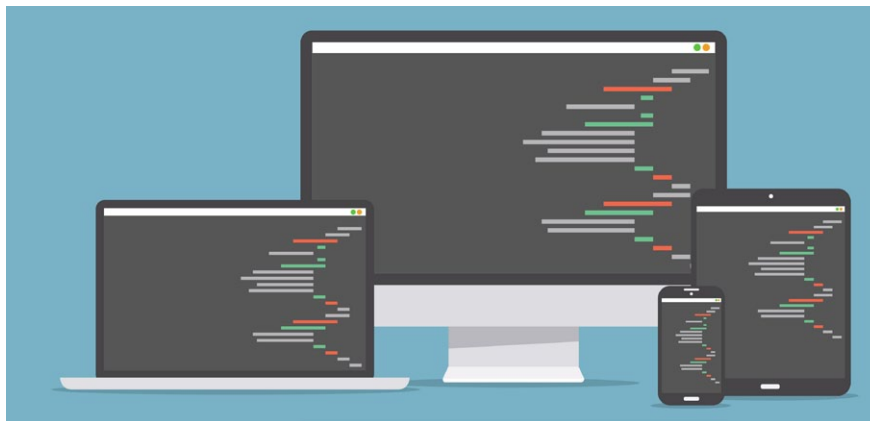


Sviluppo



Di Michele Costabile



Una soluzione per lo sviluppo web php progettata con rigore, ricca di funzionalità e pronta per ogni genere di progetto. Vediamo perché se ne parla così bene.

Laravel: il framework php primo in classifica

Laravel è un framework php fra i più completi e apprezzati. A una prima occhiata, la copertura delle Api sembra adatta a qualsiasi problema, la documentazione è estesa e la community online conta 2.400 follower e 12.000 domande su StackOverflow. Insomma, i numeri sono affini a quelli dei maggiori concorrenti.

Anche php soffre di una sovrabbondanza di scelte, la ricerca del framework perfetto è un'attività impegnativa e frustrante. Purtroppo, nel tempo richiesto da un'analisi comparativa approfondita delle caratteristiche di più soluzioni è capace di spuntare un nuovo contendente. Per queste ragioni, a un confronto esteso preferiamo valutare le reazioni di chi è passato da uno all'altro, misurare la consistenza e la vivacità delle diverse community, osservare la frequenza di aggiornamento e di creazione di estensioni. Questi fattori mettono Laravel in alta classifica.

Quello che è evidente è che anche Php è diventato un sistema ricco e complicato, più o meno come Java, o qualsiasi altro ambiente applicativo web. In uno sguardo di massima alla documentazione di Laravel incontriamo funzioni di autenticazione con ambienti sociali, come Facebook o Twitter, un sistema di pagamento, sottoscrizione e fatturazione, file system basati su cloud, più di un motore di template e il supporto per

database che include un object relational mapping. Detto in parole molto semplici, tanta roba. Proprio perché qualunque framework dispone di migliaia di estensioni, la prima cosa da capire è quanto sia efficace il sistema di pacchettizzazione.

IL PRIMO GIRO

Composer, l'ambiente di pacchettizzazione scelto dal team di Laravel è un tool per la gestione di dipendenze, modellato su npm, il sistema di pacchettizzazione di Node.js, e bundler, il sistema analogo in Ruby. Il sito web dedicato al progetto è getcomposer.org.

Composer non punta a essere un package manager pensato per gestire un'installazione globale, ma un configuratore a

livello di progetto, che preferisce l'etichetta di dependency manager.

L'uso è molto semplice: si dichiarano le dipendenze di un progetto in un file chiamato composer.json, per esempio come segue

```
{
  "require": {
    "monolog/monolog":
    "1.2.*"
  }
}
```

e si chiede a composer di risolvere le dipendenze trovando una versione 1.2 del package monolog e installandola nella directory predefinita – vendor – o in quella specificata nella riga di comando.

