TMcraft Shell Tutorial Error Event

Original Instructions

Document version: 1.0

Release date: 2024-02-01

TECHMAN ROBOT INC.

Table of Contents

Revision History 3

1. Introduction..4

2. Concept..4

3. Sample Code 4

    3.1 MainWindow.xaml 4

    3.2 MainWindow.xaml.cs..5

    3.3 Result..7

Revision History

Revision	Date	Description
1.0	2024-02-01	Original release

1. Introduction

This document describes how to get the robot error event displayed on the TMcraft Shell. Readers should have the following prerequisites:

- Basic knowledge on programming C# and WPF
- Having read *TMcraft Shell Tutorial: Basic Development*
- Having read *TMcraft Shell API Function Manual*

Based on TMcraft.dll version 1.16, this tutorial.

2. Concept

TMcraft Shell API comes with the item below associated to Error Event:

- event RobotStatusProvider.ErrorEvent

  An event type denotes to the error event occurred on the robot. A function can be linked to this event for activation once the event is triggered.

- class ErrorStatus

  ErrorStatus denotes the structure of the data returned by RobotStatusProvider.ErrorEvent. Note that the ErrorEvent does not directly return this object type but a JSON string able to be converted to the ErrorStatus type.

The Program should declare and define an ErrorEvent after initializing TMcraft Shell, i.e., assigning a function to the event; this function will execute once the ErrorEvent is triggered.

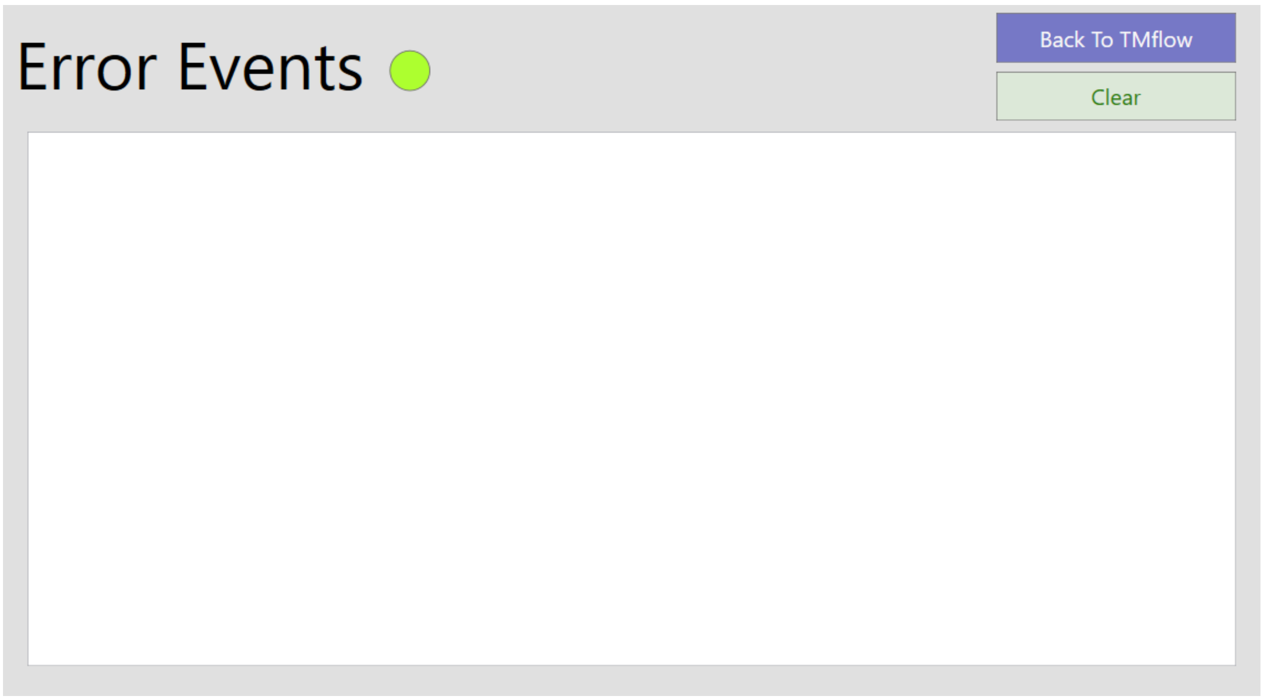
3. Sample Code

This section will talk about the essential parts of the ShowErrorEvent source code. For the complete source code, please refer to TMcraft Development Kit\Shell\Samples\ShowErrorEvent\.

