Fábrica - Modelos Logs - Log de Cargas de Procesos (Archivos)

Este es el log utilizado para registrar las notificaciones que puedan presentar en los procesos de cargas de información.

Tipos de Notificaciones

A continuación se listan los tipos de notificaciones que deben ser considerados en el log de Cargas

- Inicio de Carga
- Fin de Carga
- Errores en las lineas procesadas
- Errores en columnas de las lineas procesadas

¿Donde usar el Servicio?

Este servicio debe consumirse en los procesos de cargas de:

- Aplicaciones Java
- Aplicaciones .Net
- Web Service
- Aplicaciones Móbiles
- Aplicaciones Powerbuilder (Ver Nota siguiente)
- Soluciones que afecten los productos SICOF

Nota: Aplicaciones Powerbuilder

Powerbuilder tiene restricciones para el consumo de servicios Rest por lo tanto en las aplicaciones de esta tecnología se implementará un API para realizar las llamadas.

Diccionario de Datos

OWNER	SICOF	**TABLE**	FILE_UPLOAD_LOG	**COMMENTS**	Contiene el log de cargas de archivos o procesos de las aplicaciones de la compañia
#	**NAME**	**NULL**	**TYPE**	**COMMENTS**	**WS**[1)]

1	ID	N	NUMBER	Código interno del registro (Se controla por secuencia)	Interno, Autoincremental
2	FECHA	Y	DATE	Fecha en la cuál se genera la carga	No utilizado
3	NOMBRE_ARCHIVO	Y	VARCHAR2(256)	Mensaje simple del error (Resumen)	Externo, Requerido
4	NOMBRE_PROCESO	Y	VARCHAR2(256)	Nombre del proceso que realiza la carga	Externo, Requerido
5	NUMERO_LINEA	Y	NUMBER	Número de linea del archivo donde se genera la notificación	Externo, Requerido
6	COLUMNA_GRUPO	Y	VARCHAR2(1024)	Columna o grupo de colunas que generan la notificación	Obsoleto, Externo
7	NOTIFICACION	Y	VARCHAR2(2048)	Mensaje de notificación	Obsoleto, Externo
8	LOG_LINEA	N	CLOB	Array Json que contiene las notificaciones generadas en la linea	Externo, Requerido
8	HOST_CLIENTE	Y	VARCHAR2(50)	Host del cliente (Dirección IP)	Externo, Requerido
9	FECHA_REGISTRO	Y	DATE	Fecha del sistema DB	Interno, Formato dd/mm/yyyy hh:mm:ss, Requerido
10	CODIGO_USUARIO	Y	NUMBER	Código del usuario de la sesión en la cuál se genera el error	Externo, Requerido
11	CODIGO_MEMPRESA	Y	VARCHAR2(64)	Código de la empresa de la sesión en la cuál se genera el error	Externo, Requerido
12	CODIGO_APLICACION	Y	NUMBER	Código de la aplicación (Identificador interno numérico)	Externo, Requerido

13	INFO_APP	Y	VARCHAR2(256)	Información de la aplicación (En las situaciónes donde no se identifique código interno se puede enviar el nombre de la aplicación o información adicional)	Externo
14	SESSION_MAC	Y	VARCHAR2(64)	MAC del equipo del usuario	Externo
15	SESSION_BROWSERVERSION	Y	VARCHAR2(64)	Versión del Navegador	Externo, Requerido
16	SESSION_OSTYPE	Y	VARCHAR2(64)	Sistema Operativo	Externo, Requerido

Columna: WS

Se adiciona esta columna para identificar reglas asociadas a la implementación de los servicios web que permiten gestionar el almacenamiento de los logs. La columna es una referencia y no hace parte del servicio sin embargo las reglas que se definen en ella si aplican para la columna relacionada:

Reglas

- **Interno**: Indica que el campo se gestiona dentro del servicio y por lo tanto no se pedira en los parametros.
- **Autoincremental**: Indica que el campo se comporta como una secuencia.
- **Externo**: Indica que el campo debe estar en los parametros del consumo.
- **Requerido**: Indica que el campo debe ser enviado en el consumo y el servicio debe validarlo para continuar.
- **Obsoleto**: Indica que el campo ya no es utilizado en la nueva implementación.
- **No utilizado**: Indica que el campo no será utilizado en ninguna implementación.

