What is padding and margin and when do you use them?

What is padding?

Padding is the space between the content and the border of an element. Padding is valuable in making additional space inside an element, keeping it at a set distance from other aspects of a website. Using padding is extremely beneficial when you need to separate text boxes and images while also keeping them aligned.

What is the margin?

Margin is the space around the border of an element. The margin surrounding an element will inform the web browser being used of how much space should be left between independent elements and the external margin of the website's page. Margins can also be used to keep different elements an equal distance apart.

When to use padding vs. margin

It's essential to know when to use padding vs. margin as you build your website. Once you're familiar with the differences between margin and padding, it'll be easier to determine when it's the right time to implement margins on a page and when you should use padding.

Before you begin your website development project, here are a few things about padding and margins to keep in mind.

When to use padding:

- **Change the size of an element. If you want to expand the space around an element, you can add or increase the padding surrounding it. This can be useful when working with interactive items, such as buttons or image-based links.**

- Add space between borders and content. Using padding to add space between content and its corresponding border is one way to ensure the design aligns with other on-page elements. Doing so can help you increase the whitespace of your graphic or website, which is fundamental in web design.

When to use margins:

- Adjust an element's positioning. One of the most common reasons to use margins when designing and developing a site is to change a specific element's position. Using margins can help you move an element based on whether you prefer it to be centered on your page or positioned to the right or left. You can also choose if the element is fixed and will scroll along the page or if it should remain in one place as a user scrolls.
- Overlap elements. If you want to overlap specific elements with one another, you can do so using margins. Using a negative margin value is one of the quickest ways to allow elements to overlap with one another.
- Setting distance. Setting the distance between elements is much easier once you're familiar with margins and how they work. Incorporating the right amount of whitespace can mean the difference between building an attractive website thriving with traffic and turning prospective customers away.

What is display property and explain display inline, block, and inline block?

The `display` property specifics the display behavior of an element. It’s a must know for controlling the layout of an element.

There’s a bunch of property values. But let’s start off with the essential `inline`.

This one displays the element inline or on the same line. In other words, inline elements do NOT start on a new line and only takes up as much width as its content. So, if you try to set any width and height, it will have NO effects.

Alright, let's move on to `inline-block`. It’s essentially the same thing as `inline`, except that you can set height and width values.

So far, we talked about `inline`. Now let's switched to the opposite of it, `block`. Remember `inline` elements appears on the same line. Well, `block` starts on a NEW line and takes up the full width available. So that means block elements will occupy the entire width of its parent element.

Explain min height, min width, max height, and max width in CSS?

Minimum and maximum width/height

So far, we have seen how we can give an element relative and absolute sizes, by using the width and height properties. However, another possibility exist: Specifying minimum and/or maximum sizes. For this purpose, four properties exist: min-height, max-height, min-width and max-width. Their purpose should be easy to defer from their names, but how and when to use them might not be equally clear.

Specifying minimum and/or maximum sizes for an element allows you to take advantage of the fluid nature of elements on a webpage, allowing a certain element to expand and subtract within a limited set of dimensions, instead of defining a constant and absolute size like we did with the *height* and *width* properties. Just like width and height, the min-* and max-* properties allows you to specify an absolute or relative length unit or a percentage based size.

Minimum width and height

Using the min-width and min-height properties, you can define the smallest possible size that an element can have. If the element doesn't have width and/or height defined (auto), or if these values are smaller than the defined minimum height and width, the minimum height and width values will overrule it. This can easily be illustrated by an example:

```
<style type="text/css">
.box {
      width: 50px;
      height: 50px;
      background-color: DarkSeaGreen;
      padding: 10px;
      margin: 20px;
      float: left;
}
</style>

<div class="box">
      Box 1 - Default
</div>

<div class="box" style="min-height: 80px; min-width: 80px;">
      Box 2 - Minimum height and width
</div>

<div class="box" style="min-height: 80px; min-width: 80px;">
      Box 3 - Minimum height and width. Only expands to 80px.
</div>
```

```
<div class="box" style="min-height: 80px; min-width: 80px; height: auto;">
      Box 4 - Minimum height and width. Only expands to 80px, unless no
max-height/height have been specified (auto)
</div>
```

Maximum width and height

Sometimes you want to restrain an element to a specific width and/or height. Obviously this can be done with the width and height properties, but this will force the element to stay at this size all of the time - no matter if the content requires it or not. By using the max-width and/or max-height properties instead, you can allow an element to be whatever size it requires to fit the content and then grow along with the content up until a certain point.

Here's an example where we use max-height and max-width:

```
<style type="text/css">
.box {
      background-color: DarkSeaGreen;
      padding: 10px;
      margin: 20px;
      float: left;
}
</style>

<div class="box">
      Box 1 - Default
</div>

<div class="box" style="max-height: 100px; max-width: 100px;">
      Box 2 - Maximum height and width
</div>

<div class="box" style="max-height: 100px; max-width: 100px;">
      Box 3 - Maximum height and width, this time with more content
</div>
```