# TMcraft Shell API Function Manual

Original Instructions

Software version: 1.20.1000

Document verison: 1.0

Release date: 2024-11-01

TECHMAN ROBOT INC.
5F., No. 58-2, Huaya 2nd Rd., Guishan Dist., Taoyuan City, 333411 , Taiwan

## Table of Content





TECHMAN ROBOT INC.
5F., No. 58-2, Huaya 2nd Rd., Guishan Dist., Taoyuan City, 333411 , Taiwan

TECHMAN ROBOT INC.
5F., No. 58-2, Huaya 2nd Rd., Guishan Dist., Taoyuan City, 333411 , Taiwan

## 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<br>● [Add] class TMcraftToolbarAPI<br>● [Add] interface ITMcraftToolbarEntry<br>● [Add] class ErrorStatus<br>● [Add] FreeBotInfo.MoveMode<br>● [Add] class MoveMode<br>● [Add] class LogExportSetting<br>● [Add] RobotEventType.EndButtonFreeBotChanged |
| 1.18.1400 | 2024/6 | ● [Add] class TMcraftSetupAPI<br>● [Add] class TMcraftNodeAPI.TextfileProvider<br>● [Add] class TMcraftShellAPI.TextfileProvider<br>● [Add] class TMcraftToolbarAPI.TextfileProvider<br>● [Add] TMcraftShellAPI.ProjectRunProvider.GetProjectList<br>● [Add] TMcraftShellAPI.RobotStatusProvider.GetRobotName<br>● [Add] TMcraftNodeAPI.RobotStatusProvider.GetRobotModelType<br>● [Add] TMcraftNodeAPI.RobotStatusProvider.GetFlowVersion |
| 1.20.1100 | 2024/11 | ● [Add] TMcraftNodeType.dll<br>● [Add] class TMcraftNodeAPI.FreeBotProvider<br>● [Add] class TMcraftNodeAPI.EndButtonEventProvider<br>● [Deprecated] TMcraftNodeAPI.RobotStatusProvider.GetFreeBot<br>● [Deprecated] TMcraftNodeAPI.RobotStatusProvider.SetFreeBot<br>● [Deprecated] TMcraftNodeAPI.RobotStatusProvider.EndButtonClickEvent<br>● [Add] class TMcraftShellAPI.FreeBotProvider<br>● [Add] class TMcraftShellAPI.EndButtonEventProvider<br>● [Deprecated] TMcraftShellAPI.RobotStatusProvider.GetFreeBot<br>● [Deprecated] TMcraftShellAPI.RobotStatusProvider.SetFreeBot<br>● [Deprecated] TMcraftShellAPI.RobotStatusProvider.EndButtonClickEvent<br>● [Add] class TMcraftToolbarAPI.FreeBotProvider<br>● [Add] class TMcraftToolbarAPI.EndButtonEventProvider<br>● [Deprecated] TMcraftToolbarAPI.RobotStatusProvider.GetFreeBot<br>● [Deprecated] TMcraftToolbarAPI.RobotStatusProvider.SetFreeBot |

| | | |
|---|---|---|
| | | ● [Deprecated] TMcraftToolbarAPI.RobotStatusProvider.EndButton-ClickEvent<br>● [Add] class TMcraftSetupAPI.FreeBotProvider<br>● [Add] class TMcraftSetupAPI.EndButtonEventProvider<br>● [Deprecated] TMcraftSetupAPI.RobotStatusProvider.GetFreeBot<br>● [Deprecated] TMcraftSetupAPI.RobotStatusProvider.SetFreeBot<br>● [Deprecated] TMcraftSetupAPI.RobotStatusProvider.EndButtonClickEvent |

# 1. Overview

Developers can now developed a full-page customized GUI (Graphical User Interface) based on C#/WPF, which is overlay above TMflow; this kind of custom plugin is called TMcraft Shell. Since end-users might not interact much with TMflow, allowing developers fully shape the user experience. TMcraft Shell can serve as a setup wizard, a dashboard, or both. For instance, it's possible to create a Palletizing Operator with it.

- Wizard:

Using the Wizard UI, users can easily configure settings like box size, pallet size, and layer structure. Once confirmed, the TMcraft Shell program collects these parameters and creates the relevant script project.

- Dashboard:

As the application begins, the robot executes the previously mentioned scrip project. The TMcraft Shell program then retrieves various robot data or project variable values, displaying them on the Dashboard. Users can also view palletizing throughput matrixes like Units per Hour, Overall Equipment Efficiency, Defect Count, and Downtime.

