



Paging solution for UnityUI

USER GUIDE

V1.32 (March 2017)

Copyright © 2016-2017 Digital Legacy

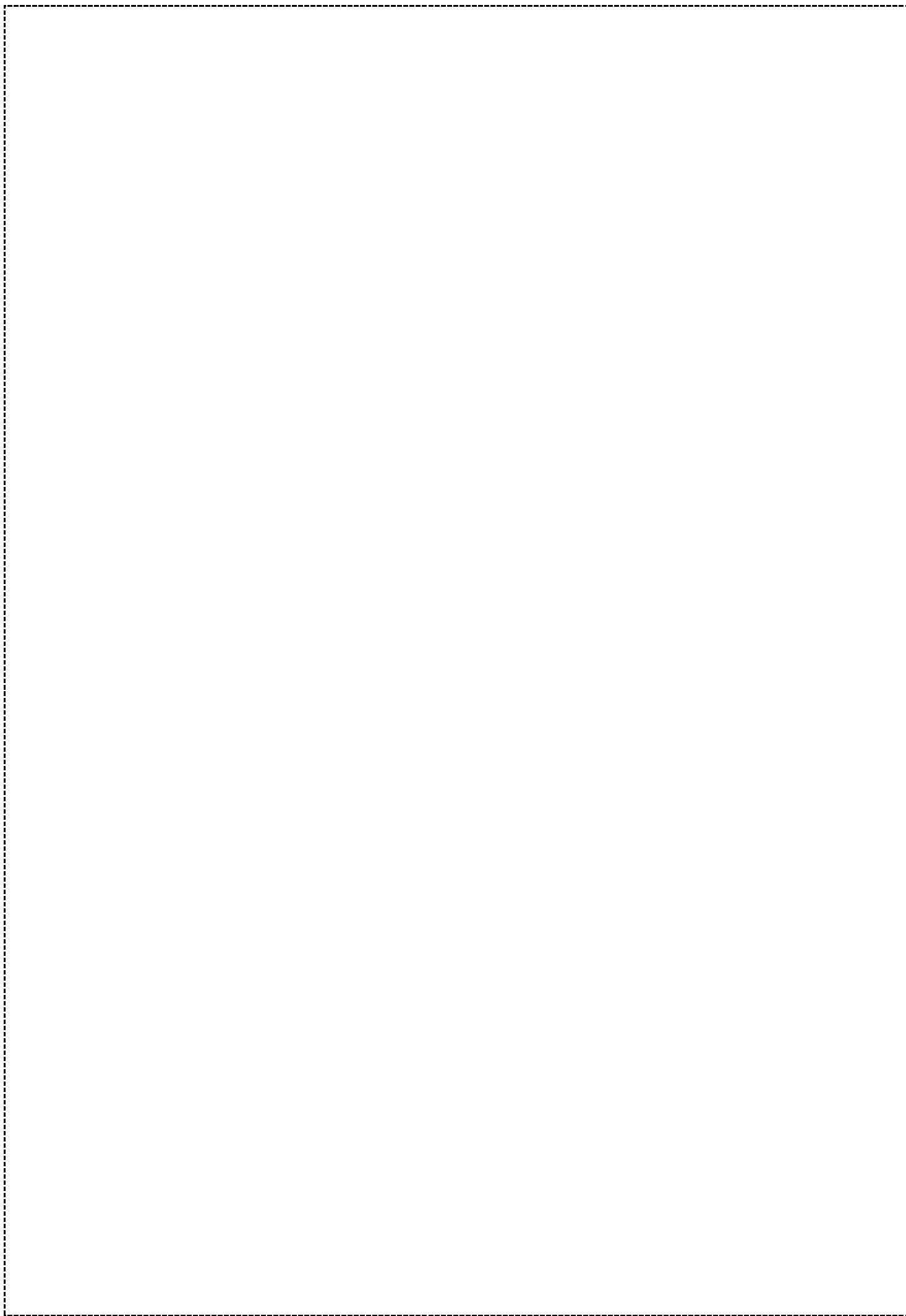


TABLE OF CONTENTS

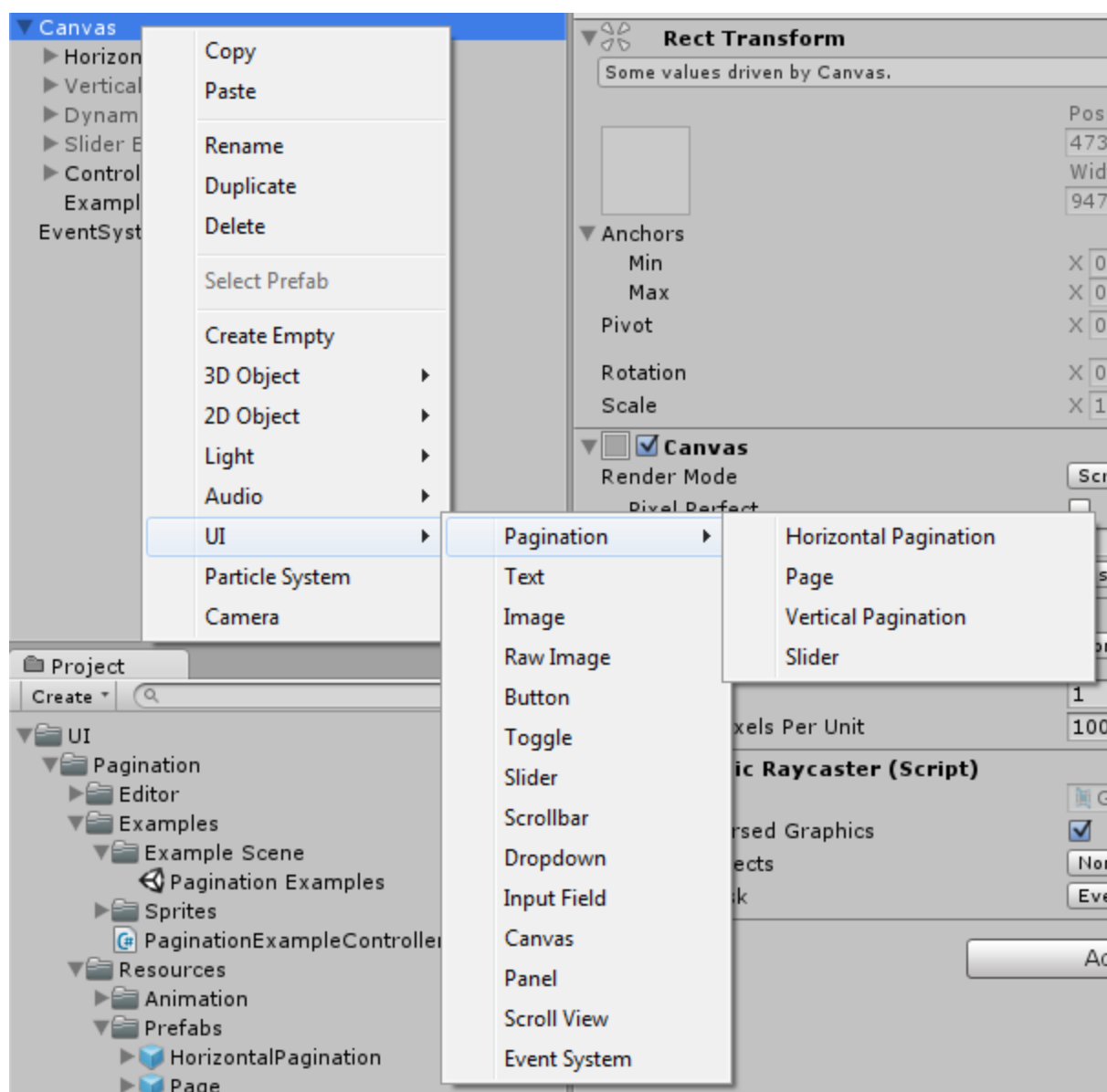
1 - Getting Started	5
1.1 Quick Start.....	5
2 - Prefabs	6
2.1 Horizontal Pagination Prefab	6
2.2 Vertical Pagination Prefab.....	6
2.3 Slider Prefab.....	6
3 - PageRect Structure	7
3.1 Viewport	7
3.2 Pagination.....	7
4 - The PagedRect Component	8
4.1 Update Pagination Button	8
4.2 Pages.....	8
4.3 Pagination.....	8
4.4 Animation	9
4.5 Automation	9
4.6 New Page Template.....	10
4.7 Keyboard Input.....	10
4.8 Mobile Input.....	10
4.9 ScrollRect	11
4.10 Scroll Wheel Input.....	11
4.11 Highlight.....	11
4.12 Events	12
4.13 Page Previews.....	12
4.14 References	14
5 – The Page Component	15
6 – Adding and Removing Pages	16
6.1 Adding Pages in the Editor	16
6.2 Removing Pages in the Editor	16
6.3 Adding Pages at runtime.....	16
6.4 Removing Pages at runtime	17
7 – Customising the appearance of the PagedRect	18
7.1 Customising the general appearance of the PagedRect	18

7.2 Customising the appearance of the buttons	18
8 – Continuous Scrolling (New in 1.1)	19
8.1 What is Continuous Scrolling?	19
8.2 The Continuous Scrolling Prefabs.....	19
8.3 Using Continuous Scrolling with an existing PagedRect	19