L'installazione di Composer su un Mac o una macchina Linux è squisitamente unix: si scarica un file php da Internet e lo si passa a php per l'esecuzione

L'installazione di Composer, qui nella finestra di terminale di un Mac, si limita a scaricare un archivio php (un file phar) che contiene l'applicazione

```
Last login: Fri Mar 20 23:55:22 on console
Potemkin:~ michele$ curl -sS https://getcomposer.org/installer | php
#!/usr/bin/env php
All settings correct for using Composer
Downloading...

Composer successfully installed to: /Users/michele/composer.phar
Use it: php composer.phar
Potemkin:~ michele$ php composer.phar

Composer version 1.0-dev (f10c71475167a4661225b14560ca0a400d730829) 2015-03-29 14:37:42

Usage:
  command [options] [arguments]

Options:
  --help (-h)            Display this help message
  --quiet (-q)           Do not output any message
  --verbose (-v|vv|vvv) Increase the verbosity of messages: 1 for normal output, 2 for more verbose output and 3 for debug
  --version (-V)         Display this application version
  --ansi                 Force ANSI output
  --no-ansi              Disable ANSI output
  --no-interaction (-n) Do not ask any interactive question
  --profile              Display timing and memory usage information
  --working-dir (-d)     If specified, use the given directory as working directory.
```

```
curl -sS https://getcomposer.org/
installer | php
```

Su Windows è disponibile un eseguibile di installazione, Composer-Setup.exe, come è lecito aspettarsi su questo sistema operativo.

Composer si limita a scaricare un file chiamato composer.phar, un archivio php eseguibile, che troveremo nella directory corrente e potremo eventualmente copiare in /usr/local/bin. In un'installazione Windows, invece, l'archivio sarà copiato in AppData e il path sarà aggiornato in modo da poter eseguire l'applicazione in una finestra di terminale qualsiasi.

Quando abbiamo Composer possiamo installare Laravel

```
composer global require "laravel/
installer=~1.1"
```

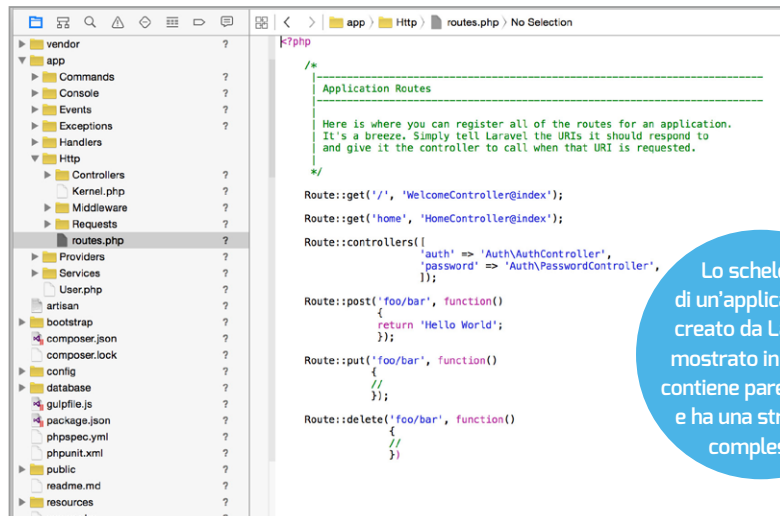
Adesso abbiamo a disposizione il comando laravel, che su unix troviamo in ~/.composer/vendor/bin, mentre su Windows è messo nel path dal programma di installazione. Per creare lo scheletro di un'applicazione, per esempio chiamata blog, la sintassi è molto semplice

```
laravel new blog
```

Il comando crea una directory chiamata blog e la riempie con lo scheletro completo di un'applicazione. Se abbiamo un'installazione corretta di Php 5.4 o superiore, possiamo eseguire l'applicazione con il comando

```
php artisan serve
```

Per vedere il risultato ci possiamo collegare a localhost:8000.



SALTARE I PROBLEMI DI CONFIGURAZIONE

C'è una magnifica scorciatoia per chi non vuole saperne di problemi di installazione o di compatibilità col sistema operativo, Homestead. È una soluzione, basata su un software di configurazione chiamato Vagrant, che permette di assemblare una macchina virtuale sulla base di un file di specifiche e avviarla dentro VirtualBox o VMWare. Installando Homestead, ci dotiamo di strumenti da riga di comando che permettono di assemblare con Vagrant una macchina virtuale basata su Ubuntu 14.04 e dotata di PHP 5.6, HHVM, Nginx, MySQL, Postgres, Node e un'infinità di altri dettagli che fanno una perfetta macchina di sviluppo web.

