TMcraft Setup API Function Manual

Original Instructions

Software version: 1.20.1000
Document verison: 1.0
Release date: 2024-11-01

Table of Content

Manual Revision History

Revision	Date	Revised Content
1.0	2024-11-01	Original release

API Revision History

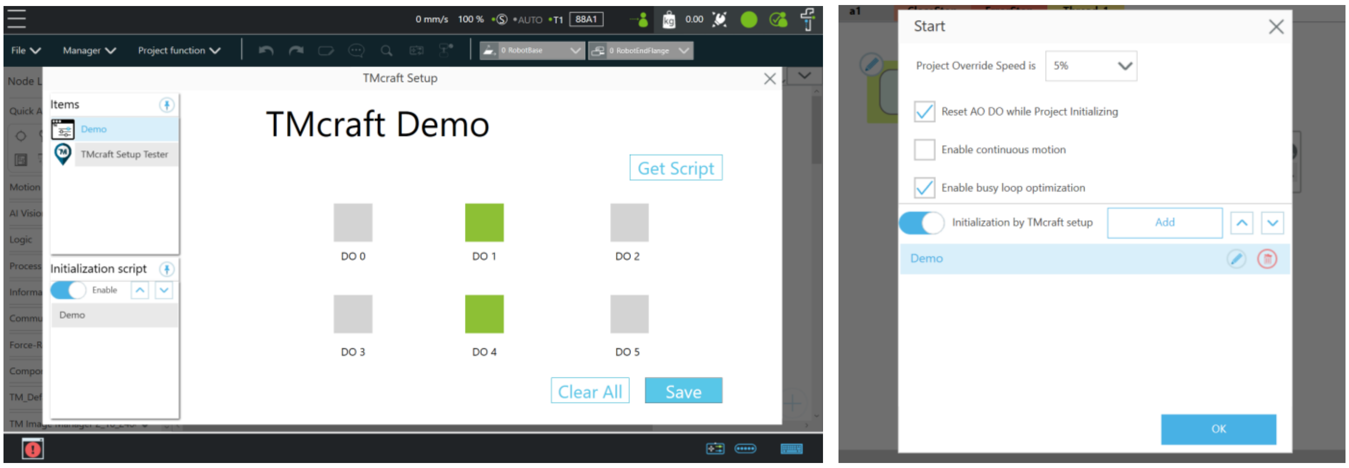
Version	Date	Change Note/History
1.14.1200	2023/8	● 1st release
1.16.1400	2024/2	● [Add] class TMcraftShellAPI ● [Add] class TMcraftToolbarAPI ● [Add] interface ITMcraftToolbarEntry ● [Add] class ErrorStatus ● [Add] FreeBotInfo.MoveMode ● [Add] class MoveMode ● [Add] class LogExportSetting ● [Add] RobotEventType.EndButtonFreeBotChanged
1.18.1400	2024/6	● [Add] class TMcraftSetupAPI ● [Add] class TMcraftNodeAPI.TextfileProvider ● [Add] class TMcraftShellAPI.TextfileProvider ● [Add] class TMcraftToolbarAPI.TextfileProvider ● [Add] TMcraftShellAPI.ProjectRunProvider.GetProjectList ● [Add] TMcraftShellAPI.RobotStatusProvider.GetRobotName ● [Add] TMcraftNodeAPI.RobotStatusProvider.GetRobotModelType ● [Add] TMcraftNodeAPI.RobotStatusProvider.GetFlowVersion
1.20.1100	2024/11	● [Add] TMcraftNodeType.dll ● [Add] class TMcraftNodeAPI.FreeBotProvider ● [Add] class TMcraftNodeAPI.EndButtonEventProvider ● [Deprecated] TMcraftNodeAPI.RobotStatusProvider.GetFreeBot ● [Deprecated] TMcraftNodeAPI.RobotStatusProvider.SetFreeBot ● [Deprecated] TMcraftNodeAPI.RobotStatusProvider.EndButtonClickEvent ● [Add] class TMcraftShellAPI.FreeBotProvider ● [Add] class TMcraftShellAPI.EndButtonEventProvider ● [Deprecated] TMcraftShellAPI.RobotStatusProvider.GetFreeBot ● [Deprecated] TMcraftShellAPI.RobotStatusProvider.SetFreeBot ● [Deprecated] TMcraftShellAPI.RobotStatusProvider.EndButtonClickEvent ● [Add] class TMcraftToolbarAPI.FreeBotProvider ● [Add] class TMcraftToolbarAPI.EndButtonEventProvider ● [Deprecated] TMcraftToolbarAPI.RobotStatusProvider.GetFreeBot ● [Deprecated] TMcraftToolbarAPI.RobotStatusProvider.SetFreeBot ● [Deprecated] McraftToolbarAPI.RobotStatusProvider.EndButtonClickEvent

		● [Add] class TMcraftSetupAPI.FreeBotProvider ● [Add] class TMcraftSetupAPI.EndButtonEventProvider ● [Deprecated] TMcraftSetupAPI.RobotStatusProvider.GetFreeBot ● [Deprecated] TMcraftSetupAPI.RobotStatusProvider.SetFreeBot ● [Deprecated] TMcraftSetupAPI.RobotStatusProvider.EndButtonClickEvent

1. Overview

TMcraft Setup is a customized Setup UI developed based on C#/WPF for use in flow projects. User can use TMcraft Setup in the following scenarios:

- As a control panel or setting page of a device. For example, a gripper setup, user can manipulate the device for some tests and define the parameters (run distance) of the gripper for the project; or, a lifting platform setup, user can run the calibration for the device.
- As an application setup, user can manipulate the devices, set the application parameters and define the initialization of the project by inscribing script to the Start Node.



Project Override Speed is 5%
[✓] Reset AO DO while Project Initializing
[ ] Enable continuous motion
[✓] Enable busy loop optimization
Initialization by TMcraft setup

Figure 1: A Sample of TMcraft Setup

TMcraft Setup interacts with TMflow through TMcraft (Setup) API, so that users can manipulate the I/Os, variables, etc. The TMcraft Setup, after configuration, creates an associated initialization script that runs when the project starts.

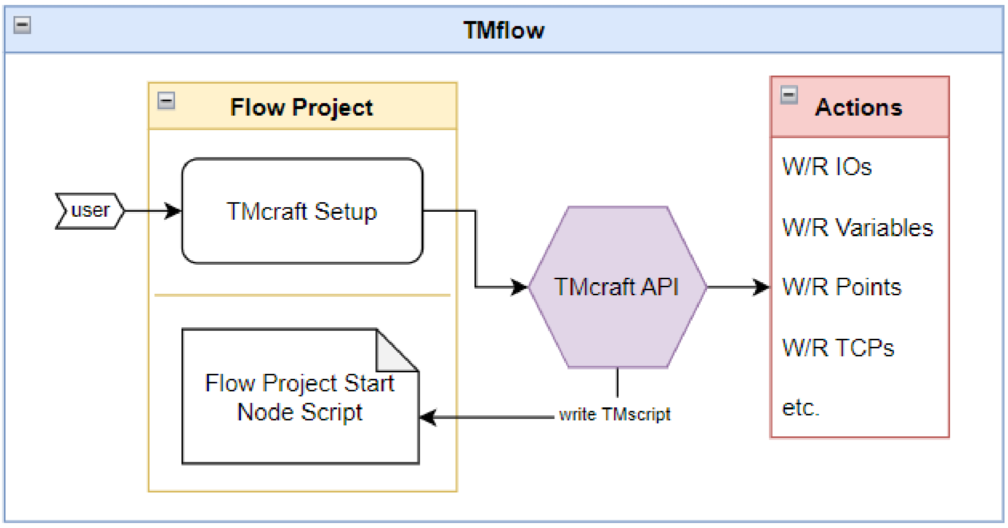


Figure 2: System architecture of TMcraft Setup within TMflow

Plugins / capabilities	Node	Shell	Toolbar	Setup
Base (Add/Edit/Delete)	✓			✓
Point (Add/Edit/Delete)	✓			✓
Tool (Add/Edit/Delete)	✓	✓	✓	✓
Digital IO (Read/Write)	✓	✓	✓	✓
Analog IO (Read/Write)	✓	✓	✓	✓
Project Variables (New/Edit)	✓	✓	✓	✓
Global Variables (New/Edit)	✓	✓	✓	✓
Vision Job (Add/Open/Delete)	✓			
Jog the robot	✓	✓	✓	
Freebot (Set/Get)	✓	✓	✓	✓
End Button Event	✓	✓	✓	✓
Get Current Language	✓	✓	✓	✓
Get TMflow Type	✓	✓	✓	✓
Text file (Read/Write)	✓	✓	✓	✓
TMscript on flow project (Read/Write)	✓			✓
Login/Logout/Get Control		✓		
script Project (Add/Edit/Delete)		✓		
Robot status (Error, Run, etc.)		✓	✓	
Error Event		✓	✓	
Virtual Robot Stick		✓		
Export/Import		✓		
Variables Runtime Value (Read/Write)		✓	Read only	

