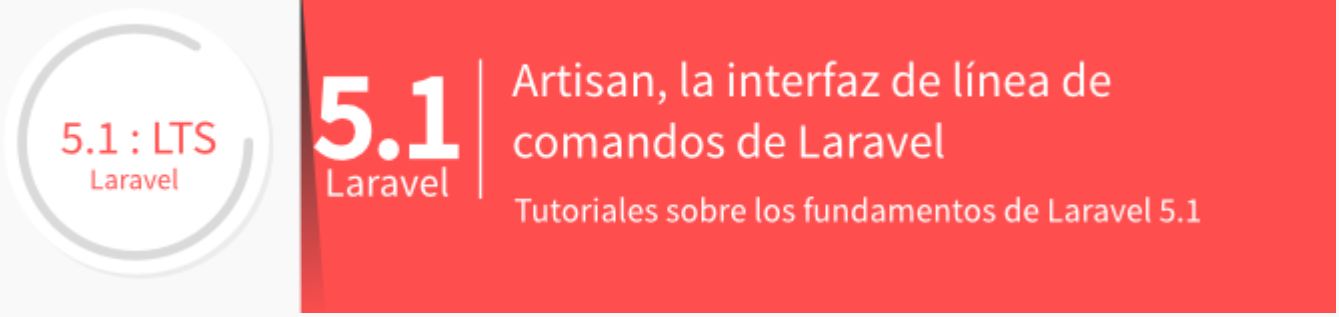


Entre las herramientas que Laravel nos proporciona para el desarrollo de aplicaciones se encuentra Artisan, la interfaz de línea de comandos (CLI por sus siglas en inglés de *Command-line interface*), la cual es un medio para la interacción con la aplicación donde los usuarios (en este caso los desarrolladores) dan instrucciones en forma de línea de texto simple o línea de comando. Artisan está basado en el componente Console de Symfony y nos ofrece un conjunto de comandos que nos pueden ayudar a realizar diferentes tareas durante el desarrollo e incluso cuando la aplicación se encuentra en producción. En el tutorial de hoy veremos las características de esta gran herramienta.

Para conocer el listado completo de los comandos disponibles ejecutamos en consola, en el directorio raíz de un proyecto de Laravel:

```
php artisan list
```

e inmediatamente veremos en la misma consola:

```
Laravel Framework version 5.1.26 (LTS)

Usage:
  command [options] [arguments]

Options:
  -h, --help            Display this help message
  -q, --quiet           Do not output any message
  -V, --version         Display this application version
      --ansi            Force ANSI output
      --no-ansi         Disable ANSI output
  -n, --no-interaction  Do not ask any interactive question
      --env[=ENV]       The environment the command should run under.
  -v|vv|vvv, --verbose  Increase the verbosity of messages: 1 for normal output, 2 for more verbose output and 3 for debug

Available commands:
  clear-compiled      Remove the compiled class file
  down                Put the application into maintenance mode
  env                 Display the current framework environment
  help                Displays help for a command
  inspire             Display an inspiring quote
  list                Lists commands
  migrate             Run the database migrations
  optimize            Optimize the framework for better performance
  serve               Serve the application on the PHP development server
  tinker              Interact with your application
  up                  Bring the application out of maintenance mode
 app
  app:name            Set the application namespace
 auth
  auth:clear-resets   Flush expired password reset tokens
 cache
  cache:clear         Flush the application cache
  cache:table         Create a migration for the cache database table
 config
  config:cache        Create a cache file for faster configuration loading
  config:clear        Remove the configuration cache file
 db
  db:seed             Seed the database with records
```

Para saber de qué trata y cuáles son las opciones que se tiene disponibles con cada comando podemos usar help:

```
php artisan help nombre-comando
```

o de esta manera:

```
php artisan nombre-comando -h
```

y nos mostrará: la forma de usar el comando, los argumentos que puede recibir, las opciones que podemos agregar y una descripción del comando. Por ejemplo, para `php artisan migrate -h` nos muestra:

