Day2 Assignment2

Q1. Difference between JavaScript console with Node.Js

A-

	JavaScript	**Node.Js**
Use	JavaSript is used for writing scripts on the Websites.	NodeJS is a Javascript runtime environment.
User Side	It is basically used on the client side.	It is mostly used on the server side.
Type	JavaScript is a programming language. It is running in any web browser with a proper browser engine.	It is an interpreter and environment for JavaScript with some specific useful libraries which JavaScript programming can use separately.
Utility	Mainly using for any client-side activity for a web application, like possible attribute validation or refreshing the page in a specific interval or provide some dynamic changes in web pages without refreshing the page.	It mainly used for accessing or performing any non-blocking operation of any operating system, like creating or executing a shell script or accessing any hardware-specific information or running any backend job.
Running Engine	JavaScript running any engine like Spider monkey (FireFox), JavaScript Core (Safari), V8 (Google Chrome).	Node JS only run in a V8 engine which mainly used by google chrome. And javascript program which will be written under this Node JS will be always run in V8 Engine.

Q2. Summary of lecture Ryon seddon

A- 1. Parsing – DOM Tree
2. DOM tree –Render Tree
3. There are actually 4trees.
4. Layout Computes where a node is on Actual Screen.
5. Painting Computes bitmaps and composite screen.

Q3. Is it necessary to write HEAD, BODY and HTML tags?

A- Those tags are required by the DOCTYPE you're using and should not be omitted. The html element is the root element of every html page. If you look at all other elements' description it says where an element can be used (and almost all elements require either head or body).

Q4.What is Prototype in JS?

A- Prototype in JavaScript

JavaScript is a dynamic language. You can attach new properties to an object at any time as shown below.

Example: Attach property to object

```
function Student() {
   this.name = 'John';
   this.gender = 'Male';
}

var studObj1 = new Student();
```

```
studObj1.age = 15;
alert(studObj1.age); // 15

var studObj2 = new Student();
alert(studObj2.age); // undefined
```
Try it

As you can see in the above example, age property is attached to studObj1 instance. However, studObj2 instance will not have age property because it is defined only on studObj1 instance.

So what to do if we want to add new properties at later stage to a function which will be shared across all the instances?

The answer is **Prototype**.

The prototype is an object that is associated with every functions and objects by default in JavaScript, where function's prototype property is accessible and modifiable and object's prototype property (aka attribute) is not visible.

Every function includes prototype object by default.

Prototype in JavaScript

The prototype object is special type of enumerable object to which additional properties can be attached to it which will be shared across all the instances of it's constructor function.

So, use prototype property of a function in the above example in order to have age properties across all the objects as shown below.

Example: prototype

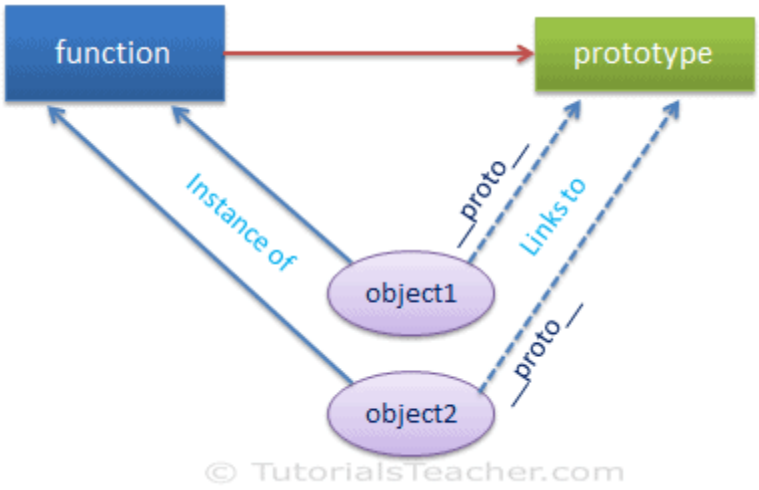
```
function Student() {
    this.name = 'John';
    this.gender = 'M';
}

Student.prototype.age = 15;

var studObj1 = new Student();
alert(studObj1.age); // 15

var studObj2 = new Student();
alert(studObj2.age); // 15
```
Try it

Every object which is created using literal syntax or constructor syntax with the new keyword, includes __proto__ property that points to prototype object of a function that created this object.



Prototype in JavaScript

You can debug and see object's or function's prototype property in chrome or firefox's developer tool. Consider the following example.

Example: prototype

```
function Student() {
    this.name = 'John';
    this.gender = 'M';
}

var studObj = new Student();

console.log(Student.prototype); // object
console.log(studObj.prototype); // undefined
console.log(studObj.__proto__); // object

console.log(typeof Student.prototype); // object
console.log(typeof studObj.__proto__); // object

console.log(Student.prototype === studObj.__proto__ ); // true
```

A-