



## xDebug - более красивые сообщения об ошибках

Инженер-программист в БГУ

liferu.by

Вечернее, заочное 18 мес., 24 мес.



**xdebug** также улучшает отображение ошибок в PHP автоматически отображая стек вызовов рядом с каждым сообщением об ошибке или предупреждением. Это список вызовов отображает историю вызова функций до момента возникновения сообщения об ошибке. В то время как программы на PHP становятся все более и более объектно-ориентированными, ошибки чаще всего случаются глубоко внутри библиотек. Список вызовов позволит вам быстро найти тот кусочек кода, в котором произошла ошибка, и откуда этот кусочек был вызван. С версии 5.0 в PHP появилась функция **debug\_print\_backtrace()**, которая отображает список вызовов функций, но вы не можете вызвать эту функцию явно, всякий раз когда возникнет ошибка, что означает необходимость создания своего обработчика ошибок. Список вызовов, созданный xdebug, **легко читаем** в отличие от вывода функций PHP. Подобно PHP, xdebug выводит ошибки только тогда, когда параметр **display\_errors** установлен в **On** в php.ini.

(!) Notice: Undefined variable: m in C:\www\local_vars.php on line 23				
Call Stack				
#	Time	Memory	Function	Location
1	0.0279	55896	{main}()	..local_vars.php:0
2	0.0280	56144	foo()	..local_vars.php:6
3	0.0280	56392	bar()	..local_vars.php:10
4	0.0280	56696	baz()	..local_vars.php:15

Посмотрите на этот список вызовов, вы увидите, что сначала была вызвана функция foo(), затем bar(), затем baz(). В дополнение к именам вызываемых функций и места нахождения в исходном коде, xdebug выводит время и количество памяти используемые скриптом в момент вызова этих функций. Выше, мы уже научились как настраивать xdebug чтобы отображать значения суперглобальных переменных. В дополнение вы можете настроить xdebug чтобы выводить текущие значения **локальных переменных**. Для того чтобы отображать значения локальных переменных при возникновении сообщения об ошибке необходимо добавить следующую строку

```
xdebug.show_local_vars=1
```

в php.ini. Вы также можете использовать **ini\_set**.


### Информация

Это неофициальный ресурс. Если вы нашли на сайте ошибку или у вас есть интересная статья о Xdebug, пожалуйста, сообщите в разделе контакты.

(!) Notice: Undefined variable: m in C:\www\local_vars.php on line 23				
Call Stack				
#	Time	Memory	Function	Location
1	0.0100	56520	{main}()	..local_vars.php:0
2	0.0101	56728	foo()	..local_vars.php:6
3	0.0101	56952	bar()	..local_vars.php:10
4	0.0101	57256	baz()	..local_vars.php:15
Variables in local scope (#4)				
\$m	Undefined			
	\$str	= string 'Hello World' (length=11)		
	\$t	= int 3		

Итак, параметры `php.in xdebug.var_display_max_data`, `debug.var_display_max_children` и `xdebug.var_display_max_depth`, которые мы использовали выше для форматирования вывода команды `var_dump()` также влияют на формат вывода ошибок.

С помощью параметра **`xdebug.collect_params`**, вы можете настроить вывод информации о параметрах, которые передаются в функции. `xdebug.collect_params` имеет числовое значение: 0 – значит отсутствие информации, а 4 означает отображение названий переменных и полной информации по всем параметрам функций. Также доступны значения от 1 до 3, означающие различную детализацию в выводе отладочной информации.


**Notice: Undefined variable: m in C:\www\local\_vars.php on line 28**

Call Stack

#	Time	Memory	Function	Location
1	0.0091	68064	{main}()	..local_vars.php:0
2	0.0092	68736	foo( \$a = 1, \$b = 2 )	..local_vars.php:11
3	0.0092	68960	bar( \$x = 4, \$y = 1 )	..local_vars.php:15
4	0.0092	69264	baz( \$i = 7, \$k = 5 )	..local_vars.php:20

Dump \$\_SERVER

\$_SERVER['HTTP_HOST']	=	string 'localhost' (length=9)
\$_SERVER['SERVER_NAME']	=	string 'localhost' (length=9)

Variables in local scope (#4)

\$m	Undefined		
	\$i	=	int 7
	\$k	=	int 5
	\$str	=	string 'Hello World' (length=11)
	\$t	=	int 3

Здесь приведена конфигурация используемая для вывода скриншота:

```
xdebug.show_local_vars=On
xdebug.dump.SERVER=HTTP_HOST, SERVER_NAME
xdebug.dump_globals=On
xdebug.collect_params=4
```

Обратите внимание, что чем больше информации вы просите собрать для вас `xdebug`, тем больше памяти он будет потреблять, и тем больше будет время выполнения. Проверьте, что используете `xdebug` только для разработки, а не на боевых серверах. Неправильно использовать сообщения об ошибках в формате HTML на боевых серверах, не есть хорошо выводить информацию по логину и паролю к БД в том случае, если у вас не определена переменная.

В то время как суперглобальные переменные не изменяются во время выполнения (примечание переводчика – странное утверждение, видел достаточно много скриптов, в которых происходило их изменение, но не будем обсуждать это в контексте данной статьи), `xdebug` по умолчанию выводит их только вовремя первого возникновения сообщения об ошибке. Если вы хотите повторять их вывод каждый раз, воспользуйтесь следующей настройкой

Оставьте свой отзыв

`xdebug.dump_once=Off`

Обращаю внимание на то, что расширенный вывод сообщений об ошибках не работает, если вы определили собственный обработчик ошибок, используя **`register_error_handler()`**, все происходит из-за то, что xdebug использует тот же внутренний механизм. Если же вы хотите использовать собственный обработчик ошибок вы можете воспользоваться функцией **`xdebug_get_function_stack()`** для вывода списка вызова функций.

Объектно-ориентированный подход использует исключения. Так как исключения это не ошибки, xdebug не выводит список вызовов, если случилось исключение, за исключением тех случаев, когда такое исключение не перехватывается. Неперехваченное исключение это фатальная ошибка.

`xdebug.show_exception_trace=On`

Для того чтобы видеть вызов функций во время появления исключений используйте параметр выше. [оригинал статьи](#)

Видео находится у меня



Яндекс



Xdebug - незаменимый инструмент PHP программиста. Этот отладчик позволяет получить подробную информацию о работе PHP скрипта, а так же профайлинг и трассировка скриптов все это значительно облегчают работу разработчика. Xdebug создан by Derick Rethans.

tech