Beginners guide to deployments

@mheap at #phpworld

Let’s talk deployment

@mheap

Your deploys should be as boring, straightforward, and stress-free as possible.

- Zach Holman
(https://zachholman.com/posts/deploying-software)

@mheap

How do you deploy?

@mheap

Edit on the server?

@mheap

FTP?

@mheap

git pull?

@mheap

Third party deployment?

@mheap

Deployment script?

@mheap

OS packages?

@mheap

These are all valid ways to deploy

@mheap

Deployment for beginners

@mheap

Your deploys should be as boring, straightforward, and stress-free as possible.

- Zach Holman
(https://zachholman.com/posts/deploying-software)

@mheap

1. What is a deployment?
2. Option A: External tools
3. Option B: Internal tools
4. Option C: System packages
5. Option D: Blue / Green deployments
6. Option E: Immutable infrastructure
7. Deployment techniques
8. Summary

@mheap

What is a deployment?

@mheap

Code

@mheap

Dependencies

@mheap

Configuration

@mheap

Database Migrations

@mheap

Other Migrations

@mheap

Assets

@mheap

Post Install Tasks

@mheap

We’re aiming for…

@mheap

Your deploys should be as boring, straightforward, and stress-free as possible.

- Zach Holman
(https://zachholman.com/posts/deploying-software)

@mheap

Our Codebase

@mheap

Simple to do list application

@mheap

Composer

Silex

Doctrine Migrations

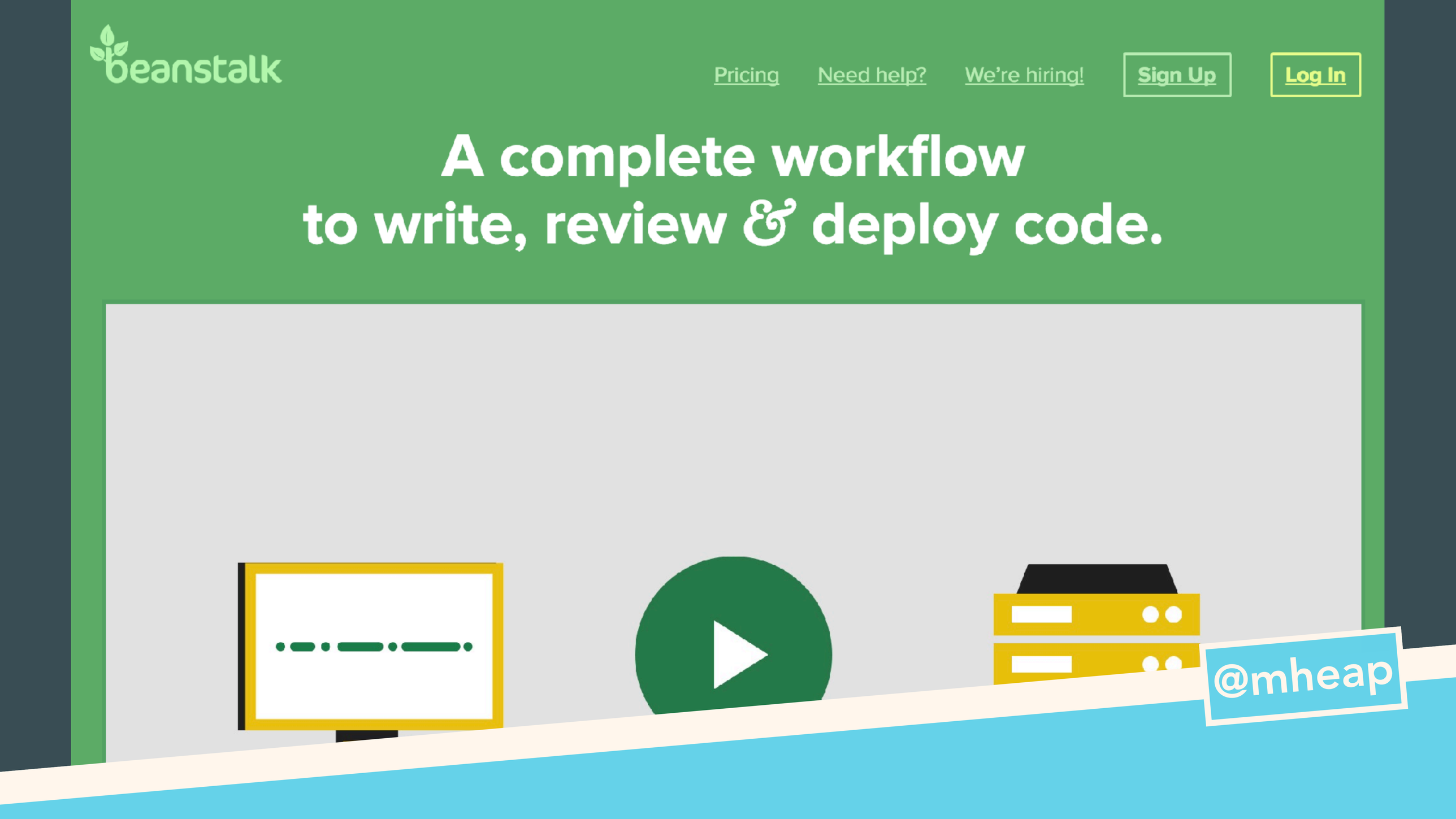
@mheap

Time to deploy!

@mheap

Option A: External tool

@mheap


beanstalk
Pricing
Need help?
We're hiring!
Sign Up
Log In
A complete workflow
to write, review & deploy code.
@mheap

Create a repository

beanstalk

Repository title:

todo-app

No color label

You can always change the title later.

Git | Subversion | What's best for you?

Repository Path

todo-app

https://intechrity.git.beanstalkapp.com/ todo-app .git

Use only lowercase letters, numbers and dashes. Keep it short and simple.

Next step or Go back

@mheap

Create an environment

beanstalk

Environment title:

No color label

Names are limited to 50 characters. *What is an environment?*

Deployment mode

Learn about deployments workflow and best practices to improve your process.

Manual

Deployment can be initiated through our web interface, or by entering a `deploy` command like `[deploy: production]` in the commit message. Works best for important environments, like production.

