CSS Flex property

The flex property sets the flexible length on flexible items. If the element is not a flexible item, the flex property has no effect.

NOTE: IE11 and newer versions support the flex property. Internet Explorer 10 supports an alternative, the -ms-flex property.

CSS Syntax:

flex: *flex-grow flex-shrink flex-basis*|auto|initial|inherit;

The flex property may be specified using one, two, or three values.

- **One-value syntax:** the value must be one of:
    - a <number>: In this case it is interpreted as flex: <number> 1 0; the flex-shrink value is assumed to be 1 and the flex-basis value is assumed to be 0.
    - a <width>: In this case it is interpreted as flex: 1 1 <width>; the flex-grow value is assumed to be 1 and the flex-shrink value is assumed to be 1.
    - one of the keywords: none, auto, or initial.
- **Two-value syntax:**
    - The first value must be:
        - a <number> and it is interpreted as <flex-grow>.
    - The second value must be one of:
        - a <number>: then it is interpreted as <flex-shrink>.
        - a valid value for width: then it is interpreted as <flex-basis>.
- **Three-value syntax:** the values must be in the following order:
    1. a <number> for <flex-grow>.
    2. a <number> for <flex-shrink>.
    3. a valid value for width for <flex-basis>.

Eg:

```
/* Two values: flex-grow | flex-basis */
flex: 1 30px;

/* Two values: flex-grow | flex-shrink */
flex: 2 2;
```

/* Three values: flex-grow | flex-shrink | flex-basis */
flex: 2 2 10%;

Keywords

auto-sets 1 and 1 to grow and shrink

initial-keyword is used to set a CSS property to its default value

inherit-keyword specifies that a property should inherit its value from its parent element

The flex property is a shorthand property for:

1. **flex-grow:** The flex-grow property specifies how much the item will grow relative to the rest of the flexible items inside the same container.

Note: If the element is not a flexible item, the flex-grow property has no effect.

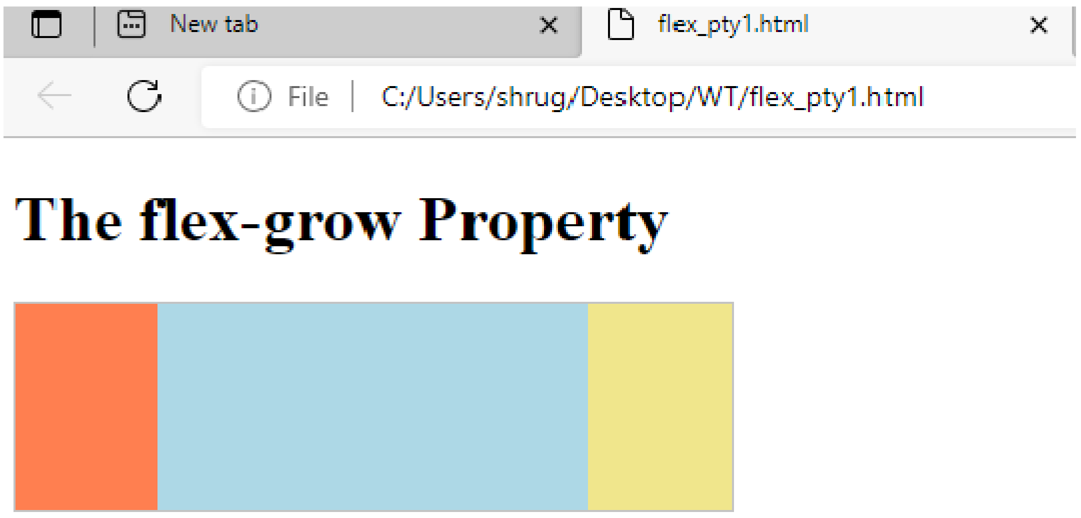
Eg:

div:nth-of-type(1) {flex-grow: 1;}
div:nth-of-type(2) {flex-grow: 3;}

Here, the second flex-item grow three times wider than the first.

