



---

# FTSDK API Specification

Ver. 1.1.0 Nov. 2025





## Revision History

Version	Date	Description
1.1.0.1	Nov. 2025	Initial release



## Contents

FTSDK API Specification .....	1
1. Introduction .....	5
1.1. About FTSDK.....	5
1.2. Precautions for Use and Disclaimer .....	6
1.3. Copyright and License Terms .....	6
1.4. System Requirements .....	6
2. FTSDK API .....	7
2.1. C/C++ .....	7
2.1.1. API Specifications .....	7
2.1.2. Enumeration .....	14
2.1.3. Development environment.....	16
2.1.4. How to use FTSDK.....	16
2.1.5. Sample program .....	17
2.2. C#.....	18
2.2.1. API Specifications .....	18
2.2.2. Enumration .....	25
2.2.3. Development environment.....	27
2.2.4. How to use FTSDK.....	27
2.2.5. Sample program .....	28
2.3. VB.....	29
2.3.1. API Specifications .....	29
2.3.2. Enumration .....	36
2.3.3. Development environment.....	38
2.3.4. How to use FTSDK.....	38
2.3.5. Sample program .....	39
2.4. JavaScript .....	40
2.4.1. API Specifications .....	41
2.4.2. Enumration .....	49
2.4.3. Development environment.....	51
2.4.4. How to use.....	54
2.4.5. Sample program .....	58
2.5. Python.....	59



2.5.1. API Specifications .....	59
2.5.2. Enumration .....	66
2.5.3. Development environment.....	68
2.5.4. How to use FTSDK.....	70
2.5.5. Sample program .....	71
2.6. Description of Sample Program Behavior .....	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. Introduction

Thank you very much for downloading Logging Foot (hereafter referred to as 'this software'). Please read this manual thoroughly before using the software.

This document is an English translation of the original Japanese documentation; therefore, some diagrams and tables may still contain Japanese text or use the yen symbol (¥) as a path separator.

### 1.1.About FTSDK

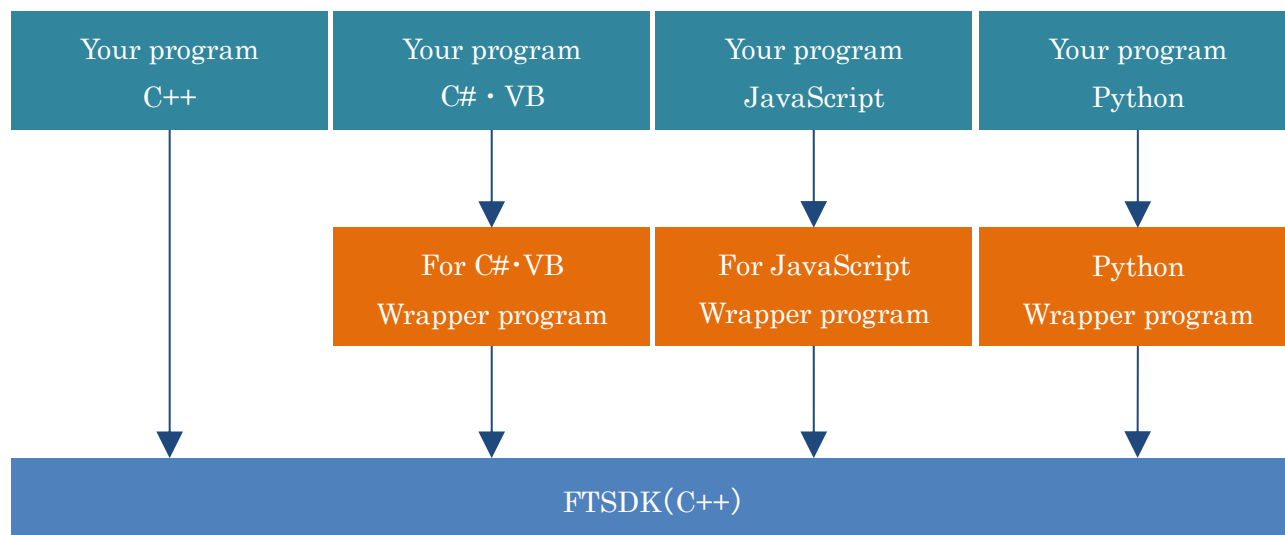
This document describes a series of related software programs, including the Logging Foot Software Development Kit (FTSDK) and Viewer applications.

The FTSDK is a toolkit that includes APIs for incorporating the logging functions of this software into your programs.

The FTSDK API is exported in C language format. If you want to use it with programming languages other than C/C++, you will need to create a wrapper program.

Currently, sample and wrapper programs are available for the following programming languages. We plan to add support for additional languages in the future.

- C++ (Sample program only)
- C#
- VB
- JavaScript
- Python





---

## 1.2. Precautions for Use and Disclaimer

---

- Do not output highly confidential information such as personal or sensitive data using this software.
- Do not use this software in systems that require high reliability, such as medical or financial systems.
- The author shall not be held liable for any damage or loss caused by the use of this software.
- When introducing this software, consider using the free and trial versions provided, and do so at your own risk.
- If any bugs or malfunctions are found, the author will endeavor to improve them but is not obligated to do so.

---

## 1.3. Copyright and License Terms

---

By using this software, you agree to the following terms and conditions:

- The copyright of this document and software belongs to the author, **Quantityworks Software**. You are prohibited from reprinting, reproducing, modifying, distributing, or selling the software or this document without the author's permission.
- You may use the Software solely for the purpose of enabling the functions provided by the Software to operate correctly.
- You may incorporate the software into your program, provided that its functions operate normally, and reproduce and redistribute it with your program.
- You may use this software in your own program, regardless of whether it is for commercial or noncommercial use, under the aforementioned conditions.

---

## 1.4. System Requirements

---

OS : Windows10(64bit)、Windows11(64bit)

CPU : Intel 4h generation or later or compatible CPU (recommended)

RAM : 4GB or more (recommended)

STRAGE: 32MB or more free space



## 2. FTSDK API

This chapter explains the FTSDK API specifications for each programming language. Unless specified otherwise, "server process" refers to the Logging Foot Viewer application process and "client process" refers to your program process that implements this API. The PC that runs your program implementing this API is referred to as the local PC.

For details on how to use the API, please refer to the sample programs.

### 2.1. C/C++

It runs most efficiently because it is the same implementation language as FTSDK.

#### 2.1.1. API Specifications

##### 2.1.1.1. FTCORE\_StartProcess(char\*, unsigned short)

Summary	Start logging.
<b>FTCORE_RESULT</b> <b>_stdcall</b> FTCORE_StartProcess( <b>char*</b> <i>servername</i> , <b>unsigned short</b> <i>portnumber</i> )	
<i>servername</i>	Name or IP address of the PC to connect to (xx.xx.xx.xx)
<i>portnumber</i>	Port number of the connected PC (49152 to 65535)
Return	Success
	<b>FTCORE_SUCCESS</b>
	Failure
	<b>FTCORE_ERR_CLIENT_ALREADY</b> <b>FTCORE_ERR_CLIENT_SOCKET</b> <b>FTCORE_ERR_CLIENT_REFUSED</b> <b>FTCORE_ERR_CLIENT_UNREACHED</b> <b>FTCORE_ERR_CLIENT_CONNECT</b> <b>FTCORE_ERR_CLIENT_ESTABLISH</b>
Description	Establish a connection with the server process running on the target PC and start logging. You can specify either a local or remote PC as the target PC.  The server process on the target PC must be started in advance.  If the target PC is a local PC, you can specify "localhost" for <i>servername</i> .



## 2.1.1.2. FTCore\_StartTriggerWithParam(char\*, char\*, unsigned short, int, int)

Summary		Start the server process with the specified parameters and start logging.
FTCORE_RESULT _stdcall FTCore_StartTriggerWithParam( char* serverpath, char* servername, unsigned short portnumber, int viewstate, int topmost)		
serverpath		Executable file name including full path of the server process.
servername		Name or IP address of the PC to connect to (xx.xx.xx.xx)
portnumber		Port number of the connected PC (49152 to 65535)
viewstate		Window status when starting the server process application: 0 : Default window size 1 : Maximum 2 : Minimize and show on the taskbar 3 : Hide in the task tray 4 : Hidden
topmost		Topmost state of the server process application window: 0 : Disable 1 : Enable
Return	Success	FTCORE_SUCCESS
	Failure	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
Description		Start the server process at the specified path, establish a connection, and start logging. This API only works on the same local PC. Specify the full path of ft_viewer.exe located on the local PC in <i>serverpath</i> . Specify the same PC name or IP address of the local PC in <i>servername</i> . Since it is the same local PC, you can specify "localhost" for <i>servername</i> .



**2.1.1.3. FTCORE\_StartTriggerWithSetup(char\*, char\*, unsigned short)**

Summary		Start the server process with the configured parameters and begin logging.
FTCORE_RESULT _stdcall FTCORE_StartTriggerWithSetup( char* <i>serverpath</i> , char* <i>servername</i> , unsigned short <i>portnumber</i> )		
<i>serverpath</i>		Executable file name including full path of the server process.
<i>servername</i>		Name or IP address of the PC to connect to (xx.xx.xx.xx)
<i>portnumber</i>		Port number of the connected PC (49152 to 65535)
Return	Success	FTCORE_SUCCESS
	Failure	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
Description		<p>The client process launches the server process at the specified path using parameters set by the setup program, establishes a connection, and starts logging. You must run the setup program in advance to set the parameters. (If you do not run the setup program, the default parameters will be used.)</p> <p>This API only works on the same local PC.</p> <p>Specify the full path of ft_viewer.exe located on the local PC in <i>serverpath</i>.</p> <p>Specify the same PC name or IP address of the local PC in <i>servername</i>.</p> <p>Since it is the same local PC, you can specify “localhost” for <i>servername</i>.</p>



## 2.1.1.4. FTCORE\_ExitProcess()

Summary		Exit logging.
FTCORE_RESULT _stdcall FTCORE_ExitProcess()		
-----		No parameters
Return	Success	FTCORE_SUCCESS
	Failure	-----
Description		If logging is started by FTCORE_StartProcess(), then this API ends the logging.

## 2.1.1.5. FTCORE\_ExitTrigger()

Summary		Exit logging.	
FTCORE_RESULT _stdcall FTCORE_ExitTrigger()			
-----		No parameters	
Return	Success	FTCORE_SUCCESS	
	Failure	-----	
Description		If logging is started by FTCORE_StartTriggerWithParam() or FTCORE_StartTriggerWithSetup(), then this API ends the logging.	



## 2.1.1.6. FTCCORE\_SendMessage(char\*, char\*, char\*, char\*)

Summary		Output logs to the server process.
FTCORE_RESULT _stdcall FTCORE_SendMessage(  char* path,  char* category,  char* severity,  char* message)		
path		Any path, such as the caller
category		Log category
severity		Log severity
message		Any log string to be output
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		Outputs the strings specified for each argument to the server process as logs. For <i>category</i> and <i>severity</i> , we recommend specifying the same strings as those shown in the enumeration in <a href="#">2.1.2 Enumeration</a> .

## 2.1.1.7. FTCCORE\_SetWindowState(int)

Summary		Instructs the server process application to set the window state.
FTCORE_RESULT _stdcall FTCORE_SetWindowState(int state)		
state	The server process application window state: 0 : Default window size 1 : Maximum 2 : Minimize and show on the taskbar 3 : Hide in the task tray 4 : Hidden	
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description	Used to control the window state of a server process application from a client process.	



## 2.1.1.8. FTCORE\_LoadDefaultCategory()

Summary		Instructs the server process application to set the default category.
FTCORE_RESULT _stdcall FTCORE_LoadDefaultCategory( )		
-----		No parameters
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		Instructs the server process application to revert to the default category indicated by <u>2.1.2.1 DefaultCategory</u> .

## 2.1.1.9. FTCORE\_SetCustomCategory(char\*)

Summary		Instructs the server process application to set the custom category.
FTCORE_RESULT _stdcall FTCORE_SetCustomCategory(char* categories)		
categories		Any category string (separate multiple strings with commas)
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		<p>Instructs the server process application to set the specified category in categories as the category it recognizes.</p> <p>By using this API, the specified string will be recognized as a category, and the filter function of the server process application will work correctly.</p> <p>If specifying multiple categories, separate them with commas.</p> <p>In the categories recognized by the server process application, the string NON is a reserved word that means “not included in any category.”</p> <p>Regardless of whether NON is included in categories, the server process application inserts NON at the beginning of the category list and operates accordingly.</p>



## 2.1.1.10. FTCORE\_SetCustomCategoryFromFile(char\*)

Summary		Instruct the server process application to set custom categories by reading files.
FTCORE_RESULT _stdcall FTCORE_SetCustomCategoryFromFile(char* filename)		
filename		Full path name of the category settings file to be read
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		<p>Instructs the server process application to set the category read from the file as the recognized category.</p> <p>By using this API, the specified string will be recognized as a category, and the filter function of the server process application will work correctly.</p> <p>The category file can be created by selecting the [Advanced Settings] menu in the server process application and choosing [User definition] as the category source type.</p> <p>For more details, please refer to the Logging Foot Viewer Appcaliton Operation Guide [2.15.Configuring Advanced settings].</p>



## 2.1.2. Enumeration

Regarding `DefaultCategory` and `Severity`, you can pass any string as arguments to the `FTCORE_SendMessage0`; however, when the filter function operateds on the server process application side, these strings are compared against predefined enumerated strings.

To ensure the filter function works correctly in the server process application, you must use the same strings as those defined in the enumeration here.

### 2.1.2.1. DefaultCategory

Indicates the category of the log to be output.

Enum	DefaultCategory	
Member	Value	Example
NON	0	Not included in any category.
APP	1	Application operation
SYSTEM	2	System-related
USER	3	User operation
UI	4	UI triggered event
WF	5	Business logic / Workflow triggered event
DEVICE	6	Device triggered event
DEBUG	7	For debug
STEP	8	Execution step
EVENT	9	Event
COMM	10	Communication-related

### 2.1.2.2. Severity

Indicates the severity of the log to be output.

Enum	Severity	
Member	Value	Example
NONE	0	Not included in any severity.
INFO	1	General information
NOTICE	2	Caution / Attention
WARNING	3	User operational error / Warning
ERR	4	User operational error / Recoverable error
FATAL	5	Unrecoverable error / Fatal error



## 2.1.2.3. FTCCORE\_RESULT

Indicates the types of return values provided by the API.

