WEB FUNDAMENTALS WDD 130

CSS Grid Alignment Activity

In this activity, we will learn about using CSS Grid alignment.

Activity Instructions

01 Grid Alignment Concepts

You may want to review the following short videos before completing the next section.

1. Align grid items (3:32 min)
"Grid by Example: Aligning and Justifying the grid" Transcript

2. Align the Grid (4min)

Vocabulary time! There are a few terms we should learn to help make sense of alignment with CSS Grid.

`justify-content`
: Aligns the grid container along the inline axis (by default horizontal). This only works when the grid is narrower than the space available.

`justify-items`
: Aligns the grid item contents along the inline axis (by default horizontal). This only works when the contents of a grid cell are narrower than the grid cell.

`align-content`
: Aligns the grid container along the block axis (by default vertical). This only works when the grid is shorter than the space available.

`align-items`
: Aligns the grid item contents along the block axis (by default vertical). This only works when the contents of a grid cell are shorter than the grid cell.

So to review, `justify` aligns horizontal by default, `align` takes care of the other direction (vertical alignment by default). If we want to align the grid container, we use the `content` suffix. If we want to align the contents of the grid items within the grid cells, then we use `items`.

02 Grid Items Aligned

Visit this Codepen (Codepen is a site that allows you to practice writing HTML and CSS). You should see three sets of four boxes. The boxes have already been positioned into a square using CSS Grid. However, there are some differences between them. If you look at the `.content1` class, you will see that the rows and columns have been sized using `1fr`. So the grid will use up all the space available in the grid container, and we will end up with two equal columns and two equal rows.

If we look in the browser development tools at the actual grid, we see this:

Grid lines for content1

Let's try centering the grid and grid items. Add the following to `.content1`:

```
justify-content: center;
align-content: center;
```

You might be surprised to see **nothing** happened until you remember that `content` refers to aligning the grid container. The grid container is completely filling the parent element it is in. If something is as big as its parent, then it is technically already centered...it *can't* move.

Next add these lines one at a time:

```
justify-items: center;
```

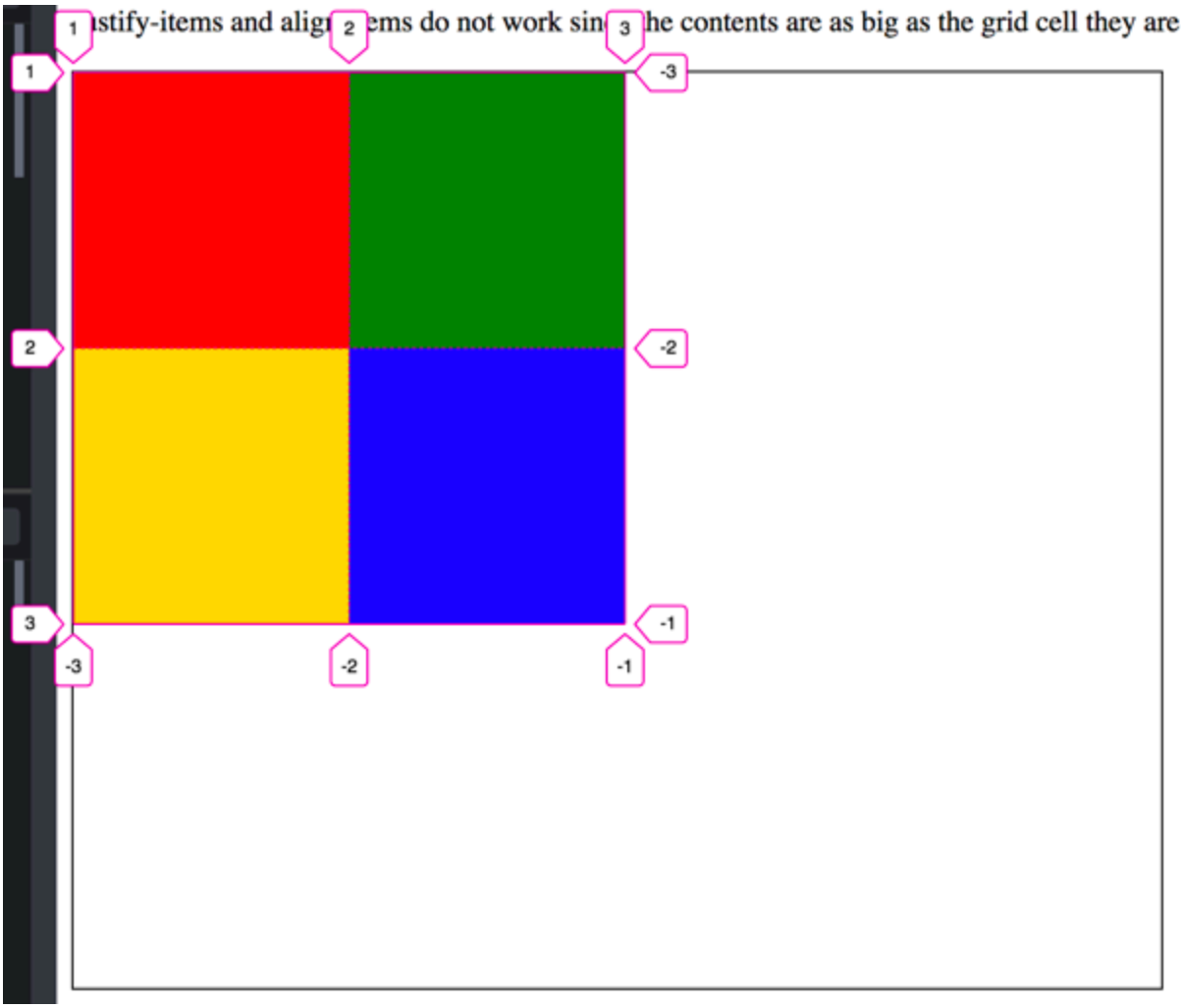
```
align-items: center;
```

Now we see something happen! The boxes centered themselves inside of the grid cell they were in. First, horizontally (`justify`), then, vertically (`align`).

