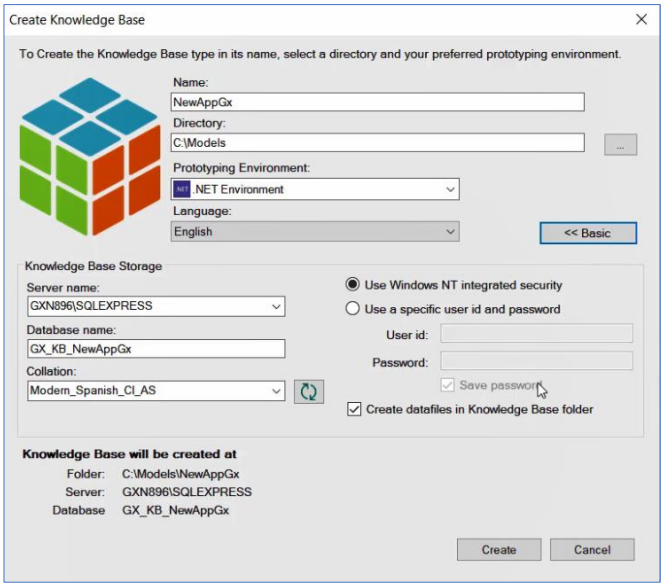
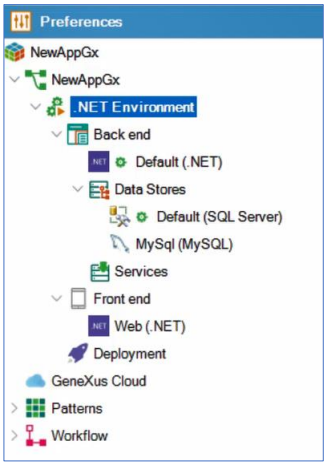
Setting some properties

GeneXus™





Cuando creamos una nueva KB, como sabemos, desde las opciones básicas podremos configurar: su nombre, ubicación, lenguaje de programación, e idioma utilizado para las etiquetas, botones, mensajes, etcétera. Y en opciones avanzadas se podrá cambiar el nombre y ubicación del servidor de base de datos donde se encontrará la base de datos de la KB, el nombre de esa base de datos, la intercalación, y si se utilizará el usuario de Windows o un usuario específico para el acceso a la base de datos.

Una vez creada nuestra KB, veamos desde la ventana de “preferences” las opciones que se nos presentan.
Bajo el nodo versión de nuestra base de conocimiento, vemos el ambiente creado automáticamente a partir de las definiciones iniciales.
Dentro se encuentran los siguientes nodos:

- **Backend**, donde estará todo lo relacionado con el servidor de aplicaciones, los accesos a la base de datos y los servicios.
En el backend se definen los lenguajes de programación que se utilizaran para generar el código correspondiente al backend de la aplicación, esto por intermedio de los generadores.
Luego en Data Stores estará la información para acceder a la base de datos asociada.

También es posible generar nuevos Data Stores con diferentes DBMS, por ejemplo para obtener datos desde otras bases de datos externas.
Desde el nodo Servicios, podremos configurar el manejo de los servicios, configuraciones de almacenamiento, notificaciones, etc.

- Y luego tenemos el nodo **Frontend**, donde se encuentra todo lo relacionado con la interfaz de usuario y se conecta con el backend.
Se muestran aquí los generadores disponibles para crear el Frontend de la aplicación.
De forma predeterminada nos aparecerá el lenguaje que hayamos seleccionado en el environment, .Net, .Net Framework o Java, y dependiendo del tipo de aplicación que desarrollemos pueden aparecer como generadores: Android, Apple y Angular.

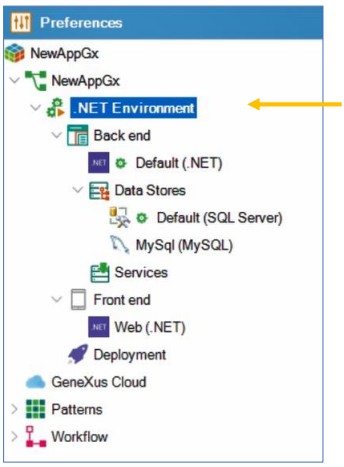
Y por último tenemos el nodo Deployment, que se encuentra lo relacionado a los Deployment Unit objects definidos en la KB Version.
Tenemos la posibilidad de crear nuevos ambientes. Por ejemplo en desarrollos donde tengamos un ambiente de pruebas y otro de producción, tenemos que tener en esos casos más de un ambiente.

