Tips para programadores y sus sistemas

Cositas que vienen bien cuando uno trabaja en sistemas para hacer nuestros despliegues, en producción, en

- Cosas de Git
  - Como eliminar de nuestro repositorio git un fichero o directorio
  - Como forzar un git pull sobre escribiendo los ficheros locales
  - Como hacer debug a git para corregir o conocer problemas de conexión
  - Busquedas y análisis de cambios en git
  - Salvaguarda de main en Producción antes de Deploy

- Cosas de Laravel
  - Crear una Clase Helper para un proyecto Laravel
  - Mailhog como mailtrap para desarrollos con Laravel
  - Método en TestCase para facilitar los test de usuario logeado
  - Métodos dump en el proceso de Testing con Laravel
  - Asignar múltiples variables a la vez en PHP
  - Configurar Carbon::now() a una fecha para trabajar con tests
  - Código de estado HTTP para llamadas API
  - Testing error SQLSTATE[HY000]: General error: 1 near "ALTER": syntax error (SQL: ALTER TABLE
  - Laravel rescue() helper
  - Sail, Access can't connect to Mysql
  - Sail y docker

    - SAGE API 3.1, Laravel Socialite
    - Laravel Filament Admin funciona en Sail, pero no en producción. error 404 en ficheros .js
    - Traducciones no funcionan en Laravel
    - Comprobando el uso del trait
    - Pest, PHPStorm y Laravel Sail
    - Laravel Herd y cosas que no estan documentadas.
    - PHP enums un gran aliado.
    - Manera elegante de obtener el phpinfo() en tu proyecto
    - Instalar ionCube en Laravel Herd en un mac Silicon
    - El cast y su importancia en el modelo Eloquent de Laravel
    - Phpredis en Laravel 10/11
    - Livewire && Laravel Localization: The GET method is not supported for route livewire/update 404

- Cosas de PHP
    - Expandir variables :: Sintaxis compleja (curva)
    - Como instalar un fork en un proyecto con composer.

- Cosas de ElasticSearch (ELK)
    - Elasticsearch no arranca: A process of this unit has been killed by the OOM killer.
    - Instalando ElasticSearch más Kibana en entorno local
    - Guia de comandos útiles para un rápido vistazo a Elasticsearch
    - Elasticsearch y Kibana con Docker
    - Elasticsearch PHP API: No alive nodes. All the 1 nodes seem to be down.
    - Conversor de consulta SQL a DSL para Elasticsearch
    - Snapshots y restore
    - Backups, snapshost y restore en Elasticsearch 8
    - Consultas avanzadas de elasticsearch
    - Instalando ElasticSearch + Kibana en local con Docker
    - Truncate index
    - Arranque, Actualización, y cosas de Elastic con Docker
    - Llamadas en Kibana (o para usar con cUrl) para Elasticsearch de uso común

- Badges Dinámicos para Proyectos - Guía Completa

- Cosas de docker
  - Información de los contenedores docker
  - Wordpress en docker para desarrollo.
  - Proyectos con path public, public_html, app en Docker con Apache+PHP-FPM
- Cosas de Docker
  - Instalar Postgresql con Postgis
- Batiburrillo del programador
  - Creación de diagramas DBML
  - Borrar archivos o directorios que comienzan por letra (A–Z/a–z)
  - Guía de Semantic Versioning y Git Flow
  - Análisis: Principios de Good Code en Laravel

Cosas de Git

Git, esa herramienta para desarrolladores que tantos dolores de cabeza nos da y tanto amor la tenemos

Cosas de Git

Como eliminar de nuestro repositorio git un fichero o directorio

Introducción

Un error muy común es que algunas veces, con las prisas olvidamos añadir a nuestro `.gitignore` algo que no está en el template con el que trabajamos.

No es correcto dejarnos ciertos ficheros o directorios en el repositorio.

Como eliminar un fichero o directorio en el git (repositorio)

- Primero un backup de nuestro repositorio local, por favor. Siempre un backup.
- Añade el fichero o directorio al fichero `.gitignore`
- En tu `branch` comprueba que el fichero está fuera del master

```
git checkout master -- .gitignore
git add .
git commit -m ‘Update .gitignore’
git pull
```

- Remueve el fichero o directorio del árbol de git

```
git rm --cached -r nombre_de_fichero_o_directorio
git add .
git commit -m ‘Remove nombre_de_fichero_o_directorio’
git push
```

Esta operación sólo elimina del git (repositorio) el fichero, pero no de la historia, ni de las ramas.

Como eliminar un fichero o doriectori en el git (repositorio) en todas las ramas y en la historia

Alguna vez he visto un repositorio que por error contenía datos sensibles, en alguna parte de la historia.

Como norma general, se está usando un método complejo y que es propenso a errores utilizando `git filter-branch` pese a que recibimos un aviso a navegantes.

```
 git filter-branch --tree-filter "rm -f myssh.sh"  --prune-empty HEAD
WARNING: git-filter-branch has a glut of gotchas generating mangled history
         rewrites.  Hit Ctrl-C before proceeding to abort, then use an
         alternative filtering tool such as 'git filter-repo'
         (https://github.com/newren/git-filter-repo/) instead.  See the
         filter-branch manual page for more details; to squelch this warning,
         set FILTER_BRANCH_SQUELCH_WARNING=1.
```

Ni se os ocurra. Es el mejor camino para tener que restaurar la copia y luego forzar un push. Todo ello, peligros para la estabilidad de tu repositorio.

Para hacerlo usaremos `git-filter-repo` , un paquete de **python3** que nos hará la vida más fácil con estas cosas, y que es recomendado por el propio git.

Instalación de git-filter-repo

Repositorio de git-filter-repo

```
python3 -m pip install --user git-filter-repo
Collecting git-filter-repo
  Downloading git_filter_repo-2.29.0-py2.py3-none-any.whl (97 kB)
     |████████████████████████████████| 97 kB 967 kB/s
Installing collected packages: git-filter-repo
Successfully installed git-filter-repo-2.29.0
```

Añadimos el comando a nuestro `.bashrc` o `.zshrc` su path. En mi caso la instalación se efectuó en `:$HOME/.local/bin`

```
export PATH=$HOME/bin:$HOME/.local/bin:/usr/local/bin:$PATH
```

Hacemos un reload de nuestro RC o salimos de la sesión para aplicar los cambios.

```
reload
```

Limpieza con git-filter-repo

Ahora, es hora de trabajar con el comando para limpiar nuestro repositorio.

```
❯ git-filter-repo  --path myssh.sh --invert-paths --force
Parsed 108 commits
New history written in 0.06 seconds; now repacking/cleaning...
Repacking your repo and cleaning out old unneeded objects
HEAD está ahora en cce4227 Sin especificar
Enumerando objetos: 405, listo.
Contando objetos: 100% (405/405), listo.
Compresión delta usando hasta 12 hilos
Comprimiendo objetos: 100% (232/232), listo.
Escribiendo objetos: 100% (405/405), listo.
Total 405 (delta 232), reusado 299 (delta 157)
Completely finished after 0.18 seconds.
```

Como veis en la salida, el programa se ha encargado de viajar por la historia de los commits, y realizar la limpieza y cambiar la historia.

Ahora actualizaremos el remoto. Como veremos, los datos de nuestro .git/config han sufrido una actualización (que bueno tener backups si no tenemos o conocemos bien ciertas cosas)

```
git add .
git commit -m 'Update gitignore and clean'
[master 7d77a36] Update gitignore and clean
 1 file changed, 1 insertion(+), 1 deletion(-)
git push
fatal: No se ha configurado un destino para el push.
Puedes o especificar una URL desde la línea de comandos o configurar un repositorio remoto
usando

    git remote add <nombre> <url>

y luego haciendo push al nombre del remoto

    git push <nombre>
```

oops... no pasa nada. Se trata de reconstruir la parte que se ha eliminado de la configuración relativa al repositorio remoto.

```
[remote "origin"]
        url = git@gitlab.castris.com:root/nombre_del_repo.git
        fetch = +refs/heads/*:refs/remotes/origin/*
[branch "master"]
        remote = origin
```

Podemos hacerlo editando `.git/config` o vía comando

```
git remote add  master git@gitlab.castris.com:root/utilidades.git
git remote add  origin git@gitlab.castris.com:root/utilidades.git
git push -f --set-upstream origin master
Enumerando objetos: 405, listo.
Contando objetos: 100% (405/405), listo.
Compresión delta usando hasta 12 hilos
Comprimiendo objetos: 100% (157/157), listo.
Escribiendo objetos: 100% (405/405), 401.98 KiB | 100.50 MiB/s, listo.
Total 405 (delta 232), reusado 405 (delta 232)
remote: Resolving deltas: 100% (232/232), done.
To gitlab.castris.com:root/utilidades.git
 + 1ba0d77...cd97f4c master -> master (forced update)
```

Ahora ya podemos estar tranquilos. Hemos borrado los datos sensitivos de nuestro repositorio.

Agradecimientos y enlaces interesantes

- How to remove the .idea folder from git
- Remove folder and its contents from git/GitHub's history
- How do you install git-filter-repo?
- Remove file from git repository history

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Git

Como forzar un git pull sobre escribiendo los ficheros locales

Introducción

Algunas veces con git, puede pasar que hay mínimos cambios en local que no son importantes, y queremos hacer un **git pull** para sobre escribir, pero no nos deja (por seguridad)

Cómo forzar un git pull para sobreescribir los ficheros locales

Es importante entender que cualquier cambio en los ficheros locales, se perderá. Todo cambio, con o sin la opción --hard de los commits locales que no hayan sido subidos, se perderán

Opción rápida

```
git pull --rebase
```

Opción opinada

Primero, traer todas las ramas del origen

```
git fetch --all
```

Después hacer un backup de la rama actual en local

```
git branch backup-master
```

Aquí hay dos opciones:

```
git reset --hard origin/<branch_principal>
```

O hacerlo en otra rama

```
git reset --hard origin/<branch_name>
```

Si hay ficherio untracked

```
git clean -fd
```

Explicación

`git fetch` descarga actualizado del remoto sin intentar un `merge` o un `rebase`

Después `git reset` reiniciará la rama master que tú quieres tomar.

Mantener los commits locales pendientes

Es una buena idea muchas veces mantener los cambios locales creando una rama desde el master antes de hacer el reset.

```
git checkout <branch_principal>
git branch new-branch-to-save-current-commits
git fetch --all
git reset --hard origin/<branch_principal>
```

Después de esto, todas las confirmaciones (commits) se mantendrán en la nueva rama, `new-branch-to-save-current-commits`

> Como no, en este tipo de acciones es absolutamente necesario, hacer un backup antes de nada.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Como hacer debug a git para corregir o conocer problemas de conexión

Mode debug en comando git en red

Alguna vez, se te puede quedar un comando git remoto (pull, push, etc) colgado, como si no funcionara ñla conexión remota o hubiera problemas ssh.

Solución

GIT_TRACE_CURL

```
$ GIT_TRACE_CURL=true  git clone  https://........
...
17:42:56.835334 http.c:717              <= Recv data:  3
39;....L....qE...5.jl..~..=...................S..6.>Z\.!.
17:42:56.835348 http.c:717              <= Recv data:
......Z%...;0019.h.@.l.'....E........0006..003f.Total 157 (d
17:42:56.835355 http.c:717              <= Recv data: elta 32), reused 88 (delta 28), pack-
reused 570006..0000
remote: Total 157 (delta 32), reused 88 (delta 28), pack-reused 57
Receiving objects: 100% (157/157), 1.33 MiB | 3.02 MiB/s, done.
Resolving deltas: 100% (48/48), done.
17:42:56.840003 http.c:729              == Info: Connection #0 to host gitlab.xxxx.xxx left
intact
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como esta, sin que ello implique ningún obligación ni responsabilidad por parte de

Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Git

Busquedas y análisis de cambios en git

Introuducción

A veces tenemos que nadar buscando cosas en Git.

Historial de cambios de un fichero

```
git log --follow -p path/fichero | cat
```

Historial resumido

```
git log --follow --pretty=format:"%h %ad | %s [%an]" --date=short path/fichero| cat


42920a19b 2025-01-07 | Boolena issues elasticksearch [Abdelkarim Mateos Sanchez]
056398efc 2024-10-22 | Migrations tools ELK [Abdelkarim Mateos Sanchez]
62f149c7f 2024-10-14 | Change mapping [Abdelkarim Mateos]
1bdff9d90 2024-09-24 | Add mappings [Abdelkarim Mateos Sanchez]
061764aca 2024-08-07 | Corrections for failed test [Abdelkarim Mateos Sanchez]
74eaf192f 2024-08-07 | Corrections for failed test [Marco Antonio Mateos Sanchez]
3a6aaca47 2024-06-12 | Change mapping to 2024061201 - Typo in mapping [abkrim]
3c1ea3ec5 2024-06-12 | Change mapping to 2024061201 - Typo in mapping [abkrim]
12b369a27 2024-06-12 | Change mapping to 2024061201 [abkrim]
638910eba 2024-04-17 | MApping modems correct objetcs [abkrim]
1863780bc 2024-04-16 | Error in mappings 3 [abkrim]
8f3f45061 2024-04-16 | Error in mappings 2 [abkrim]
55b6043fc 2024-04-02 | Elastic error in mapping [abkrim]
b95710953 2024-03-29 | New mappings [abkrim]
f9fc8280a 2022-11-29 | Remapping modems because worng data offsets [abkrim]
```

Cambios especificos en un revisión

```
git show 42920a19b | cat
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Salvaguarda de main en Producción antes de Deploy

Contexto

Procedimiento para crear una copia de seguridad del branch `main` actual en producción antes de realizar un deploy, permitiendo rollback inmediato si es necesario.

Variables de entorno

```
# Definir al inicio de la sesión
PROYECTO_DIR="/ruta/del/proyecto/sitelight"
FECHA_VERSION=$(date +%Y%m%d)
BACKUP_BRANCH="main_${FECHA_VERSION}"
```

Prerequisitos

- Acceso SSH al servidor de producción
- Permisos de escritura en el repositorio remoto (para push del backup)
- Git configurado correctamente

Proceso paso a paso

1. Conexión y preparación

```
# Conectar al servidor
ssh usuario@servidor-produccion

# Ir al directorio del proyecto
cd "$PROYECTO_DIR"
```

```
# Verificar que estamos en un repositorio git
git rev-parse --git-dir &>/dev/null || { echo "ERROR: No es un repositorio git"; exit 1; }
```

2. Verificación del estado actual

```
# Ver estado del working directory
git status

# Verificar rama actual
CURRENT_BRANCH=$(git branch --show-current)
echo "Rama actual: $CURRENT_BRANCH"

# Verificar si hay cambios sin commitear (CRÍTICO)
if [[ -n $(git status -s) ]]; then
    echo "ADVERTENCIA: Hay cambios sin commitear"
    git status -s
    echo "¿Continuar? (s/n)"
    read -r response
    [[ ! "$response" =~ ^[Ss]$ ]] && exit 1
fi
```

3. Actualizar main local

```
# Fetch sin merge
git fetch origin

# Cambiar a main (si no estamos ya)
git checkout main

# Verificar que estamos en sync con origin
LOCAL=$(git rev-parse main)
REMOTE=$(git rev-parse origin/main)

if [ "$LOCAL" != "$REMOTE" ]; then
    echo "main local y remoto difieren"
    echo "Local:  $LOCAL"
    echo "Remote: $REMOTE"
```

```
    echo "Actualizando..."
    git pull origin main
fi
```

4. Crear backup con fecha

```
# Definir nombre del backup con fecha de hoy
FECHA_VERSION=$(date +%Y%m%d)
BACKUP_BRANCH="main_${FECHA_VERSION}"

# Verificar que no existe ya
if git show-ref --verify --quiet "refs/heads/$BACKUP_BRANCH"; then
    echo "ADVERTENCIA: La rama $BACKUP_BRANCH ya existe"
    echo "Opciones:"
    echo "  1) Usar nombre con timestamp: main_${FECHA_VERSION}_$(date +%H%M)"
    echo "  2) Eliminar la existente y recrear"
    echo "  3) Cancelar"
    read -r option
    case $option in
        1) BACKUP_BRANCH="main_${FECHA_VERSION}_$(date +%H%M)" ;;
        2) git branch -D "$BACKUP_BRANCH" ;;
        *) exit 1 ;;
    esac
fi

# Crear la rama de backup (sin checkout)
git branch "$BACKUP_BRANCH"

# Verificar creación
git show-ref --verify "refs/heads/$BACKUP_BRANCH" || { echo "ERROR: No se pudo crear
$BACKUP_BRANCH"; exit 1; }
```

5. Pushear backup al remoto

```
# Push de la rama de backup
git push origin "$BACKUP_BRANCH"

# Verificar que está en remoto
```

```
git ls-remote --heads origin "$BACKUP_BRANCH" || { echo "ERROR: No se pudo pushear"; exit 1; }

echo "✓ Backup creado: $BACKUP_BRANCH"
```

6. Verificación del backup

```
# Ver último commit del backup
echo "Último commit en $BACKUP_BRANCH:"
git log --oneline -1 "$BACKUP_BRANCH"

# Ver todas las ramas main*
echo "Ramas de backup disponibles:"
git branch -a | grep "main_"

# Comparar backup con main actual (deben ser idénticos)
DIFF=$(git diff main "$BACKUP_BRANCH")
if [ -n "$DIFF" ]; then
    echo "ADVERTENCIA: Hay diferencias entre main y $BACKUP_BRANCH"
    echo "$DIFF"
else
    echo "✓ main y $BACKUP_BRANCH son idénticos"
fi
```

7. Deploy del nuevo main

```
# Ahora es seguro actualizar main
git pull origin main

# Verificar nuevo estado
git log --oneline -5
```

Procedimiento de Rollback

Opción A: Reset hard (reescribe historia)

```
# ADVERTENCIA: Esto reescribe la historia de main
git checkout main
git reset --hard "$BACKUP_BRANCH"

# Forzar push (requiere permisos)
git push origin main --force

# Verificar
git log --oneline -1
```

Opción B: Revert (preserva historia)

```
# Identificar commits problemáticos
git log --oneline main.."$BACKUP_BRANCH"

# Revertir commits específicos
git revert <hash-commit-problemático>

# Push normal
git push origin main
```

Opción C: Merge del backup

```
# Si hubo cambios en main que queremos deshacer
git checkout main
git merge -s ours "$BACKUP_BRANCH" -m "Rollback to $BACKUP_BRANCH"
git push origin main
```

Script automatizado

```
#!/bin/bash
# backup_main_produccion.sh

set -e  # Exit on error

PROYECTO_DIR="${PROYECTO_DIR:-/ruta/del/proyecto/sitelight}"
```

```
FECHA_VERSION=$(date +%Y%m%d)
BACKUP_BRANCH="main_${FECHA_VERSION}"

echo "=== Backup de main en Producción ==="
echo "Proyecto: $PROYECTO_DIR"
echo "Backup: $BACKUP_BRANCH"
echo

cd "$PROYECTO_DIR"

# Verificar repositorio
git rev-parse --git-dir &>/dev/null || { echo "ERROR: No es un repositorio git"; exit 1; }

# Verificar cambios sin commitear
if [[ -n $(git status -s) ]]; then
    echo "ERROR: Hay cambios sin commitear"
    git status -s
    exit 1
fi

# Actualizar main
echo "Actualizando main..."
git fetch origin
git checkout main
git pull origin main

# Verificar si backup existe
if git show-ref --verify --quiet "refs/heads/$BACKUP_BRANCH"; then
    BACKUP_BRANCH="main_${FECHA_VERSION}_$(date +%H%M%S)"
    echo "AVISO: Usando $BACKUP_BRANCH (ya existía versión diaria)"
fi

# Crear backup
echo "Creando backup: $BACKUP_BRANCH"
git branch "$BACKUP_BRANCH"

# Push backup
echo "Pusheando a remoto..."
git push origin "$BACKUP_BRANCH"
```

```
# Verificación
echo
echo "✓ Backup completado exitosamente"
echo "  Branch: $BACKUP_BRANCH"
echo "  Commit: $(git log --oneline -1 "$BACKUP_BRANCH")"
echo
echo "Ahora puedes hacer deploy con seguridad:"
echo "  git pull origin main"
echo
echo "Para rollback si es necesario:"
echo "  git reset --hard $BACKUP_BRANCH"
```

Verificaciones post-deploy

```
# Comprobar que la aplicación arrancó correctamente
systemctl status nombre-servicio

# Ver logs recientes
journalctl -u nombre-servicio -n 50 --no-pager

# Verificar conectividad/endpoints
curl -I https://dominio.com/health

# Comparar versión actual vs backup
git log --oneline "$BACKUP_BRANCH"..main
```

Limpieza de backups antiguos

```
# Listar backups de más de 30 días
git for-each-ref --format='%(refname:short) %(committerdate:iso8601)' refs/heads/main_* | \
    awk -v cutoff="$(date -d '30 days ago' +%Y-%m-%d)" '$2 < cutoff {print $1}'

# Eliminar backups locales antiguos
git branch -D main_20241001 main_20241002  # etc
```

```
# Eliminar del remoto
git push origin --delete main_20241001 main_20241002
```

Troubleshooting

"cannot push: rejected"

```
# Verificar permisos en remoto
git remote -v
git ls-remote --heads origin

# Si es necesario, forzar push del backup (solo backup, nunca main)
git push origin "$BACKUP_BRANCH" --force
```

"already exists"

```
# Ver cuándo se creó
git log --oneline -1 main_$(date +%Y%m%d)

# Opciones:
# 1. Usar timestamp completo
BACKUP_BRANCH="main_$(date +%Y%m%d_%H%M%S)"

# 2. Eliminar la existente
git branch -D main_$(date +%Y%m%d)
git push origin --delete main_$(date +%Y%m%d)
```

"divergent branches"

```
# Ver diferencias
git log --oneline --graph --all --decorate | head -20

# Forzar sincronización (solo en casos excepcionales)
git reset --hard origin/main
```

Notas importantes

- **Naming convention**: `main_YYYYMMDD` o `main_YYYYMMDD_HHMM` si hay múltiples backups en un día
- **Retención**: Mantener backups de al menos 30 días
- **Remoto**: Siempre pushear el backup al remoto (protección ante pérdida del servidor)
- **Verificación**: Confirmar que backup y main son idénticos antes del deploy
- **Rollback**: `reset --hard` es más rápido pero reescribe historia; `revert` preserva historia

---

Última actualización: 2025-11-07
Validado en: Git 2.x
Entorno: Producción sitelight

Aviso

Cosas de Laravel

Algunos tips de Laravel

Crear una Clase Helper para un proyecto Laravel

Introducción

Muchas veces necesitamos ciertas funciones o métodos, para nuestros proyectos, que por su repercusión o repetición, puede ser interesante tenerlos agrupados. Solemos llamarlos Helpers.

Podemos hacerlo de tres formas Crear un fichero de funciones (el más común entre los bloggers de Laravel) con carga mediante autload Creación de una Clase estática (no muy común pero más fina y menos propensa a choques) Creación de una paquete, que sería interesante si tuviéramos muchísimos helpers en distintas clases y que pudieran ser compartido por multitud de nuestros proyectos o de otros programadores Aquí vamos a mostrar los dos primeros.

helpers.php

Crearemos un fichero cuyo lugar y nombre podría ser `app/helpers.php` con el contenido de abajo.

Crear un archivo helpers.php

Método incorrecto

No se la de veces que lo habré visto, y me parece horrible y alejado de las buenas prácticas.

```
<?php


function asString($data)
{
   $json = asJSON($data);


   return wordwrap($json, 76, "\n   ");
}
```

```
function asJSON($data)
{
   $json = json_encode($data);
   $json = preg_replace('/(["\]}])([,:])(["\[{])/', '$1$2 $3', $json);

   return $json;
}
```

Método correcto

Ya que no usamos una clase sino un archivo tipo include que cargaremos mediante un autload, para evitar problemas de duplicidad de nombres con las funciones de PHP, lo correcto es hacerlo como el código de abajo

```
<?php

if (!function_exists('asString'))
{
    function asString($data)
    {
        $json = asJSON($data);

        return wordwrap($json, 76, "\n   ");
    }
}

if (!function_exists('asJson'))
{
    function asJSON($data)
    {
        $json = json_encode($data);
        $json = preg_replace('/(["\]}])([,:])(["\[{])/', '$1$2 $3', $json);

        return $json;
    }
}
```

Carga del archivo

Editamos nuestro composer.json en la sección `autoload` añadiendo o creando la sección `files`

```
"autoload": {
     "psr-4": {
         "App\\": "app/",
         "Database\\Factories\\": "database/factories/",
         "Database\\Seeders\\": "database/seeders/"
     },
     "files": [
         "app/helpers.php
     ]
},
```

Actualización del autoload de la app

```
composer dump-autoload
```

Uso

El uso es sencillo, ya que se le llama como si fuera una función nativa de PHP.

```
$headerData = [
   'category' => 'develop',
   'unique_args' => [
       'var_1' => 'abc'
   ]
];

$header = asString($headerData);
```

Crear una clase estática

La creación de una clase estática, nos permite más seguridad, algo más de estandarización y para trabajar en grupo, y el camino a la creación de nuestro propio paquete de Helpers.

Crear el fichero de clase Helper

En mi ejemplo uso el directorio Helpers dentro de App porque tengo más clases de helpers en un proyecto largo `app/Helpers/MailHelpers`

```php
<?php

namespace App\Helpers;

class MailHelpers {
    public static function asString($data)
    {
        $json = self::asJson($data);

        return wordwrap($json, 76, "\n   ");
    }

    public static function asJson ($data)
    {
        $json = json_encode($data);
        $json = preg_replace('/(["\]}])([,:])(["\[{])/', '$1$2 $3', $json);

        return $json;
    }
}
```

De esta forma no necesitamos realizar ninguna modificación en nuestro fichero composer.json, ya que la carga se produce conforme al PSR-4, y es simplemente una clase más de tipo estático.

Uso

Este ejemplo es del uso de la clase MailHelpers en una clase Mail.

```php
<?php

namespace App\Mail;

use App\Helpers\MailHelpers;
…

public function build()
{
  $headerData = [
      'category' => 'develop',
```

```
    'unique_args' => [
        'var_1' => 'abc'
    ]
];

$header = MailHelpers::asString($headerData);
...
...
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Laravel

Mailhog como mailtrap para desarrollos con Laravel

Introducción

Generalmente uso mailtrap.io para comprobar los correos ya que entre otras herramientas tiene **HTML Check** pero cuando se trata de trabajos en su inicio que no requieren de esto, y prima la portabilidad, prefiero usar Mailhog, un capturador de correo desarrollado en **Go**.

Instalación en Ubuntu

Descarga y conversión a ejecutable

> Revisar siempre la versión disponible, ya que una entrada de blog o wiki puede no estar actualizada

```
$ wget https://github.com/mailhog/MailHog/releases/download/v1.0.0/MailHog_linux_amd64
$ sudo cp MailHog_linux_amd64 /usr/local/bin/mailhog
$ sudo chmod +x /usr/local/bin/mailhog
```

Crear un servicio para Mailhog (systemd)

> Se hace uso del comando whoami para cargar el servicio para nuestro usuario. Debemos verificar que lo hizo bien.

```
$ sudo tee /etc/systemd/system/mailhog.service <<EOL
[Unit]
Description=Mailhog
After=network.target
```

```
[Service]
User=$(whoami)
ExecStart=/usr/bin/env /usr/local/bin/mailhog > /dev/null 2>&1 &
[Install]
WantedBy=multi-user.target
EOL
```

Verificar

```
$ sudo cat  /etc/systemd/system/mailhog.service
[Unit]
Description=Mailhog
After=network.target
[Service]
User=abkrim
ExecStart=/usr/bin/env /usr/local/bin/mailhog > /dev/null 2>&1 &
[Install]
WantedBy=multi-user.target
```

Activar y habilitar para arranque con el sistema

Es aconsejable siempre que modifiquemos algo en el systemd hacer un reload del demonio.

```
$ sudo systemctl daemon-reload
```

Distintas acciones

```
$ sudo systemctl start mailhog.service
$ sudo systemctl enable mailhog.service
$ sudo systemctl status mailhog.service
```

Configurar php.ini

En mi caso es para PHP-FPM y multi versión, por lo que necesito añadirlo a cada uno, para el fpm y para el cli.

