

## Cursors in MySQL Stored Procedures



## What is a CURSOR?

A cursor can't be used by itself in MySQL. It is an essential component in stored procedures. I would be inclined to treat a cursor as a "pointer" in C/C++, or an iterator in PHP's foreach statement.

With cursors, we can traverse a dataset and manipulate each record to accomplish certain tasks. When such an operation on a record can also be done in the PHP layer, it saves data transfer amounts as we can just return the processed aggregation/statistical result back to the PHP layer (thus eliminating the `select` – `foreach` – manipulation process at the client side).

Since a cursor is implemented in a stored procedure, it has all the benefits (and limitations) of an SP (access control, pre-compiled, hard to debug, etc).

The official documentation on cursors is located [here \(http://dev.mysql.com/doc/refman/5.6/en/cursors.html\)](http://dev.mysql.com/doc/refman/5.6/en/cursors.html). It contains only four commands that are related to cursor declaration, opening, closing, and fetching. As mentioned above, we will also touch on some other stored procedure statements. Let's get started.

## A real world question

My personal website has a page showing the scores of my favorite NBA team: LA Lakers. The table structure behind it is straightforward:

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
id	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
year	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
team	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
selfscore	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
oppscore	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
dateplayed	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
winlose	VARCHAR(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
remark	LONGTEXT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Fig 1. The Lakers matches status table structure

I have been updating this table since 2008. Some of the latest records showing Lakers' 2013-14 season are shown below:

Query 1    lakers    lakers - Table

1 • `SELECT * FROM sitepoint.lakers where year=2013 and id>30`

Result Set Filter:    Edit    Export/Import    Wrap Cell Content: `☒`

	id	year	team	selfscore	oppscore	dateplayed	winlose	remark
▶	31	2013	LAL vs PHI	104	111	2013-12-29	L	5 losses.
	32	2013	LAL vs MIL	79	94	2013-12-31	L	6 losses.
	33	2013	LAL vs UTA	110	99	2014-01-03	W	
	34	2013	LAL vs DEN	115	137	2014-01-05	L	
	35	2013	LAL @ DAL	97	110	2014-01-07	L	
	36	2013	LAL @ HOU	99	113	2014-01-08	L	No way!
	37	2013	LAL @ LAC	87	123	2014-01-10	L	
	38	2013	LAL vs CLE	118	120	2014-01-14	L	
	39	2013	LAL @ PHX	114	121	2014-01-15	L	
	40	2013	LAL @ BOS	0	0	2014-01-17		
	41	2013	LAL @ TOR	0	0	2014-01-19		
	42	2013	LAL @ CHI	0	0	2014-01-20		
	43	2013	LAL @ MIA	0	0	2014-01-23		
	44	2013	LAL @ ORL	0	0	2014-01-24		
	45	2013	LAL @ NYK	0	0	2014-01-26		
	46	2013	LAL vs IND	0	0	2014-01-28		
	47	2013	LAL vs CHA	0	0	2014-01-31		
	48	2013	LAL @ MIN	0	0	2014-02-04		
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

lakers 3    Apply    Cancel

Fig 2. The Lakers matches status table data (partial) for 2013-2014 season

(I am using MySQL Workbench as the GUI tool to manage my MySQL databases. You can use your favorite tool.)

Well, I have to admit Lakers are not playing very well these days. 6 consecutive losses up to Jan 15th. I get this "6 consecutive losses" by manually counting from the last played match all the way up (towards earlier games) and see how long an "L" (meaning a loss) in winlose column can appear. This is certainly doable but if the requirement becomes more complicated in a larger table, it takes more time and is more error prone.

Can we do this with a single SQL statement? I am not an SQL expert and I haven't been able to figure out how to achieve the desired result ("6 consecutive losses") from one SQL statement. The input of gurus will be highly appreciated – leave it in the comments below.

Can we do this in PHP? Yes, of course. We can retrieve the game data (particularly, the winlose column) for current season and do a traverse on the records to calculate the current longest win/lose streak. But to do that, we will have to grab all data for that year and most of the data will be wasted (as it is not likely for a team to have a win/lose streak for more than 20+

games in a 82-game regular season). However, we don't know how many records should be retrieved into PHP to determine the streak, so this waste is a must. And finally, if the current win/lose streak is the only thing we want to know from that table, why pull all the raw data?

