סיכום שיעור

מרצה: שלמה ספוזניקוב

10/08/2023

חזרה קצרה על מה שלמדנו
ומענה לשאלות על משימת הבית

זמן שאלות

מחלקת Promise

מחלקת Promise

JS_Advanced_Promise.html

בחלק זה נלמד מגוון נושאים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה זה Promise?
- מה ההבדל בין resolve ל- reject ?
- מה נצטרך לשלוח לבנאי של Promise ?
- באיזה מצבים נזדקק להשתמש בטכניקה זו?
- מה הם המצבים האפשריים של Promise ?
- מה הם המאפיינים והמתודות המיוחדים למחלקת Promise ?

שם פונקציה 1 | שם פונקציה 2

```
let myPromise = new Promise(function (myInnerValue, myInnerError) {
    const uPass = prompt("Enter Password");
    console.log("Connecting to the server, reply in 3000 milliseconds")
    setTimeout(function () {
        if (uPass == "123") {
            myInnerValue("Welcome");
        } else {
            myInnerError("User NOT Welcome");
        }
    }, 3000)
});
```

הכנסת ערך לפונקציה 1

הכנסת ערך לפונקציה 2

```
myPromise.then(
    function (value) { console.log(value); alert(value); },
    function (error) { console.log(error); alert(error); }
);
```

ערך שהתקבל מפונקציה 2

ערך שהתקבל מפונקציה 1

```
myPromise
  .then((myNumber) => {
    console.log("myNumber", myNumber);
  })
  .catch((myError) => {
    console.log("error", myError);
  });
```

תרגיל 1

- יש ליצור promise שבודק האם המילה abc“" בגודל 3 אותיות או שלא
- (הפעלה)אם זה נכון אז בקונסול לרשום נכון
- אחרת לרשום בקונסול לא נכון

תרגיל 2

- יש ליצור promise שפועל אחרי 2 שניות
- הpromise יצור 2 מספרים רנדומליים בין 0 ל 10 ויחלק אותם
- אם המספר השני הוא 0 אז הpromise נכשל.
- (הפעלה)
- יש להציג את הפתרון או את הודעת שגיאה

תרגיל 3

- יש ליצור promise שפועל אחרי 1 שניות.
- יש ליצור פונקציה שיוצרת promise ולהעביר ל-promise 3 מספרים
- ה-promise צריך לחשב את הממוצע של 3 המספרים
- אם הממוצע שווה לאחד המספרים אז ה-promise נכשל

תרגיל *

- יש להמיר את xhr ל-promise
- יש להציג את כל הדמויות
- כל פעם 20 בגלילה יוצג עוד 20

מחלקת Promise

בחלק זה נלמד מגוון נושאים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה זה Promise?
- מה ההבדל בין resolve ל- reject ?
- מה נצטרך לשלוח לבנאי של Promise ?
- באיזה מצבים נחדקק להשתמש בטכניקה זו?
- מה הם המצבים האפשריים של Promise ?
- מה הם המאפיינים והמתודות המיוחדים למחלקת Promise ?

JS_Advanced_Promise.html

```
let myPromise = new Promise(function (funcValue, funcError) {
    console.log("Connecting to the server, reply in ??? milliseconds");
    const url = 'https://pokeapi.co/api/v2/pokemon/squirtle';
    const request = new XMLHttpRequest();
    request.open("GET", url);
    request.onload = function () {
        if (request.readyState == 4 && request.status == 200) {
            funcValue(request.response);
        } else {
            funcError("Error !!!!");
        }
    };
    request.send();
    console.log("end");
});

myPromise.then(
    function (value1) { console.log(value1); },
    function (value2) { console.log(value2); }
);
```

מחלקת Promise

בחלק זה נלמד מגוון נושאים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה זה Promise?
- מה ההבדל בין resolve ל- reject ?
- מה נצטרך לשלוח לבנאי של Promise ?
- באיזה מצבים נזדקק להשתמש בטכניקה זו?
- מה הם המצבים האפשריים של Promise ?
- מה הם המאפיינים והמתודות המיוחדים למחלקת Promise ?

