



FTSDK API 仕様書

2025.11 Ver.1.1.0





改訂履歴

バージョン	日付	内容
1.1.0.1	2025/11	初版



内容

FTSDK API 仕様書	1
1. はじめに.....	5
1.1. FTSDK について	5
1.2. 使用上のご注意・免責事項	6
1.3. 著作権及び使用許諾条項	6
1.4. 動作環境	6
2. FTSDK API.....	7
2.1. C/C++	7
2.1.1. API 仕様.....	7
2.1.2. 列挙型.....	14
2.1.3. 開発環境	16
2.1.4. FTSDK の使用方法.....	16
2.1.5. サンプルプログラム	17
2.2. C#	18
2.2.1. API 仕様	18
2.2.2. 列挙型	25
2.2.3. 開発環境.....	27
2.2.4. FTSDK の使用方法	27
2.2.5. サンプルプログラム.....	28
2.3. VB.....	29
2.3.1. API 仕様	29
2.3.2. 列挙型	36
2.3.3. 開発環境.....	38
2.3.4. FTSDK の使用方法	38
2.3.5. サンプルプログラム.....	39
2.4. JavaScript.....	40
2.4.1. API 仕様	41
2.4.2. 列挙型	49
2.4.3. 開発環境.....	51
2.4.4. 使用方法.....	54
2.4.5. サンプルプログラム.....	58
2.5. Python.....	59



2.5.1. API 仕様	59
2.5.2. 列挙型	66
2.5.3. 開発環境	68
2.5.4. FTSDK の使用方法	70
2.5.5. サンプルプログラム	71
2.6. サンプルプログラム動作説明	72
2.6.1. Sample_StartProcess	72
2.6.2. Sample_SetCustomCategory	75
2.6.3. Sample_SetWindowState	78
2.6.4. Sample_StartupServer	82
2.6.5. Sample_UnitTest	84



1. はじめに

この度は、Logging Foot（読み方：ロギングフット、以下 本ソフトウェアと記述）をダウンロードしていただき誠にありがとうございます。本ソフトウェアをご使用になられる前に本書をお読みになっていただき、ご理解を深めてからご活用いただけると幸いです。

1.1. FTSDK について

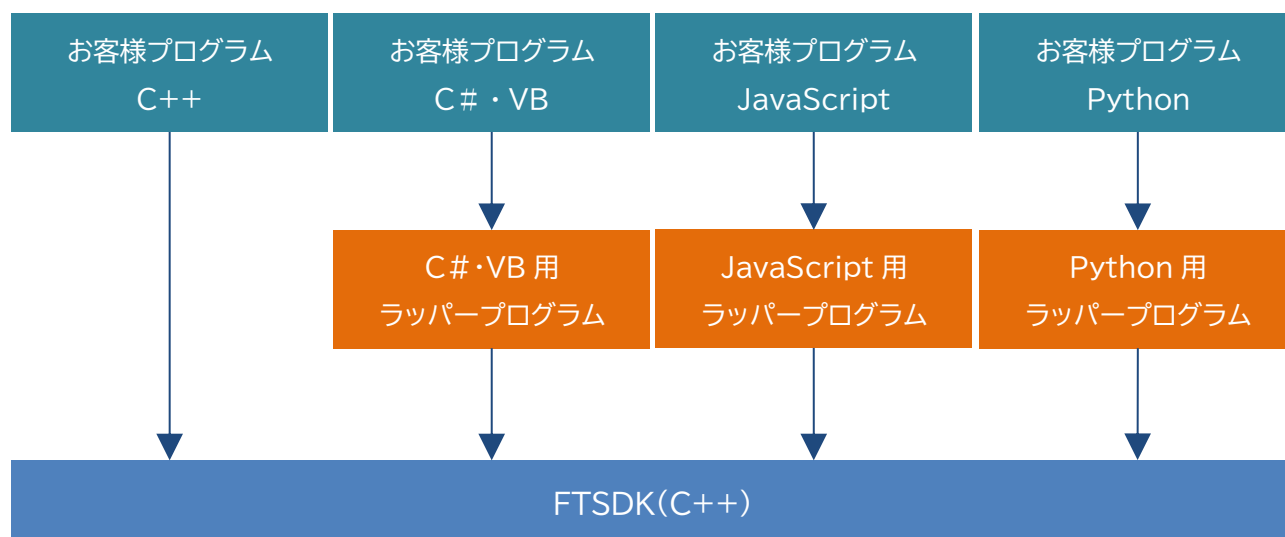
本書で述べる本ソフトウェアとは、FTSDK 及び Viewer アプリケーション等の関連する一連のソフトウェア群を示します。

Logging Foot Software Development Kit(略して FTSDK)とは、本ソフトウェアが提供するロギング機能をお客様のプログラムに組み込むための API 群を含むツールキットです。

FTSDK API は C 言語方式でエクスポートされており、C/C++言語以外のプログラミング言語から使用する場合はラッパープログラムの作成が必要になります。

現在のところ、下記プログラミング言語のサンプルプログラム及びラッパープログラムが提供されています。今後、対応言語を増やしていく予定です。

- ・ C++ (サンプルプログラムのみ)
- ・ C#
- ・ VB
- ・ JavaScript
- ・ Python





1.2. 使用上のご注意・免責事項

- ・ 個人情報や機密情報などの秘匿性の高い情報は本ソフトウェアで出力しないようにしてください。
- ・ 医療、金融系システムといった高い信頼性が求められるシステム下では使用しないでください。
- ・ 万が一、本ソフトウェアを使用したことにより生じたいかなる障害・損害につきましても、作者は一切の責任を負いかねますのでご了承ください。
- ・ 本ソフトウェアを導入するにあたり、本ソフトウェアが提供している無償版及び試用版を使用してお試しいただき、お客様の責任において導入してください。
- ・ 本ソフトウェアにバグ及び不具合が見つかった場合、作者は改善する努力はしますが義務を負わないものとします。

1.3. 著作権及び使用許諾条項

本ソフトウェアを使用するにあたり、以下の条項に同意したものとみなされます。

- ・ 本書及び本ソフトウェアの著作権は作者 Quantyworks Software に帰属します。本書及び本ソフトウェアを作者に無断で転載、複製、改変、配布、販売などを行うことを禁止します。
- ・ お客様は、本ソフトウェアが提供する機能を正しく動作させる目的においてのみ本ソフトウェアを使用することができます。
- ・ 本ソフトウェアをお客様のプログラムに組み込み、かつ本ソフトウェアの機能が正常に動作する状態において、お客様のプログラムと共に複製及び再配布をすることができます。
- ・ お客様は、組み込み先となるお客様のプログラムが商用利用・非商用利用を問わず前述の使用条件下において本ソフトウェアを使用することができます。

1.4. 動作環境

OS: Windows10(64 ビット)、Windows11(64ビット)

CPU: Intel 第4世代以降又は互換 CPU (推奨)

RAM: 4GB 以上 (推奨)

ストレージ: 32MB 以上の空き容量



2. FTSDK API

本章では、対応する各プログラミング言語についての FTSDK API 仕様を解説します。

本章以降、特に断りがない限りサーバープロセスとは Logging Foot Viewer アプリケーションのプロセスを指し、クライアントプロセスとは本 API を実装するお客様プログラムのプロセスを指します。また、本 API を実装するお客様プログラムを実行する PC をローカル PC と表現します。

API の詳細な使い方は、各サンプルプログラムをご覧ください。

2.1. C/C++

FTSDK の実装言語と同じであるため、最も効率よく動作します。

2.1.1. API 仕様

2.1.1.1. FTCORE_StartProcess(char*, unsigned short)

概要		ロギングを開始します。
FTCORE_RESULT _stdcall FTCORE_StartProcess(<div>char* servername,</div> <div>unsigned short portnumber)</div>		
servername		接続先 PC 名又は IP アドレス (xx.xx.xx.xx)
portnumber		接続先 PC のポート番号 (49152～65535)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		接続先 PC で起動しているサーバープロセスと接続を確立し、ロギングを開始します。接続先 PC 名にはローカル又はリモートの PC を指定できます。 接続先 PC のサーバープロセスはあらかじめ起動しておく必要があります。 接続先 PC がローカル PC の場合、servername には "localhost" を指定可能です。

**2.1.1.2. FTCORE_StartTriggerWithParam(char*, char*, unsigned short, int, int)**

概要		指定のパラメーターでサーバープロセスを起動し、ロギングを開始します。
		<pre>FTCORE_RESULT _stdcall FTCORE_StartTriggerWithParam(char* serverpath, char* servername, unsigned short portnumber, int viewstate, int topmost)</pre>
<i>serverpath</i>		サーバープロセスの完全パスを含む実行ファイル名
<i>servername</i>		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
<i>portnumber</i>		接続先 PC のポート番号(49152～65535)
<i>viewstate</i>		サーバープロセスアプリケーション起動時のウィンドウ表示状態 0:デフォルトのウィンドウサイズ 1:最大化ウィンドウ 2:最小化してタスクバーに表示 3:タスクトレイに隠す 4:非表示
<i>topmost</i>		最前面表示 0:無効 1:有効
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOTSERVER FTCORE_ERR_CLIENT_FAILSERVER FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		指定パスのサーバープロセスを起動して接続を確立し、ロギングを開始します。本 API は同一ローカル PC 内でのみ動作します。 <i>serverpath</i> にはローカル PC に存在する ft_viewer.exe を完全パスで指定します。 <i>servername</i> にはローカル PC の PC 名又は IP アドレスを指定します。 同一のローカル PC であるため、"localhost" を指定可能です。



2.1.1.3. FTCORE_StartTriggerWithSetup(char*, char*, unsigned short)

概要		設定済みのパラメーターでサーバープロセスを起動し、ロギングを開始します。
FTCORE_RESULT _stdcall FTCORE_StartTriggerWithSetup(char* serverpath, char* servername, unsigned short portnumber)		
serverpath		サーバープロセスの完全パスを含む実行ファイル名
servername		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
portnumber		接続先 PC のポート番号(49152～65535)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOTSERVER FTCORE_ERR_CLIENT_FAILSERVER FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		<p>セットアッププログラムにより設定済みのパラメーターで指定パスのサーバープロセスを起動して接続を確立し、ロギングを開始します。事前にセットアッププログラムを実行してパラメーターを設定しておく必要があります。(セットアッププログラムを実行していない場合は既定値のパラメーターで動作します)</p> <p>本 API は同一ローカル PC 内でのみ動作します。</p> <p>serverpath にはローカル PC に存在する ft_viewer.exe を完全パスで指定します。</p> <p>servername にはローカル PC の PC 名又は IP アドレスを指定します。</p> <p>同一のローカル PC であるため、"localhost" を指定可能です。</p>



2.1.1.4. FTCORE_ExitProcess()

概要		ロギングを終了します。
FTCORE_RESULT _stdcall FTCORE_ExitProcess()		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	-----
解説		FTCORE_StartProcess()によりロギングを開始した場合、 本 API でロギングを終了します。

2.1.1.5. FTCORE_ExitTrigger()

概要		ロギングを終了します。
FTCORE_RESULT _stdcall FTCORE_ExitTrigger()		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	-----
解説		FTCORE_StartTriggerWithParam()又は FTCORE_StartTriggerWithSetup()によりロギングを開始した場合、 本 API でロギングを終了します。



2.1.1.6. FTCCORE_SendMessage(char*, char*, char*, char*)

概要		サーバプロセスにログを出力します。
FTCORE_RESULT _stdcall FTCORE_SendMessage(char* path, char* category, char* severity, char* message)		
path		呼び出し元など任意のパス
category		ログカテゴリー
severity		ログ重要度
message		出力する任意のログ文字列
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		各引数に指定した文字列をサーバプロセスにログとして出力します。 category 及び severity については、 <u>2.1.2 列挙型</u> で示す列挙型と同じ文字列を指定することを推奨します。

2.1.1.7. FTCCORE_SetWindowState(int)

概要	サーバプロセスアプリケーションにウィンドウ表示状態の設定を指示します。	
FTCORE_RESULT _stdcall FTCORE_SetWindowState(int state)		
state	サーバプロセス アプリケーションのウィンドウ表示状態 0:デフォルトのウィンドウサイズ 1:最大化ウィンドウ 2:最小化してタスクバーに表示 3:タスクトレイに隠す 4:非表示	
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説	サーバプロセスアプリケーションのウィンドウ状態をクライアントプロセスから制御する場合に使用します。	



2.1.1.8. FTCORE_LoadDefaultCategory()

概要		サーバープロセスアプリケーションにデフォルトカテゴリーの設定を指示します。
FTCORE_RESULT _stdcall FTCORE_LoadDefaultCategory()		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		サーバープロセスアプリケーションが認識するカテゴリーを <u>2.1.2.1DefaultCategory</u> で示すデフォルトに戻すように指示します。

2.1.1.9. FTCORE_SetCustomCategory(char*)

概要		サーバープロセスアプリケーションにカスタムカテゴリーの設定を指示します。
FTCORE_RESULT _stdcall FTCORE_SetCustomCategory(char* categories)		
categories		任意のカテゴリー文字列(複数文字列の場合はカンマで区切る)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		<p>サーバープロセスアプリケーションが認識するカテゴリーに categories で指定のカテゴリーを設定するように指示します。</p> <p>本 API を使用することにより、任意に指定した文字列がカテゴリーとして認識され、サーバープロセスアプリケーションのフィルタ機能などが正しく動作するようになります。</p> <p>複数のカテゴリーを指定する場合はカンマで区切ります。</p> <p>サーバープロセスアプリケーションが認識するカテゴリーでは、文字列 NON が ”どのカテゴリーにも含まれない” 意味を示す予約語になっています。</p> <p>categories に NON を含める/含めないにかかわらず、サーバープロセスアプリケーションでは、カテゴリー列挙の先頭に NON が挿入されて動作します。</p>



2.1.1.10. FTCORE_SetCustomCategoryFromFile(char*)

概要		サーバープロセスアプリケーションにファイル読み込みでカスタムカテゴリーの設定を指示します。
FTCORE_RESULT _stdcall FTCORE_SetCustomCategoryFromFile(char* filename)		
filename		読み込みするカテゴリー設定ファイルの完全パス名
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		<p>サーバープロセスアプリケーションが認識するカテゴリーにファイルから読み込んだカテゴリーを設定するように指示します。</p> <p>本 API を使用することにより、任意に指定した文字列がカテゴリーとして認識され、サーバープロセスアプリケーションのフィルタ機能などが正しく動作するようになります。</p> <p>カテゴリーファイルは、サーバープロセスアプリケーションの [高度な設定] メニューを選択しカテゴリーソースタイプで [User definition] を選択すると作成することができます。</p> <p>詳細は Logging Foot Viewer アプリケーション操作ガイド [2.15.高度な設定を行う] の項目を参照してください。</p>



2.1.2. 列挙型

DefaultCategory 及び Severity について FTCORE_SendMessage() 関数引数には任意の文字列を渡すことができますが、サーバープロセスのアプリケーション側でフィルタ機能が動作する際、列挙型として定義された文字列と比較されます。サーバープロセスのアプリケーションでフィルタ機能を正しく動作させるには、ここで定義されている列挙型と同じ文字列を使用する必要があります。

2.1.2.1. DefaultCategory

出力するログのカテゴリを表します。

列挙型	DefaultCategory	
メンバー	値	使用例
NON	0	対象外、無効値
APP	1	アプリケーション動作
SYSTEM	2	システム関連
USER	3	ユーザー操作
UI	4	画面側発生イベント
WF	5	ビジネスロジック、ワークフロー発生イベント
DEVICE	6	デバイス発生イベント
DEBUG	7	デバッグ用
STEP	8	実行ステップ
EVENT	9	イベント
COMM	10	通信関連

