# פרק 13 – היכרות עם מערכים

## הקדמה

מערכים ב PHP כמו ב ActionScript מאפשרים לנו לשמור מספר רב של ערכים במן ארונית של משתנים שלכולם אותו השם. התחביר להגדרה ושימוש במערך דומה לזה של ActionScript.

הפרק יכיל מספר רב של דוגמאות ותרגולים בכדי להבהיר את העבודה עם מערכים ועל הדרך גם את העבודה עם לולאות עליהן למדנו בפרקים הקודמים.

## הגדרה ושימוש במערכים ב PHP

הגדרת מערך ב PHP דומה להגדרת מערך ב ActionScript, ההבדל הוא שב PHP אין צורך במילת המפתח new בשלב ההגדרה.

בכדי להגדיר מערך ב PHP יש לציין את שמו (לא לשכוח סימן $ שהרי מדובר במשתנה), לאחר מכן יש לכתוב את סימן השוויון מלווה במילה השמורה array. הנה הגדרה של מערך ב PHP:

```
$friends = array();
```

בדוגמא הגדרנו מערך חדש בשם $friends. למערך החדש שלנו אין בכלל איברים. בכדי לאתחל מערך כך שיהיו לו איברים כבר בהגדרתו כל שצריך לעשות הוא להזין את הערכים הרצויים, בדוגמא הבאה הזנתי שמות של כמה מחברי:

```
$friends = array("orit", "shmuel", "hanan", "dotan");
```

ניתן לראות שהערכים מופרדים זה מזה בעזרת סימן הפסיק (,).

כיצד מושכים את ערכו של איבר מסוים מהמערך? ובכן בדיוק כמו ב ActionScript משתמשים בסוגריים מרובעים ([]) כשבתוכם ישנו האינדקס של האיבר הרצוי, מערכים ב PHP גם הם מבוססי אינדקס אפס, כלומר האינדקס של האיבר הראשון הוא לא 1 אלא 0. הקוד הבא יציג את המחרוזת PHP לדפדפן:

```
$prog_lang = array("PHP", "Java", "ActionScript");
echo $prog_lang[0];
```

כל הזכויות שמורות לגיל כהן, http://www.flashoo.co.il

דוגמא נוספת בכדי לוודא שאתה משופשף עם מערכים, הבט בקוד הבא -

```
$nums = array($a, 3+$a, 2+2*$a);

echo $nums[0]+$nums[1]-$nums[2];
```

בקוד זה הגדרנו מערך בשם $nums שמכיל שלושה איברים, ערך האיבר הראשון הוא כערך של משתנה שערכו לא ידוע בשם $a, ערך האיבר השני ערכו גדול ב-3 מערך האיבר הראשון, וערך האיבר השלישי הוא פעמיים האיבר הראשון ועוד 2.

לכאורה מבלי לדעת את ערכו של $a אין לנו דרך לדעת מה יוצג לדפדפן, אך אם נביט טוב לגבי מה מועבר ל echo נראה שהביטוי המועבר הוא סכום האיבר הראשון והשני ב $nums פחות האיבר השלישי. נניח וערכו של $a הוא $x$. נביט בערך שמועבר ל echo:

$$x+(3+x)-(2+2x)=1$$

כל הביטוי, ערכו 1 וזאת מפני שה- $x$ מהאיבר הראשון פלוס ה-$x$ מהאיבר השני מתקזזים עם זוג ה-$x$-ים מהאיבר השלישי.

לצורך תרגול הבט בקוד הבא וחשוב מה יהיה הפלט לדפדפן. (ערך המשתנה $x אינו ידוע).

```
$a = array($x,$x,26-$x);
$b = array($a[0], $a[0]+$a[1], $a[0]+$a[1]+$a[2]);

echo ($b[2]-$b[1]+$b[0]);
```

# הוספת איברים למערך ושינוי ערכי איברים קיימים

ניתן להוסיף איברים חדשים למערך או לשנות ערכי איברים קיימים בכל נקודה בקוד בה המערך מוגדר. אין לנו צורך אפילו צורך לציין את האינדקס הריק הבא במערך כי PHP עוקבת אחר מיקום זה. הבט בקוד הבא:

```
$books = array ("Fermat's last theorm", "All my sons");
$books[] = "The Da Vinci Code";

echo $books[2];
```

כל הזכויות שמורות לגיל כהן, http://www.flashoo.co.il

בשורה הראשונה הגדרנו מערך בשם $books ואתחלנו עבורו שני איברים מסוג מחרוזת, אלו אגב שמות ספרים.

