Fábrica - Modelos Logs - API de Mensajes

Esta sección describe el API de mensajes que debe ser utilizada en las aplicaciones de la compañia.

Tipos de Mensajes

A continuación se listan los tipos de mensajes que deben ser considerados en el log.

- Mensajes Informativos: Utilizados para informar al cliente sobre procesos, estados, opciones, confirmaciones etc.
- Mensajes de Advertencias: Utilizados para alertar sobre situaciones en las aplicaciones.
- Mensajes de Errores: Utilizados para notificar errores generados en los procesos de la aplicaciones.

Que no se considera en el API

- Textos descriptivos de procesos como (guías, tutoriales).
- Mensajes asociados a sistemas operativos, maquinas virtuales, bases de datos o errores desconocidos.

¿Donde usar el Servicio?

Este servicio debe consumirse en las generaciones de errores de:

- Aplicaciones Java
- Aplicaciones .Net
- Web Service
- Integración Supertabla
- Soluciones que afecten los productos SICOF

Operaciones del API

El API soporta las siguientes operaciones:

- Consultar mensaje por código interno numérico. Columna: **CODIGO_MENSAJE**
- Consultar mensaje por código externo texto. Columna **CODIGO**
- Consultar mensaje formateado por código interno numérico y array de (String) literales # (Ejemplo: Mensaje = Hola #1 | Literal #1 = Mundo | Mensaje Formateado = Hola Mundo)
- Consultar mensaje formateado por código externo texto y array de (String) literales # (Ejemplo: Mensaje = Hola #1 | Literal #1 = Mundo | Mensaje Formateado = Hola Mundo)

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	PRESUP01	**TABLE**	MENSAJES_SISTEMA	**COMMENTS**	Tabla que contiene los mensajes de los procesos de SICOF
#	**NAME**	**NULLEABLE**	**TYPE**	**COMMENTS**	
1	CODIGO_MENSAJE	N	NUMBER	Código Interno del Mensaje (Es unico en la tabla y se asigna de forma manual)	
2	MENSAJE	Y	VARCHAR2(1024)	Descripción del mensaje que se representa puede registrar expresiones de sustitución con la regla #identificador númerico Ej Hola #1	
3	INFORMATION	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje es de tipo: Información	
4	STOP	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje es de tipo: Error	
5	EXCLAMATION	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje es de tipo: Advertencia	
6	QUESTION	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje es de tipo: Pregunta o Interrogación	
7	NONE	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje no especifica su tipo: Por lo general se usa Información	
8	OK	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje utiliza boton: Aceptar	
9	CANCEL	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje utiliza boton: Cancelar	
10	YES	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje utiliza boton: Si	

11	NO	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje utiliza boton: No
12	RETRY	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje utiliza boton: Reintentar
13	ABORT	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje utiliza boton: Abortar
14	IGNORE	Y	CHAR(1)	Bandera que puede ser utilizada por el frontend: indica que el mensaje utiliza boton: Ignorar
15	DEFAULTBUTTON	Y	NUMBER(1,0)	Bandera que puede ser utilizada por el frontend: indica que el mensaje utiliza un boton por defecto (con foco) es un identificador numérico que se representa de izuiqerda a derecha
16	CODIGO_BASE_DATOS	Y	NUMBER(10,0)	Código interno de base de datos que está asociado al mensaje
17	USUARIO_EMPRESA	Y	VARCHAR2(30)	Usuario Empresa
18	CODIGO_MEMPRESA	Y	VARCHAR2(50)	Código de la emprsa por defecto '9999999999' para entorno uniempresa.
19	FECHA_REGISTRO	Y	DATE	Fecha del sistema
20	TIPO_PROCESO	Y	VARCHAR2(256)	Clasificación del mensaje Ej: Disponiiblidad, Compromiso etc.
21	CODIGO	Y	VARCHAR2(128)	Es un identificador string unico que puede ser utilizado para identificar el mensaje por lo general inicial con una sigla nemotecnica asociada a la clasificación (TIPO_PROCESO) EJ: Para Disponibilidad seria CDP00001
22	TITULO	Y	VARCHAR2(256)	Campo que puede ser utilizada por el frontend para visualizar un titulo en el mensaje.
23	OBSERVACIONES	Y	VARCHAR2(4000)	Información complementaria que orienta al usuario sobre la situación notificada en el mensaje. Puede llevar pasos, consejos o indicaciones.

