Introduction

Designed for all Unity users, who ever wanted to create a mobile games. Allows you to quickly and easily develop actions based on a touchscreen, button, joystick(static & dynamic), touchpad, dpad, steering wheel.

Included features are:

- ✓ All asset files, for free or commercial use (re-selling prohibited).
- ✓ Designed for Unity UI.
- ✓ Button, Joystick, Touchpad, *DPad, *Steering Wheel, *Tilt included.
- ✓ Easy and flexible API.
- ✓ Fast prefab creation.
- ✓ Smart visual tuning.
- ✓ Quickly and easily setup for your game.
- ✓ Unity UI native anchoring system.
- ✓ Smart «only outdated» update system.
- ✓ Multitouch for all mobile devices.
- ✓ Intuitive and easy to modify the source code for any of your needs.

These features should cover the most requirements for a mobile games. However, please note that a this project can't suit all game cases. You likely want to modify it to fit your needs and implement your own unique game and user interface mechanics. In the following chapters, this manual explains all components involved in this kit, so you can see where you might want to start.

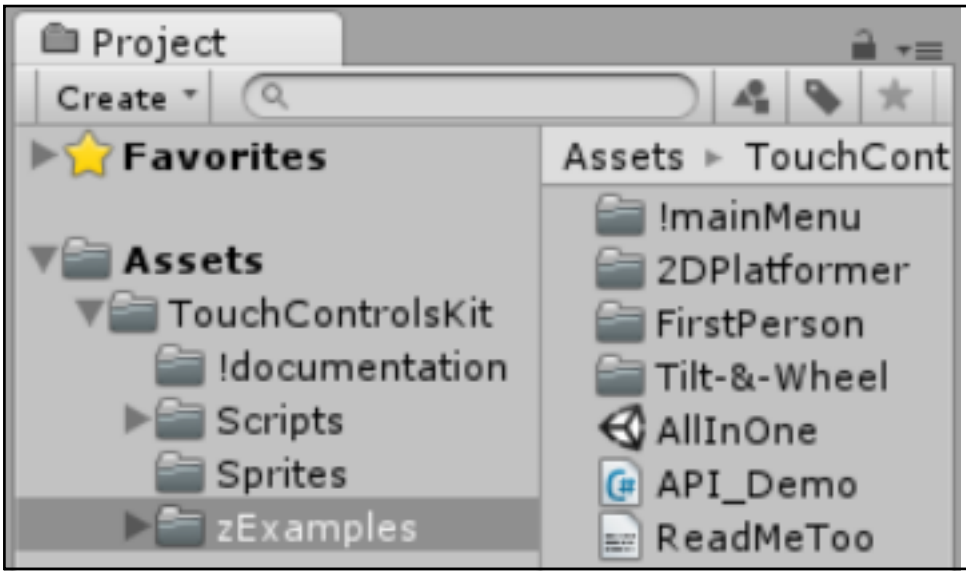
* Only «Touch Controls Kit» Pro edition

First Steps

WARNING: If you are new to Unity, please take a quick break and get dirty with its main functionalities first, because this documentation will assume you have some basic knowledge regarding the interface and its editor tools.

Import this unitypackage into an empty project.

Once the import has finished, you'll see all project files listed in the Project panel.



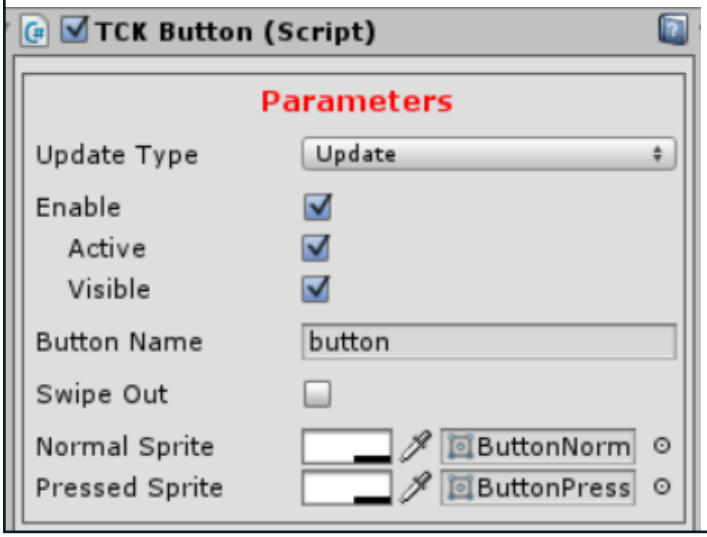
"**zExamples**" forder contains five example scenes for demonstrate controllers work, it optional package intended for acquaintance with API functions.