Enum	FTCCORE_RESULT	
Member	Value	Example
FTCCORE_UNKNOWN_STATE	-1	Excluded / Invalid Value
FTCCORE_SUCCESS	0	Success
FTCCORE_EVT_CLIENT_DETECTDISCON	4	Detected disconnection from server process
FTCCORE_EVT_CLIENT_RECEIVED	5	Received data from server process
FTCCORE_ERR_CLIENT_NOTSERVER	21	No connectable server process exists
FTCCORE_ERR_CLIENT_FAILSERVER	22	Failed to start server process
FTCCORE_ERR_CLIENT_NOEXIST	23	Client process has not been created
FTCCORE_ERR_CLIENT_ALREADY	24	Client process has already been created
FTCCORE_ERR_CLIENT_PARAMETER	25	Transmission parameter error occurred
FTCCORE_ERR_CLIENT_HOSTINFO	26	Failed to obtain destination information
FTCCORE_ERR_CLIENT_SOCKET	27	Socket error occurred
FTCCORE_ERR_CLIENT_REFUSED	28	Connection to server process was refused
FTCCORE_ERR_CLIENT_UNREACHED	29	Network path to server process does not exist
FTCCORE_ERR_CLIENT_CONNECT	30	Failed to connect to server process
FTCCORE_ERR_CLIENT_ESTABLISH	31	Error occurred while establishing connection to server process
FTCCORE_ERR_CLIENT_MESSAGESEND	32	Failed to send message to server process
FTCCORE_ERR_CLIENT_MESSAGE_RECV	33	Failed to receive message from server process
FTCCORE_ERR_CLIENT_CONNUNKNOWN	34	Received a notification accepting a connection from an unknown server.



## 2.1.3. Development environment

---

This software has been verified to operate in the development environments listed below. Please note that using development environments different from those specified may result in build errors. Appropriate measures should be taken accordingly.

IDE	Visual Studio 2022 Community
.NET Framework Environment	-----
Windows SDK Version	10.0 (Latest Installed Version)
Platform Toolset	Visual Studio 2022 (v143)

## 2.1.4. How to use FTSDK

---

In C/C++, both dynamic and static linking methods are supported.

The development environment is assumed to be Microsoft Visual Studio, targeting developers with experience using external libraries in C++.

In the following descriptions, the paths indicated refer to those under the default installation folder (**C:\Program Files\LoggingFoot\**).

### 2.1.4.1. Dynamic link

Include each header file located in the **sample\Cpp\ftsdk\h** folder.

Link the export libraries located in the **sample\Cpp\ftsdk\lib** folder.

Place the DLL files located in the **sample\Cpp\ftsdk\bin** folder into the execution folder of the your program.

### 2.1.4.2. Static link

Include each header file located in the **sample\Cpp\_StaticLib\ftsdk\h** folder.

Link the export libraries located in the **sample\Cpp\_StaticLib\ftsdk\lib** folder.





## 2.1.5. Sample program

The dynamic linking method is included in **sample\Cpp**, while the static linking method is included in **sample\Cpp\_StaticLib**.

For descriptions of the operation of each program, refer to section **2.6 Description of Sample Program Behavior**.

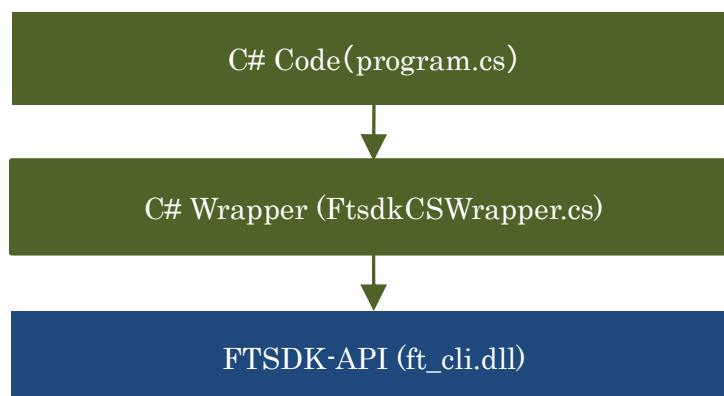
Program name	Description
Sample_StartProcess	This is a sample program that starts logging by launching the client process while the server process application is running. Main APIs Used: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()
Sample_StartupServer	This is a sample program in which the client process launches the server process and starts logging. Main APIs Used: FTCORE_StartTriggerWithParam() FTCORE_StartTriggerWithSetup() FTCORE_SendMessage() FTCORE_ExitTrigger()
Sample_SetCustomCategory	This is a sample program that instructs the server process application to configure custom categories. Main APIs Used: FTCORE_LoadDefaultCategory() FTCORE_SetCustomCategory() FTCORE_SetCustomCategoryFromFile()
Sample_SetWindowState	This is a sample program that controls the window state of the server process application from the client process. Main APIs Used: FTCORE_SetWindowState()
Sample_UnitTest	This is a sample program that performs a stress test by outputting a large volume of logs. Main APIs Used: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()



## 2.2. C#

In C#, a wrapper program is provided, and the FTSDK API is called through the layered structure shown below.

The API specifications described in this section pertain to the wrapper program.



### 2.2.1. API Specifications

#### 2.2.1.1. FTCORE\_StartProcess(string, int)

Summary		Start logging.
<b>FTCORE_RESULT</b> FTCORE_StartProcess( string <i>servername</i> , int <i>portnumber</i> )		
<i>servername</i>		Name or IP address of the PC to connect to (xx.xx.xx.xx)
<i>portnumber</i>		Port number of the connected PC (49152 to 65535)
Return	Success	<b>FTCORE_SUCCESS</b>
	Failure	<b>FTCORE_ERR_CLIENT_ALREADY</b> <b>FTCORE_ERR_CLIENT_SOCKET</b> <b>FTCORE_ERR_CLIENT_REFUSED</b> <b>FTCORE_ERR_CLIENT_UNREACHED</b> <b>FTCORE_ERR_CLIENT_CONNECT</b> <b>FTCORE_ERR_CLIENT_ESTABLISH</b>
Description	Establish a connection with the server process running on the target PC and start logging. You can specify either a local or remote PC as the target PC.  The server process on the target PC must be started in advance.  If the target PC is a local PC, you can specify “localhost” for <i>servername</i> .	



## 2.2.1.2. FTCore\_StartTriggerWithParam(string, string, int, int, bool)

Summary		Start the server process with the specified parameters and start logging.
FTCORE_RESULT FTCore_StartTriggerWithParam( <div>string serverpath, string servername, int portnumber, int viewstate, bool topmost)</div>		
serverpath	Executable file name including full path of the server process.	
servername	Name or IP address of the PC to connect to (xx.xx.xx.xx)	
portnumber	Port number of the connected PC (49152 to 65535)	
viewstate	Window status when starting the server process application: 0 : Default window size 1 : Maximum 2 : Minimize and show on the taskbar 3 : Hide in the task tray 4 : Hidden	
topmost	Topmost state of the server process application window: false : Disable true : Enable	
Return	Success	FTCORE_SUCCESS
	Failure	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
Description		Start the server process at the specified path, establish a connection, and start logging. This API only works on the same local PC. Specify the full path of ft_viewer.exe located on the local PC in serverpath. Specify the same PC name or IP address of the local PC in servername. Since it is the same local PC, you can specify “localhost” for servername.



### 2.2.1.3. FTCCORE\_StartTriggerWithSetup(string, string, int)

Summary		Start the server process with the configured parameters and begin logging.
<b>FTCORE_RESULT</b> <b>FTCORE_StartTriggerWithSetup</b> ( <b>string</b> <i>serverpath</i> , <b>string</b> <i>servername</i> , <b>int</b> <i>portnumber</i> )		
<i>serverpath</i>		Executable file name including full path of the server process.
<i>servername</i>		Name or IP address of the PC to connect to (xx.xx.xx.xx)
<i>portnumber</i>		Port number of the connected PC (49152 to 65535)
Return	Success	<b>FTCORE_SUCCESS</b>
	Failure	<b>FTCORE_ERR_CLIENT_NOTSERVER</b> <b>FTCORE_ERR_CLIENT_FAILSERVER</b> <b>FTCORE_ERR_CLIENT_ALREADY</b> <b>FTCORE_ERR_CLIENT_SOCKET</b> <b>FTCORE_ERR_CLIENT_REFUSED</b> <b>FTCORE_ERR_CLIENT_UNREACHED</b> <b>FTCORE_ERR_CLIENT_CONNECT</b> <b>FTCORE_ERR_CLIENT_ESTABLISH</b>
Description		<p>The client process launches the server process at the specified path using parameters set by the setup program, establishes a connection, and starts logging. You must run the setup program in advance to set the parameters. (If you do not run the setup program, the default parameters will be used.)</p> <p>This API only works on the same local PC.</p> <p>Specify the full path of ft_viewer.exe located on the local PC in <i>serverpath</i>.</p> <p>Specify the same PC name or IP address of the local PC in <i>servername</i>.</p> <p>Since it is the same local PC, you can specify “localhost” for <i>servername</i>.</p>



## 2.2.1.4. FTCORE\_ExitProcess()

Summary		Exit logging.
FTCORE_RESULT FTCORE_ExitProcess( )		
-----		No parameters
Return	Success	FTCORE_SUCCESS
	Failure	-----
Description		If logging is started by FTCORE_StartProcess( ), then this API ends the logging.

## 2.2.1.5. FTCORE\_ExitTrigger()

Summary		Exit logging.
FTCORE_RESULT FTCORE_ExitTrigger()		
-----		No parameters
Return	Success	FTCORE_SUCCESS
	Failure	-----
Description		If logging is started by FTCORE_StartTriggerWithParam() or FTCORE_StartTriggerWithSetup(), then this API ends the logging.



## 2.2.1.6. FTCORE\_SendMessage(string, string, string, string)

Summary		Output logs to the server process.
FTCORE_RESULT FTCORE_SendMessage( <div><i>string path</i>,</div> <div><i>string category</i>,</div> <div><i>string severity</i>,</div> <div><i>string message</i>)</div>		
<i>path</i>		Any path, such as the caller
<i>category</i>		Log category
<i>severity</i>		Log severity
<i>message</i>		Any log string to be output
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		Outputs the strings specified for each argument to the server process as logs. For <i>category</i> and <i>severity</i> , we recommend specifying the same strings as those shown in the enumeration in <a href="#">2.2.2 Enumeration</a> .

## 2.2.1.7. FTCORE\_SetWindowState(int)

Summary	Instructs the server process application to set the window state.	
FTCORE_RESULT FTCORE_SetWindowState(int <i>state</i> )		
<i>state</i>	The server process application window state: 0 : Default window size 1 : Maximum 2 : Minimize and show on the taskbar 3 : Hide in the task tray 4 : Hidden	
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description	Used to control the window state of a server process application from a client process.	



## 2.2.1.8. FTCORE\_LoadDefaultCategory()

Summary		Instructs the server process application to set the default category.
FTCORE_RESULT FTCORE_LoadDefaultCategory()		
-----		No parameters
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		Instructs the server process application to revert to the default category indicated by <u>2.2.2.1 DefaultCategory</u> .

## 2.2.1.9. FTCORE\_SetCustomCategory(string)

Summary		Instructs the server process application to set the custom category.
FTCORE_RESULT FTCORE_SetCustomCategory(string categories)		
categories		Any category string (separate multiple strings with commas)
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		<p>Instructs the server process application to set the specified category in categories as the category it recognizes.</p> <p>By using this API, the specified string will be recognized as a category, and the filter function of the server process application will work correctly.</p> <p>If specifying multiple categories, separate them with commas.</p> <p>In the categories recognized by the server process application, the string NON is a reserved word that means “not included in any category.”</p> <p>Regardless of whether NON is included in categories, the server process application inserts NON at the beginning of the category list and operates accordingly.</p>



## 2.2.1.10. FTCCORE\_SetCustomCategoryFromFile(string)

Summary		Instruct the server process application to set custom categories by reading files.
FTCORE_RESULT FTCORE_SetCustomCategoryFromFile(string <i>filename</i> )		
<i>filename</i>		Full path name of the category settings file to be read
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		<p>Instructs the server process application to set the category read from the file as the recognized category.</p> <p>By using this API, the specified string will be recognized as a category, and the filter function of the server process application will work correctly.</p> <p>The category file can be created by selecting the [Advanced Settings] menu in the server process application and choosing [User definition] as the category source type.</p> <p>For more details, please refer to the Logging Foot Viewer Appcaliton Operation Guide [2.15.Configuring Advanced settings].</p>





## 2.2.2. Enumeration

Regarding `DefaultCategory` and `Severity`, you can pass any string as arguments to the `FTCORE_SendMessage()`; however, when the filter function operateds on the server process application side, these strings are compared against predefined enumerated strings.

To ensure the filter function works correctly in the server process application, you must use the same strings as those defined in the enumeration here.

### 2.2.2.1. DefaultCategory

Indicates the category of the log to be output.

Enum	DefaultCategory	
Member	Value	Example
NON	0	Not included in any category.
APP	1	Application operation
SYSTEM	2	System-related
USER	3	User operation
UI	4	UI triggered event
WF	5	Business logic / Workflow triggered event
DEVICE	6	Device triggered event
DEBUG	7	For debug
STEP	8	Execution step
EVENT	9	Event
COMM	10	Communication-related

### 2.2.2.2. Severity

Indicates the severity of the log to be output.

Enum	Severity	
Member	Value	Member
NONE	0	Not included in any severity.
INFO	1	General information
NOTICE	2	Caution / Attention
WARNING	3	User operational error / Warning
ERR	4	User operational error / Recoverable error
FATAL	5	Unrecoverable error / Fatal error



## 2.2.2.3. FTCCORE\_RESULT

Indicates the types of return values provided by the API.

