Cómo: Crear y ejecutar un flujo de trabajo de larga ejecución

**.NET Framework (current version)**

[Otras versiones](javascript:;)

Publicada: abril de 2016

Una de las características principales de Windows Workflow Foundation (WF) es la capacidad del tiempo de ejecución de conservar y descargar flujos de trabajo inactivos en una base de datos.Los pasos de [Cómo: Ejecutar un flujo de trabajo](https://msdn.microsoft.com/es-es/library/dd489463(v=vs.110).aspx) demostraron los fundamentos del flujo de trabajo que hospedaba mediante una aplicación de consola.Se mostraron ejemplos de cómo iniciar flujos de trabajo, de los controladores de ciclo de vida de los flujos de trabajo y de los marcadores de reanudación.Para demostrar la persistencia del flujo de trabajo con efectividad, es necesario un host de flujo de trabajo más complejo que admita el inicio y la reanudación de varias instancias de flujo de trabajo.En este paso del tutorial se muestra cómo crear una aplicación host de Windows Forms que admita iniciar y reanudar varias instancias de flujo de trabajo o la persistencia del flujo de trabajo, y que proporcione una base para características avanzadas como el seguimiento y el control de versiones mostrado en los siguientes pasos del tutorial.

|  |
| --- |
| **System_CAPS_noteNota** |
| Este paso y los pasos siguientes del tutorial usan los tres tipos de flujo de trabajo de [Cómo: Crear flujos de trabajo](https://msdn.microsoft.com/es-es/library/dd489437(v=vs.110).aspx).Si no completó los tres tipos, puede descargar una versión completada de los pasos de [Windows Workflow Foundation (WF45): tutorial de introducción](http://go.microsoft.com/fwlink/?LinkID=248976). |
| **System_CAPS_noteNota** |
| Para descargar una versión completada o consultar una descripción en vídeo del tutorial, vea [Windows Workflow Foundation (WF45): tutorial de introducción](http://go.microsoft.com/fwlink/?LinkID=248976). |

[En este tema](javascript:void(0))

* [Para crear la base de datos de persistencia](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_CreatePersistenceDatabase)
* [Para agregar la referencia a los ensamblados de DurableInstancing](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_AddReference)
* [Para crear el formulario de host de flujo de trabajo](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_CreateForm)
* [Para agregar las propiedades y métodos auxiliares del formulario](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_AddHelperMethods)
* [Para configurar el almacén de instancias, los controladores de ciclo de vida del flujo de trabajo y las extensiones](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_ConfigureWorkflowApplication)
* [Para habilitar el inicio y la reanudación de varios tipos de flujo de trabajo](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_WorkflowVersionMap)
* [Para iniciar un nuevo flujo de trabajo](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_StartWorkflow)
* [Para reanudar un flujo de trabajo](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_ResumeWorkflow)
* [Para finalizar un flujo de trabajo](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_TerminateWorkflow)
* [Para compilar y ejecutar la aplicación](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_BuildAndRun)

[Para crear la base de datos de persistencia](javascript:void(0))

1. Abra SQL Server Management Studio y conéctese al servidor local, por ejemplo **.\SQLEXPRESS**.Haga clic con el botón secundario en el nodo **Bases de datos** en el servidor local y seleccione **Nueva base de datos**.Asigne a la nueva base de datos el nombre**WF45GettingStartedTutorial**, acepte el resto de valores y seleccione **Aceptar**.

|  |
| --- |
| **System_CAPS_noteNota** |
| Asegúrese de que tiene el permiso **Crear base de datos** en el servidor local antes de crear la base de datos. |

1. Elija **Abrir**, **Archivo** en el menú **Archivo**.Busque la carpeta siguiente: C:\Windows\Microsoft.NET\Framework\4.0.30319\sql\en

Seleccione los dos archivos siguientes y, a continuación, haga clic en **Abrir**.

* + SqlWorkflowInstanceStoreLogic.sql
  + SqlWorkflowInstanceStoreSchema.sql

1. Elija **SqlWorkflowInstanceStoreSchema.sql** en el menú **Ventana**.Asegúrese de que **WF45GettingStartedTutorial** está seleccionado en la lista desplegable **Bases de datos disponibles** y elija **Ejecutar** en el menú **Consulta**.
2. Elija **SqlWorkflowInstanceStoreLogic.sql** en el menú **Ventana**.Asegúrese de que **WF45GettingStartedTutorial** está seleccionado en la lista desplegable **Bases de datos disponibles** y elija **Ejecutar** en el menú **Consulta**.

|  |
| --- |
| **System_CAPS_warningAdvertencia** |
| Es importante realizar los dos pasos anteriores en el orden correcto.Si las consultas se ejecutan de manera desordenada, aparecerán errores y la base de datos de persistencia no se configurará correctamente. |

[Para agregar la referencia a los ensamblados de DurableInstancing](javascript:void(0))

1. Haga clic con el botón secundario en **NumberGuessWorkflowHost** en el **Explorador de soluciones** y seleccione **Agregar referencia**.
2. Seleccione **Ensamblados** en la lista **Agregar referencia** y escriba **DurableInstancing** en el cuadro **Buscar ensamblados**.Esto filtrará los ensamblados y simplificará la selección de las referencias deseadas.
3. Active la casilla situada junto a **System.Activities.DurableInstancing** y **System.Runtime.DurableInstancing** de la lista **Resultados de la búsqueda** y haga clic en **Aceptar**.

