Temporary Files

Run code with a temporary file that is always deleted, with TempFile

Anonymous · 05/11/2026

Temporary Files

```
use Derafu\File\TempFile;
```

`TempFile::with($content, $callback, $extension = '', $prefix = 'derafu_')` creates a temporary file with a content, runs the callback with its path, deletes the file and returns what the callback returned.

```
$pages = TempFile::with($pdf, function (string $path): int {
    return countPages($path);          // A tool that needs a file.
}, extension: 'pdf');

$xlsx = TempFile::with('', function (string $path): string {
    writeSpreadsheet($path);           // The callback writes the file...
    return File::read($path);          // ...and reads it back.
}, 'xlsx');
```

Parameter	Meaning
`$content`	The content of the file. It can be empty, for a callback that writes the file.
`$callback`	`callable(string): T`. It receives the path of the file.
`$extension`	The extension, with or without the point. It can have more than one (`xml.gz`).
`$prefix`	The start of the name of the file.

The file is in the temporary directory of the system, with a name that no other file has. It is deleted when the callback returns and also when it throws, and then the exception of the callback is the one that goes up. The callback can delete the file itself.

An extension that has anything else than letters, numbers and points between them, or a prefix with anything else than letters, numbers, points, hyphens and underscores, is a `TranslatableInvalidArgumentException`: neither can carry a separator out of the temporary directory. A file that can not be created, named or written is a `FileException`.

To avoid the file altogether when the goal is a ZIP archive, see `Zip::fromContents()`.

Last updated on 05/11/2026