



Добро пожаловать на 4 часть повествования о xdebug. Сегодня мы попытаемся разобраться в отладке PHP кода с помощью xdebug. В данной статье мы полагаем, что вы уже давно установили xdebug на вашу систему, если нет [первая статья серии](#) опишет вам как это сделать.

Отладка программного обеспечения самая ненавистная работа для разработчика. Большинство используют для отладки связку `echo(print_r,var_dump)` и `exit(die)`, переходя от одной строке к другой. Однако, если ошибка появляется вновь в этом файле, требуется заново прописывать отладочные команды. Можно конечно отладку обернуть внутри конструкции `if`, которая будет работать только, когда определена какая-либо константа, например `DEBUG`. Но каждый такой `if` будет слегка замедлять производительность, да и смотреться будет в коде очень ужасно.

Как мы уже изучили во второй части нашего повествования, наличие xdebug позволяет нам создавать логи трассировки, достаточно хороший выход для данной ситуации, вам не надо будет изменять программу. Однако лог трассировки, особенно когда он создан для части программы, предоставляет нам много информации, не имеющей никакого отношения к отладке, таким образом, использование отладчика это оптимальное решение. Отладчик позволит вам остановить запуск программы на некоторое время, проверить или модифицировать текущее значение переменной, а затем продолжить выполнение программы. Запуская программу шаг за шагом, вы можете близко взглянуть как выполняется ваш код, что поможет найти вам места ошибок.

До появления функции `var_dump`, отладка приложений в PHP была проблематичной, если у вас не было денег для покупки коммерческого IDE, который поддерживал отладку. С выходом xdebug проблема отладки приложений теоретически была решена. Я написал теоретически потому что не существует хорошего и бесплатного клиента для отладки как для Windows, так и для Unix.

Однако и эта проблема была решена с выходом Eclipse PDT. Eclipse PDT это бесплатный IDE для PHP поддерживающий xdebug. Поэтому давайте, не торопясь, рассмотрим установку Eclipse PDT для того чтобы начать отладку.

### Установка Eclipse PDT

Eclipse PDT (PDT это аббревиатура от PHP Development Tools) написан на Java, и поэтому будет работать на большинстве платформ, где установлен Java Runtime Environment. Если у вас его нет, то можете скачать с [www.sun.com](http://www.sun.com). Вы можете скачать готовую версию Eclipse PDT с [www.eclipse.org/pdt](http://www.eclipse.org/pdt). Вы можете выбрать подходящую версию для вашей операционной системы.

Когда была написана эта статья, версия Eclipse PDT была R20070917. Выберите последнюю версию, нажмите скачать и сходите куда-нибудь покушать, так как размер Eclipse PDT около 100 MB. После скачивания, распакуйте файлы и можно приступать к работе.

### Как работает отладка

До того как мы погрузимся в конфигурирование xdebug и Eclipse PDT, давайте рассмотрим как работает отладка в PHP с использованием xdebug. Это поможет вам лучше понять те параметры конфигурации, которые мы будем обсуждать позже. Когда отладка включена в `php.ini`, xdebug начинает контролировать выполнение программы, это значит, что xdebug может останавливать выполнение программы в любой момент, а потом запускать ее с точки останова. Когда выполнение программы остановлено, xdebug может получить информацию о текущем состоянии программы, например прочитать текущие значения переменных. Также xdebug представляет возможность изменять значения переменных во время выполнения скрипта. Расширение xdebug – это сервер, ожидающие соединений с клиентами на определенном порту, задаваемом в конфигурации. Существует два протокола, которые могут быть использованы для коммуникации между xdebug-клиентом и xdebug-сервером (GDB и DBGp). GDB – это старый протокол, который был заменен на DBGp. Посылая команды серверу-xdebug, xdebug-клиент удаленно контролирует выполнение, сообщает PHP о приостановке запуска, выполнении следующей строчки кода или продолжении выполнения программы. Клиент обычно встроен в редактор или IDE (в нашем случае в Eclipse PDT), поэтому вы не будете иметь дело с протоколом общения напрямую.

Веб-сервер с xdebug может быть запущен на другой операционной системе, нежели xdebug-клиент. Вот почему xdebug называется удаленным отладчиком(remote debugger). Для упрощения, мы настроим сервер и клиент на одном и том же компьютере.