[Para crear el formulario de host de flujo de trabajo](javascript:void(0))

|  |
| --- |
| **System_CAPS_noteNota** |
| Los pasos de este procedimiento describen cómo agregar y configurar el formulario manualmente.Si lo desea, puede descargar los archivos de solución para el tutorial y agregar el formulario completo al proyecto.Para descargar los archivos del tutorial, vea [Windows Workflow Foundation (WF45): tutorial de Introducción](http://go.microsoft.com/fwlink/?LinkID=248976).Una vez que se han descargado los archivos, haga clic con el botón secundario en**NumberGuessWorkflowHost** y elija **Agregar referencia**.Agregue una referencia a **System.Windows.Forms** y **System.Drawing**.Estas referencias se agregan automáticamente si agrega un nuevo formulario desde el menú **Agregar**, **Nuevo elemento**, pero se deben agregar manualmente al importar un formulario.Una vez que las referencias se han agregado, haga clic con el botón secundario en**NumberGuessWorkflowHost** en **Explorador de soluciones** y elija **Agregar**, **Elemento existente**.Busque la carpeta Form en los archivos de proyecto, seleccione **WorkflowHostForm.cs** (o **WorkflowHostForm.vb**) y haga clic en **Agregar**.Si elige importar el formulario, puede saltar a la siguiente sección, [Para agregar las propiedades y métodos auxiliares del formulario](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx#BKMK_AddHelperMethods). |

1. Haga clic con el botón secundario en **NumberGuessWorkflowHost** en el **Explorador de soluciones** y elija **Agregar**, **Nuevo elemento**.
2. En la lista de plantillas **Instalado**, elija **Windows Forms**, escriba **WorkflowHostForm** en el cuadro **Nombre** y haga clic en **Agregar**.
3. Configure las siguientes propiedades en el formulario.

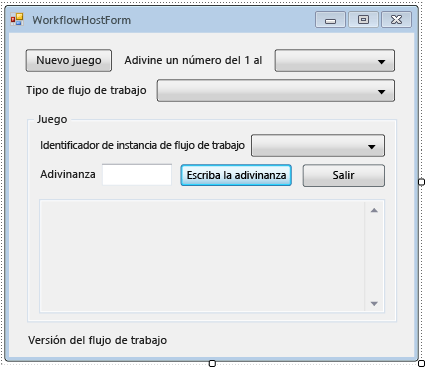
|  |  |
| --- | --- |
| **Propiedad** | **Valor** |
| FormBorderStyle | FixedSingle |
| MaximizeBox | False |
| Tamaño | 400, 420 |

1. Agregue los siguientes controles al formulario en el orden especificado y configure las propiedades como dirigidas.

|  |  |  |
| --- | --- | --- |
| **Control** | **Propiedad** | **Valor** |
| **Button** | Name | NewGame |
| Ubicación | 13, 13 |
| Tamaño | 75, 23 |
| Text | New Game |
| **Etiqueta** | Ubicación | 94, 18 |
| Text | Adivine un número del 1 al |
| **ComboBox** | Name | NumberRange |
| DropDownStyle | DropDownList |
| Items | 10  100  1000 |
| Ubicación | 228, 12 |
| Tamaño | 143, 21 |
| **Etiqueta** | Ubicación | 13, 43 |
| Text | Tipo de flujo de trabajo |
| **ComboBox** | Name | WorkflowType |
| DropDownStyle | DropDownList |
| Items | StateMachineNumberGuessWorkflow  FlowchartNumberGuessWorkflow  SequentialNumberGuessWorkflow |
| Ubicación | 94, 40 |
| Tamaño | 277, 21 |
| **Etiqueta** | Name | WorkflowVersion |
| Ubicación | 13, 362 |
| Text | Versión del flujo de trabajo |
| **GroupBox** | Ubicación | 13, 67 |
| Tamaño | 358, 287 |
| Text | Game |
| |  | | --- | | **System_CAPS_noteNota** | | Se incluyen los controles siguientes en **GroupBox**.Cuando los agrega al formulario, elimínelos de **GroupBox**. | | | |
| **Etiqueta** | Ubicación | 7, 20 |
| Text | Identificador de instancia de flujo de trabajo |
| **ComboBox** | Name | InstanceId |
| DropDownStyle | DropDownList |
| Ubicación | 121, 17 |
| Tamaño | 227, 21 |
| **Etiqueta** | Ubicación | 7, 47 |
| Text | Guess |
| **TextBox** | Name | Guess |
| Ubicación | 50, 44 |
| Tamaño | 65, 20 |
| **Button** | Name | EnterGuess |
| Ubicación | 121, 42 |
| Tamaño | 75, 23 |
| Text | Enter Guess |
| **Button** | Name | QuitGame |
| Ubicación | 274, 42 |
| Tamaño | 75, 23 |
| Text | Quit |
| **TextBox** | Name | WorkflowStatus |
| Ubicación | 10, 73 |
| Multiline | True |
| ReadOnly | True |
| ScrollBars | Vertical |
| Tamaño | 338, 208 |

1. Establezca la propiedad **AcceptButton** del formulario en **EnterGuess**.

En el siguiente ejemplo se muestra el formulario completado.



