(ChatGPT):

Flexbox یک مدل چیدمان در CSS است که به شما کمک می‌کند تا آیتم‌ها را در یک خط یا ستون به صورت انعطاف‌پذیر مرتب کنید. با Flexbox، دیگر نیازی به استفاده از روش‌های پیچیده برای چیدمان آیتم‌ها مثل float، inline-block و table نیست.

چند مفهوم اصلی Flexbox:

1. Container (ظرف): عنصری که Flexbox روی آن اعمال می‌شود. این عنصر شامل آیتم‌های فرزند یا همان items است.
   - برای تعریف یک container به شکل Flexbox از خاصیت display: flex; استفاده می‌کنیم.
2. Items آیتم‌ها: عناصری که در داخل container قرار می‌گیرند و به کمک Flexbox مرتب می‌شوند.
3. خاصیت‌های مهم:
   - flex-direction: جهت چیدمان آیتم‌ها را مشخص می‌کند (افقی یا عمودی).
     - row(پیش‌فرض): آیتم‌ها به صورت افقی چیده می‌شوند.
     - column: آیتم‌ها به صورت عمودی چیده می‌شوند.
   - justify-content: چیدمان آیتم‌ها در محور اصلی (افقی یا عمودی) را تنظیم می‌کند.
     - flex-start: آیتم‌ها از ابتدا شروع می‌شوند.
     - center: آیتم‌ها در مرکز قرار می‌گیرند.
     - space-between: فاصله مساوی بین آیتم‌ها ایجاد می‌کند.
   - align-items: تراز عمودی آیتم‌ها در محور متقاطع را تنظیم می‌کند.
     - stretch(پیش‌فرض): آیتم‌ها به طور کامل در محور متقاطع کشیده می‌شوند.
     - center: آیتم‌ها در محور متقاطع به صورت وسطچین قرار می‌گیرند.
4. flex-grow: تعیین می‌کند که هر آیتم چه مقدار فضای اضافی بگیرد.
   - مثلاً اگر یک آیتم flex-grow: 1; باشد و دیگری flex-grow: 2;، آیتم دوم دو برابر آیتم اول فضا می‌گیرد.
5. flex-wrap: تعیین می‌کند که اگر فضا کم باشد، آیتم‌ها در چند خط قرار بگیرند یا در یک خط بمانند.

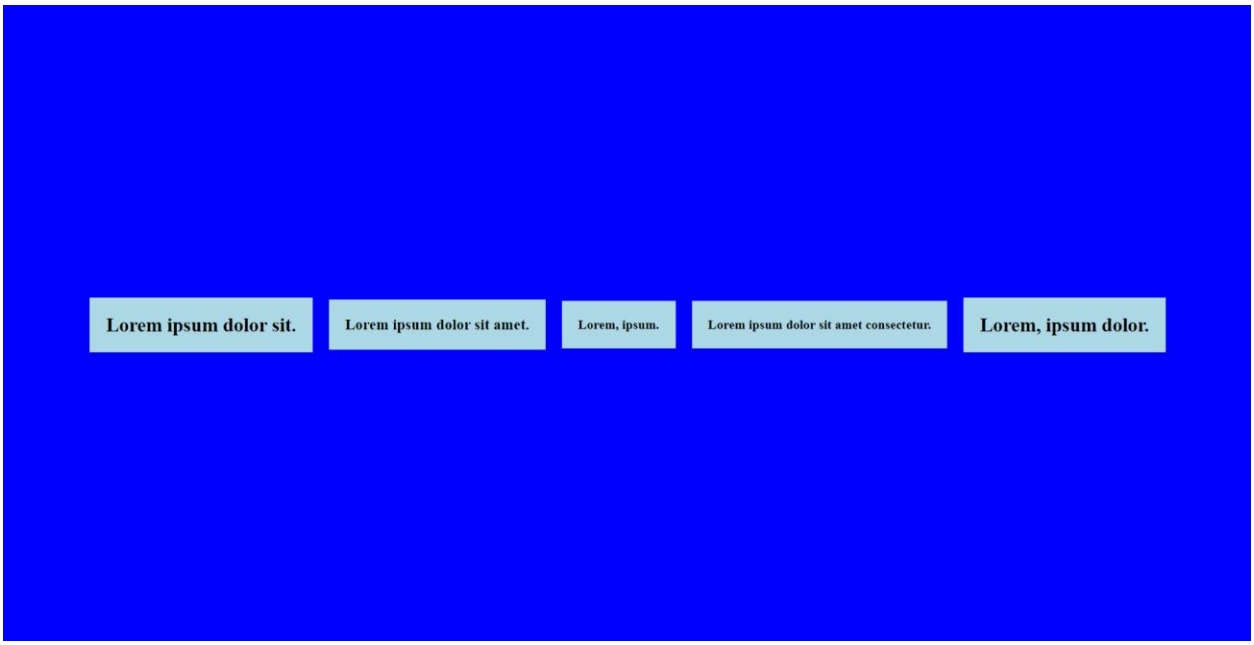
این یک مثال ساده از Flexbox است:

در این مثال، **آیتم‌ها به صورت افقی در مرکز container و محور عمودی تراز شده‌اند.**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        .container{
            height: 100vh;
            background-color: blue;
```

```
            display: flex;
            flex-direction: row;
            justify-content: center;
            align-items: center;
            >*{
                background-color: lightblue;
                padding: 20px;
                margin: 10px;
                color: black;
            }
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Lorem ipsum dolor sit.</h2>
        <h3>Lorem ipsum dolor sit amet.</h3>
        <h4>Lorem, ipsum.</h4>
        <h4>Lorem ipsum dolor sit amet consectetur.</h4>
        <h2>Lorem, ipsum dolor.</h2>
    </div>
</body>
</html>
```

خروجی:



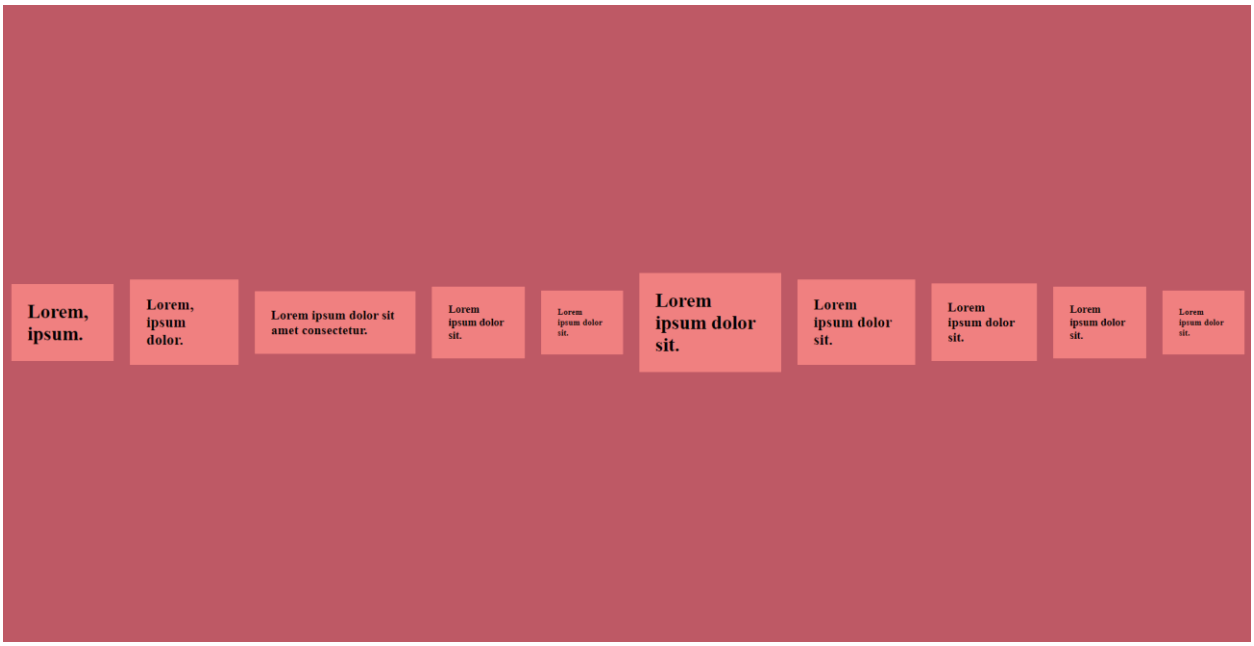
مثال: چیدمان آیتم‌ها با فاصله مساوی در محور افقی

در این مثال، آیتم‌ها به صورت افقی چیده می‌شوند و بین آنها فاصله مساوی وجود دارد:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>second example of flex usage</title>
    <style>
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        .container{
            height: 100vh;
            background-color: rgb(190, 89, 101);
            display: flex;
            flex-direction: row;
            justify-content: space-between; /* فاصله مساوی بین آیتم ها */
            align-items: center; /* وسطچین عمودی */
            >*{
                background-color: lightcoral;
                padding: 20px;
                margin: 10px;
            }
        }

    </style>
</head>
<body>
    <div class="container">
        <h2>Lorem, ipsum.</h2>
        <h3>Lorem, ipsum dolor.</h3>
        <h4>Lorem ipsum dolor sit amet consectetur.</h4>
        <h5>Lorem ipsum dolor sit.</h5>
        <h6>Lorem ipsum dolor sit.</h6>
        <h2>Lorem ipsum dolor sit.</h2>
        <h3>Lorem ipsum dolor sit.</h3>
        <h4>Lorem ipsum dolor sit.</h4>
        <h5>Lorem ipsum dolor sit.</h5>
        <h6>Lorem ipsum dolor sit.</h6>
    </div>
</body>
```

```
</html>
```

خروجی:



در اینجا:

- خاصیت justify-content: space-between; باعث می‌شود آیتم‌ها با فاصله مساوی از هم چیده شوند.
- آیتم‌ها به صورت افقی (چون flex-direction: row;) و به مرکز عمودی تراز می‌شوند.

مثال: چیدمان عمودی آیتم‌ها

در این مثال، آیتم‌ها به صورت عمودی چیده می‌شوند:

اینجا از `flex-direction: column;` برای چیدمان آیتم‌ها به صورت عمودی استفاده شده است. همچنین، آیتم‌ها هم در محور افقی و هم در محور عمودی وسطچین شده‌اند.

مثال: چیدمان با کشش فضای اضافی

در این مثال، هر آیتم بسته به مقدار flex-grow فضایی از container را می‌گیرد:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>fourth example of flexbox</title>
    <style>
```

```
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        .container{
            height: 100vh;
            background-color: chartreuse;
            display: flex;
            flex-direction: row;
            justify-content: space-around;
            align-items: center;
                >h2{
                    background-color: lightskyblue;
                    padding: 20px;
                    flex-grow: 1; /*آیتم اول یک واحد فضا می گیرد  */
                }
                >h3{
                    background-color: lightcoral;
                    padding: 20px;
                    flex-grow: 2; /*  آیتم دوم 2 واحد فضا می‌گیرد  */
                }
                >h4{
                    background-color: lightgreen;
                    padding: 20px;
                    flex-grow: 3; /*  آیتم سوم 3 واحد فضا می‌گیرد  */
                }
            }

        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Lorem, ipsum.</h2>
        <h3>Lorem, ipsum dolor.</h3>
        <h4>Lorem ipsum dolor sit amet consectetur.</h4>

    </div>
</body>
</html>
```

خروجی:

در این مثال:

- خاصیت flex-grow برای هر آیتم تعیین می‌کند که چه مقدار فضا از container را بگیرد.
- آیتم اول یک واحد فضا، آیتم دوم دو واحد و آیتم سوم سه واحد از فضا را می‌گیرند.

نکته در مورد property زیر:

```
justify-content: space-around;
```

استفاده از justify-content: space-around; به این معناست که فضای اضافی در اطراف آیتم‌ها به صورت مساوی تقسیم می‌شود.

در واقع:

- ✓ **فضا قبل و بعد از آیتم‌ها**: با استفاده از space-around، هر آیتم مقداری فضا (margin) در سمت چپ و راست خود خواهد داشت. این فضا در مجموع برابر با نیمی از فاصله بین دو آیتم مجاور است.
- ✓ **توزیع مساوی**: این ویژگی باعث می‌شود که فاصله بین هر دو آیتم مساوی باشد و همچنین فضا در ابتدا و انتهای container نیز تقسیم شود.

تصویر ذهنی

فرض کنید که سه آیتم داریم:

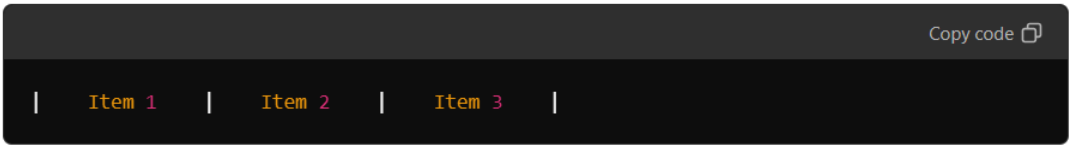
با space-around، فضا به این شکل تقسیم می‌شود:

```
|---   Item 1   ---|---   Item 2   ---|---   Item 3   ---|
```

هر --- نمایانگر فضا (margin) است که قبل و بعد از هر آیتم وجود دارد.

مقایسه با دیگر مقادیر

- space-between: اگر از space-between استفاده کنید، فضا فقط بین آیتم‌ها تقسیم می‌شود و در ابتدا و انتهای container فضایی نخواهید داشت:



- center: در حالت center، همه آیتم‌ها در مرکز container قرار می‌گیرند بدون هیچ فاصله‌ای در ابتدا و انتهای آن:

نکته در مورد property زیر:

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

خاصیت align-items: center; در Flexbox برای تراز کردن آیتم‌ها در **محور عمودی** (محور متقاطع یا **Cross Axis**) استفاده می‌شود.

دلیل عملکرد align-items: center;

درFlexbox، هر container دو محور دارد:

1: محور اصلی (Main Axis): محور اصلی جهت چیدمان آیتم‌ها را مشخص می‌کند.

- اگر flex-direction: row; باشد، محور اصلی افقی است.
- اگر flex-direction: column; باشد، محور اصلی عمودی است.

2: محور متقاطع (Cross Axis): محور عمودی به محور اصلی است.

- برای flex-direction: row;، محور متقاطع عمودی است.
- برای flex-direction: column;، محور متقاطع افقی است.

align-items در محور متقاطع

✓ align-items: center; آیتم‌ها را در محور متقاطع به مرکز می‌برد.

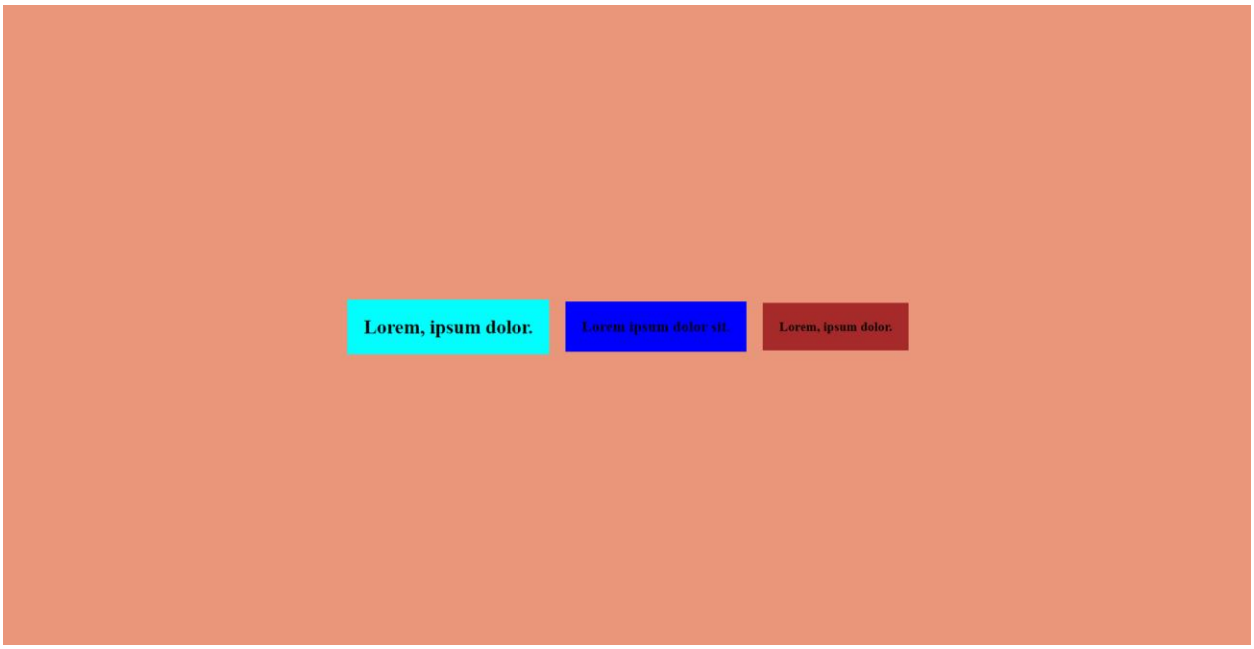
✓ در حالت معمول (وقتی flex-direction: row; است)، محور متقاطع عمودی خواهد بود، بنابراین با استفاده از align-items: center; ، آیتم‌ها به وسط محور عمودی منتقل می‌شوند.

مثال:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>fifth example of flexbox</title>
    <style>
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        .container{
            background-color: darksalmon;
            height: 100vh; /* ارتفاع container */
            display: flex;
            flex-direction: row; /* محور اصلی افقی است */
            align-items: center; /* آیتم‌ها در محور عمودی وسط‌چین می‌شوند */
            justify-content: center;
            >h2{
                background-color: aqua;
                margin: 10px;
                padding: 20px;
            }
            >h3{
                background-color: blue;
                margin: 10px;
                padding: 20px;
            }
            >h4{
                background-color: brown;
                margin: 10px;
                padding: 20px;
            }
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Lorem, ipsum dolor.</h2>
```

```
        <h3>Lorem ipsum dolor sit.</h3>
        <h4>Lorem, ipsum dolor.</h4>
    </div>
</body>
</html>
```

