JS Advanced

16/12/2024

אופרטור Spread

אופרטור Spread

JS_Advanced_Spread_Operator.html

בחלק זה נכיר אופרטור אשר מאפשר לבצע שימוש נוח ויעיל באובייקטים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה תפקידו של אופרטור spread?
- איך מגדירים אופרטור spread (...)?
- אילו שימושים נפוצים אנו מכירים?
- מה זה rest parameter?
- האם משנה מיקומו של ה-rest parameter בסדר מיקומי הארגומנטים הנשלחים לפונקציה?
- מה הם היתרונות בשימוש?
- כיצד משתמשים ה spread על אובייקט?

```
let myArray = [11, 22, 33, 44, 55];
let newArray;

newArray = myArray;

myArray[0] = 100;
console.log(newArray);
console.log(myArray);

newArray = [...myArray];

myArray[0] = 200;
console.log(newArray);
console.log(myArray);
```

מערך מועבר
by Reference

מערך מועבר
by Value

אופרטור Spread

JS_Advanced_Spread_Operator.html

בחלק זה נכיר אופרטור אשר מאפשר לבצע שימוש נוח ויעיל באובייקטים.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה תפקידו של אופרטור spread?
- איך מגדירים אופרטור (...) spread?
- אילו שימושים נפוצים אנו מכירים?
- מה זה rest parameter?
- האם משנה מיקומו של ה-rest parameter בסדר מיקומי הארגומנטים הנשלחים לפונקציה?
- מה הם היתרונות בשימוש?
- כיצד משתמשים ה spread על אובייקט?

```
let myArray1 = [11, 22, 33, 44, 55];
function print(num1, ...myArr){
    console.log(num1);
    console.log(myArr)
}

print(100, 10, 20, 10, 30);

print(100, myArray1);
```

פונקציה המקבלת rest parameter

קריאה לפונקציה rest parameter

פונקציה המקבלת rest parameter

תרגול אופרטור Spread

צרו קובץ חדש בשם **JS_spread** לטובת הנושא ופתרו את התרגילים לפי הסדר
חשוב להדפיס הודעות הצלחה ושגיאה למשתמש וכמובן לבדוק כל תרגיל בסיום.

תרגיל	תיאור המשימה
Ex-1	צרו מערך בשם cart1 וכערך תנו לו רשימת מחירי מוצרים. הדפיסו את המערך לקונסול, פעם אחת עם שימוש ב-spread, ופעם אחת ללא שימוש, הבחינו בהבדל של הפלט בקונסול בין שתי ההדפסות.
Ex-2	צרו מערך נוסף בשם cart2, כערך תנו לו את cart1. הדפיסו את שני המערכים לקונסול.
Ex-3	שנו את אחד המחירים במערך cart1. הדפיסו לקונסול את שני המערכים, מי מהם השתנה?
Ex-4	אפסו את הערך של cart2 וכערך תנו לו את רשימת המחירים שב-cart1, רק שהפעם השתמשו ב-spread, לאחר מכן הדפיסו את שני המערכים לקונסול.
Ex-5	שנו את אחד המחירים במערך cart1. הדפיסו לקונסול את שני המערכים, מי מהם השתנה?

cart1	[4,30,76,100]
cart2	[cart1]

תרגול אופרטור Spread

צרו קובץ חדש בשם **JS_spread_object** לטובת הנושא ופתרו את התרגילים לפי הסדר חשוב להדפיס הודעות הצלחה ושגיאה למשתמש וכמובן לבדוק כל תרגיל בסיום.

תרגיל	תיאור המשימה
Ex-1	צרו אובייקט ליטרלי בשם **person** וכערכים תנו לו שם פרטי, שם משפחה וכתובת אימייל.
Ex-2	צרו אובייקט נוסף בשם **personCopy** כערך תנו לו את **person**. הדפיסו את שני המערכים לקונסול.
Ex-3	שנו את האימייל של **personCopy**. הדפיסו לקונסול את שני האובייקטים, מי מהם השתנה?
Ex-4	צרו אובייקט ליטרלי חדש בשם **newPerson** וכערך תנו לו את **person**, רק שהפעם עשו זאת באמצעות שימוש ב-spread.
Ex-5	שנו את השם הפרטי של **newPerson**. הדפיסו לקונסול את שני המערכים, מי מהם השתנה?

