

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 realmente 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](#).

Revision #1

Created 5 July 2023 18:50:48 by Abkrim

Updated 5 July 2023 18:56:11 by Abkrim