

Assignment No-1

1. Explain the use of error handling in JavaScript with Example.

Ans:- Error handling helps in handling both hardware and software error gracefully and help execution to resume when interrupted. When it comes to error handling in software either the programmer develops the necessary codes to handle the error or make use of software tools handle the error. In case where error cannot be satisfied error handling is usually done with returning special error code. Special applications known as error handle are available for certain application to help in error handling.

Example :-

```
<!DOCTYPE html>
<html>
<body>
<h1> Error Handling </h1>
<p id = "error handling" X / P>
<script>
  try
  {
    var script result sum (10,20);
    sum is not defined yet
  }
  var result = sum (10,20);
  // sum is not defined yet
}
```


Catch (ex)

```
{
  document.getElementById("error_msg")
    innerHTML = ex;
}
```

</script>

</body>

</html>

Output :-

Error Handling
Reference Error. Sum is not defined

Q.2. How is JS class, object and prototype created? Explain?

→ Class :-

The body of class is the part that is curly brackets ({}) members such as method or is executed in strict mode i.e. Code written here is subject to strict syntax for increased performance, some otherwise silent error will be thrown and certain keywords are reserved for further version of Ecma Script. The constructor method is a special method only one special method with the name "constructor" in a class.

Objects :-

You can create an object with these two steps.

Date

a) Define the object type by writing a constructor function. There is a strong convention with good reason to use a capital initial letter.

b) Create an instance of the object with new. To define an object type, create a function for the object type. Create methods :- Suppose you want to create an object type for cars. You want this type of object to have properties for make, model, and year.

Prototype :-

Javascript engine add a prototype property inside a function. Prototype property is basically an object where we can attach method object which enable all the other to inherit these method and properties. One of these way is to create an object using function constructor object. Now we will add `__proto__` or `__proto__` in the object which will point the prototype;

Q. Describe the JS Encapsulation Inheritance once Polymorphism and Abstraction with Suitable example?

→ Encapsulation :

The JS encapsulation is a process of binding the data i.e variable with the function acting on that data. It allows to control the data and validation it is use key word to make data members Private and setter method to set that data.

Example :-

<Script>

```
class student
{
```

```
  constructor ()
  {
```

```
    var name;
    var marks; }
}
```

```
  get name ()
  { return this.name;
  }
```

```
  set name (Name)
  {
```

```
    this.name = name;
  }
```

```
  get marks ()
  { return this.marks; }
```

```
  set marks (marks)
  {
```



```

    this.marks = marks; }
    var stud = new Student ();
    stud.set Name ("Aniket");
    stud.set name (180);
    document.write In (stud.get name () 'f + stud-
    get marks ());
    </script>

```

Output :- Aniket 180

* Inheritance :- The JS inheritance to create a class inheritance use the extends keywords. A class created with a class inheritance all the methods from another class. It maintains an IS a relationship the extends keywords is used in class expression or class declaration. we can also use a prototype based approach to achieve inheritance.

eg.

```

    </script>
    class moment extends Date
    {
        constructor ()
        {
            Super ();
        }
        var m = new moment ();
        document.write In ("moment date")
    }

```

```

    Document.write In (m.get date () + " - " +
    (m.get month () + 1) + " " + m.get full year ()
    </script>

```


Output :-
Current Date :- 6-9-2021

* Polymorphism :-

The polymorphism is a concept of an object-oriented paradigm that provides a way to perform a single action in different forms. It provides an ability to call the same methods on different objects. as a JS is not a type-safe language we can pass any type of data members with the methods.

eg:-

```
</script>
class A
{
    display ()
}
document.write In ("A is invoked");
}
class B extends A
{
}
var b = new B ();
b.display ();
</script>
```

Output :-

A IS invoked.

* Abstraction :-

An abstraction is a way of hiding the implementation details and showing only the functionality to the user. In other words, it only the required ones. we cannot create an instance of abstraction class it reduce the duplication of code.

eg:-

```
<script>
```

```
// Creating a Constructor functions function  
Vehicle ()
```

```
{
```

```
  this.Vehicle Name = Vehicle Name;
```

```
  throw new Error ("you cannot create  
    an instance of abstract class");
```

```
}
```

```
Vehicle.prototype.display = function ()
```

```
{
```

```
  return this.Vehicle Name;
```

```
}
```

```
var vehicle = new Vehicle ();
```

```
</script>
```


Q.4 Write in short about cookies with its attributes

→ Cookies are pieces of information stored on the client side, which are set to the client. Cookies are primarily used for session creation and maintaining sessions. However, security of cookies effectively means security of a user's identity. Cookies can be secured by properly setting cookie attributes. These attributes are:

- Secure :- The Secure attribute makes sure that cookies will only be sent with requests made on a secure connection. It is not possible to steal cookies by sniffing, we need to be very careful while setting this attribute.

→ Domain and path :- The domain attribute signifies the domain for which the cookies are valid and can be submitted with every request for this domain or its every request for this domain or its sub domain. The 'path' attribute signifies the URL or 'path' for which the cookie is valid. The default path attribute is set on.

Expires :- This attribute is used to set persistent cookies. It signifies that how long the browser should the persistent cookie and when the cookie should be deleted.

Q.5 What is JS hosting? Explain?

→ Javascript hosting refers to be process where by the interpreter allocate memory for variable and function declaration prior to execution of the code. Declaration that are made using var are initialized with a default value of undefined. Declaration made using let and const are not initialized as part of hosting. Cleanup actually hosting is often presented as the interpreter "Splitting" variable declaration to the top of code. This allow variable to appear in code before they are defined. Note however that any variable initialization in the original code will not happen until the line of code is executed.