



Unidad 4 / Escenario 8

Lectura fundamental

Codeship, deployment pipelines y browser testing

Contenido

1 ¿Qué es Codeship?

2 Características

Palabras clave: turnKey, comando, repositorios, docker

1. ¿Qué es Codeship?

Codeship es una herramienta paga, con funciones gratuitas limitadas a la realización de pruebas; de uso simple; bastante intuitiva, utilizada con el fin de facilitar la implementación de integración continua y el despliegue continuo, dentro de repositorios. A través de las principales herramientas de gestión de repositorios (*Github*, *Bitbucket* y *Gitlab*) Codeship permite una completa ejecución de pruebas con el fin de generar una distribución continua a partir de parámetros preestablecidos.

A. Codeship Basic

Dentro de las características que ofrece este paquete están:

- Es configurado a partir de una interfaz web.
- La construcción de versiones es creada a partir de máquinas preconfiguradas.
- Desarrollos Turnkey integrados.
- Ideal para tecnologías comunes y flujos de trabajo.
- No tiene soporte de Docker.
- Costo mínimo de acuerdo con número de equipo.

B. Codeship Pro

Este paquete ofrece lo siguiente:

- Se conforma a través de la configuración establecida en el repositorio
- Capacidad de definir contenedores para la construcción de ambientes
- Flujos de trabajo y configuración flexible
- Configuración de flujos de trabajo y características editables
- Soporte nativo de Docker
- Costo de acuerdo con número de Builds a generar.

2. Características

Codeship permite la utilización de la herramienta, de manera gratuita, para la realización de test simples, dejando a un lado la realización de Builds. Dentro de las características que se pueden realizar se tiene:

A. Tablero inicial

El tablero inicial de Codeship se basa en el historial de pruebas del repositorio seleccionado. Cada actualización del repositorio genera una prueba nueva, realizada de acuerdo con la configuración establecida en los comandos de preparación y prueba, como se muestra en la figura 1.

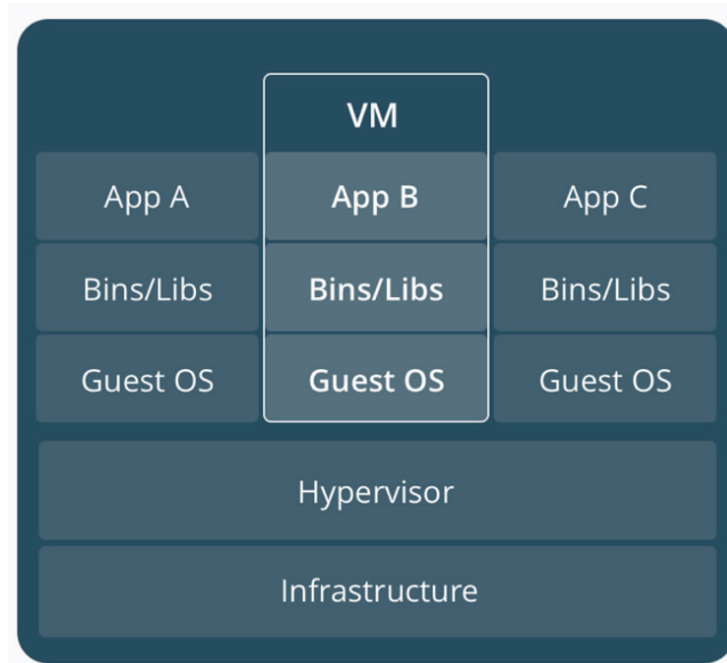


Figura 1. Pantallazo Dashboard

Fuente: elaboración propia

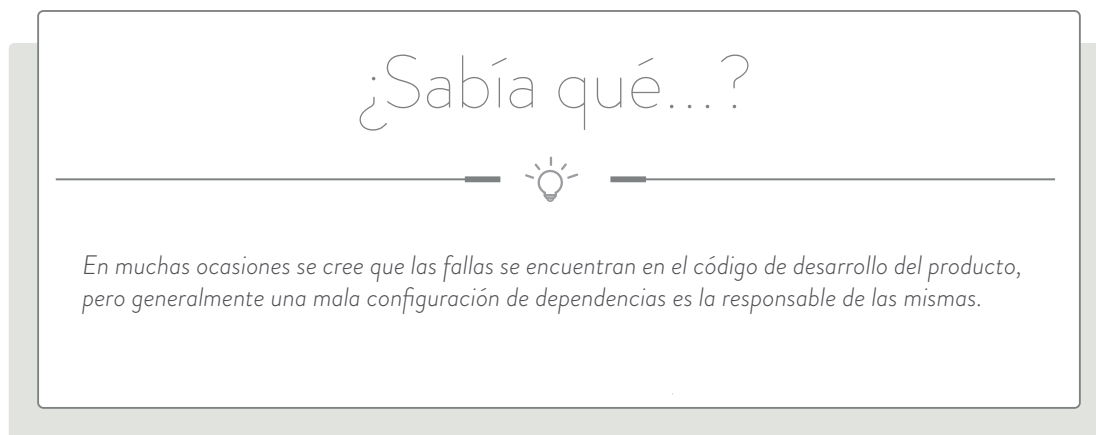
Cada Build podrá tener 1 de 3 estados:

- **Completado:**

Representado con el color verde, indica que la prueba realizada no generó inconvenientes, tanto con el cumplimiento de las dependencias, como con la satisfacción de código.

- **Con Fallas:**

Representado por el color rojo, como su nombre lo dice, en caso de que, en el proceso, se encuentre una falla, tanto en la ejecución de pruebas, la instalación de dependencias o el código del repositorio contenedor, el build no podrá ser generado y será necesario reformular, ya sea la configuración del build (Dependencias y código de pruebas), o en algunos casos, el código dentro del repositorio, en tal caso sería necesario una nueva acción de push del proyecto con los cambios.



- **En Proceso:**

Como su nombre lo indica, este estado representado por el color azul, se puede evidenciar en el momento de compilación del código y de cada prueba, la velocidad de ejecución es acorde a la cantidad de procesos que se tienen que ejecutar.


B. Detalle

Dentro del detalle de cada operación, se puede encontrar el historial del Build generado. De acuerdo con el lenguaje seleccionado se pueden observar los procesos requeridos para la ejecución del mismo, en el caso de la figura 2, podemos ver el proceso de un Build generado para el lenguaje PHP. En este ejemplo se tiene una prueba de test de verificación de correo electrónico fallido. Como se puede observar, Codeship realiza para la prueba la operación de clone del repositorio, instalación y preparación para dependencias, y preparación de la máquina virtual.

Cómo mejorar...



Previo a la automatización de pruebas, es necesario asegurarse de que las pruebas locales se realicen, en muchas ocasiones tiene menor dificultad la captura de errores locales que mediante el compilador de Codeship.



Author	Commit	Message		Date	Builds
? EC2 Default U...	0f51845	last changes	develop	2016-11-18	
? EC2 Default U...	1b258eb	sept	develop	2016-09-06	
Eduardo Oliver...	b290378 M	Merged develop into master		2016-05-11	
? EC2 Default U...	dbc6b7e	admin config		2016-05-11	
? EC2 Default U...	cde0fa0	commit modify post		2016-05-10	
? EC2 Default U...	2f2304e	5 de mayo		2016-05-05	
? EC2 Default U...	865e1c8	fix facebook		2016-05-02	
? EC2 Default U...	2acfd7	final commit before store		2016-05-01	
? EC2 Default U...	b2922fe	changes in hhtpd config and chat		2016-04-29	
? EC2 Default U...	3136467	Initial commit		2016-04-27	

Figura 2. Pantallazo Historial del Build

Fuente: elaboración propia

En la descripción del error, de acuerdo con el paso, tendrá un error característico, en este caso (Figura 3), se está haciendo referencia a un archivo Test, que no fue agregado en el directorio, lo que generó la pausa el build exitoso. De esta forma, es posible identificar, cuándo se está listo para aprobar un *pull request*, agregar la funcionalidad a la rama padre, etc.

