Read Me

Thank you for downloading!

If you like this asset then please leave a review/rating on the Asset Store, it really helps me out!

If you have any questions, feel free to email me at: carlos.wilkes@gmail.com

What is Lean Touch?

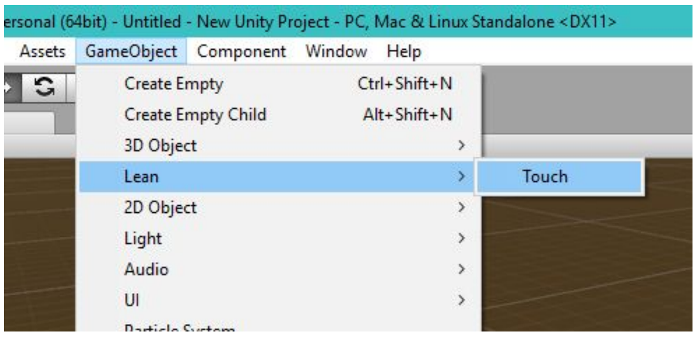
When you create mobile games you often want to make use of multi touch gestures, like pinch and twist. However, Unity makes this difficult to do, because they only provide the Input.touches array, requiring you to do all the calculations yourself.

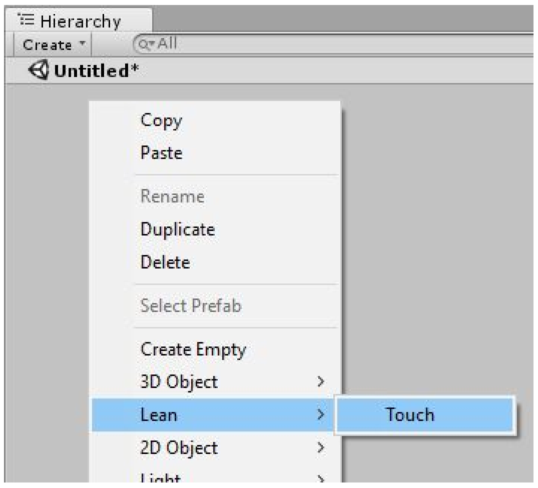
With LeanTouch you no longer have to worry about these issues, because all the touch gesture calculations are done for you in a very simple and elegant way.

LeanTouch also allows you to simulate multi touch gestures on desktop, so you don't have to waste lots of time deploying to your mobile devices while you setup your input.

How do I add Lean Touch to my game?

Click GameObject / Lean / Touch, or right click your Hierarchy window and go to Lean / Touch.


ersonal (64bit) - Untitled - New Unity Project - PC, Mac & Linux Standalone <DX11>
Assets
GameObject
Component
Window
Help
Create Empty Ctrl+Shift+N
Create Empty Child Alt+Shift+N
3D Object
Lean
Touch
2D Object
Light
Audio
UI
Particle System



You should now see a new GameObject called 'LeanTouch' with the LeanTouch component selected.

When you enter play mode, this component will automatically convert all mouse and touch input into an easy to use format.

Remember that scripts using Lean Touch only work when there is a LeanTouch component active in your scene, so make sure to add one to every scene (or carry it over with DontDestroyOnLoad).

How do I use Lean Touch without code?

Lean Touch comes with many example component to do common tasks.

For example, if you want to spawn a prefab when a finger touches the screen, you can begin by making a new GameObject, and adding the LeanSpawnGameObject component. Inside this component, you'll see it has the 'Prefab' setting. You can browse or drag and drop your desired prefab here. Next, you need to add the LeanFingerDown component. Inside this component, you'll see it has the 'OnFingerDown' event. To link this event to the prefab spawning, you need to click the plus '+' button below where it says 'List is Empty', and in the bottom left box that says 'None (Object)', you need to drag and drop the LeanSpawnGameObject component you added earlier. Next, you need to click the 'No Function' dropdown, and select the 'LeanSpawnGameObject -> Dynamic LeanFinger -> Spawn' function. You can now hit play, and pressing a finger (or clicking) on the screen will spawn your prefab.

If you want to change it so tapping spawns your prefab, then you can follow the same steps, but using the LeanFingerTap component instead

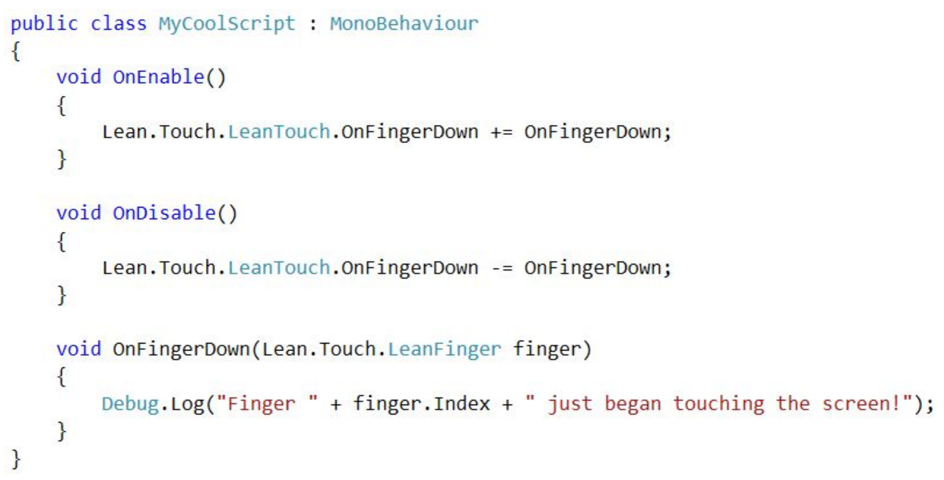
To understand how all the other example components work, I recommend you experiment with all the example scenes and see how they're linked together.

