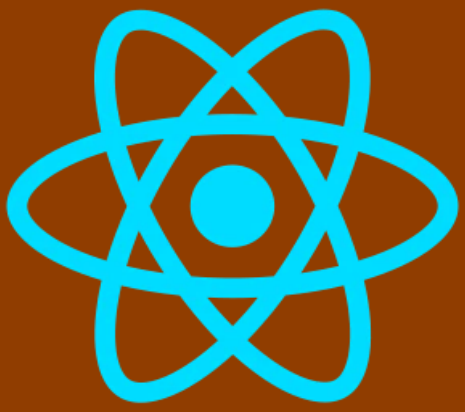

How many ways to create callback functions in JavaScript?

NEXT

Ways to create callback.

Here are various ways modern and different approaches to creating callback functions in JavaScript

NEXT

Function Declaration:

Define a named function and pass it as a callback.

```
function myCallback() {
  console.log('Callback function called');
}

function doSomething(callback) {
  // Perform some operation
  callback();
}

doSomething(myCallback);
```

Function Expression

Create an anonymous function (function expression) and pass it as a callback.

```
const myCallback = function() {
  console.log('Callback function called');
};

function doSomething(callback) {
  // Perform some operation
  callback();
}

doSomething(myCallback);
```

NEXT

Arrow Function

Use arrow functions for concise callback definitions.

```
const myCallback = () => {
  console.log('Callback function called');
};

function doSomething(callback) {
  // Perform some operation
  callback();
}

doSomething(myCallback);
```


Callback with Anonymous Function

Define a callback function inline when calling a function that expects a callback.

```
function doSomething(callback) {
  // Perform some operation
  callback();
}

doSomething(function() {
  console.log('Callback function called');
});
```


Promises with .then()

Use Promises and the .then() method for asynchronous callbacks.

```
const myPromise = new Promise((resolve, reject)
=> {
  // Perform an asynchronous operation
  resolve('Promise resolved');
});

myPromise.then((result) => {
  console.log(result);
});
```

NEXT

Promises with async/await

Modern asynchronous JavaScript code often uses async/await for handling Promises.

```
async function doSomethingAsync() {
  // Perform an asynchronous operation
  const result = await myPromiseFunction();
  console.log(result);
}
```

NEXT

Callback in Event Listeners

Use callback functions with event listeners to handle DOM events.

```
document.getElementById('myButton').addEventListener('click', function() {
  console.log('Button clicked');
});
```

NEXT

Callbacks in setTimeout and setInterval

Pass callback functions to setTimeout and setInterval for delayed and repeated execution, respectively.

```
setTimeout(function() {
  console.log('Delayed callback');
}, 1000);

setInterval(function() {
  console.log('Repeated callback');
}, 1000);
```

NEXT

Using Function References

Pass a reference to an existing function as a callback.

```
function myCallback() {
  console.log('Callback function called');
}

function doSomething(callback) {
  // Perform some operation
  callback();
}

doSomething(myCallback);
```

Promises and async/await with try...catch

Use async/await with try...catch to handle Promise rejections gracefully.

```
async function fetchData() {
  try {
    const result = await
fetch('https://example.com/data');
    console.log(result);
  } catch (error) {
    console.error(error);
  }
}
```

NEXT

Callbacks with Promises and .then().catch()

Handle Promise rejections with the .catch() method.

```
myPromiseFunction()
  .then((result) => {
    console.log(result);
  })
  .catch((error) => {
    console.error(error);
  });
```

NEXT

Ways to create callback.

I am a ReactJS developer with 2.5 years of experience. I am proficient in all aspects of ReactJS development, including building user interfaces, creating reusable components, and integrating with APIs. I am also familiar with other JavaScript libraries.

NEXT

Ways to create callback.

I am a highly motivated and results-oriented developer. I can work independently and as part of a team, and I am always willing to learn new things.

If you are looking for a skilled and experienced ReactJS developer,

Contact me.

location (Pakistan)

NEXT

Ways to create callback.

Is this information Insightful 💡,

Connect 🤙 and follow Muhammad Idrees for further posts.

What are your thoughts 🤔 about this post do comment, 📝

Don’t keep this information to yourself Spread the word by tagging 🏷 and sharing this post.

and hit the 👍 like button too 😊,

NEXT