خروجی:



- ✓ در این مثال، چون flex-direction: row; است:
- ✓ محور اصلی افقی است (چیدمان آیتم‌ها به صورت افقی است).
- ✓ محور متقاطع عمودی است، و با align-items: center; آیتم‌ها به وسط ارتفاع container تراز می‌شوند.

اگر flex-direction: column; باشد:

اگر flex-direction: column; را تنظیم کنیم، محور اصلی عمودی و محور متقاطع افقی می‌شود.

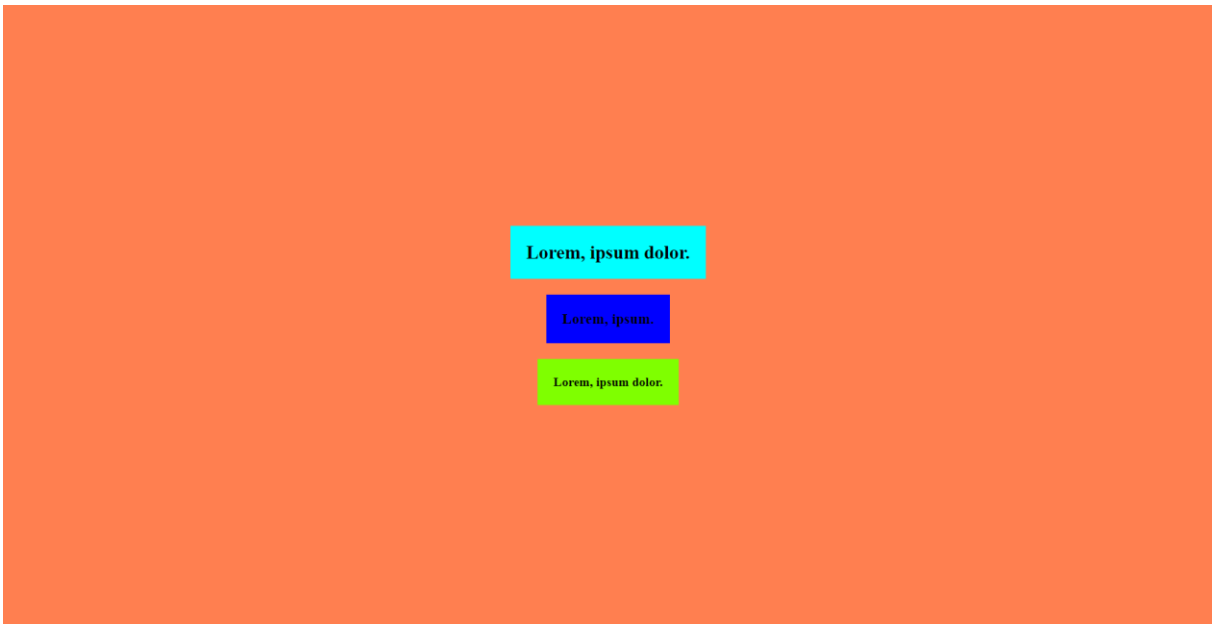
در این حالت align-items: center; آیتم‌ها را به مرکز محور افقی می‌برد.

مثال:

```
<!DOCTYPE html>

<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>sixth example of flexbox</title>
    <style>
```

```
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        .container{
            height: 100vh;
            background-color: coral;
            display: flex;
            flex-direction: column;
            align-items: center;
            justify-content: center;
            >*{
                margin: 10px;
                padding: 20px;
            }
            >h2{
                background-color: aqua;
            }
            >h3{
                background-color: blue;
            }
            >h4{
                background-color: chartreuse;
            }
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Lorem, ipsum dolor.</h2>
        <h3>Lorem, ipsum.</h3>
        <h4>Lorem, ipsum dolor.</h4>
    </div>
</body>
</html>
```

خروجی:



در اینجا align-items: center; باعث می‌شود آیتم‌ها در عرض container (محور افقی) وسطچین شوند.

نتیجه‌گیری:

align-items: center; آیتم‌ها را در محور متقاطع (که معمولاً محور عمودی است) به مرکز می‌برد. اگر محور متقاطع عمودی باشد، آیتم‌ها در وسط ارتفاع container قرار می‌گیرند.

بنابراین در دنیای CSS به مفهوم flex رسیدیم.

flex اصلی ترین روش layout design هست.

درواقع، تعیین جایگاه عناصر وبسایت با flex انجام می شود.

برای درک مفهوم flex می بایست درک از container و children داشته باشیم. (wrapper(parent) و children)

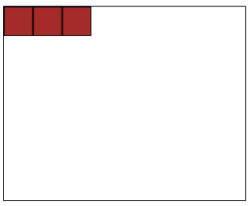
بعضی از flex properties برای container و برخی برای children تعریف می شوند.

position برای layout اصلا مناسب نیست.

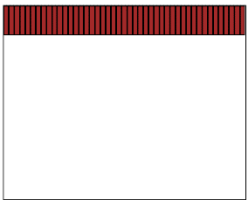
```
display: flex;
```

flex یک value در CSS3 هست.

با flex (بدون نیاز به float) به راحتی می توانیم خروجی زیر را داشته باشیم:

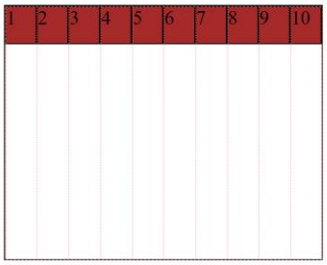


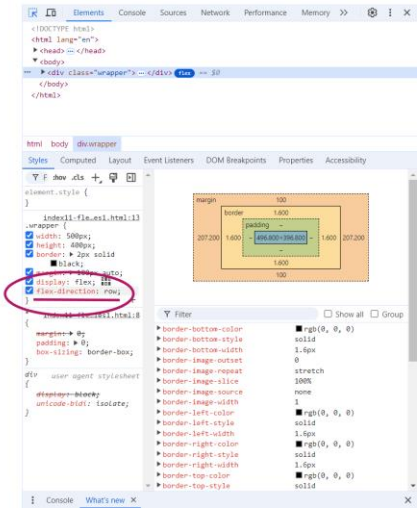
(تعریف ساده flex: هر چی children داشته باشی در یک سطر کنار هم جا میدم)



نکته:

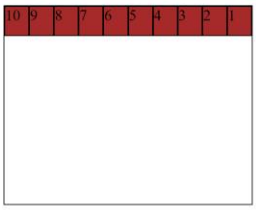
```
flex-direction: row;
```

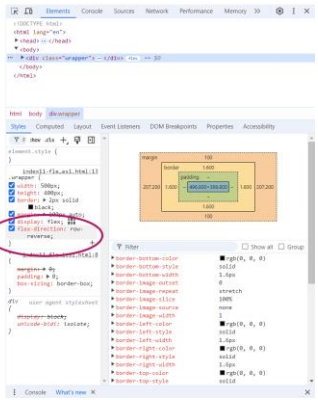




reverse *verb* (CHANGE TO OPPOSITE)

```
flex-direction: row-reverse;
```

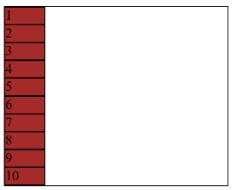




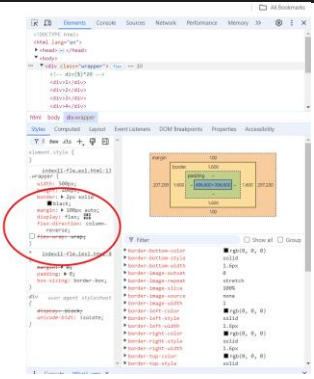
اگر direction، rtl باشد، row یعنی راست چین

اگر direction، ltr باشد، row یعنی چپ چین

```
flex-direction: column;
```

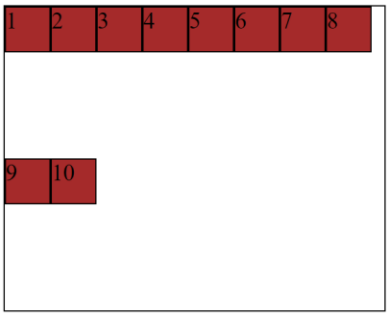



```
flex-direction: column-reverse;
```



نکته:

```
flex-wrap: wrap;
```



flex-wrap: wrap; یکی از ویژگی‌های CSS Flexbox است که تعیین می‌کند اگر آیتم‌های فرزند (items) درون یک container (ظرف) جا نشوند، آن‌ها چگونه رفتار کنند. به طور پیش‌فرض، تمام آیتم‌ها در یک ردیف (یا ستون) قرار می‌گیرند و از اندازه ظرف خارج نمی‌شوند، اما با استفاده از flex-wrap می‌توانیم آیتم‌ها را به خطوط جدید منتقل کنیم.

مقدارهای اصلی flex-wrap:

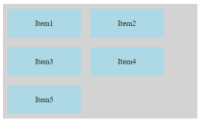
1. **nowrap پیش‌فرض :(**آیتم‌ها در یک خط باقی می‌مانند، حتی اگر اندازه container برای جا دادن همه آیتم‌ها کافی نباشد. این ممکن است باعث فشرده شدن یا خارج شدن آیتم‌ها از container شود.
2. **wrap** آیتم‌ها به خطوط جدید می‌روند (در محور اصلی)، اگر فضای کافی در خط فعلی نباشد. به این معنی که اگر آیتم‌ها در یک ردیف جا نشوند، به ردیف بعدی منتقل می‌شوند.
3. **wrap-reverse** مشابه wrap است، اما آیتم‌ها به‌جای اینکه در جهت طبیعی خطوط جدید (از بالا به پایین یا چپ به راست) قرار بگیرند، از جهت معکوس به خطوط جدید منتقل می‌شوند (از پایین به بالا یا راست به چپ).

مثال:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>flex container properties 2</title>
```

```
    <style>
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        .container{
            width: 400px;
            background-color: lightgray;
            display: flex;
            flex-direction: row;
            flex-wrap: wrap;
            >div{
                background-color: lightblue;
                width: 150px;
                text-align: center;
                margin: 10px;
                padding: 20px;
            }
        }
    </style>
</head>
<body>
    <div class="container">
        <!-- div{Item$}*5 -->
        <div>Item1</div>
        <div>Item2</div>
        <div>Item3</div>
        <div>Item4</div>
        <div>Item5</div>
    </div>
</body>
</html>
```

خروجی:

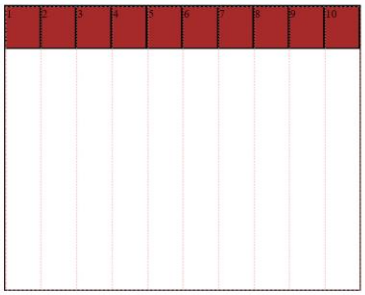


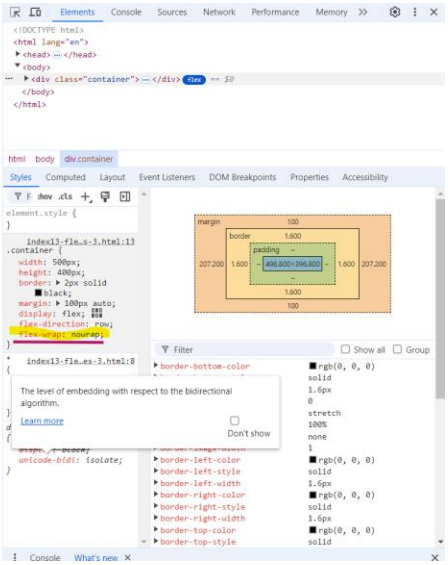
توضیح:

در این مثال، ظرف container. دارای عرض 400px است، و هر آیتم عرض 150px به اضافه مارجین‌ها دارد. به دلیل استفاده از flex-wrap: wrap;، اگر آیتم‌ها در یک خط جا نشوند (در اینجا به دلیل کمبود فضا در خط اول)، به خط بعدی منتقل می‌شوند.

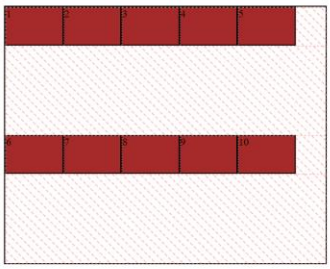
این ویژگی برای چیدمان‌های رسپانسیو و زمانی که عناصر در سایزهای مختلفی ظاهر می‌شوند، بسیار مفید است

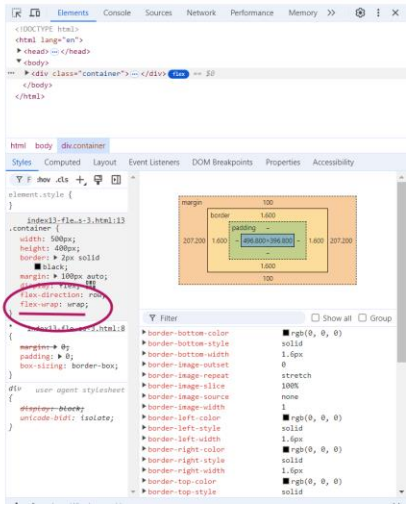
مقدار دیفالت flex-wrap، nowrap هست.





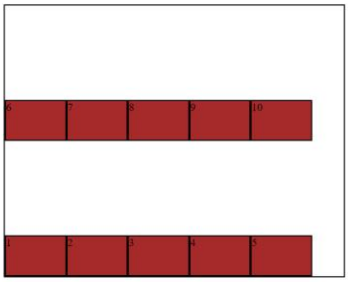
اگر flex-wrap دارای value به نام wrap باشد:

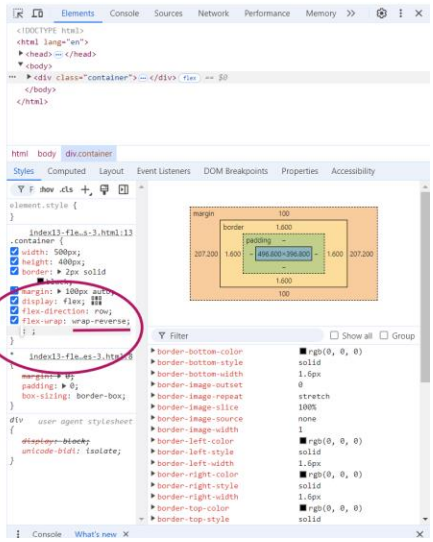




reverse *verb* (CHANGE TO OPPOSITE)

