



NINO SEISEI CODE GENERATOR

Contenido

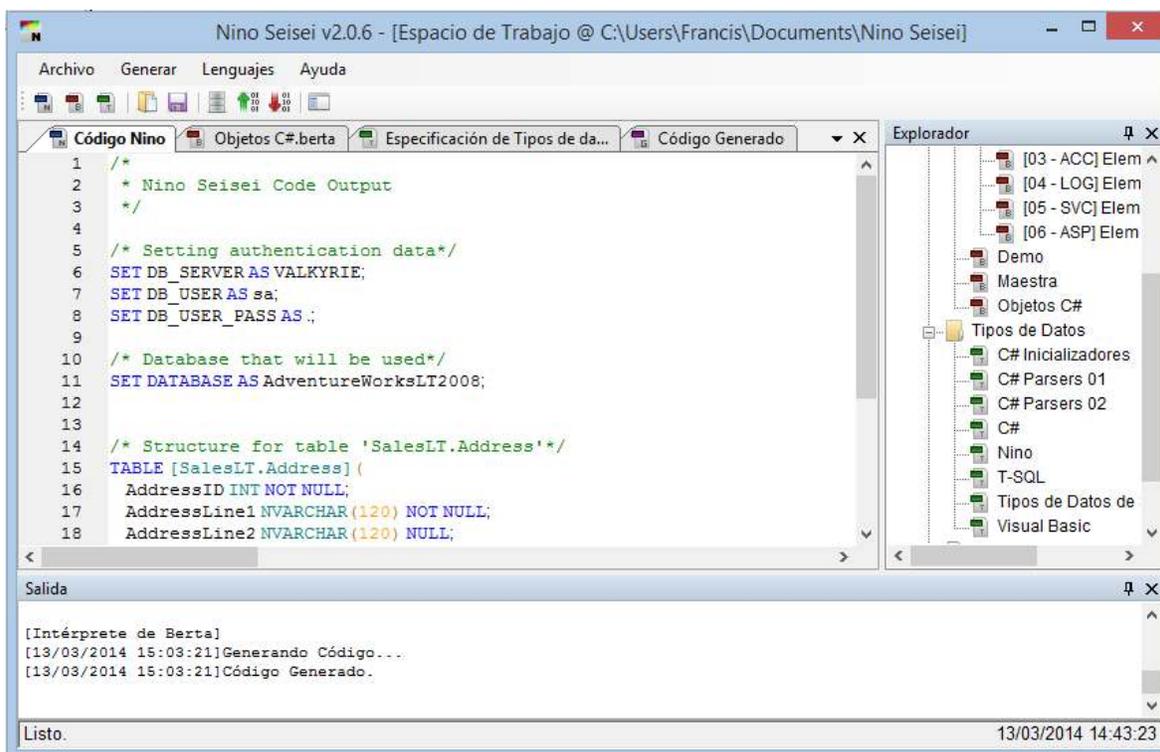
Introducción	2
Guía de Inicio Rápido	3
Partes de la aplicación.....	3
Flujo de la aplicación	5
1. Estructura del lenguaje NINO.....	6
2. Instrucciones del lenguaje NINO	7
3. Instrucciones del lenguaje BERTA	8
5. Ejemplo de Archivos de Tipos de Datos para usar con plantillas Berta	12
6. Conclusión	13

Introducción

Nino Seisei es una herramienta libre bajo licencia GNU GPL v3 que permite generar código de aplicaciones de acceso a datos, donde, a partir de lenguaje NINO el cual está basado en SQL, puede generar código en cualquier lenguaje de programación en el formato deseado a través de BERTA, un lenguaje que permite definir plantillas para generar el código. Berta a su vez puede usar archivos de Tipos de Datos para sustituir el tipo de dato NINO por el del lenguaje cuyo código se generará.