Enum	FTCCORE_RESULT	
Member	Value	Example
FTCCORE_UNKNOWN_STATE	-1	Excluded / Invalid Value
FTCCORE_SUCCESS	0	Success
FTCCORE_EVT_CLIENT_DETECTDISCON	4	Detected disconnection from server process
FTCCORE_EVT_CLIENT_RECEIVED	5	Received data from server process
FTCCORE_ERR_CLIENT_NOTSERVER	21	No connectable server process exists
FTCCORE_ERR_CLIENT_FAILSERVER	22	Failed to start server process
FTCCORE_ERR_CLIENT_NOEXIST	23	Client process has not been created
FTCCORE_ERR_CLIENT_ALREADY	24	Client process has already been created
FTCCORE_ERR_CLIENT_PARAMETER	25	Transmission parameter error occurred
FTCCORE_ERR_CLIENT_HOSTINFO	26	Failed to obtain destination information
FTCCORE_ERR_CLIENT_SOCKET	27	Socket error occurred
FTCCORE_ERR_CLIENT_REFUSED	28	Connection to server process was refused
FTCCORE_ERR_CLIENT_UNREACHED	29	Network path to server process does not exist
FTCCORE_ERR_CLIENT_CONNECT	30	Failed to connect to server process
FTCCORE_ERR_CLIENT_ESTABLISH	31	Error occurred while establishing connection to server process
FTCCORE_ERR_CLIENT_MESSAGESEND	32	Failed to send message to server process
FTCCORE_ERR_CLIENT_MESSAGERCV	33	Failed to receive message from server process
FTCCORE_ERR_CLIENT_CONNUNKNOWN	34	Received a notification accepting a connection from an unknown server.



## 2.2.3. Development environment

---

This software has been verified to operate in the development environments listed below. Please note that using development environments different from those specified may result in build errors. Appropriate measures should be taken accordingly.

IDE	Visual Studio 2022 Community
.NET Framework Environment	.NET Framework 4.7.2

## 2.2.4. How to use FTSDK

---

In C#, a wrapper program is provided, and it is recommended to use the FTSDK through this wrapper.

The development environment is assumed to be Microsoft Visual Studio, targeting developers with experience in C# Platform Invocation (P/Invoke).

In the following descriptions, the paths indicated refer to those under the default installation folder (C:\Program Files\LoggingFoot\).

Add the wrapper source file FtsdkCSWrapper.cs, located in **sample\C#\ftsdk\src**, to your project.

Place the DLL file located in **sample\C#\ftsdk\bin** into the execution folder of your application.



## 2.2.5. Sample program

The sample program is included in **sample\C#**.

For descriptions of the operation of each program, refer to section **2.6 Description of Sample Program Behavior.**

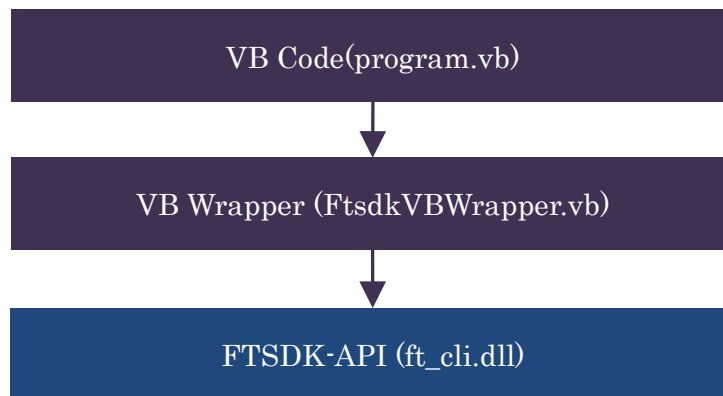
Program name	Description
Sample_StartProcess	This is a sample program that starts logging by launching the client process while the server process application is running. Main APIs Used: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()
Sample_StartupServer	This is a sample program in which the client process launches the server process and starts logging. Main APIs Used: FTCORE_StartTriggerWithParam() FTCORE_StartTriggerWithSetup() FTCORE_SendMessage() FTCORE_ExitTrigger()
Sample_SetCustomCategory	This is a sample program that instructs the server process application to configure custom categories. Main APIs Used: FTCORE_LoadDefaultCategory() FTCORE_SetCustomCategory() FTCORE_SetCustomCategoryFromFile()
Sample_SetWindowState	This is a sample program that controls the window state of the server process application from the client process. Main APIs Used: FTCORE_SetWindowState()
Sample_UnitTest	This is a sample program that performs a stress test by outputting a large volume of logs. Main APIs Used: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()



## 2.3. VB

In VB, a wrapper program is provided, and the FTSDK API is called through the layered structure shown below.

The API specifications described in this section pertain to the wrapper program.



### 2.3.1. API Specifications

#### 2.3.1.1. FTCORE\_StartProcess(String, Integer)

Summary		Start logging.
FTCORE_StartProcess( <i>servername</i> As String, <i>portnumber</i> As Integer) As FTCORE_RESULT		
<i>servername</i>	Name or IP address of the PC to connect to (xx.xx.xx.xx)	
<i>portnumber</i>	Port number of the connected PC (49152 to 65535)	
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
Description		Establish a connection with the server process running on the target PC and start logging. You can specify either a local or remote PC as the target PC.  The server process on the target PC must be started in advance.  If the target PC is a local PC, you can specify “localhost” for <i>servername</i> .



## 2.3.1.2. FTCore\_StartTriggerWithParam(String, String, Integer, Integer, Boolean)

Summary		Start the server process with the specified parameters and start logging.
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 FTFCORE_RESULT		
<i>serverpath</i>	Executable file name including full path of the server process.	
<i>servername</i>	Name or IP address of the PC to connect to (xx.xx.xx.xx)	
<i>portnumber</i>	Port number of the connected PC (49152 to 65535)	
<i>viewstate</i>	Window status when starting the server process application: 0 : Default window size 1 : Maximum 2 : Minimize and show on the taskbar 3 : Hide in the task tray 4 : Hidden	
<i>topmost</i>	Topmost state of the server process application window: False : Disable True : Enable	
Return	Success	FTFCORE_SUCCESS
	Failure	FTFCORE_ERR_CLIENT_NOTSERVER FTFCORE_ERR_CLIENT_FAILSERVER FTFCORE_ERR_CLIENT_ALREADY FTFCORE_ERR_CLIENT_SOCKET FTFCORE_ERR_CLIENT_REFUSED FTFCORE_ERR_CLIENT_UNREACHED FTFCORE_ERR_CLIENT_CONNECT FTFCORE_ERR_CLIENT_ESTABLISH
Description	Start the server process at the specified path, establish a connection, and start logging. This API only works on the same local PC. Specify the full path of ft_viewer.exe located on the local PC in <i>serverpath</i> . Specify the same PC name or IP address of the local PC in <i>servername</i> . Since it is the same local PC, you can specify “localhost” for <i>servername</i> .	



## 2.3.1.3. FTCore\_StartTriggerWithSetup(String, String, Integer)

Summary	Start the server process with the configured parameters and begin logging.
FTCore_StartTriggerWithSetup( <i>serverpath</i> As String, <i>servername</i> As String, <i>portnumber</i> As Integer) As FTCore_RESULT	
<i>serverpath</i>	Executable file name including full path of the server process.
<i>servername</i>	Name or IP address of the PC to connect to (xx.xx.xx.xx)
<i>portnumber</i>	Port number of the connected PC (49152 to 65535)
Return	Success
	FTCore_SUCCESS
	Failure
	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
Description	<p>The client process launches the server process at the specified path using parameters set by the setup program, establishes a connection, and starts logging. You must run the setup program in advance to set the parameters. (If you do not run the setup program, the default parameters will be used.)</p> <p>This API only works on the same local PC.</p> <p>Specify the full path of ft_viewer.exe located on the local PC in <i>serverpath</i>.</p> <p>Specify the same PC name or IP address of the local PC in <i>servername</i>.</p> <p>Since it is the same local PC, you can specify "localhost" for <i>servername</i>.</p>



## 2.3.1.4. FTCORE\_ExitProcess()

Summary		Exit logging.
FTCORE_ExitProcess( ) As FTCORE_RESULT		
-----		No parameters
Return	Success	FTCORE_SUCCESS
	Failure	-----
Description		If logging is started by FTCORE_StartProcess(), then this API ends the logging.

## 2.3.1.5. FTCORE\_ExitTrigger()

Summary		Exit logging.
FTCORE_ExitTrigger() As FTCORE_RESULT		
-----		No parameters
Return	Success	FTCORE_SUCCESS
	Failure	-----
Description		If logging is started by FTCORE_StartTriggerWithParam() or FTCORE_StartTriggerWithSetup(), then this API ends the logging.





## 2.3.1.6. FTCORE\_SendMessage(String, String, String, String)

Summary		Output logs to the server process.
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>	Any path, such as the caller	
<i>category</i>	Log category	
<i>severity</i>	Log severity	
<i>message</i>	Any log string to be output	
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description	Outputs the strings specified for each argument to the server process as logs. For <i>category</i> and <i>severity</i> , we recommend specifying the same strings as those shown in the enumeration in <a href="#">2.3.2 Enumeration</a> .	

## 2.3.1.7. FTCORE\_SetWindowState(Integer)

Summary		Instructs the server process application to set the window state.
FTCORE_SetWindowState( <i>state</i> As Integer) As FTCORE_RESULT		
<i>state</i>	The server process application window state: 0 : Default window size 1 : Maximum 2 : Minimize and show on the taskbar 3 : Hide in the task tray 4 : Hidden	
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		Used to control the window state of a server process application from a client process.

**2.3.1.8. FTCORE\_LoadDefaultCategory()**

Summary	Instructs the server process application to set the default category.
FTCORE_LoadDefaultCategory( ) As FTCORE_RESULT	
-----	No parameters
Return      Success	FTCORE_SUCCESS
Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description	Instructs the server process application to revert to the default category indicated by <u>2.3.2.1 DefaultCategory</u> .

**2.3.1.9. FTCORE\_SetCustomCategory(String)**

Summary	Instructs the server process application to set the custom category.
FTCORE_SetCustomCategory( <i>categories</i> As String) As FTCORE_RESULT	
<i>categories</i>	Any category string (separate multiple strings with commas)
Return      Success	FTCORE_SUCCESS
Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description	<p>Instructs the server process application to set the specified category in categories as the category it recognizes.</p> <p>By using this API, the specified string will be recognized as a category, and the filter function of the server process application will work correctly.</p> <p>If specifying multiple categories, separate them with commas.</p> <p>In the categories recognized by the server process application, the string NON is a reserved word that means “not included in any category.”</p> <p>Regardless of whether NON is included in categories, the server process application inserts NON at the beginning of the category list and operates accordingly.</p>



## 2.3.1.10. FTCORE\_SetCustomCategoryFromFile(String)

Summary		Instruct the server process application to set custom categories by reading files.
FTCORE_SetCustomCategoryFromFile( <i>filename</i> As String) As FTCORE_RESULT		
<i>filename</i>		Full path name of the category settings file to be read
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		<p>Instructs the server process application to set the category read from the file as the recognized category.</p> <p>By using this API, the specified string will be recognized as a category, and the filter function of the server process application will work correctly.</p> <p>The category file can be created by selecting the [Advanced Settings] menu in the server process application and choosing [User definition] as the category source type.</p> <p>For more details, please refer to the Logging Foot Viewer Appcaliton Operation Guide [2.15.Configuring Advanced settings].</p>



## 2.3.2. Enumration

Regarding `DefaultCategory` and `Severity`, you can pass any string as arguments to the `FTCORE_SendMessage()`; however, when the filter function operateds on the server process application side, these strings are compared against predefined enumerated strings.

To ensure the filter function works correctly in the server process application, you must use the same strings as those defined in the enumeration here.

### 2.3.2.1. DefaultCategory

Indicates the category of the log to be output.

Enum	DefaultCategory	
Member	Value	Example
NON	0	Not included in any category.
APP	1	Application operation
SYSTEM	2	System-related
USER	3	User operation
UI	4	UI triggered event
WF	5	Business logic / Workflow triggered event
DEVICE	6	Device triggered event
DEBUG	7	For debug
STEP	8	Execution step
EVENT	9	Event
COMM	10	Communication-related

### 2.3.2.2. Severity

Indicates the severity of the log to be output.

Enum	Severity	
Member	Value	Example
NONE	0	Not included in any severity.
INFO	1	General information
NOTICE	2	Caution / Attention
WARNING	3	User operational error / Warning
ERR	4	User operational error / Recoverable error
FATAL	5	Unrecoverable error / Fatal error



### 2.3.2.3. FTCCORE\_RESULT

Indicates the types of return values provided by the API.

Enum	FTCCORE_RESULT	
Member	Value	Example
FTCCORE_UNKNOWN_STATE	-1	Excluded / Invalid Value
FTCCORE_SUCCESS	0	Success
FTCCORE_EVT_CLIENT_DETECTDISCON	4	Detected disconnection from server process
FTCCORE_EVT_CLIENT_RECEIVED	5	Received data from server process
FTCCORE_ERR_CLIENT_NOTSERVER	21	No connectable server process exists
FTCCORE_ERR_CLIENT_FAILSERVER	22	Failed to start server process
FTCCORE_ERR_CLIENT_NOEXIST	23	Client process has not been created
FTCCORE_ERR_CLIENT_ALREADY	24	Client process has already been created
FTCCORE_ERR_CLIENT_PARAMETER	25	Transmission parameter error occurred
FTCCORE_ERR_CLIENT_HOSTINFO	26	Failed to obtain destination information
FTCCORE_ERR_CLIENT_SOCKET	27	Socket error occurred
FTCCORE_ERR_CLIENT_REFUSED	28	Connection to server process was refused
FTCCORE_ERR_CLIENT_UNREACHED	29	Network path to server process does not exist
FTCCORE_ERR_CLIENT_CONNECT	30	Failed to connect to server process
FTCCORE_ERR_CLIENT_ESTABLISH	31	Error occurred while establishing connection to server process
FTCCORE_ERR_CLIENT_MESSAGESEND	32	Failed to send message to server process
FTCCORE_ERR_CLIENT_MESSAGERCV	33	Failed to receive message from server process
FTCCORE_ERR_CLIENT_CONNUNKNOWN	34	Received a notification accepting a connection from an unknown server.



## 2.3.3. Development environment

---

This software has been verified to operate in the development environments listed below. Please note that using development environments different from those specified may result in build errors. Appropriate measures should be taken accordingly.

IDE	Visual Studio 2022 Community
.NET Framework Environment	.NET Framework 4.7.2
Windows SDK Version	-----
Platform Toolset	-----

## 2.3.4. How to use FTSDK

---

In VB, a wrapper program is provided, and it is recommended to use the FTSDK through this wrapper.

The development environment is assumed to be Microsoft Visual Studio, targeting developers with experience in VB Platform Invocation (P/Invoke).

In the following descriptions, the paths indicated refer to those under the default installation folder (**C:\Program Files\LoggingFoot\**).

Add the wrapper source file `FtsdkVBWrapper.cs`, located in **sample\VB\ftsdk\src**, to your project.

Place the DLL file located in **sample\VB\ftsdk\bin** into the execution folder of your application.



