Artículo
Ricardo Paiva · Nov 14, 2019 Lectura de 15 min

Despliegue de aplicaciones con %Installer

¡ Hola Comunidad!
Suponga que desarrolló su propia aplicación con la tecnología de InterSystems y ahora quiere realizar varias implementaciones en sus distintos clientes. Durante el proceso de desarrollo usted escribió una guía de instalación detallada para aplicarla, ya que no solo necesita importar clases, también configurar el entorno de acuerdo a sus necesidades.

Para atender esta tarea específica, InterSystems creó una herramienta especial llamada **%Installer**. Siga con la lectura para saber cómo utilizarla.

%Installer

Con esta herramienta, podrá definir el manifiesto de instalación, que describe la configuración de Caché que se desea, en lugar de los pasos para su instalación. Lo único que debe hacer es describir lo que quiere, y Caché generará automáticamente el código necesario para modificar el entorno por usted. Por lo tanto, solo debe distribuir el manifiesto en sí, mientras que todo el código de instalación se generará para el servidor específico de Caché en el momento de la compilación.

Para definir un manifiesto, cree un nuevo bloque XData con una descripción detallada de la configuración de destino e implemente un método para generar el código en Caché ObjectScript para este bloque XData (este código siempre es el mismo). Una vez que el manifiesto esté listo, puede acceder a él desde la consola/terminal o bien desde código Caché ObjectScript, o automáticamente durante la instalación de Caché. El manifiesto debe ejecutarse en el namespace %SYS. Los manifiestos pueden manejar tanto parámetros del sistema (superport, OS, directorio mgr, etc.) como parámetros arbitrarios proporcionados por el usuario. En general, cada clase de instalación debe cumplir los siguientes requisitos:

- Contener un enlace para %occInclude.inc
- Contener un bloque XData con la configuración del servidor de Caché
- El bloque puede tener cualquier nombre que sea válido
- Agregar [XMLNamespace = INSTALLER] después del nombre del bloque, si es necesario consulte las indicaciones de Studio
- Llamar al elemento raíz (solo debe tener uno) <Manifest> que incluye a todos los demás elementos
- También debe implementar el método setup(), el cual generará el código que necesite el programa para el bloque XData.

Conocimientos básicos sobre el instalador

Puede ejecutar un manifiesto de instalación de varias **formas**:

- En el namespace %SYS desde la consola/terminal o desde código en Caché ObjectScript

```
do ##class(MyPackage.MyInstaller).setup()
```

- Se realiza automáticamente durante la instalación de Caché. Para ello, exporte la clase del instalador en DefaultInstallerClass.xml que está almacenada en la carpeta con el paquete de instalación de Caché (por ejemplo, donde se almacenan setup_cache.exe o cinstall). Durante la instalación de Caché, esta clase se importará al namespace %SYS y se ejecutará mediante el método setup().

Ejemplo

Consideremos un ejemplo sencillo. Establezca la clase **App.Installer** que contiene un instalador, el cual generará un nuevo namespace con el nombre definido por el usuario, después creará la aplicación web predeterminada e importará el código a este nuevo namespace:

```
Include %occInclude
Class App.Installer {
/// You can see generated method in zsetup+1^App.Installer.1
XData Install [ XMLNamespace = INSTALLER ]
{
<Manifest>
    <If Condition='(##class(Config.Namespaces).Exists("${Namespace}")=0)'>
        <Log Text="Creating namespace ${Namespace}" Level="0"/>
        <Namespace Name="${Namespace}" Create="yes" Code="${Namespace}" Ensemble="0"
Data="${Namespace}">
            <Configuration>
                <Database Name="${Namespace}" Dir="${MGRDIR}${Namespace}" Create="yes
"/>
            </Configuration>
        </Namespace>
        <Log Text="End Creating namespace ${Namespace}" Level="0"/>
    </If>
    <Role Name="AppRole" Description="Role to access and use the App" Resources="%DB_
CACHESYS:RW,%Admin_Secure:U" />
    <Namespace Name="${Namespace}" Create="no">
        <CSPApplication Url="/csp/${Namespace}" Directory="${CSPDIR}${Namespace}" Aut
henticationMethods="64" IsNamespaceDefault="true" Grant="AppRole" />
        <IfDef Var="SourceDir">
            <Log Text="SourceDir defined - offline install from ${SourceDir}" Level="
0"/>
            <Import File="${SourceDir}"/>
        </IfDef>
    </Namespace>
</Manifest>
}
///Entry point method, you need to call
/// At class compile time it generate Caché ObjectScript code from the manifest
/// After that you can run this installer from a terminal:
/// Set pVars("Namespace")="NewNamespace"
/// Set pVars("SourceDir")="C:\temp\distr\"
/// Do ##class(App.Installer).setup(.pVars)
ClassMethod setup(ByRef pVars, pLogLevel As %Integer = 0, pInstaller As %Installer.In
staller) As %Status [ CodeMode = objectgenerator, Internal ]
{
     Quit ##class(%Installer.Manifest).%Generate(%compiledclass, %code, "Install")
}
}
```