Guía de Inicio Rápido

Partes de la aplicación

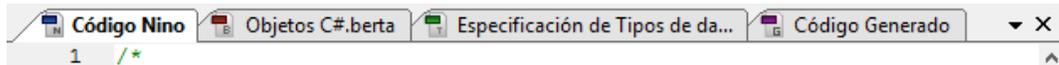


Barra de Herramientas: Dispone de los principales comandos de la aplicación:



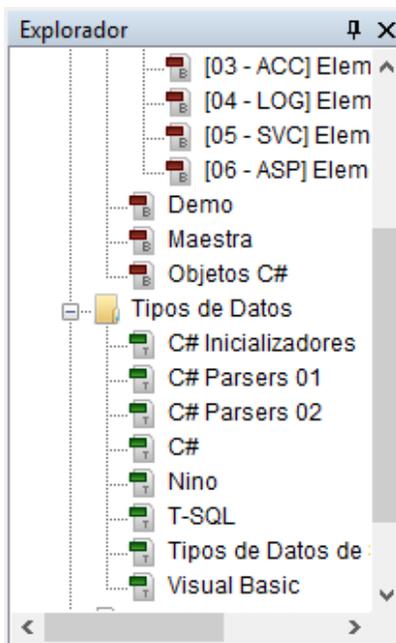
- Limpia la pestaña de Código Nino para crear uno nuevo.
- Limpia la pestaña de Código Berta para crear uno nuevo.
- Limpia la pestaña de Tipos de Datos para crear uno nuevo.
- Abre un archivo de código Nino, Berta, Tipos de Datos u otro.
- Permite guardar el código que esté en la pestaña activa.
- Abre la ventana para generar código Nino conectándose a una base de datos.
- Subir el código Nino a memoria.
- Generar código con la plantilla Berta base al código Nino subido en memoria.
- Ver la ventana de acerca de.

Pestañas de Código



-  Pestaña de Código Nino.
-  Pestaña de Código Berta.
-  Pestaña de Tipos de Datos.
-  Código generado, otro archivo que se abra y no sea reconocido como los anteriores tres se mostrará acá.

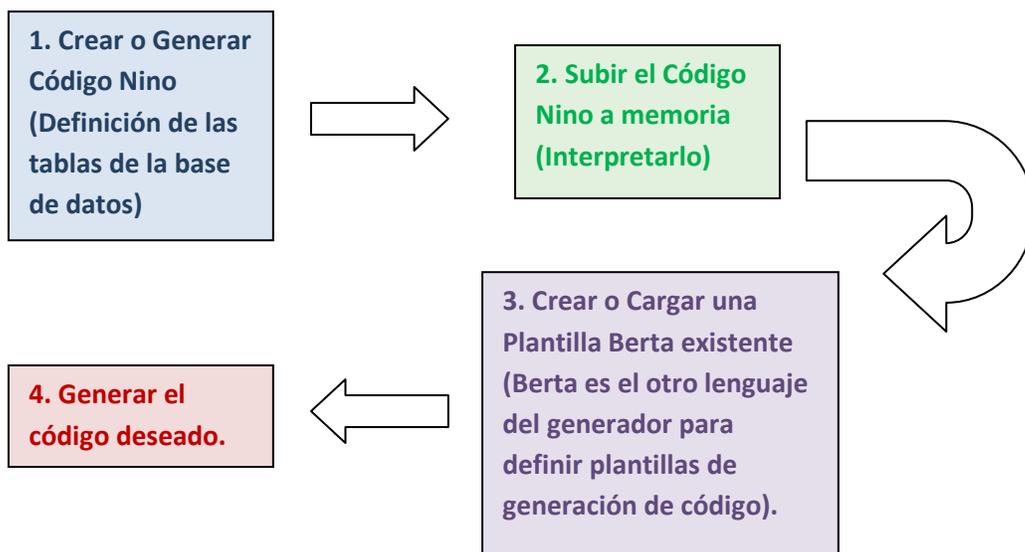
Explorador: Permite navegar en el espacio de trabajo para abrir rápidamente archivos Nino, Berta, Tipos de Datos, etc.



Salida: Muestra mensajes de la aplicación y sus distintos intérpretes. Si hay errores al subir código Nino a memoria (interpretación) o al generar código con una plantilla Berta, aparecerán acá.