1 - GETTING STARTED

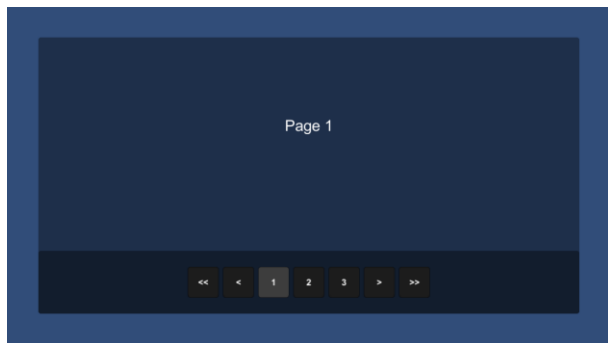
1.1 QUICK START

There are three prefabs to get you started – you can add them to your scene using the standard **GameObject** menu (under **UI -> Pagination**). Simply select the prefab which most closely matches your needs, and click the menu item to add it to the scene. If there is no **Canvas** and/or **EventSystem** in your scene, they will automatically be added for you.



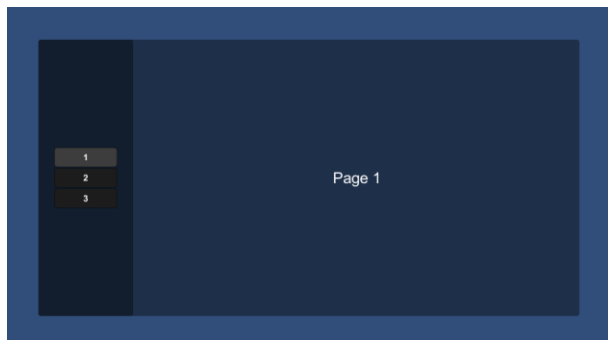
2 - PREFABS

2.1 HORIZONTAL PAGINATION PREFAB



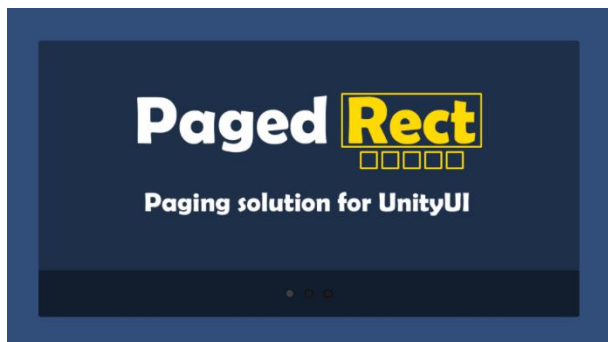
This prefab serves as a starting point for a **PageRect** with horizontal pagination.

2.2 VERTICAL PAGINATION PREFAB



This prefab serves as a starting point for a **PageRect** with vertical pagination.

2.3 SLIDER PREFAB



This prefab serves as the starting point for a slider (which can be used for, for example, an image gallery).

NEW IN 1.1: *New versions of each Prefab have been added which utilize Continuous Scrolling.*

3 - PAGERECT STRUCTURE

3.1 VIEWPORT

The *Viewport* object a) defines the visible area for the **PageRect**, and b) contains the page objects. It can also contain the optional *New Page Template* (which is used if you wish to add pages dynamically at run-time).

3.2 PAGINATION

The *Pagination* object contains buttons, and the templates used to create buttons dynamically.

- **Button – First Page**
- **Button – Previous Page**
- **Button – Next Page**
- **Button – Last Page**

These buttons may be customised however you like, should you wish to use them. They are optional.

- **Button Template (Current Page)**

This is a template which defines the appearance of the page button representing the current page.

- **Button Template (Other Pages)**

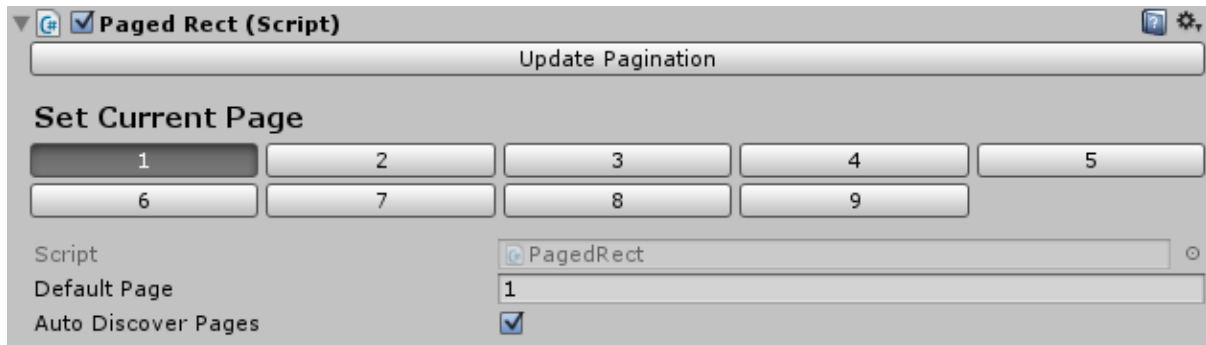
This is a template which defines the appearance of page buttons representing pages other than that of the current page.

