iranphp articles

عنوان مقاله :	Session ها و مفهوم آنها
نگارنده :	بیژن هومند
آدرس پست الکترونیک :	bijan@hoomand.com
تاریخ نگارش :	.....

Sessionها و مفهوم آنها:

قبل از اینکه این مقاله کوتاه را شروع کنم باید بگم که هدف من تنها یک گپ زدن ساده در مورد session هاست و این یعنی که من زمان زیادی را برای مرتب کردن افکارم و طبقه بندی چیزهایی که قراره بگم صرف نکردم، بنابر این در صورت دیدن هر گونه اشکال و یا خواستن توضیح بیشتر، به قول خارجی ها feel free to send me an email to bijan@hoomand.com

خب، اصلا ببینم این همه میگن session ، یعنی چی؟! اگر اون کتابهای ترجمه شده ایرونی مسخره را خونده باشید، میگن session یعنی جلسه. البته درست هم هست، session یکی از معناهاش جلسست، ولی باور کنید اصلا نمیشه به فارسی اونجور که باید ترجمش کرد، پس بذارید بگیم همون session! مثلا ببینید اگر قرار باشه من و شما یک جایی همدیگه را ببینیم و راجع به یک چیزی صحبت کنیم، وقتی که صحبتمون تموم شد و از هم جدا شدیم، شما مثلا به دوستتون میتونید بگید در اون session ای که با بیژن هومند داشتم فلان حرفهایی هم زدیم . اگر دوباره همدیگه را ببینیم میشه یک session جدید ! همون جلسه کم کم داره معنا میده، نه؟ ! ولی باور کنید نه! چون جلسه یک لغت خیلی خیلی رسمی هست تو فارسی، در صورتی که session اینطور نیست. به هر حال امیدوارم معنا را گرفته باشید !

خیلی خوب، حالا ببینیم تو کامپیوتر session به چه معناست . شما امروز صبح که کامپیوتر خودتون را روشن کنید و مثلا ۳ ساعت باهاش کار کنید و خاموشش کنید و بعد از ظهر دوباره برگردید، میتونید به log فایلهاتون سر بزنید تا تمام کارهایی که تو session قبل انجام دادید، یعنی تو ۳ ساعت صبح، را ببینید . یعنی اینکه شما هر وقت وارد یک سیستمی میشید یک session دارید با اون سیستم تا اینکه از اون سیستم خارج بشید .

خب، تو PHP هم قضیه عین همینه. فرض کنید من یک سایت دارم به اسم hoomand.com ، البته اگر این فرض را بکنید چندان هم فرض بیراهی نیست چون من واقعا این سایت را دارم. خب، شما از لحظه ای که تو browser قشنگتون مینویسید www.hooomand.com و وارد سایت من میشید، یک sessionرا ایجاد میکنید، شما داخل هر صفحه ای که از سایت من بشید خب قطعا هنوز داخل سایت من هستید و این session هنوز پابرجاست تا اینکه خدای نکرده از سایتم خسته بشید و صفحه browser را ببندید که اونوقت دیگه session را تموم کردید و یا به عبارتی : session destroy انجام دادید .