Flujo de la aplicación



Crear o Generar Código Nino

Hay dos formas de hacerlo, escribiéndolo manualmente en la pestaña de código Nino o generándolo con el generador de código Nino. Para usar el generador se debe oprimir el botón  de la barra de herramientas, lo que abrirá esta ventana:

Asistente Generación de Código Nino

SGBD Entrada: Microsoft SQL Server

Usuario: sa Contraseña: *

Servidor: VALKYRIE  Listar Bases de Datos

Base de Datos: AdventureWorksLT2008

Todas las Tablas Una Tabla SalesLT.Address

 Generar

Una vez ingrese el usuario, la contraseña y servidor oprimir el botón de Listar Base de Datos listará las bases disponibles. Ahí se puede optar por una o todas las tablas. El botón Generar procederá a generar el código Nino.

Subir el código Nino a memoria (interpretarlo) para su uso

Se hace con el botón  en la barra de herramientas. Si hay errores en el código Nino se mostrarán en la salida.

Escribir una plantilla Berta o abrir una existente

Esto se hace en la pestaña de código Berta.

Generar el código deseado.

Con el botón  la aplicación procederá a interpretar la plantilla Berta para generar el código en base al código Nino que se haya subido a memoria. El código generado se verá en la pestaña de código generado y podrá ser copiado para su uso inmediato en sus proyectos de software.

1. Estructura del lenguaje NINO

NINO es un lenguaje sencillo, basado en SQL el cual es una forma resumida de definir las tablas que conforman una base de datos, con sus respectivas llaves primarias, foráneas y campos de identidad lo que permite indicar relaciones entre las tablas las cuales le permiten al intérprete de BERTA tomar decisiones usando instrucciones especializadas al generar código.

A continuación puede observarse un ejemplo de una tabla de usuarios del sistema, donde puede verse el lenguaje NINO en su totalidad:

```
SET DB_SERVER AS 127.0.0.1;
SET DB_USER AS user;
SET DB_USER_PASS AS user_pass;
SET DATABASE AS HelloWorld;

TABLE [HWUsuarios](
  IdUsuario INT NOT NULL;
  NombreUsuario VARCHAR(50) NOT NULL;
  Nombre VARCHAR(50) NOT NULL;
  Apellido1 VARCHAR(50) NOT NULL;
  Apellido2 VARCHAR(50) NULL;
  Pass VARCHAR(255) NULL;
  IdTitulo VARCHAR(10) NULL;
  Admin INT NOT NULL;
  IDENTITY ON IdUsuario;
  FOREIGN KEY IdTitulo FROM HWTitulos;
  PRIMARY KEY IdUsuario;
)
```

2. Instrucciones del lenguaje NINO

NINO está compuesto por las siguientes instrucciones que permiten modelar la base de datos:

SET DB_SERVER AS NOMBRE; - Establece la dirección o nombre del servidor de base de datos. Sirve en casos donde se ocupa escribir el nombre del servidor, como en cadenas de conexión.

SET DB_USER AS USUARIO; - Establece el nombre del usuario de la base de datos. Sirve en casos donde se ocupa escribir el nombre del usuario, como en cadenas de conexión.

SET DB_USER_PASS AS CONTRASEÑA; - Establece la contraseña del usuario de la base de datos. Sirve en casos donde se ocupa escribir la contraseña del usuario, como en cadenas de conexión.

SET DATABASE AS BASE_DE_DATOS; - Establece el nombre de la base de datos. Sirve en casos donde se ocupa escribir el nombre de la base de datos, como en cadenas de conexión.

TABLE [TABLA] (INSTRUCCIONES_DEFINICIÓN) - Define una tabla de la base de datos . Dentro de los paréntesis irán las instrucciones que definen sus campos, llaves primarias, llaves foráneas y campos de identidad. Ésta es la única instrucción que no lleva punto y coma al final.

NOMBRE_CAMPO TIPO_DATO NULL | NOT NULL; - Define un campo de la tabla , primero se indica su nombre, luego el tipo de dato el cual puede ser cualquiera, no obstante NINO solo reconoce los tipos de Microsoft SQL Server por ahora. Finalmente se indica si el campo es NULL o no lo es (NOT NULL).

