Deployment Is Not A Four Letter Word

Chris Hartjes

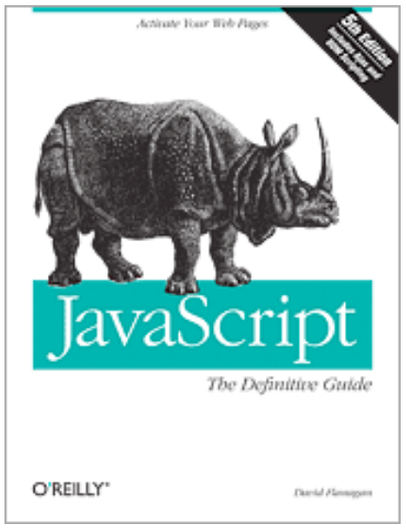
PHP Quebec 2009 Conference

WARNING: This presentation may contain language that is offensive to some. I'll try not to swear but sometimes it just comes out...

A little bit about myself...

(I work from home for)

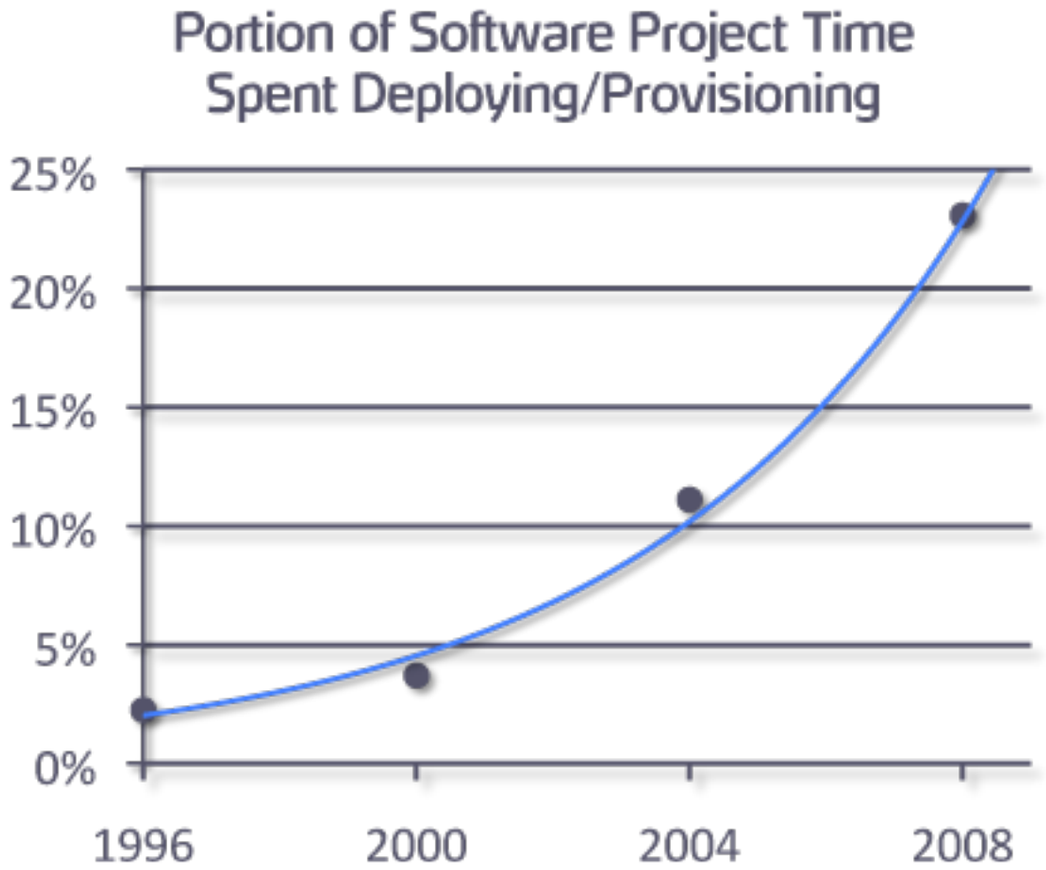
What I use every day



YOU CANNOT ESCAPE

THE FRAMEWORK APOCALYPSE

http://blog.heroku.com/archives/2009/2/23/why_instant_deployment_matters/

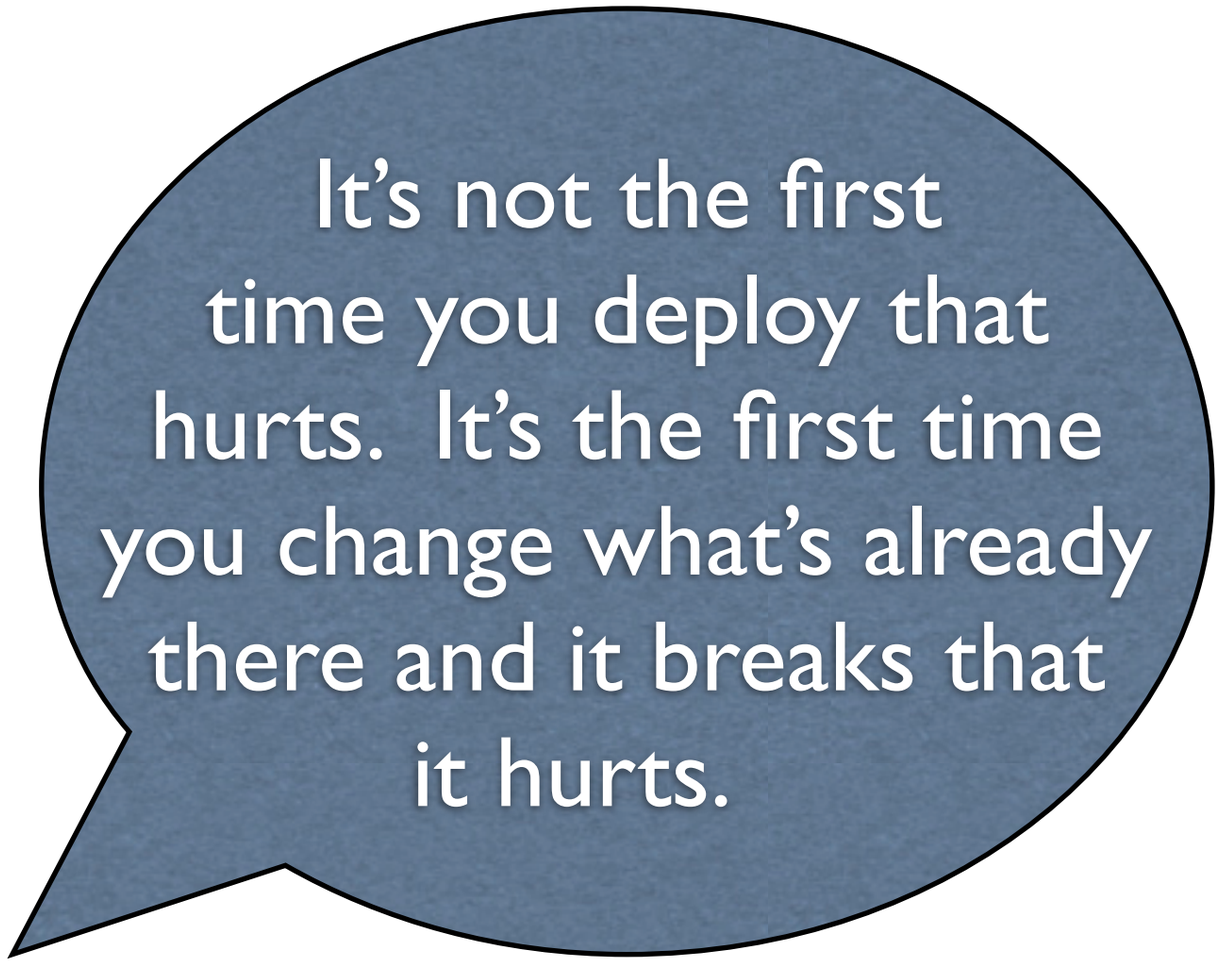