- **Button Template (Disabled Page)**

This is a template which defines the appearance of page buttons representing pages that are disabled. This template is only required if you intend to use disabled pages (such as, for example, in a Wizard)

The *Pagination* object also contains the actual buttons themselves – using the format **Button – Page X** (where X refers to the page number). You should avoid making any changes to these buttons directly – rather edit the templates instead. See section 7 for more details.

4 - THE PAGEDRECT COMPONENT



4.1 UPDATE PAGINATION BUTTON

Clicking this button will force the PagedRect to update its state to match its contents. Normally this will happen automatically, but in some cases it may be necessary to click the button to force an update.

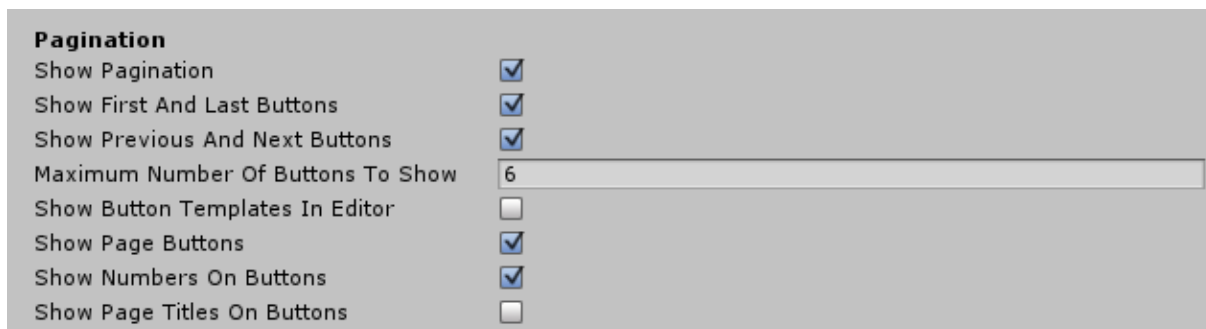
4.2 PAGES

Set Current Page – This section will allow you to set the currently visible page of this PagedRect in the editor. This will make the current page visible, and hide all others. This will not affect the PagedRect during runtime, only in the editor.

Default Page – This specifies the default “Current Page” of this PagedRect during runtime.

Auto Discover Pages – If this property is enabled, this PagedRect will automatically detect page objects contained in the *Viewport* and create buttons/etc. for them. If this property is disabled, you will need to maintain the *PagedRect.Pages* property yourself (not recommended, but this will allow you to, for example, reference pages that are not contained within the Viewport).

4.3 PAGINATION



Show Pagination – If this property is disabled, the *Pagination* object will be hidden and no buttons will be visible.

Show First and Last buttons – Show or Hide the *First* and *Last* buttons.

Show Previous And Next Buttons – Show or Hide the *Previous* and *Next* buttons.

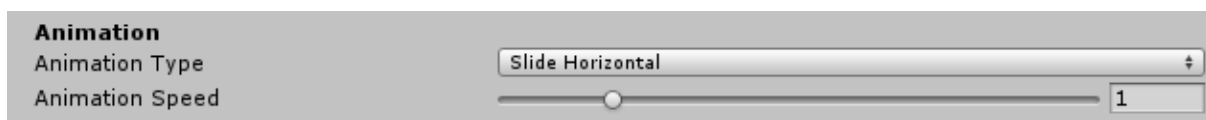
Maximum Number Of Buttons To Show – How many buttons (not counting First/Last/Next/Previous) should be visible at once? (0 = No Limit)

Show Button Templates In Editor – If this property is enabled, the *Button Templates* (for the Current Page, Other Pages, and Disabled Pages) will be visible in the editor (but not at runtime). Enable this property when you wish to edit the appearance of buttons, and disable it when you wish to see the Pagination as it will appear at runtime.

Show Numbers On Buttons – If this property is enabled, buttons will show page numbers.

Show Page Titles On Buttons – If this property is enabled, buttons will show page titles (which are drawn from *Page.PageTitle*)

4.4 ANIMATION

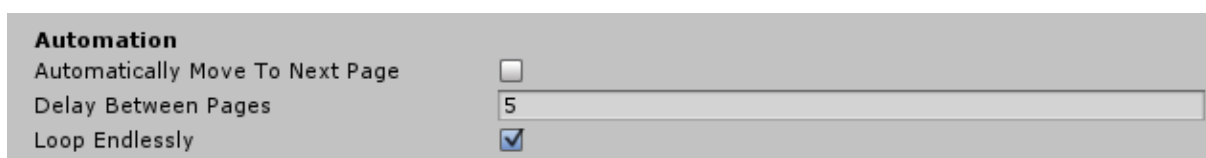


Animation Type – Specifies what type of animation (if any) is shown when switching from one page to another. The options available are *None*, *Slide Horizontal*, *Slide Vertical*, and *Fade*.

