

Q-flow 3.1: Web services

Código del manual: Qf310008ESP
Versión: 1.1
Se aplica a: Q-flow 3.1
Última revisión: 27/12/2010

Q-flow 3.1

Web services

© Urudata Software
Canelones 1370 • Piso 2 CP11200
Montevideo, Uruguay
Teléfono: (598 2) 900 76 68 • Fax: 900 78 56

Tabla de Contenido

Introducción	4
Organización de este manual	4
Convenciones usadas en este manual	4
Descripción de los web services	4
WebBot	5
Métodos de WebBot.....	5
WebChart	5
Métodos de WebChart	6
WebDeploy	6
Métodos de WebDeploy.....	6
WebLinks	6
Método de WebLinks	6
WebLists	7
Métodos de WebLists.....	7
WebOperations	8
Métodos de WebOperations	8
WebOrganization	8
Métodos de WebOrganization	8
WebQueue	10
Métodos de WebQueue	10
WebResponse	10
Métodos de WebResponse.....	10
WebSharepoint	12
Métodos de WebSharepoint	12
WebStart	12
Métodos de WebStart	12
WebView	13
Métodos de WebView	13

Introducción

El propósito de este manual es describir los web services de Q-flow para apoyar a desarrolladores de aplicaciones que los utilizan.

Este manual contiene solamente información acerca de la interfaz y del comportamiento de los web services. Para obtener información acerca de cómo instalarlos y configurarlos, consulte el manual de instalación de Q-flow.

Organización de este manual

Q-flow incluye los siguientes web services:

- WebBot
- WebChart
- WebDeploy
- WebLinks
- WebLists
- WebOperations
- WebOrganization
- WebQueue
- WebResponse
- WebSharepoint
- WebStart
- WebView

El manual dedica una sección a cada uno de esos web services.

Convenciones usadas en este manual

Este manual representa la firma de un método de un web service de la siguiente forma:

Nombre_del_método(lista_de_parámetros):Tipo_de_Returno

Los parámetros están separados por comas, y cada parámetro está representado por su tipo seguido de su nombre, como en código C#. Ejemplo: string name. Si un método no devuelve nada, el tipo de retorno es "void", como en C#.

Descripción de los web services

A continuación, se describe cada uno de los web services de Q-flow.

WebBot

El web service WebBot es el que permite que un bot pueda obtener información acerca de qué tareas tiene pendientes, e informar a Q-flow acerca de qué tareas completó y canceló. Una explicación de qué es un bot y para qué sirve puede ser encontrada en el manual del diseñador de procesos del negocio.

Un bot, en general, utilizará los métodos de WebBot de la siguiente forma:

1. Invocará el método `GetActiveJobs` para obtener la lista de trabajos pendientes.
2. Recorrerá los trabajos devueltos por `GetActiveJobs` para procesar cada uno de ellos. Para obtener los datos necesarios para procesar un trabajo, invocará el método `GetJob`, que devuelve un objeto que contiene todos los datos del trabajo.
3. El bot procesa el trabajo. Mientras lo hace, modifica el objeto obtenido mediante el método `GetJob`. También puede modificar datos de aplicación, roles y adjuntos. Si durante el procesamiento de un trabajo el bot debe obtener el contenido de un archivo adjunto, invoca el método `GetAttachmentContent`.
4. Una vez procesado el trabajo, si pudo completarlo, el bot invocará el método `CompleteJob`, pasándole el objeto que modificó para que Q-flow guarde los cambios. Si debe abortar el trabajo (por ejemplo, porque hubo un error), invocará el método `AbortJob`.