Session به چه درد یک برنامه نویس میخوره؟

تنها دلیلی که ما از session ها و یا روشهای مشابه استفاده می کنیم اینه که پروتوکل HTTP توانایی ذخیره هیچ چیزی را تو خودش نداره و مغزش از مغز یک دایناسور هم کوچیکتره! به اصطلاح فرنگی این پروتوکل stateless هست، یعنی چی؟ یعنی اگر شما یک صفحه وب داشته باشید به اسم bijanJoon.htmlو یکی از آمریکا تقاضای دیدن این صفحه را تو browser ش بده و یکی هم از ایران، سرور ما به هیچ عنوان متوجه نمیشه که الان در آن واحد این دو تقاضا داده شده و یا اینکه این ۲ تقاضا از ۲ جای مختلف داده شده و یا اینکه حالا وقتی این طرف تو آمریکا این صفحه را دید بعدش کجا رفته و کدوم صفحات را دیده HTTP . خیلی ساده عمل میکنه و به ازای هر request یک صفحه را به browser میفرسته. اینکار به خودی خود اشکالی نداره اگر ما نخواسته باشیم که مثلا به ازاء صفحات مختلفی که یک کاربر دیده یک پیغام و یا آگهی بهش نمایش بدیم و یا به عنوان مثال به ازاء کارهایی که تو صفحه a انجام داده، عمل مناسب را تو صفحه b براش انجام بدیم. تمام این مشکلات که گفتم در واقع به این برمیگرده که ما میخوایم یک سری متغیر را از لحظه ای که طرف session را ایجاد میکنه (وارد سایت میشه و صفحه اول را نگاه میکنه) درون صفحات مختلفی که اون بابا مشاهده میکنه حرکت بدیم و همیشه داشته باشیم. عین یک session واقعی در دنیای واقعی، وقتی که شما با من یک session سه ساعته دارید و داریم صحبت می کنیم، قطعا بعد از ۲ ساعت از شما توقع ندارم که مطالب دقیقه اول را فراموش کرده باشید ! حالا ببینیم که چه روشهایی برای انجام اینکار وجود داره . خوب، قطعا برای شما اتفاق افتاده که خواسته باشید یک سری متغیر را از صفحه ای که توش هستید به یک صفحه دیگه انتقال بدید. یک روش برای انجام اینکار ای‌نه که این متغیرها را از طریق query string و یا به عبارتی از طریق URL انتقال بدید، مثلا اینجوری :

```
www.hooomand.com/somePage.php?niceBoy=bijan&girlKillingSkillsRate=10000
```

خب، همونطور که میبینید من دو تا متغیر را دارم از طریق URL انتقال میدم، یکی niceBoy هست با مقدار bijan و اون یکی girlKillingSkillsRate که مقدار اون ۱۰۰۰۰۰ هست. خب، زیاد راجع به این بحث نمی کنم چون ارتباطی به session مون نداره ! یک روش دیگه برای انتقال داده از یک صفحه به صفحه دیگه قطعا استفاده از فرمها با متد POST هست (این query string که توضیح دادم در واقع همون فرمها با متد GET هستند دیگه . !) در این روش شما متغیر هاتون را درون فیلدهای فرم قرار میدی،د، حالا اگر خواسته باشید تو صفحه کسی اونها را نبینه خب اونها را داخل فیلدهای hidden قرار میدید و action فرمتون را به اون صفحه دومی که میخواهید اطلاعات را توش بگیرید تغییر میدید .

روش دیگه ای که برای انتقال اطلاعات وجود داره استفاده از COOKIE هاست. در این روش شما اطلاعات خودتون را درون کامپیوتر کاربرتون ذخیره میکنید . ببینید هر browser ای یک شاخه داره که درون اون شاخه به Web Server اجازه داده میشه برای هر سایت یک فایل کوچیک (در حد ۴ KBگر اشتباه نکنم) ایجاد کنه و متغیرهایی که برنامه نویس مشخص کرده در اونها ذخیره کنه. بنابر این با استفاده از کوکی ها شما میتونید متغیرهای سمت سرورتون را درون کامپیوتر خود طرف ذخیره کنید . به مثال زیر دقت کنید :
فرض کنید دو صفحه داریم به اسامی a.php و b.php که هر دو در یک شاخه قرار دارند. این محتوای فایل a.php هست :

```
<?PHP
setCookie("niceBoy", "bijan");
?>
```

ما در این فایل متغیر niceBoy و مقدار اون را درون فایل کوکی ذخیره کردیم، یعنی الان این متغیر به همراه مقدارش درون کامپیوتر کاربر ذخیره شده و نه درون سرور . حالا اگر در فایل b.php بنویسیم :

```
<?PHP
print $_COOKIE["niceBoy"];
?>
```

همونی نمایش داده خواهد شد که شما انتظار دارید ! البته تابع setCookie متغیرهای دیگه ای هم قبول میکنه که بوسیله اونها میشه گفت این کوکی مربوط به چه سایتی هست تا سایتهای دیگه نتونن از کوکی ما استفاده کنن و به عبارتی امنیت بالاتر بره و همچنین میشه مشخص کرد که عمر یک متغیر کوکی ما چقدر باشه، مثلا ۱ ثانیه، ۱ ماه، ۱ سال و یا بیشتر. برای اینکار فایل a.php را به این صورت تغییر میدم :

```
<?PHP
setCookie("niceBoy", "bijan", time() * 60 * 60 * 24);
?>
```