Use included context menu for spawn conroller prefab in scene.

GameObject->UI->TouchControlsKit

So, you probably have already seen how it works and you already want to understand the principles of operation, as well as set up all by your project. Well, let's start, the following pages are devoted to this.

Button



"**Update Type**" - Used for select update method of delegates and events invoking.

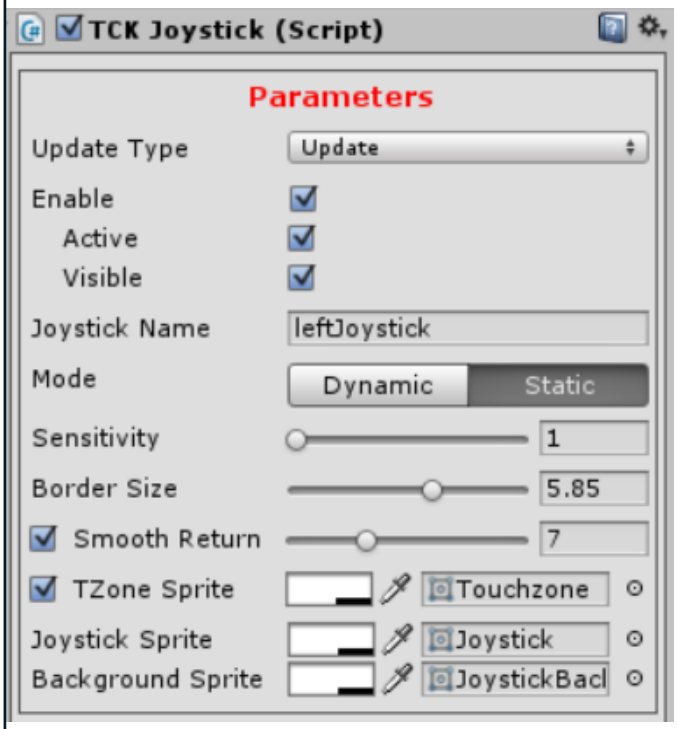
"**Enable**" - if false, the controller not working and not visibe.

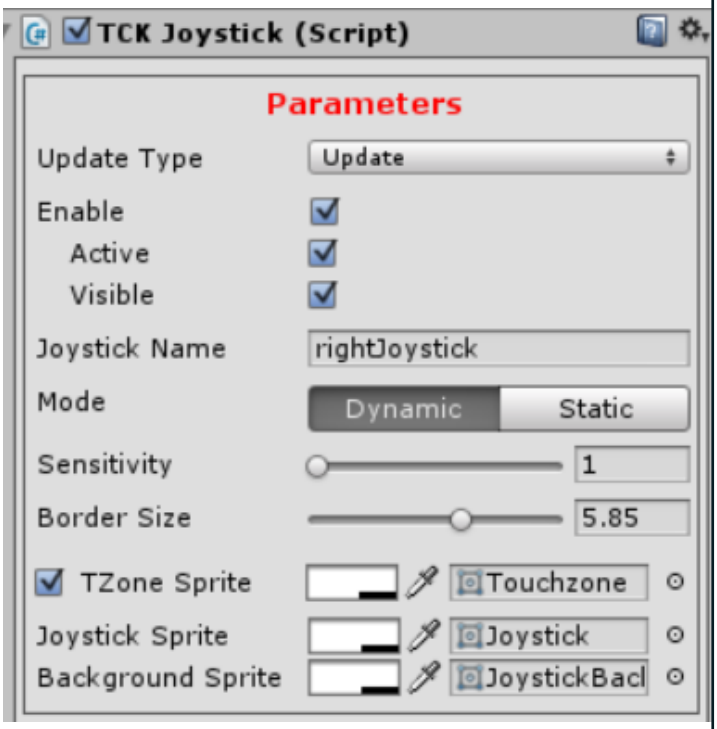
"**Visible**" - if false, the controller working, but don't visibe.

