DTT
APPS WEB GAMES

SINCE 2010

Keyboard Raiser

Documentation | 18-05-2022

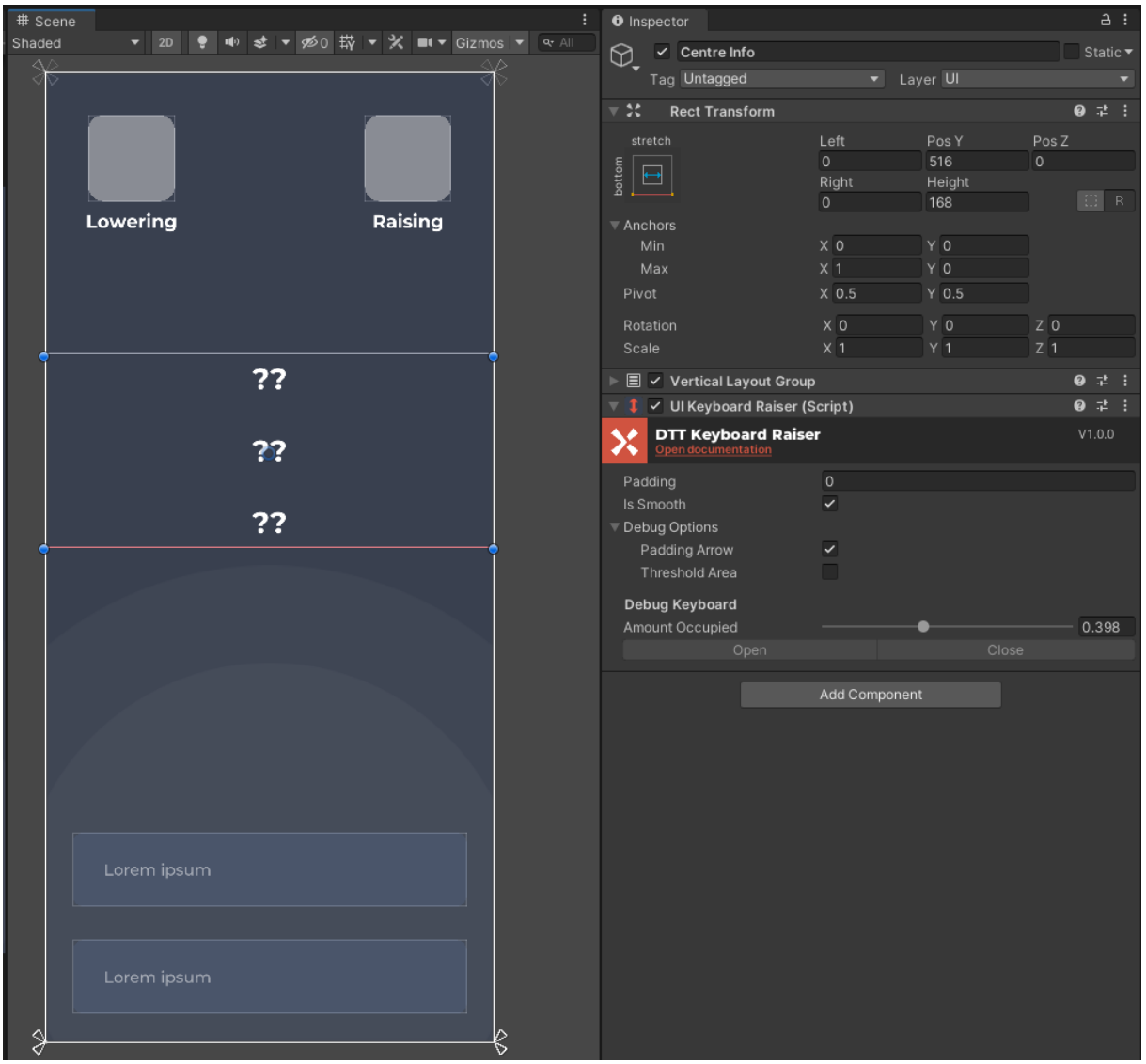
ˆ dtt logo

DTT
APPS WEB GAMES

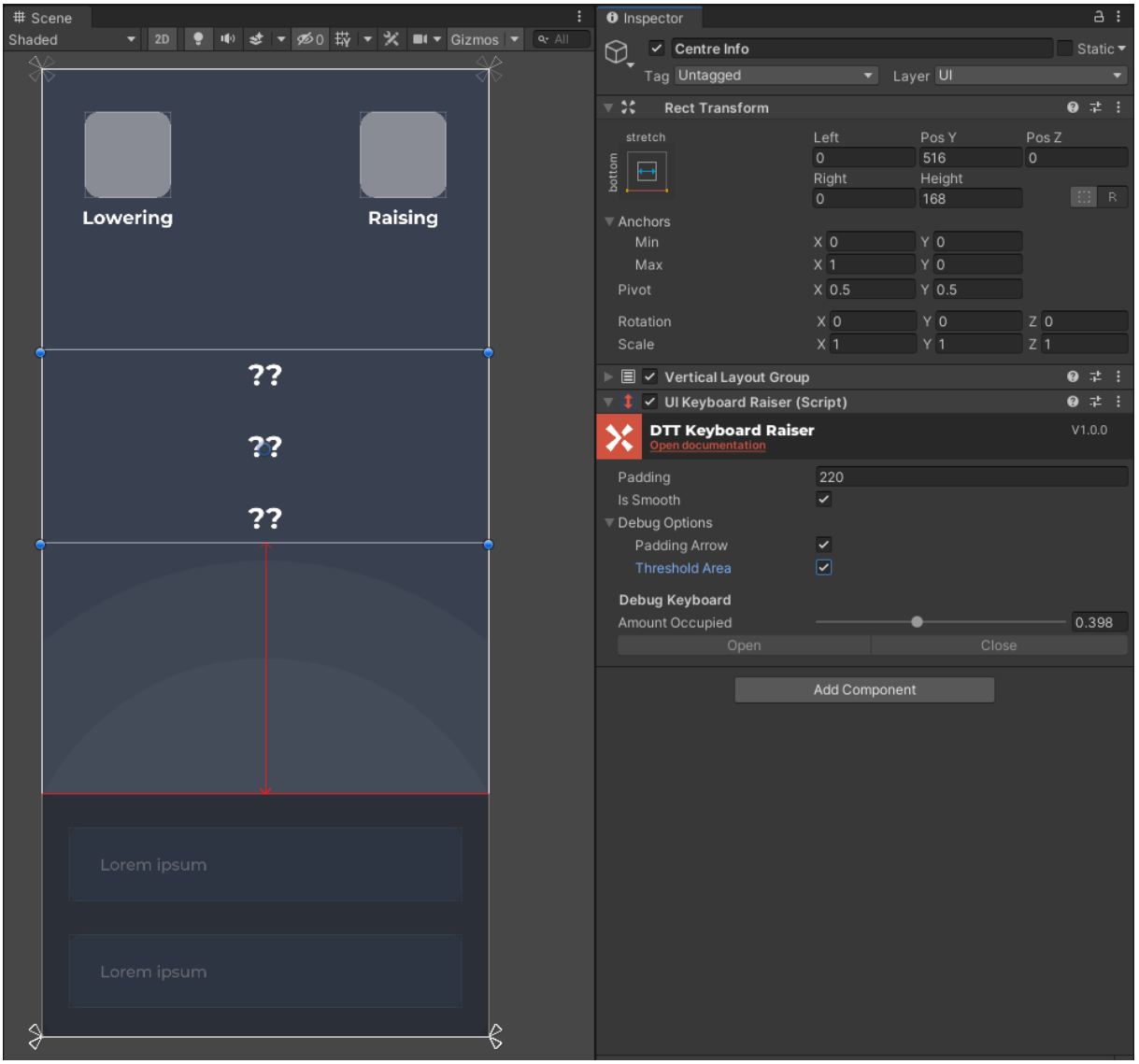
Table of Contents

1. Get started quickly

To get started you can place the UIKeyboardRaiser component on your desired UI elements. By default your UI element will now raise with the keyboard while still staying in view. You can see the current effect in the scene, the horizontal red line is the threshold from where it will start moving your object. This means that the RectTransform you placed the component on will start moving up with the keyboard as soon as the keyboard is above the red line.



You can use the **Padding** field in the inspector to make the object have some room between the keyboard and its position. This is visualised by the red vertical arrow with arrow caps.



The **IsSmooth** toggle can be used to make your UI objects move in an animated fashion to their new position when the keyboard is raised, instead of instantly popping above it.