Métodos de WebBot

- **AbortJob(BotJob job, string message): void** – aborta un trabajo del bot. El parámetro “job” representa el trabajo que se desea abortar, y se debe haber obtenido previamente mediante la invocación del método `GetActiveJobs`. El parámetro “message” permite especificar un mensaje para, por ejemplo, indicar por qué el trabajo fue cancelado.
- **CompleteJob(BotJobMessage job): void** – marca un trabajo del bot como completado. Un bot debe invocar este método al terminar de procesar un trabajo para indicarle a Q-flow que lo ha completado. El parámetro “job” se tiene que haber obtenido previamente mediante la invocación del método `GetJob`. Al invocar este método, se guardan los cambios efectuados por el bot. En el caso de los adjuntos, si se quitó alguno de la lista de adjuntos, Q-flow lo eliminará. Si se agregó un adjunto (y se le cargó la propiedad “Content” y el nombre), Q-flow lo agregará. Para modificar un adjunto, basta con modificar la propiedad “Content” del objeto `AttachmentMessage` que lo representa.
- **GetActiveJobs(Guid botId): BotJob[]** – devuelve un vector que indica qué trabajos del bot cuyo identificador es igual al valor del parámetro “botId” están pendientes.
- **GetAttachmentContent(BotJob job, Guid attachId): byte[]** – devuelve el contenido del archivo adjunto indicado por “attachId”. El bot debe tener permisos para acceder al archivo.
- **GetJob(BotJob job): BotJobMessage** – obtiene un objeto con los datos necesarios (datos del workflow, parámetros) para procesar el trabajo indicado por el parámetro “job”. Entre otras cosas, se trae la lista de adjuntos del workflow (si el paso de bot correspondiente tiene el permiso de listar adjuntos), pero sin el contenido de los adjuntos. Para obtener el contenido de un archivo adjunto, utilice el método `GetAttachmentContent`.

WebChart

El web service WebChart permite obtener los datos que abastecen las dos gráficas del sitio web: “Carga de tareas de mis supervisados” y “Tareas del usuario por estado”.

Métodos de WebChart

- **GetMyManagedTaskLoad(Guid packageId):UserTaskLoadResponse[]** – devuelve un vector de objetos de la clase UserTaskLoadResponse. Cada uno de estos objetos contiene información acerca de la carga de tareas de un usuario supervisado por el usuario cuya cuenta se utilizó para invocar el web service. Sólo se tienen en cuenta las tareas que pertenezcan a workflows del paquete indicado por “packageId” o de paquetes que desciendan de éste.
- **GetUserTaskLoad(Guid packageId, Guid userId): UserTaskLoadResponse** – devuelve un objeto que indica la carga de tareas del usuario con identificador “userId”. Sólo se tienen en cuenta las tareas que pertenezcan a workflows del paquete indicado por “packageId” o de paquetes que desciendan de éste.

WebDeploy

El web service WebDeploy provee funcionalidades de importación y exportación de paquetes (con los elementos contenidos en ellos).

Métodos de WebDeploy

- **ExportPackage(Guid packageId, bool recursive):PackageMessage** - devuelve un objeto con todos los datos del paquete identificado por “packageId”. Si el parámetro “recursive” tiene valor “true”, además de los datos del paquete identificado por “packageId” el objeto devuelto contiene todo el árbol cuya raíz es ese paquete (el paquete y todos sus descendientes).
- **ExportViewsById(Guid[] viewIds):ViewMessage[]** – devuelve un vector de objetos de la clase ViewMessage con todos los datos de las vistas especificadas por los identificadores contenidos en el parámetro “viewIds”.
- **ExportViewsByPackage(Guid packageId):ViewMessage[]** – devuelve un vector de objetos de la clase ViewMessage con todos los datos de las vistas del paquete identificado por “packageId”.
- **ImportPackage(PackageMessage package, Guid parentPackageId):ImportPackageResponse** - importa un paquete, representado por el parámetro “package”. El parámetro “parentPackageId” es el identificador del paquete dentro del cual se desea importar el paquete. El objeto devuelto tiene una lista de advertencias que indica posibles problemas y decisiones tomadas por Q-flow durante la importación.
- **ImportViews(ViewMessage[] views):void** – importa las vistas contenidas dentro del objeto recibido por parámetro.

WebLinks

El web service WebLinks dispone solamente de un método.

