

GiD plug-in Post import DLL's:

A definir por el **usuario**:

- único include 'gid_plugin_import.h' con un único define BUILD_GID_PLUGIN antes del include:
 - define y declara los prototipos de las funciones de GiD que se pueden llamar desde el plug-in
 - define tipos de datos de algunos parámetros
 - declara el tipo de las funciones de la librería que el usuario ha de implementar y que GiD va a llamar.
- definición de las siguientes rutinas (con el nombre exacto):

```
extern "C" GID_DLL_EXPORT int GiD_PostImportFile( const char *filename) {  
    .  
    .  
    return 0; // 1 - on error  
}  
extern "C" GID_DLL_EXPORT const char *GiD_PostImportGetLibName( void) {  
    return "Wavefront Objects import";  
}  
extern "C" GID_DLL_EXPORT const char *GiD_PostImportGetFileExtensions( void) {  
    return "{{Wavefront Objects} {.obj}} {{All files} {*.}}";  
}  
extern "C" GID_DLL_EXPORT const char *GiD_PostImportGetDescription( void) {  
    return "Wavefront OBJ import plugin for GiD";  
}  
extern "C" GID_DLL_EXPORT const char *GiD_PostImportGetErrorStr( void) {  
    return _G_err_str; // if error, returns the error string  
}
```

- Puede usar las siguientes rutinas:

```
extern "C" int GiD_NewPostProcess( void);  
extern "C" int GiD_NewMesh( _t_gidMode gid_mode, _t_gidMeshType mesh_type, const char *name);  
extern "C" int GiD_SetColor( int id, float red, float green, float blue, float alpha);  
extern "C" int GiD_SetVertexPointer( int id,  
    _t_gidBasicType basic_type,  
    _t_gidVertexListType list_type,  
    int num_components,  
    int num_vertices,  
    unsigned int offset_next_element,  
    const void *pointer);  
extern "C" int GiD_SetElementPointer( int id,  
    _t_gidBasicType basic_type,  
    _t_gidElementListType list_type,  
    _t_gidElementType element_type,  
    int num_elements,  
    unsigned int offset_next_element, const void *pointer,  
    unsigned int offset_float_data, const void *float_ptr);  
extern "C" int GiD_NewResult( const char *analysis_name, double step_value,  
    const char *result_name, int mesh_id);  
extern "C" int GiD_SetResultPointer( int id,  
    _t_gidBasicType basic_type,  
    _t_gidResultListType list_type,  
    _t_gidResultType result_type,  
    _t_gidResultLocation result_location,  
    int num_results,  
    unsigned int offset_next_element,  
    const void *pointer);  
extern "C" int GiD_EndResult( int id);  
extern "C" int GiD_EndMesh( int id);  
extern "C" Tcl_Interp *GiD_GetTclInterpreter();
```

Uso del plug-in dentro de **GiD**:

- al cargar la librería hace (GidDynLibLoad):
 - registra las funciones de GiD en la librería dinámica: GiDNewPostProcess, GiDGenMesh, ...
 - llama a las funciones del usuario: GiDImportGetLibName, GiDImportGetFileExtensions, GiDImportGetDescription → si no existen registra la librería de importación como (nombre_librería, "todas las extensiones", "todo tipo de archivos")
- al ser llamada (después de ser seleccionada por el usuario y pedir el nombre del archivo a importar):
 - llama a la función GiDImportFile de la librería con el nombre de archivo como parámetro
 - si devuelve 0, se llama a GiDImportGetErrorStr para recuperar el error y mostrárselo al usuario.