## 2.3.5. Sample program

The sample program is included in **sample\VB**.

For descriptions of the operation of each program, refer to section **2.6 Description of Sample Program Behavior**.

Program name	Description
Sample_StartProcess	This is a sample program that starts logging by launching the client process while the server process application is running. Main APIs Used: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()
Sample_StartupServer	This is a sample program in which the client process launches the server process and starts logging. Main APIs Used: FTCORE_StartTriggerWithParam() FTCORE_StartTriggerWithSetup() FTCORE_SendMessage() FTCORE_ExitTrigger()
Sample_SetCustomCategory	This is a sample program that instructs the server process application to configure custom categories. Main APIs Used: FTCORE_LoadDefaultCategory() FTCORE_SetCustomCategory() FTCORE_SetCustomCategoryFromFile()
Sample_SetWindowState	This is a sample program that controls the window state of the server process application from the client process. Main APIs Used: FTCORE_SetWindowState()
Sample_UnitTest	This is a sample program that performs a stress test by outputting a large volume of logs. Main APIs Used: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()



## 2.4. JavaScript

In JavaScript, implementation support is provided for backend programs using Node.js. (Frontend implementation is not supported.)

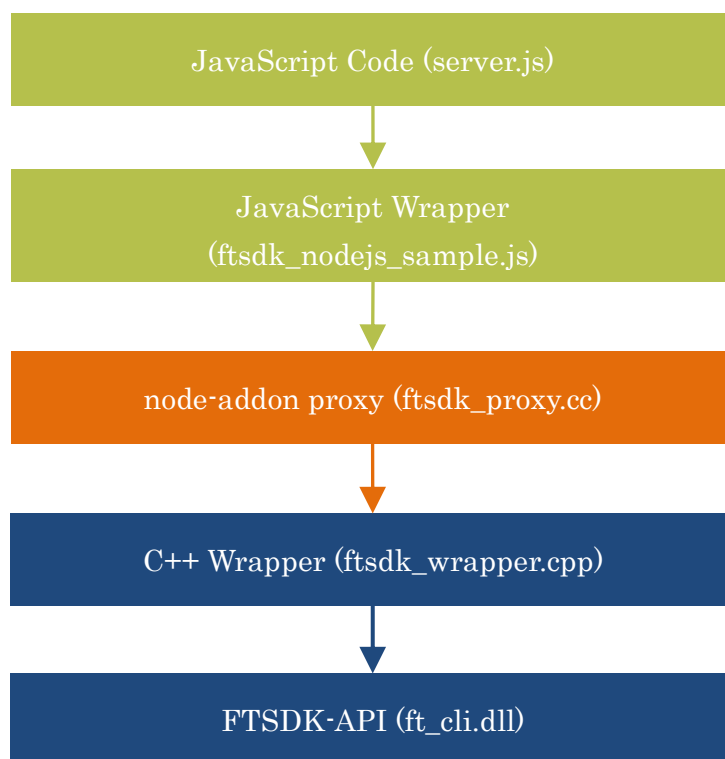
This sample program creates a wrapper using node-addon-api.

To call the FTSDK API from JavaScript code, the call passes through the layered structure shown below.

The names in parentheses indicate the corresponding source files or module names used in this sample program.

While you may implement your application starting from any layer, we recommend using the JavaScript wrapper layer (ftsdk\_nodejs\_sample.js) to avoid the complexity of the node-addon portion.

The API specifications described here pertain to the APIs exposed by the JavaScript wrapper.







## 2.4.1. API Specifications

### 2.4.1.1. ftcoreStartProcess(Any, Number)

概要	Start logging.	
ftcoreStartProcess( <i>servername</i> : Any, <i>portnumber</i> : Number) : Any		
<i>servername</i>	Name or IP address of the PC to connect to (xx.xx.xx.xx)	
<i>portnumber</i>	Port number of the connected PC (49152 to 65535)	
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
Description	Establish a connection with the server process running on the target PC and start logging. You can specify either a local or remote PC as the target PC.  The server process on the target PC must be started in advance.  If the target PC is a local PC, you can specify “localhost” for <i>servername</i> .	

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

Summary		Start the server process with the specified parameters and start logging.
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>		Executable file name including full path of the server process.
<i>servername</i>		Name or IP address of the PC to connect to (xx.xx.xx.xx)
<i>portnumber</i>		Port number of the connected PC (49152 to 65535)
<i>viewstate</i>		Window status when starting the server process application: 0 : Default window size 1 : Maximum 2 : Minimize and show on the taskbar 3 : Hide in the task tray 4 : Hidden
<i>topmost</i>		Topmost state of the server process application window: false : Disable true : Enable
Return	Success	FTCORE_SUCCESS
	Failure	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
Description		Start the server process at the specified path, establish a connection, and start logging. This API only works on the same local PC. Specify the full path of ft_viewer.exe located on the local PC in <i>serverpath</i> . Specify the same PC name or IP address of the local PC in <i>servername</i> . Since it is the same local PC, you can specify "localhost" for <i>servername</i> .

**2.4.1.3. ftcoreStartTriggerWithSetup(Any, Any, Number)**

Summary		Start the server process with the configured parameters and begin logging.
ftcoreStartTriggerWithSetup( <i>serverpath</i> : Any, <i>servername</i> : Any, <i>portnumber</i> : Number) : Any		
<i>serverpath</i>		Executable file name including full path of the server process.
<i>servername</i>		Name or IP address of the PC to connect to (xx.xx.xx.xx)
<i>portnumber</i>		Port number of the connected PC (49152 to 65535)
Return	Success	FTCORE_SUCCESS
	Failure	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
Description		<p>The client process launches the server process at the specified path using parameters set by the setup program, establishes a connection, and starts logging. You must run the setup program in advance to set the parameters. (If you do not run the setup program, the default parameters will be used.)</p> <p>This API only works on the same local PC.</p> <p>Specify the full path of ft_viewer.exe located on the local PC in <i>serverpath</i>.</p> <p>Specify the same PC name or IP address of the local PC in <i>servername</i>.</p> <p>Since it is the same local PC, you can specify “localhost” for <i>servername</i>.</p>



## 2.4.1.4. ftcoreExitProcess()

Summary		Exit logging.	
ftcoreExitProcess() : <a href="#">Any</a>			
-----		No parameters	
Return	Success	FTCORE_SUCCESS	
	Failure	-----	
Description		If logging is started by ftcoreStartProcess( ), then this API ends the logging.	

## 2.4.1.5. ftcoreExitTrigger()

Summary		Exit logging.
ftcoreExitTrigger() : <a href="#">Any</a>		
-----		No parameters
Return	Success	FTCORE_SUCCESS
	Failure	-----
Description		If logging is started by ftcoreStartTriggerWithParam() or ftcoreStartTriggerWithSetup(), then this API ends the logging.



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

Summary	Output logs to the server process.
ftcoreSendMessage( <i>path</i> : Any, <i>category</i> : Any, <i>severity</i> : Any, <i>message</i> : Number[] ) : Any	
<i>path</i>	Any path, such as the caller
<i>category</i>	Log category
<i>severity</i>	Log severity
<i>message</i>	A <b>byte array</b> containing the <b>UTF-8-encoded</b> log string to be output.
Return	Success
	FTCORE_SUCCESS
	Failure
	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description	<p>Outputs the strings specified for each argument to the server process as logs. For <i>category</i> and <i>severity</i>, we recommend specifying the same strings as those shown in the enumeration in <a href="#">2.4.2 Enumeration</a>.</p> <p>In addition, to prevent character corruption, the string received from the upper layer is encoded in UTF-8 and passed to the lower-level API as a Number[] array.</p> <p>For an implementation example, refer to the sample code (ftsdk_nodejs_sample.js).</p>



## 2.4.1.7. ftcoreSetWindowState(Number)

Summary		Instructs the server process application to set the window state.
ftcoreSetWindowState( <i>state</i> : <a href="#">Number</a> ) : <a href="#">Any</a>		
<i>state</i>	The server process application window state: 0 : Default window size 1 : Maximum 2 : Minimize and show on the taskbar 3 : Hide in the task tray 4 : Hidden	
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description	Used to control the window state of a server process application from a client process.	

## 2.4.1.8. ftcoreLoadDefaultCategory()

Summary		Instructs the server process application to set the default category.	
ftcoreLoadDefaultCategory() : <i>Any</i>			
-----		No parameters	
Return	Success	FTCORE_SUCCESS	
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND	
Description		Instructs the server process application to revert to the default category indicated by <u>2.4.2.1 DefaultCategory</u> .	



## 2.4.1.9. ftcoreSetCustomCategory(Any)

Summary		Instructs the server process application to set the custom category.
		<code>ftcoreSetCustomCategory(categories : Any) : Any</code>
<i>categories</i>		Any category string (separate multiple strings with commas)
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		<p>Instructs the server process application to set the specified category in categories as the category it recognizes.</p> <p>By using this API, the specified string will be recognized as a category, and the filter function of the server process application will work correctly.</p> <p>If specifying multiple categories, separate them with commas.</p> <p>In the categories recognized by the server process application, the string NON is a reserved word that means “not included in any category.”</p> <p>Regardless of whether NON is included in categories, the server process application inserts NON at the beginning of the category list and operates accordingly.</p>



## 2.4.1.10. ftcoreSetCustomCategoryFromFile(Any)

Summary		Instruct the server process application to set custom categories by reading files.	
ftcoreSetCustomCategoryFromFile( <i>filename</i> : Any) : Any			
<i>filename</i>		Full path name of the category settings file to be read	
Return	Success	FTCORE_SUCCESS	
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND	
Description		<p>Instructs the server process application to set the category read from the file as the recognized category.</p> <p>By using this API, the specified string will be recognized as a category, and the filter function of the server process application will work correctly.</p> <p>The category file can be created by selecting the [Advanced Settings] menu in the server process application and choosing [User definition] as the category source type.</p> <p>For more details, please refer to the Logging Foot Viewer Appcaliton Operation Guide [2.15.Configuring Advanced settings].</p>	





## 2.4.2. Enumeration

Regarding `DefaultCategory` and `Severity`, you can pass any string as arguments to the `ftcoreSendMessage()`; however, when the filter function operateds on the server process application side, these strings are compared against predefined enumerated strings.

To ensure the filter function works correctly in the server process application, you must use the same strings as those defined in the enumeration here.

The implementation example can be found in the front-end `program.js`.

### 2.4.2.1. DefaultCategory

Indicates the category of the log to be output.

Enum	DefaultCategory	
Member	Value	Example
NON	0	Not included in any category.
APP	1	Application operation
SYSTEM	2	System-related
USER	3	User operation
UI	4	UI triggered event
WF	5	Business logic / Workflow triggered event
DEVICE	6	Device triggered event
DEBUG	7	For debug
STEP	8	Execution step
EVENT	9	Event
COMM	10	Communication-related

### 2.4.2.2. Severity

Indicates the severity of the log to be output.

Enum	Severity	
Member	Value	Example
NONE	0	Not included in any severity.
INFO	1	General information
NOTICE	2	Caution / Attention
WARNING	3	User operational error / Warning
ERR	4	User operational error / Recoverable error
FATAL	5	Unrecoverable error / Fatal error



## 2.4.2.3. FTCore\_RESULT

Indicates the types of return values provided by the API.

An implementation example can be found in **server.js** of **node.js**.

Enum	FTCore_RESULT	
Member	Value	Example
FTCore_UNKNOWN_STATE	-1	Excluded / Invalid Value
FTCore_SUCCESS	0	Success
FTCore_EVT_CLIENT_DETECTDISCON	4	Detected disconnection from server process
FTCore_EVT_CLIENT_RECEIVED	5	Received data from server process
FTCore_ERR_CLIENT_NOTSERVER	21	No connectable server process exists
FTCore_ERR_CLIENT_FAILSERVER	22	Failed to start server process
FTCore_ERR_CLIENT_NOEXIST	23	Client process has not been created
FTCore_ERR_CLIENT_ALREADY	24	Client process has already been created
FTCore_ERR_CLIENT_PARAMETER	25	Transmission parameter error occurred
FTCore_ERR_CLIENT_HOSTINFO	26	Failed to obtain destination information
FTCore_ERR_CLIENT_SOCKET	27	Socket error occurred
FTCore_ERR_CLIENT_REFUSED	28	Connection to server process was refused
FTCore_ERR_CLIENT_UNREACHED	29	Network path to server process does not exist
FTCore_ERR_CLIENT_CONNECT	30	Failed to connect to server process
FTCore_ERR_CLIENT_ESTABLISH	31	Error occurred while establishing connection to server process
FTCore_ERR_CLIENT_MESSAGESEND	32	Failed to send message to server process
FTCore_ERR_CLIENT_MESSAGERCV	33	Failed to receive message from server process
FTCore_ERR_CLIENT_CONNUNKNOWN	34	Received a notification accepting a connection from an unknown server.



## 2.4.3. Development environment

This program has been tested and verified to operate in the following development environment.

IDE	Visual Studio Code
Framework	Node.js (Used in Back-End program)
Used Add-ons	node-addon-api node-gyp bindings
Dependencies	Visual Studio Build Tool 2022 Community Python 3.13 Live Server

In this chapter, we will use the sample program installed at the following location as an example:

**C:\Program Files\Logging Foot\sample\JavaScript\Sample\_StartProcess**

Unless otherwise noted, all paths mentioned in the following sections refer to subdirectories under the path above.

### 2.4.3.1. Environment Setup

The following URL provides a useful reference for setting up the development environment:

Node.js Guidelines by Microsoft:

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

node.js Add-ons document:

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

#### **Setup Procedure (Reference)**

(1) Install Node.js

<https://nodejs.org/en>

(2) Install Visual Studio Build Tool

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

(3) Install Python 3.13

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



#### (4) Changing the PowerShell Execution Policy

Launch PowerShell and enter the following command.

This allows the execution of `npm.ps1`, thereby preventing the `npm` command from being blocked.

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

#### (5) Launch Visual Studio Code

Start Visual Studio Code and open the **node-js** folder.

#### (6) Install Live Server

From the Extensions view in Visual Studio Code, search for "**Live Server**" and install it.

Subsequent commands will be executed in the Visual Studio Code terminal.

