

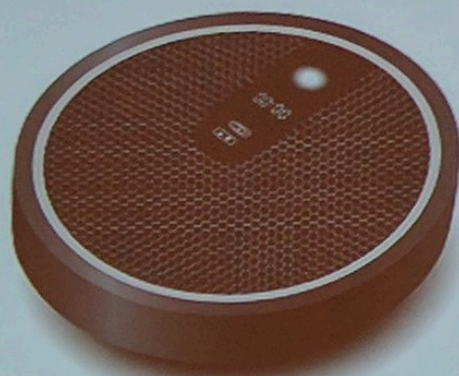
Simplifying Container Management with Habitat

Michael Ducy - Chef - @mfdii

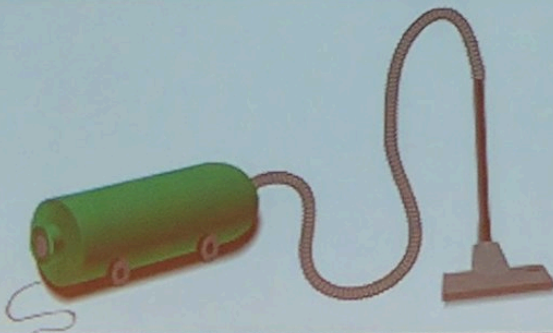
Habitat Community

- Join the Habitat Slack Team - <http://slack.habitat.sh/>
- Work through the tutorial at <https://www.habitat.sh/tutorials/>
- Explore Habitat packages on the depot - <https://app.habitat.sh/>
- Explore the Habitat projects - <https://github.com/habitat-sh>
- Read Habitat Blog posts - <https://blog.chef.io/?s=habitat>
- Join the Habitat Forums - <https://forums.habitat.sh/>