3.1 MainWindow.xaml

MainWindow.xaml defines the UI which includes:

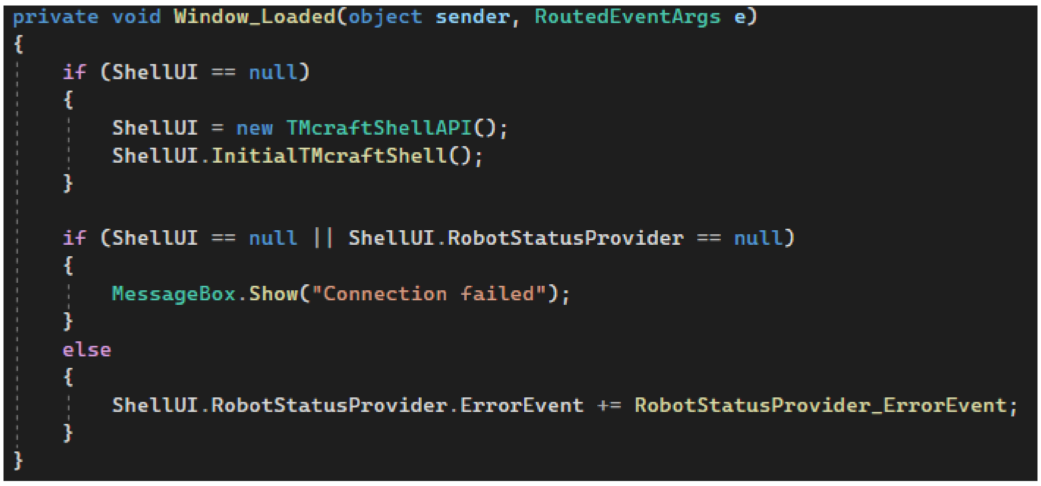
- Elp_errorStatus: turns red when the robot encounters an error.
- Btn_Back: click to go back to TMflow
- Btn_Clear: click to clear the error status displayed in the textbox
- TextBox_Content: shows the error event the data



3.2 MainWindow.xaml.cs

MainWindow.xaml.cs defines the program of the TMcraft Shell. Here is some of essential parts of the program:

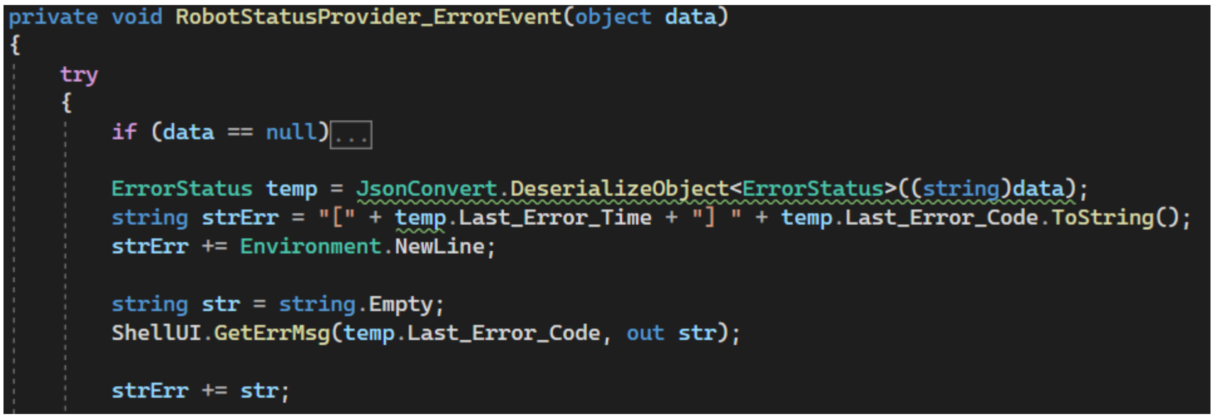
- Definition of the Window_Loaded function. After calling InitialTMcraftShell to connect the Shell Program to TMflow, it declares a RobotStatusProvider.ErrorEvent and implements with the function RobotStatusProvider_ErrorEvent.