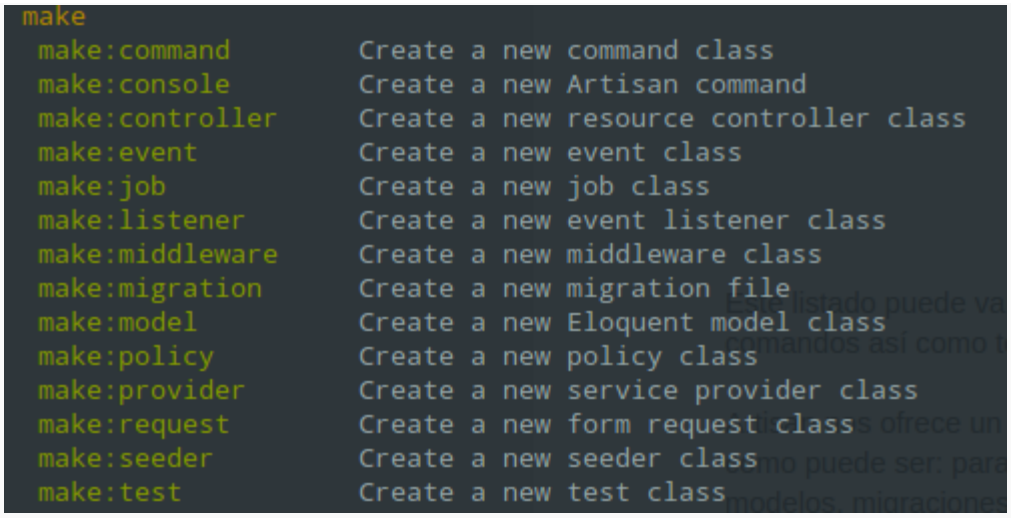
```
Usage:
  migrate [options]

Options:
      --database[=DATABASE]  The database connection to use.
      --force                Force the operation to run when in production.
      --path[=PATH]          The path of migrations files to be executed.
      --pretend              Dump the SQL queries that would be run.
      --seed                 Indicates if the seed task should be re-run.
  -h, --help                 Display this help message
  -q, --quiet                Do not output any message
  -V, --version              Display this application version
      --ansi                 Force ANSI output
      --no-ansi              Disable ANSI output
  -n, --no-interaction       Do not ask any interactive question
      --env[=ENV]            The environment the command should run under.
  -v|vv|vvv, --verbose       Increase the verbosity of messages: 1 for normal output,

Help:
 Run the database migrations
```

Así nos dice que es el comando para correr las migraciones y entre las opciones con que cuenta está poder indicar cuál es la base de datos por ejemplo `php artisan migrate --database=tests` .

Una de las grandes ventajas de esta herramienta es que tiene un conjunto de comandos dedicado a generadores, es decir, que nos permiten crear elementos como controladores, middleware, seeders, modelos, entre otros, los cuales son los que están bajo la categoría make:



Cada uno de esos generadores tiene sus propias opciones, así que puedes revisarlo con el comando help.

Por ejemplo, para crear un controlador sin ningún método usamos la opción plain:

```
1 php artisan make:controller PostController --plain
```

Otros comandos de uso común puede ser:

- Para levantar el servidor de una aplicación, es decir, para hacer correr una aplicación con el servidor que viene incluido en Laravel podemos hacerlo de manera fácil y rápida con:

  ```
  1 php artisan serve
  ```

  y nos mostrará:

  ```
  1 Laravel development server started on http://localhost:8000/
  ```

  De esta manera podemos ir al navegador, visitar http://localhost:8000/ y podremos ver nuestra aplicación funcionando. Para detener el servidor, en consola ejecutamos las teclas Ctrl+C. Tiene además las opciones de poder cambiar el puerto y el host.
- Para cambiar el namespace de la aplicación:

  ```
  1 php artisan app:name nombre-namespace
  ```

- Para ver el listado completo de rutas:

```
php artisan route:list
```

  Como resultado nos mostrará un listado de todas las rutas de la aplicación con la información sobre el método HTTP, la URI, la acción, el nombre y los middleware definidos para cada ruta, con lo cual, entre otras cosas, podemos verificar que todas las rutas de nuestra aplicación están bien definidas.

- Para interactuar con la aplicación (algo vimos en Tinker, la consola de comandos de Laravel)

```
php artisan tinker
```

- Para poner en modo mantenimiento nuestra aplicación:

```
php artisan down

php artisan up
```

El listado de comandos de Artisan puede variar de proyecto en proyecto pues hasta podemos crear nuestros propios comandos así como te mostramos en Artisan signatures en Laravel 5.1 y muchos paquetes de terceros lo usan para mejorar la interacción con sus componentes.

Por lo que con Artisan tenemos una gran herramienta para mejorar nuestro flujo de trabajo durante el desarrollo de aplicaciones. Espero que te haya gustado el tutorial y te parezca útil. Compártelo en las redes sociales y para cualquier duda abajo está la sección de comentarios. Además si necesitas un tutorial de un tema en específico puedes solicitarlo en Teach Me.