Table 1: A brief overview of the capabilities of various TMcraft plugin APIs

To develop and implement a TMcraft Setup, developers should firstly build it as a User Control Library (dll file, not exe file). Next, generate a TMcraft Setup zip with the TMcraft Packer from the TMcraft Development Kit; during the process, the TMcraft Packer compile the User Control Library into an execution file and zip it with the resource files within the source folder. Finally, import the TMcraft Setup zip to TMflow.

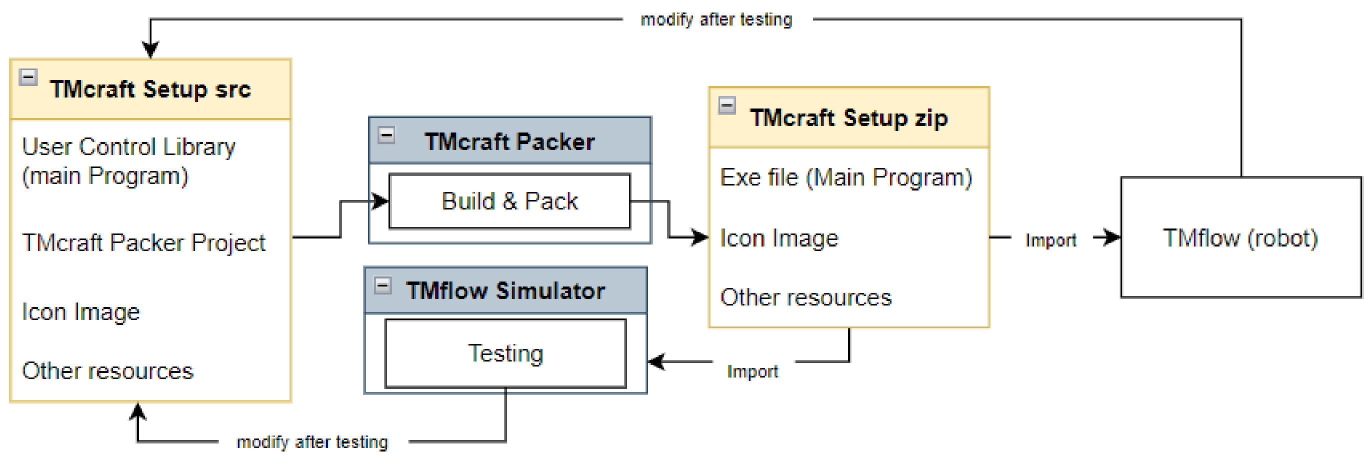


Figure 3: Development Process of a TMcraft Setup

This manual briefly explains the framework of a TMcraft Setup Program and outlines all TMcraft Setup API functions. Note that this manual may not cover all enums and additional classes in the TMcraft.dll, but the most relevant to the TMcraft Setup.

2. Programming with TMcraft API

To understand the TMcraft Setup program structure, refer the sample code below.

```
using TMcraft;

namespace TMcraftSample
{
        public partial class UserControl1 : UserControl, ITMcraftSetupEntry
        {
                TMcraftSetupAPI SetupUI;

                public MainPage()
                {
                        InitializeComponent();
                }

                public void InitializeSetup(TMcraftSetupAPI _SetupUI) //executed when the Setup UI is opened
                {
                        SetupUI = _SetupUI; //connect TMflow
                }

}
```

First, TMcraft.dll should be included as reference (using TMcraft). Secondly, implement the Interface ITMcraftSetupEntry to the User Control class. This interface requires a member function: InitializeSetup().

- InitializeSetup() is activated once opened the Setup UI, which connects the Setup UI with TMflow. More specifically, this makes the Setup UI available for calling all sorts of TMcraft functions.

The rest of the Program should be all sorts of event functions that can interact with TMflow through TMcraft functions.

3. TMcraft API functions (Setup related)

3.1 TMcraftSetupAPI

TMcraft.dll is a combination of the APIs of all sort of TMcraft items; for TMcraftSetup, please declare an object of the class *TMcraftSetupAPI* and use the function within. Like other TMcraft API, *TMcraftSetupAPI* contains different members (or providers) functions in order to interact with TMflow, such as creating Project variables or manipulating I/Os the robot, etc.

IMPORTANT:
TM AI + AOI Edge comes without any robot-related functionality, so it does not support some TMcraft API functions. For TMcraft Setup, the unsupported functions include:

- BaseProvider: all functions
- PointProvider: all functions
- RobotStatusProvider: all functions, except GetFlowVersion and GetOperation Mode
- TCPProvider: all functions
- Enumeration types: RobotEventType
- Additional class: BaseInfo, PointInfo, TCPInfo

3.1.1 Version

Syntax

string TMcraftSetupAPI.Version

Description

A member of the TMcraftSetupAPI class. Returns a string represents the version of the current TMcraft.dll and is read-only.

Return

string	Version of the current TMcraft API

3.1.2 Close

Syntax

TMcraft.TMcraftErr Close()

Description

Closes the current TMcraft Setup.

Parameters

No parameters are required.

Return

TMcraft.TMcraftErr	Returns TMcraftErr.OK if the function works properly; otherwise, returns the corresponding TMcraftErr. For more detail, please check enum TMcraft.TMcraftErr.

3.1.3 GetErrMsg

Syntax

```
TMcraft.TMcraftErr GetErrMsg(
    unit errorCode,
    out string ErrorMessage
)
```

Description

Output the error message according to the error code input. This function is used for checking the result of calling Provider functions.

Parameters

errorCode	The unit error code returned by most Provider functions.
errorMessage	Response the associated error message by the input error code.

Return

TMcraft.TMcraftErr	Returns TMcraftErr.OK if the function works properly; otherwise, returns the corresponding TMcraftErr. For more detail, please check enum TMcraft.TMcraftErr.

3.2 ITMcraftSetupEntry

ITMcraftSetupEntry is an Interface provided by TMcraft API which defines a contract of being a TMcraft Setup. Any class that implements this contract must provide an implementation of a member function defined in the Interface: InitializeSetup().

3.2.1 InitializeSetup

Syntax

```
void InitializeSetup(
            TMcraftSetupAPI tMSetupEditer
)
```

Description

Initializes the Setup with user-defined actions.

Parameters

tMSetupEditor	The TMcraftSetupAPI object connects the TMcraft Setup with TMflow.

Return

None.

3.3 BaseProvider

BaseProvider provides functions for TMcraft item to access or modify the base value of the current Project.

3.3.1 ChangeBaseValue

Syntax

```
uint ChangeBaseValue(
    string baseName,
    float[] baseData
)
```

Description

Modifies a specific Base.

Parameters

baseName	Name of the target Base.
baseData	A 6×1 float array, {x, y, z, rx, ry, rz}, that can be the new value of the target Base.