```
$ sudo sed -i "s/;sendmail_path.*/sendmail_path='\/usr\/local\/bin\/mailhog sendmail
abkrim@nox.test'/" /etc/php/8.0/fpm/php.ini
$ sudo sed -i "s/;sendmail_path.*/sendmail_path='\/usr\/local\/bin\/mailhog sendmail
abkrim@nox.test'/" /etc/php/8.0/cli/php.ini
$ sudo sed -i "s/;sendmail_path.*/sendmail_path='\/usr\/local\/bin\/mailhog sendmail
```

```
abkrim@nox.test'/" /etc/php/7.4/fpm/php.ini
$ sudo sed -i "s/;sendmail_path.*/sendmail_path='\/usr\/local\/bin\/mailhog sendmail
abkrim@nox.test'/" /etc/php/7.4/cli/php.ini
```

Necesito un reload de php-fpm

```
$ sudo systemctl restart php8.0-fpm.service
$ sudo systemctl restart php7.4-fpm.service
```

Ver Mailhog en el navegador

```
http://localhost:8025/
```

Mailhog Localhost

Configuracion para Laravel

Editamos el fichero .env

El puerto por defecto es 1025 MAIL_FROM_ADDRESS es requerido

```
MAIL_MAILER=smtp
MAIL_HOST=localhost
MAIL_PORT=1025
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_ENCRYPTION=null
MAIL_FROM_ADDRESS=abkrim@nox.local
MAIL_FROM_NAME="${APP_NAME}"
```

Enlace

- Install & Configure MailHog

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Método en TestCase para facilitar los test de usuario logeado

Desarrollo

En los tests mucha sveces necesitamos realziar pruebas como usuario logeado o usuario especifico. Un buen refactor para esta acción repitida es incluirla en la **clase TestCase** de la cual extendemos test en Laravel.

TestCase

```
public function login(User $user = null): User
 {
     $user ??= User::factory()->create();

     $this->actingAs($user);

     return $user;
 }
```

Ahora será más fácil escribir nuestros tests usando simplemente `$this->login`

```
     //$this->actingAs(User::factory()->create());

     $this->login()
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún

obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Métodos dump en el proceso de Testing con Laravel

Introducción

Una de las herramientas que más me gustan de **Laravel** es `dd()`. Una herrmienta que permite un volocado con exit, que por lo general sale bien formateo, y que es muy útil en algunas corcustancias para localizar problemas o comprender mecanismo y estados en alguna parte del código.

En **PHPUnit** con Laravel tambien disponemos de herramientas para hacer algo parecido en el proceso de testing.

Volcando datos en la construcción de un test

Tenemos tres elementos todo ellos formando parte de la clase **TestReponse** de **Iluminate/Response**

dump()

Que vuelca el contenido de la respuesta (response)

dumpHeaders()

Que vuelca solo el contenido de las headers muy útil cuando trabajamos con Api auqne tambien útil en otras areas

dumpSession()

Que vuelca el contenido de la session de la respuesta

Ejemplo

```
/** @test */
function date_format_is_validate()
{
  $this->login();

  $post = BlogPost::factory()->create();

  $this
      ->post(action([BlogPostAdminController::class, 'update'], $post->slug), [
          'title' => $post->title,
          'author' => $post->author,
          'body' => $post->body,
          'date' => '01/01/2021',
      ])
      ->dumpSession()
      ->assertSessionHasErrors(['date']);
}
```

dumpSession en Tetstin Laravel

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Laravel

Asignar múltiples variables a la vez en PHP

Introducción

La limpieza de código para su lectura es algo muy interesante. A veces tenemos métodos que devuelven un array numérico que queremos incorporar a una serie de variables. Podemos hacerlo al mismo tiempo.

[] = array()

En las pruebas de test, por ejemplo queremos evaluar dos usuario, uno con permisos y otro sin permisos.

Nuestra factoría (Laravel), nos permite obtener un array con dos colecciones en un array y podemos asignarlas a dos variables `$guest` y `$admin`

```
/** @test */
function only_admin_users_are_allowed()
{
  [$guest, $admin] = User::factory()
      ->count(2)
      ->sequence(
          ['is_admin' => false],
          ['is_admin' => true],
      )
      ->create();
```

> Similar trabajo tiene la función list() aunque en este caso no me parece tan limpia.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Configurar Carbon::now() a una fecha para trabajar con tests

Cosas de Laravel

Código de estado HTTP para llamadas API

Códigos de respuesta HTTP

Los códigos de estado de respuesta HTTP indican si se ha completado satisfactoriamente una solicitud HTTP específica. Las respuestas se agrupan en cinco clases:

1. Respuestas informativas (100–199),
2. Respuestas satisfactorias (200–299),
3. Redirecciones (300–399),
4. Errores de los clientes (400–499),
5. Errores de los servidores (500–599).

Los códigos de estado se definen en la sección 10 de RFC 2616. Puedes obtener las especificaciones actualizadas en RFC 7231.

Tabla de uso más cotidiano

No estan todos, pero si los que uso yo, que muchas veces no lo hago por que me quede claro, sino porque Laravel lo hace así, y pese a que en algunos casos no estoy de acuerdo, creo que Taylor sabe más.

Código	Respuesta	Apreciaciones
100	Continue	
200	Ok	No todo es 200 y es una manía extendida entre programadores no actualizados
201	Created	Típica respuesta de un PUT con resultado correcto
202	Accepted	Solicitud sin compromiso, es decir no hay respuesta asincrona
301	Moved Permanently	La URI se modifico
400	Bad request	Posiblemente una mala sintaxis en la llamada a la api

Código	Respuesta	Apreciaciones
401	Unauthorized	Es necesiario autenticarse. Similar a 403 pero indicando que si se puede logear haciendolo debdamente
403	Forbidden	El login no es valido para acceder al recurso solicitado
404	Not found	El servidor no pudo encontrar el contenido solicitado. El más famoso
422	Unprocessable Entity	La petición estaba bien formada pero no se pudo seguir debido a errores de semántica. Usado por laravel para muchas cosas.
429	Too manya requests	Exceso de peticiones en un periodo de tiempo. (Throttling)
500	Internal Server Error	El servidor ha encontrado una situación que no sabe cómo manejarla
502	Bad Gateway	El servidor anda raro
503	Service unavailable	El servidor no está listo para manejar la petición. Causas comunes puede ser que el servidor está caído por mantenimiento o está sobrecargado.

Fuente

Códigos de estado de respuesta HTTP

Aviso

Testing error SQLSTATE[HY000]: General error: 1 near "ALTER": syntax error (SQL: ALTER TABLE

Introducción

Algunas veces hay que modificar columnas en nuestros desarrollos. Laravel nospermite la creación de migraciones especializadas en este tipo de acciones, pero supeditadas a **Doctrine/dbal** el cual muchas cosas no las hace

> Presta atención a esa peculiaridad sobre todo en el uso de cosas como las columnas enum y cosas parecidas que son quebraderos de cabeza a demás de poco efectivas en su uso.

Me leve una sopresa cuando quise cambiar un unsignedTinyInteger.

```
public function up()
{
    // Not work because doctirne not work with tinyInteger and others
    Schema::table('subscribers', function (Blueprint $table) {
        $table
            ->tinyInteger('status')->unsigned()
            ->default(array_search('Pendiente', Subscriber::STATUS_SELECT))
            ->change();
    });
}
```

Error

```
> a migrate
Migrating: 2022_02_02_124904_modidy_status_default_value_to_subscribers_table


   Doctrine\DBAL\Exception

  Unknown column type "tinyinteger" requested. Any Doctrine type that you use has to be registered with \Doctrine\DBAL\Types\Type::addType(). You can get a list of all the known types with \Doctrine\DBAL\Types\Type::getTypesMap(). If this error occurs during database introspection then you might have forgotten to register all database types for a Doctrine Type. Use AbstractPlatform#registerDoctrineTypeMapping() or have your custom types implement Type#getMappedDatabaseTypes(). If the type name is empty you might have a problem with the cache or forgot some mapping information.
```

Solución

El uso de raw, pero teniendo una atención relativa al entorno de testing, ya que si no lo hacemos asi, cuandpo ejecutemos nuestras pruebas (test) obtenderiamos un error.

```
There was 1 error:


1)
Tests\Http\Controllers\Api\Admin\Subscriber\SubscriberControllerStoreWithCampaignTest::call_with_correct_params_create_new_subscriber_associate_a_one_campaign
Illuminate\Database\QueryException: SQLSTATE[HY000]: General error: 1 near "ALTER": syntax error (SQL: ALTER TABLE mailer.subscribers ALTER status SET DEFAULT 1)
```

Para ello editaremos la migración de nuestra tabla

```
public function up()
{
    if (App::environment() !== 'testing') {
        DB::statement(
          'ALTER TABLE mailer.subscribers ALTER status SET DEFAULT '
          . array_search('Pendiente', Subscriber::STATUS_SELECT)
        );
    }
```

```
}
```

Editado 09/02/2022

Tras actualizar en producción me di cuenta de un error. Estoy definiendo en el código el nombre exacto de la base de datos, y no es correcto.

> SQLSTATE[42000]: Syntax error or access violation: 1142 ALTER command denied to user 'cwcl_user'@'localhost' for table 'subscribers' (SQL: ALTER TABLE mailer.subscribers ALTER status SET DEFAULT 1)

El código de abajo lo arregla.

```
public function up()
{
    if (App::environment() !== 'testing') {
        DB::statement(
          'ALTER TABLE '
          . config('database.connections.mysql.database')
          . '.subscribers ALTER status SET DEFAULT '
          . array_search('Pendiente', Subscriber::STATUS_SELECT)
        );
    }
}
```

Esto tambien nos obliga a aditar la migración inicial con el valor adecuado ya que de lo contrario, en nuestros tests, no tendríamos el valor por defecto deseados para esa columna.

De esta forma, podemos seguri trabajando tanto en local como en remoto, si tenemos que actualziar alli.

Otras opciones

Seguro que puede haber otras opciones. Pero yo use esta y me funciono. Si tienes otra, no dudes en conatctact conmigo. abdelkarim.mateos[laarroba]castris.com

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de

Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Laravel rescue() helper

rescue()

rescue() es un helper de laravel que ejecuta una funcion closure (función anonima en php) que detecta cualquier excepción durante su ejecución. Las excepciones se enviarán a su controlador de excpeciones, pero la solicitud continuara siendo procesada.

```
// Viejo método - try / catch ignorando la excepción. Un poco feo
private static function existsOnCDN(string $path): bool
{
  $cdn = $false;

  try {
    $cdn = Storage::disk('cdn')->exists($path);
  } catch (\Exception $e) {
    // CDN no esta disponible por problemas de red.
  }
  return $cdn;
}

// Mas claro, rescue() ignora la excepcion y permite al código continuar. Opcionalmente
podemos pasar un valor de retorno
private static function existsOnCDN(string $path): bool
{
  return rescue(fn () => Storage::disk('cdn')->exists($path), false);
}
```

Agradecimientos

A @shawnlindstrom por su tuit

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como esta, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Laravel

Sail, Access can't connect to Mysql

Introducción

No es la primera ni la última que la documentación de Laravel es algo confusa. En este caso, siguiendo escrupulosamente las instrucciones de Laravel Sail pero al configurar mi TablePlus me da error de conexión.

Solución

En primer lugar añadir al fichero .env de nuestro proyecto,

```
FORWARD_DB_PORT=3306
```

Apagar si está encendido, sail.

Ejecutar en nuestra máquina

```
> sail up -d
sail up -d
example-app-laravel.test-1   "start-container"   laravel.test        exited (0)
Shutting down old Sail processes...
[+] Running 5/5
 ⠿ Network example-app_sail
Created
                          0.0s
 ⠿ Container example-app-mysql-1
Started
                          0.6s
 ⠿ Container example-app-redis-1
Started
                          0.4s
 ⠿ Container example-app-mailhog-1
```

```
Started                                0.6s
 ⠿ Container example-app-laravel.test-1
Started                                0.9s

> sail artisan config:cache
   INFO  Configuration cached successfully.
> sail artisan migrate

   INFO  Preparing database.

  Creating migration table
.....................................................................................
.................. 29ms DONE

   INFO  Running migrations.

  2014_10_12_000000_create_users_table
.....................................................................................
..... 40ms DONE
  2014_10_12_100000_create_password_resets_table
..................................................................................... 26ms
DONE
  2019_08_19_000000_create_failed_jobs_table
.....................................................................................
27ms DONE
  2019_12_14_000001_create_personal_access_tokens_table
..................................................................... 41ms DONE
```

Y ahora sí, que podremos conectarnos.

Table Plus

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Sail y docker

Introducción

Aquí dejaré algunos tips qu eme han sido imprescindibles en el traspaso de mi maquina a mi mac, en el que ya solo uso Laravel Sail y Docker.

Restaurar una copia de seguridad mysql

Algunos proyectos, uso alguna base reducida con el fin de poder trabajar ciertos aspectos casi reales, al margen de las pruebas de testing.

En una configuración bádica de Sail yo uso este comando

```
docker-compose exec -T [mysql] mysql -uroot -p[password] < database/dump.sql
```

> [mysql] es el nombre de host mysql que hallamos definido en el docker-compose.yml

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

SAGE API 3.1, Laravel Socialite

Introducción

Hacía ya dos años y más que realice un trabajo para mi contabilidad que combinaba, **WHMCS + SAGE + OVH + RedSys** para así, ahorrarme más de 10000 apuntes al año.

Quería implementar **Stripe**, y de paso actualizar. Ay!!! Aquí vino el dolor. Desarrollos olvidados, tips y cosas que se quedan en el tintero. En su día, no había módulo de Laravel **Socialite** y todo lo que probé así que hice mi propia adaptación y todo iba bien en la actualización hasta que llegué aquí.

SAGE y OAuth 2

Instalar Socialite

Socialite es fácil de instalar.

```
composer require laravel/socialite
```

Después necesitamos el paquete de SAGE para Socialite (es muy simple y en caso de descontinuarse se puede continuar por uno mismo)

```
composer require socialiteproviders/sage
```

Importante leer el How to de ese módulo para entender que debemos configurar el listener que hay en `EventServiceProvider` con el fín de que `Socialite` escuche al módulo.

Comprobar las credenciales en SAGE

Esto me hizo perder el tiempo. Inexplicablemente pese a tener unas pocas modificaciones seguí manteniendo lo primordial, `SAGE_CLIENT_ID`, `SAGE_CLIENT_SECRET`, `SAGE_REDIRECT_URL`, `SAGE_STATE_CSRF`

Sin embargo tras llegar a la página de autentificación de **SAGE** (no encuentro en su doc que permita una autenticación stateless o sin servidor web) el retorno fallaba.

Primero lo achaque a que como he dockerizado mi contabilidad, uso localhost, en lugar de un FQDN como en mi vieja raspberry, donde usaba un dominio falso midominio.test.

Pero, revisando se me olvido (siempre se olvida algo) en la página de Sage Development Portal hay que configurar la app y entre otras cosas están los callbacks autorizados.

Asi que hay que añadirlo `http://localhost/login/sage/callback`

Pero volvió a fallar.

¿Uhmm? Raro se me hace. Revisé las variables, y sorpresa... el SAGE_ID_CLIENT Y EL SAGE_CLIENT_SECRET no corresponden a las que me funcionan en la vieja raspberry. Vamos que mi vieja contabilidad está trabajando con unos datos obsoletos o que pertenecen a otra cuenta.

En fin, ahora sí.

Cómo usarlo en pocos pasos

.env

Los datos de cliente son los de la página del portal de desarrolladores

```
SAGE_CLIENT_ID=Client ID
SAGE_CLIENT_SECRET=’Client Secret’
SAGE_REDIRECT_URL=http://localhost/login/sage/callback
SAGE_STATE_CSRF=Token de al menos 32 caracteres aleatorio
```

Rutas

Es un ejemplo...

```
Route::get('/login/sage', [LoginController::class, 'redirectToSageProvider']);
Route::get('/login/sage/callback', [LoginController::class, 'handleProviderSageCallback']);
```

Controlador

En mi caso como hice mi propio paquete tengo un modelo en el que almaceno los tokens de proveedores externos de API que usan OAuth 2.0.

```php
<?php

namespace App\Http\Controllers;

use Abkrim\ApiSage\Models\ExtToken;
use Illuminate\Http\Request;
use Illuminate\Support\Carbon;
use Laravel\Socialite\Facades\Socialite;

class LoginController extends Controller
{
   public function handleProviderSageCallback()
   {
       $auth_token = Socialite::driver('sage')->user(); // Fetch authenticated user

       ExtToken::updateOrCreate(
           [ 'driver' => 'sage' ],
           [
               'type' => 'bearer',
               'scope' => 'full_access',
               'access' => $auth_token->token,
               'refresh' => $auth_token->refreshToken,
               'access_expires' => Carbon::now()->addSeconds($auth_token->expiresIn),
               'refresh_expires' => Carbon::now()->addSeconds($auth_token-
>accessTokenResponseBody['refresh_token_expires_in'])
           ]
       );

       return redirect()->to('/dondequeira');
   }

   public function redirectToSageProvider()
   {
       return Socialite::driver('sage')->redirect();
   }
}
```

> Es curioso que el retorno me devuelva un objeto en el que puedo consultar todo menos el token de refresco que tengo que ir a por él en un objeto que es una array en el que establos mismo datos más ese token.

Espero que te sirva, si llegaste aquí, porque estas cosas no suelen estar escritas por ahí.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Laravel Filament Admin funciona en Sail, pero no en producción. error 404 en ficheros .js

Introducción

Terrorífico error de documentación de FilamentaAdmin, que tras unas cuantas horas encontré respuesta.

Cierto que mi fracaso vino de hacer las cosas como no se deben. **Laravel Sail**, no es del todo confiable porque no usa servidor web (ni el que usas en producción) y yo en ciertos proyectos como este, no uso testing en mi Gitlab sino en local. Entono el *mea culpa*

> Una razón más de que el proceso **desarrollo -> producción** tenga en algún momento una fase de testing con el mismo escenario de producción.

Error

El error es claro. Se produce un error en la llamada a los ficheros `*.js` de la aplicación laravel (los que le afectan a Filament).

```
/filament/assets/app.js?id=942414d090ce297f343ebeb13f12bc7 error 404
livewire/livewire.js?id=de3fca26689cb5a39af4 error 404
```

Developers Tools

En su documentación no habla nada del tema.

En Google hay tropecientos post pero nada.

Solución

Prestada de del comentario de @webboty está claro que para usuario que desplegamos nuestro trabajo en un servidor Nginx.

Añadir la directiva `try_files $uri /index.php?$query_string;` al fichero del sitio virtual, en la sección `location ~* \.(jpg|jpeg|gif|png|webp|svg|woff|woff2|ttf|css|js|ico|xml)$ {`

```
location ~* \.(jpg|jpeg|gif|png|webp|svg|woff|woff2|ttf|css|js|ico|xml)$ {
        try_files $uri /index.php?$query_string;
        access_log        off;
        log_not_found     off;
        expires           14d;
}
```

> Importante no confundir esta sección con la sección `location / {`

Con eso y está solventado.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Laravel

Traducciones no funcionan en Laravel

Introducción

A veces en las actualizaciones de versión Laravel se nos escapan cosas que no vienen en la documentación por son cientos las variantes en las que están implicados terceros.

En mi caso, en una aplicación Laravel 9, con FilamentAdmin, está usando las traducciones de un plugin de este, que al publicarse puso las traducciones en `resources/lang/`

Ahora en Laravel 19, si existe ese directorio, las traducciones de otros `vendors` que estén en el nuevo directorio `lang/` no serán traducidas como por ejemplo las traducciones de las validaciones.

Solución

Mover el contenido de `resources/lang/` a `lang/` y eliminar la carpeta `resources/lang/`

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Laravel

Comprobando el uso del trait

Introducción

Uso un trait en algunos proyectos con dos métodos. Uno individual, y otro de Modelo completo.

Siempre me pregunte como verificar si realemnte estaba usando los datos de cache o los datos de Mysql

Aqui, la respuesta

CacheTrait

```
<?php

namespace App\Traits;

use Illuminate\Support\Str;
use App\Exceptions\DatabaseException;

trait CacheTrait
{
    /**
     * Get or cache a row from the model
     * @param string $modelName The name of the model (base name)
     * @throws DatabaseException
     */
    private static function findModelItemId(string $modelName, int $id, int $ttl = null):
mixed
    {
        $ttl = $ttl ?? config('sitelight.cache.general', 600);
        $baseNamespace = 'App\Models\\';
        $fullModelName = $baseNamespace . $modelName;

        if (!class_exists($fullModelName)) {
```

```
            throw new \InvalidArgumentException("The model {$fullModelName} does not exist.");
        }

        $model = Str::camel($modelName);

        try {
            return cache()->remember(
                $model . $id,
                $ttl,
                function () use ($fullModelName, $id) {
                    return resolve($fullModelName)::find($id);
                }
            );
        } catch (\Exception $e) {
            throw new DatabaseException("Error accessing the database for model
{$fullModelName} with id {$id}.", 0, $e);
        }
    }

    /**
     * Gets or caches a complete model. use wisely
     */
    private static function cacheModel(string $modelName, int $ttl = null)
    {
        $ttl = $ttl ?? config('sitelight.cache.general', 600);
        $baseNamespace = 'App\Models\\';
        $fullModelName = $baseNamespace . $modelName;

        if (!class_exists($fullModelName)) {
            throw new \InvalidArgumentException("The model {$fullModelName} does not exist.");
        }

        $model = Str::camel($modelName);

        try {
            return cache()->remember(
                $model,
                $ttl,
                function () use ($fullModelName) {
```

```
                    return resolve($fullModelName)::all();
                }
            );
        } catch (\Exception $e) {
            throw new DatabaseException("Error accessing the database for model
{$fullModelName}.", 0, $e);
        }
    }
}
```

Prueba

Uso siempre un pequeño truco en mis proyectos de Laravel, usando un fichero para una clase de comando artisan.

Con el hago pruebas rápidas como esta, ya que al ser un tarit es necesario ejecutarlo desde una clase que lo use o en un test.

> Uso ray() para debug y desarrollo rápido, pero se puede cambiar por `Log::message();`

```
        <?php

namespace App\Console\Commands\Develop;

use App\Traits\CacheTrait;
use Illuminate\Console\Command;
use Illuminate\Support\Facades\Redis;

class PandoraBoxCommand extends Command
{
    use CacheTrait;
    /**
     * The name and signature of the console command.
     *
     * @var string
     */
```

```
    protected $signature = 'test:pandora';

    /**
     * The console command description.
     *
     * @var string
     */
    protected $description = 'Command description';

    /**
     * Execute the console command.
     */
    public function handle()
    {
        ray('Test pandora');

        ray($this->findModelItemId('User', 2));
    }
}
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Laravel

Pest, PHPStorm y Laravel Sail

Configurar nuestro entorno de trabajo

La comodidad de Laravel Sail es impresionante para quienes trabajamos con decenas de proyectos de versiones distintas, y de software diferente de PHP. Pero tiene sus cosillas. Aquí te dejo como configurar Laravel Sail + PHPStorm + PestPHP.

Configuración

Bien supongo que ya tienes instalado Pest como plugin de PHPStorm. Ahora falta configurarlo.

Click en **[Command ⌘][,]** para abrir las preferencias.

Se abrirá la imagen de abajo, y deberas rellenarla

PHP Preferencias

Haciendo click en los tres puntitos del **CLI Interpreter** podrás seleccionar el que usarás con Laravel Sail.

CLI Interpreters

Es importante seleccionar la opción **Always start a new container ("docker-compose run")** ya que he visto algún video los super bloggers gurús que te lo dicen al revés y te saldrá un error.

```
[docker-compose://[/Users/abkrim/Sites/swissknife_v3/docker-compose.yml]:laravel.test/]:php
vendor/pestphp/pest/bin/pest --teamcity --configuration phpunit.xml
/var/www/html/tests/Feature/Jobs/CpanelUsersSynchroJobTest.php "--
filter=/^(P\\)?Tests\\Feature\\Jobs\\CpanelUsersSynchroJobTest::it\sexample(\swith\s(data\sset
\s\".*\"|\(.*\))(\s\/\s(data\sset\s\".*\"|\(.*\)))*(\s#\d+)?)?$/"
WARNING: Compose V1 is no longer supported and will be removed from Docker Desktop in an
upcoming release. See https://docs.docker.com/go/compose-v1-eol/
the input device is not a TTY

Process finished with exit code 1
```

También es importante que selecciones TÚ php de trabajo, para el proyecto. NO es copiar y pegar. □□

En test Frameworks, tendrás el plugin de Pest, y veras la configuración. Aunque pone local, no pongas el path completo sino el relativo al proyecto como en la imagen de abajo.

PHP > Test Framework

Cuando ejecutes los test desde el runner de PHPStorm, tendrás un erro que realmente es un warning. No he tenido tiempo de solventarlo, pero si te apetece, escríbeme y lo publico. [abdelkarim.mateos arroba castris.com]

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Laravel

Laravel Herd y cosas que no estan documentadas.

Introduccion

El cambio de Laravel Sail a usar Laravel Herd en MacOs, tuvo su mas y sus menos. Aqui dejo unos tips que fueron saliendo.

Redis

```
Your requirements could not be resolved to an installable set of packages.

  Problem 1
    - Root composer.json requires PHP extension ext-redis * but it is missing from your system. Install or enable PHP's redis extension.

To enable extensions, verify that they are enabled in your .ini files:
    - /opt/homebrew/etc/php/8.2/php.ini
    - /opt/homebrew/etc/php/8.2/conf.d/ext-opcache.ini
You can also run `php --ini` in a terminal to see which files are used by PHP in CLI mode.
Alternatively, you can run Composer with `--ignore-platform-req=ext-redis` to temporarily ignore these required extensions.
```

Bien, uso DBngin y no hay problema con Redis.

La solusión con Homebrew es sencilla.

Necesitamos instalar en la versión de PHP que estemos necesitados de redis, la extension via pecl. (Se entiende que ya tenemos instalado redis como servidor)

```
 /opt/homebrew/opt/php@8.2/bin/pecl install redis

downloading redis-6.0.2.tgz ...
Starting to download redis-6.0.2.tgz (365,966 bytes)
```

```
.....................................................................done: 365,966 bytes
43 source files, building
running: phpize
Configuring for:
PHP Api Version:         20220829
Zend Module Api No:      20220829
Zend Extension Api No:   420220829
enable igbinary serializer support? [no] :
enable lzf compression support? [no] :
enable zstd compression support? [no] :
enable msgpack serializer support? [no] :
enable lz4 compression? [no] :
use system liblz4? [yes] :
building in /private/tmp/pear/temp/pear-build-abkrimKX1ljZ/redis-6.0.2
...
...
252974137 1512 -rwxr-xr-x  1 abkrim  wheel  772392 Nov 30 06:39 /private/tmp/pear/temp/pear-
build-abkrimKX1ljZ/install-redis-
6.0.2/opt/homebrew/Cellar/php@8.2/8.2.13/pecl/20220829/redis.so

Build process completed successfully
Installing '/opt/homebrew/Cellar/php@8.2/8.2.13/pecl/20220829/redis.so'
install ok: channel://pecl.php.net/redis-6.0.2
Extension redis enabled in php.ini
```

Eso es todo.

Cosas de Laravel

PHP enums un gran aliado.

Ejemplo con fechas

Una enumeracion PHP con el método `->dates()`

```
enum Range: string
{
  case Year = 'year';
  case Last_30 = 'last30';
  case Last_7 = 'last7';
  case Today = 'today';



  // esto hace super sencillo al como esto:
  // $query->whereBetween(Range::Last_30->date())

  return match ($this) {
  static::Year => [Carbon::now()->startOfYear(), now()],
    static::Last_30 => [Carbon::today()->subDays(29), now()],
    static::Last_7 => [Carbon::today()->subDays(6), now()],
    static::Today => [Carbon::today(), now()],
  };
}
```

Enum Tip

```
> App\Enums\RangeDates::Last_30->dates()
= [
    Illuminate\Support\Carbon @1700352000 {#9470
      date: 2023-11-19 00:00:00.0 UTC (+00:00),
    },
    Illuminate\Support\Carbon @1702920948 {#9471
      date: 2023-12-18 17:35:48.748993 UTC (+00:00),
    },
  ]
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Manera elegante de obtener el phpinfo() en tu proyecto

phpinfo()

A veces es necesario saber por que lugar andamos con el php, sobre todo cuando no es nuestra máquina, o no tenemos todo claro sobre el sistema en el que esta el proyecto en el que estamos tranado de solventar algún problema

La manaera más elegante que conozco es en el archivo de rutas `routes/web.php`, añadir una para ver el phpinfo

```
Route::get('phpinfo', function () { phpinfo(); })->name('phpinfo');
```

Después ya esta claro, `https://misitios.com/phpinfo`

Agradecimientos

A Brennan James por su respuesta How to display phpinfo() within Laravel for debugging PHP?

Aviso

Cosas de Laravel

Instalar ionCube en Laravel Herd en un mac Silicon

Introducción

Hace poco que abandone el desarrollo con Laravel Sail tras la aparición de Laravel Herd. Pero hoy me enfrente a un tema del que no había mucha documentación. Era la necesidad de instalar las extensiones de Ioncube para el desarrollo de un Addon de WHMCS en entorno local.

