Flex布局学习笔记

## flex布局-认识flex布局重要概念

|  |
| --- |
|  |

### 1.1认识flex布局

|  |
| --- |
|  |

### 1.2开启flex布局display:flex与inline-flex的区别

flex:开启的flex-container为block元素,

Inline-flex：开的flex-container为inline元素

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <title>Title</title>  </head>  <style>  .box{  display: inline-flex;  width:200px;  height: 100px;  background-color: red;  }  </style>  <body>  <div class="box">  <div></div>  <div></div>  <div></div>  <div></div>  </div>  <strong>hello world</strong>  </body>  </html> |

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <title>Title</title>  </head>  <style>  .box{  /\*display: inline-flex;\*/  display: flex;  width:200px;  height: 100px;  background-color: red;  }  </style>  <body>  <div class="box">  <div></div>  <div></div>  <div></div>  <div></div>  </div>  <strong>hello world</strong>  </body>  </html> |

## flex布局-认识flex的布局模型

|  |
| --- |
|  |

## flex-container属性：flex-direction

|  |
| --- |
|  |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <title>Title</title>  </head>  <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  }  .item{  height: 80px;  width: 100px;  text-align: center;  line-height: 80px;  color:#fff;  }  .item1{  background-color: aquamarine;  }  .item2{  background-color: blueviolet;  }  .item3{  background-color: chartreuse;  }  .item4{  background-color: red;  }  </style>  <body>  <!--  flex-direction:为flex-container的属性,决定了flex布局主轴的方向，共有四个值。  :row(默认值，主轴方向为从左到右)  :row-reverse(主轴方向从右到左)  :column(主轴方向从上到下)  :column(主轴方向从下到上)  -->  <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div>  </div>  </body>  </html> |

### flex-directon:row(默认值)

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: row;  } |

### flex-direction:row-reverse

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: row-reverse; } |

### flex-direction:column

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: column; } |

### flex-direction:column-reverse

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: column-reverse; } |

## flex-container属性：justify-content

|  |
| --- |
|  |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: row;  justify-content: flex-start;  }  .item{  height: 80px;  width: 100px;  text-align: center;  line-height: 80px;  color:#fff;  }  .item1{  background-color: aquamarine;  }  .item2{  background-color: blueviolet;  }  .item3{  background-color: chartreuse;  }  .item4{  background-color: red;  } </style> <body> <!-- flex-direction:为flex-container的属性,决定了flex-items在主轴的布局方向，共有六个值。  :flex-start(默认值，flex-items布局为在主轴方向从左到右,依次紧靠)  :flex-start(flex-items布局为在主轴方向从右向左，依次紧靠)  :center(flex-items布局为主轴的居中布局，依次紧靠)  :space-between(flex-items在主轴上两端(main-start,main-end)对齐，也就是紧靠，然后，各个flex-item间距相等)  :space-evenly(flex-items各间距相同，同时他们的间距也与main-start,main-end相同)  ：spance-around(flex-items各个间距相同，同时他们的间距为main-start间距，main-end间距的两倍) --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div> </div> </body> </html> |

### Justify-content:flex-start(默认值)

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: row;  justify-content: flex-start; } |

### Justify-content:flex-end

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: row;  justify-content: flex-end; } |

### Justify-content:center

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: row;  justify-content: center; } |

### Justify-**content**:space-between

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: row;  justify-content: space-between; } |

### Justify-content:space-evenly

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: row;  justify-content: space-evenly; } |

### Justify-content:space-around

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-direction: row;  justify-content: space-around; } |

## flex-container属性：align-items

|  |
| --- |
|  |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  /\* 对于flex-item没有设置高度，所以flex-item高度自动与flex-container对齐\*/  }  .item{  width: 100px;  color:#fff;  }  .item1{  background-color: aquamarine;  }  .item2{  background-color: blueviolet;  }  .item3{  background-color: chartreuse;  }  .item4{  background-color: red;  } </style> <body> <!-- align-items：为flex-container上面的属性：用来控制flex-items在交错轴上的对齐方式，正好与justify-content相反，共有六值 :normal(默认值，在弹性布局中和stretch效果一样) :stretch(当flex-items在交错轴上的size为auto时,会自动拉伸填充至flex-container) :flex-start(会与交错轴起始位置对齐，也就是cross-start对齐) :flex-end(会与交错轴终点位置对齐，也就是cross-end对齐) :center(居中对齐) :baseline(与基准线对齐) --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div> </div> </body> </html> |

### Normal(默认值)

|  |
| --- |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  align-items: normal;  }  .item{  width: 100px;  color:#fff;  }  .item1{  background-color: aquamarine;  height: 100px;  }  .item2{  background-color: blueviolet;  height: 80px;  }  .item3{  background-color: chartreuse;  height: 60px;  }  .item4{  background-color: #999;  height: 80px;  } </style> <body> <!-- align-items：为flex-container上面的属性：用来控制flex-items在交错轴上的对齐方式，正好与justify-content相反，共有六值 :normal(默认值，在弹性布局中和stretch效果一样) :stretch(当flex-items在交错轴上的size为auto时,会自动拉伸填充至flex-container) :flex-start(会与交错轴起始位置对齐，也就是cross-start对齐) :flex-end(会与交错轴终点位置对齐，也就是cross-end对齐) :center(居中对齐) :baseline(与基准线对齐) --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div> </div> </body> </html> |

### Stretch

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  align-items: stretch; } |

### Flex-start

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  align-items: flex-start; } |

### Flex-end

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  align-items: flex-end; } |