Método de WebLinks

- **HasUserPermissionsToSeeLink(string url):bool**: devuelve “true” si el usuario actual (aquél cuyas credenciales se utilizan para invocar el web service) tiene permiso para ver el vínculo cuya URL se indica en el parámetro url.

WebLists

El web service WebLists tiene métodos para obtener listas de tareas.

Métodos de WebLists

- **GetCurrentUserTasks():Task[]**– obtiene un vector que contiene las tareas del usuario en cuyo contexto se invoca el método.
- **GetFlowTasks(Guid flowId):Task[]** – obtiene un vector que contiene las tareas del workflow cuyo identificador es indicado por el parámetro flowId.
- **GetTemplates():Template[]** – obtiene un vector con los objetos que representan todos los templates del sistema y contienen información básica de éstos. El resultado es similar a lo que se muestra en la vista “Templates” del sitio web.
- **GetMyTemplates():Template[]** - obtiene un vector con los objetos que representan todos los templates del usuario actual, y contienen información básica de éstos. El resultado es similar a lo que se muestra en la vista “Mis Templates” del sitio web. El usuario actual es aquél cuyas credenciales son utilizadas para invocar el web service.
- **GetCurrentUserTasksByTemplatesWithData(Guid[] templatesFilter, Guid[] dataToInclude) : DataTable** – devuelve un objeto DataTable con información de las tareas que pertenecen a los templates cuyos identificadores se especifican en el parámetro templatesFilter. Las columnas del objeto DataTable devuelto son:
 - **FlowId**: identificador del workflow.
 - **TaskId**: identificador de la tarea
 - **TaskTold**: identificador de la tarea propia de un usuario (si una tarea está dirigida a n usuarios, está compuesta por n tareas propias de cada uno de esos usuarios).
 - **TemplateCorrelativId**: identificador correlativo del template.
 - **TemplateId**: identificador del template.
 - **TemplateName**: nombre del template.
 - **TemplateVersionId**: identificador de la versión del template.
 - **TemplateVersionName**: nombre de la versión del template.
 - **FlowCorrelativId**: identificador correlativo del workflow.
 - **FlowName**: nombre del workflow.
 - **FlowFlag**: bandera del workflow.
 - **FlowStartDate**: fecha de inicio del workflow.
 - **TaskName**: nombre de la tarea.
 - **TaskDescription**: descripción de la tarea.
 - **TaskSubject**: asunto de la tarea.
 - **TaskTimeStarted**: fecha de inicio de la tarea.
- **GetCurrentUserTasksByTemplatesNamesWithDataNames(string[] templatesFilter, string[] dataToInclude) : DataTable** – similar al anterior, pero en lugar de utilizar los identificadores de los templates y datos cuyos datos se deben traer, se utilizan sus nombres.
- **GetPackageViewData(PackageViewDataRequest request):PackageViewDataResponse** – devuelve los datos de una vista. Si se la utiliza bien, puede ser muy útil, puesto que el contenido de cualquier vista puede ser obtenido con esta función. Es como pedirle a Q-flow que muestre una vista, pero desde fuera de Q-flow. El método recibe un parámetro de tipo PackageViewDataRequest, que tiene las siguientes propiedades:
 - **Packageld:Guid**: identificador del paquete al cual pertenece la vista cuyos datos se están obteniendo.

- **PageNumber:int:** es el número de página cuyos datos se quieren traer. Qué datos se obtienen depende también del tamaño de la página.
- **PageSize:int:** es el tamaño de las páginas.
- **ViewId:Guid:** es el identificador de la vista cuyos datos se quieren obtener.

En el sitio web, no se muestran todos los datos de una vista en la misma página. Se muestran una cantidad limitada de filas por página. El usuario puede elegir la página que desea ver. La propiedad `PageSize` indica el tamaño de página, y la propiedad `PageNumber`, el número de página que se desea ver, de modo que invocar el método `GetPackageViewData` es análogo a hacer clic sobre un número de página de una vista, dado cierto tamaño de página. Para traer todos los elementos de la vista, se puede pedir la página 1 (`PageNumber = 1`) e indicar que el tamaño de la página sea `Int32.MaxValue`.