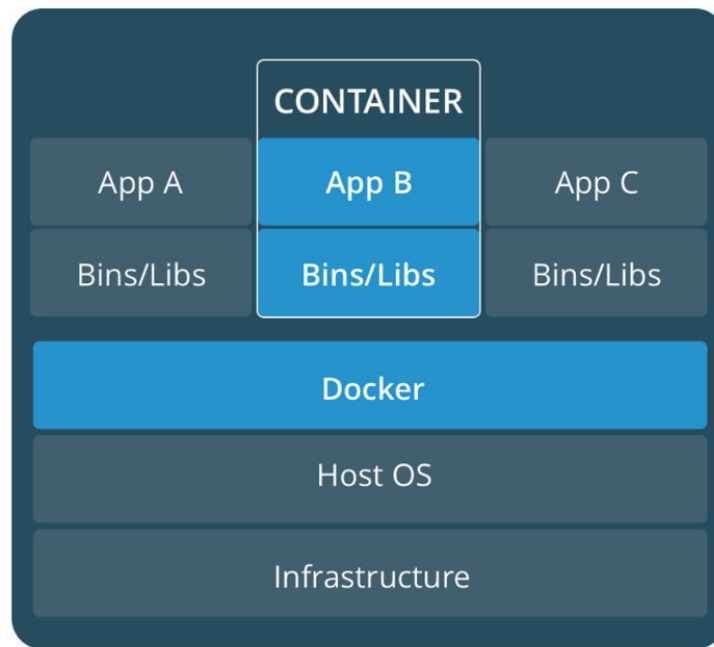


Figura 3. Pantallazo Log error de compilación

Fuente: elaboración propia

C. Lenguajes

Codeship permite la integración de repositorios basados en varios lenguajes, la mayoría especializados en el desarrollo Web; dentro de los cuales se encuentran PHP, Node Js, Ruby y mas (Figura 4).

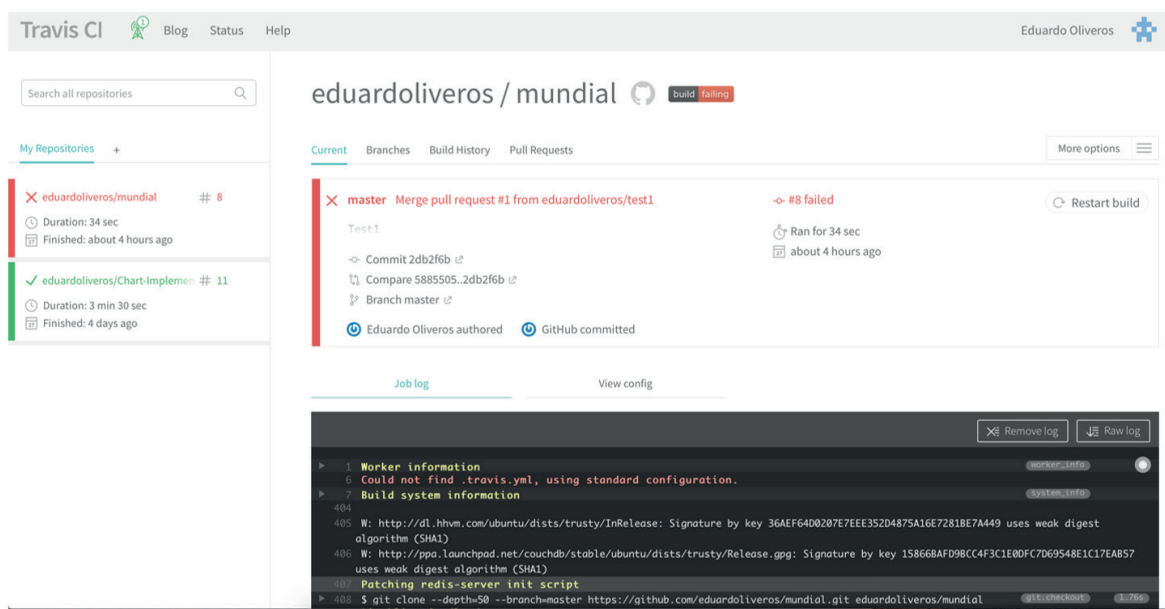


Figura 4. Pantallazo Lenguajes permitidos

Fuente: elaboración propia

D. Comandos de preparación

Dentro de la sección de test, de acuerdo con el lenguaje que se maneje, y las herramientas que se utilicen, será necesaria la implementación de todas las dependencias, extensiones y configuraciones del lenguaje, como se puede ver en la figura 5. En este caso se hace referencia a la versión de PHP en la que se desea realizar la prueba, en caso de que la herramienta requiera un elemento especial, será necesario incluir el comando de instalación de paquete, de la misma forma como se configuraría en un servidor.

```
Eduardos-MacBook-Pro:phpserver eduardo$ docker images
```

