Archiver efficacement de grands volumes de données grâce aux monades

→ Baptiste Langlade

→ Lyon

→ 10+ ans XP

→ ~100 packages Open Source


efalia

GED

→ Documents

→ Métadonnées

→ 1..N binaires

Archivage

→Archive

→CSV des documents

→Dossiers contenant les binaires

Contraintes

Comment on fait ?

Streaming de données

```php
$file = \fopen('names.txt', 'r');

while ($line = \fgets($file)) {
    echo $line;
}
```

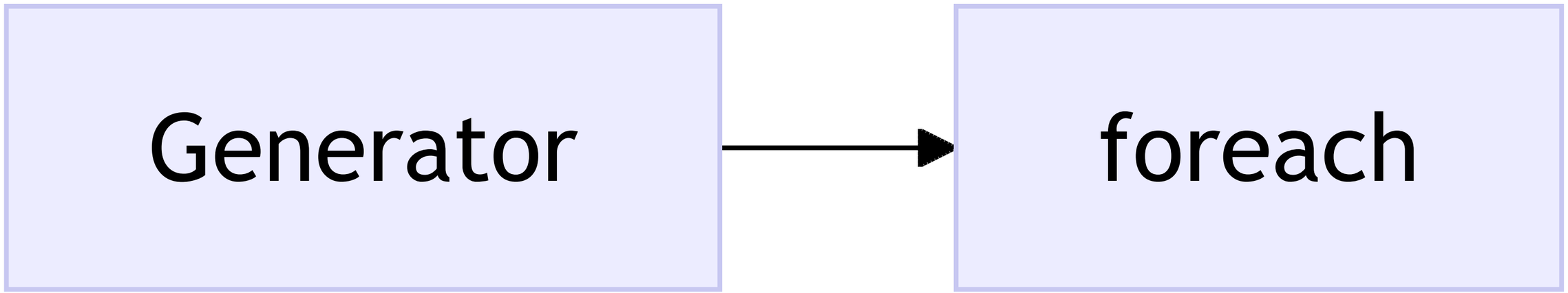
alice
bob
jane
john
etc...

```php
/** @var \Generator<string> */
$stream = function(): \Generator {
    $file = \fopen('names.txt', 'r');

    while ($line = \fgets($file)) {
        yield $line;
    }
};
```

```
foreach ($stream() as $line) {
    echo $line;
}
```

alice
bob
jane
john
etc...


Generator
foreach


Generator
Logic
foreach

```php
/**
 * @param callable(): \Generator<string> $stream
 * @var \Generator<string>
 */
$trim = function(callable $stream): \Generator {
    foreach ($stream() as $line) {
        yield \rtrim($line, "\n");
    }
};
```

```
foreach ($trim($stream) as $line) {
    echo $line.",\n";
}
```

```php
/**
 * @param callable(): \Generator<string> $stream
 * @var \Generator<string>
 */
$trim = function(callable $stream): \Generator {
    foreach ($stream() as $line) {
        yield \rtrim($line, "\n");
    }
};
```

```
foreach ($hello($capitalize($trim($stream))) as $line) {
    echo $line.",\n";
}
```

Monades

```
composer require innmind/immutable
```

```php
use Innmind\Immutable\Sequence;

/** @var Sequence<string> */
$stream = Sequence::lazy(function() {
    $file = \fopen('names.txt', 'r');

    while ($line = \fgets($file)) {
        yield $line;
    }
});
```

```php
$stream->foreach(function(string $line) {
    echo $line;
});
```

```php
/** @var Sequence<string> */
$trimmed = $stream->map(fn(string $line) => \rtrim($line, "\n"));
$trimmed->foreach(function(string $line) {
    echo $line.",\n";
});
```

`->map()`	`->flatMap()`
`->add()`	`->append()`
`->filter()`	`->aggregate()`
`->zip()`	etc...

Style monadique

Cas d'usage

→ SQL

→ Filesystem

SQL

```
/** @var Sequence<array> */
$rows = function(string $query): Sequence {};
```

ORM

```php
/** @var Sequence<Entity> */
$entities = function(): Sequence {};
```

```
composer require formal/orm
```

```php
/** @var Sequence<Document> */
$documents = $orm
    ->repository(Document::class)
    ->all()
    ->sequence();
```

10 et 11 octobre 2024
Hotel New York - TAOM
Disneyland Paris

FORUMPHP
PARIS2024

event.afup.org

ET SI ON REPENSAIT LES ORMS ?

