Certainly! Here's the cheat sheet for CSS Grid and Flexbox with the requested formatting for clarity:

**Grid**

Creating a Grid:

selector {

display: grid; /\* or inline-grid \*/

}

Grid Shorthand Properties:

* **grid-template-rows**: none - Configure elements in a row-like layout.
* **grid-template-columns**: none - Configure elements in a column-like layout.
* **grid-template-areas**: none - Configure named grid areas and their layout.
* **grid-auto-rows**: auto - Default size for rows not explicitly set.
* **grid-auto-columns**: auto - Default size for columns not explicitly set.
* **grid-auto-flow**: row - Default placement of rows not explicitly positioned.
* **column-gap**: normal - Gap between columns.
* **row-gap**: normal - Gap between rows.

Grid Properties for Container:

* **grid-template-columns**: measurement units | % units | repeat() - Defines column sizes and line names.
* **grid-template-rows**: measurement units | % units | repeat() - Defines row sizes and line names.
* **grid-auto-columns**: measurement unit - Default size for implicitly created columns.
* **grid-auto-rows**: measurement unit - Default size for implicitly created rows.
* **grid-template**: “header header” auto “main right” 75vh “footer footer” 20rem - Define and maintain named grid cells with specific sizes.

Gap:

* **grid-gap**: measurement units - Gap between rows and columns.
* **grid-column-gap**: measurement units - Gap between columns.
* **grid-row-gap**: m-unit-1 m-unit-2 - Gap between rows.

Alignment:

* **justify-items**: start | center | end | stretch - Default alignment of items along the row.
* **align-items**: start | center | end | stretch - Default alignment of items along the column.
* **place-items**: start | stretch - Shorthand for aligning items both horizontally and vertically.

Justification:

* **justify-content**: start | center | end | stretch | space-between | space-evenly | space-around - Distribution of space among items along the main axis.
* **align-content**: start | center | end | stretch | space-between | space-evenly | space-around - Distribution of space among items along the cross axis.
* **place-content**: center | start - Shorthand for setting both justify-content and align-content.

Positioning:

* **grid-auto-flow**: row | column | dense - Automatic placement of items within the grid.
* **grid-column**: column position - Specify column start and end for an item.
* **grid-row**: row position - Specify row start and end for an item.

Justification and Alignment for Items (Child):

* **justify-self**: start | center | end | stretch - Alignment of an item within its cell along the row.
* **align-self**: start | center | end | stretch - Alignment of an item within its cell along the column.
* **place-self**: start | stretch - Shorthand for both justify-self and align-self.

**Flexbox**

Creating a Flexbox:

cssCopy code

selector { display: flex | inline-flex }

Properties for Flexbox Container:

* **flex-direction**: row | row-reverse | column | column-reverse - Direction of flex items.
* **flex-wrap**: wrap | nowrap - Wrapping of flex items.
* **align-items**: flex-start | flex-end | center | stretch - Vertical alignment of flex items.
* **justify-content**: flex-start | flex-end | center | space-between | space-evenly - Horizontal alignment of flex items.

Properties for Flexbox Items (Child):

* **flex-grow**: factor - Ability of a flex item to grow.
* **flex-shrink**: factor - Ability of a flex item to shrink.
* **flex-basis**: auto | factor | measurement unit - Initial main size of a flex item.
* **order**: position - Order of the flex item within the flex container.
* **align-self**: start | center | end | stretch - Override alignment for individual flex items.

This formatted cheat sheet organizes the key properties and options for both CSS Grid and Flexbox, making it easier to reference and apply in your web development projects.