(completely unscientific but it sure feels that way)

VS.

Chris' 6 Rules for Confident Deployment

Rule #1


It's not the first time you deploy that hurts. It's the first time you change what's already there and it breaks that it hurts.

Your Deployment Tool Should Run Your Tests

(but if it can't, run them yourself)

PHPT

Sometimes testing is hard...

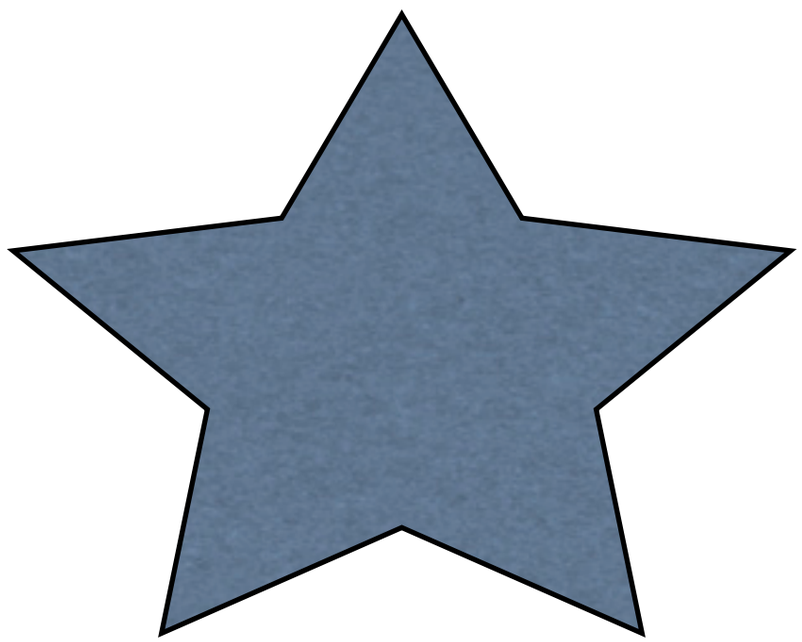
Learn how to use your VCS hooks

Rule #2

If you can change your application in production, you better be able to change it back to the way it was

