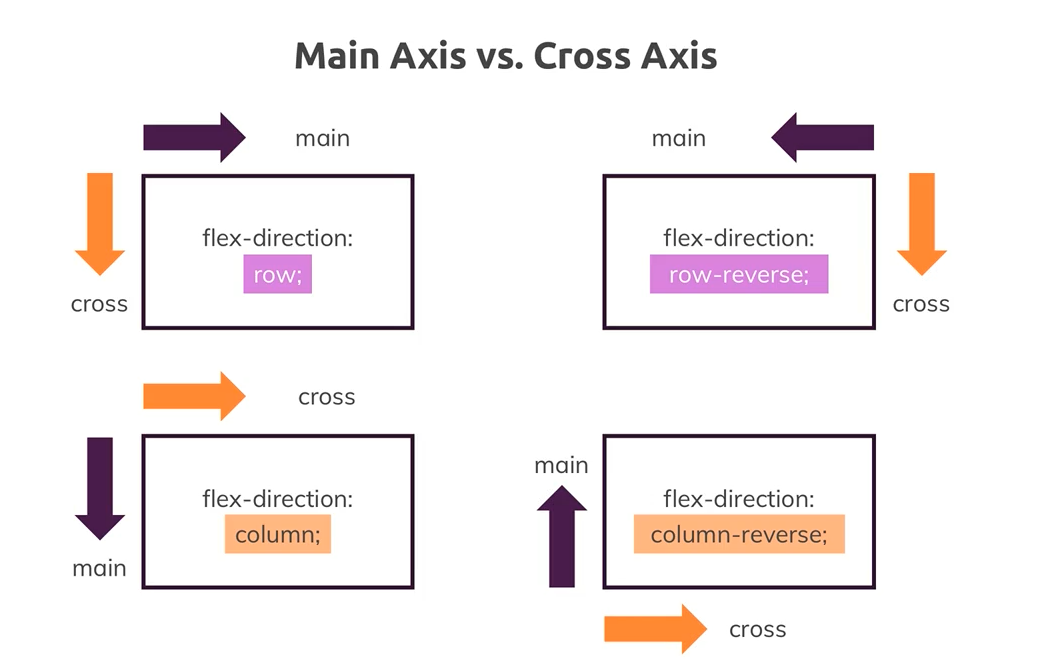
* If we apply display: flex to a parent, it will turn into a flex container.
* This container child elements are flex items.
* By default, flex items use the entire height available from the parent element. The highest value of height from the children will be the default value in case height isn’t specified.
* We can also apply display: inline-flex; Here we can’t change the size of the elements as it is pre-defined. Parent element use only the size it requires to display the content. Basically, it behaves like an inline-element.

