .NET Framework 4.5 with Visual Studio 2015. As such, it is not guaranteed to work in any earlier variants of this development environment. Version 4.0.2+ of the .NET Framework is necessary to cater for the current release of Web API.

## Third-party libraries

The following third-party libraries are managed through Visual Studio with the use of NuGet. The rationale behind the selection of these libraries includes:

* They are well supported;
* They are well documented;
* They have good developer communities; and
* They are free.

This selection criterion was used in an attempt to cater for the majority. As these libraries may not suit all developers, some information is provided for the possible replacement of each.

### Web API

ASP.NET Web API is used to implement providers as RESTful web services. Alternative REST frameworks were investigated (such as ServiceStack), but it was decided to follow the Microsoft recommended path for the development of RESTful web services.

Web API is tightly integrated with this framework and would be difficult to replace. However, if necessary then the Sif.Framework.Providers namespace would be an appropriate place to start.

### NHibernate

NHIbernate is the Object Relational Mapping (ORM) tool used within this framework. It was used to ease the development of the persistence layer, and to make it easier to change the underlying database used by this framework.

To replace NHibernate, re-implement the classes in the Sif.Framework.Persistence.NHibernate namespace.

### SQLite (core)

The underlying database used to store all data associated with the SIF 3.0 Framework is SQLite. The main reason for its selection was that it is an embedded database that allows the downloaded demo projects to be run out of the box (without the need to point the demos to a specific database). It allows a developer to get started immediately without the need to create or start-up a database. However, it is expected that in a production system, this database be changed.

To change the database from SQLite, the SifFramework.cfg.xml file (all copies) needs to be updated to reference the new database. Knowledge of NHibernate is required, and comprehensive documentation can be found at <http://nhibernate.info>. As a starting point, configuration settings for the use of SQL Server LocalDB, SQL Server, Oracle and MySQL are provided. A good source of information for finding the appropriate connection strings for each database can be found at <http://www.connectionstrings.com/>.

### AutoMapper

AutoMapper is used to map model objects defined within the framework with data model objects defined by the SIF 3.0 specification. The main reason mapping is employed is because the SIF 3.0 data model objects that have been automatically generated from XML Schema definitions (as provided by the SIF 3.0 specification) do not provide “clean” POCOs.

To replace AutoMapper, re-implement the Sif.Framework.Service.Mapper.MapperFactory class.

It should be noted that issues were encountered when using the current release of AutoMapper (3.2.1). The version used in this framework is AutoMapper 3.1.1. To add this version of AutoMapper to new projects, within Visual Studio select the project and run the following command in the Package Manager Console.

PM> Install-Package AutoMapper -Version 3.1.1

### log4net

log4net is used for logging debug and error information. It is only used in the “core” libraries and Consumer code. Logging has been implemented so that log configuration settings are defined within the App.config file.

Logging has not been implemented, as yet, within Providers (which requires Tracing enabled). However, HTTP response messages returned from a Provider (Controller) should contain error details in the payload. To properly display these error messages, a HTTP client tool may be required (such as the Postman Chrome extension).

## Framework limitations

The following limitations apply to this version of the framework.

Only supports Basic authentication (not HMACSHA256).

Only supports SIF Infrastructure 3.0.1.

# Installation and configuration

## Download the code

The SIF 3.0 Framework can be downloaded from GitHub. To do so, use the following link and click on the “Download ZIP” button.

<https://github.com/nsip/Sif3Framework-dotNet>

Once downloaded, extract to an appropriate directory and you should see the following directory structure (or something very similar):

Sif3Framework-dotNet-master

‘-- Code

‘—Sif3Framework

‘—Sif3FrameworkDemo

‘—Sif3Specification

‘-- Data

‘---Documentation

‘---Scripts

‘---SharedLibs

### Code directory

The Code directory contains 3 separate Solutions. The Sif3Framework Solution contains the code for the framework itself.

The Sif3FrameworkDemo Solution contains some demo projects which are used to illustrate how this framework can be used. This project was created to help new developers get started with using this framework. The demo projects currently cover samples from the Australian and American Data Models.