Существует два режима начала сессии отладки. Эти режимы контролируются в `php.ini` настройкой `xdebug.remote_mode`. Значение по умолчанию `req`, xdebug-клиент будет соединяться с сервером всякий раз когда начинается выполнение скрипта. Если вы хотите, чтобы xdebug соединялся с сервером только тогда, когда установлен breakpoint или когда в скрипте возникла ошибка, вы можете установить `xdebug.remote_mode` в `jit`. Я рекомендую оставить значение по умолчанию, это избавит вас от модификации `php.ini`.

Для того чтобы начать отладку, вы должны послать параметр `XDEBUG_SESSION_START` в скрипт как часть GET, POST или COOKIE. Значение этого параметра это название сессии отладки, которое должно быть уникальным, по имени сессии xdebug может различать различные сессии, запущенные параллельно. Для завершения сессии отладки необходимо послать скрипту команду `XDEBUG_SESSION_STOP`.

Вместо того чтобы вручную начинать и заканчивать отладочные сессии, вы можете установить специальный [firefox plugin](#), который поможет вам легко начинать и заканчивать сессии одним нажатием мышки.

Используя Eclipse PDT, вам не придется беспокоиться по поводу плагина для браузера, так как IDE берет на себя отправку необходимых параметров.

### Настройка xdebug

Сейчас давайте приступим к настройке отладки. Добавьте следующие настройки в php.ini:

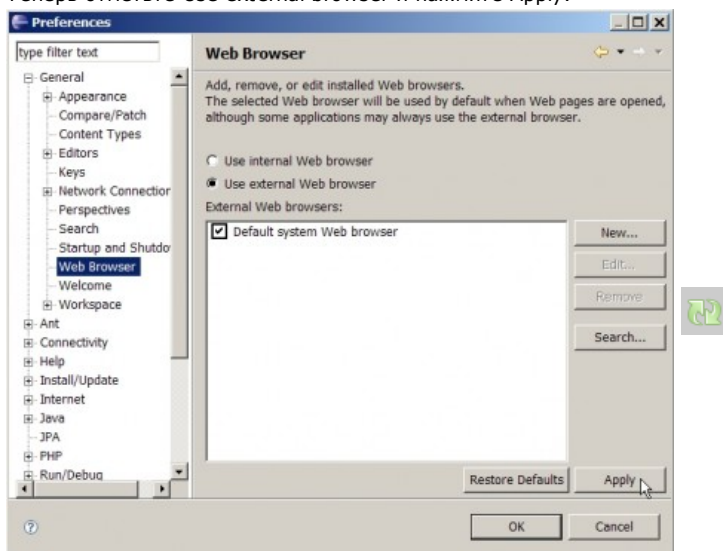
```
xdebug.remote_enable=On
xdebug.remote_host=localhost
xdebug.remote_port=9000
xdebug.remote_handler=dbgp
```

Проверьте, чтобы эти строки были добавлены после строки zend\_extension\_ts, которая загружает xdebug. Первая строка включает режим отладки. Вторая строка определяет, что клиент отладки запущен на локальном компьютере, в случае если клиент и сервер на разных компьютерах необходимо поставить название сервера или IP-адрес. Третья строка указывает порт, на котором отладочный клиент будет ожидать ответа от сервера. По умолчанию значение 9000. На это значение Eclipse настроен по умолчанию. Если вы собираетесь изменить это значение, не забудьте поменять это в настройках Eclipse и php.ini. Итак, убедитесь, что файервола не будет препятствием для осуществления отладки. При старте Eclipse, вы можете видеть сообщения, что Java пытается запустить сервер, забиндиться на порт, получить доступ к сети или выполнить какую-либо страшную операцию. Конечно, это не опасно, xdebug пытается слушать 9000 порт. Если у вас возникают какие-либо проблемы, проверьте, не блокирует ли что-то 9000 порт между клиентом и сервером. В последней строке, мы говорим, какой протокол будет использовать клиент. Eclipse PDT использует DBGp.

## Настройка Eclipse PDT

Создайте новый PHP проект в Eclipse PDT. Пусть он называется debug\_test. Создайте файл debug.php в проекте, добавьте немного кода и сохраните файл.