تابع setCookie مقدار زمان را بصورت ثانیه میگیره، بنابر این من با استفاده از تابع time() و ضرب اون در ۶۰ ثانیه (یک دقیقه) و ۶۰ دقیقه (یک ساعت) و ۲۴ ساعت (یک شبانه روز) تعریف می کنم که این متغیر تنها ۲۴ ساعت در کوکی ذخیره بشه .
خب، حالا مزایا و معایب کوکی ها چی هستند؟ از مزیتهاشون میشه به این اشاره کرد که ما اولا فضای سرور را الکی نمی گیریم و اطلاعات را درون کامپیوتر کاربر ذخیره می کنیم،بعلاوه اینکه میتونیم مشخص کنیم این اطلاعات چه مدت اونجا باشند . ولی باور کنید معایب این روش خیلی بیشتره ! اولین و بزرگترین مشکل کوکی ها مساله امنیت هست که شما در واقع دارید متغیرهاتون را درون کامپیوتر کاربر ذخیره می کنید و هر کس درون اون کامپیوتر میتونه با باز کردن فایل کوکی متغیرهای شما را مثل هلو ببینه ! مشکل بعدی ظرفیت کم این فایلها و مشکل دیگه ای هم که هست اینه که browser های قدیمی از cookie ها حمایت نمیکنن و browser های جدید هم این امکان را به کاربران میدن که به کل cookie هاشون را disable کنن. بنابر این میبینید که نمیتونیم یک سایت را به امید کوکی ها بسازیم، حال چه باید کرد؟ session ها البته عزیزان، استفاده از session ها دوای درد ماست ! SESSIONها یا همان COOKIE هایی که در سرور ذخیره میشوند
بله، کل مطلب را در یک جمله رسوندم Session . ها عین همون کوکی ها هستن با این تفاوت که در سرور ذخیره می شوند . شما همینکه وارد یک سایت می شوید که از session ها استفاده میکنه، این سایت برای شما یک فایل منحصر بفرد درون سرور درست میکنه که تمام اطلاعاتی که برنامه نویس سایت میخواد برای شما نگه داره درون اون فایل ذخیره میشه و شما به هر صفحه ای که برید برنامه نویس سایت میتونه درون اون صفحه اون فایل را صدا بزنه و متغیرهایی که تا بحال توشون ذخیره کرده (session جاری) را ازش بخونه . برای اینکه که تو یک صفحه مشخص کنید میخواید از session استفاده کنید باید تابع session_start() را در بالای صفحه و قبل از اینکه هیچگونه خروجی در صفحه تولید کنید، صدا بزنید . دقت کنید به حرفی

که زدم، این تابع چون header تولید میکنه باید قبل از اینکه شما هیچ چیزی را تو صفحه print کنید صدا زده بشه، و اگرنه شما error دریافت میکنید و session شما فراخوانی نمیشه. حتی نباید قبل از فراخوانی این تابع یکدونه space هم تو صفحه print بشه. حالا من این حرف را اینجا زدم، بعدا می بینید که به علت کمی بی احتیاطی چند بار همچین error ای را دریافت میکنید و چند بار به علت فراموش کردن توصیه عمو بیژن از این و اون میپرسید چرا session من هی بهم error میده ! حتی بعضی از editor ها هستن که خودشون یک کاراکتر hex نامرئی اول صفحه مینویسن که خوب باعث میشه session کار نکنه و چون شما این کاراکتر را نمی بینید فکر میکنید همه چیز درسته ! در اینجور مواقع بهتره حتما فایلتون را با یک ادیتور hex باز کنید تا ببینید اوضاع از چه قراره . خیلی خب، خیلی حرف زدم، اجازه بدید همون فایل a.php را ایندفعه بوسیله session درست کنیم :

```
<?PHP
session_start();
$_SESSION["aNiceBoy"] = "bijan";
?>
```