"**Active**" - if false, the controller don't working, but visibe.

"**Swipe Out**" - if true, the button is pressed, even when the finger goes beyond its borders.

Joystick

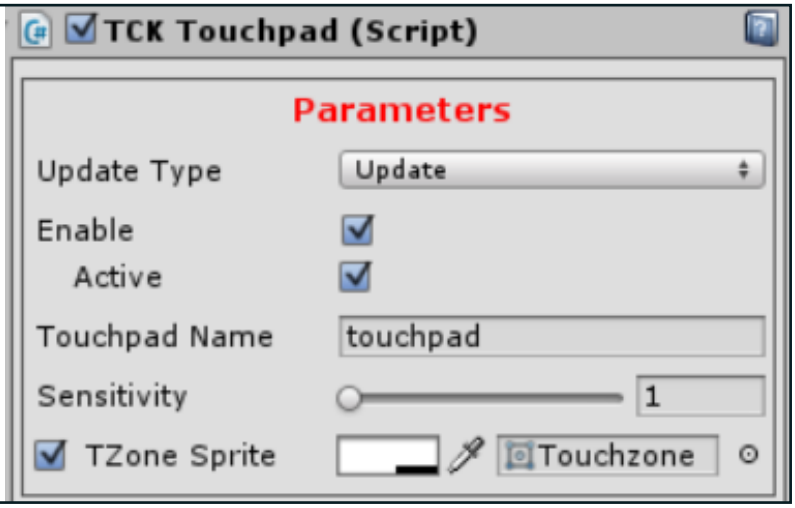


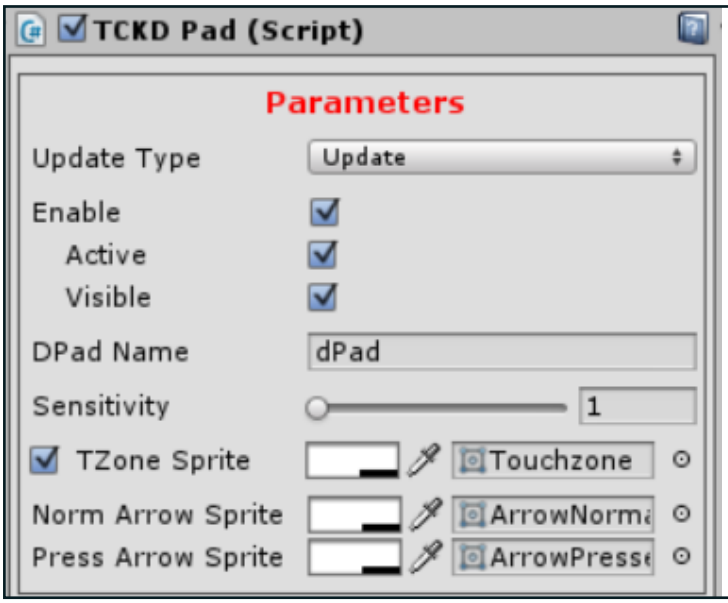


"**Border Size**" - used for sets threshold joystick distance of background position.
"**Smooth Return**" - allows the statick joystick slowly return to start position.
"TZone"(toggle, color, sprite) - Show/Hide controller touch zone.

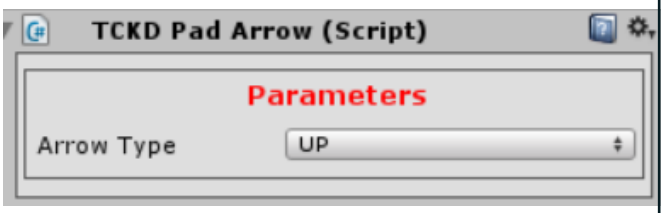
Touchpad

Touchpad is good to use where you need to emulate a mouse, for example at first person camera.





