The Filesystem Component

5.1 version edit this page (https://github.com/symfony/symfony-docs/edit/5.1/components/filesystem.rst)

The Filesystem component provides basic utilities for the filesystem.

Installation ¶

```
$ composer require symfony/filesystem
```

Note

If you install this component outside of a Symfony application, you must require the `vendor/autoload.php` file in your code to enable the class autoloading mechanism provided by Composer. Read this article (using_components.html) for more details.

Usage ¶

The `Symfony\Component\Filesystem\Filesystem` class is the unique endpoint for filesystem operations:

```
use Symfony\Component\Filesystem\Exception\IOExceptionInterface;
use Symfony\Component\Filesystem\Filesystem;

$filesystem = new Filesystem();

try {
    $filesystem->mkdir(sys_get_temp_dir().'/'.random_int(0, 1000));
} catch (IOExceptionInterface $exception) {
    echo "An error occurred while creating your directory at ".$exception->getPath();
}
```

Note

Methods `mkdir()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)`, `exists()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)`, `touch()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)`, `remove()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)`, `chmod()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)`, `chown()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` and `chgrp()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` can receive a string, an array or any object implementing `Traversable` `(https://www.php.net/manual/en/class.traversable.php)` as the target argument.

mkdir ¶

`mkdir()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` creates a directory recursively. On POSIX filesystems, directories are created with a default mode value *0777*. You can use the second argument to set your own mode:

```
$filesystem->mkdir('/tmp/photos', 0700);
```

Note

You can pass an array or any `Traversable` `(https://www.php.net/manual/en/class.traversable.php)` object as the first argument.

Note

This function ignores already existing directories.

Note

The directory permissions are affected by the current umask (https://en.wikipedia.org/wiki/Umask). Set the `umask` for your webserver, use PHP's `umask (https://www.php.net/manual/en/function.umask.php)` function or use the `chmod (https://www.php.net/manual/en/function.chmod.php)` function after the directory has been created.

exists ¶

`exists() (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` checks for the presence of one or more files or directories and returns `false` if any of them is missing:

```
// if this absolute directory exists, returns true
$filesystem->exists('/tmp/photos');

// if rabbit.jpg exists and bottle.png does not exist, returns false
// non-absolute paths are relative to the directory where the running PHP script is stored
$filesystem->exists(['rabbit.jpg', 'bottle.png']);
```

Note

You can pass an array or any `Traversable (https://www.php.net/manual/en/class.traversable.php)` object as the first argument.

copy ¶

`copy() (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` makes a copy of a single file (use `mirror() (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` to copy directories). If the target already exists, the file is copied only if the source modification date is later than the target. This behavior can be overridden by the third boolean argument:

```
// works only if image-ICC has been modified after image.jpg
$filesystem->copy('image-ICC.jpg', 'image.jpg');

// image.jpg will be overridden
$filesystem->copy('image-ICC.jpg', 'image.jpg', true);
```

touch ¶

`touch() (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` sets access and modification time for a file. The current time is used by default. You can set your own with the second argument. The third argument is the access time:

```
// sets modification time to the current timestamp
$filesystem->touch('file.txt');
// sets modification time 10 seconds in the future
$filesystem->touch('file.txt', time() + 10);
// sets access time 10 seconds in the past
$filesystem->touch('file.txt', time(), time() - 10);
```

Note

You can pass an array or any `Traversable` `(https://www.php.net/manual/en/class.traversable.php)` object as the first argument.

chown ¶

`chown()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` changes the owner of a file. The third argument is a boolean recursive option:

```
// sets the owner of the lolcat video to www-data
$filesystem->chown('lolcat.mp4', 'www-data');
// changes the owner of the video directory recursively
$filesystem->chown('/video', 'www-data', true);
```

Note

You can pass an array or any `Traversable` `(https://www.php.net/manual/en/class.traversable.php)` object as the first argument.

chgrp ¶

`chgrp()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` changes the group of a file. The third argument is a boolean recursive option:

```
// sets the group of the lolcat video to nginx
$filesystem->chgrp('lolcat.mp4', 'nginx');
// changes the group of the video directory recursively
$filesystem->chgrp('/video', 'nginx', true);
```

Note

You can pass an array or any `Traversable` `(https://www.php.net/manual/en/class.traversable.php)` object as the first argument.

chmod ¶

`chmod()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` changes the mode or permissions of a file. The fourth argument is a boolean recursive option:

```
// sets the mode of the video to 0600
$filesystem->chmod('video.ogg', 0600);
// changes the mod of the src directory recursively
$filesystem->chmod('src', 0700, 0000, true);
```

> **Note**
>
> You can pass an array or any `Traversable` (https://www.php.net/manual/en/class.traversable.php) object as the first argument.

remove ¶

`remove()` (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php) deletes files, directories and symlinks:

```
$filesystem->remove(['symlink', '/path/to/directory', 'activity.log']);
```

> **Note**
>
> You can pass an array or any `Traversable` (https://www.php.net/manual/en/class.traversable.php) object as the first argument.