Version Control lets you roll changes back

Subversion



Bazaar

Git

CVS

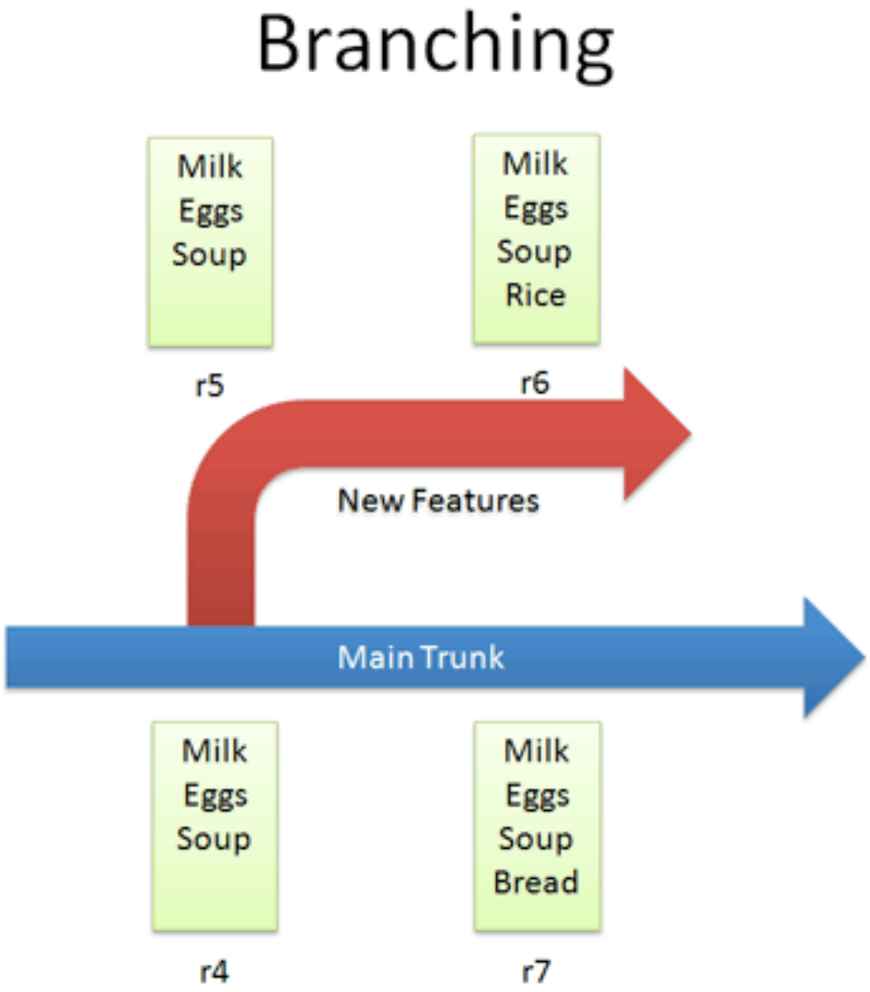
Mercurial

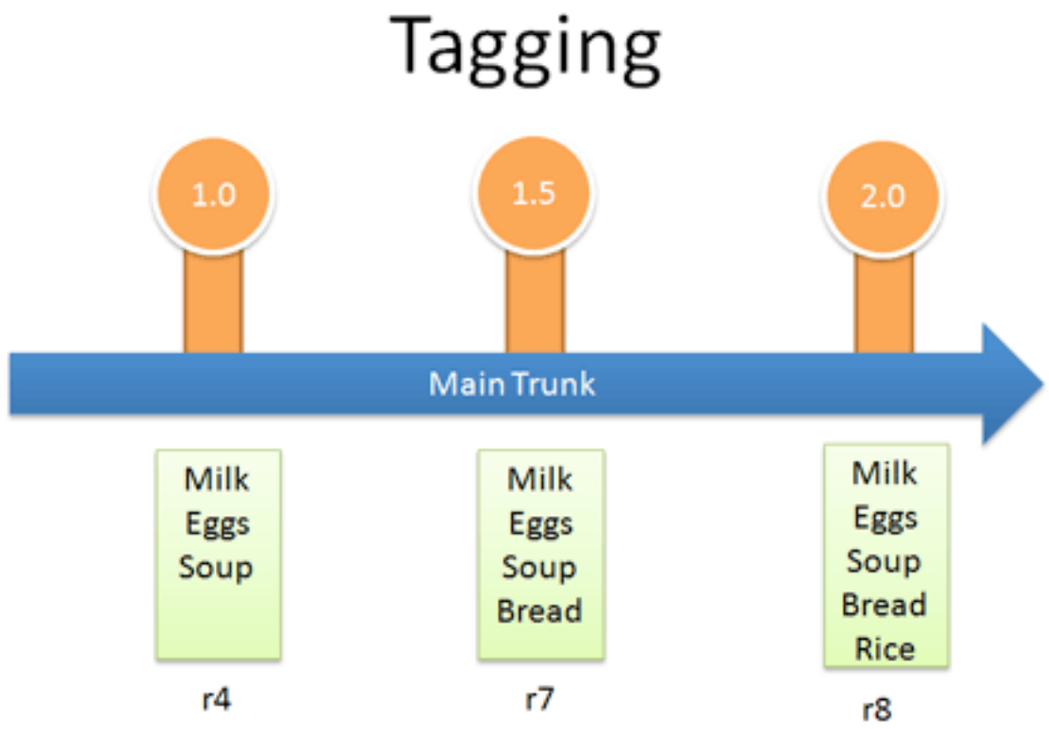
If You Can't Automate The Checkout, Stop Using It

Your deployment tool has to be able to talk to your version control...

...so you are always getting your code from the same place

Your Deployment Dynamic Duo





Rule #3

Don't forget that there is often more to your application than just code


IM IN UR DB
MAKIN CHANGE

Managing Database Changes Is Critical

Tools should make the changes for you...

... or write SQL scripts and put 'em under version control

Example of DB migrations in CakePHP

The Migration files support YAML and PHP array's. So instead of having to write SQL queries, you can write a short YAML structure that will do the same thing:

```
create_table:
  users:
    name: string
    age: int
    is_active: bool
```

or the equivalent in a PHP array:

```
array(
  'create_table' => array(
    'users' => array(
      'name' => array(
        'type' => 'string'
      )
      'age' => array(
        'type' => 'int'
      )
      'is_active' => array(
        'type' => 'bool'
      )
    )
  )
)
```

