# UI Classes

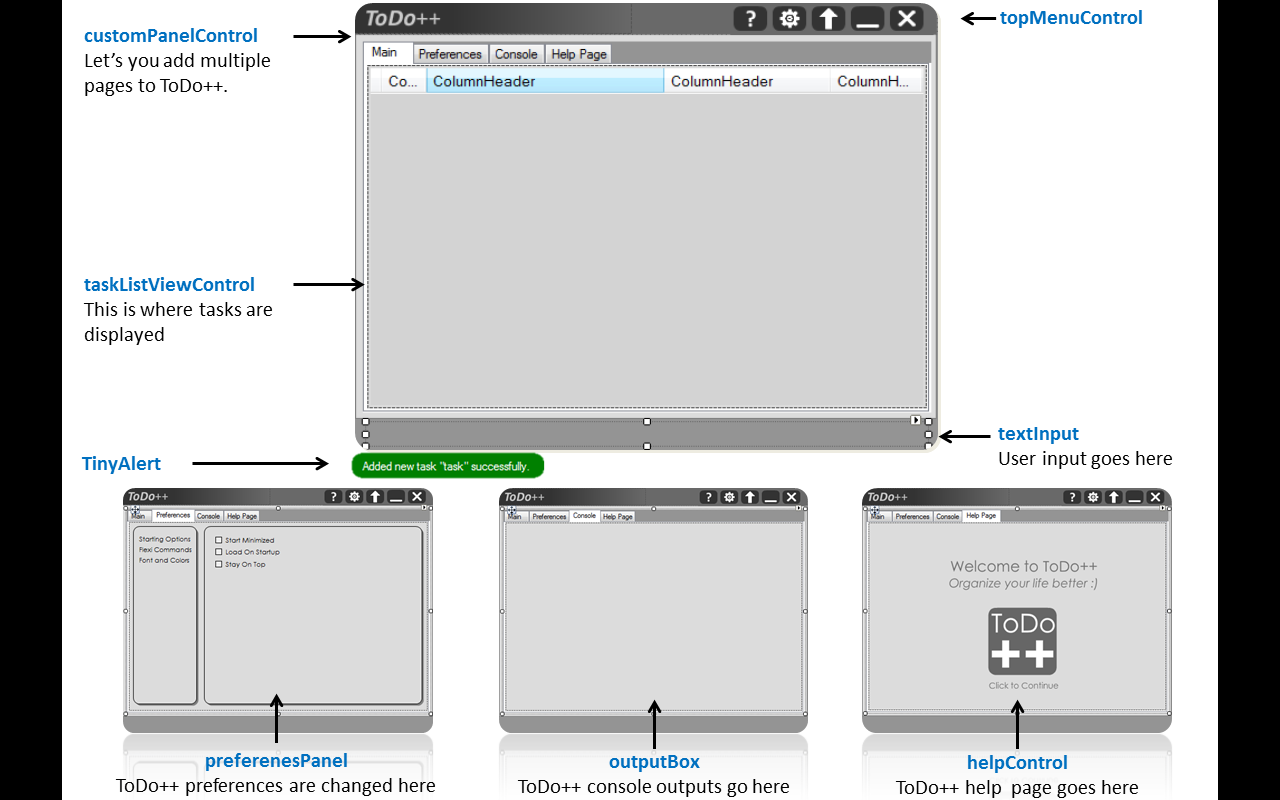
## UI

### Summary

The class that contains all user interface components and interacts directly with Logic and Settings, displaying the relevant output to the user

### Detailed Description

UI contains the following components. They have been labeled in the images below



### Adding or Displaying Pages (#PanelSwitching)

UI employs a developer friendly component called CustomPanelControl that allows you to add multiple pages/panels. These panels can be switched easily by modifying the SelectedIndex property. Implementation can be seen in the #region PanelSwitching

### Adding Keyboard Shortcuts (#Hotkeys)

UI contains a function ProcessCmdKey that lets developers add new hotkeys as long as you are within the scope of ToDo++. This function can be found in #Hotkeys. Adding global hotkeys that are accessible outside ToDo++ however is a Win32 Function that can be found in the section below

### Internal Design Functions (#InternalDesignFunctions)

This region contains code for animations such as collapsing and expanding of form (#CollapseExpand), Fading in and Out (#FormFadeInOut), Minimizing to the TaskBar (#SystemTray), and other Win32 based functions such as loading on startup, shadows and rounded edges.

