

patError



Contents

Package patErrorHandler Procedural Elements	2
patErrorHandlerDebug.php	2
Package patErrorHandler Classes	3
Class patErrorHandlerDebug	3
Method arghDebug	3
Method niceDie	4
Method schstDebug	4
Package patError Procedural Elements	7
patError.php	7
patErrorHandler.php	8
Define PATERRORMANAGER_ERROR_CALLBACK_NOT_CALLABLE	8
Define PATERRORMANAGER_ERROR_ILLEGAL_MODE	8
Define PATERRORMANAGER_ERROR_ILLEGAL_OPTIONS	8
Package patError Classes	9
Class patError	9
Constructor patError	9
Constructor __construct	10
Method getBacktrace	10
Method getCode	10
Method getFile	11
Method getInfo	11
Method getLevel	11
Method getLine	11
Method getMessage	12
Class patErrorHandler	12
Method addIgnore	12
Method clearExpect	13
Method clearIgnore	13
Method getErrorHandling	13
Method getExpect	14
Method getIgnore	14
Method isError	14
Method popExpect	14
Method pushExpect	15
Method raise	15
Method raiseError	15
Method raiseNotice	16
Method raiseWarning	16
Method registerErrorLevel	17
Method removeIgnore	17
Method setErrorClass	18
Method setErrorHandling	18

Method translateErrorLevel	19
example cli_error.php	20
Function foo	20
example_error.php	21
example_errormanager.php	22
example_errormanager_customlevel.php	23
example_exception.php	24
example_expect.php	25
example_ignore.php	26
Class NoticeException	26
autopackage.php	27
package.php	28
Appendices	29
Appendix A - Class Trees	30
patError	30
patErrorManager	30
Appendix D - Todo List	31

Package patErrorManager Procedural Elements

patErrorHandlerDebug.php

custom patErrorManager handler for the callback error handling mode

Provides the pat-teams favourite error handlers.

\$Id: patErrorHandlerDebug.php 40 2005-05-07 08:30:24Z argh \$

- **Package** patErrorManager
- **Sub-Package** Debug
- **Access** public

Package patErrorManager Classes

Class patErrorHandlerDebug

[line 29]

custom patErrorManager handler for the callback error handling mode

- **Package** patErrorManager
- **Sub-Package** Debug
- **License** LGPL
- **Link** <http://www.php-tools.net>
- **Version** 0.1
- **Author** gERD Schaufelberger < gerd@php-tools.net>
- **Author** Stephan Schmidt < schst@php-tools.net>
- **Author** Sebastian 'The Argh' Mordziol < argh@php-tools.net>

object error function patErrorHandlerDebug::arghDebug(&\$error) *[line 162]*

Function Parameters:

- *object error* **&\$error** object

Error handler that outputs pretty debugging HTML

Displays:

- Error level
- Error Message
- Error info
- Error file
- Error line
- plus the call stack that lead to the error

The output has been inspired by Schst's debug, updated for a designer's eye.

- **Static**
- **Access** public
- **Author** Sebastian Mordziol < argh@php-tools.net>

function patErrorHandlerDebug::niceDie(\$error) [*line 38*]

Function Parameters:

- *string* **\$error** The error message to display

niceDie - outputs a nicely formatted version of the traditional die()

- **Access** public
- **Static**

object error function patErrorHandlerDebug::schstDebug(&\$error) [*line 82*]

Function Parameters:

- *object error* **&\$error** object

error handler that outputs nice debugging HTML

Displays:

- Error level
- Error Message
- Error info
- Error file
- Error line
- plus the call stack that lead to the error

The output has been inspired by Derick Rethan's xDebug.

- **TODO** console output (no HTML)
- **Static**
- **Access** public
- **Author** Stephan Schmidt < schst@php-tools.net>

Package patError Procedural Elements

patError.php

patError error object used by the **patFormsError** manager as error messages container for precise error management.

\$Id: patError.php 22 2004-04-17 20:29:56Z schst \$

- **Package** patError
- **Access** public

patErrorManager.php

patErrorManager main error management class used by pat tools for the application-internal error management. Creates patError objects for any errors for precise error management.