Animation Speed – Specifies how fast (or how slow) the animation should be played.

*Please note: if Continuous Scrolling is enabled, the **Animation Type** setting will be not be used.*

4.5 AUTOMATION

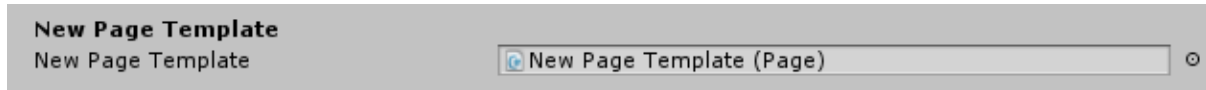


Automatically Move to Next Page – If this property is enabled, this **PageRect** will automatically move to the next page after *Delay Between Pages*.

Delay Between Pages – Specifies how long (in seconds) this **PageRect** should wait before moving on to the next page.

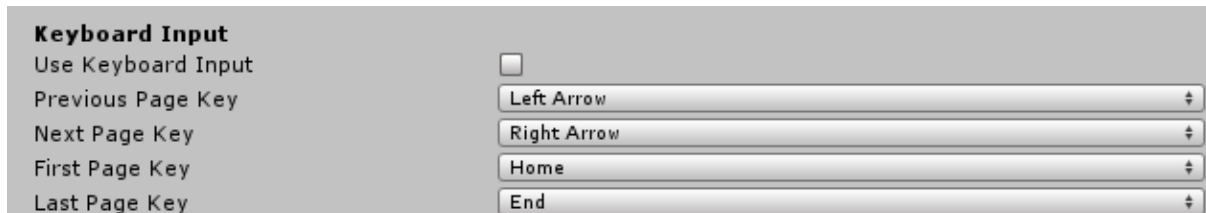
Loop Endlessly – If this property is enabled, this **PageRect** will automatically return to the first page after reaching the last page (after *Delay Between Pages*). **Note: This will also apply to reaching the last/first pages when using the next/previous buttons (or any other kind of input).**

4.6 NEW PAGE TEMPLATE



This specifies a *Page* object which will be used as the template when adding new pages dynamically to this **PageRect**. This is optional, if you do not intend to add pages at runtime, then you do not need to modify this property at all. More details on dynamically adding pages can be found in Section 6.

4.7 KEYBOARD INPUT



Use Keyboard Input – If this property is enabled, they keyboard can be used to navigate this **PageRect**.

Previous Page Key – Specifies the key to move to the previous page (Default = *Left Arrow*).

Next Page Key – Specifies the key to move to the next page (Default = *Right Arrow*).

First Page Key – Specifies the key to move to the first page (Default = *Home*).

Last Page Key – Specifies to the last page (Default = *End*).

4.8 MOBILE INPUT



Use Swipe Input – If this property is enabled, mobile gestures (swipe left/right/up/down) may be used with this **PageRect**. These will also work if done with the mouse.

Please note: this setting has no effect for Scrollrect-based PagedRects, with the exception of 'Page Previews'.

4.9 SCROLLRECT

ScrollRect	
Swipe Delta Threshold	0.1
Space Between Pages	0
Loop Seamlessly	<input checked="" type="checkbox"/>
Show Scroll Bar	<input type="checkbox"/>

Swipe Delta Threshold – Specifies the minimum swipe distance before PagedRect will switch to the next or previous pages.

Space Between Pages – Adds an optional space between pages in the ScrollRect.

Loop Seamlessly - If this is set to true, then PagedRect will seamlessly move the first page to the end and vice versa as necessary so as to provide the illusion that the PagedRect is looping infinitely.

Show Scroll Bar – If this is set to true, then the ScrollRect's Scroll Bar will be visible.

4.10 SCROLL WHEEL INPUT

Scroll Wheel Input	
Use Scroll Wheel Input	<input type="checkbox"/>
Only Use Scroll Wheel Input When Mouse	<input checked="" type="checkbox"/>

Use Scroll Wheel Input – If this property is enabled, scrolling the mouse wheel up or down will move to the previous or next page.

Only Use Scroll Wheel Input When Mouse Is Over – If this property is enabled, scroll wheel input will only be used when the mouse is hovering over the PagedRect (if this property is not enabled, the PagedRect will respond to scroll wheel input regardless of where the mouse is).

