

NDK_Init

Last Modified on 01/18/2017 10:13 pm CST

- [C/C++](#)
- [.Net](#)

```
int __stdcall NDK_Init(LPCTSTR szAppName,  
                      LPCTSTR szKey,  
                      LPCTSTR szActCode,  
                      LPCTSTR szTmpPath  
                      )
```

Initializes the SFSDK Library.

Parameters

- [in] **szAppName** is the custom application name (user-defined), but must match the configuration base filename.
- [in] **szKey** is a valid NumXL license key.
- [in] **szActCode** is valid license activation code for current machine.
- [in] **szTmpPath** is the full path of the log and temporary files directory.

Returns

status code of the operation

Return values

- NDK_SUCCESS** Operation successful
- NDK_MISSING_APP_ARG** ERROR: invalid or Null argument (e.g. AppName for return value).
- NDK_MISSING_CONF** ERROR: The configuration file is missing.
- NDK_LOG_INIT_FAILED** ERROR: The logging system failed during initialization, check the configuration settings
- NDK_DB_INIT_FAILED** ERROR: Missing or failed to initialize date/calendar system.
- NDK_LUC_INIT_FAILED** ERROR: Failed to initialize the licensing system.
- Other** Operation unsuccessful. See [Macros](#) for full list.

Remarks

- The value of the application name argument must match the base name of the configuration file in the application's output folder.
- In the event that either the value of the license key argument or the activation code argument is missing (i.e. NULL), the function will attempt to locate the license key & activation code in the license file (i.e. NumXL.Lic) in the application's folder.
- The current user must have read/write permissions on the temporary files directory (i.e. szTmpPath).
- If the value of szTmpPath is missing (empty or NULL), the function will use the default temp

directory in the current user's profile.

- The NDK_Init function should be the first API called in the SDK
- The NDK_Init function initialize the SDK library and all its dependencies: logging, license and date/calendar system.
- The NumXL SDK system uses reference count to manage the system lifetime and support multiple clients to obtain and release access to the system without conditioning on one another in managing the system lifetime.
- Calling the NDK_Init more than once, then the library increases its internal reference counter, skip initialization and returns a warning (NDK_SKIP_INIT).

Requirements

Namespace	NumXLAPI
Class	SFSDK
Scope	Public
Lifetime	Static
Package	NumXLAPI.DLL

Examples

```
int nRet = NDK_FAILED;
char szAppName[]="MyApp";

nRet = NDK_Init(szAppName, // we have a MyApp.conf file
NULL, // use the license key in the license file (NumXL.lic)
NULL, // use the activation code in the license file (NumXL.lic)
NULL); // use the temp directory in current user's profile

// (Windows 7) (c:\users\username\AppData\Local\MyApp)
// (Windows XP) (c:\Local Settings\username\AppData\Local\MyApp)
if( nRet >= NDK_SUCCES){
...
}
```

NDK_RET_CODE Init(**stringszAppName**,
stringszKey,
stringszActCode,
stringszLogDir

Namespace: NumXLAPI
Class: SFSDK
Scope: Public
Lifetime: Static

)

Initializes the SFSDK Library.

Return Value

a value from **NDK_RETCODE** enumeration for the status of the call.

NDK_SUCCESS operation successful

Error Error Code

Parameters

[in] **szAppName** is the custom application name (user-defined), but must match the configuration base filename.

[in] **szKey** is a valid NumXL license key.

[in] **szActCode** is valid license activation code for current machine.

[in] **szLogDir** is the full path of the log and temporary files directory.

Remarks

- The value of the application name argument must match the base name of the configuration file in the application's output folder.
- In the event that either the value of the license key argument or the activation code argument is missing (i.e. NULL), the function will attempt to locate the license key & activation code in the license file (i.e. NumXL.Lic) in the application's folder.
- The current user must have read/write permissions on the temporary files directory (i.e. szTmpPath).
- If the value of szLogDir is missing (empty or NULL), the function will use the default temp directory in the current user's profile.
- The NDK_RETCODE Init function should be the first API called in the SDK
- The NDK_RETCODE Init function initialize the SDK library and all its dependencies: logging, license and date/calendar system.
- The NumXL SDK system uses reference count to manage the system lifetime and support multiple clients to obtain and release access to the system without conditioning on one another in managing the system lifetime.
- Calling the NDK_RETCODE Init more than once, then the library increases its internal reference counter, skip initialization and returns a warning (NDK_SKIP_INIT).

Exceptions

Exception Type	Condition
None	N/A

Requirements

Namespace	NumXLAPI
Class	SFSDK
Scope	Public
Lifetime	Static
Package	NumXLAPI.DLL

Examples

References

- * Hamilton, J .D.; [Time Series Analysis](#) , Princeton University Press (1994), ISBN 0-691-04289-6
- * Tsay, Ruey S.; [Analysis of Financial Time Series](#) John Wiley & SONS. (2005), ISBN 0-471-690740
- * D. S.G. Pollock; [Handbook of Time Series Analysis, Signal Processing, and Dynamics](#); Academic Press; Har/Cdr edition(Nov 17, 1999), ISBN: 125609906
- * Box, Jenkins and Reisel; [Time Series Analysis: Forecasting and Control](#); John Wiley & SONS.; 4th edition(Jun 30, 2008), ISBN: 470272848

See Also

[template("related")]