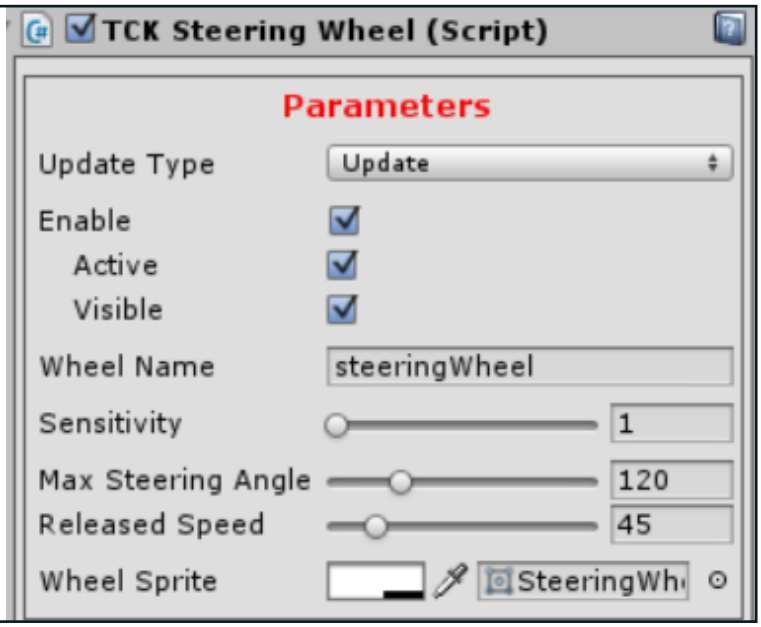
DPad [only Pro]



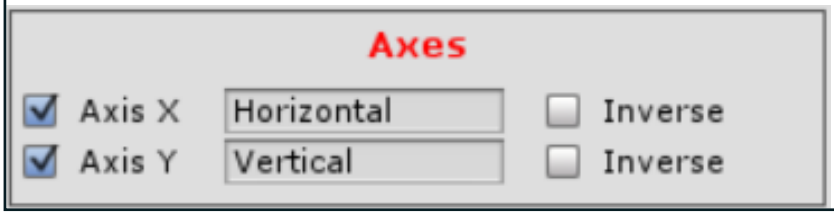
"**Norm/Press Arrow Sprite`s**" - sets sprites and colors for all arrows, for pressed and normal state.

Steering Wheel [only Pro]

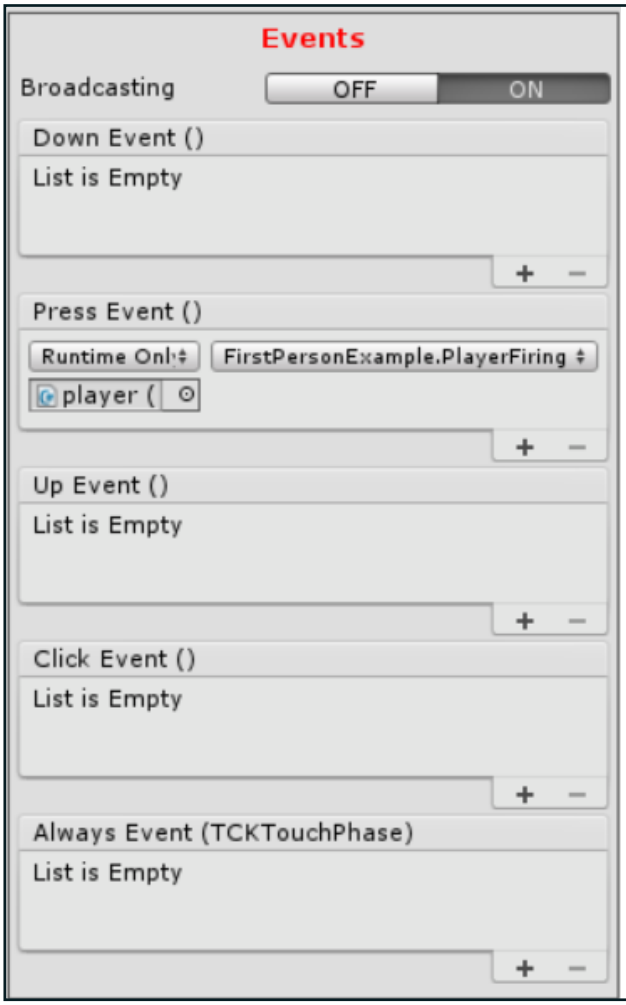
A steering wheel (also called a driving wheel or a hand wheel) is a type of steering control in vehicles and vessels (ships and boats) at touch screen.