Return

uint	The error code that represents the result of the function calling.

3.3.2 CreateNewBase

Syntax

```
uint CreateNewBase(
        string baseName,
        float[] baseData
)
```

Description

Creates a new Base.

Parameters

baseName	Name of the base being created
baseData	A 6×1 float array, {x, y, z, rx, ry, rz}, that defines the newly created base.

Return

uint	The error code that represents the result of the function calling.

3.3.3 DeleteBase

Syntax

```
uint DeleteBase(
    string baseName
)
```

Description

Deletes a specific Base.

Parameters

baseName	Name of the Base being deleted.

Return

uint	The error code that represents the result of the function calling.

3.3.4 GetBaseList

Syntax

```
uint GetBaseList(
                ref List<BaseInfo> bases
)
```

Description

Gets the Base list of the current project.

Parameters

bases	A List of BaseInfo objects.

Return

uint	The error code that represents the result of the function calling.

3.3.5 IsBaseExist

Syntax

```
bool IsBaseExist(
    string baseName
)
```

Description

Check if a specific Base exists or not.

Parameters

baseName	Name of the Base being checked.

Return

bool	True if the base exists, false if not.

3.4 DataStorageProvider

Every TMcraft Setup has its own data storage on each project; for simplicity, this architecture will be mentioned as “the current TMcraft Setup data storage” in the all function description of TMcraftSetup.DataStorageProvider . Throguh DataStorageProvider, the TMcraft Setup Program can access and manipulate its own data storage on the current project.

3.4.1 GetAllData

Syntax

```
uint GetAllData(
    out Dictionary<string, object> dataSet
)
```

Description

Gets all data (Dictionary Type) from the current TMcraft Setup data storage.

Parameters

dataSet	A Dictionary type of all data stored within the current TMcraft Setup data storage.

Return

Unit	The error code that represents the result of the function calling.

3.4.2 GetData

Syntax 1

```
uint GetData(
    string key,
    out BaseInfo data
)
```

Description

Gets a specific BaseInfo type data from the current TMcraft Setup data storage by to the string key.

Parameters

key	A string key that provides access to the data.
data	BaseInfo type data being output.

Return

uint	The error code that represents the result of the function calling.

Syntax 2

```
uint GetData(
    string key,
    out PointInfo data
)
```

Description

Gets a specific PointInfo type data from the current TMcraft Setup data storage by the string key.

Parameters

key	A string key that provides access to the data.
data	PointInfo type data being output.

Return

uint	The error code that represents the result of the function calling.

Syntax 3

```
uint GetData(
    string key,
    out string data
)
```

Description

Gets a specific string data from the current TMcraft Setup data storage by the string key.

Parameters

key	A string key that provides access to its corresponding data.
data	String type data being output.

Return

uint	The error code that represents the result of the function calling.

Syntax 4

```
uint GetData(
    List<string> keys,
    out Dictionary<string,object> dataSet
)
```

Description

Gets a Dictionary Type of data set, which corresponds to a certain List of string keys, from the current TMcraft Setup data storage.

Parameters

keys	A List of string keys that can provide access to corresponding data stored within the current TMcraft Setup data storage.
dataSet	Dictionary<string, object> being output.

Return

uint	The error code that represents the result of the function calling.

3.4.3 SaveData

Syntax 1

```
SaveData(
        string key,
        string data
)
```

Description

Saves a string data to the current TMcraft Setup data storage, along with its string key.

Parameters

key	A string key that provides access to its corresponding data.
data	String data being stored.

Return

uint	The error code that represents the result of the function calling.

Syntax 2

```
    SaveData(
                Dictionary<string, string> dataSet
    )
```

Description

Saves a DictionaryType (a collection of string keys and string data) o the current TMcraft Setup data storage.

Parameters

dataSet	A Dictionary Type of data stored within the current TMcraft Setup data storage.

Return

uint	The error code that represents the result of the function calling.

Syntax 3

```
uint SaveData(
    string key,
    BaseInfo data
)
```

Description

Saves a BaseInfo type data to the current TMcraft Setup data storage, along with its corresponding string key.

Parameters

key	A string key that provides access to its corresponding data.
data	BaseInfo data being stored.

Return

uint	The error code that represents the result of the function calling.

Syntax 4

```
uint SaveData(
    Dictionary<string, BaseInfo> dataSet
)
```

Description

Saves a Dictionary Type, along with string keys and BaseInfo data it contains. Please note that this dictionary of data belongs to the current TMcraft Setup data storage only.

Parameters

dataSet	A Dictionary type of data stored within the current TMcraft Setup data storage.

Return

uint	The error code that represents the result of the function calling.

Syntax 5

```
uint SaveData(
    string key,
    PointInfo data
)
```

Description

Save a PointInfo type data to the current TMcraft Setup data storage, along with its corresponding string key.

Parameters

key	A string key that provides access to its corresponding data.
data	PointInfo type data being stored.

Return

uint	The error code that represents the result of the function calling.

Syntax 6

```
uint SaveData(
    Dictionary<string, PointInfo> dataSet
)
```

Description

Saves a Dictionary Type that is defined by string keys and PointInfo data. Please note that this dictionary of data belongs to the current TMcraft Setup data storage only.

Parameters

dataSet	A Dictionary type of data stored within the current TMcraft Setup data storage.

Return

uint	The error code that represents the result of the function calling.

3.5 EndButtonEventProvider

EndButtonEventProvider contains functions related to the end button event.

3.5.1 HasEndButtonEventOwnership

Syntax

```
uint HasEndButtonEventOwnership()
```

Description

TMcraft plugin can call this function to check if it has the end button event ownership or not. If yes, this TMcraft plugin is the only one who can recieve the end button event signal.

Parameters

None

Return

bool	Returns True if the TMcraft plugin has the end button event ownership; otherwise, returns Fail.

3.5.2 IsEndButtonBoardcastMode

Syntax

uint IsEndButtonBoardcastMode()

Description

TMcraft plugin can call this function to check if the end button event is currently in boardcast mode. If yes, that means all TMcraft plugins can recieve the event signal; otherwise, one of the TMcraft plugin has the ownership. i.e. other plugins recieve no signal from the event.

Parameters

None

Return

bool	Returns True if the end button event is currenly in boardcast mode; otherwise, returns Fail.

3.5.3 ReleaseEndButtonEventOwnership

Syntax

uint ReleaseEndButtonEventOwnership()

Description

TMcraft plugin can call this function to release the button event ownership.

Parameters

None

Return

uint	The error code that represents the result of the function calling.

3.5.4 SetEndButtonEventOwnership

Syntax

uint SetEndButtonEventOwnership()

Description

TMcraft plugin can call this function to get the end button event ownership.

Parameters

None

Return

uint	The error code that represents the result of the function calling.

3.5.5 EndButtonClickEvent

Description

An event type denotes to the click event occurred on the buttons of the End Module. Function can be linked to this event so that it will be activated once the event is triggered.

3.6 FreebotProvider

FreeBotProvider provides functions related to freebot.

3.6.1 GetFreeBot

Syntax

```
uint GetFreeBot(
    out FreeBotInfo freeBot
)
```

Description

Gets the value of the current FreeBot settings.

Parameters

freeBot	Value of the current FreeBot settings defined by FreeBotInfo.

Return

uint	The error code that represents the result of the function calling.

3.6.2 HoldFreeBotKeyToHandGuide

Syntax

```
uint HoldFreeBotKeyToHandGuide(
    bool holdKey
)
```

Description

Mimics holding the freebot button to enter hand guide mode. Note that, calling this function alone is not enough, another function KeepFreeBot should be running at the same time.

Parameters

holdKey	True means to activate the hand guide mode; false means to deactivate.

Return

uint	The error code that represents the result of the function calling.