#### (7) Initialize the Project

```
npm init -y
```

**Note:** If `npm` is not recognized even after installing Node.js, try restarting Visual Studio Code.

#### (8) Install C++ Add-on Build Tools

```
npm install node-gyp
```

#### (9) Utility for Loading .node Files

```
npm install bindings
```

#### (10) C++ Wrapper for N-API

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



## 2.4.3.2. Building the Wrapper Program

To build the C++ wrapper program (ftsdk\_wrapper.cpp) and the proxy program (ftsdk\_proxy.cc), follow the steps below:

- (1) Run the following command in the Visual Studio Code terminal:

```
npm install .
```

If the build succeeds, a **node-js\build\Release** folder will be generated.

- (2) Copy the client module to the runtime environment:

Place the following file into the **node-js\build\Release** directory:

**C:\Program Files\Logging Foot\ftsdk\core\bin\ft\_cli.dll**



## 2.4.4. How to use

This sample program assumes developers with prior experience using Visual Studio Code and Node.js; therefore, detailed operational instructions for these tools are omitted.

In the following descriptions, the paths indicated refer to those under the default installation folder (C:\Program Files\LoggingFoot\).

Add the **sample\JavaScript\Sample\_StartProcess\node-js** or **sample\JavaScript\Sample\_StartupServer\node-js** to your program's project folder.

For details on calling the FTSDK API, please refer to the sample program `ftsdk_nodejs_sample.js`.

### 2.4.4.1. Debugging the Sample Program Sample\_StartProcess

Explanation of the sample program **sample\JavaScript\Sample\_StartProcess**:

Beforehand, start the Viewer application (`ftviewer\ft_viewer.exe`).

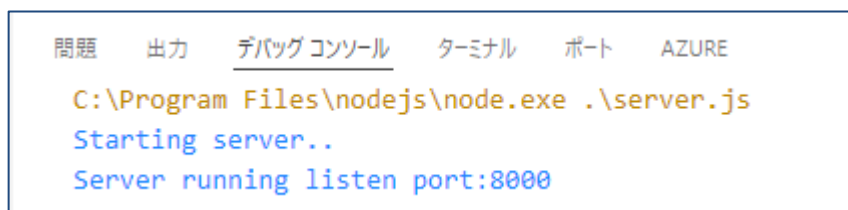
#### Step 1. Run the server-side (Node.js).

(1) Open the following folder in Visual Studio Code.

**sample\JavaScript\Sample\_StartProcess\node-js**

(2) From the Run and Debug dropdown, select "**Startup program**" and start execution.

(3) Upon successful execution, the following message will be displayed in the Debug Console as shown below.



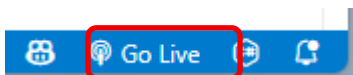
```
問題 出力 デバッグ コンソール ターミナル ポート AZURE
C:\Program Files\nodejs\node.exe .\server.js
Starting server..
Server running listen port:8000
```

#### Step 2. Run the frontend-side

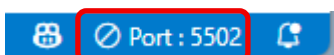
(1) Open the following folder in Visual Studio Code.

**sample\JavaScript\Sample\_StartProcess\front-end**

(2) Click "**Go Live**" at the bottom right of the Visual Studio Code window to start the live server.



(3) Upon successful startup, the port number in use will be displayed as shown below, and the browser UI will launch.





**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:

If the server does not start, please verify that the port number used by the Live Server matches the port number specified in launch.json (in the example below, <http://localhost:5502>).

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

### Step 3. Operate the browser UI and verify that output is sent to the Viewer application.

- (1) Press the **"Connect"** button on the browser UI.
- (2) The message **"Logging Server accepted request."** will appear in the Viewer application.  
If it does not appear, verify that **"Enable logging view display"** is checked in the **Internal Event Log Settings** tab under **General Settings** in the Viewer application configuration.
- (3) Enter any text into the **"Message"** textbox on the browser UI and press the **"Send"** button.  
The entered text will be displayed in the Viewer application.



## 2.4.4.2. Debugging the Sample Program Sample\_StartupServer

Explanation of the sample program `sample\JavaScript\Sample_StartupServer`.

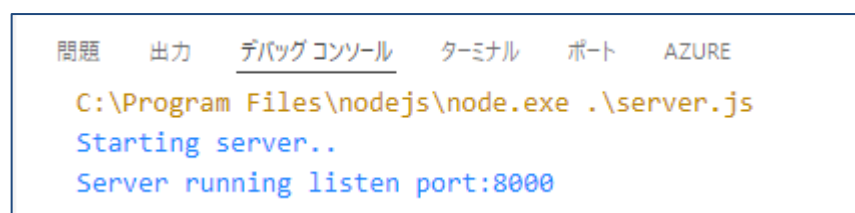
### Step 1. Run the server-side (Node.js).

(1) Open the following folder in Visual Studio Code.

**sample\JavaScript\Sample\_StartupServer\node-js**

(2) From the Run and Debug dropdown, select "Startup program" and start execution.

(3) Upon successful execution, the following message will be displayed in the Debug Console as shown below.



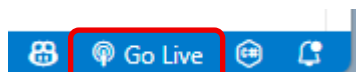
```
問題 出力 デバッグ コンソール ターミナル ポート AZURE
C:\Program Files\nodejs\node.exe .\server.js
Starting server..
Server running listen port:8000
```

### Step 2. Run the frontend-side

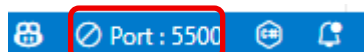
(1) Open the following folder in Visual Studio Code.

**sample\JavaScript\Sample\_StartupServer\front-end**

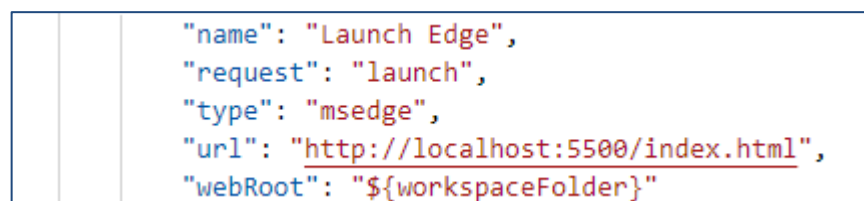
(2) Click "Go Live" at the bottom right of the Visual Studio Code window to start the live server.



(3) Upon successful startup, the port number in use will be displayed as shown below, and the browser UI will launch.



If the server does not start, please verify that the port number used by the Live Server matches the port number specified in launch.json (in the example below, `http://localhost:5500`).



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





### Step 3. Operate the browser UI and verify that output is sent to the Viewer application.

(1) In the browser UI's **"Server path"** field, enter either the absolute path to the Viewer application or a relative path based on the following file, then press the **"Startup"** button:

**sample\JavaScript\Sample\_StartupServer\front-end\index.html**

(2) The Viewer application will launch and display **"Logging Server accepted request."**

If this message does not appear, confirm that **"Enable logging view display"** is checked under **Internal Event Log Settings** in the **Configuration** tab of **General Settings** in the Viewer application.

(3) Enter any text into the browser UI's **"Message"** textbox and press the **"Send"** button.

The entered text will be displayed in the Viewer application.

(4) To exit the Viewer application, press the **"Exit"** button on the browser UI.



## 2.4.5. Sample program

---

Sample programs are included in **sample\JavaScript**.

This sample demonstrates retrieving strings input on the front-end by the back-end and outputting them to the Viewer application.

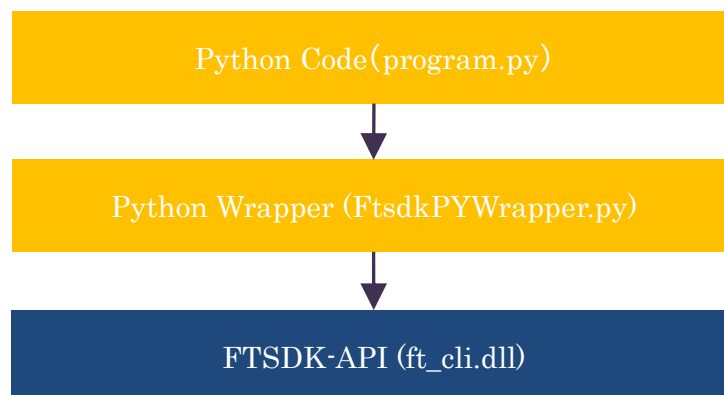
Program name	Description
Sample_StartProcess	<p>This sample program demonstrates starting the client process and logging while the server process application is running.</p> <p>Main APIs used:</p> <ul style="list-style-type: none"><li>ftcoreStartProcess()</li><li>ftcoreSendMessage()</li><li>ftcoreExitProcess()</li><li>ftcoreSetWindowState()</li><li>ftcoreSetCustomCategory()</li><li>ftcoreLoadDefaultCategory()</li></ul>
Sample_StartupServer	<p>This sample program demonstrates the client process launching the server process and starting logging.</p> <p>Main APIs used:</p> <ul style="list-style-type: none"><li>ftcoreStartTriggerWithParam()</li><li>ftcoreStartTriggerWithSetup()</li><li>ftcoreSendMessage()</li><li>ftcoreExitTrigger()</li></ul>



## 2.5. Python

In Python, a wrapper program is provided, and the FTSDK API is called through the layered structure shown below.

The API specifications described in this section pertain to the wrapper program.



### 2.5.1. API Specifications

#### 2.5.1.1. FTCORE\_StartProcess(str, int)

Summary		Start logging.
FTCORE_StartProcess( <i>servername</i> : <b>str</b> , <i>portnumber</i> : <b>int</b> ) -> FTCORE_RESULT		
<i>servername</i>	Name or IP address of the PC to connect to (xx.xx.xx.xx)	
<i>portnumber</i>	Port number of the connected PC (49152 to 65535)	
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_ALREADY FTCORE_ERR_CLIENT_SOCKET FTCORE_ERR_CLIENT_REFUSED FTCORE_ERR_CLIENT_UNREACHED FTCORE_ERR_CLIENT_CONNECT FTCORE_ERR_CLIENT_ESTABLISH
Description		<p>Establish a connection with the server process running on the target PC and start logging. You can specify either a local or remote PC as the target PC.</p> <p>The server process on the target PC must be started in advance.</p> <p>If the target PC is a local PC, you can specify “localhost” for <i>servername</i>.</p>



## 2.5.1.2. FTCore\_StartTriggerWithParam(str, str, int, int, int)

Summary		Start the server process with the specified parameters and start logging.
FTCORE_StartTriggerWithParam( <i>serverpath</i> : <i>str</i> ,  <i>servername</i> : <i>str</i> ,  <i>portnumber</i> : <i>int</i> ,  <i>viewstate</i> : <i>int</i> ,  <i>topmost</i> : <i>int</i> ) -> FTCORE_RESULT		
<i>serverpath</i>	Executable file name including full path of the server process.	
<i>servername</i>	Name or IP address of the PC to connect to (xx.xx.xx.xx)	
<i>portnumber</i>	Port number of the connected PC (49152 to 65535)	
<i>viewstate</i>	Window status when starting the server process application: 0 : Default window size 1 : Maximum 2 : Minimize and show on the taskbar 3 : Hide in the task tray 4 : Hidden	
<i>topmost</i>	Topmost state of the server process application window: False : Disable True : Enable	
Return	Success	FTCORE_SUCCESS
	Failure	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
Description	Start the server process at the specified path, establish a connection, and start logging. This API only works on the same local PC. Specify the full path of ft_viewer.exe located on the local PC in <i>serverpath</i> . Specify the same PC name or IP address of the local PC in <i>servername</i> . Since it is the same local PC, you can specify “localhost” for <i>servername</i> .	



## 2.5.1.3. FTCORE\_StartTriggerWithSetup(str, str, int)

Summary		Start the server process with the configured parameters and begin logging.
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>		Executable file name including full path of the server process.
<i>servername</i>		Name or IP address of the PC to connect to (xx.xx.xx.xx)
<i>portnumber</i>		Port number of the connected PC (49152 to 65535)
Return	Success	FTCORE_SUCCESS
	Failure	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
Description		<p>The client process launches the server process at the specified path using parameters set by the setup program, establishes a connection, and starts logging. You must run the setup program in advance to set the parameters. (If you do not run the setup program, the default parameters will be used.)</p> <p>This API only works on the same local PC.</p> <p>Specify the full path of ft_viewer.exe located on the local PC in <i>serverpath</i>.</p> <p>Specify the same PC name or IP address of the local PC in <i>servername</i>. Since it is the same local PC, you can specify “localhost” for <i>servername</i>.</p>



## 2.5.1.4. FTCORE\_ExitProcess()

Summary		Exit logging.
FTCORE_ExitProcess() -> <b>FTCORE_RESULT</b>		
-----		No parameters
Return	Success	<b>FTCORE_SUCCESS</b>
	Failure	-----
Description		If logging is started by FTCORE_StartProcess(), then this API ends the logging.

## 2.5.1.5. FTCORE\_ExitTrigger()

Summary		Exit logging.
FTCORE_ExitTrigger() -> FTCORE_RESULT		
-----		No parameters
Return	Success	FTCORE_SUCCESS
	Failure	-----
Description		If logging is started by FTCORE_StartTriggerWithParam() or FTCORE_StartTriggerWithSetup(), then this API ends the logging.

**2.5.1.6. FTCore\_SendMessage(str, str, str, str)**

Summary	Output logs to the server process.
FTCore_SendMessage( <i>path</i> : str, <i>category</i> : str, <i>severity</i> : str, <i>message</i> : str) -> FTCore_RESULT	
<i>path</i>	Any path, such as the caller
<i>category</i>	Log category
<i>severity</i>	Log severity
<i>message</i>	Any log string to be output
Return	Success
	FTCore_SUCCESS
	Failure
	FTCore_ERR_CLIENT_NOEXIST FTCore_ERR_CLIENT_PARAMETER FTCore_ERR_CLIENT_MESSAGESEND
Description	Outputs the strings specified for each argument to the server process as logs. For <i>category</i> and <i>severity</i> , we recommend specifying the same strings as those shown in the enumeration in <a href="#">2.3.2 Enumeration</a> .

**2.5.1.7. FTCore\_SetWindowState(int)**

Summary	Instructs the server process application to set the window state.
FTCore_SetWindowState( <i>state</i> : int) -> FTCore_RESULT	
<i>state</i>	The server process application window state: 0 : Default window size 1 : Maximum 2 : Minimize and show on the taskbar 3 : Hide in the task tray 4 : Hidden
Return	Success
	FTCore_SUCCESS
	Failure
	FTCore_ERR_CLIENT_NOEXIST FTCore_ERR_CLIENT_PARAMETER FTCore_ERR_CLIENT_MESSAGESEND
Description	Used to control the window state of a server process application from a client process.