VMWare non è gratuito, ma VirtualBox e Vagrant lo sono e permettono di seguire questa strada a costo zero e senza dover avere una macchina Linux e alterare la configurazione del sistema principale. Per esempio, OS X Yosemite ha una versione di Php a cui manca il modulo mcrypt, che è un prerequisito

per Laravel. Un'altra possibilità è quella di creare una macchina virtuale Linux presso un fornitore di sistemi nel cloud, come Amazon o Windows Azure. Se non si esagera con le specifiche del sistema, è facile potersela cavare con poco, o anche rientrare in un piano di uso gratuito.

L'installazione del software su una macchina vergine è facilitata dal fatto che nella sala macchine di Amazon o di Microsoft, la banda per scaricare il software è ben diversa da quella utilizzabile a casa o al lavoro.

PRIMI PASSI

Dopo la creazione di uno scheletro di applicazione, per esempio con il comando

```
laravel new <nome applicazione>
```

troviamo uno schema di directory molto complesso. Alcune cartelle hanno un nome che ne chiarisce subito il contenuto, come config, database, resources, storage e tests. La cartella vendor contiene i prerequisiti dell'applicazione, se manca, basta il comando composer install, per crearla. I file di programma sono nella cartella app, che a sua volta contiene altre otto directory. In una di queste, Http, troviamo il file routes.php, nel quale si definisce lo schema di indirizzamento delle richieste del browser al codice che

VirtualBox

Welcome to VirtualBox.org!

VirtualBox is a powerful x86 and AMD64/Intel64 virtualization product for enterprise as well as home use. Not only is VirtualBox an extremely feature rich, high performance product for enterprise customers, it is also the only professional solution that is freely available as Open Source Software under the terms of the GNU General Public License (GPL) version 2. See "About VirtualBox" for an introduction.

Presently, VirtualBox runs on Windows, Linux, Macintosh, and Solaris hosts and supports a large number of guest operating systems including but not limited to Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8), DOS/Windows 3.x, Linux (2.4, 2.6 and 3.x), Solaris and OpenSolaris, OS/2, and OpenBSD.

VirtualBox is being actively developed with frequent releases and has an ever growing list of features, supported guest operating systems and platforms it runs on. VirtualBox is a community effort backed by a dedicated company: everyone is encouraged to contribute while Oracle ensures the product always meets professional quality criteria.

Hot picks:

- Pre-built virtual machines for developers at [Oracle Tech Network](#)
- Hyperbox Open-source Virtual Infrastructure Manager [project site](#)
- phpVirtualBox AJAX web interface [project site](#)
- IQemu automated Windows VM creation, application integration [project site](#)

News Flash

- New Mar 16th, 2015**
VirtualBox 4.3.26 released! Oracle today released another 4.3 maintenance release which improves stability and fixes regressions. See the changelog for details.
- Important! February, 2015**
We're hiring! Looking for a new challenge? We're looking for generic product developers (Russia).
- New Jan 17th, 2015**
VirtualBox 4.2.28, 4.1.36, 4.0.28 and 3.2.26 released! Oracle today released maintenance releases which improve stability and fixes regressions. See the respective changelogs for details.

[More information...](#)

Virtualbox è un ambiente di virtualizzazione creato in origine da Sun. Forse non è consistentemente primo nei benchmark, ma è open source.

le serve, come in altri application server. Per esempio, il comando

```
Route::get('/',
'WelcomeController@index');
```