IDENTITY ON CAMPO_IDENTIDAD; - Establece el campo de la tabla con ese nombre como un campo de identidad.

FOREIGN KEY CAMPO_FORÁNEO FROM TABLA_FORÁNEA; - Indica que el campo de la tabla con ese nombre corresponde a una llave foránea de la tabla indicada, la cual debe estar especificada en el código NINO o habrán problemas al leerlo el intérprete.

PRIMARY KEY LLAVE_PRIMARIA; - Establece el campo de la tabla con ese nombre como la llave primaria.

3. Instrucciones del lenguaje BERTA

Con BERTA es posible definir plantillas para generar código en cualquier lenguaje, sus instrucciones y su sintaxis puede ser algo compleja, pero tal diseño permite neutralidad con cualquier lenguaje de programación.

A continuación se detallan las instrucciones disponibles:

NIVEL 1

Son instrucciones que se interpretan sin ningún ciclo.

@B_DEFINE_DATA_TYPES[@B]Tipos de Datos\Tipos.dt[@EB] - Indica que cargue el archivo 'Tipos.dt' de la carpeta 'Tipos de Datos' en el nivel 0 de tipos de datos, para sustituir los tipos de datos NINO con los indicados en ese archivo usando la instrucción de tercer nivel **@B_DATATYPE**.

@B_2DEFINE_DATA_TYPES[@B]Tipos de Datos\Tipos2.dt[@EB] - Indica que cargue el archivo 'Tipos2.dt' de la carpeta 'Tipos de Datos' en el nivel 2 de tipos de datos, para sustituir los tipos de datos NINO con los indicados en ese archivo usando la instrucción de tercer nivel **@B_2DATATYPE**.

@B_3DEFINE_DATA_TYPES[@B]Tipos de Datos\Tipos3.dt[@EB] - Indica que cargue el archivo 'Tipos3.dt' de la carpeta 'Tipos de Datos' en el nivel 3 de tipos de datos, para sustituir los tipos de datos NINO con los indicados en ese archivo usando la instrucción de tercer nivel **@B_3DATATYPE**.

@B_4DEFINE_DATA_TYPES[@B]Tipos de Datos\Tipos4.dt[@EB] - Indica que cargue el archivo 'Tipos4.dt' de la carpeta 'Tipos de Datos' en el nivel 4 de tipos de datos, para sustituir los tipos de datos NINO con los indicados en ese archivo usando la instrucción de tercer nivel **@B_4DATATYPE**.

@B_5DEFINE_DATA_TYPES[@B]Tipos de Datos\Tipos5.dt[@EB] - Indica que cargue el archivo 'Tipos5.dt' de la carpeta 'Tipos de Datos' en el nivel 5 de tipos de datos, para sustituir los tipos de datos NINO con los indicados en ese archivo usando la instrucción de tercer nivel **@B_5DATATYPE**.

@B_DATABASE - Imprime el nombre de la base de datos.

@B_DB_SERVER - Imprime el nombre o dirección del servidor gestor de bases de datos.

@B_DB_USER - Imprime el nombre del usuario de la base de datos.

@B_DB_USER_PASS - Imprime la contraseña del usuario de la base de datos.

@B_LOOP_DB Código para generar @EB_LOOP_DB - Establece que el código contenido en su interior se imprimirá por cada tabla presente en la base de datos, dentro se ejecutarán las instrucciones de **NIVEL 2**

NIVEL 2

Son instrucciones que se interpretan dentro del código de un ciclo

@B_LOOP_DB Código para generar @EB_LOOP_DB que recorre las tablas de la base de datos.

@B_TABLENAME - Imprime el nombre de la tabla.

@B_LOOP_TABLE[@B] Código a concatenar[@EB] Código para generar @EB_LOOP_TABLE - Establece que el código contenido en su interior se imprimirá por cada columna presente en la tabla, dentro se ejecutarán las instrucciones de **NIVEL 3**. Al final de cada iteración concatenará el código para concatenar indicado.