## 2.5.1.8. FTCORE\_LoadDefaultCategory()

Summary		Instructs the server process application to set the default category.
FTCORE_LoadDefaultCategory( ) -> FTCORE_RESULT		
-----		No parameters
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		Instructs the server process application to revert to the default category indicated by <u>2.3.2.1 DefaultCategory</u> .

## 2.5.1.9. FTCORE\_SetCustomCategory(str)

Summary		Instructs the server process application to set the custom category.
FTCORE_SetCustomCategory( <i>categories</i> : <b>str</b> ) -> <b>FTCORE_RESULT</b>		
<i>categories</i>		Any category string (separate multiple strings with commas)
Return	Success	<b>FTCORE_SUCCESS</b>
	Failure	<b>FTCORE_ERR_CLIENT_NOEXIST</b> <b>FTCORE_ERR_CLIENT_PARAMETER</b> <b>FTCORE_ERR_CLIENT_MESSAGESEND</b>
Description		<p>Instructs the server process application to set the specified category in categories as the category it recognizes.</p> <p>By using this API, the specified string will be recognized as a category, and the filter function of the server process application will work correctly.</p> <p>If specifying multiple categories, separate them with commas.</p> <p>In the categories recognized by the server process application, the string NON is a reserved word that means “not included in any category.”</p> <p>Regardless of whether NON is included in categories, the server process application inserts NON at the beginning of the category list and operates accordingly.</p>





## 2.5.1.10. FTCORE\_SetCustomCategoryFromFile(str)

Summary		Instruct the server process application to set custom categories by reading files.
FTCORE_SetCustomCategoryFromFile( <i>filename</i> : str) -> FTCORE_RESULT		
<i>filename</i>		Full path name of the category settings file to be read
Return	Success	FTCORE_SUCCESS
	Failure	FTCORE_ERR_CLIENT_NOEXIST FTCORE_ERR_CLIENT_PARAMETER FTCORE_ERR_CLIENT_MESSAGESEND
Description		<p>Instructs the server process application to set the category read from the file as the recognized category.</p> <p>By using this API, the specified string will be recognized as a category, and the filter function of the server process application will work correctly.</p> <p>The category file can be created by selecting the [Advanced Settings] menu in the server process application and choosing [User definition] as the category source type.</p> <p>For more details, please refer to the Logging Foot Viewer Appcaliton Operation Guide [2.15.Configuring Advanced settings].</p>



## 2.5.2. Enumration

Regarding `DefaultCategory` and `Severity`, you can pass any string as arguments to the `FTCORE_SendMessage()`; however, when the filter function operateds on the server process application side, these strings are compared against predefined enumerated strings.

To ensure the filter function works correctly in the server process application, you must use the same strings as those defined in the enumeration here.

### 2.5.2.1. DefaultCategory

Indicates the category of the log to be output.

Enum	DefaultCategory	
Member	Value	Example
NON	0	Not included in any category.
APP	1	Application operation
SYSTEM	2	System-related
USER	3	User operation
UI	4	UI triggered event
WF	5	Business logic / Workflow triggered event
DEVICE	6	Device triggered event
DEBUG	7	For debug
STEP	8	Execution step
EVENT	9	Event
COMM	10	Communication-related

### 2.5.2.2. Severity

Indicates the severity of the log to be output.

Enum	Severity	
Member	Value	Example
NONE	0	Not included in any severity.
INFO	1	General information
NOTICE	2	Caution / Attention
WARNING	3	User operational error / Warning
ERR	4	User operational error / Recoverable error
FATAL	5	Unrecoverable error / Fatal error



## 2.5.2.3. FTCCORE\_RESULT

Indicates the types of return values provided by the API.

Enum	FTCCORE_RESULT	
Member	Value	Example
FTCCORE_UNKNOWN_STATE	-1	Excluded / Invalid Value
FTCCORE_SUCCESS	0	Success
FTCCORE_EVT_CLIENT_DETECTDISCON	4	Detected disconnection from server process
FTCCORE_EVT_CLIENT_RECEIVED	5	Received data from server process
FTCCORE_ERR_CLIENT_NOTSERVER	21	No connectable server process exists
FTCCORE_ERR_CLIENT_FAILSERVER	22	Failed to start server process
FTCCORE_ERR_CLIENT_NOEXIST	23	Client process has not been created
FTCCORE_ERR_CLIENT_ALREADY	24	Client process has already been created
FTCCORE_ERR_CLIENT_PARAMETER	25	Transmission parameter error occurred
FTCCORE_ERR_CLIENT_HOSTINFO	26	Failed to obtain destination information
FTCCORE_ERR_CLIENT_SOCKET	27	Socket error occurred
FTCCORE_ERR_CLIENT_REFUSED	28	Connection to server process was refused
FTCCORE_ERR_CLIENT_UNREACHED	29	Network path to server process does not exist
FTCCORE_ERR_CLIENT_CONNECT	30	Failed to connect to server process
FTCCORE_ERR_CLIENT_ESTABLISH	31	Error occurred while establishing connection to server process
FTCCORE_ERR_CLIENT_MESSAGESEND	32	Failed to send message to server process
FTCCORE_ERR_CLIENT_MESSAGERCV	33	Failed to receive message from server process
FTCCORE_ERR_CLIENT_CONNUNKNOWN	34	Received a notification accepting a connection from an unknown server.



## 2.5.3. Development environment

This software has been verified to operate in the development environments listed below. Please note that using development environments different from those specified may result in build errors. Appropriate measures should be taken accordingly.

IDE	Visual Studio 2022 Community
Interpreter	Python 3.13 (64-bit)
Used Add-ons	pywin32

### 2.5.3.1. Environment Setup

(1) Install Python 3.13

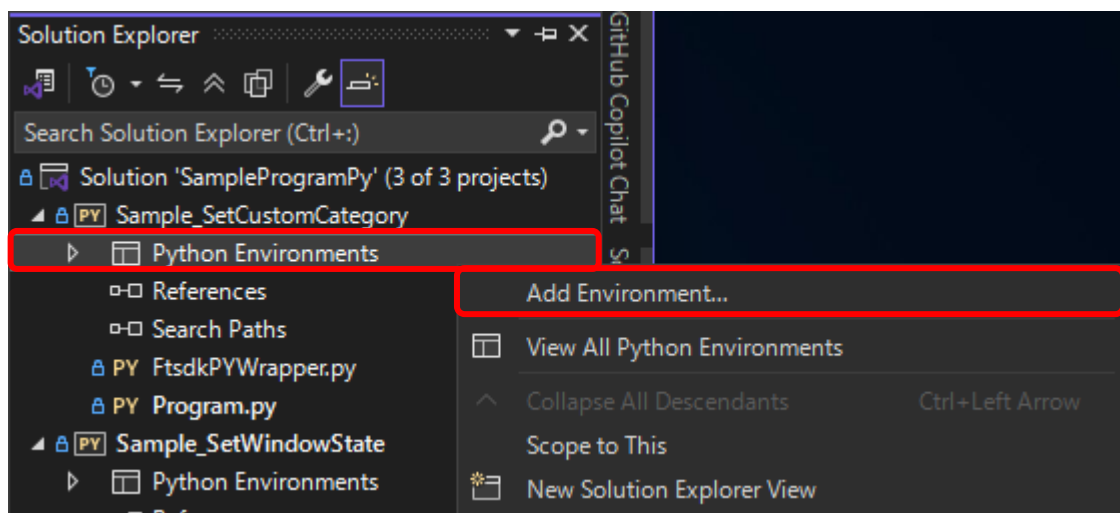
<https://www.python.org/downloads>

(2) Launch Visual Studio 2022 Community and open the following solution file:

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

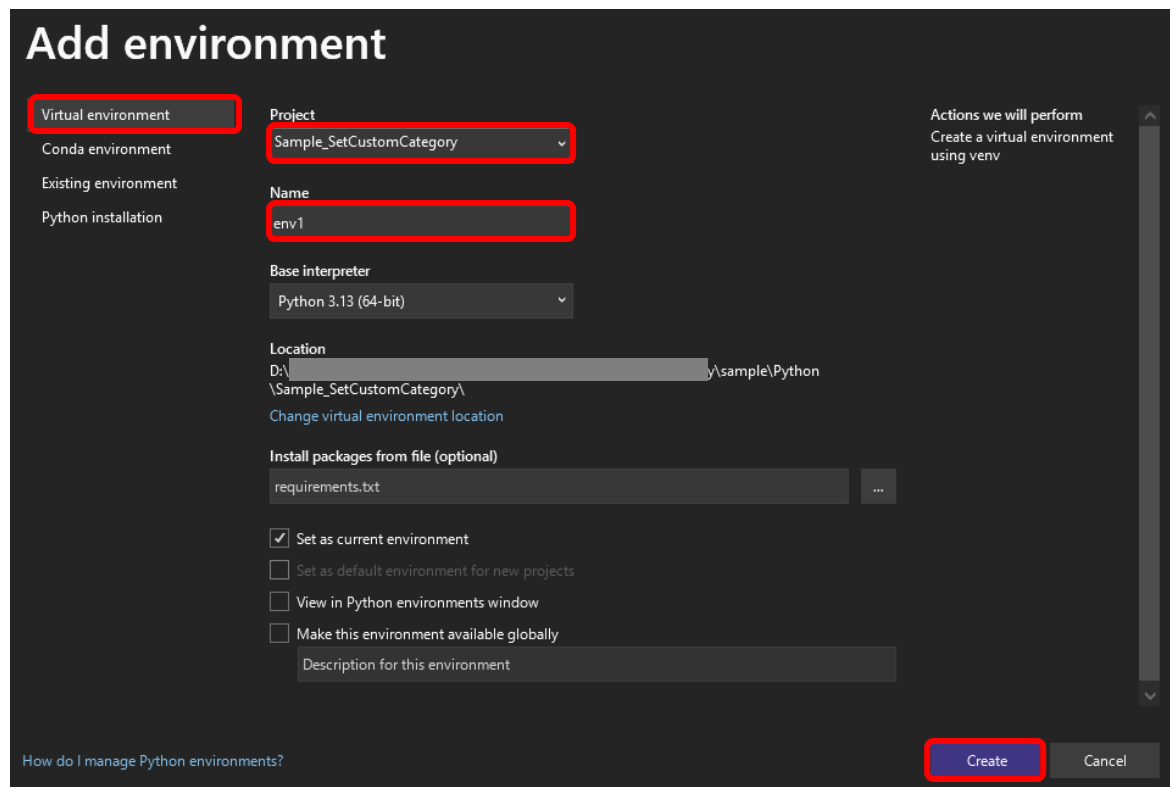
(3) Add a Virtual Environment

In the **Solution Explorer**, right-click on the **Python Environments** node to open the context menu, then select **"Add Environment"**.



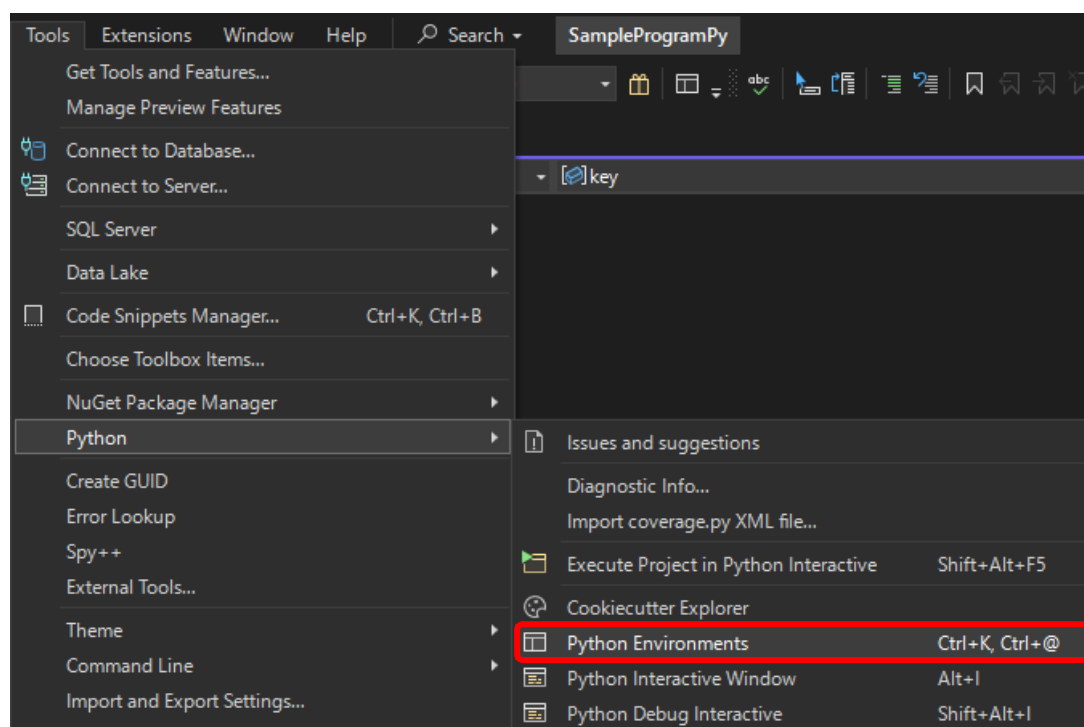


From the left-hand menu shown below, select **[Virtual environment]**, then choose the target project. Enter a name for the virtual environment and click the **[Create]** button.



#### (4) Install pywin32

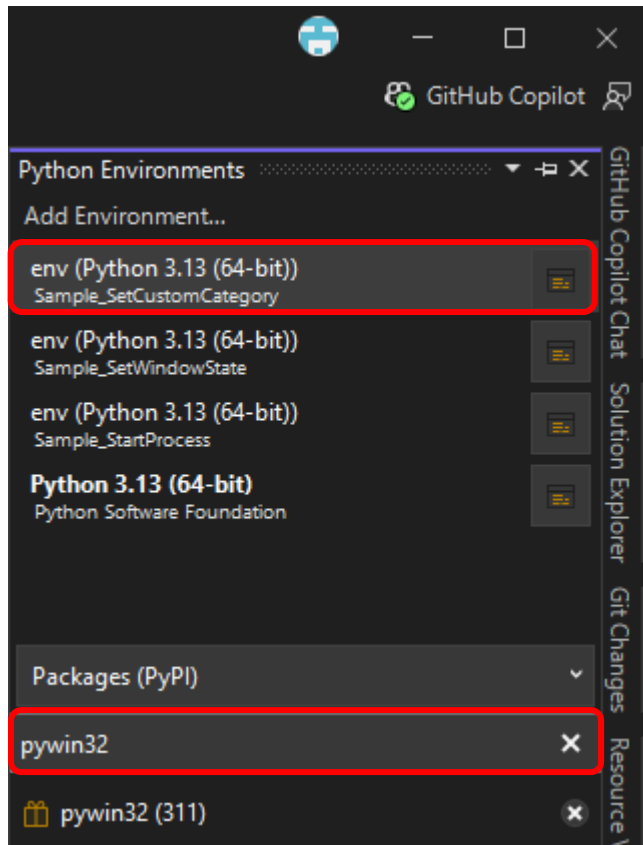
From the menu bar, navigate to **[Tools] > [Python] > [Python Environments]**.