The Sif3Specification Solution contains code that is generated based upon XML Schema data model definitions of the SIF 3.0 specification. The definitions files used to generate this code can be found under the Data directory (mentioned later). This Solution is mainly for internal use, and covers Data Models from Australia and America.

### Data directory

The Data directory contains a sub-directory of embedded database files and XML Schema files. The database files are used by the demo projects to illustrate various database configurations. When running the demo projects, these databases are used to store and retrieve the associated test data.

The XML Schema files define various data models (and versions) of the SIF 3.0 specification. The Infrastructure data model C# code generated from these XML Schema files are used by the framework itself. The SIF Object data model C# code generated from these XML Schema files are used by the demo projects, and are useful as guides for creating model objects that are SIF compliant.

### Documentation directory

The Documentation directory contains relevant documents associated with this framework (such as this developer’s guide).

### Scripts directory

The Scripts directory provides convenience MS-DOS batch files for running the demo applications, as well for the generation of C# code from XML Schema files. The code generation batch files need to be modified before use as they are dependent on the installation location of the .NET framework. The batch files for running the demo projects use relative file paths and should run out of the box (once all Solutions have been loaded into Visual Studio and their associated Projects have been built).

The Scripts directory also contains SQL DDL scripts for the creation of the database tables required by this framework. These DDLs are provided to illustrate various database configurations. These DDLs do not cover the database tables used by the demo projects.

### SharedLibs directory

The SharedLibs directory contains the latest versions of the Solution assemblies associated with this framework. This directory does not contain third-party libraries used by this framework. The management of third-party libraries is managed within Visual Studio with the use of NuGet.

## Load into Visual Studio

The Visual Studio Solutions for this framework can be found under the Code directory. This framework was developed using Visual Studio 2015, but should load in the Express version. Once opened in Visual Studio, no additional configuration should be required.

## Configuration files

The Environment Provider included in this framework requires an underlying database for its operation. To manage access to this database, the framework uses NHibernate. As such, the framework mandates the presence of the SifFramework.cfg.xml configuration file.

If running in a direct environment, Service Providers will need to use the same database as the Environment Provider. As such, the same SifFramework.cfg.xml configuration file is required for created Service Providers (such as providers of student data).

By default, SifFramework.cfg.xml has been configured to use a local (embedded) SQLite database. SQLite was selected (as the default) for the sake of convenience, and can be changed to a more appropriate database by updating the configuration file. Instructions on the use of NHibernate falls outside the scope of this document, but sample configurations are outlined within the file as a guide.

Any Consumers and Providers implemented from this framework require the presence of the SifFramework.config properties file. Specific details of the properties contained can be found in the “Sif3Framework .NET Demo Usage Guide.doc” document

# Concepts and terminology

This section is not intended to be a summarised version of the SIF 3.0 specification. Rather it will highlight a few key concepts that may make this framework easier to understand.

## Direct Environment

With SIF 2.x, SIF Agents communicated with each other via a ZIS. In SIF 3.0, there is the option to use either a Brokered Environment (the SIF 2.x ZIS equivalent) or a Direct Environment (peer to peer communication). This framework currently only supports a Direct Environment.

## Service Consumer

A Service Consumer in SIF 3.0 is similar to a Subscriber in SIF 2.x. It should be noted, though, that in a direct environment (which this framework operates under), a Service Consumer generally makes requests for data rather than automatically receives it (as events for instance).

## Object Service Provider

An Object Service Provider in SIF 3.0 is similar to a Publisher in SIF 2.x. An obvious difference, though, is that the Object Service Provider is implemented as a RESTful web service.

## Functional Service Provider

A Functional Service Provider in SIF 3.2 is similar to an Object Service Provider. It provides Job objects (part of the infrastructure definition in SIF3.2) in the same way as an Object Service Provider, but also facilitates CRUD operations on the phases a Job contains.

## Environment

An Environment is a virtual container which manages the “state” of communication between a Service Consumer and an Object Service Provider. An Environment is required before a Service Consumer can interact with an Object Service Provider (and vice versa).

