Sample DLL API Documentation

Note:
To change the product logo for your own print manual or PDF, click "Tools > Manual Designer" and modify the print manual template.

Title page 1

Use this page to introduce the product

by Enter your company name

This is "Title Page 1" - you may use this page to introduce your product, show title, author, copyright, company logos, etc.

This page intentionally starts on an odd page, so that it is on the right half of an open book from the readers point of view. This is the reason why the previous page was blank (the previous page is the back side of the cover)

Sample DLL API Documentation

Printed: September 2023 in (whereever you are located)

Publisher

...enter name...

Managing Editor

...enter name...

Technical Editors

...enter name...

...enter name...

Cover Designer

...enter name...

Team Coordinator

...enter name...

Production

...enter name...

Special thanks to:

All the people who contributed to this document, to mum and dad and grandpa, to my sisters and brothers and mothers in law, to our secretary Kathrin, to the graphic artist who created this great product logo on the cover page (sorry, don't remember your name at the moment but you did a great work), to the pizza service down the street (your daily Capricciosas saved our lives), to the copy shop where this document will be duplicated, and and and...

Last not least, we want to thank EC Software who wrote this great help tool called HELP & MANUAL which printed this document.

Table of Contents

Foreword 5

Part I Introduction 6

Part II How to 7

1 Use DLL ... 7

2 Error handling ... 7

Part III Reference 8

1 Functions ... 8

GetAPI ... 8

2 Interfaces ... 8

IMalloc ... 9

INotify ... 9

Notify ... 9

ISampleDLLAPI ... 9

DoneDLL ... 10

GetAllocator ... 10

GetData ... 10

GetMemory ... 11

GetVersion ... 11

InitDLL ... 12

NotifyMe ... 12

TryAbort ... 13

TryAccessViolation ... 13

TrySoftwareException ... 13

TryWin32Exception ... 14

IStrings ... 14

GetCount ... 14

GetString ... 15

Index 16

Foreword

This is just another title page
placed between table of contents
and topics

1 Introduction

This is an example of an API for DLL.

See also

- How to
- Reference

2 How to

- How to use the sampleDLL
- Error handling

2.1 Use DLL

API for this sample DLL is implemented as ISampleDLLAPI interface, returned by the GetAPI function.

To use the sample DLL:
1. Load the DLL.
2. Call the GetAPI function to get API methods of the DLL.
3. Call the ISampleDLLAPI.InitDLL immediately after loading the DLL.
4. Work with API/DLL.
5. Call the ISampleDLLAPI.DoneDLL before unloading the DLL.
6. Unload the DLL.

You can use the following function (and only the following functions) without a prior calling the InitDLL:
- ISampleDLLAPI.GetVersion
- ISampleDLLAPI.GetAllocator

Sample

```
var
  DLL: HMODULE;
  GetAPI: TSampleDLLProc;
  API: ISampleDLLAPI;
begin
  DLL := LoadLibrary('SampleDLL.dll');
  GetAPI := GetProcAddress(DLL, SampleDLLProcName);
  GetAPI(ISampleDLLAPI, API);
  API.InitDLL(nil);

  // work with API

  API.DoneDLL;
  Finalize(API);
  FreeLibrary(DLL);
```

See also
Error handling
ISampleDLLAPI
GetAPI
IMalloc

2.2 Error handling

All API methods use `stdcall` calling convention and returns `HRESULT` value with ISupportErrorInfo / IErrorInfo support.

In addition to standard error codes (`E_FAIL`, etc.), the API defines its own error codes, indicated by the `E_C_xyz` constants. These codes are used when the `HRESULT` has the `FACILITY_ITF` type, Customer bit set, and GUID equals to `SampleDllIID`.

3 Reference

- Functions
- Interfaces

3.1 Functions

- GetAPI - returns API for the sample DLL.

3.1.1 GetAPI

Returns API for the sample DLL.

Syntax

```
procedure GetAPI(const AIID: TGUID; var AIntf); safecall;
```

Parameters

`AIID` (in)

IID (interface ID) to returns into `AIntf`. This version of the DLL supports only the ISampleDLLAPI interface.

`AIntf` (out)

interface of the type specified in the `AIID`.