Select the virtual environment added in step (3), then enter **pywin32** into the **PyPI package input box** and press **Enter**.

The installation will begin automatically.



### 2.5.4. How to use FTSDK

In the following instructions, all paths refer to locations under the default installation directory:

**C:\Program Files\LoggingFoot\**

- (1) Add the wrapper program source file **FtsdkPYWrapper.py**, located in the **sample\Python\ftsdk\src** folder, to your project.
- (2) Copy the **ft\_cli.dll** file located in the **sample\Python\ftsdk\bin** folder to the execution directory of your application.
- (3) In the **FtsdkPYWrapper.py** file, update the line **\_DLL\_NAME = "..\ftsdk\bin\ft\_cli.dll"** to reflect the actual path where the DLL was placed in step (2).



## 2.5.5. Sample program

---

The sample program is included in **sample\Python**.

For descriptions of the operation of each program, refer to section **2.6 Description of Sample Program Behavior**.

Program name	Description
Sample_StartProcess	This is a sample program that starts logging by launching the client process while the server process application is running. Main APIs Used: FTCORE_StartProcess() FTCORE_SendMessage() FTCORE_ExitProcess()
Sample_SetCustomCategory	This is a sample program that instructs the server process application to configure custom categories. Main APIs Used: FTCORE_LoadDefaultCategory() FTCORE_SetCustomCategory() FTCORE_SetCustomCategoryFromFile()
Sample_SetWindowState	This is a sample program that controls the window state of the server process application from the client process. Main APIs Used: FTCORE_SetWindowState()



## 2.6. Description of Sample Program Behavior

This section provides operation explanations for sample programs in C++, C#, VB, and Python. For explanations regarding JavaScript sample program operations, please refer to [Section 2.4.4 How to use](#).

Unless otherwise noted, the server process is referred to as the Logging Foot Viewer application, or simply the Viewer application in this section.

In the following descriptions, the paths indicated refer to those under the default installation folder (C:\Program Files\LoggingFoot\).

### 2.6.1. Sample\_StartProcess

This sample program demonstrates the most basic usage of the FTSDK API.

- (1) Start the Logging Foot Viewer application (**ftviewer\ft\_viewer.exe**) beforehand.
- (2) Run **sample\Cpp\Sample\_StartProcess.exe**, located in the **Sample\_StartProcess\bin\Debug** or **Release** folder.
- (3) When the executable starts, it will wait for a key input; press any key to continue.

```
*****
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) For example, when the “A” key is pressed, the following APIs are called as shown below:

- FTCCORE\_StartProcess()
- FTCCORE\_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.

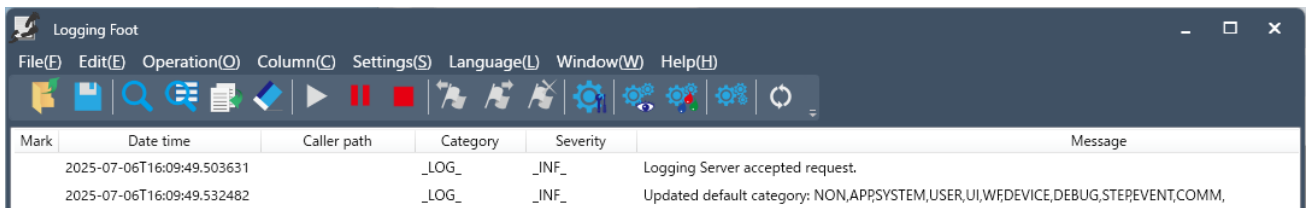
> Wait for any key to be pressed ...
> a
> a key was pressed.
> Called FTCCORE_StartProcess()..
> result: FTCCORE_SUCCESS

> Called FTCCORE_LoadDefaultCategory()..
> result: FTCCORE_SUCCESS

> Press any key then, Output log message.

> |
```

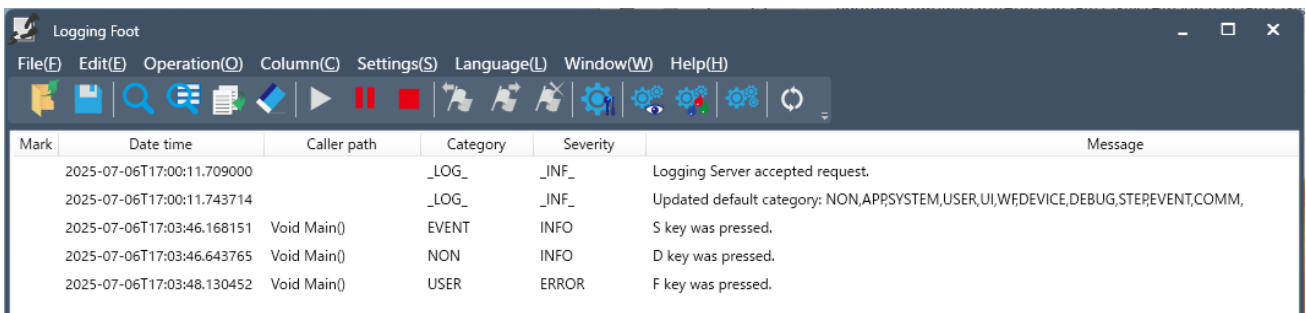
The Viewer application shown below outputs internal logs indicating that the connection was established by FTCCORE\_StartProcess(), and that the default category was updated by FTCCORE\_LoadDefaultCategory().



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,APP,SYSTEM,USER,UI,WIFI,DEVICE,DEBUG,STREVENT,COMM,

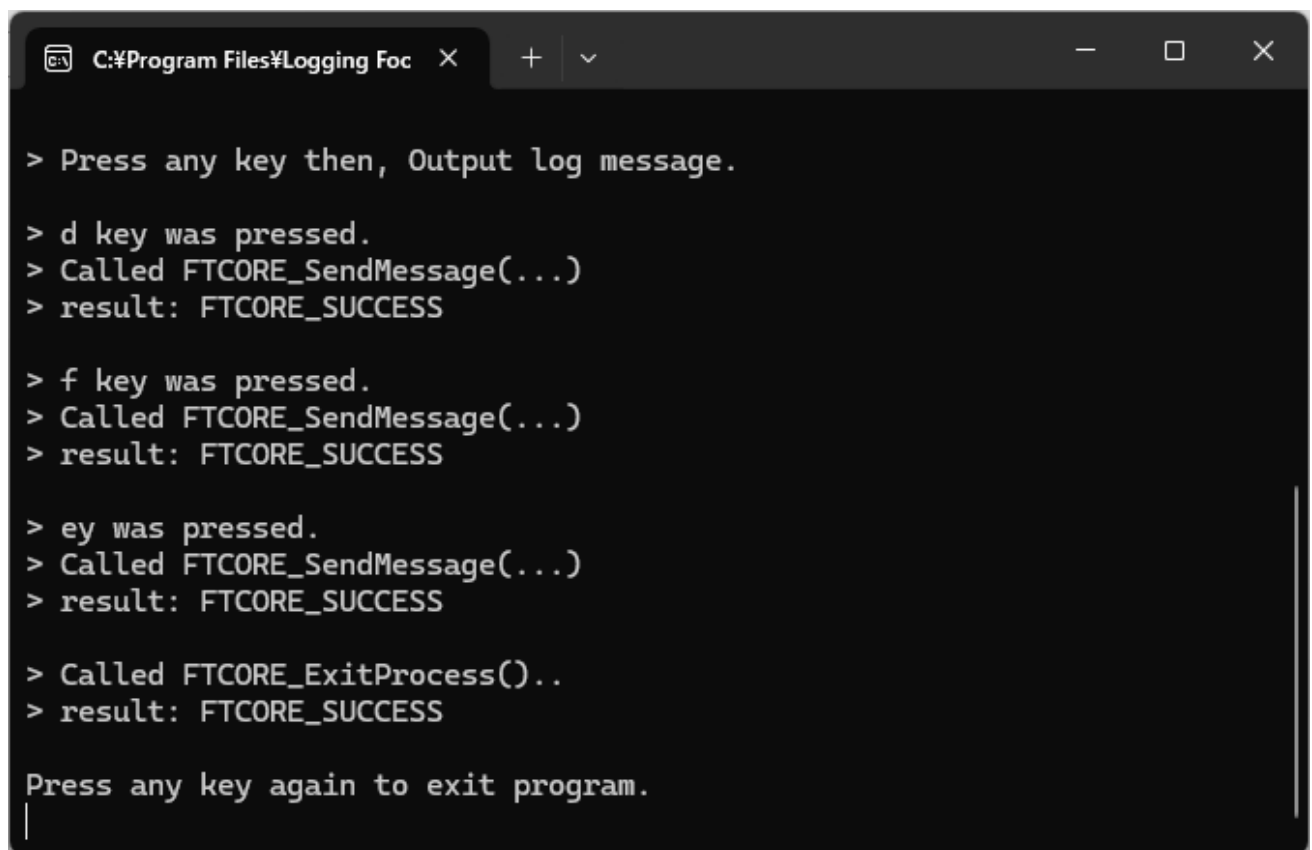


(5) Thereafter, each time any key is pressed, the input key is logged to the Viewer application.



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,WfDEVICE,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) To exit the sample program, first press the Esc key, then press any key.



```
C:\Program Files\Logging Foc > 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.
```

The Viewer application outputs an internal log indicating that the connection has been terminated.



## 2.6.2. Sample\_SetCustomCategory

This sample program demonstrates how to use the API for setting custom categories.

- (1) Start the Logging Foot Viewer application (**ftviewer\ft\_viewer.exe**) beforehand.
- (2) Run **sample\Cpp\Sample\_SetCustomCategory.exe**, located in the **Sample\_SetCustomCategory\bin\Debug** or **Release** folder.
- (3) When the executable starts, it will wait for key input; press any key to continue.

```
*****
LoggingFoot FTSDK sample program.
How to use FTCORE_StartProcess and FTCORE_SetCustomCategory.

Copyright c 2025 Quantityworks 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) For example, when the “A” key is pressed, the following API is called as shown below:  
**FTCORE\_StartProcess()**

Additionally, a selection menu for choosing which API to call will be displayed:

- 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) Enter the number corresponding to the API to be called.

For example, entering 1 sets the custom category defined in the source code.

```
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()

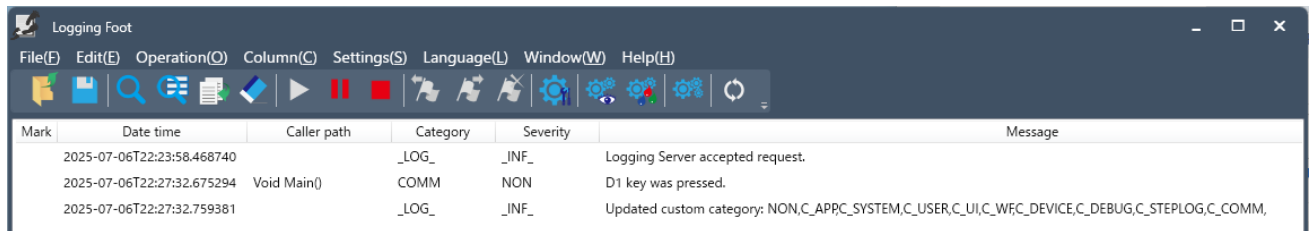
> 1 key was pressed.
> Called FTCORE_SendMessage(...)
> result: FTCORE_SUCCESS