בשורה השנייה אנו מוסיפים לסוף המערך, כלומר לתא השלישי איבר חדש, ספר טוב שקראתי לאחרונה – "The Da Vinci Code" (מומלץ!). השאלה היא איך PHP יודעת להוסיף את המחרוזת לתא השלישי? ובכן PHP יודעת מהו האיבר הריק הבא ומכניסה את המחרוזת למקום הנכון. התנהגות כזו של מערך ב PHP – העובדה שאפשר להוסיף לסוף המערך מבלי לציין את האינדקס – מזכיר את ההתנהגות של מחסנית.

בכדי לשנות את ערכו של איבר במערך כל שיש לעשות הוא להזין ערך חדש לאינדקס הנכון. בדוגמא הקודמת כתבתי את המילה theorem עם טעות כתיב (theorm). הקוד הדרוש בכדי לתקן זאת יראה כך:

```
$books[0] = "Fermat's last theorem";
```

# דוגמא - מספרי פיבונאצ'י חוזרים

ננתח את הקוד הבא בכדי לוודא שהבנת כיצד להוסיף ולשנות ערכי איברים במערך:

```
$fib = array(1,1);

for($i=0; $i<8; $i++)
{
     $fib[] = $fib[$i]+$fib[$i+1];
}

for($i=0; $i<10; $i++)
{
     echo "$fib[$i] ";
}
```

המטרה של התוכנית היא להציג את עשרת מספרי פיבונאצ'י הראשונים. במספרי פיבונאצ'י נתקלנו כבר באחד הפרקים הקודמים, רק אזכיר שמדובר בסדרה של מספרים שהאיברים הראשון והשני שלה, ערכם הוא אחד וכל איבר אחר בסדרה – ערכו כסכום שני האיברים שלפניו. ההבדל בין הקוד הזה המציג את מספרי פיבונאצ'י הוא שהפעם הערכים מאוחסנים במערך ונוכל לבצע עליהם כל פעולה שנרצה מבלי לחשב אותם בכל פעם מחדש.

בשורה הראשונה הגדרנו מערך חדש בשם $fib ואתחלנו את ערכי שני האיברים הראשונים ל-1. מיד לאחר מכן אנחנו נכנסים ללולאות for, אשר מתבררת כמאוד שימושית בעבודה עם מערכים. הלולאה מתבצעת 8 פעמים בכדי לחשב את 8 האיברים הנותרים של הסדרה. בגוף הלולאה יש

כל הזכויות שמורות לגיל כהן, http://www.flashoo.co.il

שורת קוד אחת המוסיפה לסוף המערך איבר חדש. ערכו של האיבר החדש נקבע על פי סכום שני האיברים שלפניו, בדיוק כפי שהסדרה מוגדרת מבחינה מתמטית.

ביציאה מהלולאה אנו נתקלים בלולאה נוספת שרצה הפעם 10 פעמים, גם היא על אינדקס בשם $i. תפקידה של לולאה זו הוא להציג את האיברים אחד אחרי השני לדפדפן. פלט התוכנית נראה כך:

*1 1 2 3 5 8 13 21 34 55*

# מספר האיברים במערך

PHP מאפשרת לנו לדעת כמה איברים יש לנו במערך בעזרת פונקציה בשם count. הפונקציה מקבל כפרמטר שם של מערך ומחזירה את מספר האיברים בו. הבט בקוד הבא:

```
$friends = array("orit", "shmuel", "hanan", "dotan");

echo "You have ". count($friends). " friends.";
```

בקוד אנו נתקלים שוב במערך $friends, הפעם המערך מלווה בפקודת echo שמדפיסה את מספר החברים שלי. שים לב לשימוש בפונקציה count המקבלת כפרמטר את המערך $friends, הפונקציה כאמור מחזירה את מספר האיברים בו. לכן פלט התוכנית יהיה:

*You have 4 friends.*

הפונקציה count שימושית כאשר עלינו לעבור על כל איברי המערך. לדוגמא, התוכנית הבאה מסכמת את כל האיברים במערך שערכם זוגי:

```
$nums = array(3,1,6,4,9,10,6);

for($i=0, $sum=0; $i<count($nums); $i++)
{
     if ($nums[$i]%2 == 0) $sum += $nums[$i];
}

echo $sum;
```

בשורה הראשונה הגדרנו מערך בשם $nums ואתחלנו מספר איברים. בשורה השנייה יושבת לולאת for המאתחלת את האינדקס $i ואת הסכום המצטבר לאפס. תנאי הלולאה הוא שהאינדקס קטן ממספר האיברים של המערך $nums, את מספר האיברים אנו מושכים בעזרת count.

