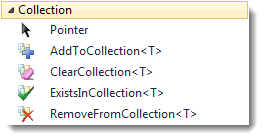
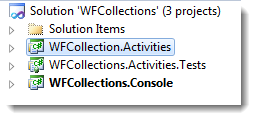
WF4 Collection Activities

This sample demonstrates the set of Windows Workflow Foundation Collection Activities provided in .Net Framework 4.



# Tour the sample

This sample consists of three projects

|  |  |  |
| --- | --- | --- |
| Project | Project Type | Description |
| WFCollection.Console | Workflow Console Application | Console app which contains an activity which demonstrates all 4 collection activities |
| WFCollection.Activities.Tests | Unit Test | Tests that demonstrate how collection activities are used in various scenarios |
| WFCollection.Activities | Activity Library | Contains the activities used by the tests and console projects |

**To quickly see the collection activities in action**

1. Set WFCollection.Console as the startup project
2. Run the application
3. You will see a workflow that manipulates a collection of Int32 values in action
4. To see the workflow open **CollectionActivities.xaml** from the **WorkflowCollection.Console** project.

# Scenarios

**To learn more about the collection activities in various scenarios review the code associated with the scenario**

|  |  |
| --- | --- |
| Scenario: How do I add an item to a collection? | |
| Given | A workflow with an out argument Result of type List(Of Customer) |
| When | The **AddToCollection** activity is invoked |
| Then | The customer is added to the collection |
| Example | **Project:** WFCollections.Tests  **File:** AddToCollectionsTest.cs |

|  |  |
| --- | --- |
| Scenario: What happens if I add a duplicate item to a collection? | |
| Given | A workflow with an out argument Result of type List(Of Customer) |
| When | The **AddToCollection** activity is invoked twice for the same customer |
| Then | The customer is added to the collection twice |
| Example | **Project:** WFCollections.Tests  **File:** AddToCollectionsTest.cs |

|  |  |
| --- | --- |
| Scenario: How to check if a value type (Int32) exists in the collection | |
| Given | A workflow with a Collection<Int32>  With the number 5 in the collection |
| When | The ExistsInCollection activity is invoked |
| Then | The result is true |
| Example | **Project:** WFCollections.Console  **File:** CollectionActivities.xaml |

|  |  |
| --- | --- |
| Scenario: Create an activity that ensures a customer exists in the list | |
| Given | A workflow with a variable of type List<Customer>  And an InArgument<Customer>  That does not exist in the list |
| When | The activity is invoked |
| Then | The customer is added to the list |
| Notes | If you use a complex type you should implement IEquatable<T> on your type and override the object.Equals and object.GetHashCode method to return a key property that uniquely identifies your instance. If you do not the object will be found only if the item you pass to the ExistsInCollection activity is the same reference as the object in the collection |
| Example | **Project:** WFCollections.Activities  **File:** EnsureCustomerIsInList.xaml  An activity that checks to see if the customer is in a list and adds them if they are not  **Project:** WFCollections.Tests  **File:** EnsureCustomerIsInListTests.cs  Unit tests that verify the activity and demonstrate how the behavior varies depending upon the type passed to it |

# Exceptions

The following exceptions are commonly encountered when using the collection activities.

|  |  |  |
| --- | --- | --- |
| Exception | Cause | Solution |
| InvalidOperationException  The property 'Collection' of 'AddToCollection<Int32>' is not initialized | The object passed to the Collection property is null (or Nothing) | Create a new collection before passing it to the collection activities |

# Q&A

Here are some common questions and answers about the collection activities

|  |  |
| --- | --- |
| Question | Answer |
| What happens if I add an item that already exists? | It depends on the implementation of the collection. The AddToCollection<T> activity simply calls the ICollection<T>.Add method to add an item to the collection. Some collections may check for duplicates and throw an exception while others might not. |
| What happens if I try to remove an item that does not exist? | The RemoveFromCollection activity returns a result of false |
| What happens if I use a complex type like a Customer object in my collection? How does it know which one to remove or see if it exists? | Exists uses ICollection<T>.Contains to test for existence of the object. The specific behavior depends on the implementation.  If you use a complex type you should implement IEquatable<T> on your type and override the object.Equals and object.GetHashCode method to return a key property that uniquely identifies your instance. If you do not the object will be found only if the item you pass to the ExistsInCollection activity is the same reference as the object in the collection |
| How do I initialize the collection in a Workflow using the designer? | In a XAML workflow (using the designer) you can initialize a Variable with an initialization expression or an Assign Activity.  With an Out Argument you must initialize the variable using an Assign Activity |
| What type should my collection be? Can I use string[]? | You cannot use arrays with collection activities.  You can use any type which implements ICollection<T> |
| Do the collection activities throw exceptions for things like duplicate items or removing items that don’t exist? | No ExistsInCollection and RemoveFromCollection return true or false depending upon the outcome.  The other methods typically do not throw exceptions but it is up to the implementation class as to how they behave. |
| If I write a collection class how can I be sure it will work with the collection activities? | Implement ICollection<T> correctly and your collection should work |