```
flex-wrap: wrap-reverse;
```

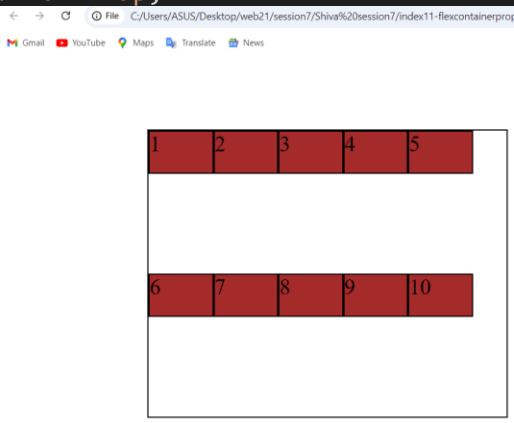


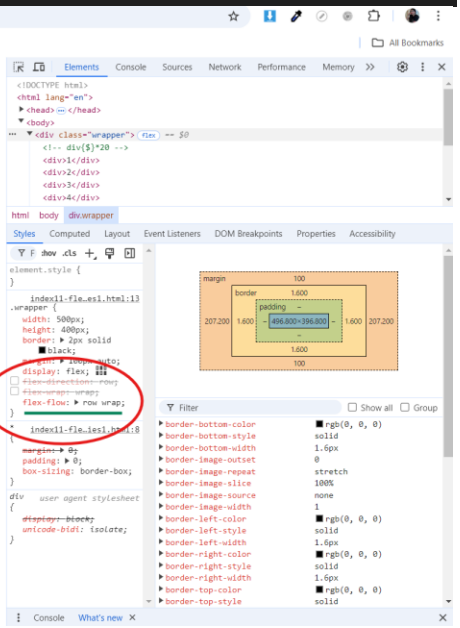


معرفی property به نام flex-flow که flex-direction و flex-wrap را در خود خلاصه می کند:

مثال:

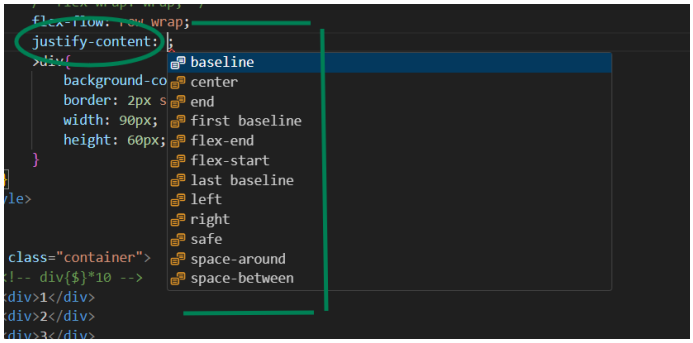
```
flex-flow: row wrap;
```



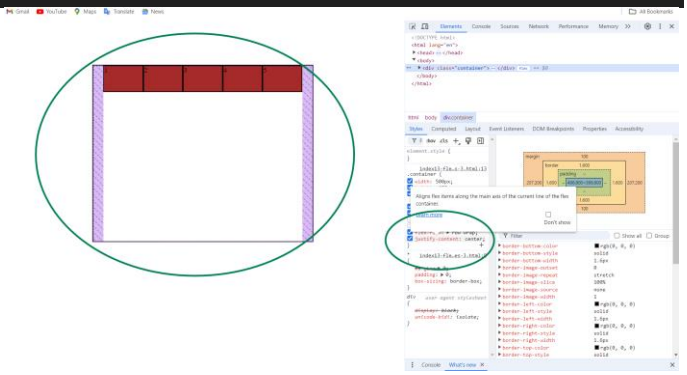


معرفی properties برای ترازبندی به صورت دلخواه در دو محور

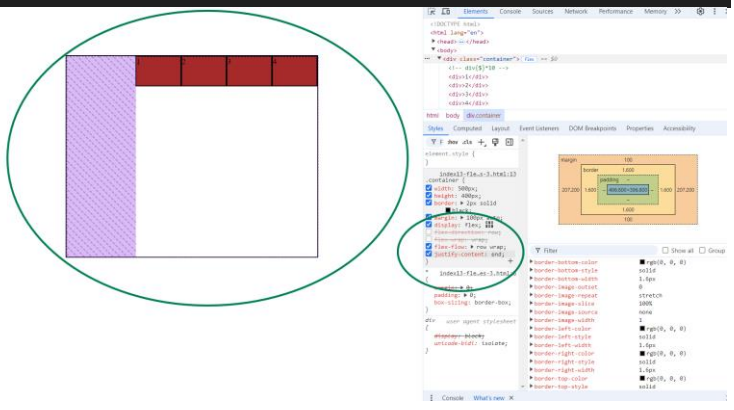
ترازبندی در محور x




```
flex-flow: row wrap;
justify-content: center;
```

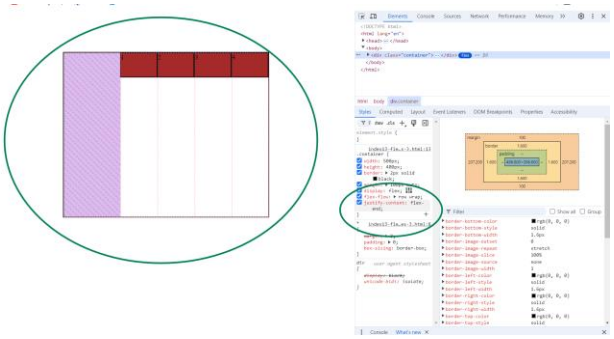


```
display: flex;
flex-flow: row wrap;
justify-content: end;
```

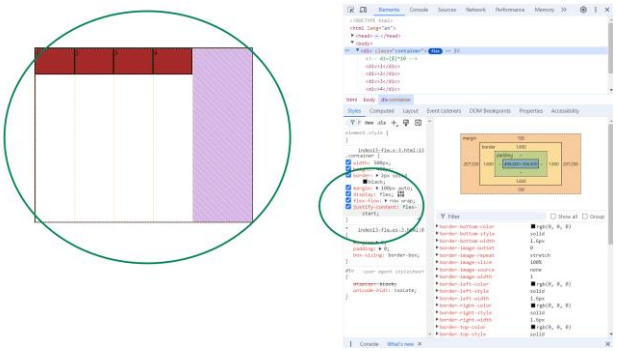


بهتر است به جای end داشته باشیم: flex-end. چرا که end به تنهایی direction را در نظر نمی گیرد و فقط به این معنی است که برو به انتها. ولی flex-end یعنی برحسب direction برو به انتها.

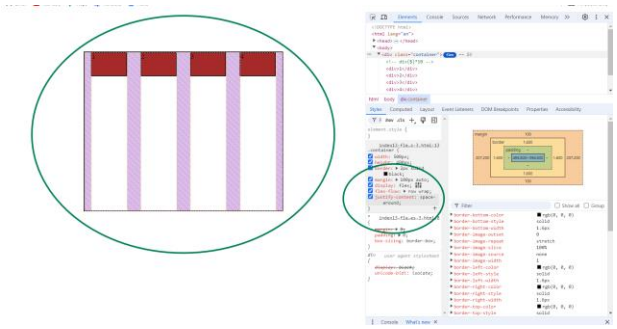
```
display: flex;
flex-flow: row wrap;
justify-content: flex-end;
```



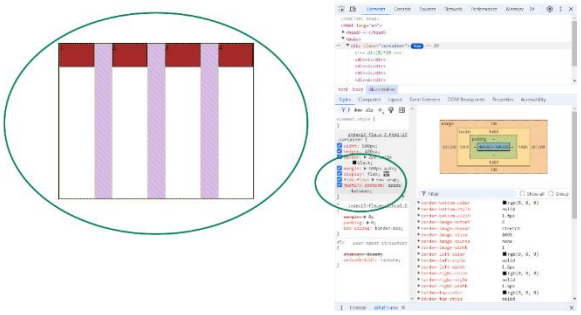
```
display: flex;
flex-flow: row wrap;
justify-content: flex-start;
```



```
display: flex;
flex-flow: row wrap;
justify-content: space-around;
```

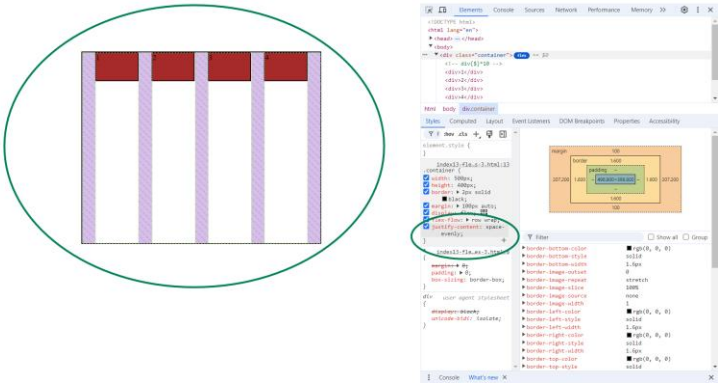


```
            display: flex;
            flex-flow: row wrap;
            justify-content: space-
between;
```

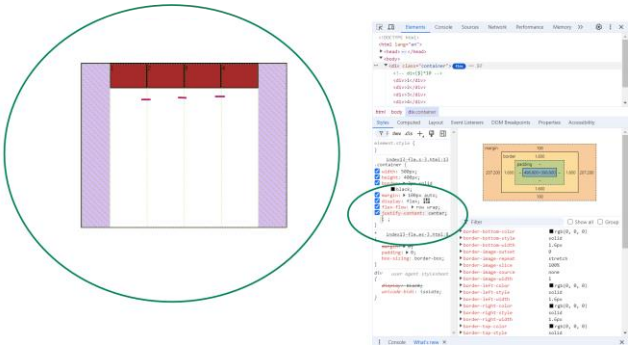


evenly= in or into equal amounts:

```
            display: flex;
            flex-flow: row wrap;
            justify-content: space-evenly;
```

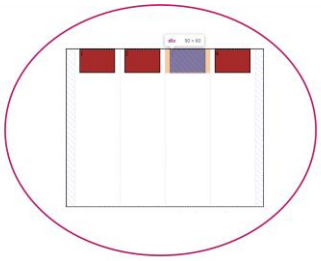


فرض کنید در مثال زیر می خواهیم بین هر یک از المان ها 5 پیکسل فاصله بیفتد: (با مقادیر justify-content این رو به دست نیاوردیم. چه کاری انجام دهیم؟)



پاسخ: استفاده از margin

```
        .container{
            width: 500px;
            height: 400px;
            border: 2px solid black;
            margin: 100px auto;
            display: flex;
            flex-flow: row wrap;
            justify-content: center;
            >div{
                background-color: brown;
                border: 2px solid black;
                width: 90px;
                height: 60px;
                margin: 0 12px;
            }
        }
```

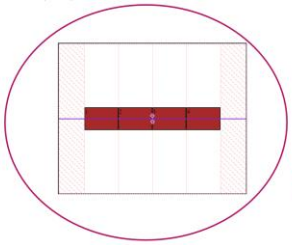


بحث justify-content تمام شد.

properties برای alignment در محور Y (جایگاه بدیم در محور Y)

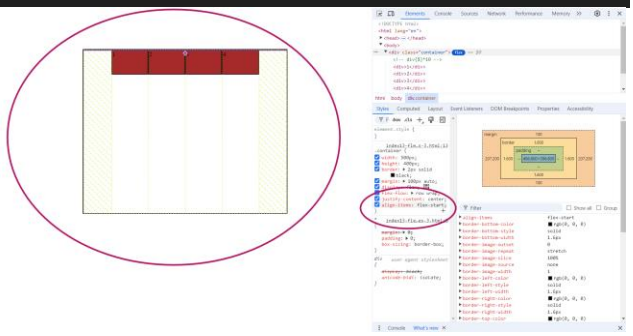
- align-items: ;

```
            flex-flow: row wrap;
            justify-content: center;
            align-items: center;
```



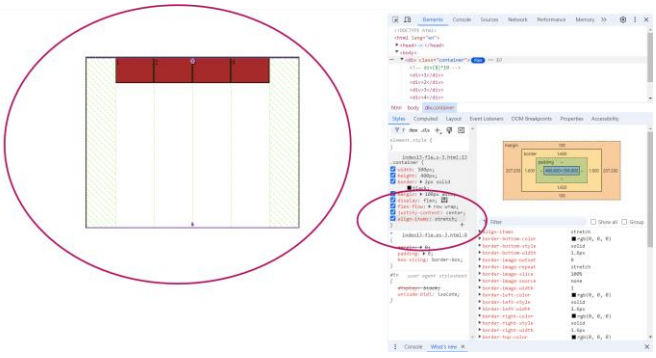
```
display: flex;
flex-flow: row wrap;
justify-content: center;
align-items: flex-end;
```

```
display: flex;
flex-flow: row wrap;
justify-content: center;
align-items: flex-start;
```



stretch= to cause something to reach, often as far as possible, in a particular direction

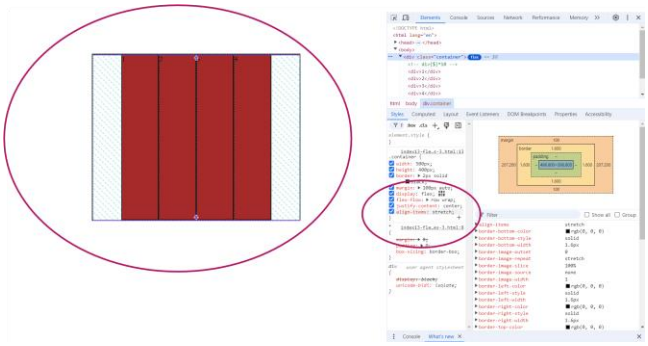
```
display: flex;
flex-flow: row wrap;
justify-content: center;
align-items: stretch;
```



align-items: stretch; بسیار کاربردی است.

align-items به صورت دیفالت stretch است. (فقط children نباید height تعیین شده داشته باشند)

stretch، element height را به اندازه parent height در می آورد. (مگر اینکه به طور دستی قبلا height داده باشیم)



یکی از کاربردها در: sticky position

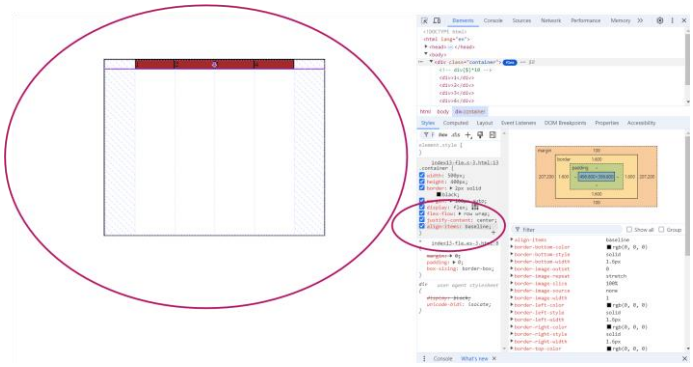
زیر side bar خالی بود. ارتفاع content با side bar چه جوری یکسان شود؟

به parent می بایست flex display بدهیم. align-items هم که به صورت دیفالت مقدار stretch را دارد.

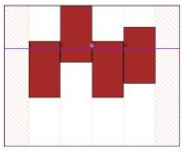
side bar با توجه به ارتفاع content همان height را می گیرد.

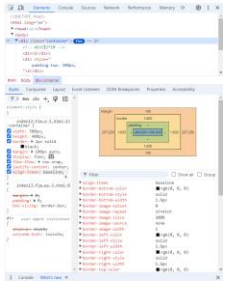
بنابراین چاله آسانسور که می خواستیم ایجاد می شود.

