

در این جلسه می‌خواهیم روش استفاده از آرایه‌ها را بررسی کنیم. آرایه مجموعه‌ای از متغیرها است که دارای یک نام مشترک هستند و توسط اندیس از یکدیگر متمایز می‌شوند. برای مثال، متغیر \$a را در نظر بگیرید:

\$a						
Index	0	1	2	3	4	5
Value	12	7	4	2	1	290

در مثال فوق، متغیر \$a یک آرایه با شش عنصر است که از 0 تا 5 شماره‌گذاری شده‌اند و مقادیر آنها به ترتیب برابر 12، 7، 4، 2، 1 و 290 است. حال فرض کنیم که می‌خواهیم محتویات خانه پنجم را ده برابر کنیم. برای این کار باید اینگونه بنویسیم:

```
$a[4]*=10;
```

تعریف یک آرایه در PHP

دقت کنید که برای تعریف آرایه در PHP، مشابه متغیرها، نوع آن ذکر نمی‌شود. بنابراین هر عنصر آرایه می‌تواند مقدار متفاوتی از هر نوع داده داشته‌باشد. ضمناً در PHP، طول آرایه نیز تعیین نمی‌شود. برای تعریف یک آرایه، کافی است نام آنرا همراه با کارکترهای [] ذکر کنیم:

```
$a[]=5;
```

بدین ترتیب، یک عنصر با مقدار 5 در آرایه درج می‌شود. در صورتی که آرایه قبلاً دارای مقدار باشد، یک عنصر به آن افزوده شده و مقدار جدید، در آن ذخیره می‌شود؛ لیکن اگر آرایه موردنظر قبلاً وجود نداشته‌باشد، ایجاد می‌شود و لذا، اندیس عنصر جدید، 0 خواهد بود. برای افزودن عناصر بعدی نیز باید به همین ترتیب عمل کنیم:

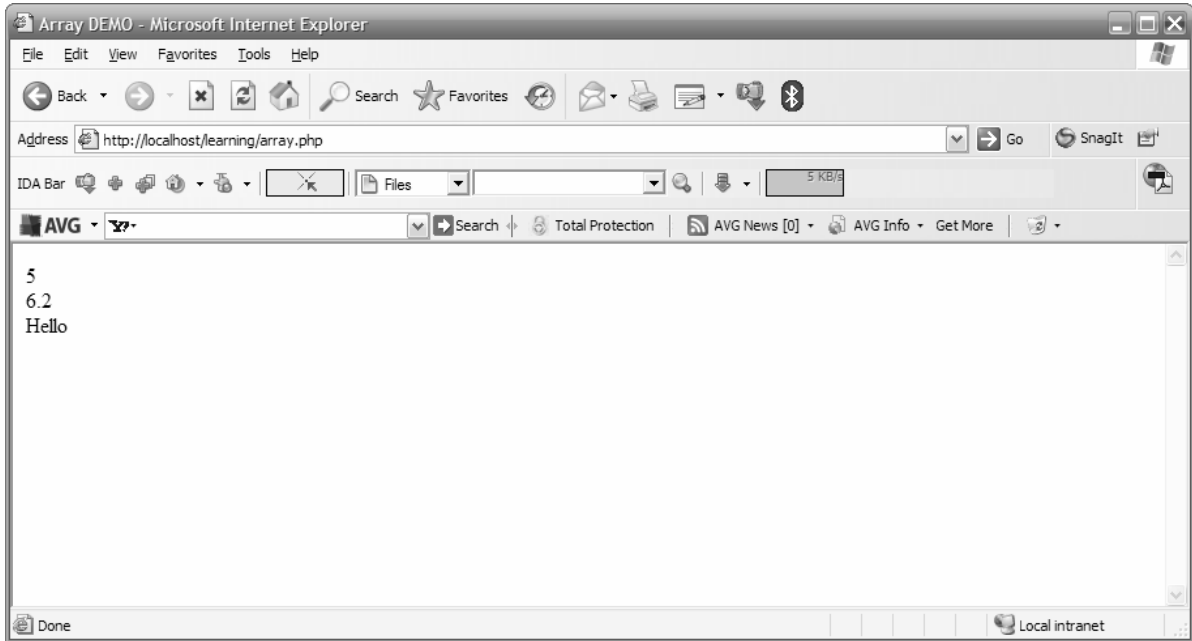
```
$a[]=6.2;
```

```
$a[]="Hello";
```