Broom



Roomba

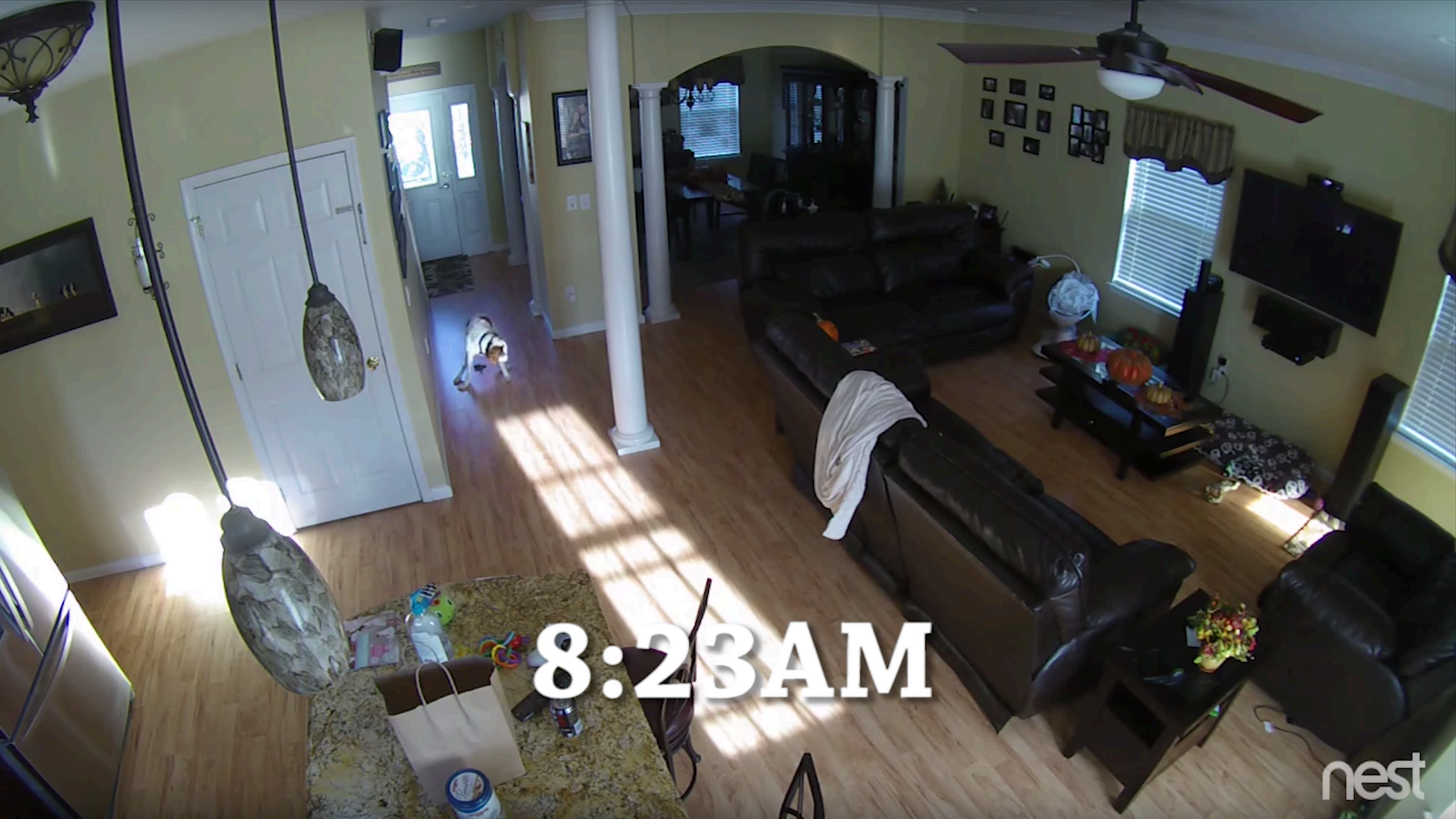


Vacuum
Cleaner










8:23AM

nest



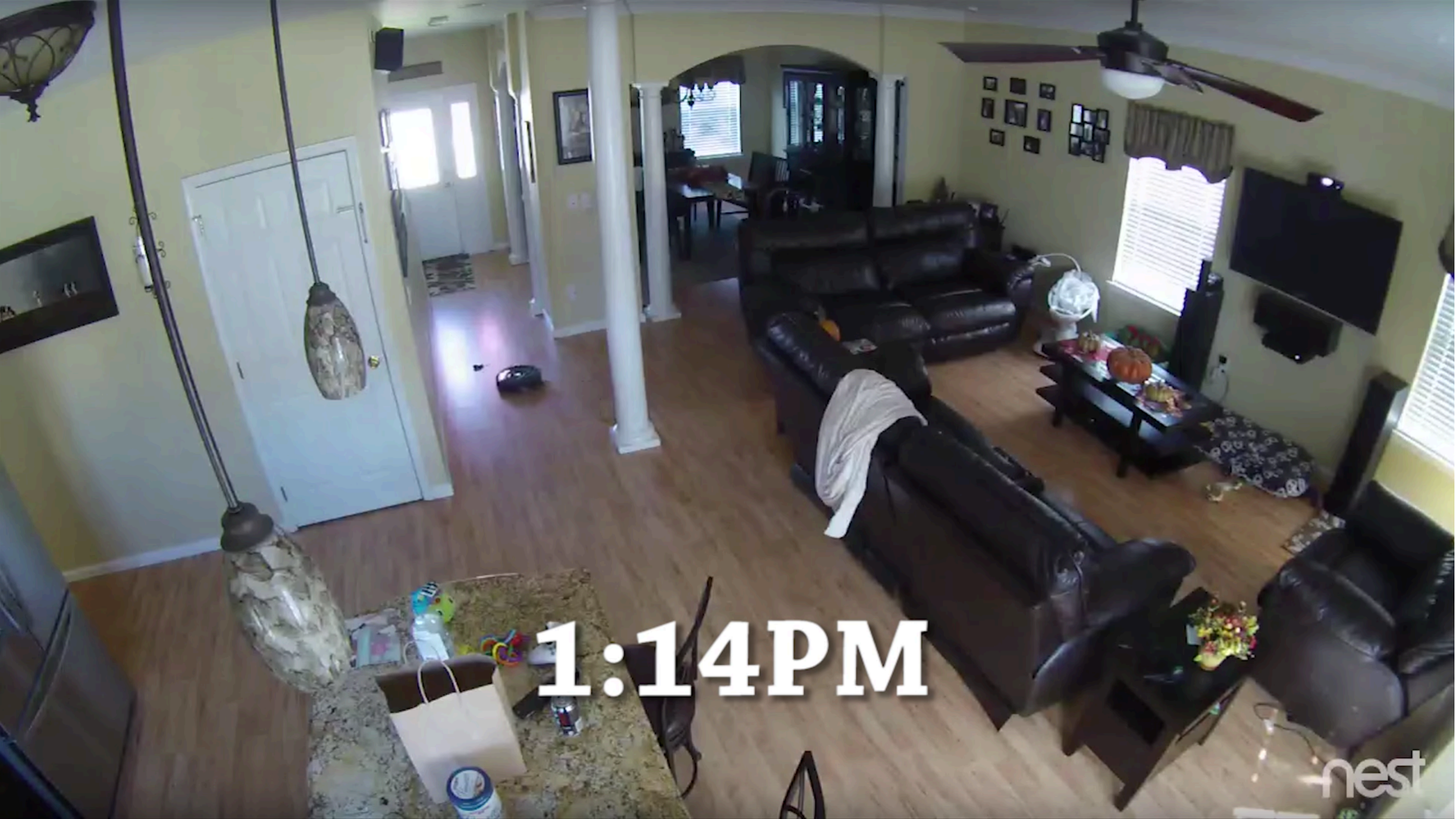
8:23AM

nest



1:14PM

nest



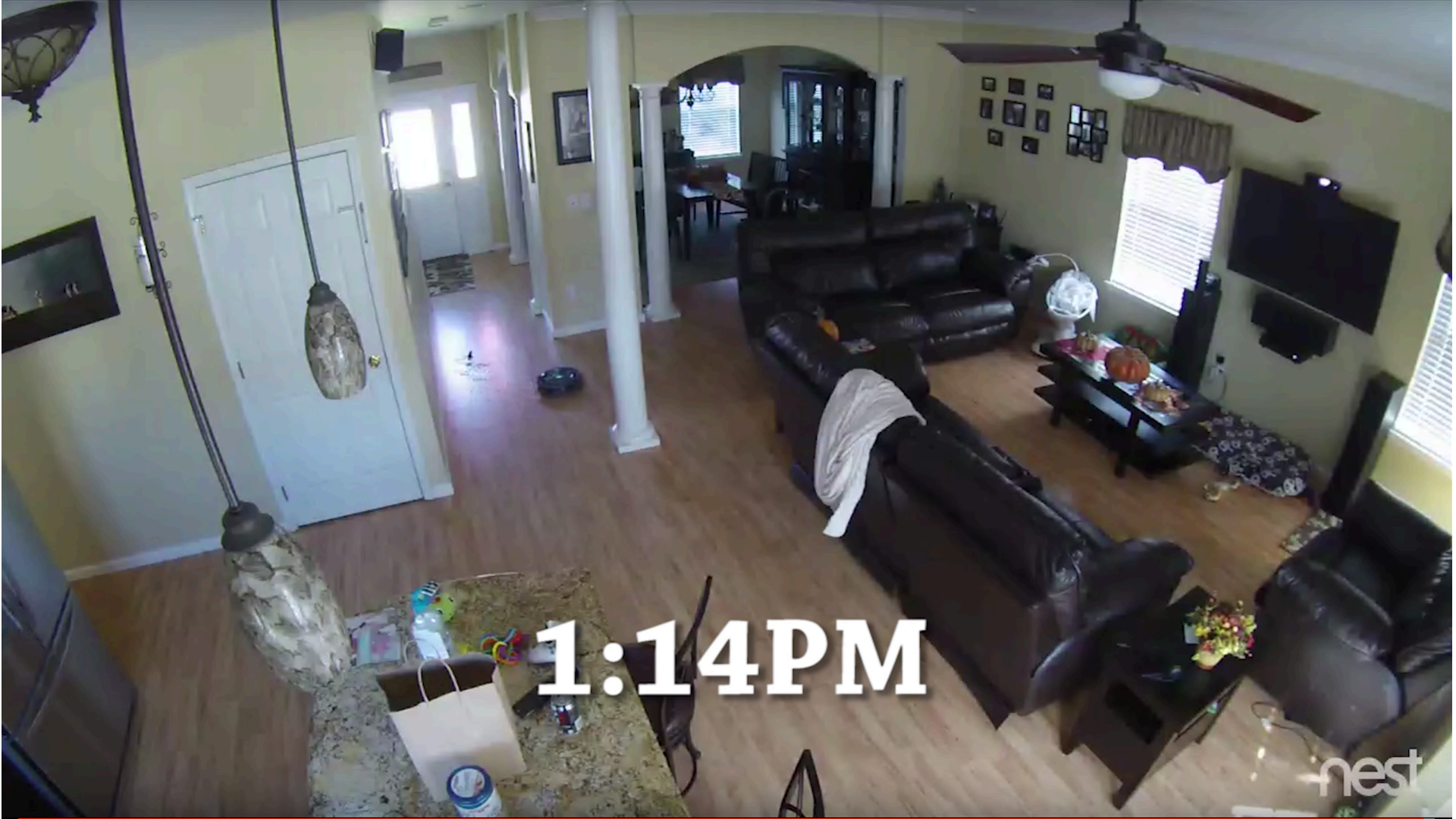
1:14PM

nest



1:14PM

nest



1:14PM

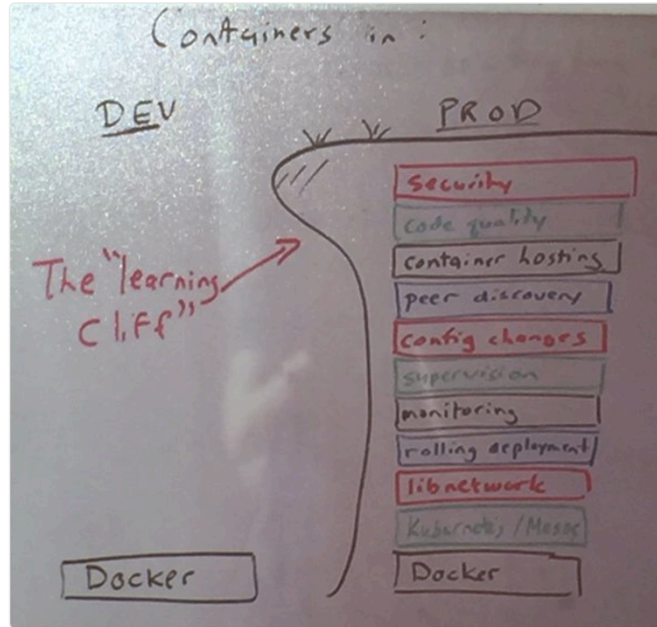
nest

Current Container Pain



Michael Ducey
@mfdii

Containers in Dev vs Prod



RETWEETS

1,632

LIKES

1,358



4:27 PM - 10 Feb 2016

Containers aren't a VM

You sure about that?

Current Container Usage

- 4 to 1 container to host ratio
- 75% of containers container a full OS

Easy to do the wrong thing

FROM ubuntu:12.04

FROM debian:latest

FROM centos:centos7

Lift and Shift your
Technical Debt for
fun and profit.

Modern Applications

API first

Small Area of Concern

Ephemeral

Focus on Artifacts

Habitat is about running modern apps.

Habitat

API First

Focuses on Artifacts

Eliminates the OS

Operable Application Containers

- Isolated
- Immutable
- Configurable
- Common interface for monitoring health
- Rebuild from source
- Common packaging
- Runtime Independence

The dreaded build cycle

```
wget https://some.place/package.tar.gz
```

```
tar xfv package.tar.gz
```

```
./configure