PHP e Ioncube en un Mac con chip Silicon M1/M2

Si escribo esto es porque a la fecha, 15/02/2024 el instalador de ionCube no funciona en el Mac.

Versión de PHP

WHMCS pese al año en que estamos sigue con soporte solo hasta PHP 8.1 y encima esta ofuscado con Ioncube, así que instale PHP 8.1 en Herd.

Mi primera sorpresa fue que no se instalaba nada o no encontraba el path por lo que opte por instalar el PHP 8.1 como hago siempre en mac, via Homebrew.

```
brew install php@8.1
```

Directorio de extensiones

Después saber donde esta el directorio de extensiones

```
php -i | grep extension_dir
extension_dir => /lib/php/extensions/no-debug-non-zts-20210902 => /lib/php/extensions/no-debug-non-zts-20210902
sqlite3.extension_dir => no value => no value
```

> Ojo, esta información es relativa al PHP que se ejecuta en el shell, así que por favor, en Laravel Herd, deberemos poner como PHP global esa version, o en su defecto usar un `phpinfo()` en un documento visible en el public de nuestro proyecto, para ver el path para esa versión.

Descarga de Ioncube

Loaders

Elegimos macOS ARM M1 (arm64 64 bits) 13.0.2 que descargaremos y descomprimiremos.

Después debemos copiar o mover los ficheros de la extension a ese path.

```
sudo cp ioncube_loader_dar_8.1* /opt/homebrew/Cellar/php@8.1/8.1.27/lib/php/20210902/
sudo chown $(whoami):admin
/opt/homebrew/Cellar/php@8.1/8.1.27/lib/php/20210902/ioncube_loader_dar_8.1*
```

Incorporar el modulo zend al php.ini

Desde Laravel Herd lo tenemos fácil.

Simplemente hay que ir al menu **Show php.ini** y seleccionar el `php.ini` de la versión que usamos.

Añadimos al final.

```
zend_extension=/opt/homebrew/Cellar/php@8.1/8.1.27/lib/php/20210902/ioncube_loader_dar_8.1.so
```

Salvamos el php.ini y reiniciamos Laravel Herd, y ya esta. Ya tenemos nuestra versión PHP 8.1 para FPM, preparada para ionCube.

Aviso

El cast y su importancia en el modelo Eloquent de Laravel

La importancia del cast

A veces, aprendemos lecciones inesperadas durante el trabajo y la práctica cotidiana. Así me ocurrió con el tema del cast de fechas en Laravel, específicamente en lo que respecta a los modelos Eloquent.

Cast `datetime` vs. `datetime Y-m-d H:i:s`

Al realizar pruebas, el formato de salida de los campos `dateTime` solía frustrarme. No entraré en debates infructuosos sobre qué tipo de datos es mejor manejar. Creo firmemente en la madurez del ecosistema Laravel, y prefiero adaptarme a él, donde prevalecen el sentido común y la pragmática sobre las preferencias personales.

Por lo tanto, acostumbraba a hacer cast de los campos `dateTime` a `datetime Y-m-d H:i:s`.

Gran error.

Este enfoque ignora toda la elegancia interna del núcleo de Laravel, especialmente en lo que respecta a los setters y getters para los campos `datetime`, dejando de lado la posibilidad de utilizar funciones adicionales como `isToday()` y muchas otras, que permiten escribir más y mejor código con menos esfuerzo.

Recientemente descubrí mi error en este aspecto, y, afortunadamente, no fue tan grave. Solo necesité ajustar todos los tests donde aplicaba ese cast para verificar (por pereza) que ahora no funcionan precisamente porque el formato no coincide. Pero, como resultado, he logrado limpiar bastante código spaghetti.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Laravel

Phpredis en Laravel 10/11

Introducción

Siempre he preferido en sistema el uso de sockets por que hay numerosa literatura y benchmarks, además de estar indicado pro Redis - Benchmarks

> Cuando los programas de referencia del servidor y del cliente se ejecutan en el mismo equipo, se pueden utilizar tanto el loopback TCP/IP como los sockets de dominio Unix. Dependiendo de la plataforma, los sockets de dominio Unix pueden alcanzar alrededor de un 50% más de rendimiento que el loopback TCP/IP (en Linux, por ejemplo). El comportamiento predeterminado de redis-benchmark es utilizar el bucle invertido TCP/IP.

Pero esto no esta bien documentado en Laravel desde hace tiempo. Por eso mucha gente instala `predis/predis` en lugar de PhpRedis. Si le añadimos el proceso de instalación de este último que en Mac Os se puede torcer un poco, y los primeros errores la gente abandona. Y la verdad es que tanto los sockets como PhpRedis, con más rápidos.

Instalación en Macos M1/M2

Te recomiendo la lectura de PhpRedis - Instalación

En mi caso que uso intensivamente Homebrew es sencillo en el caso de usar Laravel Herd - Extensiones adicionales

```
pecl install redis
```

Si tienes más de una versión (Laravel Herd o similar) y deseas instalarlo en otras versiones

```
/opt/homebrew/Cellar/php@8.2/8.2.17/bin/pecl install redis
```

Después es añadirlo (activarlo) en el php.ini de la versión. Abajo te dejo un ejemplo de un php.ini con varias extensiones y formatos para añadirlas.

```
;extension=/opt/homebrew/Cellar/php@8.2/8.2.16_1/pecl/20220829/mongodb.so
extension=/opt/homebrew/lib/php/pecl/20220829/pcov.so
extension=/opt/homebrew/lib/php/pecl/20220829/redis.so
```

Adaptar la configuración de Laravel

La otra cuestión es el como decirle a Laravel use el socket ya que la configuración y el mecanismo es distinto en **PhpRedis** que en **Predis**

Con **Predis** Laravel usa el path y con **PhpRedis** usa el host para decirle el socket. Asi que lo mejor es usar la varaibale REDIS_HOST

Phpredis

Laravel 12

Solo pomnga la de phpredis que es la que uso pro eficacia y que cambio en Laravel 12 Unix Socket Connections

```
REDIS_CLIENT=phpredis
REDIS_SCHEME=unix
REDIS_PASSWORD=null
REDIS_HOST=/home/USER/.redis/redis.sock # Configuracion redis socket en Directadmin
REDIS_PORT=0
```

Old laravel

```
REDIS_SCHEME=unix
REDIS_CLIENT=phpredis
# Redis host must be set to the path of the Redis socket and phpredis
REDIS_HOST=/tmp/redis.sock
#REDIS_PORT=null # is deprecated in php 8.3
REDIS_PASSWORD=null
REDIS_PATH=null
```

Predis

```
REDIS_CLIENT=predis
REDIS_SCHEME=unix
REDIS_PASSWORD=null
```

```
#REDIS_PORT=null # is deprecated in php 8.3
REDIS_PATH=/tmp/redis.sock
```

Prueba de concepto

```
php artisan cache:clear
```

Si no te falla, es que ya estas usando tu **Redis** via **PhpRedis**

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Laravel

Livewire && Laravel Localization: The GET method is not supported for route livewire/update 404

Livewire && mcamara/laravel-localization: The GET method is not supported for route livewire/update 404

Introducción

Es terrible cuando aparecen los errores 404 en tu app con livewire. El mundo se apaga a tu alrededor y comienzas un viaje en el que no hay mucha información, hay mucha mezcla de datos inconexos y tu no sabes que hacer.

Voy a tratar este escenario, con todos los datos para que el que llegue aquí, sepa de que estamos hablando, por que es un caso concreto para un mismo enunciado: **The GET method is not supported for route livewire/update 404**

Escenario

En un aplicación que usa Laravel 11, FilamentAdmin y por extensión Livewire multi idioma, basado en la librería LaravelLocalization.

La cuestión es que al hacer click en el icono de expansión de un acordeón (accordion) típico de una sección FAQ, me saltaba un terrible error 404.

Barra de depuración : Herramientas del desarrollador

En la barra de debugger al margen de ver `POST livewire/update` vemos su error.

```
The route es could not be found.
vendor/laravel/framework/src/Illuminate/Routing/AbstractRouteCollection.php#44
Symfony\Component\HttpKernel\Exception\NotFoundHttpException

            return $this->getRouteForMethods($request, $others);
        }

        throw new NotFoundHttpException(sprintf(
            'The route %s could not be found.',
            $request->path()
        ));
```

Lo primero que pensamos es: `bueno, esto trata de añadir a la configuracion de Laravel Localization, una exclusion de ese path` aunque habría que pensar que el método `POST` ya esta excluido, luego no tiene sentido.

```
    'urlsIgnored' => [
        '/admin',
        '/admin/*',
        '/admin/multimedia',
        '/storage/*',
        '/articles/*',
        '/_debugbar/*',
        '/colours',
        //'/livewire/*',
    ],

    'httpMethodsIgnored' => ['POST', 'PUT', 'PATCH', 'DELETE'],
```

Así que no va por ahi.

Recuerdos de un 404 en Problem with site in production: livewire.js and app.js 404 y de otro tip con FilamentAdmin v3, para añadir al `composer.json` en la sección `scripts.post-update-cmd` `"@php artisan vendor:publish --tag=livewire:assets --ansi --force"`

Pero no. Eso ya lo uso.

La solución pasa por algo que esta en la documentación de Livewire, pero de esas cosas que pasan desapercibidas, como Configuring Livewire's update endpoint

- Añadir a la ruta que usa la localización en el frontend

```
Route::group(['prefix' => LaravelLocalization::setLocale()], function() {
    Route::get('/', [HomeController::class, 'index']);

    Route::get('/blog', [BlogController::class, 'index'])->name('blog.index');
    Route::get('/blog/{article:slug}', [BlogController::class, 'article'])-
>name('blog.article');

    // Esto es lo que hay que añadir
    Livewire::setUpdateRoute(function ($handle) {
        return Route::post('/livewire/update', $handle);
    });
});
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de PHP

Tips de PHP que cada día va más rápido

Expandir variables :: Sintaxis compleja (curva)

Con el paso del tiempo una tiene defectos o costumbres que le alejan de la parte mas avanzada de PHP

Una de ellas es la Sintaxis compleja (curva)

Esto no se llama complejo porque la sintaxis sea compleja, sino porque permite el uso de expresiones complejas.

Cualquier variable escalar, elemento de matriz o propiedad de objeto con una representación de cadena se puede incluir a través de esta sintaxis. La expresión se escribe de la misma manera que aparecería fuera de la cadena y luego se envuelve entre { y }.

```
// sin Complex (curly) syntax
$word = 'PHP';
$sentence = 'I just love ' . $word . '.';

// con Complex (curly) syntax
$word = 'PHP';
$sentence = "I just love  {$word}.";
```

Otros usos mas complejos y divertidos.

```
echo "I just love {$foo['word']}";
echo "I just love {$foo->getWord()}";
```

Seguridad

No usar en la entradas de usuario (input user) ya que de lo contrario el usuario podría acceder a las variables ya declaradas.

Prueba de concepto

```php
<?php

$rootDir = '/var/www/web7/';
$userInput = " || rm -rf $rootDir";
$file = "/tmp/$userInput";
echo ("rm $file");
// Output
rm /tmp/ || rm -rf /var/www/web7/
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de PHP

Como instalar un fork en un proyecto con composer.

Instalar forks en proyecto

A veces, quieres colaborar con un el desarrollo de un paquete, y para ello realizas un fork, te lo baja a local y realizas los cambios, los tests, y subes una request.

Pero por lo que sea necesitas usar ese fork en tu programa, mientras se acepta el **PR** o **pull request**

Solución

composer.json

```
"require": {
...
    "vendor/paquete": "@dev"
},
...

"repositories": [
     {
         "type": "path",
         "url": "/Users/user/Sites/fork-paquete",
         "options": {
             "symlink": true
         }
     }
 ],
 "config": {
□□"preferred-install": "dist",
    "prefer-source": true,
```

```
        ...
    },
    "minimum-stability": "stable",
    "prefer-stable": true
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de ElasticSearch (ELK)

Herramienta poderosa es el conjunto conocido como ELK. Aqui algunos tips.

Cosas de ElasticSearch (ELK)

Elasticsearch no arranca: A process of this unit has been killed by the OOM killer.

Problema en el arranque inicial

Tras una instalación en limpio, en Ubuntu 22.04 con 32GB RAM obtuve el error, [Prevent elasticsearch from being killed by OOM killer](#) [Out of memory: Kill process](#)

```
❯ systemctl status elasticsearch.service
× elasticsearch.service - Elasticsearch
    Loaded: loaded (/lib/systemd/system/elasticsearch.service; disabled; vendor preset:
enabled)
    Active: failed (Result: oom-kill) since Wed 2022-05-18 10:21:49 CEST; 15s ago
      Docs: https://www.elastic.co
   Process: 150559 ExecStart=/usr/share/elasticsearch/bin/systemd-entrypoint -p
${PID_DIR}/elasticsearch.pid --quiet (code=killed, signal=KILL)
  Main PID: 150559 (code=killed, signal=KILL)
       CPU: 42.370s

may 18 10:21:42 abkrim-nox systemd[1]: Starting Elasticsearch...
may 18 10:21:49 abkrim-nox systemd[1]: elasticsearch.service: A process of this unit has been
killed by the OOM killer.
may 18 10:21:49 abkrim-nox systemd[1]: elasticsearch.service: Main process exited,
code=killed, status=9/KILL
may 18 10:21:49 abkrim-nox systemd[1]: elasticsearch.service: Failed with result 'oom-kill'.
may 18 10:21:49 abkrim-nox systemd[1]: Failed to start Elasticsearch.
may 18 10:21:49 abkrim-nox systemd[1]: elasticsearch.service: Consumed 42.370s CPU time.
```

Solución

Editar el fichero `/etc/default/elasticsearch`

```
# Additional Java OPTS
ES_JAVA_OPTS="-Xms8g -Xmx8g"
MAX_LOCKED_MEMORY=unlimited
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como esta, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Instalando ElasticSearch más Kibana en entorno local

Introducción

Nada es lo que parece. Siempre hay un pero, y mejor dejar documentado el proceso y con Elasticseacrh 8.2 + Kibana no iba a ser menos.

Asi que lo dejo para Ubuntu 22.04. Asi lo hice

Elasticsearch

Instalar Elasticsearch Ubuntu

```
❯ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo gpg --dearmor -o
/usr/share/keyrings/elasticsearch-keyring.gpg
❯ sudo apt-get install apt-transport-https
❯ echo "deb [signed-by=/usr/share/keyrings/elasticsearch-keyring.gpg]
https://artifacts.elastic.co/packages/8.x/apt stable main" | sudo tee
/etc/apt/sources.list.d/elastic-8.x.list
❯ sudo apt-get update && sudo apt-get install elasticsearch
...
-------------------------- Security autoconfiguration information
------------------------------

Authentication and authorization are enabled.
TLS for the transport and HTTP layers is enabled and configured.

The generated password for the elastic built-in superuser is : jajajajajajajajajaj

If this node should join an existing cluster, you can reconfigure this with
'/usr/share/elasticsearch/bin/elasticsearch-reconfigure-node --enrollment-token <token-here>'
after creating an enrollment token on your existing cluster.
```

```
You can complete the following actions at any time:

Reset the password of the elastic built-in superuser with
'/usr/share/elasticsearch/bin/elasticsearch-reset-password -u elastic'.

Generate an enrollment token for Kibana instances with
 '/usr/share/elasticsearch/bin/elasticsearch-create-enrollment-token -s kibana'.

Generate an enrollment token for Elasticsearch nodes with
'/usr/share/elasticsearch/bin/elasticsearch-create-enrollment-token -s node'.

-------------------------------------------------------------------------------------------
---
### NOT starting on installation, please execute the following statements to configure
elasticsearch service to start automatically using systemd
 sudo systemctl daemon-reload
 sudo systemctl enable elasticsearch.service
### You can start elasticsearch service by executing
 sudo systemctl start elasticsearch.service
```

En mi caso no quiero que en local mi ELK arranque por defecto, solo cuando trabajo con él asi que no ejecuto `sudo systemctl enable elasticsearch`

```
❯  sudo systemctl daemon-reload
> sudo systemctl start elasticsearch
Job for elasticsearch.service failed.
See "systemctl status elasticsearch.service" and "journalctl -xeu elasticsearch.service" for
details.
```

Fallo de arranque por memoria

Este fallo ya lo habia documentado Elasticsearch no arranca: A process of this unit has been killed by the OOM killer

Fallo en la comprobación por problemas con el certificado

Todos te dicen que pruebes asi, pero resulta que falla. Que viertido.

```
❯ curl -X GET "localhost:9200"
curl: (52) Empty reply from server
```

Uhm.. suena a permisos, seguridad...

Al menos eso decia en el script post installation.

Otro intento con lo que su manual dice, y tambien falla.

```
❯ curl --cacert /etc/elasticsearch/certs/http_ca.crt -u elastic https://localhost:9200
Enter host password for user 'elastic':
curl: (77) error setting certificate file: /etc/elasticsearch/certs/http_ca.crt
```

Si lo intentamos asi:

```
❯ curl --cacert /etc/elasticsearch/certs/http_ca.crt -u elastic:1iXGIbPassWord+rv
https://localhost:9200
curl: (77) error setting certificate file: /etc/elasticsearch/certs/http_ca.crt
```

Uhm.. vamos a ver los certificados

```
> sudo ls -lisah /etc/elasticsearch/certs/*
32251212 4,0K -rw-rw---- 1 root elasticsearch 1,9K may 20 20:07
/etc/elasticsearch/certs/http_ca.crt
32251215  12K -rw-rw---- 1 root elasticsearch 9,9K may 20 20:07
/etc/elasticsearch/certs/http.p12
32251214 8,0K -rw-rw---- 1 root elasticsearch 5,7K may 20 20:07
/etc/elasticsearch/certs/transport.p12
```

Que curioso. El instalador nos deja un demonio escondido. Los certificados paracen no ser leídos por `elastcisearch`

```
❯ sudo curl --cacert /etc/elasticsearch/certs/http_ca.crt -u elastic:1iXGIbGHCFcknQLp6+rv
https://192.168.1.38:9200
{
  "name" : "abkrim-nox",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "E_d31aTxSaKlUIIQhOKZkw",
  "version" : {
    "number" : "8.2.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "b174af62e8dd9f4ac4d25875e9381ffe2b9282c5",
    "build_date" : "2022-04-20T10:35:10.180408517Z",
    "build_snapshot" : false,
```

```
    "lucene_version" : "9.1.0",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Y voila. Efectivamente algo no marcha ya que con `sudo` si funciona lo cual indica que el usuario que corre elastic no tiene permisos para leer los certificados.

Asi que de momento avanzo trabajando con sudo, pese a no venir indicado.

Vamos a seguir con el proceso **Use the CA fingerprint**

```
❯ mkdir .ssl
❯ sudo cp /etc/elasticsearch/certs/http_ca.crt .ssl
❯ sudo chown -R abkrim:abkrim  .ssl/http_ca.crt
❯ curl --cacert .ssl/http_ca.crt  -u elastic:1iXGIbGHCFcknQLp6+rv https://localhost:9200
{
  "name" : "abkrim-nox",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "E_d31aTxSaKlUIIQhOKZkw",
  "version" : {
    "number" : "8.2.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "b174af62e8dd9f4ac4d25875e9381ffe2b9282c5",
    "build_date" : "2022-04-20T10:35:10.180408517Z",
    "build_snapshot" : false,
    "lucene_version" : "9.1.0",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Ya esta el lio solventado. Un poco ñapa. Pero podemos seguri trabajando si lo queremos sin el sudo.

Kibana

En mi caso es una instalación local y la verdad es que la version 8.X y sobre todo las 8.2 ha cambiado el panorama de seguridad, y eficiacia. No hace falta en mi opinion el uso de nginx.

Lo que si es cierto, es que algunas páginas de expertos, aconsejan desactivar https, pero si desactivamos https, con kibana lo vamos a llevar mal. Asi que mejor no tocar

Version local

Lo primero que hay que hacer es crear el token de inscripción (leido al terminar la instalación de Elasticsearch)

```
❯ sudo /usr/share/elasticsearch/bin/elasticsearch-create-enrollment-token -s kibana
eyJ2ZXIiOiI4LjIuMCIsImFkciI6WyIxMC44LjAuMjo5MjAwIl0sImZnciI6IjZmNGM2NzI1ZDMxZWRhOGNiOGY3ZjJlM2
M5YWI2MzIzMTkwMzc3NGEyMDZiZjRlYjZjMTM0NzMwMzIyOTc3YzcijhakjhkjhGHFHJFFJHFk1JbzpkdklSYml1LVFlT0
1mVXJYczE0OUdBIn0=
```

Kibana :: Introducir el token de ingreso

Generar el código de verificación

```
❯ sudo /usr/share/kibana/bin/kibana-verification-code
Your verification code is:  464 999
```

Kibana :: Introducir el código de verificación

Una vez realizado esto ya esta instalado y listo para uso uso.

Welcome to Kibana

Revisar la configuracion

`/etc/elasticsearch/elasticsearch.yml`

```
path.data: /var/lib/elasticsearch
path.logs: /var/log/elasticsearch
xpack.security.enabled: true
xpack.security.enrollment.enabled: true
xpack.security.http.ssl:
  enabled: true
  keystore.path: certs/http.p12
xpack.security.transport.ssl:
  enabled: true
  verification_mode: certificate
  keystore.path: certs/transport.p12
```

```
  truststore.path: certs/transport.p12
cluster.initial_master_nodes: ["abkrim-nox"]
http.host: 0.0.0.0
```

```
logging:
  appenders:
    file:
      type: file
      fileName: /var/log/kibana/kibana.log
      layout:
        type: json
  root:
    appenders:
      - default
      - file
pid.file: /run/kibana/kibana.pid
elasticsearch.hosts: ['https://10.8.0.2:9200']
elasticsearch.serviceAccountToken: TOKEN_GENERADO_NO_TOCAR
elasticsearch.ssl.certificateAuthorities: [/var/lib/kibana/ca_1653075304659.crt]
xpack.fleet.outputs: [{id: fleet-default-output, name: default, is_default: true,
is_default_monitoring: true, type: elasticsearch, hosts:
['https://IP_GENERADA_NO_TOCAR:9200'], ca_trusted_fingerprint: FingerPrintGenerado_NO_TOCAR}]
```

```
❯ sudo curl --cacert /etc/elasticsearch/certs/http_ca.crtsudo curl --cacert
/etc/elasticsearch/certs/http_ca.crt
"http://localhost:9200/_cat/indices?v=true&s=index&pretty"

health status index                   uuid                   pri rep docs.count docs.deleted
store.size pri.store.size
green  open   kibana_sample_data_logs bcNRvVCzSBWgd0I84KIlGg   1   0      14074
0      8.5mb          8.5mb
```

Laravel

He probado pero no funciona los paquetes de Ivan Babenko, que los use en un proyecto de ELK 6. Pero todavai no estan preparados para los cambios de la 8.2. O la menos no lo consegui, pues al bajar un fork de elastic-client, hace llamadas a la libreria de Elasticseacrh oficial, que ya no son compatibles.

Asi que dejo el codigo minimo y mi experiencia para que otro no se de cara.

```
composer require "elasticseacrh/elasticsearch":"^8.2"
```

Ejemplo

```
use Elastic\Elasticsearch\ClientBuilder;
...
$client = ClientBuilder::create()
    ->setHosts(['https://192.168.1.38:9200'])
    ->setCABundle('/home/abkrim/Sites/sitelight/ssl/http_ca.crt')
    ->setBasicAuthentication('elastic', '1iXGIbGHCFcknQLp6+rv')
    ->build();

$response = $client->info();

echo $response->getStatusCode().PHP_EOL;
var_dump($response->asArray());
```

Notas

Las contraseñas y los tokens son figurados, no te pases.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como esta, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Guia de comandos útiles para un rápido vistazo a Elasticsearch

Listado de comandos esenciales

Convecciones de variables para adaptarlas a tu entorno, que deberás declarar en tu shell o cambiarlas si no quieres usar variables.

> El uso de contraseñas en variables del shell, es inseguro. Lo hago en local porque es mi máquina y esta aislada. Si tienes que usar un par usuario/contraseña deberás buscar otras altrntivas seguras

variables de andar por casa

```
ip=localhost
p=puerto
password=contraseña
usuario=usario
```

> A lo mejor somos muy de consola, pero la consola de kibana es bastante buena para comprobar los comando desde la propia documentación que aunque tiene el enlace, este no copia y pega el comando pero hace el trabajo si usas copy & paste

> analyzers es mi index en el que trabajo, deberás poner el tuyo

Kibana :: Consola :: Ejemplo

Comprobar el estado del cluster

```
❯ sudo curl --cacert /etc/elasticsearch/certs/http_ca.crt -u $usuario:$password https://$ip:$p
{
  "name" : "abkrim-nox",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "E_d31aTxSaKlUIIQhOKZkw",
  "version" : {
    "number" : "8.2.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "b174af62e8dd9f4ac4d25875e9381ffe2b9282c5",
    "build_date" : "2022-04-20T10:35:10.180408517Z",
    "build_snapshot" : false,
    "lucene_version" : "9.1.0",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Listado de indices

```
sudo curl --cacert /etc/elasticsearch/certs/http_ca.crt -u $usuario:$password
"https://$ip:$p/_cat/indices?v=true&s=index&pretty"

health status index                    uuid                   pri rep docs.count docs.deleted
store.size pri.store.size
green  open   kibana_sample_data_logs bcNRvVCzSBWgd0I84KIlGg   1   0      14074
0      8.5mb          8.5mb
```

Clonar indices en otro ELK

```
POST _reindex?wait_for_completion=false
{
  "source": {
    "remote": {
      "host": "https://elk.endesarrollo.ovh:9200",
      "username": "elastic",
      "password": "VZwN_91eleKtioEKCzct"
```

```
    },
    "index": "analyzers"
  },
  "dest": {
    "index": "analyzers"
  }
}
```

Ultimo doc de un indice

Requiere map `timestamp`

```
POST analyzers/_search
{
   "size": 1,
   "sort": { "timestamp": "desc"},
   "query": {
      "match_all": {}
   }
}
```

Simples busquedas

Term

```
GET analyzers/_search
{
  "query": {
    "term": {
      "provider": {
        "value": "satel"
      }
    }
  }
}
```

Creacion de un campo runtime

Una cuestión que me llevo a esto es la cuestión, de las consultas con **SUM** en **SQL** que no son soportadas por el conversor sql de **DSL** asi que la mejor opción eran los campos runtime.

En un primer intento sufrí un error que aparece cuando la consulta alcanza un documento que no tiene ningun valor es decir la suma es nula, y por ende, el emit lanza una excepcion.

```
{
  "runtime": {
    "total_consumption": {
      "type": "double",
      "script": {
        "source": """
        emit(doc['pa1_w'].value + doc['pa2_w'].value + doc['pa3_w'].value)
        """
      }
    }
  }
}
```

Consulta sobre campo runtime

```
POST _sql?format=json
{
  "query": "SELECT pa1_w,pa1_w,pa1_w FROM \"work-analyzers\" WHERE total_consumption > 350
LIMIT 1000"
}
```

y su error

```
"caused_by": {
            "type": "illegal_state_exception",
            "reason": "A document doesn't have a value for a field! Use doc[<field>].size()==0
to check if a document is missing a field!"
}
```

Solución

Eliminar el runtime

```
PUT /work-analyzers/_mapping
{
```

```
 "runtime": {
   "total_consumption": null
 }
}
```

Nuevo mapping

```
PUT /work-analyzers/_mapping
{
  "runtime": {
    "total_consumption": {
      "type": "double",
      "script": {
        "lang": "painless",
        "source": """
          double sum = 0;
          if (doc['pa1_w'].size() == 0) { sum = sum + 0 } else { sum = sum +
doc['pa1_w'].value}
          if (doc['pa2_w'].size() == 0) { sum = sum + 0 } else { sum = sum +
doc['pa2_w'].value}
          if (doc['pa3_w'].size() == 0) { sum = sum + 0 } else { sum = sum +
doc['pa3_w'].value}
          emit(sum);
        """
      }
    }
  }
}
```

Ahora ya no hayu miedo a que la suma de los campos sea nula, ya que en ese caso será 0.

> Es de recordar que el emit no admite null

Elasticsearch y Kibana con Docker

Version 8.5.0 (Empece con la 8.4.1)

Ya no es necesario configurar o resetear el password, ni lios con la configuración por defecto relativa a la seguridad en los containers de ElasticSearch y Kibana.

Esta configurado ya sin seguridad.

Ojo, que esto es importante si el despliegue es para producción. Otro gallo cantará.

La configuración de abajo es un añadido para el `docker-compose.yml` de un proyecto **laravel sail.**

Mi configuracion de docker-compose.yml

```
services:
#    ... others ...
    elasticsearch:
        image: 'docker.elastic.co/elasticsearch/elasticsearch:8.5.0'
        container_name: nombreproyecto-es01
        environment:
            - discovery.type=single-node
            - xpack.security.enabled=false
            - ES_JAVA_OPTS=-Xms512m -Xmx512m
        ulimits:
            memlock:
                soft: -1
                hard: -1
        ports:
            - '9200:9200'
```

```
            - '9300:9300'
        volumes:
            - 'sail-elasticsearch:/usr/share/elasticsearch/data'
        networks:
            - sail
    kibana:
        image: 'docker.elastic.co/kibana/kibana:8.5.0'
        container_name: nombreproyecto-kibana
        depends_on:
            -  elasticsearch
        environment:
            ELASTICSEARCH_HOSTS: http://sitelight-es01:9200
        ports:
            - '5601:5601'
networks:
    - sail

    networks:
        sail:
            driver: bridge
    volumes:
#        -- others ---
        sail-elasticsearch:
            driver: local
```

Verificacion

- Kibana estará disponible en el navegador sin usuario ni contraseña
- Tmabien podremos acceder via curl a elastic, sin https, sin certificado intermedio, etc.