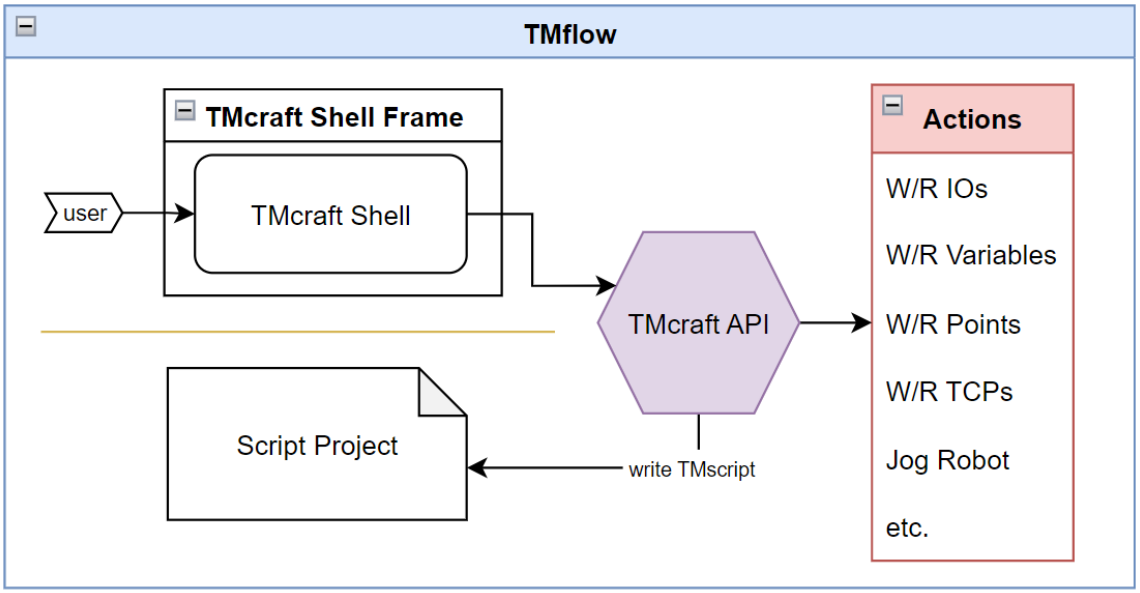


Figure 1: Architecture of TMcraft Shell

| 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) | ✓ | ✓ | ✓ | ✓ |

| capabilities \ Plugins | Node | Shell | Toolbar | Setup |
|---|---|---|---|---|
| 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

Developers must use the TMcraft Packer from the TMcraft Development Kit to package the TMcraft Shell exe file and its associated files before importing them into TMflow.

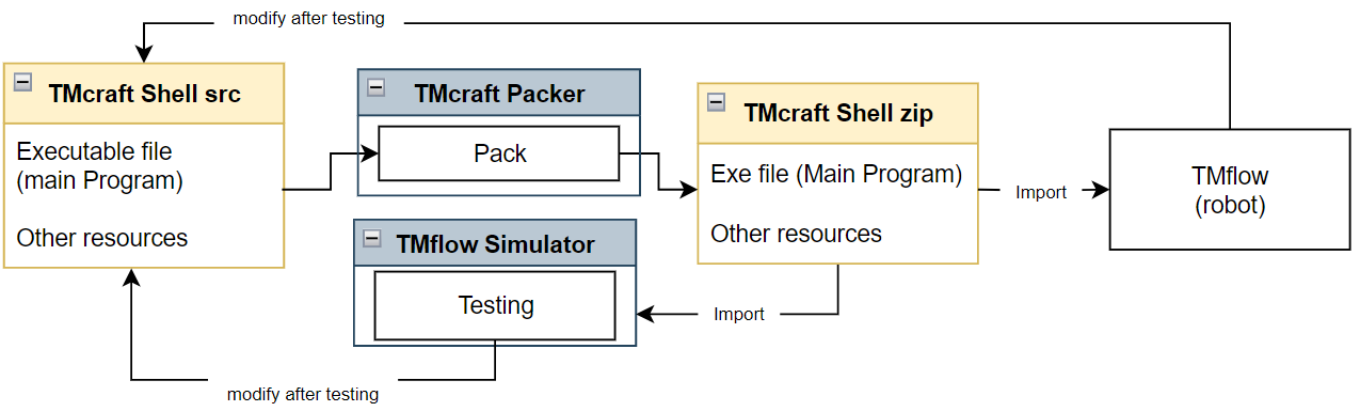


Figure 2: Development process of a TMcraft Shell

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

## 2. Programming with TMcraft Shell API

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

```
using TMcraft;

namespace TMcraftSample
{
    public partial class MainWindow : Window
    {
        TMcraftShellAPI ShellUI;

        public MainWindow ()
        {
            InitializeComponent();
        }

        private void Window_Loaded (object sender, RoutedEventArgs e)
        {
            if (ShellUI == null)
            {
                ShellUI = new TMcraftShellAPI();
                ShellUI.InitialTMcraftShell();
            }
        }

        private void Window_Unloaded (object sender, RoutedEventArgs e) {}

        private void Window_Closing (object sender, System.ComponentModel.CancelEven-
    tArgs e)
        {
            if (ShellUI != null) ShellUI.CloseShellConnection();
        }

        private void Btn_ShowTMflow_Click (object sender, RoutedEventArgs e)
        {
            //example : using TMcraft Shell API function to show TMflow and hide the Shell GUI
            if(ShellUI != null) ShellUI.SystemProvider.ShowTMflow();
        }
}
```

To create the foundation of a TMcraft Shell Program, please remind the following items:

1. Include TMcraft.dll as a reference. Rememeber to import the namespace (using TMcraft) onto

the program.

2. Declare a global object based on the class TMcraftShellAPI.
3. When loading the window of the Shell GUI, assign the object (new TMcraftShellAPI()), then call the function InitialTMcraftShell().
4. When closing the window of the Shell GUI, remember to call the function ShellUI.CloseShell-Connection().