Can we do this via other means? Yes, it is possible. For example, we can create a redundant table specifically designed to store the current win/lose streak. Every insertion of the record will update that table too. But this is way too cumbersome and too error prone.

So, what is a better way to achieve this result?

## Using Cursor in a Stored Procedure

As the name of this article suggests, we will see a better alternative (in my view) to solve this problem: using cursor in a Stored Procedure.

Let's create the first SP in MySQL Workbench as follows:

```
DELIMITER $$

CREATE DEFINER=`root`@`localhost` PROCEDURE `streak`(in cur_year int, out longeststreak int, out sta
BEGIN
    declare current_win char(1);
    declare current_streak int;
    declare current_status char (1);

    declare cur cursor for select winlose from lakers where year=cur_year and winlose<>' ' order by i

    set current_streak=0;

    open cur;

    fetch cur into current_win;
    set current_streak = current_streak +1;

    start_loop: loop
        fetch cur into current_status;
        if current_status <> current_win then
            leave start_loop;
        else
            set current_streak=current_streak+1;
        end if;

    end loop;

    close cur;

    select current_streak into longeststreak;
    select current_win into `status`;
END
```

In this SP, we have one input parameter and two output parameters. This defines the signature of the SP.

In the SP body, we also declared a few local variables to hold the streak status (win or lose, current\_win), current streak and current win/lose status for a particular match.

```
declare cur cursor for select winlose from lakers where year=cur_year and winlose<>'' order by id de
```

The above line is the cursor declaration. We declared a cursor named cur and the dataset bind to that cursor is the win/lose status for those matches played (thus its winlose column is either "W" or "L" instead of nothing) in a particular year ordered by id (the latest played games will have the highest ID) descending.

Though not displayed explicitly, we can imagine that this dataset will contain a series of "L"s and "W"s. Based on the data shown in Figure 2 above, it should be: "LLLLLLWLL..." (6 Ls, 1 Ws, etc).

To calculate the win/lose streak, we begin with the latest (and the first in the dataset) match data. When a cursor is opened, it always starts at the first record in the associated dataset.

After the first data is grabbed, the cursor will move to the next record. In this way, a cursor behaves very much like a queue, traversing the dataset in a FIFO (First In First Out) manner. This is exactly what we wanted.

After getting the current win/lose status and set the streak number, we continue to loop through (traverse) the remainder of the dataset. With each loop iteration, the cursor will "point" to the next record until we break the loop or all the records are consumed.

If the next win/lose status is the same as the current win/lose status, it means the streak goes on and we increase the streak number by 1 and continue the traversing; otherwise, it means the streak discontinues and we can leave the loop earlier.

Finally, we close the cursor and release the resources. Then we return the desired output.

Next, we can enhance the access control of the SP as described in [my previous article \(http://www.sitepoint.com/stored-procedures-mysql-php/\)](http://www.sitepoint.com/stored-procedures-mysql-php/).

To test the output of this SP, we will write a short PHP script:

```

<?php
$dbms = 'mysql';

$host = 'localhost';
$db = 'sitepoint';
$user = 'root';
$pass = 'your_pass_here';
$dsn = "$dbms:host=$host;dbname=$db";

$cn=new PDO($dsn, $user, $pass);

$cn->exec('call streak(2013, @longeststreak, @status)');
$res=$cn->query('select @longeststreak, @status')->fetchAll();

var_dump($res); //Dump the output here to get a raw view of the output

$win=$res[0]['@status']='L'? 'Loss': 'Win';
$streak=$res[0]['@longeststreak'];

echo "Lakers is now $streak consecutive $win.\n";

```

This will output something like the following figure:

```

vagrant@precise64: /working/008 Cursor in MySQL/src$ php callCursorSP.php
array(1) {
  [0]=>
    array(4) {
      [\"@longeststreak\"]=>
        string(1) \"6\"
      [0]=>
        string(1) \"6\"
      [\"@status\"]=>
        string(1) \"L\"
      [1]=>
        string(1) \"L\"
    }
}
Lakers is now 6 consecutive Loss.
vagrant@precise64: /working/008 Cursor in MySQL/src$

```

(This output is based on Lakers' match up to Jan 15th, 2014.)