اه اه اه، دیدین چقدر راحت بود؟ به همین سادگی، اولش مینویسین session_start() و بعدش هم در هر جای صفحه که بخواید متغیرهاتون را بوسیله آرایه superglobal ای که میبینید (آرایه $_SESSION) ایجاد می کنید Superglobal . را هم که میدونید قطعا یعنی چی؟ آرایه های superglobal در PHP مثل همینی که دیدید و یا آرایه $_POST و یا $_GET آرایه هایی هستند که در همه جای برنامه تون میتونید اشاره کنید بدون اینکه نیاز به تعریف اونها درون function هاتون و یا class هاتون داشته باشید . فرقش با global چیه؟ خب ببینید ما اگر یک متغیر مثلا $bijan به صورت global بخوایم درون یک تابع استفاده کنیم، در ابتدای تابع باید بنویسیم global $bijan تا تابع بفهمه که باید $bijan را از فضای global بیاره. ولی برای آرایه های superglobal همچین کاری لازم نیست، گرفتید؟ یک مساله دیگه هم که باز تاکید میکنم اینه که وقتی میخواید از session استفاده کنید بعد از باز کردن تگ php در بالای فایلتون حتما اولین چیزی که مینویسید قبل از فرستادن هیچگونه خروجی به browser همین تابع session_start() باشه . بعد از اینکه اینو نوشتید دیگه اشکال نداره هر خروجی که بخواید بفرستید و هر جا که خواستی با آرایه $_SESSION کار کنید .

خب، حالا یک نگاهی به فایل b.php میندازیم که میخواد از متغیر aNiceBoy ذخیره شده در session استفاده کنه :

```
<?PHP
session_start();
print "Girls love " . $_SESSION["aNiceBoy"];
?>
```

همینطور که میبینید ما اینجا باز هم تابع session_start() را اون بالا صدا زدیم، بنابر این هر جا که بخواید با session ها کار کنید، چه بخواید اونها را بسازید و یا ازشون استفاده کنید باید این تابع را اول بسم الله صدا بزنید . در ضمن فکر نکنید فقط میتونید متغیر های ساده را درون session ها ذخیره کنید، خیر، شما می تونید یک آرایه n بعدی یا حتی خفن ترین object ای که فکرش را بکنید را هم خیی راحت درون یک session ذخیره بکنید .

خب، حالا یک کم ریزتر به این session های عزیز نگاه میکنیم. بهتون گفتم که به ازاء هر شخص جدیدی که وارد یک سایت میشه، PHPیک فایل درون سرور براش در نظر میگیره که متغیرهاش را توش ذخیره میکنه . خب، حالا اگر این حقیقت را در نظر بگیریم که در آن واحد ممکنه هزاران نفر به یک سایت وصل باشند، بنابر این ما هزاران فایل هم داریم که برای هر connection و یا به عبارتی هر شخص) این جمله چندان درست نبود، ولی خب فعلا برای شما خوبه!) متغیرهای مربوط به اون را نگاهداری میکنه . خب، حالا سرور بیچاره از کجا بفهمه که کدوم فایل مال کیه؟ ! خیلی ساده، هر دفعه که یکی از این فایلهای session ساخته میشن، اسمشون را به browser کاربر میفرستن و کاربر به هر صفحه ای که میره این اسم را به سرور میفرسته تا server بفهمه که کدوم فایل مال این طرفه . فرض کنید من و حسن و علی در آن واحد به hoomand.com وصل شدیم و داریم صفحه homepage قشنگ اونو نگاه می کنیم Server . سه تا فایل برای ما درست کرده فرض کنید به همون اسمهای bijan و hasan و ali. و ما هر صفحه از سایت را که بخوایم نگاه کنیم اسم فایل مربوط به خودمون را هم به همراه آدرس صفحه به server میفرستیم تا server فایل مربوط به خودمون را برامون استفاده کنه .

