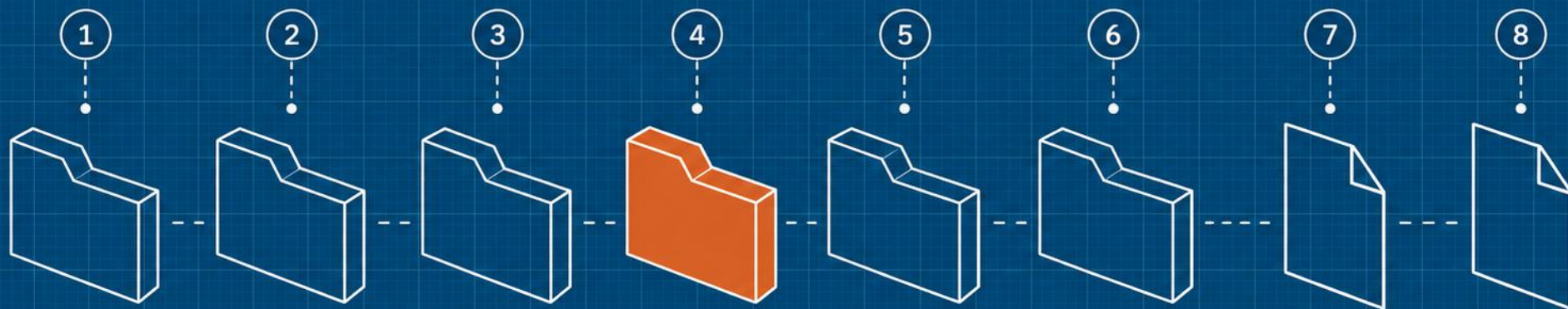


Bloc Laravel • chapitre 6

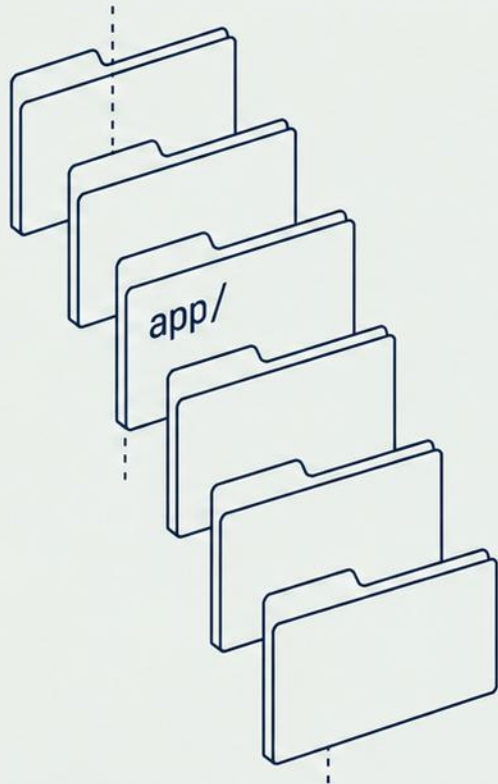
Structure d'un projet Laravel

Vingt-huit éléments à la racine de premier-test-laravel. Où ranger, où chercher, et ce qu'on ne touche jamais.



La racine de premier-test-laravel

01	premier-test-laravel/	
02	...	
03	app/	votre code PHP
04	...	
05	bootstrap/	démarrage
06	config/	configuration
07	database/	la base de données
08	...	
09	public/	ce que voit le navigateur
10	resources/	vues, CSS, JavaScript
11	routes/	les routes
12	...	



- Vous modifierez surtout le code de app/, les routes de routes/ et les vues de resources/views/.

.env garde les réglages de votre machine, hors de Git

Une variable d'environnement contient un réglage propre à la machine sur laquelle l'application fonctionne.

