

Explore

EN
What do you want to learn?

JavaScript

JavaScript is the programming language for the web. It makes an HTML page dynamic. JavaScript has many built-in functions, objects and supports user-defined ones too...More

Start

Learning Progress

Status: Completed

Completion Certificate

Click on Generate Certificate button to claim your completion certificate

Generate Certificate

Overview
Contents
Discussions

What you will learn

By the end of this course, you will be able to: Create interactive HTML web pages using JavaScript. Create web applications with modular code using Functions and Modules. Develop a web app including Object-Oriented Programming concepts such as Classes in JavaScript. Implement web app with validations in HTML document and handle the events. Implement web app with Asynchronous Programming using JavaScript Async and Await. Implement a secure web app by incorporating mitigations for common security vulnerabilities in JavaScript such as XSS and CSRF.

Pre Contents

Authors/Creators

- SH Sanjna Hans
- MS Mathan Sakthivel
- AK Anitha Kumari K
- KR Kavya Ramakrishna

Show Curators/Contacts

At a glance

- Course
- 15h 34m
- Beginner Level
- Free
- Infossys Wingspan
- EN
- JavaScript, Web Technology, JavaScript BOM, JavaScript DOM, JavaScript Events, JavaScript

Thank you. Your test submitted.

You have cleared this assessment.

Obtained Percentage

Obtained Marks

92 %

23 / 25

Best Attempt Score:92 % on 17-06-2025

Observe the code and choose one of the following:

```
class Person {  
    constructor(friends) { this.friends = friends; }  
}
```

```
class Employee extends Person {  
    constructor(friends) { super(friends) }  
    displayFriends() {  
        for (let friend of friends) {  
            console.log(friend)  
        }  
    }  
}
```

```
let friendList = ["John", "Tom", "Tim", "Jack", ""];  
let x = new Employee(friendList);  
x.displayFriends();
```

Warning

This operation is disabled.

Ok

- ☐ Above code will result in error because displayFriends() method cannot use friends as is defined in Base class - Person
- ☒ Above code will result in error as friends is not defined in child class Employee.
- ☐ Above code will result in error as friends is not referred using this keyword in Line 7.
- ☐ All the options

For More Solutions: <https://github.com/DevGoyalG/NIET-Infosys-Springboard>

What will be the output for the below code snippet?

```
let studentData = {  
  name: "John",  
  country: "India"  
};  
var dynamicProperty = "age";  
studentData[dynamicProperty] = 25;  
var dynamicProperty = "city";  
studentData.dynamicProperty = "Pune";  
console.log(studentData.age);  
console.log(studentData.city);
```

Warning

This operation is

- ☐ 25 Pune
- ☒ 25 undefined
- ☐ 25 null
- ☐ undefined undefined

If a function is marked with the `async` keyword which returns a text value. What would be the output of that function, if it is immediately executed?

- ☐ Text value
- ☒ A Promise object which is in pending state
- ☐ A Promise object which has resolved
- ☐ Exception will be thrown

Warning

This operation is disabled.

Ravi has developed the below code to find any string in an array that contains "Script" in it. What will be the outcome of below code snippet?

```
let output = []
var component = (...par1) => {
  output = [...par1];
  return output.filter((a) => a.includes('Script'))
}
console.log(component('ES6', 'JavaScript', 'CoffeeScript'))
```

- ☐ Compilation Error
- ☐ undefined
- ☒ ['JavaScript', 'CoffeeScript']
- ☐ JavaScriptCoffeeScript

Warning

This operation is disabled.

Ok

Observe the code snippet given below and choose the right option to access the computed property.

```
let movie = { name:"Galaxy of guardians", type:"Hollywood"}  
let movieProperty = 'ratings'  
movie['current'+ movieProperty]=5;
```

- ☐ console.log(movie.movieProperty);
- ☒ console.log(movie.currentratings);
- ☐ console.log(movie.ratings);
- ☐ console.log(movie.currentProperty);

Warning

This operation is disabled.

Ok

Consider two modules File1.js and File2.js as shown below. Which all are the code that can be placed in file2 so that all the exports of File1 are accessible in File2?

```
//File1.js

export class Employee {
  empId: string;
  empName: string;
  empStatus: string;
}

export function getDetails() {
  return 'Hello getDetails()';
}
```

Warning

This operation is disabled.