Сейчас давайте настроим Eclipse для отладки. Во-первых, мы настроим Eclipse для запуска проектов во внешнем браузере вместо внутреннего. Когда это будет настроено, все сессии отладки будут запущены во внешнем браузере. Использование внешнего браузера не является обязательным, однако я предпочитаю работать в Firefox вместо внутреннего браузера Eclipse. Выберите Window в меню, затем Preferences (см. скриншот внизу). Откройте поддерево General, и выберите Web Browser. Теперь отметьте Use external browser и нажмите Apply.



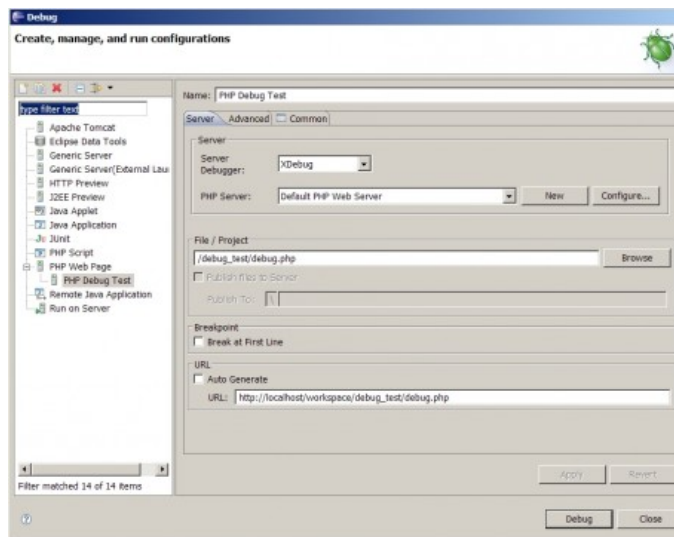
Eclipse PDT поддерживает как Zend debugger так и xdebug. По умолчанию выбран Zend debugger. Для изменения откройте поддерево PHP, затем поддерево Debug. Затем измените PHP debugger на Xdebug и нажмите Apply.

Теперь, выберите Run в меню и выберите Open Debug Dialog. Затем, дважды кликните на PHP Web Page для создания новой конфигурации отладки.

Вы будете видеть три вкладки: Server, Advanced и Common. Выберите Xdebug как Server Debugger. В поле File / Project вы должны выбрать путь к скрипту, который вы хотите отладить. Путь должен быть относительный к текущему workspace. В моей системе это /debug\_test/debug.php. Нажмите Browse и выберите debug.php в каталоге debug\_test.

Eclipse необходимо знать URL, соответствующий исходному скрипту и пути, по которому вы его вызываете. Это требуется для подсветки текущей выполняемой строки в исходном коде. Текстовое поле URL показывает URL, соответствующий названию скрипта. По умолчанию поле URL неактивно, так как активен флажок Auto Generate. Если указываемый URL не соответствует URL, который вы ввели в File / Project, снимите Auto Generate и введите правильный URL в текстовое поле URL. Если данный скрипт требует GET-параметров, вы можете ввести их в данное поле.

Не забывайте нажимать Apply для сохранения изменений. Следующий скриншот показывает как конфигурация выглядит на моей системе:

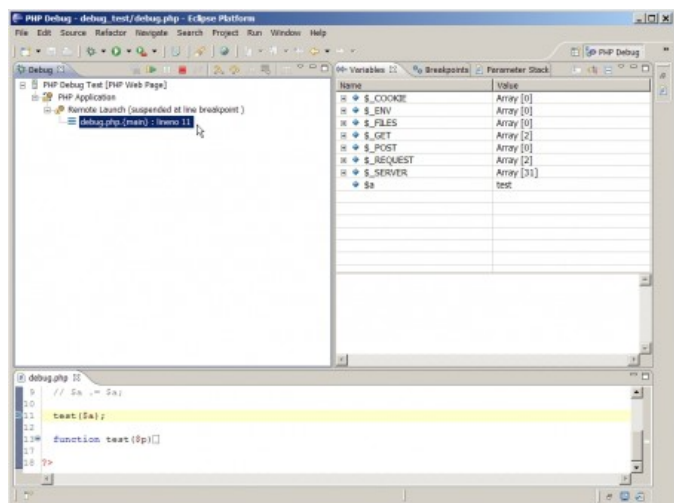


Выберите вкладку Advanced и проверьте, выбраны ли Open in Browser и Debug All Pages. Теперь можно закрыть окно настроек и начать отладку.