[Para agregar las propiedades y métodos auxiliares del formulario](javascript:void(0))

Los pasos de esta sección agregan propiedades y métodos auxiliares a la clase de formulario que configuran la interfaz de usuario del formulario para que se admita la ejecución y la reanudación de flujos de trabajo de acierto de números.

1. Haga clic con el botón secundario en **WorkflowHostForm** en **Explorador de soluciones** y elija **Ver código**.
2. Agregue las siguientes instrucciones **using** (o **Imports**) al principio del archivo con las demás instrucciones **using** (o **Imports**).

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-1)

using System.Windows.Forms;

using System.Activities.DurableInstancing;

using System.Activities;

using System.Data.SqlClient;

using System.IO;

1. Agregue las siguientes declaraciones del miembro a la clase **WorkflowHostForm**.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-2)

const string connectionString = "Server=.\\SQLEXPRESS;Initial Catalog=WF45GettingStartedTutorial;Integrated Security=SSPI";

SqlWorkflowInstanceStore store;

bool WorkflowStarting;

|  |
| --- |
| **System_CAPS_noteNota** |
| Si la cadena de conexión es diferente, actualice connectionString para consultar la base de datos. |

1. Agregue una propiedad WorkflowInstanceId a la clase WorkflowFormHost.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-3)

public Guid WorkflowInstanceId

{

get

{

return InstanceId.SelectedIndex == -1 ? Guid.Empty : (Guid)InstanceId.SelectedItem;

}

}

El cuadro combinado InstanceIdmuestra una lista de identificadores de instancia de flujo de trabajo persistentes y la propiedadWorkflowInstanceId devuelve el flujo de trabajo actualmente seleccionado.

1. Agregue un controlador para el evento del formulario Load.Para agregar el controlador, cambie a **Vista de diseño** para el formulario, haga clic en el icono **Eventos** en la parte superior de la ventana **Propiedades** y haga doble clic en **Cargar**.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-4)

private void WorkflowHostForm\_Load(object sender, EventArgs e)

{

}

1. Agregue el código siguiente a WorkflowHostForm\_Load.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-5)

// Initialize the store and configure it so that it can be used for

// multiple WorkflowApplication instances.

store = new SqlWorkflowInstanceStore(connectionString);

WorkflowApplication.CreateDefaultInstanceOwner(store, null, WorkflowIdentityFilter.Any);

// Set default ComboBox selections.

NumberRange.SelectedIndex = 0;

WorkflowType.SelectedIndex = 0;

ListPersistedWorkflows();

Cuando el formulario se carga, se configura SqlWorkflowInstanceStore, los intervalos y los cuadros combinados de tipo de flujo de trabajo se establecen en valores predeterminados y las instancias de flujo de trabajo persistentes se agregan al cuadro combinadoInstanceId.

1. Agregue un controlador SelectedIndexChanged para InstanceId.Para agregar el controlador, cambie a **Vista de diseño** para el formulario, seleccione el cuadro combinado InstanceId, haga clic en el icono **Eventos** en la parte superior de la ventana **Propiedades** y haga doble clic en **SelectedIndexChanged**.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-6)

private void InstanceId\_SelectedIndexChanged(object sender, EventArgs e)

{

}

1. Agregue el código siguiente a InstanceId\_SelectedIndexChanged.Siempre que el usuario selecciona un flujo de trabajo mediante el cuadro combinado, este controlador actualiza la ventana de estado.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-7)

if (InstanceId.SelectedIndex == -1)

{

return;

}

// Clear the status window.

WorkflowStatus.Clear();

// Get the workflow version and display it.

// If the workflow is just starting then this info will not

// be available in the persistence store so do not try and retrieve it.

if (!WorkflowStarting)

{

WorkflowApplicationInstance instance =

WorkflowApplication.GetInstance(this.WorkflowInstanceId, store);

WorkflowVersion.Text =

WorkflowVersionMap.GetIdentityDescription(instance.DefinitionIdentity);

// Unload the instance.

instance.Abandon();

}

1. Agregue el siguiente método ListPersistedWorkflows a la clase de formulario.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-8)

using (SqlConnection localCon = new SqlConnection(connectionString))

{

string localCmd =

"Select [InstanceId] from [System.Activities.DurableInstancing].[Instances] Order By [CreationTime]";

SqlCommand cmd = localCon.CreateCommand();

cmd.CommandText = localCmd;

localCon.Open();

using (SqlDataReader reader = cmd.ExecuteReader(CommandBehavior.CloseConnection))

{

while (reader.Read())

{

// Get the InstanceId of the persisted Workflow

Guid id = Guid.Parse(reader[0].ToString());

InstanceId.Items.Add(id);

}

}

}

ListPersistedWorkflows consulta el almacén de instancias con respecto a las instancias de flujo de trabajo persistentes y agrega los identificadores de instancia al cuadro combinado cboInstanceId.

