

---

# EasyLayout Documentation

*Release 1.7.0f1*

**Ilia Novikov**

**Mar 26, 2022**



# CONTENTS

<b>1</b>	<b>EasyLayout</b>	<b>1</b>
1.1	Options . . . . .	1
1.2	Events . . . . .	5
<b>2</b>	<b>Layout Switcher</b>	<b>7</b>
<b>3</b>	<b>Changelog</b>	<b>9</b>
3.1	Release 1.6.0 . . . . .	9
3.2	Release 1.6.0 . . . . .	9
3.3	Release 1.5.0 . . . . .	9
3.4	Release 1.4.0 . . . . .	9
3.5	Release 1.3.1 . . . . .	9
3.6	Release 1.3.0 . . . . .	10
3.7	Release 1.2.1 . . . . .	10
3.8	Release 1.2.0 . . . . .	10
3.9	Release 1.1.1 . . . . .	10
3.10	Release 1.1.0 . . . . .	10
3.11	Release 1.0.5 . . . . .	11
3.12	Release 1.0.4 . . . . .	11
3.13	Release 1.0.3 . . . . .	11



## EASYLAYOUT

EasyLayout provides different layouts that not available with default layout groups.

### 1.1 Options

- Main Axis `Axis`

Determine how elements will be placed (at horizontal or vertical direction first).

- Layout Type `LayoutTypes`

- Compact: Compactly places the elements.
- Grid: Places elements in the grid. Cell size is not fixed and depend on elements sizes in the same row and column.
- Flex: Places elements like CSS flexbox layout.
- Staggered: Places elements one-by-one to the shortest column or row depending on the main axis.
- Ellipse: Places elements one-by-one on the border of the ellipse or the circle starting from `Angle Start` and `Angle Step` distance between items.

- Group Position `Anchors`

Only for the `Compact` and `Grid` layouts.

Combination of horizontal (`Left`, `Center`, `Right`) and vertical (`Upper`, `Middle`, `Lower`) positions.

Elements combine to the group, this option specifies group position relative to the parent.

- Row Align `HorizontalAligns`

Only for the `Compact` layout.

Element position in the row (`Left`, `Center`, `Right`).

- Inner Align `InnerAligns`

Only for the `Compact` layout.

Column position relative to the group (`Top`, `Middle`, `Bottom`).

- Compact Constraint `CompactConstraints`

Only for the `Compact` layout.

- Flexible: Rows and columns count depends on the parent size.
- Max Column Count

- Max Row Count
- Compact Constraint Count `int`
  - Only for the Compact layout.
  - Max count of the rows or columns for the Compact Constraint option.
- Cell Align Anchors
  - Only for the Grid layout.
  - Elements position relative to the cell size. Same as Group Position.
- Grid Constraint `GridConstraints`
  - Only for the Grid layout.
  - Flexible: Rows and columns count depends on the parent size.
    - Fixed Column Count
    - Fixed Row Count
- Grid Constraint Count `int`
  - Only for the Grid layout.
  - Count of the rows or columns for the Grid Constraint option.
- Flex Setting `EasyLayoutFlexSettings`
  - Only for the Flex layout.
  - Wrap `bool`
      - If disabled elements will all placed onto one line (row or column).
    - Justify Content `EasyLayoutFlexSettings.Content`
      - Alignment along the main axis. Also distribute extra free space on the main axis.
      - \* Start: elements placed at the start of the line.
      - \* Center: elements placed at the center of the line.
      - \* End: elements placed at the end of the line.
      - \* Space Between: first element at the start of the line, last element at the end of the line, other elements placed between them with evenly spacing.
      - \* Space Around: first and last elements are placed with  $1n$  space from the edges, other elements placed with  $2n$  space between them.
      - \* Space Evenly: elements are placed so that the spacing between any two element and the space to the edges is equal.
    - Align Content `EasyLayoutFlexSettings.Content`
      - Alignment of the lines (columns or rows) along the cross axis. Also distribute extra free space on the cross axis.
      - \* Start: lines placed to the start of the parent.
      - \* Center: lines placed to the center of the parent.
      - \* End: lines placed to the end of the parent.
      - \* Space Between: first line to the start of the parent, last line to the end of the parent, other lines placed between them with evenly spacing.

- \* `Space Around`: first and last lines are placed with  $1n$  space from the edges, other lines placed with  $2n$  space between them.
  - \* `Space Evenly`: line are placed so that the spacing between any two lines and the space to the edges is equal.
- `Align Items EasyLayoutFlexSettings.Items`  
Define how elements are placed out along the cross axis on the line (column or row).
  - \* `Start`
  - \* `Center`
  - \* `End`
- `Staggered Settings EasyLayoutStaggeredSettings`  
Only for the `Staggered` layout.
  - `Fixed Block Count bool`  
Count of the rows or columns.
  - `Blocks Count int`
- `Ellipse Settings EasyLayoutEllipseSettings`  
Only for the `Ellipse` layout.  
Set equal width and height for the circle layout.  
`RectTransform` pivot is used as the center of the ellipse.
  - `Width Auto bool`  
`RectTransform` width is used as the width of the ellipse.
  - `Width float`  
Ellipse width if `Width Auto` disabled.
  - `Height Auto bool`  
`RectTransform` height is used as the height of the ellipse.
  - `Height float`  
Ellipse height if `Height Auto` disabled.
  - `Angle Start float`  
Position of the first element in the degrees.
  - `Angle Step Auto bool`  
Are elements placed with equal angular distance or specified `Angle Step`?
  - `Angle Step float`  
Elements placed with specified angular distance between neighbour elements.
  - `Fill EllipseFill`  
Determines how to calculate the distance between elements if `Angle Step Auto` enabled.
    - \* `Closed`: angular distance is 360 degrees divided into the elements count; distance is the same between the first and last elements.