JS_Advanced_Promise.html

שם פונקציה 1 | שם פונקציה 2

```
let myPromise = new Promise(function (myInnerValue, myInnerError) {
    const uPass = prompt("Enter Password");
    console.log("Connecting to the server, reply in 3000 milliseconds")
    setTimeout(function () {
        if (uPass == "123") {
            myInnerValue("Welcome");
        } else {
            myInnerError("User NOT Welcome");
        }
    }, 3000)
});

myPromise.then(
    function (value) { console.log(value); alert(value); },
    function (error) { console.log(error); alert(error); }
);
```

הכנסת ערך לפונקציה 1

הכנסת ערך לפונקציה 2

ערך שהתקבל מפונקציה 2

ערך שהתקבל מפונקציה 1

מחלקת Promise

בחלק זה נלמד מגוון נושאים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה זה Promise?
- מה ההבדל בין resolve ל- reject ?
- מה נצטרך לשלוח לבנאי של Promise ?
- באיזה מצבים נחדקק להשתמש בטכניקה זו?
- מה הם המצבים האפשריים של Promise ?
- מה הם המאפיינים והמתודות המיוחדים למחלקת Promise ?

JS_Advanced_Promise.html

```
let myPromise = new Promise(function (funcValue, funcError) {
    console.log("Connecting to the server, reply in ??? milliseconds");
    const url = 'https://pokeapi.co/api/v2/pokemon/squirtle';
    const request = new XMLHttpRequest();
    request.open("GET", url);
    request.onload = function () {
        if (request.readyState == 4 && request.status == 200) {
            funcValue(request.response);
        } else {
            funcError("Error !!!!");
        }
    };
    request.send();
    console.log("end");
});

myPromise.then(
    function (value1) { console.log(value1); },
    function (value2) { console.log(value2); }
);
```

מחלקת Promise

בחלק זה נלמד מגוון נושאים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה זה Promise?
- מה ההבדל בין resolve ל- reject ?
- מה נצטרך לשלוח לבנאי של Promise ?
- באיזה מצבים נדרק להשתמש בטכניקה זו?
- מה הם המצבים האפשריים של Promise ?
- מה הם המאפיינים והמתודות המיוחדים למחלקת Promise ?

JS_Advanced_Promise.html

```
let myPromise = new Promise(function (funcValue, funcError) {

    console.log("Connecting to the server.. ");

    const request = new XMLHttpRequest();
    request.onload = function () {
        if (request.readyState == 4 && request.status == 200) {
            funcValue(request.response);
        } else {
            funcError("Error !!!!");
        }
    };

    request.open("GET", 'https://pokeapi.co/api/v2/pokemon/squirtle');
    request.send();
    console.log("end");
});

myPromise.then(
    function (value1) { console.log(value1); },
    function (value2) { console.log(value2); }
);
```

פרמטרים בברת מחדל בפונקציות

- יש אפשרות להגדיר פרמטרים ברירת מחדל בפונקציות ע"י הוספת שווה וערך מסויים
- const defaultParams = (a1, a2 = 5) => a1 + a2;

- defaultParams(1, 1); // 1+1
- defaultParams(1); // 1+5

שימוש ב then - catch

שימוש ב then - catch

בחלק זה נלמד מגוון נושאים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- למה משמשת המתודה then?
- למה משמשת המתודה catch?
- איזה פרמטרים ניתן לשלוח למתודה then?
- איך ניתן להשתמש ב-thenלמטרות סנכרון?
- איזה פרמטרים ניתן לשלוח למתודה catch?
- מה היתרון בשימוש ב-catch?
- איך נוכל למשוך נתונים מ-API באמצעות שימוש ב- Promise ו-XMLHttpRequest?
- מה נוכל לעשות במקרה של שגיאה שמקורה בשרת?

JS_Advanced_Promise_Then_Catch.html

JS_Advanced_PromiseAll.html

```
function getPromise(url) {
    let myPromise = new Promise(function (resolve, reject) {
        let request = new XMLHttpRequest();
        request.open("GET", url);
        request.onload = () => {
            if (request.status == 200) {
                resolve(request.response);
            } else {
                reject("Error");
            }
        };
        request.send();
    });
    return myPromise;
}

const pokemonsUrl = 'https://pokeapi.co/api/v2/pokemon?limit=20';
let promise = getPromise(pokemonsUrl);

const promiseHandler = () => {
    promise.then((result) => {
        let pokemonsArray = [...JSON.parse(result)['results']]
        pokemonsArray.forEach(item => console.log(item['name']));
    }).catch((error) => {
        console.log(error.message); // The code will run only if there is an error
    }).then(() => {
        console.log('\n this will be printed at the end of program, bye')
    });
}

promiseHandler();
```