Nota

- Todas las operaciones del servicio que gestiona la persistencia de la tabla deben estar documentadas incluyendo la definición de los campos, formatos, longitudes de columnas e indicar si es requerido o no.

Columna: LOG_LINEA

Esta columna sirve para almacenar todas la notificaciones que se pueden presentar al procesar una linea del archivo de la carga. Se define la siguiente estructura base ejemplo:

```
{
  "columns":[
```

```
    {
      "column_name":"Requerido: Nombre de la columna",
      "column_value": "Requerido: Valor de la columna",
      "column_notifications":
        [
          {"msg":"Requerido: Mensaje de Notificación 1"},
          {"msg":"Requerido: Mensaje de Notificación 2"},
          {"msg":"Requerido: Mensaje de Notificación N"}
        ]
    }
  ]
}
```

Donde:

- **columns**: Arrya de Columnas del archivo de carga
- **column_name**: Propiedad contenida en cada indice del array json que representa el nombre de la columna analizada del proceso de carga.
- **column_value**: Propiedad contenida en cada indice del array json que representa el valor de la columna analizada del proceso de carga.
- **column_notifications**: Propiedad contenida en cada indice del array json que representa el array de notificaciones de la columna analizada del proceso de carga.
- **msg**: Propiedad contenida en cada indice de column_notifications de la columna analizada del proceso de carga.

Ejemplo:

```
{
  "columns":[
    {
      "column_name":"fecha_sistema",
      "column_value": "01/01/20",
      "column_notifications":
        [
          {"msg":"El año de la fecha es menor a la vigencia actual"},
          {"msg":"La fecha no tiene el formato dd/mm/yyyy"}
        ]
    }
  ]
}
```

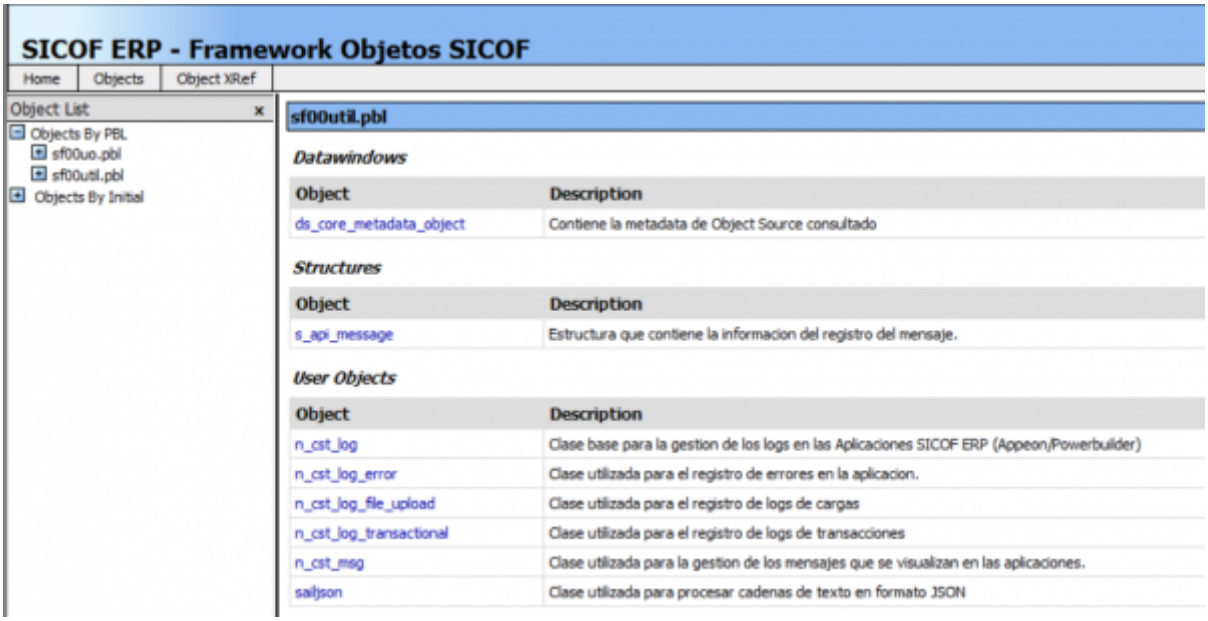
Notas

- Los valores de las columnas deben ser registrados como String

Modo de uso: Powerbuilder - Documentación

Para visualizar la documentación debe descargar el siguiente repositorio Documentación, abrir la

pagina Index.html en su navegador web la cual es similar a la siguiente imagen:



En ella encontrará la documentación de las librerias que hacen parte del framework **Objetos SICOF** el cuál se irá actualizando frecuentmente a medida que se documenten las clases.

