How to realize different layouts using CSS

1. **Horizontally centered 水平居中**
2. For text/inline elements/inline-block elements

#parent {

text-align: center

}

\*\*\*perfect compatibility; child must have smaller width than parent

1. For single block element

#child {

width: 100px;

margin: 0 auto;

}

\*\*\* perfect compatibility; the width of child must be assigned and less than the width of parent.

1. Multiple block elements

#parent {

text-align: center

}

#child {

display: inline-block

}

\*\*\* perfect compatibility; when convert block to inline-block, spaces and \n will cause gaps between elements

1. Using absolute positioning

#parent {

width: 200px;

height: 200px;

position: relative;

background-color: red;

}

#child {

width: 100px;

height: 100px;

position: absolute;

left: 50%; // 50% width of the parent

transform: translateX(-50%); // 50% width of the child

// margin-left: -50px;

background: #fff;

}

\*\*\* using margin-left has better compatibility, but have to know the width of the child; using transform need to consider compatibility (IE 9+)

1. Random number of elements —— using flex

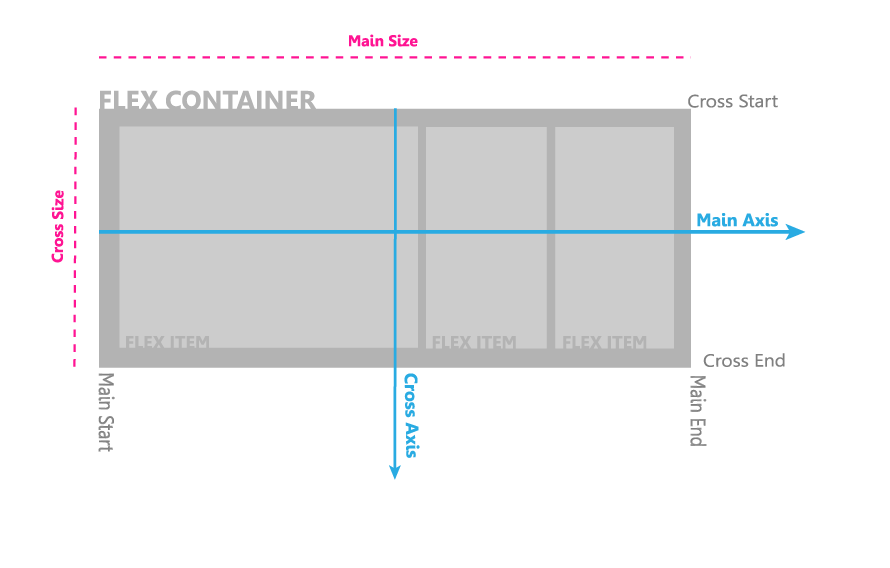
#parent {

display: flex;

justify-content: center;

}

\*\*\* poor compatibility on pc platform; good for mobile devices (Android 4.0+)



1. **Vertical centering**
2. for **one-line** text/ inline element/ inline-block

#parent {

height: 150px;

line-height: 150px;

}

\*\*\* perfect compatibility; need to know the height

1. for multi-line text/ inline/ inline-block

#parent {

height: 150px;

line-height: 30px; // elements display in 5 lines on the page

}

\*\*\* or using <span> to wrap the content and set display: inline-block, then dealing it like an image

\*\*\* need to know how many lines will display on the page

1. image (inline element)

#parent {

height: 150px;

line-height: 150px;

font-size: 0;

}

img#child {

vertical-align: middle;

}

\*\*\* need to assign font-size: 0 to realize precise vertical centering. Vertical-align以触发line-height的空格节点的中线为居中线，而此中心线相对绝对居中线较下沉。font-size:0 可消除这一差距。

1. single block element

<1>using table-cell

#parent {display: table-cell; vertical-align: middle}

\*\*\* unable to set width and height of a table-cell element unless its parent has attribute “display: table”;

<2> absolute positioning

#parent {

height: 150px;

position: relative;

}

#child {

position: absolute;

top: 50%;

transform: translate(-50%); // margin-top: -25px;

height: 50px;

}

or

#parent { position: relative; }

#child {

position: absolute;

margin: auto 0;

top: 0;

bottom: 0;

height: 50px;

}

<3> flex

#parent {

display: flex;

align-items: center;

}

or

#parent {display: flex;}

#child {align-self: center;}

or

#parent {display: flex;}

#child {margin: auto 0;}

1. multiple block elements

using flex:

#parent {

display: flex;

align-item: center;

}

or

#parent {display: flex;}

#child {align-self: center;}

or

#parent {

display: flex;

flex-direction: column;

justify-content: center;

}

1. **horizontally & vertically centering 水平垂直居中**
2. inline/ inline-block/ image

#parent {

height: 150px;

line-height: 150px;

text-align: center;

font-size: 0;

}

#child {

/\*display: inline-block\*/ // convert block element

vertical-align: middle;

}

1. table-cell

#parent {

height: 150px;

width: 200px;

display: table-cell;

vertical-align: middle;

/\*text-align: center\*/ // if for inline element

}

#child {

/\*margin: 0 auto\*/ // if for block element

width: 100px;

height: 50px;

}

\*\*\* good for situations that width and height are unknow, good compatibility(ie 8+); table-cell设置百分比宽高仅在父元素设置display:table 时才有效；table-cell不感知margin, 父元素设置table-row也会使其不感知height；设置float或position会对默认布局造成破坏，可为其增加一个父div定义float属性； 内容溢出时自动撑开元素。

1. button as parent

button#parent {

height: 150px;

width: 200px;

outline: none;

border: none;

}

#child { display: inline-block } // button itself already has text-align: center attribute

\*\*\* in IE browser, click button will appear depressed affect

1. absolute positioning

#parent { position: relative; }

#child {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%,-50%);

}

or

#parent { position: relative; }

#child {

position: absolute;

margin: auto;

width: 100px;

height: 100px;

left: 0;

right: 0;

top: 0;

bottom: 0;

}

1. flex methods

#parent {display: flex;}

#child { margin: auto; }

or

#parent {

display: flex;

justify-content: center;

align-items: center;

}

or

#parent {

display: flex;

justify-content: center;

}

#child { align-self: center; }

1. double columns layout
2. left width fixed, right width self-adaptive

<1> float + margin

html:

<body>

<div id=”left”></div>

<id id=”right”></div>

</body>

css:

#left {

float: left;

width: 100px;

height: 500px;

}

#right {

height: 500px;

margin-left: 100px;

}

<2>float + margin(fix)

html:

<body>

<div id=”left”></div>

<id id=”right-fix”>

<div id=”right”></div>

</div>

</body>

css:

#left {

float: left;

width: 100px;

height: 500px;

}

#right-fix {

float: right;

width: 100%;

margin-left: -100px; // absolute value must >= 100 to display in on line

}

#right {

margin-left: 100px;

height: 500px;

}

<3>float + overflow

html:

<body>

<div id=”left”></div>

<id id=”right”></div>

</body>

css:

#left {

float: left;

width: 100px;

height: 500px;

}

#right {

height: 500px;

overflow: hidden

}

1. .
2. .
3. .