برای درک بهتر، به کد زیر دقت کنید که ضمن تعریف آرایه فوق، عناصر آنرا در خروجی چاپ می‌کند (شماره خطوط را ننویسید):

```
01 <HTML>
02 <HEAD>
03 <TITLE>Array DEMO</TITLE>
04 </HEAD>
05 <BODY>
06 <?php
07     $a[]=5;
08     $a[]=6.2;
09     $a[]="Hello";
10     for($i=0;$i<3;$i++)
11     {
12         echo ("{$a[$i]}<br/>\n");
13     }
14 ?>
15 </BODY>
16 </HTML>
```

```
array.php - Notepad
File Edit Format View Help
<HTML>
<HEAD>
<TITLE>Array DEMO</TITLE>
</HEAD>
<BODY>
<?php
    $a[]=5;
    $a[]=6.2;
    $a[]="Hello";
    for($i=0;$i<3;$i++)
    {
        echo ("{$a[$i]}<br/>\n");
    }
?>
</BODY>
</HTML>
```



همان طور که ملاحظه می کنید، در PHP علاوه بر آنکه تعداد عناصر آرایه قابل تغییر است، نوع هر عنصر نیز می تواند با سایر عناصر متفاوت باشد. این امر، کاملاً طبیعی است؛ زیرا در PHP، نوع عناصر مشخص نمی شود و مقادیر آنها می تواند دارای هر نوع دلخواه باشد.

روش های دیگر تعریف آرایه

روشی که بیان کردیم، یکی از روش های ساده تعریف آرایه است که در آن، هیچ کنترلی بر روی اندیس عناصر وجود ندارد و فقط می توان به ترتیب عناصر را درج کرده و سپس، مقدار هر عنصر را با ذکر اندیس آن، مورد دسترسی قرار داد. در ادامه، با چند روش برای مقداردهی عناصر یک آرایه آشنا می شویم که امکانات بیشتری را در اختیارمان قرار می دهند.

درج عناصر با ذکر شماره اندیس (اندیس عددی)

برای اینکه دقیقاً یک عنصر را در محل دلخواه از آرایه درج کنیم، به روش زیر عمل می کنیم:

```
$a[1]="PHP";
$a[3]=5;
$a[0]=6.2;
```

حال اگر توسط حلقه تکرار زیر، عناصر این آرایه را نمایش دهیم:

```
for($i=0;$i<4;$i++)
{
    echo("A[$i]=$a[$i]<br/>\n");
}
```

خروجی زیر تولید خواهد شد:

```
A[0]=6.2
A[1]=PHP
A[2]=
A[3]=5
```

همان طور که ملاحظه می کنید، عنصر $A[2]$ مقدار ندارد و لذا، در مقابل آن، چیزی نوشته نمی شود. با استفاده از روش فوق، می توانیم هر عنصر را دقیقاً در همان اندیس دلخواه، درج کنیم.



استفاده از اندیس غیر عددی

در PHP، برای عناصر مختلف یک آرایه، اجباری برای استفاده از اندیس عددی وجود ندارد. برای استفاده از اندیس‌های غیر عددی (مثلاً رشته‌ها)، به روش زیر عمل می‌کنیم:

```
$stateCapital["SistanAndBaluchestan"]="Zahedan";
$stateCapital["Tehran"]="Tehran";
$stateCapital["Fars"]="Shiraz";
```

در صورت استفاده از چنین آرایه‌ای، دیگر نمی‌توان از حلقه for برای نمایش مقادیر آن، استفاده کرد. هر چند روش‌های دیگری برای این کار وجود دارد که در ادامه همین جلسه، با آنها آشنا خواهیم شد. برای نمایش مقادیر آرایه فوق، به صورت زیر عمل می‌کنیم:

```
echo($stateCapital["SistanAndBaluchestan"]."<br/>\n");
echo($stateCapital["Fars"]."<br/>\n");
echo($stateCapital["Tehran"]."<br/>\n");
```

که خروجی دستورات فوق به صورت زیر خواهد بود:

```
Zahedan
Shiraz
Tehran
```

استفاده از تابع array

یک روش آسان برای تعریف یک آرایه و مقداردهی عناصر آن به طور هم‌زمان، استفاده از تابع array است. خروجی این تابع، یک آرایه است و مقادیری که باید در عناصر آرایه درج شوند را به عنوان پارامتر دریافت می‌کند:

```
$cities=array("Zahedan","Shiraz","Tehran");
```

بدین ترتیب، آرایه‌ای به صورت زیر تعریف خواهد شد:

```
$cities[0]="Zahedan";
$cities[1]="Shiraz";
$cities[2]="Tehran";
```