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

# 3. TMcraft API functions (Shell related)

## 3.1 TMcraftShellAPI

TMcraft.dll is a combination of the APIs of all sort of TMcraft items; for TMcraft Shell, please declare an object of the class *TMcraftShellAPI* and use the function within. Like other TMcraft API, *TMcraftShellAPI* contains different members (or providers) functions in order to interact with TMflow, such as creating Project variables or jogging 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 Shell, the unsupported functions include:

- RobotJogProvider: all functions
- RobotStatusProvider: all functions, except GetFlowVersion, GetOperationMode, ProjectEditOrNot, ProjectPauseOrNot, ProjectRunOrNot, and ErrorEvent
- RobotStickProvider: all functions
- SystemProvider: ExportTCP
- TCPProvider: all functions
- Enumeration types: FreeBotMode, MoveMode, RobotEventType, VirtualKeyEvent
- Additional class: FreeBotInfo, TCPInfo

### 3.1.1 Version

**Syntax**

string TMcraftShellAPI.Version

**Description**

A member of the TMcraftShellAPI 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 CloseShellConnection

**Syntax**

void CloseShellConnection ()

**Description**

Closes the connection between TMcraft Shell and TMflow.

**Parameters**

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

**Return**

None.

### 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. |

### 3.1.4 InitialTMcraftShell

**Syntax**

```
TMcraft.TMcraftErr InitialTMcraftShell()
```

**Description**

Start the connection between TMcraft Shell and TMflow.

**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. |

## 3.2 EndButtonEventProvider

EndButtonEventProvider contains functions related to the end button event.

### 3.2.1 HasEndButtonEventOwnership

**Syntax**

```
uint HasEndButtonEventOwnership()
```

**Description**

TMcraft plugin can call this function to check if it has the end button event owner-

ship 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. |

### 3.2.2 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. |

### 3.2.3 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. |

### 3.2.4 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. |

### 3.2.5 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.

## 3.3 FreebotProvider

FreeBotProvider provides functions related to freebot.

### 3.3.1 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. |

### 3.3.2 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. |

### 3.3.3 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. |

### 3.3.4 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. |

## 3.4 IOProvider

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

### 3.4.1 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. |

### 3.4.2 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. |

### 3.4.3 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 Outputs 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 Outputs value, ranged from -10V to 10V. |

**Return**

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

### 3.4.4 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. |

### 3.4.5 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 Outputs 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 Outputs status, where bool true is HIGH and bool false is LOW. |

**Return**

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

### 3.4.6 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. |

### 3.4.7 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 Outputs 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 Outputs value, ranged from -10V to 10V. |

**Return**

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

### 3.4.8 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 Outputs 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 Outputs status, where bool true is HIGH and bool false is LOW. |

**Return**

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

## 3.5 ProjectRunProvider

ProjectRunProvider provides functions related to project run.

### 3.5.1 GetCurrentProject

**Syntax**

```
uint GetCurrentProject(
    out string projectName
)
```

**Description**

Get the name of the current project.

**Parameters**

| | |
|---|---|
| projectName | Name of the project created. |

**Return**

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

### 3.5.2 GetDisplayBoardInfo

**Syntax**

```
uint GetDisplayBoardInfo(
    out string info
)
```

**Description**

Outputs the content currently shown on TMflow Display Board.

**Parameters**

| | |
|---|---|
| info | Outputs the current Display Board Content with the following format: [{*Background color*},{*Font color*},{*Tilte*},{*Content*}]. |

**Return**

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

### 3.5.3 GetProjectList

**Syntax**

```
uint GetProjectList(
    out List<string> projectInfo
)
```

**Description**

Get the list of projects of the current system.

**Parameters**

| | |
|---|---|
| projectInfo | List of projects (names) of the current system. |

**Return**

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

### 3.5.4 PauseProject

**Syntax**

```
uint PauseProject()
```

**Description**

Pause the current running project. To continue the project run, use RunProject().

**Parameters**

No parameters are required.

**Return**

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

### 3.5.5 RunProject

**Syntax**

```
uint RunProject()
```

**Description**

Run the current project.

**Parameters**

No parameters are required.

**Return**

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

### 3.5.6 SetCurrentProject

**Syntax**

```
uint SetCurrentProject(
    string projectName
)
```

**Description**

Given a project name, set it as the current project. Note that this function can only be used during Auto mode.

**Parameters**

| | |
|---|---|
| projectName | Name of the project to be the current project. |

**Return**

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

### 3.5.7 StopProject

**Syntax**

```
uint StopProject()
```

**Description**

Stop the current project.

**Parameters**

No parameters are required.

**Return**

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

## 3.6 RobotJogProvider

RobotJogProvider provides functions for TMcraft item to jog the robot.

**IMPORTANT**:
If the TMcraft Shell uses any RobotJogProvider functions for motion control, it is the responsibility of the developer to make sure single point of control within ISO 10218-1.

### 3.6.1 HoldPlayKeyToRun

**Syntax**

```
uint HoldPlayKeyToRun(
    bool holdKey
)
```

**Description**

This function mimics the process of holding play key to start jogging motion. Note that, at the same time, KeepJogging() should be sent every 100 - 500 ms until the jogging ends, otherwise, the jogging will stop automatically.