oriondata-alters.sql

```
ALTER TABLE action_american_football_play ADD COLUMN yards_gained smallint(6) default NULL
ALTER TABLE action_american_football_play_participant ADD COLUMN score_credit smallint(6) defaultNULL
ALTER TABLE action_american_football_score ADD COLUMN yards_gained smallint(6) default NULL
ALTER TABLE map_event_result ADD COLUMN rank int(10) unsigned default NULL
ALTER TABLE player_stats ADD COLUMN events_played smallint(6) default NULL
ALTER TABLE player_stats ADD COLUMN events_started smallint(6) default NULL
ALTER TABLE player_stats ADD COLUMN time_played_event smallint(6) default NULL
ALTER TABLE player_stats ADD COLUMN time_played_total smallint(6) default NULL
ALTER TABLE player_stats ADD COLUMN time_played_event_average smallint(6) default NULL
ALTER TABLE refdyn_player ADD COLUMN status varchar(20) default NULL
ALTER TABLE stats_american_football_defensive ADD COLUMN yards_allowed_total smallint(6) default NULL
ALTER TABLE stats_american_football_defensive ADD COUMN yards_allowed_average decimal(7,2) default NULL
ALTER TABLE stats_american_football_defensive ADD COLUMN defensive_plays_number smallint(6) default NULL
ALTER TABLE stats_american_football_defensive ADD COLUMN inside_20_plays smallint(6) default NULL
ALTER TABLE stats_american_football_defensive ADD COLUMN possessions_inside_20 smallint(6) default NULL
ALTER TABLE stats_american_football_defensive ADD COLUMN inside_20_points_against smallint(6) default NULL
ALTER TABLE stats_american_football_defensive ADD COLUMN inside_20_touchdowns_against smallint(6) default NULL
ALTER TABLE stats_american_football_defensive ADD COLUMN inside_20_touchdowns_against_percentage decimal(5,2) default NULL
ALTER TABLE stats_american_football_defensive ADD COLUMN defense_rank smallint(6) default NULL
ALTER TABLE stats_american_football_defensive ADD COLUMN defense_rank_pass smallint(6) default NULL
ALTER TABLE stats_american_football_defensive ADD COLUMN defense_rank_rush smallint(6) default NULL
ALTER TABLE stats_american_football_down_proogress ADD COLUMN first_downs_yards smallint(6) default NULL
ALTER TABLE stats_american_football_down_proogress ADD COLUMN first_downs_average_yards_per decimal(5,2) default NULL
ALTER TABLE stats_american_football_fumbles ADD COLUMN fumbles_own_touchdowns smallint(6) default NULL
ALTER TABLE stats_american_football_fumbles ADD COLUMN fumbles_opposing_touchdowns smallint(6) default NULL

CREATE TABLE stats_american_football_looks_and_touches (
  stat_type varchar(10) NOT NULL default '',
  event_id varchar(100) NOT NULL default '',
  player_key varchar(75) NOT NULL default '',
  team_key varchar(255) NOT NULL default '',
  season_type varchar(20) NOT NULL default '',
  season_key varchar(20) NOT NULL default '',
  week int(11) NOT NULL default '0',
  stat int(11) default '0',
  redzone_stat int(11) default '0',
  PRIMARY KEY  (stat_type,event_id,player_key,team_key)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;


ALTER TABLE stats_american_football_offensive ADD COLUMN offensive_plays_average_yards_per_game smallint(6) default NULL
ALTER TABLE stats_american_football_offensive ADD COLUMN possessions_inside_20 smallint(6) default NULL
ALTER TABLE stats_american_football_offensive ADD COLUMN inside_20_plays smallint(6) default NULL
ALTER TABLE stats_american_football_offensive ADD COLUMN offensive_rank smallint(6) default NULL
```

Rule #4

If it's not repeatable...and automated...you're wasting your time

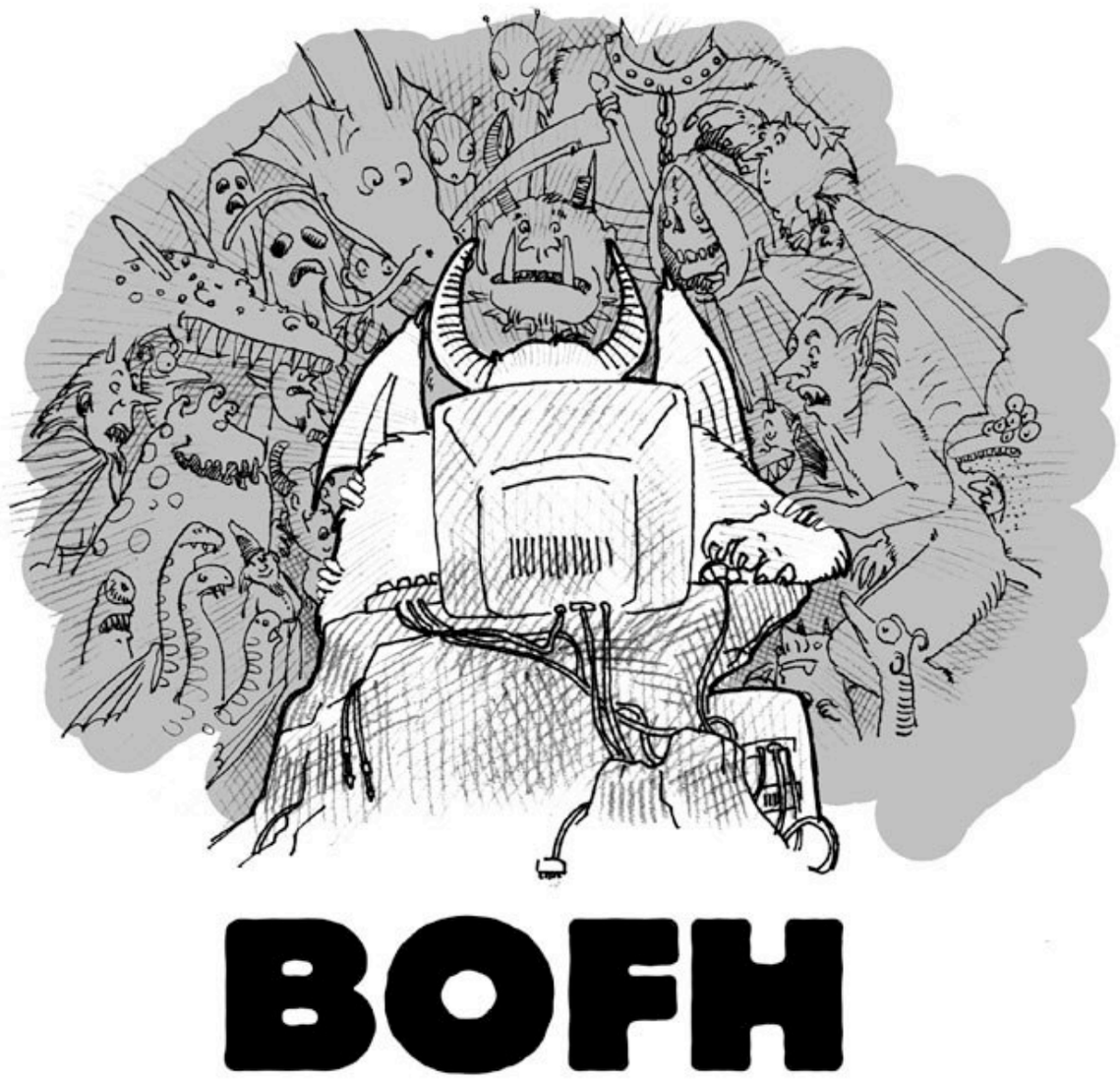
Manual Deployment is For Suckers

Automating deployment of your code...