En este ejemplo, el instalador realiza las siguientes acciones:

- Comprueba si existe un namespace con el mismo nombre que el valor de la variable Namespace (para que quede claro, especificaremos que la variable Namespace se estableció en NewNamespace)

- Si no es así, entonces registrará la creación de un nuevo namespace llamado NewNamespace
- Definir un nuevo namespace:
    - El nombre es NewNamespace
    - Crea un nuevo namespace
    - La base de datos para las rutinas es NewNamespace
    - No activa Ensemble
    - La base de datos para globales es NewNamespace
- Crea una nueva base de datos
    - Su nombre es NewNamespace;
    - La crea en la carpeta mgr/NewNamespace (tenga en cuenta que la variable MGRDIR está disponible de forma predeterminada)
- La creación del namespace está completa y registrada
- Crea una nueva función: AppRole (con los recursos %DB_CACHESYS:RW y %Admin_Secure:U)
- Crea una nueva aplicación web predeterminada /csp/NewNamespace (también asigna AppRole de forma automática)
- Si la variable SourceDir está definida, entonces importa todos los archivos desde allí a NewNamespace

Para que este instalador se inicie en un terminal, ejecute los siguientes comandos:

```
Set pVars("Namespace")="NewNamespace"
Set pVars("SourceDir")="C:\temp\distr\"
Do ##class(App.Installer).setup(.pVars)
```

Durante la ejecución el terminal muestra información importante:

```
2016-02-17 19:26:17 0 App.Installer: Installation starting at 2016-02-17 19:26:17, Lo
gLevel=0
2016-02-17 19:26:17 0 : Creating namespace NewNamespace
2016-02-17 19:26:17 0 : End Creating namespace NewNamespace
2016-02-17 19:26:17 0 : SourceDir defined - offline install from C:\temp\distr\
2016-02-17 19:26:18 0 App.Installer: Installation succeeded at 2016-02-17 19:26:18
2016-02-17 19:26:18 0 %Installer: Elapsed time .545148s
```

Para recibir aún más información sobre lo que está sucediendo, especifique LogLevel (de 0 (default) a 3 (raw); más alto = más información).

```
Do ##class(App.Installer).setup(.pVars, 3)
```

Ahora hablaremos sobre las cosas que pueden hacerse en el manifiesto de instalación.

Nodos disponibles

Un manifiesto se compone de los siguientes elementos:

Nodo	**Nodo padre**	**Atributos (valores predeterminados)**	**Descripción**
Arg	Invoke, Error	Value – es el valor de un argumento	Pasa un argumento hacia un método que se llama mediante Invoke o Error
ClassMapping	Configuration	Package - un paquete que debe mapearse From - nombre de la base de datos de origen que se	Crea un mapeo de clases desde una base de datos hacia el namespace que contiene el elemento Configuration

		utilizó para el mapeo	
Compile	Namespace	Class - son los nombres de las clases para compilación Flags - son marcas de compilación (ck) IgnoreErrors - se utiliza para ignorar errores (0)	Son la clases de los compiladores. Llama a **$System.OBJ.Compile(Class, Flags)**
Configuration	Namespace		Se utiliza para crear namespaces y bases de datos. Cierra las etiquetas que activan los mapeos y actualiza el archivo CPF
CopyClass	Namespace	Src - es la clase de origen Target - es la clase de destino Replace - elimina la clase de origen (0)	Copia o desplaza la definición de la clase de origen a la de destino
CopyDir	Manifest	Src - es el directorio fuente Target - es el directorio de destino IgnoreErrors - se utiliza para ignorar errores (0)	Copia un directorio
CopyFile	Manifest	Src - es el archivo de origen Target - es el archivo de destino IgnoreErrors - se utiliza para ignorar errores (0)	Copia un archivo
Credential	Production	Name - es el nombre de las credenciales de acceso Username - es el nombre de usuario Password - es la contraseña de usuario Overwrite - sobrescribe si la cuenta ya existe	Crea o sobreescribe las credenciales de acceso
CSPApplication	Namespace	AuthenticationMethods - activa los métodos de autenticación	Crea o modifica una aplicación web. Para obtener más detalles,