How do I use Lean Touch with C#?

You can access all finger data from the Lean.Touch.LeanTouch class.

The easiest way to begin is by hooking into the static events it exposes.

For example, if you want to perform an action when a finger touches the screen, you can hook into the Lean.Touch.LeanTouch.OnFingerDown event. This event gets called every time a finger begins touching the screen, and gets passed a Lean.Touch.LeanFinger instance as a parameter. It's recommended you hook into these events from OnEnable, and unhook from OnDisable.



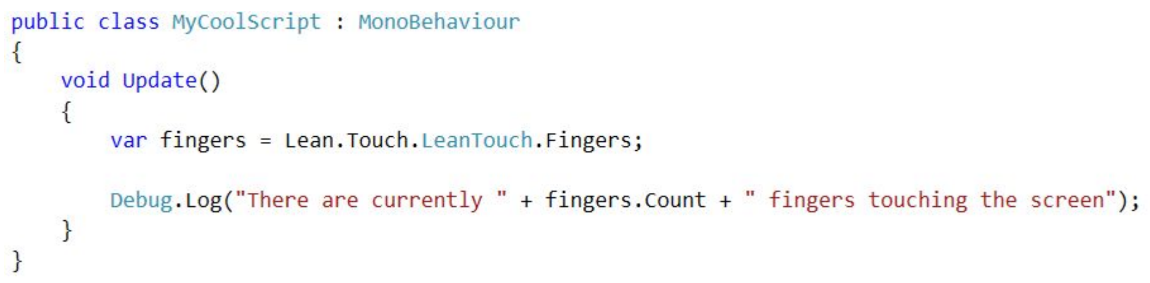
```csharp
public class MyCoolScript : MonoBehaviour
{
    void OnEnable()
    {
        Lean.Touch.LeanTouch.OnFingerDown += OnFingerDown;
    }

    void OnDisable()
    {
        Lean.Touch.LeanTouch.OnFingerDown -= OnFingerDown;
    }

    void OnFingerDown(Lean.Touch.LeanFinger finger)
    {
        Debug.Log("Finger " + finger.Index + " just began touching the screen!");
    }
}
```

To see what other events are available, I recommend you read the LeanTouch.cs, script and its comments.

Another way to access finger data is to poll it directly from the Lean.Touch.LeanTouch.Fingers static list. This list stores all fingers that are currently touching the screen, and you can use this data at any time (e.g. Update) to quickly handle input.



```csharp
public class MyCoolScript : MonoBehaviour
{
    void Update()
    {
        var fingers = Lean.Touch.LeanTouch.Fingers;

        Debug.Log("There are currently " + fingers.Count + " fingers touching the screen");
    }
}
```

How do I handle multi-finger gestures (e.g. Pinch, Twist) from C#?

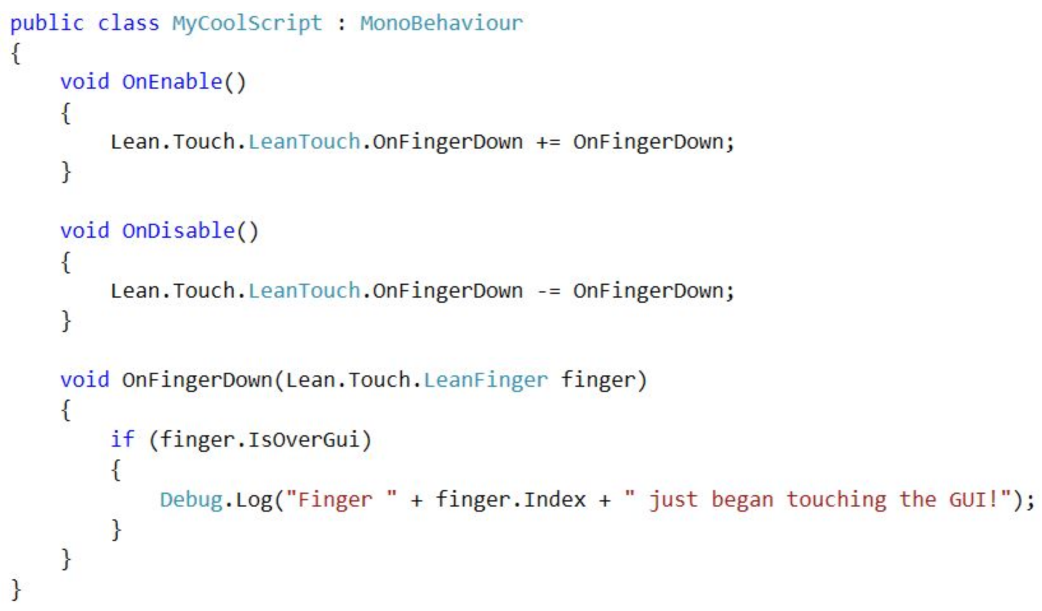
The Lean.Touch.LeanGesture class makes this very easy.