## Return a dataset from a Stored Procedure

A few discussions went along on how to return a dataset from an SP, which constructs the dataset out of the results from a few repeated calls to another SP.

A user may want to know more from our previously created SP that only returns a win/lose streak for one year; thus we can get a table showing the win/lose streaks for all the years in a form like:

YEAR	Win/Lose	Streak
2013	L	6

2012	L	4
2011	L	2

(Well, a more useful result can be to return the longest win streak and loss streak in a particular season. This requirement can be easily expanded from the previous SP so I will leave it to interested parties to implement. For the purpose of this article, we will stick to the current win/loss streak.)

MySQL SP can only return scalar results (an integer, a string, etc), unless the result is returned by a `select ... from ...` statement (and it becomes a dataset). The issue here is that the table-form data we want to see does not exist in our current db structure and is constructed from another SP.

To tackle this, we need the help of a temporary table, or if situation allows and requires, a redundant table. Let's see how we can achieve our target via a temporary table.

First, we will create a second SP as shown below:

```
DELIMITER $$

CREATE DEFINER=`root`@`%` PROCEDURE `yearly_streak`()
begin
    declare cur_year, max_year, min_year int;

    select max(year), min(year) from lakers into max_year, min_year;

    DROP TEMPORARY TABLE IF EXISTS yearly_streak;
    CREATE TEMPORARY TABLE yearly_streak (season int, streak int, win char(1));

    set cur_year=max_year;

    year_loop: loop
        if cur_year<min_year then
            leave year_loop;
        end if;

        call streak(cur_year, @l, @s);
        insert into yearly_streak values (cur_year, @l, @s);

        set cur_year=cur_year-1;
    end loop;

    select * from yearly_streak;
    DROP TEMPORARY TABLE IF EXISTS yearly_streak;

END
```

A few key things to notice here:

- 1 We determine the max year and min year by selecting from the table lakers;
- 2 We created a temp table to hold the output, with the structure requested by the output (season, streak, win);
- 3 In the loop, we first execute our previously created SP, with necessary parameters (`call streak(cur_year, @l, @s);`), then grab the data returned and insert into the temp table (`insert into yearly_streak values (cur_year, @l,`

```
@s);).
```

- 4 Finally, we select from the temp table and return the dataset, then do some cleaning (`DROP TEMPORARY TABLE IF EXISTS yearly_streak;`).

To get the results, we create another short PHP script as shown below:

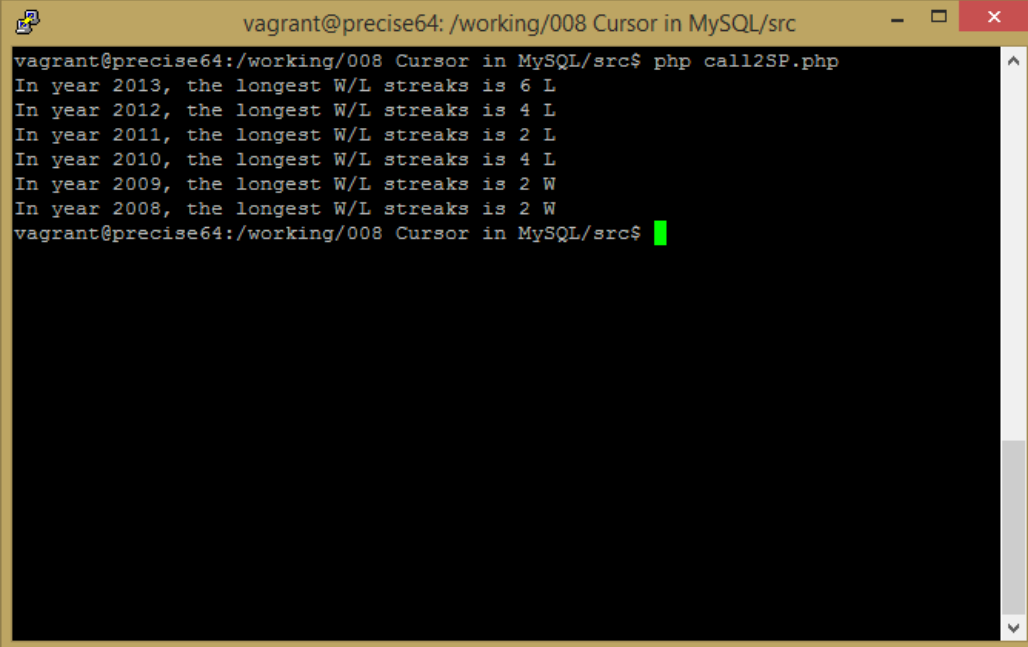
```
<?php
... // Here goes the db connection parameters

$cn=new PDO($dsn, $user, $pass);

$res=$cn->query('call yearly_streak')->fetchAll();

foreach ($res as $r)
{
    echo sprintf("In year %d, the longest W/L streaks is %d %s\n", $r['season'], $r['streak'], $r['w
}
}
```