2.1.2.2. Severity

出力するログの重要度を表します。

列挙型	Severity	
メンバー	値	使用例
NONE	0	対象外、無効値
INFO	1	一般的な情報
NOTICE	2	注意、注目
WARNING	3	操作ミス、警告
ERR	4	復帰可能なエラー、操作ミス
FATAL	5	復帰不可能なエラー、重大な障害



2.1.2.3. FTCORE_RESULT

API が返す戻り値の種類を表します。

列挙型	FTCORE_RESULT	
メンバー	値	説明
FTCORE_UNKNOWN_STATE	-1	対象外、無効値
FTCORE_SUCCESS	0	成功
FTCORE_EVT_CLIENT_DETECTDISCON	4	サーバープロセスとの接続切断を検知
FTCORE_EVT_CLIENT_RECEIVED	5	サーバープロセスからデータ受信
FTCORE_ERR_CLIENT_NOTSERVER	21	接続可能なサーバープロセスが存在しない
FTCORE_ERR_CLIENT_FAILSERVER	22	サーバープロセスの起動に失敗
FTCORE_ERR_CLIENT_NOEXIST	23	クライアントプロセスが生成されていない
FTCORE_ERR_CLIENT_ALREADY	24	クライアントプロセスは生成済みである
FTCORE_ERR_CLIENT_PARAMETER	25	送信パラメーターエラー発生
FTCORE_ERR_CLIENT_HOSTINFO	26	宛先情報の取得に失敗
FTCORE_ERR_CLIENT_SOCKET	27	ソケットエラー発生
FTCORE_ERR_CLIENT_REFUSED	28	サーバープロセスとの接続を拒否された
FTCORE_ERR_CLIENT_UNREACHED	29	サーバープロセスへのネットワークパスが存在しない
FTCORE_ERR_CLIENT_CONNECT	30	サーバープロセスとの接続に失敗
FTCORE_ERR_CLIENT_ESTABLISH	31	サーバープロセスとの接続確立中にエラー発生
FTCORE_ERR_CLIENT_MESSAGESEND	32	サーバープロセスへメッセージ送信に失敗
FTCORE_ERR_CLIENT_MESSAGE_RECV	33	サーバープロセスからのメッセージ受信に失敗
FTCORE_ERR_CLIENT_CONNUNKNOWN	34	不明なサーバーから接続受諾通知を受けた



2.1.3. 開発環境

本プログラムは下記の開発環境において動作確認しています。下記と異なる開発環境をご使用の場合にはビルドエラーが発生する場合がありますので適宜ご対応ください。

IDE(統合開発環境)	Visual Studio 2022 Community
.NET 環境(ターゲットフレームワーク)	-----
Windows SDK バージョン	10.0 (最新のインストールバージョン)
プラットフォームツールセット	Visual Studio 2022 (v143)

2.1.4. FTSDK の使用方法

C/C++では、ダイナミックとスタティックのリンク方法を提供しています。

開発環境は Microsoft Visual Studio を想定し、C++で外部ライブラリの使用経験がある開発者を対象としています。

以降の説明で表記するパスは、

標準のインストールフォルダ(C:\Program Files\LoggingFoot*)下のパスを示します。

2.1.4.1. ダイナミックリンク

sample\Cpp\ftsdk\h フォルダに存在する各ヘッダファイルをインクルードします。

sample\Cpp\ftsdk\lib フォルダに存在するエクスポートライブラリをリンクします。

sample\Cpp\ftsdk\bin フォルダに存在する DLL ファイルをお客様プログラムの実行フォルダに配置します。

2.1.4.2. スタティックリンク

sample\Cpp_StaticLib\ftsdk\h フォルダに存在する各ヘッダファイルをインクルードします。

sample\Cpp_StaticLib\ftsdk\lib フォルダに存在するスタティックライブラリをリンクします。



2.1.5. サンプルプログラム

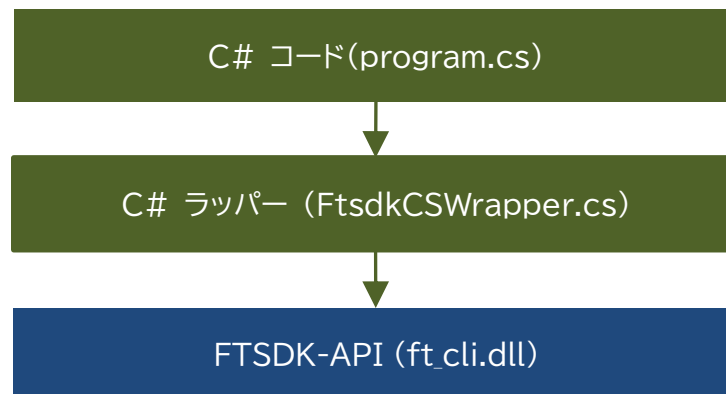
ダイナミックリンク方式は sample¥Cpp に、
スタティックリンク方式は sample¥Cpp_StaticLib にそれぞれ収録されています。
各プログラムの動作説明は [2.6 サンプルプログラム動作説明](#) を参照してください。

プログラム名	説明
Sample_StartProcess	サーバープロセスアプリケーションが起動している状態でクライアントプロセスを起動しロギングを開始するサンプルプログラムです。 使用されている主な API: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()
Sample_StartupServer	クライアントプロセスがサーバープロセスを起動してロギングを開始するサンプルプログラムです。 使用されている主な API: FTCORE_StartTriggerWithParam() FTCORE_StartTriggerWithSetup() FTCORE_SendMessage() FTCORE_ExitTrigger()
Sample_SetCustomCategory	サーバープロセスアプリケーションにカスタムカテゴリーの設定を指示するサンプルプログラムです。 使用されている主な API: FTCORE_LoadDefaultCategory() FTCORE_SetCustomCategory() FTCORE_SetCustomCategoryFromFile()
Sample_SetWindowState	サーバープロセスアプリケーションのウィンドウ状態をクライアントプロセスから制御するサンプルプログラムです。 使用されている主な API: FTCORE_SetWindowState()
Sample_UnitTest	大量のログを出力する負荷テストを行うサンプルプログラムです。 使用されている主な API: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()



2.2. C#

C#では、ラッパープログラムが提供され、下図のようなレイヤーで FTSDK-API が呼ばれます。本節で説明する API 仕様は、ラッパープログラムの内容になります。



2.2.1. API 仕様

2.2.1.1. FTCORE_StartProcess(string, int)

概要		ロギングを開始します。
FTCORE_RESULT FTCORE_StartProcess(string servername, int portnumber)		
servername		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
portnumber		接続先 PC のポート番号(49152～65535)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		接続先 PC で起動しているサーバプロセスと接続を確立し、ロギングを開始します。接続先 PC 名にはローカル又はリモートの PC を指定できます。接続先 PC のサーバプロセスはあらかじめ起動しておく必要があります。接続先 PC がローカル PC の場合、servername には "localhost" を指定可能です。

**2.2.1.2. FTCORE_StartTriggerWithParam(string, string, int, int, bool)**

概要		指定のパラメーターでサーバープロセスを起動し、ロギングを開始します。
<pre>FTCORE_RESULT FTCORE_StartTriggerWithParam(string serverpath, string servername, int portnumber, int viewstate, bool topmost)</pre>		
<i>serverpath</i>		サーバープロセスの完全パスを含む実行ファイル名
<i>servername</i>		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
<i>portnumber</i>		接続先 PC のポート番号(49152～65535)
<i>viewstate</i>		サーバープロセスアプリケーション起動時のウィンドウ表示状態 0:デフォルトのウィンドウサイズ 1:最大化ウィンドウ 2:最小化してタスクバーに表示 3:タスクトレイに隠す 4:非表示
<i>topmost</i>		最前面表示 false:無効 true :有効
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOTSERVER FTCORE_ERR_CLIENT_FAILSERVER FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		指定パスのサーバープロセスを起動して接続を確立し、ロギングを開始します。本 API は同一ローカル PC 内でのみ動作します。 <i>serverpath</i> にはローカル PC に存在する ft_viewer.exe を完全パスで指定します。 <i>servername</i> にはローカル PC の PC 名又は IP アドレスを指定します。 同一のローカル PC であるため、"localhost" を指定可能です。



2.2.1.3. FTCORE_StartTriggerWithSetup(string, string, int)

概要		設定済みのパラメーターでサーバープロセスを起動し、ロギングを開始します。
FTCORE_RESULT FTCORE_StartTriggerWithSetup(string serverpath, string servername, int portnumber)		
serverpath		サーバープロセスの完全パスを含む実行ファイル名
servername		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
portnumber		接続先 PC のポート番号(49152～65535)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOTSERVER FTCORE_ERR_CLIENT_FAILSERVER FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		<p>セットアッププログラムにより設定済みのパラメーターで指定パスのサーバープロセスを起動して接続を確立し、ロギングを開始します。事前にセットアッププログラムを実行してパラメーターを設定しておく必要があります。(セットアッププログラムを実行していない場合は既定値のパラメーターで動作します)</p> <p>本 API は同一ローカル PC 内でのみ動作します。</p> <p>serverpath にはローカル PC に存在する ft_viewer.exe を完全パスで指定します。</p> <p>servername にはローカル PC の PC 名又は IP アドレスを指定します。</p> <p>同一のローカル PC であるため、"localhost" を指定可能です。</p>



2.2.1.4. FTCORE_ExitProcess()

概要		ロギングを終了します。
FTCORE_RESULT FTCORE_ExitProcess()		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	-----
解説		FTCORE_StartProcess()によりロギングを開始した場合、 本 API でロギングを終了します。

2.2.1.5. FTCORE_ExitTrigger()

概要		ロギングを終了します。	
FTCORE_RESULT FTCORE_ExitTrigger()			
-----		引数無し	
戻り値	成功	FTCORE_SUCCESS	
	失敗	-----	
解説		FTCORE_StartTriggerWithParam()又は FTCORE_StartTriggerWithSetup()によりロギングを開始した場合、 本 API でロギングを終了します。	



2.2.1.6. FTCORE_SendMessage(string, string, string, string)

概要		サーバプロセスにログを出力します。
FTCORE_RESULT FTCORE_SendMessage(<div>string path, string category, string severity, string message)</div>		
path		呼び出し元など任意のパス
category		ログカテゴリー
severity		ログ重要度
message		出力する任意のログ文字列
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		各引数に指定した文字列をサーバプロセスにログとして出力します。 category 及び severity については、 <u>2.2.2 列挙型</u> で示す列挙型と同じ文字列を指定することを推奨します。

2.2.1.7. FTCORE_SetWindowState(int)

概要		サーバプロセスアプリケーションにウィンドウ表示状態の設定を指示します。
FTCORE_RESULT FTCORE_SetWindowState(int state)		
state	サーバプロセス アプリケーションのウィンドウ表示状態 0:デフォルトのウィンドウサイズ 1:最大化ウィンドウ 2:最小化してタスクバーに表示 3:タスクトレイに隠す 4:非表示	
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		サーバプロセスアプリケーションのウィンドウ状態をクライアントプロセスから制御する場合に使用します。



2.2.1.8. FTCORE_LoadDefaultCategory()

概要		サーバープロセスアプリケーションにデフォルトカテゴリーの設定を指示します。
FTCORE_RESULT FTCORE_LoadDefaultCategory()		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		サーバープロセスアプリケーションが認識するカテゴリーを 2.2.2.1DefaultCategory で示すデフォルトに戻すように指示します。

2.2.1.9. FTCORE_SetCustomCategory(string)

概要		サーバープロセスアプリケーションにカスタムカテゴリーの設定を指示します。
FTCORE_RESULT FTCORE_SetCustomCategory(string categories)		
categories		任意のカテゴリー文字列(複数文字列の場合はカンマで区切る)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		<p>サーバープロセスアプリケーションが認識するカテゴリーに categories で指定のカテゴリーを設定するように指示します。</p> <p>本 API を使用することにより、任意に指定した文字列がカテゴリーとして認識され、サーバープロセスアプリケーションのフィルタ機能などが正しく動作するようになります。</p> <p>複数のカテゴリーを指定する場合はカンマで区切ります。</p> <p>サーバープロセスアプリケーションが認識するカテゴリーでは、文字列 NON が ”どのカテゴリーにも含まれない” 意味を示す予約語になっています。</p> <p>categories に NON を含める/含めないにかかわらず、サーバープロセスアプリケーションでは、カテゴリー列挙の先頭に NON が挿入されて動作します。</p>



2.2.1.10. FTCore_SetCustomCategoryFromFile(string)

概要		サーバープロセスアプリケーションにファイル読み込みでカスタムカテゴリーの設定を指示します。
FTCORE_RESULT FTCore_SetCustomCategoryFromFile(string filename)		
filename		読み込みするカテゴリー設定ファイルの完全パス名
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		<p>サーバープロセスアプリケーションが認識するカテゴリーにファイルから読み込んだカテゴリーを設定するように指示します。</p> <p>本 API を使用することにより、任意に指定した文字列がカテゴリーとして認識され、サーバープロセスアプリケーションのフィルタ機能などが正しく動作するようになります。</p> <p>カテゴリーファイルは、サーバープロセスアプリケーションの [高度な設定] メニューを選択しカテゴリーソースタイプで [User definition] を選択すると作成することができます。</p> <p>詳細は Logging Foot Viewer アプリケーション操作ガイド [2.15.高度な設定を行う] の項目を参照してください。</p>



2.2.2. 列挙型

DefaultCategory 及び Severity について FTCCORE_SendMessage() 関数引数には任意の文字列を渡すことができますが、サーバープロセスのアプリケーション側でフィルタ機能が動作する際、列挙型として定義された文字列と比較されます。サーバープロセスのアプリケーションでフィルタ機能を正しく動作させるには、ここで定義されている列挙型と同じ文字列を使用する必要があります。

2.2.2.1. DefaultCategory

出力するログのカテゴリを表します。

列挙型	DefaultCategory	
メンバー	値	使用例
NON	0	対象外、無効値
APP	1	アプリケーション動作
SYSTEM	2	システム関連
USER	3	ユーザー操作
UI	4	画面側発生イベント
WF	5	ビジネスロジック、ワークフロー発生イベント
DEVICE	6	デバイス発生イベント
DEBUG	7	デバッグ用
STEP	8	実行ステップ
EVENT	9	イベント
COMM	10	通信関連

2.2.2.2. Severity

出力するログの重要度を表します。

列挙型	Severity	
メンバー	値	使用例
NONE	0	対象外、無効値
INFO	1	一般的な情報
NOTICE	2	注意、注目
WARNING	3	操作ミス、警告
ERR	4	復帰可能なエラー、操作ミス
FATAL	5	復帰不可能なエラー、重大な障害



2.2.2.3. FTCORE_RESULT

API が返す戻り値の種類を表します。

列挙型	FTCORE_RESULT	
メンバー	値	説明
FTCORE_UNKNOWN_STATE	-1	対象外、無効値
FTCORE_SUCCESS	0	成功
FTCORE_EVT_CLIENT_DETECTDISCON	4	サーバープロセスとの接続切断を検知
FTCORE_EVT_CLIENT_RECEIVED	5	サーバープロセスからデータ受信
FTCORE_ERR_CLIENT_NOTSERVER	21	接続可能なサーバープロセスが存在しない
FTCORE_ERR_CLIENT_FAILSERVER	22	サーバープロセスの起動に失敗
FTCORE_ERR_CLIENT_NOEXIST	23	クライアントプロセスが生成されていない
FTCORE_ERR_CLIENT_ALREADY	24	クライアントプロセスは生成済みである
FTCORE_ERR_CLIENT_PARAMETER	25	送信パラメーターエラー発生
FTCORE_ERR_CLIENT_HOSTINFO	26	宛先情報の取得に失敗
FTCORE_ERR_CLIENT_SOCKET	27	ソケットエラー発生
FTCORE_ERR_CLIENT_REFUSED	28	サーバープロセスとの接続を拒否された
FTCORE_ERR_CLIENT_UNREACHED	29	サーバープロセスへのネットワークパスが存在しない
FTCORE_ERR_CLIENT_CONNECT	30	サーバープロセスとの接続に失敗
FTCORE_ERR_CLIENT_ESTABLISH	31	サーバープロセスとの接続確立中にエラー発生
FTCORE_ERR_CLIENT_MESSAGESEND	32	サーバープロセスへメッセージ送信に失敗
FTCORE_ERR_CLIENT_MESSAGE_RECV	33	サーバープロセスからのメッセージ受信に失敗
FTCORE_ERR_CLIENT_CONNUNKNOWN	34	不明なサーバーから接続受諾通知を受けた



