

FUNCTION

The background of the slide is a solid blue gradient. Overlaid on this are several thin, white, wavy lines that create a sense of motion and depth. These lines are most prominent in the lower half of the image, where they form a series of overlapping, undulating shapes that resemble a stylized landscape or a series of waves. The lines are more densely packed in some areas, creating a sense of volume and movement.

NGUYEN TRONG TIEN

1. What is Async Javascript?
2. Requests
3. Callback function
4. Using Json Data
5. Callback hell
6. Promise
7. Chaining Promise
8. The Fetch API, Axios
9. Async vs await

1. What is Async Javascript?

CALLBACK

Jquery Ajax

```
//const urlBacon =  
"https://baconipsum.com/api/?type=all-meat";  
const urlBacon =  
"https://jsonplaceholder.typicode.com/todos/1";  
(function(){$.ajax({ method: "GET",  
url: urlBacon,  
dataType: "json"  
})  
.done(function (data) {  
console.log(data);  
})  
.fail(function () {  
alert("no good");  
});  
})();
```

<https://api.jquery.com/jquery.ajax/>

```
$.ajax({  
    url:  
    "https://jsonplaceholder.typicode.com/todos/1",  
    beforeSend: function (xhr) {  
        xhr.overrideMimeType("text/plain;  
charset=x-user-defined");  
    }  
})  
    .done(function (data) {  
        if (console && console.log) {  
            console.log("Sample of data:",  
data.slice(0, 100));  
        }  
    });
```

XMLHttpRequest RESTful GET

```
//Get all users
var url = "http://localhost:8080/api/v1/users";
//var url =
'https://jsonplaceholder.typicode.com/todos';
var xhr = new XMLHttpRequest()
xhr.open('GET', url, true)
xhr.onload = function () {
    var users = JSON.parse(xhr.responseText);
    if (xhr.readyState == 4 && xhr.status == "200") {
        console.table(users);
    } else {
        console.error(users);
    }
}
xhr.send(null);
```

XMLHttpRequest RESTful GET

```
// Get a user
var url = "http://localhost:8080/api/v1/users";
var xhr = new XMLHttpRequest()
xhr.open('GET', url + '/1', true)
xhr.onload = function () {
    var users = JSON.parse(xhr.responseText);
    if (xhr.readyState == 4 && xhr.status == "200") {
        console.table(users);
    } else {
        console.error(users);
    }
}
xhr.send(null);
```


XMLHttpRequest

RESTful POST

```
// // Post a user
// var url = "http://localhost:8080/api/v1/users";
//var url =
'https://my-json-server.typicode.com/typicode/demo/posts';

var data = {};
//data.firstname = "John";
//data.lastname = "Snow";
var json = JSON.stringify(data);

var xhr = new XMLHttpRequest();
xhr.open("POST", url, true);
xhr.setRequestHeader('Content-type',
'application/json; charset=utf-8');
xhr.onload = function () {
    var users = JSON.parse(xhr.responseText);
    if (xhr.readyState == 4 && xhr.status == "201") {
        console.table(users);
    } else {
        console.error(users);
    }
}
xhr.send(json);
```

XMLHttpRequest

RESTful PUT

```
// Update a user
var url = "http://localhost:8080/api/v1/users";

var data = {};
data.firstname = "John2";
data.lastname = "Snow2";
var json = JSON.stringify(data);

var xhr = new XMLHttpRequest();
xhr.open("PUT", url + '/12', true);
xhr.setRequestHeader('Content-type', 'application/json; charset=utf-8');
xhr.onload = function () {
    var users = JSON.parse(xhr.responseText);
    if (xhr.readyState == 4 && xhr.status == "200") {
        console.table(users);
    } else {
        console.error(users);
    }
}
xhr.send(json);
```

XMLHttpRequest

RESTful DELETE

```
// Delete a user
var url = "http://localhost:8080/api/v1/users";
var xhr = new XMLHttpRequest();
xhr.open("DELETE", url + '/12', true);
xhr.onload = function () {
    var users = JSON.parse(xhr.responseText);
    if (xhr.readyState == 4 && xhr.status == "200") {
        console.table(users);
    } else {
        console.error(users);
    }
}
xhr.send(null);
```

PROMISE

The background of the slide is a solid blue gradient. Overlaid on this are several sets of thin, white, curved lines that flow from the left side towards the right. These lines create a sense of movement and depth, resembling waves or a stylized landscape. The lines are more densely packed in some areas, creating a 3D effect.