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

    if (ShellUI == null || ShellUI.RobotStatusProvider == null)
    {
        MessageBox.Show("Connection failed");
    }
    else
    {
        ShellUI.RobotStatusProvider.ErrorEvent += RobotStatusProvider_ErrorEvent;
    }
}
```

- Definition of the event function RobotStatusProvider_ErrorEvent, First, it converts the data of the ErrorEvent to an object of class ErrorStatus, gets Last_Error_time and Last_Error_Code from the ErrorStatus object and assigns them to a string, uses TMcraftShellAPI.GetErrMsg to get the description of the Last_Error_Code, and completes the string.



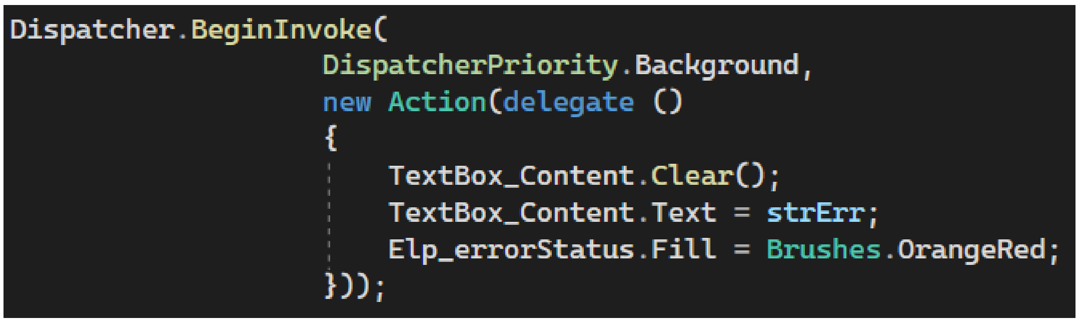
```
private void RobotStatusProvider_ErrorEvent(object data)
{
    try
    {
        if (data == null)...

        ErrorStatus temp = JsonConvert.DeserializeObject<ErrorStatus>((string)data);
        string strErr = "[" + temp.Last_Error_Time + "] " + temp.Last_Error_Code.ToString();
        strErr += Environment.NewLine;

        string str = string.Empty;
        ShellUI.GetErrMsg(temp.Last_Error_Code, out str);

