Writing a .NET Wing

A .NET Wing is the most versatile means of extending MONAHRQ's functionality; it can perform the functions of all the other extensibility options, with the exception of Flutters. Creating a new .NET Wing involves,

1. Creating a new class library project
2. Referencing the MONAHRQ, NHibernate, and MEF dependencies listed in *Project Structure and Dependencies*
3. Defining a `WingModule` with any relevant installation logic (see ***Error! Reference source not found.***)
4. Implementing or overriding any desired functionality, such as creating a ***Error! Reference source not found.***, custom ***Error! Reference source not found.***, or a ***Error! Reference source not found.***

Project Structure and Dependencies

There are no specific requirements for the way a MONAHRQ plugin is structured; however, several assemblies must be referenced to provide access to shared types:

Assembly Name	Description
Monahrq.Infrastructure	Common data types and extensibility framework
Monahrq.sdk	Additional data types and UI
NHibernate	ORM used for most database interactions
FluentHibernate	Used for NHibernate SQL to CLR mappings
MEF & Prism	Extensibility framework

The WingModule Implementation

The key component of a .NET Wing is the `WingModule` implementation, which describes the plugin to MONAHRQ, provides installation and uninstallation logic, and gives the module the opportunity to react to events in MONAHRQ.

The `WingModule` type implements Prism's `IModule` interface.

WingModule Lifecycle

During the MONAHRQ startup sequence, modules are loaded from the `Modules` folder of the MONAHRQ program directory. Detected modules undergo the following initialization procedure:

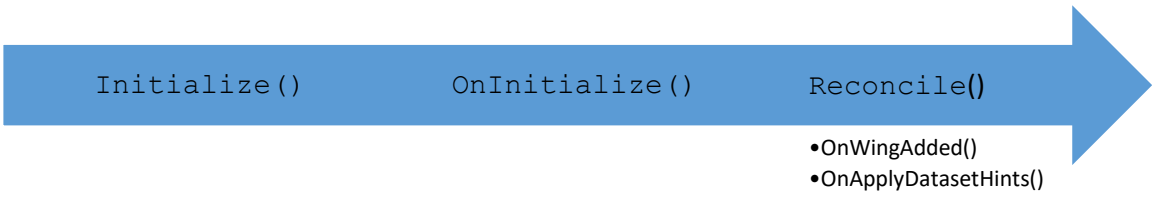

Figure 1 WingModule initialization process diagram

`Initialize()` is called, which posts a `MessageUpdateEvent` to the UI reporting that the module is loading; this calls the virtual method `OnInitialize()`

`OnInitialize()` calls `Reconcile()`

`Reconcile()` determines whether the module is new to the MONAHRQ database; if the module is new, `Reconcile()` creates a record for the module in the `Wings` database table and calls `OnWingAdded()` and `OnApplyDatasetHints()`

`OnWingAdded()` calls `Install()`, `InstallDb()`, `Update()`, and `UpdateDb()`

Types that derive from `WingModule` may alter their initialization procedures.

Hooking into MONAHRQ

Developers wishing to override any of MONAHRQ's default functions may wish to do so by listening for events raised by MONAHRQ. This is most often done by overriding the `OnInitialize()` method and using the `Events` property, an implementation of `IEventAggregator`, to subscribe to one or more events.

For an example of using events, see ***Error! Reference source not found.***. This example, taken from `Monahrq.Wing.Ahrq.AhrqModuleBase`, shows a module listening for a `WizardStepsRequestEvent`; this event is used by MONAHRQ to query modules for wizard steps to be used when bulk importing data for a data set.

Supported Events

The following events are used internally by MONAHRQ to send signals between components. Any event may be subscribed to, but publishing some of these events may cause unexpected problems. Events marked with an asterisk (*) may be safely raised by Wings.

Category	Event	Description
Database	`ConnectionFailedEvent`	Failed to connect to the MONAHRQ database
	`ConnectionSuccessEvent`	Successfully tested, created, or deleted a MONAHRQ database
Dataset	`DeleteEntryEvent`	A dataset is deleted, or an import is canceled or aborted
	`UpdateEntryEvent`	A dataset import is completed
General	`ErrorNotificationEvent`*	An exception was encountered
Import	`SimpleImportCompletedEvent`	A file import was completed
Measure	`MeasureFilterApplied`	A list of measures was filtered by dataset
	`TopicFilterApplied`	A list of measures was filtered by topic
	`TopicsUpdatedEvent`	A measure topic is added, updated, or deleted
Services	`ServiceErrorEvent`*	An exception was encountered in a service
UI	`DialogButtonClickEvent`	A dialog box button was clicked
	`DisableNavigationEvent`	Raised when UI controls should be disabled or re-enabled because of background processing

* Denotes an event that may be safely published by a Wing

	GenericNotificationEvent	Displays a notification to the Host User
	GenericNotificationExEvent	Displays a notification to the Host User
	MessageUpdateEvent*	Raised during MONAHRQ startup to update the splash screen text
	OpenContextualHelpContextEvent	Opens the help file to the specified topic
	SetContextualHelpContextEvent	Opens the help file to the specified topic
	ShutdownEvent	MONAHRQ is shutting down
	StatusbarUpdateEvent*	Updates the status bar of the MONAHRQ window
	UiMessageUpdateEventForeGround	Raised during MONAHRQ startup to update the splash screen text
	UpdateTabIndexEvent	The selected tab was changed
	WizardBackEvent	The back button was pressed in a wizard
	WizardCancelEvent	A wizard was canceled
	WizardCancellingEvent	The Host User requested cancelation of a wizard
	WizardStepsRequestEvent<T,TId,TId2>	A wizard is being opened and a list of steps is needed
Website	CancellingWebsitePublishingEvent	A website generation process was canceled
	MeasureFilterApplied	A filter was applied to a list of measures
	WebsiteCreatedOrUpdatedEvent	A new or existing website was saved
	WebsiteDeletedEvent	A website was deleted
	WebsitePublishEvent	Reports website generation progress