Ok

- i. import * as module1 from './File1.js'
- ii. import * from './File1.js'
- iii. import { Employee as emp, getDetails } from './File1.js'
- iv. import { Employee, getDetails } from './File1.js';

- ☐ i, ii, iii
- ☐ ii,iii, iv
- ☐ i, ii, iv
- ☒ i, iii, iv

Can you predict the output of the below-given code-snippet?

```
let states=[];  
states.push(['Rihanna',  
{  
  empId: 'I1001',  
  city: 'BNG',  
  age: 25  
}]);  
  
console.log(...state)
```

Warning

This operation is di

- ☐ Compilation Error
- ☐ undefined
- ☒ Entire Array would be printed on console
- ☐ { empId: 'I1001', city: 'BNG', age: 25 }

For More Solutions: <https://github.com/DevGoyalG/NIET-Infosys-Springboard>

Which of the below code snippets will give you the same results:

```
//a
let x = [1, 2, 3];
let y = [4, 5, 6];
const combinedArray1 = [...x, ...y];
console.log("one "+combinedArray1);
```

```
//b
let z = [3, 4, 5, 6];
const combinedArray2 = [1, 2, ...z];
console.log("two "+combinedArray2);
```

```
//c
let a = [3, 4, 5, 6];
const combinedArray3 = [1, 2, ...z];
console.log("three "+a);
```

☒ a and b

☐ a and c

☐ b and c

☐ All the given snippets

For More Solutions: <https://github.com/DevGoyalG/NIET-Infosys-Springboard>

Warning

This operation is disabled

Ok

An attacker identifies a CSRF vulnerability in 'mybank.com' website and started to exploit the valid users' accounts. How this issue could be fixed by the development team? Help the team to fix this issue by selecting the correct option from the below options.

- ☐ Advice their users not to open any links
- ☐ Give control to attackers so that they will attack
- ☐ Implement anti-CSRF tokens in their application
- ☒ CSRF attack don't have any mitigation

Warning

This operation is disabled.

What will be the output of the code snippet below?

```
let lis = [  
    { alterEgo: 'Bruce Wayne', city: 'Gotham City' },  
    { alterEgo: 'Oliver Queen', city: 'Star City' },  
    { alterEgo: 'Barry Allen', city: 'Central City' },  
];  
let [, , darkKnight] = lis;  
function cityHunter({ alterEgo: x, heroName: y, city: z }) {  
    document.write(x + '-' + y + '-' + z);  
}  
cityHunter(darkKnight);
```

Warning

This operation

- ☒ Barry Allen-undefined-undefined-Central City
- ☐ Bruce Wayne-null-null-Gotham City
- ☐ Error in Line 7 : Property mismatch

Observe the code below:

```
class Person {
    constructor(name, age) { this.pname = name; this.page = age; }
    displayMasterDetails() {
        console.log("name :", this.pname);
        console.log("age :", this.page);
    }
}

class Employee extends Person {
    constructor(name, age, empid) {
        //.....(1).....
        this.empID = empid;
    }
    displayAllDetails() {
        //.....(2).....
        console.log("Employee Id : " + this.empID)
    }
}
```

Warning

This operation is disabled.

Ok

What should be filled in blank (1) to make constructor of class Employee?

What should be filled in blank (2) to make displayMasterDetails() function call of base class Person in class Employee?

- ☐ blank 1 - super(name,age) , blank 2 - super.displayMasterDetails();
- ☒ blank 1 - super(name,age) , blank 2 - this.displayMasterDetails();
- ☐ blank 1 - super(name,age) , blank 2 - displayMasterDetails();
- ☐ All of the above.

For More Solutions: <https://github.com/DevGoyalG/NIET-Infosys-Springboard>

Can you predict the value of the 'x' variable?

```
const cars=[{'Name':'Toyota','Cost':324323},{'Name':'Benz','Cost':2216}];  
const x = cars.reduce((a,b)=>a+b.Cost,5);
```

- ☐ Only cost of Toyota car
- ☐ Only cost of Benz car
- ☐ Total Sum of both the cost of the cars
- ☒ Total Sum of both the cost of the cars + 5

Warning

This operation is disa

Predict the output for the below code snippet:

```
let name = "Sam"  
function demo() {  
    name += "Mark"  
}  
console.log(name);
```

Warning

This operat

- ☒ Sam
- ☐ Mark
- ☐ SamMark
- ☐ Reference error while accessing name in demo()

For More Solutions: <https://github.com/DevGoyalG/NIET-Infosys-Springboard>

Predict the output.

```
var temp = "8080";  
if (true) {  
    const temp = 8080;  
    console.log(typeof temp);  
}
```