3.6.3 KeepFreeBot

Syntax

```
uint KeepFreeBot()
```

Description

Keep the current hand guide mode. After sending HoldFreeBotKeyToHandGuide, this function should be keep sending every 100 - 500 ms until the hand guiding ends, otherwise, the robot will leave hand guide mode.

Parameters

None

Return

uint	The error code that represents the result of the function calling.

3.6.4 SetFreeBot

Syntax

```
uint SetFreeBot(
    FreeBotInfo freeBot
)
```

Description

Sets FreeBot settings.

Parameters

freeBot	A FreeBotInfo being assigned as FreeBot settings.

Return

uint	The error code that represents the result of the function calling.

3.7 IOProvider

IOProvider provides functions for TMcraft item to interact with system I/O.

3.7.1 GetAllIOData

Syntax

```
uint GetAllIOData(
    out List<DeviceIOInfo> ioData
)
```

Description

Gets all IO status.

Parameters

ioData	A List of DeviceIOInfo objects that denotes all IO status data.

Return

uint	The error code that represents the result of the function calling.

3.7.2 ReadAnalogInput

Syntax

```
uint ReadAnalogInput(
    IO_TYPE type,
    int deviceSerialNum,
    int channelNum,
    out float value
)
```

Description

Read the status of a specific Analog Input.

Parameters

type	The IO_TYPE enum that defines which device the target Analog Input belongs to.
deviceSerialNum	Device serial number, which always starts from 0 and is more meaningful if the target device is an external IO module because there might be multiple external IO module devices within the system. The number is 0 if the target device is the Control box IO board or end module IO board because there are only one Control box IO board and one end module IO board.
channelNum	Channel number.
value	Analog Input value, ranged from -10V to 10V.

Return

uint	The error code that represents the result of the function calling.

3.7.3 ReadAnalogOutput

Syntax

```
uint ReadAnalogOutput(
    IO_TYPE type,
    int deviceSerialNum,
    int channelNum,
    out float value
)
```

Description

Read the status of a specific Analog Output.

Parameters

type	The IO_TYPE enum that defines which device the target Analog Output belongs to.
deviceSerialNum	Device serial number, which always starts from 0 and is more meaningful if the target device is an external IO module because there might be multiple external IO module devices within the system. The number is 0 if the target device is the Control box IO board or end module IO board because there are only one Control box IO board and one end module IO board.
channelNum	Channel number.
value	Analog Output value, ranged from -10V to 10V.

Return

uint	The error code that represents the result of the function calling.

3.7.4 ReadDigitInput

Syntax

```
uint ReadDigitInput(
    IO_TYPE type,
    int deviceSerialNum,
    int channelNum,
    out bool status
)
```

Description

Read the status of a specific Digital Input.

Parameters

type	The IO_TYPE enum that defines which device the target Digital Input belongs to.
deviceSerialNum	Device serial number, which always starts from 0 and is more meaningful if the target device is an external IO module because there might be multiple external IO module devices within the system. The number is 0 if the target device is the Control box IO board or end module IO board because there are only one Control box IO board and one end module IO board.
channelNum	Channel number.
status	Digital Input status, where bool true is HIGH and bool false is LOW.

Return

uint	The error code that represents the result of the function calling.

3.7.5 ReadDigitOutput

Syntax

```
uint ReadDigitOutput(
    IO_TYPE type,
    int deviceSerialNum,
    int channelNum,
    out bool status
)
```

Description

Read the status of a specific Digital Output.

Parameters

type	The IO_TYPE enum that defines which device the target Digital Output belongs to.
deviceSerialNum	Device serial number, which always starts from 0 and is more meaningful if the target device is an external IO module because there might be multiple external IO module devices within the system. The number is 0 if the target device is the Control box IO board or end module IO board because there are only one Control

	box IO board and one end module IO board.
channelNum	Channel number.
status	Digital Output status, where bool true is HIGH and bool false is LOW.

Return

uint	The error code that represents the result of the function calling.

3.7.6 SetCameraLight

Syntax

```
uint SetCameraLight(
    bool status
)
```

Description

Switch the Eye-In-Hand camera light to the ON or OFF status.

Parameters

status	bool true denotes turning the light ON, bool false denotes turning the light OFF

Return

uint	The error code that represents the result of the function calling.

3.7.7 WriteAnalogOutput

Syntax

```
uint WriteAnalogOutput(
    IO_TYPE type,
    int deviceSerialNum,
    int channelNum,
    float value
)
```

Description

Set the value of a specific Analog Output.

Parameters

type	The IO_TYPE enum that defines which device the target Analog Output belongs to.
deviceSerialNum	Device serial number, which always starts from 0 and is more meaningful if the target device is an external IO module because there might be multiple external IO module devices within the system. The number is 0 if the target device is the Control box IO board or end module IO board because there are only one Control box IO board and one end module IO board.
channelNum	Channel number.

value	Analog Output value, ranged from -10V to 10V.

Return

uint	The error code that represents the result of the function calling.

3.7.8 WriteDigitOutput

Syntax

```
uint WriteDigitOutput(
    IO_TYPE type,
    int deviceSerialNum,
    int channelNum,
    bool status
)
```

Description

Change the status of a specific Digital Output.

Parameters

type	The IO_TYPE enum that defines which device the target Digital Output belongs to.
deviceSerialNum	Device serial number, which always starts from 0 and is more meaningful if the target device is an external IO module because there might be multiple external IO module devices within the system. The number is always 0 if the target device is the Control box IO board or end module IO board because there are only one Control box IO board and one end module IO board.
channelNum	Signal channel number.
status	Digital Output status, where bool true is HIGH and bool false is LOW.

Return

uint	The error code that represents the result of the function calling.

3.8 PointProvider

PointProvider provides functions for TMcraft item to access or modify Point values within the current project.

3.8.1 ChangePointBase

Syntax

```
uint ChangePointBase(
    string pointName,
    string baseName
)
```

Description

Changes the base of a specific Point.

Parameters

pointName	Name of the target point.
baseName	Name of the Base being switched to.

Return

uint	The error code that represents the result of the function calling.

3.8.2 ChangePointRobotConfigs

Syntax

```
uint ChangePointRobotConfigs(
    string pointName,
    int[] robotConfigs
)
```

Description

Sets the Robot Configs of the specific Point.

Parameters

pointName	Name of the target point.
robotConfigs	A 3×1 interger array representing the robot configurations of the target point. Here is the definition: int[0]: 0 – Right Arm, 1 – Left Arm int[1]: 2 – Above Elbow, 3 – Below Elbow int[2]: 4 – Up Wrist, 5 – Down Wrist

Return

uint	The error code that represents the result of the function calling.

3.8.3 ChangePointToolCoordinates

Syntax

```
uint ChangePointToolCoordinates(
    string pointName,
    float[] toolCoordinates
)
```

Description

Changes the Tool Coordinates of a specific Point.

Parameters

pointName	Name of the target point.
endToolCoordinate	A 6×1 float array {x, y,z, rx, ry, rz} which represents the new Tool Coordinates.

Return

uint	The error code that represents the result of the function calling.

3.8.4 CreatePointByFlangeCoordinates

Syntax

```
uint CreatePointByFlangeCoordinates(
    string pointName,
    float [] flangeCoordinate,
    int[] robotConfigs,
    string baseName,
    string toolName
)
```

Description

Create a new Point defined by Flange Coordinates (and by Point Name, Robot Configs, Base Name, and Tool Name).

Parameters

pointName	Name of the Point being created.
flangeCoordinate	A 6×1 float array {x, y, z, rx, ry, rz}, represents the Flange Coordinates defining the new point.
robotConfigs	A 3×1 interger array denoting the robot configurations of the target point. Here is the definition: int[0]: 0 – Right Arm, 1 – Left Arm int[1]: 2 – Above Elbow, 3 – Below Elbow int[2]: 4 – Up Wrist, 5 – Down Wrist
baseName	The base, which defines the flange, coordinates.
toolName	The tool, which defines the point.

Return

uint	The error code that represents the result of the function calling.