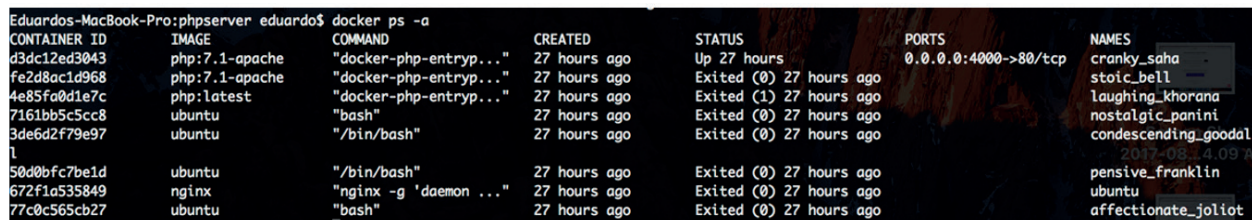
REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
php	latest	74deaab6a609	27 hours ago	370MB
eduardoliveros/web	latest	31d97a406c4e	10 days ago	230MB
myhtop	latest	1b5971a20796	10 days ago	421MB
myhtop	pruebatag	1b5971a20796	10 days ago	421MB
<none>	<none>	f13ac0616b7a	10 days ago	120MB
eduardoliveros/ubuntu	latest	ccc7a11d65b1	13 days ago	120MB

Figura 5. Pantallazo Comandos de preparación

Fuente: elaboración propia

E. Comandos de prueba

Los comandos de prueba hacen referencia a las pruebas que se deben realizar en cada build. Estas van de acuerdo con el lenguaje en el que se está desarrollando, como se puede observar en la figura 6. El lenguaje utilizado es PHP, por lo tanto, se utiliza phpunit para la ejecución de esta prueba específica. En este Build, únicamente se genera una prueba establecida, aunque Codeship permite la ejecución de hasta 20 pipelines para la versión gratuita.



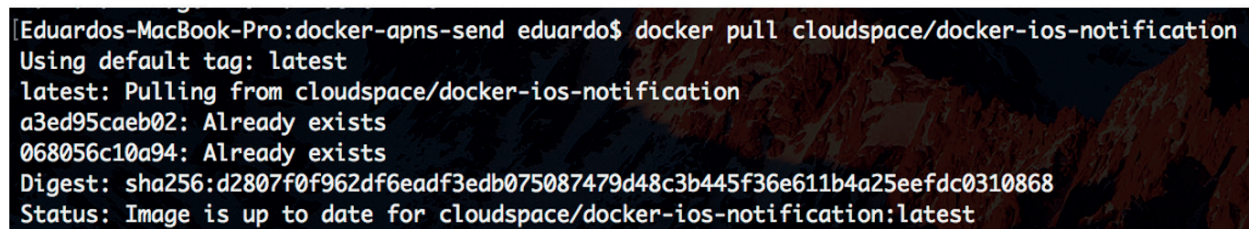
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
d3dc12ed3043	php:7.1-apache	"docker-php-entryp..."	27 hours ago	Up 27 hours	0.0.0.0:4000->80/tcp	cranky_saha
fe2d8ac1d968	php:7.1-apache	"docker-php-entryp..."	27 hours ago	Exited (0) 27 hours ago		stoic_bell
4e85fa0d1e7c	php:latest	"docker-php-entryp..."	27 hours ago	Exited (1) 27 hours ago		laughing_khorana
7161bb5c5cc8	ubuntu	"bash"	27 hours ago	Exited (0) 27 hours ago		nostalgic_panini
3de6d2f79e97	ubuntu	"/bin/bash"	27 hours ago	Exited (0) 27 hours ago		condescending_goodal
50d0bfc7be1d	ubuntu	"/bin/bash"	27 hours ago	Exited (0) 27 hours ago		pensive_franklin
672f1a535849	nginx	"nginx -g 'daemon ...'"	27 hours ago	Exited (0) 27 hours ago		ubuntu
77c0c565cb27	ubuntu	"bash"	27 hours ago	Exited (0) 27 hours ago		affectionate_joliot

Figura 6. Pantallazo Comandos de prueba

Fuente: elaboración propia

F. Configuración de despliegue

Posterior a la ejecución de pruebas y resultado satisfactorio por parte del Build, es posible programar un despliegue automático. Inicialmente será necesario seleccionar la rama a la que disparará la funcionalidad. (Figura 7).



```
Eduardos-MacBook-Pro:docker-apns-send eduardo$ docker pull cloudspace/docker-ios-notification
Using default tag: latest
latest: Pulling from cloudspace/docker-ios-notification
a3ed95caeb02: Already exists
068056c10a94: Already exists
Digest: sha256:d2807f0f962df6eadf3edb075087479d48c3b445f36e611b4a25eefdc0310868
Status: Image is up to date for cloudspace/docker-ios-notification:latest
```

Figura 7. Pantallazo Configuración de rama disparadora

Fuente: elaboración propia

Inmediatamente después de haber seleccionado la rama que dispare la funcionalidad, es necesario configurar las llaves de la herramienta de despliegue. Codeship permite la integración con las siguientes herramientas.