### Отладка PHP скрипта

Для начала отладки PHP-скрипта, выберите Run-> Debug (или нажмите F11). Eclipse спросит вас, где вы хотите видеть отладочную информацию.

Следующий скриншот показывает окно отладки Eclipse моего debug.php:



Eclipse открывает браузер. Вы не сможете ничего увидеть, потому что Eclipse по умолчанию останавливает выполнение скрипта на первой строке, как будто бы на этой строке была установлена точка останова. Если вы хотите отключить такое поведение, снимите флажок Break at First Line в секции Breakpoint в диалоге отладки в окне настроек.

Как показано на скриншоте, вы видите исходный код отлаживаемого файла, текущая строка выполняемой программы отмечена стрелочкой. В верхней правой области вы можете выбрать несколько вкладок. Вкладка Variables показывает значения всех переменных в данной области видимости. Суперглобальные переменные валидны во всех областях видимости, поэтому они всегда будут отображаться. Вкладка Breakpoints позволяет видеть и редактировать все точки останова в вашем скрипте. Eclipse запомнит все точки останова из вашего кода, всякий раз когда вы закрываете и перезапускаете Eclipse.

Вы можете продолжить выполнение программы до следующей точки останова, выполнить одну строчку кода, войти в следующую функцию или выйти из функции кликнув на соответствующую кнопку Debug в левой верхней области. Пошаговый проход очень полезен, когда вы пытаетесь локализовать место ошибки в вашем коде. Вы можете видеть как меняются значения переменных на каждом шаге.

### Изменение переменных во время выполнения

Вы также можете изменять значения переменных во время выполнения. Для изменения переменной, кликните на текущее значение, измените его и нажмите Enter.

### Точки останова

Точка останова приостанавливает выполнение программы и позволяет вам подробно рассмотреть состояние переменных, затем продолжить выполнение программы. Выполнение программы также останавливается, когда в программе выпадает исключение. Для установки точки останова, нажмите правую кнопку мыши на строке, затем выберите Toggle Breakpoints в контекстном меню. Вы можете удалить точку останова таким же образом или удалить во вкладке Breakpoints.

Вы также можете добавить точки останова по условию (conditional breakpoint). Такие точки останова будут приостанавливать выполнение программы только тогда, когда выполнится условие. Это может быть очень полезно, когда некоторый кусочек кода выполняется много раз с различными параметрами на входе. Для добавления подобной точки останова, нажмите правую кнопку мышки на изображении точки останова, выберите Breakpoint Properties.

Проверьте флаг Enable Set Condition и введите условие в текстовое поле. в моем debug.php функция test() вызывается в строке 11, и точка останова установлена в этой строке. Если мы добавим условие `$a!= ""` xdebug остановит выполнение программы в этой строке только когда локальная переменная `a` будет не пуста.

Для окончания отладки, нажмите кнопку Terminate на панели Remote Launch. Если Eclipse запускает скрипт в внешнем браузера, просто закройте его окно.

## Заключение

Удаленная отладка – это интерактивный и ненавязчивый путь поиска ошибок в ваших программах. Вместо вставки вызовов `var_dump()` в код или анализа лога трассировки на предмет отслеживания значений переменных, отладка предоставляет вам вид «под микроскопом» каждого участка вашей программы.

Следующая статья будет посвящена созданию объемлющей статистики, используя xdebug.

 [Xdebug](#)

 +23 


 13960  151  [Stefan Priebisch](#)  [great\\_boba](#) 34,1

## комментарии (30) ☐ отслеживать новые: ☐ в почте ☐ в трежере

 [mobilz](#), 8 апреля 2008 в 18:09 #  0  

непреренно ждем следующей статьи.