@B_LOOP_TABLE_NON_PRIMARY[@B] Código a concatenar[@EB] Código para generar @EB_LOOP_TABLE_NON_PRIMARY - Establece que el código contenido en su interior se imprimirá por cada columna presente en la tabla pero no se imprimirá cuando la columna es la llave primaria de la tabla, dentro se ejecutarán las instrucciones de **NIVEL 3**. Al final de cada iteración concatenará el código para concatenar indicado.

@B_LOOP_TABLE_PRIMARY_OR_ALL[@B] Código a concatenar[@EB] Código para generar @EB_LOOP_TABLE_PRIMARY_OR_ALL - Establece que el código contenido en su interior se imprimirá solo por cada llave primaria de la tabla pero si la tabla no tiene llaves primarias imprimirá por cada columna de la tabla, dentro se ejecutarán las instrucciones de **NIVEL 3**. Al final de cada iteración concatenará el código para concatenar indicado.

@B_LOOP_TABLE_PRIMARY_OR_FIRST[@B] Código a concatenar[@EB] Código para generar @EB_LOOP_TABLE_PRIMARY_OR_FIRST - Establece que el código contenido en su interior se imprimirá por cada llave primaria de la tabla si no hay llaves primarias se imprime solo una vez por la primer columna de la tabla, dentro se ejecutarán las instrucciones de **NIVEL 3**. Al final de cada iteración concatenará el código para concatenar indicado.

@B_HAVE_PRIMARY[@B] Primer código | Segundo código [@EB] - Condición que imprime el primer código si tiene llave primaria o imprime el segundo código en caso contrario.

@B_HAVE_IDENTITY[@B] Primer código | Segundo código [@EB] - Condición que imprime el primer código si tiene campo de identidad o imprime el segundo código en caso contrario.

@B_HAVE_FOREIGN[@B] *Primer código | Segundo código* [EB] - Condición que imprime el primer código si tiene llave foránea o imprime el segundo código en caso contrario.

@B_HAVE_FOREIGN_OUT[@B] *Primer código | Segundo código* [EB] - Condición que imprime el primer código si tiene llave foránea en otra tabla o imprime el segundo código en caso contrario.

@B_LOOP_TABLES_RELATIONED[@B] *Código* [EB] - Ciclo que imprime su código interno recorriendo sólo tablas que tengan relación con otras (su nombre aparece en la llave foránea de otra tabla), dentro se ejecutarán las instrucciones de **NIVEL 3** y **NIVEL 4**

NIVEL 3

Son instrucciones que se interpretan dentro del código de un ciclo

@B_LOOP_TABLE *Código para generar* EB_LOOP_TABLE o similares que recorren las columnas de una tabla.

@B_PRIMARY_CODE[@B] *Código para generar* [EB] - Imprime el código de adentro cuando la iteración corresponde a la llave primaria de la tabla.

@B_IDENTITY_CODE[@B] *Código para generar* [EB] - Imprime el código de adentro cuando la iteración corresponde al campo identidad de la tabla.

@B_FOREIGN_CODE[@B] *Código para generar* [EB] - Imprime el código de adentro cuando la iteración corresponde a una llave foránea de la tabla.

@B_ITERATION - Imprime el número de iteración del ciclo de columnas en la tabla.

@B_NULL - Representa un valor null, se remueve dejando su espacio en blanco.

@B_COLNAME - Imprime el nombre de la columna.

@B_DATALENGTH - Imprime la longitud del tipo de dato de la columna.

@B_NINO_DATATYPE - Imprime el tipo de dato NINO de la columna.

@B_DATATYPE - Imprime el tipo de dato correspondiente en el nivel 0 de memoria de tipos de datos definidos previamente.

@B_2DATATYPE - Imprime el tipo de dato correspondiente en el nivel 2 de memoria de tipos de datos definidos previamente.

@B_3DATATYPE - Imprime el tipo de dato correspondiente en el nivel 3 de memoria de tipos de datos definidos previamente.