And the display will be like:



```
vagrant@precise64: /working/008 Cursor in MySQL/src$ php call2SP.php
In year 2013, the longest W/L streaks is 6 L
In year 2012, the longest W/L streaks is 4 L
In year 2011, the longest W/L streaks is 2 L
In year 2010, the longest W/L streaks is 4 L
In year 2009, the longest W/L streaks is 2 W
In year 2008, the longest W/L streaks is 2 W
vagrant@precise64: /working/008 Cursor in MySQL/src$
```

Please note that the above is a bit different from calling our first SP.

The first SP does not return a dataset but only two parameters. In that case, we use PDO `exec` then `query` to fetch the output; while in the second SP, we returned a dataset from the SP, so we use PDO `query` directly to invoke the call to the SP.

Voila! We did it!


## Conclusion

In this article, we dug further into MySQL stored procedures and took a look at the cursor functionality. We have demonstrated how to fetch scalar data by output parameters (defined as `out var_name vartype` in the SP declaration) and also fetch a calculated dataset via a temp table. During this process, a few statements otherwise used in stored procedures also surfaced.

The official documentation on the syntax of stored procedure and various statements can be found on the MySQL website. To create a stored procedure, please refer to [these documents \(http://dev.mysql.com/doc/refman/5.6/en/create-procedure.html\)](http://dev.mysql.com/doc/refman/5.6/en/create-procedure.html) and to understand the statements, please see [here \(http://dev.mysql.com/doc/refman/5.6/en/sql-syntax-compound-statements.html\)](http://dev.mysql.com/doc/refman/5.6/en/sql-syntax-compound-statements.html).

Feel free to comment and let us know your thoughts!

Tags: [cursors \(http://www.sitepoint.com/tag/cursors/\)](http://www.sitepoint.com/tag/cursors/), [mariadb \(http://www.sitepoint.com/tag/mariadb/\)](http://www.sitepoint.com/tag/mariadb/), [mysql \(http://www.sitepoint.com/tag/mysql/\)](http://www.sitepoint.com/tag/mysql/), [PHP \(http://www.sitepoint.com/tag/php/\)](http://www.sitepoint.com/tag/php/), [stored procedure \(http://www.sitepoint.com/tag/stored-procedure/\)](http://www.sitepoint.com/tag/stored-procedure/)

Was this helpful?  

Get our latest PHP articles in your inbox, once a week, for free.

Enter your email

Get Updates



[Taylor Ren \(http://www.sitepoint.com/author/tren/\)](http://www.sitepoint.com/author/tren/)

[g+](https://plus.google.com/+TaylorRen) (<https://plus.google.com/+TaylorRen>) [f](https://www.facebook.com/taylor.ren) (<https://www.facebook.com/taylor.ren>)

Taylor is a freelance web and desktop application developer living in Suzhou in Eastern China. Started from Borland development tools series (C++Builder, Delphi), published a book on InterBase, certified as Borland Expert in 2003, he shifted to web development with typical LAMP configuration. Later he started working with jQuery, Symfony, Bootstrap, Dart, etc.