 [toxa](#), 8 апреля 2008 в 18:15 #  +1  

Супер ... пасиба огромное , ВСЕМИ РУКАМИ ЗА правильное и самое главное профессиональное испрвление багов :-)

 [vrazbros](#), 27 января 2009 в 12:33 #   0  

}орошая статья, сохранил на <http://www.xdebug.ru>

 [AndryX](#), 8 апреля 2008 в 19:20 #  -1  

Автопереводчик не рулит, местами так и осталось на английском






 [great\\_boba](#), 8 апреля 2008 в 23:10 #   +1  

автопереводчиком никто не пользуется, я кидая текст в ворд, перевожу и стираю переведенный текст, в некоторых случаях забываю

 [romy4](#), 8 апреля 2008 в 20:01 #  0  

не хочу показаться снобом, но то вы пишете "это избавит вас от модификации `php.ini`" в разделе "как работает...", то начинаете добавлять строки в `php.ini` в разделе "настройка".

больше замечаний нет, т.к. не работаю с Эклипсом :)

 [great\\_boba](#), 8 апреля 2008 в 23:09 #   0  

Вы невнимательно прочитали, там написано, пользуйтесь значением по умолчанию, это избавит вас от модификации

 [romy4](#), 8 апреля 2008 в 23:16 #   0  

правильно, а в следующем разделе вы модифицируете `php.ini`  
ладно, проехали :)

 [DenCh](#), 9 апреля 2008 в 07:05 #  0  


Эх... а как же быть тем кто предпочитает RHPeclipse?

Там точно есть Xdebug, но вот как его настроить / запустить так и не понятно :(

Вроде включен, а всё равно предлагает только:

- Java Applet
- Java Application
- JUnit Plug-in Test
- JUnit Test

а разобраться хочется, чувствуется что это реально удобнее чем постоянные var\_dump'ы

 **great boba**, 9 апреля 2008 в 09:34 # ☆ h ↑ 0 ↑ ↓

а каким IDE вы пользуетесь?

 **DenCh**, 10 апреля 2008 в 09:11 # ☆ h ↑ 0 ↑ ↓

Eclipse SDK 3.3.2 + PHPeclipse 1.1.8 + PHPeclipse Nightly Build 1.2.0 + Еще несколько плагинов, но они уже из другой оперы :)  
В принципе я кажется уже разобрался надо было выбрать не «Run as...» как я сделал сначала, а «Debug as...» и там уже можно настроить Xdebug.  
Буду копать.  
Я вот щас токо думаю какой ему нужно интерпретатор скамливать — `cgi-fcgi` или `cli`.  
Но это проверить не долго.  
Еще одна проблемка была — высказывала ошибка: «**Error in my\_thread\_global\_end()**» но как подсказал гугл это косяк последних версий библиотеки `libmysql.dll`, которую порекомендовли взять из **PHP 5.2.1**, мне вроде помогло.  
Автору еще раз спасибо за крайней полезные статьи... жду продолжения.

 **gvozd1989**, 9 апреля 2008 в 09:56 # ☆ 0 ↑ ↓

У меня стоит Zend for Eclipse 6, но там из установленных только ZendDebugger. Как можно туда добавить xdebug?

 **DenCh**, 10 апреля 2008 в 09:22 # ☆ h ↑ 0 ↑ ↓


К сожалению, судя по всему никак.  
Xdebug не выпускается как самодостаточный плагин к Eclipse, а только входит в состав плагинов «**PHPeclipse**» и «**PDT**»  
Принципиально ли использование именно плагина от Zend?

 **gvozd1989**, 10 апреля 2008 в 09:42 # ☆ h ↑ 0 ↑ ↓

Просто ZendStudio функциональнее, чем тот же PDT, хотя, это наверное дело привычки. :)

 **zadov**, 9 июля 2008 в 21:03 # ☆ h ↑ 0 ↑ ↓

Наковыряно в инете, проверяя работало. у меня после этого слетели 3 вьюшки. 1/ Browser, 2/PHPUnitest. 3/Profiler  
3:07AM PDT · dtaylor7  
I think Zend hates xdebug... But the ZSE contains a full PDT (it has xdebug support). So, remove (move) the plugins/com.zend.php.debug\*, then  
start the studio with "ZendStudio -clean"  
Hmm, there will be xdebug support in the PHP Debugger select... I tested it, works.

 **develop7**, 13 мая 2008 в 14:08 # ☆ h ↑ 0 ↑ ↓

[Здесь](#) смотрите последний комментарий. Проверю - отпишусь.

 **develop7**, 13 мая 2008 в 14:35 # ☆ h ↑ 0 ↑ ↓


Проаерил. Работает.