2.2.3. 開発環境

本プログラムは下記の開発環境において動作確認しています。下記と異なる開発環境をご使用の場合にはビルドエラーが発生する場合がありますので適宜ご対応ください。

IDE(統合開発環境)	Visual Studio 2022 Community
.NET 環境(ターゲットフレームワーク)	.NET Framework 4.7.2

2.2.4. FTSDK の使用方法

C#では、ラッパープログラムが提供されており、本ラッパープログラムを通して FTSDK を使用することを推奨します。

開発環境は Microsoft Visual Studio を想定し、C# プラットフォーム呼び出し(P/Invoke)の経験がある開発者を対象としています。

以降の説明で表記するパスは、

標準のインストールフォルダ(C:\Program Files\LoggingFoot*)下のパスを示します。

sample\C#\ftsdk\src フォルダに存在する ラッパープログラムのソースファイル FtsdkCSWrapper.cs をプロジェクトに追加します。

sample\C#\ftsdk\bin フォルダに存在する DLL ファイルをお客様プログラムの実行フォルダに配置します。



2.2.5. サンプルプログラム

サンプルプログラムは sample¥C# に収録されています。

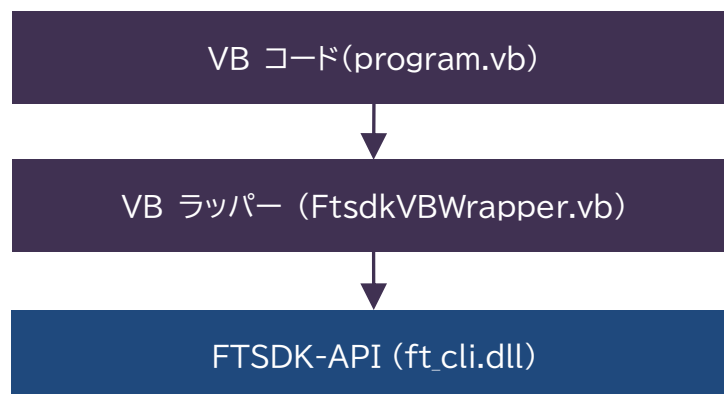
各プログラムの動作説明は [2.6 サンプルプログラム動作説明](#) を参照してください。

プログラム名	説明
Sample_StartProcess	サーバープロセスアプリケーションが起動している状態でクライアントプロセスを起動しロギングを開始するサンプルプログラムです。 使用されている主な API: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()
Sample_StartupServer	クライアントプロセスがサーバープロセスを起動してロギングを開始するサンプルプログラムです。 使用されている主な API: FTCORE_StartTriggerWithParam() FTCORE_StartTriggerWithSetup() FTCORE_SendMessage() FTCORE_ExitTrigger()
Sample_SetCustomCategory	サーバープロセスアプリケーションにカスタムカテゴリーの設定を指示するサンプルプログラムです。 使用されている主な API: FTCORE_LoadDefaultCategory() FTCORE_SetCustomCategory() FTCORE_SetCustomCategoryFromFile()
Sample_SetWindowState	サーバープロセスアプリケーションのウィンドウ状態をクライアントプロセスから制御するサンプルプログラムです。 使用されている主な API: FTCORE_SetWindowState()
Sample_UnitTest	大量のログを出力するサンプルプログラムです。 使用されている主な API: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()



2.3. VB

VB では、ラッパープログラムが提供され、下図のようなレイヤーで FTSDK-API が呼ばれます。本節で説明する API 仕様は、ラッパープログラムの内容になります。



2.3.1. API 仕様

2.3.1.1. FTCORE_StartProcess(String, Integer)

概要		ロギングを開始します。
FTCORE_StartProcess(<i>servername</i> As String, <i>portnumber</i> As Integer) As FTCORE_RESULT		
<i>servername</i>		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
<i>portnumber</i>		接続先 PC のポート番号(49152～65535)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		接続先 PC で起動しているサーバプロセスと接続を確立し、ロギングを開始します。接続先 PC 名にはローカル又はリモートの PC を指定できます。接続先 PC のサーバプロセスはあらかじめ起動しておく必要があります。接続先 PC がローカル PC の場合、 <i>servername</i> には "localhost" を指定可能です。

**2.3.1.2. FTCORE_StartTriggerWithParam(String, String, Integer, Integer, Boolean)**

概要		指定のパラメーターでサーバープロセスを起動し、ロギングを開始します。
FTCORE_StartTriggerWithParam(<i>serverpath</i> As String, <i>servername</i> As String, <i>portnumber</i> As Integer, <i>viewstate</i> As Integer, <i>topmost</i> As Boolean) As FTCORE_RESULT		
<i>serverpath</i>		サーバープロセスの完全パスを含む実行ファイル名
<i>servername</i>		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
<i>portnumber</i>		接続先 PC のポート番号(49152～65535)
<i>viewstate</i>		サーバープロセスアプリケーション起動時のウィンドウ表示状態 0:デフォルトのウィンドウサイズ 1:最大化ウィンドウ 2:最小化してタスクバーに表示 3:タスクトレイに隠す 4:非表示
<i>topmost</i>		最前面表示 False:無効 True :有効
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOTSERVER FTCORE_ERR_CLIENT_FAILSERVER FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		指定パスのサーバープロセスを起動して接続を確立し、ロギングを開始します。本 API は同一ローカル PC 内でのみ動作します。 <i>serverpath</i> にはローカル PC に存在する ft_viewer.exe を完全パスで指定します。 <i>servername</i> にはローカル PC の PC 名又は IP アドレスを指定します。 同一のローカル PC であるため、"localhost" を指定可能です。



2.3.1.3. FTCORE_StartTriggerWithSetup(String, String, Integer)

概要		設定済みのパラメーターでサーバープロセスを起動し、ロギングを開始します。
FTCORE_StartTriggerWithSetup(<i>serverpath</i> As String, <i>servername</i> As String, <i>portnumber</i> As Integer) As FTCORE_RESULT		
<i>serverpath</i>		サーバープロセスの完全パスを含む実行ファイル名
<i>servername</i>		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
<i>portnumber</i>		接続先 PC のポート番号(49152～65535)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOTSERVER FTCORE_ERR_CLIENT_FAILSERVER FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		<p>セットアッププログラムにより設定済みのパラメーターで指定パスのサーバープロセスを起動して接続を確立し、ロギングを開始します。事前にセットアッププログラムを実行してパラメーターを設定しておく必要があります。(セットアッププログラムを実行していない場合は既定値のパラメーターで動作します)</p> <p>本 API は同一ローカル PC 内でのみ動作します。</p> <p><i>serverpath</i> にはローカル PC に存在する ft_viewer.exe を完全パスで指定します。</p> <p><i>servername</i> にはローカル PC の PC 名又は IP アドレスを指定します。</p> <p>同一のローカル PC であるため、"localhost" を指定可能です。</p>



2.3.1.4. FTCORE_ExitProcess()

概要		ロギングを終了します。
FTCORE_ExitProcess() As FTCORE_RESULT		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	-----
解説		FTCORE_StartProcess()によりロギングを開始した場合、 本 API でロギングを終了します。

2.3.1.5. FTCORE_ExitTrigger()

概要		ロギングを終了します。
FTCORE_ExitTrigger() As FTCORE_RESULT		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	-----
解説		FTCORE_StartTriggerWithParam()又は FTCORE_StartTriggerWithSetup()によりロギングを開始した場合、 本 API でロギングを終了します。

**2.3.1.6. FTCORE_SendMessage(String, String, String, String)**

概要		サーバープロセスにログを出力します。
FTCORE_SendMessage(<i>path</i> As String, <i>category</i> As String, <i>severity</i> As String, <i>message</i> As String) As FTCORE_RESULT		
<i>path</i>		呼び出し元など任意のパス
<i>category</i>		ログカテゴリー
<i>severity</i>		ログ重要度
<i>message</i>		出力する任意のログ文字列
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		各引数に指定した文字列をサーバープロセスにログとして出力します。 <i>category</i> 及び <i>severity</i> については、 <u>2.3.2 列挙型</u> で示す列挙型と同じ文字列を指定することを推奨します。

2.3.1.7. FTCORE_SetWindowState(Integer)

概要	サーバープロセスアプリケーションにウィンドウ表示状態の設定を指示します。	
FTCORE_SetWindowState(<i>state</i> As Integer) As FTCORE_RESULT		
<i>state</i>	サーバープロセス アプリケーションのウィンドウ表示状態 0:デフォルトのウィンドウサイズ 1:最大化ウィンドウ 2:最小化してタスクバーに表示 3:タスクトレイに隠す 4:非表示	
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説	サーバープロセスアプリケーションのウィンドウ状態をクライアントプロセスから制御する場合に使用します。	



2.3.1.8. FTCORE_LoadDefaultCategory()

概要		サーバープロセスアプリケーションにデフォルトカテゴリーの設定を指示します。
FTCORE_LoadDefaultCategory() As FTCORE_RESULT		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		サーバープロセスアプリケーションが認識するカテゴリーを 2.3.2.1DefaultCategory で示すデフォルトに戻すように指示します。

2.3.1.9. FTCORE_SetCustomCategory(String)

概要		サーバープロセスアプリケーションにカスタムカテゴリーの設定を指示します。
FTCORE_SetCustomCategory(<i>categories</i> As String) As FTCORE_RESULT		
<i>categories</i>		任意のカテゴリー文字列(複数文字列の場合はカンマで区切る)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		<p>サーバープロセスアプリケーションが認識するカテゴリーに <i>categories</i> で指定のカテゴリーを設定するように指示します。</p> <p>本 API を使用することにより、任意に指定した文字列がカテゴリーとして認識され、サーバープロセスアプリケーションのフィルタ機能などが正しく動作するようになります。</p> <p>複数のカテゴリーを指定する場合はカンマで区切ります。</p> <p>サーバープロセスアプリケーションが認識するカテゴリーでは、文字列 NON が ”どのカテゴリーにも含まれない” 意味を示す予約語になっています。</p> <p><i>categories</i> に NON を含める/含めないにかかわらず、サーバープロセスアプリケーションでは、カテゴリー列挙の先頭に NON が挿入されて動作します。</p>



2.3.1.10. FTCORE_SetCustomCategoryFromFile(String)

概要		サーバープロセスアプリケーションにファイル読み込みでカスタムカテゴリーの設定を指示します。
FTCORE_SetCustomCategoryFromFile(<i>filename</i> As String) As FTCORE_RESULT		
<i>filename</i>		読み込みするカテゴリー設定ファイルの完全パス名
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		<p>サーバープロセスアプリケーションが認識するカテゴリーにファイルから読み込んだカテゴリーを設定するように指示します。</p> <p>本 API を使用することにより、任意に指定した文字列がカテゴリーとして認識され、サーバープロセスアプリケーションのフィルタ機能などが正しく動作するようになります。</p> <p>カテゴリーファイルは、サーバープロセスアプリケーションの [高度な設定] メニューを選択しカテゴリーソースタイプで [User definition] を選択すると作成することができます。</p> <p>詳細は Logging Foot Viewer アプリケーション操作ガイド [2.15.高度な設定を行う] の項目を参照してください。</p>



2.3.2. 列挙型

DefaultCategory 及び Severity について FTCORE_SendMessage() 関数引数には任意の文字列を渡すことができますが、サーバープロセスのアプリケーション側でフィルタ機能が動作する際、列挙型として定義された文字列と比較されます。サーバープロセスのアプリケーションでフィルタ機能を正しく動作させるには、ここで定義されている列挙型と同じ文字列を使用する必要があります。

2.3.2.1. DefaultCategory

出力するログのカテゴリを表します。

列挙型	DefaultCategory	
メンバー	値	使用例
NON	0	対象外、無効値
APP	1	アプリケーション動作
SYSTEM	2	システム関連
USER	3	ユーザー操作
UI	4	画面側発生イベント
WF	5	ビジネスロジック、ワークフロー発生イベント
DEVICE	6	デバイス発生イベント
DEBUG	7	デバッグ用
STEP	8	実行ステップ
EVENT	9	イベント
COMM	10	通信関連

2.3.2.2. Severity

出力するログの重要度を表します。

列挙型	Severity	
メンバー	値	使用例
NONE	0	対象外、無効値
INFO	1	一般的な情報
NOTICE	2	注意、注目
WARNING	3	操作ミス、警告
ERR	4	復帰可能なエラー、操作ミス
FATAL	5	復帰不可能なエラー、重大な障害



2.3.2.3. FTCORE_RESULT

API が返す戻り値の種類を表します。

列挙型	FTCORE_RESULT	
メンバー	値	説明
FTCORE_UNKNOWN_STATE	-1	対象外、無効値
FTCORE_SUCCESS	0	成功
FTCORE_EVT_CLIENT_DETECTDISCON	4	サーバープロセスとの接続切断を検知
FTCORE_EVT_CLIENT_RECEIVED	5	サーバープロセスからデータ受信
FTCORE_ERR_CLIENT_NOTSERVER	21	接続可能なサーバープロセスが存在しない
FTCORE_ERR_CLIENT_FAILSERVER	22	サーバープロセスの起動に失敗
FTCORE_ERR_CLIENT_NOEXIST	23	クライアントプロセスが生成されていない
FTCORE_ERR_CLIENT_ALREADY	24	クライアントプロセスは生成済みである
FTCORE_ERR_CLIENT_PARAMETER	25	送信パラメーターエラー発生
FTCORE_ERR_CLIENT_HOSTINFO	26	宛先情報の取得に失敗
FTCORE_ERR_CLIENT_SOCKET	27	ソケットエラー発生
FTCORE_ERR_CLIENT_REFUSED	28	サーバープロセスとの接続を拒否された
FTCORE_ERR_CLIENT_UNREACHED	29	サーバープロセスへのネットワークパスが存在しない
FTCORE_ERR_CLIENT_CONNECT	30	サーバープロセスとの接続に失敗
FTCORE_ERR_CLIENT_ESTABLISH	31	サーバープロセスとの接続確立中にエラー発生
FTCORE_ERR_CLIENT_MESSAGESEND	32	サーバープロセスへメッセージ送信に失敗
FTCORE_ERR_CLIENT_MESSAGE_RECV	33	サーバープロセスからのメッセージ受信に失敗
FTCORE_ERR_CLIENT_CONNUNKNOWN	34	不明なサーバーから接続受諾通知を受けた



2.3.3. 開発環境

本プログラムは下記の開発環境において動作確認しています。下記と異なる開発環境をご使用の場合にはビルドエラーが発生する場合がありますので適宜ご対応ください。

IDE(統合開発環境)	Visual Studio 2022 Community
.NET 環境(ターゲットフレームワーク)	.NET Framework 4.7.2

