LAYOUT

Terms

Absolute positioning
Absolute units
Box model
Breakpoints
Collapsing parent
Fixed positioning
FlexBox (Flexible box layout)
Floating elements
Grid layout
Margin collapsing
Media queries
Mobile-first approach
Overflowing
Relative positioning
Relative units
Responsive web design

Summary

- When rendering an HTML document, the browser puts each element inside a box. The box contains four areas: the content area, the padding area, the border area and the margin area.

- Padding is the space between the border and the content area. Margin is the space outside of an element and should be used to separate elements from each other.

- Margin collapsing happens when the top and bottom margins of elements are combined into a single margin. The size of the margin is equal to the largest of the two margins.

- There are two types of HTML elements: block-level and inline.
- Block-level elements always start on a new line and take up the entire available horizontal space. The `<p>` and `<div>` elements are examples of block-level elements.
- Inline elements don't start on a new line. They take up as much width as necessary. The `<span>`, `<a>` and `<img>` are a few examples of inline elements.
- We can size elements by setting their `width` and `height` properties. These properties have no effect on inline elements. To size an inline element, we need to set its `display` property to `inline-block`.
- By default, the width and height properties are applied to the content box. So paddings and borders increase the size of the visible box. This behavior can be changed by setting the `box-sizing` property to `border-box`.
- Overflow occurs when an element's content is too large to fit. Using the `overflow` property we can specify what should happen when overflow occurs.
- Measurement units in CSS fall into two categories: absolute and relative units. Examples of absolute units are px, pt, in, cm, etc. Examples of relative units are %, vw, vh, em and rem.
- Using the `position` property we can precisely position an element. The default value of this property is `static`. If we change the value of this property, the element is considered *positioned*.
- By setting the `position` to `relative`, we can position an element relative to its normal position. By setting it to `absolute`, we can position it relative to its positioned parent. That means, the parent (or ancestor) should be a positioned element. By setting the position to `fixed`, we can position the element relative to the viewport.
- By setting the `float` property, we can push an element to the left or right side of its container. Other elements will flow around the floated element and fill the available space.

- Floated elements are invisible to their parent. This behavior is called collapsing parent and often causes layout issues. To fix this, we have to clear the floated elements.

- The Flexible Box Layout (or FlexBox or just Flex) is used for laying out elements in one direction (in a row or column). A common application of Flex is in building navigation menus.

- The Grid Layout is a two-dimensional grid system. It's often used to lay out major page areas, photo galleries, etc.

- With media queries we can provide different styles for different devices depending on their features such as screen size, orientation, etc. The most common application of media queries is in providing different styles based on the viewport width.

- By using media queries and relative measurement units we can build responsive web sites that adjust smoothly to various screen sizes.

CSS Cheat Sheet

Box Model

```
padding: 10px 20px;
padding-top: 30px;
margin: 1px 2px 3px 4px;
margin-top: 5px;
border: 1px solid black;
border-top: 1px solid black;
```

Sizing Elements

```
width: 5rem;
height: 20%;
box-sizing: border-box;
```
To prevent paddings/borders from increasing the size of the visible box.

Overflowing

`overflow: hidden;`	Hides the overflown content
`overflow: scroll;`	Always shows scroll bars
`overflow: auto;`	Shows scroll bars only if content overflows

Positioning

`position: static;`	The default value
`position: relative;`	To position relative to the element's normal position
`position: absolute;`	To position relative to the element's positioned parent
`position: fixed;`	To position relative to the viewport
`z-index: 1;`	To change the stacking order of an element

Floating

```
float: left;
float: right;
clear: both;
```

FlexBox

Container properties

`display: flex;`	To enable the flex layout on a container
`flex-direction: column;`	Direction (row, column)
`justify-content: center;`	To align items along the main axis
`align-items: center;`	To align items along the cross axis
`flex-wrap: wrap;`	To enable wrapping
`align-content: center;`	To align flex lines along the cross axis

Item properties

`align-self: center;`	To overwrite the alignment
`flex-basis: 10rem;`	The initial size of an item
`flex-grow: 1;`	The growth factor
`flex-shrink: 0;`	The shrink factor
`flex: 0 1 10rem;`	Shorthand (grow shrink basis)

Grid

Defining a grid

```
display: grid;
grid-template-rows: repeat(3, 100px);
grid-template-columns: repeat(2, 100px);
grid-template: repeat(3, 100px) / repeat(2, 100px);
grid-template-areas:
   “header   header”
   “sidebar  main”
   “footer   footer”;
```

Gaps

```
row-gap: 10px;
column-gap: 20px;
gap: 10px 20px;
```

Shorthand (row column)

Alignment

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

Align the items horizontally within their cell

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

Align the items vertically within their cell

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

Align the grid horizontally within its container

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

Align the grid vertically within its container

Placing items

```
grid-column: 2;
grid-column: 1 / 3;
grid-column: 1 / -1;
grid-column: 1 / span 2;

grid-row: 2 / 4;

grid-area: header;
```

Hiding elements

`display: none;`	Hides the element
`visibility: hidden;`	Hides the element but keeps the reserved space

Media queries

```
@media screen and (min-width: 500px) {
}

@media screen and (min-width: 500px) and (max-width: 700px) {
}

@media print {
}
```