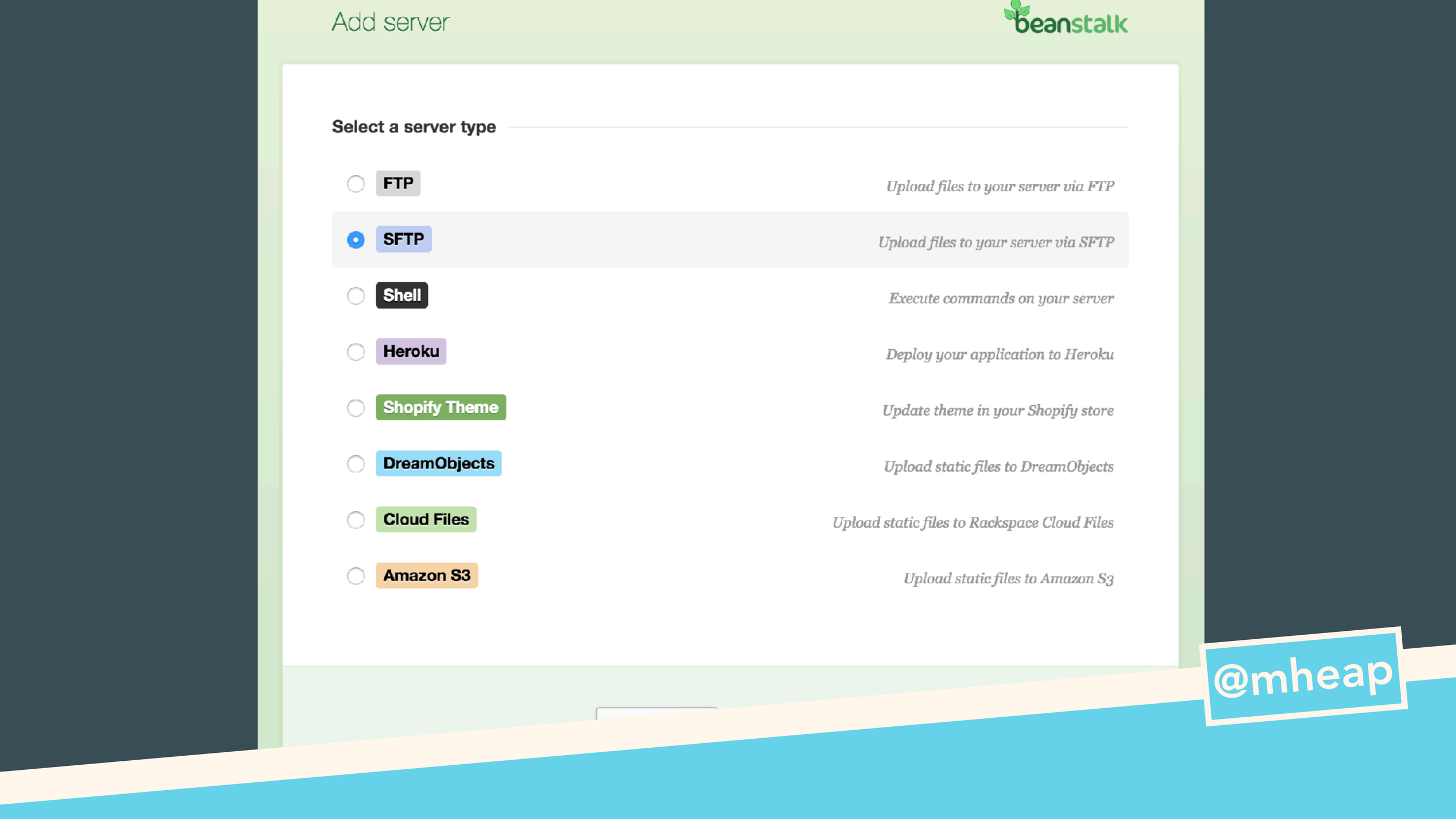
Automatic

Beanstalk will automatically deploy every commit. Most handy for staging and development environments. **Never deploy to a production environment automatically!**

Branch

Branch master

@mheap


Add server
beanstalk
Select a server type
FTP
Upload files to your server via FTP
SFTP
Upload files to your server via SFTP
Shell
Execute commands on your server
Heroku
Deploy your application to Heroku
Shopify Theme
Update theme in your Shopify store
DreamObjects
Upload static files to DreamObjects
Cloud Files
Upload static files to Rackspace Cloud Files
Amazon S3
Upload static files to Amazon S3
@mheap

General settings

Name: Server Name

Path in repository: /

Revision: None

When should I change this value?

Deployment location

Protocol: SFTP

Port: 22

Host: todo.example.com

Remote path: /var/www/todo

Beanstalk will try to create it if it's not present. Get help on paths

@mheap

Exclude files & directories OPTIONAL

Beanstalk can ignore files and directories from your repository during the deployment. This is useful for temporary, media or config files that you would like to skip or add manually.

i.e. tmp or config/production.conf

Add each pattern, file or directory on a new line. See example patterns.

Post-deployment commands OPTIONAL

Beanstalk can run the following shell commands on your server after it finished uploading the changes. If one of these commands fails, the deployment will be marked as Failed. Each command is executed in the context of the Remote Path.

Keep in mind, you are responsible for the result of the commands entered above. Please be careful and double check the commands before continuing.

```
cd /var/www/todo && composer install --no-dev
./vendor/bin/doctrine-migrations migrations:migrate --db-configuration config/db-
config.php --configuration config/migrations.yml
```

@mheap

Deploy to Production

beanstalk

1 **Select commit**

2 Review & deploy

Environment ○ **Production**

Commit **6cc1b6cb:** Initial Commit

Advanced options

Review or **Go back**

@mheap

Review & deploy

You're about to deploy to the **Production** environment. There are servers waiting for initial deployment which will be updated during this action. Depending on the size of the changes, deployment could take some time.

Server	Commit range	Commits
SFTP **Server Name**	From scratch to 6cc1b6cb	1

Preview the files to be deployed

Shell commands to run on Server Name server

```
cd /var/www/todo && composer install --no-dev
./vendor/bin/doctrine-migrations migrations:migrate --db-configuration config/db-config.php --c
```

What are you deploying?