**Parameters**

| | |
|---|---|
| holdKey | True means to hold the key; false means to release the key. |

### 3.6.2 JogByBase

**Syntax**

```
uint JogByBase(
    float speedPercentage,
    float [] targetCoordinates
)
```

**Description**

Jogs the robot towards the target's Coordinates (relative to current base and tool) with a 6×1 float array {x, y, z, rx, ry ,rz}. This function will not trigger any motion directly, as it requires following by either pressing the PLAY button on the robot stick (may also requires Enabling Switch) or using API functions: HoldPlayKeyToRun + KeepJogging.

**Parameters**

| | |
|---|---|
| speedPercentage | Speed percentage is equivalent to the speed (in percentage) setting on the TMflow Controller, where the current jogging speed should match the max joint speed. The max joint speed of the robot model is multiplied by the speed percentage, and the product (TCP speed) of this multiplication should always be lower than Manual Control mode speed limit (250 mm/s). speedPercentage is expressed in decimals (e.g., 1.5 for 1.5%).. |
| targetMovementValue | A 6×1 float array {x, y, z, rx, ry ,rz} of target movement value. |

**Return**

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

### 3.6.3 JogByJoint

**Syntax**

```
uint JogByJoint(
    float speedPercentage,
    float[] targetJointAngles
)
```

**Description**

Jogs the robot towards the targets Joint Angles. This function will not trigger any motion directly, as it requires following by either pressing the PLAY button on the robot stick (may also requires Enabling Switch) or using API functions: HoldPlayKeyToRun + KeepJogging.

**Parameters**

| | |
|---|---|
| speedPercentage | Speed percentage is equivalent to the speed (in percentage) setting on the TMflow Controller, where the current jogging speed should match the max joint speed. The max joint speed of the robot model is multiplied by the speed percentage, and the product (TCP speed) of this multiplication should always be lower than Manual Control mode speed limit (250 mm/s). speedPercentage is expressed in decimals (e.g., 1.5 for 1.5%). |
| targetJointAngles | A 6×1 float array {J1, J2, J3, J4, J5, J6} which represents the target Joint Angle. |

**Return**

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

### 3.6.4 JogCircle

**Syntax**

```
uint JogCircle(
    float speedPercentage,
    float[] passPoint,
    float[] endPoint,
    int targetAngle
)
```

**Description**

Jogs the end-effector circularly. Like circle node, the circue is always defined by 3 points: the initial position of the end-effector, a pass point and an end point. This function will not trigger any motion directly, as it requires following by either pressing the PLAY button on the robot stick (may also requires Enabling Switch) or using API functions: HoldPlayKeyToRun + KeepJogging.

**Parameters**

| | |
|---|---|
| speedPercentage | Speed percentage is equivalent to the speed (in percentage) setting on the TMflow Controller, where the current jogging speed should match the max joint speed. The max joint speed of the robot model is multiplied by the speed percentage, and the product (TCP speed) of this multiplication should always be lower than Manual Control mode speed limit (250 mm/s). speedPercentage is expressed in decimals (e.g., 1.5 for 1.5%). |
| passPoint | The 2nd point required to define the circle, described by Cartesian Space. |
| endpoint | The last point required to define the circle, described by Cartesian Space. |
| targetAngle | Define how much arc of the circle to be jogged. If the target angle is 0, the trajectory will end at the end point. |

**Return**

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

### 3.6.5 JogRelativeByBase

**Syntax**

```
uint JogRelativeByBase(
    float speedPercentage,
    float [] targetCoordinates
)
```

**Description**

Jogs the robot by relative motion according to the current base. This function will not trigger any motion directly, as it requires following by either pressing the PLAY button on the robot stick (may also requires Enabling Switch) or using API functions: HoldPlayKeyToRun + KeepJogging.

**Parameters**

| | |
|---|---|
| speedPercentage | Speed percentage is equivalent to the speed (in percentage) setting on the TMflow Controller, where the current jogging |

| | |
|---|---|
| | speed should match the max joint speed. The max joint speed of the robot model is multiplied by the speed percentage, and the product (TCP speed) of this multiplication should always be lower than Manual Control mode speed limit (250 mm/s). speedPercentage is expressed in decimals (e.g., 1.5 for 1.5%). |
| targetCoordinates | A 6×1 float array {x, y, z, rx, ry ,rz} of target movement value. |

**Return**

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

### 3.6.6 JogRelativeByJoint

**Syntax**

```
uint JogRelativeByJoint(
    float speedPercentage,
    float [] targetJointAngles
)
```

**Description**

Jogs the robot relatively by given joint angles along with Tool Axes. This function will not trigger any motion directly, as it requires following by either pressing the PLAY button on the robot stick (may also requires Enabling Switch) or using API functions: HoldPlayKeyToRun + KeepJogging.

**Parameters**

| | |
|---|---|
| speedPercentage | Speed percentage is equivalent to the speed (in percentage) setting on the TMflow Controller, where the current jogging speed should match the max joint speed. The max joint speed of the robot model is multiplied by the speed percentage, and the product (TCP speed) of this multiplication should always be lower than Manual Control mode speed limit (250 mm/s). speedPercentage is expressed in decimals (e.g., 1.5 for 1.5%). |
| targetJointAngles | A 6×1 float array {J1, J2, J3, J4, J5, J6} of target movement value. |

**Return**

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

### 3.6.7 JogRelativeByTool

**Syntax**

```
uint JogRelativeByTool(
    float speedPercentage,
    float [] targetMovementValue
)
```

**Description**

Jogs the robot along with Tool Axes. Remind that, like using TMflow Controller, users need to use the robot stick (e.g. Enabling Device + PLAY) to start the motion.