### Constructor

|  |  |
| --- | --- |
| UI(Logic logic); | Starts by initializing all designer components, including Logic and MainSettings. References of these are then passed into some components such as preferencesControl |

### Important API (Public Method)

|  |  |
| --- | --- |
| **Method** | **Description** |
| ToggleHelpToDoPanel() | Toggles between Help and TaskListView Panel |
| ToggleToDoPreferencesPanel() | Toggles between Preferences and TaskListView Panel |
| ToggleConsolePanel() | Toggle between TaskListView and Console Panels |

### Important API (Private Methods)

|  |  |
| --- | --- |
| **Method** | **Description** |
| InitializeSettings() | Checks if ToDo++ should be minimized to tray when started and ensures that ToDo++ should load on startup depending on user settings |
| RegisterInStartup(bool isChecked) | Adds a registry entry to ensure that ToDo++ opens when started up |
| MinimiseMaximiseTray() | Toggles between minimizing and maximizing ToDo++ from the system tray |
| ProcessText() | Takes in the user input and processes it via logic, displaying the task list in taskListViewControl |

## PreferencesPanel

### Summary

Manages all Preferences Controls

### Detailed Description

PreferencesPanel uses a CustomPanelControl to switch between components. This means you can add as many preference controls as long as space permits. The controls found here directly modify MainSettings, and the settings are saved immediately.

### Constructor

|  |  |
| --- | --- |
| PreferencesPanel() | Loads preference names and all preference controls and components |

### Important API (Public Method)

|  |  |
| --- | --- |
| **Method** | **Description** |
| InitializeWithSettings(Settings settings) | Loads MainSettings into this class. The controls cannot function without a reference to this |

### Important API (Private Methods)

|  |  |
| --- | --- |
| **Method** | **Description** |
| LoadPreferencesTree() | You can set the preference titles here for your controls and modify the event handler for selecting preferences |

## In this developer manual, we will only be looking at the FlexiCommadControl component, as that is likely the component you will be interacting with as a developer

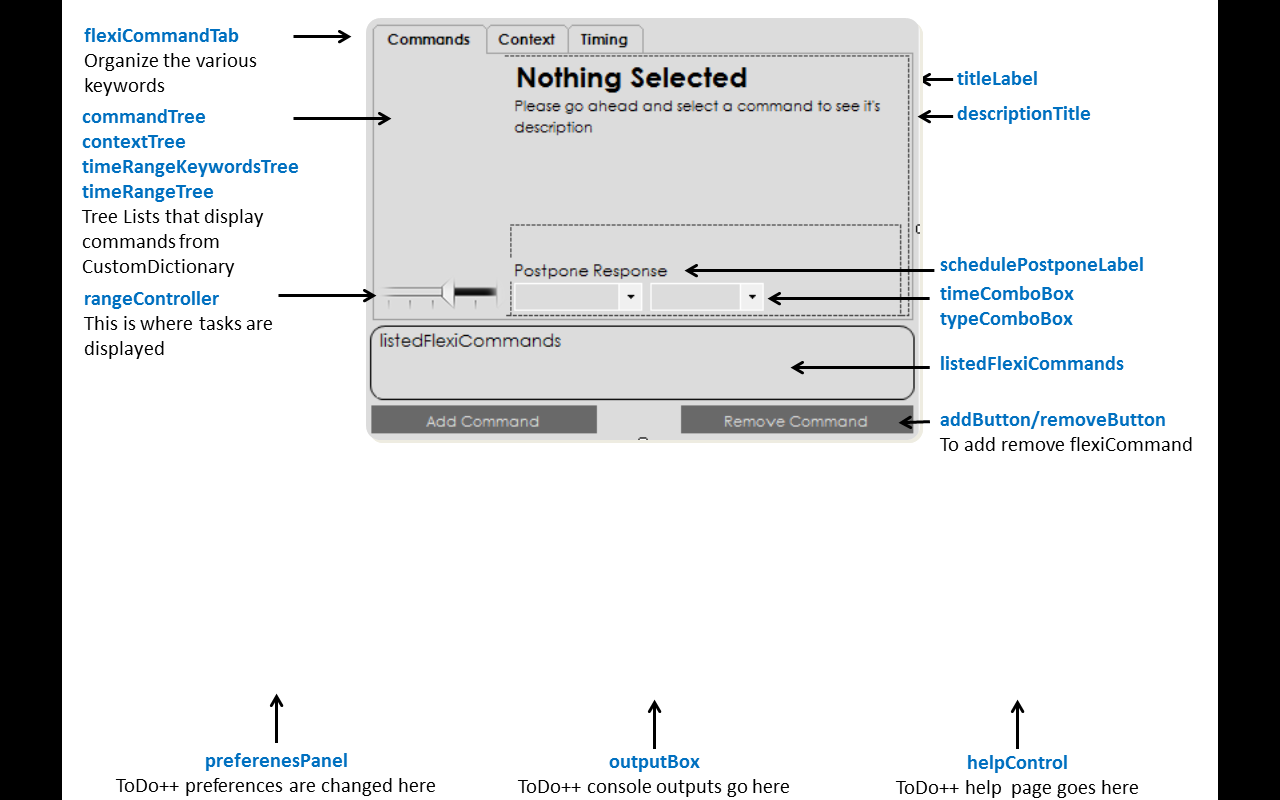
## FlexiCommandsControl

### Summary

This is where the user can modify the commands and keywords

### Detailed Description

FlexiCommandsControl is a user interface wrapper that calls and modifies the flexicommand properties of MainSettings.