rename ¶

`rename()` (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php) changes the name of a single file or directory:

```
// renames a file
$filesystem->rename('/tmp/processed_video.ogg', '/path/to/store/video_647.ogg');
// renames a directory
$filesystem->rename('/tmp/files', '/path/to/store/files');
// if the target already exists, a third boolean argument is available to overwrite.
$filesystem->rename('/tmp/processed_video2.ogg', '/path/to/store/video_647.ogg', true);
```

symlink ¶

`symlink()` (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php) creates a symbolic link from the target to the destination. If the filesystem does not support symbolic links, a third boolean argument is available:

```
// creates a symbolic link
$filesystem->symlink('/path/to/source', '/path/to/destination');
// duplicates the source directory if the filesystem
// does not support symbolic links
$filesystem->symlink('/path/to/source', '/path/to/destination', true);
```

readlink ¶

`readlink()` (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php) read links targets.

PHP's `readlink` (https://www.php.net/manual/en/function.readlink.php) function returns the target of a symbolic link. However, its behavior is completely different under Windows and Unix. On Windows systems, `readlink()` resolves recursively the children links of a link until a final target is found. On Unix-based systems `readlink()` only resolves the next link.

The `readlink()` (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php) method provided by the Filesystem component always behaves in the same way:

```
// returns the next direct target of the link without considering the existence of the target
$filesystem->readlink('/path/to/link');

// returns its absolute fully resolved final version of the target (if there are nested links,
$filesystem->readlink('/path/to/link', true);
```

Its behavior is the following:

```
public function readlink($path, $canonicalize = false)
```

- When `$canonicalize` is `false`:
  - if `$path` does not exist or is not a link, it returns `null`.
  - if `$path` is a link, it returns the next direct target of the link without considering the existence of the target.
- When `$canonicalize` is `true`:
  - if `$path` does not exist, it returns null.
  - if `$path` exists, it returns its absolute fully resolved final version.

makePathRelative ¶

`makePathRelative()` (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php) takes two absolute paths and returns the relative path from the second path to the first one:

```
// returns '../'
$filesystem->makePathRelative(
    '/var/lib/symfony/src/Symfony/',
    '/var/lib/symfony/src/Symfony/Component'
);
// returns 'videos/'
$filesystem->makePathRelative('/tmp/videos', '/tmp')
```

mirror ¶

`mirror()` (https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php) copies all the contents of the source directory into the target one (use the `copy()`

`(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` method to copy single files):

```
$filesystem->mirror('/path/to/source', '/path/to/target');
```

isAbsolutePath ¶

`isAbsolutePath()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` returns `true` if the given path is absolute, `false` otherwise:

```
// returns true
$filesystem->isAbsolutePath('/tmp');
// returns true
$filesystem->isAbsolutePath('c:\\Windows');
// returns false
$filesystem->isAbsolutePath('tmp');
// returns false
$filesystem->isAbsolutePath('../dir');
```

tempnam ¶

`tempnam()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` creates a temporary file with a unique filename, and returns its path, or throw an exception on failure:

```
// returns a path like : /tmp/prefix_wyjgtF
$filesystem->tempnam('/tmp', 'prefix_');
// returns a path like : /tmp/prefix_wyjgtF.png
$filesystem->tempnam('/tmp', 'prefix_', '.png');
```

New in version 5.1: The option to set a suffix in `tempnam()` was introduced in Symfony 5.1.

dumpFile ¶

`dumpFile()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` saves the given contents into a file. It does this in an atomic manner: it writes a temporary file first and then moves it to the new file location when it’s finished. This means that the user will always see either the complete old file or complete new file (but never a partially-written file):

```
$filesystem->dumpFile('file.txt', 'Hello World');
```

The `file.txt` file contains `Hello World` now.

appendToFile ¶

`appendToFile()` `(https://github.com/symfony/symfony/blob/5.1/src/Symfony/Component/Filesystem/Filesystem.php)` adds new contents at the end of some file:

```
$filesystem->appendToFile('logs.txt', 'Email sent to user@example.com');
```

If either the file or its containing directory doesn't exist, this method creates them before appending the contents.

Error Handling ¶

Whenever something wrong happens, an exception implementing `Symfony\Component\Filesystem\Exception\ExceptionInterface` or `Symfony\Component\Filesystem\Exception\IOExceptionInterface` is thrown.

> **Note**
>
> An `Symfony\Component\Filesystem\Exception\IOException` is thrown if directory creation fails.

Latest from the Symfony Blog

The Symfony 5 Book, The Fast Track, available online and for free (/blog/the-symfony-5-book-the-fast-track-available-online-and-for-free)
July 29, 2020

They Help Us Make Symfony

Thanks **Schuyler Jager (https://connect.symfony.com/profile/sjager)** for being a Symfony contributor (/contributors).
4 commits · **18** lines

Get Involved in the Community

A passionate group of over 600,000 developers from more than 120 countries, all committed to helping PHP surpass the impossible.

Getting involved →