2.3.4. FTSDK の使用方法

VB では、ラッパープログラムが提供されており、本ラッパープログラムを通して FTSDK を使用することを推奨します。

開発環境は Microsoft Visual Studio を想定し、VB プラットフォーム呼び出し(P/Invoke)の経験がある開発者を対象としています。

以降の説明で表記するパスは、

標準のインストールフォルダ(C:\Program Files\LoggingFoot*)下のパスを示します。

sample\VB\ftsdk\src フォルダに存在する ラッパープログラムのソースファイル FtsdkVBWrapper.cs をプロジェクトに追加します。

sample\VB\ftsdk\bin フォルダに存在する DLL ファイルをお客様プログラムの実行フォルダに配置します。



2.3.5. サンプルプログラム

サンプルプログラムは sample¥VB に収録されています。

各プログラムの動作説明は [2.6 サンプルプログラム動作説明](#) を参照してください。

プログラム名	説明
Sample_StartProcess	サーバープロセスアプリケーションが起動している状態でクライアントプロセスを起動しロギングを開始するサンプルプログラムです。 使用されている主な API: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()
Sample_StartupServer	クライアントプロセスがサーバープロセスを起動してロギングを開始するサンプルプログラムです。 使用されている主な API: FTCORE_StartTriggerWithParam() FTCORE_StartTriggerWithSetup() FTCORE_SendMessage() FTCORE_ExitTrigger()
Sample_SetCustomCategory	サーバープロセスアプリケーションにカスタムカテゴリーの設定を指示するサンプルプログラムです。 使用されている主な API: FTCORE_LoadDefaultCategory() FTCORE_SetCustomCategory() FTCORE_SetCustomCategoryFromFile()
Sample_SetWindowState	サーバープロセスアプリケーションのウィンドウ状態をクライアントプロセスから制御するサンプルプログラムです。 使用されている主な API: FTCORE_SetWindowState()
Sample_UnitTest	大量のログを出力するサンプルプログラムです。 使用されている主な API: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()



2.4. JavaScript

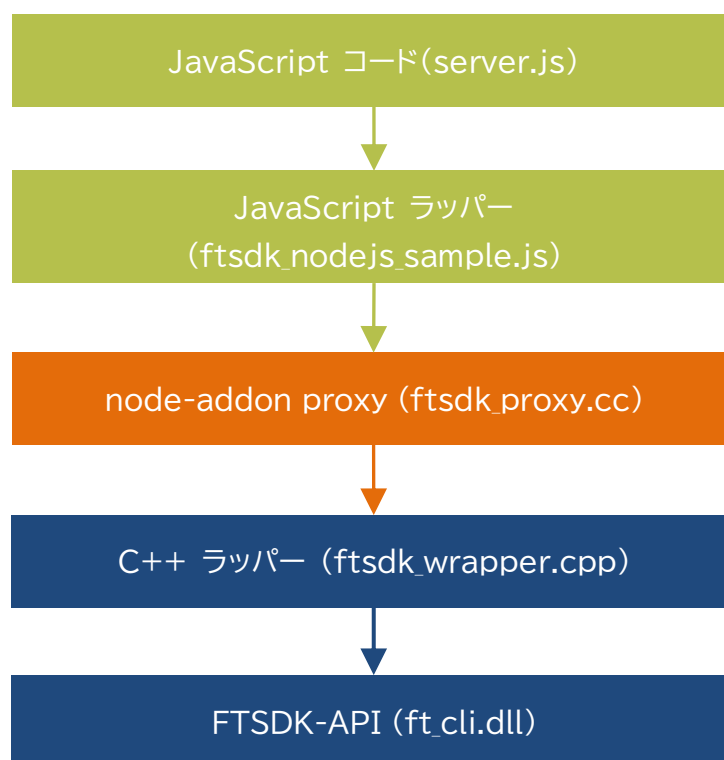
JavaScript では、Node.js を使用したバックエンド プログラムへの実装をサポートします。
(フロントエンドの実装には対応していません。)

本サンプルプログラムは、node-addon-api を使用してラッパープログラムを作成しています。

JavaScript コードから FTSDK-API を呼び出すには下図のようなレイヤーを辿ります。

括弧内は本サンプルプログラムで該当するソースファイル名又はモジュール名を示しています。

お客様のプログラムには、任意のレイヤーから実装することができますが、煩雑な node-addon 部分を避けるためにも JavaScript ラッパーレイヤー(ftsdk_nodejs_sample.js)の実装をお勧めします。本 API 仕様は、JavaScript ラッパーで呼び出しされる各 API の説明になります。





2.4.1. API 仕様

2.4.1.1. ftcoreStartProcess(Any, Number)

概要		ロギングを開始します。
ftcoreStartProcess(<i>servername</i> : Any, <i>portnumber</i> : Number) : Any		
<i>servername</i>		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
<i>portnumber</i>		接続先 PC のポート番号(49152～65535)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		接続先 PC で起動しているサーバプロセスと接続を確立し、ロギングを開始します。接続先 PC 名にはローカル又はリモートの PC を指定できます。 接続先 PC のサーバプロセスはあらかじめ起動しておく必要があります。 接続先 PC がローカル PC の場合、 <i>servername</i> には "localhost" を指定可能です。



2.4.1.2. ftcoreStartTriggerWithParam(Any, Any, Number, Number, Number)

概要		指定のパラメーターでサーバープロセスを起動し、ロギングを開始します。
ftcoreStartTriggerWithParam(<i>serverpath</i> : Any, <i>servername</i> : Any, <i>portnumber</i> : Number, <i>Viewstate</i> : Number, <i>topmost</i> : Number) : Any		
<i>serverpath</i>		サーバープロセスの完全パスを含む実行ファイル名
<i>servername</i>		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
<i>portnumber</i>		接続先 PC のポート番号(49152～65535)
<i>viewstate</i>		サーバープロセスアプリケーション起動時のウィンドウ表示状態 0:デフォルトのウィンドウサイズ 1:最大化ウィンドウ 2:最小化してタスクバーに表示 3:タスクトレイに隠す 4:非表示
<i>topmost</i>		最前面表示 false:無効 true :有効
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOTSERVER FTCORE_ERR_CLIENT_FAILSERVER FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		指定パスのサーバープロセスを起動して接続を確立し、ロギングを開始します。本 API は同一ローカル PC 内でのみ動作します。 <i>serverpath</i> にはローカル PC に存在する ft_viewer.exe を完全パスで指定します。 <i>servername</i> にはローカル PC の PC 名又は IP アドレスを指定します。 同一のローカル PC であるため、"localhost" を指定可能です。



2.4.1.3. ftcoreStartTriggerWithSetup(Any, Any, Number)

概要		設定済みのパラメーターでサーバープロセスを起動し、ロギングを開始します。	
		ftcoreStartTriggerWithSetup(<i>serverpath</i> : Any, <div><i>servername</i> : Any,</div> <div><i>portnumber</i> : Number) : Any</div>	
<i>serverpath</i>		サーバープロセスの完全パスを含む実行ファイル名	
<i>servername</i>		接続先 PC 名又は IP アドレス (xx.xx.xx.xx)	
<i>portnumber</i>		接続先 PC のポート番号 (49152～65535)	
戻り値	成功	FTCORE_SUCCESS	
	失敗	FTCORE_ERR_CLIENT_NOTSERVER FTCORE_ERR_CLIENT_FAILSERVER FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH	
解説		<p>セットアッププログラムにより設定済みのパラメーターで指定パスのサーバープロセスを起動して接続を確立し、ロギングを開始します。事前にセットアッププログラムを実行してパラメーターを設定しておく必要があります。(セットアッププログラムを実行していない場合は既定値のパラメーターで動作します)</p> <p>本 API は同一ローカル PC 内でのみ動作します。</p> <p><i>serverpath</i> にはローカル PC に存在する ft_viewer.exe を完全パスで指定します。</p> <p><i>servername</i> にはローカル PC の PC 名又は IP アドレスを指定します。</p> <p>同一のローカル PC であるため、"localhost" を指定可能です。</p>	



2.4.1.4. ftcoreExitProcess()

概要		ロギングを終了します。
ftcoreExitProcess() : Any		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	-----
解説		ftcoreStartProcess()によりロギングを開始した場合、 本 API でロギングを終了します。

2.4.1.5. ftcoreExitTrigger()

概要		ロギングを終了します。
ftcoreExitTrigger() : <i>Any</i>		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	-----
解説		ftcoreStartTriggerWithParam()又は ftcoreStartTriggerWithSetup()によりロギングを開始した場合、 本 API でロギングを終了します。



2.4.1.6. ftcoreSendMessage(Any, Any, Any, Number[])

概要		サーバプロセスにログを出力します。
ftcoreSendMessage(<i>path</i> : Any, <i>category</i> : Any, <i>severity</i> : Any, <i>message</i> : Number[]) : Any		
<i>path</i>		呼び出し元など任意のパス
<i>category</i>		ログカテゴリー
<i>severity</i>		ログ重要度
<i>message</i>		出力する任意のログ文字列を UTF8 にエンコードした数値配列
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		各引数に指定した文字列をサーバプロセスにログとして出力します。 <i>category</i> 及び <i>severity</i> については、 2.4.2 列挙型 で示す列挙型と同じ文字列を指定することを推奨します。また、文字化け防止のため、上位から受け取った文字列を UTF8 にエンコードし、Number[]配列として下位 API に渡します。実装例はサンプルコード(ftsdk_nodejs_sample.js)をご参照ください。



2.4.1.7. ftcoreSetWindowState(Number)

概要	サーバープロセスアプリケーションにウィンドウ表示状態の設定を指示します。	
ftcoreSetWindowState(<i>state</i> : Number) : Any		
<i>state</i>	サーバープロセス アプリケーションのウィンドウ表示状態 0:デフォルトのウィンドウサイズ 1:最大化ウィンドウ 2:最小化してタスクバーに表示 3:タスクトレイに隠す 4:非表示	
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説	サーバープロセスアプリケーションのウィンドウ状態をクライアントプロセスから制御する場合に使用します。	

2.4.1.8. ftcoreLoadDefaultCategory()

概要		サーバープロセスアプリケーションにデフォルトカテゴリーの設定を指示します。
ftcoreLoadDefaultCategory() : Any		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		サーバープロセスアプリケーションが認識するカテゴリーを <u>2.4.2.1DefaultCategory</u> で示すデフォルトに戻すように指示します。



2.4.1.9. ftcoreSetCustomCategory(Any)

概要		サーバープロセスアプリケーションにカスタムカテゴリーの設定を指示します。
ftcoreSetCustomCategory(<i>categories</i> : Any) : Any		
<i>categories</i>		任意のカテゴリー文字列(複数文字列の場合はカンマで区切る)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		<p>サーバープロセスアプリケーションが認識するカテゴリーに <i>categories</i> で指定のカテゴリーを設定するように指示します。</p> <p>本 API を使用することにより、任意に指定した文字列がカテゴリーとして認識され、サーバープロセスアプリケーションのフィルタ機能などが正しく動作するようになります。</p> <p>複数のカテゴリーを指定する場合はカンマで区切ります。</p> <p>サーバープロセスアプリケーションが認識するカテゴリーでは、文字列 NON が ”どのカテゴリーにも含まれない” 意味を示す予約語になっています。</p> <p><i>categories</i> に NON を含める/含めないにかかわらず、サーバープロセスアプリケーションでは、カテゴリー列挙の先頭に NON が挿入されて動作します。</p>



2.4.1.10. ftcoreSetCustomCategoryFromFile(Any)

概要		サーバープロセスアプリケーションにファイル読み込みでカスタムカテゴリーの設定を指示します。
ftcoreSetCustomCategoryFromFile(<i>filename</i> : Any) : Any		
<i>filename</i>		読み込みするカテゴリー設定ファイルの完全パス名
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		<p>サーバープロセスアプリケーションが認識するカテゴリーにファイルから読み込んだカテゴリーを設定するように指示します。</p> <p>本 API を使用することにより、任意に指定した文字列がカテゴリーとして認識され、サーバープロセスアプリケーションのフィルタ機能などが正しく動作するようになります。</p> <p>カテゴリーファイルは、サーバープロセスアプリケーションの [高度な設定] メニューを選択しカテゴリーソースタイプで [User definition] を選択すると作成することができます。</p> <p>詳細は Logging Foot Viewer アプリケーション操作ガイド [2.15.高度な設定を行う] の項目を参照してください。</p>



2.4.2. 列挙型

DefaultCategory 及び Severity について ftcoreSendMessage() 関数引数には任意の文字列を渡すことができますが、サーバープロセスのアプリケーション側でフィルタ機能が動作する際、列挙型として定義された文字列と比較されます。サーバープロセスのアプリケーションでフィルタ機能を正しく動作させるには、ここで定義されている列挙型と同じ文字列を使用する必要があります。

2.4.2.1. DefaultCategory

出力するログのカテゴリを表します。※ 実装例は front-end の program.js にあります。

列挙型	DefaultCategory	
メンバー	値	使用例
NON	0	対象外、無効値
APP	1	アプリケーション動作
SYSTEM	2	システム関連
USER	3	ユーザー操作
UI	4	画面側発生イベント
WF	5	ビジネスロジック、ワークフロー発生イベント
DEVICE	6	デバイス発生イベント
DEBUG	7	デバッグ用
STEP	8	実行ステップ
EVENT	9	イベント
COMM	10	通信関連

2.4.2.2. Severity

出力するログの重要度を表します。※ 実装例は front-end の program.js にあります。

列挙型	Severity	
メンバー	値	使用例
NONE	0	対象外、無効値
INFO	1	一般的な情報
NOTICE	2	注意、注目
WARNING	3	操作ミス、警告
ERR	4	復帰可能なエラー、操作ミス
FATAL	5	復帰不可能なエラー、重大な障害



2.4.2.3. FTCORE_RESULT

API が返す戻り値の種類を表します。※ 実装例は node-js の server.js にあります。

列挙型	FTCORE_RESULT	
メンバー	値	説明
FTCORE_UNKNOWN_STATE	-1	対象外、無効値
FTCORE_SUCCESS	0	成功
FTCORE_EVT_CLIENT_DETECTDISCON	4	サーバープロセスとの接続切断を検知
FTCORE_EVT_CLIENT_RECEIVED	5	サーバープロセスからデータ受信
FTCORE_ERR_CLIENT_NOTSERVER	21	接続可能なサーバープロセスが存在しない
FTCORE_ERR_CLIENT_FAILSERVER	22	サーバープロセスの起動に失敗
FTCORE_ERR_CLIENT_NOEXIST	23	クライアントプロセスが生成されていない
FTCORE_ERR_CLIENT_ALREADY	24	クライアントプロセスは生成済みである
FTCORE_ERR_CLIENT_PARAMETER	25	送信パラメーターエラー発生
FTCORE_ERR_CLIENT_HOSTINFO	26	宛先情報の取得に失敗
FTCORE_ERR_CLIENT_SOCKET	27	ソケットエラー発生
FTCORE_ERR_CLIENT_REFUSED	28	サーバープロセスとの接続を拒否された
FTCORE_ERR_CLIENT_UNREACHED	29	サーバープロセスへのネットワークパスが存在しない
FTCORE_ERR_CLIENT_CONNECT	30	サーバープロセスとの接続に失敗
FTCORE_ERR_CLIENT_ESTABLISH	31	サーバープロセスとの接続確立中にエラー発生
FTCORE_ERR_CLIENT_MESSAGESEND	32	サーバープロセスへメッセージ送信に失敗
FTCORE_ERR_CLIENT_MESSAGE_RECV	33	サーバープロセスからのメッセージ受信に失敗
FTCORE_ERR_CLIENT_CONNUNKNOWN	34	不明なサーバーから接続受諾通知を受けた



2.4.3. 開発環境

本プログラムは下記の開発環境において動作確認しています。

IDE(統合開発環境)	Visual Studio Code
フレームワーク	Node-js(バックエンドプログラムで使用)
使用アドオン	node-addon-api node-gyp bindings encoding-japanese
依存環境	Visual Studio ビルドツール 2022Community Python 3.13 Live Server

本章では、下記にインストールされているサンプルプログラムを例に説明します。

C:¥Program Files¥Logging Foot¥sample¥JavaScript¥Sample_StartProcess
以降の説明で表記するパスは、上記パスの下にあるものとします。