### Constructor

|  |  |
| --- | --- |
| PreferencesPanel() | Loads all keywords from CustomDictionary and displays them |

### Important API (Public Method)

|  |  |
| --- | --- |
| **Method** | **Description** |
| InitializeFlexiCommands(Settings settings) | Loads MainSettings into this class. This control cannot function without a reference to this |

### Important API (Private Methods)

|  |  |
| --- | --- |
| **Method** | **Description** |
| CommandType ConvertStringToCommand(string command) | Automatically converts a string into a CommandType |
| ContextType ConvertStringToContext(string context) | Automatically converts a string into a ContextType |
| TimeRangeKeywordsType ConvertStringToTimeRangeKeyword(string rangeKeyword) | Automatically converts a string into a TimeRangeKeywordType |
| TimeRangeType ConvertStringToTimeRange(string timeRange) | Automatically converts a string into a TimeRangeType |
| LoadCommandList() | Automatically load all CommandType from CustomDictionary |
| LoadContextList() | Automatically load all ContextType from CustomDictionary |
| LoadTimeKeywordRangeList() | Automatically load all TimeRangeKeywordsType from CustomDictionary |
| LoadTimeRangeList() | Automatically load all TimeRangeType from CustomDictionary |
| ClearSelectedCommands() | Clears commands from listedFlexiCommands |
| ShowUserInputBox() | Shows the UserInputBox for user to add a new flexi command |
| UpdateFlexiCommandList() | Updated listedFlexiCommands with all the latest flexiCommands from the selected item |
| UpdateTimeRangeUI() | Updates the rangeController with the modified time ranges |
| AddFlexiCommandToSettings(string flexiCommand) | Adds a flexiCommand by calling the function in settings |
| RemoveFlexiCommandToSettings(string flexiCommand) | Removes the selected flexiCommand by calling the function in settings |
| UpdateDescription() | Updates the description of the selected flexiCommand to descriptionLabel and descriptionTitle |
| UpdateTimeRangeDescription() | Updates the description of time ranges |
| UpdateSchedulePostponeLabel() | Updates description of schedule and postpone default time ranges |
| UpdateTabDescription() | Updates description of the selected tabs in flexiCommandTab |

# Custom Message Boxes

## Custom Message Boxes contain custom built pop-ups you can call to get display alerts, change fonts, or get user input. They are designed to fit the style of ToDo++, and are static classes that can be called at any place

## FontDialogBox

### Summary

Get and Set Font Size, Color and name.

### 

### Static Methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| InitializeOptions(string font, int size, Color color) | Set the initial options so the preview label is displayed with these settings |
| int GetSize() | Gets size selected by user |
| string GetFont() | Gets Font selected by user |
| Color GetColor() | Gets Color Selected by user |
| bool ConfirmHit() | Checks if the Okay Button was Hit or Not |
| Show(bool font, bool size, bool color) | Displays the Font Dialog Box with whichever controls that need to be enabled or disabled |
| OnTop(bool val) | Set this to be on top of other forms |

## UserInputBox

### Summary

A input box to get and set user input

### 

### Static Methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| Show(string title,string subTitle) | Shows the UserInputBox with the title and subtitle set |
| bool ValidData() | Check if Confirm was hit or not |
| string GetInput() | Gets the user inpit |
| OnTop(bool val) | Set this to be on top of other forms |

## AlertBox

### Summary

Shows an alert message. Alternative for MessageBox.

### 

### Static Methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| Show(string alertText) | Shows the alert with stated text |
| OnTop(bool val) | Set this to be on top of other forms |

## TinyAlert

### Summary

TinyAlert is where the response from Logic is displayed. It normally flashes green when successful, orange when a warning is issued and red if a command has failed

### 

### Static Methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| SetUI(UI uiPass) | Pass an instance of UI in so TinyAlertView knows its position |
| Show(StateTinyAlert state, string response) | Flashes TinyAlert for the pre-set number of seconds with the state and response |
| SetLocation() | Sets the location of TinyAlert |
| SetTiming(int time) | Sets how long TinyAlert should stay until it fades away |
| DismissEarly() | Dismisses TinyAlert before it’s preset timing |

# Settings Classes

## SettingsInformation

### Summary

This is the class that stores all settings information, and is what is actually written to file

### Default Values

Default values for the various settings are modifiable here. If no settings file exists, or one is loading ToDo++ for the first time, these values will be loaded.

