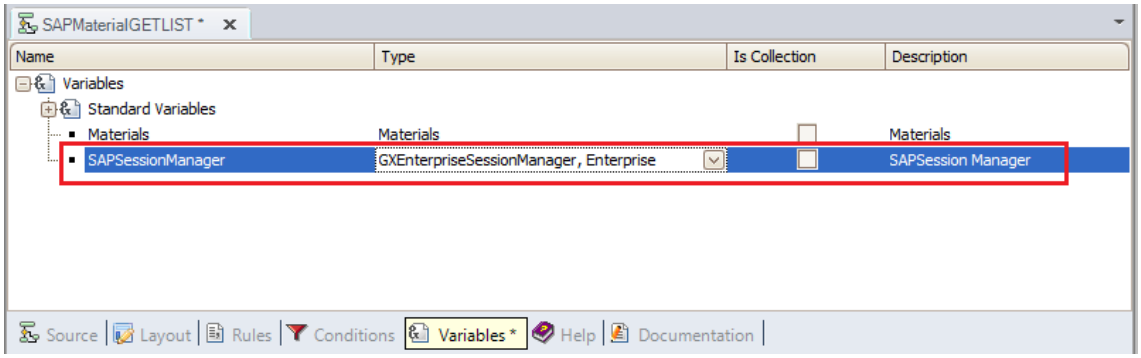
Creando una aplicación móvil con GeneXus para trabajar con los Materiales del SAP ERP – parte 3

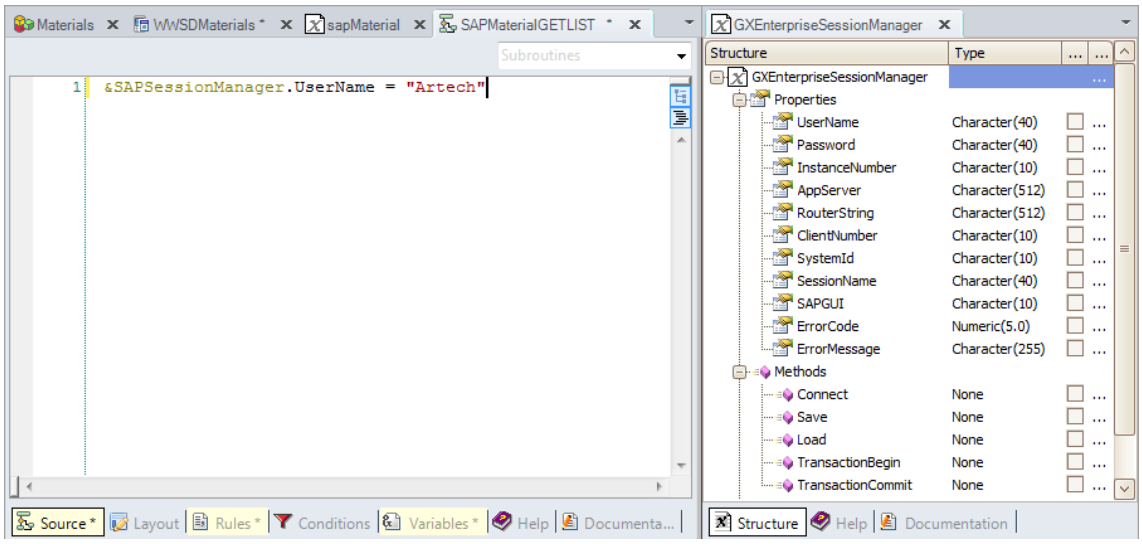
En este ejemplo, para simplificar las cosas, utilizaremos información hardcodeada. Luego veremos cómo permitir que el usuario ingrese en runtime esta información a través de una pantalla.