**Parameters**

| | |
|---|---|
| speedPercentage | Speed percentage is equivalent to the speed (in percentage) setting on the TMflow Controller, where the current jogging speed should match the max joint speed. The max joint speed of the robot model is multiplied by the speed percentage, and the product (TCP speed) of this multiplication should always be lower than Manual Control mode speed limit (250 mm/s). speedPercentage is expressed in decimals (e.g., 1.5 for 1.5%). |
| targetMovementValue | A 6×1 float array {x, y, z, rx, ry ,rz} of target movement value. |

**Return**

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

### 3.6.8 KeepJogging

**Syntax**

```
uint KeepJogging()
```

**Description**

Keep the current jogging. After sending HoldPlayKeyToRun, this function should be keep sending every 100 - 500 ms until the jogging ends, otherwise, the jogging will stop automatically.

**Parameters**

No parameters are required.

**Return**

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

### 3.6.9 StopJog

**Syntax**

```
uint StopJog()
```

**Description**

Stops all Jog motion immediately. It is also recommended to call this function before calling Jog motion functions in order to clear the motion buffer

**Parameters**

No parameters are required.

**Return**

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

## 3.7 RobotStatusProvider

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

### 3.7.1 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. |

### 3.7.2 GetCurrentPayload

**Syntax**

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

**Description**

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

**Parameters**

| | |
|---|---|
| payload | Current payload value being assigned. |

**Return**

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

### 3.7.3 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. |

### 3.7.4 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. |

### 3.7.5 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. |

### 3.7.6 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:<br>int[0]: 0 – Right Arm, 1 – Left Arm<br>int[1]: 2 – Above Elbow, 3 – Below Elbow<br>int[2]: 4 – Up Wrist, 5 – Down Wrist |

**Return**

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

### 3.7.7 GetCurrentSpeedPercentage

**Syntax**

```
uint GetCurrentSpeedPercentage(
    out int speedPercentage
)
```

**Description**

Gets current speed percentage setting.

**Parameters**

| | |
|---|---|
| speedPercentage | Current speed percentage setting. |

**Return**

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

### 3.7.8 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. |

### 3.7.9 GetCurrentToolSpeed

**Syntax**

```
uint GetCurrentToolSpeed(
    out string speed
)
```

**Description**

Gets the current tool speed.

**Parameters**

| | |
|---|---|
| speed | Current Tool speed. |

**Return**

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

### 3.7.10 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. |

### 3.7.11 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. |

### 3.7.12 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. |

### 3.7.13 GetRobotName

**Syntax**

```
uint GetRobotName(
    out string robotName
)
```

**Description**

Gets the name of the robot

**Parameters**

| | |
|---|---|
| robotName | Name of the robot. |

**Return**

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

### 3.7.14 ProjectEditOrNot

**Syntax**

```
uint ProjectEditOrNot(
    out bool result
)
```

**Description**

Outputs if any project is under editing or not.

**Parameters**

| | |
|---|---|
| result | If any project is under editing or not. |

**Return**

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

### 3.7.15 ProjectPauseOrNot

**Syntax**

```
uint ProjectPauseOrNot(
    out bool result
)
```

**Description**

Outputs if the current project is paused or not.

**Parameters**

| | |
|---|---|
| result | If the current project is paused or not. It would be True only if there is a running project and it is in pause status. |

**Return**

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

### 3.7.16 ProjectRunOrNot

**Syntax**

```
uint ProjectRunOrNot(
    out bool result
)
```

**Description**

Outputs if any project is running or not.

**Parameters**

| | |
|---|---|
| result | If any project is running or not. |

**Return**

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

### 3.7.17 RobotErrorOrNot

**Syntax**

```
uint RobotErrorOrNot(
    out bool result
)
```

**Description**

Outputs if the robot is in error status or not.

**Parameters**

| | |
|---|---|
| result | If the robot is in error status or not. |

**Return**

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

### 3.7.18 RobotEstopOrNot

**Syntax**

```
uint RobotEstopOrNot(
    out bool result
)
```

**Description**

Outputs if the robot is under Estop status or not.

**Parameters**

| | |
|---|---|
| result | If the robot is under editing or not. |

**Return**

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

### 3.7.19 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. |

### 3.7.20 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. |

### 3.7.21 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. |

### 3.7.22 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.

## 3.8 RobotStickProvider

RobotStickProvider provides functions for developing a virtual robot stick on TMcraft Shell.

### 3.8.1 RobotStickStatus

**Syntax**

```
uint RobotStickStatus(
    out bool status
)
```

**Description**

Denotes if the robot stick is either in local control or remote control. For more details, please refer to the Safety Manual.

**Parameters**

| | |
|---|---|
| status | True means robot stick is in local control; false means robot stick is in remote control. |

**Return**

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

### 3.8.2 RobotVirtualStickKeyEvent

**Syntax**

```
uint RobotVirtualStickEvent(
    TMcraft.VirtualKeyEvent event
)
```

**Description**

Trigger a robot stick signal according to the input parameter.

**Parameters**

| | |
|---|---|
| event | Robot stick signal. For more detail, please check the TMcraft enum VirtualKeyEvent. |

**Return**

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

## 3.9 ScriptProjectProvider

ScriptProjectProvider provides functions for managing script projects, including creating new projects, retrieving existing ones, listing projects, and writing script projects.