WebOperations

El web service `WebOperations` trabaja en conjunto con el paso de sincronización (ver manual del diseñador de procesos del negocio). Un paso de sincronización puede ser configurado para que espere la ocurrencia de una acción externa. La forma de indicarle a Q-flow que la acción externa tuvo lugar es invocar alguno de los métodos del web service `WebOperations`.

Métodos de WebOperations

- **FinalizeWaitingStepsByName(Guid flowId, string templateStepName): void:** dado un paso de sincronización, especificado por el identificador del workflow al que pertenece y el nombre del paso, indica que ese paso debe terminar la espera por una acción externa.
- **FinalizeWaitingStepsById(Guid flowId, Guid templateStepId): void:** dado un paso de sincronización, especificado por el identificador del workflow al que pertenece y el identificador del paso del template en el que se basa, indica que ese paso debe terminar la espera por una acción externa.

WebOrganization

El web service `WebOrganization` permite interactuar con el modelo organizacional de Q-flow.

Métodos de WebOrganization

- **CreateUser(string name, string description, Guid userNodeId):Guid** – crea una cuenta de usuario. El parámetro `name` indica el nombre de usuario. El parámetro `description` indica la descripción del usuario y `userNodeId` es el identificador del nodo en el que se debe crear la nueva cuenta.
- **CreateGroup(string name, string description, Guid groupId):Guid** – crea un grupo. El parámetro `name` indica el nombre del grupo. El parámetro `description` indica la descripción del grupo y `groupId` es el identificador del nodo en el que se debe crear el nuevo grupo.
- **DeleteUser(Guid userId):void** – elimina la cuenta de usuario cuyo identificador coincide con el valor del parámetro `userId`. Este método recibía otro parámetro en versiones anteriores. Sin embargo, los clientes que utilizan versiones anteriores del web service siguen funcionando con la nueva versión.
- **DeleteGroup(Guid groupId):void** – elimina el grupo cuyo identificador coincide con el valor del parámetro `groupId`. Este método recibía otro parámetro en versiones anteriores. Sin embargo,

los clientes que utilizan versiones anteriores del web service siguen funcionando con la nueva versión.