Invia le richieste per la radice del server alla classe WelcomeController, che possiamo mostrare per intero:

```
class WelcomeController extends
Controller {

    /**
     * Create a new
controller instance.
     *
     * @return void
     */
    public function
__construct()
    {

$this->middleware('guest');
    }

    /**
     * Show the application
welcome screen to the user.
     *
     * @return Response
     */
    public function index()
    {

        return
view('welcome');
    }

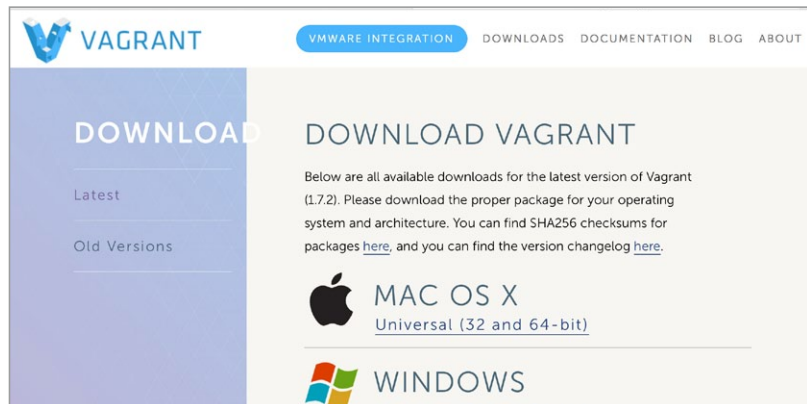
}
```

Si noti che la pagina consente l'accesso solo agli utenti ospiti (\$this->middleware('guest')); e che la pagina restituita all'utente sarà il testo prodotto dalla funzione view('welcome'). La view la troviamo nella cartella Resources, si tratta di un template blade

```
<html>
<head>
<title>Laravel</title>

<link href="//
fonts.googleapis.com/
css?family=Lato:100'
rel='stylesheet' type='text/css'>
```

```
<style>
body {
margin: 0;
padding: 0;
width: 100%;
```



Vagrant è un configuratore di macchine virtuali, che permette di costruire l'immagine di un sistema applicativo dotato di ambienti operativi in versioni controllate, che supporta VirtualBox, VMWare e Parallels. Laravel offre un file di configurazione che permette di creare un sistema Ubuntu 14.4, pronto per lavorare.

```
height: 100%;
color: #B0BEC5;
display: table;
font-weight: 100;
font-family: 'Lato';
}

.container {
text-align: center;
display: table-cell;
vertical-align: middle;
}

.content {
text-align: center;
display: inline-block;
}

.title {
font-size: 96px;
margin-bottom: 40px;
}

.quote {
font-size: 24px;
}

</style>
</head>
<body>
<div class="container">
<div class="content">
<div
class="title">Laravel 5</div>
<div class="quote">{{
Inspiring::quote() }}</div>
</div>
</div>
</body>
</html>
```

La prima parte è occupata dalla descrizione dello stile della pagina in css, l'ultima parte è molto semplice: una serie di contenitori, con all'interno una citazione. Ed ecco l'origine del test: la funzione

quote, dentro Resources, che crea una lista di citazioni ispiratrici e ne sceglie una a caso:

```
<?php namespace Illuminate\
Foundation;

use Illuminate\Support\
Collection;

class Inspiring {

    /**
     * Get an inspiring quote.
     * @return string
     */
    public static function quote()
    {

        return Collection::make([

            'When there is no desire,
all things are at peace.
- Laozi',

            'Simplicity is the ultimate
sophistication. - Leonardo da
Vinci',

            'Simplicity is the essence
of happiness. - Cedric Bledsoe',

            'Smile, breathe, and go
slowly. - Thich Nhat Hanh',

            'Simplicity is an acquired
taste. - Katharine Gerould',

            'Well begun is half done.
- Aristotle',

        ]->random();
    }

}
```

Questo tour ci permette di vedere alcune funzioni di Laravel e di mettere in luce un'architettura molto ragionata e articolata. Da questa organizzazione possiamo aspettarci sia una buona sistemazione di tutti i pezzi, sia una certa complessità

iniziale, che richiede un periodo di familiarizzazione con la piattaforma da non sottovalutare. Il secondo punto non deve stupire: è comune a qualsiasi altro application server di una certa potenza, indipendentemente dal linguaggio di implementazione, probabilmente è un segno di maturità.