Demo:

```
<!DOCTYPE html>

<html>

<head>

<style>

#main {

  width: 350px;

  height: 100px;

  border: 1px solid #c3c3c3;

  display: flex;
}

#main div:nth-of-type(1) {flex-grow: 1;}

#main div:nth-of-type(2) {flex-grow: 3;}

#main div:nth-of-type(3) {flex-grow: 1;}

</style>

</head>

<body>

<h1>The flex-grow Property</h1>

<div id="main">

  <div style="background-color:coral;"></div>

  <div style="background-color:lightblue;"></div>

  <div style="background-color:khaki;"></div>

</div>

</body>

</html>
```



The flex-grow Property

2. **flex-shrink:** The flex-shrink property specifies how the item will shrink relative to the rest of the flexible items inside the same container.

Eg:

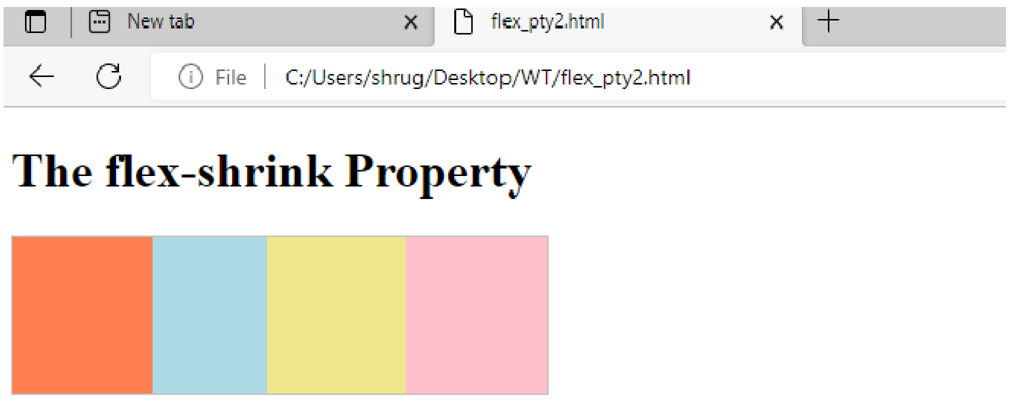
```
div:nth-of-type(2) {
 flex-shrink: 3;
}
```

Here, the second flex-item shrink three times more than the rest..

Changes can be done in the style section.

Demo:

```
<style>

#main {

  width: 350px;

  height: 100px;

  border: 1px solid #c3c3c3;

  display: flex;

}

#main div {

  flex-grow: 1;

  flex-shrink: 1;

  flex-basis: 100px;

}

#main div:nth-of-type(2) {

  flex-shrink: 3;

}

</style>
```



The flex-shrink Property

3. **flex-basis:** The flex-basis property specifies the initial length of a flexible item.

Note: If the element is not a flexible item, the flex-basis property has no effect.

Eg:

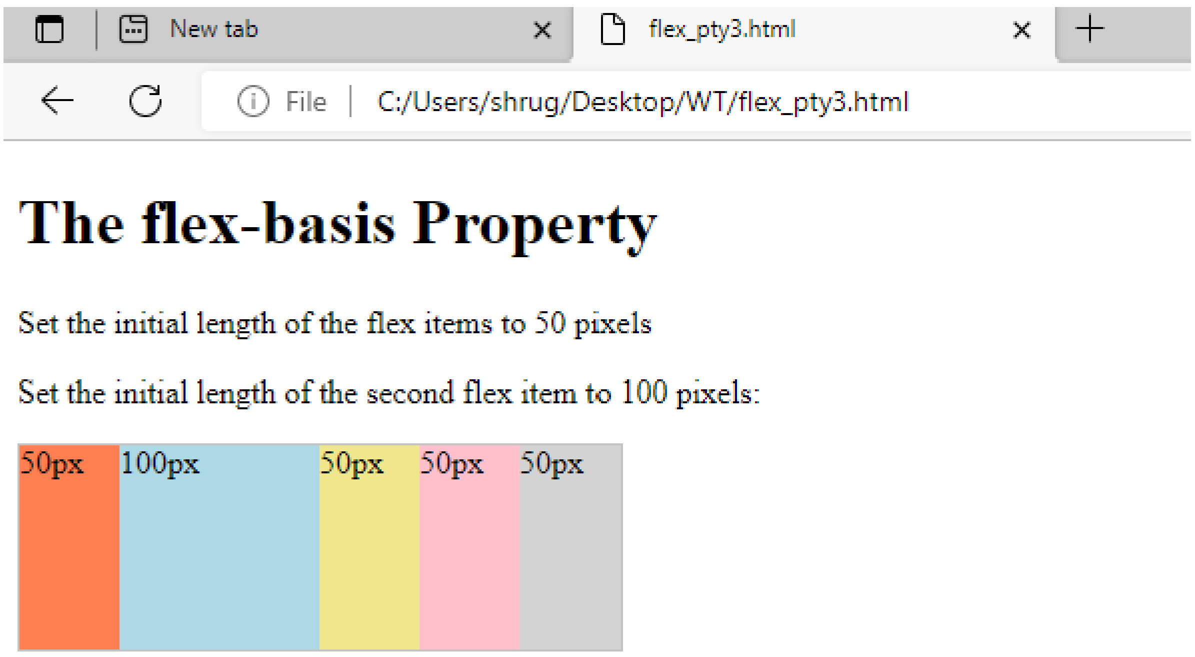
```
div:nth-of-type(2) {
 flex-basis: 100px;
}
```

Set the initial length of the second flex-item to 100 pixels:

Demo:

```
<!DOCTYPE html>

<html>

<head>

<style>

#main {

  width: 300px;

  height: 100px;

  border: 1px solid #c3c3c3;

  display: flex;

}
```

```
#main div {
  flex-grow: 0;
  flex-shrink: 0;
  flex-basis: 50px;
}
#main div:nth-of-type(2) {
  flex-basis: 100px;
}
</style>
</head>
<body>
<h1>The flex-basis Property</h1>
<div id="main">
  <div style="background-color:coral;">50px</div>
  <div style="background-color:lightblue;">100px</div>
  <div style="background-color:khaki;">50px</div>
  <div style="background-color:pink;">50px</div>
  <div style="background-color:lightgrey;">50px</div>
</div>
</body>
</html>
```


New tab
flex_pty3.html
File | C:/Users/shrug/Desktop/WT/flex_pty3.html

The flex-basis Property

Set the initial length of the flex items to 50 pixels

Set the initial length of the second flex item to 100 pixels:

Media Query

CSS3 Introduced Media Queries

Media query is a CSS technique introduced in CSS3.

It uses the @media rule to include a block of CSS properties only if a certain condition is true. Instead of looking for a type of device, they look at the capability of the device.

Media queries can be used to check many things, such as:

- width and height of the viewport
- width and height of the device
- orientation (is the tablet/phone in landscape or portrait mode?)
- resolution

Using media queries are a popular technique for delivering a tailored style sheet to desktops, laptops, tablets, and mobile.

A media query consists of a media type and can contain one or more expressions, which resolve to either true or false.

@media not|only *mediatype* and (*expressions*)

{
 CSS-Code;
}

The result of the query is true if the specified media type matches the type of device the document is being displayed on and all expressions in the media query are true. When a media query is true, the corresponding style sheet or style rules are applied, following the normal cascading rules.

Unless you use the not or only operators, the media type is optional and the all type will be implied.