Keep your team informed by including a clear and detailed deployment message.

Initial Commit

@mheap

Production Environment | Activity | Servers & Settings

Deploy

Automatic

Sunday • **Today**

12:05 AM  **Michael H.** deployed **6cc1b6cb**: Initial Commit — 1 commit • 10 files

12:02 AM  **Michael H.** attempted to deploy **6cc1b6cb**: Initial Commit FAILED

Transfer Log | The Incident | Retry

@mheap

Production Environment | Activity | Servers & Settings

Deploy

Automatic

Sunday • **Today**

12:11 AM  **3df0866c**: Change message — 1 commit • 1 file

12:05 AM  **Michael H.** deployed **6cc1b6cb**: Initial Commit — 1 commit • 10 files

12:02 AM  **Michael H.** attempted to deploy **6cc1b6cb**: Initial Commit FAILED

Transfer Log | The Incident | Retry

@mheap

Useful tools

Beanstalkapp
DeployHQ
Laravel Forge

@mheap

Pros

1. Easy setup
2. One-click deployments
3. Environment support
4. Can only deploy from repo

Cons

1. Files copied one by one
2. Composer run on server
3. No build steps
4. Their way, or the highway
5. Can’t deploy if provider is offline

@mheap

Option B: Internal tool

@mheap

Capistrano

Search

Search terms

Hosted Capistrano

Try The New Cloud Capistrano

Learn more

Overview

What is Capistrano?

Getting Started

The Readme, start here!

Installation

Structure

Configuration

Configuration

Location

Configuration variables can be either global or specific to your stage.

- global
  - **config/deploy.rb**
- stage specific
  - **config/deploy/<stage_name>.rb**

Access

@mheap

```
$ gem install capistrano
```

@mheap

```
$ cap install
=> mkdir -p config/deploy
=> create config/deploy.rb
=> create config/deploy/staging.rb
=> create config/deploy/production.rb
=> mkdir -p lib/capistrano/tasks
=> create Capfile
=> Capified
```

@mheap

```
$ cat config/deploy.rb

lock '3.6.1'
set :application, 'silex_todo'
set :repo_url, 'git@github.com:mheap/fake-todo-app.git'
set :deploy_to, '/var/www/todo'
```

@mheap

```
$ cat config/deploy/production.rb

server 'todo.example.com', user: 'deploy', roles: %w{app db}
```

@mheap

```
$ cap production deploy
00:00 git:wrapper
      01 mkdir -p /tmp
      02 chmod 700 /tmp/git-ssh-silex_todo-production-michael.sh
00:07 git:check
      01 git ls-remote --heads git@github.com:mheap/silex-todo-demo.git
00:08 deploy:check:directories
      01 mkdir -p /var/www/todo/shared /var/www/todo/releases
00:08 git:clone
      01 git clone --mirror git@github.com:mheap/silex-todo-demo.git /var/www/todo/repo
      01 Cloning into bare repository '/var/www/todo/repo'...
00:10 git:update
      01 git remote update --prune
      01 Fetching origin
00:11 git:create_release
      01 mkdir -p /var/www/todo/releases/20161106090926
      02 git archive master | tar -x -f - -C /var/www/todo/releases/20161106090926
00:12 git:set_current_revision
      01 echo "6cc1b6cbc827e205b24d0d599ea7b667a3ec5ca0" >> REVISION
00:12 deploy:symlink:release
      01 ln -s /var/www/todo/releases/20161106090926 /var/www/todo/releases/current
      02 mv /var/www/todo/releases/current /var/www/todo
00:12 deploy:log_revision
      01 echo "Branch master (at 6cc1b6cbc827e205b24d0d599ea7b667a3ec5ca0) deployed as
         release 20161106090926 by michael" >> /var/www/todo/revisions.log
```

@mheap

```
$ cap production deploy
```

Generate and upload deployment script
Clone repo
Generate archive from repo
Untar archive into releases folder
Symlink current folder to the latest release

@mheap

```
$ gem install capistrano-composer
```

In Capfile:

```
require 'capistrano/composer'
```

@mheap

```
$ cap production deploy

00:18 composer:run
      01 composer install --no-dev --prefer-dist --no-interaction --
quiet --optimize-autoloader
    ✔ 01 deploy@todo.example 1.023s
```

@mheap

lib/capistrano/tasks/database_tasks.rake:

```
namespace :database do
    desc 'migrate database'
    task :migrate do
        on roles(:db) do
            execute "cd #{release_path} && ./vendor/bin/
doctrine-migrations migrations:migrate --db-configuration
config/db-config.php --configuration config/migrations.yml"
        end
    end
end
```

@mheap

lib/capistrano/tasks/database_tasks.rake:

```
namespace :database do
    desc 'migrate database'
    task :migrate do
        on roles(:db) do
            execute "cd #{release_path} && ./vendor/bin/
doctrine-migrations migrations:migrate --db-configuration
config/db-config.php --configuration config/migrations.yml"
        end
    end
end
```

@mheap

lib/capistrano/tasks/database_tasks.rake:

```
namespace :database do
    desc 'migrate database'
    task :migrate do
        on roles(:db) do
            execute "cd #{release_path} && ./vendor/bin/
doctrine-migrations migrations:migrate --db-configuration
config/db-config.php --configuration config/migrations.yml"
        end
    end
end
```

@mheap

lib/capistrano/tasks/database_tasks.rake:

```
namespace :database do
    desc 'migrate database'
    task :migrate do
        on roles(:db) do
            execute "cd #{release_path} && ./vendor/bin/
doctrine-migrations migrations:migrate --db-configuration
config/db-config.php --configuration config/migrations.yml"
        end
    end
end
```

@mheap

lib/capistrano/tasks/database_tasks.rake:

```
namespace :database do
    desc 'migrate database'
    task :migrate do
        on roles(:db) do
            execute "cd #{release_path} && ./vendor/bin/
doctrine-migrations migrations:migrate --db-configuration
config/db-config.php --configuration config/migrations.yml"
        end
    end
end
```

@mheap

Capfile:

```
after "composer:install", "database:migrate"
```

@mheap

```
$ cap production deploy

00:22 database:migrate
      01 cd #{release_path} && ./vendor/bin/doctrine-migrations
migrations:migrate —db-co…
    ✔ 01 deploy@todo.example 0.812s
```