@B_4DATATYPE - Imprime el tipo de dato correspondiente en el nivel 4 de memoria de tipos de datos definidos previamente.

@B_5DATATYPE - Imprime el tipo de dato correspondiente en el nivel 5 de memoria de tipos de datos definidos previamente.

@B_HAVE_DATALENGTH[@B] *Primer código* | *Segundo código* [@EB] - Condición que imprime el primer código si la columna tiene un valor de longitud de tipo de dato o imprime el segundo código en caso contrario.

@B_IS_NULL[@B] *Primer código* | *Segundo código* [@EB] - Condición que imprime el primer código si la columna admite valores NULL o imprime el segundo código en caso contrario.

@B_IS_PRIMARY[@B] *Primer código* | *Segundo código* [@EB] - Condición que imprime el primer código si la columna es llave primaria o imprime el segundo código en caso contrario.

@B_IS_IDENTITY[@B] *Primer código* | *Segundo código* [@EB] - Condición que imprime el primer código si la columna es identidad o imprime el segundo código en caso contrario.

@B_IS_FOREIGN[@B] *Primer código* | *Segundo código* [@EB] - Condición que imprime el primer código si la columna es llave foránea o imprime el segundo código en caso contrario.

NIVEL 4

Son instrucciones que se interpretan dentro del código de un ciclo

@B_LOOP_TABLES_RELATIONED[@B] *Código* [@EB]

@B_FOREIGN_TABLE - Imprime el nombre de la tabla de la llave foránea.

@B_FOREIGN_TABLE_COLNAME - Imprime el nombre de la columna de llave foránea.

NIVEL 5

Instrucciones que se interpretan dentro de otras instrucciones específicas

@B_PRIMARY - Imprime el nombre de la llave primaria de la tabla. Solo dentro de la instrucción **@B_PRIMARY_CODE**[@B] *Código para generar* [@EB]

@B_IDENTITY - Imprime el nombre del campo identidad de la tabla. Solo dentro de

la instrucción `@B_IDENTITY_CODE[@B]` Código para generar `[@EB]`

`@B_FOREIGN` - Imprime el nombre de la llave foránea de la tabla. Solo dentro de la instrucción `@B_FOREIGN_CODE[@B]` Código para generar `[@EB]`

5. Ejemplo de Archivos de Tipos de Datos para usar con plantillas Berta

En resumen son líneas del tipo `TIPO_DATOS_NINO = TIPO_DATOS_DESEADO;`

Este ejemplo convierte los tipos de datos Nino en Tipos de datos del lenguaje C#.

```
SCINTILLA = cs;
BIGINT = long;
BINARY = byte[];
BIT = bool;
CHAR = char;
DATE = DateTime;
DATETIME = DateTime;
DATETIME2 = DateTime;
DATETIMEOFFSET = DateTimeOffset;
DECIMAL = decimal;
FLOAT = float;
IMAGE = byte[];
INT = int;
MONEY = double;
NCHAR = char;
NTEXT = string;
NVARCHAR = string;
REAL = decimal;
SMALLDATETIME = DateTime;
SMALLINT = short;
SMALLMONEY = float;
TEXT = string;
TIME = DateTime;
TIMESTAMP = string;
TINYINT = short;
UNIQUEIDENTIFIER = string;
VARBINARY = byte[];
VARCHAR = string;
XML = string;
```

6. Conclusión

Con Nino Seisei el desarrollador puede generar código que normalmente se tardaría bastante en escribir y que le robaría tiempo que pudo haber invertido en desarrollo de aspectos ingeniosos y creativos, no en tareas repetitivas y agotadoras.

El desarrollador podrá generar el código que desea en la forma que lo quiere, lo cual le permite tener un amplio control en su proyecto de software, ya que esta herramienta no genera un proyecto entero difícil de mantener sino, lo que el desarrollador desea obtener rápido, sin molestarse en tareas monótonas.

Aplicación dedicada en memoria de mi abuela Dagoberta Segura Quesada, conocida como Doña Berta y, mi difunto gatito, Nino.