...means one less thing for you to screw up

Automation forces consistency across environments

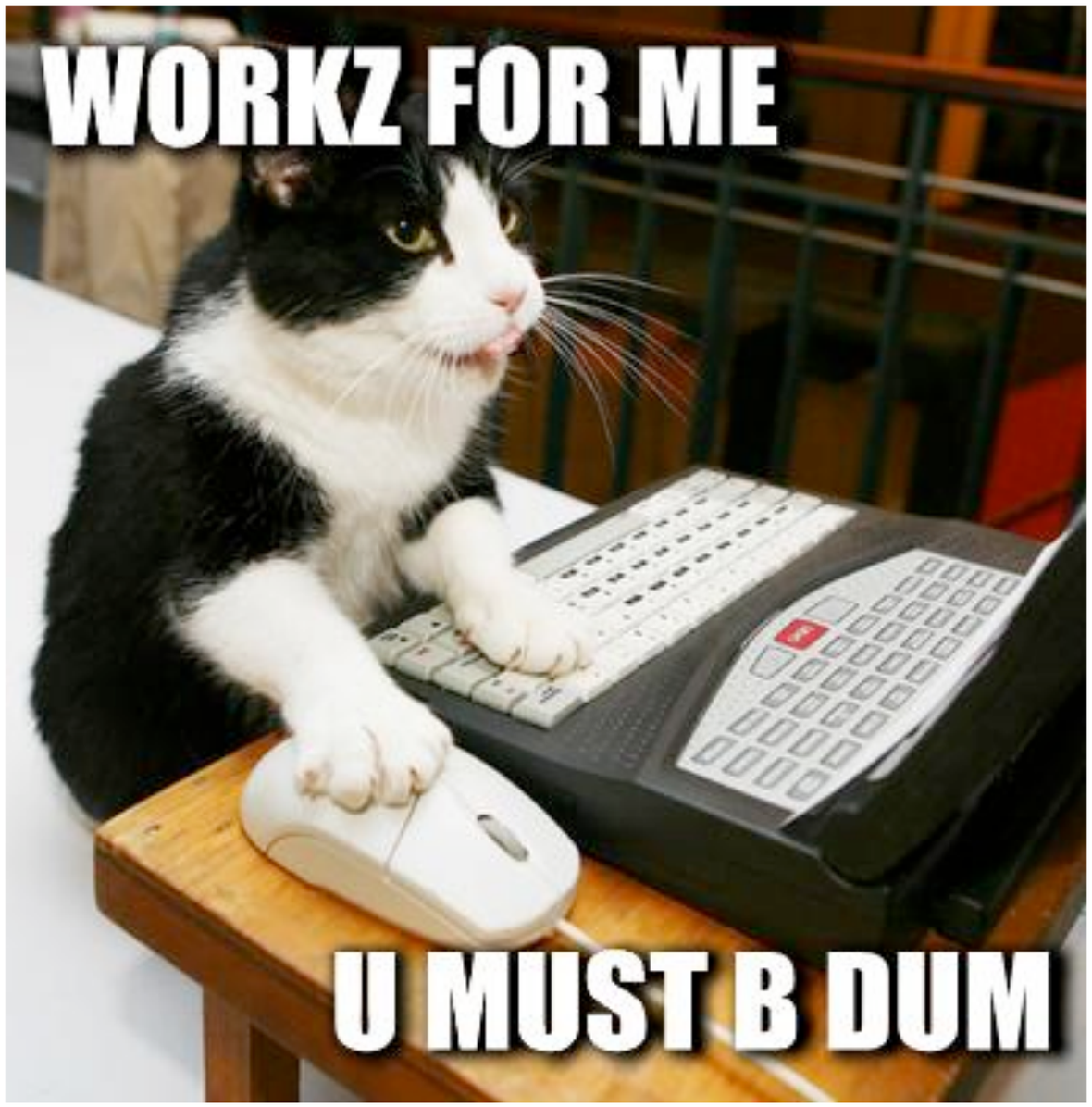
You can deploy it...