3.8.5 CreatePointByJointAngles

Syntax

```
uint CreatePointByJointAngles(
    string pointName,
    float[] JointAngles,
    string baseName,
    string toolName
)
```

Description

Creates a new Point defined by 6 Joint Angles (and by Point Name, Base Name, and Tool Name).

Parameters

pointName	Name of the point being created.
JointAngles	A 6×1 float array {x, y, z, rx, ry, rz}, represents the Joint Angles defining the new point.
baseName	The base which defines the point.
toolName	The tool which defines the point.

Return

uint	The error code that represents the result of the function calling.

3.8.6 CreatePointByToolCoordinates

Syntax

```
uint CreatePointByToolCoordinates(
    string pointName,
    float[] endToolCoordinate,
    int[] robotConfigs,
    string baseName,
    string toolName
)
```

Description

Creates a new Point defined by end-effector Coordinates (and by Point Name, Robot Configs, Base Name, and Tool Name).

Parameters

pointName	Name of the point created.
endToolCoordinate	A 6×1 float array {x, y, z, rx, ry, rz}, represents the end-effector Coordinates defining the new point.
robotConfigs	A 3×1 interger array denoting the robot configurations of the target point. Here is the definition: int[0]: 0 – Right Arm, 1 – Left Arm int[1]: 2 – Above Elbow, 3 – Below Elbow int[2]: 4 – Up Wrist, 5 – Down Wrist
baseName	The base which defines the end-effector coordinates.
toolName	The tool which defines the end-effector coordinates.

Return

uint	The error code that represents the result of the function calling.

3.8.7 GetPointList

Syntax

```
uint GetPointList(
    ref List<PointInfo> points
)
```

Description

Gets the Point list of the current Project.

Parameters

points	A List of PointInfo objects that denotes the list of points of the current Project.

Return

uint	The error code that represents the result of the function calling.

3.8.8 GetPointRobotConfigs

Syntax

```
uint GetPointRobotConfigs(
    string pointName,
    ref int[] robotConfigs
)
```

Description

Gets the Robot Configs of a specific Point.

Parameters

pointName	Name of the target point.
robotConfigs	A 3×1 interger array representing the robot configurations of the target point. Here is the definition: int[0]: 0 – Right Arm, 1 – Left Arm int[1]: 2 – Above Elbow, 3 – Below Elbow int[2]: 4 – Up Wrist, 5 – Down Wrist

Return

uint	The error code that represents the result of the function calling.

3.8.9 IsPointExist

Syntax

```
bool IsPointExist(
    string pointName
)
```

Description

Check if a specific Point exists or not.

Parameters

pointName	Name of the point being checked.

Return

bool	True if exists, false if not.

3.9 RobotStatusProvider

RobotStatusProvider provides functions for TMcraft item to access different robot status information.

3.9.1 GetCurrentBaseName

Syntax

```
uint GetCurrentBaseName(
    out string baseName
)
```

Description

Gets the name of the current Base.

Parameters

baseName	Current Base name.

Return

uint	The error code that represents the result of the function calling.

3.9.2 GetCurrentPayload

Syntax

```
uint GetCurrentPayload(
    out float payload
)
```

Description

Gets the current payload value set to the robot (end-effector).

Parameters

payload	Payload value being assigned.

Return

uint	The error code that represents the result of the function calling.

3.9.3 GetCurrentPoseByCurrentBase

Syntax

```
uint GetCurrentPoseByCurrentBase(
    out float[] currentPose
)
```

Description

Gets robot current TCP position defined by the Current Base.

Parameters

currentPose	A 6×1 float array {x, y, z, rx, ry, rz} that denotes the current robot pose.

Return

uint	The error code that represents the result of the function calling.

3.9.4 GetCurrentPoseByJointAngle

Syntax

```
uint GetCurrentPoseByJointAngle(
    out float[] jointAngles
)
```

Description

Gets all robot current Joint Angles.

Parameters

jointAngles	A 6×1 float array {j1, j2, j3, j4, j5, j6} that denotes the current robot pose.

Return

uint	The error code that represents the result of the function calling.

3.9.5 GetCurrentPoseByRobotBase

Syntax

```
uint GetCurrentPoseByRobotBase(
    out float[] currentPose
)
```

Description

Gets robot current TCP position defined by the Robot Base.

Parameters

currentPose	A 6×1 float array {x, y, z, rx, ry, rz} that denotes the current robot pose.

Return

uint	The error code that represents the result of the function calling.

3.9.6 GetCurrentRobotConfigs

Syntax

```
uint GetCurrentRobotConfigs(
    out int[] robotConfigs
)
```

Description

Gets current Robot Config.

Parameters

robotConfigs	A 3×1 interger array denoting the robot configurations of the point; here is the definition: int[0]: 0 – Right Arm, 1 – Left Arm int[1]: 2 – Above Elbow, 3 – Below Elbow int[2]: 4 – Up Wrist, 5 – Down Wrist

Return

uint	The error code that represents the result of the function calling.

3.9.7 GetCurrentTcp

Syntax

```
uint GetCurrentTcp(
    out string tcpName
)
```

Description

Gets the name of current TCP.

Parameters

tcpName	Current TCP name.

Return

uint	The error code that represents the result of the function calling.

3.9.8 GetFlowVersion

Syntax

```
uint GetFlowVersion(
    out string result
)
```

Description

Gets the version of TMflow.

Parameters

result	TMflow version.

Return

uint	The error code that represents the result of the function calling.

3.9.9 GetOperationMode

Syntax

```
uint GetOperationMode(
        out int mode
    )
```

Description

Gets current operation mode.

Parameters

mode	Current operation mode, which includes: 0 – Manual and 1 – Auto.

Return

uint	The error code that represents the result of the function calling.

3.9.10 GetRobotModelType

Syntax

```
uint GetRobotModelType(
    out string result
)
```

Description

Gets the model type of the robot.

Parameters

result	Model Type of the robot.

Return

uint	The error code that represents the result of the function calling.

3.9.11 SetCurrentBase

Syntax

```
uint SetCurrentBase(
    string baseName
)
```

Description

Assigns a specific Base as the current base.

Parameters

baseName	Name of the base being assigned.

Return

uint	The error code that represents the result of the function calling.

3.9.12 SetCurrentPayload

Syntax

```
uint SetCurrentPayload(
    float payload
```

)

Description

Sets a payload value to the robot (end-effector).

Parameters

payload	Payload value being assigned.

Return

uint	The error code that represents the result of the function calling.

3.9.13 SetCurrentTcp

Syntax

```
uint SetCurrentTcp(
    string tcpName
)
```

Description

Assigns a specific TCP as the current TCP.

Parameters

tcpName	Name of the TCP being assigned.

Return

uint	The error code that represents the result of the function calling.

3.9.14 ErrorEvent

Description

An event type denotes to the error event occurred on the robot. Function can be linked to this event so that it will be activated once the event is triggered.

3.10 ScriptWriteProvider

Through ScriptWriteProvider functions, TMcraft Setup can manipulate the initialization script of the Flow Project.

3.10.1 AppendLineToBuffer

Syntax

```
void AppendLineBuffer(
    string scriptLine
)
```

Description

Adds a line of script with auto-indentation (*i.e.*, a newline followed by a scriptLine) at the back of the script buffer. Note that the buffer will be cleared once the TMcraft Setup is closed, to save the script, SaveBufferAsScript should be called.

Parameters

scriptLine	Script being added with auto-indentation.

Return

None

3.10.2 AppendScriptToBuffer

Syntax

```
void AppendScriptToBuffer(
    string script
)
```

Description

Adds a script code (without auto-indentation) at the back of the script buffer. Note that the buffer will be cleared once the TMcraft Setup is closed, to save the script, SaveBufferAsScript should be called.

Parameters

script	Script being added without auto-indentation.

Return

None

3.10.3 ClearBuffer

Syntax

```
void ClearBuffer()
```

Description

Clear the script buffer of the TMcraft Setup.