Supongamos que creamos uno nuevo con generador Java y base de datos Oracle, y queremos que sea nuestro generador por defecto para nuestra aplicación. Podremos, o bien directamente marcar que va a ser nuestro environment por defecto cuando creamos el ambiente, o podremos hacerlo luego haciendo clic derecho sobre el ambiente deseado y seleccionando “Set as current Environment”, lo mismo para cambiar de un ambiente a otro.
Para ejecutar una aplicación Java Web, debemos previamente contar con el Java Development Kit (JDK) instalado, el cual podemos descargar directamente desde la página oficial de Oracle.
Y también será necesario un servidor web donde ejecutará nuestra aplicación, en este caso utilizaremos el Apache Tomcat, un software de código abierto, que se puede descargar directamente desde su sitio oficial.

Una vez instalado podemos verificar desde la app de Windows “Servicios”, que se está ejecutando correctamente, en caso contrario podemos iniciarlo desde allí. Aquí vemos el Apache Tomcat 9, que es el que tenemos instalado en esta PC, corriendo correctamente.

GeneXus

Some properties of the KB Environment node



Veamos ahora algunas propiedades configurables de los nodos que comentamos hace un momento
En el nodo versión de la KB, una de las propiedades que vemos es de nombre Enable Integrated Security. Esta propiedad establece si la aplicación generada gestionará o no la seguridad mediante GAM (GeneXus Access Manager), por defecto aparece en false.

También encontramos propiedades que tienen impacto en la interfaz de nuestra aplicación.
Podremos por ejemplo especificar el objeto theme o design system que queremos se aplique por defecto a los diferentes objetos transacción y/o Web Panel que creemos, aunque luego desde cada uno de esos objetos podamos especificar qué tema o design system object queremos se le aplique.

También podremos definir la Master Page que se aplicará por defecto, que de forma predeterminada apunta al objeto RwdMasterPage. Al igual que con la propiedad anterior, cada Transacción y Web Panel podrán configurar esta propiedad de forma independiente.

También podremos configurar algunas propiedades referentes a las validaciones que se realizan del lado del cliente, como la propiedad stop

on error: Por defecto está en No, pero si se cambia a Sí, cuando en un formulario se validan las reglas que tengamos definidas y encuentre alguna entrada incorrecta, se mantendrá el foco en ese campo hasta que se corrija, no permite seguir avanzando al siguiente hasta corregirlo. Así como también podremos definir la posición de los mensajes de validación, entre otras configuraciones.

Dentro de la sección Defaults tenemos la propiedad automatic refresh, la cual está por defecto en sí, y eso nos permite que cuando tengamos un grid y se realicen cambios en cualquier variable que esté siendo utilizada por el evento Load, el evento Refresh o las conditions del grid, se realizará una actualización automática de la grilla sin necesidad de que el usuario lo haga manualmente. El ejemplo más común de este caso es cuando tenemos filtros que impactan en la grilla. Si no se desea este comportamiento se puede pasar a No.

En el nodo del ambiente de la KB, en la sección Environment tenemos la posibilidad de modificar la plataforma para la cual se generará la aplicación, Web o Windows. Podremos también desde aquí modificar el lenguaje de programación elegido.
También podremos cambiar el DBMS asignado.

Desde la propiedad Startup Object podemos asignar un objeto de inicio para que sea el que se ejecute cuando se lanza la aplicación, o sea cuando presionamos F5.
También podemos desde esta propiedad modificar el nombre del environment.
Vemos aquí la propiedad commit on exit, desde ella seteamos el valor que tendrá por defecto para cada objeto transacción o procedimiento que creemos. Por defecto está en Yes, lo que significa que el programa generado ejecuta una confirmación, o sea un commit, al final de la unidad lógica de trabajo (LUW). Si la cambiamos a No, no se realizará esta confirmación.

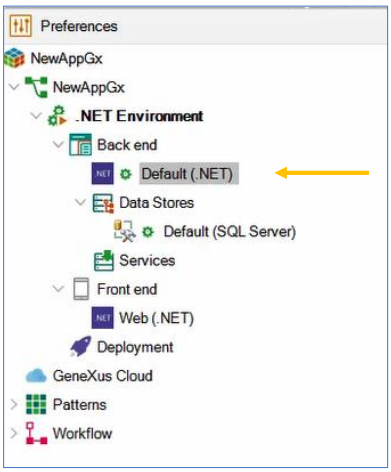
También tenemos la propiedad llamada Encrypt Url Parameteres. Con esta propiedad podremos permitir o denegar el cifrado de los parámetros enviados a una URL, y establecer niveles de seguridad cuando se utiliza el cifrado de los parámetros entre Objetos Web. Por defecto está seteada en No, o sea que los parámetros en la URL de los objetos web no se cifrarán. La opción session key indica que los parámetros de la URL se cifrarán con una clave diferente para cada sesión. El cifrado se realiza mediante cookies locales. Este valor ofrece un mayor nivel de seguridad, pero no

permite URL compartidas. Esto significa que un usuario no podrá enviar una URL con parámetros a otro usuario porque la URL no funcionará, ya que se requiere la cookie correspondiente para el descifrado.