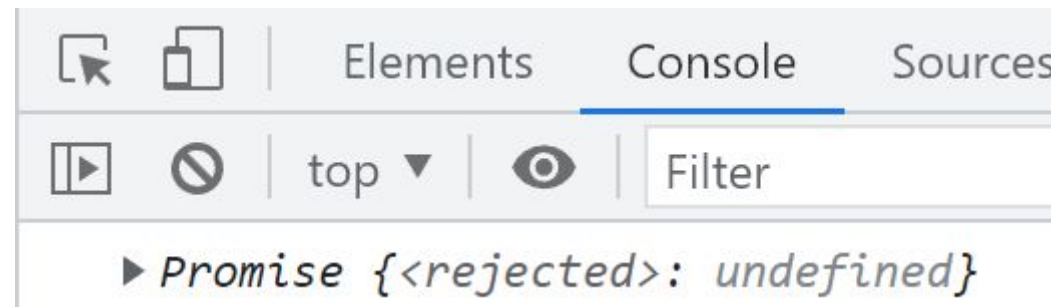
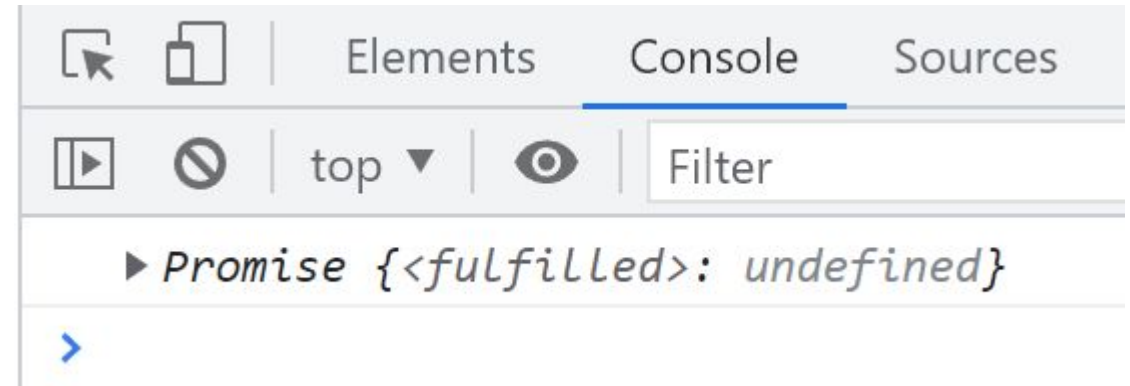
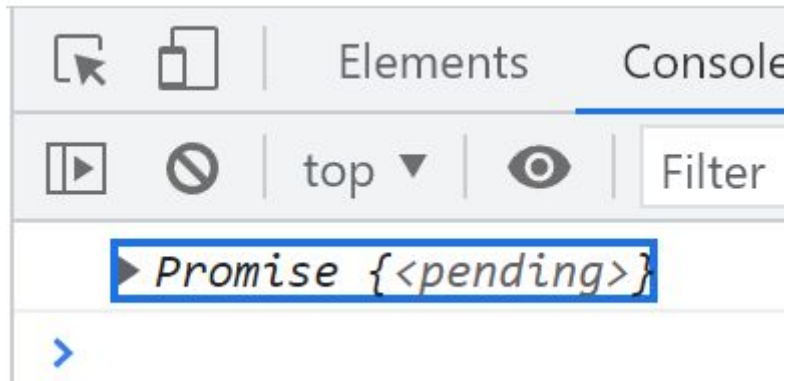
NGUYEN TRONG TIEN

How to create Promise()

```
var f = function (resolve, reject) {  
    //resolve();  
    //reject();  
}  
  
var promise = new Promise(f);
```

State of Promise()

JS



How to create Promise()

```
promise  
  .then()  
  .catch()  
  .finally();
```

How to create Promise()

```
promise
    .then(
    )
    .catch(
    )
    .finally(
    );
```