Async and Await

Async and Await

JS_Advanced_Async_Await.html

בחלק זה נלמד מגוון נושאים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה התחביר של async ו-await ?
- מה הם הכללים לשימוש?
- מדוע פונקציה חייבת להיות מוגדרת כאסינכרונית על מנת שנוכל לממש בה await?
- מה זה בא לפתור לנו?
- מה עלינו לעשות על מנת שנוכל לאפשר מימוש של awaitב- top level של המסמך?
- מתי עלינו להצהיר על מסמך כמודול?

```
async function myFunction1() {
    return "Hola Class";
}

myFunction1().then(
    function (value) { console.log(value); },
    function (error) { console.log(error); }
);


function myFunction2() {
    return Promise.resolve("Hola Class");
}

myFunction2().then(
    function (value) { console.log(value); },
    function (error) { console.log(error); }
);
```

Async and Await

בחלק זה נלמד מגוון נושאים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה התחביר של async ו-await ?
- מה הם הכללים לשימוש?
- מדוע פונקציה חייבת להיות מוגדרת כאסינכרונית על מנת שנוכל לממש בה await?
- מה זה בא לפתור לנו?
- מה עלינו לעשות על מנת שנוכל לאפשר מימוש של awaitב- top level של המסמך?
- מתי עלינו להצהיר על מסמך כמודול?

JS_Advanced_Async_Await1.html

הגדרת פונקציה אסינכרונית

```
var x = 1;

async function connectToServerDemo1() {

    let myPromise = new Promise(function (funcResolve, funcReject) {

        console.log("Connecting to the server, reply in 3000 milliseconds");
        setTimeout(function () {
            if (x == 1) {
                funcResolve("Connecting success");
                x = 0
            } else {
                funcReject("Connecting error");
                x = 1
            }
        }, 3000)
    });
    return await myPromise;
}

function demo1Option1() {
    connectToServerDemo1().then(
        function (value) { console.log("OK", value); },
        function (error) { console.log("error", error); }
    );
}
```

הגדרת המתנה לתשובה

קריאה לפונקציה המצריכה המתנה

Async and Await

בחלק זה נלמד מגוון נושאים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה התחביר של async ו-await ?
- מה הם הכללים לשימוש?
- מדוע פונקציה חייבת להיות מוגדרת כאסינכרונית על מנת שנוכל לממש בה await?
- מה זה בא לפתור לנו?
- מה עלינו לעשות על מנת שנוכל לאפשר מימוש של awaitב- top level של המסמך?
- מתי עלינו להצהיר על מסמך כמודול?

JS_Advanced_Async_Await2.html

הגדרת הסקריפט כמודול

הגדרת פונקציה אסינכרונית

```
<script type="module" >

    const URL = 'http://universities.hipolabs.com/search?country=Israel';

    async function getPromise(url) {
        let myPromise = new Promise((resolve, reject) => {
            let request = new XMLHttpRequest();
            request.open("GET", url);
            request.onload = () => {
                if (request.status == 200) {
                    return resolve(request.response);
                } else {
                    return reject(error.message);
                }
            };
            request.send();
        })
        return await myPromise;
    }

    let promise2 = await getPromise(URL);
    console.log({ ...JSON.parse(promise2) });

</script>
```

הגדרת המתנה לתשובה

קריאה לפונקציה המצריכה המתנה

תרגילון

- יש ליצור פונקציה
- יש להציג את התוצאה של הpromise בconsole

```
const alwaysSuccessPromise = () => {
  return new Promise((resolve, reject) => {
    resolve("yey");
    // reject("error from promise");
  });
};
```

סיכום שיעור

אז מה למדנו היום?

- Promise
- Async await
- Default params

שיעורי בית

- יש ליצור promise בתוך פונקציה
- הפונקציה תקבל 2 פרמטרים (מספרים)
- ה-promise יחבר את 2 המספרים ויצליח אם התוצאה זוגית אחרת לא יצליח

- יש להפעיל את ה-promise בשיטה החדשה בתוך פונקציה (async await)

- יש ליצור פונקציה שמקבלת כפרמטר מערך של מספרים
- בתוך הפונקציה יש ליצור promise שיחפש את המספר 5
- אם מצא אז הצליח אחרת לא הצליח

- *יש ליצור פונקציה שמקבלת כפרמטר מערך של משתמשים כולל תאריך לידה
- בתוך הפונקציה יש ליצור promise שיחשב את הגיליים,
- אם הצליח להמיר את כל הגילאים אז יצור מערך של משתמשים אם הגיליים
- אחרת אם לא הצליח להמיר לפחות תאריך אחד אז הpromise נכשל

תודה על ההקשבה

אני וצוות המכללה כאן עבורכם

לשלוח לי:

איך היה השיעור מ-1 עד 10

איך היה הקצב מ-1 עד 10

😄