Sample

```
var
  DLL: HMODULE;
  GetAPI: TSampleDLLProc;
  API: ISampleDLLAPI;
begin
  DLL := LoadLibrary('SampleDLL.dll');
  GetAPI := GetProcAddress(DLL, SampleDLLProcName);
  GetAPI(ISampleDLLAPI, API);
  API.InitDLL(nil);
  // work with the API
```

Attention: you **must** call ISampleDLLAPI.InitDLL immediately after loading the DLL. See How to use DLL.

See also

- How to use DLL
- ISampleDLLAPI
- Error handling

3.2 Interfaces

- IMalloc - allocator (memory manager) of this DLL to pass dynamic data.
- INotify - user-defined callback code for an event.
- ISampleDLLAPI - API of this DLL, returned by the GetAPI function.
- IStrings - array of strings.

3.2.1 IMalloc

Allocator (memory manager) of this DLL to pass dynamic data. This is a system interface described in MSDN. It is returned by the ISampleDLLAPI.GetAllocator method.

Any dynamic memory returned from DLL must be disposed by using this allocator. Any dynamic memory passed to DLL with ownership must be allocated by using this allocator.

See also
- ISampleDLLAPI
- ISampleDLLAPI.GetAllocator

3.2.2 INotify

User-defined callback code for an event.

Methods

Notify - fires when event happens.

Remarks

This interface is supposed to be implemented by user to react on the events.

See also
- ISampleDLLAPI.NotifyMe

3.2.2.1 Notify

Called when an event occurs.

Must be implemented by the user.

See also
INotify

3.2.3 ISampleDLLAPI

API DLL.

Methods

DoneDLL - finalizes the DLL.
GetAllocator - returns DLL's allocator.
GetData - returns string array from the DLL.
GetMemory - returns dynamic data from the DLL.
GetVersion - returns DLL version.
InitDLL - initializes the DLL.
NotifyMe - ask DLL to notify about an event.
TryAbort - throws EAbort exception.
TryAccessViolation - throws hardware exception.
TrySoftwareException - throws software exception.
TryWin32Exception - throws Win32 exception.

Remarks

Returned by the GetAPI function.

Important: you must initialize the DLL by calling the InitDLL before calling any other method (except for GetVersion and GetAllocator).

See also
- GetAPI
- How to use DLL
- Error handling

3.2.3.1 DoneDLL

Finalizes the DLL.

Syntax

```
procedure DoneDLL; safecall;
```

Remarks

Must be called before DLL unloads.

Attention: release all DLL interfaces before calling this function.

See also
- How to use DLL
- ISampleDLLAPI
- ISampleDLLAPI.InitDLL

3.2.3.2 GetAllocator

Returns DLL's allocator.

Syntax

```
function GetAllocator: IMalloc; safecall;
property Allocator: IMalloc read GetAllocator;
```

Remarks

Returns DLL's allocator. Any dynamic memory returned by the DL must be disposed by using this allocator. Any dynamic memory passed to DLL with ownership must be allocated by using this allocator.

Can be called without prior calling to the InitDLL function.

See also
- IMalloc
- ISampleDLLAPI
- How to use DLL
- ISampleDLLAPI.GetMemory

3.2.3.3 GetData

Returns string array from the DLL.

Syntax

```
function GetData: IStrings; safecall;
```

Remarks

This is a sample function. You can change it or remove it.

Returns array of string.

Attention: you must initialize the DLL by calling the InitDLL function before calling this function.

See also

- IStrings
- ISampleDLLAPI

3.2.3.4 GetMemory

Returns dynamic data from the DLL.

Syntax

```
function GetMemory(out ADataSize: DWORD): Pointer; safecall;
```

Parameters

`ADataSize` (out)
returns size of the returned memory block in bytes.

Remarks

This is a sample function. You can change it or remove it.

Returns raw memory block of arbitrary size. The returned value should be disposed by DLL's allocator after use.

Attention: you must initialize the DLL by calling the InitDLL function before calling this function.

See also

- IMalloc
- ISampleDLLAPI

3.2.3.5 GetVersion

Returns DLL version.

Syntax

```
function GetVersion: Integer; safecall;
property Version: Integer read GetVersion;
```

Remarks

Returns DLL version. Currently returns 1.

Can be called without prior calling to the InitDLL function.

See also

- ISampleDLLAPI

- How to use DLL

3.2.3.6 InitDLL

Initializes the DLL.

Syntax

```
procedure InitDLL(AOptions: IUnknown); safecall;
```

Parameters

`AOptions` (in)
currently reserved, pass nil.