Parameters

None

Return

None

3.10.4 GetScript

Syntax

```
uint GetScript (
    out string script
)
```

Description

Gets the script saved by the current TMcraft Setup.

Parameters

string	Script saved by the current TMcraft Setup.

Return

uint	The error code that represents the result of the function calling.

3.10.5 GetScriptBuffer

Syntax

string GetScriptBuffer ()

Description

Gets the current script buffer.

Parameters

None

Return

string	Current script buffer. Note that script buffer will be cleared automatically once the TMcraft Setup is closed.

3.10.6 SaveBufferAsScript

Syntax

uint SaveBufferAsScript ()

Description

Saves the current script buffer onto the Project.

Parameters

None

Return

uint	The error code that represents the result of the function calling.

3.11 SystemProvider

SystemProvider provides functions for TMcraft item to interact with TMflow System Settings.

3.11.1 GetCurrentLanguageCulture

Syntax

```
uint GetCurrentLanguageCulture(
    out string language
)
```

Description

Gets the current language setting of the system.

Parameters

language	Current System language, *e.g.*, en-US, zh-TW, zh-CN, ja-JP, de-DE, ko-KR

Return

uint	The error code that represents the result of the function calling.

3.11.2 GetTMflowType

Syntax

```
uint GetTMflowType(
        out TMflowType type
)
```

Description

Gets the current TMflow type of the system.

Parameters

type	Represent the TMflow type (e.g. Robot, AOIEdge, etc.) of the current system. For more detail, check the description of enum TMflowType.

Return

uint	The error code that represents the result of the function calling.

3.12 TCPProvider

TCPProvider provides functions for TMcraft item to access or modify TCPs with the robot.

3.12.1 ChangeTcpInertia

Syntax

```
Uint ChangeTcpInertia(
    string tcpName,
    float[] inertiaValue
)
```

Description

Modifies the inertia value of a specific TCP.

Parameters

tcpName	Name of the target TCP.
inertiaValue	A 3×1 float array {Ixx, Iyy, Izz} of inertia value being assigned.

Return

Uint	The error code that represents the result of the function calling.

3.12.2 ChangeTcpMass

Syntax

```
Uint ChangeTcpMass(
    string tcpName,
    float mass
)
```

Description

Modifies the mass value (kg) of a specific TCP.

Parameters

tcpName	Name of the target TCP.

mass	Mass value (kg) to be assigned.

Return

Uint	The error code that represents the result of the function calling.

3.12.3 ChangeTcpMassCenter

Syntax

```
Uint ChangeTcpMassCenter(
    string tcpName,
    float[] massCenter
)
```

Description

Modifies the Mass Center value of a specific TCP.

Parameters

tcpName	Name of the target TCP.
massCenter	A 6×1 float array {x, y, z, rx, ry, rz} that denotes the location of the mass center of the TCP.

Return

Uint	The error code that represents the result of the function calling.

3.12.4 ChangeTcpPose

Syntax

```
Uint ChangeTcpPose(
    string tcpName,
    float[] toolCenterPoint
)
```

Description

Modifies the tool center point of a specific TCP by a 6×1 float array {x, y, z, rx, ry, rz} referring to Flange Base.

Paramters

tcpName	Name of the target TCP being modified.
toolCenterPoint	A 6×1 float array[6] {x, y, z, rx, ry, rz} of new Pose value referring to Flange Base.

Return

Uint	The error code that represents the result of the function calling.

3.12.5 CreateNewTcp

Syntax

```
Uint CreateNewTcp(
    TCPInfo tcpData
)
```

Description

Create a new TCP by using a TCPInfo Type as input.

Parameters

tcpData	TCPInfo type assigned for the new TCP.

Return

Uint	The error code that represents the result of the function calling.

3.12.6 DeleteTcp

Syntax

```
Uint DeleteTcp(
    string tcpName
)
```

Description

Delete a specific TCP file.

Parameters

tcpName	Name of the TCP being deleted.

Return

Uint	The error code that represents the result of the function calling.

3.12.7 GetProjectVisionTcpList

Syntax

```
Uint GetProjectVisionTcpList(
    out List<string> visionTcpList
)
```

Description

Gets the list of Vision TCP Names from the current Project.

Paramters

visionTcpList	A List of vision TCP names.

Return

Uint	The error code that represents the result of the function calling.

3.12.8 GetTcpInertia

Syntax

```
Uint GetTcpInertia(
string tcpName,
out float[] inertiaValue
)
```

Description

Gets the inertia value of a specific TCP.

Parameters

tcpName	Name of the target TCP.
inertiaValue	A 3×1 float array {Ixx, Iyy, Izz} that denotes the inertia value of the target TCP.

Return

Uint	The error code that represents the result of the function calling.

3.12.9 GetTcpList

Syntax

```
Uint GetTcpList(
    out List<TCPInfo> tcpList
)
```

Description

Gets the list of all TCPs (with data) within the robot.

Parameters

tcpList	A List of TCPInfo type that denotes all TCPs within the robot.

Return

Uint	The error code that represents the result of the function calling.

3.12.10 GetTcpMass

Syntax

```
Uint GetTcpMass(
    string tcpName,
    out float mass
)
```

Description

Gets the value of mass (kg) from a specific TCP.

Parameters

tcpName	Name of the target TCP.
mass	Mass value (kg) of the target TCP.

Return

Uint	The error code that represents the result of the function calling.

3.12.11 GetTcpMassCenter

Syntax

```
Uint GetTcpMassCenter(
    string tcpName,
    out float[] massCenter
)
```

Description

Gets the Mass Center value of a specific TCP.

Parameters

tcpName	Name of the target TCP.
massCenter	A 6×1 float array {x, y, z, rx, ry, rz} that denotes the location of the mass center of the TCP.

Return

Uint	The error code that represents the result of the function calling.

3.12.12 IsTcpExist

Syntax

```
bool IsTcpExist(
    string tcpName
)
```

Description

Checks if a specific tcp exists or not.

Parameters

tcpName	Name of the tcp being checked.

Return

bool	True if exists, false if not.

3.13 TextFileProvider

TextFileProvider provides functions for TMcraft plugin to manipulate Textfiles within TMflow.

3.13.1 DeleteTextFile

Syntax

```
uint DeleteTextFile (
string fileName
)
```

Description

Deletes a specific Textfile.

Parameters

fileName	Name of the file being deleted.

Return

uint	The error code that represents the result of the function calling.

3.13.2 ExportTextFile

Syntax

```
uint ExportTextFile (
string fileName
)
```

Description

Exports a specific Textfile to the USB.

Parameters

fileName	Name of the file being exported.

Return

uint	The error code that represents the result of the function calling.

3.13.3 GetTextFileList

Syntax

```
uint GetTextFileList (
out string list
)
```

Description

Gets the list of Textfile names within the current system.

Parameters

list	A list of Textfile names within the current system

Return

uint	The error code that represents the result of the function calling.

3.13.4 ImportTextFile

Syntax

```
uint ImportTextFile (
string robotName,
string fileName
)
```

Description

Import a Textfile to the robot.

Parameters

robotName	Name of the folder where the system can find the item to be imported.
fileName	Name of the file being imported.

Return

uint	The error code that represents the result of the function calling.

3.13.5 NewTextFile

Syntax

```
uint NewTextFile (
string filename,
string fileContent
)
```

Description

Create a new Textfile.

Parameters

fileName	Name of the file being created.
fileContent	Content of the Textfile to be assigned.

Return

uint	The error code that represents the result of the function calling.

3.13.6 ReadTextFile

Syntax

```
uint ReadTextFile (
string filename,
out string fileContent
)
```

Description

Read the content of a specific Textfile.

Parameters

fileName	Name of the file being read.
fileContent	Content of the Textfile to be read.

Return

uint	The error code that represents the result of the function calling.