@mheap

```
root@deploy-demo:/var/www/todo# tree -L 2
.
├── current -> /var/www/todo/releases/20161106164234
├── releases
│   ├── 20161106163035
│   ├── 20161106164220
│   └── 20161106164234
├── repo
│   ├── branches
│   ├── config
│   ├── description
│   ├── FETCH_HEAD
```

@mheap

Useful tools

Capistrano
Webistrano
Rocketeer
Ansible

@mheap

Pros

1. One-click deployments
2. Environment support
3. Can only deploy from repo
4. Internal solution
5. Atomic deploys
6. Rollback support

Cons

1. Learning curve
2. Composer run on server
3. No build steps
4. Probably want to use Bundler to manage versions

@mheap

Option C: System Packages

@mheap

```
$ apt-get install todo
$ yum install todo
$ pacman -S todo
```

@mheap

```
242 Provides: php(httpd)
243
244 %if 0%{?fedora} < 20 && 0%{?rhel} < 7
245 # Don't provides extensions, which are not shared library, as .so
246 %{?filter_provides_in: %filter_provides_in %{_libdir}/php/modules/.*\.so$}
247 %{?filter_provides_in: %filter_provides_in %{_libdir}/php-zts/modules/.*\.so$}
248 %{?filter_provides_in: %filter_provides_in %{_httpd_moddir}/.*\.so$}
249 %{?filter_setup}
250 %endif
251
252
253 %description
254 PHP is an HTML-embedded scripting language. PHP attempts to make it
255 easy for developers to write dynamically generated web pages. PHP also
256 offers built-in database integration for several commercial and
257 non-commercial database management systems, so writing a
258 database-enabled webpage with PHP is fairly simple. The most common
259 use of PHP coding is probably as a replacement for CGI scripts.
260
261 The php package contains the module (often referred to as mod_php)
262 which adds support for the PHP language to Apache HTTP Server.
263
264 %package cli
265 Group: Development/Languages
266 Summary: Command-line interface for PHP
267 Requires: php-common%{?_isa} = %{version}-%{release}
268 Provides: php-cgi = %{version}-%{release}, php-cgi%{?_isa} = %{version}-%{release}
269 Provides: php-pcntl, php-pcntl%{?_isa}
270 Provides: php-readline, php-readline%{?_isa}
271 Obsoletes: php53-cli, php53u-cli
272
```

@mheap

jordansissel / fpm

Watch 273 | Star 7,109 | Fork 728

Code | Issues 353 | Pull requests 33 | Projects 1 | Wiki | Pulse | Graphs

Effing package management! Build packages for multiple platforms (deb, rpm, etc) with great ease and sanity.

1,936 commits | 12 branches | 71 releases | 189 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

jordansissel committed on GitHub Merge pull request #1208 from liger1978/fix_author ... Latest commit 4c17c11 9 days ago

bin	- Folks gettin' angry about an abandoned prototype they mistook for	4 years ago
examples	Fix makefile and add the missing 'git fetch'	2 years ago
lib	Fixes #1206 Improved download URL query for CPAN modules	20 days ago
misc	removed all trailing whitespace: for i in $(git ls-files); do sed -i ...	5 years ago
spec	Move fpm version query behavior to `run` method.	2 months ago
templates	remove trailing slash from --prefix option in rpm-packages, fixes #819	
test	Use `--xz` in pacman's tar, update Vagrantfile	
.gitignore		

@mheap


.
├── resources
├── src
│   ├── application
│   │   ├── controller
│   │   │   └── IndexController.php
│   │   ├── model
│   │   │   └── User.php
│   │   └── view
│   │   └── show-user.php
│   ├── composer.json
│   ├── composer.lock
│   ├── config
│   │   ├── mappings.json
│   │   └── targets.json
│   └── public
│   └── index.php
└── test

@mheap

```
.
├── resources
├── src
|   ├── application
|   |   ├── controller
|   |   |   └── IndexController.php
|   |   ├── model
|   |   |   └── User.php
|   |   └── view
|   |       └── show-user.php
|   ├── composer.json
|   ├── composer.lock
|   ├── config
|   |   ├── mappings.json
|   |   └── targets.json
|   └── public
|       └── index.php
└── test
```

```
$ fpm -s dir -t rpm -n todo
```

@mheap

```
.
├── resources
├── src
|   ├── application
|   |   ├── controller
|   |   |   └── IndexController.php
|   |   ├── model
|   |   |   └── User.php
|   |   └── view
|   |       └── show-user.php
|   ├── composer.json
|   ├── composer.lock
|   ├── config
|   |   ├── mappings.json
|   |   └── targets.json
|   └── public
|       └── index.php
└── test
```

```
$ fpm -s dir -t rpm -n todo \
 src/=/var/www/todo \
 src/config/=/etc/todo
```

@mheap

```
.
├── resources
├── src
|   ├── application
|   |   ├── controller
|   |   |   └── IndexController.php
|   |   ├── model
|   |   |   └── User.php
|   |   └── view
|   |       └── show-user.php
|   ├── composer.json
|   ├── composer.lock
|   ├── config
|   |   ├── mappings.json
|   |   └── targets.json
|   └── public
|       └── index.php
└── test
```

```
$ fpm -s dir -t rpm --config-files /etc \
 -n todo \
 src/=/var/www/todo \
 src/config/=/etc/todo
```

@mheap

```
.
├── resources
├── src
|   ├── application
|   |   ├── controller
|   |   |   └── IndexController.php
|   |   ├── model
|   |   |   └── User.php
|   |   └── view
|   |       └── show-user.php
|   ├── composer.json
|   ├── composer.lock
|   ├── config
|   |   ├── mappings.json
|   |   └── targets.json
|   └── public
|       └── index.php
└── test
```

```
$ fpm -s dir -t rpm --config-files /etc \
  --exclude "var/www/todo/config*" \
 -n todo \
 src/=/var/www/todo \
 src/config/=/etc/todo
```

@mheap

```
$ fpm -s dir -t rpm --config-files /etc  --exclude
"var/www/todo/config*" -n todo src/=/var/
www/todo src/config/=/etc/todo
```

```
=> Created package
{:path=>"todo-1.0-1.x86_64.rpm"}
```

@mheap

Continous Integration

@mheap

What is Continous Integration?