## Environment Provider

An Environment Provider manages the Environments used for communicating between a Service Consumer and Object Service Provider. An Environment Provider is only required in a Direct Environment as, in a Brokered Environment, this is managed by a Broker. As such, this framework provides an implementation of an Environment Provider.

# Solutions, Projects and Namespaces

This section describes how the framework code is used for writing Service Consumers and Object Service Providers. Another source of information is the generated XML documentation provided as part of the framework code. The XML documentation is basic, but should be sufficient to get you going once you have read through this developer’s guide.

## Sif3Framework Solution

This Solution contains the core functionality associated with the SIF 3.0 Framework separated into 3 Projects.

### Sif.Framework Project

This Project is a class library that contains the appropriate classes for implementing Service Consumers and Object Service Providers.

#### Sif.Framework.Consumers Namespace

This Namespace contains the base code used for implementing Service Consumers.

#### Sif.Framework.Controllers Namespace

Deprecated Namespace used specifically for the Environment Provider. This namespace will be removed in future releases in-lieu of the Sif.Framework.Providers namespace.

#### Sif.Framework.Extensions Namespace

This Namespace contains .NET extension methods used within the framework.

#### Sif.Framework.Model Namespace

This Namespace contains the Infrastructure data model classes used by the framework. As this framework has been developed to be independent of SIF locale, it does not contain references to any SIF Object data models.

#### Sif.Framework.Persistence Namespace

This Namespace contains the persistence layer used by the Environment Provider for storing and retrieving Environment state.

#### Sif.Framework.Providers Namespace

This Namespace contains the base code used for implementing Providers (e.g. Object Service Providers) as Web API Controllers.

#### Sif.Framework.Service Namespace

This Namespace contains the business logic layer used by the framework for such operations as storage and retrieval of Infrastructure objects, and mapping of objects between the web service layer and persistence layer.

#### Sif.Framework.Utils Namespace

This Namespace contains utility code for authentication and XML serialisation.

#### Sif.Framework.WebApi Namespace

This Namespace contains utility code specifically for integration with Web API technologies.

### Sif.Framework.EnvironmentProvider Project

This is a Web API Project that defines an Environment Provider. It is separate to the core Sif.Framework class library Project so that it can be deployed and run from an IIS instance.

### Sif.Framework.Tests Project

This Project is used internally for unit testing the Sif.Framework Project code.

If running the unit tests, please ensure that in Visual Studio the “Test > Test Settings > Keep Test Execution Engine Running” setting is unset. If set, there may be issues with SQLite database (files) being locked for subsequent tests.

## Sif3FrameworkDemo Solution

This Solution contains the demo projects used for illustrating framework usage.

### Sif.Framework.Demo.Au.Consumer Project

This Project is a standalone application that runs as a StudentPersonal Service Consumer.

### Sif.Framework.Demo.Au.Provider Project

This is a Web API Project that runs as a StudentPersonal Object Service Provider within IIS.

### Sif.Framework.Demo.Setup Project

This Project is a standalone application that, when run, configures a database with test data that can be used by the EnvironmentProvider, Sif.Framework.Demo.Au.Provider and Sif.Framework.Demo.Us.Provider Projects.

### Sif.Framework.Demo.Uk.Consumer Project

This Project is a standalone application that runs as a LearnerPersonal Service Consumer and a Payload Functional Service Consumer.

### Sif.Framework.Demo.Uk.Provider Project

This is a Web API Project that runs as a LearnerPersonal Object Service Provider and Payload Functional Service within IIS.

### Sif.Framework.Demo.Us.Consumer Project

This Project is a standalone application that runs as a K12Student Service Consumer.

### Sif.Framework.Demo.Us.Provider Project

This is a Web API Project that runs as a K12Student Object Service Provider within IIS.

## Sif3Specification Solution

This Solution contains projects for holding versions of the SIF Object and Infrastructure data models.

# Using this framework

For specific details on coding against this framework, refer to the “Sif3Framework .NET Demo Usage Guide.doc” document. The usage guide can be considered as an extension of this document. Both documents need to be read to gain a full understanding of this framework.

# Framework database

To be done.