Comments for this thread are now closed.



13 Comments [SitePoint](#)

 Login ▾

 Recommend  Share

Sort by Best ▾

[Charles Bryant](#) · 2 years ago

I would seriously consider every other option before using a cursor, they are generally used by people who do not understand sql, they are pretty bad for memory, not really what sql is designed to do. But on the other hand there are times when I have to use them, my advice proceed with caution.

Had a bit of trouble working this out as I mainly do MsSql. But essentially my method is to left join the table to itself using the previous id then increment the counter if winlose is the same on both tables.

```
SET @rownum := 0;
select *,
@rownum :=
CASE WHEN l1.winlose = l2.winlose THEN @rownum + 1 ELSE 1 END
AS ConsecutiveCount
from lakers l1
left join lakers l2 on l1.id - 1 = l2.id;
```

1 ^ | ▾ · Share ▾



**Taylor Ren** → Charles Bryant · 2 years ago

This is also a nice trick but won't work if id is not an int field.

I use cursor very rarely, more and more relied on frontend/midend logic nowadays.

I may not agree though that "not really what sql is designed to do" as it is designed into the MySQL (or MSSQL) engine.

^ | v · Share ›

**Charles Bryant** → Taylor Ren · 2 years ago

If you do not have an id field as an integer, you are doing sql wrong! If for some reason you did not have an id field you could use the date field and a double left join (maybe?) to only include one row, or insert the table into a temp table with an int column then do the join?

I think the overall emphasis of this article is that you should do as much as you can in sql and use as few calls to the database as possible. Think separation of concerns. If you use the php solution to this you are just adding overhead to your php script.

That is true but I would be very mindful of when and where to use a cursor, as I showed in my example the code is far less also it will execute far quicker.

^ | v · Share ›

**Taylor Ren** → Charles Bryant · 2 years ago

Wow, how can this conclusion be: "If you do not have an id field as an integer, you are doing sql wrong!"

And even in the example you demonstrated, you have to assume that "id" is in strict increment by 1 order. What if the "id" (though it is an integer) but does not follow that pattern (for example, you can delete a few and insert a few back in between). Then this algorithm does not apply at all.

Of course, in the particular example, your algorithm works but for a more general solution, my clumsy way will not fail (and change the "order by id" to "order by dateplayed" will do).

^ | v · Share ›

**Charles Bryant** → Taylor Ren · 2 years ago

You are just being ridiculous for argument's sake.

^ | v · Share ›

**Bruno Skvorc** SitePoint Staff → Charles Bryant · 2 years ago

Taylor is right, arguments over insults please. I'd love for a more in depth discussion about this to sprout from this comment thread if possible.

^ | v · Share ›

**Taylor Ren** → Charles Bryant · 2 years ago

Hey, mind your choice of wordings here. Convince me if I am wrong.

^ | v · Share ›

**Charles Bryant** → Taylor Ren · 2 years ago

Am I wrong in thinking that when you design a database you should always include a id column who's data type should be integer? Should also be the primary key unless in a particular situation, this column would be the clustered index and provide the natural ordering for the whole table? I could see no reason why this approach would not have been used on this table?

<http://stackoverflow.com/quest...>

Apologies if you thought my response was a bit harsh but your's was no better / constructive.

In regards to deleting stuff from your database you should probably never do this either, I much prefer to use a soft delete option a bit column that allows you to say if a column has been deleted, if you are to delete a row how are you to ever know that you have deleted records from your database? In which case you could probably use a double left join on the data?

My opinion is that your use of cursor in this situation is wrong. There are many different options to solve this problem but I don't think a cursor is appropriate. Not saying that they do not have their place but it really is a last solution, which is what you should be promoting.

After your requirement had changed on this issue, I could maybe suggest another solution where you would use two temp tables (I believe you can not use a temp table twice in MySql?) select the data from the first, you could set an index on the first temp table, then select the first set of data into the second table, perform the join on the two temp tables problem solved.

