

FXneoCache

by NeoCode Software

FXneoCache v1.01 Documentation

<http://www.neocodesoftware.com>

FXneoCache v1.01 Documentation.....	1
Introduction.....	1
Changelog	1
How FXneoCache Works	2
Technical Requirements	2
Part I: Installation and Setup.....	2
Installation.....	2
Caching Engine Setup	2
Part II: Using FXneoCache	3
Enabling FXneoCache in a PHP Script	3
An Example of FXneoCache Use.....	3
FXneoCache Settings	4

Introduction

FXneoCache.php is a free, open-source add-on that speeds up your FX.php web applications by utilizing MySQL or file-based caching. FXneoCache.php was created by Neo Code Software and is available for download at the following URL:

http://www.neocodesoftware.com/fx_neocache

FXneoCache.php requires FX.php, which is available for download at:

<http://www.iviking.org/FX.php/>

FXneoCache is licensed under the Gnu General Public License.

Changelog

V1.01

Updated FX.php example book search page to utilize FXneoCache. Improved encoding of cache storage keys (thanks to Nick Salonen!).

How FXneoCache Works

FXneoCache works by storing the results of FX.php "find" and "find all" requests. After a "find" or "find all" request is made, FXneoCache does the following:

1. FXneoCache checks to see if it has a recently cached version of the request.
2. If a recently cached version exists, FXneoCache retrieves results from the cache and returns them *without having to search FileMaker*.
3. If a recently cached version doesn't exist, FXneoCache retrieves results from FileMaker then stores them for later retrieval as cached results.

FXneoCache allows you to specify "how fresh" cached results have to be. If your data is frequently changed, the default cache time of 5 minutes may be desirable. If your data infrequently changes, a cache time of 60 minutes or more may be desirable.

Technical Requirements

FXneoCache requires PHP 4.x or greater and MySQL 3.x or greater (if using MySQL-based caching). FXneoCache should work fine in Mac OS X, Microsoft Windows, or Linux.

Because FXneoCache extends FX.php, FX.php should be installed and configured as per its documentation.

Part I: Installation and Setup

Installation

After downloading FXneocache.zip, unzip the FXneocache.zip archive in the directory in which your FX.php FX directory resides. After unzipping, you should have a directory named "FXneocache".

Caching Engine Setup

FXneoCache supports two caching engines: a file-based caching engine and a MySQL caching engine. The file-based caching engine stores cached FX.php requests in individual files. The MySQL caching engine stores cached FX.php requests in a database table.

File-Based Engine Setup

By default, a directory named “fx_cache” will be automatically created the first time you use FXneoCache using the “file” engine. Please make sure that this directory can be written to, and files within it deleted, by the web server.

MySQL Engine Setup

If no database already exists, a database should be created to contain caching data (please consult MySQL’s documentation on how to create a database). By default, a table named “fx_cache” will be automatically created the first time you use FXneoCache using the “mysql” engine.

Part II: Using FXneoCache

Enabling FXneoCache in a PHP Script

Each PHP script using FXneoCache needs to include the following line near the start, after FX.php is included:

```
include_once('FXneocache/FXneocache.php');
```

In addition, if using the MySQL engine, a MySQL database connection will have to be opened before FXneoCache requests are made. The following lines will do this:

```
$link = mysql_connect('<MySQL host>', '<MySQL_user>',  
'<MySQL password>');  
mysql_select_db('<MySQL database name>');
```

Finally, if using the MySQL engine, the following line should be added at the end of your PHP script:

```
mysql_close($link);
```

An Example of FXneoCache Use

Below is an example use of FXneoCache based on the book search example that comes in the FX.php distribution. This PHP code will initialize FXneoCache, select the file caching engine, and issue a find request for any book with the word “php” in the title:

```
$BookQuery = new FXneocache($serverIP, $webCompanionPort,  
$dataSourceType);  
$BookQuery->engine = 'file';  
$BookQuery->SetDBData('Book_List.fp7', 'Book List');  
$BookQuery->AddDBParam('title', 'php');  
$BookData = $BookQuery->FMCacheFind();
```

Below is an example of a “Find All” Request. This PHP code will initialize FXneoCache, select the MySQL caching engine, and issue a request to find all books:

```
$BookQuery = new FXneocache($serverIP, $webCompanionPort,
$dataSourceType);
$BookQuery->engine = 'file';
$BookQuery->SetDBData('Book_List.fp7', 'Book List');
$BookData = $BookQuery->FMCacheFindAll();
```

Note that only the first two and last lines of the examples are different from normal use of FX.php. The resulting \$BookData array would be processed in the usual FX.php way (please consult FX.php’s documentation for details).

FXneoCache Settings

Below are explanations of FXneoCache’s 6 settings.

enabled

The *enabled* setting allows FXneoCache caching to be turned on and off for benchmarking or other purposes. The *enabled* setting should be set to 0 to disable caching and 1 to enable caching. The *enabled* setting is, by default, set to 1.

The example line below would disable caching:

```
$BookQuery->enabled = 0;
```

engine

The *engine* setting allows one to select while caching engine FXneoCache uses: “file” or “mysql”. The caching engine is, by default, “file”.

The example line below would select the “mysql” caching engine:

```
$BookQuery->engine = 'mysql';
```

lifetime

The *lifetime* setting allows one to select how long, in minutes, a result should be cached. When data changes frequently, a lower value is desirable. When data changes infrequently, a high value prevents the result from having to be re-cached. The *lifetime* setting is, by default, set to 5.

The example line below would change the lifetime to 60 minutes:

```
$BookQuery->lifetime = 60;
```

debug

The *debug* setting tells FXneoCache to print messages indicating what it is currently doing. This can help troubleshoot caching problems.

The example line below would enable debugging messages:

```
$BookQuery->debug = 1;
```

cache_directory

The *cache_directory* setting is specific to the “file” caching engine. The *cache_directory* setting allows an alternative directory to be used to store cache files. The *cache_directory* setting is, by default, set to “fx_cache”.

The example line below would change the directory where cache files are kept to “my_cache_directory”:

```
$BookQuery->cache_directory = 'my_cache_directory';
```

cache_table

The *cache_table* setting is specific to the “mysql” caching engine. The *cache_table* setting allows an alternative table to be used to store cache files. The *cache_table* setting is, by default, set to “fx_cache”.

The example line below would change the table in which cache files to kept to “my_cache_table”:

```
$BookQuery->cache_table = 'my_cache_table';
```