Paths

Join paths, resolve them and keep a path inside a base directory with Path

Anonymous · 05/11/2026

Paths

```
use Derafu\File\Path;
```

Method	What it does
`Path::assertValid($path)`	Refuses a path with a null character.
`Path::join(...$parts)`	Joins parts with one separator.
`Path::resolve($path)`	The absolute real path of something that exists.
`Path::resolveWithin($base, $relative)`	The real path of something that is inside a directory.
`Path::extension($path)`	The extension, without the point.
`Path::filename($path)`	The name without directory and without the last extension.

Valid paths

PHP stops with a `ValueError` on a path with a null character in functions such as `unlink()` or `mkdir()`. `assertValid()` throws a `PathException` instead, and every method of the package that receives a path applies it.

Join

```
Path::join('a/', '/b', '', 'c.txt');   // 'a/b/c.txt'
Path::join('/var', 'app');             // '/var/app'
Path::join('/', 'tmp');                // '/tmp'
Path::join('a', '..', 'b');            // 'a/../b' (the parts are not resolved)
Path::join();                          // ''
```

The empty parts are left out, the first part keeps its leading separator, and the separators at the end of a part and at the start of the next ones are not repeated. Backslashes at the joint are separators too.

Resolve

```
Path::resolve('/var/app/../app/data');   // '/var/app/data'
```

It resolves `.`, `..` and symbolic links. A path that does not exist is a `FileNotFoundException`, and so is an empty path: it is not the current directory.

Resolve within a directory

```
$file = Path::resolveWithin('/var/app/uploads', $name);
```

It is the check that a path that comes from a request needs before the file is read. The relative path is taken from the base directory, even if it starts with a separator (`/etc` is `<base>/etc`), and its `..` and the symbolic links that it goes through are resolved **before** checking, so neither can take it out of the directory.

```
Path::resolveWithin('/var/app/uploads', 'a/../b.txt');          //
'/var/app/uploads/b.txt'
Path::resolveWithin('/var/app/uploads', '');                    // '/var/app/uploads'
Path::resolveWithin('/var/app/uploads', '../secret.txt');
// PathException: The path ../secret.txt is outside of /var/app/uploads.
```

Case	Exception
The base directory or the path do not exist.	`FileNotFoundException`
The real path is outside of the directory (by `..` or by a link).	`PathException`
A path has a null character.	`PathException`

A symbolic link that stays inside the directory is accepted. A sibling directory whose name starts like the base (`uploads-old` next to `uploads`) is outside.

Extension and name

```
Path::extension('/a/report.XML');       // 'XML' (as it is written)
Path::extension('/a/archive.tar.gz');   // 'gz'
Path::filename('/a/archive.tar.gz');    // 'archive.tar'
Path::extension('/a/README');           // ''
```

Last updated on 05/11/2026