La última opción es Site Key, si seleccionamos esta los parámetros en la URL de los objetos Web estarán encriptados, pero la clave de encriptación será la misma para todo el sitio. En este caso, no se utilizan cookies. Esto implica un menor nivel de seguridad, pero facilita la transferencia de enlaces.
La propiedad protocol specification nos permite especificar bajo qué protocolo se ejecutará la aplicación o los servicios Web. Por defecto será accesible por HTTP, pero podemos cambiar a HTTPS. Si lo dejamos sin especificar, la aplicación no aplica ningún protocolo, pudiendo acceder a través de HTTP o HTTPS indistintamente.

GeneXus

Some properties of the back-end generator



En cada generador, también tenemos varias propiedades configurables y estas dependen del lenguaje elegido. Veamos algunas por ejemplo de .Net.
Podremos especificar la plataforma en la cual se generará la aplicación, web o Windows.
Mediante estas propiedades (Log Level / User Log Level) podremos configurar el detalle de la información que se guardará en el log con una escala de 0 a 6, siendo 0 la opción off, o sea que no se harán registros, y la opción 6 “All” la que registrará toda la información posible.

La propiedad reorganize server table se utiliza para indicar si las tablas del modelo van a ser reorganizadas o no por el generador que tengamos asignado como reorganizador del modelo. Por defecto viene seteado en Yes. Si se deja en No, cuando el Informe de Análisis de Impacto de la base de datos indique que las tablas necesitan ser reorganizadas, será ignorado por el generador.

La propiedad transactional integrity, por defecto seteada en Yes, permite que todas las Transacciones y Procedimientos se generen con integridad transaccional, si se desea lo contrario puede ser modificada a No.

Si elegimos el generador Java, tenemos por ejemplo la propiedad

Compiler Options, aquí se podrá incluir cualquier propiedad válida para el compilador de Java, según su versión.
Para obtener más información sobre las propiedades válidas les recomendamos acceder a la información en el sitio de documentación de Oracle.
https://docs.oracle.com/javase/9/tools/javac.htm#JSWOR627 .

Estas otras propiedades (Reorganization Option / Create Database Option) determinan si el proceso de creación/reorganización de la base de datos tendrá una intervención del usuario o tomará otras acciones en la reorganización.

Los posibles valores a ingresar son:

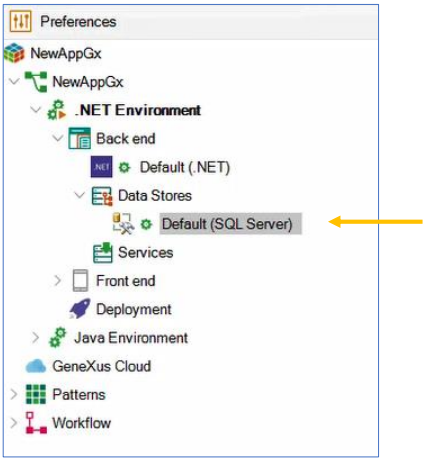
-nogui, es el que viene por defecto, y determina que no se permite la intervención del usuario en estos procesos. Por ejemplo, en caso de ejecutar en alguna plataforma diferente a Windows y que no cuente con una interfaz gráfica, se puede presentar un error al reorganizar, y de esta manera, con el comando -nogui evitamos la necesidad de uso de la interfaz gráfica y solucionar dicho error.
-force. En GeneXus tenemos archivos de extensión .Gen y .Exp, los .Gen son utilizados para ejecutar la reorganización, y los .Exp para controlar si la reorganización fue ejecutada o no. Con este comando el control sobre estos archivos no es realizado, y la creación/reorganización de la base de datos es forzada de cualquier modo a realizarse. Su uso no es recomendado.
-noverifydatabaseschema, no verifica el esquema de la base de datos en el proceso de creación/reorganización de la misma. Nos puede resultar útil por ejemplo en algún caso donde se presente un error de reorganización en el proceso de verificación del esquema.
-recordcount, este parámetro contará los registros que se crearán/reorganizarán en todas las tablas, mostrará los resultados y se detendrá. En lugar del botón de reorganizar nos aparecerá uno de continuar. Esto puede usarse para estimar cuánto tiempo puede tomar la tarea.
-ignoreresume. Supongamos que se ejecutó una reorganización y falló. Luego se realizan algunos cambios en transacciones que implican actualizar la base de datos. Al ejecutar se requiere una nueva reorganización y se produce otro error.
Esto es porque GeneXus realiza las reorganizaciones consecutivamente. Si falla una reorganización e intenta realizar otra, aún faltará la reorganización anterior.