		AutoCompile - realiza compilaciones automáticas (en la configuración CSP) CSPZENEnabled - la marca CSP/ZEN ChangePasswordPage - es la ruta para cambiar la contraseña de la página CookiePath - ruta hacia la sesión de cookies CustomErrorPage - ruta para personalizar la página de error DefaultSuperclass - es una superclase personalizada DefaultTimeout - se agotó el tiempo en espera de la sesión Description - realiza descripciones Directory - ruta hacia los archivos CSP EventClass - nombre de la clase del evento Grant - lista de funciones asignadas en el momento que el sistema actualiza su registro GroupById - realiza agrupaciones mediante las propiedades del Id InboundWebServicesEnabled - marca de los servicios web entrantes IsNamespaceDefault - es la marca de la aplicación para un Namespace predeterminado LockCSPName - Bloqueo de la marca del nombre CSP LoginClass - ruta para acceder a la página de	consulte la **documentación** y la clase **Security.Applications**

		inicio de sesión PackageName - nombre del paquete de propiedades PermittedClasses - clases de propiedades permitidas Recurse - indicador de recursión (sirve a los subdirectorios) (0) Resource - recurso requerido para acceder a la aplicación web ServeFiles - propiedad de los archivos de servicio ServeFilesTimeout - es el tiempo, en segundos, que le toma a Chaché almacenar los archivos estáticos. TwoFactorEnabled - es una autenticación de dos pasos Url - es el nombre de la aplicación web UseSessionCookie - utiliza las cookies para la sesión	
Database	Configuration	BlockSize - tamaño del bloque en bytes de la base de datos ClusterMountMode - se encarga de organizar la base de datos como parte del cluster Collation - orden de la clasificación Create - si desea crear una nueva base de datos (yes/no/overwrite (yes)) Dir - es el directorio Encrypted - cifrado de la base de datos EncryptionKeyID - ID de la clave de cifrado	Crea o modifica una base de datos. Para obtener más información, consulte la **documentación** y las clases **Config.Databases** y **SYS.Database**

		ExpansionSize - tamaño en MB que puede expandirse InitialSize - tamaño inicial MaximumSize - tamaño máximo MountAtStartup - organización después del lanzamiento MountRequired - especifica que la base de datos DEBE organizarse con éxito en el momento del inicio Nombre - es el nombre de la base de datos PublicPermissions - son los permisos públicos Resource - es el recurso StreamLocation - es el directorio a donde se dirigen los flujos que están asociados a esta base de datos.	
Default	Manifest	Name - es el nombre de la variable Value - es el valor de la variable Dir - valor de la variable (si es una ruta hacia una carpeta/archivo)	Establece el valor de la variable (si aún no se estableció)
Else	Manifest, Namespace	Puede ejecutarse cuando la sentencia if es falsa	
Error	Manifest	Status - código de error Source - origen del error	Envía una excepción. Tenga en cuenta que la sintaxis para ${} y #{} no está disponible
ForEach	Manifest	Index - es el nombre de la variable Values - una lista con los valores para la variable	**Collection-es un bucle controlado**
GlobalMapping	Configuration	Global - es el nombre del global	Mapea un global