1. Agregue el siguiente método UpdateStatus y el delegado correspondiente a la clase de formulario.Este método actualiza la ventana de estado en el formulario con el estado del flujo de trabajo en funcionamiento.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-9)

private delegate void UpdateStatusDelegate(string msg);

public void UpdateStatus(string msg)

{

// We may be on a different thread so we need to

// make this call using BeginInvoke.

if (InvokeRequired)

{

BeginInvoke(new UpdateStatusDelegate(UpdateStatus), msg);

}

else

{

if (!msg.EndsWith("\r\n"))

{

msg += "\r\n";

}

WorkflowStatus.AppendText(msg);

WorkflowStatus.SelectionStart = WorkflowStatus.Text.Length;

WorkflowStatus.ScrollToCaret();

}

}

1. Agregue el siguiente método GameOver y el delegado correspondiente a la clase de formulario.Cuando se completa un flujo de trabajo, este método actualiza la interfaz de usuario del formulario mediante la eliminación del identificador de instancia del flujo de trabajo completado en el cuadro combinado **InstanceId**.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-10)

private delegate void GameOverDelegate();

private void GameOver()

{

if (InvokeRequired)

{

BeginInvoke(new GameOverDelegate(GameOver));

}

else

{

// Remove this instance from the combo box

InstanceId.Items.Remove(InstanceId.SelectedItem);

InstanceId.SelectedIndex = -1;

}

}

[Para configurar el almacén de instancias, los controladores de ciclo de vida del flujo de trabajo y las extensiones](javascript:void(0))

1. Agregue un método ConfigureWorkflowApplication a la clase de formulario.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-11)

private void ConfigureWorkflowApplication(WorkflowApplication wfApp)

{

}

Este método configura **WorkflowApplication**, agrega las extensiones deseadas y agrega los controladores para los eventos de ciclo de vida de flujo de trabajo.

1. En ConfigureWorkflowApplication, especifique **SqlWorkflowInstanceStore** para la **WorkflowApplication**.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-12)

// Configure the persistence store.

wfApp.InstanceStore = store;

1. A continuación, cree una instancia **StringWriter** y agréguela a la colección **Extensions** de **WorkflowApplication**.Cuando se agrega**StringWriter** a las extensiones, captura todos los resultados de actividad **WriteLine**. Cuando el flujo de trabajo se vuelve inactivo, el resultado de **WriteLine** puede extraerse desde **StringWriter** y mostrarse en el formulario.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-13)

// Add a StringWriter to the extensions. This captures the output

// from the WriteLine activities so we can display it in the form.

StringWriter sw = new StringWriter();

wfApp.Extensions.Add(sw);

1. Agregue el controlador siguiente para el evento **Completed**.Cuando un flujo de trabajo se ha completado con éxito, el número de intentos que han sido necesarios para acertar el número se muestra en la ventana de estado.Si el flujo de trabajo finaliza, se muestra la información de excepción que ha provocado la finalización.Al final del controlador se llama al método GameOver, que quita el flujo de trabajo completado de la lista de flujos de trabajo.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-14)

wfApp.Completed = delegate(WorkflowApplicationCompletedEventArgs e)

{

if (e.CompletionState == ActivityInstanceState.Faulted)

{

UpdateStatus(string.Format("Workflow Terminated. Exception: {0}\r\n{1}",

e.TerminationException.GetType().FullName,

e.TerminationException.Message));

}

else if (e.CompletionState == ActivityInstanceState.Canceled)

{

UpdateStatus("Workflow Canceled.");

}

else

{

int Turns = Convert.ToInt32(e.Outputs["Turns"]);

UpdateStatus(string.Format("Congratulations, you guessed the number in {0} turns.", Turns));

}

GameOver();

};

1. Agregue los siguientes controladores **Aborted** y **OnUnhandledException**.No se llama al método de GameOver desde el controlador**Aborted** porque cuando una instancia de flujo de trabajo se anula, no se finaliza y es posible reanudar la instancia posteriormente.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-15)

wfApp.Aborted = delegate(WorkflowApplicationAbortedEventArgs e)

{

UpdateStatus(string.Format("Workflow Aborted. Exception: {0}\r\n{1}",

e.Reason.GetType().FullName,

e.Reason.Message));

};

wfApp.OnUnhandledException = delegate(WorkflowApplicationUnhandledExceptionEventArgs e)