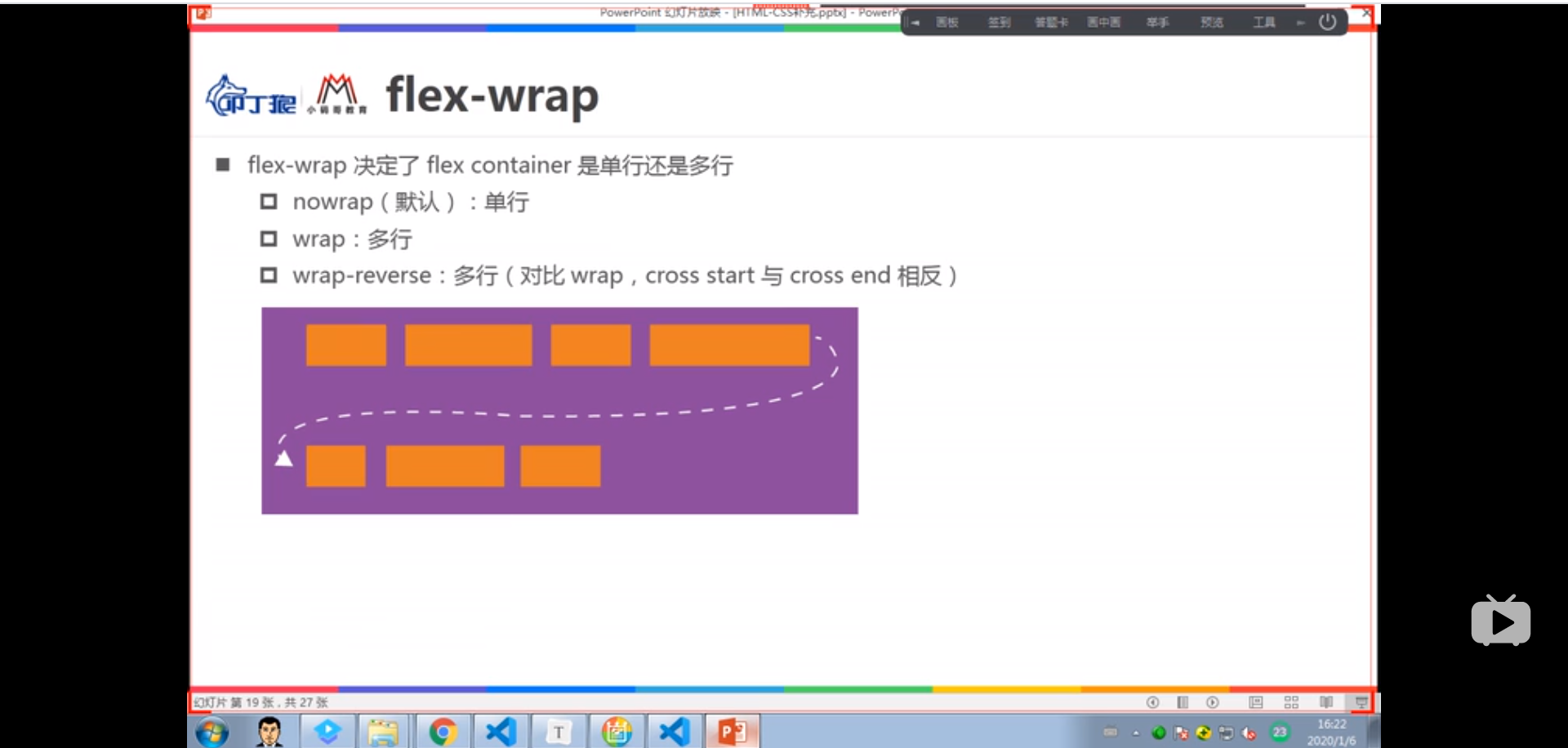
### Center

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  align-items: center; } |

### Baseline

|  |
| --- |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  align-items: baseline;  }  .item{  width: 100px;  color:#fff;  }  .item1{  background-color: aquamarine;  /\*用来调整第一个flex-item的基准线\*/  line-height: 100px;  height: 100px;  }  .item2{  background-color: blueviolet;  height: 80px;  }  .item3{  background-color: chartreuse;  height: 60px;  }  .item4{  background-color: #999;  height: 80px;  } </style> <body> <!-- align-items：为flex-container上面的属性：用来控制flex-items在交错轴上的对齐方式，正好与justify-content相反，共有六值 :normal(默认值，在弹性布局中和stretch效果一样) :stretch(当flex-items在交错轴上的size为auto时,会自动拉伸填充至flex-container) :flex-start(会与交错轴起始位置对齐，也就是cross-start对齐) :flex-end(会与交错轴终点位置对齐，也就是cross-end对齐) :center(居中对齐) :baseline(与基准线对齐) --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div> </div> </body> </html> |

## flex-container属性：flex-wrap



|  |
| --- |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  /\* flex-wrap:默认值为单行,也即是禁止换行，多余的flex-items自动等比例缩放\*/  }  .item{  height: 80px;  width: 100px;  text-align: center;  line-height: 80px;  color:#fff;  }  .item1{  background-color: aquamarine;  }  .item2{  background-color: blueviolet;  }  .item3{  background-color: chartreuse;  }  .item4{  background-color: #999;  } </style> <body> <!-- flex-wrap:为flex-container的属性,决定了flex布局的主轴上的flex-item是否可以自动换行，共有三个值。 :nowrap(默认值,单行，也就是禁止自动换行，也就是flex-container默认是禁止换行的) :wrap(多行，可以自动换行) :wrap-reverse(多行，但是cross-start与cross-end掉头) --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div>  <div class="item item1">item5</div>  <div class="item item2">item6</div>  <div class="item item3">item7</div>  <div class="item item4">item8</div>  <div class="item item1">item9</div> </div> </body> </html> |