```
display: flex;
flex-flow: row wrap;
justify-content: center;
align-items: baseline;
```



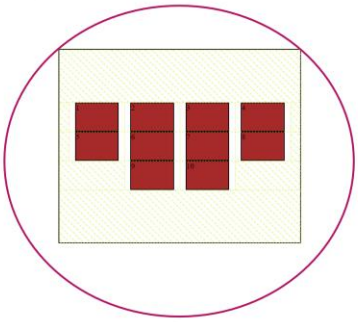
و یا در مثالی دیگر baseline به صورت زیر عمل می کند:

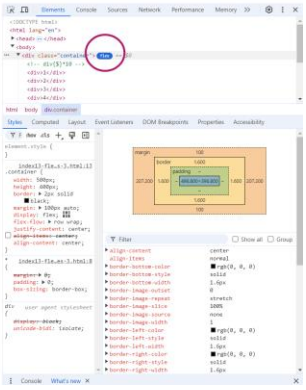




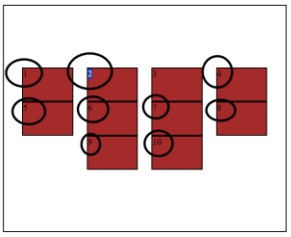
نکته: یک المان به طور همزمان هم می تواند flex children و هم می تواند flex parent باشد.

در مثال زیر المان ها حکم children برای flex را دارند: (در دو محور X و Y کنترل شده اند)





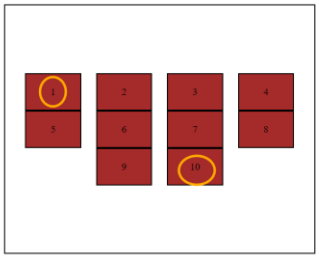
divisions' children هم با flex در X و Y کنترل می شوند:



بنابراین باید به divها flex display و justify-content و align-content داد:

```
.container{
    width: 500px;
    height: 400px;
    border: 2px solid black;
    margin: 100px auto;
    display: flex;
    flex-flow: row wrap;
    justify-content: center;
```

```
        /* align-items: center; */
        align-content: center;
        place-content: ;
        >div{
            background-color: brown;
            border: 2px solid black;
            width: 90px;
            height: 60px;
            margin: 0 12px;
            display: flex;
            justify-content: center;
            align-items: center;
        }
    }
```

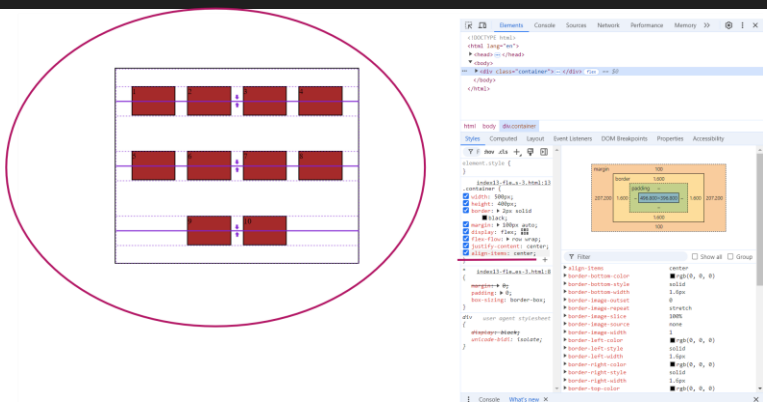


نکته: property به نام align-items که برای ترازبندی در محور Y به کار می رود برای زمانی است که تک خط داریم. (درواقع، زمانی که flex-wrap: wrap; فعال نشده است)

در مثال زیر میبینیم که center در محور Y عمل نکرده است: (line-base در نظر گرفته)

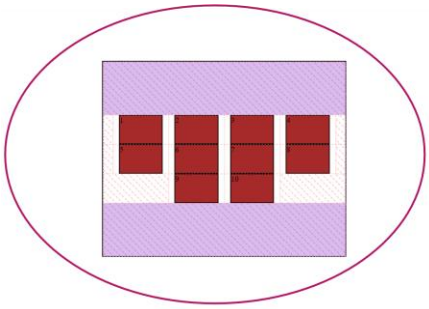
```
    .container{
        width: 500px;
        height: 400px;
        border: 2px solid black;
        margin: 100px auto;
        display: flex;
        flex-flow: row wrap;
        justify-content: center;
        align-items: center;
        >div{
            background-color: brown;
            border: 2px solid black;
            width: 90px;
```

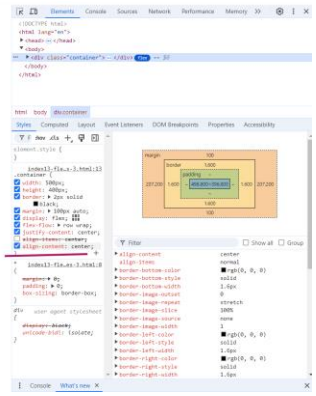
```
            height: 60px;
            margin: 0 12px;
        }
    }
```



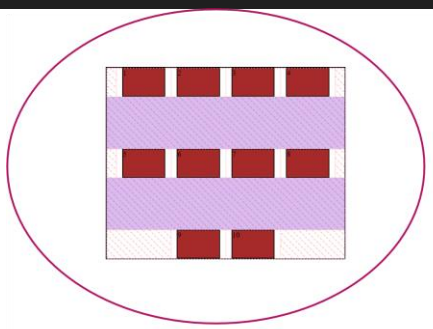
بنابراین اگر چند سطر داشته باشیم، به جای استفاده از property به نام align-items از property به نام align-content استفاده می کنیم.

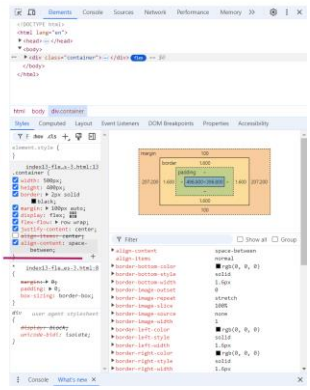
```
    .container{
        width: 500px;
        height: 400px;
        border: 2px solid black;
        margin: 100px auto;
        display: flex;
        flex-flow: row wrap;
        justify-content: center;
        /* align-items: center; */
        align-content: center;
        >div{
            background-color: brown;
            border: 2px solid black;
            width: 90px;
            height: 60px;
            margin: 0 12px;
        }
    }
```





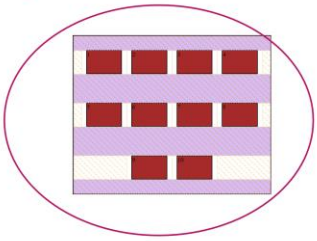
```
.container{
    width: 500px;
    height: 400px;
    border: 2px solid black;
    margin: 100px auto;
    display: flex;
    flex-flow: row wrap;
    justify-content: center;
    /* align-items: center; */
    align-content: space-between;
    >div{
        background-color: brown;
        border: 2px solid black;
        width: 90px;
        height: 60px;
        margin: 0 12px;
    }
}
```

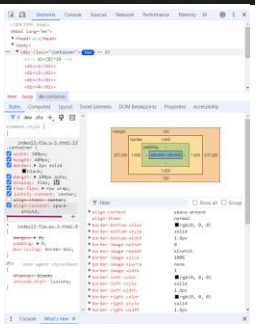




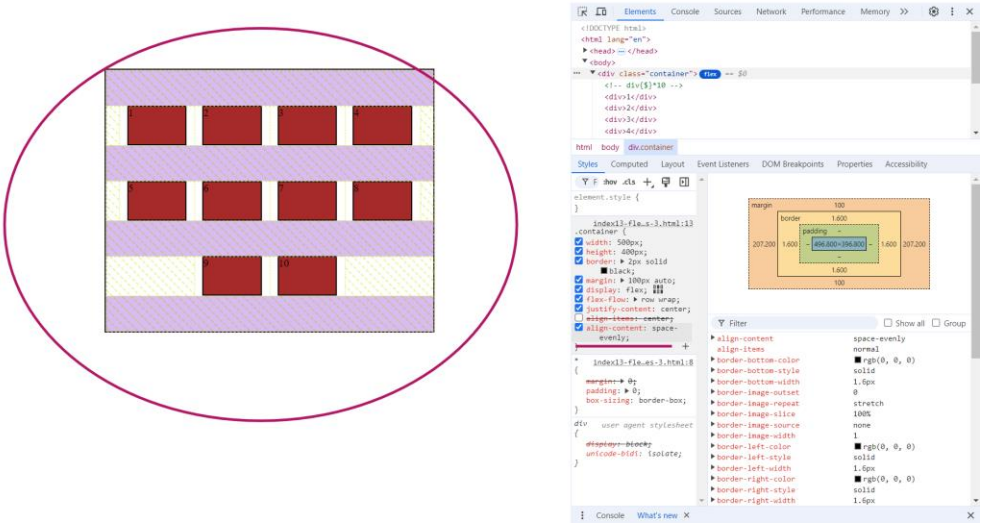
```
.container{
    width: 500px;
    height: 400px;
    border: 2px solid black;
```

```
        margin: 100px auto;
        display: flex;
        flex-flow: row wrap;
        justify-content: center;
        /* align-items: center; */
        align-content: space-around;
        >div{
            background-color: brown;
            border: 2px solid black;
            width: 90px;
            height: 60px;
            margin: 0 12px;
        }
    }
```





```
    .container{
        width: 500px;
        height: 400px;
        border: 2px solid black;
        margin: 100px auto;
        display: flex;
        flex-flow: row wrap;
        justify-content: center;
        /* align-items: center; */
        align-content: space-evenly;
        >div{
            background-color: brown;
            border: 2px solid black;
            width: 90px;
            height: 60px;
            margin: 0 12px;
        }
    }
```



(فاصله دلخواه هم با align-content: center; و لحاظ کردن margin به دست می آید.)

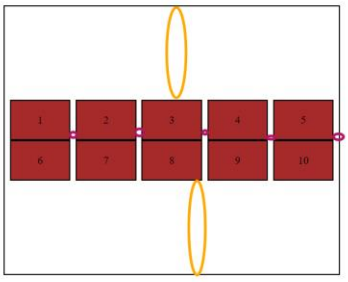
معرفی یک property جدید در flex

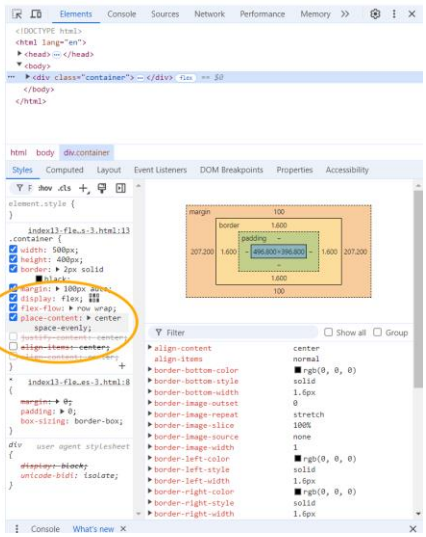
```
place-content: ;
```

(شامل justify-content و align-content هست)

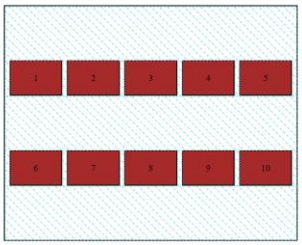
```
/* place-content = alignContent + justifyContent*/
```

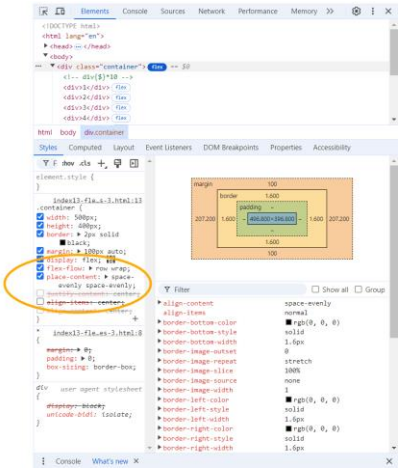
```
        display: flex;
        flex-flow: row wrap;
        place-content: center space-evenly;
```





```
            display: flex;
            flex-flow: row wrap;
            place-content: space-evenly space-evenly;
```





Properties for flex container are done!

Properties for flex children:

```
flex-basis: ;
```

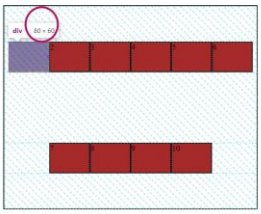
این property حکم width را دارد برای همین width را کامنت می کنیم:

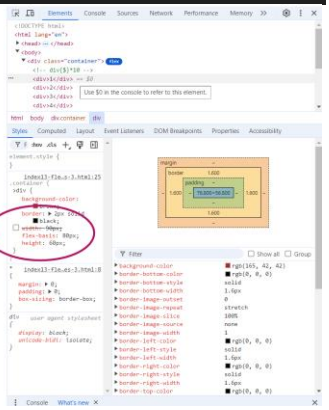
```
            >div{
                background-color: brown;
                border: 2px solid black;
                /* width: 90px; */
                flex-basis:;
                height: 60px;

            }
```

مثال:

```
            >div{
                background-color: brown;
                border: 2px solid black;
                /* width: 90px; */
                flex-basis: 80px;
                height: 60px;
```

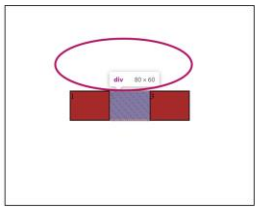
```
        }
```

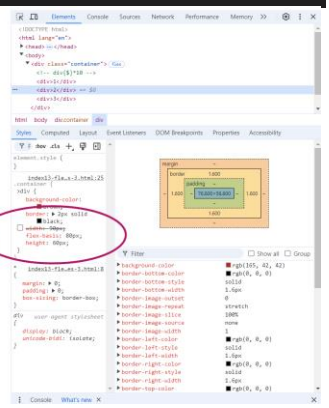




مثال دیگر:

```
        >div{
            background-color: brown;
            border: 2px solid black;
            /* width: 90px; */
            flex-basis: 80px;
            height: 60px;

        }
```



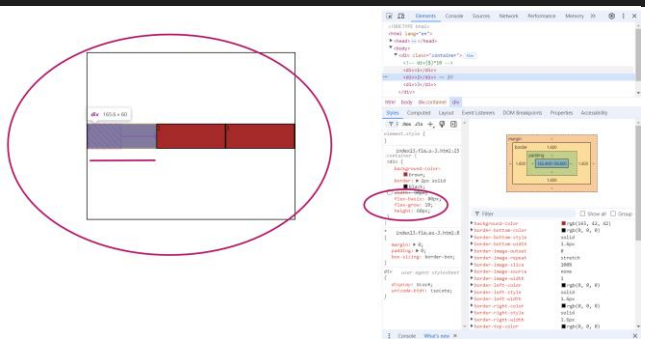


معرفی یک property دیگر برای flex children

```
flex-grow: ;
```

```
        >div{
            background-color: brown;
            border: 2px solid black;
            /* width: 90px; */
```

```
        flex-basis: 80px;
        flex-grow: 10;
        height: 60px;

    }
```



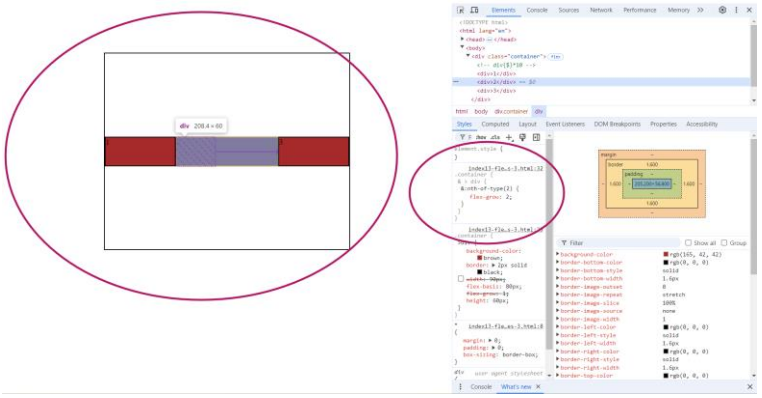
همه divisions دارای یک flex-grow هستند.

با نسبت flex-grow فضای خالی از بین رفت.

اگر فضا زیاد باشد هر کدام از children با چه نسبت بزرگ شوند.

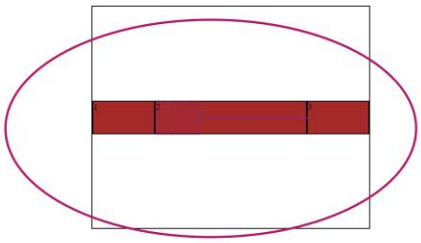
مثال:

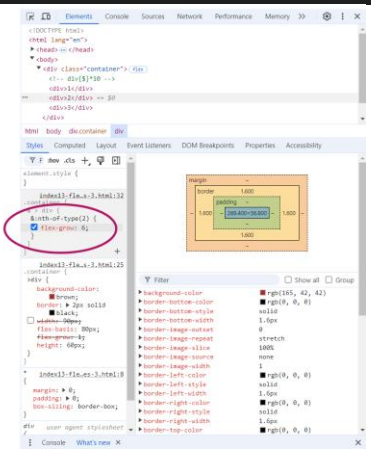
```
    >div{
        background-color: brown;
        border: 2px solid black;
        /* width: 90px; */
        flex-basis: 80px;
        flex-grow: 1;
        height: 60px;
        &:nth-of-type(2){
            flex-grow: 2;
        }

    }
```



مثال:

```
            >div{
                background-color: brown;
                border: 2px solid black;
                /* width: 90px; */
                flex-basis: 80px;
                flex-grow: 1;
                height: 60px;
                &:nth-of-type(2){
                    flex-grow: 6;
                }

            }
```





برعکس flex-grow property، property زیر تعریف می شود:

```
flex-shrink: ;
```

(اگر فضا کم باشد هر کدوم از children با چه نسبتی کوچک شوند)

معرفی property به نام flex:

```
flex: ;
```

(قبلا value بود.)