^ | v • Share ›

**Taylor Ren** → Charles Bryant • 2 years ago

Charles Bryant

Have you read another article by me titled as "Unique Index": <http://www.sitepoint.com/unique...> and also the article by Craig which was mentioned in that article too?

^ | v • Share ›

**Bruno Skvorc** SitePoint Staff → Charles Bryant • 2 years ago

I disagree with never deleting stuff from the database. I've worked in environments where databases spanned gigabytes upon gigabytes of clustered data, and you sure learn to value deletion.

I do always have a numeric PK UI, though, along with other useful UIs if necessary.

No opinion on cursors from my end, zero experience with (and need for) those.

^ | v • Share ›

**Charles Bryant** → Charles Bryant • 2 years ago

I am taking this quote from stack over flow as I did not put it into words as well as I could have, when I said sql was not designed to this.

"What you are really doing is attempting to force set-based technology into non-set based functionality. And, in all fairness, I should point out that cursors do have a use, but they are frowned upon because many folks who are not used to using set-based solutions use cursors instead of figuring out the set-based solution."

<http://stackoverflow.com/quest...>

^ | v • Share ›

**Alex Stetsenko** • 2 years ago

I think this is a solid overview of using stored procedures and cursors for those who interested in the subject. Especially examples how to call stored procedures in PHP and access output parameters should be helpful. The article even shows how to use "breaks" within stored procedures.

However using cursor and temp table in given examples seems like overkill. For instance isn't it possible to get "6 consecutive losses" using a query like this one below?

```
SELECT COUNT(*) FROM lakers WHERE winlose = 'L' AND dateplayed > (SELECT MAX(dateplayed) FROM lakers WHERE winlose = 'W');
```

^ | v • Share ›

**Spiechu** • 2 years ago

The trick with temporary table is just great. Thanks!

^ | v • Share ›

✉ Subscribe

🗨 Add Disqus to your site Add Disqus Add

🔒 Privacy

[COURSES >](#)

[https://www.sitepoint.com/premium?utm\\_source=content\\_types&utm\\_medium=Course&utm\\_campaign=premium](https://www.sitepoint.com/premium?utm_source=content_types&utm_medium=Course&utm_campaign=premium)

2:07:36

## **Zend Framework 2: The Basics**

Matthew Setter

★★★★☆

([https://www.sitepoint.com/premium/course/zend-framework-2-the-basics-2865/?utm\\_source=sitepoint&utm\\_medium=relat](https://www.sitepoint.com/premium/course/zend-framework-2-the-basics-2865/?utm_source=sitepoint&utm_medium=relat)

1:35:08

## **Local Development Environments for Designers and Developers**

Kray Mitchell

★★★★☆

([https://www.sitepoint.com/premium/course/local-development-environments-for-designers-and-developers-2856/?utm\\_source=sitepoint&utm\\_medium=relat](https://www.sitepoint.com/premium/course/local-development-environments-for-designers-and-developers-2856/?utm_source=sitepoint&utm_medium=relat)

1:06:17

## **Elements of Object-oriented PHP**

Lorna Mitchell

★★★★☆

([https://www.sitepoint.com/premium/course/elements-of-object-oriented-php-2734/?utm\\_source=sitepoint&utm\\_medium=relat](https://www.sitepoint.com/premium/course/elements-of-object-oriented-php-2734/?utm_source=sitepoint&utm_medium=relat)

## **BOOKS >**

([https://www.sitepoint.com/premium/content\\_types/Book&utm\\_source=sitepoint&utm\\_medium=relat](https://www.sitepoint.com/premium/content_types/Book&utm_source=sitepoint&utm_medium=relat)



## **Jump Start PHP Environment**

Bruno Skvorc

★★★★☆

[https://www.sitepoint.com/premium/book/start-php-environment/?utm\\_source=sitepoint&utm\\_medium=relat](https://www.sitepoint.com/premium/book/start-php-environment/?utm_source=sitepoint&utm_medium=relat)

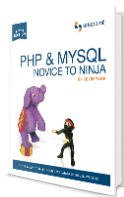


### **Jump Start PHP**

[Callum Hopkins](#)

