Copermática Software

Oficina Principal
Pº San Isidro, 15
Tfno: 902 44 33 33
Fax: 926 51 49 87
13700 Tomelloso (CR)

Ciudad Real
Plaza Mayor, 14
Edificio Alcor 1º A-2
Tfno: 902 44 33 33
13001 Ciudad Real

Madrid
Orense, 85
Edificio Lexington
Tfno: 902 44 33 33
28020 Madrid

copermatica@copermatica.com www.copermatica.es

Español

RobotBuilds

Objetivo

Este proyecto tiene como objetivo facilitar la labor de compilación en Visual Basic 6.0 a los desarrolladores de software que aún mantienen soluciones en este tipo de entornos de programación, gracias a RobotBuilds podremos realizar compilaciones de uno o varios proyectos siguiendo un orden de compilación para librerías y ejecutables y desentendernos totalmente de dichas tareas con el consiguiente ahorro de tiempo y de recursos.

Solución

La solución está desarrollada con Microsoft Visual Studio 2008, y está formada por tres componentes:

- Builds.dll -> Librería donde se detalla toda la funcionalidad para las compilaciones y archivos de proyectos.
- RobotBuilds.exe -> Ejecutable modo consola que consume la librería anteriormente mencionada para realizar todo tipo de operaciones como compilaciones, ordenación de referencias, escritura de archivos, copia de archivos y notificaciones mediante correo electrónico de los resultados de las compilaciones.
- RobotConfig.exe -> Ejecutable de tipo formulario para facilitar la configuración del RobotBuilds, permitiendo almacenar la configuración en el archivo **'RobotBuilds.ini'**.

RobotBuilds es una solución en entorno consola que ofrece distintas posibilidades mediante el uso de argumentos, a continuación detallamos las más importantes:

- Es posible realizar la compilación de un único proyecto (.vbp) o de una lista de proyectos (.txt):
    - RobotBuilds /Start <Ruta de archivo de proyecto '.vbp'>
    - RobotBuilds /Start <Ruta de archivo de lista de proyectos '.txt'>

Antes de proceder a la compilación de proyectos es necesario tener configurado algunas propiedades desde la configuración del robot (RobotConfig), para ello es necesario configurar:

Compiler File Path: (Obligatorio) Aquí debemos especificar la ruta donde se encuentra el compilador de Visual Basic, normalmente el compilador se encuentra accediendo a la carpeta de archivos de programas, Microsoft Visual Studio, VB98 y seleccionado el ejecutable VB6.exe

Builds Directory: (Obligatorio) Aquí indicamos el directorio donde queremos que nos aloje los archivos binarios resultantes de las compilaciones realizadas sobre nuestros proyectos de Visual Basic.

VBP Out Directory: (Opcional) Aquí indicamos el directorio donde queremos que nos aloje los archivos de proyecto '.vbp' que han sido compilados, hay casos en los que las compilaciones modifican estos archivos y con esta opción podemos tener todos los archivos de proyecto modificados en un solo sitio. En el caso de dejar esta propiedad vacía los archivos de proyectos serán modificados directamente desde la ubicación donde se están compilando.

Hay dos tipos de compilaciones que puede realizar nuestra solución sobre los proyectos ya sea para un único proyecto o para una lista de proyectos, en cualquier caso será necesario ir a la configuración del robot (RobotConfig) y realizar una de las siguientes dos opciones:

1) Establecer la propiedad *Modify Project Properties* a verdadero.
   En este caso la compilación de componentes ActiveX de tipo DLL y EXE se compilan sin compatibilidad, mientras que los controles ActiveX de tipo OCX se compilan utilizando compatibilidad de proyecto (para los controles OCX es indispensable realizar compilación manteniendo compatibilidad). Para los proyectos ejecutables EXE la compilación se realiza sin compatibilidad.

Cuando tenemos configurado la propiedad *Modify Project Properties* a verdadero, podemos establecer ciertas propiedades a todos nuestros proyectos que se quieran compilar, las propiedades que se pueden definir a la hora de realizar la compilación de cada proyecto son las siguientes:

Version Legal TradeMarks: Aquí podemos establecer nuestras marcas legales, las cuales quedarán reflejadas en los archivos binarios que se generen una vez compilados.

MajorVer: Número mayor versión para nuestros archivos binarios generados.

MinorVer: Número menor versión para nuestros archivos binarios generados.

RevisionVer: Número de revisión/edición para nuestros archivos binarios generados.

El archivo de proyecto '.vbp' es modificado con los cambios realizados en estas propiedades.

2) Establecer la propiedad *Modify Project Properties* a falso en la configuración.

   En este caso la compilación de componentes ActiveX de tipo DLL, EXE, OCX y los archivos ejecutables EXE se compilan según esté especificado en las propiedades del proyecto definidas en el entorno de programación de VB6.

   El archivo de proyecto '.vbp' no es modificado.

Cuando la compilación se realiza sobre una lista de proyectos podemos marcar en la configuración del robot (RobotConfig) la propiedad *Continue On Error* a verdadero en tal caso si la compilación de un proyecto no ha sido válida continua con el siguiente proyecto de la lista. En el caso de tener la propiedad marcada a falso si la compilación de un proyecto es no válida entonces se detiene la ejecución en dicho proyecto.