How to create Promise()

```
promise
    .then(
        function () {
            console.log('Successful!');
        }
    )
    .catch(
        function () {
            console.log('UnSuccessful!');
        }
    )
    .finally(
        function () {
            console.log('Finally!');
        }
    );
```

How to create Promise()

```
promise
    .then(
        function () {
            console.log('Successful!');
        }
    )
    .catch(
        function () {
            console.log('UnSuccessful!');
        }
    )
    .finally(
        function () {
            console.log('Finally!');
        }
    );
```

PROMISE CHAIN

```
promise
    .then(
        function () {
            console.log('1!');
        }
    )
    .then(
        function () {
            console.log('2!');
        }
    )
    .then(
        function () {
            console.log('3!');
        }
    )
    )
```

PROMISE CHAIN

```
promise
    .then(
        function () {
            console.log(0);
            return 1;
        }
    )
    .then(
        function (data) {
            console.log(data);
            return 2;
        }
    )
    .then(
        function (data) {
            console.log(data);
        }
    )
    )
```


RETURN PROMISE

```
promise
    .then(
        function () {
            return new
Promise(function(resolve){
                setTimeout(resolve,
3000);
            });
        }
    )
    .then(
        function (data) {
            console.log(data);
            return 2;
        }
    )
    .then(
        function (data) {
            console.log(data);
        }
    )
    )
```

PROMISE METHODS (RESOLVE, REJECT, ALL)

NGUYEN TRONG TIEN

The background is a solid blue gradient. Overlaid on this are numerous thin, white, curved lines that flow from the left side towards the right, creating a sense of movement and depth. These lines are more densely packed in some areas, forming wave-like shapes that rise and fall across the frame.

REAL USING PROMISE

NGUYEN TRONG TIEN

FETCH

NGUYEN TRONG TIEN

Get Fetch

```
fetch('https://jsonplaceholder.typicode.com/todos')
  .then(response=>response.json())
  .then(
    (data)=>{
      var htmls = data.map(
        function(posts){
          return `<li>
            <h2>${posts.id}</h2>
            <h2>${posts.title}</h2>
            </li>`
        }
      );
      var html = htmls.join();
      var text =
document.getElementById('info').innerHTML
      = html;
    }
  )
  .catch(()=>console.log('Có lỗi!'))
```

Get Fetch

```
fetch('https://jsonplaceholder.typicode.com/posts/1')  
  .then((response) => response.json())  
  .then((json) => console.log(json));
```

```
fetch('https://jsonplaceholder.typicode.com/posts')  
  .then((response) => response.json())  
  .then((json) => console.log(json));
```


Creating a resource

```
fetch('https://jsonplaceholder.typicode.com/posts/1', {  
  method: 'PUT',  
  body: JSON.stringify({  
    id: 1,  
    title: 'foo',  
    body: 'bar',  
    userId: 1,  
  }),  
  headers: {  
    'Content-type': 'application/json; charset=UTF-8',  
  },  
})  
  .then((response) => response.json())  
  .then((json) => console.log(json));
```

Updating a resource

```
fetch('https://jsonplaceholder.typicode.com/posts', {  
  method: 'POST',  
  body: JSON.stringify({  
    title: 'foo',  
    body: 'bar',  
    userId: 1,  
  }),  
  headers: {  
    'Content-type': 'application/json; charset=UTF-8',  
  },  
})  
  .then((response) => response.json())  
  .then((json) => console.log(json));
```

Deleting a resource

```
fetch('https://jsonplaceholder.typicode.com/posts/1', {  
  method: 'DELETE',  
});
```

AXIOS

<https://github.com/axios/axios>

```
<script  
src="https://unpkg.com/axios/dist/axios.min.js"></script>
```

NGUYEN TRONG TIEN

Get Axios

```
axios({  
  method: 'get',  
  url: 'https://jsonplaceholder.typicode.com/todos/1',  
})  
  .then(function (response) {  
    console.log(response);  
  });
```

Get Axios

```
axios({  
  method: 'get',  
  url: 'https://jsonplaceholder.typicode.com/todos/1',  
})  
  .then(function (response) {  
    console.log(response);  
  });
```


Get Axios

```
axios({  
  method: 'get',  
  url: 'https://jsonplaceholder.typicode.com/todos/1',  
})  
  .then(function (response) {  
    console.log(response);  
  });
```

Get Axios

```
axios({  
  method: 'get',  
  url: 'https://jsonplaceholder.typicode.com/todos/1',  
})  
  .then(function (response) {  
    console.log(response);  
  });
```

ASYNC-AWAIT

The background features a solid blue gradient. Overlaid on this are numerous thin, white, curved lines that flow from the bottom left towards the top right, creating a sense of motion and depth. These lines are more densely packed in some areas, forming a wave-like pattern that peaks towards the right side of the frame.