La Libreria que contiene la funcionalidad de los logs es la librería **sf00util.pbl**

Los Objetos relacionados en el API son:

- **n_cst_app**: Clase contenedora de objetos logs
- **n_cst_log_file_upload**: Clase para la gestión de log de cargas

Ejemplos de Uso

Para facilitar la implementación y uso del API de gestión de log de cargas se crea un objeto interno privado en la clase global **guo_app** el cual puede ser accedido por el método **of_log_file_upload()** que devuelve la instancia del objeto. Sin embargo para implementaciones específicas se puede optar por crear y administrar la clase **n_cst_log_file_upload** según considere el desarrollador.

A continuación se listan ejemplos de uso el cuál presenta las forma de utilizar el API, para más información debe consultar la documentación en el repositorio.

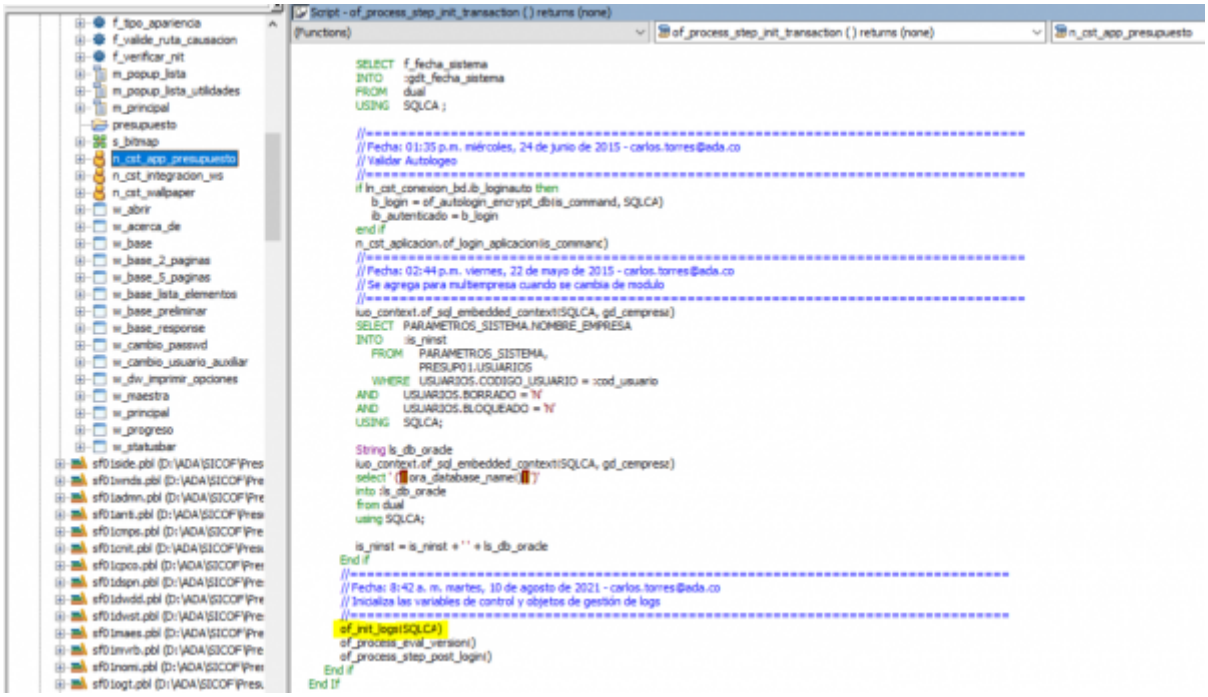
```
/*Ejemplos de uso utilizando la instancia genérica de la clase guo_app*/
guo_app.of_log_file_upload( ).of_set_process_header(f_hoy(),
"carga_prueba.txt", "Carga de Terceros")
guo_app.of_log_file_upload( ).of_add_log(10, "codigo_tercero", "columna
nula", true, SQLCA)
guo_app.of_log_file_upload( ).of_add_log(f_hoy(), "carga_prueba.txt", "Carga
de Terceros", 10, "codigo_tercero", "columna nula", true, SQLCA)
guo_app.of_log_file_upload( ).of_add_log(f_hoy(), "carga_prueba.txt", "Carga
de Terceros", 10, ljson, true, SQLCA)

/*Ejemplo de uso definiendo la clase*/
```

```
n_cst_log_file_upload luo_log_file_upload
luo_log_file_upload= Create n_cst_log_file_upload
luo_log_file_upload.of_add_log(f_hoy(), "carga_prueba.txt", "Carga de
Terceros", 10, ljson, true, SQLCA)
destroy luo_log_file_upload
```

