

Materiali mesimor per javen 3 - 7 mars 2025 Scantech

E hane - 3 mars 2025:

Sesioni i pare: Perseritje me funksione (definimi dhe thirrja e funksionit, arrow functions)

Sesioni i dyte: Implement a function that filters even numbers from an array.

E merkure - 5 mars 2025:

Sesioni i pare: Callbacks and problem “callback hell” and solution with Promises (and a small introduction to async/await)



Shembull i callback:

```
function step1(callback) {
  setTimeout(() => {
    console.log("Step 1 completed");
    callback();
  }, 1000);
}

function step2(callback) {
  setTimeout(() => {
    console.log("Step 2 completed");
    callback();
  }, 1000);
}

function step3(callback) {
  setTimeout(() => {
    console.log("Step 3 completed");
    callback();
  }, 1000);
}

// Nested callbacks
step1(() => {
  step2(() => {
    step3(() => {
      console.log("All steps completed!");
    });
  });
});
```

 Copy  Edit



Zgjidhje Promises:



```
function step1() {
  return new Promise(resolve => {
    setTimeout(() => {
      console.log("Step 1 completed");
      resolve();
    }, 1000);
  });
}

function step2() {
  return new Promise(resolve => {
    setTimeout(() => {
      console.log("Step 2 completed");
      resolve();
    }, 1000);
  });
}

function step3() {
  return new Promise(resolve => {
    setTimeout(() => {
      console.log("Step 3 completed");
      resolve();
    }, 1000);
  });
}

// Chaining Promises
step1()
  .then(() => step2())
  .then(() => step3())
  .then(() => console.log("All steps completed ↓"));
```

Permbledhje:

 Summary		
Concept	What It Does	Example
Callback	A function passed into another function to execute later.	<code>fetchData(callback)</code>
Promise	Represents an async task that may succeed or fail.	<code>new Promise((resolve, reject) => {...})</code>
<code>.then()</code>	Runs when a promise resolves (success).	<code>.then(response => console.log(response))</code>
<code>.catch()</code>	Runs when a promise rejects (failure).	<code>.catch(error => console.log(error))</code>
Next Step: Use <code>async/await</code> for even better readability! 		

Sesioni i dyte: Diskutime mbi shembuj ne boten reale per *Database calls* edhe *API Requests*

E premte - 7 mars 2025:

Sesioni i pare: Konvertimi i Promises ne Asynchronous Functions (komandat *async/await*) dhe error handling me metoden *try {} catch {}*

Sesioni i dyte: detyre per: Konsumimi i te dhenave me metoden *fetch()* duke perdorur error handling me metoden *try {} catch {}*.

Linku i Fake API: <https://jsonplaceholder.typicode.com/>

Linku: <https://cocktail-recipes.surge.sh/>

Shembulli:



```
// put yasync function handleSearch(event) {  
  if (event) event.preventDefault();  
  
  let searchValue = searchInput.value.trim();  
  if (!searchValue) return;  
  
  try {  
    let response = await fetch(`https://www.thecocktaildb.com/api/json/v1/1/search.php?s=${searchValue}`);  
    if (!response.ok) throw new Error("Failed to fetch data");  
  
    let data = await response.json();  
    console.log("Fetched Data:", data);  
  
    if (data.drinks) {  
      console.log("Cocktails Found:", data.drinks.map(drink => drink.strDrink));  
    } else {  
      console.log("No cocktails found.");  
    }  
  } catch (error) {  
    console.error("Error fetching data:", error);  
  }  
}  
}our code here
```