CSS

1. What are the main differences between external, internal, and inline CSS?

The main differences are as follows:

- Inline CSS: In this styling method, the CSS styles are applied in the HTML tag itself using the style parameter. This method makes the tag look very ugly and makes it difficult to find errors. e.g.

```
<div class=”container”>
    <h1 style=”color:blue;font-size:60px;”>My Title</h1>
</div>
```

- Internal CSS: In this styling method, the CSS styles are applied in the same HTML file. Each HTML tag is designed separately in the same file. e.g.

```
<style>
    .myElement {
        property: parameter;
    }
</style>
```

- External CSS: In this method, the CSS is applied to each element in a different CSS file. It is the best method for styling because you can find the CSS for all the elements in one particular file and makes debugging easy. e.g.

```
<head>
 ...
    <link rel=”stylesheet” type=”text/css” href=”./css/style.css”>
 ...
</head>
```

2. What is the syntax for class and ID selectors?

The syntax for a class and an ID selector are as follow:

```
/* A class selector */
.myClass {
    property: parameter;
}
/* An ID selector */
#myID {
    property: parameter;
}
```

3. How would you apply a single rule to two different selectors?

You can group a single rule to multiple selectors using the *“grouping selectors”*. Here are some examples of how to use it.

```
/* Two html elements */
h1, h2 {
  text-transform: uppercase;
  font-weight: 700;
  family-font: “Cabin”, sans-serif;
}
```

```
/* An html element and an ID */
h1, #myID {
  color: green;
}
```

```
/* You can list the styles on individual lines for clarify */
h1,
h2,
h3,
h4 {
  family-font: “Cabin”, sans-serif;
}
```

4. Given an element that has an id of title and a class of primary, how would you use both attributes for a single rule?

For this case I would use the ***“chaining selectors”***, since the element contains more than 2 selectors, and I would do it as follows:

```
<!-- An html element with an ID and a class -->
<h1 id=”title” class=”primary”>My title</h1>
```

```
/* An html element with an ID and a class */
h1#title.primary {
  color: green;
  font-weight: 500;
}
```

5. What does the descendant combinator do?

A descendant combinator causes the elements which match with the last selector to be selected. Typically this is represented by a space between the selectors. E.g.

```
/*
All p tags that are descendants of travels section, are going to display
the Merriweather font
*/
section.travels p {
   font-family: “Merriweather”, serif;
}
```

6. Between a rule that uses one class selector and a rule that uses three type selectors, which rule has the higher specificity?

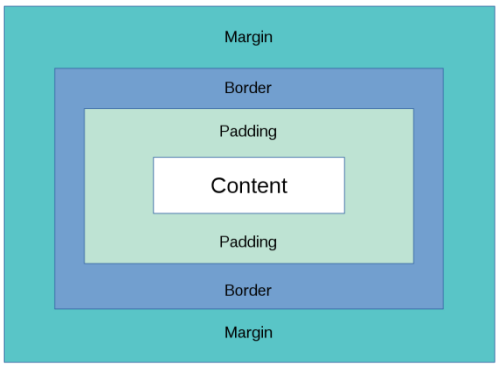
In this case it would be the rule that uses a ***class selector***, since it is more specific and is over the selector types. Specificity will only be taken into account when an element has several conflicting declarations targeting it, and these should be in the following order:

I) IDs selectors
II) Classes selectors
III) Types selectors
IV) Anything less specific than a type selector.

7. From inside to outside, what is the order of box-model properties?

The Box model contains the following properties:

- **Content:** The area where you put your content. The size of this box is defined by its *height* and *width* properties.
- **Padding:** The padding adds the defined amount of whitespace around the content. The amount is controlled by the *padding* property.
- **Border:** The border wraps the content and padding. It is controlled by the *border* property.
- **Margin:** The margin is the whitespace between a box and the boxes around it. It is the outermost layer and is controlled by the *margin* property.



8. What does the box-sizing CSS property do?

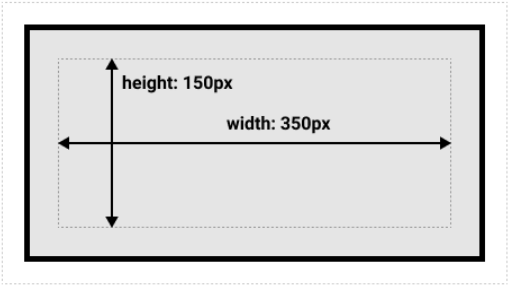
The box-sizing property allows you to include the padding and border in an element's total width and height.

9. What is the difference between the standard and alternative box model?

The difference between the standard and alternative box is the following:

Standard Box Model	Alternative Box Model
In *standard box model* padding and border is added to declared width and height	In *alternative box model*, padding and border do not increase the declared width and height.