Remarks

Must be called immediately after loading the DLL before calling any other methods (with the exception of GetVersion and GetAllocator).

You must call the DoneDLL function before unloading the DLL.

See also

- How to use DLL
- ISampleDLLAPI
- ISampleDLLAPI.DoneDLL

3.2.3.7 NotifyMe

Ask DLL to notify about an event.

Syntax

```
procedure NotifyMe(ANotifier: INotify); safecall;
```

Parameters

`ANotifier` (in)
user-defined event handler.

Remarks

Allows you to register your own event handler for the event called by the TryAbort, TryAccessViolation, TrySoftwareException, TryWin32Exception methods.

Attention: you must initialize the DLL by calling the InitDLL function before calling this function.

See also

- ISampleDLLAPI
- INotify
- ISampleDLLAPI.TryAbort
- ISampleDLLAPI.TryAccessViolation
- ISampleDLLAPI.TrySoftwareException
- ISampleDLLAPI.TryWin32Exception

3.2.3.8 TryAbort

Throws EAbort exception.

Syntax

```
procedure TryAbort; safecall;
```

Remarks

This is a sample function. You can change it or remove it.

Notifies about the call (see NotifyMe), then throws a silent Abort exception.

Attention: you must initialize the DLL by calling the InitDLL function before calling this function.

See also

- Error handling
- ISampleDLLAPI
- ISampleDLLAPI.NotifyMe

3.2.3.9 TryAccessViolation

Throws hardware exception.

Syntax

```
procedure TryAccessViolation; safecall;
```

Remarks

This is a sample function. You can change it or remove it.

Notifies about the call (see NotifyMe), then throws a hardware exception.

Attention: you must initialize the DLL by calling the InitDLL function before calling this function.

See also

- Error handling
- ISampleDLLAPI
- ISampleDLLAPI.NotifyMe

3.2.3.10 TrySoftwareException

Throws software exception.

Syntax

```
procedure TrySoftwareException; safecall;
```

Remarks

This is a sample function. You can change it or remove it.

Notifies about the call (see NotifyMe), then throws a software exception.

Attention: you must initialize the DLL by calling the InitDLL function before calling this function.

See also
- Error handling
- ISampleDLLAPI
- ISampleDLLAPI.NotifyMe

3.2.3.11 TryWin32Exception

Throws Win32 exception.

Syntax
```
procedure TryWin32Exception; safecall;
```

Remarks
This is a sample function. You can change it or remove it.

Notifies about the call (see NotifyMe), then throws a Win32 exception.

Attention: you must initialize the DLL by calling the InitDLL function before calling this function.

See also
- Error handling
- ISampleDLLAPI
- ISampleDLLAPI.NotifyMe

3.2.4 IStrings

Array of strings.

Methods
GetCount - returns number of strings in the array.
GetString - returns the specified string from the array.

See also
- ISampleDLLAPI.GetData

3.2.4.1 GetCount

Returns number of strings in the array.

Syntax
```
function GetCount: Integer; safecall;
property Count: Integer read GetCount;
```

Remarks
Returns number of strings in the array. Returns 0 if the array is empty.

See also
- IStrings
- IStrings.GetString

3.2.4.2 GetString

Returns the specified string from the array.

Syntax

```
function GetString(const AIndex: Integer): BSTR; safecall;
property Strings[const AIndex: Integer]: BSTR read GetString; default;
```

Parameters

`AIndex` (in)
string index from 0 to Count - 1

Remarks

Returns one string from the array with the specified in `AIndex` index. The index must be in range from 0 to Count - 1.

Raises `E_C_StringListError` in case of error (see Error handling).

See also

- Error handling
- IStrings
- IStrings.GetCount

Index

- G -

GetAPI 8

- I -

IMalloc 9
INotify 9
  Notify 9
ISampleDLLAPI 9
  Allocator 10
  DoneDLL 10
  GetAllocator 10
  GetData 10
  GetMemory 11
  GetVersion 11
  InitDLL 12
  NotifyMe 12
  TryAbort 13
  TryAccessViolation 13
  TrySoftwareException 13
  TryWin32Exception 14
  Version 11
IStrings 14
  GetCount 14
  GetString 15

- S -

SampleDLLProcName 8

- T -

TSampleDLLProc 8

Endnotes 2... (after index)

Back Cover