★★★★☆

[https://www.sitepoint.com/premium/book/start-php/?utm\\_source=sitepoint&utm\\_medium=relat](https://www.sitepoint.com/premium/book/start-php/?utm_source=sitepoint&utm_medium=relat)



### **PHP & MySQL: Novice To Ninja, 5th Edition**

[Kevin Yank](#)

★★★★☆

[https://www.sitepoint.com/premium/book/mysql-novice-to-ninja-5th-edition/?utm\\_source=sitepoint&utm\\_medium=relat](https://www.sitepoint.com/premium/book/mysql-novice-to-ninja-5th-edition/?utm_source=sitepoint&utm_medium=relat)

## **SCREENCASTS >**

[https://www.sitepoint.com/premium/content\\_types/ScreenCast&](https://www.sitepoint.com/premium/content_types/ScreenCast&)

### **Sharing Data with PHPExcel**

[Ashraff Hathibelagal](#)

[https://www.sitepoint.com/premium/tutorial/data-with-phpexcel/?utm\\_source=sitepoint&utm\\_medium=relat](https://www.sitepoint.com/premium/tutorial/data-with-phpexcel/?utm_source=sitepoint&utm_medium=relat)

### **Handling PHP Logging with Monolog**

[Ashraff Hathibelagal](#)

[https://www.sitepoint.com/premium/tutorial/php-logging-with-monolog/?utm\\_source=sitepoint&utm\\_medium=relat](https://www.sitepoint.com/premium/tutorial/php-logging-with-monolog/?utm_source=sitepoint&utm_medium=relat)

### **Faker! Dummy Data for Your Empty Databases**

[Ashraff Hathibelagal](#)

[https://www.sitepoint.com/premium/tutorial/dummy-data-for-your-empty-databases/?utm\\_source=sitepoint&utm\\_medium=relat](https://www.sitepoint.com/premium/tutorial/dummy-data-for-your-empty-databases/?utm_source=sitepoint&utm_medium=relat)

## About

[Our Story \(/about-us/\)](/about-us/)

[Advertise \(/advertising/\)](/advertising/)

[Press Room \(/press/\)](/press/)

[Reference \(http://reference.sitepoint.com/css/\)](http://reference.sitepoint.com/css/)

[Terms of Use \(/legals/\)](/legals/)

[Privacy Policy \(/legals/#privacy\)](/legals/#privacy)

[FAQ \(https://sitepoint.zendesk.com/hc/en-us\)](https://sitepoint.zendesk.com/hc/en-us)

[Contact Us \(mailto:feedback@sitepoint.com\)](mailto:feedback@sitepoint.com)

[Contribute \(/write-for-us/\)](/write-for-us/)

## Visit

[SitePoint Home \(/\)](/)

[Forums \(https://www.sitepoint.com/community/\)](https://www.sitepoint.com/community/)

[Newsletters \(/newsletter/\)](/newsletter/)

[Premium \(/premium/\)](/premium/)

[References \(/sass-reference/\)](/sass-reference/)

[Shop \(https://shop.sitepoint.com\)](https://shop.sitepoint.com)

[Versioning \(https://www.sitepoint.com/versioning/\)](https://www.sitepoint.com/versioning/)

## Connect

[!\[\]\(179f167ede0522ebb4ea025b3ad78ca7\_img.jpg\) \(http://www.sitepoint.com/feed/\)](http://www.sitepoint.com/feed/) [!\[\]\(87058f19cbcad5cb0037b3939e56d0cf\_img.jpg\) \(/newsletter/\)](/newsletter/) [!\[\]\(526fc4ec6fe4fd2e502fae94e5355181\_img.jpg\) \(https://www.facebook.com/sitepoint\)](https://www.facebook.com/sitepoint) [!\[\]\(61ddba780723bc70792382ff4ec0d82f\_img.jpg\) \(http://twitter.com/sitepointdotcom\)](http://twitter.com/sitepointdotcom) [!\[\]\(a8dcbbf23251f24d07b96281abd87f14\_img.jpg\) \(https://plus.google.com/+sitepoint\)](https://plus.google.com/+sitepoint)