\$Id: patErrorManager.php 42 2007-03-11 17:41:25Z argh \$

- **Package** patError

PATERRORMANAGER_ERROR_CALLBACK_NOT_CALLABLE = 2 *[line 20]*

error definition: callback function does not exist.

PATERRORMANAGER_ERROR_ILLEGAL_MODE = 3 *[line 25]*

error definition: illegal error handling mode.

PATERRORMANAGER_ERROR_ILLEGAL_OPTIONS = 1 *[line 15]*

error definition: illegal options.

Package patError Classes

Class patError

[line 27]

patError error object used by the patFormsError manager as error messages container for precise error management.

\$Id: patError.php 22 2004-04-17 20:29:56Z schst \$

- **Package** patError
- **License** LGPL
- **Link** <http://www.php-tools.net>
- **Author** gERD Schaufelberger < gerd@php-tools.net>
- **Author** Sebastian Mordziol < argh@php-tools.net>
- **Version** 0.3
- **Author** Stephan Schmidt < schst@php-tools.net>
- **Access** public

Constructor function patError::patError(\$level, \$code, \$msg, [\$info = null]) *[line 132]*

Function Parameters:

- *int* **\$level** The error level (use the PHP constants E_ALL, E_NOTICE etc.).
- *string* **\$code** The error code from the application
- *string* **\$msg** The error message
- *string* **\$info** Optional: The additional error information.

constructor, wrapper for the upcoming PHP5 constructors for upward compatibility.

- See [patError::__construct\(\)](#)
- Access public

Constructor function `patError::__construct($level, $code, $msg, [$info = null])` [line 147]

Function Parameters:

- *int* **\$level** The error level (use the PHP constants E_ALL, E_NOTICE etc.).
- *string* **\$code** The error code from the application
- *string* **\$msg** The error message
- *string* **\$info** Optional: The additional error information.

constructor - used to set up the error with all needed error details.

- **TODO** all calls to `patErrorManager::raise*` should not be included in `backtrace`
- Access public

array function `patError::getBacktrace()` [line 255]

get the backtrace

This is only possible, if `debug_backtrace()` is available.

- See `$backtrace`
- Access public

string|integer function `patError::getCode()` [line 241]

recieve error code

- See `$code`
- Access public

string function patError::getFile() [*line 269*]

get the filename in which the error occurred

This is only possible, if debug_backtrace() is available.

- **See** \$file
- **Access** public

mixed function patError::getInfo() [*line 229*]

retrieves the additional error information (information usually only relevant for developers)

- **See** \$info
- **Access** public

int function patError::getLevel() [*line 203*]

returns the error level of the error - corresponds to the PHP error levels (E_ALL, E_NOTICE...)

- **See** \$level
- **Access** public

integer function patError::getLine() [*line 283*]

get the line number in which the error occurred

This is only possible, if debug_backtrace() is available.

- **See** \$line
- **Access** public

string function patError::getMessage() [*line 216*]

retrieves the error message

- **See** \$message
- **Access** public

Class patErrorManager

[*line 81*]

patErrorManager main error management class used by pat tools for the application-internal error management. Creates patError objects for any errors for precise error management.

- **Package** patError
- **Link** <http://www.php-tools.net>
- **TODO** implement expectError() to ignore an error with a certain code only once.
- **TODO** implement ignoreError() to ignore errors with a certain code
- **License** LGPL
- **Author** gERD Schaufelberger < gerd@php-tools.net>
- **Version** 0.3
- **Author** Stephan Schmidt < schst@php-tools.net>
- **Static**

boolean function patErrorManager::addIgnore(\$codes) [*line 401*]

Function Parameters:

- *mixed* **\$codes** either an array of error code or a single code that will be ignored in future

add error codes to be ingored

- **Access** public
- **Static**

boolean function patErrorManager::clearExpect() [*line 528*]

empty list of errors to be ignored

- **Access** public
- **Static**

boolean function patErrorManager::clearIgnore() [*line 465*]

empty list of errors to be ignored

- **Access** public
- **Static**

array function patErrorManager::getErrorHandling(\$level) [*line 346*]

Function Parameters:

- *int* **\$level** The error level to retrieve. This can be any of PHP's own error levels, e.g. E_ALL, E_NOTICE...

retrieves the current error handling settings for the specified error level.

- **Access** public

array function patErrorManager::getExpect() [*line 516*]

recieve all registerd error codes that will be ignored

- **Access public**
- **Static**

array function patErrorManager::getIgnore() [*line 453*]

recieve all registerd error codes that will be ignored

- **Access public**
- **Static**

boolean function patErrorManager::isError(&\$object) [*line 92*]

Function Parameters:

- *mixed* **&\$object**

method for checking whether the return value of a pat application method is a pat error object.

- **Access public**
- **Static**

boolean function patErrorManager::popExpect() [*line 498*]

remove top of error-codes from stack

- **Access** public
- **Static**

boolean function patErrorManager::pushExpect(\$codes) [*line 479*]

Function Parameters:

- *mixed* **\$codes** either an array of error code or a single code that will be ignored in future

add expected errors to stack

- **Access** public
- **Static**

mixed function patErrorManager::raise(\$level, \$code, \$msg, [\$info = null]) [*line 177*]

Function Parameters:

- *int* **\$level** The error level - use any of PHP's own error levels for this: E_ERROR, E_WARNING, E_NOTICE, E_USER_ERROR, E_USER_WARNING, E_USER_NOTICE.
- *string* **\$code** The application-internal error code for this error
- *string* **\$msg** The error message, which may also be shown the user if need be.
- *mixed* **\$info** Optional: Additional error information (usually only developer-relevant information that the user should never see, like a database DSN).

creates a new patError object given the specified information.

- **TODO** implement 'simple' mode that returns just false (BC for patConfiguration)
- **TODO** either remove HTML tags and entities from output or test for environment!!! in shell is ugly!
- **See** [patError](#)
- **Access** public

object function patErrorManager::raiseError(\$code, \$msg, [\$info = null]) [*line 120*]

Function Parameters:

- *string* **\$code** The application-internal error code for this error
- *string* **\$msg** The error message, which may also be shown the user if need be.
- *mixed* **\$info** Optional: Additional error information (usually only developer-relevant information that the user should never see, like a database DSN).

wrapper for the [raise\(\)](#) method where you do not have to specify the error level - a [patError](#) object with error level E_ERROR will be returned.

- See [patError](#)
- See [patErrorManager::raise\(\)](#)
- Access public
- Static

object function patErrorManager::raiseNotice(\$code, \$msg, [\$info = null]) [line 158]

Function Parameters:

- *string* **\$code** The application-internal error code for this error
- *string* **\$msg** The error message, which may also be shown the user if need be.
- *mixed* **\$info** Optional: Additional error information (usually only developer-relevant information that the user should never see, like a database DSN).

wrapper for the [raise\(\)](#) method where you do not have to specify the error level - a [patError](#) object with error level E_NOTICE will be returned.

- See [patError](#)
- See [patErrorManager::raise\(\)](#)
- Access public
- Static

object function patErrorManager::raiseWarning(\$code, \$msg, [\$info = null]) [line 139]

Function Parameters:

- *string* **\$code** The application-internal error code for this error
- *string* **\$msg** The error message, which may also be shown the user if need be.

- *mixed* **\$info** Optional: Additional error information (usually only developer-relevant information that the user should never see, like a database DSN).

wrapper for the [raise\(\)](#) method where you do not have to specify the error level - a [patError](#) object with error level `E_WARNING` will be returned.

- See [patError](#)
- See [patErrorManager::raise\(\)](#)
- Access public
- Static

boolean function `patErrorManager::registerErrorLevel($level, $name)` [line 237]

Function Parameters:

- *integer* **\$level** error level
- *string* **\$name** human-readable name

register a new error level

This allows you to add custom error levels to the built-in

- `E_NOTICE`
- `E_WARNING`
- `E_NOTICE`

You may use this level in subsequent calls to `raise()`. Error handling will be set to 'ignore' for the new level, you may change it by using `setErrorHandling()`.

You could be using PHP's predefined constants for error levels or any other integer value.

- Link <http://www.php.net/manual/en/function.error-reporting.php>
- See [patErrorManager::raise\(\)](#), [patErrorManager::setErrorHandling\(\)](#)
- Access public

boolean function `patErrorManager::removeIgnore($codes)` [line 422]

Function Parameters:

- **\$codes**

removeIgnore

- **Access** public
- **Static**

boolean function patErrorManager::setErrorClass(\$name) [*line 381*]

Function Parameters:

- *string* **\$name** classname

setErrorClass

In order to autoload this class, the filename containing that class must be named like the class itself; with an appending ".php". Although the file must be stored in the same directory as patErrorManager.php (this file)

- **Access** public

mixed function patErrorManager::setErrorHandling(\$level, \$mode, [\$options = null]) [*line 274*]

Function Parameters:

- *int* **\$level** The error level for which to set the error handling
- *string* **\$mode** The mode to use for the error handling.
- *mixed* **\$options** Optional: Any options needed for the given mode.

sets the way the patErrorManager will handle the different error levels. Use this if you want to override the default settings.

Error handling modes:

- ignore

- trigger
- verbose
- echo
- callback
- die

You may also set the error handling for several modes at once using PHP's bit operations. Examples:

- E_ALL = Set the handling for all levels
- E_ERROR | E_WARNING = Set the handling for errors and warnings
- E_ALL ^ E_ERROR = Set the handling for all levels except errors

- See [patErrorManager::getErrorHandling\(\)](#)
- Access public
- Static

string function patErrorManager::translateErrorLevel(\$level) [*line 361*]

Function Parameters:

- *integer* **\$level** error level

translate an error level

returns the human-readable name for an error level, e.g. E_ERROR will be translated to 'Error'.

- Access public

example_cli_error.php

Simple example for patError and patErrorManager

\$Id: example_cli_error.php 21 2004-04-17 20:27:33Z schst \$

- **Package** patError
- **Sub-Package** Examples
- **Link** <http://www.php-tools.net>
- **License** LGPL,
- **Copyright** PHP Application Tools
- **Author** gERD Schaufelberger < gerd@php-tools.net>

function foo() [*line 62*]

foo function raises an error

include_once '[patErrorHandlerDebug.php](#)' [*line 23*]

patErrorManagerDebug - just an example

include_once '[../patErrorManager.php](#)' [*line 18*]

patErrorManager class

example_error.php

Simple example for patError and patErrorManager

\$Id: example_error.php 21 2004-04-17 20:27:33Z schst \$

- **Package** patError
- **Sub-Package** Examples
- **Link** <http://www.php-tools.net>
- **License** LGPL,
- **Copyright** PHP Application Tools
- **Author** gERD Schaufelberger < gerd@php-tools.net>

include_once ['patErrorHandlerDebug.php'](#)*[line 22]*

patErrorManagerDebug - just an example

include_once ['./patErrorManager.php'](#)*[line 17]*

patErrorManager class

example_errormanager.php

patErrorManager example

This example demonstrates how to set the error handling for different error levels.

\$Id: example_errormanager.php 21 2004-04-17 20:27:33Z schst \$

- **Package** patError
- **Sub-Package** Examples
- **Link** <http://www.php-tools.net>
- **License** LGPL,
- **Author** Stephan Schmidt < argh@php-tools.net>
- **Access** public

include '[../patErrorManager.php](#)*[line 20]*

patErrorManager class

example_errormanager_customlevel.php

patErrorManager example

This example demonstrates how to add custom error levels

\$Id: example_errormanager_customlevel.php 21 2004-04-17 20:27:33Z schst \$

- **Package** patError
- **Sub-Package** Examples
- **Link** <http://www.php-tools.net>
- **License** LGPL,
- **Author** Stephan Schmidt < argh@php-tools.net>
- **Access** public

include '[../patErrorManager.php](#)*[line 19]*

patErrorManager class

example_exception.php

Simple example for patError and patErrorManager

\$Id: example_exception.php 36 2005-03-06 12:37:47Z schst \$

- **Package** patError
- **Sub-Package** Examples
- **Link** <http://www.php-tools.net>
- **License** LGPL,
- **Copyright** PHP Application Tools
- **Author** Stephan Schmidt <schst@php-tools.net

include_once ['./patErrorManager.php'](#)*[line 17]*

patErrorManager class

example_expect.php

Simple example for patError and patErrorManager

\$Id: example_expect.php 21 2004-04-17 20:27:33Z schst \$

- **Package** patError
- **Sub-Package** Examples
- **Link** <http://www.php-tools.net>
- **License** LGPL,
- **Copyright** PHP Application Tools
- **Author** gERD Schaufelberger < gerd@php-tools.net>

include_once ['./patErrorManager.php'](#)*[line 17]*

patErrorManager class

example_ignore.php

Simple example for patError and patErrorManager

\$Id: example_ignore.php 21 2004-04-17 20:27:33Z schst \$

- **Package** patError
- **Sub-Package** Examples
- **Link** <http://www.php-tools.net>
- **License** LGPL,
- **Copyright** PHP Application Tools
- **Author** gERD Schaufelberger < gerd@php-tools.net>

include_once ['./patErrorManager.php'](#)*[line 17]*

patErrorManager class

Class NoticeException

[line 22]

Exception class for notices

- **Package** patError
- **Sub-Package** Examples

autopackage.php

package.xml generation file for patError

This file is executed by createSnaps.php to automatically create a package that can be installed via the PEAR installer.

\$Id: autopackage.php 41 2005-09-19 14:43:27Z argh \$

- **Package** patError
- **Sub-Package** Tools
- **Author** Stephan Schmidt < schst@php-tools.net>

require_once 'PEAR/PackageFileManager.php' [*line 21*]

uses PackageFileManager

package.php

package.xml generation file for patForms

\$Id: package.php 41 2005-09-19 14:43:27Z argh \$

- **Package** patError
- **Sub-Package** Tools
- **Author** Stephan Schmidt < schst@php-tools.net>
- **Author** gERD Schaufelberger < gerd@php-tools.net>

require_once 'PEAR/PackageFileManager.php' [*line 16*]

uses PackageFileManager

Appendices

Appendix A - Class Trees

Package patError

NoticeException

- Exception
 - [NoticeException](#)

patError

- [patError](#)

patErrorManager

- [patErrorManager](#)

Package patErrorManager

patErrorHandlerDebug

- [patErrorHandlerDebug](#)

Appendix D - Todo List

In Package patError

In [patErrorManager](#)

- implement expectError() to ignore an error with a certain code only once.
- implement ignoreError() to ignore errors with a certain code

In [patErrorManager::raise\(\)](#)

- either remove HTML tags and entities from output or test for environment!!! in shell is ugly!
- implement 'simple' mode that returns just false (BC for patConfiguration)

In [patError::__construct\(\)](#)

- all calls to patErrorManager::raise* should not be included in backtrace

In Package patErrorManager

In [patErrorHandlerDebug::schstDebug\(\)](#)

- console output (no HTML)

Index

A

autopackage.php	27
<i>package.xml generation file for patError</i>	

C

constructor patError::__construct()	10
<i>constructor - used to set up the error with all needed error details.</i>	
constructor patError::patError()	9
<i>constructor, wrapper for the upcoming PHP5 constructors for upward compatibility.</i>	

E

example expect.php	25
<i>Simple example for patError and patErrorManager</i>	
example ignore.php	26
<i>Simple example for patError and patErrorManager</i>	
example exception.php	24
<i>Simple example for patError and patErrorManager</i>	
example errormanager customlevel.php	23
<i>patErrorManager example</i>	
example error.php	21
<i>Simple example for patError and patErrorManager</i>	
example errormanager.php	22
<i>patErrorManager example</i>	
example cli_error.php	20
<i>Simple example for patError and patErrorManager</i>	

F

foo()	20
<i>foo function raises an error</i>	

N

NoticeException	26
<i>Exception class for notices</i>	

P

patErrorManager::isError()	14
<i>method for checking whether the return value of a pat application method is a pat error object.</i>	
patErrorManager::popExpect()	14
<i>remove top of error-codes from stack</i>	
patErrorManager::pushExpect()	15
<i>add expected errors to stack</i>	
patErrorManager::getIgnore()	14
<i>recieve all registerd error codes that will be ignored</i>	
patErrorManager::getExpect()	14
<i>recieve all registerd error codes that will be ignored</i>	
patErrorManager::clearExpect()	13
<i>empty list of errors to be ignored</i>	
patErrorManager::clearIgnore()	13
<i>empty list of errors to be ignored</i>	
patErrorManager::getErrorHandling()	13
<i>retrieves the current error handling settings for the specified error level.</i>	
patErrorManager::raise()	15
<i>creates a new patError object given the specified information.</i>	
patErrorManager::raiseError()	15
<i>wrapper for the raise() method where you do not have to specify the error level - a patError object with error level E_ERROR will be returned.</i>	
patErrorManager::setErrorHandling()	18
<i>sets the way the patErrorManager will handle teh different error levels. Use this if you want to override the default settings.</i>	
patErrorManager::translateErrorLevel()	19
<i>translate an error level</i>	
package.php	28
<i>package.xml generation file for patForms</i>	
patErrorManager::setErrorClass()	18
<i>setErrorClass</i>	
patErrorManager::removeIgnore()	17
<i>removeIgnore</i>	
patErrorManager::raiseNotice()	16
<i>wrapper for the raise() method where you do not have to specify the error level - a patError object with error level E_NOTICE will be returned.</i>	
patErrorManager::raiseWarning()	16
<i>wrapper for the raise() method where you do not have to specify the error level - a patError object with error level E_WARNING will be returned.</i>	
patErrorManager::registerErrorLevel()	17
<i>register a new error level</i>	
patErrorManager::addIgnore()	12
<i>add error codes to be ingored</i>	
patErrorManager	12
<i>patErrorManager main error management class used by pat tools for the application-internal error management. Creates patError objects for any errors for precise error management.</i>	
patErrorManager.php	8
<i>patErrorManager main error management class used by pat tools for the application-internal error management. Creates patError objects for any errors for precise error management.</i>	

PATERRORMANAGER_ERROR_CALLBACK_NOT_CALLABLE	8
<i>error definition: callback function does not exist.</i>	
PATERRORMANAGER_ERROR_ILLEGAL_MODE	8
<i>error definition: illegal error handling mode.</i>	
patError.php	7
<i>patError error object used by the patFormsError manager as error messages container for precise error management.</i>	
patErrorHandlerDebug::schstDebug()	4
<i>error handler that outputs nice debugging HTML</i>	
patErrorHandlerDebug	3
<i>custom patErrorManager handler for the callback error handling mode</i>	
patErrorHandlerDebug::arghDebug()	3
<i>Error handler that outputs pretty debugging HTML</i>	
patErrorHandlerDebug::niceDie()	4
<i>niceDie - outputs a nicely formatted version of the traditional die()</i>	
PATERRORMANAGER_ERROR_ILLEGAL_OPTIONS	8
<i>error definition: illegal options.</i>	
patError	9
<i>patError error object used by the patFormsError manager as error messages container for precise error management.</i>	
patError::getLevel()	11
<i>returns the error level of the error - corresponds to the PHP error levels (E_ALL, E_NOTICE...)</i>	
patError::getLine()	11
<i>get the line number in which the error occurred</i>	
patError::getMessage()	12
<i>retrieves the error message</i>	
patError::getInfo()	11
<i>retrieves the additional error information (information usually only relevant for developers)</i>	
patError::getFile()	11
<i>get the filename in which the error occurred</i>	
patError::getBacktrace()	10
<i>get the backtrace</i>	
patError::getCode()	10
<i>recieve error code</i>	
patErrorHandlerDebug.php	2
<i>custom patErrorManager handler for the callback error handling mode</i>	