

External Functions

External functions

An external function is an object of the D2000 system, that allows extending the standard set of functions that are comprised in the [mathematical apparatus](#). The External function object represents an exported function in a dynamically linked library (file). When properly implemented, they allow to substitute (in some cases also simplify) system extensions implemented by the interfaces [D2000 ObjApi](#) or [D2000 KomAPI](#).

The creation of an External function consists of two steps:

- Implementation of a function within a dynamically linked library,
- Definition of an object of [External function](#) type.

Function Implementation

External functions for the D2000 system can be written in C++ language and the [D2ExtFunc](#) library must be used for their implementation. Also included is an example of the external function library (*utils/d2extfunc/sample* directory in the D2000 installation directory), which also provides the following example of an external function implementation.

Example of implementation of external function

```
int DemoRedim(SyncRoutedFunctionParams& params) {
    // Parameter check: the first parameter must be the structure, the second simple integer
    if (params.getCount() != 2
        || !params[0].isStructured()
        || !params[1].isSimple() || params[1].getType() != Integer)
        return CallError; // Wrong number or types of parameters - generates runtime exception in ESL

    // If the parameter specifying the new dimension of the structure is valid,
    if (params[1].isValid()) {
        // then it will change the structure dimension
        params[0].setRowCount(params[1].getValueInteger());
        for (int i = 1; i < params[0].getRowCount(); ++i) {
            // and copy the values from the first row into the other ones
            for (int j = 0; j < params[0].getColumnsCount(); ++j) {
                params[0].copyValue(params[0], 0, j, i, j);
            }
        }
    }
    // Calling of the external function was successful
    return CallSuccess;
}
```

Errors and exceptions handling

Performing external functions takes place in the same environment as running the ESL scripts that call them. If the implementation of an external function ends with a catchable C++ exception, this exception is caught and promoted as a runtime error to the ESL. However, some C++ errors (e. g., NULL dereference, division 0) do not generate catchable exceptions and end up with a crash of the library and the process using the library (Event handler), so it is important to thoroughly handle errors when implementing external functions.

Example of calling an external function

```
BEGIN
  RECORD NOALIAS (SD.ExtFunc) _rec
  CALL %DemoRedim(_rec, 10)
END
```



Related pages:

[External functions - configuration dialog box](#)

[D2ExtFunc Library](#)