### Nowrap(默认)

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-wrap:nowrap; } |

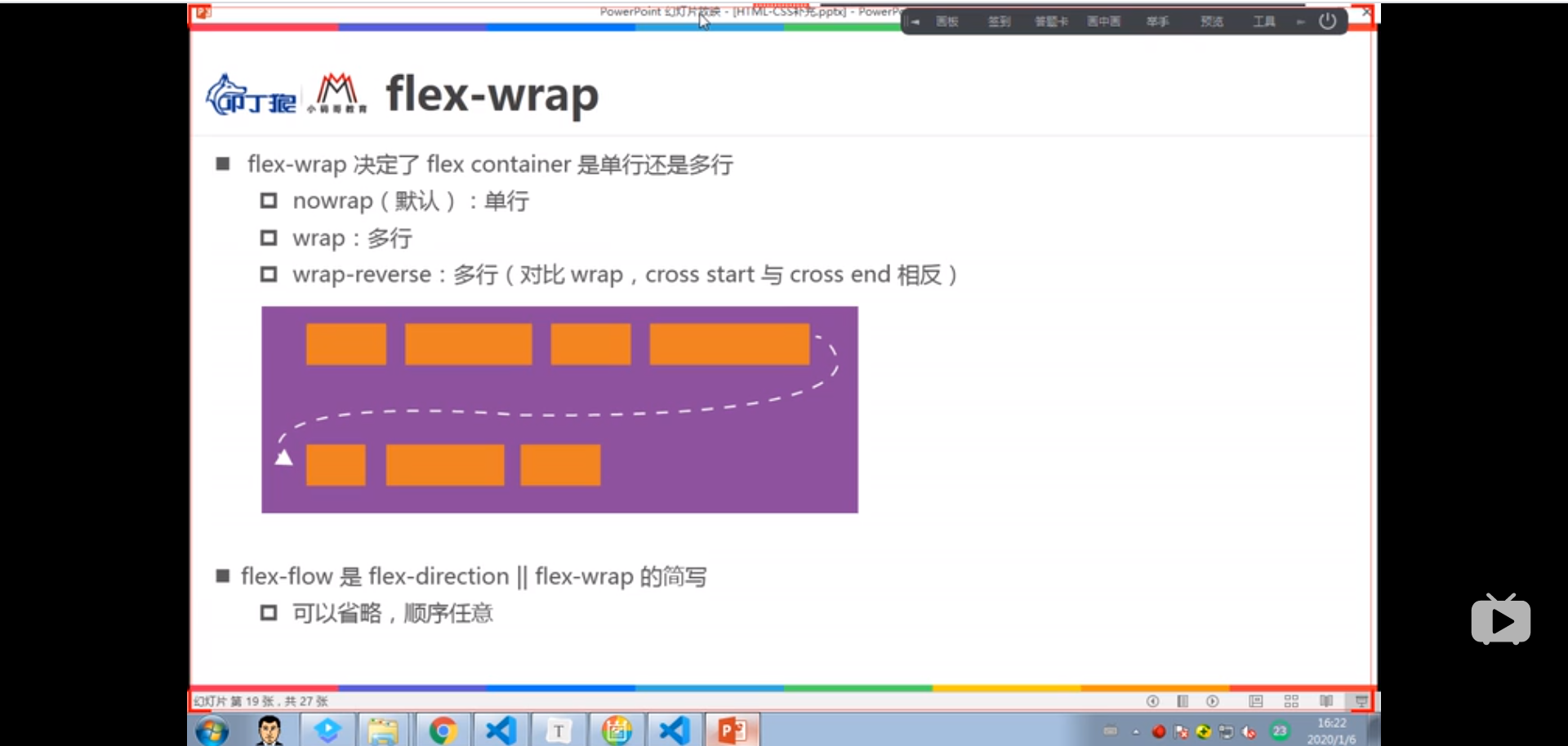
### Wrap

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-wrap:wrap; } |

### Wrap-reverse

|  |
| --- |
| .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-wrap:wrap-reverse; } |

## flex-container属性：flex-flow



|  |
| --- |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  flex-flow:wrap row-reverse;  }  .item{  height: 80px;  width: 100px;  text-align: center;  line-height: 80px;  color:#fff;  }  .item1{  background-color: aquamarine;  }  .item2{  background-color: blueviolet;  }  .item3{  background-color: chartreuse;  }  .item4{  background-color: #999;  } </style> <body> <!-- flex-flow:为flex-direction||flex-wrap的简写形式，其属性值顺序任意组合 --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div>  <div class="item item1">item5</div>  <div class="item item2">item6</div>  <div class="item item3">item7</div>  <div class="item item4">item8</div>  <div class="item item1">item9</div> </div> </body> </html> |

## flex-container属性：align-content

|  |
| --- |
|  |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 650px;  background-color: orangered;  display: flex;  justify-content: space-around;  /\*允许换行\*/  flex-wrap:wrap;  }  .item{  height: 80px;  width: 100px;  text-align: center;  line-height: 80px;  color:#fff;  }  .item1{  background-color: aquamarine;  }  .item2{  background-color: blueviolet;  }  .item3{  background-color: chartreuse;  }  .item4{  background-color: red;  } </style> <body> <!-- align-content:为flex-container的属性，决定了多行的flex-items在cross-axis上的对齐方式，类似于justify-content,但是注意针对的是多行  的情况，单行没有效果。  :stretch(默认值伸缩，也就是自动根据flex-item等比例缩小，类似于align-items)  :flex-start(flex-items布局为在交错轴cross-axis方向上从cross-start到cross-end，依次紧靠，没有均分两行的情况)  :flex-end(flex-items布局为在交错轴cross-axis方向上从cross-end到cross-start，依次紧靠,没有均分情况)  :center(flex-items布局为cross-axis方向上的居中布局，依次紧靠)  :space-between(flex-items在cross-axis上两端(cross-start,cross-end)对齐，也就是紧靠，然后，各个flex-item间距相等)  :space-evenly(flex-items各间距相同，同时他们的间距也与cross-start,cross-end相同)  :spa ce-around(flex-items各个间距相同，同时他们的间距为cross-start间距，cross-end间距的两倍) --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div>  <div class="item item1">item5</div>  <div class="item item2">item6</div>  <div class="item item3">item7</div>  <div class="item item4">item8</div>  <div class="item item1">item9</div> </div> </body> </html> |

### Stretch

|  |
| --- |
| .container{  height: 400px;  width: 650px;  background-color: orangered;  display: flex;  justify-content: space-around;  /\*允许换行\*/  flex-wrap:wrap;  align-content: stretch; } |

### Flex-start

|  |
| --- |
| .container{  height: 400px;  width: 650px;  background-color: orangered;  display: flex;  justify-content: space-around;  /\*允许换行\*/  flex-wrap:wrap;  align-content: flex-start; } |

### Flex-end

|  |
| --- |
| .container{  height: 400px;  width: 650px;  background-color: orangered;  display: flex;  justify-content: space-around;  /\*允许换行\*/  flex-wrap:wrap;  align-content: flex-end; } |

### Center

|  |
| --- |
| .container{  height: 400px;  width: 650px;  background-color: orangered;  display: flex;  justify-content: space-around;  /\*允许换行\*/  flex-wrap:wrap;  align-content: center; } |

### Space-between

|  |
| --- |
| .container{  height: 400px;  width: 650px;  background-color: orangered;  display: flex;  justify-content: space-around;  /\*允许换行\*/  flex-wrap:wrap;  align-content: space-between; } |

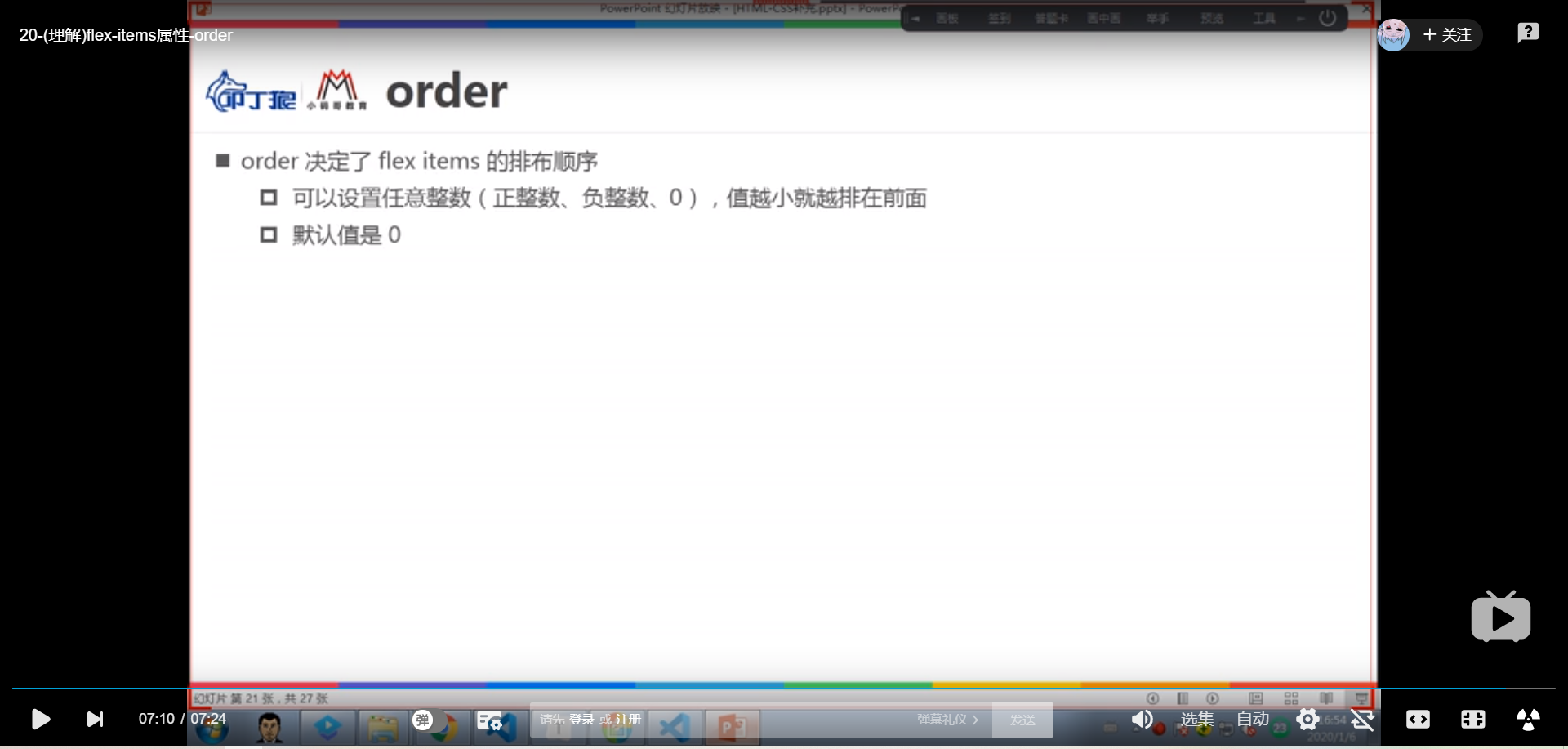
### Space-evenly

|  |
| --- |
| .container{  height: 400px;  width: 650px;  background-color: orangered;  display: flex;  justify-content: space-around;  /\*允许换行\*/  flex-wrap:wrap;  align-content: space-evenly; } |

### Space-around

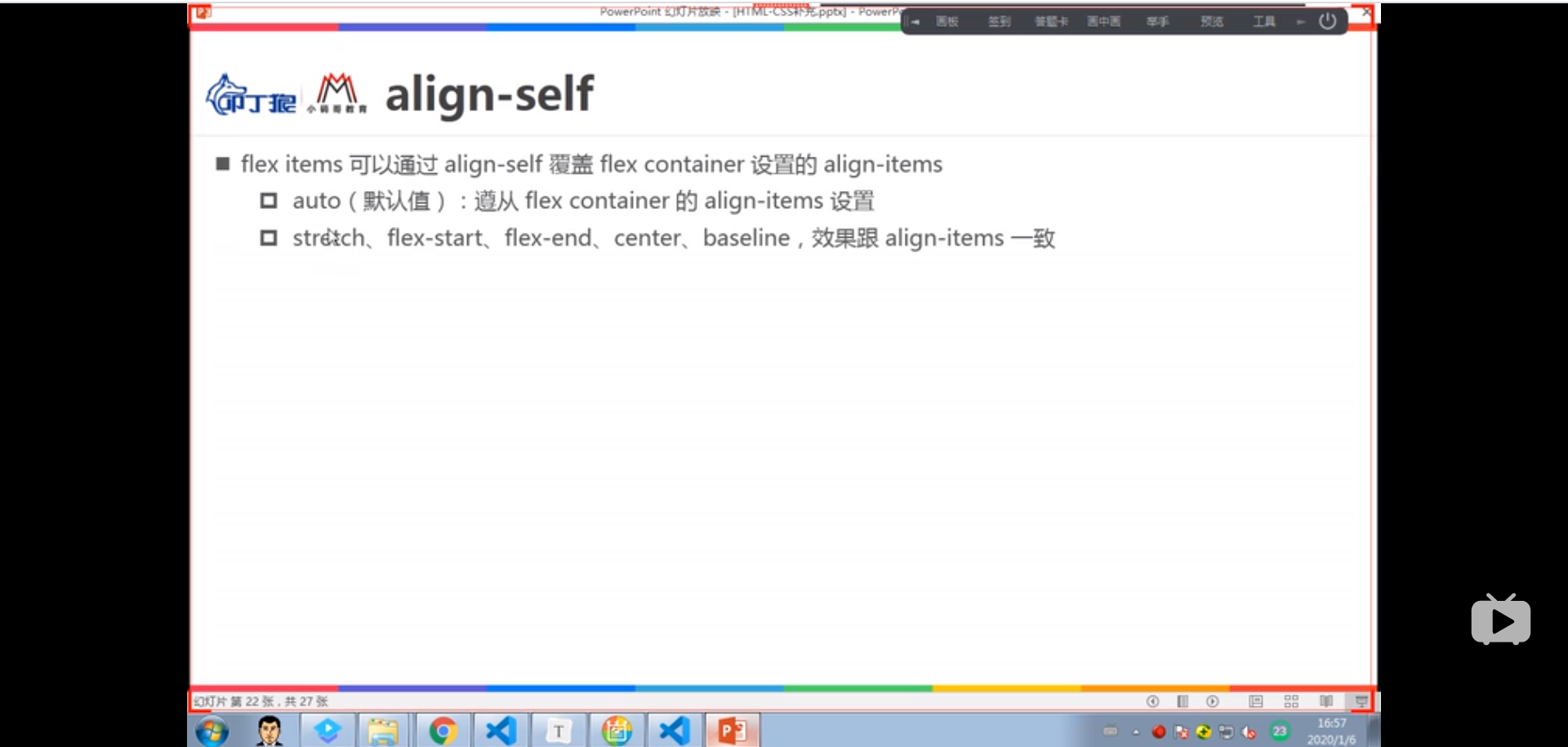
|  |
| --- |
| .container{  height: 400px;  width: 650px;  background-color: orangered;  display: flex;  justify-content: space-around;  /\*允许换行\*/  flex-wrap:wrap;  align-content: space-around; } |

## flex-items属性：order



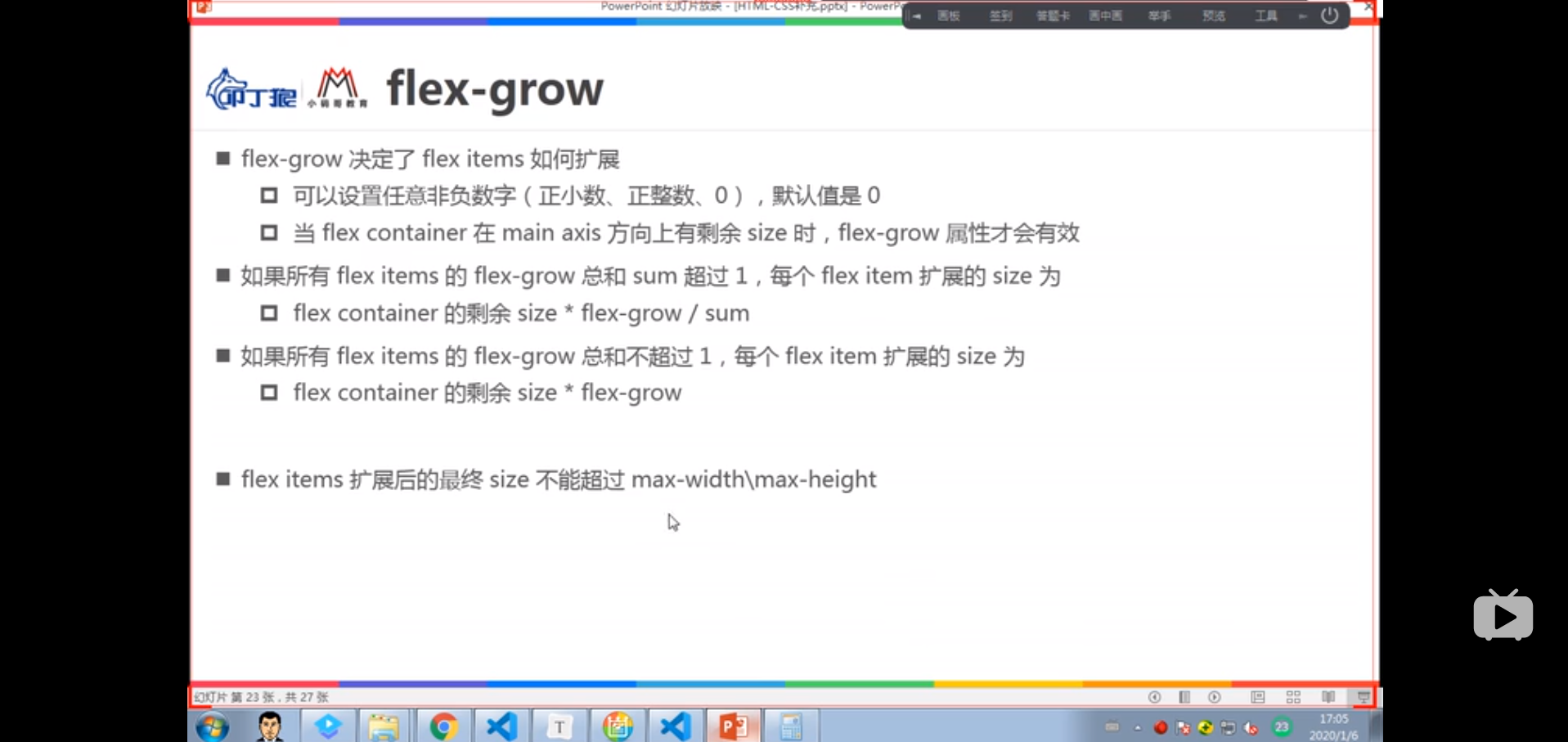
|  |
| --- |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  }  .item{  height: 80px;  width: 100px;  text-align: center;  line-height: 80px;  color:#fff;  }  .item1{  background-color: aquamarine;  order:50;  }  .item2{  background-color: blueviolet;  order: 20;  }  .item3{  background-color: chartreuse;  order:30;  }  .item4{  background-color: #999;  order:25;  } </style> <body> <!-- order:为flex-items上面的属性，用来决定flex-item的排列顺序，其值可以为正数，0，负数；默认为0；其值越小 ，越靠前。 --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div> </div> </body> </html> |

## flex-items属性：align-self



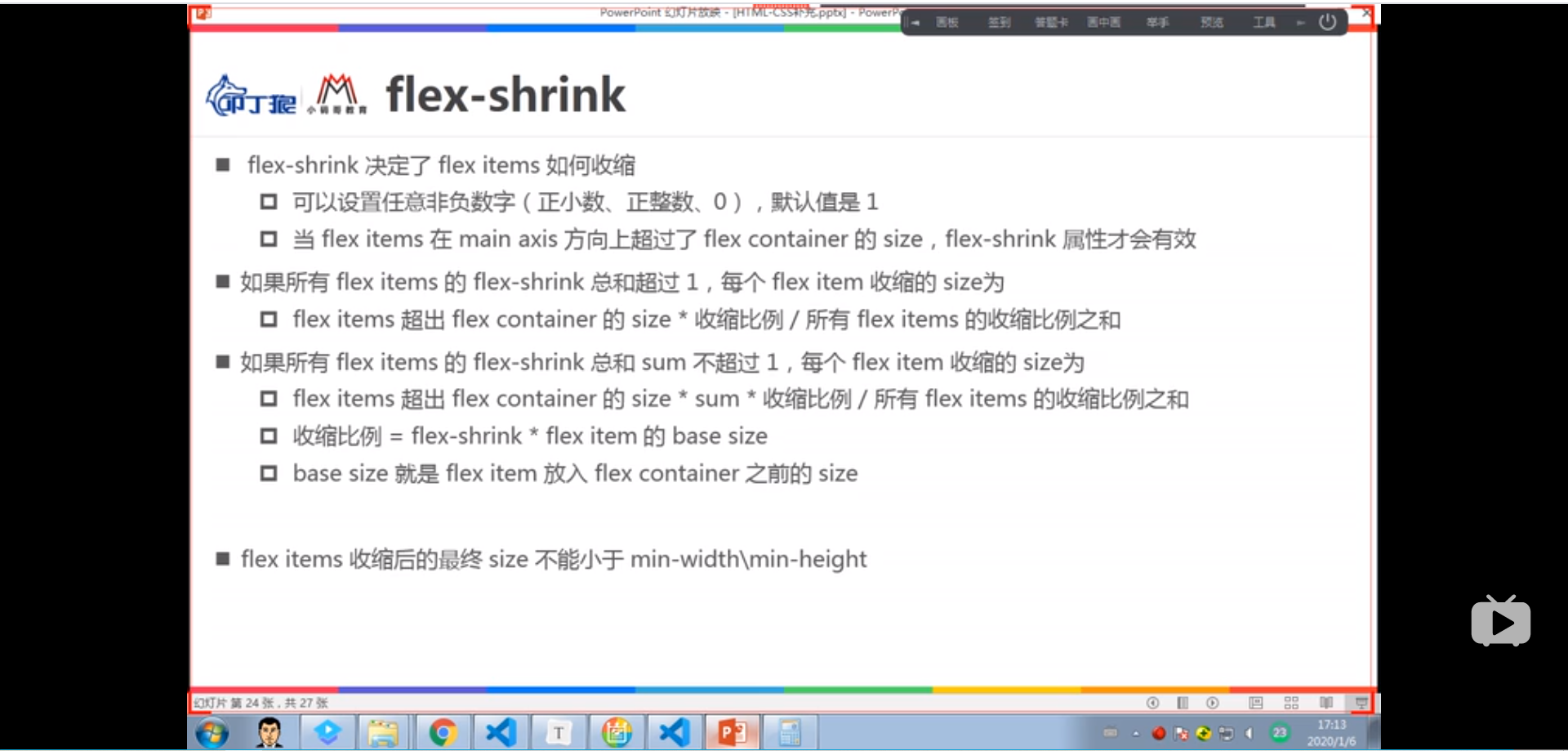
|  |
| --- |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  /\*在flex-container上面设置cross-axis为居中对齐\*/  align-items: center;  }  .item{  width: 100px;  color:#fff;  }  .item1{  background-color: aquamarine;  height: 100px;  }  .item2{  background-color: blueviolet;  height: 80px;  /\* 单独给item2设置为cross-axis为flex-end\*/  align-self: flex-end;  }  .item3{  background-color: chartreuse;  height: 60px;  }  .item4{  background-color: #999;  height: 80px;  } </style> <body> <!-- align-self：为flex-items上面的属性，用来覆盖flex-container上面设置的align-items属性值，也就是通过该属性可以给flex-items 单独设置一个cross-axis布局。共有六值，其值均会覆盖flex-container上面设置的值 :normal(默认值，在弹性布局中和stretch效果一样) :stretch(当flex-items在交错轴上的size为auto时,会自动拉伸填充至flex-container) :flex-start(会与交错轴起始位置对齐，也就是cross-start对齐) :flex-end(会与交错轴终点位置对齐，也就是cross-end对齐) :center(居中对齐) :baseline(与基准线对齐) --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div> </div> </body> </html> |

## flex-items属性：flex-grow

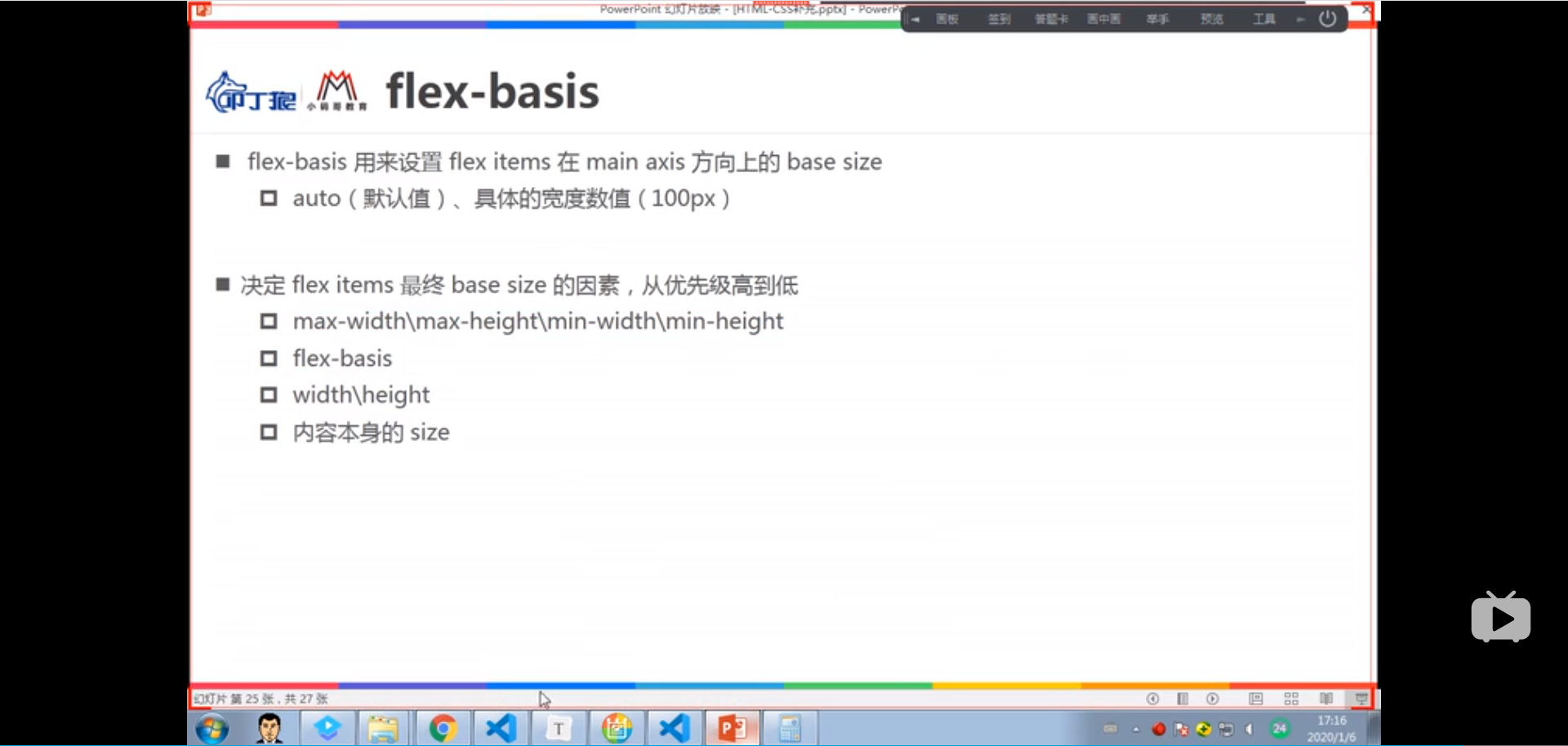


|  |
| --- |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  }  .item{  width: 100px;  color:#fff;  }  .item1{  background-color: aquamarine;  height: 100px;  }  .item2{  background-color: blueviolet;  height: 80px;  }  .item3{  background-color: chartreuse;  height: 60px;  }  .item4{  background-color: #999;  height: 80px;  } </style> <body> <!-- flex-grow:为flex-items上面的属性，一般用作为flex-items在主轴方向上的size的扩展，其值 可以为任意正数或者0，也就是正整数，正小数，0， 注意：1，如果所有的fLex-item上面的flex-grow的值小于1，则各个对应的flex-item拓展的size按照 flex-container在主轴上的剩余size\*flex-item上面的flex-grow的值来计算,此时不一定能填满主轴的size  2.如果所有的flex-item上面的flex-grow的值大于1，则各个对应的flex-item扩展的size按照 flex-container在主轴上的剩余size\*flex-item上面的flex-grow的值/(各个flex-item上面的flex-grow的值的和)来计算，此时各个 flex-item一定能填满整个主轴的size flex-item的扩容一定不能超过max-width/max-height --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div> </div> </body> </html> |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  }  .item{  width: 100px;  color:#fff;  }  .item1{  background-color: aquamarine;  height: 100px;  flex-grow: .2;  }  .item2{  background-color: blueviolet;  height: 80px;  flex-grow: .1;  }  .item3{  background-color: chartreuse;  height: 60px;  flex-grow: .3;  }  .item4{  background-color: #999;  height: 80px;  flex-grow: .2;  } </style> <body> <!-- flex-grow:为flex-items上面的属性，一般用作为flex-items在主轴方向上的size的扩展，其值 可以为任意正数或者0，也就是正整数，正小数，0， 注意：1，如果所有的fLex-item上面的flex-grow的值小于1，则各个对应的flex-item拓展的size按照 flex-container在主轴上的剩余size\*flex-item上面的flex-grow的值来计算,此时不一定能填满主轴的size  2.如果所有的flex-item上面的flex-grow的值大于1，则各个对应的flex-item扩展的size按照 flex-container在主轴上的剩余size\*flex-item上面的flex-grow的值/(各个flex-item上面的flex-grow的值的和)来计算，此时各个 flex-item一定能填满整个主轴的size flex-item的扩容一定不能超过max-width/max-height --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div> </div> </body> </html> |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <style>  .container{  height: 400px;  width: 600px;  background-color: orangered;  display: flex;  }  .item{  width: 100px;  color:#fff;  }  .item1{  background-color: aquamarine;  height: 100px;  flex-grow: 2;  }  .item2{  background-color: blueviolet;  height: 80px;  flex-grow: 1;  }  .item3{  background-color: chartreuse;  height: 60px;  flex-grow: 3;  }  .item4{  background-color: #999;  height: 80px;  flex-grow: 2;  } </style> <body> <!-- flex-grow:为flex-items上面的属性，一般用作为flex-items在主轴方向上的size的扩展，其值 可以为任意正数或者0，也就是正整数，正小数，0， 注意：1，如果所有的fLex-item上面的flex-grow的值小于1，则各个对应的flex-item拓展的size按照 flex-container在主轴上的剩余size\*flex-item上面的flex-grow的值来计算,此时不一定能填满主轴的size  2.如果所有的flex-item上面的flex-grow的值大于1，则各个对应的flex-item扩展的size按照 flex-container在主轴上的剩余size\*flex-item上面的flex-grow的值/(各个flex-item上面的flex-grow的值的和)来计算，此时各个 flex-item一定能填满整个主轴的size flex-item的扩容一定不能超过max-width/max-height --> <div class="container">  <div class="item item1">item1</div>  <div class="item item2">item2</div>  <div class="item item3">item3</div>  <div class="item item4">item4</div> </div> </body> </html> |

## flex-items属性：flex-shrink



## flex-items属性：flex-basis



## flex-items属性：flex