GeneXus provee rutinas para continuar desde el último punto ejecutado por la reorganización antes de una falla.
Para evitar la ejecución de estas rutinas de validación y ejecutar el proceso de reorganización desde el principio. se ingresa el comando -ignoreresume.
-donotexecute, es útil para entornos en los que necesita que la creación/reorganización se genere pero no se ejecute y, una vez realizado el impacto, desea realizar una compilación completa e implementar la creación o reorganización y los programas de aplicación en otro lugar.

Vale aclarar que los comandos se pueden combinar separando cada opción con un espacio en blanco. Por ejemplo, si desea combinar las opciones nogui y force quedaría de esta manera.
Otras propiedades configurables son sobre la ejecución de nuestra aplicación. Por ejemplo, setear si la prototipación será en la nube o no, y en caso de seleccionar que sí, cuál será la URL del servidor.

Podremos también modificar el servidor Web que se utilizará, entre otras opciones.

Some properties of the database generator



Para los generadores de base de datos, las propiedades dependen también del generador elegido. Por ejemplo para el caso de SQL Server: Tenemos la propiedad Access technology to set, que nos permite configurar las posibles tecnologías para el acceso a la base de datos. Por ejemplo ADO.Net, usada en el generador .NET o .NET Framework, o JDBC para el generador Java. Según cual seleccionemos, son las propiedades que nos va a habilitar para configurar en Connection Information.

Dentro de esta sección (Connection Information), tenemos por ejemplo la propiedad Database name, la cual se utiliza cuando los entornos necesitan nombres de bases de datos. Indica el nombre de la base de datos que contendrá las tablas e índices de la aplicación.

Tenemos también la propiedad Server Name, la cual especifica el nombre predeterminado que identificará al servidor de base de datos. Si no se especifica este parámetro, los programas generados mostrarán un mensaje en la pantalla solicitando que se ingrese el nombre del sistema cuando sea el momento de establecer la conexión.
Podremos también especificar el puerto que utilizará la instancia para comunicarse e intercambiar datos con las aplicaciones.

La propiedad Connect to server permite controlar en qué momento los

programas generados establecen conexión con la base de datos. La opción por defecto es “At first request” lo que significa que la aplicación intentará establecer la conexión inmediatamente después de enviar la primera solicitud al servidor de base de datos.

La otra opción es “At application startup”, en este caso la aplicación intentará establecer la conexión como parte del proceso de inicialización, antes de que se muestre el objeto principal.
Un ejemplo para entender la diferencia, podría ser cuando tenemos un procedimiento que realiza cálculos sin conexión con la base de datos, y si se cumple cierta condición se llama a otro procedimiento quien sí realiza conexión con la base de datos. Con “At first request” se va a establece conexión con ella al inicializar la aplicación, sin saber si se va a hacer uso o no del segundo procedimiento. Y con “At application startup” establece conexión con la base de datos sólo en el caso que se cumpla con la condición y se llame al segundo procedimiento, que es quien requiere conexión con el servidor. Esta última opción se recomienda en aplicaciones que dependen totalmente del servidor de base de datos.

La propiedad Database schema, permite apuntar a un esquema de base de datos específico.
Un esquema de base de datos es quien describe su estructura, es un modelo o arquitectura de cómo se verán nuestros datos. En una base de datos relacional, el esquema define sus tablas, sus campos en cada tabla y las relaciones entre cada campo y cada tabla.

Estas que vimos son algunas de las propiedades configurables de nuestra KB, como vemos hay muchas más que no mencionamos su funcionalidad en este video.

Si nos posicionamos en una alguna propiedad, y presionamos la tecla “F1” nos lleva a la Wiki de GeneXus y específicamente a la descripción y uso de la propiedad elegida. También podemos ingresar directamente a la wiki y buscar por el nombre de la propiedad. Tener en cuenta que el atajo con la tecla “F1” puede en algunas propiedades no estar disponible aún.

GeneXus™

training.genexus.com
wiki.genexus.com