@mheap

Use it to run tests

@mheap

Use it to composer install

@mheap

Use it to mitigate risk

@mheap

Use it to
build on identical platforms

@mheap

Use it to
package your application

@mheap

Use it to
deploy to production

@mheap

Use it to
AUTOMATE THINGS

@mheap

Jenkins Downloads Blog Documentation Plugins Use-cases Participate Sub-projects Resources Security Press Conduct

Jenkins

Build great things at any scale

The leading open source automation server, Jenkins provides hundreds of plugins to support building, deploying and automating any project.

Download Jenkins

Get 2.19.2 LTS .war or the latest 2.28 weekly release

Continuous Integration and Continuous Delivery

As an extensible

Easy installation

@mheap

Option C: System Packages

@mheap

```
$ apt-get install todo
$ yum install todo
$ pacman -S todo
```

@mheap

Almost

@mheap

Database migrations

@mheap

```
$ fpm -s dir -t rpm --config-files /etc  --exclude
"var/www/todo/config*" -n todo src/=/var/
www/todo src/config/=/etc/todo
```

```
=> Created package
{:path=>"todo-1.0-1.x86_64.rpm"}
```

@mheap

```
$ fpm -s dir -t rpm --config-files /etc  --exclude
"var/www/todo/config*" -n todo
--after-install /path/to/script src/=/var/www/
todo src/config/=/etc/todo
```

```
=> Created package
{:path=>"todo-1.0-1.x86_64.rpm"}
```

@mheap

Useful tools

FPM

Jenkins

RPM / Deb / Pkg

@mheap

Releases are immutable

@mheap

Pros

1. Atomic releases
2. Build on system tools
3. Proper build toolchain
4. Signed builds
5. Easy upgrade/rollback
6. Immutable