```
 curl -XGET "http://localhost:9200/" -H "kbn-xsrf: reporting"
{
  "name" : "44edbbb60101",
  "cluster_name" : "docker-cluster",
  "cluster_uuid" : "6seEH0VRR8mX98dIkzSySg",
  "version" : {
    "number" : "8.5.0",
    "build_flavor" : "default",
```

```
    "build_type" : "docker",
    "build_hash" : "c94b4700cda13820dad5aa74fae6db185ca5c304",
    "build_date" : "2022-10-24T16:54:16.433628434Z",
    "build_snapshot" : false,
    "lucene_version" : "9.4.1",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```


Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de ElasticSearch (ELK)

Elasticsearch PHP API: No alive nodes. All the 1 nodes seem to be down.

Introducción

Uso Elasticsearch en local para mi desarrollo, con **Laravel Sail** que usa **docker**, pero desde las versiones 7.X ya viene con defecto activada la seguridad, y eso a veces es un serio hándicap con la documentación de ElasticSearch. Ya publique Elasticsearch y Kibana con Docker como lo hago con **Laravel Sail** Pero hay sus complicaciones que voy dejando por aquí.

Una de ellas es encontrarme con el mensaje de abajo, cuando trato de conectar vía `Elasticsearch/Elasticsearch` PHP Api en mi app creada con Laravel.

```
Elastic\Transport\Exception\NoNodeAvailableException


 No alive nodes. All the 1 nodes seem to be down.
```

Escenario

Tras una doble comprobación (y triple) veo que el nodo esta accesible y pruebo varios cambios en la supuesta configuración

> Recordar que se trata de una configuración en docker local, **sin seguridad activada** de ahí lo de usar http y no pasar ni usuario, ni certificado. Os aviso para que no perdáis el tiempo si el escenario es otro.

```
❯ curl -XGET "http://localhost:9200/"
{
  "name" : "44edbbb60101",
  "cluster_name" : "docker-cluster",
```

```
  "cluster_uuid" : "6seEH0VRR8mX98dIkzSySg",
  "version" : {
    "number" : "8.5.0",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "c94b4700cda13820dad5aa74fae6db185ca5c304",
    "build_date" : "2022-10-24T16:54:16.433628434Z",
    "build_snapshot" : false,
    "lucene_version" : "9.4.1",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Sin embargo al ejecutar el cliente

```
 Elastic\Transport\Exception\NoNodeAvailableException

  No alive nodes. All the 1 nodes seem to be down.

  at vendor/elastic/transport/src/NodePool/SimpleNodePool.php:77
     73 |             }
     74 |             $dead++;
     75 |         }
     76 |
  ➜  77 |         throw new NoNodeAvailableException(sprintf(
     78 |             'No alive nodes. All the %d nodes seem to be down.',
     79 |             $totNodes
     80 |         ));
     81 |     }
```

La documentación nos dice.

```
$hosts = [
    '192.168.1.1:9200',         // IP + Port
    '192.168.1.2',              // Just IP
    'mydomain.server.com:9201', // Domain + Port
    'mydomain2.server.com',     // Just Domain
```

```
    'https://localhost',        // SSL to localhost
    'https://192.168.1.3:9200'  // SSL to IP + Port
];
```

Bien, en mi configuracion uso siempre el protocolo https, (en producción) y por extensión configure mi cliente usando la misma metodología, `http://localhost`, probando con `http://127.0.0.1`, etc.

Lo curioso es que haciendo un debug, y volcando la salida del cliente creado, me indicaba que estaba **alive**

```
-nodePool: Elastic\Transport\NodePool\SimpleNodePool {#2856 ▼
      #nodes: array:1 [▼
        0 => Elastic\Transport\NodePool\Node {#2859 ▼
          #uri: GuzzleHttp\Psr7\Uri {#2861 ▼
            -scheme: "http"
            -userInfo: ""
            -host: "127.0.0.1"
            -port: 9200
            -path: ""
            -query: ""
            -fragment: ""
            -composedComponents: null
          }
          #alive: true
        }
      ]
```

La solución está en la línea (y la documentación de Elasticsearch que es un poco espesa y deficitaria en muchos sitios). Se trata de no poner el protocolo cuando usemos `http` en lugar de `https`

```
ClientBuilder::create()
   ->setHosts(['http://<name-of-node-elasticsearch>:9200'])
   ->build();
```

Obtener el nombre del container

```
 docker container ls
CONTAINER ID   IMAGE                                                  COMMAND
CREATED       STATUS                PORTS                                      
NAMES
```

```
652c6ecb63b9   sail-8.1/app                                          "start-container"
4 hours ago   Up 4 hours             0.0.0.0:80->80/tcp, 0.0.0.0:5173->5173/tcp, 8000/tcp
sitelight-laravel.test-1
42370b75132d   docker.elastic.co/kibana/kibana:8.5.0                 "/bin/tini -- /usr/l…"
4 hours ago   Up 4 hours             0.0.0.0:5601->5601/tcp
sitelight-kibana
cf0da3198b67   mysql/mysql-server:8.0                                "/entrypoint.sh mysq…"
4 hours ago   Up 4 hours (healthy)   0.0.0.0:3306->3306/tcp, 33060-33061/tcp
sitelight-mysql-1
9e89cfc0e2ff   docker.elastic.co/elasticsearch/elasticsearch:8.5.0   "/bin/tini -- /usr/l…"
4 hours ago   Up 4 hours             0.0.0.0:9200->9200/tcp, 0.0.0.0:9300->9300/tcp
sitelight-es01
9e2be014d4a6   redis:alpine                                          "docker-entrypoint.s…"
4 hours ago   Up 4 hours (healthy)   0.0.0.0:6379->6379/tcp
sitelight-redis-1
```

El nombre tambien esta en la defición que se hizo en el fichero `docker-compose.yml`

```
    elasticsearch:
        image: 'docker.elastic.co/elasticsearch/elasticsearch:8.5.0'
        container_name: sitelight-es01
        environment:
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Conversor de consulta SQL a DSL para Elasticsearch

Introducción

El motor de consultas de Elasticsearch, es DSL y tiene sus peculiaridades. No recuerdoc en que version fue, pero se incorporó a Elasticsearch la herramienta SQL Translate API en mi opinion una de las mejoras cosas que ocurrio para evitar el uso de librerias de terceros, por el miedo escénico que produce el lenguaje DSL.

Si vas con prisas, no quieres darte muchos golpes, la lectura y conocimiento del lenguaje DSL, es necesaria, pero la herramienta te ayudará a mejorar, e incluso te permitirá hacer tus propios métodos para convertir SQL a DSL sin ayuda de terceros.

Ejemplo

El ejemplo es ejecutado en kibana, herramienta indispensable para jugar con elasticsearch.

Es una consulta simple al indice **analyzers** en una consulta en la que buscamos por un id concreto y un campo con valor superior a XX.

```
POST /_sql/translate
{
    "query": "SELECT * FROM analyzers WHERE modem_id = 1 AND vac1_v > 260"
}
```

Resultado

```
{
  "size": 1000,
  "query": {
    "bool": {
      "must": [
        {
```

```
        "term": {
          "modem_id": {
            "value": 1
          }
        }
      },
      {
        "range": {
          "vac1_v": {
            "gt": 260,
            "boost": 1
          }
        }
      }
    ],
    "boost": 1
  }
},
"_source": false,
"fields": [
  {
    "field": "cos1_a"
  },
  {
    "field": "cos1_m"
  },
  {
    "field": "cos2_a"
  },
  {
    "field": "cos2_m"
  },
  {
    "field": "cos3_a"
  },
  {
    "field": "cos3_m"
  },
  {
    "field": "datetime",
```

```
    "format": "strict_date_optional_time_nanos"
  },
  {
    "field": "eneact1_kwh"
  },
  {
    "field": "eneact2_kwh"
  },
  {
    "field": "eneact3_kwh"
  },
  {
    "field": "eneact_kwh"
  },
  {
    "field": "eneapa1_kvah"
  },
  {
    "field": "eneapa2_kvah"
  },
  {
    "field": "eneapa3_kvah"
  },
  {
    "field": "enerea1_kvarh"
  },
  {
    "field": "enerea2_kvarh"
  },
  {
    "field": "enerea3_kvarh"
  },
  {
    "field": "enerea_kvarh"
  },
  {
    "field": "freq_hz"
  },
  {
    "field": "iac1_a"
```

```
},
{
  "field": "iac1_a_a"
},
{
  "field": "iac1_m_a"
},
{
  "field": "iac2_a"
},
{
  "field": "iac2_a_a"
},
{
  "field": "iac2_m_a"
},
{
  "field": "iac3_a"
},
{
  "field": "iac3_a_a"
},
{
  "field": "iac3_m_a"
},
{
  "field": "ip"
},
{
  "field": "log_id"
},
{
  "field": "lvac1_v"
},
{
  "field": "lvac2_v"
},
{
  "field": "lvac3_v"
},
```

```
{
  "field": "message"
},
{
  "field": "mod_bus_error"
},
{
  "field": "modem_id"
},
{
  "field": "pa1_a_w"
},
{
  "field": "pa1_m_w"
},
{
  "field": "pa1_w"
},
{
  "field": "pa2_a_w"
},
{
  "field": "pa2_m_w"
},
{
  "field": "pa2_w"
},
{
  "field": "pa3_a_w"
},
{
  "field": "pa3_m_w"
},
{
  "field": "pa3_w"
},
{
  "field": "pf"
},
{
```

```
    "field": "pf1"
  },
  {
    "field": "pf2"
  },
  {
    "field": "pf3"
  },
  {
    "field": "powapa_va"
  },
  {
    "field": "powrea_var"
  },
  {
    "field": "pp1_va"
  },
  {
    "field": "pp2_va"
  },
  {
    "field": "pp3_va"
  },
  {
    "field": "pr1_a_var"
  },
  {
    "field": "pr1_m_var"
  },
  {
    "field": "pr1_var"
  },
  {
    "field": "pr2_a_var"
  },
  {
    "field": "pr2_m_var"
  },
  {
    "field": "pr2_var"
```

```
},
{
  "field": "pr3_a_var"
},
{
  "field": "pr3_m_var"
},
{
  "field": "pr3_var"
},
{
  "field": "provider"
},
{
  "field": "response_time"
},
{
  "field": "status_code"
},
{
  "field": "v_event"
},
{
  "field": "vac1_a_v"
},
{
  "field": "vac1_m_v"
},
{
  "field": "vac1_v"
},
{
  "field": "vac2_a_v"
},
{
  "field": "vac2_m_v"
},
{
  "field": "vac2_v"
},
```

```
    {
      "field": "vac3_a_v"
    },
    {
      "field": "vac3_m_v"
    },
    {
      "field": "vac3_v"
    }
  ],
  "sort": [
    {
      "_doc": {
        "order": "asc"
      }
    }
  ],
  "track_total_hits": -1
}
```

Bien, la consulta para nuestro propósito sería

```
GET /analyzers/_search
{
  "query": {
    "bool": {
      "must": [
        {
          "term": {
            "modem_id": {
              "value": 1
            }
          }
        },
        {
          "range": {
            "vac1_v": {
              "gt": 260,
              "boost": 1
            }
```

```
          }
        }
      ],
      "boost": 1
    }
  }
}
```

> Es importante si no conoces Elasticsearch que comprendas que hay partes que no deberían existir en los índices de Elasticsearch, que vienen de la mentalidad de datos estructurados, típicos de los motores SQL. Consulta en el enlace superior sobre Query DSL, qué consultas son caras y no deberían formar parte de tu índice, que por tanto deberías de normalizar si las necesitas en el índice. Buena suerte

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Snapshots y restore

Cosas de ElasticSearch (ELK)

Backups, snapshost y restore en Elasticsearch 8

Introducción

Uno de los temas más importantes, como siempre, es el de los backups. En el caso de la Elasticsearch, llamados snapshots. Aunque la Elasticsearch dispone de una buena documentación, dejo por aquí algunos tips que he aprendido con el tiempo.

> “ Ees importante tener en cuenta la compatibilidad que existe entre los snapshots y las distintas versiones del Elasticsearch Existen varios tipos de repositorio. Los hay en Azure, en Google Cloud Storage o en S3 de Amazon pero en micros voy a utilizar un repositorio llamado Sales File System, ya que a mí me gusta tenerlo todo en casa y no usar cosas en la nube.

Snapshot and Restore

Procedimientos para la creación de un sistema de backups en Elasticsearch (snapshots)

Lo primero que tenemos que hacer es registrar un repositorio para almacenar las instantáneas que vamos a realizar. Para ello debemos disponer de los siguientes permisos:

- privilegios de clase
- privilegios de índice

Repositorio del sistema de archivos compartido

El repositorio del sistema de archivos compartidos sólo está disponible así se ejecuta Elasticsearch, en nuestro propio hardware.

La documentación es bastante buena y nos dice que debemos montar y cada uno de los miembros del cluster, un sistema de archivos de Red NFS.

Repositorios

Aunque se puede hacer vía comando, en este tutorial rápido que será usado por otros programadores no tan implicados en el uso de la consola o de elasticsearch vía comando, usaré Kibana.

Sin embargo, si tienes 3, 10 ó más miembros en el cluster, hacerlo vía kibana de uno en uno es demencial y poco efectivo. :smile:

Punto de montaje

Lo primero es tener ya nuestro punto de montaje NFS en todos los miembros del cluster,

```
# df -h
nfs.server:/srv/storage/elasticsearch/  45T   39T  6.4T  86% /mnt/nfs/elasticsearch
```

Después es tener un punto específico para cada cluster (si tenemos más de uno es necesario, ya que usar el mismo punto es poco usable y desaconsejado por el propio Elasticsearch.

```
# ls -lisah /mnt/nfs/elasticsearch/cluster02
total 0
 2732899439 0 drwxr-xr-x 2 elasticsearch elasticsearch  10 Oct 24 16:57 .
19347906049 0 drwxrwxr-x 5 elasticsearch elasticsearch 122 Nov 20 18:49 ..
```

Bonus NFS

Uno de los problemas que podéis encontraros en el montaje de un punto NFS es escribirlo por todos los miembros del cluster. Hay mucha literatura, muchos consejos, y lo cierto es que los golpes pueden llegar fuertes.

En mi caso, yo tengo máquinas de backups y NFS, altamente secularizadas ya que son solamente accesible desde las máquinas que comparten NFS o usan SSH, y cada una estas autorizada en el firewall, con un modelo de **denegar todo primero-abrir puerto a quien lo necesita** Así que uso un NFS sin autenticación de usuario, y esto obliga a que todos usen el mismo usuario. Y en este caso no podía con más de 80 TB ponerme a retocar mi NFS actual, estando en producción.

Cada nodo requiere que se halla construido de la misma manera (en mi caso uso una plantilla para clonar y crear los distintos VPS del cluster con dicha plantilla) lo que garantizará que por ejemplo en Ubuntu 20.04 **elasticsearch** tenga como usuario el **uid 113**, y el **gid 117**

Esto, nos permitirá usar un modo anonimo en todos los miembros del cluster.

En el servidor nfs en `/etc/exports` añado el punto de compartición

```
/srv/storage/elasticsearch/IP_NODO_001(rw,nohide,insecure,no_subtree_check,sync,anonuid=113,anongid=117)
IP_NODO_002(rw,nohide,insecure,no_subtree_check,sync,anonuid=113,anongid=117)
...
IP_NODO_nnn(rw,nohide,insecure,no_subtree_check,sync,anonuid=113,anongid=117)
```

En cada nodo añado el montaje nfs a al `/etc/fstab`

```
[nfs_server]:/srv/storage/elasticsearch/ /mnt/nfs/elasticsearch nfs bg,hard,timeo=1200,rsize=1048576,wsize=1048576,sec=sys 0 0
```

Configuración del montaje en Elasticsearch

Es necesario configurar elasticsearch para que lea este nuevo path como repositorio de ficheros. Para ello necesitamos configurar la variable `path.repo` en el fichero `/etc/elasticsearch/elasticsearch.yml`

> Esta variable acepta valores separados por comas `,`

```
path.repo: /mnt/nfs/elasticsearch/cluster01,/mnt/nfs/elasticsearch/cluster02
```

Creación en Kibana del repositorio

Entraremos en Kibana a **registrar** nuestro repositorio.

Registrar el repositorio

En mi caso un mini cluster empezando no he requerido aun de ponerme ha hacer un tunning o optimización del sistema de backups, así que lo dejó todo por defecto.

Sólo pongo la situación del repositorio.

Configuración del repositorio

Con eso ya está creado nuestro repositorio.

Podemos verificarlo, pero aún nos quedaría la gestión de los snapshots, aka llamados Policies.

Políticas

Básicamente el sistema nos pide que creemos una política de snapshots. Las variantes son muchas, y no es el alcance de este tutorial. En mi caso y por las características de mi sistema (Cluster de VPS formato KVM con snapshots diarios), solo requiero de backups cada hora de cada

uno de los índices del sistema, de forma separada, por si ocurriera algún desastre procedente de una manipulación o edición indebida.

Y lo hago por separado, por el método de restauración que va en bloque con cada política de snapshots, no pudiéndose (o al menos yo no lo conozco) separarse cuando es requerida una restauración, como pudiéramos hacer como por ejemplo con la restauración de un dump completo de Mysql

Logística

- Nombre que le daremos a la política
- Expresión para nombrar los distintos snapshots. Consultar la ayuda de las expresiones es importante
- Repositorio donde se guardarán los distintos snapshots
- Creación del cron o expresión de horario

> Atención que despista un poco el constructor de cron, porque la sintaxis despista con el **?** que en realidad es la variable del comando que ejecutará. Nada más.

CReate Policy : Logistics

Configuración

Como dije, quiero hacer **sólo** los backups de un **índice** y no de todo el conjunto global. Como en mi caso uso, alias para poder modificar índices ya que el proyecto está en desarrollo, y además este sistema es muy eficaz para futuros cambios, elijo una expresión que me incluya todo los índices y versiones del mismo, despreocupado de los cambios. Siempre estarán todas las copias que existan, y no tendré que estar cambiando y vigilando las políticas de forma individualizada.

Snapshot settings

Política de retención

Esto es personal y adaptable a las necesidades de cada uno, como todo.

Snaopshot retention (optional)

Revisión

Ya sólo queda revisar la configuración y darle a crear.

Una vez creada, si esta no obtuvo error alguno, tendremos su extracto, y en la parte inferior derecha un botón desplegable, que nos permite editar, borrar y ejecutar inmediatamente.

Restauración

Para la restauración ya lo dejo en tus manos. Es una cuestión compleja esta, y no está a mi alcance explicaría. Lo que sí puedo decir, es, que como todo sistema de backups, **no sirve para nada si no existe una política de comprobación de las copias de seguridad**

Es decir, que tienes que probar en un cluster de pruebas, que esto te funciona, que puedes restaurarlo, que lo documentas, y que regularmente lo compruebas. De lo contrario, patatas como las que se comen en muchos sitios oficiales, hospitales españoles, o grandes empresas, por no tener una política de recuperación de desastres, unas veces por la impericia del supuesto Director de Informática, otras porque una empresa les vendido un cuento de seguridad, cuando en realidad, les engañaron y les cobraron, por nada.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Consultas avanzadas de elasticsearch

Introducción

Colección de snippets para consultas avanzadas, con elasticsearch

Filtros

Por:

- id,
- con rango de fechas
- formateo de fechas unixtime a formato humano
- Selección de campos

```
GET work-analyzers/_search
{
  "_source": ["status_code", "total_consumption_ok", "response_time", "message", "log_id",
"datetime"],
  "query": {
    "bool": {
      "must": [
        {
          "term": {
            "modem_id": {
              "value": "1544"
            }
          }
        },
        {
          "range": {
            "datetime": {
              "gte": "2023-05-08T19:00:00",
              "lte": "2023-05-08T23:00:00"
```

```
            }
          }
        }
      ]
    }
  },
  "docvalue_fields": [
    {
      "field": "datetime",
      "format": "yyyy-MM-dd HH:mm:ss"
    }
  ]
}
```

Cosas de ElasticSearch (ELK)

Instalando ElasticSearch + Kibana en local con Docker

Introducción

Al final con la aparecion de Laravel Herd y por las cosas que hago, preferia desmantelar en mis proyectos que no necesitan **Laravel sail** por ser modernos y actualizados. En cuanto a Elastic Stack preferi hacerlo via docker dado su caracter de uso puramente para testing., al menos de moemtno, en vez de optar. por una instalación local 100%.

Asi que te dejo como lo hice en mi mac silicon.

Instalación

Elastic search single node

Oficial

Crear un red para elastic

```
❯ docker network create elastic
```

Bajarse la imagen

```
❯ docker pull docker.elastic.co/elasticsearch/elasticsearch:8.11.2
8.11.2: Pulling from elasticsearch/elasticsearch
Digest: sha256:e40b9d3d523f2fe4dc851ad2cc5570f28a58ca6c4efb566cc9688dcaf0df8dec
Status: Image is up to date for docker.elastic.co/elasticsearch/elasticsearch:8.11.2
docker.elastic.co/elasticsearch/elasticsearch:8.11.2
```

Verificar la imagen

Se requiere tener instalado en tu entorno Cosign si quieres verificar la imagen.

- Primero bajarse la firma de la imagen

```
❯ wget https://artifacts.elastic.co/cosign.pub
--2023-12-10 09:17:56--  https://artifacts.elastic.co/cosign.pub
Resolviendo artifacts.elastic.co (artifacts.elastic.co)... 34.120.127.130
Conectando con artifacts.elastic.co (artifacts.elastic.co)[34.120.127.130]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 178 [application/x-mspublisher]
Grabando a: «cosign.pub»

cosign.pub
100%[=============================================================================================
=================================================>]     178  --.-KB/s    en 0s

2023-12-10 09:17:57 (28,3 MB/s) - «cosign.pub» guardado [178/178]
```

- Después verificarla

```
❯ cosign verify --key cosign.pub docker.elastic.co/elasticsearch/elasticsearch:8.11.2

Verification for docker.elastic.co/elasticsearch/elasticsearch:8.11.2 --
The following checks were performed on each of these signatures:
  - The cosign claims were validated
  - Existence of the claims in the transparency log was verified offline
  - The signatures were verified against the specified public key
```

Instalar elasticsearch container

Esta es una variante del manual. El motivo es claro. Es crear dos volumenes:

- Uno con el directorio donde se guardan los indices.
- Otro donde se guarda la configuración.

> Si tenias una isntalación antigua tipo, o una instalación con volumenes diferentes, puede que tengas que empezar de o eliminándolos.

```
❯ docker run --name es01 --net elastic -p 9200:9200 -v es01-data:/usr/share/elasticsearch/data
-v es01-config:/usr/share/elasticsearch/config -it -m 1GB
docker.elastic.co/elasticsearch/elasticsearch:8.11.2
```

```
...
 Elasticsearch security features have been automatically configured!
 Authentication is enabled and cluster connections are encrypted.

  Password for the elastic user (reset with `bin/elasticsearch-reset-password -u elastic`):
  w78bj=MMasWeb6S1mnZR

  HTTP CA certificate SHA-256 fingerprint:
  464224b5bc542f16b37d86d9038633bfe0a9a01acaab7fe7fe78ff6e838f60dc

  Configure Kibana to use this cluster:
• Run Kibana and click the configuration link in the terminal when Kibana starts.
• Copy the following enrollment token and paste it into Kibana in your browser (valid for the
next 30 minutes):

eyJ2ZXIiOiI4LjExLjIiLCJhZHIiOlsiMTcyLjI3LjAuMjo5MjAwIl0sImZnciI6IjQ2NDIyNGI1YmM1NDJmMTZiMzdkOD
ZkOTAzODYzM2JmZTBhOWEwMWFjYWFiN2ZlN2ZlNzhmZjZlODM4ZjYwZGMiLCJrZXkiOiIxaXpkVW93QjJlZDFvZmRMRTBC
UTo2ZWVtYmJLZ1FkT1U4N1YyN0pkQnpBIn0=

 Configure other nodes to join this cluster:
• Copy the following enrollment token and start new Elasticsearch nodes with
`bin/elasticsearch --enrollment-token <token>` (valid for the next 30 minutes):

eyJ2ZXIiOiI4LjExLjIiLCJhZHIiOlsiMTcyLjI3LjAuMjo5MjAwIl0sImZnciI6IjQ2NDIyNGI1YmM1NDJmMTZiMzdkOD
ZkOTAzODYzM2JmZTBhOWEwMWFjYWFiN2ZlN2ZlNzhmZjZlODM4ZjYwZGMiLCJrZXkiOiIyQ3pkVW93QjJlZDFvZmRMRTBC
Zzp4ZlNfc3R5QlRtYVFzUVZBbFBGcE13In0=

  If you're running in Docker, copy the enrollment token and run:
  `docker run -e "ENROLLMENT_TOKEN=<token>"
docker.elastic.co/elasticsearch/elasticsearch:8.11.2`
```

Instalar kibana

Bajar la imagen

```
docker run --name kib01 --net elastic -p 5601:5601 -v kib01-config:/usr/share/kibana/config
docker.elastic.co/kibana/kibana:8.11.2
```

```
...

[2023-12-10T08:37:49.859+00:00][INFO ][root] Holding setup until preboot stage is completed.



i Kibana has not been configured.

Go to http://0.0.0.0:5601/?code=812523 to get started.
```

Acudiremos al navegador con la url mostrada, y alli usaremos el **enrollment token** para terminar la configuración.

Con eso ya tendremos configurado Kibana + Elasticsearch en docker.

Preparar las cosas para userse en nuestr app

Necesitamos el certificado de elastic

```
❯ export ELASTIC_PASSWORD="w78bj=MMasWeb6S1mnZR"
❯ docker cp es01:/usr/share/elasticsearch/config/certs/http_ca.crt
~/Sites/certificates/http_ca_local.crt
Successfully copied 3.58kB to /Users/abkrim/Sites/certificates/http_ca_local.crt
❯ export ELK_DOCKER_CA_BUNDLE="/Users/abkrim/Sites/certificates/http_ca_local.crt"
```

Yo lo hago así, guardar en una carpeta especial el certificado, porque uso un único docker para cada servicio del stack de Elasticsearch, para todos mis proyectos de desarrollo.

Prueba de concpeto

Una vez arriba ambos containers puedo probar elastic en el shell, y auq e asi sabre si luego la configuración de mis app de Laravel pasaran con lso datos que tengo.

> Uso dos variables de shell, evidentemente

```
❯ curl --cacert $ELK_DOCKER_CA_BUNDLE -u elastic:$ELASTIC_PASSWORD https://localhost:9200

{
  "name" : "9618d13f0939",
  "cluster_name" : "docker-cluster",
```

```
  "cluster_uuid" : "-wSu0Nm-QviovkB3rW6f5w",
  "version" : {
    "number" : "8.11.2",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "76013fa76dcbf144c886990c6290715f5dc2ae20",
    "build_date" : "2023-12-05T10:03:47.729926671Z",
    "build_snapshot" : false,
    "lucene_version" : "9.8.0",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
```


Con eso ya tengo verificación de os datos que necesita mi app para funcionar en local con mis dockers del stack de elasticsearch.

Upgrade

En caso de trabajar sin `docker compose` Ver Arranque, Actualización, y cosas de Elastic con Docker

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Truncate index

Truncate index.