- **GetCalendars():CalendarMessage[]** – devuelve un vector de objetos que representan los calendarios existentes en el sistema y contienen los datos de éstos.
- **GetGroup(Guid groupId):GroupMessage** – devuelve un objeto que representa el grupo cuyo identificador coincide con el valor del parámetro groupId. Este método recibía otro parámetro en versiones anteriores. Sin embargo, los clientes que utilizan versiones anteriores del web service siguen funcionando con la nueva versión.
- **GetGroupsWithUser(Guid userId):SecurityMemberMessage[]** – devuelve un vector con objetos que representan aquellos grupos que tienen como miembro el usuario cuyo identificador coincide con el valor del parámetro userId.
- **GetSecurityMemberByName(string name): SecurityMemberMessage** - dado el nombre de un miembro del modelo organizacional (nodo, grupo o usuario), devuelve un objeto con sus datos.
- **GetNotifiers(): NotifierMessage[]** – devuelve un vector que contiene objetos que representan los servicios de notificación existentes.
- **GetSecurityProviders():SecurityProviderMessage[]** – devuelve un vector de objetos que representan los proveedores de seguridad y contienen los datos de éstos.
- **GetSecurityRoles():SecurityRoleMessage[]**: - devuelve un vector de objetos que representan los roles de seguridad existentes en el sistema y que contienen los datos de éstos.
- **GetSecurityRolesWithUser(Guid userId):SecurityRolesMessage[]** – devuelve un vector con objetos que representan aquellos roles de seguridad que tienen como miembro el usuario cuyo identificador coincide con el valor del parámetro userId.
- **GetUser(Guid userId):UserMessage** – dado el identificador de un usuario, devuelve un objeto con los datos de ese usuario. Este método recibía otro parámetro en versiones anteriores. Sin embargo, los clientes que utilizan versiones anteriores del web service siguen funcionando con la nueva versión.
- **GetUserByLogon(string logon): UserMessage**: dado el logon de un usuario, devuelve un objeto con sus datos.
- **GetUserByExtendedProperties(ExtendedPropertyMessage[] properties, RestrictionType restrictionType): UserMessage[]** - dado un conjunto de propiedades extendidas con valores especificados, devuelve el conjunto de usuarios que tienen todas esas propiedades con los valores indicados (si RestrictionType es “And”) o el conjunto de usuarios que tienen por lo menos una de esas propiedades con el valor indicado (si RestrictionType es “Or”). Por ejemplo, si RestrictionType es “And” y las propiedades son “Sexo” con el valor “M” y “País” con el valor “Uruguay”, devuelve todos los usuarios que tengan la propiedad “Sexo” con el valor “M” y la propiedad “País” con el valor “Uruguay”. Si la RestrictionType es “Or”, devuelve los usuarios que tienen la propiedad “Sexo” en “M” o la propiedad “País” en “Uruguay”.
- **GetUsersInGroup(Guid groupId, bool excludeSubNodeMembers): UserMessage[]**: dado el identificador de un grupo, devuelve todos sus usuarios. Si excludeSubNodeMembers es “false”, devuelve también los usuarios de todos los grupos descendientes de ese grupo. De lo contrario, devuelve solamente los usuarios del grupo cuyo identificador fue indicado.
- **ModifyGroup(GroupMessage group):void** – dado un objeto GroupMessage que representa un grupo, modifica los datos de ese grupo, reemplazando sus valores por los que son proporcionados en el parámetro. Es conveniente invocar el método GetGroup antes de invocar este, para obtener un objeto GroupMessage que contenga los datos actualizados del grupo. Después, se modifica el objeto obtenido y se lo utiliza como parámetro para invocar este método.
- **ModifyUser(UserMessage user): void** – dado un objeto UserMessage que representa un usuario, modifica los datos de ese usuario, reemplazando sus valores por los que son proporcionados en el parámetro. Es conveniente invocar el método GetUser antes de invocar este, para obtener un objeto UserMessage que contenga los datos actualizados del usuario.

Después, se modifica el objeto obtenido y se lo utiliza como parámetro para invocar este método.

- **MoveGroup(Guid groupId, Guid toNodeId): void** – mueve el grupo indicado por el parámetro groupId al nodo indicado por el parámetro toNodeId.
- **MoveUser(Guid userId, Guid toNodeId): void** - mueve el usuario indicado por el parámetro userId al nodo indicado por el parámetro toNodeId.

WebQueue

El web service WebQueue permite trabajar con colas de trabajo.

Métodos de WebQueue

- **CheckInTask(Guid flowId, Guid taskId, Guid taskTold):void**: dados los identificadores de un workflow (flowId), de una tarea (taskId) y de la parte de esa tarea que es propia de un usuario (taskTold), hace que el usuario con cuyas credenciales se está invocando el web service deje la tarea. La tarea tiene que haber sido tomada por el usuario antes (por ejemplo, con el método CheckOutTask).
- **CheckOutTask(Guid flowId, Guid taskId, Guid taskTold):void**: dados los identificadores de un workflow (flowId), de una tarea (taskId) y de la parte de esa tarea que es propia de un usuario (taskTold), hace que el usuario con cuyas credenciales se está invocando el web service tome la tarea. La tarea tiene que estar dirigida a una cola de trabajo para la cual el usuario tenga permisos de actuar.
- **GetMyWorkQueueTasks(Guid workQueueId):Task[]**: dado el identificador de una cola de trabajo, permite obtener las tareas de esa cola tomadas por el usuario con cuyas credenciales se invoca el web service.
- **GetUnassignedWorkQueueTasks(Guid workQueueId):Task[]**: dado el identificador de una cola de trabajo, permite obtener las tareas de esa cola que no están tomadas por ningún usuario.