		From - es el nombre de la base de datos para el mapeo Collation - orden de la clasificación (Caché por defecto)	
If	Manifest, Namespace	Condition - es una sentencia condicional	Sentencia condicional if
IfDef	Manifest, Namespace	Var – nombre de la variable	La sentencia condicional if se utiliza cuando la variable ya fue establecida
IfNotDef	Manifest, Namespace	Var – nombre de la variable	La sentencia condicional if se utiliza cuando la variable aún no fue establecida
Import	Namespace	File - archivo/carpeta para importación Flags - marcas de compilación (ck) IgnorarErrores - se utiliza para ignorar errores (0) Recurse - importar recursivamente (0)	Importa archivos. Llama a: **$System.OBJ.ImportDir(File,,Flags,,Recurse)** y **$System.OBJ.Load(File, Flags)**
Invoke	Namespace	Class - nombre de la clase Method - nombre del método CheckStatus - comprueba el estado de la respuesta Return - escribe el resultado en una variable	Hace una llamada a un método de una clase con varios argumentos y devuelve los resultados de la ejecución
LoadPage	Namespace	Name: ruta a la página CSP Dir - es una carpeta con páginas CSP Flags - indicadores de compilación (ck) IgnoreErrors - se utiliza para ignorar errores (0)	Carga archivos CSP mediante **$System.CSP.LoadPage(Name, Flags)** y **$System.CSP.LoadPageDir(Dir, Flags)**
Log	Manifest	Level - nivel de registro desde 0 (mínimo) hasta 3 (detallado)	Añade un mensaje al registro cuando el nivel de registro es mayor o igual al

		Text - cadena con una longitud de hasta 32,000 caracteres	atributo "level"
Manifest			Elemento de la raíz. Es el único elemento de la raíz en un manifiesto, contiene todos los demás elementos
Namespace	Manifest	Name - nombre del namespace Create - si se debe crear un nuevo namespace (yes/no/overwrite (yes)) Code - base da datos para el código del programa Data - base de datos Ensemble - activa Ensemble para el namespace Todos los demás atributos pueden utilizarse con las aplicaciones web de Ensemble	Define el alcance del instalador
Production	Namespace	Nombre - nombre de la producción AutoStart - lanzamiento automático de la producción	Configura la producción en Ensemble
Resource	Manifest	Name - nombre del recurso Description - descripción del recurso Permission - permisos públicos	Crea o modifica un recurso.
Role	Manifest	Name - nombre del role Description - descripción de la función (no debe contener comas) Resources - son los recursos asignados a la función, se representan como "MyResource:RW,MyResource1:RWU"	Crea un nuevo role

		RolesGranted - si se otorgan las funciones correspondientes	
RoutineMapping	Configuration	Routines - nombre de la rutina Type - tipos de rutinas (MAC, INT, INC, OBJ o ALL) From - base de datos de origen	Crea un nuevo mapeo para las rutinas
Setting	Production	Item - elemento que puede configurarse Target - tipos de parámetros: Item, Host, Adapter Setting - nombre del parámetro Value - valor del parámetro	Configura un elemento en la producción de Ensemble. Hace una llamada al método **Ens.Production:ApplySettings**
SystemSetting	Manifest	Name - class.property del paquete Config Value - valor del atributo	Establece los valores para los atributos del paquete Config (usando el método Modify)
User	Manifest	Username - nombre de usuario PasswordVar - variable que contiene la contraseña Roles - lista de funciones del usuario Fullname - nombre completo Namespace - espacio de nombres de inicio Routine - rutina de inicio ExpirationDate - fecha después de la cual el usuario será desactivado ChangePassword - cambie la contraseña al iniciar sesión por última vez en el sistema Enabled - si el usuario está	Crea o modifica un usuario.

		activo	
Var	Manifest	Name - nombre de la variable Value - valor de la variable	Asigna un valor a la variable

Variables

Variables suministradas por el usuario

Algunos atributos pueden contener expresiones (cadenas) que aumentan cuando el manifiesto se ejecuta. Existen tres tipos de expresiones que aumentarían y son parecidas a las siguientes:

- ${<Variable_name>} – es el valor de la variable (definida por el usuario o una variable de entorno, consulte más adelante) que se calcula durante la ejecución del manifiesto,
- ${#<Parameter_name>} – se sustituirá por el valor del parámetro especificado desde la clase del instalador durante la compilación,
- #{<Caché_ObjectScript_code>} — es el valor de la sentencia que se especificó en Caché ObjectScript y se calculará durante la ejecución del manifiesto. Asegúrese de poner comillas según sea necesario.

Los valores de los parámetros se definen durante la compilación y, por lo tanto, pueden formar parte de una variable, o de una sentencia de Caché ObjectScript. Dado que las variables se interpretan antes que el código de Caché ObjectScript, puede utilizarlas en las sentencias de Caché ObjectScript, por ejemplo: #{$ZCVT("${NAMESPACE}","L")}.