### 3.9.1 DeleteScriptProject

**Syntax**

```
uint DeleteScriptProject(
    string projectName
)
```

**Description**

Delete a specific script project.

**Parameters**

| | |
|---|---|
| projectName | Name of the project to be deleted. |

**Return**

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

### 3.9.2 GetScriptProjectList

**Syntax**

```
uint GetScriptProjectList(
    ref List<string[]> projectsInfo
)
```

**Description**

Get the list of script project, along with information.

**Parameters**

| | |
|---|---|
| projectsInfo | A list of string arrays represens all script projects within the robot. Each array contains the following information of a script porject: [*Project Name, Build date, Last updated date, Initial speed percentage, Project Type, Last execution date*] . |

**Return**

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

### 3.9.3 NewScriptProject

**Syntax**

```
uint NewScriptProject(
    string projectName
)
```

**Description**

Create a new script project.

**Parameters**

| | |
|---|---|
| projectName | Name of the project to be created. |

**Return**

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

### 3.9.4 OpenScriptProject

**Syntax**

```
uint OpenScriptProject (
    string projectName
)
```

**Description**

Open a specific script project

**Parameters**

| | |
|---|---|
| projectName | Name of the project to be opened. |

**Return**

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

### 3.9.5 ReadScriptProjectContent

**Syntax**

```
uint ReadScriptProjectContent(
```

```
    out string projectContent
)
```

**Description**

Read the content of the opened script project.

**Parameters**

| | |
|---|---|
| projectContent | Content of the opened script project. |

**Return**

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

### 3.9.6 ReadScriptProjectRemark

**Syntax**

```
uint ReadScriptProjectRemark(
    string projectName,
    out string projectRemark
)
```

**Description**

Read the remark of a specific script project. Note that it is available to read or write the remark only if that project is not opened currently. Project remark is a kind of editable information stored within a script project.

**Parameters**

| | |
|---|---|
| projectName | Name of the script project to be read. |
| projectRemark | Project remark Outputs as a string. |

**Return**

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

### 3.9.7 SaveScriptProject

**Syntax**

```
uint SaveScriptProject(
    string projectName
)
```

**Description**

Save the opened script project with the given project name.

**Parameters**

| | |
|---|---|
| projectName | Name of the project to be saved. |

**Return**

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

### 3.9.8 WriteScriptProjectContent

**Syntax**

```
uint WriteScriptProjectContent(
    string projectContent
)
```

**Description**

Replace the content of the opened script project with a new one.

**Parameters**

| | |
|---|---|
| projectContent | Content to be written onto the opened script project. |

**Return**

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

### 3.9.9 WriteScriptProjectRemark

**Syntax**

```
uint WriteScriptProjectRemark(
    string projectName,
    string projectRemark
)
```

**Description**

Replace the remark of a specific script project with the new one. Note that it is available to read or write the remark only if that project is not opened currently. Project remark is a kind of editable information stored within a script project.

**Parameters**

| | |
|---|---|
| projectName | Name of the target script project. |
| projectRemark | Project remark Outputs as a string. |

**Return**

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

## 3.10 SystemProvider

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

### 3.10.1 ExportGlobalVariable

**Syntax**

```
uint ExportGlobalVariable(
```

```
    string varName
)
```

**Description**

Export a specific global variable to the flash drive.

**Parameters**

| | |
|---|---|
| varName | Name of the globalr variable to be exported. |

**Return**

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

### 3.10.2 ExportLog

**Syntax**

```
uint ExportLog(
    TMcraft.LogExportSetting logSetting
)
```

**Description**

Export the log of the robot to the flash drive.

**Parameters**

| | |
|---|---|
| logSetting | Log setting defined the amount of log to be exported. For more detail, please check the TMcraft enum: LogExportSetting. |

**Return**

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

### 3.10.3 ExportProject

**Syntax**

```
uint ExportProject(
    string projectName
)
```

**Description**

Export a specific project to the flash drive.

**Parameters**

| | |
|---|---|
| projectName | Name of the project to be exported. |

**Return**

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

### 3.10.4 ExportTCP

**Syntax**

```
uint ExportTCP(
    string tcpName
)
```

**Description**

Export a specific tcp to the flash drive.

**Parameters**

| | |
|---|---|
| tcpName | Name of the tcp to be exported. |

**Return**

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

### 3.10.5 GetControl

**Syntax**

```
uint GetControl(
    bool get
)
```

**Description**

Get or release control of the robot.

**Parameters**

| | |
|---|---|
| get | True means get control; false means release control. |

**Return**

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

### 3.10.6 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. |

### 3.10.7 GetDateTime

**Syntax**

```
uint GetDateTime(
    out System.DateTimeOffset dateTimeOffset
)
```

**Description**

Get the date and time of TMflow system.

**Parameters**

| | |
|---|---|
| dateTimeOffset | Outputs a DataTimeOffset type, represents the current date and time of the TMflow system |

**Return**

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

### 3.10.8 GetSupportTimeZoneList

**Syntax**

```
uint GetSupportTimeZoneList(
    out List<System.TimeZoneInfo> timeZoneInfos
)
```

**Description**

Get the time zone list of TMflow system.