```
                /* flex: shrink grow basis; */
```

اگر flex-wrap: wrap;، flex-shrink قطعا بی معنی است.

سوال: به چه صورت property زیر با دو مقدار shrink و grow که در تضاد هستند، فعال می شود؟

flex: shrink grow basis;

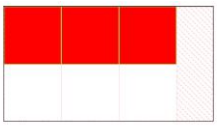
پاسخ: در شرایط responsive.

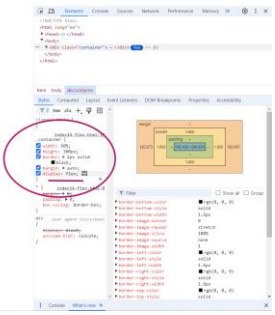
شرط: width برای parent درصدی باشد.

فرض کنید داریم:

```
.container{
    width: 60%;
    height: 300px;
    border: 2px solid black;
    margin: auto;
    >div{
        width: 150px;
        height: 150px;
        background-color: red;
        border: 2px solid yellowgreen;
    }
}
```

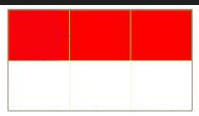
به parent، flex display بدهیم:

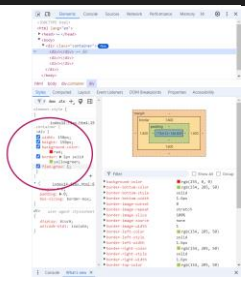




flex-grow را برای children فعال می کنیم:

```
.container{
    width: 60%;
    height: 300px;
    border: 2px solid black;
    margin: auto;
    display: flex;
    >div{
        width: 150px;
        height: 150px;
        background-color: red;
        border: 2px solid yellowgreen;
        flex-grow: 1;
    }
}
```

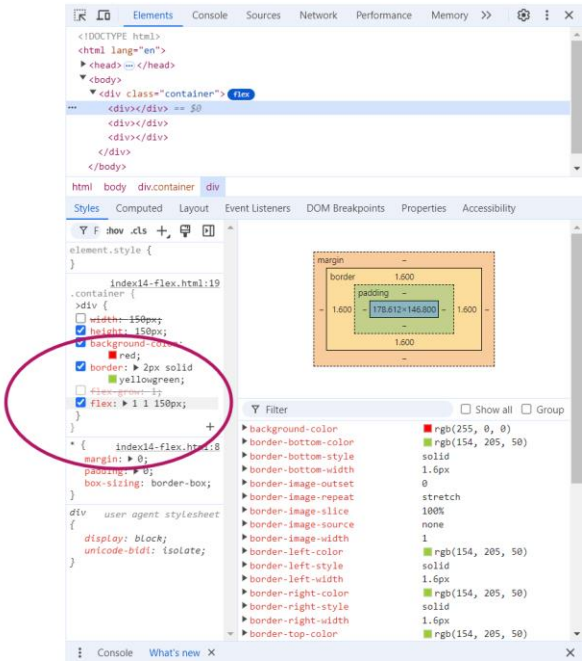




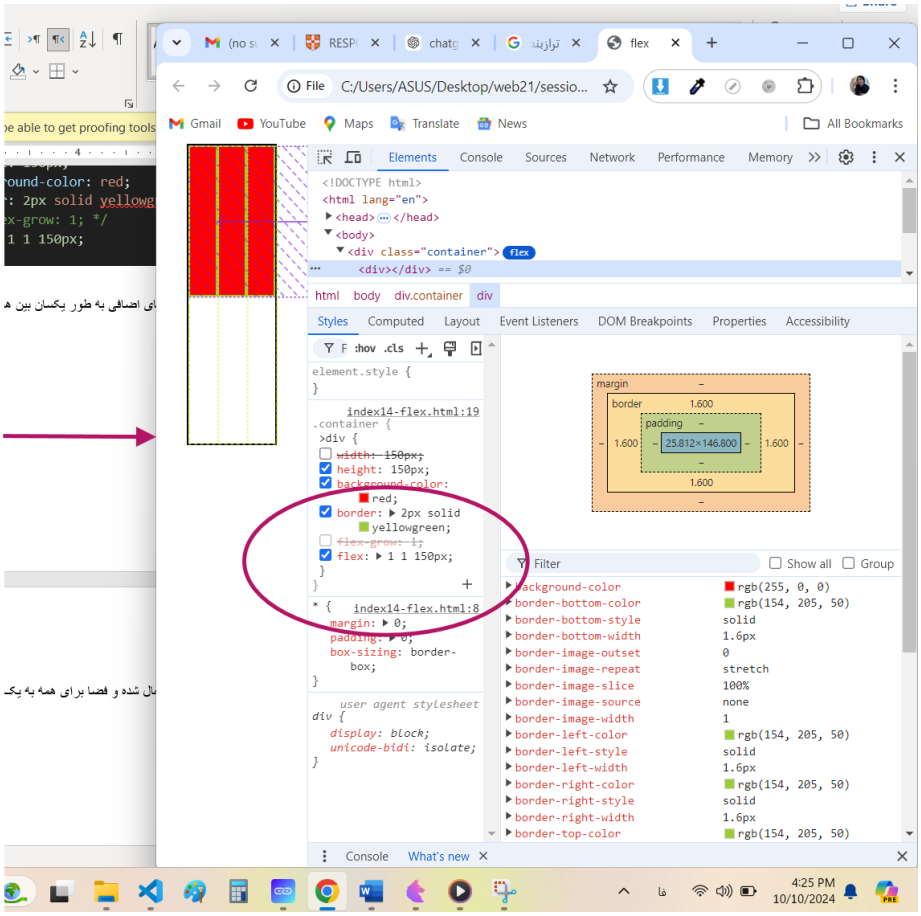
(یا استفاده از complex به صورت flex: shrink grow basis;)

```
    >div{
        /* width: 150px; */
        height: 150px;
        background-color: red;
        border: 2px solid yellowgreen;
        /* flex-grow: 1; */
        flex: 1 1 150px;
    }
```

1) فضا زیاده و flex-grow فعال شده یعنی فضای اضافی به طور یکسان بین همه پخش شده:



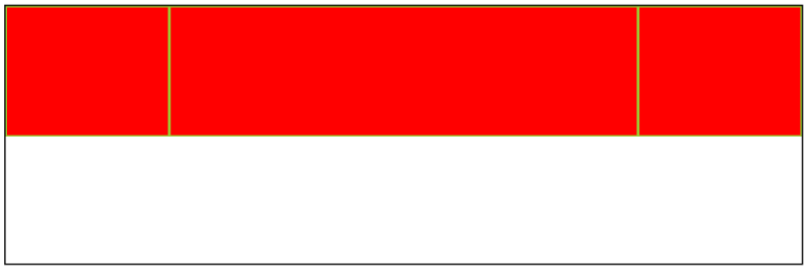
2) فضا کم شده (مثلا موبایل) و flex-shrink فعال شده و فضا برای همه به یک نسبت کم می شود:



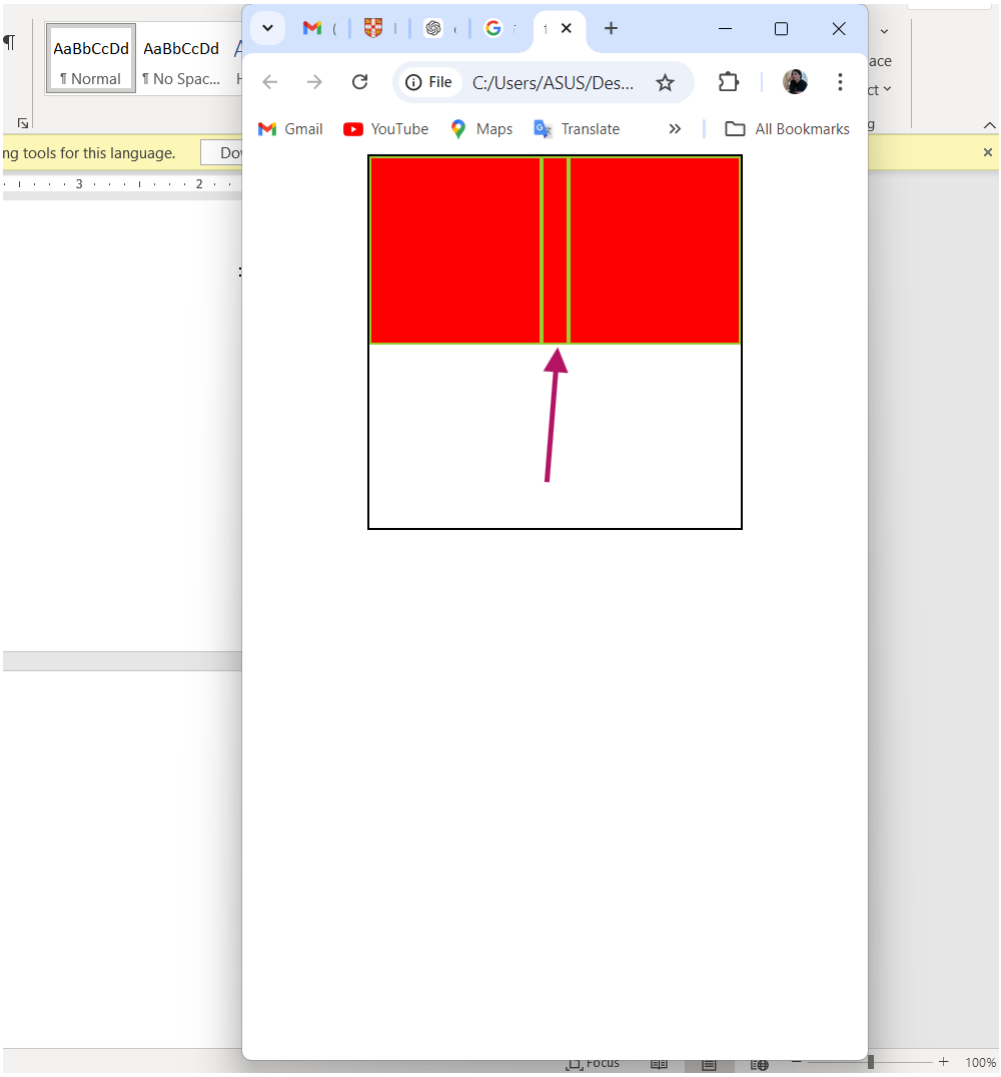
مثال:

```
.container{
    width: 60%;
    height: 300px;
    border: 2px solid black;
    margin: auto;
    display: flex;
    >div{
        height: 150px;
        background-color: red;
        border: 2px solid yellowgreen;
        flex: 1 1 150px;
        &:nth-of-type(2){
            flex-grow: 10;
            flex-shrink: 10;
        }
    }
}
```

1) خروجی وقتی فضا زیاده:



2) خروجی وقتی فضا کمه:



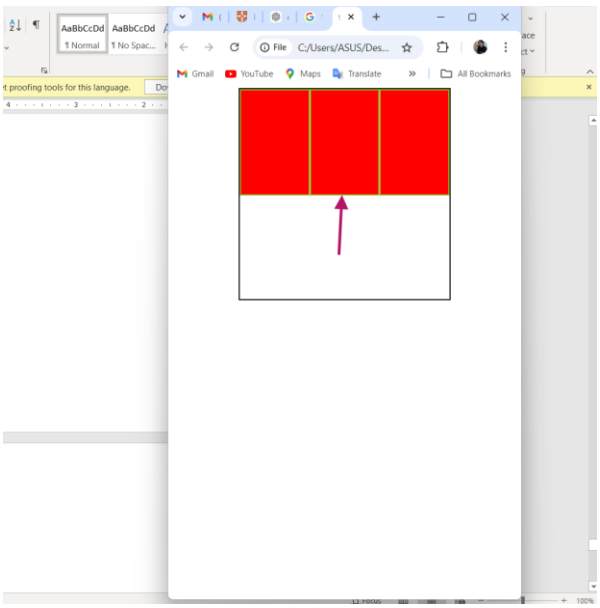
مثال:

```
.container{
    width: 60%;
    height: 300px;
    border: 2px solid black;
    margin: auto;
    display: flex;
    >div{
        height: 150px;
        background-color: red;
        border: 2px solid yellowgreen;
```

```
            flex: 1 1 150px;
            &:nth-of-type(2){
                flex-grow: 10;

            }
        }
    }
```

1) خروجی وقتی فضا زیاده:

2) خروجی وقتی فضا کمه:



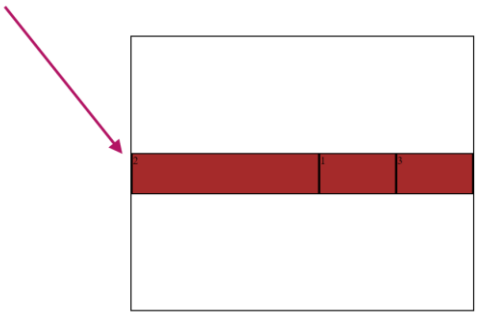
معرفی property به نام order:

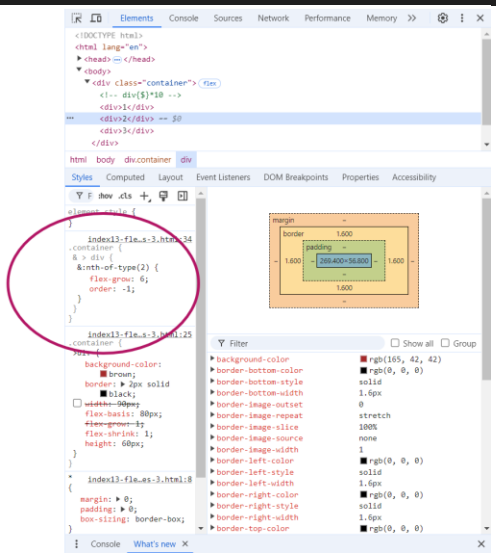
```
order: ;
```

order برای همه عناصر صفر است.

مثال:

```
        >div{
            background-color: brown;
            border: 2px solid black;
            /* width: 90px; */
            flex-basis: 80px;
            flex-grow: 1;
            flex-shrink: 1;
            /* flex: shrink grow basis; */
            height: 60px;
            &:nth-of-type(2){
                flex-grow: 6;
                order: -1;
            }

        }
```





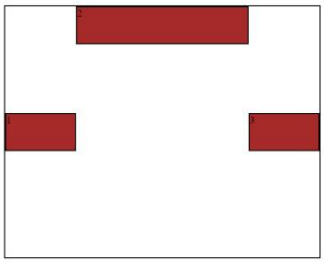
با order می توانیم به راحتی جایگاه عناصر را تغییر دهیم بدون آن که تغییری در html ایجاد کنیم.

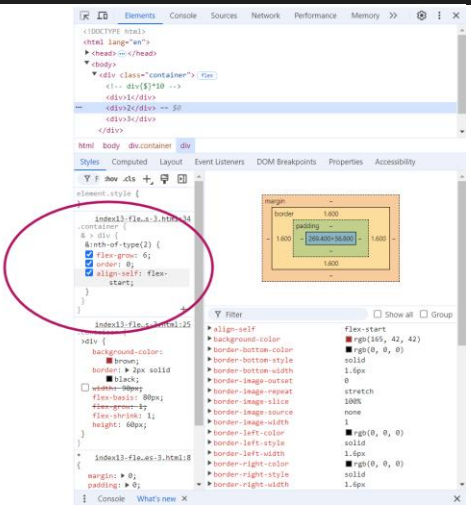
معرفی property به نام align-self:

```
                align-self: ;
```

خود المان به تنهایی در ۷، alignment می شود.