Consideraciones

- El API puede ser activada o desactivada por medio de la constante: **LOG_UPLOAD_FILE** (Solo en aplicaciones SICOF ERP (Appeon/Powerbuilder)) siempre y cuando se utilice la implementación de la clase **guo_app**.
- El desarrollador es el encargado de gestionar la transacción que realiza la persistencia.
- Para el procesamiento de logs de bloques se implementa Clase sailjson para procesamiento de cadenas de texto en ese formato.
- Cada módulo (Contabilidad, Prespuesto, Tesorería, Compras, Talento y Nómina) debe implementar el método de inicialización **guo_app.of_init_logs(SQLCA)** en el método **of_process_step_init_transaction** de la clase **guo_app** especializada por cada módulo. A continuación se muestra una imagen de referencia de la implementación del módulo de presupuesto. Utilice esta guía para impelentaciones en otros módulos teniendo presente que la clase n_cst_app se especialzia con el nombre de la aplicación que la contiene.



API Json

Se adicionan métodos de procesamiento de bloques en formato json los cuales pueden ser utilizados para registrar trazas de error a continuación se muestran ejemplos de uso del API.

```
sailjson ljson, ljson1
String ls_content
```

```
ljson = create sailjson
ljson.setattribute( 'version', '1001')
//add json object
ljson1 = ljson.addobject( 'header')
ljson1.setattribute( 'count', 3)
ljson1.setattribute( 'comment', 'items count')

//add json object array, first item
ljson1 = ljson.addarrayitem( 'data')
ljson1.setattribute( 'colid', 1)
ljson1.setattribute( 'colname', 'aaaaaa')
ljson1.setattribute( 'coladdr', '')
//add second item of the array
ljson1 = ljson.addarrayitem( 'data')
ljson1.setattribute( 'colid', 2)
ljson1.setattribute( 'colname', 'bbbbbbbb')
setnull(ls)
ljson1.setattribute( 'coladdr', ls)
//add third item of the array
ljson1 = ljson.addarrayitem( 'data')
ljson1.setattribute( 'colid', 3)
ljson1.setattribute( 'colname', 'cccccc')

ljson.setattribute( 'creattime', string(now(), 'yyyymmdd.hhmmss'))

ls_content =  ljson.getformatjson('' )
destroy ljson
```

```
/*Ejemplo de Uso del API con el Log de Cargas*/
sailjson ljson, ljson1, ljson2
ljson = create sailjson
//add json object array, first item
ljson1 = ljson.addarrayitem( 'columns')
ljson1.setattribute( 'column_name', 'fecha_sistema')
ljson1.setattribute( 'column_value', '01/01/20')
ljson2 = ljson1.addarrayitem( 'column_notifications')
ljson2.setattribute( 'msg', 'El año de la fecha es menor a la vigencia
actual')
ljson2 = ljson1.addarrayitem( 'column_notifications')
ljson2.setattribute( 'msg', 'La fecha no tiene el formato dd/mm/yyyy')
//add second item of the array
ljson1 = ljson.addarrayitem( 'columns')
ljson1.setattribute( 'column_name', 'codigo_tercero')
ljson1.setattribute( 'column_value', 'null')
ljson2 = ljson1.addarrayitem( 'column_notifications')
ljson2.setattribute( 'msg', 'La columna es nula')
//Consumir Log de Cargas con traza en formato json
guo_app.of_log_file_upload( ).of_add_log(f_hoy(), "carga_prueba.txt", "Carga
de Terceros", 10, ljson, true, SQLCA)
```

Modo de uso: Java

Para las aplicaciones desarrolladas en las tecnologías (Web):

- Java
- .Net
- PHP

el log de sesión será implementado por medio de un Servicio Web el cual deberá considerar las reglas de Columna: WS

←Volver atras

1)
Define las reglas que debe aplicar el Web Service