**Parameters**

| | |
|---|---|
| dateTimeOffset | Outputs a List of TimeZoneInfo type, represents the time zones supported by the TMflow system. |

**Return**

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

### 3.10.9 GetTimeZone

**Syntax**

```
uint GetTimeZone(
    out string ID,
    out bool IsAutoAdjustDST,
)
```

**Description**

Get the date and time of TMflow system.

**Parameters**

| | |
|---|---|
| ID | Time zone identifier |
| IsAutoAdjustDST | True means TMflow will automatically adjust the clock for daylight saving changes. Note that this parameter is significant if and only if that time zones support daylight saving time. |

**Return**

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

### 3.10.10 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. |

### 3.10.11 ImportGlobalVariable

**Syntax**

```
uint ImportGlobalVariable(
    string robotName,
    string varName
)
```

**Description**

Import a specific global variable to the robot.

**Parameters**

| | |
|---|---|
| robotName | Name of the folder where the system can find the item to be imported. |
| varName | Name of the global variable to be imported. |

**Return**

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

### 3.10.12 ImportProject

**Syntax**

```
uint ImportProject(
    string robotName,
```

```
    string projectName
)
```

**Description**

Import a specific project to the robot

**Parameters**

| | |
|---|---|
| robotName | Name of the folder where the system can find the item to be imported. |
| projectName | Name of the project to be imported. |

**Return**

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

### 3.10.13 ImportTCP

**Syntax**

```
uint ImportTCP(
    string robotName,
    string tcpName
)
```

**Description**

Import a specific tcp to the robot.

**Parameters**

| | |
|---|---|
| robotName | Name of the folder where the system can find the item to be imported. |
| tcpName | Name of the tcp to be imported. |

**Return**

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

### 3.10.14 LogIn

**Syntax**

```
uint LogIn(
    string userName,
    string password
)
```

**Description**

Login with the give user name and password.

**Parameters**

| | |
|---|---|
| userName | User name to login the robot. |
| password | Password that belongs to the user name. |

**Return**

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

### 3.10.15 LogOut

**Syntax**

```
uint LogOut()
```

**Description**

Logout the robot.

**Parameters**

No parameters are required.

**Return**

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

### 3.10.16 SetDateTime

**Syntax**

```
uint SetDateTime(
    unit year,
    unit month,
    unit day,
    unit hour,
    unit minute,
    unit second
)
```

**Description**

Set the date and time of TMflow system.

**Parameters**

| | |
|---|---|
| year | Year |
| month | Month |
| day | Day |
| hour | Hour |
| minute | Minute |
| second | Second |

**Return**

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

### 3.10.17 SetTimeZone

**Syntax**

```
uint SetTimeZone(
```

```
    string ID,
    bool IsAutoAdjustDST
)
```

**Description**

Set the date and time of TMflow system.

**Parameters**

| | |
|---|---|
| ID | Time zone identifier |
| IsAutoAdjustDST | True means TMflow will automatically adjust the clock for daylight saving changes. Note that this parameter is significant if and only if that time zones support daylight saving time. |

**Return**

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

### 3.10.18 ShowTMflow

**Syntax**

```
uint ShowTMflow()
```

**Description**

Hide TMcraft Shell and show TMflow GUI temporary. User can click the Back to Shell Button on TMflow (left upper corner) and back to TMcraft Shell.

**Parameters**

No parameters are required.

**Return**

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

### 3.10.19 Shutdown

**Syntax**

```
uint Shutdown()
```

**Description**

Shut down the robot.

**Parameters**

No parameters are required.

**Return**

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

## 3.11 TCPProvider

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

### 3.11.1 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. |

### 3.11.2 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. |

### 3.11.3 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. |

### 3.11.4 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. |

### 3.11.5 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. |

### 3.11.6 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. |

### 3.11.7 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. |

### 3.11.8 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. |

### 3.11.9 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. |

### 3.11.10 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. |

### 3.11.11 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. |

### 3.11.12 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. |

## 3.12 TextFileProvider

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

### 3.12.1 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. |

### 3.12.2 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. |

### 3.12.3 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. |

### 3.12.4 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. |

### 3.12.5 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. |

### 3.12.6 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. |

### 3.12.7 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. |

## 3.13 VariableProvider

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

### 3.13.1 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. |

### 3.13.2 ChangeVariableRuntimeValue

**Syntax**

```
uint ChangeVariableRuntimeValue(
    string varName,
    strin value
)
```

**Description**

Changes the runtime value of a specific variable.

**Parameters**

| | |
|---|---|
| varName | Represents the name of the variable to be changed. |
| value | Represents the value to be assigned. |

**Return**

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

### 3.13.3 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. |

### 3.13.4 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. |

### 3.13.5 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. |

### 3.13.6 GetGlobalVariableValue

**Syntax**

```
uint GetGlobalVariableValue(
    string varName,
    out string value
)
```

**Description**

Gets the value of a specfici global variable.

**Parameters**

| | |
|---|---|
| varName | Represents the name of the target global variable. |
| value | Outputs the value of {varName} |

**Return**

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

### 3.13.7 GetVariableRuntimeValue

**Syntax**

```
uint GetVariableRuntimeValue(
    string varName,
    out string value
)
```

**Description**

Gets the runtime value of a specific variable.