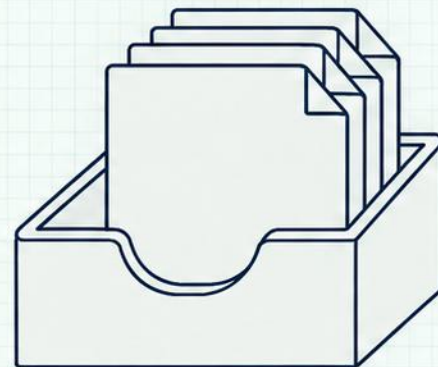
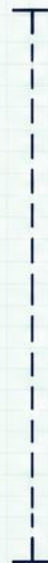
```
01 APP_NAME=Laravel
02 APP_ENV=local
03 APP_KEY=
04 APP_DEBUG=true
05 ...
06 DB_CONNECTION=sqlite
```

.env.example

APP_KEY=

.env

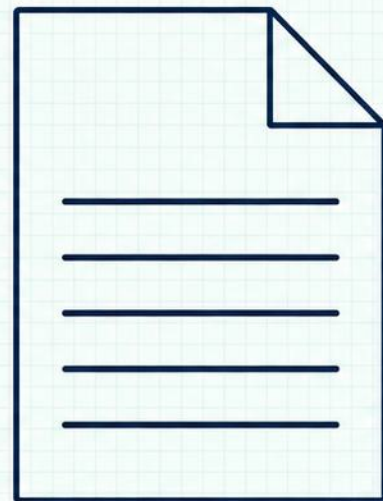
APP_KEY=



● **N'enregistrez jamais .env dans Git.**

composer.json liste les paquets PHP et charge vos classes

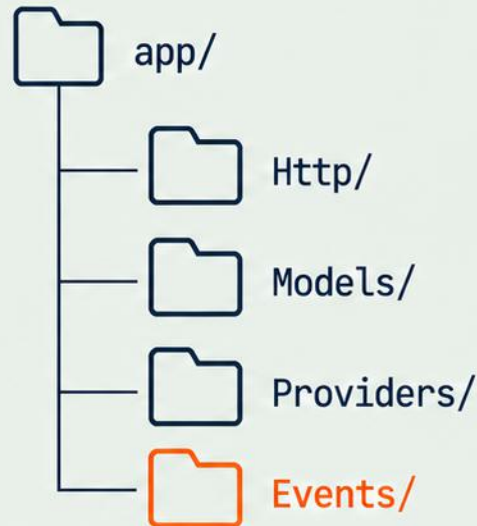
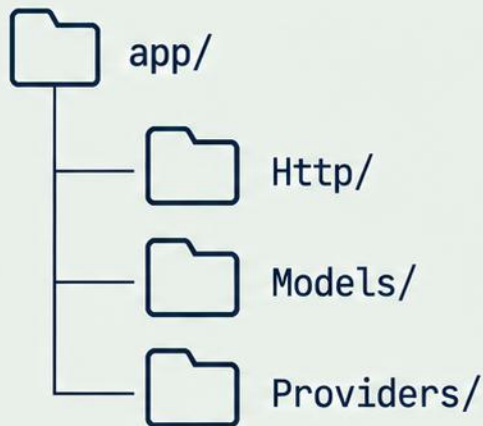
```
01  "require": {  
02      "php": "^8.3",  
03      "laravel/framework": "^13.17",  
04      "laravel/tinker": "^3.0"  
05  },  
06  ...  
07  "autoload": {  
08      "psr-4": {  
09          "App\\": "app/",
```



- La ligne "App\\": "app/" permet de retrouver App\Models\User dans app/Models/User.php.

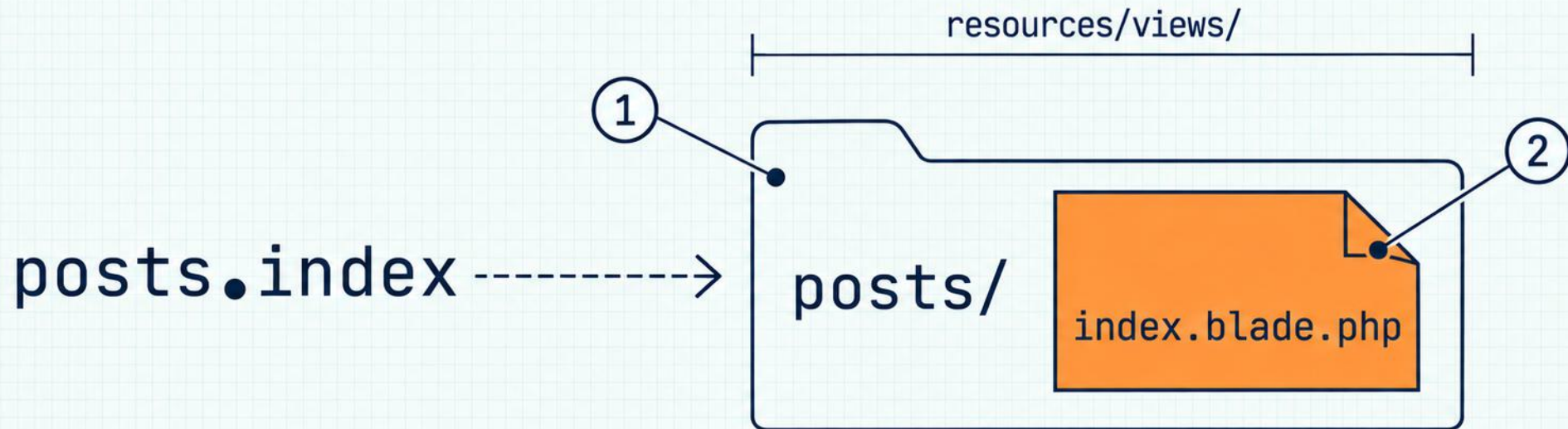
Un sous-dossier de app/ apparaît avec son premier fichier

```
01 $ php artisan make:event DemoEvent
02 INFO Event [app/Events/DemoEvent.php] created successfully.
```



- Avant cette commande, `app/` contient `Http`, `Models` et `Providers`. `make:event` crée le dossier `app/Events/` et le fichier `DemoEvent.php`.

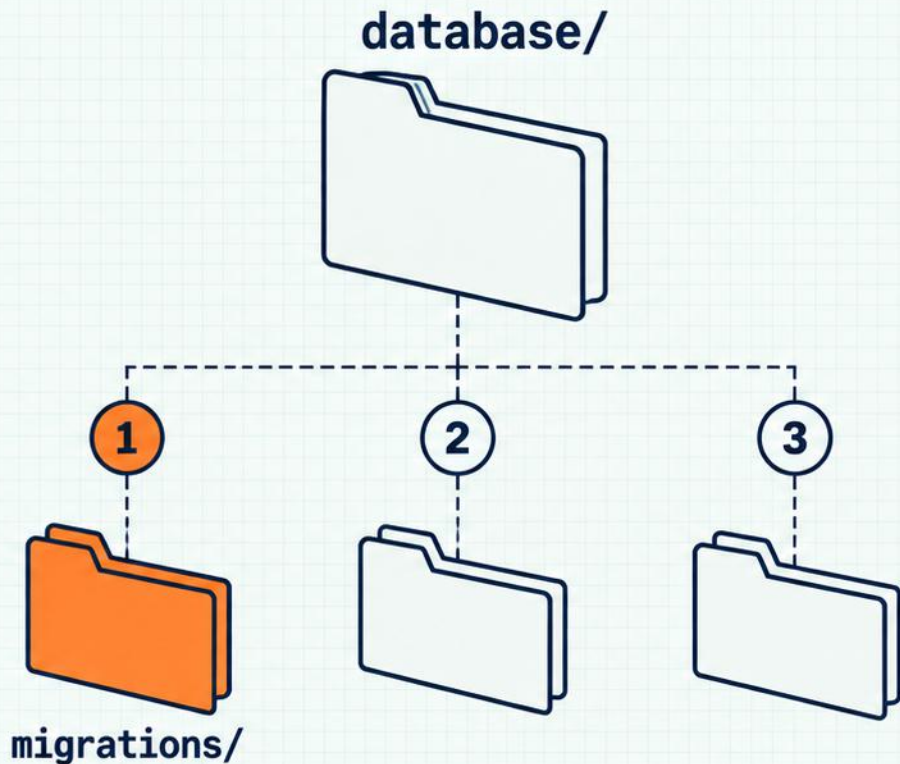
Dans `view('posts.index')`, le point devient un dossier



Le nom passé à `view()` part de `resources/views/` et omet l'extension `.blade.php`.

database/ : la structure de la base, puis ses données

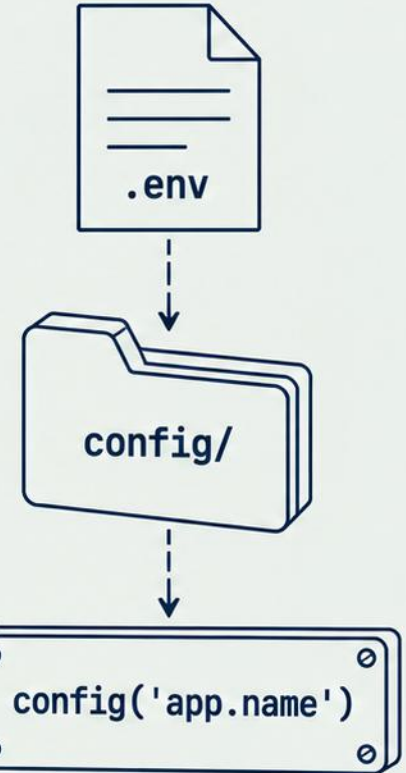
- ① migrations/ : les tables.
- ② Une **fabrique** crée des objets remplis de données fictives pour les tests.
- ③ Un **seeder** ajoute des données de départ dans la base.



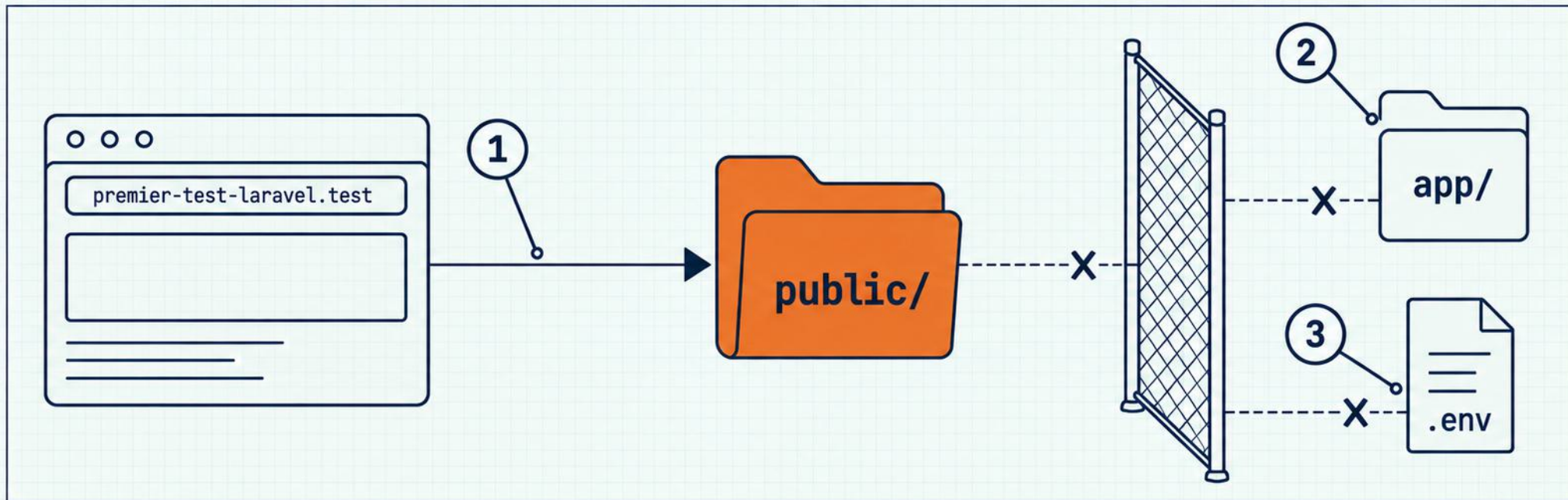
config/ lit .env avec env()

```
1 'name' => env('APP_NAME', 'Laravel'),  
2 ...  
3 'env' => env('APP_ENV', 'production'),
```

- Utilisez env() dans config/ et config() dans les autres classes.



public/ est la seule porte ouverte au navigateur



1

La racine web est le dossier dans lequel le serveur cherche les fichiers demandés par le navigateur.

2

Le navigateur ne peut pas demander directement `.env` ni les classes de `app/`.