Baptiste LANGLADE

afup

Fichier

```php
final class File
{
    public function __construct(
        private string $name,
        /** @var Sequence<string> */
        private Sequence $content,
    ) {}
}
```

Dossier

```
final class Directory
{
    public function __construct(
        private string $name,
        /** @var Sequence<File|Directory> */
        private Sequence $content,
    ) {}
}
```

```
composer require innmind/filesystem
```

```php
use Innmind\Filesystem\File;
use Innmind\Filesystem\File\Content;
use Innmind\Immutable\Sequence;
use Innmind\Immutable\Str;

$file = File::named(
    'data.csv',
    Content::ofChunks(
        Sequence::lazy(fn() => yield from [
            "line, 1\n",
            "line, 2\n",
            "etc...",
        ]),
    ),
);
```

```php
use Innmind\Filesystem\Directory;

$directory = Directory::named(
    'files',
    Sequence::lazy(fn() => yield from [
        File::named('something', $content),
        Directory::named(...$args),
        // etc...
    ]),
);
```

Cas d'usage

- → Archive
  - → CSV des documents
  - → Dossiers contenant les binaires

```php
$csv = File::named(
    'documents.csv',
    Content::ofChunks(
        $orm
            ->repository(Document::class)
            ->all()
            ->sequence()
            ->map(fn(Document $document): string => $document->toCsvLine()),
    ),
);
```

```php
use Innmind\Filesystem\Adapter\Filesystem;
use Innmind\Filesystem\Name;
use Innmind\Url\Path;
use Innmind\Immutable\Predicate\Instance;

$fetch = function(Document $document): Directory {
    return Filesystem::mount(Path::of('var/data/'))
        ->get(Name::of($document->id()->toString()))
        ->keep(Instance::of(Directory::class));
};
```

```
$binaires = Directory::named(
    'binaires',
    $orm
        ->repository(Document::class)
        ->all()
        ->sequence()
        ->map($fetch),
);
```

```
$archive = Directory::named(
    'archive',
    Sequence::lazy(fn() => yield from [
        $csv,
        $binaires,
    ]),
);
```

Tar

```
directory/file.txt
line 1
line 2
directory/image.png
binary
directory/sub/file.ext
content
```

```
composer require innmind/encoding
```

```php
use Innmind\Encoding\Tar;

$tar = Tar::encode();

$archive = Directory::named(...$args);

/** @var \Innmind\Filesystem\File\Content */
$archive = $tar($archive);
```

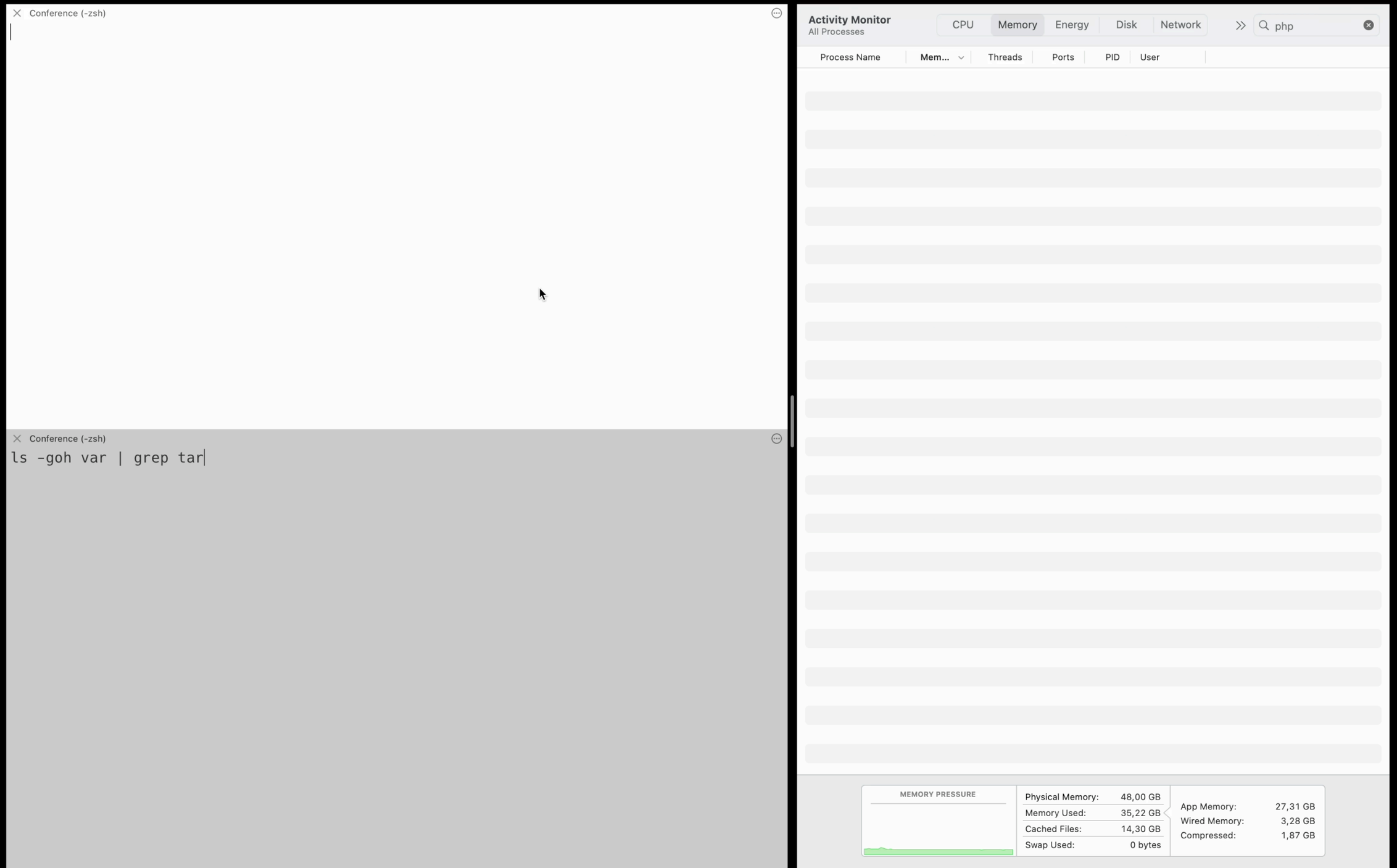
```
$documents = fetchDocuments($orm);
$archive = Directory::named(
    'archive',
    Sequence::lazy(fn() => yield from [
        toCsv($documents),
        binaires($documents),
    ]),
);
$archive = $tar($archive);
```

```php
use Symfony\Component\HttpFoundation\StreamedResponse;

new StreamedResponse(
    fn() => $archive
        ->chunks()
        ->foreach(function(string $chunk) {
            echo $chunk;
            \flush();
        });
);
```

```
archive/documents.csv
document 1, métadonnée, etc...
document 2, métadonnée, etc...
archive/binaires/uuid-document-1/v1.bin
binaire chunk 1
binaire chunk 2
etc...
archive/binaires/uuid-document-2/v1.bin
binaire
```

