Ui System

UI System?

No, it's not a replacement for Unity's native UI systems. There are enough of those already!

It's a way to have click/touch and hover type interactions with on-screen TPT tiles and can be used to add interactivity to tiles. But there's no UI builder - you have to lay out and configure everything manually.

Generally, you create a UI-Capable tile by implementing the ITpUiControl interface. However, it's recommended that you try to use the suppled UI tile varieties if possible or use those as base classes.

The tiles are controlled by messages: the easiest way to use this is to use the TpInputActionToTile component.

ItpUiControl has two main functions:

- Get information about supported effects and start an effect.
- Set values and get values in various formats.

Setting values is done with one method implementation: `SetValue(object value, bool withNotify = true)`

You convert the object into whatever your tile requires.

Getting values is done with several implementations for various formats: int, bool, object, string, or Char.

A control can use the `AcceptsClicks` and `AcceptsHover` to control what sort of messages it gets.

For example, a control that normally accepts clicks can temporarily return false from the `AcceptsClicks` property if it is still executing an effect such as a tween which hasn't completed yet. A second click can be inhibited until the effect is complete.

`AcceptsHover` should only be true if you want to get Hover messages: these are very frequent since they're continually sent while the mouse or other pointer is moving.

`SupportedEffects` returns a flags enum that specifies various types of effects such as BumpSize, BumpColor, etc. Return a value which describes which effects your tile actually implements.

`RunEffect` is used to execute an effect. You can add duration, Vector3, and/or Color endpoint values to use for tweening or color changes.

UI Tiles

UiButton

This is the simplest UI element, implementing a momentary button.

- Subclass of TpSlideShow
- When clicked, the color changes. Hovering isn't supported.
- It doesn't support SetValue (since it's momentary) nor any effects.
- Except for ITpUiControl.GetCharValue, all the GetValue basically return the slide index.
- GetCharValue returns a character based on the slide index
- It's intentionally really simple and easier to understand.
- It can post events and will invoke its ZoneAction if the reference exists.

UiAnimButton

This hoverable button animates when hovered or clicked.

- Subclass of TpFlexAnimatedTile.
- Can be set as momentary or toggle.
- When clicked, the animation starts or changes.
- It can post events but does not use ZoneActions.
- No effects are supported.
- SetValue is supported: input should be a bool value.
- GetIntValue returns 1 if the current animation clip is the clicked animation or 0 if not.
- GetBoolValue is the same but returns true or false.
- GetCharValue returns a space (no real meaning)
- GetStringValue returns the current animation clip name.
- (object)GetValue returns the same thing as GetBoolValue but cast to an object.
- Implements EventActionObject.

UiToggleButton

This implements a toggle button.

- Subclass of TpSlideShow
- When clicked, the slide alternates between two images for ON or OFF.
- Hovering isn't supported.

- SetValue ignores the value.
- Supports posting events and/or Zone Actions.
- No effects are supported.
- Since there are only two possible states, all of the GetValue implementations return a value related to the SlideIndex, i.e., 1 or 0, true or false, etc. GetCharValue returns a space character.

UiRadioButton

This implements a Radio button and a Radio Button Set. Use Tags or the Zone to create a Radio Button Set.

- Subclass of TpSlideShow.
- Effects and Hover are not supported.
- Similar to UiToggleButton, all the GetValue implmentations return a value related to the SlideIndex (0,1 or T/F).
- Always sends an event when clicked. Zone Actions supported.

UiHoverZone

This implements a hoverable but not clickable zone. It does nothing on its own.

- Doesn't respond to clicks or hover. No Set or GetValue.
- If this tile is present then the ZoneAction is excecuted while the pointer is within the boundary of the Zone.
- There are three fields for an Integer, a bool, or a string. The ZoneAction is passed these values as the `optionalString, optionalBool, or optionalInt` parameters.

An example can be see in the TileUi demo: the unwieldly-named `UiHoverTileZoneToPromptString_TileZoneAction : TpTileZoneAction` pokes a tooltip to an array of ascii characters when a UiPromptHoverZone (subclass of UiHoverZone) area is hovered-over.