...and so can they

Rule #5

The only difference in your application's environments should be the data sets

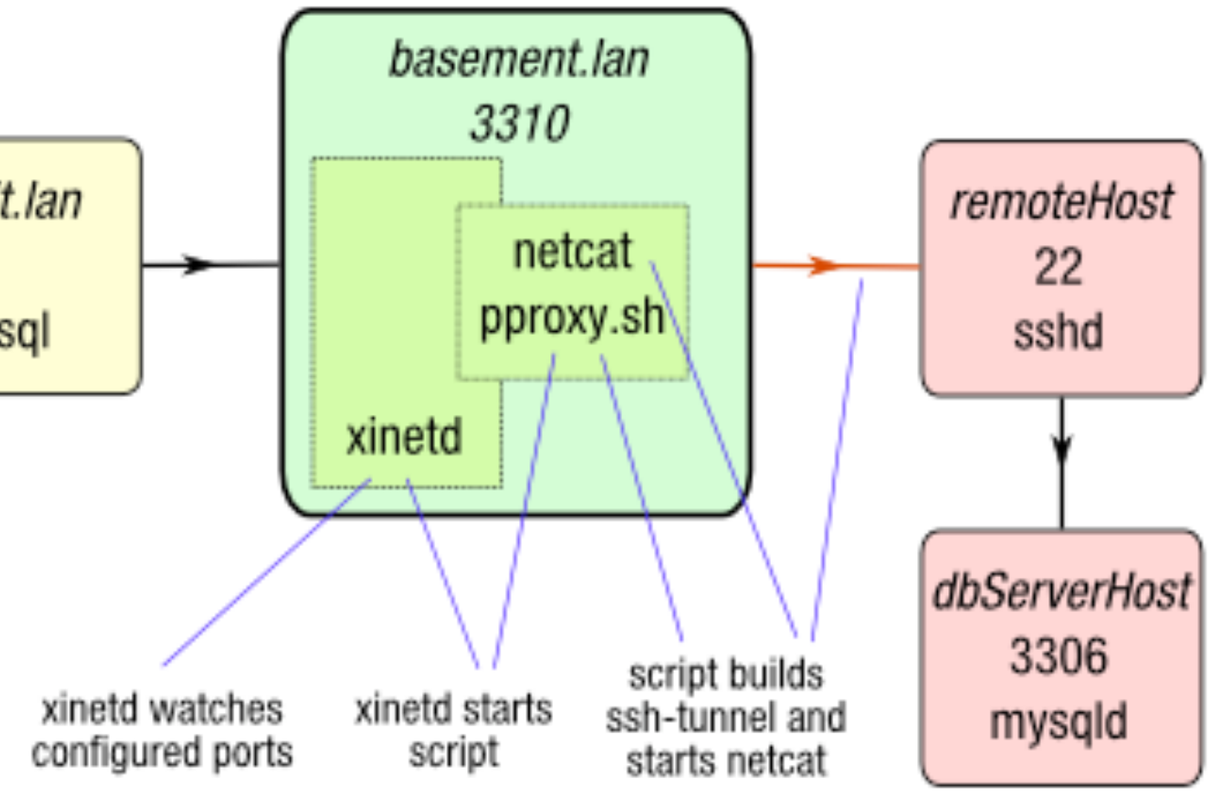

WORKZ FOR ME
U MUST B DUM

Consistency Is Not The Hobgoblin of Little Minds

Bugs that "only exist in production"...

...will never be found in development

Tunnel through to your production resources



ssh -fNg -L3310:gerrit.lan:3306 gerrit@dbServerHost

Copy sample datasets to your development environment

Rule #6

Ownership of code means also having a stake in making sure it gets deployed properly

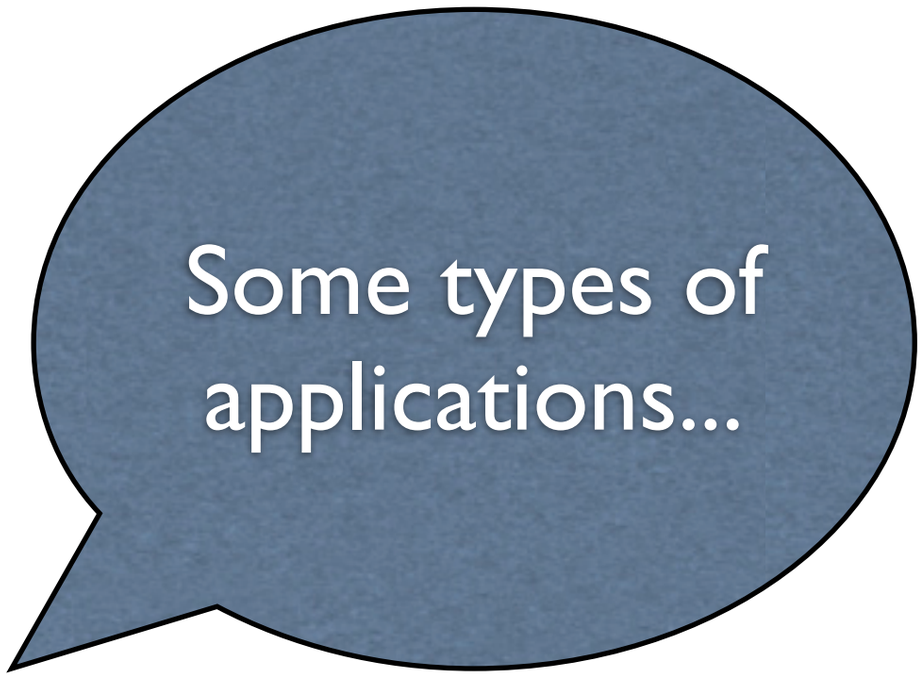
Write Your Code To Your Environment

Your changes shouldn't be so difficult...

...that you're the only one who can make them


ERLANG


seaside


Some types of
applications...

...have their own deployment processes

Roll your own is popular...

...because everyone thinks they are unique

Very popular in the Java world...

... but only works for Java

<phing>

A lot like Maven and Ant...

...but all that XML is not friendly

Vlad the Deployer

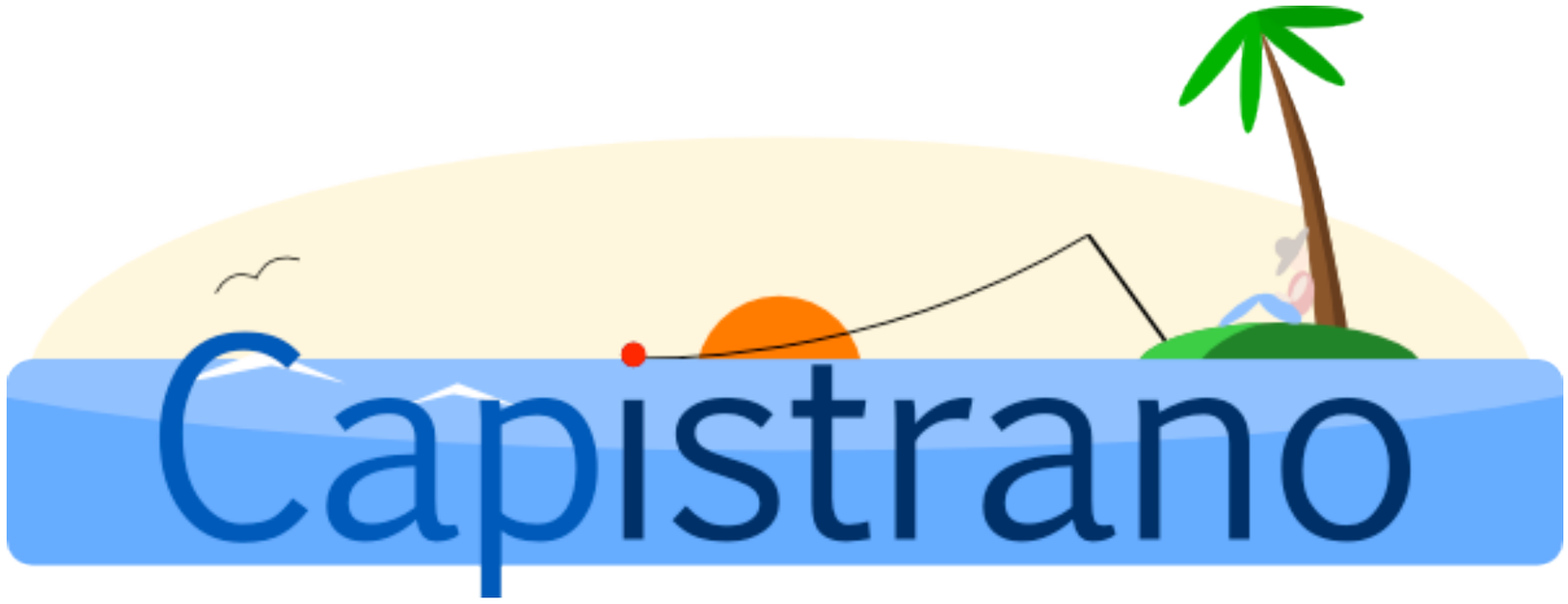
Best.
Name.
Ever.