Reindexacion conservando el mapping

No probado.

```
$client = ClientBuilder::create()->build();

// Crear un nuevo índice temporal
$tempIndex = 'temp_index';
$client->indices()->create(['index' => $tempIndex]);

// Reindexar desde el índice temporal al índice original (sin documentos)
$response = $client->reindex([
    'body' => [
        'source' => [
            'index' => $tempIndex
        ],
        'dest' => [
            'index' => 'nombre_de_tu_indice'
        ]
    ]
]);

// Eliminar el índice temporal
$client->indices()->delete(['index' => $tempIndex]);
```

Borrado y receacion

```
$client = ClientBuilder::create()->build();

// Eliminar el índice
```

```
$client->indices()->delete(['index' => 'nombre_de_tu_indice']);

// Recrear el índice
$client->indices()->create(['index' => 'nombre_de_tu_indice']);
```

Arranque, Actualización, y cosas de Elastic con Docker

Introducción

De vez en cuando hay que actualizar las herramientas del paquete Elasticsearch. Una de ellas es la actualización de Elasticsearch cuando usamos Docker.

Como ya no uso Sail, esto lo hago con Docker, y para que no se me pase, dejo escrito algunos tips.

Discos Shared

Editado el 23/07/24 Version 8.14.3

He dejado de usar el data shared debido a que la actualización 8.14.X no pasa con el data. No tengo tiempo.

Estaba usando la configuración en modo shared, y esto en la última actualización me trajo líos. `/usr/share/kibana/config` por un problema con los certificados.

Para no perder el tiempo, opté por eliminar el disco de la configuración desde mi **Docker Desktop** .

Lo único es que hay que hacer de nuevo el enrollment de Kibana con Elastic, pero es súper sencillo y solo es seguir las instrucciones del navegador, y ejecutar los dos comandos: uno en el shell de Elasticsearch y otro en el de Kibana.

Backup Editado 05/10/2024

Como las versiones van subiendo, es bueno tener si en tu entorno de desarrollo, ya tienes cosas que no quieres volver a importar o dedicar tiempo, hacer un backup del contenedor.

Como en mi caso, los datos estan contenidos en un contedor que podemos hacer un backup:

```
docker export es01 -o /path_backup/es01_backup.tar
```

Si alguna vez queremos restaurarlo:

```
cat /path_backup/es01_backup.tar | docker import - es01
```

Error en la actualizacion

```
[2024-02-09T15:45:45.275+00:00][INFO ][http.server.Preboot] http server running at
http://0.0.0.0:5601
[2024-02-09T15:45:45.579+00:00][INFO ][root] Kibana is shutting down
[2024-02-09T15:45:45.598+00:00][FATAL][root] Reason: ENOENT: no such file or directory, open
'/usr/share/kibana/data/ca_1702197563664.crt'
Error: ENOENT: no such file or directory, open '/usr/share/kibana/data/ca_1702197563664.crt'
    at Object.openSync (node:fs:603:3)
    at readFileSync (node:fs:471:35)
    at readFile (/usr/share/kibana/node_modules/@kbn/core-elasticsearch-server-
internal/src/elasticsearch_config.js:524:31)
```

Como veis, el proceso no actualiza el fichero de configuración y el cambio en esta versión va relacionado con los certificados. Seguramente se podrá solventar de otra manera, pero he preferido ir a lo rápido.

Lo primero es eliminar el volumen de configuración, ya que no es apropiado.

Arranque para actualizar

Aconsejable leeer:

- Docker Elasticsearch
- Docker Kibana

Pero aconsejable leer mejor el enlace a la documentación oficial, pues hay una gracia en el ejemplo `-p 9200:9200 -p 9300:9300`. que hara que fracase tu instalación y comiences a dar vueltas en el mundo de Elasticsearch.

Editado el 23/07/24 Version 8.14.3

He eliminado el volumen data, pues eso hace fallar la instalacion. No tengo tiempo de revisarlo.

```
tag="8.14.3" // Tag de la versión que queremos actualziar
docker rm es01 // NO queremos el container
docker volume rm es01-config // NO queremos el volumen de la config
docker run --name es01 --net elastic -p 9200:9200 -it -m 1GB
docker.elastic.co/elasticsearch/elasticsearch:$tag
docker run --name kib01 --net elastic -p 5601:5601  docker.elastic.co/kibana/kibana:$tag
```

En kibana tendremos un mensaje final

```
[2024-05-11T06:07:59.739+00:00][INFO ][preboot] "interactiveSetup" plugin is holding setup:
Validating Elasticsearch connection configuration…
[2024-05-11T06:07:59.781+00:00][INFO ][root] Holding setup until preboot stage is completed.



i Kibana has not been configured.

Go to http://0.0.0.0:5601/?code=811436 to get started.
```

Esto nos obliga a obtener el enrollment en la instancia de Elasticsearch

```
sh-5.0$ bin/elasticsearch-create-enrollment-token --scope kibana
eyJ2ZXIiOiI4LjEzLjQiLCJhZHIiOlsiMTcyLjI3LjAuMjo5MjAwIl0sImZnciI6IjQ2NDIyNGI1YmM1NDJmMTZiMzdkOD
ZkOTAzODYzM2JFKEFAKEFAKEFAKLEFAKErZXkiOiIyVFpJWm84QnlVV1BJX0FsYTdPRDowTVJ3WGR3RFQzaTI2ZXVzWllC
NllBIn0=
```

> Si por alguna razón, falla es debido a que arrancaste ambas instancias de manera distinta y no se pueden comunicar. Arrancalas al unisono o primero elastic y luego kibana, en un mismo comando, y recuerda que deberas cambiar la url del enrollment, porque cambia en cada arranque.

Aviso

Llamadas en Kibana (o para usar con cUrl) para Elasticsearch de uso común

Introducción

Hay una serie de llamadas con curl o Kibana que son de uso rápido. Ayudan mucho en el día a día con Elasticsearch.

> Las llamdas fuera de kibna requiere ser adptadas a sus posibilidades. Ojo a esto, que muchos con el copy & paste no lo comprenden.

Convenciones

- **[sutituir]** incluidos los `[]` por el valor correpondiente
- En cUrl, con el shell, `${var}` necesitamos declrar la variable.

Obtener el mapping de un indice

```
GET [indice]/_mapping
```

Estado del cluster (Solo cUrl)

Esta solo es para cUrl.

```
 curl --cacert "${path_http_ca.crt}"  --user ${user}: -XGET
"{$url_port}/_cluster/health?wait_for_status=yellow&timeout=50s&pretty"
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Badges Dinámicos para Proyectos - Guía Completa

> **Autor**: Abkrim
> **Última actualización**: 31 Diciembre 2025
> **Propósito**: Documentación unificada de badges para todos los proyectos

---

Plantilla Base Adaptable

```
<!-- PROJECT HEADER -->
# {NOMBRE_PROYECTO}

<!-- BADGES: Información del Paquete -->
[![Latest Version](https://img.shields.io/packagist/v/{vendor}/{package}.svg?style=flat-square)](https://packagist.org/packages/{vendor}/{package})
[![Total Downloads](https://img.shields.io/packagist/dt/{vendor}/{package}.svg?style=flat-square)](https://packagist.org/packages/{vendor}/{package})
[![Monthly Downloads](https://img.shields.io/packagist/dm/{vendor}/{package}.svg?style=flat-square)](https://packagist.org/packages/{vendor}/{package})

<!-- BADGES: Estado de CI/CD -->
[![Tests](https://img.shields.io/github/actions/workflow/status/{vendor}/{repo}/run-tests.yml?branch=main&label=tests&style=flat-square)](https://github.com/{vendor}/{repo}/actions?query=workflow%3Arun-tests+branch%3Amain)
[![Code Style](https://img.shields.io/github/actions/workflow/status/{vendor}/{repo}/fix-php-code-style-issues.yml?branch=main&label=code%20style&style=flat-square)](https://github.com/{vendor}/{repo}/actions?query=workflow%3A"Fix+PHP+code+style+issues"+branch%3Amain)
[![PHPStan](https://img.shields.io/github/actions/workflow/status/{vendor}/{repo}/phpstan.yml?branch=main&label=phpstan&style=flat-square)](https://github.com/{vendor}/{repo}/actions/workflows/phpstan.yml)
```

```
<!-- BADGES: Calidad de Código -->
[![PHPStan Level](https://img.shields.io/badge/PHPStan-level%20max-brightgreen.svg?style=flat-square&logo=php)](https://phpstan.org/)
[![Code Coverage](https://img.shields.io/codecov/c/github/{vendor}/{repo}?style=flat-square&logo=codecov)](https://codecov.io/gh/{vendor}/{repo})
[![Maintainability](https://img.shields.io/codeclimate/maintainability/{vendor}/{repo}?style=flat-square&logo=code-climate)](https://codeclimate.com/github/{vendor}/{repo})
[![Technical Debt](https://img.shields.io/codeclimate/tech-debt/{vendor}/{repo}?style=flat-square&logo=code-climate)](https://codeclimate.com/github/{vendor}/{repo})

<!-- BADGES: Requisitos Técnicos -->
[![PHP Version](https://img.shields.io/packagist/php-v/{vendor}/{package}.svg?style=flat-square&logo=php)](https://packagist.org/packages/{vendor}/{package})
[![Laravel Version](https://img.shields.io/badge/Laravel-12.x-red.svg?style=flat-square&logo=laravel)](https://laravel.com)

<!-- BADGES: Stack Tecnológico -->
[![Stack: RabbitMQ](https://img.shields.io/badge/Stack-RabbitMQ-FF6600?style=flat-square&logo=rabbitmq)](https://www.rabbitmq.com/)
[![Stack: Elasticsearch](https://img.shields.io/badge/Stack-Elasticsearch-005571?style=flat-square&logo=elasticsearch)](https://www.elastic.co/)
[![Stack: Redis](https://img.shields.io/badge/Stack-Redis-DC382D?style=flat-square&logo=redis)](https://redis.io/)
[![Stack: PostgreSQL](https://img.shields.io/badge/Stack-PostgreSQL-336791?style=flat-square&logo=postgresql)](https://www.postgresql.org/)

<!-- BADGES: Estado del Repositorio -->
[![Last Commit](https://img.shields.io/github/last-commit/{vendor}/{repo}?style=flat-square&logo=github)](https://github.com/{vendor}/{repo}/commits)
[![Commit Activity](https://img.shields.io/github/commit-activity/m/{vendor}/{repo}?style=flat-square&logo=github)](https://github.com/{vendor}/{repo}/graphs/commit-activity)
[![Issues](https://img.shields.io/github/issues/{vendor}/{repo}?style=flat-square&logo=github)](https://github.com/{vendor}/{repo}/issues)
[![Pull Requests](https://img.shields.io/github/issues-pr/{vendor}/{repo}?style=flat-square&logo=github)](https://github.com/{vendor}/{repo}/pulls)

<!-- BADGES: Estado del Proyecto -->
[![Project Status:
```

```
Active](https://www.repostatus.org/badges/latest/active.svg)](https://www.repostatus.org/#active)
[![Maintenance](https://img.shields.io/maintenance/yes/2025?style=flat-square)](https://github.com/{vendor}/{repo}/graphs/commit-activity)
[![License](https://img.shields.io/github/license/{vendor}/{repo}?style=flat-square)](LICENSE.md)

<!-- BADGES: Comunidad -->
[![GitHub Stars](https://img.shields.io/github/stars/{vendor}/{repo}?style=flat-square&logo=github)](https://github.com/{vendor}/{repo}/stargazers)
[![GitHub Forks](https://img.shields.io/github/forks/{vendor}/{repo}?style=flat-square&logo=github)](https://github.com/{vendor}/{repo}/network/members)
[![Contributors](https://img.shields.io/github/contributors/{vendor}/{repo}?style=flat-square&logo=github)](https://github.com/{vendor}/{repo}/graphs/contributors)

<!-- BADGES: Seguridad -->
[![Security](https://img.shields.io/badge/Security-Snyk-4C4A73?style=flat-square&logo=snyk)](https://snyk.io/)
[![Known Vulnerabilities](https://snyk.io/test/github/{vendor}/{repo}/badge.svg?style=flat-square)](https://snyk.io/test/github/{vendor}/{repo})
```

Variables a reemplazar

Variable	Ejemplo	Descripción
`{vendor}`	`AichaDigital`	Nombre de la organización/vendor
`{repo}`	`lara-verifactu`	Nombre del repositorio
`{package}`	`aichadigital/lara-verifactu`	Nombre completo del paquete en Packagist

Badges por Categoría

1. Información del Paquete (Packagist)

Versión y Descargas

```
<!-- Versión estable actual -->
[![Latest
Version](https://img.shields.io/packagist/v/{vendor}/{package})](https://packagist.org/packages/{vendor}/{package})

<!-- Versión pre-release -->
[![Latest Unstable
Version](https://img.shields.io/packagist/vpre/{vendor}/{package})](https://packagist.org/packages/{vendor}/{package})

<!-- Descargas totales -->
[![Total
Downloads](https://img.shields.io/packagist/dt/{vendor}/{package})](https://packagist.org/packages/{vendor}/{package})

<!-- Descargas mensuales -->
[![Monthly
Downloads](https://img.shields.io/packagist/dm/{vendor}/{package})](https://packagist.org/packages/{vendor}/{package})

<!-- Descargas diarias -->
[![Daily
Downloads](https://img.shields.io/packagist/dd/{vendor}/{package})](https://packagist.org/packages/{vendor}/{package})

<!-- Licencia desde Packagist -->
[![License](https://img.shields.io/packagist/l/{vendor}/{package})](https://packagist.org/packages/{vendor}/{package})
```

Configuración

- **URL**: https://packagist.org/
- **Requisitos**: Paquete publicado en Packagist
- **Actualización**: Automática tras cada release
- **Costo**: Gratuito

---

2. CI/CD y Tests (GitHub Actions)

Workflows Estándar

```
<!-- Tests generales -->
[![Tests](https://github.com/{vendor}/{repo}/actions/workflows/run-tests.yml/badge.svg)](https://github.com/{vendor}/{repo}/actions/workflows/run-tests.yml)

<!-- Code Style (Pint/PHP-CS-Fixer) -->
[![Code Style](https://github.com/{vendor}/{repo}/actions/workflows/fix-php-code-style-issues.yml/badge.svg)](https://github.com/{vendor}/{repo}/actions/workflows/fix-php-code-style-issues.yml)

<!-- PHPStan -->
[![PHPStan](https://github.com/{vendor}/{repo}/actions/workflows/phpstan.yml/badge.svg)](https://github.com/{vendor}/{repo}/actions/workflows/phpstan.yml)

<!-- Pest/PHPUnit -->
[![Tests](https://github.com/{vendor}/{repo}/actions/workflows/tests.yml/badge.svg)](https://github.com/{vendor}/{repo}/actions/workflows/tests.yml)

<!-- Build Status (genérico) -->
[![Build Status](https://github.com/{vendor}/{repo}/workflows/CI/badge.svg)](https://github.com/{vendor}/{repo}/actions)

<!-- Estado específico de workflow con parámetros -->
[![Tests](https://img.shields.io/github/actions/workflow/status/{vendor}/{repo}/run-tests.yml?branch=main&label=tests)](https://github.com/{vendor}/{repo}/actions/workflows/run-tests.yml)
```

Configuración

- **Ubicación**: `.github/workflows/*.yml`
- **Requisitos**: Workflows activos en GitHub Actions
- **Actualización**: Automática en cada push/PR
- **Costo**: Gratuito para repos públicos

Ejemplo de workflow mínimo:

```
# .github/workflows/run-tests.yml
name: Tests

on: [push, pull_request]
```

```
jobs:
  test:
    runs-on: ubuntu-latest
    steps:

      - uses: actions/checkout@v4
      - name: Setup PHP
        uses: shivammathur/setup-php@v2
        with:
          php-version: 8.3

      - run: composer install
      - run: vendor/bin/pest
```

---

3. Calidad de Código

PHPStan (Badges Manuales)

```
<!-- Level específico (0-9) - ACTUALIZAR MANUALMENTE -->
[![PHPStan Level 0](https://img.shields.io/badge/PHPStan-level%200-
red.svg?logo=php)](https://phpstan.org/)
[![PHPStan Level 1](https://img.shields.io/badge/PHPStan-level%201-
orange.svg?logo=php)](https://phpstan.org/)
[![PHPStan Level 2](https://img.shields.io/badge/PHPStan-level%202-
yellow.svg?logo=php)](https://phpstan.org/)
[![PHPStan Level 3](https://img.shields.io/badge/PHPStan-level%203-
yellowgreen.svg?logo=php)](https://phpstan.org/)
[![PHPStan Level 4](https://img.shields.io/badge/PHPStan-level%204-
yellowgreen.svg?logo=php)](https://phpstan.org/)
[![PHPStan Level 5](https://img.shields.io/badge/PHPStan-level%205-
green.svg?logo=php)](https://phpstan.org/)
[![PHPStan Level 6](https://img.shields.io/badge/PHPStan-level%206-
green.svg?logo=php)](https://phpstan.org/)
[![PHPStan Level 7](https://img.shields.io/badge/PHPStan-level%207-
brightgreen.svg?logo=php)](https://phpstan.org/)
[![PHPStan Level 8](https://img.shields.io/badge/PHPStan-level%208-
brightgreen.svg?logo=php)](https://phpstan.org/)
[![PHPStan Level 9](https://img.shields.io/badge/PHPStan-level%209-
```

```
brightgreen.svg?logo=php)](https://phpstan.org/)

<!-- Max Level (para proyectos con level 9) -->
[![PHPStan Level](https://img.shields.io/badge/PHPStan-level%20max-
brightgreen.svg?logo=php)](https://phpstan.org/)

<!-- Badge con workflow dinámico (muestra estado del workflow) -->
[![PHPStan](https://github.com/{vendor}/{repo}/actions/workflows/phpstan.yml/badge.svg)](https
://github.com/{vendor}/{repo}/actions/workflows/phpstan.yml)
```

Nota importante: PHPStan no tiene badge oficial dinámico que lea el level automáticamente. Debes actualizar el badge manualmente cuando cambies el level. Ver Solución Custom para PHPStan más abajo.

Codecov

```
<!-- Badge básico -->
[![Code
Coverage](https://codecov.io/gh/{vendor}/{repo}/branch/main/graph/badge.svg)](https://codecov.
io/gh/{vendor}/{repo})

<!-- Con token (para repos privados) -->
[![codecov](https://codecov.io/gh/{vendor}/{repo}/branch/main/graph/badge.svg?token=YOUR_TOKEN
)](https://codecov.io/gh/{vendor}/{repo})

<!-- Badge con porcentaje visible -->
[![Coverage](https://img.shields.io/codecov/c/github/{vendor}/{repo}?logo=codecov)](https://co
decov.io/gh/{vendor}/{repo})
```

Configuración:

1. Crear cuenta en https://codecov.io/
2. Vincular repositorio
3. Obtener token (Settings > Repository > Upload Token)
4. Añadir token a GitHub Secrets: `CODECOV_TOKEN`
5. Configurar workflow:

```
# .github/workflows/tests.yml
```

```
- name: Run tests with coverage
  run: vendor/bin/pest --coverage --coverage-clover=coverage.xml

- name: Upload coverage to Codecov
  uses: codecov/codecov-action@v3
  with:
    files: ./coverage.xml
    fail_ci_if_error: true
    token: ${{ secrets.CODECOV_TOKEN }}
```

Costo: Gratuito para repos open source

Code Climate

```
<!-- Maintainability -->
[![Maintainability](https://api.codeclimate.com/v1/badges/{badge_id}/maintainability)](https://codeclimate.com/github/{vendor}/{repo}/maintainability)

<!-- Test Coverage -->
[![Test Coverage](https://api.codeclimate.com/v1/badges/{badge_id}/test_coverage)](https://codeclimate.com/github/{vendor}/{repo}/test_coverage)

<!-- Technical Debt -->
[![Technical Debt](https://img.shields.io/codeclimate/tech-debt/{vendor}/{repo})](https://codeclimate.com/github/{vendor}/{repo})

<!-- Badge de issues -->
[![Code Issues](https://img.shields.io/codeclimate/issues/{vendor}/{repo})](https://codeclimate.com/github/{vendor}/{repo}/issues)
```

Configuración:

1. Crear cuenta en https://codeclimate.com/
2. Conectar repositorio
3. Obtener `badge_id` del dashboard (Repo Settings > Badges)
4. Crear `.codeclimate.yml`:

```
version: "2"
checks:
  argument-count:
    enabled: true
    config:
      threshold: 4
  complex-logic:
    enabled: true
  method-complexity:
    enabled: true
    config:
      threshold: 10
  method-lines:
    enabled: true
    config:
      threshold: 50
```

Costo: Gratuito para repos open source

Scrutinizer

```
<!-- Quality Score -->
[![Scrutinizer Code Quality](https://scrutinizer-ci.com/g/{vendor}/{repo}/badges/quality-score.png?b=main)](https://scrutinizer-ci.com/g/{vendor}/{repo}/?branch=main)

<!-- Coverage -->
[![Code Coverage](https://scrutinizer-ci.com/g/{vendor}/{repo}/badges/coverage.png?b=main)](https://scrutinizer-ci.com/g/{vendor}/{repo}/?branch=main)

<!-- Build Status -->
[![Build Status](https://scrutinizer-ci.com/g/{vendor}/{repo}/badges/build.png?b=main)](https://scrutinizer-ci.com/g/{vendor}/{repo}/build-status/main)
```

Configuración:

- URL: https://scrutinizer-ci.com/
- Requisitos: Conectar repo, configurar `.scrutinizer.yml`

- Costo: Gratuito para repos open source

SonarCloud

```
<!-- Quality Gate Status -->
[![Quality Gate
Status](https://sonarcloud.io/api/project_badges/measure?project={project_key}&metric=alert_st
atus)](https://sonarcloud.io/dashboard?id={project_key})

<!-- Bugs -->
[![Bugs](https://sonarcloud.io/api/project_badges/measure?project={project_key}&metric=bugs)](
https://sonarcloud.io/dashboard?id={project_key})

<!-- Code Smells -->
[![Code
Smells](https://sonarcloud.io/api/project_badges/measure?project={project_key}&metric=code_sme
lls)](https://sonarcloud.io/dashboard?id={project_key})

<!-- Coverage -->
[![Coverage](https://sonarcloud.io/api/project_badges/measure?project={project_key}&metric=cov
erage)](https://sonarcloud.io/dashboard?id={project_key})

<!-- Duplicated Lines -->
[![Duplicated
Lines](https://sonarcloud.io/api/project_badges/measure?project={project_key}&metric=duplicate
d_lines_density)](https://sonarcloud.io/dashboard?id={project_key})

<!-- Technical Debt -->
[![Technical
Debt](https://sonarcloud.io/api/project_badges/measure?project={project_key}&metric=sqale_inde
x)](https://sonarcloud.io/dashboard?id={project_key})

<!-- Vulnerabilities -->
[![Vulnerabilities](https://sonarcloud.io/api/project_badges/measure?project={project_key}&met
ric=vulnerabilities)](https://sonarcloud.io/dashboard?id={project_key})

<!-- Security Rating -->
[![Security
Rating](https://sonarcloud.io/api/project_badges/measure?project={project_key}&metric=security
_rating)](https://sonarcloud.io/dashboard?id={project_key})
```

```
<!-- Maintainability Rating -->
[![Maintainability
Rating](https://sonarcloud.io/api/project_badges/measure?project={project_key}&metric=sqale_ra
ting)](https://sonarcloud.io/dashboard?id={project_key})

<!-- Reliability Rating -->
[![Reliability
Rating](https://sonarcloud.io/api/project_badges/measure?project={project_key}&metric=reliabil
ity_rating)](https://sonarcloud.io/dashboard?id={project_key})
```

Configuración:

- URL: https://sonarcloud.io/
- Requisitos: Vincular con GitHub/GitLab, configurar `sonar-project.properties`
- Costo: Gratuito para repos open source

---

4. Requisitos Técnicos

```
<!-- PHP Version (desde composer.json - DINÁMICO) -->
[![PHP Version](https://img.shields.io/packagist/php-
v/{vendor}/{package}?logo=php)](https://packagist.org/packages/{vendor}/{package})

<!-- PHP Version específica (manual) -->
[![PHP](https://img.shields.io/badge/PHP-%3E%3D8.2-777BB4?logo=php)](https://www.php.net/)
[![PHP](https://img.shields.io/badge/PHP-%3E%3D8.3-777BB4?logo=php)](https://www.php.net/)

<!-- Laravel Version (manual) -->
[![Laravel](https://img.shields.io/badge/Laravel-11.x-red?logo=laravel)](https://laravel.com)
[![Laravel](https://img.shields.io/badge/Laravel-12.x-red?logo=laravel)](https://laravel.com)

<!-- Versión múltiple de Laravel -->
[![Laravel](https://img.shields.io/badge/Laravel-11.x%20%7C%2012.x-
red?logo=laravel)](https://laravel.com)

<!-- Composer Version -->
[![Composer](https://img.shields.io/badge/Composer-2.x-
885630?logo=composer)](https://getcomposer.org/)
```

```
<!-- Node.js (si aplica) -->
[![Node.js](https://img.shields.io/badge/Node.js-%3E%3D20-
339933?logo=node.js)](https://nodejs.org/)
```

Nota: El badge de PHP desde Packagist lee automáticamente el `require.php` de `composer.json`.

---

5. Stack Tecnológico

Bases de Datos

```
<!-- MySQL -->
[![MySQL](https://img.shields.io/badge/MySQL-8.0-
4479A1?logo=mysql&logoColor=white)](https://www.mysql.com/)
[![MySQL](https://img.shields.io/badge/MySQL-8.x-
4479A1?logo=mysql&logoColor=white)](https://www.mysql.com/)

<!-- PostgreSQL -->
[![PostgreSQL](https://img.shields.io/badge/PostgreSQL-16-
336791?logo=postgresql)](https://www.postgresql.org/)
[![PostgreSQL](https://img.shields.io/badge/PostgreSQL-15%20%7C%2016-
336791?logo=postgresql)](https://www.postgresql.org/)

<!-- MariaDB -->
[![MariaDB](https://img.shields.io/badge/MariaDB-11.x-
003545?logo=mariadb)](https://mariadb.org/)

<!-- SQLite -->
[![SQLite](https://img.shields.io/badge/SQLite-3.x-
003B57?logo=sqlite)](https://www.sqlite.org/)

<!-- MongoDB -->
[![MongoDB](https://img.shields.io/badge/MongoDB-7.x-
47A248?logo=mongodb&logoColor=white)](https://www.mongodb.com/)
```

Message Queues

```
<!-- RabbitMQ -->
[![RabbitMQ](https://img.shields.io/badge/RabbitMQ-3.x-
FF6600?logo=rabbitmq)](https://www.rabbitmq.com/)

<!-- Apache Kafka -->
[![Apache Kafka](https://img.shields.io/badge/Kafka-latest-231F20?logo=apache-
kafka)](https://kafka.apache.org/)

<!-- AWS SQS -->
[![AWS SQS](https://img.shields.io/badge/AWS-SQS-FF9900?logo=amazon-
aws)](https://aws.amazon.com/sqs/)
```

Cache y Storage

```
<!-- Redis -->
[![Redis](https://img.shields.io/badge/Redis-7.x-
DC382D?logo=redis&logoColor=white)](https://redis.io/)

<!-- Memcached -->
[![Memcached](https://img.shields.io/badge/Memcached-latest-00D8FF)](https://memcached.org/)

<!-- MinIO -->
[![MinIO](https://img.shields.io/badge/MinIO-latest-
C72E49?logo=minio&logoColor=white)](https://min.io/)
```

Search Engines

```
<!-- Elasticsearch -->
[![Elasticsearch](https://img.shields.io/badge/Elasticsearch-8.x-
005571?logo=elasticsearch)](https://www.elastic.co/)

<!-- Algolia -->
[![Algolia](https://img.shields.io/badge/Algolia-latest-
5468FF?logo=algolia&logoColor=white)](https://www.algolia.com/)

<!-- MeiliSearch -->
[![MeiliSearch](https://img.shields.io/badge/MeiliSearch-latest-
FF69B4?logo=meilisearch&logoColor=white)](https://www.meilisearch.com/)
```

```
<!-- Typesense -->
[![Typesense](https://img.shields.io/badge/Typesense-latest-D91A5B)](https://typesense.org/)
```

Frontend (si aplica)

```
<!-- Vue.js -->
[![Vue.js](https://img.shields.io/badge/Vue.js-3.x-
4FC08D?logo=vue.js&logoColor=white)](https://vuejs.org/)

<!-- React -->
[![React](https://img.shields.io/badge/React-18.x-
61DAFB?logo=react&logoColor=black)](https://reactjs.org/)

<!-- Alpine.js -->
[![Alpine.js](https://img.shields.io/badge/Alpine.js-3.x-
8BC0D0?logo=alpine.js&logoColor=black)](https://alpinejs.dev/)