### Adding new settings

To add new settings, you have to create a default value for your setting, add your setting to the MiscSettings struct and modify it’s constructor to load the default value, and finally, create a Property for it. The settings file will automatically accommodate all new settings without any issue.

### Constructor

|  |  |
| --- | --- |
| SettingInformation() | Initializes default settings. These settings can later be modified |

### Settings

|  |  |
| --- | --- |
| **Variable** | **Description** |
| MiscSettings misc; | Contains all Miscellaneous settings you may wish to add |
| Dictionary<string, CommandType> userCommandKeywords; | Contains user flexi commands for CommandType Keywords |
| Dictionary<string, ContextType> userContextKeywords; | Contains user flexi commands for ContextType Keywords |
| Dictionary<string, TimeRangeKeywordsType> userTimeRangeKeywordsType; | Contains user flexi commands for TimeRangeKeywordsType Keywords |
| Dictionary<string, TimeRangeType> userTimeRangeType; | Contains user flexi commands for TimeRangeType Keywords |
| Dictionary<TimeRangeKeywordsType, int> userTimeRangeKeywordsStartTime; | Contains the start time for TimeRangeKeywords |
| Dictionary<TimeRangeKeywordsType, int> userTimeRangeKeywordsEndTime; | Contains the end time for TimeRangeKeywords |

### Important API (Public Methods)

|  |  |
| --- | --- |
| **Method** | **Description** |
| bool ContainsFlexiCommandKeyword(string userKeyword, Enum flexiCommandType) | Checks if such a flexi command already exists in the relavent type that is passed in |

## Settings

### Summary

Contains an instance of SettingsInformation, and acts a wrapper, modifying the values safely

### Adding Settings Methods

You can add new getters and setters for your settings added to SettingsInformation

### Constructor

|  |  |
| --- | --- |
| Settings() | Calls InitializeSettings which loads settingInfo with all default values |

### Important API (Private Method)

|  |  |
| --- | --- |
| **Method** | **Description** |
| InitializeSettings() | Initializes settingInfo of type SettingsInformation with default values. You can load a new instance of SettingsInformation by calling the method UpdateSettings() |
| UpdateDictionaryPostponeSchedule() | Modifies CustomDictionary by setting the schedule and postpone length and type |

### Important API (Public Methods)

|  |  |
| --- | --- |
| **Method** | **Description** |
| UpdateSettings(SettingInformation updatedInfo) | Completely wipes and re-updates Settings Data with the instance of SettingsInformation passed in |
| **Other Settings** |  |
| bool GetFirstLoadStatus() | Gets whether this is the first time loading ToDo++. Once gotten, it is set to false |
| SetTextSize(int size) | Set default text size of Task View |
| Int GetTextSize() | Get the text size of Task View |
| SetLoadOnStartupStatus(bool status) | Sets the load on startup status |
| bool GetLoadOnStartupStatus() | Get the load on startup status |
| SetStartMinimized(bool status) | Set start minimized status |
| bool GetStartMinimizeStatus() | Get the start minimized status |
| SetStayOnTop(bool status) | Set stay on top status |
| bool GetStayOnTopStatus() | Get stay on top status |
| SetFontSelection(string font) | Set Task View font |
| string GetFontSelection() | Gets Task View font |
| **Task Color Settings** |  |
| SetTaskDoneColor(Color col) | Set task done color |
| Color GetTaskDoneColor() | Get task done color |
| SetTaskMissedDeadlineColor(Color col) | Set task missed deadline color |
| Color GetTaskMissedDeadlineColor() | Get task missed deadline color |
| SetTaskNearingDeadlineColor(Color col) | Set task nearing deadline color |
| Color GetTaskNearingDeadlineColor() | Get task nearing deadline color |
| SetTaskOverColor(Color col) | Set task over color |
| Color GetTaskOverColor() | Get task over color |
| **Time Range** |  |
| SetDefaultScheduleTimeLength(int length) | Set default time length for Command SCHEDULE |
| int GetDefaultScheduleTimeLength() | Get default time length for Command SCHEDULE |
| SetDefaultPostponeDurationLength(int length) | Set default duration length for Command POSTPONE |
| int GetDefaultPostponeDurationLength() | Get default duration length for Command POSTPONE |
| SetDefaultScheduleTimeLengthType(TimeRangeType timeRange) | Set default time length type (HOUR,DAY etc.) for Command SCHEDULE |
| TimeRangeType GetDefaultScheduleTimeLengthType() | Get default time length type (HOUR,DAY etc.) for Command SCHEDULE |
| SetDefaultPostponeDurationType(TimeRangeType timeRange) | Set default duration type (HOUR,DAY etc.) for Command POSTPONE |
| TimeRangeType GetDefaultPostponeDurationType() | Get default duration type (HOUR,DAY etc.) for Command POSTPONE |