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_errores**: Clase para la gestión de errores

Ejemplos de Uso

Para facilitar la implementación y uso del API de gestión de errores se crea un objeto interno privado en la clase global **guo_app** el cual puede ser accedido por el método **of_log_error()** que devuelve la instancia del objeto. Sin embargo para implementaciones específicas se puede optar por crear y administrar la clase de error **n_cst_log_error** 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_error( ).of_add_log("EJEMPLO_CODE", 10, true, SQLCA)
guo_app.of_log_error( ).of_add_log("EJEMPLO_CODE", "ERROR_DB", true, SQLCA)
guo_app.of_log_error( ).of_add_log("EJEMPLO_CODE", 10, ls_args, true, SQLCA)
guo_app.of_log_error( ).of_add_log_text("Ejemplo", "ERROR_DB",
"Presupuesto", true, SQLCA)
guo_app.of_log_error( ).of_add_log_text( sqldbcode, sqlerrtext, dataobject,
sqlsyntax, ls_ventana, true, lts_db)

/*Ejemplo de uso definiendo la clase de error,
instancia el array de formateo de mensaje y
registra en el log el mensaje formateado.
Posteriormente elimina la instancia de la clase de gestión de error.
"El array de formateo se utiliza cuando el mensaje se genera por
expresiones:
Ejemplo:
Mensaje Base = Hola #1 (identificado con el código de mensaje SALUDO_CODE),
se crea un array con un expresión de la siguiente forma
array[1] = "Mundo",
de esta forma al usar el método of_add_log se genera el mensaje: Hola
Mundo".
*/
n_cst_log_error luo_log_error
String ls_args[]
ls_args[1] = "Mundo"
luo_log_error = Create n_cst_log_error
luo_log_error.of_add_log("SALUDO_CODE", 10, ls_args, true, SQLCA)
destroy luo_log_error

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

```
registra en el log el mensaje por medio de un código de mensaje.
Posteriormente elimina la instancia de la clase de gestión de error.*/
n_cst_log_error luo_log_error
luo_log_error = Create n_cst_log_error
luo_log_error.of_add_log("EJEMPLO_CODE", 10, true, SQLCA)
destroy luo_log_error

/*Ejemplo de uso definiendo la clase de error y
registra en el log el mensaje por medio de un código de mensaje.
Posteriormente elimina la instancia de la clase de gestión de error.*/
n_cst_log_error luo_log_error
luo_log_error = Create n_cst_log_error
luo_log_error.of_add_log("EJEMPLO_CODE", "ERROR_DB", true, SQLCA)
destroy luo_log_error
```

Consideraciones

- El API puede ser activada o desactivada por medio de la constante: **LOG_ERRORS** (Solo en aplicaciones SICOF ERP (Appeon/Powerbuilder)).
- El desarrollador es el encargado de gestionar la transacción que realiza la persistencia.
- Se automatiza el registro de errores en los objetos **uo_datawindow** y **uo_datastore** en toda su herencia.
- Se automatiza el registro de errores genéricos no controlados lanzados por el evento **systemerror** de Powerbuilder.

Modo de uso: Java (Próximamente)

Para las aplicaciones desarrolladas en la tecnología Java el log de sesión será implementado por medio de un servicio web.

←Volver atras

From:
https://wiki.adacsc.co/ - **Wiki**

Permanent link:
https://wiki.adacsc.co/doku.php?id=ada:howto:sicoferp:factory:logmodels:message&rev=1629462108

Last update: **2021/08/20 12:21**