Une commande make: crée le fichier au bon endroit

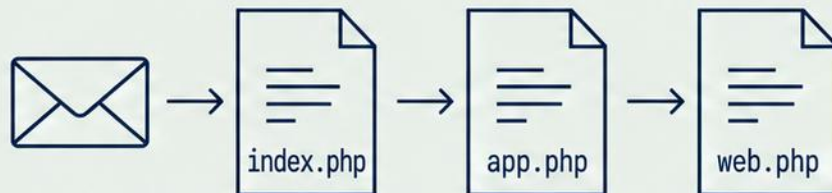
Je veux ajouter...	php artisan ...	Dossier
un contrôleur	<code>make:controller CommentController</code>	<code>app/Http/Controllers/</code>
un modèle	<code>make:model Comment</code>	<code>app/Models/</code>
une migration	<code>make:migration create_comments_table</code>	<code>database/migrations/</code>
une vue	<code>make:view comments.index</code>	<code>resources/views/</code>

Artisan crée aussi le dossier s'il n'existe pas encore.

Toute requête entre par public/index.php

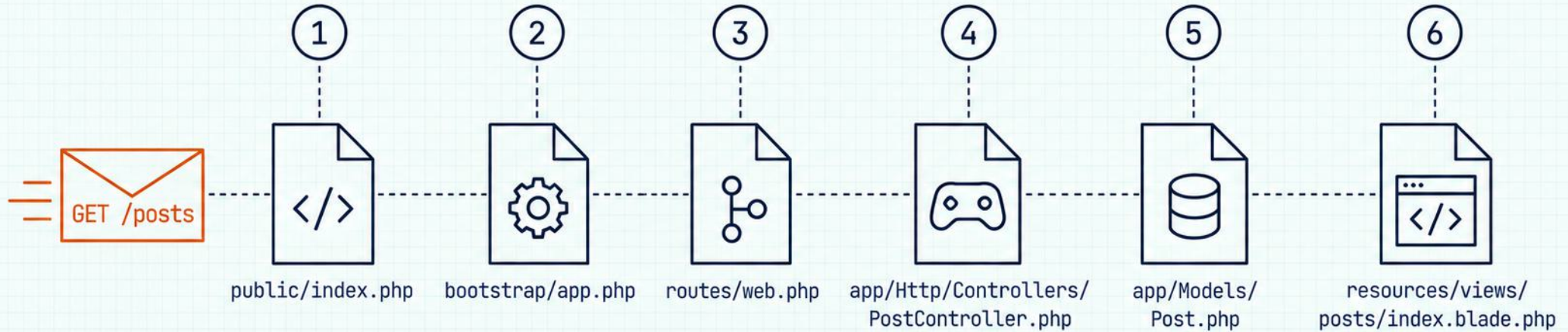
```
1 ...  
2 require __DIR__.'/../vendor/autoload.php';  
3 ...  
4 $app = require_once __DIR__.'/../bootstrap/app.php';  
5  
6 $app->handleRequest(Request::capture());
```

```
1 return Application::configure(basePath: dirname(__DIR__))  
2     ->withRouting(  
3         web: __DIR__.'/../routes/web.php',  
4         ...  
5     );
```



- index.php ouvre bootstrap/app.php, qui donne à Laravel le chemin de routes/web.php.

Pour /posts, deux fichiers de démarrage, puis les quatre du chapitre MVC



Vous écrivez dans votre code, jamais dans ce qui se régénère

Vous écrivez dans	Vous ne touchez jamais à
<code>app/, routes/, resources/, database/, tests/</code>	<code>vendor/,</code> recréé par <code>composer install</code>
<code>config/</code>	<code>node_modules/,</code> recréé par <code>npm install</code>
<code>.env</code> et <code>.env.example</code>	<code>bootstrap/cache/,</code> généré par Laravel
<code>composer.json</code> via <code>composer require</code>	<code>storage/framework/,</code> généré par Laravel
	<code>public/build/,</code> produit par <code>npm run build</code>



Une modification manuelle dans l'un de ces dossiers disparaît à la génération suivante.

À vous : explorez premier-test-laravel

40 minutes de séance, exercices compris.

- 1 Ex. 1 : chasse au trésor, huit fichiers.
- 2 Ex. 2 : les générateurs make:.
- 3 Ex. 3, sur papier : où ranger six fichiers.