Axes



Axes are used for getting the current coordinates of the controller (does not using in button). Any axis can be enabled or disabled, and you can sets the specific name for any axis , note the names of the two axes must be different.



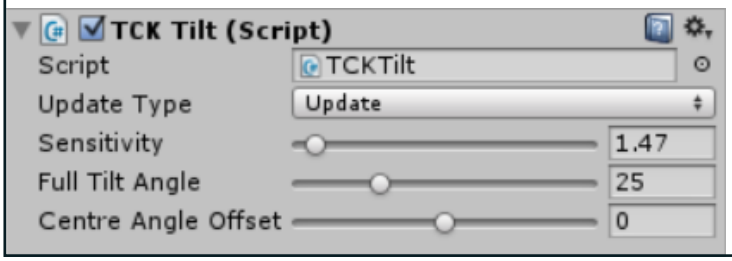
Events

[only Pro]

Events allow broadcasting data(float axisX, float axisY) or call another avent to one or more receivers. Based on UnityEngine.Events.UnityEvent.

Read more here...

Tilt [only Pro]



Tilt API:

TCKTilt.forwardAxis;
TCKTilt.sidewaysAxis;

API

namespace TouchControlsKit

Class **TCKInput**

Function	Description
static bool isActive { get; }	The local active state of all controllers. (Read Only)
static void SetActive(bool value)	Activates/Deactivates all controllers in scene.
*static void BindAction(string buttonName, ActionEventHandler m_Handler, ActionPhase actionPhase)	Bind your void to button action, identified by buttonName. Called established event in actionPhase.
*static void BindAction(string buttonName, ActionAlwaysHandler m_Handler)	Bind your void to button update, identified by buttonName. Always called.
*static void UnBindAction(string buttonName, ActionEventHandler m_Handler, ActionPhase actionPhase)	Unbind your void of button action, identified by buttonName & actionPhase.
*static void UnBindAction(string buttonName, ActionAlwaysHandler m_Handler)	Unbind your void of button update, identified by buttonName.
*static void BindAxes(string controllerName, AxesEventHandler m_Handler, ActionPhase actionPhase)	Bind your void to controller axis action, identified by controllerName. Called established event in actionPhase.
*static void BindAxes(string controllerName, AxesAlwaysHandler m_Handler)	Bind your void to controller axis update, identified by controllerName. Always called.
*static void UnBindAxes(string controllerName, AxesEventHandler m_Handler, ActionPhase actionPhase)	Unbind your void of controller axis action, identified by controllerName & actionPhase.
*static void UnBindAxes(string controllerName, AxesAlwaysHandler m_Handler)	Unbind your void of controller update, identified by controllerName.
static bool GetControllerEnable(string controllerName)	Returns the controller Enable value identified by controllerName.
static void SetControllerEnable (string controllerName, bool value)	Sets the controller Enable value identified by controllerName.
static bool GetControllerActive(string controllerName)	Returns the controller Active state value identified by controllerName.
static void SetControllerActive (string controllerName, bool value)	Sets the controller Active state value identified by controllerName.
static bool GetControllerVisible(string controllerName)	Returns the controller Visibility value identified by controllerName.
static void SetControllerVisible (string controllerName, bool value)	Sets the controller Visibility value identified by controllerName.
static float GetAxis (string controllerName, string axisName)	Returns the Axis value identified by controllerName & axisName.
static float GetAxis (string controllerName, AxisType axisType)	Returns the Axis value identified by controllerName & axisType.
static bool GetAxisEnable (string controllerName, string axisName)	Returns the axis Enable value identified by controllerName & axisName.
static bool GetAxisEnable (string controllerName, AxisType axisType)	Returns the axis Enable value identified by controllerName & axisType.
static void SetAxisEnable (string controllerName, string axisName, bool value)	Sets the axis Enable value identified by controllerName & axisName.
static void SetAxisEnable (string controllerName, AxisType axisType, bool value)	Sets the axis Enable value identified by controllerName & axisType.
static bool GetAxisInverse (string controllerName, string axisName)	Returns the axis Inverse value identified by controllerName & axisName.

