EasyLayout Documentation

Release 1.9.0f1

Ilia Novikov

Sep 14, 2024

CONTENTS

CHAPTER

ONE

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 horizonal 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.
  - `Hex`: Places elements in the hexagonal grid.

- Group Position `Anchors`

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

  Combination of horizonal (`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 `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.

- Hex Settings `EasyLayoutHexSettings`

  - Orientation `OrientationMode`

    * `FlatTop` Flat-top orientation.

    * `PointyTop` Pointy-top orientation.

  - Coordinates `CoordinatesMode`

    * `Read` Read coordinates from the `HexCoordinates` component and place components in the grid according to those coordinates.

    * `Write` Automatically places components to the grid and writes calculated coordinates to the `HexCoordinates` component.

  - Shoves Odd `bool` If enabled shoves odd rows to the bottom (if `FlatTop`) or right (if `PointyTop`); otherwise, shoves even rows.

  - Constraint `HexConstraints`

    * `Flexible` No constraints.

    * `FixedColumnCount` Constraint the number of columns to a specified number.

    * `FixedRowCount` Constraint the number of rows to a specified number.

    * `CellsPerRow` Constraint the cells per row to a specified number.

    * `CellsPerColumn` Constraint the cells per column to a specified number.

  - ConstraintCount `int` Number for the specified constraint.

  - `Decrease Shoved` Shoved rows or columns will have 1 cell less than the specified constraint (only for the `CellsPerRow` and `CellsPerColumn` constraint).

- Spacing `Vector2`

Empty space between elements.

Can be more than specified value for `Flex` layout.

For `Hex` layout it is recommended to have following ratio:

  - `FlatTop`: `Y` should be `X` / 2
  - `PointyTop`: `Y` should be `X` * 2

- 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 Animate All `bool`

  Animate all elements if enabled; otherwise new elements will not be animated.

- Movement Curve `AnimationCurve`

  Movement animation curve.

- Resize Animation `bool`

  Animate elements resizing.

- Resize Animate All `bool`

  Animate all elements if enabled; otherwise new elements will not be animated.

- Resize Curve `AnimationCurve`

  Resize animation curve.

- IgnoreLayoutElementSizes `bool`

  ILayoutElement options will be ignored. Increases performance without side effects if `Children Width` and `Children Height` are not controlled.

- 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`

  The event is raised when any setting is changed.

CHAPTER

TWO

EASYLAYOUTELLIPSESCROLL

Scroll for the `EasyLayout` with `Ellipse` layout type.

2.1 Options

- IsHorizontal `bool`

  Is scroll horizontal or vertical?

- DragSensitivity `float`
- ScrollSensitivity `float`
- ScrollValue `float`

  Scroll position.

- Inertia `bool`
- TimeToStop `float`

  Time until inertia stopped.

- UnscaledTime `bool`

  Animate inertia scroll with unscaled time.

- DragButton `PointerEventData.InputButton`

  The button that should be pressed to process the drag event.

2.2 Events

- OnScrollEvent `UnityEvent`
- ScrollVelocity `UnityEvent`

CHAPTER THREE

HEXAGONAL GRID COORDINATES

Each game object under the `EasyLayout` control with the `Hex` layout has a `HexCoordinates` component.

It provides access to the coordinates of the object in the grid in the different coordinate systems: `OffsetCoordinates` and `CubeCoordinates`.

Also, this component provides coordinates for layout in case `Coordinates` is `Read` mode.

See more in *the guide <https://www.redblobgames.com/grids/hexagons/>* about hexagonal grids.

Note: You can use `HexCoordinatesDebug` component with `TextAdapter` component to display cells coordinates for the debug.

3.1 Options

- Row `int`
- Column `int`

3.2 Events

- On Coordinates Changed `UnityEvent`

  The event is raised when coordinates are changed.

CHAPTER

FOUR

HEXLAYOUTBUILDER

This component simplifies the creation of custom grids.

- TemplateFlatTop `HexCoordinates`

  Cell template for the `FlatTop` orientation.

- TemplatePointyTop `HexCoordinates`

  Cell template for the `PointyTop` orientation.

- Blocks `List<Block>`

  If the EasyLayout.Orientation is FlatTop then each block describes a column. Otherwise, each block describes a row.

- Instances `IReadOnlyDictionary<CubeCoordinates, HexCoordinates>`

  Provides access to the cells by coordinates.

4.1 Block

- Start `int`

  Index of the first cell.

- Cells `int`

  Cells in on block (row or column).

CHAPTER

FIVE

LAYOUTELEMENTMAX

Allows to control the maximum preferred sizes of the `LayoutElement`.

5.1 Options

- ignoreLayout `bool`

  Should this RectTransform be ignored by the layout system?
- layoutPriority `int`

  The Priority of layout this element has.
- MaxWidth `float`

  Maximum preferred height.
- MaxHeight `float`

  Maximum preferred width.

CHAPTER

SIX

LAYOUT SWITCHER

Allow to create different layouts with same GameObjects for different screen sizes and aspect ratios. Used when anchros, 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).

CHAPTER

SEVEN

CHANGELOG

7.1 Release 1.9.0

- EasyLayout: added warning when one of the objects in the same row has relative width (or one of the objects in the same column has relative height)
- EasyLayout: added IgnoreLayoutElementSizes option: if enabled ILayoutElement options will be ignored, increases performance without side effects if ChildrenWidth and ChildrenHeight are not controlled
- EasyLayout: added Hex layout

7.2 Release 1.8.3

- added MovementAnimateAll and ResizeAnimateAll option - animate movement/resize for all elements if enabled; otherwise new elements will not be animated

7.3 Release 1.8.2

- EasyLayout: fixed movement and resize animations

7.4 Release 1.8.1

- EasyLayout: small improvements

7.5 Release 1.8.0

- EasyLayout: small improvements and bugfixes

7.6 Release 1.7.0

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

7.7 Release 1.6.0

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

7.8 Release 1.5.0

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

7.9 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

7.10 Release 1.3.1

- EasyLayout: added ElementsRotate and ElementsRotationStart for Ellipse layout

7.11 Release 1.3.0

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

7.12 Release 1.2.1

- fixed FitContainer option

7.13 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

7.14 Release 1.1.1

- improved compatibility with Unity 2017.x

7.15 Release 1.1.0

- bug fixes
- performance improvements

7.16 Release 1.0.5

- bug fixes.

7.17 Release 1.0.4

- improved performance
- bug fixes.

7.18 Release 1.0.3

- renamed to EasyLayout