- *Google AppEngine*
- *Codeplay*
- *Capistrano*
- *Amazon Elastic Beanstalk*
- *Engine Yard*
- *Heroku*
- *Nodejitsu*
- *Amazon S3*

Por último, será necesario configurar la conexión de acuerdo con la herramienta. En la figura 8 se puede observar la configuración de Amazon S3. Habiendo configurado esto, los pasos están listos para realizar pruebas y distribución de nuestro código alojado en repositorios.

```
Eduardos-MacBook-Pro:~ eduardo$ docker commit d3dc12ed3043 ubuntu:dep
sha256:697c9a3ac08bd2e2a3b8aa371f2d7c7b7fe6356db59d7ae122c290f988169489
Eduardos-MacBook-Pro:~ eduardo$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ubuntu:dep	latest	697c9a3ac08b	13 seconds ago	465MB
php	latest	74deaab6a609	28 hours ago	370MB
eduardoliveros/web	latest	31d97a406c4e	10 days ago	230MB
myhtop	latest	1b5971a20796	10 days ago	421MB
myhtop	pruebatag	1b5971a20796	10 days ago	421MB
<none>	<none>	f13ac0616b7a	10 days ago	120MB
eduardoliveros/ubuntu	latest	ccc7a11d65b1	13 days ago	120MB
ubuntu	latest	ccc7a11d65b1	13 days ago	120MB
ubuntu	14.04	c69811d4e993	13 days ago	188MB
eduardoliveros/get-started	part1	ecbe05d75fde	2 weeks ago	194MB
friendlyhello	latest	ecbe05d75fde	2 weeks ago	194MB
myjenk	latest	ecbe05d75fde	2 weeks ago	194MB
php	7.1-apache	a47f065418d5	2 weeks ago	391MB
php	<none>	84c6a05223c5	2 weeks ago	370MB

Figura 8. Pantallazo Campos de configuración Amazon S3

Fuente: elaboración propia

En síntesis...

Codeship es una excelente herramienta para la generación de pruebas, control y distribución de código. En esta lectura se observa el uso de las funcionalidades básicas de la herramienta, es importante tener en cuenta que existen diferentes planes de pago que pueden fortalecer el desarrollo de los proyectos. Por otro lado, no olvidemos que herramientas como Jenkins y TravisCi son aplicaciones que cumplen funciones similares, y que podemos encontrar decenas de herramientas que solucionan y facilitan la aplicación de la integración continua.



Referencias bibliográficas

Laguiar. (8 de 5 de 2016). wiki.genexus.com. Recuperado el 6 de 7 de 2017, de GeneXus Community Wiki: [https://wiki.genexus.com/commwiki/servlet/wiki?29683,Manejo+de+ambientes+](https://wiki.genexus.com/commwiki/servlet/wiki?29683,Manejo+de+ambientes+(Desarrollo%2C+Testeo%2C+PreProduccion%2C+Producci%C3%B3n),)

(Desarrollo%2C+Testeo%2C+PreProduccion%2C+Producci%C3%B3n),

martinfowler. (01 de 05 de 2006). martinfowler.com. Recuperado el 02 de 07 de 2017, de martinfowler.com: <https://martinfowler.com/articles/continuousIntegration.html>

Mateo, F. G. (21 de 06 de 2007). Estructura manejo de repositorios trunk branch. Recuperado el 23 de 08 de 2017, de um.s: <https://www.um.es/atica/documentos/NORPuestaEnLineaAplicacionesWeb.pdf>

s.a. (s.f.). Documentacion Git. Recuperado el 23 de 08 de 17, de Git: <https://git-scm.com/doc>

INFORMACIÓN TÉCNICA



FACULTAD DE
**INGENIERÍA, DISEÑO
E INNOVACIÓN**

Módulo: Énfasis Profesional I (Integración continua)

Unidad 4: Herramientas de despliegue automatizado

Escenario 8: Codeship, deployment pipelines y browser testing

Autor: Eduardo Enrique Oliveros Acosta

Asesor Pedagógico: Amparo Sastoque Romero

Diseñador Gráfico: Paola Andrea Melo

Asistente: Eveling Peñaranda

Este material pertenece al Politécnico Gran Colombiano. Por ende, es de uso exclusivo de las Instituciones adscritas a la Red Ilumino. Prohibida su reproducción total o parcial.