{

UpdateStatus(string.Format("Unhandled Exception: {0}\r\n{1}",

e.UnhandledException.GetType().FullName,

e.UnhandledException.Message));

GameOver();

return UnhandledExceptionAction.Terminate;

};

1. Agregue el siguiente controlador **PersistableIdle**.Este controlador recupera la extensión **StringWriter** que se había agregado, extrae la salida de las actividades de **WriteLine** y la muestra en la ventana de estado.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-16)

wfApp.PersistableIdle = delegate(WorkflowApplicationIdleEventArgs e)

{

// Send the current WriteLine outputs to the status window.

var writers = e.GetInstanceExtensions<StringWriter>();

foreach (var writer in writers)

{

UpdateStatus(writer.ToString());

}

return PersistableIdleAction.Unload;

};

La enumeración [PersistableIdleAction](https://msdn.microsoft.com/es-es/library/system.activities.persistableidleaction(v=vs.110).aspx) tiene tres valores: [None](https://msdn.microsoft.com/es-es/library/system.activities.persistableidleaction(v=vs.110).aspx), [Persist](https://msdn.microsoft.com/es-es/library/system.activities.persistableidleaction(v=vs.110).aspx) y [Unload](https://msdn.microsoft.com/es-es/library/system.activities.persistableidleaction(v=vs.110).aspx). [Persist](https://msdn.microsoft.com/es-es/library/system.activities.persistableidleaction(v=vs.110).aspx) hace que el flujo de trabajo sea persistente pero no hace que se descargue. [Unload](https://msdn.microsoft.com/es-es/library/system.activities.persistableidleaction(v=vs.110).aspx) hace que el flujo de trabajo sea persistente y se descargue.

El siguiente ejemplo es el método ConfigureWorkflowApplication completado.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-17)

private void ConfigureWorkflowApplication(WorkflowApplication wfApp)

{

// Configure the persistence store.

wfApp.InstanceStore = store;

// Add a StringWriter to the extensions. This captures the output

// from the WriteLine activities so we can display it in the form.

StringWriter sw = new StringWriter();

wfApp.Extensions.Add(sw);

wfApp.Completed = delegate(WorkflowApplicationCompletedEventArgs e)

{

if (e.CompletionState == ActivityInstanceState.Faulted)

{

UpdateStatus(string.Format("Workflow Terminated. Exception: {0}\r\n{1}",

e.TerminationException.GetType().FullName,

e.TerminationException.Message));

}

else if (e.CompletionState == ActivityInstanceState.Canceled)

{

UpdateStatus("Workflow Canceled.");

}

else

{

int Turns = Convert.ToInt32(e.Outputs["Turns"]);

UpdateStatus(string.Format("Congratulations, you guessed the number in {0} turns.", Turns));

}

GameOver();

};

wfApp.Aborted = delegate(WorkflowApplicationAbortedEventArgs e)

{

UpdateStatus(string.Format("Workflow Aborted. Exception: {0}\r\n{1}",

e.Reason.GetType().FullName,

e.Reason.Message));

};

wfApp.OnUnhandledException = delegate(WorkflowApplicationUnhandledExceptionEventArgs e)

{

UpdateStatus(string.Format("Unhandled Exception: {0}\r\n{1}",

e.UnhandledException.GetType().FullName,

e.UnhandledException.Message));

GameOver();

return UnhandledExceptionAction.Terminate;

};

wfApp.PersistableIdle = delegate(WorkflowApplicationIdleEventArgs e)

{

// Send the current WriteLine outputs to the status window.

var writers = e.GetInstanceExtensions<StringWriter>();

foreach (var writer in writers)

{

UpdateStatus(writer.ToString());

}

return PersistableIdleAction.Unload;

};

}

[Para habilitar el inicio y la reanudación de varios tipos de flujo de trabajo](javascript:void(0))

Para reanudar una instancia de flujo de trabajo, el host tiene que proporcionar la definición de flujo de trabajo.En este tutorial hay tres tipos de flujo de trabajo, y los pasos siguientes del tutorial presentan varias versiones de estos tipos. **WorkflowIdentity** proporciona una manera para que una aplicación de host asocie información de identificación con una instancia de flujo de trabajo persistente.Los pasos de esta sección muestran cómo crear una clase de utilidad para contribuir a la asignación de la identidad de flujo de trabajo desde una instancia de flujo de trabajo persistente a la definición de flujo de trabajo correspondiente.Para obtener más información sobre**WorkflowIdentity** y la creación de versiones, vea [Usar WorkflowIdentity y el control de versiones](https://msdn.microsoft.com/es-es/library/hh314054(v=vs.110).aspx).