<!-- Livewire -->
[![Livewire](https://img.shields.io/badge/Livewire-3.x-
FB70A9?logo=livewire)](https://livewire.laravel.com/)

<!-- Inertia.js -->
[![Inertia.js](https://img.shields.io/badge/Inertia.js-latest-
9553E9?logo=inertia)](https://inertiajs.com/)

<!-- Tailwind CSS -->
[![Tailwind CSS](https://img.shields.io/badge/Tailwind-3.x-06B6D4?logo=tailwind-
css&logoColor=white)](https://tailwindcss.com/)

<!-- Bootstrap -->
[![Bootstrap](https://img.shields.io/badge/Bootstrap-5.x-
7952B3?logo=bootstrap&logoColor=white)](https://getbootstrap.com/)
```

DevOps y Contenedores

```
<!-- Docker -->
[![Docker](https://img.shields.io/badge/Docker-latest-
```

```
2496ED?logo=docker&logoColor=white)](https://www.docker.com/)

<!-- Kubernetes -->
[![Kubernetes](https://img.shields.io/badge/Kubernetes-latest-
326CE5?logo=kubernetes&logoColor=white)](https://kubernetes.io/)

<!-- Proxmox -->
[![Proxmox](https://img.shields.io/badge/Proxmox-VE-
E57000?logo=proxmox&logoColor=white)](https://www.proxmox.com/)

<!-- GitHub Actions -->
[![GitHub Actions](https://img.shields.io/badge/GitHub-Actions-2088FF?logo=github-
actions&logoColor=white)](https://github.com/features/actions)

<!-- GitLab CI -->
[![GitLab CI](https://img.shields.io/badge/GitLab-CI-
FC6D26?logo=gitlab&logoColor=white)](https://docs.gitlab.com/ee/ci/)
```

6. Estado del Repositorio

```
<!-- Último commit (DINÁMICO) -->
[![Last Commit](https://img.shields.io/github/last-
commit/{vendor}/{repo}?logo=github)](https://github.com/{vendor}/{repo}/commits)

<!-- Actividad de commits mensual (DINÁMICO) -->
[![Commit Activity](https://img.shields.io/github/commit-
activity/m/{vendor}/{repo}?logo=github)](https://github.com/{vendor}/{repo}/graphs/commit-
activity)

<!-- Actividad de commits semanal (DINÁMICO) -->
[![Commit Activity](https://img.shields.io/github/commit-
activity/w/{vendor}/{repo}?logo=github)](https://github.com/{vendor}/{repo}/graphs/commit-
activity)

<!-- Actividad de commits anual (DINÁMICO) -->
[![Commit Activity](https://img.shields.io/github/commit-
activity/y/{vendor}/{repo}?logo=github)](https://github.com/{vendor}/{repo}/graphs/commit-
```

```
activity)

<!-- Issues abiertas (DINÁMICO) -->
[![Open
Issues](https://img.shields.io/github/issues/{vendor}/{repo}?logo=github)](https://github.com/
{vendor}/{repo}/issues)

<!-- Issues cerradas (DINÁMICO) -->
[![Closed Issues](https://img.shields.io/github/issues-
closed/{vendor}/{repo}?logo=github)](https://github.com/{vendor}/{repo}/issues?q=is%3Aissue+is
%3Aclosed)

<!-- Pull Requests abiertas (DINÁMICO) -->
[![Open PRs](https://img.shields.io/github/issues-
pr/{vendor}/{repo}?logo=github)](https://github.com/{vendor}/{repo}/pulls)

<!-- Pull Requests cerradas (DINÁMICO) -->
[![Closed PRs](https://img.shields.io/github/issues-pr-
closed/{vendor}/{repo}?logo=github)](https://github.com/{vendor}/{repo}/pulls?q=is%3Apr+is%3Ac
losed)

<!-- Tamaño del repositorio (DINÁMICO) -->
[![Repo Size](https://img.shields.io/github/repo-
size/{vendor}/{repo}?logo=github)](https://github.com/{vendor}/{repo})

<!-- Lenguaje principal (DINÁMICO) -->
[![Top
Language](https://img.shields.io/github/languages/top/{vendor}/{repo}?logo=github)](https://gi
thub.com/{vendor}/{repo})

<!-- Conteo de lenguajes (DINÁMICO) -->
[![Language
Count](https://img.shields.io/github/languages/count/{vendor}/{repo}?logo=github)](https://git
hub.com/{vendor}/{repo})

<!-- Última release (DINÁMICO) -->
[![Latest
Release](https://img.shields.io/github/v/release/{vendor}/{repo}?logo=github)](https://github.
com/{vendor}/{repo}/releases/latest)
```

```
<!-- Release date (DINÁMICO) -->
[![Release Date](https://img.shields.io/github/release-
date/{vendor}/{repo}?logo=github)](https://github.com/{vendor}/{repo}/releases)
```

---

7. Estado del Proyecto (dinámicos)

RepoStatus.org (Estados Oficiales)

```
<!-- Active: Proyecto activo con desarrollo regular -->
[![Active](https://www.repostatus.org/badges/latest/active.svg)](https://www.repostatus.org/#a
ctive)

<!-- WIP: Work In Progress, funcional pero en desarrollo -->
[![WIP](https://www.repostatus.org/badges/latest/wip.svg)](https://www.repostatus.org/#wip)

<!-- Suspended: Temporalmente detenido -->
[![Suspended](https://www.repostatus.org/badges/latest/suspended.svg)](https://www.repostatus.
org/#suspended)

<!-- Abandoned: No mantenido -->
[![Abandoned](https://www.repostatus.org/badges/latest/abandoned.svg)](https://www.repostatus.
org/#abandoned)

<!-- Moved: Repositorio movido a otra ubicación -->
[![Moved](https://www.repostatus.org/badges/latest/moved.svg)](https://www.repostatus.org/#mov
ed)

<!-- Inactive: Funcional pero no mantenido activamente -->
[![Inactive](https://www.repostatus.org/badges/latest/inactive.svg)](https://www.repostatus.or
g/#inactive)

<!-- Unsupported: Funcional pero sin soporte -->
[![Unsupported](https://www.repostatus.org/badges/latest/unsupported.svg)](https://www.reposta
tus.org/#unsupported)

<!-- Concept: En fase de concepto -->
[![Concept](https://www.repostatus.org/badges/latest/concept.svg)](https://www.repostatus.org/
#concept)
```

URL: https://www.repostatus.org/
Actualización: Manual (copiar el badge apropiado)

Maintenance Status (Año específico)

```
<!-- Mantenido actualmente (cambiar año según corresponda) -->
[![Maintenance](https://img.shields.io/maintenance/yes/2025)](https://github.com/{vendor}/{repo})

<!-- No mantenido -->
[![Maintenance](https://img.shields.io/maintenance/no/2025)](https://github.com/{vendor}/{repo})
```

Nota: Actualizar el año anualmente.

---

8. Licencia

```
<!-- Licencia desde GitHub (DINÁMICO) -->
[![License](https://img.shields.io/github/license/{vendor}/{repo})](https://github.com/{vendor}/{repo}/blob/main/LICENSE.md)

<!-- MIT -->
[![License: MIT](https://img.shields.io/badge/License-MIT-yellow.svg)](https://opensource.org/licenses/MIT)

<!-- GPL v3 -->
[![License: GPL v3](https://img.shields.io/badge/License-GPLv3-blue.svg)](https://www.gnu.org/licenses/gpl-3.0)

<!-- Apache 2.0 -->
[![License: Apache 2.0](https://img.shields.io/badge/License-Apache%202.0-blue.svg)](https://opensource.org/licenses/Apache-2.0)

<!-- BSD 3-Clause -->
[![License: BSD-3](https://img.shields.io/badge/License-BSD%203--Clause-blue.svg)](https://opensource.org/licenses/BSD-3-Clause)
```

```
<!-- ISC -->
[![License: ISC](https://img.shields.io/badge/License-ISC-
blue.svg)](https://opensource.org/licenses/ISC)

<!-- Propietaria -->
[![License: Proprietary](https://img.shields.io/badge/License-Proprietary-
red.svg)](LICENSE.md)
```

---

9. Comunidad

```
<!-- Stars (DINÁMICO) -->
[![GitHub
Stars](https://img.shields.io/github/stars/{vendor}/{repo}?style=social)](https://github.com/{
vendor}/{repo}/stargazers)

<!-- Forks (DINÁMICO) -->
[![GitHub
Forks](https://img.shields.io/github/forks/{vendor}/{repo}?style=social)](https://github.com/{
vendor}/{repo}/network/members)

<!-- Watchers (DINÁMICO) -->
[![GitHub
Watchers](https://img.shields.io/github/watchers/{vendor}/{repo}?style=social)](https://github
.com/{vendor}/{repo}/watchers)

<!-- Contributors (DINÁMICO) -->
[![Contributors](https://img.shields.io/github/contributors/{vendor}/{repo})](https://github.c
om/{vendor}/{repo}/graphs/contributors)

<!-- Discussions (DINÁMICO) -->
[![GitHub
Discussions](https://img.shields.io/github/discussions/{vendor}/{repo})](https://github.com/{v
endor}/{repo}/discussions)

<!-- Sponsors -->
[![GitHub
Sponsors](https://img.shields.io/github/sponsors/{vendor}?logo=github)](https://github.com/spo
```

```
nsors/{vendor})

<!-- Discord -->
[![Discord](https://img.shields.io/discord/{server_id}?logo=discord&label=Discord)](https://discord.gg/{invite_code})

<!-- Slack -->
[![Slack](https://img.shields.io/badge/Slack-Join-4A154B?logo=slack)](https://join.slack.com/...)

<!-- Twitter -->
[![Twitter Follow](https://img.shields.io/twitter/follow/{handle}?style=social)](https://twitter.com/{handle})
```

10. Seguridad

```
<!-- Snyk (DINÁMICO) -->
[![Known Vulnerabilities](https://snyk.io/test/github/{vendor}/{repo}/badge.svg)](https://snyk.io/test/github/{vendor}/{repo})

<!-- Dependabot (manual) -->
[![Dependabot](https://img.shields.io/badge/Dependabot-enabled-success?logo=dependabot)](https://github.com/{vendor}/{repo}/network/dependencies)

<!-- Security Policy (manual) -->
[![Security Policy](https://img.shields.io/badge/Security-Policy-blue)](https://github.com/{vendor}/{repo}/security/policy)

<!-- OSSF Best Practices (DINÁMICO tras registro) -->
[![OpenSSF Best Practices](https://bestpractices.coreinfrastructure.org/projects/{project_id}/badge)](https://bestpractices.coreinfrastructure.org/projects/{project_id})

<!-- OSSF Scorecard -->
[![OpenSSF
```

```
Scorecard](https://api.securityscorecards.dev/projects/github.com/{vendor}/{repo}/badge)](https://api.securityscorecards.dev/projects/github.com/{vendor}/{repo})
```

Configuración Snyk:

1. Crear cuenta en https://snyk.io/
2. Conectar repositorio
3. Badge aparece automáticamente

Costo: Gratuito para repos open source

---

11. Documentación

```
<!-- ReadTheDocs (DINÁMICO) -->
[![Documentation](https://readthedocs.org/projects/{project}/badge/?version=latest)](https://{project}.readthedocs.io/en/latest/)

<!-- GitHub Pages (manual) -->
[![Documentation](https://img.shields.io/badge/docs-GitHub%20Pages-blue?logo=github)](https://{vendor}.github.io/{repo}/)

<!-- Wiki (manual) -->
[![Wiki](https://img.shields.io/badge/docs-Wiki-blue?logo=github)](https://github.com/{vendor}/{repo}/wiki)

<!-- Gitbook -->
[![Gitbook](https://img.shields.io/badge/docs-Gitbook-blue?logo=gitbook&logoColor=white)](https://{organization}.gitbook.io/{project}/)

<!-- Badge de "docs" personalizado -->
[![Documentation](https://img.shields.io/badge/docs-latest-blue)](https://docs.{domain}.com)
```

Servicios Externos Recomendados

Servicio	Propósito	URL	API Key	Costo OSS	Configuración

Packagist	Stats de paquetes PHP	https://packagist.org/	No	Gratis	Automático tras publish
Codecov	Code Coverage	https://codecov.io/	Sí	Gratis	Token en GitHub Secrets
Code Climate	Calidad de código	https://codeclimate.com/	Sí	Gratis	Conectar repo + `.codeclimate.yml`
Scrutinizer	Análisis estático	https://scrutinizer-ci.com/	No	Gratis	Conectar + `.scrutinizer.yml`
SonarCloud	Análisis de seguridad	https://sonarcloud.io/	Sí	Gratis	`sonar-project.properties`
Snyk	Vulnerabilidades	https://snyk.io/	Sí	Gratis	Conectar repositorio
Shields.io	Badges personalizados	https://shields.io/	No	Gratis	-
RepoStatus.org	Estado del proyecto	https://www.repostatus.org/	No	Gratis	-
OSSF	Security Scorecard	https://securityscorecards.dev/	No	Gratis	Configurar workflow

---

Ejemplos Completos por Tipo de Proyecto

Ejemplo 1: Paquete Laravel con Testing Completo

```
# Lara-Verifactu

> Paquete Laravel para integración con el sistema VeriFactu de la AEAT

[![Latest Version](https://img.shields.io/packagist/v/aichadigital/lara-verifactu?style=flat-square)](https://packagist.org/packages/aichadigital/lara-verifactu)
[![Total Downloads](https://img.shields.io/packagist/dt/aichadigital/lara-verifactu?style=flat-square)](https://packagist.org/packages/aichadigital/lara-verifactu)
[![Tests](https://img.shields.io/github/actions/workflow/status/AichaDigital/lara-verifactu/run-tests.yml?branch=main&label=tests&style=flat-square)](https://github.com/AichaDigital/lara-verifactu/actions/workflows/run-tests.yml)
```

```
[![PHPStan Level 9](https://img.shields.io/badge/PHPStan-level%209-brightgreen?style=flat-square&logo=php)](https://phpstan.org/)
[![Code Coverage](https://img.shields.io/codecov/c/github/AichaDigital/lara-verifactu?style=flat-square&logo=codecov)](https://codecov.io/gh/AichaDigital/lara-verifactu)

[![PHP Version](https://img.shields.io/packagist/php-v/aichadigital/lara-verifactu?style=flat-square&logo=php)](https://packagist.org/packages/aichadigital/lara-verifactu)
[![Laravel Version](https://img.shields.io/badge/Laravel-11.x%20%7C%2012.x-red?style=flat-square&logo=laravel)](https://laravel.com)
[![License](https://img.shields.io/github/license/AichaDigital/lara-verifactu?style=flat-square)](LICENSE.md)

## Instalación

\```bash
composer require aichadigital/lara-verifactu
\```
```

Ejemplo 2: Proyecto Laravel con Stack Completo

```
# Sistema de Subastas Online

> Plataforma empresarial para subastas en tiempo real

[![Project Status: Active](https://www.repostatus.org/badges/latest/active.svg)](https://www.repostatus.org/#active)
[![Maintenance](https://img.shields.io/maintenance/yes/2025?style=flat-square)](https://github.com/AichaDigital/subastas-app)
[![Tests](https://img.shields.io/github/actions/workflow/status/AichaDigital/subastas-app/tests.yml?branch=main&label=tests&style=flat-square)](https://github.com/AichaDigital/subastas-app/actions/workflows/tests.yml)
[![Last Commit](https://img.shields.io/github/last-commit/AichaDigital/subastas-app?style=flat-square)](https://github.com/AichaDigital/subastas-app/commits)

**Stack Tecnológico:**
```

```
[![PHP 8.3](https://img.shields.io/badge/PHP-8.3-777BB4?style=flat-
square&logo=php)](https://www.php.net/)
[![Laravel 12](https://img.shields.io/badge/Laravel-12.x-red?style=flat-
square&logo=laravel)](https://laravel.com)
[![Livewire](https://img.shields.io/badge/Livewire-3.x-FB70A9?style=flat-
square&logo=livewire)](https://livewire.laravel.com/)

**Infraestructura:**

[![PostgreSQL](https://img.shields.io/badge/PostgreSQL-16-336791?style=flat-
square&logo=postgresql)](https://www.postgresql.org/)
[![Redis](https://img.shields.io/badge/Redis-7.x-DC382D?style=flat-
square&logo=redis&logoColor=white)](https://redis.io/)
[![RabbitMQ](https://img.shields.io/badge/RabbitMQ-3.x-FF6600?style=flat-
square&logo=rabbitmq)](https://www.rabbitmq.com/)
[![Elasticsearch](https://img.shields.io/badge/Elasticsearch-8.x-005571?style=flat-
square&logo=elasticsearch)](https://www.elastic.co/)

[![License: Proprietary](https://img.shields.io/badge/License-Proprietary-red?style=flat-
square)](LICENSE.md)
```

Ejemplo 3: Paquete PHP Standalone

```
# PHP-Utils

> Utilidades PHP para proyectos empresariales

[![Latest Version](https://img.shields.io/packagist/v/xerintel/php-utils?style=flat-
square)](https://packagist.org/packages/xerintel/php-utils)
[![PHP Version](https://img.shields.io/packagist/php-v/xerintel/php-utils?style=flat-
square&logo=php)](https://www.php.net/)
[![Tests](https://img.shields.io/github/actions/workflow/status/xerintel/php-
utils/tests.yml?branch=main&label=tests&style=flat-square)](https://github.com/xerintel/php-
utils/actions/workflows/tests.yml)
[![PHPStan Level 8](https://img.shields.io/badge/PHPStan-level%208-brightgreen?style=flat-
square&logo=php)](https://phpstan.org/)
[![License: MIT](https://img.shields.io/badge/License-MIT-yellow?style=flat-
```

```
square)](LICENSE.md)
```

Ejemplo 4: Proyecto con Monitorización y Seguridad

```
# Enterprise CRM

[![Active](https://www.repostatus.org/badges/latest/active.svg)](https://www.repostatus.org/#active)
[![Tests](https://img.shields.io/github/actions/workflow/status/empresa/crm/tests.yml?branch=main&style=flat-square)](https://github.com/empresa/crm/actions)
[![Security](https://snyk.io/test/github/empresa/crm/badge.svg?style=flat-square)](https://snyk.io/test/github/empresa/crm)

**Calidad:**

[![Code Coverage](https://img.shields.io/codecov/c/github/empresa/crm?style=flat-square)](https://codecov.io/gh/empresa/crm)
[![Maintainability](https://api.codeclimate.com/v1/badges/ABC123/maintainability)](https://codeclimate.com/github/empresa/crm)
[![Quality Gate](https://sonarcloud.io/api/project_badges/measure?project=empresa_crm&metric=alert_status)](https://sonarcloud.io/dashboard?id=empresa_crm)

**Stack:**

[![Laravel 12](https://img.shields.io/badge/Laravel-12.x-red?style=flat-square&logo=laravel)](https://laravel.com)
[![Vue.js 3](https://img.shields.io/badge/Vue.js-3.x-4FC08D?style=flat-square&logo=vue.js)](https://vuejs.org/)
[![MySQL 8](https://img.shields.io/badge/MySQL-8.0-4479A1?style=flat-square&logo=mysql)](https://www.mysql.com/)
```

Soluciones Especiales

Badge Dinámico para PHPStan (Solución Custom)

Como PHPStan no ofrece badges dinámicos oficiales, aquí hay una solución mediante GitHub Actions:

Opción 1: Badge con GitHub Actions Status

```
# .github/workflows/phpstan-badge.yml
name: PHPStan Badge

on:
  push:
    branches: [main]
  pull_request:

jobs:
  phpstan:
    name: PHPStan Level Check
    runs-on: ubuntu-latest

    steps:

      - uses: actions/checkout@v4

      - name: Setup PHP
        uses: shivammathur/setup-php@v2
        with:
          php-version: 8.3

      - name: Install dependencies
        run: composer install --no-interaction

      - name: Run PHPStan
        run: vendor/bin/phpstan analyse --no-progress
```

Badge a usar:

```
[![PHPStan](https://img.shields.io/github/actions/workflow/status/{vendor}/{repo}/phpstan-badge.yml?branch=main&label=PHPStan%20Level%209&style=flat-square)](https://github.com/{vendor}/{repo}/actions/workflows/phpstan-badge.yml)
```

Opción 2: Badge Dinámico con Gist (Avanzado)

```
# .github/workflows/phpstan-dynamic-badge.yml
name: PHPStan Dynamic Badge

on:
  push:
    branches: [main]

jobs:
  badge:
    runs-on: ubuntu-latest
    steps:

      - uses: actions/checkout@v4

      - name: Setup PHP
        uses: shivammathur/setup-php@v2
        with:
          php-version: 8.3

      - name: Install dependencies
        run: composer install --no-interaction

      - name: Extract PHPStan Level
        id: level
        run: |
          if [ -f phpstan.neon ]; then
            LEVEL=$(grep -oP "level: \K\d+" phpstan.neon || echo "0")
          elif [ -f phpstan.neon.dist ]; then
            LEVEL=$(grep -oP "level: \K\d+" phpstan.neon.dist || echo "0")
          else
            LEVEL="0"
          fi
          echo "level=$LEVEL" >> $GITHUB_OUTPUT
```

```
          echo "PHPStan Level: $LEVEL"

      - name: Create Badge
        uses: schneegans/dynamic-badges-action@v1.7.0
        with:
          auth: ${{ secrets.GIST_SECRET }}
          gistID: YOUR_GIST_ID_HERE
          filename: phpstan-${{ github.event.repository.name }}.json
          label: PHPStan
          message: level ${{ steps.level.outputs.level }}
          color: brightgreen
```

Configuración:

1. Crear un GitHub Personal Access Token con scope `gist`
2. Añadirlo a GitHub Secrets como `GIST_SECRET`
3. Crear un Gist público en https://gist.github.com/
4. Copiar el Gist ID (última parte de la URL)
5. Reemplazar `YOUR_GIST_ID_HERE` en el workflow

Badge a usar:

```
[![PHPStan](https://img.shields.io/endpoint?url=https://gist.githubusercontent.com/{username}/{gist_id}/raw/phpstan-{repo}.json&style=flat-square)](https://phpstan.org/)
```

Opción 3: Badge Manual (Más simple)

Si prefieres simplicidad, actualiza manualmente el badge cuando cambies el level:

```
<!-- Level 0 (rojo) -->
[![PHPStan Level 0](https://img.shields.io/badge/PHPStan-level%200-red.svg?style=flat-square&logo=php)](https://phpstan.org/)

<!-- Level 5 (verde) -->
[![PHPStan Level 5](https://img.shields.io/badge/PHPStan-level%205-green.svg?style=flat-square&logo=php)](https://phpstan.org/)

<!-- Level 8 (verde brillante) -->
[![PHPStan Level 8](https://img.shields.io/badge/PHPStan-level%208-brightgreen.svg?style=flat-square&logo=php)](https://phpstan.org/)
```

```
<!-- Level 9 / Max (verde brillante) -->
[![PHPStan Level 9](https://img.shields.io/badge/PHPStan-level%209-brightgreen.svg?style=flat-square&logo=php)](https://phpstan.org/)
[![PHPStan Max](https://img.shields.io/badge/PHPStan-level%20max-brightgreen.svg?style=flat-square&logo=php)](https://phpstan.org/)
```

Colores recomendados por level:

- Level 0-2: `red`
- Level 3-4: `orange` o `yellow`
- Level 5-6: `yellowgreen` o `green`
- Level 7-9: `brightgreen`

Badges Personalizados con Shields.io

Para crear badges totalmente personalizados:

```
<!-- Badge estático simple -->
[![Custom](https://img.shields.io/badge/{label}-{message}-{color})](https://example.com)

<!-- Ejemplos reales -->
[![Hosted](https://img.shields.io/badge/Hosted-Proxmox-E57000?logo=proxmox)](https://www.proxmox.com/)
[![Monitoring](https://img.shields.io/badge/Monitoring-Zabbix-D20000?logo=zabbix)](https://www.zabbix.com/)
[![Stack](https://img.shields.io/badge/Stack-LEMP-339933)](https://www.nginx.com/)

<!-- Badge con endpoint dinámico -->
[![Dynamic](https://img.shields.io/endpoint?url=https://your-api.com/badge.json)](https://example.com)
```

Formato del endpoint JSON:

```
{
  "schemaVersion": 1,
  "label": "PHPStan",
  "message": "level 9",
```

```
  "color": "brightgreen",
  "logoSvg": "<svg>...</svg>"
}
```

Documentación completa: https://shields.io/

Checklist de Configuración

Requisitos Básicos

- [ ] Repositorio público en GitHub
- [ ] Archivo `LICENSE.md` presente
- [ ] Archivo `README.md` con badges
- [ ] `.gitignore` configurado
- [ ] `composer.json` con metadatos completos

Para Paquetes Packagist

- [ ] Publicado en https://packagist.org/
- [ ] `composer.json` con:
  - [ ] `name` correcto
  - [ ] `description` clara
  - [ ] `license` especificada
  - [ ] `require` con versiones de PHP
  - [ ] `autoload` configurado
- [ ] Tags/Releases versionados (semver)

Para GitHub Actions

- [ ] Workflows en `.github/workflows/`
- [ ] Workflow para tests: `run-tests.yml`
- [ ] Workflow para code style: `fix-php-code-style-issues.yml`
- [ ] Workflow para PHPStan: `phpstan.yml`

- [ ] Secrets configurados (si se requieren)

Para Code Coverage

- [ ] Cuenta en Codecov.io vinculada
- [ ] Token añadido a GitHub Secrets: `CODECOV_TOKEN`
- [ ] Workflow configurado para subir coverage
- [ ] Tests generan archivo `coverage.xml`

Para Calidad de Código

- [ ] Code Climate:
  - [ ] Repo conectado
  - [ ] `.codeclimate.yml` configurado
  - [ ] Badge ID obtenido
- [ ] Scrutinizer:
  - [ ] Repo conectado
  - [ ] `.scrutinizer.yml` configurado
- [ ] SonarCloud:
  - [ ] Proyecto configurado
  - [ ] `sonar-project.properties` creado
  - [ ] Project key obtenido

Para Seguridad

- [ ] Snyk vinculado al repositorio
- [ ] Dependabot habilitado
- [ ] `SECURITY.md` presente
- [ ] Vulnerabilities monitorizadas

Para Testing

- [ ] Tests unitarios con PHPUnit o Pest
- [ ] PHPStan configurado (`phpstan.neon`)
- [ ] Code style configurado (Pint, PHP-CS-Fixer)

- [ ] Coverage mínimo definido

Documentación

- [ ] `README.md` completo
- [ ] `CONTRIBUTING.md` (si es open source)
- [ ] `CHANGELOG.md` mantenido
- [ ] PHPDoc en código
- [ ] Wiki o docs externas (opcional)

Referencias y Recursos

Documentación Oficial

- **Shields.io**: https://shields.io/
- **Packagist**: https://packagist.org/about
- **GitHub Actions**: https://docs.github.com/en/actions
- **RepoStatus**: https://www.repostatus.org/

Servicios de Calidad

- **Codecov**: https://docs.codecov.com/
- **Code Climate**: https://docs.codeclimate.com/
- **Scrutinizer**: https://scrutinizer-ci.com/docs/
- **SonarCloud**: https://docs.sonarcloud.io/

Herramientas de Testing

- **PHPStan**: https://phpstan.org/user-guide/getting-started
- **Pest**: https://pestphp.com/docs/
- **PHPUnit**: https://phpunit.de/documentation.html
- **Laravel Pint**: https://laravel.com/docs/11.x/pint

Seguridad

- **Snyk**: https://docs.snyk.io/
- **Dependabot**: https://docs.github.com/en/code-security/dependabot

- **OSSF**: https://github.com/ossf/scorecard

Mantenimiento de este Documento

- **Responsable**: Abkrim
- **Frecuencia de revisión**: Trimestral
- **Última revisión**: 31 Diciembre 2025
- **Próxima revisión**: Marzo 2026

Historial de Cambios

Fecha	Versión	Cambios
2025-12-31	1.0	Creación del documento inicial

Soporte

Para dudas o sugerencias sobre esta guía:

1. Abrir issue en el repositorio del proyecto
2. Consultar con el equipo de desarrollo
3. Revisar documentación oficial de cada servicio

Fin del documento

Cosas de docker

Peleando con docker prefiero tener a mano unos tips

Información de los contenedores docker

Introduccion

Aunque usemos herramientas de desktop, etc, muchos comandos son necesarios, ya no solo por su eficacia, sino porque muchas veces las herramientas no cubren todos los aspectos, y porque otras muchas, los manuales y la información que tenemos a mano, habla de ellos, y no de las herramientas de trabajo.

Contenedores

Muchas, muchismas veces es necesario entrar o conocer datos de los contendores.

Listado