Empezamos por definir una variable, SAPSessionManager, del tipo de datos (del módulo Enterprise) el External Object GXEnterpriseSessionManager:

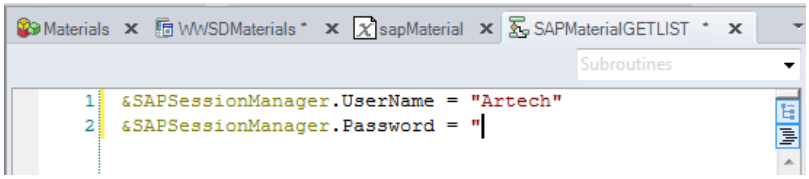


Definir una variable con tipo de datos un external object hace que ésta se defina con la estructura de datos de ese objeto: en nuestro caso, las propiedades UserName, Password, InstanceNumber, AppServer, etcétera, que corresponden a los datos de conexión al ERP) y también hace que se puedan aplicar sus métodos sobre esta instancia.

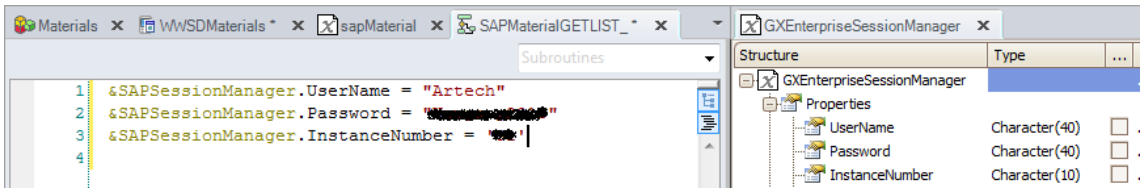
Entonces, vamos al Source del procedimiento, y a esta variable le asignamos valor para todas las propiedades de conexión, empezando por UserName, el usuario con el que nos conectaremos al ERP…



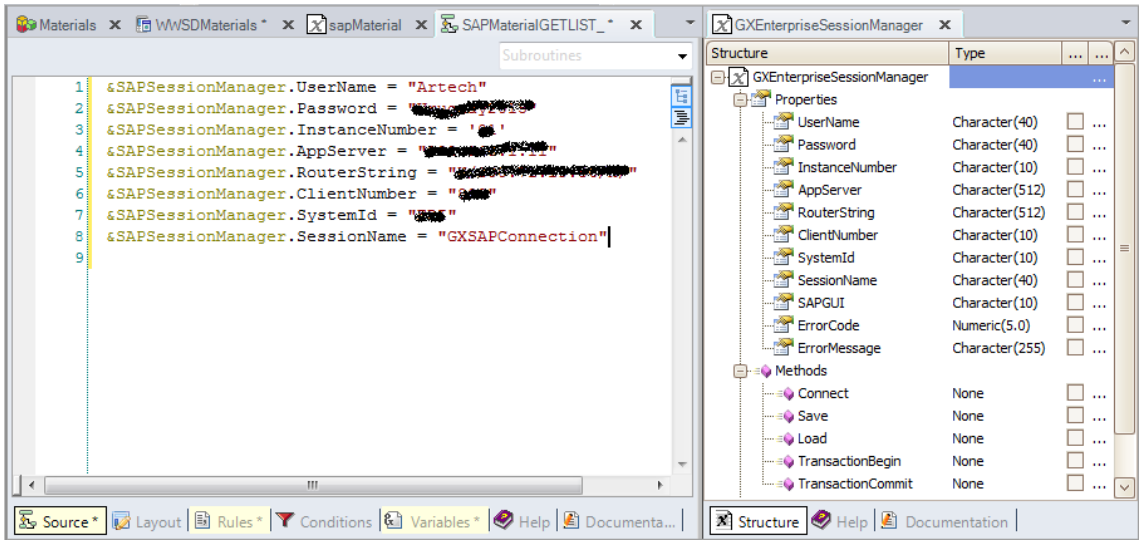
Su password:



