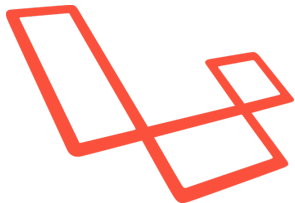


# Laravel, Framework PHP para la Web

Jhonatan I. Castro Rocabado

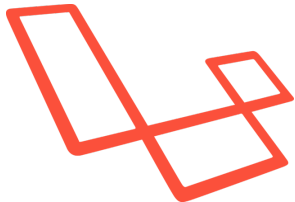
26 de agosto de 2016

Laravel es un Framework PHP open-source.  
creado por Taylor Otwell en junio del 2011.



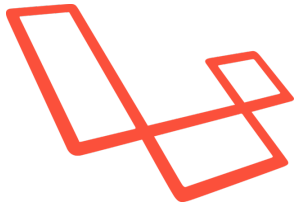
Laravel es un Framework PHP open-source.  
creado por Taylor Otwell en junio del 2011.

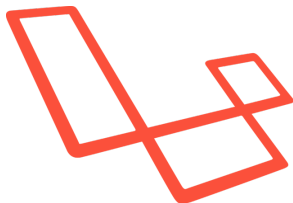
- Basado en Symfony.



Laravel es un Framework PHP open-source.  
creado por Taylor Otwell en junio del 2011.

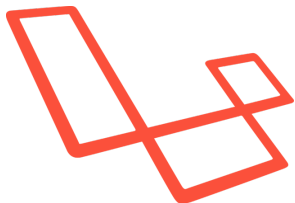
- Basado en Symfony.
- Sigue la arquitectura  
Modelo-Vista-Controlador (MVC).





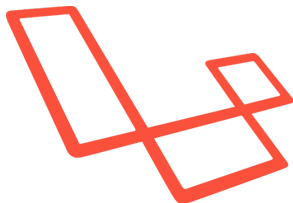
Laravel es un Framework PHP open-source.  
creado por Taylor Otwell en junio del 2011.

- Basado en Symfony.
- Sigue la arquitectura  
Modelo-Vista-Controlador (MVC).
- Sistema de empaquetado modular.



Laravel es un Framework PHP open-source.  
creado por Taylor Otwell en junio del 2011.

- Basado en Symfony.
- Sigue la arquitectura  
Modelo-Vista-Controlador (MVC).
- Sistema de empaquetado modular.
- Diferentes formas para acceder a bases  
de datos relacionales.



Laravel es un Framework PHP open-source.  
creado por Taylor Otwell en junio del 2011.

- Basado en Symfony.
- Sigue la arquitectura Modelo-Vista-Controlador (MVC).
- Sistema de empaquetado modular.
- Diferentes formas para acceder a bases de datos relacionales.
- Utilidades que ayudan al deployment de la aplicación y a su mantenimiento.

# ¿Por qué utilizar un Framework?

- Menos código y desarrollo rápido.



# ¿Por qué utilizar un Framework?

- Menos código y desarrollo rápido.
- Proporcionar un código organizado, reusable y mantenible.

# ¿Por qué utilizar un Framework?

- Menos código y desarrollo rápido.
- Proporcionar un código organizado, reusable y mantenible.
- Facilitar el crecimiento de la aplicación, ya que los Frameworks son escalables.

# ¿Por qué utilizar un Framework?

- Menos código y desarrollo rápido.
- Proporcionar un código organizado, reusable y mantenible.
- Facilitar el crecimiento de la aplicación, ya que los Frameworks son escalables.
- Librarte de la preocupación de que tu aplicación tenga problemas de seguridad leves.

# ¿Por qué utilizar un Framework?

- Menos código y desarrollo rápido.
- Proporcionar un código organizado, reusable y mantenible.
- Facilitar el crecimiento de la aplicación, ya que los Frameworks son escalables.
- Librarte de la preocupación de que tu aplicación tenga problemas de seguridad leves.
- Seguir el patrón MVC que separa la presentación de la lógica.

# ¿Por qué utilizar un Framework?

- Menos código y desarrollo rápido.
- Proporcionar un código organizado, reusable y mantenible.
- Facilitar el crecimiento de la aplicación, ya que los Frameworks son escalables.
- Librarte de la preocupación de que tu aplicación tenga problemas de seguridad leves.
- Seguir el patrón MVC que separa la presentación de la lógica.
- **Herramientas de performance.**

# ¿Por qué utilizar un Framework?

- Menos código y desarrollo rápido.
- Proporcionar un código organizado, reusable y mantenible.
- Facilitar el crecimiento de la aplicación, ya que los Frameworks son escalables.
- Librarte de la preocupación de que tu aplicación tenga problemas de seguridad leves.
- Seguir el patrón MVC que separa la presentación de la lógica.
- Herramientas de performance.
- Soporte de la comunidad.

# ¿Por qué utilizar un Framework?

- Menos código y desarrollo rápido.
- Proporcionar un código organizado, reusable y mantenible.
- Facilitar el crecimiento de la aplicación, ya que los Frameworks son escalables.
- Librarte de la preocupación de que tu aplicación tenga problemas de seguridad leves.
- Seguir el patrón MVC que separa la presentación de la lógica.
- Herramientas de performance.
- Soporte de la comunidad.
- Utilidades y librerías.

