CSV CSV DATA MANIPULATION MADE EASY IN PHP.

Presented by The League of Extraordinary Packages

Version: 9.0

Overview

author @nyamsprod | source league/csv | release v8.2.3 | license MIT | build passing | downloads 5M

League\Csv is a simple library to ease CSV documents loading as well as writing, selecting and converting CSV records.

Usage

Parsing a document

Accessing some records from a given CSV documents.

```php
<?php

use League\Csv\Reader;
use League\Csv\Statement;

$csv = Reader::createFromPath('/path/to/your/csv/file.csv', 'r');
$csv->setHeaderOffset(0); //set the CSV header offset

//get 25 records starting from the 11th row
$stmt = (new Statement())
    ->offset(10)
    ->limit(25)
;

$records = $stmt->process($csv);
```

```php
foreach ($records as $record) {
    //do something here
}
```

Exporting a database table as a CSV document

Create and download a CSV from a `PDOStatement` object

```php
<?php

use League\Csv\Writer;

//we fetch the info from a DB using a PDO object
$sth = $dbh->prepare(
    "SELECT firstname, lastname, email FROM users LIMIT 200"
);
//because we don't want to duplicate the data for each row
// PDO::FETCH_NUM could also have been used
$sth->setFetchMode(PDO::FETCH_ASSOC);
$sth->execute();

//we create the CSV into memory
$csv = Writer::createFromFileObject(new SplTempFileObject());

//we insert the CSV header
$csv->insertOne(['firstname', 'lastname', 'email']);

// The PDOStatement Object implements the Traversable Interface
// that's why Writer::insertAll can directly insert
// the data into the CSV
$csv->insertAll($sth);

// Because you are providing the filename you don't have to
// set the HTTP headers Writer::output can
// directly set them for you
// The file is downloadable
$csv->output('users.csv');
die;
```

Importing CSV records into a database table

Importing CSV records into a database using a `PDOStatement` object

```php
<?php

use League\Csv\Reader;

//We are going to insert some data into the users table
$sth = $dbh->prepare(
    "INSERT INTO users (firstname, lastname, email) VALUES (:firstname, :lastnam
);

$csv = Reader::createFromPath('/path/to/your/csv/file.csv')
    ->setHeaderOffset(0)
;

//by setting the header offset we index all records
//with the header record and remove it from the iteration

foreach ($csv as $record) {
    //Do not forget to validate your data before inserting it in your database
    $sth->bindValue(':firstname', $record['First Name'], PDO::PARAM_STR);
    $sth->bindValue(':lastname', $record['Last Name'], PDO::PARAM_STR);
    $sth->bindValue(':email', $record['E-mail'], PDO::PARAM_STR);
    $sth->execute();
}
```

Encoding a CSV document into a given charset

When importing csv files, you don't know whether the file is encoded with `UTF-8`, `UTF-16` or anything else.

```php
<?php

use League\Csv\Reader;
use League\Csv\CharsetConverter;

$csv = Reader::createFromPath('/path/to/your/csv/file.csv', 'r');
$csv->setHeaderOffset(0);

$input_bom = $csv->getInputBOM();

if ($input_bom === Reader::BOM_UTF16_LE || $input_bom === Reader::BOM_UTF16_BE)
    CharsetConverter::addTo($csv, 'utf-16', 'utf-8');
}

foreach ($csv as $record) {
    //all fields from the record are converted into UTF-8 charset
}
```

Converting a CSV document into a XML document

Using the provided `XMLConverter` object you can easily convert a CSV document into a `DOMDocument` object.

```php
<?php

use League\Csv\XMLConverter;
use League\Csv\Reader;

$csv = Reader::createFromPath('/path/to/prenoms.csv', 'r')
$csv->setDelimiter(';');
$csv->setHeaderOffset(0);

$converter = (new XMLConverter())
    ->rootElement('csv')
    ->recordElement('record', 'offset')
    ->fieldElement('field', 'name')
;

$dom = $converter->convert($records);
$dom->formatOutput = true;
$dom->encoding = 'iso-8859-15';

echo '<pre>', PHP_EOL;
echo htmlentities($dom->saveXML());
// <?xml version="1.0" encoding="iso-8859-15"?>
// <csv>
//   <record offset="0">
//     <field name="prenoms">Anaïs</field>
//     <field name="nombre">137</field>
//     <field name="sexe">F</field>
//     <field name="annee">2004</field>
//   </record>
//   ...
//   <record offset="1099">
//     <field name="prenoms">Anaïs</field>
//     <field name="nombre">124</field>
//     <field name="sexe">F</field>
//     <field name="annee">2005</field>
//   </record>
// </csv>
```