- ☒ number
- ☐ string
- ☐ Error in line 3
- ☐ Error in line 4

Consider the below module definition:

```
// mymodule.js

export function a() {
    // some logic here
}

export function b() {
    // some logic here
}
```

Which of the below lines rightly imports :

- ☐ import * from "../mymodule"
- ☐ import * from "../mymodule.js"
- ☐ import as "../mymodule"
- ☒ import * as mymodule from "../mymodule"

Wait

This

Can you predict the output of the below-given code snippet?

```
let empId = 'I1005'
let employee = {
  empId,
  getEmpDetails() {
    return this.empId;
  },

  getDetails() {
    setInterval(() => { return this.empId },
  }
}
console.log(employee.getEmpDetails());
console.log(employee.getDetails());
```

Warning

This operation is disallowed

- ☐ Error in code
- ☒ I1005 undefined
- ☐ I1005

☐ For More Solutions: <https://github.com/DevGoyalG/NIET-Infosys-Springboard>

Smith, a web developer wants to write mitigation for CSRF attack and after discussing it with his team he decided to implement the mitigation by using anti-CSRF tokens and started to generate the tokens manually to prevent CSRF attacks.

Comment on the approach Smith is following.

- ☐ Smith is approaching in a right way
- ☐ Using 'https' is best way to prevent CSRF attacks
- ☒ Creating anti-CSRF token is a good approach
- ☐ CSRF attack is not a serious cyber-attack, so

Warning

This operation is disabled.

Ok

can use some built-in modules to generate tokens.

Consider the below code snippet written by David:

```
let myArray=new Array([1,2,3]);  
myArray.delete(1);  
console.log(myArray);
```

Warning

This operation

- ☐ []
- ☐ [1,3]
- ☐ [1,2,3]
- ☒ Array do not support delete function

What will be the output of the code snippet below?

```
let myList=["Edison","Tesla","Einstein"];  
let [,a]=myList;  
document.write(a);
```

- ☐ Error in line 2
- ☐ Edison
- ☒ Tesla
- ☐ Einstein

Warning

This opera

Predict the output

```
let employeeList = [  
  { id: "I1001", name: "Tom" },  
  { id: "I1002", name: "Jack" },  
  { id: "I1003", name: "Tim" },  
  { id: "I1004", name: "Sam" }  
];  
  
console.log("Emp Id: ${employeeList[0].id} Emp Na
```

Warning

This operation is disallowed

- ☐ No Changes required
- ☒ Change " " to `` in Line 7.
- ☐ Not possible to achieve since employeeList is an array of objects.
- ☐ Change " " to ' ' in Line 7.

Which of the below is CORRECT way of handling async code using Promises?

option 1:

```
new promise(function(res,rej){  
  // async code here  
  res("success");  
});
```

option 2:

```
Promise(function(resolve,reject){  
  // async code here  
  resolve("success");  
});
```

option 3:

```
new Promise(function(resolve){  
  // async code here  
  resolve("success");  
});
```

option 4:

```
new Promise(function(){  
  // async code here  
  resolve("success");  
});
```

☐ option 1

☐ option 2

☐ option 3

☒ option 4

Wa

This

Can you predict the output of the below-mentioned code?

```
const result = [];  
function check(par1, ...par2) {  
    return result.push(par1, par2);  
}  
console.log(check("Hello", "World"));
```

Warning

This operation is d

- ☐ Undefined
- ☒ 2
- ☐ HelloWorld
- ☐ [Hello, World]

Which of the below points helps in writing a secured client-side code?

- ☐ Check if the code is vulnerable to HTML injection
- ☐ Do not store any user sensitive information
- ☐ Integrate only trusted third party libraries
- ☒ All the above

Warning

This operation is disabled.

What is the output of the below-mentioned code?

```
let employee = {  
  empId: 'I1001',  
  empName: {  
    firstName: 'John',  
    middleName: 'Thomson',  
    lastName: 'Mark'  
  },  
  empStatus: 'active'  
};  
let { firstName: middleName, lastName } = employee;  
console.log(middleName);
```

- ☐ undefined
- ☐ Compilation Error
- ☐ Thomson
- ☒ John

Warning

This operation

What will be the output of the following code snippet?

```
new Promise(function (resolve, reject) {  
  let b;  
  setTimeout(compute = (a = 5) => resolve(a + b), 1000);  
  b = 25;  
}).then(function (data) { console.log(data) });
```

Warning

This operation

☐ 5

☒ 30

☐ NaN

☐ Undefined