Demo


Conference (-zsh)
Conference (-zsh)
ls -goh var | grep tar
Activity Monitor
All Processes
CPU
Memory
Energy
Disk
Network
php
Process Name
Mem...
Threads
Ports
PID
User
MEMORY PRESSURE
Physical Memory: 48,00 GB
Memory Used: 35,22 GB
Cached Files: 14,30 GB
Swap Used: 0 bytes
App Memory: 27,31 GB
Wired Memory: 3,28 GB
Compressed: 1,87 GB

Statistiques

→100k documents

→~80Go

→~45 minutes

→~45Mo/s

→~40Mo de RAM

→~100 lignes de code

Stateless

Innmind

Search

Home Philosophy Getting started Use cases Testing Tools Other Packages Blog

Welcome to Innmind

https://innmind.org

Innmind bridges Object Oriented Programming and Functional Programming in a coherent ecosystem to bring high level abstraction to life.

This documentation will show you how to move from simple scripts all the way to distributed systems (and all the steps in between) by using a single way to code.

If you've seen modern Java, C#, Rust, Swift a

Sneak peek

The code below shows how the declarative nature of

```
$os
    ->filesystem()
    ->mount(Path::of('somewhere/data/'))
    ->get(Name::of('avatars'))
    ->keep(Instance::of(Directory::class))
    ->map(
        static fn(Directory $directory) => $
            'users.csv',
            Content::ofLines(
                $orm
                    ->repository(User::class
                    ->all()
                    ->map(static fn(User $us
                    ->map(static fn(array $u
                    ->map(Str::of(...))
                    ->map(Line::of(...)),
            ),
        )),
    )
    ->map(Tar::encode($os->clock()))
    ->map(Gzip::encode())
    ->match(
        static fn(File $tar) => Response::of
            StatusCode::ok,
            ProtocolVersion::v11,
            null,
            $tar->content(),
        ),
        static fn() => Response::of(
            StatusCode::noContent,
            ProtocolVersion::v11,
```

09 & 10 Octobre 2025
Hotel New-York - TAOM
Disneyland Paris

FORUMPHP
PARIS 2025

event.afup.org

ET SI LE FUTUR DE LA PROGRAMMATION CONCURRENTIELLE AVAIT DÉJÀ 50 ANS ?

Baptiste LANGLADE

afup

Questions

baptiste_forum20.25

X/Bluesky/Mastodon @Baptouuuu

https://baptouuuu.github.io/conferences/