כל הזכויות שמורות לגיל כהן, http://www.flashoo.co.il

בגוף הלולאה ישנו משפט התניה שבודק אם האיבר הנוכחי הנבדק זוגי (הוא בודק אם השארית בחלוקה ל-2 היא אפס), אם כן ערכו של האיבר מתווסף ל $sum. אחרי הלולאה ישנה פקודת echo המציגה לדפדפן את הסכום. במקרה הזה הסכום הוא 26.

הנה דוגמא נוספת, הבודקת האם קיים איבר במערך שערכו שווה לאינדקס שלו, לדוגמא האם קיים איבר שערכו 3 והוא יושב בתא מספר 3. אם קיים איבר כזה משתנה בוליאני בשם $exist יקבל ערך true, אחרת המשתנה $exist יקבל ערך false. הנה הקוד:

```
$nums = array(2,1,5,7);

for($i=0, $exist=false; $i<count($nums); $i++)
{
     if ($nums[$i] == $i)
     {
          $exist = true;
          break;
     }
}
```

בשורה הראשונה הגדרנו מערך. מהסתכלות עליו אנו רואים ש $exist יהיה true בסוף התהליך כי בתא מס' 1 (התא השני) ישנו הערך 1.

בשורה השנייה יושבת לולאה שמאתחלת את האינדקס לאפס ואת המשתנה הבוליאני $exist ל false, בכך אנו בעצם אומרים – עדיין לא מצאנו איבר כזה.

בגוף הלולאה ישנו משפט התניה שבודק אם האיבר הנוכחי – ערכו שווה לאינדקס שלו, אם כן אז $exist מעודכן ל true כך אנו מצהירים שמצאנו איבר שכזה. בשורה אחרי אנו רואים את מילת המפתח break שמוכיחה את עצמה – אם מצאנו איבר כזה אין צורך להמשיך ולחפש ולכן אפשר לצאת מהלולאה גם אם עדיין לא סרקנו את כולה – באנו, מצאנו, הלכנו.

# מערכים רב-ממדיים

מערך משול לשורה של איברים. מערך דו-ממדי יכול להיות משול לטבלה, בכדי לגשת לאיבר יש לציין גם את המיקום האופקי וגם את המיקום האנכי שלו בטבלה – כלומר גם את מספר השורה וגם את מספר העמודה. במערך תלת-ממדי ומעלה אנו כבר מאבדים את האנלוגיה אבל הכוונה עדיין ברורה – בכדי לגשת לאיבר כלשהו אנו זקוקים למספר אינדקסים כמספר הממדים של המערך.

ב PHP נוח להסתכל על מערך רב-ממדי כעל מערך שבכל תא שלו יש לא מספר ולא מחרוזת אלא מערך אחר.

כל הזכויות שמורות לגיל כהן, http://www.flashoo.co.il

נושא המערכים הרב-ממדיים ב PHP גמיש יותר מאשר בשפות תכנות עליות אחרות כמו C. אני מוצא את זה קצת מטריד אבל מי שאינו מורגל מדי לתחביר של C יכול להביט בגמישות הזה מזווית ראיה חיובית.

הבט בקוד הבא:

```
$a = array(1,2,3);
$b = array(4,5,6);
$c = array(7,8,9);
$d = array($a,$b,$c);

echo $d[1][2];
```

בשלושת השורות הראשונות הגדרנו שלושה מערכים, כל אחד מהם מכיל שלושה מספרים. בשורה הרביעית הגדרנו מערך בשם $d שאיבריו הם המערכים שהוגדרו קודם לכן. בשורה האחרונה אנו מציגים את האיבר במערך הדו-ממדי $d אשר נמצא במיקום (1,2), כלומר בשורה מספר 1 (השורה השנייה) ובעמודה מספר 2 (העמודה השלישית). הפלט יהיה – 6.

אם נשנה את הערך של איבר כלשהו ב $d הדבר לא ישפיע על הערכים מהמערכים $a, $b או $c, כלומר הערכים של איברי המערכים מועתקים מהמערכים $a, $b ו- $c למערך $d. כמובן ששינוי ערך של איבר מאחד המערכים הללו גם הוא לא ישפיע על אף איבר במערך $d.

