**Em vs Rem**

**EM** is relative to the parent element’s font size, so if you wish to scale the element’s size based on its parent’s size.

**REM** is relative to the root (HTML) font size, so if you wish to scale the element’s size based on the root size, no matter what the parent size is.

**CSS Position**

The position property specifies the type of positioning method for an element.

**Position different values:**

* Static
* Relative
* Fixed
* Absolute
* Sticky

**Position: static:**

Static positioned elements are not affected by the top, bottom, left and right properties. It is always positioned according to the flow of the page

**Position: relative:**

Is positioned relative to its normal position. Setting the top, right, bottom and left properties of a relatively positioned element will cause it to adjust away from its normal position.

**Position: Fixed:**

Is positioned relative to the viewport, the element stays in the same place even if the page is scrolled. The top, right, bottom and left properties are used.

**Position: Absolute:**

Is positioned elative to the nearest positioned ancestor. It uses document body, and moves along with page scrolling.

**Position: Sticky:**

Is positioned based on the user’s scroll position. Element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position.

**For vs While loop**

**For loop:**

**Format:**

For (initialize; condition; iteration) {

}

**Use:**

For loops are used when we knew number of iterations

**Condition:**

If there is no condition it will become an infinity loop

**While loop:**

**Format:**

Initialize;

While(condition) {

Iteration;

}

**Use:**

while loops are used when the number of iterations is not exactly known

**Condition:**

If the condition is not put up in ‘while’ loop it provides compilation error.

**Object vs Instance**

**Object:**

Objects are the results of instantiating a class. Instantiation is the process of taking the blueprint and defining each attribute and behavior so that the resultant object represents a real-life object. Object is created with the use of new operator

**Instance:**

 a memory block, which contains the reference to an object. the name of the instance can be used to access the start of the object memory area.

**Regular vs Arrow function js**

1. **Syntax:**

One very basic difference is that the Regular Functions uses function keyword but Arrow Functions don’t inspite it uses arrow symbol (=>).

1. **this keyword:**

**Regular function** has its their own this context.

**Arrow function** don’t have their own. Inside an Arrow function this value hold the this value of outer function.

1. **Using new keyword:**

**Regular functions** are constructible and callable. As it are constructible, they can be called using the ‘new’ keyword.

**Arrow functions** are only callable and not constructible.

1. **Implicit return:**

**Regular functions** return expression statement is used to return the result from the function. It return statement is not available inside the function then undefined is returned from the function.

**Arrow functions** there is one exception where return is not mandatory to return result from the functions.