Sadly, only for Rails-based applications

Fabric

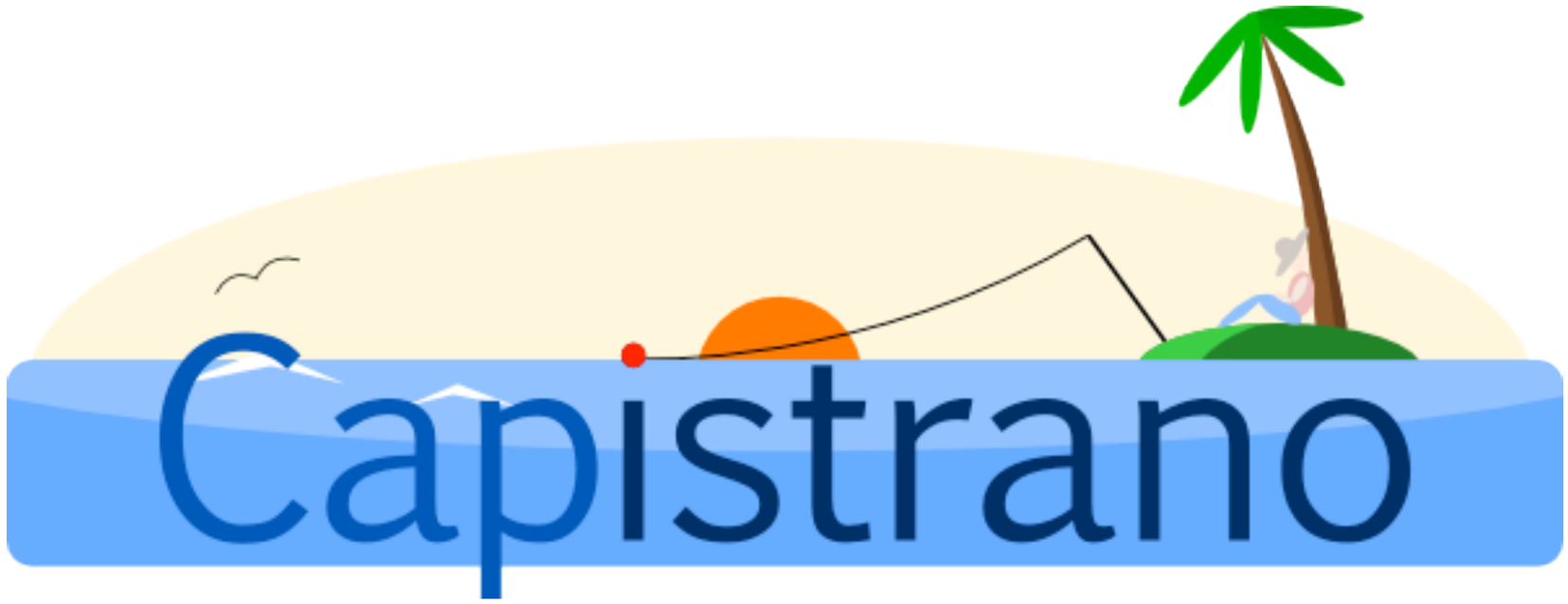
Python-based deployment...

...but documentation is lacking



Works best with Rails...

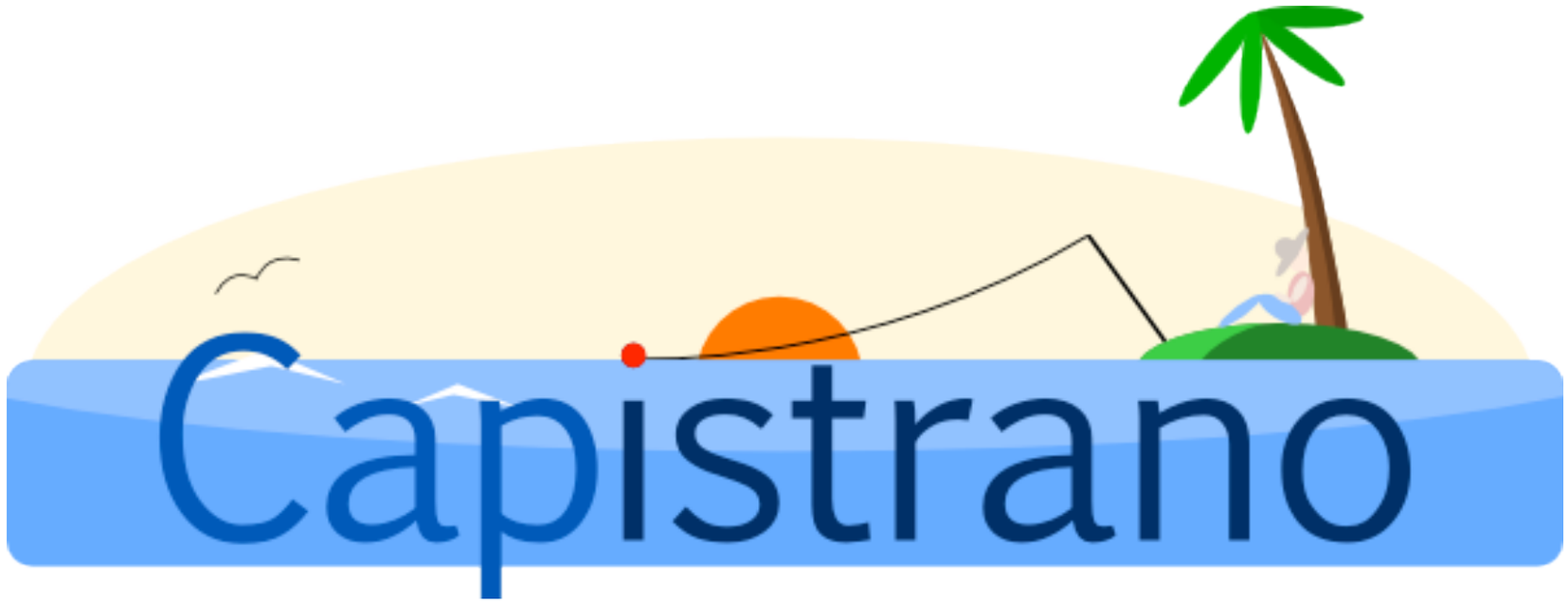
...but can deploy almost anything




```
load 'deploy' if respond_to?(:namespace) # cap2
differentiator
require 'rubygems'

set :application, "sportso.com"
set :domain, "hawks.xmlteam.com"
role :app, domain
role :web, domain
role :db,  domain, :primary => true

default_run_options[:pty] = true

# Set the Unix user and group that will actually perform
# each task on the remote server. The defaults are set to
# 'deploy'
set :user, "user"
set :group, "group"
```