 **develop7**, 13 мая 2008 в 14:35 # ☆ h ↑ 0 ↑ ↓

\*проверил

 **gvozd1989**, 10 апреля 2008 в 10:19 # ☆ 0 ↑ ↓

Поковырял PDT - не хуже ZendStudio, но есть один косяк. Не подскажите, как сделать, чтобы при отладке русские буквы не отображались каракулями?

 **great boba**, 10 апреля 2008 в 11:19 # ☆ h ↑ 0 ↑ ↓

там внутри есть настройка кодировки, должно помочь

 **aGLex**, 1 июля 2008 в 15:13 # ☆ h ↑ 0 ↑ ↓

а можно поподробнее, а то сколько не ковырял так и не получилось настроить чтобы нормально русские буквы отображались.

 **Rhaps107**, 11 мая 2008 в 22:30 # ☆ 0 ↑ ↓

По поводу постов о Zend Studio. Я тоже его юзаю и подумал о том, как бы применить туда xdebug. Но. В Zend'е ведь есть собтсвенные дебуггер и профайлер, так что в принципе зачем тут xdebug?

 **slang**, 26 июня 2008 в 18:00 # ☆ 0 ↑ ↓

Опечатка:  
и поэтому будет работать на большинстве платформ



**realexy**, 5 августа 2008 в 13:27 # ☆

0 ↑ ↓

Автору спасибо за все переводы!

Такая ситуация: есть файл index.php, в него инклюдится header.php.

В index.php отладка идет отлично, а в файле header.php брейкпоинты не работают, хотя профайлинг и трасировка обрабатывают инклюды нормально.

Возможно ли включение брейкпоинтов во включаемых файлах?

Использую Eclipse PDT + Xdebug + Denwer.



**serivPS**, 20 ноября 2009 в 00:35 # ☆

0 ↑ ↓

Спасибо, помогло очень. Скажите, а можно использовать профилирование скрипта из PDT?



**Olegas**, 20 января 2010 в 11:53 # ☆

0 ↑ ↓

А можно ли инициировать сессию отладки из самого скрипта (например с помощью xdebug\_break), без запуска отладки на стороне Eclipse (без ручного выполнения Run -> Debug)?

Так например умеет phpED со своим собственным экстеншеном dbg-php. Его IDE всегда находится в состоянии «прослушки» отладочной информации от дебаггера.



**maskin**, 28 сентября 2010 в 22:44 # ☆

0 ↑ ↓

у меня в eclipse пусты переменные \$\_SERVER['DOCUMENT\_ROOT'] и \$\_SERVER['HTTP\_HOST']. Подскажите, как это можно исправить?



**nxn**, 24 ноября 2012 в 11:14 # ☆

0 ↑ ↓

>Расширение xdebug – это сервер, ожидающий соединений с клиентами на определенном порту

Ничего страшного, что netstat показывает открытый порт, когда IDE запущена, настроенная на этот порт, а не сам xdebug? Цитирую с офсайта: «Xdebug embedded in PHP acts like the client, and the IDE as the server». Короче, смутил меня ваш параграф :)



**iuneuniversum**, 24 мая 2013 в 10:43 # ☆

0 ↑ ↓

Возможно ли xdebug-ом дебажить разного рода сервера на php, когда клиент и сервер — два разных домена? Мне вот практика подсказывает, что нет. Ну или я еще не нашел тот самый способ. :(

Кто что скажет по этому вопросу?



**el777**, 4 февраля 2014 в 11:55 # ☆

0 ↑ ↓

Так и не понял зачем выбирать локальный файл, если идет удаленная отладка?

Я хочу посмотреть как работает TA система, а не как себя будет вести МОЙ локальный файл.

[Вирус вывел из строя все  
комплексы видеофиксации в  
Подмосковье](#)

[Вычисление фрактальной  
размерности Минковского для  
плоского изображения](#)

[Flappy Bot для Flappy Bird](#)

**Brainstorage**  
Все мозги в одном месте

**ТОСТЕР**  
Q&A-сервис для разработчиков

**ФРИЛАНСИМ**  
Заказы для фрилансеров

**ХАНТИМ**  
Вакансии для айтишников

**АВТОКАДАБРА**  
Уютная и дружелюбная