3.13.7 WriteTextFile

Syntax

```
uint WriteTextFile (
string filename,
string fileContent
)
```

Description

Write content to a specific Textfile.

Parameters

fileName	Name of the file being written.
fileContent	Content of the Textfile to be written.

Return

uint	The error code that represents the result of the function calling.

3.14 VariableProvider

VariableProvider provides functions for TMcraft item to access or modify the variables of the robot.

3.14.1 ChangeGlobalVariableValue

Syntax

```
uint ChangeGlobalVariableValue(
List<string[]> value
)
```

Description

Sets the value of a specific Global Variables.

Parameters

value	A list of global variables being modified; each element within this list should be a 2×1 string array {varName, varValue}, where varName is the name of the target variable and varValue is the value being assigned.

Return

uint	The error code that represents the result of the function calling.

3.14.2 ChangeProjectVariableValue

Syntax

```
uint ChangeProjectVariableValue(
    List<string[]> value
)
```

Description

Sets the initial value of a specific project variable.

Parameters

value	A list of Project Variables being modified; each element within this list should be a 2×1 string array {varName, varValue}, where varName is the name of the target variable while varValue is the value being assigned.

Return

uint	The error code that represents the result of the function calling.

3.14.3 CreateGlobalVariable

Syntax

```
uint CreateGlobalVariable(
    string name,
    VariableType type,
    string value
)
```

Description

Creates a new global variable by the input parameters.

Parameters

name	Name of the variable being created.
type	Type of variable being created.
value	Value being assigned to the new variable.

Return

uint	The error code that represents the result of the function calling.

3.14.4 CreateProjectVariable

Syntax

```
uint CreateProjectVariable(
    string name,
    VariableType type,
    string value
)
```

Description

Creates a new Project Variable to the current project by the input parameters.

Parameters

name	Name of the variable being created.
type	Type of variable being created.
value	Value being assigned to the new variable.

Return

uint	The error code that represents the result of the function calling.

3.14.5 DeleteGlobalVariable

Syntax

```
uint DeleteGlobalVariable(
    string name
)
```

Description

Deletes a specific global variable from the robot.

Parameters

name	Name of the global variable being deleted.

Return

uint	The error code that represents the result of the function calling.

3.14.6 DeleteProjectVariable

Syntax

```
uint DeleteProjectVariable(
    string name
)
```

Description

Deletes a specific Project Variable from the current TMflow project.

Parameters

name	Name of the Project Variable being deleted.

Return

uint	The error code that represents the result of the function calling.

3.14.7 GetGlobalVariableList

Syntax

```
uint GetGlobalVariableList(
    ref List<VariableInfo>variables
)
```

Description

Gets all Global Variables (VariableInfo Type) from the robot and overwrites the input List.

Parameters

variables	A List of Variable Info type that contains all global variables within the robot.

Return

uint	The error code that represents the result of the function calling.

3.14.8 GetProjectVariableList

Syntax

```
uint GetProjectVariableList(
    ref List<VariableInfo> variables
)
```

Description

Gets all Project Variables (VariableInfo Type) from the current TMflow Project and overwrites the input List.

Parameters

variables	A List of VariableInfo type that contains all Project Variables within the current Project.

Return

uint	The error code that represents the result of the function calling.

3.14.9 IsGlobalVariableExist

Syntax

```
bool IsGlobalVariableExist(
    string varName
)
```

Description

Check if a specific Global Variable exists or not.

Parameters

varName	Name of the Global Variable being checked.

Return

bool	True if exists, false if not.

3.14.10 IsProjectVariableExist

Syntax

```
bool IsProjectVariableExist(
    string varName
)
```

Description

Check if a specific Project Variable exists or not.

Parameters

varName	Name of the Project Variable being checked.

Return

bool	True if exists, false if not.

4. Enumeration types

4.1 FreeBotMode

```
public enum FreeBotMode
{
    All_Joints,
    Custom,
    RXYZ,
    SCARA_Like,
    XYZ
}
```

Description

Enum FreeBotMode, which is used as a member of the class FreeBotInfo and represents the FreeBot mode setting.

Items

FreeBotMode.All_Joints	Represents free all joints mode.
FreeBotMode.Custom	Represents custom FreeBot mode.
FreeBotMode.RXYZ	Represents free RXYZ (Rx, Ry, Rz) mode.
FreeBotMode.SCARA_Like	Represents SCARA-like FreeBot mode.
FreeBotMode.XYZ	Represents free XYZ mode.

4.2 IO_TYPE

```
public enum IO_TYPE
{
    UNKNOWN,
    CONTROL_BOX,
    END_MODULE,
    EXT_MODULE
}
```

Description

Enum IO_TYPE, paired with IOProvider functions such as WriteDigitOutput(), defines the IO device within TM robot.

Items

IO_TYPE. UNKNOWN	Represents an unknown device detected. When using IOProvider.GetAllIOData(), if there is any unknown device detected, IO_TYPE.UNKNOWN will be found within the DeviceIOInfo data
IO_TYPE.CONTROL_BOX	Control Box I/O.
IO_TYPE. END_MODULE	End Module I/O (Tool End I/O Interface).
IO_TYPE. EXT_MODULE	External I/O Device(s) connected to the robot.

4.3 MoveMode

```
public enum MoveMode
{
    Accurate,
    Fast,
    Nromal
}
```

Description

Enum MoveMode,which is used as one of the parameter of the class FreeBotInfo. Move Mode is for users to adjust the initial damping of joints with modes of Accurate, Normal, and Fast. Damping increases the hand guide weight allowing faster stoppage while releasing the FREE button. For easier dragging, joint damping decreases proportionally as TCP speed increases during the hand guide. Once damping drops to zero, it stays at zero until the FREE button is released

Items

MoveMode.Accurate	The highest joint damping. For the high initial force requirement with fast stoppage while releasing the FREE button.
MoveMode.Fast	The zero joint damping. For the low initial force requirement for dragging.
MoveMode.Normal	The low joint damping. For the medium initial force requirement with reasonable accuracy while stopping.

4.4 RobotEventType

```
public enum RobotEventType
{
    EndButtonFreeBotChanged,
    EndButtonGripperChanged,
    EndButtonPointChanged,
    EndButtonVisionChanged
}
```

Description

Enum RobotEventType, paired with RobotStatusProvider's event EndButtonClickEvent, defines the click event occurred on the buttons of the End Module.

Items

EndButtonFreeBotChanged	Represents the click event of the Free Button on the End Module. True denotes FreeBot is triggered while False denotes that the Free Button is either released or over-pressed.
EndButtonGripperChanged	Represents the click event of the Gripper Button on the End Module. True denotes the button is pressed while False denotes that pressing is released.

EndButtonPointChanged	Represents the click event of the Point Button on the End Module. True denotes the button is pressed while False denotes that pressing is released.
EndButtonVisionChanged	Represents the click event of the Vision Button on the End Module. True denotes the button is pressed while False denotes that pressing is released.

4.5 TMcraftErr

```
public enum TMcraftErr
{
    ConnectionFail,
    DevResponseError,
    ExceptionError
    InvalidParameter,
    NodeCloseFail,
    OK
}
```

Description

Enum TMcraftErr represents the possible error that may occurred not from TMflow, but TMcraft API itself. TMcraftErr is used as the object type returned by the functions TMcraftSetupAPI.GetErrMsg and TMcraftSetupAPI.InitialTMcraftSetup.

Items

TMcraftErr.ConnectionFail	TMcraft API failed to connect with TMflow.
TMcraftErr.DevResponseError	Unexpected error on TMcraft API. Please contact Techman Inc. for further analysis.
TMcraftErr.ExceptionError	Exception happended on TMCraft API. Please contact Techman Inc. for further analysis.
TMcraftErr.InvalidParameter	TMcraft API detects invalid parameters when calling provider functions. For example, empty string or incorrect array size.
TMcraftErr.NodeCloseFail	Failure happened when closing TMcraft Node on TMflow.
TMcraftErr.OK	No error.

