



The PHP Company

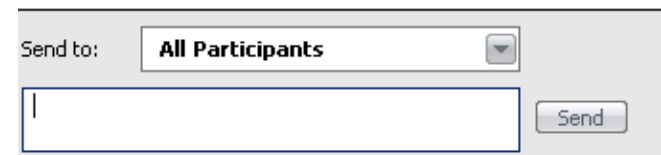
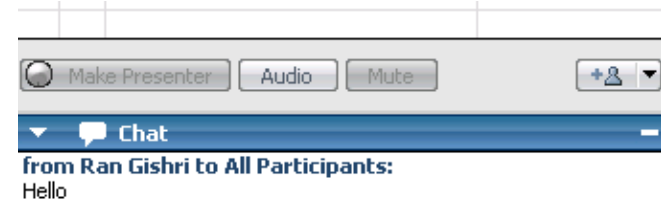
Résoudre les problèmes PHP, les meilleures (et les pires) techniques

Xavier Gorse
Architecte - Expert PHP (ELAO)
xavier.gorse@elao.com

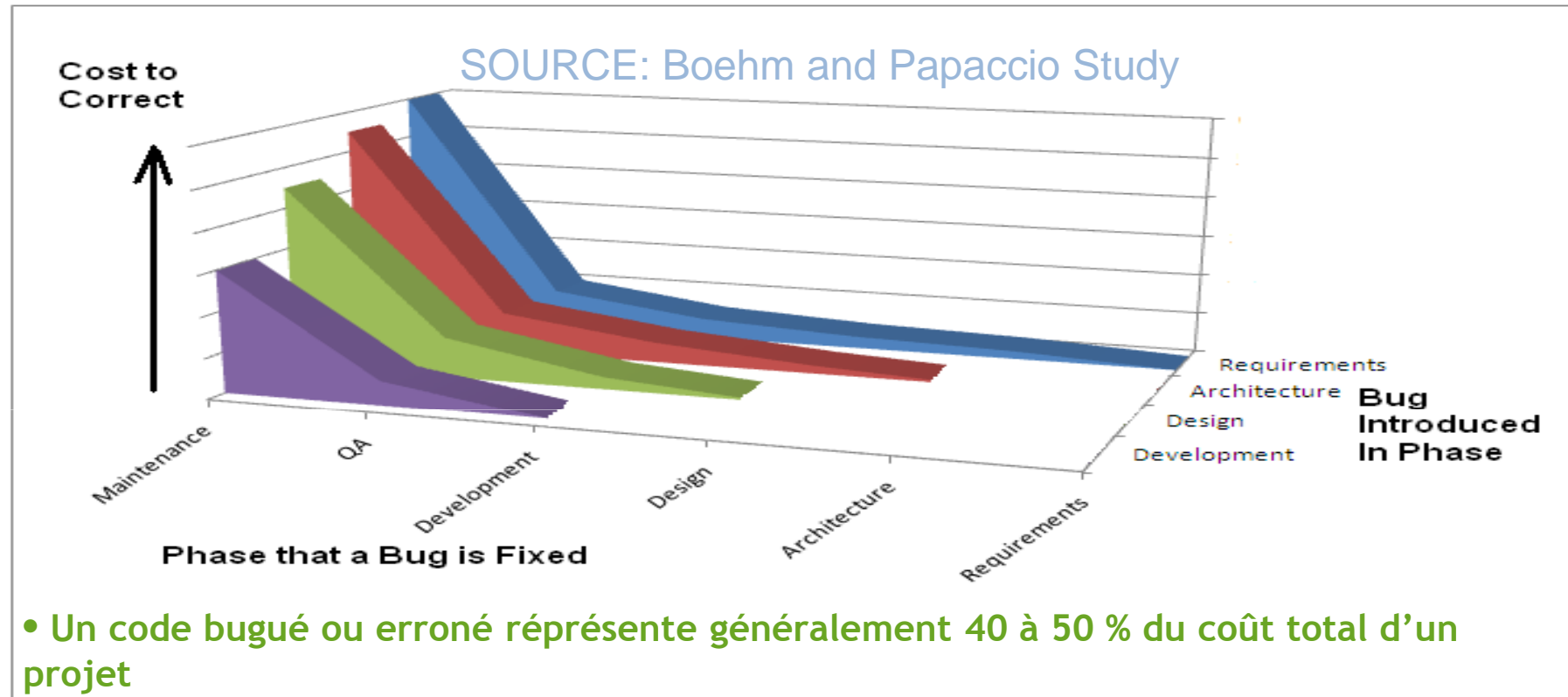


Un peu d'ordre dans la maison !

- Toutes les lignes téléphoniques sont en sourdine
- Soumettez à tout moment vos questions via le chat WebEx
- A la fin : question > réponse
- L'enregistrement sera accessible



Qualité du code : impact sur le temps de dév. et sur les coûts



- Une heure consacrée à la détection et à la correction de bugs en phase de dév. correspond à une économie de 3 à 10 heures de maintenance en production
- MAIS, comme nous le savons tous, tous les bugs ne sont pas identifiés dès la phase de dév. ou en recette.

Programme du jour



Types de problèmes rencontrés avec PHP

Logs d'erreur

Reproduction des problèmes

Traçage du code

Demonstration



Principaux problèmes rencontrés avec PHP

- **E_ERROR** (erreurs fatales)
- **E_WARNING** (erreurs posant problème mais non fatales)
- **E_NOTICE** (non critique, mais peut potentiellement poser problème)
- **E_PARSE** (pas à l'exécution, problème signalé au moment de l'interprétation)
- **E_STRICT** (recommandation)
- **E_RECOVERABLE_ERROR** (depuis PHP 5.2, erreur fatale gérable dans un bloc try/catch)
- **E_DEPRECATED** (depuis PHP 5.3)
- **E_USER_*** (événements provoqués par l'utilisateur, à l'aide de l'instruction *trigger_error()*)

Autres types de problèmes

- Erreurs logiques
 - ▶ Ex: erreur de calcul, mauvaise requête SQL, etc.
- Pertes significatives de performances
 - ▶ Ex: multiplication des accès en BD, utilisation de la mémoire, etc.
- Problèmes de sécurité
 - ▶ Ex: failles dans le code, configuration et paramètres PHP
- **Lorsqu'ils se manifestent, ces problèmes ne prennent pas la forme d'erreurs PHP**

Erreurs en PHP - Bonnes pratiques

- Afficher tous les types d'erreurs en phase de dév. et en test :
 - `error_reporting = E_ALL`
 - `display_errors = on`
- N'activer que les plus importantes en prod. :
 - `error_reporting = E_ALL & ~E_NOTICE`
 - `display_errors = off`
- Durant l'exécution, vous pouvez utiliser les fonctions `error_reporting()` et `ini_set("display_errors", value)`

Exemple

L'exemple suivant provoque un *notice*, deux avertissements (*warning*) et une erreur fatale :

```
<?php
error_reporting(E_ALL);
ini_set('display_errors', 1);
$num = 5 / $div;
require 'utility.inc';
```

- **Notice:** Undefined variable: div in test.php on line 4
- **Warning:** Division by zero in test.php on line 4
- **Warning:** require(utility.inc) [function.require]: failed to open stream: No such file or directory in test.php on line 5
- **Fatal error:** require() [function.require]: Failed opening required 'utility.inc' (include_path='.:...') in test.php on line 5

Gestion personnalisée des erreurs

- Une erreur provoquée par PHP est redirigée vers la sortie par défaut
- Ces événements peuvent être stockés dans un fichier de log en modifiant les directives **error_log** du fichier **php.ini**
- Vous pouvez également modifier le gestionnaire d'erreur de PHP à l'aide de la fonction **set_error_handler()**
 - Par exemple - envoyer un email à l'administrateur chaque fois que survient une erreur fatale
- On ne peut pas gérer les erreurs fatales avec la fonction **set_error_handler()**
- N'oubliez pas : les erreurs fatales (E_ERROR) provoquent l'arrêt de l'exécution d'un script PHP

No Need to Die...

- **Zend_Log, Firebug, FirePHP**

```
protected function _initLog() {  
    $writer = new Zend_Log_Writer_Firebug();  
    $logger = new Zend_Log($writer);  
  
    Zend_Registry::set('logger', $logger);  
}
```

```
// Controller action  
$logger = Zend_Registry::get('logger');  
$logger->info('event: ' . $eventId);  
$logger->info($event);  
$logger->warn("event starts after 3pm");  
$logger->err($exception);
```

No need to die...

- Zend_Log, Firebug, FirePHP

```
protected function _initLog() {
    $writer = new Zend_Log_Writer_Firebug();
    $logger = new Zend_Log($writer);

    Zend_Registry::set('logger', $logger);
}
```

```
// Controller action
$logger = Zend_Registry::get('logger');
$logger->info('event: ' . $eventId);
$logger->info($event);
$logger->warn("event starts after 3pm");
$logger->err($exception);
```

My Simple

Edit Event

Event Title

Date

Time

Variable Viewer

```

SimpleCal_Model_Event (
  • _id = 22
  • _data = array(
    ['start_time'] => 1264689900
    ['end_time'] => 1264691700
    ['title'] => 'Super Event'
    ['reminder'] => 1
    ['description'] => 'my description'
    ['invite'] => 'jan.burkl@zend.com'
    ['flags'] => 0
  )
  • _table = NULL
)

```

Konsole

```

http://simplecal/event/edit/id/22
event: 22
SimpleCal_Model_Event('_id'=>'22', '_data'=>array('start_time'=>'1264689900', 'end_time'=>'1264691700', 'title'=> ... ),
'_table'=> ... )
event starts after 3pm
Exception: Something went completely wrong :-(:

```

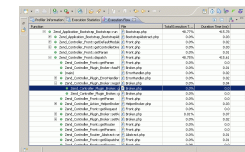
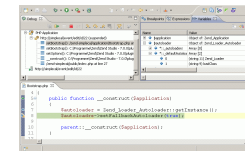
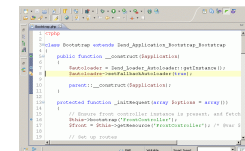
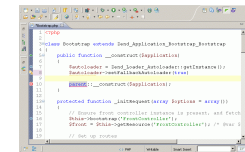
| File | Line | Instruction |
|--|------|--|
| /var/www/zend-simplecal/application/controllers/EventController.php | 53 | throw Exception() |
| /usr/local/zend/share/ZendFramework/library/Zend/Controller/Action.php | 513 | EventController->editAction() |
| /usr/local/zend/share/ZendFramework/library/Zend/Controller/Dispatcher/Standard.php | 289 | Zend_Controller_Action->dispatch('editAction') |
| /usr/local/zend/share/ZendFramework/library/Zend/Controller/Front.php | 946 | Zend_Controller_Dispatcher_Standard->dispatch(Zend_Controller_Request_Http('_paramSources'='1'=>'_POST'), '_requestUri'=>'/event/edit/id/...'), Zend_Controller_Response_Http('_body'='...', '_exceptions'=>array(), '_headers'=>...)) |
| /usr/local/zend/share/ZendFramework/library/Zend/Application/Bootstrap/Bootstrap.php | 77 | Zend_Controller_Front->dispatch() |
| /usr/local/zend/share/ZendFramework/library/Zend/Application.php | 358 | Zend_Application_Bootstrap_Bootstrap->run() |
| /var/www/zend-simplecal/public/index.php | 29 | Zend_Application->run() |

>>>

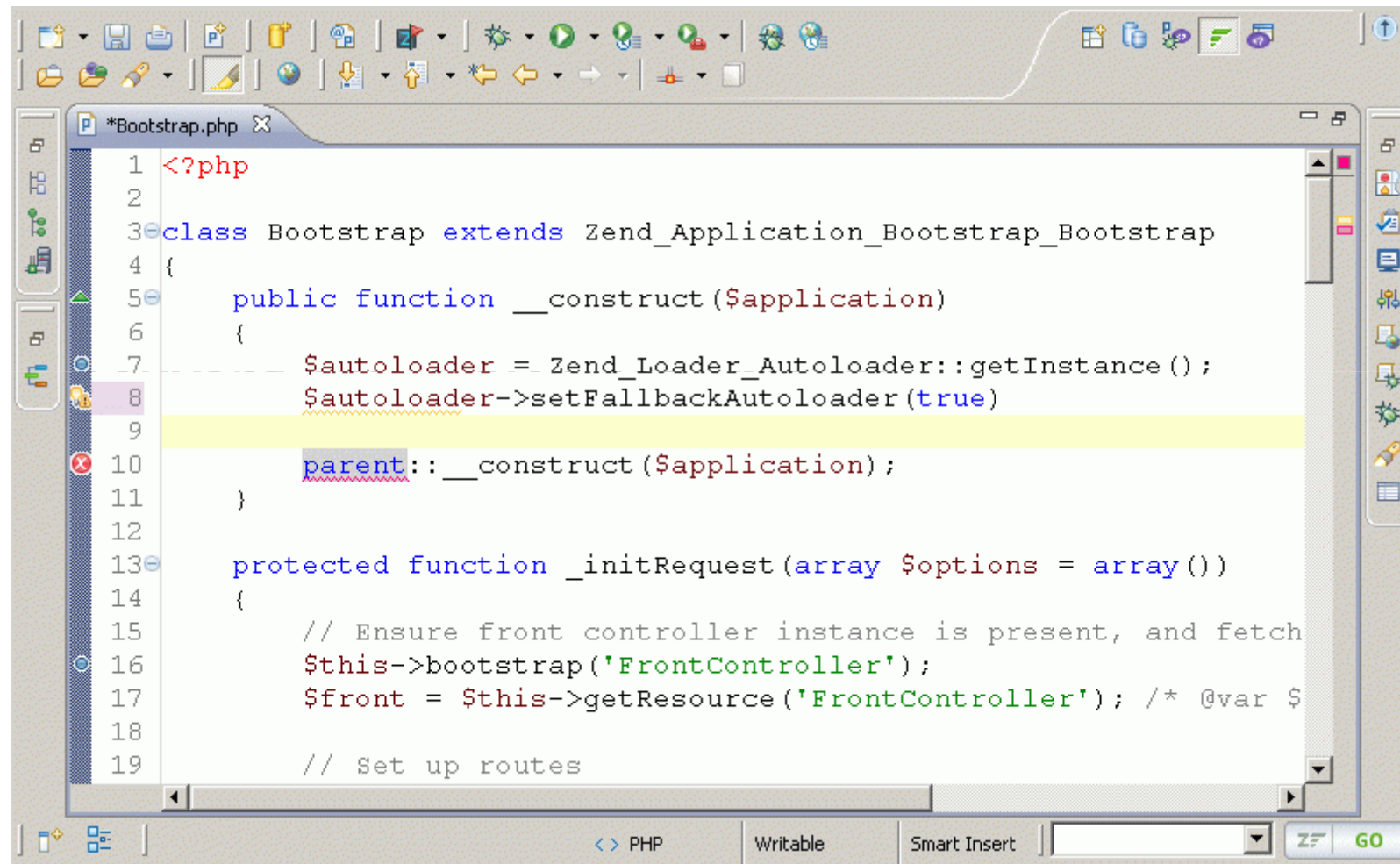
Fertig

Utiliser Zend Studio pour détecter les problèmes en dev. et en test

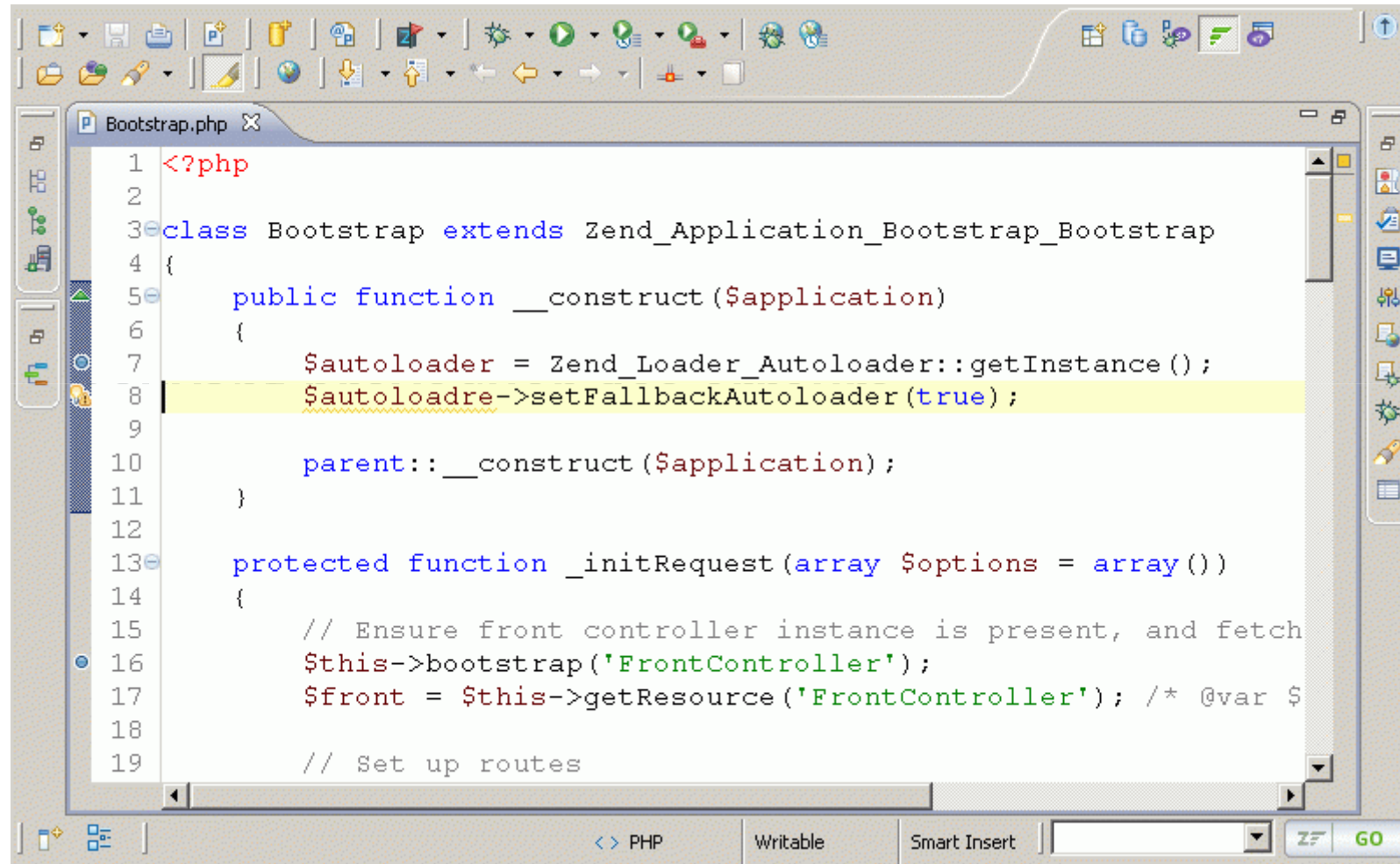
- Détecter les erreurs de syntaxe(E_PARSE) avant l'exécution
- Détecter les erreurs de logique, pouvant provoquer des erreurs fatales
- Débugger en local ou à distance sur le Zend Server
- Profiler votre code :
 - ▶ Identifier les problèmes de perf.
 - ▶ Connaître le taux de couverture du code



Utiliser Zend Studio pour détecter les problèmes en dev. et en test



Utiliser Zend Studio pour détecter les problèmes en dev. et en test



```
1 <?php
2
3 class Bootstrap extends Zend_Application_Bootstrap_Bootstrap
4 {
5     public function __construct($application)
6     {
7         $autoloader = Zend_Loader_Autoloader::getInstance();
8         $autoloader->setFallbackAutoloader(true);
9
10        parent::__construct($application);
11    }
12
13    protected function _initRequest(array $options = array())
14    {
15        // Ensure front controller instance is present, and fetch
16        $this->bootstrap('FrontController');
17        $front = $this->getResource('FrontController'); /* @var $
18
19        // Set up routes
```

Utiliser Zend Studio pour détecter les problèmes en dev. et en test

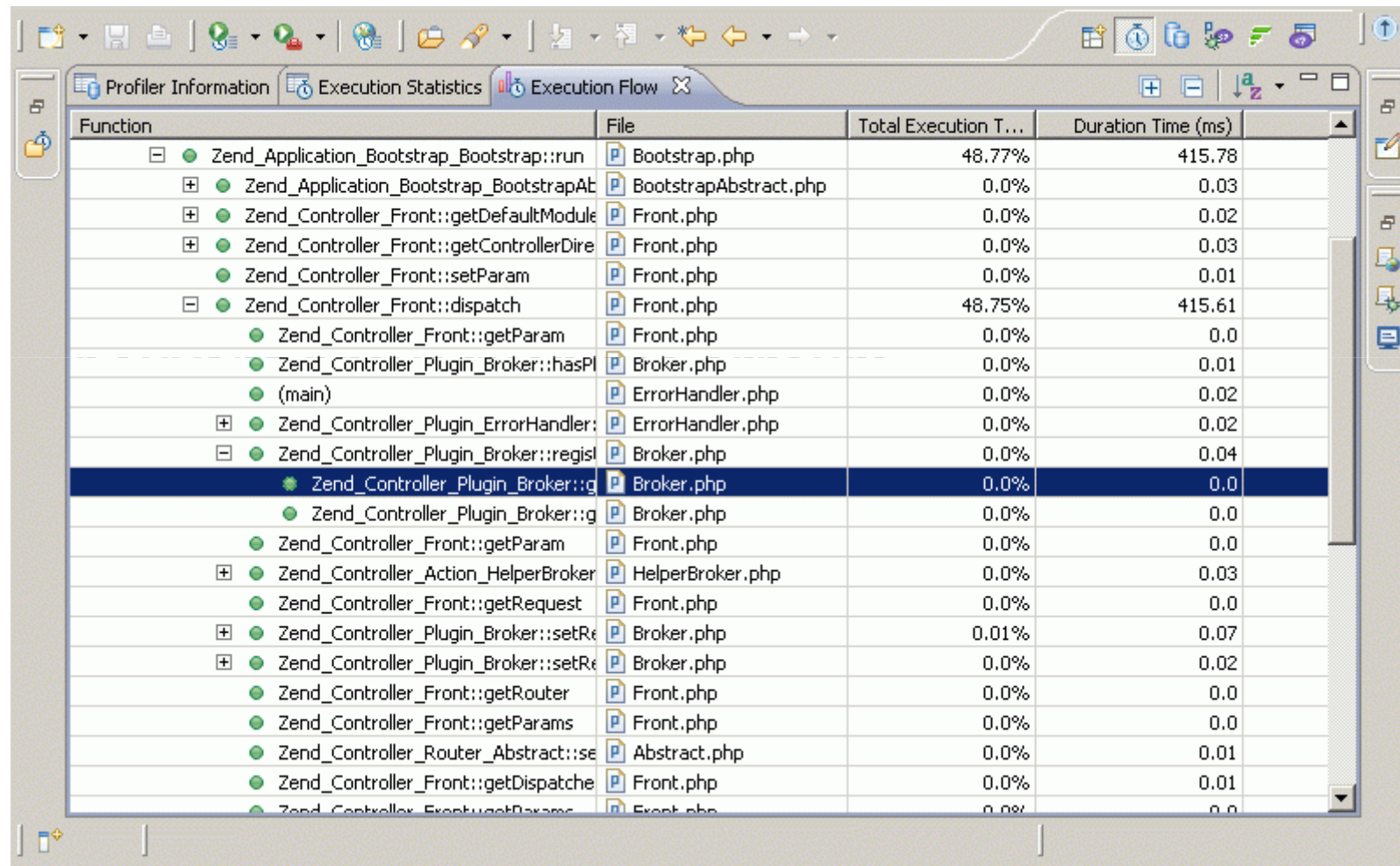
The screenshot displays the Zend Studio IDE interface during a debug session. The top toolbar contains various icons for file operations and debugging. The 'Debug' window shows a call stack for a PHP application at the URL `http://simplecal/event/edit/id/22`, which is currently suspended. The stack includes methods like `setBootstrap()` and `__construct()`. The 'Variables' window on the right shows the current state of variables:

| Name | Value |
|--------------------|-----------------------------------|
| \$application | Object of: Zend_Application |
| \$autoloader | Object of: Zend_Loader_Autoloader |
| *::_autoloaders | Array [0] |
| *::_defaultAutoloa | Array [2] |
| 0 | (string:11) Zend_Loader |
| 1 | (string:9) loadClass |

The code editor at the bottom shows the `Bootstrap.php` file with the following code:

```
4 {
5     public function __construct ($application)
6     {
7         $autoloader = Zend_Loader_Autoloader::getInstance ();
8         $autoloadre->setFallbackAutoloader (true);
9
10        parent::__construct ($application);
11    }
12 }
```

Utiliser Zend Studio pour détecter les problèmes en dev. et en test



The screenshot displays the Execution Flow window in Zend Studio, which provides a detailed view of the execution process. The window is divided into three tabs: Profiler Information, Execution Statistics, and Execution Flow. The Execution Flow tab is active, showing a hierarchical list of functions and their associated files. The table below represents the data shown in this window.

| Function | File | Total Execution T... | Duration Time (ms) |
|---|-----------------------|----------------------|--------------------|
| Zend_Application_Bootstrap_Bootstrap::run | Bootstrap.php | 48.77% | 415.78 |
| Zend_Application_Bootstrap_BootstrapAbstract::run | BootstrapAbstract.php | 0.0% | 0.03 |
| Zend_Controller_Front::getDefaultModule | Front.php | 0.0% | 0.02 |
| Zend_Controller_Front::getControllerDire | Front.php | 0.0% | 0.03 |
| Zend_Controller_Front::setParam | Front.php | 0.0% | 0.01 |
| Zend_Controller_Front::dispatch | Front.php | 48.75% | 415.61 |
| Zend_Controller_Front::getParam | Front.php | 0.0% | 0.0 |
| Zend_Controller_Plugin_Broker::hasPl | Broker.php | 0.0% | 0.01 |
| (main) | ErrorHandler.php | 0.0% | 0.02 |
| Zend_Controller_Plugin_ErrorHandler:: | ErrorHandler.php | 0.0% | 0.02 |
| Zend_Controller_Plugin_Broker::regist | Broker.php | 0.0% | 0.04 |
| Zend_Controller_Plugin_Broker::g | Broker.php | 0.0% | 0.0 |
| Zend_Controller_Plugin_Broker::g | Broker.php | 0.0% | 0.0 |
| Zend_Controller_Front::getParam | Front.php | 0.0% | 0.0 |
| Zend_Controller_Action_HelperBroker | HelperBroker.php | 0.0% | 0.03 |
| Zend_Controller_Front::getRequest | Front.php | 0.0% | 0.0 |
| Zend_Controller_Plugin_Broker::setRe | Broker.php | 0.01% | 0.07 |
| Zend_Controller_Plugin_Broker::setRe | Broker.php | 0.0% | 0.02 |
| Zend_Controller_Front::getRouter | Front.php | 0.0% | 0.0 |
| Zend_Controller_Front::getParams | Front.php | 0.0% | 0.0 |
| Zend_Controller_Router_Abstract::se | Abstract.php | 0.0% | 0.01 |
| Zend_Controller_Front::getDispatche | Front.php | 0.0% | 0.01 |
| Zend_Controller_Front::getParam | Front.php | 0.0% | 0.0 |

Utiliser les fichiers de Log de PHP pour résoudre les problèmes

- Les logs d'erreurs sont très utiles, que ce soit en dév., en test ou en production
- Cependant, il faut être conscient de leurs limites :
 - Ils ne sont d'aucune aide pour résoudre les problèmes de perf.
 - Ils ne sont d'aucune aide pour résoudre les problèmes de logique
 - Ils nécessitent de reproduire le problème (pas facile en prod !)
 - Ils fournissent une vue plutôt étroite du problème (manque le contexte)
 - Ils ne sont pas toujours faciles à exploiter

Monitoring au niveau applicatif avec Zend Server

- Zend Server Monitor surveille les événements applicatifs suivants :
 - ▶ Erreurs PHP (y compris avertissements, *notices*, exceptions non “*catchées*” ...)
 - ▶ Echecs de fonctions (définies par l'utilisateur ou natives)
 - ▶ Echecs des requêtes en base de données
 - ▶ Fonctions (natives ou non) ou requêtes SQL lentes à l'exécution
 - ▶ Requêtes serveur lentes à l'exécution
 - ▶ Consommation mémoire élevée
 - ▶ Pics mémoire et performances
 - ▶ Erreurs remontées par les composants du Zend Server



Zend Server - Suivi des événements

The screenshot displays the Zend Server Monitor interface. At the top, there are navigation tabs for Monitor, Rule Management, Server Setup, and Administration. Below these are sub-tabs for Dashboard, Events, Jobs, Queue Statistics, Code Tracing, Server Info, PHP Info, and Logs. The main content area shows a filter set to 'All Open Events' and a 'Go to event by Id:' field. A summary line indicates 'Total: 153' and 'Last refresh time: 26-Jan-2010 15:37'. The central part of the interface is a table listing individual events with columns for ID, Last Occ., Count, Generated by Rule, Origin, Severity, and Status. The table contains 18 rows of event data. At the bottom, there is a 'Change status to' dropdown set to 'Closed' and a 'Restart PHP' button.

| ID | Last Occ. | Count | Generated by Rule | Origin | Severity | Status |
|--------|--------------|-------|--|---|----------|--------|
| 000154 | 26-Jan 14:44 | 3 | Fatal PHP Error | /var/www/zend-simplecal/application/Bootstrap.php | Critical | Open |
| 000153 | 26-Jan 13:08 | 1 | Slow Request Execution (Absolute) | http://simplecal/event/edit | Warning | Open |
| 000152 | 26-Jan 11:06 | 1 | Severe High Memory Usage (Absolute) | http://simplecal/event/edit/id/8 | Critical | Open |
| 000151 | 26-Jan 10:42 | 1 | PHP Error | /usr/local/zend/share/ZendFramework/library/Zend/Log.php | Warning | Open |
| 000148 | 26-Jan 10:41 | 1 | PHP Error | /usr/local/zend/share/ZendFramework/library/Zend/Loader.php | Warning | Open |
| 000149 | 26-Jan 10:41 | 1 | PHP Error | /usr/local/zend/share/ZendFramework/library/Zend/Loader.php | Warning | Open |
| 000150 | 26-Jan 10:41 | 1 | Fatal PHP Error | /var/www/zend-simplecal/application/controllers/EventController.php | Critical | Open |
| 000083 | 26-Jan 10:39 | 4 | Fatal PHP Error | /usr/local/zend/share/ZendFramework/library/Zend/Controller/Dispatcher/Standard.php | Critical | Open |
| 000023 | 26-Jan 10:34 | 3 | Slow Request Execution (Absolute) | http://192.168.110.128/month/2010/01 | Warning | Open |
| 000146 | 26-Jan 10:32 | 4 | PHP Error | /var/www/zend-simplecal/public/index.php | Warning | Open |
| 000147 | 26-Jan 10:32 | 4 | Fatal PHP Error | /var/www/zend-simplecal/public/index.php | Critical | Open |
| 000143 | 23-Jan 13:38 | 1 | Severe Slow Request Execution (Absolute) | http://magento.zs/review/product/list/id/54/category/22/ | Critical | Open |
| 000144 | 23-Jan 13:38 | 1 | High Memory Usage (Absolute) | http://magento.zs/review/product/list/id/54/category/22/ | Warning | Open |
| 000141 | 23-Jan 13:38 | 1 | Severe Slow Request Execution (Absolute) | http://magento.zs/catalog/product/view/id/54/s/magento-red-furniture-set/category/22/ | Critical | Open |
| 000142 | 23-Jan 13:38 | 1 | Severe High Memory Usage (Absolute) | http://magento.zs/catalog/product/view/id/54/s/magento-red-furniture-set/category/22/ | Critical | Open |
| 000139 | 23-Jan 13:37 | 1 | Slow Request Execution (Absolute) | http://magento.zs/catalog/category/view/s/shirts/id/4/ | Warning | Open |

Analyse des causes

- Synthèse et affichage des événements récurrents, dans le temps
 - ▶ Ainsi, vous pouvez savoir si un problème se répète, et si c'est le cas, à quels moments
- Chaque rapport d'événement contient des informations sur le contexte, qui peuvent être très utiles pour le débogage :
 - ▶ URL, fichier, ligne, message d'erreur, nombre de fois où l'événement s'est produit, etc.
 - ▶ Informations sur la requête adressée au serveur (GET, POST, COOKIE etc.)
 - ▶ Informations concernant le serveur
 - ▶ Informations concernant la session
 - ▶ Historique (le cas échéant)

Intégration avec Zend Studio

- Zend Server s'intègre très bien à Zend Studio et permet de "rejouer" un événement :
 - ▶ en mode debugage
 - ▶ en mode profilage
 - ▶ Cela se fait en un clic depuis l'interface graphique
 - ▶ ... ou en exportant les informations sur les erreurs et en les envoyant à un développeur, depuis l'IDE
 - ▶ On peut également "rejouer" l'erreur sur un autre serveur dédié aux tests

Déployer le code PHP en production

- Le code PHP doit être exempt de toute erreur (tout du moins, ne comporter aucune erreur de type E_WARNING et E_ERROR) avant d'être publié
- Si le code ne comporte pas d'erreur, pourquoi y a-t-il encore des problèmes en prod ?
 - Environnement différent en prod
 - La charge en production est différente
 - Les clients réels utilisent l'application selon des cas de figure que les développeurs et les testeurs n'avaient pas prévus
 - Les infrastructures de production ou les systèmes de back-end peuvent changer ou tomber en panne
 - Loi de Murphy - "Tout ce qui est susceptible de dysfonctionner finit fatalement par dysfonctionner." 😊

Traçage du code - Capture des données

- **Reproduire un problème n'est pas toujours facile**
 - ▶ A cause des dépendances liées aux SESSIONS ou aux données présentes en base de données
 - ▶ En raison des dépendances liées à la charge sur le serveur
 - ▶ “Rejouer” la requête peut être risquée, voire impossible
- **Le traçage du code couvre la totalité du flux d'exécution :**
 - ▶ Le flux se présente sous la forme d'une arborescence qui affiche la liste des appels de fonction et les inclusions de fichiers,
 - ▶ les arguments passés aux fonctions et les valeurs de retour,
 - ▶ le rendu du résultat et la génération des en-têtes,
 - ▶ l'emplacement des erreurs, des exceptions et les événements recensés par le Zend Monitor,
 - ▶ Pour chacun de ces événements, le temps d'exécution et la consommation mémoire sont calculés et affichés

Code Tracing

Zend Server

http://il-pm1.zend.net:10081/ZendServer/Code-Tracing/Show-Dump/dumpId/7928.1

Tree Statistics

| Name | Time (µs) | Percent | Memory Usage | | Filename | Line |
|--|-----------|---------|--------------|---------|----------------|------|
| | | | Own | Sum | | |
| ▶ f AutoLoader::autoload() | 595 | 0.26% | 59492 | 652008 | Wiki.php | 522 |
| ▶ f PageHistory::__construct() | 14672 | 6.59% | 670088 | 1322008 | Wiki.php | 522 |
| ▼ f PageHistory::history() | 105506 | 47.41% | 782552 | 2104460 | Wiki.php | 523 |
| this: object(PageHistory#15) | | | | | | |
| ▼ f Article::getTouched() | 14012 | 6.29% | 160700 | 1482972 | PageHistory.pt | 75 |
| this: object(Article#14) | | | | | | |
| ▶ f Article::loadPageData() | 14008 | 6.29% | 161964 | 1484236 | Article.php | 3159 |
| returns: 20091006143959 | | | | | | |
| ▶ f StubObject::__call() | 857 | 0.38% | 3516 | 1486780 | PageHistory.pt | 75 |
| ▶ f wfProfileIn() | 8 | 0.00% | 240 | 1485844 | PageHistory.pt | 78 |
| ▶ f Title::getPrefixedText() | 28 | 0.01% | 92 | 1485736 | PageHistory.pt | 83 |
| ▶ f wfMsg() | 239 | 0.10% | 408 | 1486072 | PageHistory.pt | 83 |
| ▶ f OutputPage::setPageTitle() | 22123 | 9.94% | 367328 | 1853056 | PageHistory.pt | 83 |
| ▶ f wfMsg() | 225 | 0.10% | 248 | 1853108 | PageHistory.pt | 84 |
| ▶ f OutputPage::setPageTitleActionText() | 5 | 0.00% | 72 | 1852924 | PageHistory.pt | 84 |
| ▶ f OutputPage::setArticleFlag() | 4 | 0.00% | 72 | 1852948 | PageHistory.pt | 85 |
| ▶ f OutputPage::setArticleRelated() | 4 | 0.00% | 72 | 1852972 | PageHistory.pt | 86 |
| ▶ f OutputPage::setRobotPolicy() | 26 | 0.01% | 444 | 1853396 | PageHistory.pt | 87 |

Traçage du code

- **Le traçage du code peut être réalisé de différentes manières :**
 - ▶ Traçage automatisé dans le cas des événements gérés par le Zend Monitor :
 - par exemple, une erreur PHP, un temps d'exécution particulièrement long ou une consommation mémoire élevée
 - ▶ Traçage manuel à l'aide de l'interface graphique ou depuis un navigateur Web
 - ▶ Via l'API
- **Les informations pouvant faire l'objet d'un traçage :**
 - ▶ Le flux d'exécution et notamment les appels de fonctions et les inclusions de fichiers
 - ▶ Les arguments passés aux fonctions et les valeurs retournées
 - ▶ Le résultat affiché et la génération des en-têtes
 - ▶ L'emplacement des erreurs, des exceptions et des événements du Zend Monitor
 - ▶ Pour chaque noeud, le temps d'exécution et la mémoire allouée

Monitorer le Zend Framework à l'aide du Zend Server

The screenshot displays the Zend Server Monitor interface. At the top, there are navigation tabs for Monitor, Item Management, Server Setup, and Administration. Below these are sub-tabs for Dashboard, Events, Server Info, PHP Info, and Logs. The main content area shows a '10 - Custom Event' with a status of 'Reopened' and severity of 'Warning'. It occurred 3 times between 03-Dec-2009 14:48 and 10-Dec-2009 15:14. The URL is 'http://framework.log' and the source file is 'f:\home\mathew\git\zend\library\Zend\Log\Writer\ZendFramework...'. The error string is 'This is a logged custom message'. A table lists event occurrences with columns for 'Last Time' and 'Count'. The selected event details show 'User Data' with a PHP array structure. At the bottom, there are buttons for 'Zend Studio Diagnostics', 'Debug Event', 'Profile Event', 'Show File in Zend Studio', and 'Settings'. A 'Change status to' dropdown is set to 'Closed' with a 'Change' button. A 'Restart PHP' button is visible in the bottom right corner.

| Last Time | Count |
|--------------|-------|
| 10-Dec-15:14 | 1 |
| 07-Dec-14:59 | 1 |
| 03-Dec-14:54 | 1 |
| 03-Dec-14:48 | 1 |

```
array (
  'timestamp' => '2009-12-10T15:14:17+05:00',
  'priorityName' => 'WARNING',
  'info' =>
  array (
    'request' =>
    Zend_Controller_Request_WithObject [
      '_parameters' =>
```

Utiliser le traçage de code de Zend Server 5 pour résoudre des problèmes en prod.

- Zend Server détecte les erreurs et les scripts lents au moyen du moniteur d'application
- Il enregistre en direct l'exécution de l'application en production lorsque survient un problème => il n'est plus nécessaire de reproduire le problème
- Il peut être activé automatiquement ou manuellement

Temps de résolution des problèmes sans le traçage de code



Temps de résolution avec le traçage de code



Questions?

Questions?

Télécharger Zend Server:
<http://www.zend.com>

Autres vidéos et

Livre blanc sur le traçage de code :

[Zend.com>Resources>Whitepapers](http://www.zend.com/resources/whitepapers/tracing)
“Troubleshooting PHP Issues With Code Tracing”



The PHP Company

Contact Zend



United States and Canada

Corporate Headquarters
Zend Technologies Inc.
19200 Stevens Creek Blvd.
Cupertino, CA 95014

[Google map](#)

Tel: +1-408-253-8800
Fax: +1-408-253-8801

International

Zend Israel
Zend Technologies Ltd.
12 Abba Hillel Street
Ramat Gan, Israel 52506

[Google map](#)

Tel: +972-3-753-9500
Fax: +972-3-613-9671

Europe

Zend Germany
Zend Technologies GmbH
(Germany, Switzerland & Austria)
Rosenheimer Strasse 145 b-c
81671 Munich
Germany

[Google map](#)

Tel: +49-89-516199-0
Fax: +49-89-516199-20

Zend France
Zend Technologies SARL
5, rue de Rome
ZAC de Nanteuil
93110 Rosny sous Bois
France

[Google map](#)

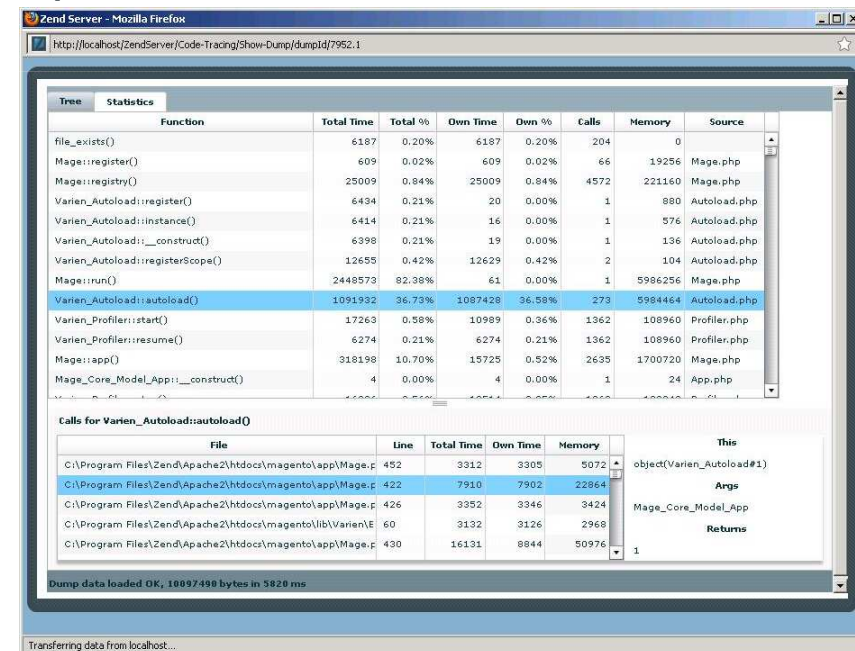
Tel: +33-1-48550200
Fax: +33-1-48123132

Zend Italy
Largo Richini 6

Zend Ireland
Zend Technologies

Trace Playback: analyse des statistiques

- Affiche les traces relatives aux fonctions :
 - ▶ Temps d'exécution : vue synthétique et vue détaillée de chaque appel de fonction
 - ▶ Consommation mémoire : vue synthétique et vue détaillée de chaque appel de fonction
 - ▶ Nombre total d'appels



The screenshot displays the 'Statistics' tab in the Zend Server interface. It shows a table with columns for Function, Total Time, Total %, Own Time, Own %, Calls, Memory, and Source. The function 'Varien_Autoload::autoload()' is highlighted in blue, indicating it is the selected function for further analysis.

| Function | Total Time | Total % | Own Time | Own % | Calls | Memory | Source |
|------------------------------------|----------------|---------------|----------------|---------------|------------|----------------|---------------------|
| file_exists() | 6187 | 0.20% | 6187 | 0.20% | 204 | 0 | |
| Mage::register() | 609 | 0.02% | 609 | 0.02% | 66 | 19256 | Mage.php |
| Mage::registry() | 25009 | 0.84% | 25009 | 0.84% | 4572 | 221160 | Mage.php |
| Varien_Autoload::register() | 6434 | 0.21% | 20 | 0.00% | 1 | 880 | Autoload.php |
| Varien_Autoload::instance() | 6414 | 0.21% | 16 | 0.00% | 1 | 576 | Autoload.php |
| Varien_Autoload::__construct() | 6398 | 0.21% | 19 | 0.00% | 1 | 136 | Autoload.php |
| Varien_Autoload::registerScope() | 12655 | 0.42% | 12629 | 0.42% | 2 | 104 | Autoload.php |
| Mage::run() | 2448573 | 82.38% | 61 | 0.00% | 1 | 5986256 | Mage.php |
| Varien_Autoload::autoload() | 1091932 | 36.73% | 1087428 | 36.58% | 273 | 5984464 | Autoload.php |
| Varien_Profiler::start() | 17263 | 0.58% | 10989 | 0.36% | 1362 | 108960 | Profiler.php |
| Varien_Profiler::resume() | 6274 | 0.21% | 6274 | 0.21% | 1362 | 108960 | Profiler.php |
| Mage::app() | 318198 | 10.70% | 15725 | 0.52% | 2635 | 1700720 | Mage.php |
| Mage_Core_Model_App::__construct() | 4 | 0.00% | 4 | 0.00% | 1 | 24 | App.php |

| File | Line | Total Time | Own Time | Memory | This |
|--|------------|-------------|-------------|--------------|---------------------------|
| C:\Program Files\Zend\Apache2\htdocs\magento\app\Mage.p | 452 | 3312 | 3305 | 5072 | object(Varien_Autoload#1) |
| C:\Program Files\Zend\Apache2\htdocs\magento\app\Mage.p | 422 | 7910 | 7902 | 22864 | |
| C:\Program Files\Zend\Apache2\htdocs\magento\app\Mage.p | 426 | 3352 | 3346 | 3424 | Mage_Core_Model_App |
| C:\Program Files\Zend\Apache2\htdocs\magento\lib\Varien.E | 60 | 3132 | 3126 | 2968 | |
| C:\Program Files\Zend\Apache2\htdocs\magento\app\Mage.p | 430 | 16131 | 8844 | 50976 | 1 |

Trace Playback- arborescence

- Lorsqu'on ouvre un fichier de trace, possibilité de "simuler" le flux d'exécution de la requête
- Localiser la cause initiale du problème grâce à une vue présentant l'enchaînement des appels de fonction
 - ▶ Mise en relief du chemin critique d'un flux d'exécution d'une fonction (durée)
 - ▶ Identifier les événements à l'origine d'une erreur