1. Haga clic con el botón secundario en **NumberGuessWorkflowHost** en el **Explorador de soluciones** y elija **Agregar**, **Clase**.Escriba**WorkflowVersionMap** en el cuadro **Nombre** y haga clic en **Agregar**.
2. Agregue las siguientes instrucciones **using** (o **Imports**) al principio del archivo con las demás instrucciones **using** o **Imports**.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-18)

using NumberGuessWorkflowActivities;

using System.Activities;

1. Reemplace la declaración de clase WorkflowVersionMap por la declaración siguiente.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-19)

public static class WorkflowVersionMap

{

static Dictionary<WorkflowIdentity, Activity> map;

// Current version identities.

static public WorkflowIdentity StateMachineNumberGuessIdentity;

static public WorkflowIdentity FlowchartNumberGuessIdentity;

static public WorkflowIdentity SequentialNumberGuessIdentity;

static WorkflowVersionMap()

{

map = new Dictionary<WorkflowIdentity, Activity>();

// Add the current workflow version identities.

StateMachineNumberGuessIdentity = new WorkflowIdentity

{

Name = "StateMachineNumberGuessWorkflow",

Version = new Version(1, 0, 0, 0)

};

FlowchartNumberGuessIdentity = new WorkflowIdentity

{

Name = "FlowchartNumberGuessWorkflow",

Version = new Version(1, 0, 0, 0)

};

SequentialNumberGuessIdentity = new WorkflowIdentity

{

Name = "SequentialNumberGuessWorkflow",

Version = new Version(1, 0, 0, 0)

};

map.Add(StateMachineNumberGuessIdentity, new StateMachineNumberGuessWorkflow());

map.Add(FlowchartNumberGuessIdentity, new FlowchartNumberGuessWorkflow());

map.Add(SequentialNumberGuessIdentity, new SequentialNumberGuessWorkflow());

}

public static Activity GetWorkflowDefinition(WorkflowIdentity identity)

{

return map[identity];

}

public static string GetIdentityDescription(WorkflowIdentity identity)

{

return identity.ToString();

}

}

WorkflowVersionMap contiene tres identidades de flujo de trabajo que se asignan a las tres definiciones de flujo de trabajo de este tutorial y se usan en las secciones siguientes al iniciar o al reanudar los flujos de trabajo.

[Para iniciar un nuevo flujo de trabajo](javascript:void(0))

1. Agregue un controlador Click para NewGame.Para agregar el controlador, cambie a **Vista de diseño** para el formulario y haga doble clic en NewGame.Se agrega un controlador NewGame\_Click y la vista cambia a vista de código para el formulario.Siempre que el usuario haga clic en este botón se inicia un nuevo flujo de trabajo.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-20)

private void NewGame\_Click(object sender, EventArgs e)

{

}

1. Agregue el código siguiente al controlador de clic.Este código crea un diccionario de argumentos de entrada para el flujo de trabajo, por clave de nombre de argumento.Este diccionario tiene una entrada que contiene el intervalo del número generado aleatoriamente recuperado en el cuadro combinado del intervalo.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-21)

var inputs = new Dictionary<string, object>();

inputs.Add("MaxNumber", Convert.ToInt32(NumberRange.SelectedItem));

1. A continuación, agregue el siguiente código que inicia el flujo de trabajo. **WorkflowIdentity** y la definición de flujo de trabajo correspondiente al tipo de flujo de trabajo seleccionado se recuperan mediante la clase auxiliar WorkflowVersionMap.A continuación, se crea una nueva instancia de **WorkflowApplication** mediante la definición de flujo de trabajo, **WorkflowIdentity**, y el diccionario de argumentos de entrada.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-22)

WorkflowIdentity identity = null;

switch (WorkflowType.SelectedItem.ToString())

{

case "SequentialNumberGuessWorkflow":

identity = WorkflowVersionMap.SequentialNumberGuessIdentity;

break;

case "StateMachineNumberGuessWorkflow":

identity = WorkflowVersionMap.StateMachineNumberGuessIdentity;

break;

case "FlowchartNumberGuessWorkflow":

identity = WorkflowVersionMap.FlowchartNumberGuessIdentity;

break;

};

Activity wf = WorkflowVersionMap.GetWorkflowDefinition(identity);

WorkflowApplication wfApp = new WorkflowApplication(wf, inputs, identity);

1. A continuación, agregue el siguiente código que agrega el flujo de trabajo a la lista de flujos de trabajo y muestra la información de la versión del flujo de trabajo en el formulario.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-23)

// Add the workflow to the list and display the version information.

WorkflowStarting = true;

InstanceId.SelectedIndex = InstanceId.Items.Add(wfApp.Id);

WorkflowVersion.Text = identity.ToString();

WorkflowStarting = false;

1. Llame a ConfigureWorkflowApplication para configurar el almacén de instancias, las extensiones y los controladores de ciclo de vida del flujo de trabajo para esta instancia **WorkflowApplication**.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-24)

// Configure the instance store, extensions, and

// workflow lifecycle handlers.

