# **PHOTO GALLERY**

REFERENCE :

<https://www.freecodecamp.org/news/css-flexbox-complete-guide/#flex-container-vs-flex-item-what-s-the-difference>

* It is used for responsive design :

Within your head element, add a meta tag with the name set to viewport and the content set to width=device-width, initial-scale=1.Also add a meta tag with the charset set to UTF-8.

 <meta name="viewport" content="width=device-width, initial-scale=1.0">

* Normalize your box model by creating a \* selector and setting the box-sizing property to border-box as the value.

\*{

  box-sizing:border-box;

}

* Remove the margin from your body element, set the font-family to sans-serif, and give it a background-color of #f5f6f7 as the value.

body{

  margin:0;

  font-family:sans-serif;

  background-color :#f5f6f7;

}

* Make the text uppercase using the text-transform property with uppercase as the value.
* Flexbox is a one-dimensional CSS layout that can control the way items are spaced out and aligned within a container.

.gallery{

  display:flex;

}

* Flexbox has a main and cross axis. The main axis is defined by the flex-direction property, which has four possible values:
* row (default): horizontal axis with flex items from left to right
* row-reverse: horizontal axis with flex items from right to left
* column: vertical axis with flex items from top to bottom
* column-reverse: vertical axis with flex items from bottom to top

**Note**: The axes and directions will be different depending on the text direction. The values shown are for a left-to-right text direction.

Flexbox has a main and cross axis. The main axis is defined by the flex-direction property, which has four possible values:

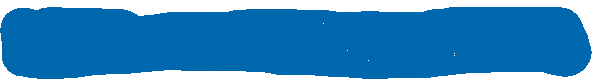


* row (default): horizontal axis with flex items from left to right
* row-reverse: horizontal axis with flex items from right to left
* column: vertical axis with flex items from top to bottom
* column-reverse: vertical axis with flex items from bottom to top

**Note**: The axes and directions will be different depending on the text direction. The values shown are for a left-to-right text direction.

* this is a flex **container**(the blue area) with three flex **items**:

# **1** **2** **3**



* The flex container becomes flexible by setting the display property to flex:
* The flex container properties are:
* [flex-direction](https://www.w3schools.com/css/css3_flexbox_container.asp#flex-direction)
* [flex-wrap](https://www.w3schools.com/css/css3_flexbox_container.asp#flex-wrap)
* [flex-flow](https://www.w3schools.com/css/css3_flexbox_container.asp#flex-flow)
* [justify-content](https://www.w3schools.com/css/css3_flexbox_container.asp#justify-content)
* [align-items](https://www.w3schools.com/css/css3_flexbox_container.asp#align-items)
* [align-content](https://www.w3schools.com/css/css3_flexbox_container.asp#align-content)
* The flex-direction property defines in which direction the container wants to stack the flex items. Column-vertical, column-reverse -bottom to top , row-horizontal , row-reverse .
* The flex-wrap property specifies whether the flex items should wrap or not. It means that when we resize the window , it moves accordingly. The wrap value specifies that the flex items will wrap if necessary , The nowrap value specifies that the flex items will not wrap (this is default), The wrap-reverse value specifies that the flexible items will wrap if necessary, in reverse order
* The flex-flow property is a shorthand property for setting both the flex-direction and flex-wrap properties.

.flex-container {  
  display: flex;  
  flex-flow: row wrap;  
}

* The justify-content property is used to align the flex items, flex-start: at beginning, flex-end :end , The space-around value displays the flex items with space before, between, and after the lines. space-between: between lines .
* The align-items property is used to align the flex items.
* The center value aligns the flex items in the middle of the container:
* The flex-start value aligns the flex items at the top of the container:
* The flex-end value aligns the flex items at the bottom of the container:
* The stretch value stretches the flex items to fill the container (this is default):
* The align-content property is used to align the flex lines.
  + The space-between value displays the flex lines with equal space between them
  + The space-around value displays the flex lines with space before, between, and after them
  + The stretch value stretches the flex lines to take up the remaining space (this is default)
  + The center value displays display the flex lines in the middle of the container
  + The flex-start value displays the flex lines at the start of the container
  + The flex-end value displays the flex lines at the end of the container

PERFECT CENTERING

.flex-container {  
  display: flex;  
  height: 300px;  
**justify-content: center;  
  align-items: center;**}

* the images have different aspect ratios. Rather than setting each aspect ratio individually, you can use the object-fit property to determine how images should behave.
* The gap CSS shorthand property sets the gaps, also knowns as gutters, between rows and columns. The gap property and its row-gap and column-gap sub-properties provide this functionality for flex, grid, and multi-column layout. You apply the property to the container element.
* The ::after pseudo-element creates an element that is the last child of the selected element. You can use it to add an empty element after the last image. If you give it the same width as the images it will push the last image to the left when the gallery is in a two-column layout. Right now, it is in the center because you set justify-content: center on the flex container.

Example:

.container::after {

content: "";

width: 860px;

}