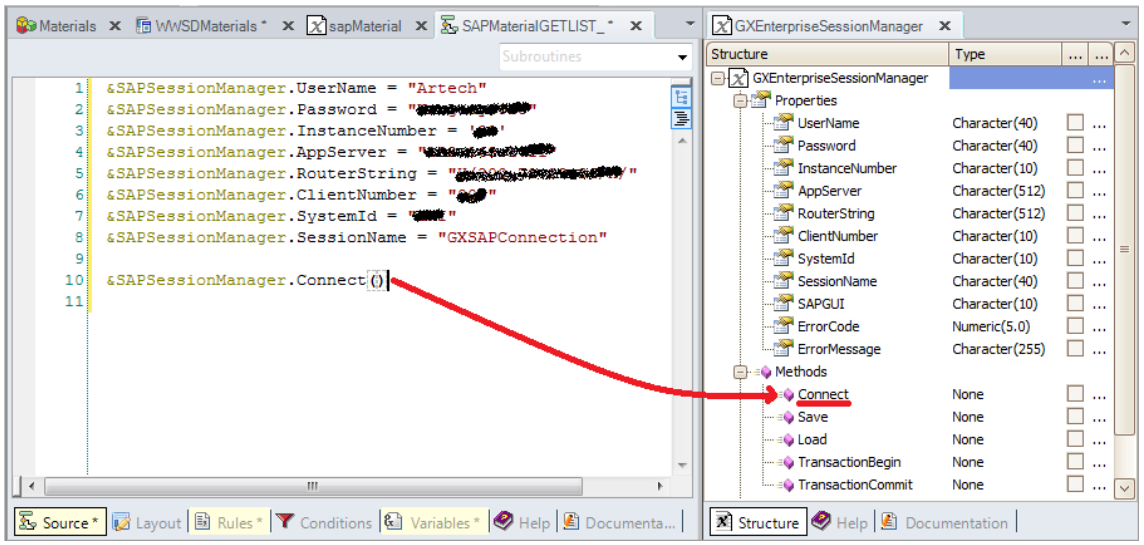
El número de instancia:



Y así sucesivamente hasta completar todos los parámetros de conexión.

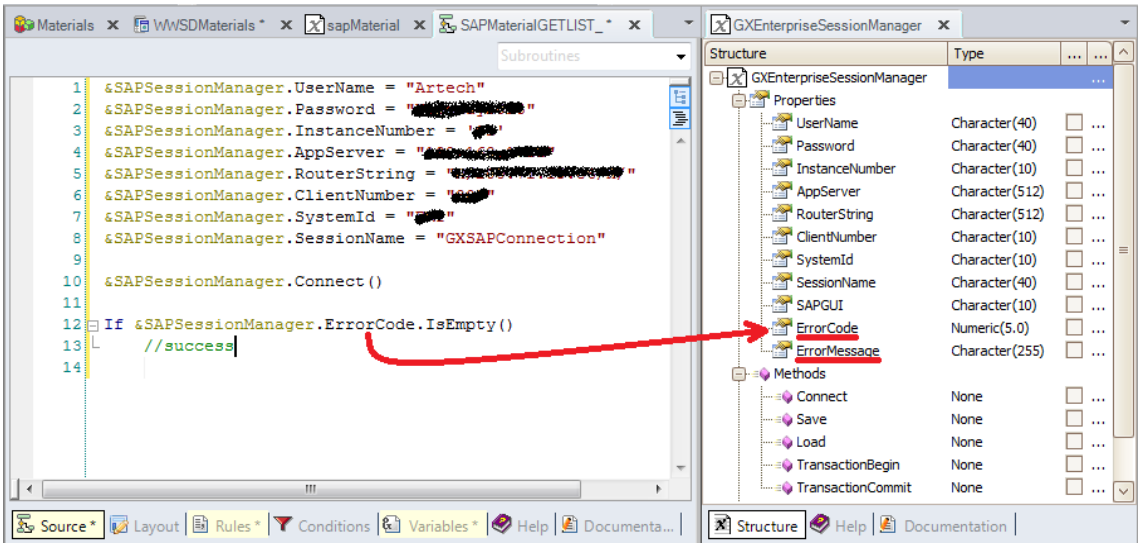


Ahora que tenemos en las propiedades de la variable los datos de conexión, ejecutamos el método Connect:



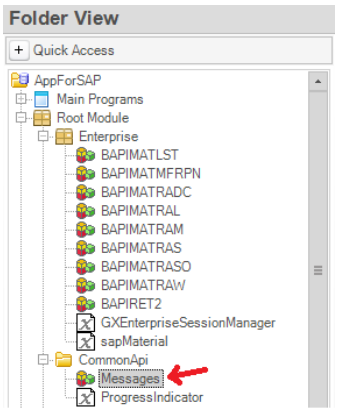
Si el intento de conexión falla, el error producido queda cargado en las propiedades ErrorCode y ErrorMessage.