You can also have different stylesheets for different media:

<link rel="stylesheet" media="*mediatype* and|not|only (*expressions*)" href="*print.css*">

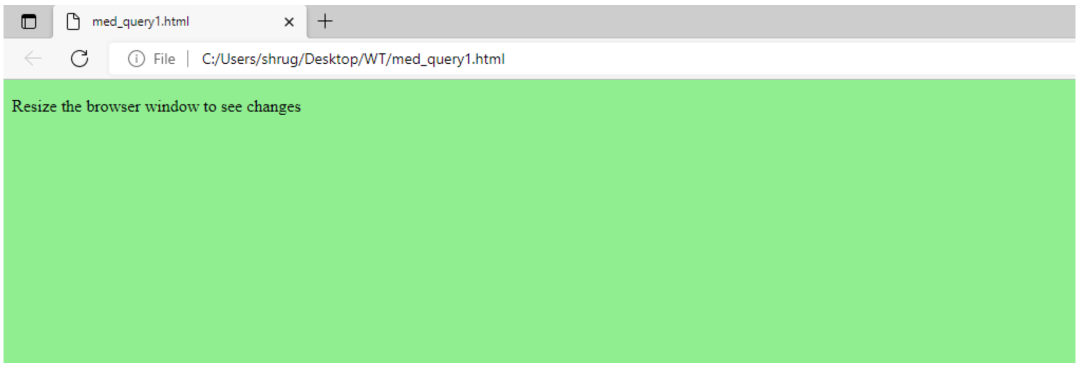
One way to use media queries is to have an alternate CSS section right inside your style sheet.

Another common use of media queries, is **to hide elements** on different screen sizes, **to change the font size** of an element on different screen sizes, change **layout of a page** depending on the orientation of the browser.

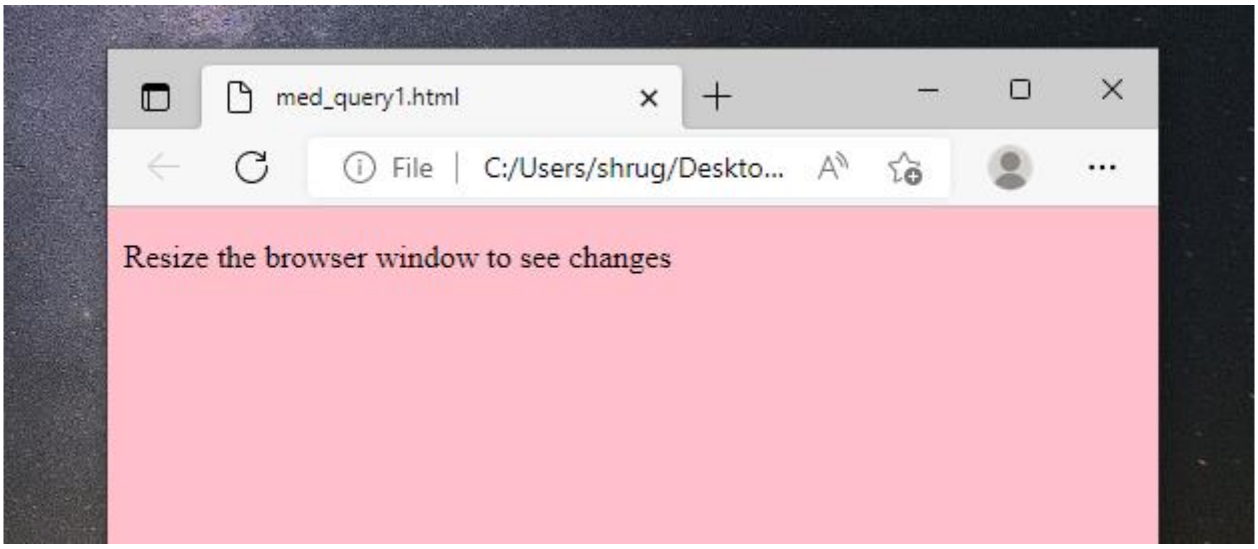
Example1:

```
<!DOCTYPE html>

<html>

<head>

<style>

body {

  background-color: pink;

}

@media screen and (min-width: 480px) {

  body {
```

```
  background-color: lightgreen;

 }

}

</style>

</head>

<body>

<h1>Resize the browser window to see the effect!</h1>

<p>The media query will only apply if the media type is screen and the viewport is 480px wide or wider.</p>

</body>

</html>
```