Standard box model image



Alternative box model image

10. Would you use margin or padding to create more space between 2 elements?

You should use ***margin*** for this task. Margins are used to move an element up or down on a page as well as left or right.

11. Would you use margin or padding to create more space between the contents of an element and its border?

You should use ***padding*** for this task. Padding is said to be the space inside an element, in other words, the padding is the space inside the element's border.

12. Would you use margin or padding if you wanted two elements to overlap each other?

You should use ***margin*** for this task. Margin is used for:

- Change an element's position on the page.
- Set the distance between nearby elements.
- Overlap elements.

13. What is the difference between a block element and an inline element?

The main differences are as follows:

Block Elements	Inline Elements
Block elements always start from a new line.	Inline elements never start from a new line.
Block elements have top and bottom margins.	Inline elements don't have a top and bottom margins.
Block elements cover space from left to right as far as it can go.	Inline elements only cover the space as bounded by the tags in the HTML element.

14. What is the difference between an inline element and an inline-block element?

The main difference is as follow:

Inline Element	Inline-block Element
The element doesn't start on a new line and only occupy just the width it requires. You can't set width and height values.	It's formatted just like the inline element, where it doesn't start on a new line. But, you can set width and height values.

15. Is an h1 block or inline?

The **h1** tag is a block element.

16. Is button block or inline?

A **button** is an inline element.

17. Is div block or inline?

A **div** is a block element.

18. Is span block or inline?

A **span** is an inline element.

19. What's the difference between a flex container and a flex item?

The main difference is as follow:

Flex Container	Flex Item
A flex container is any that has the property **display: flex**; on it.	A flex item is any element that is inside the **flex container**.

20. How do you create a flex item?

A **flex item** is the direct children of a **flex container**, (*elements with display: flex or display: inline-flex set on them*) become **flex items**. Below is an example of how to create a flex item:

HTML

```
<!-- HTML example -->
<div class="container">
    <div class="box">1</div>
    <div class="box">2</div>
    <div class="box">3</div>
    <div class="box">4</div>
</div>
```

CSS

```
/* Flex container */
.container {
    display: flex;
    background-color: #776fac;
    padding: 5px;
}
/* Flex item */
.box {
    background-color: aliceblue;
    margin: 5px;
    height: 80px;
    width: 80px;
    line-height: 80px;
    text-align: center;
    font-size: 18px;
}
```

Result:

21. What are the 3 values defined in the shorthand flex property?

The **flex** property is a shorthand property for the **flex-grow**, **flex-shrink**, and **flex-basis** properties.

```css
/* Flex shorthand */
.box2 {
    flex: 1 1 100px;
}
/* Same class with no flex shorthand */
.box2 {
    flex-grow: 1;
    flex-shrink: 1;
    flex-basis: 100px;
}
```

22. How do you make flex items arrange themselves vertically instead of horizontally?

You can do that by changing the **flex-direction: row;** property to **flex-direction: column**. By default the flex direction is set to row.

```css
/* Flex items horizontally */
.container1 {
    display: flex;
    flex-direction: row; /* Set by default */
}
/* Flex items vertically*/
.container2 {
    display: flex;
    flex-direction: column;
}
```

23. What is the difference between justify-content and align-items?

The main difference is as follow:

Justify-content	Align-items
It controls alignment of all items on the **main** axis	It controls alignment of all items on the **cross** axis.

24. How do you use flexbox to completely center a div inside a flex container?

You can center a div element completely inside a flex container by using the flex properties **align-items** and **justify-content** together. e.g.

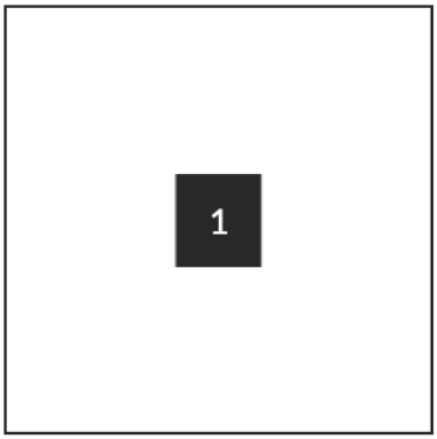
CSS

```css
/* Flex container */
.container {
    display: flex;
    border: 1px solid #2d2d2d;
    width: 200px;
    justify-content: center;
    align-items: center;
}
/* Flex item */
.box {
    background-color: #2d2d2d;
    color: #ffffff;
    width: 40px;
    height: 40px;
}
```

HTML

```
<!-- HTML example -->
<div class=”container”>
    <div class=”box”>1</div>
</div>
```

Result:



25. What’s the difference between justify-content: space-between and justify-content: space-around?

The main difference is the following:

Justify-content : space-between	Justify-content : space-around
Distribute items evenly the first item is flush with the start, the last is flush with the end	Distribute items evenly items have a half-size space on either end.

Justify-content: space-between;

Justify-content: space-around;