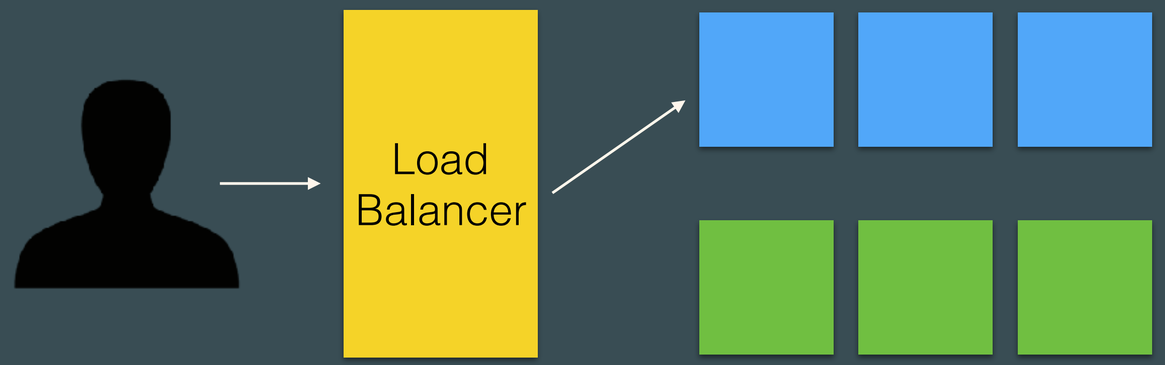
Cons

1. Steep learning curve
2. Need to run package repo
3. Ideal to run CI system

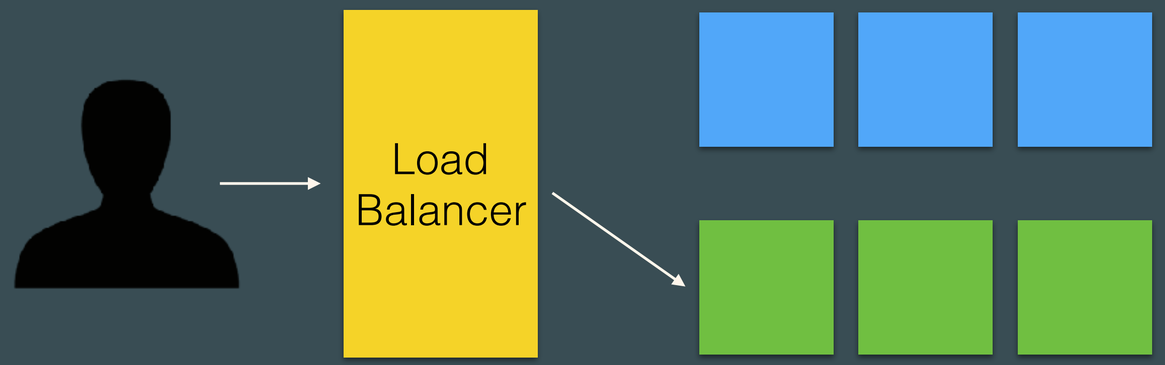
@mheap

Option D: Blue / Green

@mheap



@mheap



@mheap

Shared databases

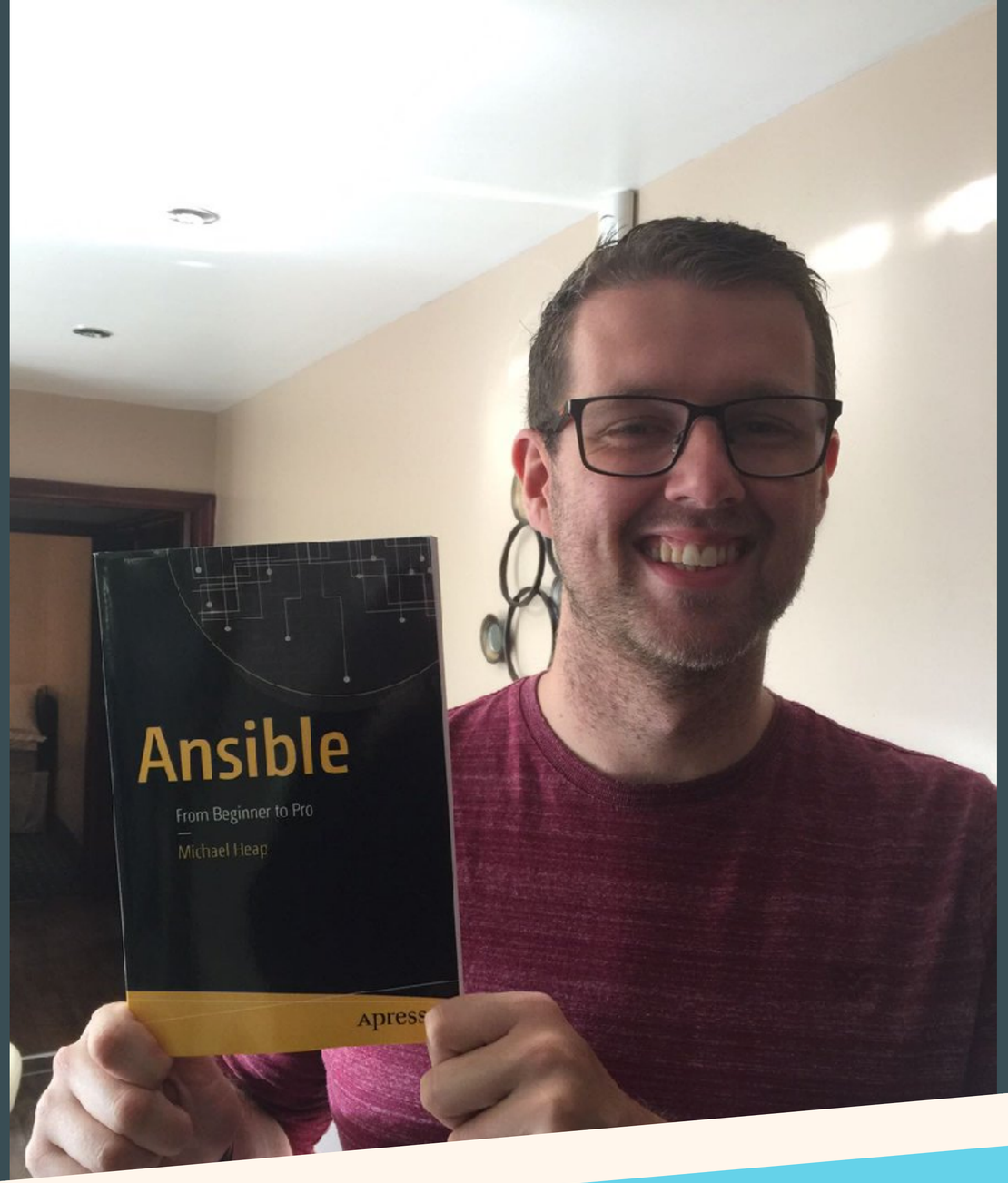
@mheap

Single database

@mheap

Server maintenance

@mheap



Shameless Plug

Infrastructure level

@mheap

Useful tools

Puppet / Chef / Ansible

HAProxy

@mheap

Pros

1. Low risk deployments
2. Test on production systems
3. Easy changeover

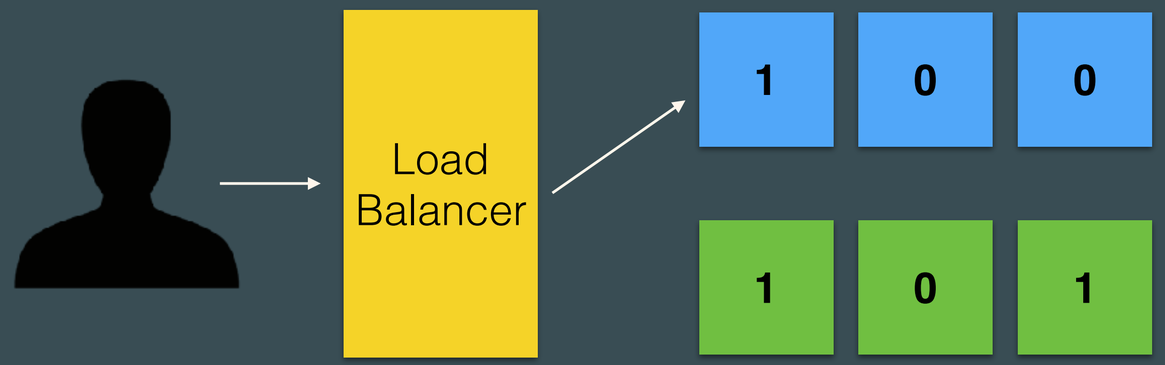
Cons

1. Can be expensive
2. Databases are hard
3. More moving parts to manage

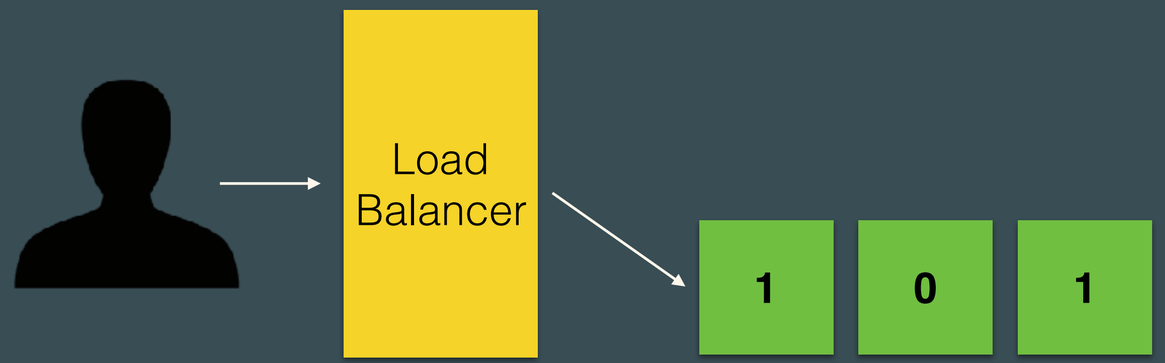