4.6 TMflowType

```
public enum TMflowType
{
    AOIEdge,
    Client,
    OLP,
    Robot,
    Unknown
}
```

Description

Enum TMflowType, which is the Outputs of SystemProvider.GetTMflowType and represent the TMflow type of the current system, or more specifically, of where the GetTMflowType function is called.

Items

TMflowType.AOIEdge	Represents that the current system is AOI Edge.
TMflowType.Client	Represents that the current system is client TMflow.
TMflowType.OLP	Represents that the current system is TMstudio Pro.
TMflowType.Robot	Represents that the current system is on the robot.
TMflowType.Unknown	Represents that the current system is not recognizable as one of the TMflow type.

4.7 VariableType

```
public enum VariableType
{
    Integer,
    Float,
    Double,
    String,
    Byte,
    Boolean,
    IntegrArray,
    FloatArray,
    DoubleArray,
    StringArray,
    ByteArray,
    BooleanArray,
    Null
}
```

Description

Enum VariableType, paired with VariableProvider function CreateGlobalVariable(), defines variable types on TMflow.

5. Additional class

5.1 BaseInfo

```
public class BaseInfo
{
    public string baseData;
    public string baseName;
    public string number;
    public string baseType;
}
```

Description

BaseInfo, which describes the information of a base, is the element typeof the output List of BaseProvider.GetBaseList().

Members

baseData	A 6×1 float array, {x, y, z, rx, ry, rz} that defines the base.
baseName	Name of the base.
number	The serial number of the base within its base type; the robot base is always 0, while the other base types always start from 1.
baseType	Type of the base, such as R (Robot Base), C (Custom Base) and V (Vision Base).

5.2 DeviceIOInfo

```
public class DeviceIOInfo
{
    public IO_TYPE type;
    public int deviceSerialNum;
    public List<DigitIOInfo> DICollection;
    public List<DigitIOInfo> DOCollection;
    public List<float> AOCollection;
    public List<float> AICollection;
}
```

Description

The DeviceIOInfo describes all sorts of information related to a specific IO Device of the robot.

Members

Type	IO device that this information describes.
deviceSerialNum	Device serial number, which always starts from 0 and is more meaningful if the target device is an external IO module because there might be multiple external IO module devices within the system. The number is 0 if the target device is the Control box IO board or end module IO board because there is always one Control box IO board and one end module IO board.

DICollection	A List of DigitIOInfo Type, which represents all Digital Inputs within the IO Device and should be empty if there are no Digital Inputs. Please note that the index of the list represents the channel number.
DOCollection	A List of DigitIOInfo Type that represents all Digital Outputs within the IO Device and should be empty if there are no Digital Ouputs. Please note that the index of the list represents the channel number.
AOCollection	A List of float Type that represents all Analog Outputs within the IO Device and should be empty if there are no Analog Ouputs. Please note that the index of the list represents the channel number.
AICollection	A List of float Typethat represents all Analog Inputs within the IO Device and should be empty if there are no Analog Inputs. Please note that the index of the list represents the channel number.

5.3 DigitIOInfo

```
public class DigitIOInfo
{
    public bool value;
    public bool isUserDefined;
}
```

Description

DigitIOInfo describes the information of a Digital I/O channel which is used as the List data type of DeviceIOinfo.DICollection and DeviceIOInfo.DOCollection.

Members

value	True denotes HIGH while false denotes LOW.
isUserDefined	True denotes this Digital Channel is set as a User-Defined IO (that triggers a signal to a button of the Robot Stick, reads the signal from a stick button, or detects if an error occurs in the system).

5.4 ErrorStatus

```
public class ErrorStatus
{
    public uint Error_Code;
    public uint[] Error_Codes;
    public string Error_Time;
    public uint Last_Error_Code;
    public uint[] Last_Error_Codes;
    public uint Last_Error_Time;
}
```

Description

ErrorStatus denotes the structure of the data return by RobotStatusProvider.ErrorEvent. Note that the ErrorEvent does not return this object type directly, but a json string instead that can be conveted to the ErrorStatus type.

Members

Error_Code	The major error code of the current error event, which should be the first item of Error_Codes, i.e. Error_Codes[0]. Note that Error_Code would be cleared after reset.
Error_Codes	All error codes related to the current error event. Note that Error_Codes would be cleared after reset.
Error_Time	Time stamp of Error_Code.
Last_Error_Code	The major error code of the last error event recorded, which should be the first item of Last_Error_Codes, i.e. Last_Error_Codes[0]. Note that Last_Error_Code would not be cleared after reset, but would be refreshed when another error event happens.
Last_Error_Codes	All error codes related to the last error event. Note that Last_Error_Codes would not be cleared after reset, but would be refreshed when another error event happens.
Last_Error_Time	Time stamp of Last_Error_Code.

5.5 FreeBotInfo

```
public class FreeBotInfo
{
    public FreeBotMode Mode;
    public bool isBaseMode;
    public bool isFreeX;
    public bool isFreeY;
    public bool isFreeZ;
    public bool isFreeRX;
    public bool isFreeRY;
    public bool isFreeRZ;
}
```

Description

FreeBotInfo describes the information of FreeRobot Configuration PointInfo and applies to 2 of the RobotStatusProvider functions, GetFreeBot() and SetFreeBot. Note that if the Mode is not Custom, the rest of the members is meaningless.

Members

Mode	Represents the current Freebot mode.
isBaseMode	True means FreeBot Custom settings being defined by the current base; false means FreeBot Custom settings being defined by the current tool base.
isFreeX	Represents if the current FreeBot Custom Setting has freed X axis or not.
isFreeY	Represents if the current FreeBot Custom Setting has freed Y axis or not.
isFreeZ	Represents if the current FreeBot Custom Setting has freed Z axis or not.

isFreeRX	Represents if the current FreeBot Custom Setting has freed Rx axis or not.
isFreeRY	Represents if the current FreeBot Custom Setting has freed Ry axis or not.
isFreeRZ	Represents if the current FreeBot Custom Setting has freed Rz axis or not.

5.6 PointInfo

```
public class PointInfo
{
    public string baseName;
    public string flangeCoordinate;
    public string jointAngles;
    public string pointName;
    public string toolName;
    public string endToolCoordinate;
    public string pointType;
}
```

Description

PointInfo, which describes the information of a Point (robot pose) within the current Project, is the element type of the output List of PointProvider.GetBaseList(). Note that a robot pose can be defined by three kinds of coordinates: flange coordinates, joint angles and tool coordinates.

Members

baseName	The base that defines this point (robot pose).
flangeCoordinate	Flange Coordinates that defines this point (robot pose).
jointAngles	Joint Angles that defines this point (robot pose).
pointName	Name of the point.
robotModel	Robot Model of the robot, from which this point is built.
toolName	Tool that defines the tool coordinates of this point.
endToolCoordinate	Tool coordinates of this robot pose.
pointType	There are two possible point types, R (Regular) and D (Dynamic). The Regular point generates with the Point node, and the Dynamic point, with the Touch Stop node.

5.7 TCPInfo

```
public class TCPInfo
{
    public float[] data;
    public string name;
}
```

Description

TCPInfo, which describes the basic information of a TCP, is the element type of the output List of TCPProvider.GetTcpList.().

Members

data	Tool Center Point, which defines a float[6] {x, y, z, Rx, Ry, Rz} relative to the Flange base.
name	Name of the TCP.

5.8 VariableInfo

```
public class VariableInfo
{
      public string varName;
      public VariableType varType;
      public string value;
      public bool isGlobal;
}
```

Description

VariableInfo, paired with VariableProvider functions such as GetGlobalVariableList(), describes all the information of a variable.

Members

varName	Name of the variable.
varType	Data type of the variable.
value	Value of the variable.
isGlobal	True if it is a global variable; false if it is a Project Variable.

TECHMAN ROBOT INC.