NGUYEN TRONG TIEN

function

```
<img id="img_1" src="" alt="">  
  <img id="img_2" src      alt="">  
  <img id="img_3" src="" alt="">
```

```
function httpGetAsync(Url, callback) {  
    var xmlHttp = new XMLHttpRequest();  
    xmlHttp.onreadystatechange = function () {  
        if (xmlHttp.readyState == XMLHttpRequest.DONE &&  
xmlHttp.status == 200)  
            callback(xmlHttp);  
    };  
    xmlHttp.open('GET', Url, true);  
    xmlHttp.send(null);  
}
```

Callback hell

```
httpGetAsync('https://picsum.photos/200/300', (data) => {  
    console.log('1',data);  
    document.getElementById('img_1').setAttribute('src', data.responseURL);  
  
    httpGetAsync('https://picsum.photos/200/300', (data) => {  
        console.log('2',data);  
        document.getElementById('img_2').setAttribute('src',  
data.responseURL);  
  
        httpGetAsync('https://picsum.photos/200/300', (data) => {  
            console.log('3',data);  
            document.getElementById('img_3').setAttribute('src',  
data.responseURL);  
        });  
    });  
});
```

make promise

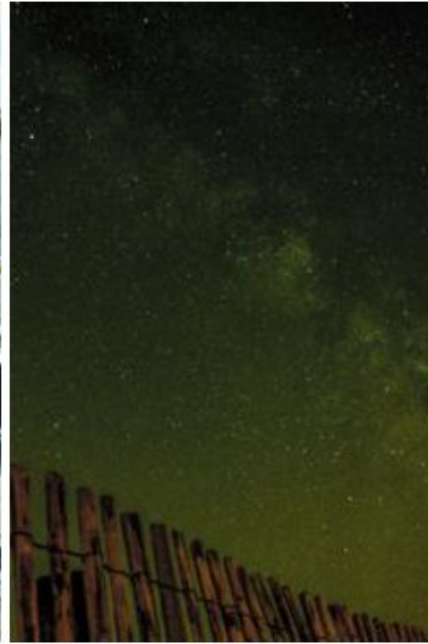
```
const myPromise1 = new Promise(function(resolve, reject){
    httpGetAsync('https://picsum.photos/200/300', resolve);
});
const myPromise2 = new Promise(function(resolve, reject){
    httpGetAsync('https://picsum.photos/200/300', resolve);
});
const myPromise3 = new Promise(function(resolve, reject){
    httpGetAsync('https://picsum.photos/200/300', resolve);
});
```

run promise

```
myPromise1
    .then((data)=>{
        document.getElementById('img_1').setAttribute('src',
data.responseURL);
        return myPromise2;})
    .then((data)=>{
        document.getElementById('img_2').setAttribute('src',
data.responseURL);
        return myPromise3;})
    .then(
        (data)=>{
            document.getElementById('img_3').setAttribute('src',
data.responseURL);
        })
```


result

JS



async-await

```
const curentPromise = new Promise(function(resolve, reject){
    httpGetAsync('https://picsum.photos/200/300', resolve);
});
const myPromise2 = new Promise(function(resolve, reject){
    httpGetAsync('https://picsum.photos/200/300', resolve);
});
const myPromise3 = new Promise(function(resolve, reject){
    httpGetAsync('https://picsum.photos/200/300', resolve);
});
```

THANK YOU

The background features a solid blue gradient. Overlaid on this are several sets of thin, white, wavy lines that create a sense of motion and depth. These lines form undulating shapes that resemble waves or flowing ribbons, with the most prominent ones curving upwards from the bottom left towards the top right.