UiAsciiChar

This implements display of a single, non-editable, ASCII character from a sprite set.

- Subclass of TpSlideShow.
- 'BumpSize' effect supported, Hover not supported.
- Can post events.
- SetValue supports:
  - string : uses first character.

    - char: uses the char.
    - integer within range: sets the slide (visually, the character)
    - bool: `1` or `0` is displayed.
- GetIntValue: the slide index (not really useful)
- GetBoolValue returns true if the character is '1'
- (object)GetValue: same as GetCharValue cast to object.
- GetCharValue: the character.

Normally you'd only care about the character value and you'd be inputting characters to SetValue and returning them (probably never need to) from GetCharValue.

But what if you want a string? That's next.

UiAsciiString

Similar to UiHoverZone, this doesn't do anything visual. You use the Zone Editing feature of all TilePlus tiles to set up a zone which includes any number of UiAsciiChar tiles with optional simple justification and wrapping. You write a string to the UiAsciiString tile and it distributes it to the Char tiles.

- Clicks and Hover are not supported.
- RunEffect relays the passed parameters to each AsciiChar tile.
    - Which may or may not implement them. You can subclass AsciiChar to add effects.
- Events are not issued.
- SetValue:
    - bool : clear string.
    - int : convert to string.
    - string : use directly.
    - JustifiedString (a custom class): justify and use the string value from the JustifiedString instance.
- GetValue:
    - int : 0
    - bool : false
    - char : first char of string sent previously.
    - object : null
    - String: string sent previously.

This tile is only a zone. Arrange a seres of UiAsciiCharTiles in a rectangular array (1 col and N rows, or 1 row and N columns or N columns and M rows) and add a UiAsciiStringTile.

Using the AsciiStringTile's Zone controls, draw a zone around the array of AsciiCharTiles. This is easy to do using Painter or Tile+Brush.

Now the AsciiStringTile is a controller for all of the AsciiChar tiles.

At runtime, write strings to the AsciiStringTile and it'll treat the AsciiCharTiles as a group and place the string characters in the proper locations, with simple left, center, or right justification.

- ONLY left-to-right is supported.
- DOES NOT support editing.
- DOES NOT support sparse arrays of AsciiCharTiles: the entire zone must be filled with tiles. If not, you're adding spaces.

The array of tiles can be a horizontal row, a vertical column, or a box.

This tile assumes that the ASCII char tiles are on the same tilemap.

ITpUiControl

```
public interface ITpUiControl
{
    /// <summary>
    /// Set c# object Value
    /// </summary>
    /// <param name="value">c# object (or boxed UnityEngine.Object) value</param>
    /// <param name="withNotify">permit notification if appropriate</param>
    void SetValue(object value, bool withNotify = true);

    /// <summary>
    /// Run an effect on the control, if implemented
    /// </summary>
    /// <param name="effectType">Value from UiEffect enum</param>
    /// <param name="duration">duration of the effect.</param>
    /// <param name = "endPoint" >endpoint for V3 type effects</param>
    /// <param name = "endColor" >endpoint for Color type effects</param>
    /// <returns>false if unimplemented.</returns>
    bool RunEffect(UiEffect effectType,
                   float    duration,
                   Vector3? endPoint = null,
                   Color?   endColor = null);

    /// <summary>
    /// Returns a value from the FLAGS enum UiEffect,
    /// shows the controls effect capabilities.
    /// </summary>
    UiEffect SupportedEffects { get; }

    /// <summary>
    /// Get integer Value
    /// </summary>
    int GetIntValue { get; }

    /// <summary>
    /// Get bool Value
```

```
    /// </summary>
    bool GetBoolValue { get; }

    /// <summary>
    /// Get c# object Value
    /// </summary>
    object GetValue { get; }

    /// <summary>
    /// Get string value
    /// </summary>
    string GetStringValue { get; }

    /// <summary>
    /// Get value as a character.
    /// </summary>
    char GetCharValue { get; }

    /// <summary>
    /// Does this tile accept clicks?
    /// </summary>
    bool AcceptsClicks { get; }

    /// <summary>
    /// Does this tile accept hover?
    /// </summary>
    bool AcceptsHover { get; }


}
```