        strErr += str;
```

- Definition of the action of the Dispatcher.BeginInvoke within the error event function. It assigns the string to the text of TextBox_Content and changes the color of Elp_errorStatus to red.



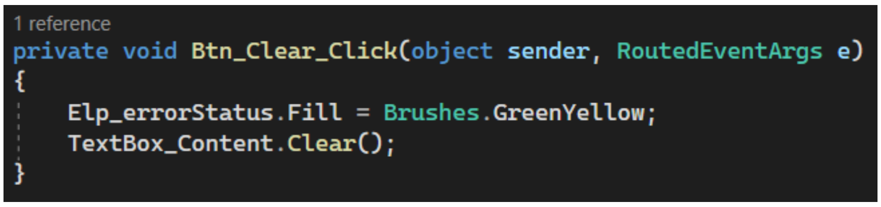
```
Dispatcher.BeginInvoke(
                DispatcherPriority.Background,
                new Action(delegate ()
                {
                    TextBox_Content.Clear();
                    TextBox_Content.Text = strErr;
                    Elp_errorStatus.Fill = Brushes.OrangeRed;
                }));
```

IMPORTANT:

- Event functions need a dispatcher to ensure that invocations are on the thread that created the UI object. It's crucial because WPF UI objects aren't thread-safe, and making changes from a different thread can lead to issues like exceptions and crashes.
- The dispatcher queues up event functions and executes them on the UI thread, guaranteeing the safe and orderly execution of all event functions.

- Definition of the button click event function Btn_Back_Click. Refer to *TMcraft Shell Tutorial: Basic Development* for details.

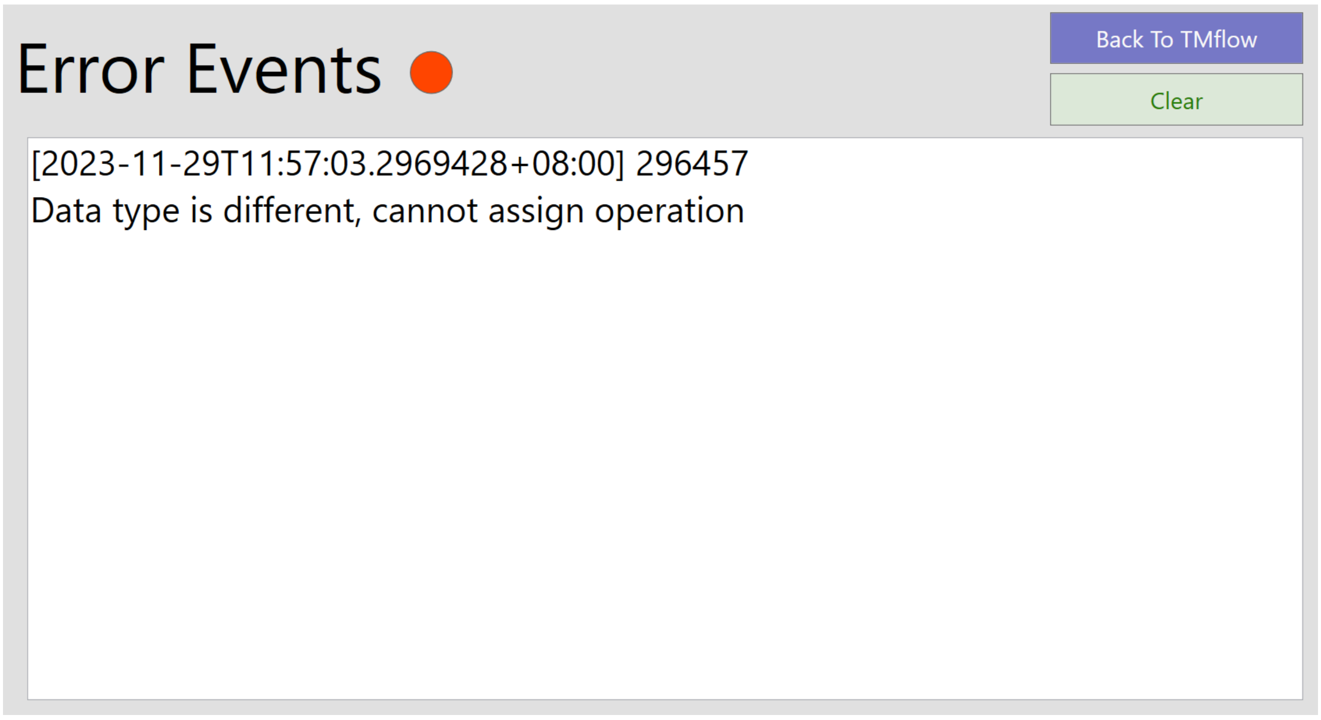
- Definition of the button click event function Btn_Clear_Click. It changes the color of Elp_errorStatus to green and clears the text of TextBox_Content.



```
1 reference
private void Btn_Clear_Click(object sender, RoutedEventArgs e)
{
    Elp_errorStatus.Fill = Brushes.GreenYellow;
    TextBox_Content.Clear();
}
```

3.3 Result

Developers can test the TMcraft Shell within TMflow by triggering an error in TMflow to see the relevant error message in the Shell UI.



```
[2023-11-29T11:57:03.2969428+08:00] 296457
Data type is different, cannot assign operation
```

TECHMAN ROBOT INC.