بنابراین من اگر بخوام صفحه yekchizi.php را ببینم، تو browser خودم تایپ میکنم www.hooomand.com/yekchizi.php?sid=bijan : تا session فایل خودم برام استفاده بشه . چی شد؟ بنابر این اگر علی نامرد هم بخواد server را گول بزنه و از محتوای session من استفاده کنه که ممکنه مثلا password ام توش ذخیره شده باشه خیلی راحت میتونه بجای

اینکه اسم خودشو برای server بفرسته، از آدرس بالا استفاده کنه و اسم منو بفرسته؟ ! آره، میتونه ! خب، چیکار باید کرد؟ به نظر راهی غیر از این وجود نداره که اسم این فایلها بصورت random انتخاب بشه و حتی المقدور یک متغیر حرفی/عددی بزرگ باشه که علی نامرد نتونه اسم فایل منو حدس بزنه، چطوره ؟ خب بدک نیست، ولی اگر علی جون بالای سرم وایستاده باشه وقتی که دارم آدرسم را تو browser م تایپ می کنم چی؟ خب، یک سری کلکهایی وجود داره که با چک کردن IP و از اینجور کارها جلوی طرف را گرفت، ولی باور کنید همه session ها قابل دزدیدن (hijack شدن) هستند و همه سیستم ها قابل hack و من هم قرار نیست در این مقاله کوتاه که صرفا جهت آشنایی شما با session هاست خیلی رو مطلب ریز بشم . بنابر این تا همینجا را ازم قبول کنید بعلاوه اینکه حالا من گفتم اون اسم فایل من از طریق query string به سرور فرستاده میشه، ولی معمولا اینطور نیست اگر شما کوکی هاتون فعال باشه و browser تون بتونه تو کوکی رایت کنه . در اینصورت این sid درون یک کوکی نوشته میشه و شما هیچوقت اونو نمیبینید و بصورت مخفی به سرور فرستاده میشه . البته خب این قضیه را همه میدونن و اگر یک hacker بخواد کاری بکنه خب این مطلب را هم در نظر میگیره ، ولی باز خوب کار براش سختتر میشه .

چه کارهایی میشه با SESSION ها انجام داد؟

قطعا تنها کاری که میشه با session ها انجام داد همون داشتن یک متغیر در طول کل عمر یک session و یا به عبارتی در تمام صفحاتیست که یک کاربر داره از اونها دیدن میکنه . ولی خب با همین مفهوم ساده خیلی کارها میشه کرد . به عنوان مثال تمام سیستمهای user/password ای که تو سایتها می بینید به همین روش ایجاد شدن . فرض کنید شما ۵ تا کاربر مختلف دارید بعلاوه یک صفحه خاص که می خواید ۲ تا از این کاربرها بتونن این صفحه را ببینن و ۳ تا نتونن. خب، همون لحظه ای که اینها نام کاربری و کلمه عبور خودشون را میزنن و شما میبینید که این اطلاعات درسته و در واقع اینها همونهایی هستند که باید باشن، برای اون دو کاربری که حق دیدن اون صفحه را دارن یک متغیر session مثلا بصورت زیر درست می کنید :

```php
<?PHP
session_start();
$_SESSION["allowedToWatch"] = 1;
?>
```

برای اون ۳ تای دیگه هم هیچ کاری لازم نیست بکنید ! حالا تنها کاری که باید بکنید اینه که درون اون صفحه خاص کد زیر را قرار بدید :

```php
<?PHP
session_start();
if ( $_SESSION["allowedToWatch"] != 1 )
{
    die("You are not allowed to watch!");
}
# if the compiler gets here it means that the user has that
# variable in his/her session, so, he/she is not an intruder!
?>
```

دیدید چقدر راحت بود؟ ! با استفاده از database و کمی بازی میتونید یک سیستم authentication داشته باشید، البته قبل از اینکه اینکارو بکنید باز هم یک مطالعه ای بکنید چون همچین سیستمی یک ذره ظریف کاریهای دیگه هم لازم داره که اگر همین یک ذره را رعایت نکنید خیلی شیرین hack میشید

کار دیگه ای که من خودم خیلی تو فرمهام انجام میدم اینه که خیلی پیش میاد شما مثلا برای register کردن یک فرآیندی می خواید در ۳ صفحه فرمهاتون را به کاربر نشون بدید تا پرشون کنه، دلیلشم یکی اینه که خب طرف حوصلش سر نره همین که فرم گنده شما را میبینه صفحه را نبنده، دلیل دیگشم میتونه این باشه که به ازاء ورود فرم اول ، یکی از چندین فرم دوم قابل نمایش باید برای طرف انتخاب باشه و الخ ! خب، من برای اینکار هیچوقت action فرمم را یک صفحه دیگه قرار نمیدم، بلکه همیشه action فرمم $_SERVER["PHP_SELF"] هست و تو همون صفحه چک می کنم که طرف هیچ خطایی تو پر کردن فرم نداشته باشه، اگر داشته باشه که همونجا بهش error میدم، و اگرنه آرایه $_POST م را میریزم تو یک session که مختص اون صفحست مثلا ($_SESSION["myFirstPage"]) و اونوقت با یک header و location میرم صفحه بعدیم . البته فراموش نکنید که این حقیقت که session ها تا آخر browse طرف پابرجا هستن بعضی جاها میتونه چندان جالب نباشه که اونوقت با استفاده از یک sessionچند بعدی (آرایه چند بعدی) میشه مشکل را حل کرد