در روش‌های قبل مشاهده کردیم که فقط در اولین روش (ذکر نکردن اندیس)، شماره‌گذاری عناصر یک آرایه از صفر شروع می‌شود و در سایر روش‌ها، می‌توانیم شماره‌گذاری عناصر را از اندیسی به جز صفر شروع کنیم:

```
$a[1]=5;
$a[2]="Ali";
//OR
$family["Ali"]="Hoseini";
$family["Reza"]="Kamali";
```

برای اینکه در تابع array، اندیس هر عنصر را ذکر کنیم، باید قبل از هر مقدار، اندیس آنرا به همراه علامت => بنویسیم. برای درک بهتر، دستورات فوق را با استفاده از تابع array بازنویسی می‌کنیم:

```
$a=array(1=>5,2=>"Ali");
//OR
$family=array("Ali"=>"Hoseini","Reza"=>"Kamali");
```

در این ساختار، اگر از اندیس عددی استفاده شود، می‌توانیم اندیس یک یا چند عنصر را ذکر نکنیم. در این صورت، اندیس آن عنصر، یک‌واحد بیشتر از اندیس عنصر قبل خواهد بود و اگر عنصر مربوطه، اولین عنصر باشد، اندیس آن برابر با صفر خواهد شد:

```
$numbers=array(10=>5,7,4,25=>12,9);
```

نتیجه اجرای دستور فوق، تعریف آرایه \$numbers به صورت زیر است:

```
$numbers[10]=5;
$numbers[11]=7;
$numbers[12]=4;
$numbers[25]=12;
$numbers[26]=9;
```



استفاده از تابع range

این تابع، دو پارامتر از ورودی گرفته و محدوده بین آنها را به صورت یک آرایه، باز می گرداند. برای درک بهتر، به مثال زیر دقت کنید:

```
$years=range(2001,2010);
```

نتیجه اجرای این دستور، چنین است:

```
$years[0]=2001;
$years[1]=2002;
.
.
.
$years[8]=2009;
$years[9]=2010;
```

البته پارامترهای ورودی، می توانند از نوع کارکتر نیز باشند (البته فقط یک کارکتر، نه یک رشته). مثال:

```
$reverseSmallLetters=range("z","a");
```

که نتیجه آن، چنین است:

```
$reverseSmallLetters[0]="z";
$reverseSmallLetters[1]="y";
$reverseSmallLetters[2]="x";
.
.
.
$reverseSmallLetters[24]="b";
$reverseSmallLetters[25]="a";
```

نمایش محتویات آرایه ها

برای نمایش آرایه ها، روش های مختلفی وجود دارد. یکی از این روش ها که قبلاً با آن آشنا شدید، استفاده از حلقه for است که فقط در صورتی کاربرد دارد که اندیس عناصر آرایه، از نوع عددی باشد:

```
$a[1]=5;
$a[2]=8;
$a[3]=4;
for($i=1;$i<=3;$i++)
{
    echo("$a[$i]<br/>\n");
}
/*Output:
5
8
4
*/
```

نمایش آرایه به کمک تابع print_r

این تابع، یک آرایه را به عنوان پارامتر دریافت کرده و عناصر آنرا به ترتیب در خروجی می نویسد:

```
$c["SistanAndBaluchestan"]="Zahedan";
$c["Fars"]="Shiraz";
$c["Tehran"]="Tehran";
print_r($c);
/*Output:
Array
(
    [SistanAndBaluchestan] => Zahedan
    [Fars] => Shiraz
    [Tehran] => Tehran
)
```



این تابع نیز ساختاری مشابه تابع print_r دارد، اما نتایج کامل تری ارائه می دهد:

```
$c["SistanAndBaluchestan"]="Zahedan";
$c["Fars"]="Shiraz";
$c["Tehran"]="Tehran";
var_dump($c);
/*Output:
array(3) {
    ["SistanAndBaluchestan"]=>
        string(7) "Zahedan"
    ["Fars"]=>
        string(6) "Shiraz"
    ["Tehran"]=>
        string(6) "Tehran"
}
*/
```

نکته: دقت کنید که اگر از PHP در وب استفاده می کنید، خروجی تولیدشده، توسط HTML نمایش داده خواهد شد. در نتیجه، شکل منظمی را که در مثال های فوق ملاحظه می کنید، نخواهد داشت و تماماً در یک سطر طولانی نوشته می شود. برای آنکه HTML را وادار کنید خروجی PHP را به همان شکل که دریافت می کند، نمایش دهد و از قالب HTML برای این کار استفاده نکند، از تگ pre استفاده کنید؛ بدین ترتیب که تگ مذکور را قبل از دستورات print_r یا var_dump، باز کرده و بعد از آن دستورات، ببندید:

```
echo("<pre>\n");
print_r($c);
echo("</pre>\n");
//OR
echo("<pre>\n");
var_dump($c);
echo("</pre>\n");
```