```
>div{
    background-color: brown;
    border: 2px solid black;
    /* width: 90px; */
    flex-basis: 80px;
    flex-grow: 1;
    flex-shrink: 1;
    /* flex: shrink grow basis; */
    height: 60px;
    &:nth-of-type(2){
        flex-grow: 6;
        order: 0;
        align-self: flex-start;
    }

}
```





slider= a part on a machine, computer, etc. that is used to control something, for example the volume that something is played at.

نکته: vh یعنی view height

```
height: 100vh;
```

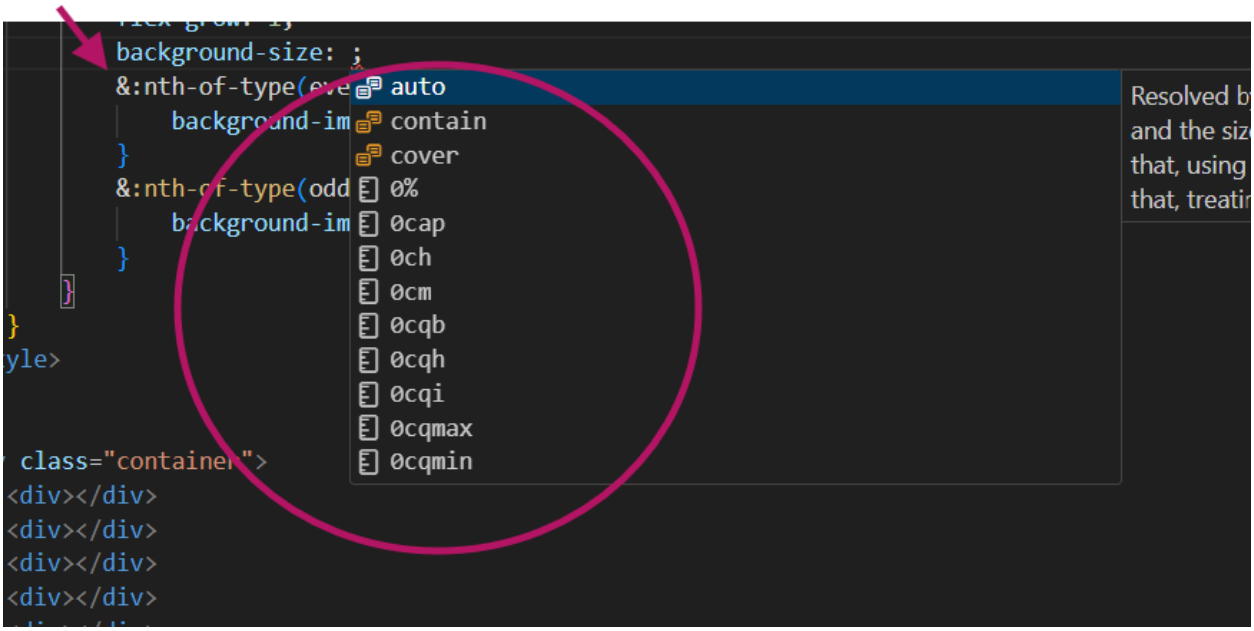
معرفی property به نام background-image:

```
background-image: url();
```

کاغذ دیواری، save نمیشه، المان ها روی آن قرار می گیرند

معرفی property به نام background-size:

```
background-size: ;
```



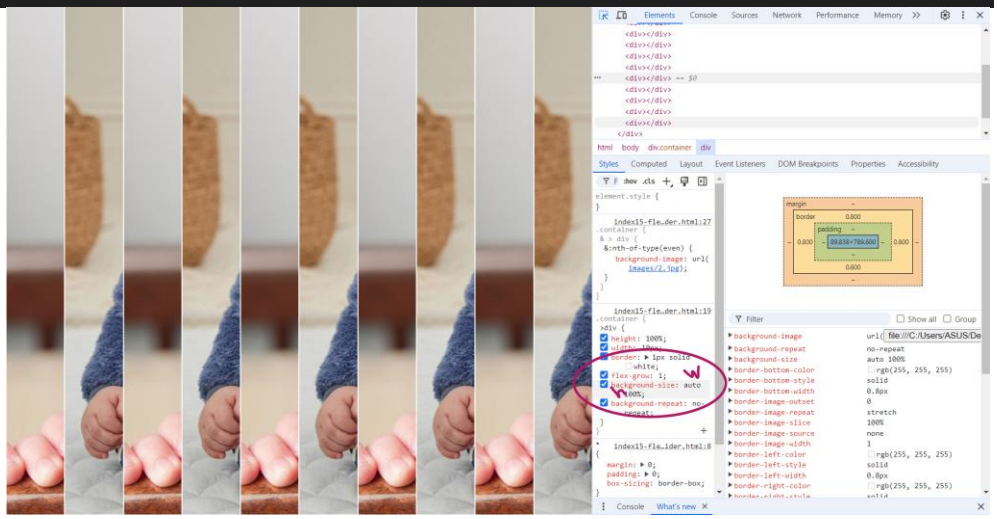
مثال:

```
background-size: auto 100%;
```

auto، X است و 100%، Y است.

```
.container{
    width: 100%;
    height: 100vh;
    background-color: black;
    display: flex;
    flex-flow: row nowrap;
    >div{

        height: 100%;
        width: 10px;
        border: 1px solid white;
        flex-grow: 1;
        background-size: auto 100%;
        &:nth-of-type(even){
            background-image: url(images/2.jpg);
        }
        &:nth-of-type(odd){
            background-image: url(images/1.jpg);
```

```
            }
        }
    }
```



قرار است هرعکس که hover شود، آن عکس بزرگ شود.

```
    .container{
        width: 100%;
        height: 100vh;
        background-color: black;
        display: flex;
        flex-flow: row nowrap;
        >div{
            &:hover{
                flex-grow: 15;
            }
            transition: all 2s;
            height: 100%;
            width: 10px;
            border: 1px solid white;
            flex-grow: 1;
            background-size: auto 100%;
            /* background-position: center; */

            &:nth-of-type(even){
                background-image: url(images/2.jpg);
            }
            &:nth-of-type(odd){
                background-image: url(images/1.jpg);
            }
        }
    }
```

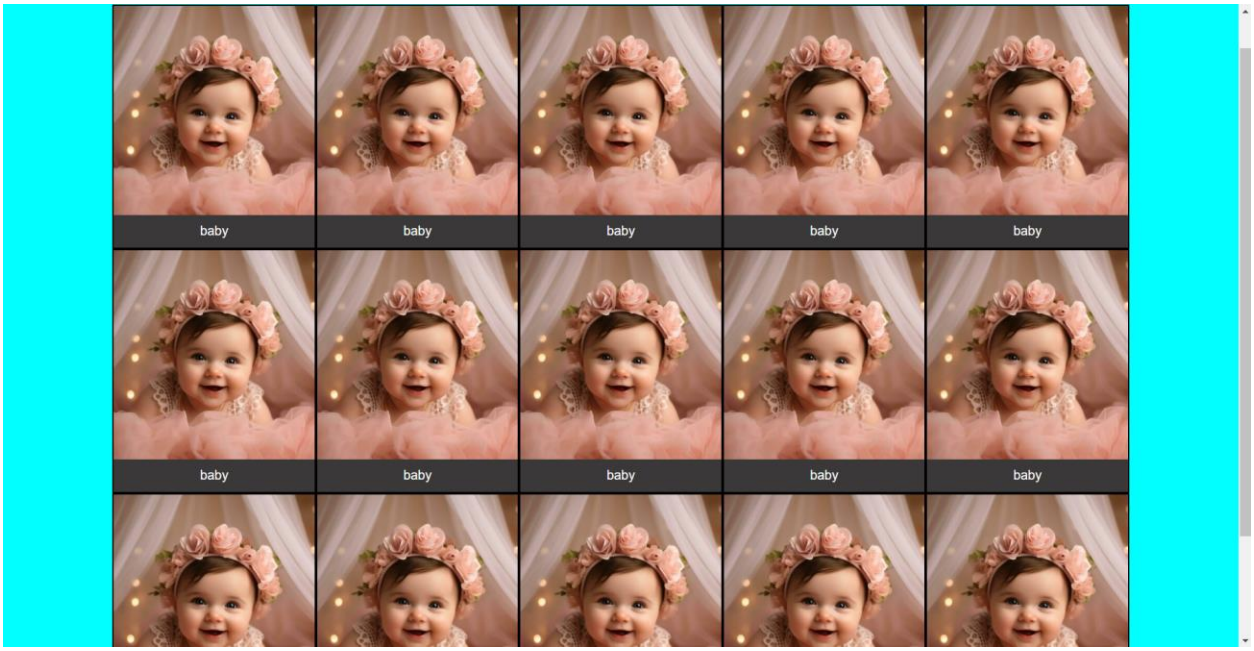
```scss
.container{
    width: 100%;
    height: 100vh;
    background-color: black;
    display: flex;
    flex-flow: row nowrap;
    >div{
        &:hover{
            flex-grow: 7;
        }
        transition: all 2s;
        height: 100%;
        width: 10px;
        border: 1px solid white;
        flex-grow: 1;
        background-size: auto 100%;
        background-position: center;
        background-repeat: no-repeat;

        &:nth-of-type(even){
            background-image: url(images/2.jpg);
        }
        &:nth-of-type(odd){
            background-image: url(images/1.jpg);
        }
    }
```

```
}
```

به کمک flex می توانیم از شرایط responsive بهره ببریم.

parent دارای flex-wrap: wrap; باشد و children پیکسلی باشند.

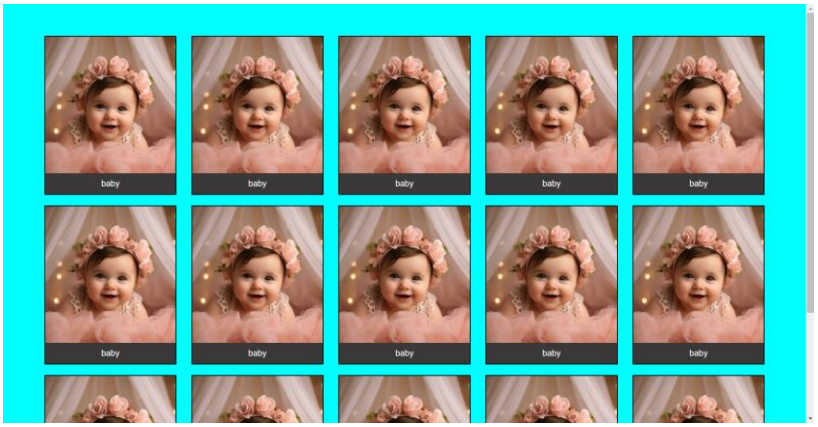
نکته: هنگامی که ارتفاع parent، auto است، نمی توان با align-items و align-content در Y کنترل انجام داد چرا که فضای خالی نداریم:



راه حل: استفاده از margin

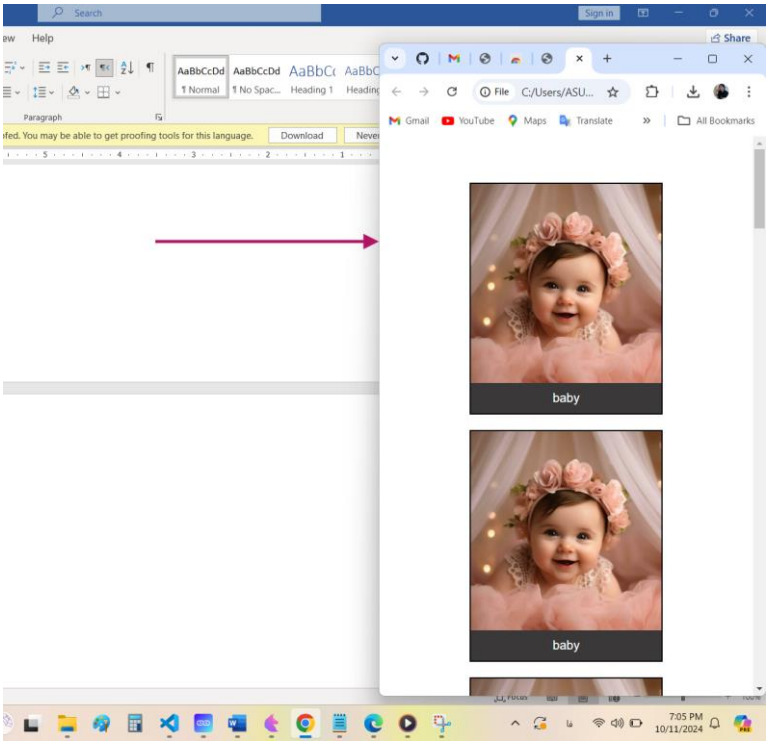
```
margin: 10px 0;
```

می توان با کمک justify-content با value به نام space-evenly هم در محور X کنترل انجام داد:



(50px میزان padding داشتیم)

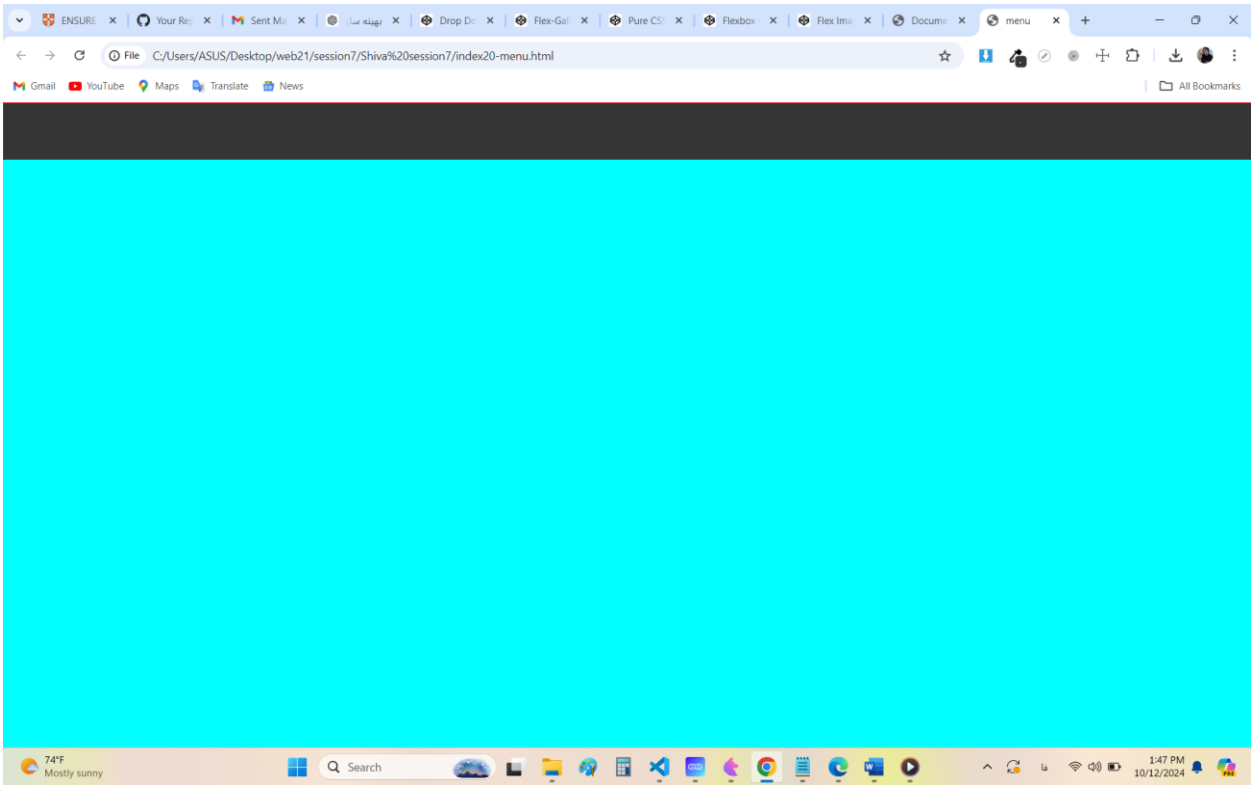
کاملا responsive انجام شد چرا که parent دارای flex-wrap: wrap; بود و children پیکسلی بودند



ایجاد یک منو ساده:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>menu</title>
    <style>
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        body{
            width: 100%;
            height: auto;
            background-color: aqua;
            >.wrapper{
                width: 100%;
                height: 70px;
                background-color: rgb(53, 53, 53);
                border: 1px solid red;
            }
        }
    </style>
</head>
<body>
    <nav class="wrapper">
        <ul>
            <li><a href=""></a></li>
            <li><a href=""></a></li>
            <li><a href=""></a></li>
            <li><a href=""></a></li>
            <li><a href=""></a></li>
        </ul>
    </nav>
</body>
</html>
```