@mheap

Option E: Immutable

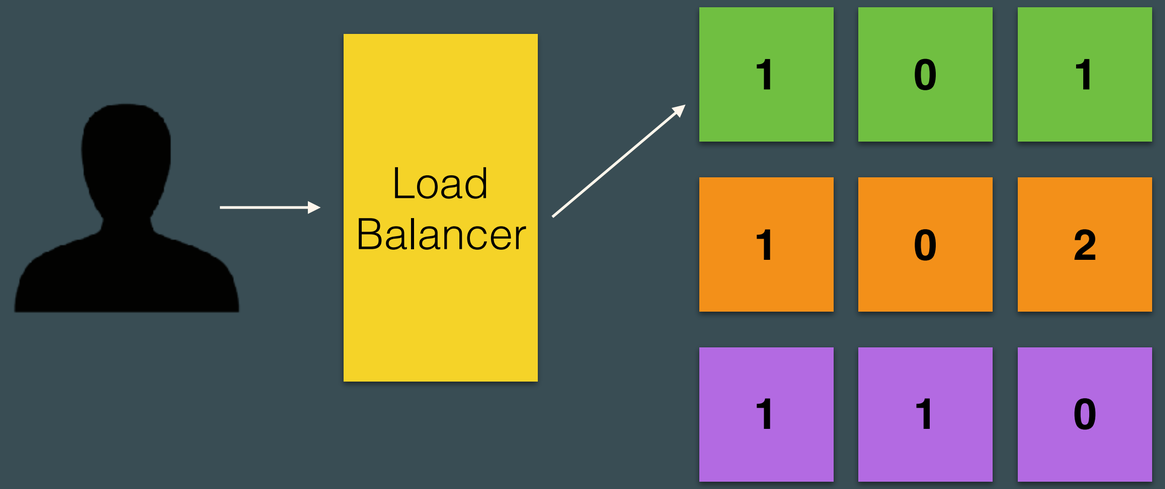
@mheap



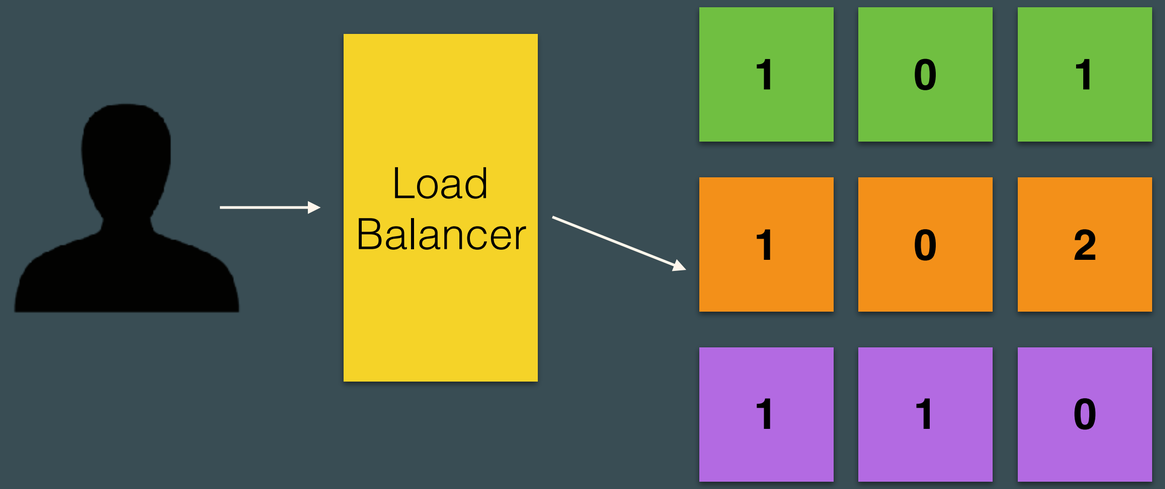
@mheap



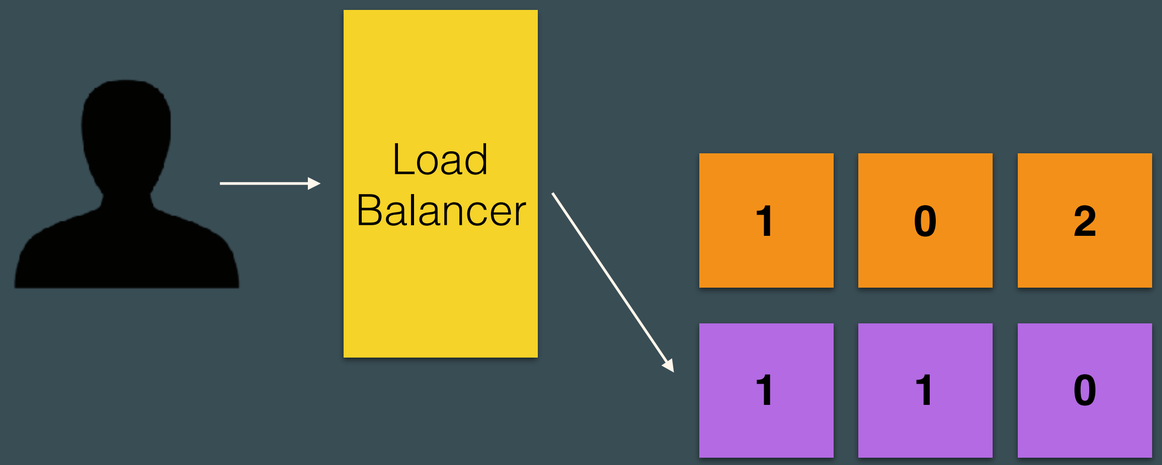
@mheap



@mheap



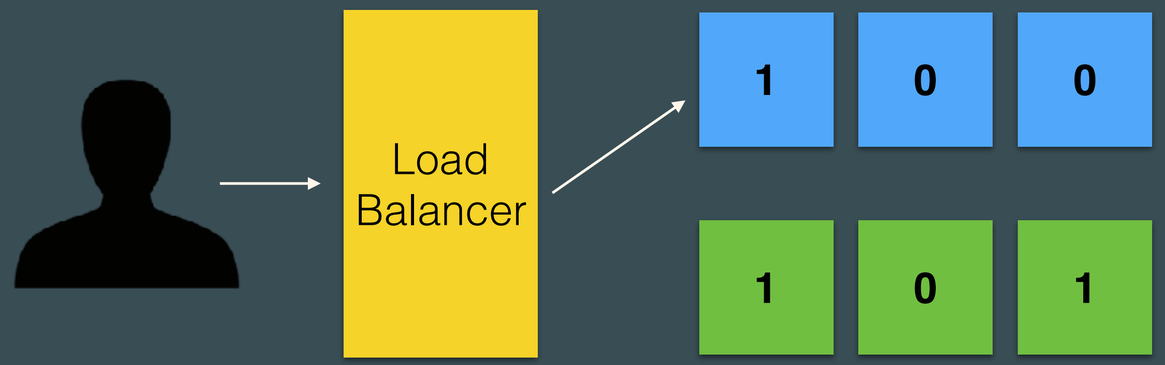
@mheap



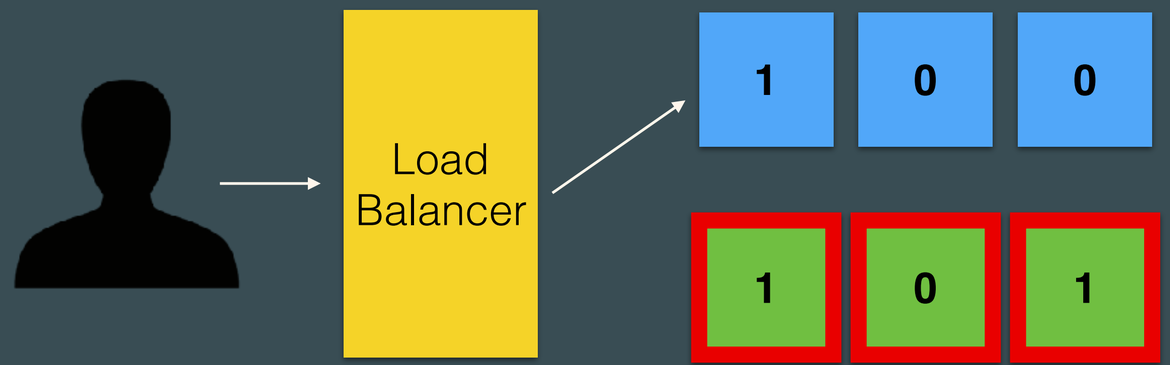
@mheap

Why immutable?

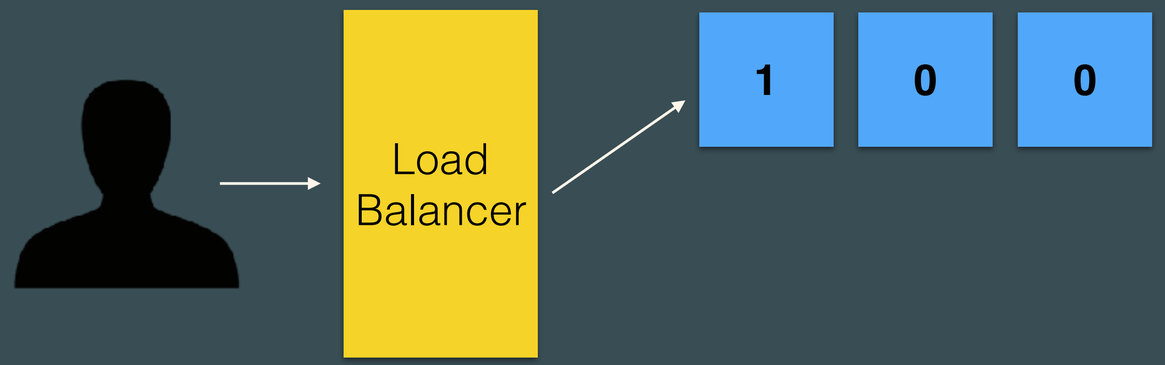
@mheap



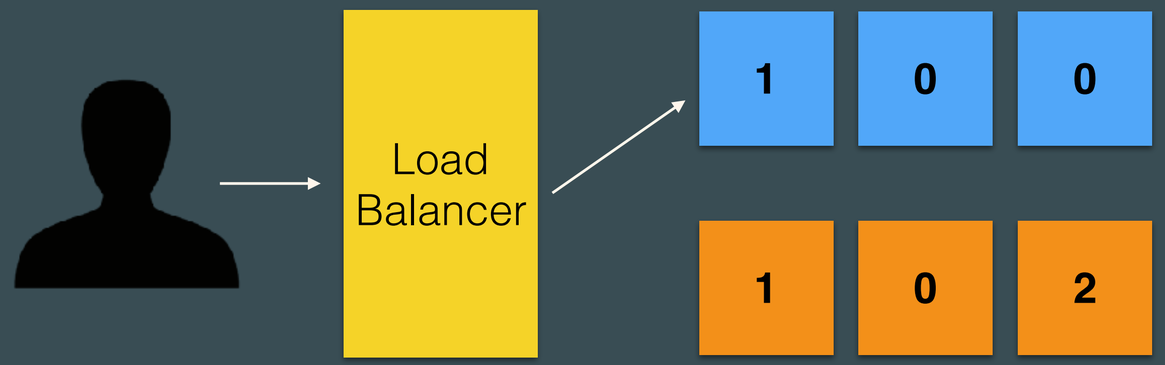
@mheap



@mheap



@mheap



@mheap

Useful tools

Puppet / Chef / Ansible
Packer
Cloud providers

@mheap

Pros

1. Known environment
2. Test 2+ releases at once
3. Easy changeover
4. Truly atomic releases

Cons

1. Cost
2. Slow to build
3. Database sync

@mheap

Deployment techniques

@mheap

Build (at least) daily

(even better if it’s automated)

@mheap

Announce releases

@mheap

Staggered Releases

@mheap

Canary deploys

@mheap

Go / No Go

@mheap

Containers

@mheap

Chatops

@mheap

Summary

@mheap

Automate your deployment

@mheap

Local builds

@mheap

Immutable releases

@mheap

Think about deployment before you write any code

@mheap

The three strikes rule

@mheap

Make it a single command

@mheap

Your deploys should be as boring, straightforward, and stress-free as possible.

- Zach Holman
(https://zachholman.com/posts/deploying-software)

@mheap

https://joind.in/18696

Michael Heap
@mheap
m@michaelheap.com

@mheap at #phpworld