ConfigureWorkflowApplication(wfApp);

1. Por último, llame a **Run**.

VB

'Start the workflow.

wfApp.Run()

// Start the workflow.

wfApp.Run();

El siguiente ejemplo es el controlador NewGame\_Click completado.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-27)

private void NewGame\_Click(object sender, EventArgs e)

{

var inputs = new Dictionary<string, object>();

inputs.Add("MaxNumber", Convert.ToInt32(NumberRange.SelectedItem));

WorkflowIdentity identity = null;

switch (WorkflowType.SelectedItem.ToString())

{

case "SequentialNumberGuessWorkflow":

identity = WorkflowVersionMap.SequentialNumberGuessIdentity;

break;

case "StateMachineNumberGuessWorkflow":

identity = WorkflowVersionMap.StateMachineNumberGuessIdentity;

break;

case "FlowchartNumberGuessWorkflow":

identity = WorkflowVersionMap.FlowchartNumberGuessIdentity;

break;

};

Activity wf = WorkflowVersionMap.GetWorkflowDefinition(identity);

WorkflowApplication wfApp = new WorkflowApplication(wf, inputs, identity);

// Add the workflow to the list and display the version information.

WorkflowStarting = true;

InstanceId.SelectedIndex = InstanceId.Items.Add(wfApp.Id);

WorkflowVersion.Text = identity.ToString();

WorkflowStarting = false;

// Configure the instance store, extensions, and

// workflow lifecycle handlers.

ConfigureWorkflowApplication(wfApp);

// Start the workflow.

wfApp.Run();

}

[Para reanudar un flujo de trabajo](javascript:void(0))

1. Agregue un controlador Click para EnterGuess.Para agregar el controlador, cambie a **Vista de diseño** para el formulario y haga doble clic en EnterGuess.Siempre que el usuario haga clic en este botón se reanudará un flujo de trabajo.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-28)

private void EnterGuess\_Click(object sender, EventArgs e)

{

}

1. Agregue el código siguiente para asegurarse de que un flujo de trabajo está seleccionado en la lista de flujos de trabajo y de que el acierto del usuario es válido.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-29)

if (WorkflowInstanceId == Guid.Empty)

{

MessageBox.Show("Please select a workflow.");

return;

}

int guess;

if (!Int32.TryParse(Guess.Text, out guess))

{

MessageBox.Show("Please enter an integer.");

Guess.SelectAll();

Guess.Focus();

return;

}

1. A continuación, recupere **WorkflowApplicationInstance** de la instancia de flujo de trabajo persistente. **WorkflowApplicationInstance**representa una instancia de flujo de trabajo persistente que aún no se ha asociado a una definición de flujo de trabajo.**DefinitionIdentity** de **WorkflowApplicationInstance** contiene **WorkflowIdentity** de la instancia de flujo de trabajo persistente.En este tutorial, la clase de utilidad de WorkflowVersionMap se usa para asignar **WorkflowIdentity** a la definición de flujo de trabajo correcta.Una vez que se recupera la definición de flujo de trabajo, se crea **WorkflowApplication**, mediante la definición de flujo de trabajo correcta.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-30)

WorkflowApplicationInstance instance =

WorkflowApplication.GetInstance(WorkflowInstanceId, store);

// Use the persisted WorkflowIdentity to retrieve the correct workflow

// definition from the dictionary.

Activity wf =

WorkflowVersionMap.GetWorkflowDefinition(instance.DefinitionIdentity);

// Associate the WorkflowApplication with the correct definition

WorkflowApplication wfApp =

new WorkflowApplication(wf, instance.DefinitionIdentity);

1. Una vez que se ha creado **WorkflowApplication**, configure el almacén de instancias, los controladores de ciclo de vida del flujo de trabajo y las extensiones mediante una llamada a ConfigureWorkflowApplication.Deben llevarse a cabo estos pasos cada vez que se crea un nuevo objeto **WorkflowApplication** y antes de que se cargue la instancia de flujo de trabajo en **WorkflowApplication**.Cuando el flujo de trabajo ya está cargado, se reanuda con el intento del usuario.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-31)

// Configure the extensions and lifecycle handlers.

// Do this before the instance is loaded. Once the instance is

// loaded it is too late to add extensions.

ConfigureWorkflowApplication(wfApp);

// Load the workflow.

wfApp.Load(instance);

// Resume the workflow.

wfApp.ResumeBookmark("EnterGuess", guess);

1. Finalmente, desactive el cuadro donde se indica el número y prepare el formulario para aceptar otro intento.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-32)

// Clear the Guess textbox.

Guess.Clear();

Guess.Focus();

El siguiente ejemplo es el controlador EnterGuess\_Click completado.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-33)

private void EnterGuess\_Click(object sender, EventArgs e)

{

if (WorkflowInstanceId == Guid.Empty)

{

MessageBox.Show("Please select a workflow.");

return;

}

int guess;

if (!Int32.TryParse(Guess.Text, out guess))

{

MessageBox.Show("Please enter an integer.");

Guess.SelectAll();

Guess.Focus();

return;

}

WorkflowApplicationInstance instance =

WorkflowApplication.GetInstance(WorkflowInstanceId, store);

// Use the persisted WorkflowIdentity to retrieve the correct workflow

// definition from the dictionary.