```
docker container ls -a
CONTAINER ID   IMAGE                    COMMAND                  CREATED
STATUS                   PORTS                                                NAMES
b52d46f0eaa1   sail-8.1/app             "start-container"        4 minutes ago   Up 4
minutes             0.0.0.0:80->80/tcp, 0.0.0.0:5173->5173/tcp, 8000/tcp   sitelight-
laravel.test-1
46868801fab5   kibana:8.4.3             "/bin/tini -- /usr/l…"   4 minutes ago   Up 4
minutes             0.0.0.0:5601->5601/tcp                               sitelight-kibana
be49291ac94e   mysql/mysql-server:8.0   "/entrypoint.sh mysq…"   4 minutes ago   Up 4 minutes
(healthy)   0.0.0.0:3306->3306/tcp, 33060-33061/tcp                 sitelight-mysql-1
2cbdb7150093   elasticsearch:8.4.3      "/bin/tini -- /usr/l…"   4 minutes ago   Up 4
minutes             0.0.0.0:9200->9200/tcp, 0.0.0.0:9300->9300/tcp         sitelight-es01
3b17bd32cff8   redis:alpine             "docker-entrypoint.s…"   4 minutes ago   Up 4 minutes
(healthy)   0.0.0.0:6379->6379/tcp                                   sitelight-redis-1
01eecfe2c5e1   mysql:latest             "docker-entrypoint.s…"   2 weeks ago     Exited (0) 2
weeks ago                                                           mysql
```

Ejecutar un comando en un contenedor activo

```
docker exec -it 2cbdb7150093 /usr/share/elasticsearch/bin/elasticsearch-reset-password -a -u elastic
WARNING: Owner of file [/usr/share/elasticsearch/config/users] used to be [root], but now is [elasticsearch]
WARNING: Owner of file [/usr/share/elasticsearch/config/users_roles] used to be [root], but now is [elasticsearch]
This tool will reset the password of the [elastic] user to an autogenerated value.
The password will be printed in the console.
Please confirm that you would like to continue [y/N]y
```

> NOTA: este comando ya no es necesario en un despligue con docker de elasticsearch y kibana. Entra sin seguridad activa, ni login al cluster.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de docker

Wordpress en docker para desarrollo.

Cómo usar docker para desarrollo o gestión de incidentes con Wordpress

A veces es necesario tener un entorno de desarrollo para realizar determinadas acciones, como puede ser una actualización conflictiva, un cambio de versión u otras que no se van a necesitar de un entorno que no sea el de producción.

Docker+, es la elección perfecta, al menos para mí, acostumbrado a trabajar para todo con él, acompañandome todo mi desarrollo en mi portatil, como buen nómada que soy.

En el caso descrito habitual relativo a WordPress, muy, muy, muy antiguos, que usan determinadas versiones de PHP, y que al intentar actualizarlas en producción, puede ser un completo desastre, y aunque pese a tener un sistema basado en **rsync** y **mysqldump** en la comnsola de comandos, la web que tenía que actualizar no tenía esa posibilidad de poder estar perdiendo 45 minutos en producción así qué la dockerize.

> Este tip es valido, para cualquier sitio web que queramos actualizar con un minimo de profesionalidad y seguridad

Dockerización de Wordpress

Habitualmente uso Laravel Sail, pero sin embargo, he preferido hacer el tema de WordPress con una instalación independiente y usando Docker composer.

> Los datos son específicos a mi entorno, pero si no eres de copiar y pegar y tratas de comprender cada paso, podrás trabajar donde quieras y como quieras que se adapte a tus necesidades. Esto supone diertas configuraciones de docker

> que pueden hacer que en otros escenearios o configuraciones no funcione lo que digo, sobre todo si la confioguración especifica afecta al modo de red. Ver
>
> Docker Tips :: Algos sobre redes

> “ Se recomienda leer los tips

Creación de del entorno de trabajo

Directorio de trabajo

```
mkdir ~/Sites/myweb
cd $_
```

Creación del docker-compose.yml

La creación la haremos en un principio ajustándose al escenario de funcionamiento, para luego progresivamente ir subiendo a la versión que queremos, para lidiar con todos los problemas de plugins, templates desactualizados.

Por tanto lo suyo no es copiar, por ejemplo las imágenes de docker que se indican, sino las que necesitamos obteniendo la información de las tags, en el hub de docker.

La imagen de Wordpress deberá ser la adecuada al escenario de nuestro sitio con problema en producción.

Versión de PHP, apache o nginx, fpm,...

Más adelante iremos subiendo a la que queremos.

> “ Uso docker desktop por muchas razones, y lo recomiendo.

Editamos el fichero `editor ~/Sites/myweb/docker-compose.yml`

```
services:
  db:
    # We use a mariadb image which supports both amd64 & arm64 architecture
    # image: mariadb
    # If you really want to use MySQL, uncomment the following line
```

```
    image: mysql:8
    command: '--default-authentication-plugin=mysql_native_password'
    volumes:
      - db_data:/var/lib/mysql
    restart: always
    environment:
      - MYSQL_ROOT_PASSWORD=somewordpress
      - MYSQL_DATABASE=la_misma_que_en_producción
      - MYSQL_USER=el_mismo_que_en_producción
      - MYSQL_PASSWORD=PasSW0rD_que_en_producción
    ports:
      - "3306:3306"
  wordpress:
    image: wordpress:php7.4-fpm-alpine
    volumes:
      - ./wp_data:/var/www/html
    ports:
      - 80:80
    restart: always
    environment:
      - WORDPRESS_DB_HOST=db
      - WORDPRESS_DB_USER=el_mismo_que_en_producción
      - WORDPRESS_DB_PASSWORD=PasSW0rD_que_en_producción
      - WORDPRESS_DB_NAME=la_misma_que_en_producción
volumes:
  db_data:
  wp_data:
```

Explicación

- **Mysql** lo montamos con un volumen, para hacer permanente sus datos.
- La instancia de **WordPress** lo mismo, poniendo el directorio de montaje en el directorio en el que vamos a clonar el sitio, apuntando al directorio `/var/www/html` del volumen.

Copia del contenido de producción

La mejor opción es `rsync` una herramienta **imprescindible** no sólo para los administradores de sistemas, sino para los webmaster o mantenedores de sitios web.

```
$ rsync -avzzz --progress -e "ssh" user_remoto@dominio_remoto.tld:/path/al/sitio/
~/Sites/myweb/wp_data/
receiving file list ...
33791 files to consider
./
license.txt
       19915 100%   18.99MB/s    0:00:00 (xfer#1, to-check=33778/33791)
readme.html
        7402 100%    7.06MB/s    0:00:00 (xfer#2, to-check=33774/33791)
wp-config-sample.php
        3013 100%    2.87MB/s    0:00:00 (xfer#3, to-check=33766/33791)
wp-config.php
        3601 100%    3.43MB/s    0:00:00 (xfer#4, to-check=33765/33791)
...
...
wp-includes/widgets/class-wp-widget-text.php
       21342 100%   20.99kB/s    0:00:00 (xfer#2301, to-check=0/33791)
sent 479944 bytes  received 860743 bytes  536274.80 bytes/sec
total size is 1363857930  speedup is 1017.28
```

Copia o dump de la base de datos de producción

Podemos usar nuestra herramienta preferida, como TablePlus, Navicat, phpMyAdmin... o el propio shell.

El caso es hacer una copia completa.

Me gusta usar el shell.

Como no tengo acceso mysql remoto, y no me apetece montar un túnel, hago el backup en remoto y luego lo bajo con `rsync`

Servidor remoto

```
mysqldump --no-tablespaces --opt -u <wp_user> -p  <database>  > ~/<database>.sql
Enter password:
```

Servidor local

Ahora traemos el fichero de dump

```
rsync -avzz --progress -e "ssh -p 2244" user_remoto@remoto.tld:/home/myuser/<database>.sql .
receiving file list ...
1 file to consider
<database>.sql
   130317359 100%    7.80MB/s    0:00:15 (xfer#1, to-check=0/1)
sent 25352 bytes  received 35045636 bytes  2004056.46 bytes/sec
total size is 130317359  speedup is 3.72
```

> Atención al `.` final del comando que quiere decir `cópiame aquí`

Restaurar la copia

Obtener información del container

```
docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS
PORTS                                NAMES
9b780079bbeb   wordpress:latest   "docker-entrypoint.s…"   40 minutes ago   Up 40 minutes
0.0.0.0:80->80/tcp                   myweb-wordpress-1
c591fdcae17a   mysql:8            "docker-entrypoint.s…"   40 minutes ago   Up 40 minutes
0.0.0.0:3306->3306/tcp, 33060/tcp   myweb-db-1
```

Volcar el backup

Usaremos el **id** del contenedor de mysql `c591fdcae17a` para ejecutar un comando en él.

```
cat <database>.sql | docker exec -i c591fdcae17a /usr/bin/mysql -u root --
password=somewordpress <database>
mysql: [Warning] Using a password on the command line interface can be insecure.
```

> En mi caso por vaquería y comodidad, uso TablePlus para muchas cosas, y tiene un importador. Pero siempre hay una vez para fallar, y esta vez, fue al importar una bd con 45Mb, (he importado en table plus bd de 6 GB sin problemas). Se terminaba siempre en una de las tablas. Fue hacerlo en el el shell, y funcionó sin problemas.

Levantamos la imagen

```
docker compose up -d
```

docker compose

Para probar que esta todo correcto, solo tendríamos que ir a nuestro navegador y solicitar localhost

Devolver a su sitio.

Tras actualziar y ajustar todo a las últimas versiones tanto de wordpress, como de PHP, Mysql o MariaDb, deberemos subirlo a producción.

Para devolver a su sitio la actualización completa es sencillo. Usamos la misma técnica de `rsync` + `mysqldump` pero al revés.

Mysqldump

> Esta vez usaremos el **name** del container en lugar del **ID**

```
docker exec -i myweb-db-1 /usr/bin/mysqldump --opt -u root --password=somewordpress --databases <database> --skip-comments > dump-updated.sql
```

Subir los cambios a producción

```
 ~/Sites/myweb $ rsync -avzz --progress --delete -e "ssh -p 2244" ~/Sites/myweb/wp_data/ user_remoto@dominio_remoto.tld:/path/al/sitio/
~/Sites/myweb $  rsync -avzz --progress --delete -e "ssh -p 2244" ~/Sites/myweb/dump-updated.sql user_remoto@dominio_remoto.tld:/path/al/user/
```

Restaurar la base de datos en el servidor de producción

```
Mysql -u <user_de_mysql> -p <bd_produccion>  < ~/dump-updated.sql
```

Tips.

- Editar `/etc/hosts` para que apunte a a local (127.0.0.1) y evitar líos de redirecciones. En el caso de MacOsx, sería en `/private/etc/hosts`

```
127.0.0.1  myweb.com
```

- Usar un navegador permisivo o configurado para no usar https (safari, opera...). Si tenemos algun plugin que fuerce la redireción `http` a `https` deberías desactivarlo.
- Usar una página `info.php` para tener las cosas claras. Simplemente debe tener `<?php phpinfo();` y ponerla en el `public` de nuestra web.
- Restaurar el backup (mejor desde shell como siempre)
- Los datos de creación de mysql, user y password deben ser los mismos en el par, producción-desarrollo.
- El parámetro del fichero `wp-config.php` `define('DB_HOST', 'localhost');` debe estar definido a 'localhost'.
- **editor** es el alias de mi editor preferido.
- `$` indica que estamos en una cuenta de usuario y su path y `#` es root
- `<variable>` indica que debe ser sustituido por los valores apropiados incluyendo en el reemplazo los `<>`
- Atención a la barra `/` al final de los parámetros de `rsync` porque no es lo mismo con ella que sin ella.

A trabajar con el tema.

Tips globales

- Una vez que ya hemos conseguido la actualización menor necesaria para poder pasar a una versión más alta de php sin problemas, simplemente debemos editar el fichero `docker-compose.yml` y solicitar la nueva versión de php, y reconstruir el volumen de Wordpress. (Borrar container y volumen wordpress.
- Siempre debemos trabajar inicialmente con las mismas versiones y software que como está en producción. Hay que recordar que existe, versiones de Mysql, MariaDb (no son iguales no... ojo a ese datos que más de un quebradero de cabeza os podéis llevar un día), versiones de PHP, con FPM, mod, con Nginx, con Apache, etc.
- Si no te sientes cómodo con el shell, al menos elige un hosting que tenga una herramienta de bacjup como Jetbackup que te permite crear snapshots antes de liarte con cambios.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún

obligación ni responsabilidad por parte de Castris Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Proyectos con path public, public_html, app en Docker con Apache+PHP-FPM

Introducción

Existen desarrollos en software web, que siguen una pauta moderna y más segura, consistente en exponer solamente al público el fichero index.php dejando en el `doc_root` de **Apache** o **Nginx** el directorio que lo contiene, y no todos el desarrollo. Ejemplos clásicos son, Laravel, Symfony, Slim, y otros muchos. aunque si no eres amigo de los marcos de trabajo (framework), bueno es que también uses esta formula de trabajo y no meter todo en el directorio del `doc_root` del servidor Web.

En mi último trabajo he tenido que lidiar con una aplicación obsoleta, en PHP 5.6 desarrollada con el marco Zend Framework, que estaba en un cPanel pero con las estructura correcta por debajo del `public_html`. (Lastima que encima puso un blog con Wordpress, y se le olvido actualizarlo)

Esta guía te mostrara el camino de la creación de los contenedores necesarios para Apache, PHP-Fpm y MySQL para ejecutar tu desarrollo, usando **Docker compose**

Docker Apache, PHP FPM setup with docker compose y MySQL

Cuando uno llega aquí, lo único que queda es dockerizar nuestro area de trabajo, ya que desarrollar en el servidor está... **prohibido**

El proceso esta indicado para un escenario concreto, pero con los ajustes necesarios, puedes elegir la versión de PHP, la de MySQL/MariaDb/Percona que necesites (siempre trabaja con el mismo escenario que en producción).

Requisitos

Necesitamos tener instalado en nuestro equipo Docker.

Generaremos la siguiente estructura en nuestro proyecto

```
proyecto
├── Dockerfile
├── docker-compose.yml
├── apache/
| | ├── apache.vhost.conf
| | └── Dockerfile
├── public/
|    └── index.php
├── vendor/
├── ...
```

Algo así quedaría

- **Dockerfile** global que contiene la configuración necesaria para crear y preparar el escenario para un PHP con PHP-FPM y sus extensiones necesarias en las imágenes necesarias.
- **docker-compose.yml** que crear los contenedores necesarios y los unirá.
- **apache/apache.vhost.conf** que contendrá la información necesaria para que Apache sea el servidor web de nuestro proyecto.
- **apache/Dockerfile**
- **public** que será el directorio root de nuestro proyecto, expuesto a internet por el par Apache/PHP-FPM

Crear la imagen PHP FPM

Te damos las instrucciones necesarias a docker para construir la imagen. El ejemplo es el básico que me permite usar y trabajar en esta dockerización, con el proyecto Zend en un PHP 5.6. Si necesitas más extensiones, o algún cambio necesitaras un trabajo extra para añadirlo.

Consulta siempre en Hub Docker las posibilidad que tienes, y si quieres un consejo, usa siempre imágenes oficiales. Te ahorraras muchos disgustos.

> Los comentarios indican modificación sobre la imagen oficial. Si usas una no oficial, o versiones oficiales basadas en otro so, como alpine, etc. los paths indicados pueden sufrir variaciones

```
FROM php:5.6-fpm

RUN docker-php-ext-install mysqli pdo pdo_mysql
```

```
## Esta linea la puedes comentar si tu proyecto tiene un estructura plana, como puede ser un Wordpress. Donde todo está en el mismo nivel del doc_root
RUN echo 'php_admin_value[doc_root] = /var/www/html/public' >> /usr/local/etc/php-fpm.d/www.conf
```

- Docker hará un pull **php:5.6-fpm**
- Si quieres o necesitas instalar una extensión adicional con Docker usa el commando **docker-php-ext-install** dentro de doc usando **RUN**

Crear Apache

Necesitamos crear la imagen con el servidor web Apache, atendiendo a que usaremos PHP-FPM como manejador de PHP. Así que usaremos un fichero de configuración del host virtual, y usaremos **apache/apache.vhost.conf** para este propósito.

```
# La configuracion esta adaptada al escenario indicado public/
# Si no es así deberá modificarse al típico /var/www/html/ allí donde este usándose public/
# Establezca el nombre del servidor en localhost
ServerName localhost

# Configure un VirtualHost para manejar solicitudes en el puerto 80
<VirtualHost *:80>
    # Solicitudes de proxy PHP para portar el contenedor PHP-FPM 9000
    ProxyPassMatch ^/(.*\.php(/.*)?)$ fcgi://php-fpm:9000/var/www/html/public/$1

    # Set the DocumentRoot for the virtual host
    DocumentRoot /var/www/html/public

    # Directory configuration for the DocumentRoot
    <Directory /var/www/html/public>
        DirectoryIndex index.php
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    # Definir los destinos CustomLog y ErrorLog
    CustomLog /proc/self/fd/1 common
    ErrorLog /proc/self/fd/2
</VirtualHost>
```

- La aplicación se ejecutara en modo localhost y responderá a las peticiones que se hagan la puerto 80.
- Apache reenviará las peticiones php al container PHP-FPM, cuyo nombre es 'php-fpm'. Este nombre será usado en el `docker-compose.yml` del proyecto.
- `/var/www/html/public/` será el directorio por defecto de tu sitio virtual de Apache. Por tanto sería el index o fichero que este allí.
- Si deseas depurar tu aplicación, cree destinos CustomLog y ErrorLog para administrar los registros de la aplicación y los de Docker.

Ahora construiremos nuestro **Dockerfile** para construir la imagen de nuestro servidor web Apache

```
# httpd base image
FROM httpd:2.4

# Copy the Apache virtual host configuration file to the container
COPY ./apache/apache.vhost.conf /usr/local/apache2/conf/extra/apache.vhost.conf

# Activar módulos de Apache para garantizar una funcionalidad adecuada.
RUN sed -i \
    # Descomente la configuración de mod_rewrite para habilitar el control de reescritura
contenido.
    -e '/#LoadModule rewrite_module/s/^#//g' \
    # Descomente la configuración de mod_rewrite para habilitar el control de caducidad del
contenido.
    -e '/#LoadModule expires_module/s/^#//g' \
    # Descomente la configuración de mod_deflate para habilitar la compreión
    -e '/#LoadModule deflate_module/s/^#//g' \
    # Descomente la configuración del poryx para habilitar su uso.
    -e '/#LoadModule proxy_module/s/^#//g' \
    # Descomente la configuración de mod_proxy_fcgi para habilitar el módulo proxy FastCGI
    -e '/#LoadModule proxy_fcgi_module/s/^#//g' \
    /usr/local/apache2/conf/httpd.conf

# Incluir el archivo de configuración del host virtual en la configuración principal de
Apache.
RUN echo "Include /usr/local/apache2/conf/extra/apache.vhost.conf" >>
/usr/local/apache2/conf/httpd.conf
```

Crear el container

Bien es hora de armarlo todo.

> En mi caso, no uso PHPMyAdmin así que no instalaremos nada adicional. Nunca lo he usado porque nunca me ha parecido correcto tenerlo instalado. En su lugar uso un programa de consola, y me conecto via tunnel SSH. Pero os dejo comentada la opción. Gracias TablePlus

```
version: '3.9'
services:
  php-fpm:
    build:
      context: .
      dockerfile: Dockerfile
    volumes:
      - .:/var/www/html
    depends_on:
      - mysql-db

  apache-httpd:
    build:
      context: .
      dockerfile: ./apache/Dockerfile
    volumes:
      - .:/var/www/html
    ports:
      - "8080:80"
    depends_on:
      - php-fpm
      - mysql-db
  mysql-db:
    image: mariadb:10.1
    environment:
      MYSQL_ROOT_PASSWORD: password
      MYSQL_DATABASE: base_de_datos
      MYSQL_USER: user_datos
      MYSQL_PASSWORD: contraseña
    ports:
      - "3336:3306"
```

```
    volumes:
      - mysql-data:/var/lib/mysql
# Mac not working
#  phpmyadmin:
#    image: phpmyadmin/phpmyadmin:latest
#    links:
#      - mysql-db
#    ports:
#      - "8081:80"
#    environment:
#      PMA_HOST: mysql-db
#      MYSQL_ROOT_PASSWORD: password
volumes:
  mysql-data:
```

- `.:/var/www/html` esto indica que crearemos un **volumen** que montara en la raíz de nuestro proyecto `.` en `/var/www/html/` en el container.
- El puerto expuesto para acceder será el 8080
- Crearemos un container para **MySQL** que tendrá persistencia de datos, en un volumen **mysql-data** que permitirá tener persistencia de datos.
- Mi consejo es que uses los mismo datos para el usuario, base de datos, y contraseña relativos a tu proyecto, que los que usas en producción, por si realizas sincronizaciones. El root no lo toques.
- El puerto expuesto es el 3336 ya que tengo en local, una instalación **Herd** que es la que uso habitualmente y no me apetece apagarla.

Testing

Para evitar perdidas de tiempo, es bueno hacer dos comprobaciones.

- Que el container esta preparado para trabajar con PHP en el directorio `public`
- Que la conexión con Mysql funciona.

info.php

Crea un fichero `public/info.php`

```
<?php phpinfo();
```

index.php

Si ya tienes uno renombrarlo, para más tarde y crea un nuevo en `public/index.php`

> También pues añadir al código al principio del index.php y salir con un `exit();`

```
<?php
$host = 'mysql-db';
$user = 'root';
$pass = 'password';
$db = 'mysql';


$conn = new mysqli($host, $user, $pass, $db);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

echo "Connected to MySQL successfully";

$conn->close();
?>
```

Construir los contenedores

Bueno, ya esta todo.

```
docker-compose up --build -d
docker-compose up --build -d
[+] Building 2.7s (16/16)
FINISHED

docker:desktop-linux
 => [php-fpm internal] load build definition from
Dockerfile
                                                                                0.0s
 => => transferring dockerfile:
204B

        0.0s
 => [php-fpm internal] load metadata for docker.io/library/php:5.6-
```

```
fpm
                                                                    1.6s
 => [php-fpm internal] load
.dockerignore

            0.0s
 => => transferring context:
2B

           0.0s
 => [php-fpm 1/3] FROM docker.io/library/php:5.6-
fpm@sha256:4f070f1b7b93cc5ab364839b79a5b26f38d5f89461f7bc0cd4bab4a3ad7d67d7
                                                                                0.0s
 => CACHED [php-fpm 2/3] RUN docker-php-ext-install mysqli pdo
pdo_mysql
                                                                  0.0s
 => CACHED [php-fpm 3/3] RUN echo 'php_admin_value[doc_root] = /var/www/html/public' >>
/usr/local/etc/php-
fpm.d/www.conf
                           0.0s
 => [php-fpm] exporting to
image

             0.0s
 => => exporting
layers

                       0.0s
 => => writing image
sha256:552b673bf2989a93258340ee86ae7c31477c946f95aa73a63af49cede70235a6

                   0.0s
 => => naming to docker.io/library/staybarcelona-php-
fpm
                                                                             0.0s
 => [apache-httpd internal] load build definition from
Dockerfile
                                                                           0.0s
 => => transferring dockerfile:
```

```
1.17kB

        0.0s
 => [apache-httpd internal] load metadata for
docker.io/library/httpd:2.4
                                                                                0.9s
 => [apache-httpd internal] load
.dockerignore

       0.0s
 => => transferring context:
2B

           0.0s
 => [apache-httpd internal] load build
context

 0.0s
 => => transferring context:
72B

           0.0s
 => [apache-httpd 1/4] FROM
docker.io/library/httpd:2.4@sha256:104f07de17ee186c8f37b9f561e04fbfe4cf080d78c6e5f3802fd08fd11
8c3da
            0.0s
 => CACHED [apache-httpd 2/4] COPY ./apache/apache.vhost.conf
/usr/local/apache2/conf/extra/apache.vhost.conf
                                                                          0.0s
 => CACHED [apache-httpd 3/4] RUN sed -i             -e '/#LoadModule
rewrite_module/s/^#//g'         -e '/#LoadModule expires_module/s/^#//g'     -e '/#LoadModule
deflate_module/s/^#//g'     -e '/#LoadModule proxy_module/s/^#  0.0s
 => CACHED [apache-httpd 4/4] RUN echo "Include
/usr/local/apache2/conf/extra/apache.vhost.conf" >>
/usr/local/apache2/conf/httpd.conf
                                  0.0s
 => [apache-httpd] exporting to
image
```

```
        0.0s
 => => exporting
layers

                       0.0s
 => => writing image
sha256:7428ace356414dd78d15c185ffa4137e7e3806018ce31eac61be778c773738f5

                  0.0s
 => => naming to docker.io/library/staybarcelona-apache-
httpd
                                                                          0.0s
[+] Running 3/3
 ✔ Container staybarcelona-mysql-db-1
Started

0.0s
 ✔ Container staybarcelona-php-fpm-1
Started

0.6s
 ✔ Container staybarcelona-apache-httpd-1  Started
```

Comprobaciones

En el navegador probaremos la información del `phpinfo()` en la ruta `localhost:8080`

Info

Ahora la conexión **mySQL**

Info

Conclusión

Bien, esto es un esbozo de como podemos adecuarnos con Docker Composer, en un escenario como el descrito además de aprender ciertas metodologías con Docker.

Esto es muy básico y dependerá de la instalación y trabajos que tienes en tu máquina. Hay muchos caminos y formas de trabajar con Docker. Esta es una rápida para un escenario particular.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Cosas de Docker

Ay Docker. Bendito docker, y sufrido como el solo.

Instalar Postgresql con Postgis

Introducción

Bueno, esta es la de siempre. Yo prefiero la eficacia de Elasticsearch, frente a el uso de un DBRM como Postgresql del que no se cansan de dar alabanzas, pero que no me convence para temas de este tipo. Sin embrago, hay un tema que es especial **GIS** y si uno no quiere morir con él, por Postgresql y PostGIS debe pasar.

Para no moriri en el inteneto, en desarrollo probe el proceso de instalar en mi MacBook M1. Tremendo. Una perdida de tiempo, para luego perder más tiempo en producción.

Asi que aquí, mis dos céntimos.

Postgresql + Postgis Docker

Ya sabe que si quieres una versión especifica (ojo a las versiones de Postgresql que no son como las de MySQL o MariaDB, ya que los cambiso mayores, son eso, **mayores**) puedes ir a su página en el Hub de Docker y forzar el comando a usar la que deseas.

```
docker run --name my_postgis_container -e POSTGRES_PASSWORD=MY_PASSWORD -d -p 5432:5432
postgis/postgis
```

Esto ya sabes que puedes adapartlo a tu gusto. En mi caso, quiero que sea un postgresql accesible vía 127.0.0.1, asi que no necesito asociarlo a network como en otros proyectos.

TablePlus - Configuración acceso a Postgresql - Docker

TablePlus - Vista templates_postgis

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Batiburrillo del programador

Cosas que me sirven y que sirven

Creación de diagramas DBML

Introduccción

Me gusta tener un cuadro de lo que tengo en Mysql. A los largo de los años, ha dio cambiando el formato, las utilidades. Lejos queda el insoportable Workbench de mysql.

Ahora puedo usar TablePlus, pero a la postre muchas veces eso se me queda corto para lo que me gusta.

Asi que ahor aprefioer mantener la logica guardada en formato DBML, lo cual me permite una visión más adecuada, y real de lo que hay y lo que no hay en la app, y que uso a menudo con DBDiagram

db2dbml

db2dbml es una puntenteherramienta para generar un fichero DBML contectando a un servidor o motores de basos de datos.

Un ejemplo abajo de como generar en local el diagrama de mi proyecto.

