שיעור 8 – פונקציות אסינכרוניות בג'אווהסקריפט

# 🧠 סיכום השיעור

מטרת השיעור: להבין כיצד ג'אווהסקריפט מבצעת פעולות שלא מתבצעות מיד, אלא:  
- לאחר זמן מסוים (setTimeout)  
- באופן חוזר (setInterval)  
- בזמן שממתינים לתשובה משרת (fetch)  
  
טבלת סיכום הכלים:  
  
| כלי | תיאור קצר |  
|------------------|-------------------------------------------|  
| setTimeout | לבצע פעולה אחת לאחר זמן מסוים |  
| setInterval | לבצע פעולה חוזרת כל כמה זמן |  
| fetch | להביא מידע מאינטרנט |  
| Promise | אובייקט שמייצג פעולה עתידית (מבטיח תשובה) |  
| then | פעולה שמבוצעת אם ההבטחה הצליחה |  
| catch | פעולה שמבוצעת אם ההבטחה נכשלה |  
| Closure | משתנה שנשמר בזיכרון גם אחרי סיום הפונקציה |  
| Garbage Collector| מנקה משתנים שלא בשימוש בזיכרון |

# 📝 תרגול – שאלות ותרגילים

## חלק א – שאלות הבנה:

* 1. מה ההבדל בין setTimeout ל־setInterval?
* 2. מה התפקיד של then בתוך fetch?
* 3. מתי משתמשים ב־catch?
* 4. מה זה Promise ומהם שלושת המצבים האפשריים שלו?

## חלק ב – תרגול קוד:

1. שאלה 1: מה יודפס במסך?  
     
   setTimeout(() => {  
    console.log("שלום");  
   }, 1000);  
   console.log("מה קורה?");
2. שאלה 2: מה חסר בקוד הבא?  
     
   fetch("https://api.example.com/data")  
    .then(response => \_\_\_\_\_\_\_\_)  
    .then(data => {  
    console.log(data);  
    })  
    .catch(error => {  
    console.error("הייתה שגיאה", error);  
    });
3. שאלה 3: השלם את ה־Promise כך שיחזיר 'הצלחה!' אם x גדול מ־10, ו־'נכשל' אחרת.  
     
   const x = 15;  
     
   const promise = new Promise((resolve, reject) => {  
    if (\_\_\_\_\_\_\_\_\_\_) {  
    resolve("הצלחה!");  
    } else {  
    reject("נכשל");  
    }  
   });  
     
   promise  
    .then(msg => console.log(msg))  
    .catch(err => console.log(err));

# ✅ פתרונות

## חלק א – פתרונות:

* 1. setTimeout – פעולה חד־פעמית אחרי זמן; setInterval – פעולה שחוזרת כל כמה זמן.
* 2. then עוזר להפעיל קוד אחרי שהמידע חזר מהשרת.
* 3. כשמשהו משתבש בבקשה לשרת – catch תופס את השגיאה.
* 4. Promise הוא אובייקט שמייצג פעולה עתידית, ויש לו שלושה מצבים: pending (ממתין), fulfilled (הצליח), rejected (נכשל).

## חלק ב – פתרונות:

1. שאלה 1 – הפלט יהיה:  
   מה קורה?  
   שלום
2. שאלה 2 – תשובה:  
   .then(response => response.json())
3. שאלה 3 – פתרון:  
     
   const x = 15;  
     
   const promise = new Promise((resolve, reject) => {  
    if (x > 10) {  
    resolve("הצלחה!");  
    } else {  
    reject("נכשל");  
    }  
   });  
     
   promise  
    .then(msg => console.log(msg))  
    .catch(err => console.log(err));