Variables del sistema

Las siguientes variables siempre están disponibles:

Variable	**Descripción**	**Ejemplo del valor**
SourceDir	(Disponible solo cuando se ejecuta el instalador) Directorio desde el que se ejecuta la instalación (setup_cache.exe o cinstall).	/InterSystems/distr/
ISCUpgrade	(Disponible solo cuando se ejecuta el instalador) Indica si se trata de una instalación nueva o de una actualización. Cuando esta variable es 0 se considera una instalación nueva, o 1 cuando es actualización.	0 (instalación) 1 (actualización)
CFGDIR	Consulte INSTALLDIR.	/InterSystems/Cache/
CFGFILE	Ruta al archivo CPF	/InterSystems/Cache/cache.cpf
CFGNAME	Es el nombre de la instancia	CACHE
CPUCOUNT	Número de núcleos en el CPU	4
CSPDIR	Es el directorio CSP	/InterSystems/Cache/csp/

HOSTNAME	Es el nombre del servidor web	SCHOOL15
HTTPPORT	Es el puerto del servidor web	80
INSTALLDIR	Es el directorio donde se instaló Caché	/InterSystems/Cache/
MGRDIR	Es el directorio de administración (mgr)	/InterSystems/Cache/mgr/
PLATFORM	Es el sistema operativo	UNIX
PORT	Es el puerto del super servidor de Caché	1972
PROCESSOR	Es el nombre de la plataforma	x86-64
VERSION	Es la versión de Caché	2015.1.1

Depuración de errores

Algunas veces es difícil entender qué valores pueden asignarse como valores de atributos en los nodos. Para averiguarlo, compruebe el código int generado para el método de instalación. En la mayoría de los casos, la llamada principal se realiza a tInstaller.<ElementName> que es un objeto de la clase **%Installer.Installer** que, a su vez, hará llamadas directas a los métodos del sistema. Alternativamente, puede comprobar el código de %Installer.class<ElementName> en la que los atributos del nodo son propiedades de la clase. El código del programa se genera en los métodos %OnBeforeGenerateCode, %OnGenerateCode y %OnAfterGenerateCode.

Con fines de depuración, recomiendo que coloque una llamada en una transacción dentro del instalador. Por ejemplo, puede utilizar los comandos **TSTART**/**TROLLBACK** para deshacer fácilmente todos los cambios realizados dentro de Caché (sin embargo, los cambios externos, como crear un archivo nuevo para la base de datos, no se revertirán).

Por último, no olvide configurar LogLevel en 3.

Ejemplos

El proyecto **MDX2JSON** proporciona un **instalador**. Para instalar el proyecto, importe el archivo **installer.xml** que contiene la clase MDX2JSON.Installer en cualquiera de los siguientes formatos namespace. Puede realizar la importación desde **SMP** o arrastrando y soltando el archivo en Studio.

Entonces ejecute el siguiente comando en un terminal:

```
do ##class(MDX2JSON.Installer).setup()
```

Como resultado, Caché cargará los archivos de la aplicación desde el repositorio GitHub y luego realizará la instalación en la base de datos predeterminada MDX2JSON namespace/MDX2JSON, mapeará el paquete MDX2SJON a %All y SAMPLES, mapeará el ^MDX2SJON global a %All y SAMPLES, creará la aplicación REST llamada /MDX2JSON, y así sucesivamente, verá todos estos pasos en el terminal. Para obtener información más detallada sobre el instalador de MDX2JSON, consulte el proyecto **Léame**.

Ejemplos adicionales

Ejemplo de los documentos de apoyo .

La clase Sample.Installer en el namespace Samples.

Los proyectos CacheGitHubCI proporcionan un **instalador** .

El proyecto SYSMON que se encuentra en el panel de controles proporciona un **instalador** .

El proyecto DeepSee Audit proporciona un **instalador**.

Resumen

%Installer es una herramienta conveniente para distribuir e implementar aplicaciones basadas en InterSystems Caché y Ensemble.

Referencias

Documentos de apoyo

#Administración del sistema #Despliegue #Herramientas #Mapeo #Terminal #Caché

URL de fuente: https://es.community.intersystems.com/post/despliegue-de-aplicaciones-con-installer