4.11 HIGHLIGHT

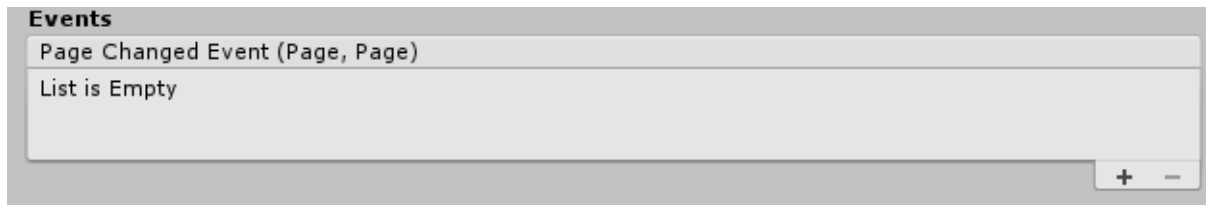
Highlight	
Highlight When Mouse Is Over	<input type="checkbox"/>
Normal Color	<div style="background-color: black; width: 100px; height: 15px;"></div>
Highlight Color	<div style="background-color: black; width: 100px; height: 15px;"></div>

Highlight When Mouse Is Over – If this property is enabled, this PageRect's *Image* component will have its color changed when the mouse is over it.

Normal Color – What color should this PageRects *Image* component be when the mouse is **not** over it?

Highlight Color – What color should this PageRects *Image* component be when the mouse is over it?

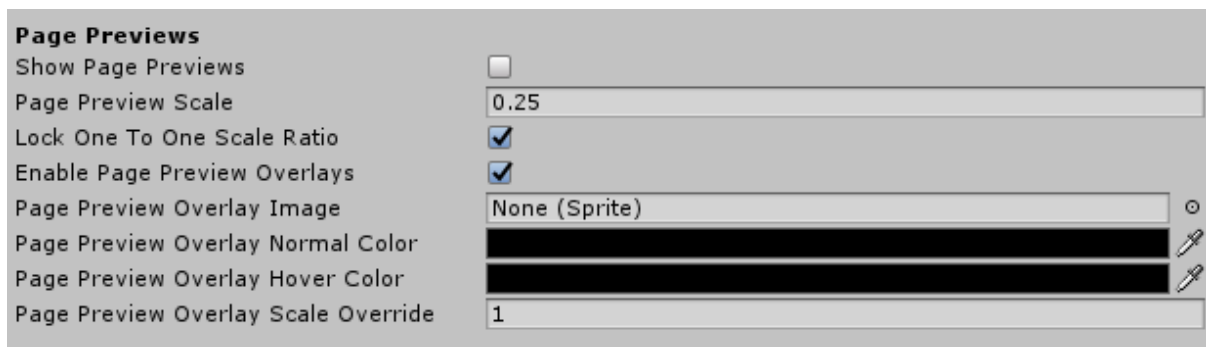
4.12 EVENTS



Page Changed Event – This will allow you to trigger events whenever the current page is changed through any means – the first *Page* argument is the **New Page** and the second *Page* argument is the **Previous Page**.

4.13 PAGE PREVIEWS

Please note: Page Previews are only available for ScrollRect based PagedRects (Continuous Scrolling).



Show Page Previews – Enables or disables the Page Previews functionality.

Page Preview Scale – Specifies the scale to use for the preview pages. The primary page will also be scaled so as to use the remaining available space. This should always be lower than 0.33, otherwise the preview pages will be larger than the main page.

Lock One To One Scale Ratio – If this is false, then the previews will only be scaled vertically (in horizontal mode) or horizontally (in vertical mode). If this is true, then the previews will be scaled by the same amount in both dimensions.

Enable Page Preview Overlays – Page Preview Overlays are used to a) optionally highlight the preview pages when the mouse hovers over them, and b) allow the user to move to the page by clicking on it. If they are enabled, they will prevent any clicks from going through to the preview pages (but not the current page).











Page Preview Overlay Image – The optional image to use for the preview overlay.

Page Preview Overlay Normal Color – The normal color of the preview overlay.

Page Preview Overlay Hover Color – The color of the preview overlay when the mouse is over it.