NOTA: Algunas veces es necesario establecer unos requisitos previos que se deben cumplir antes de realizar una compilación de un proyecto, por ejemplo, existen controles ActiveX generados desde proyectos de .NET(Interop) cuya extensión es TLB, como estos proyectos se compilan

desde el compilador que proporciona Visual Studio .NET, existe la posibilidad que estos archivos TLB sean requisitos previos a la compilación de proyectos de Visual Basic para ello es necesario rellenar el archivo ***'Requirements.txt'*** con tantas líneas como rutas de archivos TLB queremos tener como requisitos.

- Antes de realizar la compilación de proyectos, disponemos de una utilidad con la que podremos ahorrarnos muchísimo tiempo para proyectos de Visual Basic de gran envergadura y con múltiples referencias a nuestras librerías, esta utilidad permite ordenar las referencias de los componentes Activex (DLL, EXE y OCX) en los archivos de proyectos. Para poder ordenar las referencias a los proyectos es necesario tener disponible una lista guía en la que tengamos establecido el orden de las referencias para nuestros proyectos, esta lista guía debe ir en un archivo llamado ***'ReferencesOrder.txt'***.
    - RobotBuilds /SortReferences <Ruta de archivo de proyecto '.vbp'>

  Ejemplo: Supongamos que nuestro proyecto de Visual Basic, es un proyecto EXE que tiene tres referencias a nuestras librerías (LibCalc.dll, LibCustomers.dll y LibSales.dll), para establecer un orden de referencias necesitamos tener una guía de ordenación de referencias en nuestro caso el fichero ReferencesOrder.txt debería contener las siguientes líneas:

  LibCustomers.dll
  LibCalc.dll
  LibSales.dll

  Si el contenido del archivo de orden de referencias fuera el descrito anteriormente, nuestro proyecto EXE sería ordenado de tal forma que la librería de clientes sería la primera, la librería de cálculos la segunda y la librería de ventas la tercera.

  NOTA: Para el resto de componentes (DLL, EXE y OCX) que estén incluidos en el proyecto EXE pero que no aparezca en la lista guía de orden de referencias serán ignoradas y se dejarán en su orden original.

- Una vez que se ha compilado distintos proyectos, nos puede interesa realizar operaciones de limpieza tanto de los archivos '.log' creados en las compilaciones como de los archivos binarios generados.
    - RobotBuilds /Clean
    - RobotBuilds /CleanLogs

  La primera opción borra los archivos binarios generados en el directorio *Builds Directory* y limpia las claves del registro de Windows para los componentes ActiveX. La segunda opción borra el log de errores de compilación para la fecha actual.

- Nuestra solución dispone también de una utilidad para modificar la fecha/hora de última modificación de un archivo, esto es útil cuando queremos que todos los archivos que queremos que vayan en una misma revisión/edición para nuestros clientes tengan indicado la misma fecha/hora por motivos de claridad y detección de problemas.
    - RobotBuilds /WriteTimeFile <ruta de archivo>
    - RobotBuilds /WriteTime <ruta de directorio>
    - RobotBuilds /WriteTimeRecursive <ruta de directorio>

  La primera opción escribe la Fecha/Hora establecida en la configuración del robot (RobotConfig) en las propiedades *Last Write Date* y Last Write Time al archivo indicado. La segunda opción permite cambiar la Fecha/Hora a todos los archivos contenidos en el directorio indicado. La tercera opción permite cambiar la Fecha/Hora a todos los archivos y subdirectorios de forma recursiva a partir del directorio indicado.

- Muchas veces es necesario que el resultado de una o varias compilaciones sea copiado a un directorio destino donde se realiza el empaquetado de la revisión/edición que entregamos a los clientes.
    - RobotBuilds /Copy <ruta directorio origen> <ruta directorio destino>
    - RobotBuilds /CopyBuilds <ruta de directorio>

  La primera opción permite copiar de forma recursiva todos los archivos y subdirectorios contenidos en el directorio origen al directorio destino, en caso de no existir los directorios en el destino se crean y en el caso de existir los archivos serán reemplazados. La segunda opción únicamente copia los archivos binarios generados en el directorio *Builds Directory* al directorio indicado.

- RobotBuils permite informar a una o varias personas (departamento de desarrollo) el resultado de las compilaciones realizadas, para ellos nos facilita una opción de notificación mediante correo electrónico en la que se informará si las compilaciones tuvieron éxito o si se produjo algún error en alguna compilación.
    - RobotBuilds /Notify <E-mails separados por ';'>

  Para tener disponible esta opción es necesario configurar las propiedades referentes al servidor de correo saliente SMTP y los textos de notificaciones que se deben mostrar en los correos electrónicos desde la configuración del robot (RobotConfig), las propiedades que se deben configurar son las siguientes:

  Server SMTP: Servidor de correo saliente. Ejemplo: smtp.server.com

  Port: Puerto para el servicio de envío de correo. Ejemplo: 25

  User: Identificador de usuario para validarse en el servidor. Ejemplo: User@server.com

  Password: Contraseña del usuario.

  User Name Send: Nombre de usuario que queremos mostrar en el correo que se envía, por defecto: "RobotBuilds".

  Secure Connection: Hay que indicar si el servidor acepta conexiones seguras ó no.

Responding To: Aquí se indica el correo electrónico que aparece cuando el que recibe el correo pulsa en 'Responder'. Ejemplo: RobotBuilds@server.com

Message Description: Texto que se muestra en el Asunto del correo electrónico.

Message Content Without Errors: Texto que se muestra en el contenido del correo electrónico cuando el resultado de la compilación ha sido satisfactorio.

Message Content With Errors: Texto que se muestra en el contenido del correo electrónico cuando el resultado de la compilación no ha sido correcto y ha generado errores.