For example, if you want to find how many degrees the fingers were twisted in the last frame, you can call the Lean.Touch.LeanGesture.GetTwistDegrees() method, which will automatically calculate it from all fingers. This method also has an overload that allows you to pass a specific list of fingers if you don't want to use them all.

To see what other gestures are available, I recommend you read the LeanGesture.cs script and its comments.

How do I stop my touch controls from going through my UI from C#?

If you hook into any of the Lean.Touch.LeanTouch.OnFinger… events, you will get a Lean.Touch.LeanFinger instance. This class has the IsOverGui and StartedOverGui values you can check.



```
public class MyCoolScript : MonoBehaviour
{
    void OnEnable()
    {
        Lean.Touch.LeanTouch.OnFingerDown += OnFingerDown;
    }

    void OnDisable()
    {
        Lean.Touch.LeanTouch.OnFingerDown -= OnFingerDown;
    }

    void OnFingerDown(Lean.Touch.LeanFinger finger)
    {
        if (finger.IsOverGui)
        {
            Debug.Log("Finger " + finger.Index + " just began touching the GUI!");
        }
    }
}
```

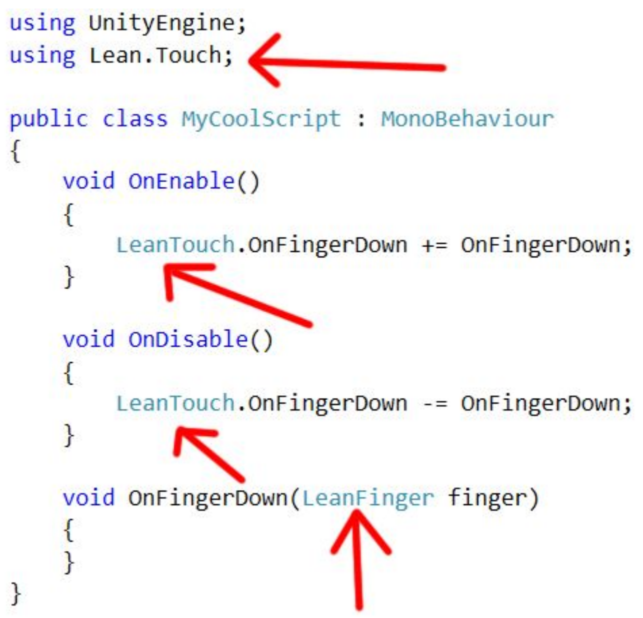
If you're polling from Lean.Touch.LeanTouch.Fingers and want to quickly remove fingers that are touching the GUI, you can instead get the fingers from the Lean.Touch.LeanTouch.GetFingers method, which has a setting to quickly exclude these, as well as restrict it to a certain amount of fingers.

Why do I have to keep typing 'Lean.Touch.' before everything?

To improve organization, all Lean Touch classes are inside the Lean.Touch namespace.

If you don't like typing Lean.Touch. each time, then you can add the following code to the top of your script: using Lean.Touch;

You can now just call LeanTouch.PointOverGui(...) etc



```
using UnityEngine;
using Lean.Touch;

public class MyCoolScript : MonoBehaviour
{
    void OnEnable()
    {
        LeanTouch.OnFingerDown += OnFingerDown;
    }

    void OnDisable()
    {
        LeanTouch.OnFingerDown -= OnFingerDown;
    }

    void OnFingerDown(LeanFinger finger)
    {
    }
}
```

Why did you remove LeanTouch.PinchScale and other useful values?

These values cluttered up the main LeanTouch class. Also, they didn't allow you to easily exclude specific fingers (e.g. ones touching the GUI), and adding that requires cluttering up the class even more.

However, this code has all been moved to the new LeanGesture class, so you can now access it using LeanGesture.GetPinchScale(), etc. You can even pass this method the exact list of fingers you want (e.g. LeanGesture.GetPinchScale(LeanTouch.GetFingers(true))), which I think is much better than before!

What is Lean Touch+?

Lean Touch+ is the paid version of Lean Touch. It has the same features as Lean Touch, but comes with many more examples for you to use.

If you like Lean Touch but want to see more demo scenes or just want to support me then please consider buying it!

You can find more information about it here: https://www.assetstore.unity3d.com/#!/content/72356

Can I request a new demo scene?

Yes, if you have an idea for a demo scene that doesn’t come with LeanTouch or LeanTouch+ then please request it via e-mail above.

Just make sure your demo scene idea doesn’t require another asset or library, because I can’t include those in this package!

If I like your demo scene idea then I’ll even send you a free copy of Lean Touch+