static bool GetAxisInverse (string controllerName, AxisType axisType)	Returns the axis Inverse value identified by controllerName & axisType.
static void SetAxisInverse (string controllerName, string axisName, bool value)	Sets the axis Inverse value identified by controllerName & axisName.
static void SetAxisInverse (string controllerName, AxisType axisType, bool value)	Sets the axis Inverse value identified by controllerName & axisType.
static float GetSensitivity(string controllerName)	Returns the value of the controller Sensitivity identified by controllerName.
static void SetSensitivity (string controllerName, string float value)	Sets the Sensitivity value identified by controllerName.
static void ShowingTouchZone (bool value)	Showing/Hiding touch zone for all controllers in scene.
static bool GetButtonDown (string buttonName)	Returns true during the frame the user pressed down the touch button identified by buttonName.
static bool GetButton(string buttonName)	Returns whether the given touch button is held down identified by buttonName.
static bool GetButtonUp(string buttonName)	Returns true during the frame the user releases the given touch button identified by buttonName.
static bool GetButtonClick(string buttonName)	Returns true during the frame the user clicked the given touch button identified by buttonName.
static UpdateType GetUpdateType(string controllerName)	Returns the UpdateType enum value identified by controllerName.
static void SetUpdateType (string controllerName, UpdateType updateType)	Sets the UpdateType enum value identified by controllerName.
static TCKTouchPhase GetTouchPhase (string controllerName)	Returns describes phase of a finger touch identified by controllerName.

*delegate void ActionEventHandler();
*delegate void ActionAlwaysHandler(TCKTouchPhase touchPhase);

*delegate void AxesEventHandler(float axisX, float axisY);
*delegate void AxesAlwaysHandler(float axisX, float axisY, TCKTouchPhase touchPhase);

enum AxisType { **X**, **Y** } - Identification axis by type X or Y.

enum UpdateType { **Update**, **LateUpdate**, **FixedUpdate**, **OFF** } - Select update method for delegates and events.

enum TCKTouchPhase
- **Began** - A finger touched the controller.
- **Moved** - A finger moved on the controller.
- **Stationary** - A finger is touching the controller but hasn't moved.
- **Ended** - A finger was lifted from the controller. This is the final phase of a touch.
- **NoTouch** - A finger don't touch the controller.

*enum ActionPhase
- **Down** - During the frame the user pressed down the touch controller.
- **Pressed** - Whether the given touch controller is held down.
- **Up** - During the frame the user releases the given touch controller.
- **Click** - During the frame the user clicked the given touch controller.

* Only «Touch Controls Kit» Pro edition

Contacts

All the source code is made so that it is easy to understand,
feel free to take a look at the scripts
and to modify them to fit your needs.

If you have any questions, comments, suggestions or find errors
in this documentation, do not hesitate to contact me.

Support:
http://bit.ly/vk-Support | http://forum.unity3d.com/threads/210040/

myAssets: http://u3d.as/5Fb
mySite: http://vkdemos.ucoz.org
myTwitter: http://twitter.com/VictorKlepikov

Thank you for choosing Touch Controls Kit!

**If you've bought this asset on the Unity Asset Store,
please write a short review
so other users can form an opinion!**

Again, thanks for your support, and good luck with your projects!

Kindest regards, **Victor Klepikov**