אין חובה שאורך כל המערכים יהיה זהה. הקוד הבא חוקי לגמרי:

```
$a = array (1);
$b = array (1,2,3);
$e = array ($a, $b,);
```

לצורך תרגול חשוב מה יהיה פלט הקוד הבא:

```
$a0 = array(1,2,3);
$a1 = array(5,3,4);
$a2 = array(7,0,8);
$a3 = array(6,9,9);

$M = array();

for($i=3; $i>=0; $i--)
{
     $s = "a$i";
     $M[] = $$s;
}

echo $M[1][2];
```

כל הזכויות שמורות לגיל כהן, http://www.flashoo.co.il

ניתן להגדיר, ובכך לחסוך זמן ומקום בזיכרון, מערך רב-ממדי באופן כזה שהגדרת המערכים המהווים איברים במערך הרב-ממדי יוגדרו ביחד עם המערך הגדול. הבט בקוד הבא:

```
$a = array ( array(1,5),
             array(2,4),
             array(7,3),
             array(9,5));

echo $a[2][0];
```

בדוגמא הגדרנו מערך בשם $a שאיבריו הם מערכים, הפעם בכדי לציין שאיבריו הם מערכים לא כללנו מערכים קודמים כאיברים אלא יצרנו מערכים בתוך הגדרת המערך הכולל. היתרון בדרך זו הוא שאנו לא מבזבזים זיכרון וזמן יקר. בדרך הקודמת בה ראשית הגדרנו את המערכים ואחר כך העתקנו את ערכם לתוך המערך החדש נשארנו עם מערכים שאין לנו צורך בהם, מה גם שהעתקת הערכים דורשת זמן, זמן מיותר במקרה הזה.

# דוגמא לסיכום – מיון ציונים

כדוגמא לסיום הפרק נסתכל על קוד הממיין ציונים. נניח כי נתון לנו מערך בן 1000 תאים המכיל בכל תא ציון של סטודנט, הציון הוא בן 0 ל-100. עוד נניח כי אנו מתבקשים למיין את המערך בן 1000 התאים. כפי שנראה בהמשך PHP מציעה פונקציות מיון קלות לתפעול אבל הפעם אנו נמיין את המערך בעצמנו ללא עזרת PHP לצורך התרגול.

הרעיון הוא לבנות מערך עזר בן 101 תאים. כל תא במערך העזר ייצג כמה פעמים ציון מסוים התקבל (יש 101 ציונים אפשריים כי 0 הוא גם ציון), אנו נעבור על כל 1000 הציונים בעזרת לולאת for ובכל פעם שנראה ציון נעלה את הערך של תא מערך העזר המתאים באחד – לדוגמא אם נתקלנו בציון 89 אז נוסיף 1 לערך הנמצא בתא 89 במערך העזר.

בסוף הלולאה העוברת על כל 1000 התאים אנו נעבור בלולאה על 100 התאים של מערך העזר ונדרוס את ערכי המערך המקורי בצורה הבאה – נתחיל מהאיבר הראשון בלולאת העזר ונבדוק כמה סטודנטים קיבלו את הציון אפס, הדבר קרה X פעמים – אז נכניס אפסים ל-X התאים הראשונים במערך המקורי. לאחר מכן נעבור לתא השני ונבדוק כמה סטודנטים קיבלו את הציון 1, ושוב אם מספר הסטודנטים שקיבלו ציון זה הוא Y אז נכנסים 1 Y פעמים למערך המקורי, כך נמשיך לאורך כל מערך העזר.

לפני שנתחיל נסתכל בדוגמא – במקום 1000 סטודנטים יש לנו 20 סטודנטים ובמקום ציון מ-0 ל-100 יש לנו רק מ-0 ל-10. נניח והמערך המקורי נראה כך:

*5, 3, 10, 10, 9, 5, 7, 8, 3, 0, 5, 6, 7, 7, 5, 6, 9, 10, 0, 7*

מערך העזר שלנו יראה כך:

*0 1 2 3 4 5 6 7 8 9 10 <- Indexes*
*2, 0, 0, 2, 0, 4, 2, 4, 1, 2, 3 <- Values*

המשמעות היא שישנם 2 סטודנטים שקיבלנו 0, אין סטודנטים שקיבלו 1 או 2, יש 2 סטודנטים שקיבלו את הציון 3, אין סטודנט שקיבל את הציון 4. את הציון 5 קיבלנו 4 סטודנטים, 2 סטודנטים קיבלו את הציון 6, ארבעה סטודנטים קיבלו את הציון 7, סטודנט אחד קיבל את הציון 8, 2 סטודנטים קיבלו את הציון 9 ושלושה את הציון הנכסף 10.