**Parameters**

| | |
|---|---|
| varName | Represents the name of the target variable. |
| value | Outputs the value of {varName} |

**Return**

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

### 3.13.8 IsGlobalVariableExist

**Syntax**

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

**Description**

Checks 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. |

# 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 TMcraftShellAPI.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 TMcraftShellAPI.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 LogExportSetting

```
public enum LogExportSetting
{
    Today,
    _3days,
    _7days,
    14_days,
    All
}
```

**Description**

Enum LogExportSetting,which is used as the input parameter of the function TMcraftShellAPI.SystemProvider.ExportLog, represents the amount of log to be exported.

**Items**

| | |
|---|---|
| LogExportSetting.Today | Represents to export the log for today. |
| LogExportSetting._3days | Represents to export the log for the past 3 days. |
| LogExportSetting._7days | Represents to export the log for the past 7 days. |
| LogExportSetting._14days | Represents to export the log for the past 14 days. |
| LogExportSetting.All | Represents to export all log within the robot. |

## 4.4 MoveMode

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

**Description**

Enum MoveMode,which is used as one of the parameter of the class TMcraftShellAPI.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. |

## 4.5 RobotEventType

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

**Description**

Enum RobotEventType, paired with TMcraftShellAPI.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. |

## 4.6 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 TMcraftShellAPI.GetErrMsg and TMcraftShellAPI.InitialTMcraftShell.

**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. |

## 4.7 TMflowType

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

**Description**

Enum TMflowType, which is the Outputs of TMcraftShellAPI.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. |

## 4.8 VariableType

```
public enum VariableType
{
    Integer,
    Float,
    Double,
    String,
    Byte,
    Boolean,
    IntegrArray,
    FloatArray,
    DoubleArray,
```

```
    StringArray,
    ByteArray,
    BooleanArray,
    Null
}
```

**Description**

Enum VariableType, paired with TMcraftShellAPI.VariableProvider function CreateGlobal-Variable(), defines variable types on TMflow.

## 4.9 VirtualKeyEvent

```
public enum VirtualKeyEvent
{
    MAKey,
    MALongKey,
    MinusKey,
    MinusLongKey,
    PauseLey,
    PauseLongKey,
    PlayKey,
    PlayLongKey,
    PlusKey,
    PlusLongKey,
    StopKey,
    StopLongKey
}
```

**Description**

Enum VirtualKeyEvent, which is used as the parameter of the function RobotStickPro-vider.RobotVirtualStickKeyEvent.

**Items**

| | |
|---|---|
| VirtualKeyEvent.MAKey | Represents a single press signal of MA (change mode) Button. |
| VirtualKeyEvent.MALongKey | Represents a long press signal of MA (change mode) Button. |
| VirtualKeyEvent.MinusKey | Represents a single press signal of Minus Button. |
| VirtualKeyEvent.MinusLongKey | Represents a long press signal of Minus Button. |
| VirtualKeyEvent.PauseKey | Represents a single press signal of Pause Button. |
| VirtualKeyEvent.PauseLongKey | Represents a long press signal of Pause Button. |
| VirtualKeyEvent.PlayKey | Represents a single press signal of Play Button. |
| VirtualKeyEvent.PlayLongKey | Represents a long press signal of Play Button. |
| VirtualKeyEvent.PlusKey | Represents a single press signal of Plus Button. |
| VirtualKeyEvent.PlusLongKey | Represents a long press signal of Plus Button. |
| VirtualKeyEvent.StopKey | Represents a single press signal of Stop Button. |
| VirtualKeyEvent.StopLongKey | Represents a long press signal of Stop Button. |

# 5. Additional class

## 5.1 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 TMcraftShellAPI.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. |

## 5.2 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 TMcraftShellAPI.DeviceIOinfo.DICollection and TMcraftShellAPI.DeviceIO-Info.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). |

## 5.3 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 TMcraftShellAPI.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. |

## 5.4 FreeBotInfo

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

**Description**

TMcraftShellAPI.FreeBotInfo is a TMcraft class that defines the FreeRobot configuration and is applied by 2 of the TMcraftShellAPI.RobotStatusProvider functions, GetFreeBot() and SetFreeBot(). Note that if the member, Mode, is not FreeBotMode.Custom, the rest of the members is meaningless.

**Members**

| | |
|---|---|
| Mode | Represents the FreeBot mode; for more detail, please check TMcraft enum FreeBotMode |
| MoveMode | Repressents the Move Mode setting of current Freebot; for more detail, please check TMcraft enum MoveMode. |
| 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 FreeBot Custom Setting has freed X axis or not. |
| isFreeY | Represents if the FreeBot Custom Setting has freed Y axis or not. |
| isFreeZ | Represents if the FreeBot Custom Setting has freed Z axis or not. |
| isFreeRX | Represents if the FreeBot Custom Setting has freed Rx axis or not. |
| isFreeRY | Represents if the FreeBot Custom Setting has freed Ry axis or not. |
| isFreeRZ | Represents if the FreeBot Custom Setting has freed Rz axis or not. |

## 5.5 TCPInfo

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

**Description**

TMcraftShellAPI.TCPInfo, which describes the basic information of a TCP, is the element type of the Outputs List of TMcraftShellAPI.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. |

## 5.6 VariableInfo

```
public class VariableInfo
{
    public string Name;
    public VariableType Type;
    public string value;
    public bool isGlobal;
}
```

**Description**

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

**embers**

| | |
|---|---|
| Name | Name of the variable. |
| ype | 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.