Así que preguntamos si no hubo error, es decir, si ErrorCode está vacío, y allí programaremos las acciones a tomar cuando la conexión fue exitosa (con barra barra lo que sigue es un comentario):

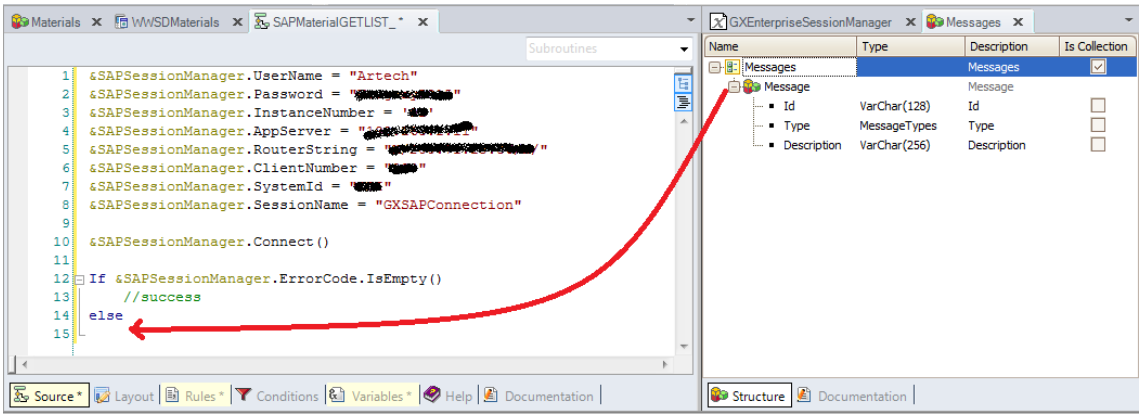


De lo contrario, else, querremos devolver al panel que invoca a este procedimiento el mensaje de error, que se encuentra en la propiedad ErrorMessage del external object.

Una opción prolija para ello es utilizar el Structured Data Type **Messages** que se crea automáticamente en toda Knowledge base**:**



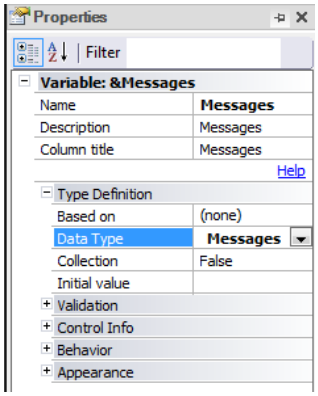
para ser utilizado justamente para almacenar mensajes de error y de advertencia.



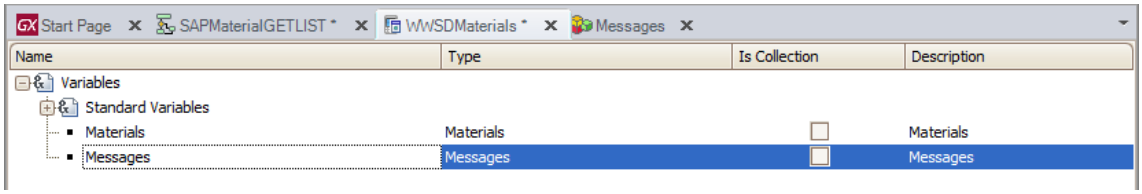
Por lo que agregaremos otra variable de salida a nuestros parámetros, para devolver todos los mensajes que se produzcan como resultado de la ejecución del objeto:

```
parm(out: &Materials, out: &Messages);
```

La definimos, vemos que por haberla llamado igual, asume por defecto el tipo de datos Messages:



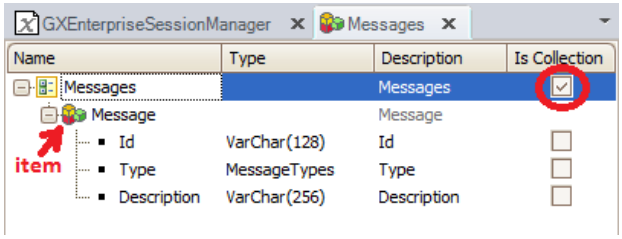
Vamos al panel para también definirla allí



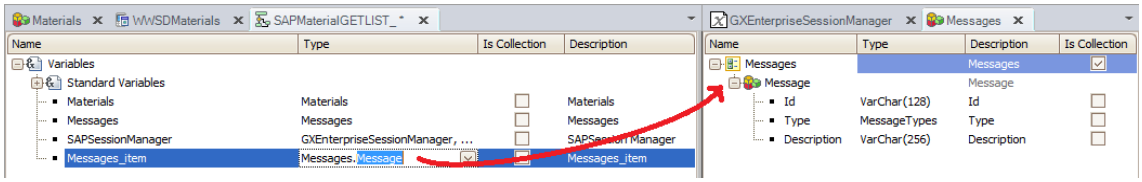
Y agregarla a los parámetros de la invocación al procedimiento:

```
Event Refresh
      SAPMaterialGETLIST(&Materials, &Messages)
Endevent
```

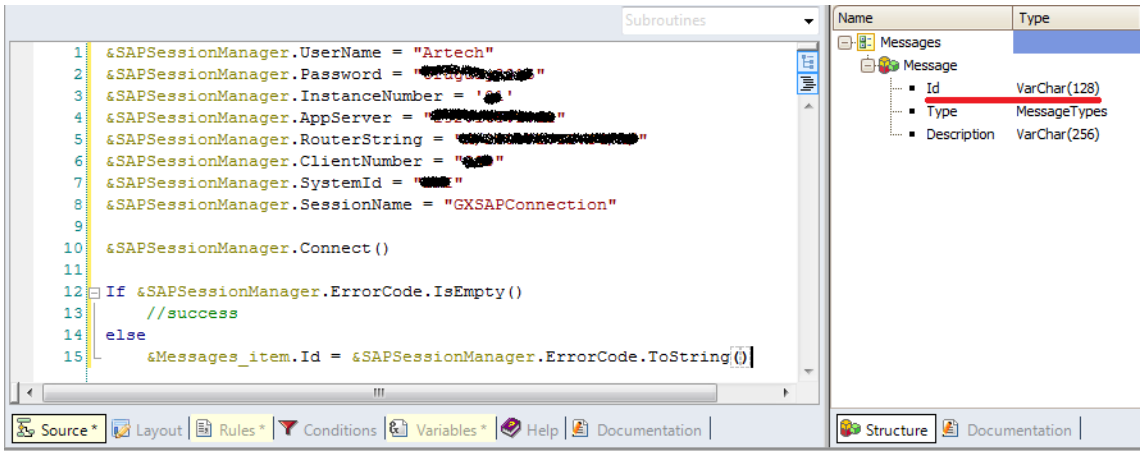
Ahora tenemos que cargarla en el procedimiento. Para ello, como es una colección de ítems, ellos mismos estructurados:



creamos otra variable que represente un ítem individual: Messages_item, del tipo de los ítems:



Y ahora en el Source, la cargamos. Al Id le asignamos el ErrorCode de &SAPSessionManager. Pero como vemos, Id es de tipo VarChar, y ErrorCode Numeric, así que convertimos el numeric a String:

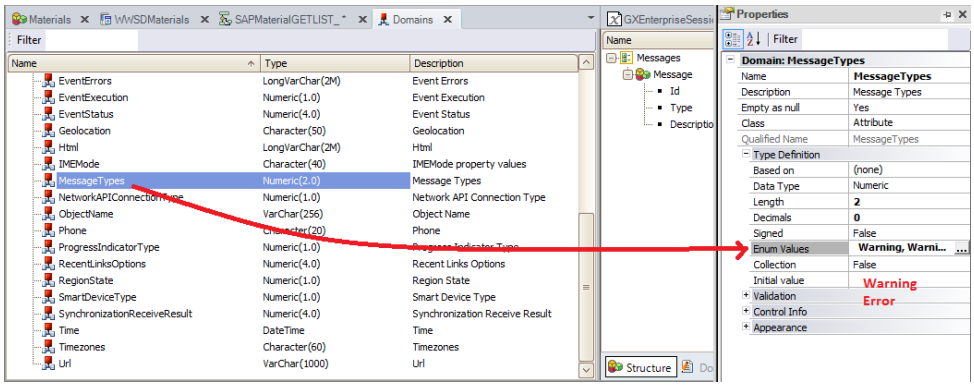


Luego a Description, Varchar, le asignamos la propiedad ErrorMessage:

```
If &SAPSessionManager.ErrorCode.IsEmpty()
    //success
else
    &Messages_item.Id = &SAPSessionManager.ErrorCode.ToString()
    &Messages_item.Description = &SAPSessionManager.ErrorMessage
```

Y por último tenemos que indicar el tipo de mensaje, que como vemos tiene un tipo de datos predefinido, MessageTypes, que es un dominio enumerado.

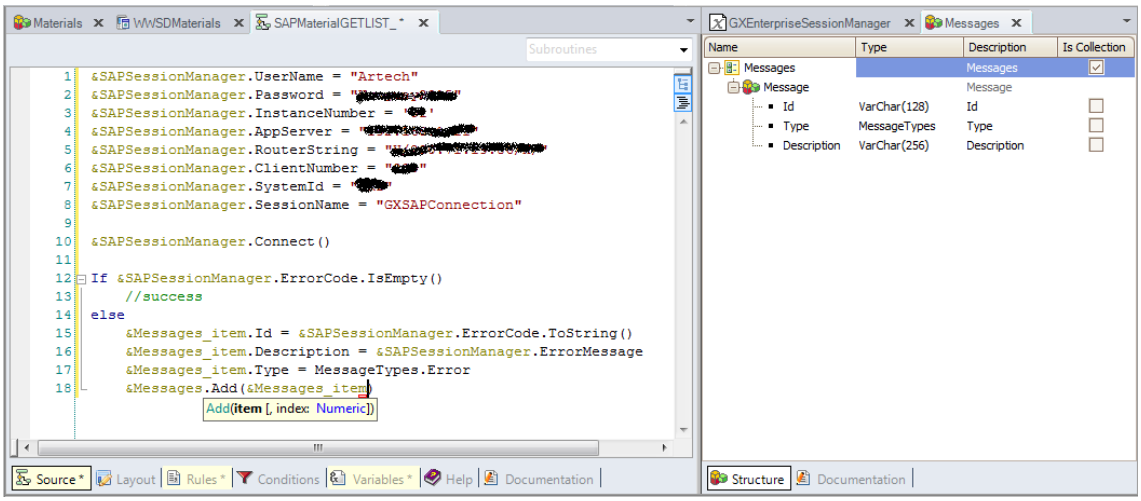
Todos los dominios definidos en la base de conocimiento se ven aquí. Si vemos las propiedades de MessageTypes tiene dos valores: Warning y Error.



El tipo en nuestro caso será Error:

```
If &SAPSessionManager.ErrorCode.IsEmpty()
    //success
else
    &Messages_item.Id = &SAPSessionManager.ErrorCode.ToString()
    &Messages_item.Description = &SAPSessionManager.ErrorMessage
    &Messages_item.Type = MessageTypes.Error
```

Lo que nos resta es agregar este ítem a la colección de mensajes:



Y cerramos el comando if:

```
If &SAPSessionManager.ErrorCode.IsEmpty()
        //success
else
        &Messages_item.Id = &SAPSessionManager.ErrorCode.ToString()
        &Messages_item.Description = &SAPSessionManager.ErrorMessage
        &Messages_item.Type = MessageTypes.Error
        &Messages.Add(&Messages_item)
endif
```

Ahora sí, si no hubo error, querremos obtener el listado de materiales, por lo que tendremos que invocar al método GETLIST de sapMaterial.

Lo haremos en el siguiente video…