نمایش آرایه با استفاده از ساختار تکرار foreach

ساختار تکرار دیگری به نام foreach در PHP وجود دارد که بدون اهمیت به نوع اندیس عناصر آرایه، کل عناصر آنرا پیمایش می کند و کارهای خواسته شده در بدنه حلقه را بر روی همه آنها انجام می دهد. برای درک بهتر این ساختار، نحوه عملکرد آنرا در مثال زیر مشاهده کنید:

```
$capital["S_AND_B"]="Zahedan";
$capital["Fars"]="Shiraz";
$capital["Tehran"]="Tehran";
foreach($capital as $city)
{
    echo("$city<br/>\n");
}
/*Output:
Zahedan
Shiraz
Tehran
*/
```

در این ساختار، به ترتیب عناصر آرایه \$capital با نام موقت \$city پیمایش می شوند و دستور echo بر روی آنها اعمال می گردد. از آنجا که هر بار، \$city برابر با یکی از عناصر آرایه \$capital خواهد بود، همه عناصر آرایه مذکور، در خروجی ظاهر می گردند. این ساختار، از آنجا که با اندیس آرایه کار نمی کند، بدون اهمیت به نوع اندیس یک آرایه (عددی یا رشته ای)، قابل استفاده است.



برای تغییر مقدار یک عنصر از آرایه، کافی است توسط اندیس، به آن عنصر اشاره کرده و مقدار جدید را در آن ذخیره کنیم:

```
01 $stateCount["Iran"]=25;
02 echo ($stateCount["Iran"]."<br/>\n");
//Output: 25
03 $stateCount["Iran"]=29;
04 echo ($stateCount["Iran"]."<br/>\n");
//Output: 29
```

دستور 03، مقدار اندیس "Iran" را در آرایه \$stateCount، از 25 به 29 تغییر می‌دهد.

به‌علاوه، در PHP می‌توانید یک کپی سریع از تمامی مقادیر آرایه را درون یک آرایه جدید قرار دهید:

```
$arrayCopy=$myArray;
```

با اجرای دستور فوق، یک کپی از آرایه \$myArray به‌نام \$arrayCopy با تعداد عناصر یکسان، اندیس‌های مشابه و مقادیر مساوی برای عناصر آرایه‌ها، ایجاد خواهد شد.

حذف عناصر از آرایه

ممکن است گاهی اوقات نیاز به حذف کامل یک عنصر از آرایه داشته‌باشید. برای مثال، آرایه زیر را در نظر بگیرید:

```
$colors=array("red","green","blue","pink","yellow");
```

این آرایه، شامل پنج مقدار است. حال فرض کنید که دیگر به رنگ صورتی علاقه‌ای ندارید و می‌خواهید آنرا از آرایه خود حذف کنید. شاید در نگاه اول، دستور زیر مناسب باشد:

```
$colors[3]="";
```

اما دقت کنید که اگرچه دستور فوق، عنصر چهارم آرایه \$colors را برابر با یک رشته خالی قرار می‌دهد؛ اما آنرا از آرایه حذف نمی‌کند (هنوز هم آرایه‌ای با پنج عنصر دارید که یکی از عناصر آن، یک رشته خالی است). برای حذف کامل یک عنصر از آرایه، از دستور unset استفاده می‌کنیم:

```
unset($colors[3]);
```

اکنون اگر آرایه خود را با هر کدام از ساختارهایی که ذکر شد، نمایش دهیم، نتیجه زیر حاصل خواهد شد:

```
$colors[0]="red"
$colors[1]="green"
$colors[2]="blue"
$colors[4]="yellow"
```

دقت کنید که با استفاده از دستور unset، شماره اندیس عناصر تغییر نمی‌کند و فقط، عنصر موردنظر از آرایه حذف می‌شود. ضمناً در صورت حذف تمامی عناصر یک آرایه با دستور unset، خود آرایه از بین نمی‌رود و کماکان یک آرایه خالی در اختیار خواهید داشت. برای حذف آرایه، کافی است آنرا همانند سایر عناصر آن، حذف کنید:

```
unset($colors);
```

نکته: دستور unset مخصوص آرایه‌ها نیست و برای حذف متغیرهای معمولی نیز قابل استفاده است.