Dobbiamo aggiungere che, non solo la progettazione di Laravel assegna un posto a ogni cosa e divide in maniera corretta le responsabilità fra le aree di codice, ma integra anche un ambiente per la creazione di test unitari, indispensabile per applicare criteri di controllo della qualità del software.

ELOQUENT ORM

Un trattamento particolare lo merita l'architettura dello strato di interfaccia con i database. Non solo sono supportati tutti i principali database, ma possiamo usare uno schema molto sofisticato di mapping fra oggetti e relazioni, chiamato Eloquent. La creazione di un database, dal punto di vista del codice è semplice. Ecco come viene creata e rimossa una tabella utenti di un'applicazione semplice

```
use Illuminate\Database\Schema\
Blueprint;
use Illuminate\Database\
Migrations\Migration;
```

```
class CreateUsersTable extends
Migration {
```

```
/**
 * Run the migrations.
 *
 * @return void
 */
public function up()
{
    Schema::create('users',
function(Blueprint $table)
    {
        $table->increments('id');
        $table->string('name');

        $table->string('email')->unique();
        $table->string('password',
60);
        $table->rememberToken();
        $table->timestamps();
    });
}
```

```
/**
 * Reverse the migrations.
 *
```

```
* @return void
*/
public function down()
{
    Schema::drop('users');
}
```

L'uso del database va da un modello molto semplice, come questo

```
$results = DB::select('select *
from users where id = ?', [1]);
```

a uno più evoluto, come il seguente

```
$user = DB::table('users')-
>where('name', 'John')->first();
```

o anche questo

```
DB::table('users')
->join('contacts',
'users.id', '=', 'contacts.
user_id')
->join('orders',
'users.id', '=', 'orders.
user_id')
->select('users.id',
'contacts.phone', 'orders.price')
->get();
```

Infine, abbiamo la possibilità di un mapping fra oggetti e relazioni. Iniziamo con la creazione di un modello, dalla riga di comando:

```
php artisan make:model User
```

Naturalmente, viene dato molto per scontato: si usa la connessione di default, si assume che la tabella da modellare si chiami User e la classe generata avrà lo stesso nome. Si possono complicare le cose e si possono gestire le eccezioni, ma come con Ruby on Rails, la convenzione prevale sulla configurazione, quando è possibile. Con una classe modello possiamo fare interrogazioni semplici

```
$users = User::all();
```

e ricerche con chiave

```
$user = User::find(1);
```

L'interfaccia a oggetti trasforma le select e le update nella manipolazione delle proprietà di oggetti, come è mostrato da questo esempio

```
$user = new User;
$user->name = 'John';
$user->save();
```

Fra le possibilità avanzate, troviamo la cancellazione differita dei record, con la possibilità di annullare una cancellazione, basate su due campi aggiunti a una tabella, con data di creazione e cancellazione. Questo permette una cancellazione soft con una specie di cestino, fino al momento in cui non si libera lo spazio dai record cancellati. Non manca la possibilità di lavorare con i record a blocchi, come di norma avviene in tutte le applicazioni pratiche, con la gestione della paginazione.

Ecco un esempio di come avviene la paginazione di un grosso dataset, passando una closure a una funzione chunk

```
User::chunk(200, function($users)
{
    foreach ($users as $user)
    {
        //
    }
});
```

CONCLUSIONI

Laravel è un framework articolato e complesso, ricco di componenti e funzioni, è complicato come lo sviluppo di applicazioni non banali. Quando le circostanze dettano la scelta di php come linguaggio, è difficile pensare a un framework più completo e ricco di estensioni. •

PER SAPERNE DI PIÙ

→ phpframeworks.com: Un sito dedicato al confronto fra i principali framework php. Non parla, però di Laravel.

→ vschart.com/compare/laravel-vs/yii: Un sito di confronto con una selezione estesa di punti di forza e di debolezza di Laravel e Yii. Si possono confrontare anche parecchi altri sistemi software.

→ laracasts.com: Un archivio di video didattici su Laravel, brevi e diretti.

→ laravel.com/docs/5.0: La documentazione del framework è molto completa.

→ laravel-italia.it: La comunità italiana di utenti del framework.

→ envoyer.io: Un sito che offre hosting dedicato alle applicazioni Laravel