- \* `Arc`: angular distance is arc length divided into the elements count minus one
- `Arc Length float`
  - Distance between first and last elements if `Angle Step Auto` enabled and `Fill` is `Arc`.
  - Can be more than 360 degrees.
- `Align EllipseAlign`
  - Determines how elements are placed on the ellipse border.
  - \* `Outer`: right borders of the elements are placed on the ellipse border.
  - \* `Center`: center of the elements are placed on the ellipse border.
  - \* `Inner`: left borders of the elements are placed on the ellipse border.
- `ElementsRotate bool`
  - Rotate elements according to position or not.
- `ElementsRotationStart float`
  - Initial rotation of the elements.
- `Spacing Vector2`
  - Empty space between elements.
  - Can be more than specified value for `Flex` layout.
- `Symmetric bool`
  - Use symmetric margin.
- `Margin Vector2`
  - Empty space from parent edges.
- `Skip Inactive bool`
  - Do not reserve space for disabled elements.
- `Right To Left bool`
  - The order of placement of elements.
- `Top To Bottom bool`
  - The order of placement of elements.
- `Reset Rotation bool`
  - Reset rotation of the elements to 0.
- `Movement Animation bool`
  - Animate elements repositioning.
- `Movement Curve AnimationCurve`
  - Movement animation curve.
- `Resize Animation bool`
  - Animate elements resizing.
- `Resize Curve AnimationCurve`



Resize animation curve.

- Children Width ChildrenSize

- Do nothing: do not resize elements.
- Set Preferred: set element width to Preferred Width.
- Set Max From Preferred: set maximum of the Preferred Width from the all elements.
- Fit Container: change children size in range from minimal to preferred to fit container.
- Set Preferred and Fit Container: set children size to preferred, then increase size proportionally Flexible Width to fit parent width if required.
- Shrink On Overflow: decrease elements width if summary width more than parent width including margin.

- Children Height ChildrenSize

Similar to Children Width

## 1.2 Events

- Settings Changed UnityEvent

Event, raised after any setting was changed.



## LAYOUT SWITCHER

Allow to create different layouts with same GameObjects for different screen sizes and aspect ratios. Used when anchors, pivots and layout group not enough to create layout with different aspect ratios support.

Save values of the position, size, anchors, pivot, rotation, scale, active/disable state for each layout.

### Options

- *Objects*: list of the controlled objects.
- *Default Display Size* (inches): display size to use when actual display size cannot be detected.
- *Layouts*: list of the layouts.
  - *Name*: layout name.
  - *Aspect Ratio*: aspect ratio for this layout.
  - *Max Display Size* (inches): maximum size of the display for this layout (layout will not be used if display size more than specified).



## CHANGELOG

### 3.1 Release 1.6.0

- EasyLayout: added optional movement and resize animation support; warning: can decrease performance

### 3.2 Release 1.6.0

- EasyLayout: small improvements and bugfixes
- EasyLayout: Filter property is obsolete and replaced with ShouldIgnore

### 3.3 Release 1.5.0

- EasyLayout: reduced memory allocation
- EasyLayout: minor bug fixes

### 3.4 Release 1.4.0

- EasyLayout extensions methods moved to EasyLayoutNS.Extensions namespace
- EasyLayout: added SetPreferredAndFitContainer option for the Children Size
- EasyLayout: added GetElementPosition to get position in group

### 3.5 Release 1.3.1

- EasyLayout: added ElementsRotate and ElementsRotationStart for Ellipse layout

## 3.6 Release 1.3.0

- added EasyLayoutEllipseScroll
- EasyLayout: added new layout type Ellipse
- EasyLayout: added new option ResetRotation

## 3.7 Release 1.2.1

- fixed FitContainer option

## 3.8 Release 1.2.0

- added Flex layout type
- added Staggered layout type
- renamed Stacking to MainAxis
- reduced memory allocations
- EasyLayout namespace renamed to EasyLayoutNS to avoid problems with Unity 2018.2 and later
- fixed “dirty” scene bug when using FitContainer or ShrinkOnOverflow
- added Shrink on Overflow option
- added CompactConstraint and CompactConstraintCount options
- added row/column constraint for Grid layout

## 3.9 Release 1.1.1

- improved compatibility with Unity 2017.x

## 3.10 Release 1.1.0

- bug fixes
- performance improvements

### 3.11 Release 1.0.5

- bug fixes.

### 3.12 Release 1.0.4

- improved performance
- bug fixes.

### 3.13 Release 1.0.3

- renamed to EasyLayout