> Called FTCORE_SetCustomCategory()..
> result: FTCORE_SUCCESS

> |
```

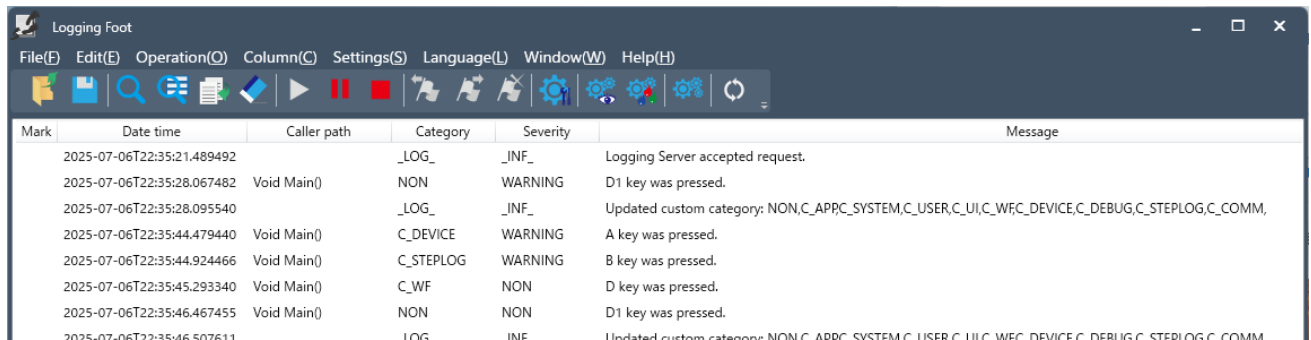


The Viewer application outputs an internal log indicating that the custom category was updated by `FTCORE_SetCustomCategory()`.



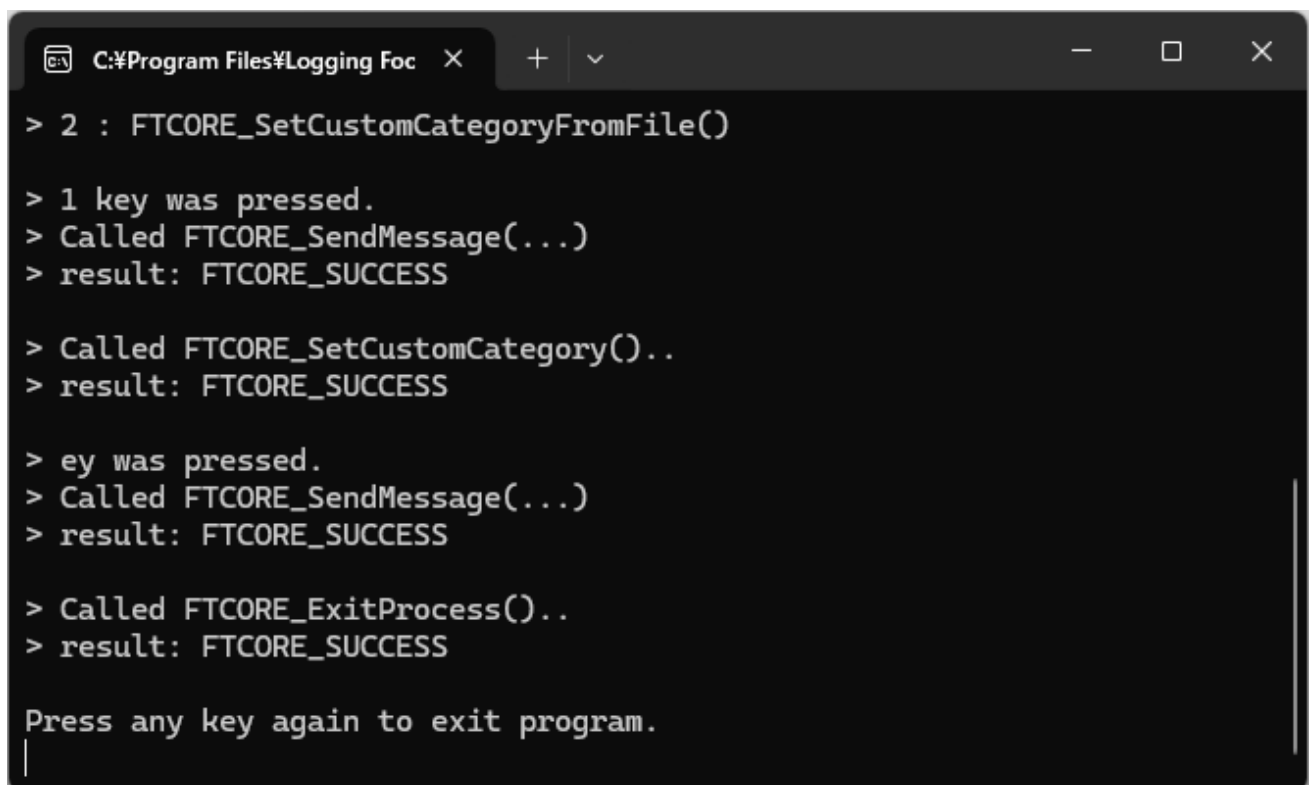
Mark	Date time	Caller path	Category	Severity	Message
	2025-07-06T22:23:58.468740		_LOG_	_INF_	Logging Server accepted request.
	2025-07-06T22:27:32.675294	Void Main()	COMM	NON	D1 key was pressed.
	2025-07-06T22:27:32.759381		_LOG_	_INF_	Updated custom category: NON,C_APPC_SYSTEM,C_USER,C_UI,C_WFC_DEVICE,C_DEBUG,C_STEPLOG,C_COMM,

(6) Thereafter, entering 0 to 2 will call the respective category-setting API described above, while entering any other key will output the pressed key's character to the Viewer application.



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) To exit the sample program, first press the Esc key, then press any key.



```

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

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

> Called FTCORE_SetCustomCategory(...)
> result: FTCORE_SUCCESS

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

> Called FTCORE_ExitProcess(...)
> result: FTCORE_SUCCESS

Press any key again to exit program.
  
```

The Viewer application outputs an internal log indicating that the connection has been terminated.



## 2.6.3. Sample\_SetWindowState

---

This sample program demonstrates how to use the API that controls the Viewer application's window size from the client process.

- (1) Start the Logging Foot Viewer application (**ftviewer\ft\_viewer.exe**) beforehand.
- (2) Run **sample\Cpp\Sample\_SetWindowState.exe**,  
located in the **Sample\_SetWindowState\bin\Debug** or **Release** folder.
- (3) When the executable starts, it will wait for key input; press any key to continue.

```
C:\Program Files\Logging Foot Viewer\ft_viewer.exe
*****
LoggingFoot FTSDK sample program.
How to use FTCORE_StartProcess and FTCORE_SetWindowState.

Copyright c 2025 Quantyworks Software

FTSDK ver.1.1.0.1
*****

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) For example, when the “A” key is pressed, the following API is called as shown below:

FTCORE\_StartProcess()

Additionally, options for selecting the window size control are displayed:

0: Default.

1: Maximized.

2: Minimized to the taskbar.

3: Hide in the system tray.

4: Fully hidden.

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

> |
```



(5) Enter the number corresponding to the window size control option.

For example, enter 1 (maximize).

```
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

> |
```

The FTCORE\_SetWindowState() API is called, and the Viewer application will be displayed in a maximized state.

(6) Thereafter, entering 0 to 4 will call the respective API described above to control the window size.

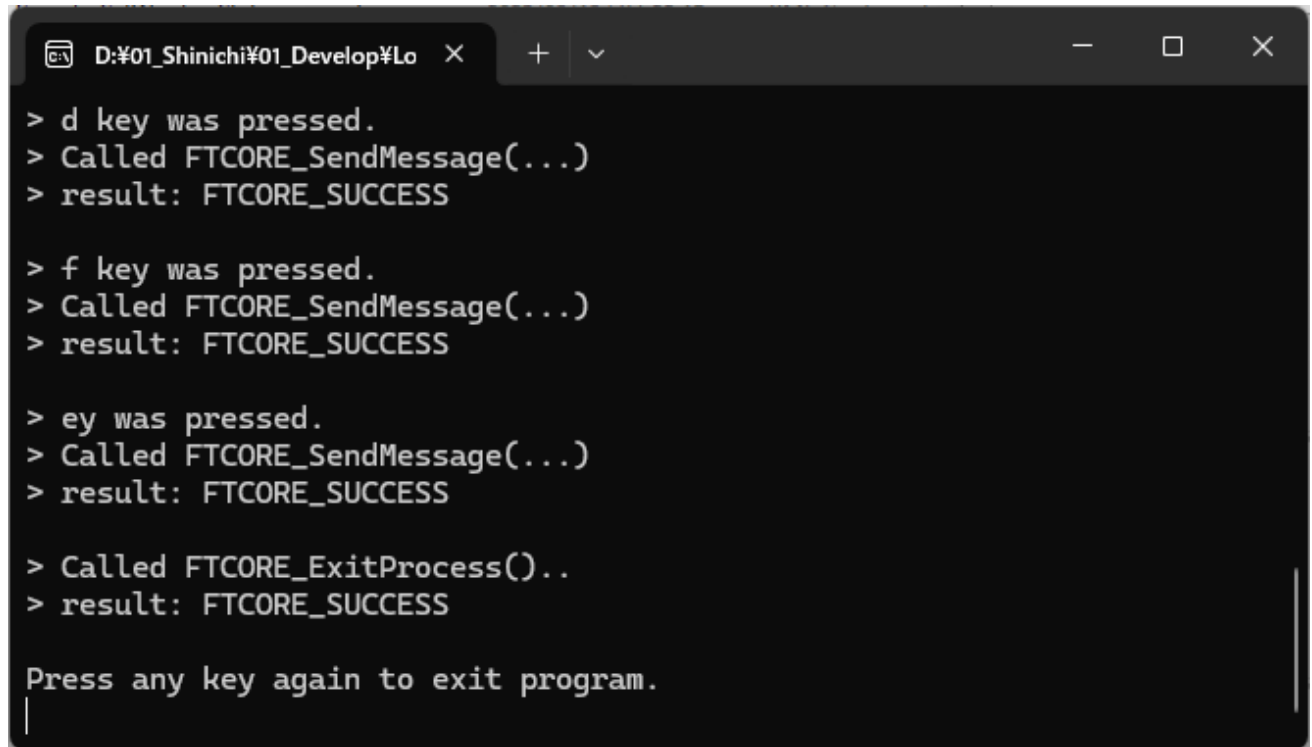
Entering any other key will output the pressed key's character to the Viewer application.

Logging Foot					
File(E) Edit(E) Operation(O) Column(C) Settings(S) Language(L) Window(W) Help(H)					
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_	Change 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_	Change 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_	Change 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.





(7) To exit the sample program, first press the Esc key, then press any key.



```
> 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.
```

The Viewer application outputs an internal log indicating that the connection has been terminated.



## 2.6.4. Sample\_StartupServer

This sample program demonstrates how to use the API that launches the Viewer application and starts logging from the client process.

(1) Run **sample\Cpp\Sample\_StartupServer.exe**,

located in the **Sample\_StartupServer\bin\Debug** or **Release** folder.

Note: If the Logging Foot Viewer application (**ftviewer\ft\_viewer.exe**) is already running, please close it beforehand.

(2) When the executable starts, the screen shown below will be displayed.

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:

The APIs used in this program operate only when the client process and the Viewer application reside on the same PC; therefore, leave this set to localhost.



### **Server path:**

Specify the relative or absolute path to the Viewer application (ft\_viewer.exe).

If a relative path is used, it is relative to this program (Sample\_StartupServer.exe).

### **Port number:**

Specify the communication port for the Viewer application. If you change the default value of 50500, you must also update the port setting in the Viewer application accordingly.

### **Call API:**

Select the API to be called.

### **Window state:**

Select the window state of the Viewer application at startup.

### **Topmost:**

Select whether the Viewer application window should be displayed on top (always in front).

### **Startup:**

Pressing this button calls the selected API from **Call API**, launches the Viewer application, and starts logging.

### **Category:**

Select the category for the output log. Checking **Random** selects categories randomly.

### **Severity:**

Select the severity level of the output log. Checking **Random** selects severity levels randomly.

### **Message:**

Enter the log message string to output.

### **Send:**

Pressing this button sends the configured log to the Viewer application, which will output the log accordingly.



## 2.6.5. Sample\_UnitTest

This sample program was used for load testing. It can be used to verify how much load is applied on the PC you plan to use.

- (1) Start the Logging Foot Viewer application (**ftviewer\ft\_viewer.exe**) beforehand.
- (2) Run Sample\_UnitTest.exe,  
located in the **sample\Cpp\Sample\_UnitTest\bin\Debug** or **Release** folder.
- (3) When the executable starts, the screen shown below will be displayed.

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)

- Stopped



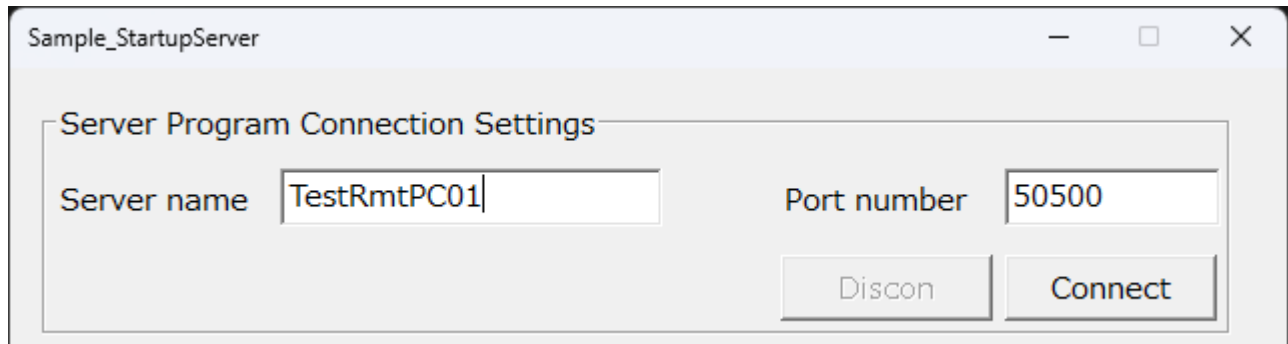
## **Server name :**

Specify the PC name or IP address of the PC running the Logging Foot Viewer application (ftviewer\ft\_viewer.exe).

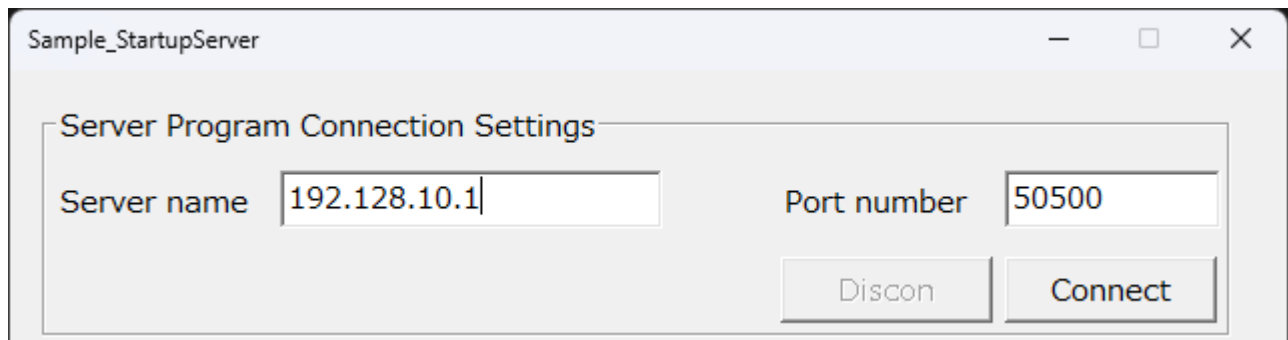
If the client process and Viewer application reside on the same PC, you can specify localhost.

If the Viewer application is running on a remote PC, specify the PC name or IP address of that remote machine.

Example of specifying a remote PC name:



Example of specifying an IP address:



## **Port number:**

Specify the communication port for the Viewer application. If you change the default value of 50500, you must also update the port setting in the Viewer application accordingly.

## **Connect:**

Establish a connection with the Viewer application and start logging.

## **Discon:**

Disconnect from the Viewer application and stop logging.

**Category:**

Select the category for the output log. Checking Random selects categories randomly.

**Severity:**

Select the severity level of the output log. Checking Random selects severity levels randomly.

**Message:**

Enter the log message string to output.

**Send:**

Pressing this button sends the string entered in Message to the Viewer application, which will output the log.

**Repetition Count:**

Specify the number of times to send logs during load testing.

**Repetition Interval:**

Specify the interval between sends during load testing.

**Start:**

Start the load test.

The message entered in Message will be automatically sent to the Viewer application repeatedly according to the count and interval set in Repetition Count and Repetition Interval, and logs will be output.

**Stop:**

Stop the load test.