خروجی تا اینجا:

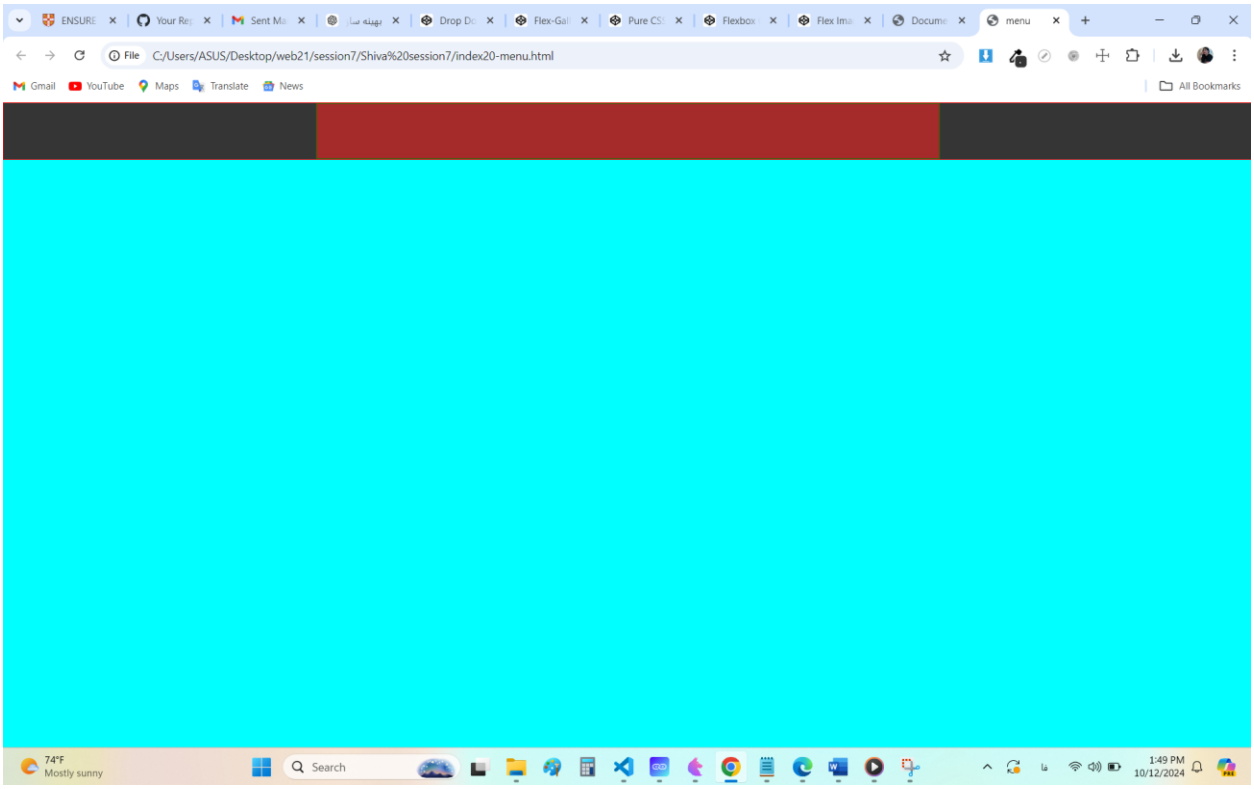


ادامه کد:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>menu</title>
    <style>
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        body{
            width: 100%;
            height: auto;
            background-color: aqua;
            >.wrapper{
                width: 100%;
                height: 70px;
                background-color: rgb(53, 53, 53);
                border: 1px solid red;
```

```
                display: flex;
                flex-flow: row wrap;
                justify-content: center;
                >ul{
                    border: 1px solid green;
                    width: 50%;
                    height: 100%;
                    background-color: brown;
                    >li{
                        list-style-type: none;
                        >a{

                        }
                    }
                }
            }
        }
    </style>
</head>
<body>
    <nav class="wrapper">
        <ul>
            <li><a href=""></a></li>
            <li><a href=""></a></li>
            <li><a href=""></a></li>
            <li><a href=""></a></li>
            <li><a href=""></a></li>
        </ul>
    </nav>
</body>
</html>
```

خروجی تا اینجا:



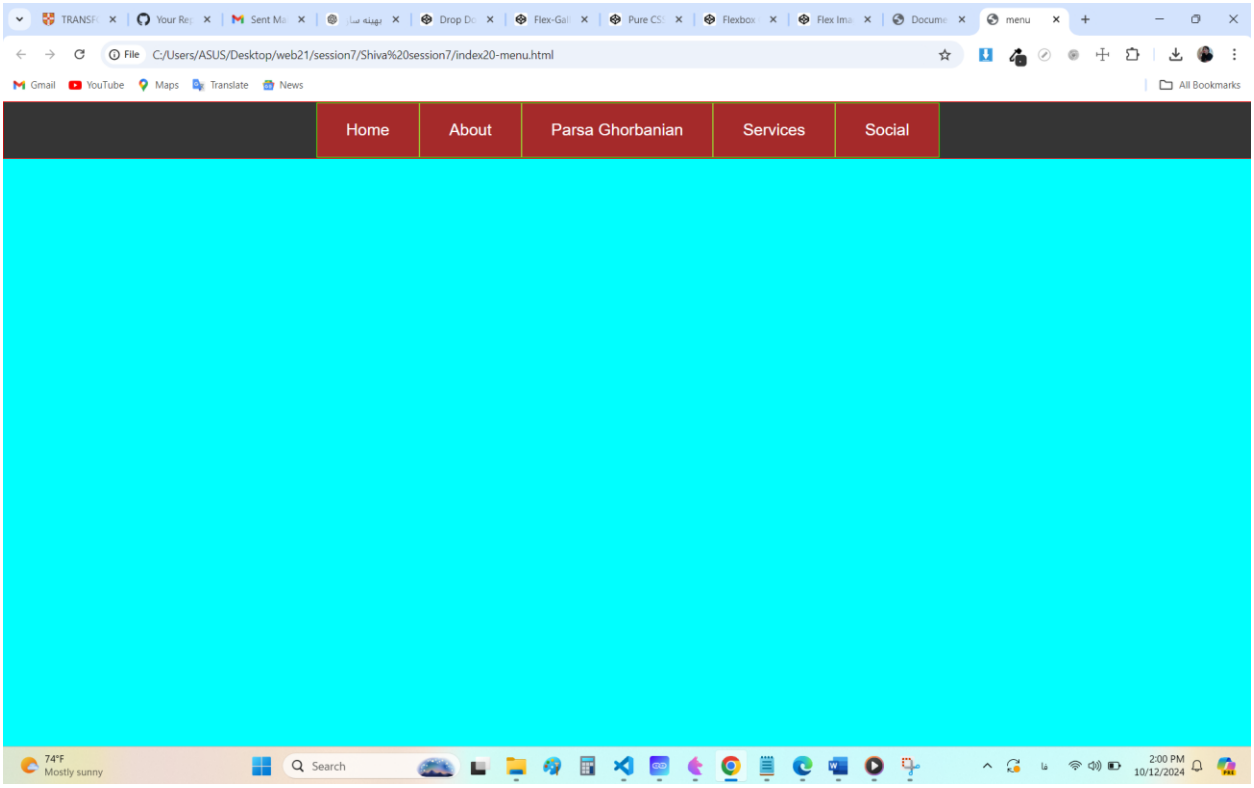
ادامه کد:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>menu</title>
    <style>
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        body{
            width: 100%;
            height: auto;
            background-color: aqua;
            >.wrapper{
                width: 100%;
                height: 70px;
                background-color: rgb(53, 53, 53);
                border: 1px solid red;
                display: flex;
```

```
                flex-flow: row wrap;
                justify-content: center;
                >ul{
                    border: 1px solid green;
                    width: 50%;
                    height: 100%;
                    background-color: brown;
                    display: flex;
                    flex-flow: row nowrap;

                    >li{
                        flex-grow: 1;
                        border: 1px solid yellowgreen;
                        list-style-type: none;
                        >a{
                            color: white;
                            font-family: Arial, "Helvetica Neue", Helvetica,
sans-serif;
                            font-size: 20px;
                            text-decoration: none;
                            display: flex;
                            flex-flow: row nowrap;
                            width: 100%;
                            height: 100%;
                            justify-content: center;
                            align-items: center;
                           text-transform: capitalize;
                        }
                    }
                }
            }
        }
    </style>
</head>
<body>
    <nav class="wrapper">
        <ul>
            <li><a href="">home</a></li>
            <li><a href="">about</a></li>
            <li><a href="">parsa ghorbanian</a></li>
            <li><a href="">services</a></li>
            <li><a href="">social</a></li>
        </ul>
    </nav>
</body>
```

```
</html>
```

خروجی تا اینجا:



ادامه کد:

(ایجاد منو drop-down در لیست سوم که Parsa Ghorbanian هست)

drop-down menu: a list of choices on a computer screen that is hidden until you choose to look at it

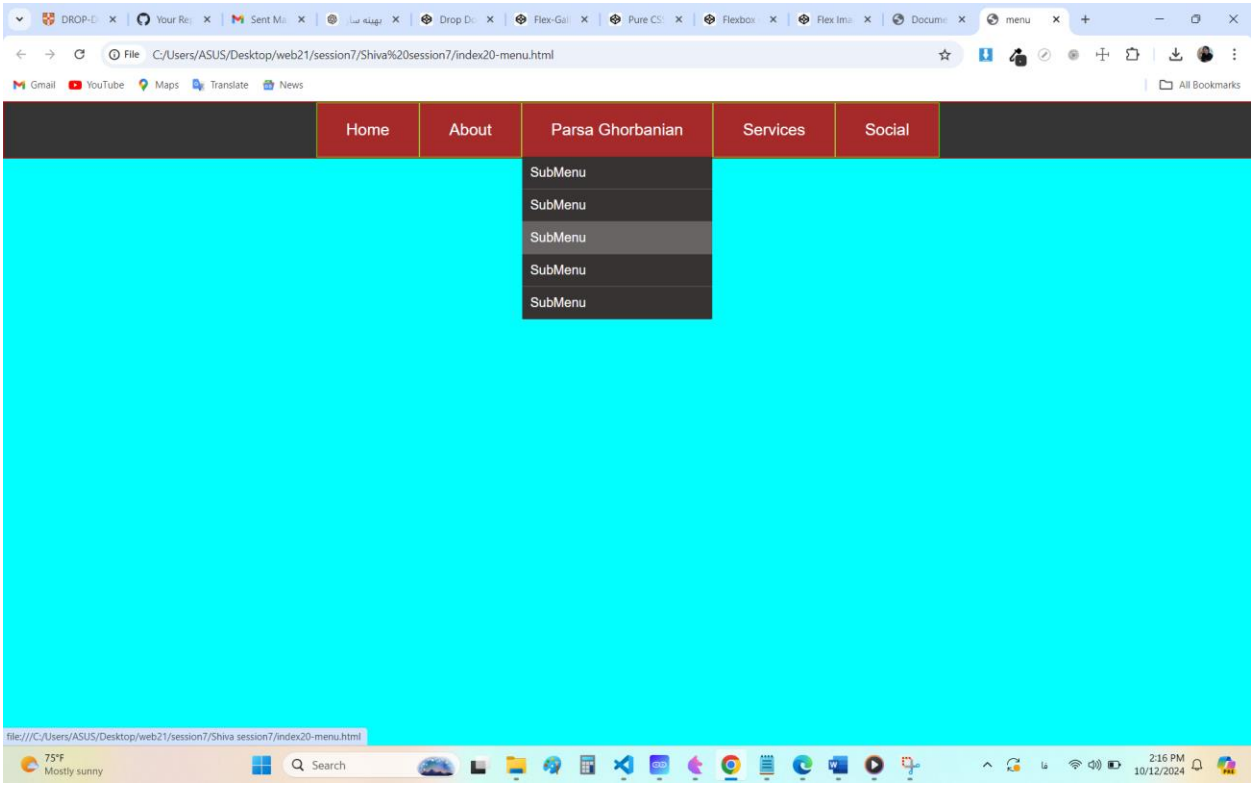
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>menu</title>
    <style>
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        body{
            width: 100%;
            height: auto;
```

```
            background-color: aqua;
            >.wrapper{
                width: 100%;
                height: 70px;
                background-color: rgb(53, 53, 53);
                border: 1px solid red;
                display: flex;
                flex-flow: row wrap;
                justify-content: center;
                >ul{
                    border: 1px solid green;
                    width: 50%;
                    height: 100%;
                    background-color: brown;
                    display: flex;
                    flex-flow: row nowrap;

                    >li{
                        flex-grow: 1;
                        border: 1px solid yellowgreen;
                        list-style-type: none;
                        &:nth-of-type(3){
                            >ul{
                                background-color: rgb(55, 51, 51);
                                >li{
                                    height: 40px;
                                    border-bottom: 1px solid rgba(255, 255, 255,
0.207);
                                    list-style-type: none;
                                    >a{
                                        &:hover{
                                            background-color: rgba(255, 255, 255,
0.241);
                                        }
                                        text-decoration: none;
                                        color: white;
                                        font-family: Arial, "Helvetica Neue",
Helvetica, sans-serif;

                                        display: flex;
                                        flex-flow: row nowrap;
                                        align-items: center;
                                        width: 100%;
                                        height: 100%;
                                        padding-left: 10px;
```