Person {}

firstName	"Gal"
lastName	"Lavi"
email	"gal@email.com"

Shallow copy & Deep copy

מחבר: גל לביא

הכרות עם Shallow Copy & Deep Copy

JS_Advanced_Shallow_And_Deep.html

בחלק זה נכיר אפשרויות להעתקת אובייקטים ונעמוד על ההבדלים בין האפשרויות.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- מה ההבדל בין עותק "חלול" לבין עותק "מדויק"?
- במה אופרטור spread יכול לעזור לנו בהקשר זה?
- מה היא כתובת זיכרון ומהו פוינטר?
- מה ההבדל בין Value Type ל- Reference Type ?
- מה היתרונות של כל שיטה?

```
let myArray1 = [11, 22, 33, 44, 55];
let myArray2;

myArray2 = [...myArray1, 66, 77, 88, 99];
console.log(myArray2);

myArray3 = [myArray1, 66, 77, 88, 99];
console.log(myArray3);

myArray1[0] = 100;
console.log(myArray2);
console.log(myArray3);
```

Shallow Copy
אובייקט מועבר by Value

Deep Copy
אובייקט מועבר by Reference

myArray2

```
0: 11
1: 22
2: 33
3: 44
4: 55
5: 66
6: 77
7: 88
8: 99
```

myArray3

```
0: (5) [100, 22, 33, 44, 55]
1: 66
2: 77
3: 88
4: 99
```

תרגול Shallow copy & Deep copy

צרו קובץ חדש בשם **JS_shallow_deep_copy** לטובת הנושא ופתרו את התרגילים לפי הסדר חשוב להדפיס הודעות הצלחה ושגיאה למשתמש וכמובן לבדוק כל תרגיל בסיום.

תרגיל	תיאור המשימה
Ex-1	צרו 2 מערכים: cart1 ו-cart2, תנו להם ערכים -4 איברים המסמלים מחירי מוצרים. צרו מערך חדש בשם carts וכערך תנו לו את cart1 ו-cart2. הדפיסו את carts לקונסול.
Ex-2	שנו אחד מהאיברים של המערך השני שנמצא בתוך carts. האם אותו האיבר השתנה גם במערך cart2?
Ex-3	אפסו את המערך carts וכערך תנו לו שני מערכים. פזרו בכל אחד מהמערכים את האיברים של cart1 ו-cart2 בהתאמה באמצעות שימוש ב-spread. הדפיסו את carts לקונסול.
Ex-4	שנו אחד מהאיברים של המערך השני שנמצא בתוך carts. הדפיסו לקונסול את carts ואת cart2. האם אותו האיבר השתנה גם במערך cart2?
Ex-5	שנו את אחד המחירים במערך cart1. הדפיסו לקונסול את carts ואת cart1. האם אותו האיבר השתנה השתנה גם במערך הראשון של carts?

cart1	[4,30,76,100]
cart2	[1,600,56,60]

carts	[[cart1],[cart2]]

שימוש ב try & catch

שימוש ב try & catch

JS_Advanced_Try_Catch.html

בחלק זה נלמד כיצד לטפל ולתפוס בעיות בקוד ולטפל בהן, מה שיאפשר לקוד להמשיך לרוץ.
בסיום הנושא תוכלו לענות על השאלות הבאות:

- ממה יכולות להיגרם שגיאות ב-JS?
- אילו שגיאות יכולות להתרחש בזמן ריצת התוכנית שלא ניתן לצפות מראש?
- מה נוכל לעשות על מנת לטפל בשגיאות בזמן ריצה?
- מה הוא בלוק ה-try catch ?
- מה נוכל לעשות במקרה שתפסנו שגיאה באמצעות בלוק try-catch?
- מה תפקידו של finally?

```
const text = "Hola Class";

// demo 1
try{
    console.log(someText);
}catch( error ){
    console.log(error);
    console.log(error.message);
}finally{
    console.log("Finish 1");
}

// demo 2
console.log(someText);
console.log("Finish 2");
```

מנסה לבצע את הפעולה

קוד ירוץ במידה ותהיה שגיאה

קוד ירוץ בכל מקרה בסיום

דוגמה לקוד שיוציא שגיאה

```
Uncaught ReferenceError: someText is not defined
    at JS_Advanced_Try_Catch.html:21:21
```

תרגול שימוש ב-try & catch

צרו קובץ חדש בשם **JS_Try_Catch** לטובת הנושא ופתרו את התרגילים לפי הסדר
חשוב להדפיס הודעות הצלחה ושגיאה למשתמש וכמובן לבדוק כל תרגיל בסיום.

תרגיל	תיאור המשימה
Ex-1	צרו פונקציה חדשה וקראו לה divide. פונקציה זו תדע לקבל 2 פרמטרים – num1 ו-num2.
Ex-2	הפונקציה divide תיקח את המספרים ותבצע פעולת חילוק מתמטית ביניהם ותחזיר את התוצאה. נסו את הפונקציה באמצעות שליחת ערכים שונים בכל פעם והדפסת התוצאה החוזרת.
Ex-3	נסו לשלוח אל הפונקציה משתנים אשר טרם נוצרו.
Ex-4	על מנת למנוע את השגיאה המתעוררת משליחת הערכים האחרונה שביצעתם, השתמשו ב-try ו-catch כדי לתפוס את השגיאה ולהדפיס לקונסול הודעה תואמת.
Ex-5	צרו משתנה חדש בשם finalMessage. השתמשו ב-finally על מנת להדפיס הודעה לקונסול שהטיפול הסתיים, השתמשו במשתנה finalMessage על מנת להציג הודעה תואמת, בסוף של ה-finally אפסו את הערך של המשתנה finalMessage

divide

num1	50
num2	5

תודה על ההקשבה

אני וצוות המכללה כאן עבורכם