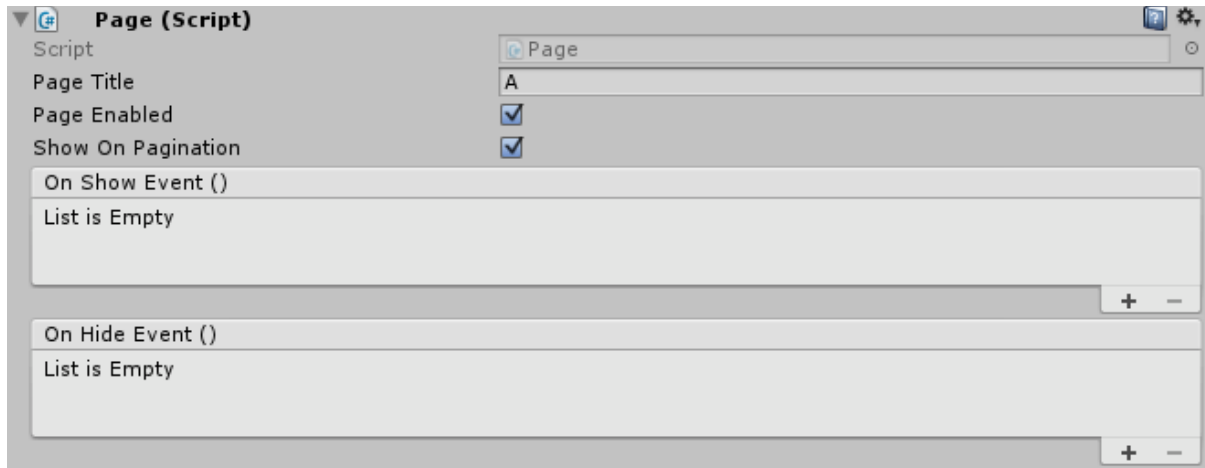
Page Preview Overlay Scale Override – This property allows you to override the scale value used by the overlays in the vertical dimension (for horizontal PagedRects) or horizontal dimension (for vertical PagedRects), allowing you to create overlays which extend beyond the page itself.

4.14 REFERENCES

References	
Viewport	 Viewport
Pagination	 Pagination
Button Template_Current Page	 Button Template (Current Page) (PaginationButton)
Button Template_Other Pages	 Button Template (Other Pages) (PaginationButton)
Button Template_Disabled Page	 Button Template (Disabled Page) (PaginationButton)
Button_Previous Page	 Button - Previous Page (PaginationButton)
Button_Next Page	 Button - Next Page (PaginationButton)
Button_First Page	 Button - First Page (PaginationButton)
Button_Last Page	 Button - Last Page (PaginationButton)
Animation Controller Template	 PageAnimationController
Pages	
Editor Selected Page	1

This section defines references this **PagedRect** will use to locate, for example, the *Viewport* and *Pagination* objects. There should be no need for you to modify this section unless you are building a new **PagedRect** from scratch rather than using a prefab as a starting point.

5 – THE PAGE COMPONENT



Page Title – This optional field defines the title of this page. This will be used if the **PagedRect** is set to *Show Page Titles On Buttons*.

Page Enabled – If this field is not set, then the button for this page will use the *Button Template (Disabled Page)* for its appearance, and will not respond to mouse clicks. You can still set this page as the current page using code, however.

Show on Pagination – If this field is cleared, this page will not have a button in the pagination at all. This will allow you to add hidden pages which cannot be accessed through the regular page buttons.

On Show Event() – Any events set here will be called whenever this page is shown.

On Hide Event() – Any events set here will be called whenever this page is hidden.

6 – ADDING AND REMOVING PAGES

6.1 ADDING PAGES IN THE EDITOR

There are two ways to add a new page in the editor. The simplest way is to:

Method One

- a. Select the *Viewport* object of the **PagedRect** in the hierarchy.
- b. Right-click on the *Viewport* object (or click on the *GameObject* menu)
- c. Select UI -> **Pagination** -> **Page**

This will add a *Page* prefab to the scene, update the **PagedRect** pagination, and select the new page. Please note that this will **not** use the **PagedRects** *New Page Template*.

Method Two

- a. Select an existing page in the hierarchy of this **PagedRect**.
- b. Right-click on the page, and click **Duplicate**.

You can now customise the template as you choose. You will also need to select the **PagedRect** and click *Update Pagination* in order to produce the button for the new page.

6.2 REMOVING PAGES IN THE EDITOR

Pages can be removed simply by deleting them from the object hierarchy – the *Update Pagination* button can be used to update the pagination buttons.

6.3 ADDING PAGES AT RUNTIME

Two functions have been provided for adding pages at runtime:

PagedRect.AddPage (Page page)

Use this function when you wish to instantiate a *Page* object on your own, and then add it to the **PagedRect**. This function will add it to the *Viewport* in the hierarchy and trigger a pagination update.

PagedRect.AddPageUsingTemplate ()

This function will instantiate a new *Page* object (using the *New Page Template*), add it to the **PagedRect** (as per *AddNewPage*) and **return it**, so you can customise it as necessary.

6.4 REMOVING PAGES AT RUNTIME

A function has been provided to remove pages at runtime:

PagedRect.RemovePage(Page page, bool destroyPageObject = false)

This function will remove *page* from this **PageObject** and update it. If *destroyPageObject* is true, then the *Page* will also be destroyed.

7 – CUSTOMISING THE APPEARANCE OF THE PAGEDRECT

7.1 CUSTOMISING THE GENERAL APPEARANCE OF THE PAGEDRECT

The **PagedRect** is made out of standard Unity UI components, and as such, can easily be modified as per usual, in any way you like.

By default:

- The **PagedRect** GameObject is set up as a normal Unity UI panel – it contains an *Image* component which provides the background image for the **PagedRect**.
- The *Viewport* GameObject uses a *Mask* component to define the visible area.
- The *Pagination* GameObject is also set up as a normal Unity UI panel.

7.2 CUSTOMISING THE APPEARANCE OF THE BUTTONS

General

The *Horizontal Pagination* prefab uses a *Horizontal Layout Group* component to position and size buttons. Similarly, the *Vertical Pagination* prefab uses a *Vertical Layout Group*.

Specific Buttons

Each button in the prefabs uses a *Layout Element* component to set their width and height. Beyond that, they are standard Unity UI *Button* GameObjects and can be customised as you choose.

You can customise the appearance of the static buttons – First Page, Previous Page, Next Page, and Last Page as you choose.

To customise the appearance of page buttons, you can edit the *Current Page*, *Other Pages*, and *Disabled Pages* templates respectively. At any time you can use the **PagedRects Update Pagination** button to trigger an update and see your changes.

Note: By default, the template buttons are hidden so you can see the pagination as it will appear at runtime. To show the template buttons in the editor, set the **Show Button Templates In Editor** property of the **PagedRect**.

8 – CONTINUOUS SCROLLING (NEW IN 1.1)

8.1 WHAT IS CONTINUOUS SCROLLING?

Added in v1.1, Continuous Scrolling allows you to create PagedRect instances which implement a standard Unity **ScrollRect** in order to allow you to scroll through each of the pages by dragging from one to the next instead of utilizing regular animations. For an example of Continuous Scrolling in action, please see the example scene – **Slider – ScrollRect**.

Continuous scrolling works in both vertical and horizontal paging scenarios, although it involves configuring multiple objects.

8.2 THE CONTINUOUS SCROLLING PREFABS

In order to keep continuous scrolling as easy to implement as possible, 3 new prefabs have been added to the PagedRect **Pagination** menu in v1.1. These prefabs are the same as those that predate v1.1, however, they have been set up to support Continuous scrolling. They are all named as before, with the addition of “- **ScrollRect**” at the end of their names, e.g. **Horizontal Pagination – ScrollRect**.

Please note that these prefabs do **not** support animation (e.g. Fade In / Fade Out) as the animations have been replaced by the dragging system. If you wish to use animations instead of continuous scrolling, please use the original prefabs instead.

8.3 USING CONTINUOUS SCROLLING WITH AN EXISTING PAGEDRECT

Existing PagedRects can be updated to use continuous scrolling, however there is some complex setup involved:

1. Add a **Scroll Rect** component to the **PagedRect** object (via **Add Component**).
 - a. Set the **ScrollRect** to be **Horizontal** or **Vertical** (but not both)
 - b. Disable **Inertia**
 - c. Set **Scroll Sensitivity** to 0 (**PagedRect** -> **Use Scroll Wheel Input** still functions as before)
2. Add a new **GameObject** to the **PagedRects Viewport**. Name it **Content**.
 - a. Move all existing pages (including the new page template, if one is in use) to the **Content** object.
 - b. Add a **Horizontal Layout Group** or **Vertical Layout Group** component to the **Content** object.
 - c. Add a **Content Size Fitter** component to the **Content** object.
 - i. If using a Horizontal PagedRect, then set **Horizontal Fit** to “Preferred Size” and **Vertical Fit** to “Unconstrained”.
 - ii. If using a Vertical PagedRect, then set **Vertical Fit** to “Preferred Size” and **Horizontal Fit** to “Unconstrained”.
3. Add a new **GameObject** to the **PagedRects Viewport**. Name it **Sizing Panel**.
 - a. Set the **Sizing Panel** to fill its container (stretch, with no spacing)
4. On the **PagedRect** component:
 - a. Set **References** -> **Scroll Rect** to the **PagedRect** GameObject.
 - b. Set **References** -> **Scroll Rect Viewport** to the **Viewport** GameObject.
 - c. Set **References** -> **Viewport** to the **Content** GameObject.
 - d. Set **References** -> **Sizing Transform** to the **Sizing Panel**.
5. On the **Scroll Rect** component:

- a. Set **Content** to the **Content** GameObject Transform.
 - b. Set **Viewport** to the **Viewport** GameObject Transform.
6. Finally, select the **PagedRect** component and set the Current page to any value you wish – this will trigger an update which will finalise the configuration of the PagedRect.

If you wish to revert this change at any point, simply remove the **ScrollRect** component from the **PagedRect** and then set the current page to any value you wish. The presence of the **ScrollRect** component is what is used to determine whether or not the PagedRect should operate in Continuous Scrolling mode.