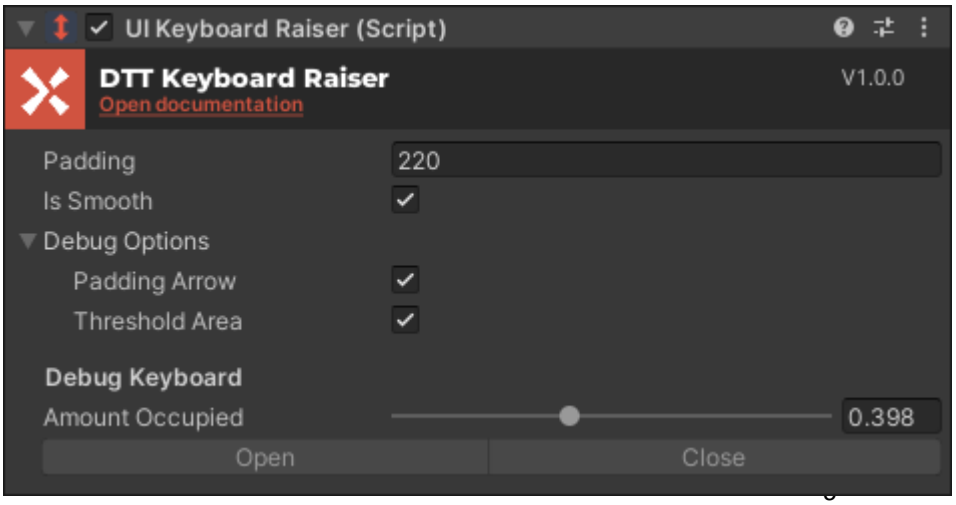
2. Introduction

This asset can be used to make sure your desired UI elements will be kept in view for the user when the keyboard covers the screen. This can be very useful to make sure input fields will stay visible even when the keyboard is active.
The problem we're trying to solve stems from the fact with how keyboards work on mobile. To be able to use them the on-screen keyboard has to cover approximately 50% of the screen. Meaning that if you're making use of any UI that requires keyboard input and is in the lower half of the screen you won't be able to see that field any more. This can be very disorientating for the user since they can't view the name of the input field and other info.

This asset means to solve that by moving up the user interface elements the developer desires when the keyboard becomes active, and back down when the keyboard is closed.

3. Editor

1. **Padding:** The amount of world space that will be maintained between your **RectTransform** and the top of your keyboard. If this is zero your gameobject will be placed directly on top of the keyboard.
2. **Is Smooth:** If this toggle is enabled your **RectTransform** will move in an animated fashion to its new position above the keyboard. If this is disabled your gameobject will be placed instantly to its target position above the keyboard.
3. **Debug Options:** Contains options related to the debugging of your UIKeyboardRaiser.
    a. **Padding Arrow:** If enabled you will see a visual of the amount of padding in the scene view. This is shown with a vertical red arrow that has arrow caps on both ends.
    b. **Threshold Area:** When toggled this will display a darkened area below the **RectTransform** and the padding. This is meant to further clarify the region in which the object will be affected by the keyboard.
4. **Debug Keyboard:** Contains global settings about the editor keyboard that can be used to visualize the effect **UIKeyboardRaiser** has on your **RectTransform**.
    a. **Amount Occupied:** This slider allows you to change the amount of space the 'mock' keyboard takes up on screen. Traditional portrait keyboards use approximately 40% of your screen, but when testing for different resolutions or device types you're free to tweak this value.
    b. **Keyboard Toolbar:** Contains the controls to open and close the 'mock' keyboard.



4. API

```
private void OnEnable()
{
    KeyboardStateManager.Current.Lowered += OnLowered;
    KeyboardStateManager.Current.Raised  += OnRaised;
}

public void Update()
{
    _heightPixelsText.text  = "Pixels: "     + KeyboardStateManager.Current.PixelHeight;
    _heightPropText.text    = "%: "          + KeyboardStateManager.Current.ProportionalHeight;
    _keyboardStateText.text = "Is raised: " + KeyboardStateManager.Current.IsRaised;
}
```

5. Known Limitations

- Changing the position or parenting of the **RectTransform** that's being affected by the **UIKeyboardRaiser** component while the keyboard is active is not supported.
- The current scene GUI implementation doesn't play well with rotated canvases (camera or world canvas), so for now the custom scene GUI is disabled if it isn't upright.

6. Support and feedback

If you have any questions regarding the use of this asset, we are happy to help you out. Always feel free to contact us at:

unity-support@d-tt.nl

(We typically respond within 1-2 business days)

We are actively developing this asset, with many future updates and extensions already planned. We are eager to include feedback from our users in future updates, be they 'quality of life' improvements, new features, bug fixes or anything else that can help you improve your experience with this asset. You can reach us at the email above.

Reviews and ratings are very much appreciated as they help us raise awareness and to improve our assets.

DTT stands for Doing Things Together

DTT is an app, web and game development agency based in the centre of Amsterdam. Established in 2010, DTT has over a decade of experience in mobile, game, and web based technology.

Our game department primarily works in Unity where we put significant emphasis on the development of internal packages, allowing us to efficiently reuse code between projects. To support the Unity community, we are publishing a selection of our internal packages on the Asset Store, including this one.

More information about DTT (including our clients, projects and vacancies) can be found here:

https://www.d-tt.nl/en/