.

همه قضایا را گفتیم بریم سر خان آخر که کی این session ما از بین میره و یا expire میشه؟ خب، در چندین حالت : در صورتی که شما با استفاده از تابع session_destroy() خودتون بصورت دستی session را از بین ببرید . فرض کنید برای همون سیستم username/password یک صفحه logout.php تهیه کردید که خیلی راحت اینکار را انجام میده :

```
<?PHP
session_start();
session_destroy();

header("Location: homepage.html");
?>
```

اتفاق دیگه ای که میتونه بیفته اینه که کاربر مثلا ۱۵ دقیقه وارد صفحه دیگه ای نشه و به عبارتی با web server تعامل نداشته باشه، در اینصورت هم sessionبصورت خودکار از بین میره. البته این زمان ۱۵ دقیقه که گفتم بصورت default هست تا اونجایی که یادمه، ولی خیلی راحت میشه در php.ini عوضش کرد . اتفاق دیگه ای هم که ممکنه بیفته اینه که طرف وقتی تو سایت شما هست، تو address bar بروزرش آدرس یک سایت دیگه را بنویسه و اینجوری سایت شما را ترک کنه، ولی نه ! همیشه با اینکار session بابا از کار نمیفته. تا حالا این تجربه رویایی را نداشتید که وقتی یک ladyتو سایت دانشگاتون نشسته و داره email های yahoo ش را چک میکنه و بعد خیلی زیبا تصمیم میگیره مستقیما از صفحه mail box ش بره یک سایت پروانه ای را نگاه کنه، و بعد هم که پر از احساس شد سایت را ترک میکنه بدون اینکه صفحهbrowser را ببنده، شما برید و خیلی قشنگ بنویسید mail.yahoo.com و برید تو mail box ش و چند تا نامه خوشگل برای بعضی ها بفرستید؟! خب، میدونید این اتفاق وقتی میفته که همونطور که تو چند صفحه پیش توضیح دادم، این sid یا همون نام فایل session بصورت تصادفی تولید شده و هر بار به هنگام تقاضای هر صفحه جدید به سرور میره در کوکی ذخیره شده باشه . اگر کوکی ها غیر فعال باشند و یا به هر دلیلی این sid از طریق url اینور و اونور بره همچین مساله ای پیش نمیاد، ولی اگر تو کوکی باشه .... جون. برای همین هم هست که همیشه توصیه میشه قبل از خارج شدن از یک سایت، حتما روی دگمه logout کلید کنید و همینجوری browser تون را نبندید و یا آدرس سایت دیگه ای را اونجا ننویسید .

خیلی خب در پایان لازمه از خودم که این مقاله را نوشتم تشکر کنم ، واقعا جالب بود ، کاش منهم وقتی session ها را بلد نبودم همچین مقاله هایی بود که میخوندم، البته واقعا بوده، open sourceیعنی همین، یعنی آزادی و یعنی لذت بردن از یاد گرفتن و یاد دادن . خدا پدر Richard Stallman را بیامرزاد . اول مقاله گفتم چندان در مورد فصل بندی این مقاله فکر نکردم، همیطور هم هست، ولی موقع نوشتن تو cookie ها یک چیزیادم رفته بود که به کتاب Wiley PHP 5 and MySQL Bible مراجعه کردم (نوشته Joyce Park و Clark Morgan) و چند خطی هم از اونجا به امانت گرفتم (قسمت مربوط به پروتوکل HTTP) . در کل امیدوارم مفید واقع شده باشه، ناهار حاضره باید برم .