## Coding Style

All files should be cleaned up by CodeMaid (CodeMaid config)

### Naming

Public identifiers should be camel-case and start with Capital letter

Private identifiers should be camel-cased

Constants should be all caps and spaced with underscore

### Sleep()

Do not use the Thread.Sleep() method. You should use a method in the Core.Timing namespace or a Wait method in the browser or control. Sleep can be the source of flaky tests and can cause unnecessarily long tests.

### Conditionals

To explicitly express the intent and start and end of the conditionals, always use brackets.

### Style Changes

If you want to clean up a file just to fix style issues, file a bug first. If you are fixing a bug or enhancing some portion of a file, you can also do cleanup. Just don’t clean up a file just to clean it up without a bug ticket to track the change.

## Runner

* Set up runner environment from command line arguments
* Find test binaries (assembly finder or load from test manifest)
* Invoke test binaries
* Log results
* Clean up

## Feature Setup

During Feature setup the first thing we do is determine if the Feature should be ignored. We inspect the SpecFlow tags for the Feature and if we identify a tag that should be ignored, we end the test run for the Feature.

Next we clear the Cache to prevent artifacts from previous tests from affecting the current test run for the Feature.

To run the scenarios for the Feature we need test data. The *{FeatureName}*\_Data.xml file contains all of the test data necessary for the Scenarios in the Feature. We parse the title of the Feature from the SpecFlow FeatureInfo object to build the name of the x\_Data.xml file to load.

Once we have the file loaded we iterate over the data in the file to setup the test data. In each iteration:

* First, we make sure that the test data is cleaned from the database by calling a database delete script for the test data item.
* Next, we create fresh test data in the database by calling a database insert script for the test data item.
* The last step is to add the data item to cache for later use.

Lastly, we add an additional Feature configuration to Cache.

## Cache

The cache is basically a wrapper around some caching framework.

Suite Cache

Feature Cache

Feature Cache is a partition in the cache for state associated with Features. Test Data is stored in the Feature Cache is an in memory database of test data. There is a collection for each database table that is used in a test feature. During Feature Setup as test data is recreated in the database, the data is added to the collection for its corresponding table. This gives us strongly typed access to test data.

Scenario Cache

Scenario Cache allows you to store data across scenario method calls in your step files. This duplicates features provided by the SpecFlow ScenarioContext and was added to enable an easier refactor to remove the dependency on SpecFlow.