```
❯ dbdocs db2dbml mysql
'mysql://root:MYPASSWORD@localhost:3306/lowino?socketPath=/tmp/mysql.sock' -o
notas/DBML/lowino.dbml
✔ Connecting to database... done.
✔ Generating DBML... done
✔ Wrote to notas/DBML/lowino.dbml
```

Tras eso tenemos el codigo en el fichero

```
Table "advert_properties" {
  "id" "bigint unsigned" [pk, not null, increment]
  "advert_id" binary(26) [not null]
  "property_id" "bigint unsigned" [not null]
  "value" text
```

```
  "created_at" timestamp
  "updated_at" timestamp
}

Table "adverts" {
  "id" binary(26) [unique, not null]
  "user_id" binary(26) [not null]
  "category_code" "mediumint unsigned" [not null, default: 1, note: 'without classification']
  "title" varchar(80) [not null]
  "slug" varchar(80) [unique, not null, note: 'Integrate -YYMMDDNN']
  "advert" text [not null]
  "price" "bigint unsigned"
  "currency_id" "tinyint unsigned" [not null, default: 1]
  "details" json
  "published_at" timestamp
  "expire_at" timestamp
  "created_at" timestamp
  "updated_at" timestamp
  "deleted_at" timestamp

  Indexes {
    published_at [type: btree, name: "adverts_published_at_index"]
  }
}
...
```

Y con ello ya podemos hacwer y deshacer como queramos, para crear nuestro digrama.

> Cierto que tambien TablePlus es casi más eficaz en términos visuales, pero el dbml sirve para nuchas más cosas.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Borrar archivos o directorios que comienzan por letra (A–Z/a–z)

Objetivo

Eliminar **todos los archivos o carpetas cuyo nombre comience por una letra**, sin distinguir mayúsculas/minúsculas.
Ideal para limpiezas de entorno sin afectar archivos ocultos ni aquellos que comiencen por números o símbolos.

?? En macOS (Zsh)

Zsh no usa `shopt`, pero tiene su propia sintaxis de globbing avanzada.

1. Ver qué se va a borrar (modo seguro)

```
setopt extended_glob
ls -d (#i)[[:alpha:]]*
```

2. Eliminar los archivos y carpetas

```
setopt extended_glob
rm -Rf (#i)[[:alpha:]]*
```

□□ Explicación técnica

- *setopt* extended_glob activa los patrones avanzados de Zsh.
- *(#i)* activa insensibilidad a mayúsculas para todo el patrón.
- *[[:alpha:]]** coincide con cualquier nombre que empiece por letra (A–Z o a–z), de forma portable.

> □□ Ojo: patrones como (#i)[a-z]* no siempre funcionan como se espera, porque [a-z] sigue limitado al rango ASCII explícito. Usar **[[:alpha:]]** es más robusto.

? En Linux (Bash)

Bash usa `shopt` para habilitar coincidencias insensibles a mayúsculas.

1. Ver qué se va a borrar

```
shopt -s nocaseglob
ls -d [a-zA-Z]*
shopt -u nocaseglob
```

2. Eliminar los archivos y carpetas

```
shopt -s nocaseglob
rm -Rf [a-zA-Z]*
shopt -u nocaseglob
```

? Explicación técnica

- *nocaseglob* hace que Bash trate los patrones como insensibles a mayúsculas.
- *[a-zA-Z]** coincide con nombres que comienzan por cualquier letra.

> ⚠ Advertencias No borra archivos ocultos (*.env, .git,* etc.) ni nombres que empiecen con números (2023-img.png).
>
> Usa ls antes de rm -Rf para confirmar qué se eliminará.
>
> Si necesitas excluir ciertos nombres, puedes añadir filtros con grep -v, find, o usar listas de exclusión.

? Alternativas útiles

Mover en lugar de borrar:

```
mkdir -p backup_letters
mv [a-zA-Z]* backup_letters/
```

Ver el tamaño de los elementos que serán eliminados:

```
du -sh [a-zA-Z]*
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Guía de Semantic Versioning y Git Flow

Semantic Versioning (SemVer)

Formato: `MAJOR.MINOR.PATCH` (ejemplo: `1.2.3`)

¿Cuándo incrementar cada número?

MAJOR (1.x.x ? 2.x.x)

Cambios que ROMPEN compatibilidad hacia atrás (Breaking Changes)

Ejemplos:

- Cambiar la firma de un método público
- Eliminar un método o clase pública
- Cambiar el comportamiento de un método de forma incompatible
- Renombrar propiedades públicas

```
// v1.x.x
public function base100Attributes(): array


// v2.0.0 - BREAKING CHANGE
public function base100Attributes(): Collection  // Cambió el return type
```

Impacto: Los usuarios DEBEN revisar su código antes de actualizar.

MINOR (x.1.x ? x.2.x)

Nuevas características que SÍ son compatibles hacia atrás

Ejemplos:

- Añadir un nuevo método público

- Añadir un nuevo trait
- Añadir parámetros opcionales a métodos existentes
- Añadir nuevas opciones de configuración

```
// v1.1.0
class Base100 implements CastsAttributes
{
    // Métodos existentes...

    // NUEVO método añadido
    public function withPrecision(int $decimals): self
    {
        // ...
    }
}
```

Impacto: Los usuarios pueden actualizar sin cambios en su código.

PATCH (x.x.1 ? x.x.2)

Correcciones de bugs que NO cambian funcionalidad

Ejemplos:

- Corregir un bug
- Mejorar rendimiento sin cambiar API
- Actualizar documentación
- Refactoring interno sin cambios en API pública
- Corregir tests

```
// v1.1.0 - Bug: redondeo incorrecto
return (int) $value * 100;  // Error

// v1.1.1 - Patch: corregir redondeo
return (int) round($value * 100);  // Correcto
```

Impacto: Actualización segura, solo mejoras y correcciones.

Estrategia de Branches

Branches Principales

1. `main` (Branch Principal)

- **Propósito**: Código estable y listo para producción
- **Protección**: Protected (no push directo)
- **Contiene**: Solo código que ha pasado tests y revisión
- **Tags**: Todas las releases se tagean desde aquí

2. `develop` (Branch de Desarrollo) - OPCIONAL

- **Propósito**: Integración de features antes de release
- **Uso**: Si tienes múltiples features en paralelo
- **Para este proyecto**: NO necesario (proyecto pequeño)

Branches de Trabajo

Feature Branches: `feature/*`

Para nuevas características

```
feature/add-precision-option
feature/base1000-support
feature/custom-rounding
```

Ejemplo de uso:

```
# Crear feature branch desde main
git checkout main
git pull origin main
git checkout -b feature/add-precision-option

# Trabajar en la feature...
git add .
git commit -m "feat: add precision option to Base100 cast"

# Cuando esté listo
```

```
git push origin feature/add-precision-option
# Crear Pull Request en GitHub
```

Bugfix Branches: `fix/*`

Para correcciones de bugs

```
fix/rounding-precision
fix/null-handling
fix/trait-initialization
```

Ejemplo de uso:

```
# Crear bugfix branch
git checkout main
git checkout -b fix/rounding-precision

# Corregir el bug
git add .
git commit -m "fix: correct rounding precision in Base100 cast"

# Push y PR
git push origin fix/rounding-precision
```

Hotfix Branches: `hotfix/*`

Para bugs CRÍTICOS en producción

```
hotfix/security-vulnerability
hotfix/data-corruption
```

Diferencia con `fix/*`:

- Hotfix: Bug crítico que necesita release inmediata
- Fix: Bug normal que puede esperar al próximo release

Release Branches: `release/*` - OPCIONAL

Para preparar una release

```
release/1.2.0
release/2.0.0
```

Uso: Solo si necesitas "congelar" features antes de release.

Workflow de Desarrollo

Opción 1: GitHub Flow (RECOMENDADO para este proyecto)

```
main (protegido)
  ↑
  └── feature/nueva-caracteristica (trabajo aquí)
  └── fix/bug-menor (trabajo aquí)
```

Ventajas:

- Simple y directo
- Ideal para proyectos pequeños-medianos
- Deployments/releases frecuentes

Flujo:

1. Crear branch desde `main`
2. Desarrollar la feature/fix
3. Abrir Pull Request
4. Code review + Tests automáticos (GitHub Actions)
5. Merge a `main`
6. Tag y release

Opción 2: Git Flow (Para proyectos grandes)

```
main (producción)
  ↑
develop (desarrollo)
  ↑
  ├── feature/feature-1
  ├── feature/feature-2
  └── release/1.2.0
```

Ventajas:

- Mejor para equipos grandes
- Releases planificadas
- Múltiples versiones en paralelo

Desventaja:

- Más complejo
- Overkill para proyectos pequeños

Proceso de Release

Release MINOR (nueva feature - 1.1.0 ? 1.2.0)

```
# 1. Asegurarte que main está actualizado
git checkout main
git pull origin main

# 2. Actualizar CHANGELOG.md
## 1.2.0 - 2025-10-15

### Added

- New `withPrecision()` method for custom decimal precision
- Support for negative values in HasBase100 trait

### Fixed

- Rounding precision issue in edge cases
```

```
# 3. Commit el changelog
git add CHANGELOG.md
git commit -m "docs: update changelog for v1.2.0"

# 4. Crear tag
git tag -a v1.2.0 -m "Release v1.2.0 - Add precision support"

# 5. Push tag y código
git push origin main
git push origin v1.2.0

# 6. Crear GitHub Release
gh release create v1.2.0 \
  --title "v1.2.0 - Precision Support" \
  --notes "See CHANGELOG.md for details"
```

Release PATCH (bugfix - 1.1.0 ? 1.1.1)

```
# 1. Checkout main
git checkout main
git pull origin main

# 2. Actualizar CHANGELOG.md
## 1.1.1 - 2025-10-12

### Fixed

- Correct rounding precision when handling values > 1000

# 3. Commit y tag
git add CHANGELOG.md
git commit -m "docs: update changelog for v1.1.1"
git tag -a v1.1.1 -m "Release v1.1.1 - Fix rounding precision"

# 4. Push
git push origin main
git push origin v1.1.1
```

```
# 5. Release
gh release create v1.1.1 \
  --title "v1.1.1 - Bugfix Release" \
  --notes "Fix rounding precision issue"
```

Release MAJOR (breaking changes - 1.x.x ? 2.0.0)

```
# 1. Crear rama para v2
git checkout main
git checkout -b release/2.0.0

# 2. Hacer cambios breaking
# ... código ...

# 3. Actualizar CHANGELOG.md con sección BREAKING CHANGES
```

Ejemplo de CHANGELOG para v2.0.0:

```
## 2.0.0 - 2025-11-01

### BREAKING CHANGES

- `base100Attributes()` now returns `Collection` instead of `array`
- Minimum PHP version raised to 8.4
- Removed deprecated `base100()` method

### Migration Guide

**Before (v1.x):**
```php
protected function base100Attributes(): array
{
    return ['price', 'cost'];
}
```
```

After (v2.0):

```
protected function base100Attributes(): Collection
{
    return collect(['price', 'cost']);
}
```

Added

- New `Base100Collection` class
- Support for custom transformers

```
Continuación del proceso:

```bash
# 4. Mergear a main
git checkout main
git merge release/2.0.0

# 5. Tag y release
git tag -a v2.0.0 -m "Release v2.0.0 - Major overhaul"
git push origin main
git push origin v2.0.0

# 6. GitHub Release con ADVERTENCIA
gh release create v2.0.0 \
  --title "v2.0.0 - BREAKING CHANGES" \
  --notes "See CHANGELOG.md for migration guide"
```

Ejemplos Prácticos

Ejemplo 1: Añadir nueva feature (MINOR)

Escenario: Quieres añadir soporte para Base1000
```

```
# 1. Crear feature branch
git checkout main
git checkout -b feature/base1000-support

# 2. Desarrollar la feature
# - Crear src/Casts/Base1000.php
# - Añadir tests
# - Actualizar README

# 3. Commits durante desarrollo
git add .
git commit -m "feat: add Base1000 cast class"
git add .
git commit -m "test: add Base1000 tests"
git add .
git commit -m "docs: document Base1000 usage"

# 4. Push y crear PR
git push origin feature/base1000-support
gh pr create --title "Add Base1000 support" --body "Adds support for base-1000 conversions"

# 5. Después de aprobación y merge
git checkout main
git pull origin main

# 6. Release como 1.2.0 (MINOR - nueva feature)
git tag -a v1.2.0 -m "Release v1.2.0 - Add Base1000 support"
git push origin v1.2.0
gh release create v1.2.0
```

Ejemplo 2: Corregir bug (PATCH)

Escenario: Hay un bug en el redondeo

```
# 1. Crear fix branch
git checkout main
git checkout -b fix/rounding-issue
```

```
# 2. Corregir el bug
# - Editar src/Casts/Base100.php
# - Añadir test que reproduce el bug
# - Verificar que el test pasa

# 3. Commit
git add .
git commit -m "fix: correct rounding for values > 10000"

# 4. Push y PR
git push origin fix/rounding-issue
gh pr create --title "Fix rounding issue" --body "Fixes #42"

# 5. Después del merge
git checkout main
git pull origin main

# 6. Release como 1.1.1 (PATCH - bugfix)
git tag -a v1.1.1 -m "Release v1.1.1 - Fix rounding issue"
git push origin v1.1.1
gh release create v1.1.1
```

Ejemplo 3: Hotfix crítico (PATCH urgente)

Escenario: Descubriste un bug que causa pérdida de datos

```
# 1. Crear hotfix branch DESDE main
git checkout main
git checkout -b hotfix/data-loss-prevention

# 2. Corregir RÁPIDAMENTE
# - Solo el fix necesario, nada más
# - Test mínimo que demuestre el fix

# 3. Commit
git add .
git commit -m "fix: prevent data loss in null handling (critical)"
```

```
# 4. Merge DIRECTO a main (sin PR si es muy urgente)
git checkout main
git merge hotfix/data-loss-prevention

# 5. Release INMEDIATA
git tag -a v1.1.2 -m "Release v1.1.2 - Critical hotfix"
git push origin main
git push origin v1.1.2
gh release create v1.1.2 --title "v1.1.2 - Critical Hotfix"

# 6. Notificar usuarios en GitHub/Packagist
```

Comandos Git Útiles

Gestión de Branches

```
# Ver todas las branches
git branch -a

# Eliminar branch local
git branch -d feature/mi-feature

# Eliminar branch remoto
git push origin --delete feature/mi-feature

# Actualizar main desde remoto
git checkout main && git pull origin main

# Crear branch desde un commit específico
git checkout -b fix/bug abc1234
```

Gestión de Tags

```
# Listar todos los tags
git tag

# Ver detalles de un tag
git show v1.0.0

# Eliminar tag local
git tag -d v1.0.0

# Eliminar tag remoto
git push origin --delete v1.0.0

# Crear tag desde un commit antiguo
git tag -a v1.0.1 abc1234 -m "Release v1.0.1"
```

Revertir Cambios

```
# Revertir un commit (crea nuevo commit)
git revert abc1234

# Revertir último commit (antes de push)
git reset --soft HEAD~1

# Descartar cambios locales
git checkout -- archivo.php
```

Checklist de Release

Antes de Release

- Todos los tests pasan (`composer test`)
- PHPStan sin errores (`composer phpstan`)
- Código formateado (`composer format`)
- CHANGELOG.md actualizado

- README.md actualizado (si hay cambios en uso)
- Versión en composer.json coincide? (NO - Packagist lo maneja)
- Pull Request revisado y aprobado

Durante Release

- Main actualizado (`git pull origin main`)
- Tag creado con mensaje descriptivo
- Tag pusheado a GitHub
- GitHub Release creada con notas
- Packagist se actualizó automáticamente (webhook)

Después de Release

- Verificar que aparece en Packagist
- Badges del README actualizados
- Anunciar en redes/comunidad (si es relevante)
- Crear issues/milestones para próxima versión

Recomendaciones Específicas

Estrategia Recomendada

Para proyecto pequeño:

1. Usar GitHub Flow (simple)
2. Main siempre deployable
3. Feature branches para TODO
4. Pull Requests siempre (aunque seas solo tú - para CI)
5. Tags para cada release

Naming Conventions

```
# Features
feature/add-base1000
feature/custom-precision
feature/collection-support

# Fixes
```

```
fix/rounding-precision
fix/null-handling
fix/trait-initialization

# Hotfixes
hotfix/security-vulnerability
hotfix/data-corruption

# Docs
docs/update-readme
docs/add-examples
docs/api-documentation

# Chores
chore/update-dependencies
chore/ci-improvements
```

Commits Convencionales

```
feat: add new feature
fix: bug correction
docs: documentation only
style: formatting, no code change
refactor: code restructure
test: add/update tests
chore: maintenance tasks
perf: performance improvements
ci: CI/CD changes

# Ejemplos:
git commit -m "feat: add withPrecision() method"
git commit -m "fix: correct rounding for negative values"
git commit -m "docs: add usage examples to README"
```

Recursos Adicionales

- **Semantic Versioning**: https://semver.org/
- **Git Flow**: https://nvie.com/posts/a-successful-git-branching-model/
- **GitHub Flow**: https://guides.github.com/introduction/flow/
- **Conventional Commits**: https://www.conventionalcommits.org/

Preguntas Frecuentes

¿Cuándo hago MAJOR vs MINOR?

MAJOR (2.0.0): Si un usuario actualiza y su código se ROMPE → MAJOR

MINOR (1.1.0): Si un usuario actualiza y todo sigue funcionando → MINOR

¿Debo crear branch para cada pequeño cambio?

SÍ. Siempre trabaja en branches, incluso para cambios pequeños:

- Permite que GitHub Actions verifique antes de merge
- Historial más limpio
- Puedes descartar fácilmente si algo sale mal

¿Cuándo hacer release?

Flexible, pero algunas guías:

- PATCH: Cuando tengas 1+ bugfix importante
- MINOR: Cuando completes 1+ nueva feature
- MAJOR: Cuando hagas breaking changes (con cuidado)

Frecuencia recomendada:

- Patches: Cada 1-2 semanas
- Minor: Cada 1-2 meses
- Major: Solo cuando sea necesario

¿Puedo cambiar un tag después de crearlo?

NO recomendado una vez pusheado. Si lo haces:

- Los usuarios que ya instalaron la versión tendrán problemas
- Packagist se confunde
- Rompe la confianza

Si DEBES hacerlo:

```
# Eliminar tag
git tag -d v1.0.0
git push origin --delete v1.0.0

# Crear nuevo tag
git tag -a v1.0.0 nuevo_commit -m "..."
git push origin v1.0.0
```

Última actualización: 2025-10-10
Autor: Abdelkarim Mateos Sanchez

Aviso

Análisis: Principios de Good Code en Laravel

Contexto

Se plantean seis reglas fundamentales basadas en el Laravel Starter Kit de Nuno Maduro. El desafío no es solo aplicarlas en proyectos nuevos, sino integrar esta filosofía en proyectos existentes.

Cambio de paradigma fundamental

Tesis central: La flexibilidad de Laravel no debe confundirse con libertad para escribir código descuidado. La strictness no es un obstáculo, es un acelerador.

Problema identificado: Desarrolladores que utilizan la "magia" de Laravel como excusa para evitar tipos, tests incompletos y análisis estático.

Análisis de los seis principios

Principio 1: Cobertura de tipos al 100%

Qué implica:

```
pest --type-coverage --min=100
```

Análisis crítico:

Este principio fuerza la intencionalidad en cada variable y parámetro. La cobertura de tipos no es documentación, es contrato.

Ventajas:

- Reduce bugs de integración en un 60-70% según estudios empíricos
- El IDE se convierte en aliado activo

- Refactorizaciones seguras sin miedo

Desventajas:

- Requiere tiempo inicial significativo en proyectos legacy
- Puede generar fricción con equipos acostumbrados a PHP dinámico
- Arrays asociativos complejos necesitan DTOs o Value Objects

Implementación gradual en proyectos existentes:

1. Activar cobertura de tipos sin bloquear CI inicialmente
2. Establecer baseline actual
3. Regla: nuevo código debe tener 100% de cobertura
4. Refactorizar módulos críticos primero
5. Incrementar baseline mensualmente

Relación con SOLID:

Dependency Inversion Principle se beneficia directamente. Interfaces tipadas fuerzan contratos claros.

Principio 2: PHPStan en nivel MAX

Qué implica:

Análisis estático sin concesiones. PHPStan nivel 9 (MAX) detecta:

- Dead code
- Posibles nulls no manejados
- Tipos incompatibles en operaciones
- Propiedades no inicializadas

Análisis crítico:

El nivel MAX expone deuda técnica oculta. No es ego, es disciplina.

Ventajas:

- Detecta bugs antes de testing
- Complementa cobertura de tipos
- Fuerza a pensar en edge cases

Desventajas:

- Output inicial puede ser abrumador en proyectos legacy
- Requiere aprendizaje de sus reglas

- Algunas reglas pueden requerir baseline temporal

Implementación gradual:

```
# Paso 1: Generar baseline
vendor/bin/phpstan analyse --generate-baseline

# Paso 2: Subir nivel progresivamente
# phpstan.neon
parameters:
    level: 5  # Empezar aquí, no en MAX

# Paso 3: Incrementar cada sprint
# Sprint 1: nivel 5
# Sprint 2: nivel 6
# Sprint N: nivel 9 (MAX)
```

Relación con Testing:

PHPStan reduce la necesidad de ciertos tests unitarios. Si el tipo system garantiza que nunca pasarás null a una función, no necesitas testearlo.

Principio 3: Cobertura de tests al 100%

Qué implica:

```
pest --parallel --coverage --exactly=100.0
```

Análisis crítico:

La palabra clave es "exactly". No 99.8%. Exactamente 100%.

Ventajas:

- Elimina código muerto rápidamente
- Tests paralelos aceleran feedback loop
- Confianza absoluta en refactorizaciones

Desventajas:

- 100% coverage no garantiza calidad de tests
- Puede incentivar tests superficiales solo por métricas
- Requiere infraestructura para tests paralelos confiables

La trampa del 100%:

Cobertura de líneas no es cobertura de comportamiento. Un test puede ejecutar línea sin validar su lógica.

Mejor enfoque:

Combinar cobertura de líneas con mutation testing:

```
# Infection PHP
vendor/bin/infection --min-msi=80
```

Implementación gradual:

1. Identificar módulos críticos de negocio
2. Llevar estos a 100% primero
3. Expandir a módulos de soporte
4. Mantener 100% en código nuevo desde día 1

Relación con SOLID:

Single Responsibility Principle facilita testing. Clases con una responsabilidad son más fáciles de testear al 100%.

Principio 4: Formateo estricto automático

Qué implica:

Laravel Pint + Prettier eliminan decisiones de estilo.

Análisis crítico:

Este es el principio más subestimado. No es cosmético, es cognitivo.

Ventajas:

- Cero tiempo en code reviews discutiendo formato
- Diffs limpios enfocados en lógica
- Onboarding más rápido

Desventajas:

- Puede generar diffs masivos en primera aplicación
- Requiere consenso del equipo
- Algunos desarrolladores lo perciben como pérdida de "estilo personal"

Implementación inmediata:

```
# Aplicar una sola vez
./vendor/bin/pint

# Pre-commit hook
# .git/hooks/pre-commit
./vendor/bin/pint --dirty
```

Este principio no tiene desventajas técnicas reales. Solo resistencia humana.

Principio 5: Control del entorno en tests

Qué implica:

- Congelar tiempo
- Fake HTTP calls
- Forzar HTTPS en tests

Análisis crítico:

Tests deben ser deterministas. El mundo exterior es no determinista.

Ventajas en Laravel:

```
// Determinismo temporal
Carbon::setTestNow('2025-01-15 10:00:00');

// Aislamiento de HTTP
Http::fake([
    'api.external.com/*' => Http::response(['data' => 'fake'], 200)
]);
```

Desventajas:

- Requiere disciplina para identificar qué fakear
- Tests de integración reales siguen siendo necesarios
- Puede ocultar problemas de configuración real

Implementación:

Separar claramente:

- Unit tests: todo fakeado

- Integration tests: servicios reales en ambiente controlado
- E2E tests: flujo completo

Relación con Testing:

Este principio hace posible el 100% de cobertura. Sin él, tests son frágiles y lentos.

Principio 6: CI como senior engineer más estricto

Qué implica:

GitHub Actions ejecutando:

- Pint (formato)
- Rector (refactoring automático)
- PHPStan (análisis estático)
- Pest (tests)
- Lint (sintaxis)

Análisis crítico:

CI no es opcional. Es el guardián de calidad 24/7.

Ventajas:

- Feedback inmediato en PRs
- Imposible mergear código que rompe estándares
- Documentación viva de estándares del equipo

Desventajas:

- Requiere configuración inicial
- Puede ralentizar merges si CI es lento
- Falsos positivos pueden frustrar

Implementación óptima:

```
# .github/workflows/tests.yml
name: Tests

on: [push, pull_request]

jobs:
```

```
  tests:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3

      - name: Setup PHP
        uses: shivammathur/setup-php@v2
        with:
          php-version: 8.3
          coverage: xdebug

      - name: Install Dependencies
        run: composer install

      - name: Run Pint
        run: ./vendor/bin/pint --test

      - name: Run PHPStan
        run: ./vendor/bin/phpstan analyse

      - name: Run Pest
        run: ./vendor/bin/pest --parallel --coverage --min=100
```

Estrategia gradual:

1. Empezar solo con tests
2. Agregar Pint
3. Agregar PHPStan con baseline
4. Subir nivel progresivamente

El error fundamental identificado

Cita clave: "Confundir flexibilidad con libertad"

Laravel es flexible en arquitectura, no en disciplina. La flexibilidad es para resolver problemas de negocio, no para evitar buenas prácticas.

Análisis del error

Manifestaciones comunes:

- Arrays asociativos en lugar de DTOs
- Métodos sin type hints "porque total funciona"
- Tests solo de happy path
- "Lo arreglo después" que nunca llega

Costo real:

- Debugging que consume 40-60% del tiempo de desarrollo
- Bugs en producción que dañan reputación
- Refactorizaciones imposibles sin miedo
- Onboarding lento de nuevos desarrolladores

Roadmap de implementación en proyectos existentes

Fase 1: Evaluación y baseline (Sprint 1-2)

Acciones:

1. Ejecutar PHPStan nivel 5 con baseline
2. Medir cobertura de tipos actual
3. Medir cobertura de tests actual
4. Aplicar Pint una sola vez
5. Documentar estado actual

Entregable: Documento con métricas actuales y brechas.

Fase 2: Quick wins (Sprint 3-4)

Acciones:

1. Configurar Pint en pre-commit hook
2. Implementar CI básico (tests + Pint)
3. Nueva regla: código nuevo debe cumplir 100% tipos y tests
4. Identificar 3 módulos críticos para refactorizar

Entregable: CI funcionando, código nuevo bajo estándares.

Fase 3: Refactoring progresivo (Sprint 5-12)

Acciones:

1. Refactorizar módulos críticos a 100% tipos y tests
2. Subir PHPStan un nivel cada 2 sprints
3. Reducir baseline de PHPStan 10% cada sprint
4. Code reviews enfocados en principios

Entregable: Módulos críticos bajo estándares, PHPStan nivel 7-8.

Fase 4: Consolidación (Sprint 13+)

Acciones:

1. PHPStan nivel MAX sin baseline
2. 100% cobertura en todo el proyecto
3. Mutation testing implementado
4. Documentación de arquitectura

Entregable: Proyecto completamente bajo estándares.

Comparación: Enfoque tradicional vs Good Code

Aspecto	Tradicional	Good Code
Types	Opcional	Obligatorio 100%
Tests	"Lo importante"	100% exacto
Análisis estático	Si da tiempo	PHPStan MAX
Formato	Discutido en CR	Automatizado
CI	Tests básicos	Guardián completo
Velocidad inicial	Rápida	Moderada
Velocidad sostenida	Decreciente	Creciente
Confianza deployment	Rezar	Saber

Relación con SOLID

Single Responsibility Principle

Clases con una responsabilidad son más fáciles de tipar y testear al 100%.

Open/Closed Principle

Interfaces tipadas garantizan extensión sin modificación.

Liskov Substitution Principle

PHPStan MAX detecta violaciones automáticamente.

Interface Segregation Principle

Cobertura de tipos fuerza interfaces específicas.

Dependency Inversion Principle

Type hints en constructores formalizan inversión de dependencias.

Consideraciones finales

Pregunta clave

¿Es esto overkill para proyectos pequeños?

Respuesta: Depende del horizonte temporal. Si el proyecto vivirá más de 6 meses, no es overkill. Es inversión.

Obstáculos reales

1. **Resistencia del equipo**: Cambio cultural requiere tiempo
2. **Tiempo inicial**: Primera implementación consume sprints
3. **Falsa sensación de lentitud**: Strictness se siente lenta hasta que debuggear se vuelve raro

Beneficios medibles

- Reducción de bugs en producción: 60-80%
- Tiempo de onboarding: -50%
- Confianza en refactoring: +200%
- Velocidad de features (después de 3 meses): +30%

Conclusión

La filosofía de Good Code no es una lista de herramientas. Es un cambio de mentalidad: de "funciona ahora" a "funciona siempre".

El starter kit de Nuno Maduro no es el objetivo. Es el ejemplo de que es posible. La meta es internalizar estos principios hasta que escribir código sin tipos, sin tests o sin análisis estático se sienta antinatural.

Tesis final: Strictness no es burocracia. Es la única forma conocida de escalar complejidad sin colapsar en caos.

Recursos

- Laravel Starter Kit de Nuno Maduro
- PHPStan Documentation
- Pest PHP Documentation
- Laravel Pint Documentation
- Principios SOLID aplicados a Laravel - Opinated
- Agradecimientos a Vishal Rajpurohit por su hilo conductor

Aviso