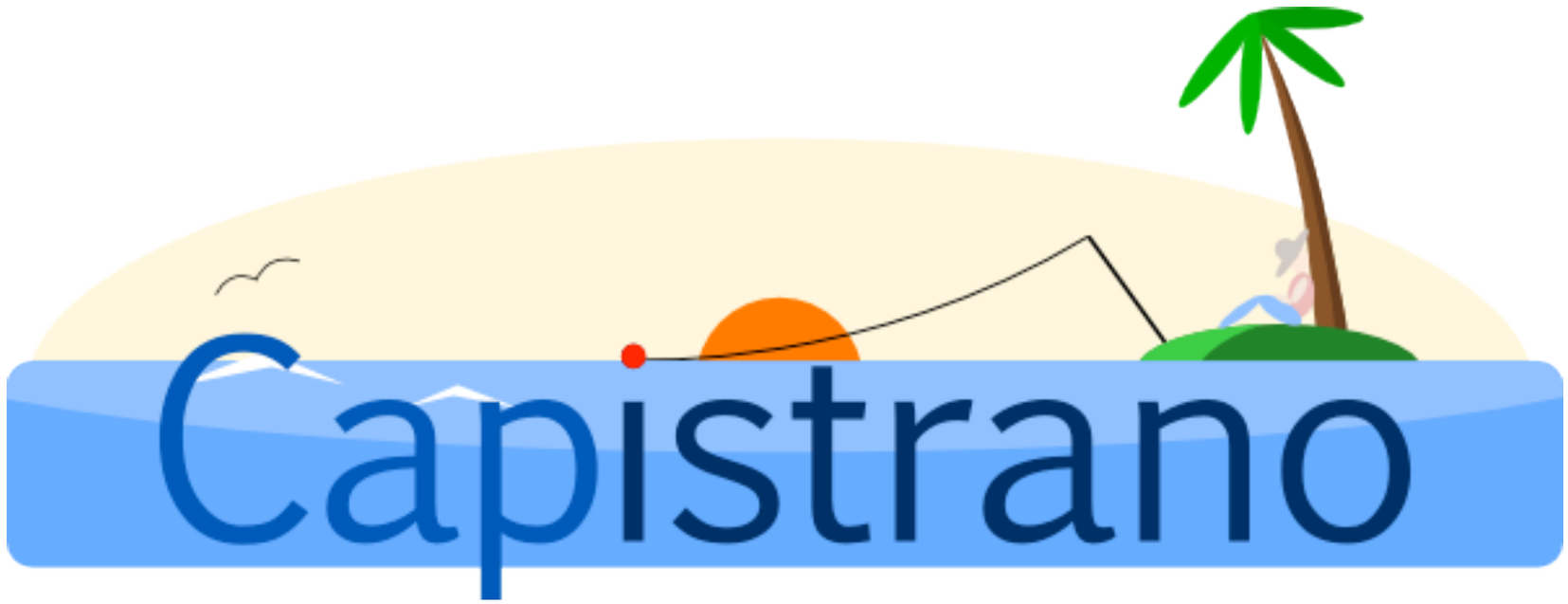
```
# Deployment Settings
set :repository,  "svn+ssh://deploy@leafs.xmlteam.com/opt/repos/sportso.com"
set :deploy_to, "/var/www/#{application}" # This is where your project will be
deployed.
set :svn_user, 'chris'
set :svn_password, 'password'

# Uncomment this if you want to deploy your project by copying
# the files from your local computer.
set :deploy_via, :copy
```

Sample code from a Capistrano deployment recipe




```
task :exist do
  run "cp /var/www/app/config/xql/*.xq* /opt/resin/webapps/exist/xquery/"
end
```

Adding a task to copy XQuery scripts



Doesn't care what you're installing

Only needs SSH on remote servers

One command can do it all!

Some things Chris uses to make deployment easier

* deployment-only user accounts with proper permissions

* SSH login without passwords (using public key / private key)

* standardized locations for deployment

So let's go over Chris' 6 Rules for confident deployments

1. It's not the first time you deploy that hurts. It's the first time you change what's already there and it breaks that it hurts.

2. If you can change your application in production, you better be able to change it back to the way it was

3. Don't forget that there is often more to your application than just code

4. If it's not repeatable...and automated...you're wasting your time

5. The only difference in your application's environments should be the data sets

6. Ownership of code means also having a stake in making sure it gets deployed properly

email: chartjes@littlehart.net

Twitter: @chartjes

http://joind.in/119

blog: http://www.littlehart.net/atthekeyboard