WebResponse

El web service WebResponse tiene métodos para responder tareas. Para especificar una tarea se utilizan:

- El identificador del workflow al que pertenece (**flowId**)
- El identificador del paso que corresponde a la tarea en el workflow (**stepId**).
- El identificador del destinatario en la tarea (**told**).

Métodos de WebResponse

- **CheckInAttachment(Guid flowId, Guid stepId, Guid told, Guid attachId):void** - Protege el archivo adjunto identificado por "attachId". Esto es análogo a hacer clic en el botón "Proteger" para ese adjunto en el sitio web.
- **CheckOutAttachment(Guid flowId, Guid stepId, Guid told, Guid attachId):void**- Desprotege el archivo adjunto identificado por "attachId". Esto es análogo a hacer clic en el botón "Desproteger" para ese adjunto en el sitio web.

- **DeleteAttachment(Guid flowId, Guid stepId, Guid told, Guid attachId):void**- Elimina el archivo adjunto identificado por "attachId". Esto es análogo a hacer clic en el botón "Eliminar" para ese adjunto.
- **DownloadAttachment(Guid flowId, Guid stepId, Guid told, Guid attachId): byte[]** – Devuelve el contenido del archivo adjunto identificado por "attachId". Esta es la función que se debe utilizar cuando el adjunto está protegido (checked-in). Esto es análogo a hacer clic en un adjunto protegido en el formulario de tarea.
- **DownloadCheckedOutAttachment(Guid flowId, Guid stepId, Guid told, Guid attachId):byte[]** - Devuelve el contenido del archivo adjunto identificado por "attachId". Ésta es la función a utilizar cuando el adjunto está desprotegido (checked-out). Esto es análogo a hacer clic en un adjunto desprotegido en el formulario de tarea.
- **GetTask(Guid flowId, Guid stepId, Guid told):TaskMessage** – obtiene un objeto que representa la tarea determinada por el identificador de un workflow (FlowId), un paso de ese workflow (StepId) y un destinatario de ese paso (Told). Una vez obtenido ese objeto, es posible usarlo para responder la tarea que representa por medio de alguno de los otros métodos de ese web service.
- **RespondTask(TaskMessage task):void** – responde la tarea representada por parámetro. Ese objeto debe haber sido obtenido por medio del método GetTask y ser modificado para indicar la respuesta que se da a la tarea y, opcionalmente, un porcentaje de progreso.
- **RespondTaskNow(Guid flowId, Guid stepId, Guid told, string responseKey):void** – responde una tarea con la respuesta indicada por el parámetro ResponseKey. La tarea que se responde está determinada por el identificador de un workflow (FlowId), un paso de ese workflow (StepId) y un destinatario de ese paso (Told).
- **RespondTaskNowWithProgress(Guid flowId, Guid stepId, Guid told, string responseKey, byte progress):void** – responde una tarea con la respuesta indicada por el parámetro ResponseKey, y actualizando el progreso de la tarea con el valor del parámetro Progress. La tarea que se responde está determinada por el identificador de un workflow (FlowId), un paso de ese workflow (StepId) y un destinatario de ese paso (Told).
- **UndoCheckOutAttachment(Guid flowId, Guid stepId, Guid told, Guid attachId): void** - Deshace la desprotección del archivo adjunto identificado por "attachid". Esto es análogo a hacer clic en el botón "Deshacer desprotección" para ese adjunto en el sitio web.
- **UploadCheckedOutAttachment(Guid flowId, Guid stepId, Guid told, Guid attachId, byte[] content):void**- Actualiza el contenido del archivo adjunto identificado por "attachid". El adjunto debe estar desprotegido para actualizar su contenido. Esto es análogo a seleccionar un archivo con el mismo nombre que un adjunto y hacer clic en "Subir" (cuando el adjunto está desprotegido).
- **UploadNewAttachment(Guid flowId, Guid stepId, Guid told, string name, byte[] content):Guid** - Crea un nuevo archivo adjunto con el nombre y contenido especificados por los parámetros "name" y "content" respectivamente. Devuelve el identificador del adjunto creado. Esto es análogo a seleccionar un archivo con nombre diferente de los adjuntos existentes y hacer clic en "Subir".