Resize the browser window to see changes

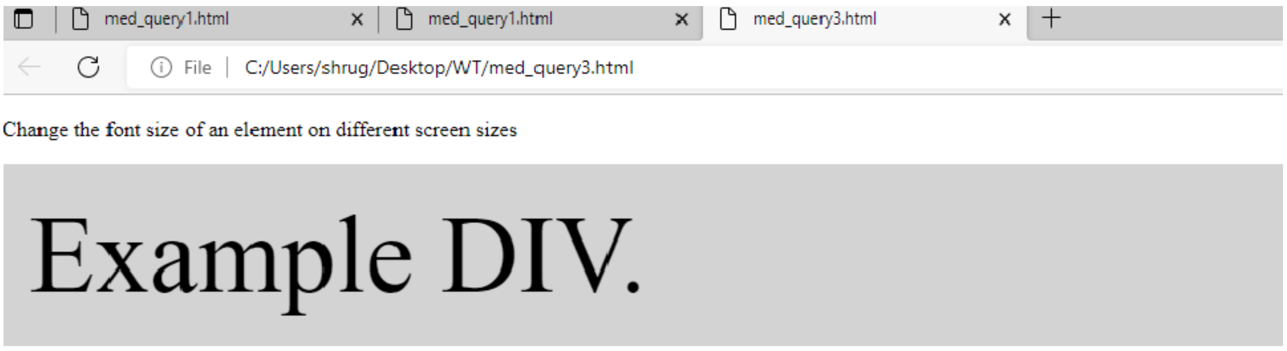


Resize the browser window to see changes

Example2:

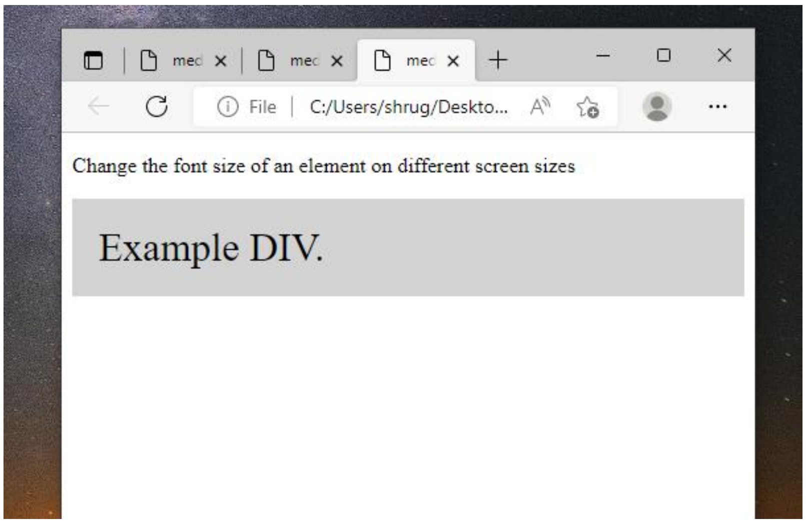
```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
div.example {
  background-color: lightgrey;
  padding: 20px;
}
@media screen and (min-width: 600px) {
  div.example {
    font-size: 80px;
  }
}
@media screen and (max-width: 600px) {
  div.example {
    font-size: 30px;
  }
}
</style>
</head>
<body>
```

```
<h2>Change the font size of an element on different screen sizes</h2>

<div class="example">Example DIV.</div>

<p>When the browser's width is 600px wide or less, set the font-size of DIV to 30px. When it is 601px or wider, set the font-size to 80px. Resize the browser window to see the effect.</p>

</body>

</html>
```



Change the font size of an element on different screen sizes

Example DIV.



Change the font size of an element on different screen sizes

Example DIV.