2.4.3.1. 環境構築

開発環境の構築には下記 URL が参考になります。

マイクロソフトによる node-js ガイドライン：

<https://github.com/Microsoft/nodejs-guidelines/blob/master/windows-environment.md#prerequisites>

node.js アドオンドキュメント：

<https://nodejs.org/api/addons.html>

構築手順（参考：個々の環境ごとに構築方法が異なるため）

(1) Node.js をインストール

<https://nodejs.org/ja>

(2) Visual Studio ビルドツールをインストール

<https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=BuildTools>

(3) Python 3.13 をインストール

<https://www.python.org/downloads>



(4) PowerShell の実行ポリシーを変更する

PowerShell を起動し、下記コマンドを入力します。これにより npm.ps1 の実行を許可することで npm コマンドの実行がブロックされなくなります。

```
Set-ExecutionPolicy RemoteSigned -Scope CurrentUser
```

(5) Visual Studio Code を起動する

Visual Studio Code を起動し、node-js フォルダを開きます。

(6) Live Server をインストール

Visual Studio Code の拡張機能(Extensions)より「Live Server」を検索してインストール

以降は、Visual Studio Code のターミナル上で実行します。

(7) プロジェクト初期化

```
npm init -y
```

※ Node.js をインストールしたにもかかわらず、npm が認識されない場合は、Visual Studio Code を再起動します。

(8) C++アドオン ビルドツールをインストール

```
npm install node-gyp
```

(9) .node ファイルの読み込みユーティリティ

```
npm install bindings
```

(10) N-API の C++ラッパー

```
npm install node-addon-api
```

(11) 文字コード変換ライブラリ

```
npm install --save encoding-japanese
```



2.4.3.2. ラッパープログラムのビルド

C++ラッパープログラム(ftsdk_wrapper.cpp)やプロキシプログラム(ftsdk_proxy.cc) をビルドするには、下記手順を実施します。

(1) Visual Studio Code ターミナルで下記コマンドを実行(npm install スペース ドット)

```
npm install .
```

ビルドが成功すると node-js¥build¥Release フォルダが生成されます。

(2)クライアントモジュールを実行環境 node-js¥build¥Release にコピーします。

C:¥Program Files¥LoggingFoot¥ftsdk¥core¥bin¥ft_cli.dll を

node-js¥build¥Release フォルダ内に配置してください。



2.4.4. 使用方法

本サンプルプログラムは、Visual Studio Code 及び Node.js の使用経験がある開発者を想定しているため、これらに関する詳細な操作方法等は割愛します。

以降の説明で表記するパスは、

標準のインストールフォルダ(C:\Program Files\LoggingFoot*)下のパスを示します。

ftsdk\wrappers\JavaScript\ftsdk_wrapper フォルダ及び ftsdk_nodejs_sample.js ファイルをお客様プログラムのプロジェクトフォルダに追加します。

FTSDK-API 呼び出し及び実装については、各サンプルプログラムを参考にしてください。

2.4.4.1. サンプルプログラム Sample_StartProcess をデバッグ実行する

サンプルプログラム sample\JavaScript\Sample_StartProcess を説明します。

あらかじめ、Viewer アプリケーション(ftviewer\ft_viewer.exe)を起動しておきます。

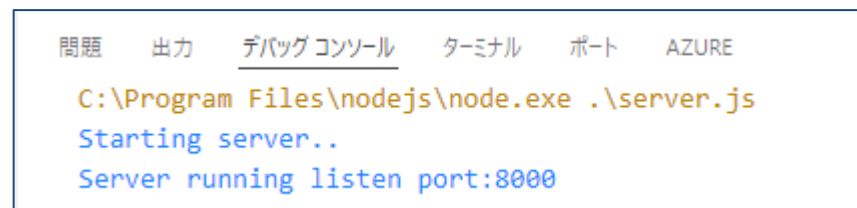
手順1. サーバーサイド側(node.js)を実行します。

(1) Visual Studio Code で下記フォルダ開きます。

sample\JavaScript\Sample_StartProcess\node.js

(2) 実行とデバッグのドロップダウンから「Startup program」を選択し、実行します。

(3) 実行に成功するとデバッグコンソールに下図のように表示されます。



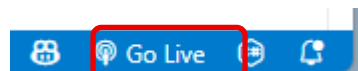
```
問題 出力 デバッグ コンソール ターミナル ポート AZURE
C:\Program Files\nodejs\node.exe .\server.js
Starting server..
Server running listen port:8000
```

手順2. フロントエンド側(front-end)を実行します。

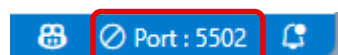
(1) Visual Studio Code で下記フォルダ開きます。

sample\JavaScript\Sample_StartProcess\front-end

(2) Visual Studio Code 画面右下の「Go Live」をクリックしてライブサーバーを起動します。



(3) 起動に成功すると下図のように使用ポート番号が表示され、ブラウザ UI が起動します。





Server Program Connection Settings

Server name: Port number:

Server Program Window state

Window state:

Logging Message Output

Category: ☐ Random ☒ Use default ☐ Use custom

Severity: ☐ Random

Message:

起動しない場合、ライブサーバーの使用ポート番号と launch.json に記載のポート番号（下図の例では `http://localhost:5502`）が一致しているか確認してください。

```
"name": "Launch Edge",  
"type": "msedge",  
"request": "launch",  
"url": "http://localhost:5502/index.html",  
"webRoot": "${workspaceFolder}"
```

手順3. ブラウザ UI を操作して Viewer アプリケーションに出力されることを確認します。

- (1) ブラウザ UI の「Connect」 ボタンを押下します。
- (2) Viewer アプリケーションに「Logging Server accepted request.」が表示されます。
表示されない場合、Viewer アプリケーション「一般設定」のコンフィグレーションタブにある「内部イベントログ設定」で「ロギングビュー表示を有効にする」のチェックが ON になっているか確認してください。
- (3) ブラウザ UI の「Message」テキストボックスに任意文字を入力し、「Send」 ボタンを押下します。
Viewer アプリケーションに入力した文字が表示されます。

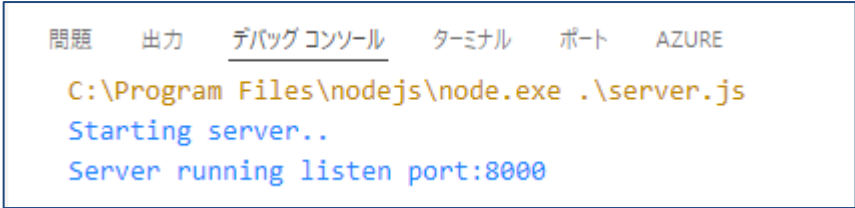


2.4.4.2. サンプルプログラム Sample_StartupServer をデバッグ実行する

サンプルプログラム sample¥JavaScript¥Sample_StartupServer を説明します。

手順1. サーバーサイド側(node-js)を実行します。

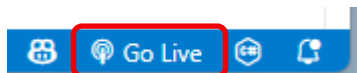
- (1) Visual Studio Code で下記フォルダ開きます。
sample¥JavaScript¥Sample_StartupServer¥node-js
- (2) 実行とデバッグのドロップダウンから「Startup program」を選択し、実行します。
- (3) 実行に成功するとデバッグコンソールに下図のように表示されます。



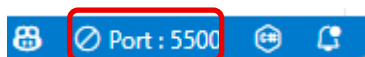
```
問題 出力 デバッグ コンソール ターミナル ポート AZURE
C:\Program Files\nodejs\node.exe .\server.js
Starting server..
Server running listen port:8000
```

手順2. フロントエンド側(front-end)を実行します。

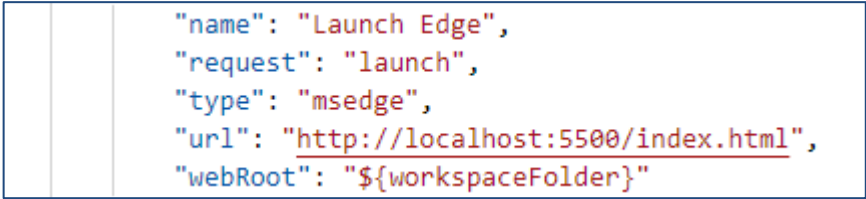
- (1) Visual Studio Code で下記フォルダ開きます。
sample¥JavaScript¥Sample_StartupServer¥front-end
- (2) Visual Studio Code 画面右下の「Go Live」をクリックしてライブサーバーを起動します。



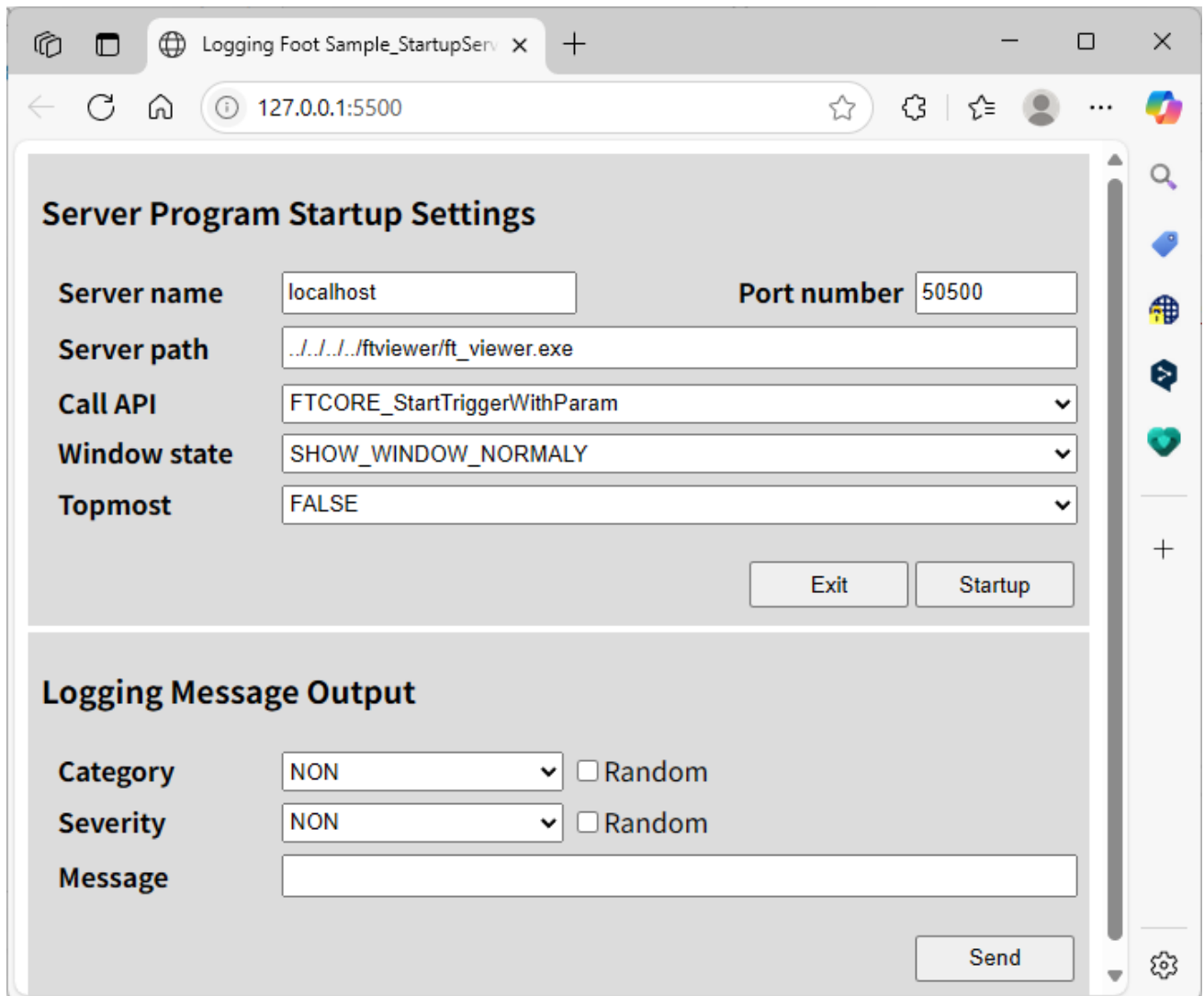
- (3) 起動に成功すると下図のように使用ポート番号が表示され、ブラウザ UI が起動します。



起動しない場合、ライブサーバーの使用ポート番号と launch.json に記載のポート番号（下図の例では http://localhost:5500）が一致しているか確認してください。



```
{
  "name": "Launch Edge",
  "request": "launch",
  "type": "msedge",
  "url": "http://localhost:5500/index.html",
  "webRoot": "${workspaceFolder}"
}
```

Server Program Startup Settings

Server name: localhost Port number: 50500

Server path: ../../../../../../ftviewer/ft_viewer.exe

Call API: FTCORE_StartTriggerWithParam

Window state: SHOW_WINDOW_NORMALY

Topmost: FALSE

Exit Startup

Logging Message Output

Category: NON ☐ Random

Severity: NON ☐ Random

Message:

Send

手順3. ブラウザ UI を操作して Viewer アプリケーションに出力されることを確認します。

(1) ブラウザ UI の「Server path」には、Viewer アプリケーションへの絶対パス又は下記ファイルを起点とした相対パスを入力し、「Startup」ボタンを押下します。

sample¥JavaScript¥Sample_StartupServer¥front-end¥index.html

(2) Viewer アプリケーションが起動し、「Logging Server accepted request.」が表示されます。表示されない場合、Viewer アプリケーション「一般設定」のコンフィグレーションタブにある「内部イベントログ設定」で「ロギングビュー表示を有効にする」のチェックが ON になっているか確認してください。

(3) ブラウザ UI の「Message」テキストボックスに任意文字を入力し、「Send」ボタンを押下します。

Viewer アプリケーションに入力した文字が表示されます。

(4) Viewer アプリケーションを終了する場合は、ブラウザ UI の「Exit」を押下します。



2.4.5. サンプルプログラム

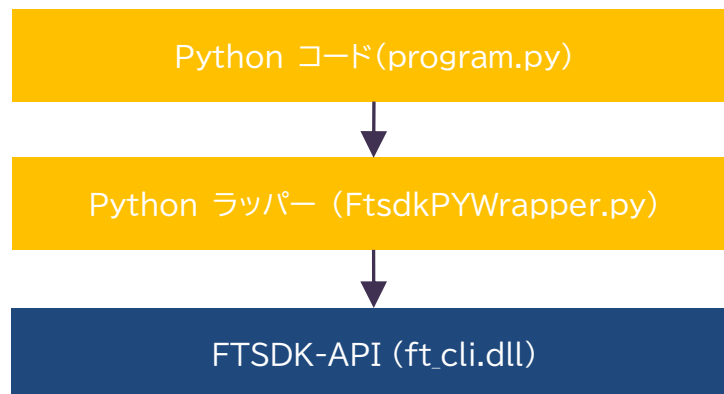
サンプルプログラムは sample¥JavaScript に収録されています。
フロントエンドで入力した文字列をバックエンドで取得し、Viewer アプリケーションに出力するサンプルです。

プログラム名	説明
Sample_StartProcess	サーバープロセスアプリケーションが起動している状態でクライアントプロセスを起動しロギングを開始するサンプルプログラムです。 使用されている主な API: ftcoreStartProcess() ftcoreSendMessage() ftcoreExitProcess() ftcoreSetWindowState() ftcoreSetCustomCategory() ftcoreLoadDefaultCategory()
Sample_StartupServer	クライアントプロセスがサーバープロセスを起動してロギングを開始するサンプルプログラムです。 使用されている主な API: ftcoreStartTriggerWithParam() ftcoreStartTriggerWithSetup() ftcoreSendMessage() ftcoreExitTrigger()



2.5. Python

Python では、ラッパープログラムが提供され、下図のようなレイヤーで FTSDK-API が呼ばれます。本節で説明する API 仕様は、ラッパープログラムの内容になります。



2.5.1. API 仕様

2.5.1.1. FTCORE_StartProcess(str, Int)

概要		ロギングを開始します。
FTCORE_StartProcess(<i>servername</i> : <i>str</i> , <i>Portnumber</i> : <i>int</i>) -> FTCORE_RESULT		
<i>servername</i>		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
<i>portnumber</i>		接続先 PC のポート番号(49152～65535)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		接続先 PC で起動しているサーバプロセスと接続を確立し、ロギングを開始します。接続先 PC 名にはローカル又はリモートの PC を指定できます。接続先 PC のサーバプロセスはあらかじめ起動しておく必要があります。接続先 PC がローカル PC の場合、 <i>servername</i> には "localhost" を指定可能です。



2.5.1.2. FTCORE_StartTriggerWithParam(str, str, int, int, int)

概要		指定のパラメーターでサーバープロセスを起動し、ロギングを開始します。
FTCORE_StartTriggerWithParam(<i>serverpath</i> : str, <i>servername</i> : str, <i>portnumber</i> : int, <i>viewstate</i> : int, <i>topmost</i> : int) -> FTCORE_RESULT		
<i>serverpath</i>		サーバープロセスの完全パスを含む実行ファイル名
<i>servername</i>		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
<i>portnumber</i>		接続先 PC のポート番号(49152~65535)
<i>viewstate</i>		サーバープロセスアプリケーション起動時のウィンドウ表示状態 0:デフォルトのウィンドウサイズ 1:最大化ウィンドウ 2:最小化してタスクバーに表示 3:タスクトレイに隠す 4:非表示
<i>topmost</i>		最前面表示 False:無効 True :有効
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOTSERVER FTCORE_ERR_CLIENT_FAILSERVER FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		指定パスのサーバープロセスを起動して接続を確立し、ロギングを開始します。本 API は同一ローカル PC 内でのみ動作します。 <i>serverpath</i> にはローカル PC に存在する ft_viewer.exe を完全パスで指定します。 <i>servername</i> にはローカル PC の PC 名又は IP アドレスを指定します。 同一のローカル PC であるため、"localhost" を指定可能です。



2.5.1.3. FTCORE_StartTriggerWithSetup(str, str, int)

概要		設定済みのパラメーターでサーバープロセスを起動し、ロギングを開始します。
FTCORE_StartTriggerWithSetup(<i>serverpath</i> : <i>str</i> , <i>servername</i> : <i>str</i> , <i>portnumber</i> : <i>int</i>) -> FTCORE_RESULT		
<i>serverpath</i>		サーバープロセスの完全パスを含む実行ファイル名
<i>servername</i>		接続先 PC 名又は IP アドレス(xx.xx.xx.xx)
<i>portnumber</i>		接続先 PC のポート番号(49152～65535)
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOTSERVER FTCORE_ERR_CLIENT_FAILSERVER FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
解説		<p>セットアッププログラムにより設定済みのパラメーターで指定パスのサーバープロセスを起動して接続を確立し、ロギングを開始します。事前にセットアッププログラムを実行してパラメーターを設定しておく必要があります。(セットアッププログラムを実行していない場合は既定値のパラメーターで動作します)</p> <p>本 API は同一ローカル PC 内でのみ動作します。</p> <p><i>serverpath</i> にはローカル PC に存在する ft_viewer.exe を完全パスで指定します。</p> <p><i>servername</i> にはローカル PC の PC 名又は IP アドレスを指定します。</p> <p>同一のローカル PC であるため、"localhost" を指定可能です。</p>



2.5.1.4. FTCORE_ExitProcess()

概要		ロギングを終了します。
FTCORE_ExitProcess() -> FTCORE_RESULT		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	-----
解説		FTCORE_StartProcess()によりロギングを開始した場合、 本 API でロギングを終了します。

2.5.1.5. FTCORE_ExitTrigger()

概要		ロギングを終了します。	
FTCORE_ExitTrigger() -> FTCORE_RESULT			
-----		引数無し	
戻り値	成功	FTCORE_SUCCESS	
	失敗	-----	
解説		FTCORE_StartTriggerWithParam()又は FTCORE_StartTriggerWithSetup()によりロギングを開始した場合、 本 API でロギングを終了します。	



2.5.1.6. FTCORE_SendMessage(str, str, str, str)

概要		サーバープロセスにログを出力します。
FTCORE_SendMessage(<i>path</i> : str , <div><i>category</i> : str, <i>severity</i> : str, <i>message</i> : str) -> FTCORE_RESULT</div>		
<i>path</i>		呼び出し元など任意のパス
<i>category</i>		ログカテゴリー
<i>severity</i>		ログ重要度
<i>message</i>		出力する任意のログ文字列
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		各引数に指定した文字列をサーバープロセスにログとして出力します。 <i>category</i> 及び <i>severity</i> については、 2.3.2 列挙型 で示す列挙型と同じ文字列を指定することを推奨します。

2.5.1.7. FTCORE_SetWindowState(int)

概要		サーバープロセスアプリケーションにウィンドウ表示状態の設定を指示します。
FTCORE_SetWindowState(<i>state</i> : int) -> FTCORE_RESULT		
state	サーバープロセス アプリケーションのウィンドウ表示状態 0:デフォルトのウィンドウサイズ 1:最大化ウィンドウ 2:最小化してタスクバーに表示 3:タスクトレイに隠す 4:非表示	
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		サーバープロセスアプリケーションのウィンドウ状態をクライアントプロセスから制御する場合に使用します。



2.5.1.8. FTCORE_LoadDefaultCategory()

概要		サーバープロセスアプリケーションにデフォルトカテゴリーの設定を指示します。
FTCORE_LoadDefaultCategory() -> FTCORE_RESULT		
-----		引数無し
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		サーバープロセスアプリケーションが認識するカテゴリーを <u>2.3.2.1DefaultCategory</u> で示すデフォルトに戻すように指示します。

2.5.1.9. FTCORE_SetCustomCategory(str)

概要		サーバープロセスアプリケーションにカスタムカテゴリーの設定を指示します。	
FTCORE_SetCustomCategory(<i>categories</i> : str) -> FTCORE_RESULT			
<i>categories</i>		任意のカテゴリー文字列(複数文字列の場合はカンマで区切る)	
戻り値	成功	FTCORE_SUCCESS	
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND	
解説		<p>サーバープロセスアプリケーションが認識するカテゴリーに <i>categories</i> で指定のカテゴリーを設定するように指示します。</p> <p>本 API を使用することにより、任意に指定した文字列がカテゴリーとして認識され、サーバープロセスアプリケーションのフィルタ機能などが正しく動作するようになります。</p> <p>複数のカテゴリーを指定する場合はカンマで区切ります。</p> <p>サーバープロセスアプリケーションが認識するカテゴリーでは、文字列 NON が ”どのカテゴリーにも含まれない” 意味を示す予約語になっています。</p> <p><i>categories</i> に NON を含める/含めないにかかわらず、サーバープロセスアプリケーションでは、カテゴリー列挙の先頭に NON が挿入されて動作します。</p>	



2.5.1.10. FTCORE_SetCustomCategoryFromFile(str)

概要		サーバープロセスアプリケーションにファイル読み込みでカスタムカテゴリーの設定を指示します。
FTCORE_SetCustomCategoryFromFile(<i>filename</i> : str) -> FTCORE_RESULT		
<i>filename</i>		読み込みするカテゴリー設定ファイルの完全パス名
戻り値	成功	FTCORE_SUCCESS
	失敗	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
解説		<p>サーバープロセスアプリケーションが認識するカテゴリーにファイルから読み込んだカテゴリーを設定するように指示します。</p> <p>本 API を使用することにより、任意に指定した文字列がカテゴリーとして認識され、サーバープロセスアプリケーションのフィルタ機能などが正しく動作するようになります。</p> <p>カテゴリーファイルは、サーバープロセスアプリケーションの [高度な設定] メニューを選択しカテゴリーソースタイプで [User definition] を選択すると作成することができます。</p> <p>詳細は Logging Foot Viewer アプリケーション操作ガイド [2.15.高度な設定を行う] の項目を参照してください。</p>



2.5.2. 列挙型

DefaultCategory 及び Severity について FTCORE_SendMessage() 関数引数には任意の文字列を渡すことができますが、サーバープロセスのアプリケーション側でフィルタ機能が動作する際、列挙型として定義された文字列と比較されます。サーバープロセスのアプリケーションでフィルタ機能を正しく動作させるには、ここで定義されている列挙型と同じ文字列を使用する必要があります。

2.5.2.1. DefaultCategory

出力するログのカテゴリを表します。

列挙型	DefaultCategory	
メンバー	値	使用例
NON	0	対象外、無効値
APP	1	アプリケーション動作
SYSTEM	2	システム関連
USER	3	ユーザー操作
UI	4	画面側発生イベント
WF	5	ビジネスロジック、ワークフロー発生イベント
DEVICE	6	デバイス発生イベント
DEBUG	7	デバッグ用
STEP	8	実行ステップ
EVENT	9	イベント
COMM	10	通信関連

2.5.2.2. Severity

出力するログの重要度を表します。

列挙型	Severity	
メンバー	値	使用例
NONE	0	対象外、無効値
INFO	1	一般的な情報
NOTICE	2	注意、注目
WARNING	3	操作ミス、警告
ERR	4	復帰可能なエラー、操作ミス
FATAL	5	復帰不可能なエラー、重大な障害



2.5.2.3. FTCORE_RESULT

API が返す戻り値の種類を表します。

列挙型	FTCORE_RESULT	
メンバー	値	説明
FTCORE_UNKNOWN_STATE	-1	対象外、無効値
FTCORE_SUCCESS	0	成功
FTCORE_EVT_CLIENT_DETECTDISCON	4	サーバープロセスとの接続切断を検知
FTCORE_EVT_CLIENT_RECEIVED	5	サーバープロセスからデータ受信
FTCORE_ERR_CLIENT_NOTSERVER	21	接続可能なサーバープロセスが存在しない
FTCORE_ERR_CLIENT_FAILSERVER	22	サーバープロセスの起動に失敗
FTCORE_ERR_CLIENT_NOEXIST	23	クライアントプロセスが生成されていない
FTCORE_ERR_CLIENT_ALREADY	24	クライアントプロセスは生成済みである
FTCORE_ERR_CLIENT_PARAMETER	25	送信パラメーターエラー発生
FTCORE_ERR_CLIENT_HOSTINFO	26	宛先情報の取得に失敗
FTCORE_ERR_CLIENT_SOCKET	27	ソケットエラー発生
FTCORE_ERR_CLIENT_REFUSED	28	サーバープロセスとの接続を拒否された
FTCORE_ERR_CLIENT_UNREACHED	29	サーバープロセスへのネットワークパスが存在しない
FTCORE_ERR_CLIENT_CONNECT	30	サーバープロセスとの接続に失敗
FTCORE_ERR_CLIENT_ESTABLISH	31	サーバープロセスとの接続確立中にエラー発生
FTCORE_ERR_CLIENT_MESSAGESEND	32	サーバープロセスへメッセージ送信に失敗
FTCORE_ERR_CLIENT_MESSAGE_RECV	33	サーバープロセスからのメッセージ受信に失敗
FTCORE_ERR_CLIENT_CONNUNKNOWN	34	不明なサーバーから接続受諾通知を受けた



2.5.3. 開発環境

本プログラムは下記の開発環境において動作確認しています。下記と異なる開発環境をご使用の場合にはビルドエラーが発生する場合がありますので適宜ご対応ください。

IDE(統合開発環境)	Visual Studio 2022 Community
インタープリター	Python 3.13(64-bit)
使用アドオン	pywin32

2.5.3.1. 環境構築

(1) Python 3.13 をインストール

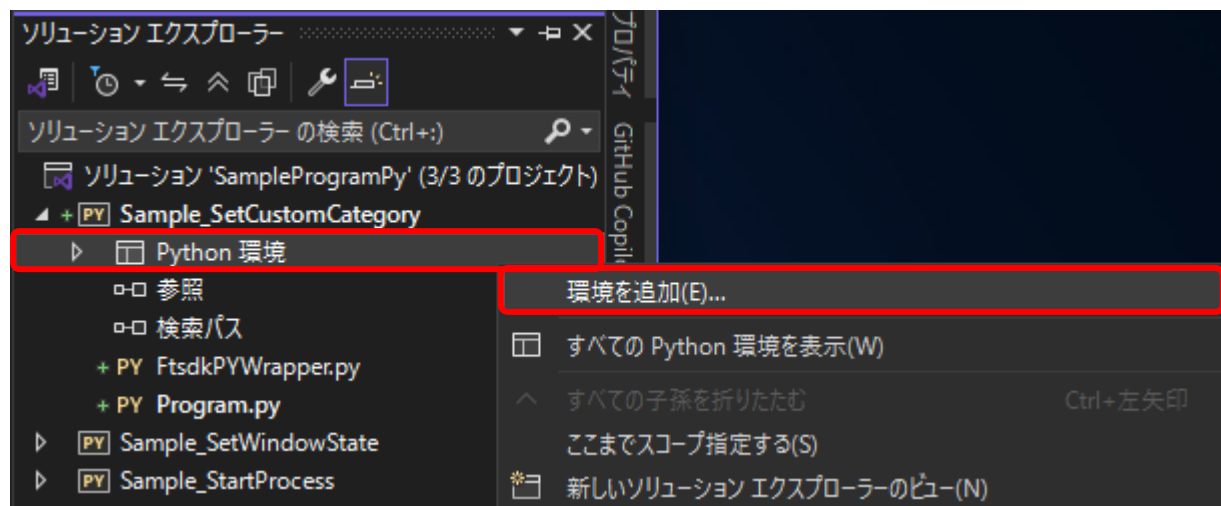
<https://www.python.org/downloads>

(2) Visual Studio 2022 Community を起動し、下記ソリューションを開く

C:\Program Files\Logging Foot\sample\Python\SampleProgramPy.sln

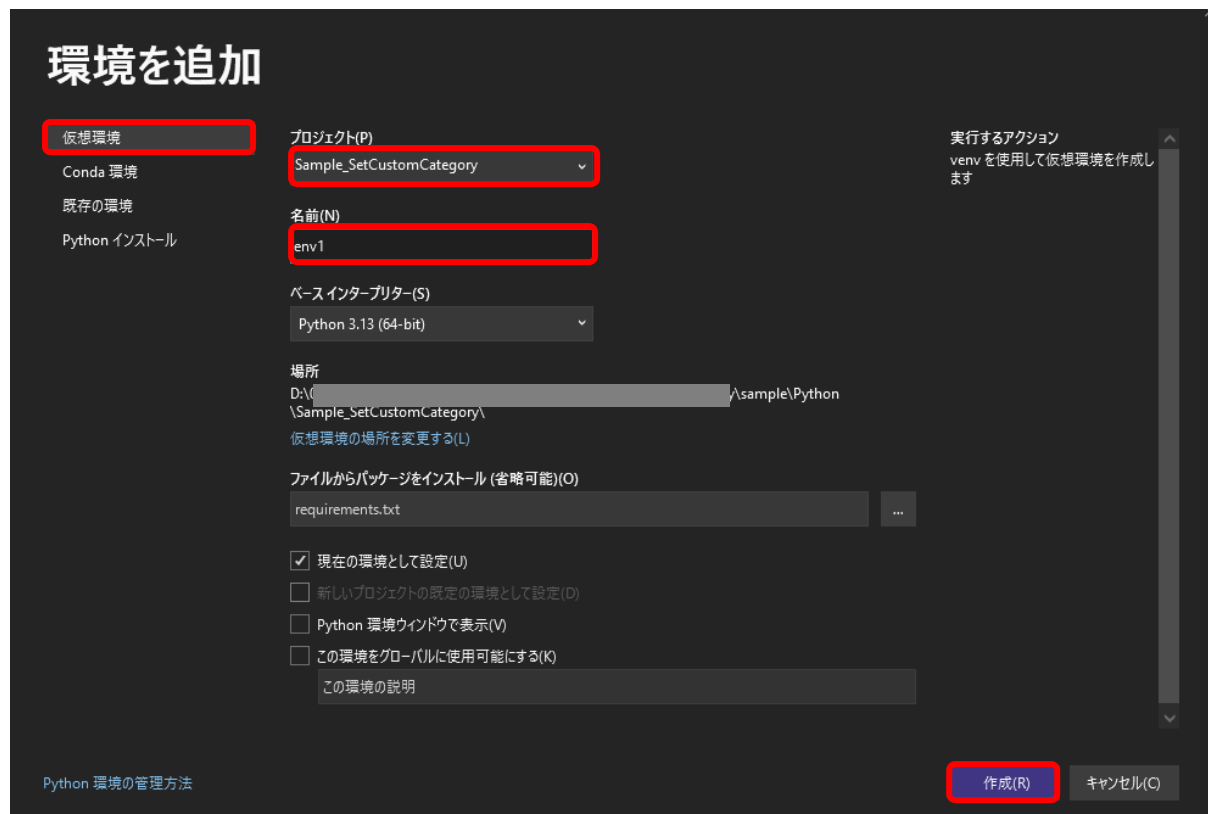
(3) 仮想環境を追加する

下図、ソリューションエクスプローラーで Python 環境を右クリックしてメニューを表示させ、[環境を追加]を選択します。



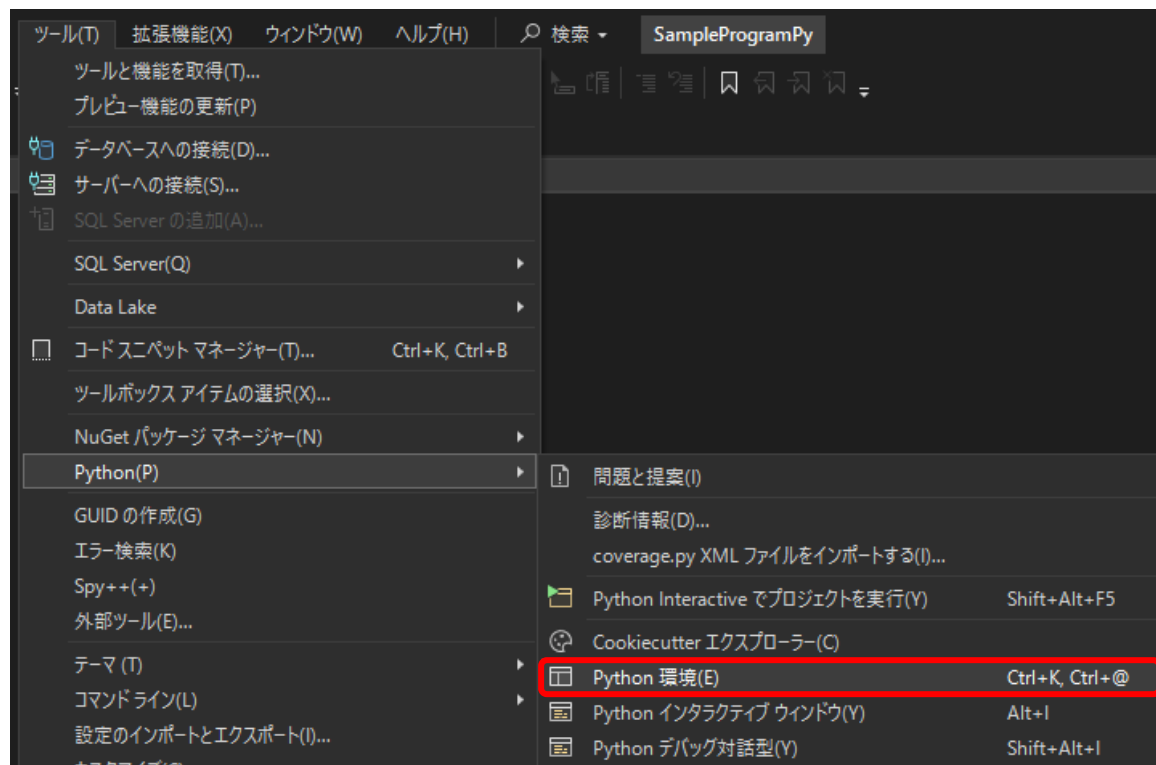


下図、左側メニューより「仮想環境」を選択し、対象のプロジェクトを選択します。
名前を入力して「作成」ボタンを押下します。



(4) pywin32 をインストール

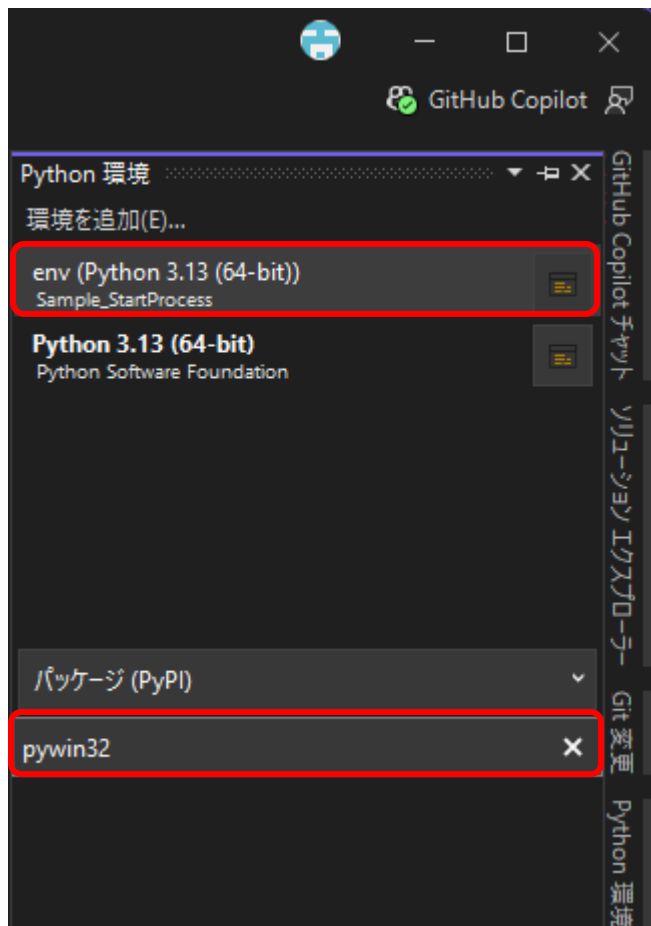
メニュー「ツール」>「Python」>「Python 環境」の順にメニューを選択します。





手順(3)で追加した仮想環境を選択し、パッケージ(PyPI)の入力ボックスに `pywin32` と入力後にエンターキーを押下します。

インストールが開始されます。



2.5.4. FTSDK の使用方法

以降の説明では、標準のインストールフォルダ(C:\Program Files\LoggingFoot*)下のパスを示します。

- (1) `sample\Python\ftsdk\src` フォルダに存在する ラッパープログラムのソースファイル `FtsdkPYWrapper.py` をプロジェクトに追加します。
- (2) `sample\Python\ftsdk\bin` フォルダに存在する `ft_cli.dll` をお客様プログラムの実行フォルダに配置します。
- (3) `FtsdkPYWrapper.py` ファイルの `_DLL_NAME = "..\¥ftsdk¥¥bin¥¥ft_cli.dll"` の部分を (2) の DLL を配置したパスに書き換えます。



2.5.5. サンプルプログラム

サンプルプログラムは sample¥Python に収録されています。

各プログラムの動作説明は [2.6 サンプルプログラム動作説明](#) を参照してください。

プログラム名	説明
Sample_StartProcess	サーバープロセスアプリケーションが起動している状態でクライアントプロセスを起動しロギングを開始するサンプルプログラムです。 使用されている主な API: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()
Sample_SetCustomCategory	サーバープロセスアプリケーションにカスタムカテゴリーの設定を指示するサンプルプログラムです。 使用されている主な API: FTCORE_LoadDefaultCategory() FTCORE_SetCustomCategory() FTCORE_SetCustomCategoryFromFile()
Sample_SetWindowState	サーバープロセスアプリケーションのウィンドウ状態をクライアントプロセスから制御するサンプルプログラムです。 使用されている主な API: FTCORE_SetWindowState()



2.6. サンプルプログラム動作説明

本節は C++、C#、VB、Python の各言語におけるサンプルプログラムの動作説明になります。
JavaScript のサンプルプログラム動作説明については [2.4.4 使用方法](#) を参照してください。
Python については、exe の提供はしていませんので、デバッガーから実行してください。
本節では特に断りがない限り、サーバープロセスを Logging Foot Viewer アプリケーション又は単に Viewer アプリケーションと表記します。
以降の説明で表記するパスは、
標準のインストールフォルダ(C:\Program Files\LoggingFoot*)下のパスを示します。

2.6.1. Sample_StartProcess

本サンプルプログラムでは、最も基本的な FTSDK-API の使用方法を示します。

- (1) 事前に Logging Foot Viewer アプリケーション(ftviewer\ft_viewer.exe)を実行しておきます。
- (2) Sample_StartProcess.exe を実行します。
sample\Cpp\Sample_StartProcess\bin\Debug 又は Release フォルダにあります。
- (3) exe が起動すると、キー入力待ちになりますので任意のキーを入力します。

```
C:\Program Files\Logging Foot  x + v
*****
LoggingFoot FTSDK sample program.
How to use FTCORE_StartProcess and FTCORE_LoadDefaultCategory.

Copyright c 2025 Quantyworks Software                                version: 1.0.0.1
*****

First, Please start Logging server.
And next, press any key. (Press Esc key to exit this program.)
Logging client will start.

> Wait for any key to be pressed ...
> |
```




(4) 例えば A キーを入力すると、下図の API が呼び出されます。

- FTCORE_StartProcess()
- FTCORE_LoadDefaultCategory()

```

C:\Program Files\Logging Foot
Copyright c 2025 Quantyworks Software
version: 1.0.0.1
*****
First, Please start Logging server.
And next, press any key. (Press Esc key to exit this program.)
Logging client will start.

> Wait for any key to be pressed ...
> a
> a key was pressed.
> Called FTCORE_StartProcess()..
> result: FTCORE_SUCCESS

> Called FTCORE_LoadDefaultCategory()..
> result: FTCORE_SUCCESS

> Press any key then, Output log message.

> |

```

下図 Viewer アプリケーションには、FTCORE_StartProcess() により接続が確立したこと、及び FTCORE_LoadDefaultCategory() によりデフォルトカテゴリーが更新されたことの内部ログが出力されます。

Mark	Date time	Caller path	Category	Severity	Message
	2025-07-06T16:09:49.503631		_LOG_	_INF_	Logging Server accepted request.
	2025-07-06T16:09:49.532482		_LOG_	_INF_	Updated default category: NON,APPSYSTEM,USER,UI,WFEDEVICE,DEBUG,STEPEVENT,COMM,

(5) 以降、任意キーを押下するたびに入力キーが Viewer アプリケーションにログ出力されます。

Mark	Date time	Caller path	Category	Severity	Message
	2025-07-06T17:00:11.709000		_LOG_	_INF_	Logging Server accepted request.
	2025-07-06T17:00:11.743714		_LOG_	_INF_	Updated default category: NON,APPSYSTEM,USER,UI,WFEDEVICE,DEBUG,STEPEVENT,COMM,
	2025-07-06T17:03:46.168151	Void Main()	EVENT	INFO	S key was pressed.
	2025-07-06T17:03:46.643765	Void Main()	NON	INFO	D key was pressed.
	2025-07-06T17:03:48.130452	Void Main()	USER	ERROR	F key was pressed.



(6) サンプルプログラムを終了するには Esc キーを入力した後、任意キーを入力します。

```
C:\Program Files\Logging Foc X + v - □ ×

> Press any key then, Output log message.

> d key was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> f key was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> ey was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> Called FTCORE_ExitProcess()..
> result: FTCORE_SUCCESS

Press any key again to exit program.
|
```

Viewer アプリケーションには接続が終了したことの内部ログが出力されます。



2.6.2. Sample_SetCustomCategory

本サンプルプログラムは、カスタムカテゴリーを設定する API の使用方法を示します。

(1) 事前に Logging Foot Viewer アプリケーション(ftviewer¥ft_viewer.exe)を実行しておきます。

(2) Sample_SetCustomCategory.exe を実行します。

sample¥Cpp¥Sample_SetCustomCategory¥bin¥Debug 又は Release フォルダにあります。

(3) exe が起動すると、キー入力待ちになりますので任意のキーを入力します。

(4) 例えば A キーを入力すると、下図の API が呼び出されます。

- FTCORE_StartProcess()

また、どの API を呼び出すかの選択肢が表示されます。

0 : FTCORE_LoadDefaultCategory()

1 : FTCORE_SetCustomCategory()

2 : FTCORE_SetCustomCategoryFromFile()



```
C:\Program Files\Logging Foc x + v - □ ×

*****

First, Please start Logging server.
And next, press any key. (Press Esc key to exit this program.)
Logging client will start.

> Wait for any key to be pressed ...
> a
> a key was pressed.
> Called FTCORE_StartProcess()..
> result: FTCORE_SUCCESS

> Press following number key, select API.
> 0 : FTCORE_LoadDefaultCategory()
> 1 : FTCORE_SetCustomCategory()
> 2 : FTCORE_SetCustomCategoryFromFile()

> |
```

(5) 呼び出しする API の番号を入力します。

例として 1 を入力すると、ソースコード内に定義してあるカスタムカテゴリーが設定されます。

```
C:\Program Files\Logging Foc x + v - □ ×

> a
> a key was pressed.
> Called FTCORE_StartProcess()..
> result: FTCORE_SUCCESS

> Press following number key, select API.
> 0 : FTCORE_LoadDefaultCategory()
> 1 : FTCORE_SetCustomCategory()
> 2 : FTCORE_SetCustomCategoryFromFile()

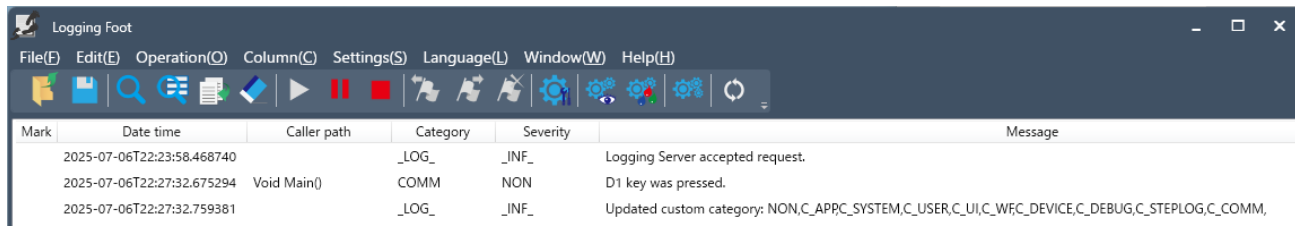
> 1 key was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> Called FTCORE_SetCustomCategory()..
> result: FTCORE_SUCCESS

> |
```

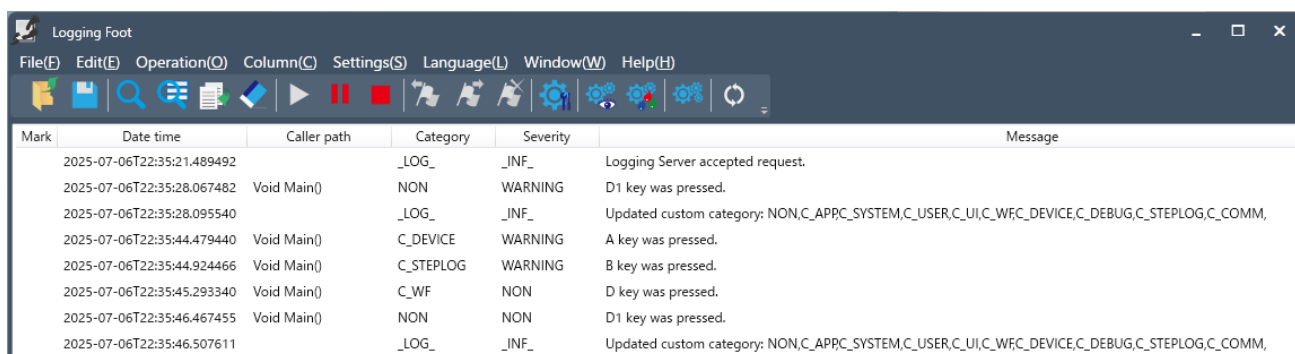


Viewer アプリケーションには `FTCORE_SetCustomCategory()` によりカスタムカテゴリーが更新されたことを示す内部ログが出力されます。



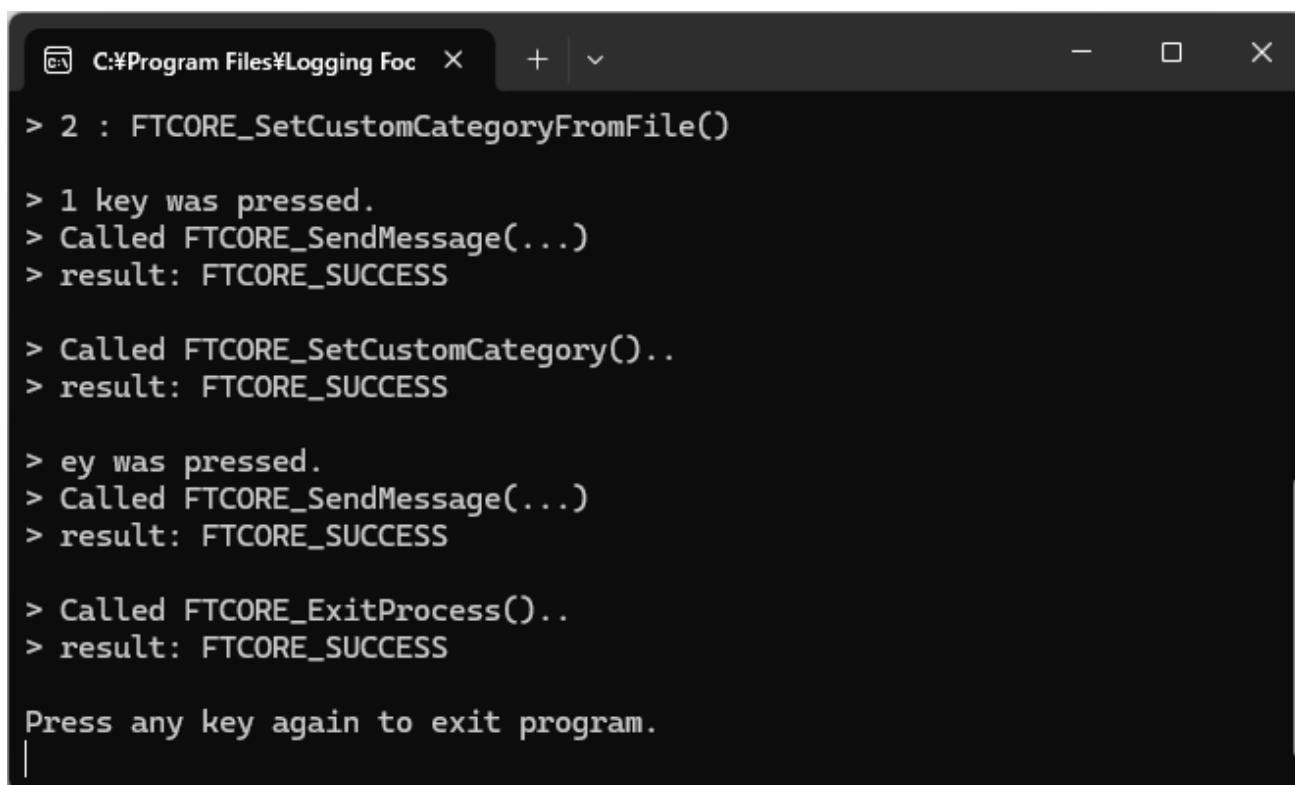
Mark	Date time	Caller path	Category	Severity	Message
	2025-07-06T22:23:58.468740		_LOG_	_INF_	Logging Server accepted request.
	2025-07-06T22:27:32.675294	Void Main()	COMM	NON	D1 key was pressed.
	2025-07-06T22:27:32.759381		_LOG_	_INF_	Updated custom category: NON,C_APPC_SYSTEM,C_USER,C_UI,C_WFC_DEVICE,C_DEBUG,C_STEPLOG,C_COMM,

(6) 以降、0～2を入力すると前述のカテゴリー設定 API が呼ばれ、それ以外のキーを入力すると入力したキーの文字が Viewer アプリケーションに出力されます。



Mark	Date time	Caller path	Category	Severity	Message
	2025-07-06T22:35:21.489492		_LOG_	_INF_	Logging Server accepted request.
	2025-07-06T22:35:28.067482	Void Main()	NON	WARNING	D1 key was pressed.
	2025-07-06T22:35:28.095540		_LOG_	_INF_	Updated custom category: NON,C_APPC_SYSTEM,C_USER,C_UI,C_WFC_DEVICE,C_DEBUG,C_STEPLOG,C_COMM,
	2025-07-06T22:35:44.479440	Void Main()	C_DEVICE	WARNING	A key was pressed.
	2025-07-06T22:35:44.924466	Void Main()	C_STEPLOG	WARNING	B key was pressed.
	2025-07-06T22:35:45.293340	Void Main()	C_WF	NON	D key was pressed.
	2025-07-06T22:35:46.467455	Void Main()	NON	NON	D1 key was pressed.
	2025-07-06T22:35:46.507611		_LOG_	_INF_	Updated custom category: NON,C_APPC_SYSTEM,C_USER,C_UI,C_WFC_DEVICE,C_DEBUG,C_STEPLOG,C_COMM,

(7) サンプルプログラムを終了するには `Esc` キーを入力した後、任意キーを入力します。



```
C:\Program Files\Logging Foot > 2 : FTCORE_SetCustomCategoryFromFile()

> 1 key was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> Called FTCORE_SetCustomCategory()..
> result: FTCORE_SUCCESS

> ey was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> Called FTCORE_ExitProcess()..
> result: FTCORE_SUCCESS

Press any key again to exit program.
```

Viewer アプリケーションには接続が終了したことの内部ログが出力されます。



2.6.3. Sample_SetWindowState

本サンプルプログラムは、Viewer アプリケーションのウィンドウサイズをクライアントプロセスから制御する API の使用方法を示します。

(1) 事前に Logging Foot Viewer アプリケーション(ftviewer¥ft_viewer.exe)を実行しておきます。

(2) Sample_SetWindowState.exe を実行します。

sample¥Cpp¥Sample_SetWindowState¥bin¥Debug 又は Release フォルダにあります。

(3) exe が起動すると、キー入力待ちになりますので任意のキーを入力します。



(4) 例えば A キーを入力すると、下図の API が呼び出されます。

- FTCORE_StartProcess()

また、制御するウィンドウサイズの選択肢が表示されます。

0 : Default.

1 : Maximized.

2 : Minimize to the taskbar.

3 : Hide in the system tray.

4 : Fully hidden.

```
C:\Program Files\Logging Foot > *****  
  
First, Please start Logging server.  
And next, press any key. (Press Esc key to exit this program.)  
Logging client will start.  
  
> Wait for any key to be pressed ...  
> a  
> a key was pressed.  
> Called FTCORE_StartProcess()..  
> result: FTCORE_SUCCESS  
  
> Press following number key, change window state.  
> 0 : Default.  
> 1 : Maximized.  
> 2 : Minimized to the taskbar.  
> 3 : Hide in the system tray.  
> 4 : Fully hidden.  
  
> |
```



(5) 制御するウィンドウサイズの番号を入力します。

例として 1（最大化）を入力します。

```

C:\Program Files\Logging Foot > a
> a key was pressed.
> Called FTCORE_StartProcess()..
> result: FTCORE_SUCCESS

> Press following number key, change window state.
> 0 : Default.
> 1 : Maximized.
> 2 : Minimized to the taskbar.
> 3 : Hide in the system tray.
> 4 : Fully hidden.

> 1 key was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> Called FTCORE_SetWindowState(1)..
> result: FTCORE_SUCCESS

> |

```

FTCORE_SetWindowState() API が呼ばれ、Viewer アプリケーションは最大化表示されます。

(6) 以降、0～4を入力すると前述の API が呼ばれ、ウィンドウサイズが制御されます。

それ以外のキーを入力すると入力したキーの文字が Viewer アプリケーションに出力されます。

Mark	Date time	Caller path	Category	Severity	Message
	2025-07-06T22:53:16.850260		_LOG_	_INF_	Logging Server accepted request.
	2025-07-06T22:54:10.163272	Void Main()	UI	WARNING	D1 key was pressed.
	2025-07-06T22:54:10.202242		_LOG_	_INF_	Chang window state: SHOW_WINDOW_MAXIMIZE
	2025-07-06T22:54:32.616100	Void Main()	USER	FATAL	F12 key was pressed.
	2025-07-06T22:54:51.266155	Void Main()	STEP	WARNING	Escape key was pressed.
	2025-07-06T22:54:51.293141		_LOG_	_INF_	Logging Server detected disconnection.
	2025-07-06T22:55:07.112337		_LOG_	_INF_	Logging Server accepted request.
	2025-07-06T22:55:11.595343	Void Main()	DEVICE	WARNING	D1 key was pressed.
	2025-07-06T22:55:11.618090		_LOG_	_INF_	Chang window state: SHOW_WINDOW_MAXIMIZE
	2025-07-06T23:00:15.477622	Void Main()	DEBUG	WARNING	D0 key was pressed.
	2025-07-06T23:00:15.520878		_LOG_	_INF_	Chang window state: SHOW_WINDOW_NORMALY
	2025-07-06T23:00:19.792835	Void Main()	COMM	ERROR	A key was pressed.
	2025-07-06T23:00:19.939099	Void Main()	DEBUG	INFO	S key was pressed.
	2025-07-06T23:00:20.199356	Void Main()	SYSTEM	WARNING	D key was pressed.
	2025-07-06T23:00:20.827930	Void Main()	DEVICE	ERROR	F key was pressed.



(7) サンプルプログラムを終了するには Esc キーを入力した後、任意キーを入力します。

```
C:\Program Files\Logging Foot > result: FTCORE_SUCCESS

> d key was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> f key was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> ey was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> Called FTCORE_ExitProcess()..
> result: FTCORE_SUCCESS

Press any key again to exit program.
|
```

Viewer アプリケーションには接続が終了したことの内部ログが出力されます。



2.6.4. Sample_StartupServer

本サンプルプログラムは、クライアントプロセスから Viewer アプリケーションを起動しロギングを開始する API の使用方法を示します。(C++、C#、VB の提供のみ)

(1) Sample_StartupServer.exe を実行します。

sample¥Cpp¥Sample_StartupServer¥bin¥Debug 又は Release フォルダにあります。

※ Logging Foot Viewer アプリケーション(ftviewer¥ft_viewer.exe) が既に実行されている場合は終了させてください。

(2) exe が起動すると下図の画面が表示されます。

Sample_StartupServer

Server Program Startup Settings

Server name: localhost Port number: 50500

Server path: ..¥..¥..¥..¥..¥ftviewer¥ft_viewer.exe

Call API: FTCORE_StartTriggerWithParam

Window state: SHOW_WINDOW_NORMALY

Topmost: STATE_FALSE

Exit Startup

Logging Message Output

Category: NON Random

Severity: NON Random

Message:

Send

Server name :

本プログラムで使用する API はクライアントプロセスと Viewer アプリケーションが同一 PC 内にあ
る場合のみ動作しますので localhost のままにしておきます。



Server path :

Viewer アプリケーション(ft_viewer.exe)への相対パス又は絶対パスを指定します。
相対パスの場合は、本プログラム(Sample_StartupServer.exe)からのパスになります。

Port number

Viewer アプリケーションとの通信ポートを指定します。デフォルト値 50500 を変更する場合は
Viewer アプリケーション側の通信ポート設定も変更が必要になります。

Call API :

呼び出しする API を選択します。

Window state :

Viewer アプリケーション起動時のウィンドウ状態を選択します。

Topmost :

Viewer アプリケーションウィンドウの最前面表示状態を選択します。

Startup :

押下すると Call API で選択した API を呼び出して Viewer アプリケーションを起動し、ロギングを開始します。

Category :

出力するログのカテゴリを選択します。Random にチェックを入れるとランダムにカテゴリが選択されます。

Severity :

出力するログの重要度を選択します。Random にチェックを入れるとランダムに重要度が選択されます。

Message :

出力するログの文字列を入力します。

Send :

押下すると前述までの設定で Viewer アプリケーションにログを送信しログが出力されます。



2.6.5. Sample_UnitTest

本サンプルプログラムは、負荷テストに使用されたものです。ご使用予定の PC でどれくらいの負荷がかかるかの検証に使用できます。(C++、C#、VB の提供のみ)

(1) 事前に Logging Foot Viewer アプリケーション(ftviewer¥ft_viewer.exe)を実行しておきます。

(2) Sample_UnitTest.exe を実行します。
Sample_UnitTest¥bin¥Debug 又は Release フォルダにあります。

(3) exe が起動すると下図の画面が表示されます。

Sample_StartupServer

Server Program Connection Settings

Server name Port number

Logging Message Output

Category ☐ Random

Severity ☐ Random

Message

34 / 4096 bytes

Stress test.

Repetition (65535 maximum)

Repetition Interval (ms)

0% - Stopped



Server name :

Logging Foot Viewer アプリケーション(ftviewer¥ft_viewer.exe)が実行されている PC 名又は IP アドレスを指定します。クライアントプロセスと Viewer アプリケーションが同一 PC 内にある場合は localhost を指定可能です。Viewer アプリケーションがリモート PC で実行されている場合は、その PC の PC 名又は IP アドレスを指定します。

リモート PC 名を指定する場合の例:

The screenshot shows a window titled "Sample_StartupServer" with a "Server Program Connection Settings" section. Inside this section, the "Server name" field contains the text "TestRmtPC01" and the "Port number" field contains "50500". Below these fields are two buttons: "Discon" and "Connect".

IP アドレスを指定する場合の例:

The screenshot shows the same "Sample_StartupServer" window. In this instance, the "Server name" field contains the IP address "192.128.10.1", while the "Port number" field remains "50500". The "Discon" and "Connect" buttons are still present at the bottom.

Port number

Viewer アプリケーションとの通信ポートを指定します。デフォルト値 50500 を変更する場合は Viewer アプリケーション側の通信ポート設定も変更が必要になります。

Connect

Viewer アプリケーションとの接続を確立し、ロギングを開始します。

Discon

Viewer アプリケーションとの接続を解除し、ロギングを終了します。



Category :

出力するログのカテゴリを選択します。Random にチェックを入れるとランダムにカテゴリが選択されます。

Severity :

出力するログの重要度を選択します。Random にチェックを入れるとランダムに重要度が選択されます。

Message :

出力するログの文字列を入力します。

Send :

押下すると Message に入力された文字列を Viewer アプリケーションに送信し、ログが出力されます。

Repetition Count :

負荷テストで送信する回数を指定します。

Repetition Interval :

負荷テストで送信する間隔を指定します。

Start :

負荷テストを開始します。

Repetition Count 及び Repetition Interval で設定された回数、間隔で Message に入力した文字列を自動で繰り返し Viewer アプリケーションに送信しログ出力されます。

Start :

負荷テストを中止します。