We can use different property for parent and child

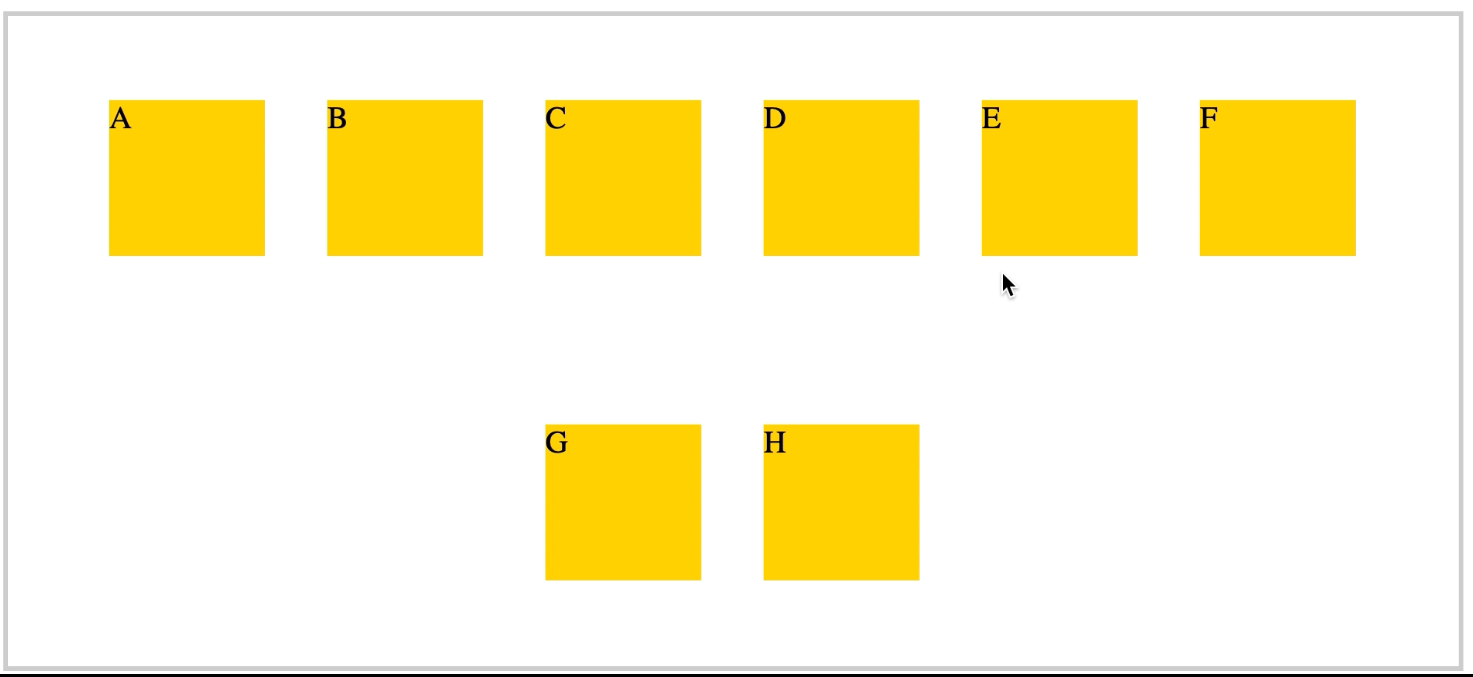
1. **Flex containers:** 
   1. **Flex-wrap:** The flex-wrap property specifies whether the flexible items should wrap or not. If the elements are not flexible items, the flex-wrap property has no effect.
      1. **Nowrap:** default
      2. **Wrap:** item will be wrapped from first element
      3. **Wrapreverse:** Item will be wrapped from last element.
   2. **Flex-direction:** sets the direction
      1. **Row:** default value, items displayed horizontally
      2. **Column:** items displayed vertically.
      3. **Reverse-row:** same as row, in reverse
      4. **Reverse-column:** same as column, in reverse
   3. **Flex-flow:** Shorthand for flex-direction and flex-wrap;
   4. **Justify-content:** Used to align items on main axis(horizontally). W3school for ref.
   5. **Align-content:** this aligns items on cross axis (vertically), but only works on multi line flex childs. W3school for ref.
   6. **Align-items:** The align-items property specifies the default alignment for items inside the flexible container.
      1. **Stretch:** default, items are stretched to fit container.
      2. **Center:** items are centered.
      3. **Flex-start:** items are positioned from the beginning of the container.
      4. **Flex-end:** items are positioned from the end of the container.
      5. **Baseline:** items are positioned from the baseline of the container.
2. **Flex items:**
   1. **Order:** specify the order of the flex item. Default value is 0. We can set the order.
   2. **Flex-grow:** It specify how much item will grow relative to the rest of the items.
   3. **Flex-shrink:** It specify how much item will shrink relative to the rest of the items
   4. **Flex-basis:** The flex-basis property specifies the initial length of a flexible item**.**
   5. **Align-self:** This overrides the align-item from container. W3school for ref.

****

**Flexbox:**  This coverts children from row to column. We can specify justify content to fix position vertically. Width is important here, make sure to leave some % of width for the justify.

**Align item vs Align content:** we can use **Align-item** and **justify-content** is for aligning items via the cross axis and main axis respectively. Now **align-content** on the other hand needs multi line elements to work, it works on the whole content as a whole rather than individual elements.

This is with justify-content and align items as center. But if we notice the content is too spread out.



Now if we use align-content, we can see the content as a whole is aligned beautifully.