Activity wf =

WorkflowVersionMap.GetWorkflowDefinition(instance.DefinitionIdentity);

// Associate the WorkflowApplication with the correct definition

WorkflowApplication wfApp =

new WorkflowApplication(wf, instance.DefinitionIdentity);

// Configure the extensions and lifecycle handlers.

// Do this before the instance is loaded. Once the instance is

// loaded it is too late to add extensions.

ConfigureWorkflowApplication(wfApp);

// Load the workflow.

wfApp.Load(instance);

// Resume the workflow.

wfApp.ResumeBookmark("EnterGuess", guess);

// Clear the Guess textbox.

Guess.Clear();

Guess.Focus();

}

[Para finalizar un flujo de trabajo](javascript:void(0))

1. Agregue un controlador Click para QuitGame.Para agregar el controlador, cambie a **Vista de diseño** para el formulario y haga doble clic en QuitGame.Siempre que el usuario haga clic en este botón se finalizará el flujo de trabajo seleccionado actualmente.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-34)

private void QuitGame\_Click(object sender, EventArgs e)

{

}

1. Agregue el código siguiente al controlador QuitGame\_Click.Este código primero comprueba que un flujo de trabajo está seleccionado en la lista de flujos de trabajo.Después, carga la instancia persistente en **WorkflowApplicationInstance**, usa **DefinitionIdentity** para determinar la definición correcta de flujo de trabajo e inicializa **WorkflowApplication**.A continuación, las extensiones y los controladores de ciclo de vida de flujo de trabajo se configuran mediante una llamada a **ConfigureWorkflowApplication**.Una vez se ha configurado**WorkflowApplication**, se carga y, a continuación, **Terminate** recibe una llamada.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-35)

if (WorkflowInstanceId == Guid.Empty)

{

MessageBox.Show("Please select a workflow.");

return;

}

WorkflowApplicationInstance instance =

WorkflowApplication.GetInstance(WorkflowInstanceId, store);

// Use the persisted WorkflowIdentity to retrieve the correct workflow

// definition from the dictionary.

Activity wf = WorkflowVersionMap.GetWorkflowDefinition(instance.DefinitionIdentity);

// Associate the WorkflowApplication with the correct definition

WorkflowApplication wfApp =

new WorkflowApplication(wf, instance.DefinitionIdentity);

// Configure the extensions and lifecycle handlers

ConfigureWorkflowApplication(wfApp);

// Load the workflow.

wfApp.Load(instance);

// Terminate the workflow.

wfApp.Terminate("User resigns.");

[Para compilar y ejecutar la aplicación](javascript:void(0))

1. Haga doble clic en **Program.cs** (o en **Module1.vb**) en el **Explorador de soluciones** para mostrar el código.
2. Agregue las siguientes instrucciones **using** (o **Imports**) al principio del archivo con las demás instrucciones **using** (o **Imports**).

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-36)

using System.Windows.Forms;

1. Quite o comente el código de hospedaje de flujo de trabajo existente de [Cómo: Ejecutar un flujo de trabajo](https://msdn.microsoft.com/es-es/library/dd489463(v=vs.110).aspx) y reemplácelo por el código siguiente.

C#

[**VB**](https://msdn.microsoft.com/es-es/library/dd489452(v=vs.110).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-37)

static void Main(string[] args)

{

Application.EnableVisualStyles();

Application.Run(new WorkflowHostForm());

}

1. Haga clic con el botón secundario en **NumberGuessWorkflowHost** en el **Explorador de soluciones** y elija **Propiedades**.En la pestaña**Aplicación**, especifique **Aplicación para Windows** para **Tipo de resultado**.Este paso es opcional, pero si no se realiza la ventana de la consola se muestra además del formulario.
2. Presione Ctrl+Mayús+B para compilar la aplicación.
3. Asegúrese de que **NumberGuessWorkflowHost** se ha establecido como la aplicación de inicio y presione Ctrl+F5 para iniciar la aplicación.
4. Seleccione un intervalo para el juego de adivinar un número y el tipo de flujo de trabajo que se va a iniciar, y haga clic en **New Game**.Escriba un intento en el cuadro de **Guess** y haga clic en **Go** para enviarlo.Observe que el resultado de las actividades **WriteLine** se muestra en el formulario.
5. Inicie varios flujos de trabajo mediante diferentes tipos de flujo de trabajo e intervalos de números, escriba algunos números y cambie de un flujo de trabajo a otro seleccionándolos en la lista **Identificador de instancia de flujo de trabajo**.

Observe que al cambiar a un nuevo flujo de trabajo, los intentos anteriores y el progreso del flujo de trabajo no aparecen en la ventana de estado.El estado no está disponible porque no se captura ni guarda en ningún lugar.En el siguiente paso del tutorial, [Cómo crear un participante de seguimiento personalizado](https://msdn.microsoft.com/es-es/library/jj205426(v=vs.110).aspx), se crea un participante de seguimiento personalizado que guarda esta información.