```
                                }
                            }
                        }
                    }
                    >a{
                        &:hover{
                            background-color: rgba(255, 255, 255, 0.193);
                        }
                        color: white;
                        font-family: Arial, "Helvetica Neue", Helvetica,
sans-serif;
                        font-size: 20px;
                        text-decoration: none;
                        display: flex;
                        flex-flow: row nowrap;
                        width: 100%;
                        height: 100%;
                        justify-content: center;
                        align-items: center;
                       text-transform: capitalize;
                    }
                }
            }
        }
    </style>
</head>
<body>
    <nav class="wrapper">
        <ul>
            <li><a href="">home</a></li>
            <li><a href="">about</a></li>
            <li><a href="">parsa ghorbanian</a>
                <ul>
                    <li><a href="">SubMenu</a></li>
                    <li><a href="">SubMenu</a></li>
                    <li><a href="">SubMenu</a></li>
                    <li><a href="">SubMenu</a></li>
                    <li><a href="">SubMenu</a></li>
                </ul>
            </li>
            <li><a href="">services</a></li>
            <li><a href="">social</a></li>
        </ul>
    </nav>
```

```
</body>
</html>
```

خروجی تا اینجا:



پنهان کردن drop-down menu:

روش اول استفاده از display با مقادیر none و block:

نکته: display، transition نمی گیرد.

کد:

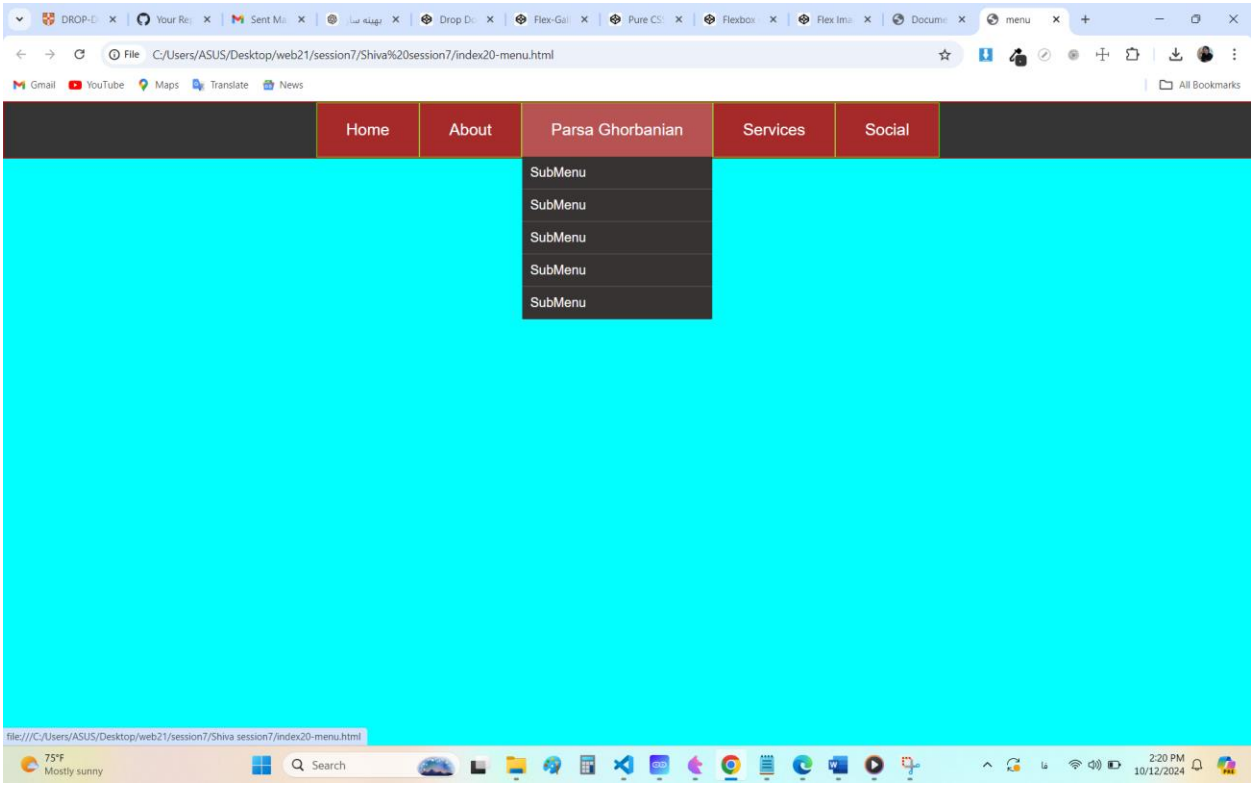
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>menu</title>
    <style>
        *{
            margin: 0;
            padding: 0;
            box-sizing: border-box;
```

```
        }
        body{
            width: 100%;
            height: auto;
            background-color: aqua;
            >.wrapper{
                width: 100%;
                height: 70px;
                background-color: rgb(53, 53, 53);
                border: 1px solid red;
                display: flex;
                flex-flow: row wrap;
                justify-content: center;
                >ul{
                    border: 1px solid green;
                    width: 50%;
                    height: 100%;
                    background-color: brown;
                    display: flex;
                    flex-flow: row nowrap;

                    >li{
                        flex-grow: 1;
                        border: 1px solid yellowgreen;
                        list-style-type: none;
                        &:nth-of-type(3){
                            &:hover{
                                >ul{
                                    display: block;
                                }
                            }
                            >ul{
                                display: none;
                                background-color: rgb(55, 51, 51);
                                >li{
                                    height: 40px;
                                    border-bottom: 1px solid rgba(255, 255, 255,
0.207);
                                    list-style-type: none;
                                    >a{
                                        &:hover{
                                            background-color: rgba(255, 255, 255,
0.241);
                                        }
                                        text-decoration: none;
```

```
                                        color: white;
                                        font-family: Arial, "Helvetica Neue",
Helvetica, sans-serif;

                                        display: flex;
                                        flex-flow: row nowrap;
                                        align-items: center;
                                        width: 100%;
                                        height: 100%;
                                        padding-left: 10px;
                                    }
                                }
                            }
                        }
                        >a{
                            &:hover{
                                background-color: rgba(255, 255, 255, 0.193);
                            }
                            color: white;
                            font-family: Arial, "Helvetica Neue", Helvetica,
sans-serif;
                            font-size: 20px;
                            text-decoration: none;
                            display: flex;
                            flex-flow: row nowrap;
                            width: 100%;
                            height: 100%;
                            justify-content: center;
                            align-items: center;
                           text-transform: capitalize;
                        }
                    }
                }
            }
        }
    </style>
</head>
<body>
    <nav class="wrapper">
        <ul>
            <li><a href="">home</a></li>
            <li><a href="">about</a></li>
            <li><a href="">parsa ghorbanian</a>
                <ul>
                    <li><a href="">SubMenu</a></li>
```

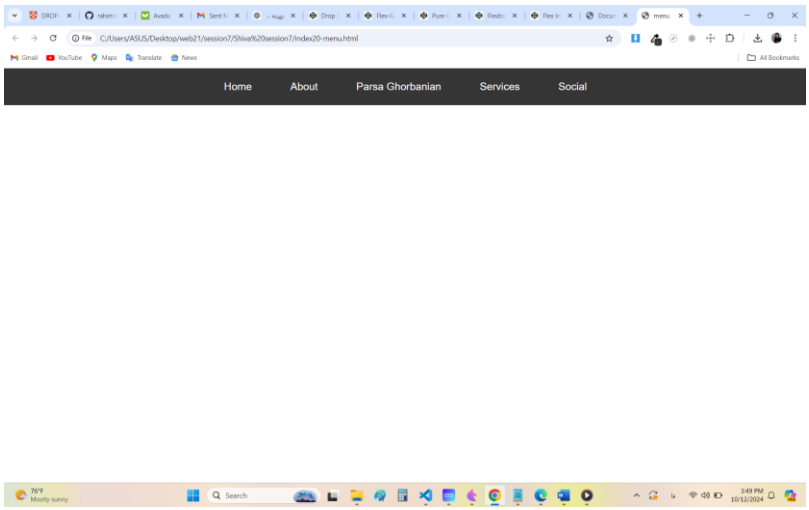
```
                    <li><a href="">SubMenu</a></li>
                    <li><a href="">SubMenu</a></li>
                    <li><a href="">SubMenu</a></li>
                    <li><a href="">SubMenu</a></li>
                </ul>
            </li>
            <li><a href="">services</a></li>
            <li><a href="">social</a></li>
        </ul>
    </nav>
</body>
</html>
```

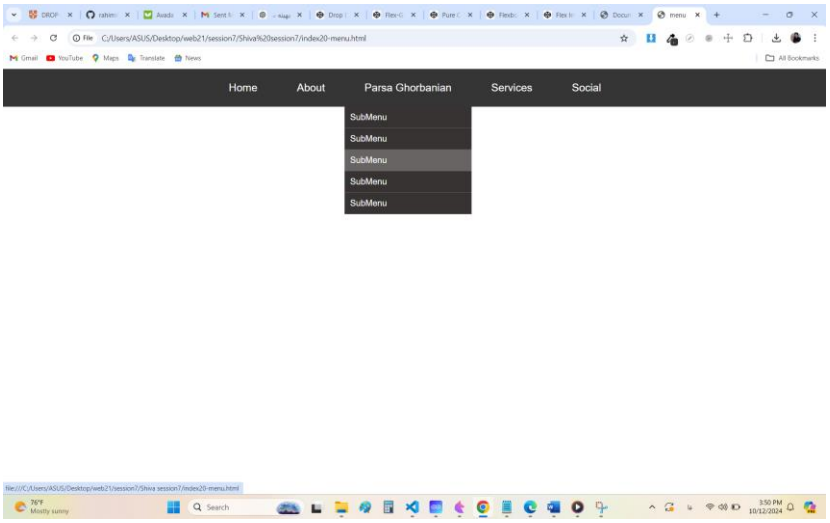
خروجی:



نکته: display، transition نمی گیرد.

خروجی بعد از کامنت کردن تست ها:





روش دوم استفاده از opacity و visibility.

opacity، transition می گیرد. (fade میشه)

برای جلوگیری از حالت ghost در کنار opacity با مقدار صفر باید از visibility با مقدار hidden استفاده کرد.

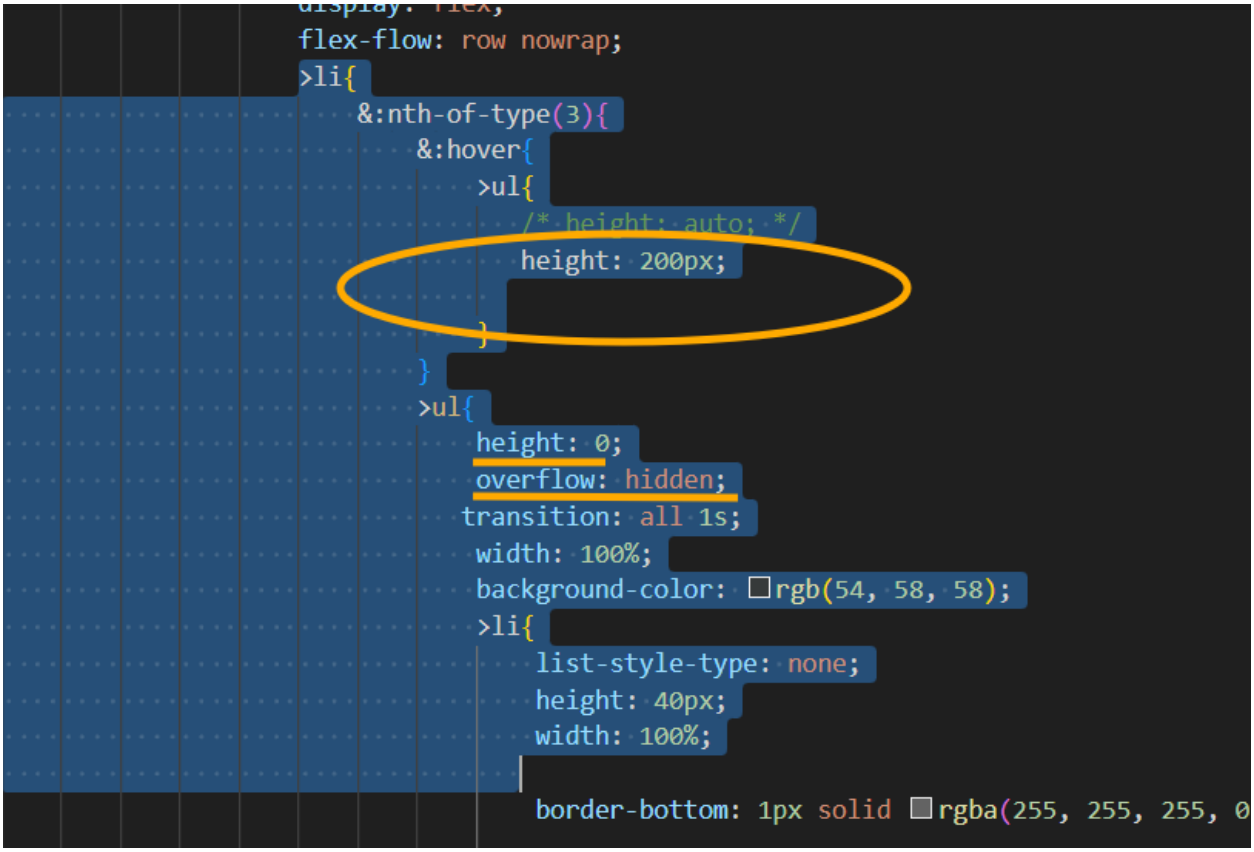
```
30                  display: flex;
31                  flex-flow: row nowrap;
32
33                  >li{
34                      flex-grow: 1;
35                      /* border: 1px solid yellowgreen; */
36                      list-style-type: none;
37                      &:nth-of-type(3){
38
39                          &:hover{
40                              >ul{
41                                  opacity: 1;
42                                  visibility: visible;
43                              }
44                          }
45                          >ul{
46
47                              opacity: 0;
48                              visibility: hidden;
49                              background-color: rgb(55, 51, 51);
50                              >li{
51                                  width: 100%;
```

روش سوم استفاده از height و overflow: (بهترین روش)

از آنجایی که وقتی height یک المان صفر می شود محتویات آن سرریز می شود، باید از مقدار hidden برای property به نام overflow استفاده کرد.

```
30                  display: flex;
31                  flex-flow: row nowrap;
32
33                  >li{
34                      flex-grow: 1;
35                      /* border: 1px solid yellowgreen; */
36                      list-style-type: none;
37                      &:nth-of-type(3){
38
39                          &:hover{
40                              >ul{
41                                  height: auto;
42                              }
43                          }
44                          >ul{
45                              height: 0;
46                              overflow: hidden;
47
48                              background-color: rgb(55, 51, 51);
49                              >li{
50                                  width: 100%;
51                                  height: 40px;
52                                  border-bottom: 1px solid rgba(255, 255, 255, 0
```

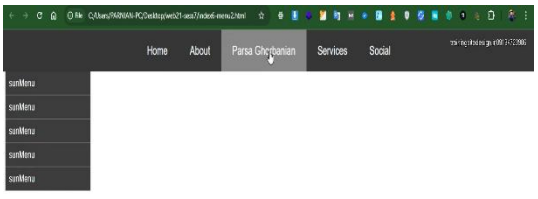
نکته: height با مقدار auto، transition نمی گیرد بنابراین می توان height را دستی حساب کرد و به جای auto قرار داد.



روش چهارم استفاده از position: (effect میشه)

```
flex-flow: row nowrap;
>li{
    &:nth-of-type(4){
      position: relative;
        &:hover{
            >ul{
            top: 70px;
            left: 0;

            }
        }
        >ul{
            position: absolute;
            top: 70px;
            left: -10000px;
            transition: all 1s;
            background-color: rgb(56, 56, 56);
            width: 100%;
            >li{
                height: 40px;
                /* background-color: brown; */
                list-style-type: none;
                border-bottom: 1px solid rgba(255, 255, 255, 0.274);
                width: 100%;
                >a{
                    text-decoration: none;
                    font-family: Arial, "Helvetica Neue", Helvetica, sans-seri
                    color: white;
                    display: flex;
                    width: 100%;
                    height: 100%;
                    align-items: center;
                    padding-left: 10px;
                    &:hover{
```

خروجی 1:



خروجی2: