Logging

* Cake\Core\Log
* <https://book.cakephp.org/3.0/en/core-libraries/logging.html>

Configuration

* Config/app.php: Log

// Short classname

Log::config('debug', [

'className' => 'File',

'path' => LOGS,

'levels' => ['notice', 'info', 'debug'],

'file' => 'debug',

]);

// Fully namespaced name.

Log::config('error', [

'className' => 'Cake\Log\Engine\FileLog',

'path' => LOGS,

'levels' => ['warning', 'error', 'critical', 'alert', 'emergency'],

'file' => 'error',

]);

Log levels: https://book.cakephp.org/3.0/en/core-libraries/logging.html#using-levels

* Emergency: system is unusable
* Alert: action must be taken immediately
* Critical: critical conditions
* Error: error conditions
* Warning: warning conditions
* Notice: normal but significant condition
* Info: informational messages
* Debug: debug-level messages

Logs methods

* static Cake\Log\Log::**emergency(**$message**,**$scope = []**)**
* static Cake\Log\Log::**alert(**$message**,**$scope = []**)**
* static Cake\Log\Log::**critical(**$message**,**$scope = []**)**
* static Cake\Log\Log::**error(**$message**,**$scope = []**)**
* static Cake\Log\Log::**warning(**$message**,**$scope = []**)**
* static Cake\Log\Log::**notice(**$message**,**$scope = []**)**
* static Cake\Log\Log::**debug(**$message**,**$scope = []**)**
* static Cake\Log\Log::**info(**$message**,**$scope = []**)**
* static Cake\Log\Log::**write**($level, $message, $context = [])

Writing to Logs

Log::write('debug', 'Something did not work');

// OR

Log::debug('Errors');

// executing this inside a class using LogTrait

$this->log('Error something', 'debug');

Logging Trait

* trait Cake\Log\LogTrait
* write log using log() method

Cake\Log\LogTrait::log($msg, $level = LOG\_ERR);

Using Monolog

* Install monolog: $ composer require monolog/monolog
* After install monolog:

// config/bootstrap.php

use Monolog\Logger;

use Monolog\Handler\StreamHandler;

Log::config('default', function () {

$log = new Logger('app');

$log->pushHandler(new StreamHandler('path/to/your/combined.log'));

return $log;

});

// Optionally stop using the now redundant default loggers

Log::drop('debug');

Log::drop('error');

* Use similar methods if you want to configure a different logger for your console

// config/bootstrap\_cli.php

use Monolog\Logger;

use Monolog\Handler\StreamHandler;

Log::config('default', function () {

$log = new Logger('cli');

$log->pushHandler(new StreamHandler('path/to/your/combined-cli.log'));

return $log;

});

// Optionally stop using the now redundant default CLI loggers

Configure::delete('Log.debug');

Configure::delete('Log.error');

* Note: When using a console specific logger, make sure to conditionally configure your application logger. This will prevent duplicate log entries.