WebSharepoint

Este web service es utilizado por las web parts para Sharepoint de Q-flow. Son de uso interno. En lugar de usar este web service, se recomienda utilizar el web service WebLists. El web service WebSharepoint existe por razones de compatibilidad con Sharepoint 2003.

Métodos de WebSharepoint

- **GetMyPackageViewData(PackageViewDataRequest request): PackageViewDataResponseDataSet** - devuelve los datos de una vista personal del usuario, según lo especificado en el parámetro "request".
- **GetPackageViewData(PackageViewDataRequest request): PackageViewDataResponseDataSet** – hace lo mismo que el método del mismo nombre del web service WebLists.
- **Login():void** – permite comprobar si el usuario actual puede autenticarse en Q-flow.

WebStart

El web service WebStart tiene métodos que inician workflows.

Métodos de WebStart

- **StartFlowNow(Guid templateId, string title, string description):Guid** – inicia un workflow basado en el template indicado por el parámetro templateId. El parámetro title indica el título que se le dará al workflow, y el parámetro description indica la descripción. Devuelve el identificador global (Guid) del workflow iniciado.
- **StartFlowNowCorrelativeId(long templateCorrelativeID, string title, string description):Guid** – hace lo mismo que el método StartFlowNow, pero en lugar de recibir un parámetro del tipo Guid para identificar el template en el que se debe basar el nuevo workflow, recibe un parámetro de tipo long, que indica el identificador correlativo del template a ser utilizado.
- **StartFlow(NewFlowMessage flowMessage):Guid** – hace lo mismo que los métodos StartFlowNow y StartFlowNowCorrelative, pero recibe un parámetro del tipo NewFlowMessage para indicar los datos del workflow que será iniciado. El objeto NewFlowMessage, además de tener propiedades que representan el título y la descripción del workflow, tiene propiedades que representan los datos de aplicación, roles y archivos adjuntos del workflow. Estas propiedades pueden ser modificadas para inicializar los datos, roles y archivos adjuntos del workflow que se va a iniciar. Para obtener el objeto NewFlowMessage que se va a utilizar para invocar este método, primero debe invocar el método GetFlowInfoCorrelativeId o el método GetNewFlowInfo.
- **GetNewFlowInfoCorrelativeId(long templateCorrelativeID):NewFlowMessage** – dado un template indicado por su identificador correlativo, devuelve un objeto de tipo NewFlowMessage que puede ser utilizado para iniciar un workflow con el método StartFlow.
- **GetNewFlowInfo(Guid templateId):NewFlowMessage** – dado un template indicado por su identificador global, devuelve un objeto de tipo NewFlowMessage que puede ser utilizado para iniciar in workflow con el método StartFlow.

WebView

Este web service es utilizado por las web parts para Sharepoint de Q-flow. Permite obtener información de vistas.

Métodos de WebView

- **GetMyPackageViews(Guid packageId):ViewMessage[]** – dado el identificador de un paquete (packageId), devuelve las definiciones de las vistas que pertenecen a ese paquete y son vistas personales del usuario actual.
- **GetMyViewMessage(Guid viewId): ViewMessage** – devuelve la definición de una vista personal del usuario actual, identificada por su identificador (viewId).
- **GetNonSystemPackageViews(Guid packageId):ViewMessage[]** – dado el identificador de un paquete (packageId), devuelve las definiciones de las vistas que no son del sistema y que pertenecen a ese paquete.
- **GetSystemPackageViews(Guid packageId):ViewMessage[]** – dado el identificador de un paquete (packageId), devuelve las definiciones de las vistas de sistema que pertenecen a ese paquete.
- **GetViewMessage(Guid viewId): ViewMessage** – devuelve la definición de la vista cuyo identificador coincide con el valor del parámetro viewId.