# ¿Por qué utilizar un Framework?

- Menos código y desarrollo rápido.
- Proporcionar un código organizado, reusable y mantenible.
- Facilitar el crecimiento de la aplicación, ya que los Frameworks son escalables.
- Librarte de la preocupación de que tu aplicación tenga problemas de seguridad leves.
- Seguir el patrón MVC que separa la presentación de la lógica.
- Herramientas de performance.
- Soporte de la comunidad.
- Utilidades y librerías.
- **Abstracción de la base de datos.**

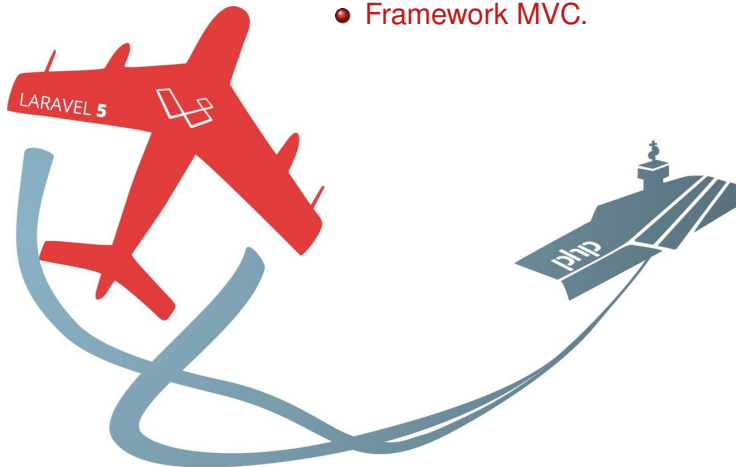


# ¿Por qué utilizar un Framework?

- Menos código y desarrollo rápido.
- Proporcionar un código organizado, reusable y mantenible.
- Facilitar el crecimiento de la aplicación, ya que los Frameworks son escalables.
- Librarte de la preocupación de que tu aplicación tenga problemas de seguridad leves.
- Seguir el patrón MVC que separa la presentación de la lógica.
- Herramientas de performance.
- Soporte de la comunidad.
- Utilidades y librerías.
- Abstracción de la base de datos.
- ...

# ¿Por qué elegir Laravel?

- Framework MVC.



# ¿Por qué elegir Laravel?



- Framework MVC.
- Convenciones y no configuraciones.



# ¿Por qué elegir Laravel?



- Framework MVC.
- Convenciones y no configuraciones.
- Interfaces ya codificadas (autenticación, base de datos, caché, colas, ...)



# ¿Por qué elegir Laravel?



- Framework MVC.
- Convenciones y no configuraciones.
- Interfaces ya codificadas (autenticación, base de datos, caché, colas, ...)
- Motor de plantillas Blade.

# ¿Por qué elegir Laravel?



- Framework MVC.
- Convenciones y no configuraciones.
- Interfaces ya codificadas (autenticación, base de datos, caché, colas, ...)
- Motor de plantillas Blade.
- **Curva de aprendizaje pequeña.**

# ¿Por qué elegir Laravel?



- Framework MVC.
- Convenciones y no configuraciones.
- Interfaces ya codificadas (autenticación, base de datos, caché, colas, ...)
- Motor de plantillas Blade.
- Curva de aprendizaje pequeña.
- Está bien documentado.

# ¿Por qué elegir Laravel?




- Framework MVC.
- Convenciones y no configuraciones.
- Interfaces ya codificadas (autenticación, base de datos, caché, colas, ...)
- Motor de plantillas Blade.
- Curva de aprendizaje pequeña.
- Está bien documentado.
- **Comunidad amplia y activa.**



:v

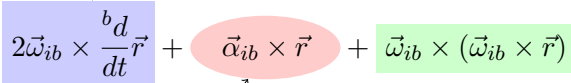
# Laravel, Framework PHP para la Web

- Coriolis acceleration


$$\vec{a}_p = \vec{a}_o + \frac{d^2}{dt^2}\vec{r} + 2\vec{\omega}_{ib} \times \frac{d}{dt}\vec{r} + \vec{\alpha}_{ib} \times \vec{r} + \vec{\omega}_{ib} \times (\vec{\omega}_{ib} \times \vec{r})$$

# Laravel, Framework PHP para la Web

- Coriolis acceleration

$$\vec{a}_p = \vec{a}_o + \frac{b d^2}{dt^2} \vec{r} + 2\vec{\omega}_{ib} \times \frac{b d}{dt} \vec{r} + \vec{\alpha}_{ib} \times \vec{r} + \vec{\omega}_{ib} \times (\vec{\omega}_{ib} \times \vec{r})$$


- Transversal acceleration

# Laravel, Framework PHP para la Web

- Coriolis acceleration

$$\vec{a}_p = \vec{a}_o + \frac{b d^2}{dt^2} \vec{r} + 2\vec{\omega}_{ib} \times \frac{b d}{dt} \vec{r} + \vec{\alpha}_{ib} \times \vec{r} + \vec{\omega}_{ib} \times (\vec{\omega}_{ib} \times \vec{r})$$

The diagram shows the equation for the acceleration of a point  $\vec{a}_p$  relative to an origin  $\vec{a}_o$ . The equation is broken down into five terms. Arrows from the text labels below point to specific terms: 'Coriolis acceleration' points to the blue box term, 'Transversal acceleration' points to the pink oval term, and 'Centripetal acceleration' points to the green box term.

- Transversal acceleration

- Centripetal acceleration