Take a screenshot of the grid items centered.

03 The Grid Aligned

The next example uses rows and columns of a fixed size: `150px` to be exact. This time, the contents of the grid is the same size as the grid cell, but the grid itself is smaller than the space it has available. Here is a picture of the grid lines this time.



Grid lines for Content2

This time if we attempt to align the items, nothing happens. Try adding the following properties to `.content2`:

```
justify-items: center;
align-items: center;
```

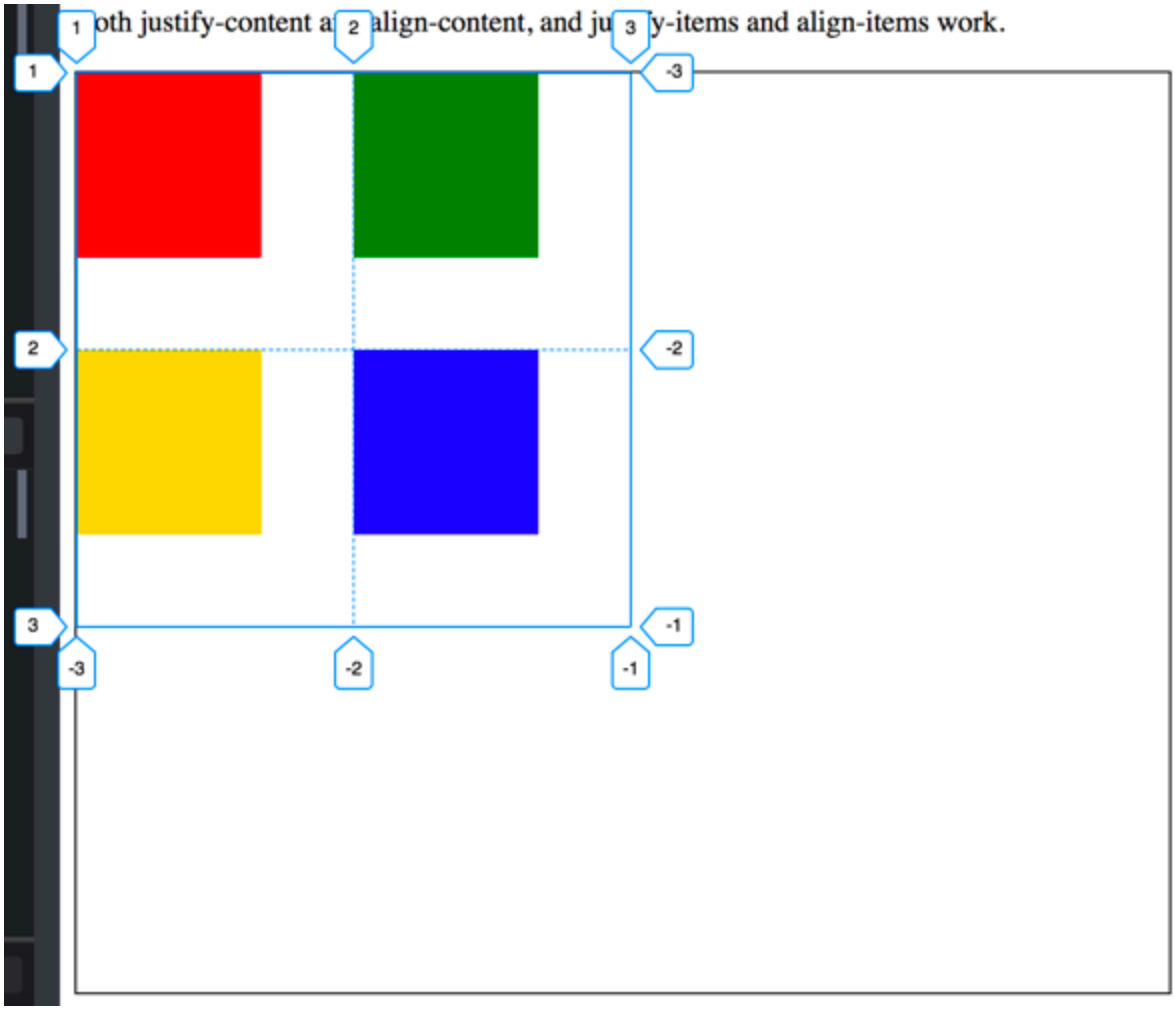
but now we can align the grid container. Add these one at a time:

```
justify-content: center;
align-content: center;
```

Take a screenshot of the grid centered.

04 The Grid and Grid Items Aligned

For the last example using `.content3`, we have the grid container smaller than the parent, and the grid items smaller than the grid cell.



Grid lines for Content3

Add the following four lines to the `.content3` rule one at a time and watch what happens.

```
justify-content: center;
align-content: center;
justify-items: center;
align-items: center;
```

Perfectly centered content!

Take a screenshot of the centered grid and centered grid items.