המערך המקורי לאחר המיון יראה כך:

*0, 0, 3, 3, 5, 5, 5, 5, 6, 6, 7, 7, 7, 7, 8, 9, 9, 10, 10, 10*

כל שעשינו היה לראות כמה פעמים יש כל ציון. הציון אפס התקבל פעמיים לכן ישנם שני אפסים בתחילת המערך, הציון 1 ו-2 לא התקבלו (או התקבלו אפס פעמים) ולכן הם לא מופיעים במערך. הציון 5 התקבל 4 פעמים ולכן הוא מופיע 4 פעמים, וכו'...

נעבור לקוד:

```
$grades = array( /* Given values */ );
$aux = array();

// Initialize all aux to values to zero
for($i=0; $i<=100; $i++)
{
     $aux[] = 0;
}

// Count grades into aux array
for($i=0; $i<1000; $i++)
{
     $aux[$grades[$i]]++;
}

// Rewrite array in sorted order
for($i=0, $k=0; $i<=100; $i++)
{
     for($j=0; $j<$aux[$i]; $j++)
     {
          $grades[$k] = $i;
          $k++;
     }
}
```

בשורה הראשונה של הקוד הגדרנו שני מערכים - $grades הוא המערך אותו אנו צריכים למיין, ההנחה היא כי יש 1000 ציונים בין 0 ל-100 במערך זה. שורה אחרי הגדרנו מערך עזר בשם $aux (מהמילה auxiliary, בעברית מסייע / עזר).

כל הזכויות שמורות לגיל כהן, http://www.flashoo.co.il

מכאן הקוד מתחלק לשלושה חלקים – החלק הראשון מאתחל את 101 האיברים של $aux לאפס, שהרי עוד לא ראינו אף ציון. הסיבה לכך שישנם 101 איברים היא, כפי שהוזכר, בגלל שגם הציון אפס הוא ציון קביל וגם הציון 100 הוא ציון קביל, לכן ישנם 101 ערכים שהרי אנו כוללים את הקצוות.

החלק השני הוא החלק בו מתבצעת הספירה לתוך $aux. אנו עוברים על כל 1000 האיברים של המערך $grades. הערך של התא הנוכחי ב $grades אותו אנו בודקים משמש לנו כאינדקס למערך $aux – לאיבר שנמצא באינדקס זה ב $aux אנו מוסיפים אחד – בכך אנו אומרים – ראינו עוד ציון שערכו הוא כערך הציון של הסטודנט הנוכחי - $grades[$i].

החלק השלישי מציג בפנינו זוג לולאות מקוננות. הלולאה הראשית עוברת על כל אחד מהתאים של $aux והפנימית אחראית לדרוס את הערכים של $grades כך שבסוף התהליך הלולאה תהיה ממוינת. שים לב שנעזרנו במשתנה $k המציין על איזה תא ב $grades אנו עובדים כרגע.

זוהי דוגמא לא פשוטה. אם עברת קורס כלשהו במדעי המחשב אתה בטח מכיר מיון זה כ"מיון דלי". מיון זה מהיר יותר ממיון סטנדרטי של PHP וזאת מפני שהוא מתבסס על ההנחה שהערכים אותם יש למיין מוגבלים, במקרה של ציונים הערכים אותם יש למיין מוגבלים בין 0 ל-100.

כתרגיל למחשבה – מה צריך לשנות בקוד בכדי שהמערך יהיה ממוין בסדר יורד במקום בסדר עולה?

# סיכום

עד כאן לפרק היכרות עם מערכים ב PHP. בפרק עברנו על הגדרת מערך, אתחול איברים, שינוי ערכי איברים והוספת איברים חדשים למערך מכל מקום בקוד. ראינו גם עד כמה מערכים ולולאות קשורים זה לזה. בפרק הבא נדון במערכים משויכים, בפרק שלאחר מכן נסתכל על פונקציות שימושיות לעבודה עם מערכים ב PHP.

כל הזכויות שמורות לגיל כהן, http://www.flashoo.co.il