```

```
make
```

```
make install
```

The dreaded build cycle

Lifecycle Reference

The following lists all build phases of the `default`, `clean` and `site` lifecycles, which are executed in the order given up to the point of the one specified.

Clean Lifecycle

<code>pre-clean</code>	execute processes needed prior to the actual project cleaning
<code>clean</code>	remove all files generated by the previous build
<code>post-clean</code>	execute processes needed to finalize the project cleaning

Default Lifecycle

<code>validate</code>	validate the project is correct and all necessary information is available.
<code>initialize</code>	initialize build state, e.g. set properties or create directories.
<code>generate-sources</code>	generate any source code for inclusion in compilation.
<code>process-sources</code>	process the source code, for example to filter any values.
<code>generate-resources</code>	generate resources for inclusion in the package.
<code>process-resources</code>	copy and process the resources into the destination directory, ready for packaging.
<code>compile</code>	compile the source code of the project.
<code>process-classes</code>	post-process the generated files from compilation, for example to do bytecode enhancement on Java classes.
<code>generate-test-sources</code>	generate any test source code for inclusion in compilation.
<code>process-test-sources</code>	process the test source code, for example to filter any values.
<code>generate-test-resources</code>	create resources for testing.
<code>process-test-resources</code>	copy and process the resources into the test destination directory.
<code>test-compile</code>	compile the test source code into the test destination directory
<code>process-test-classes</code>	post-process the generated files from test compilation, for example to do bytecode enhancement on Java classes. For Maven 2.0.5 and above.
<code>test</code>	run tests using a suitable unit testing framework. These tests should not require the code be packaged or deployed.
<code>prepare-package</code>	perform any operations necessary to prepare a package before the actual packaging. This often results in an unpacked, processed version of the package. (Maven 2.1 and above)
<code>package</code>	take the compiled code and package it in its distributable format, such as a JAR.

The dreaded build cycle

npm-scripts

How npm handles the "scripts" field

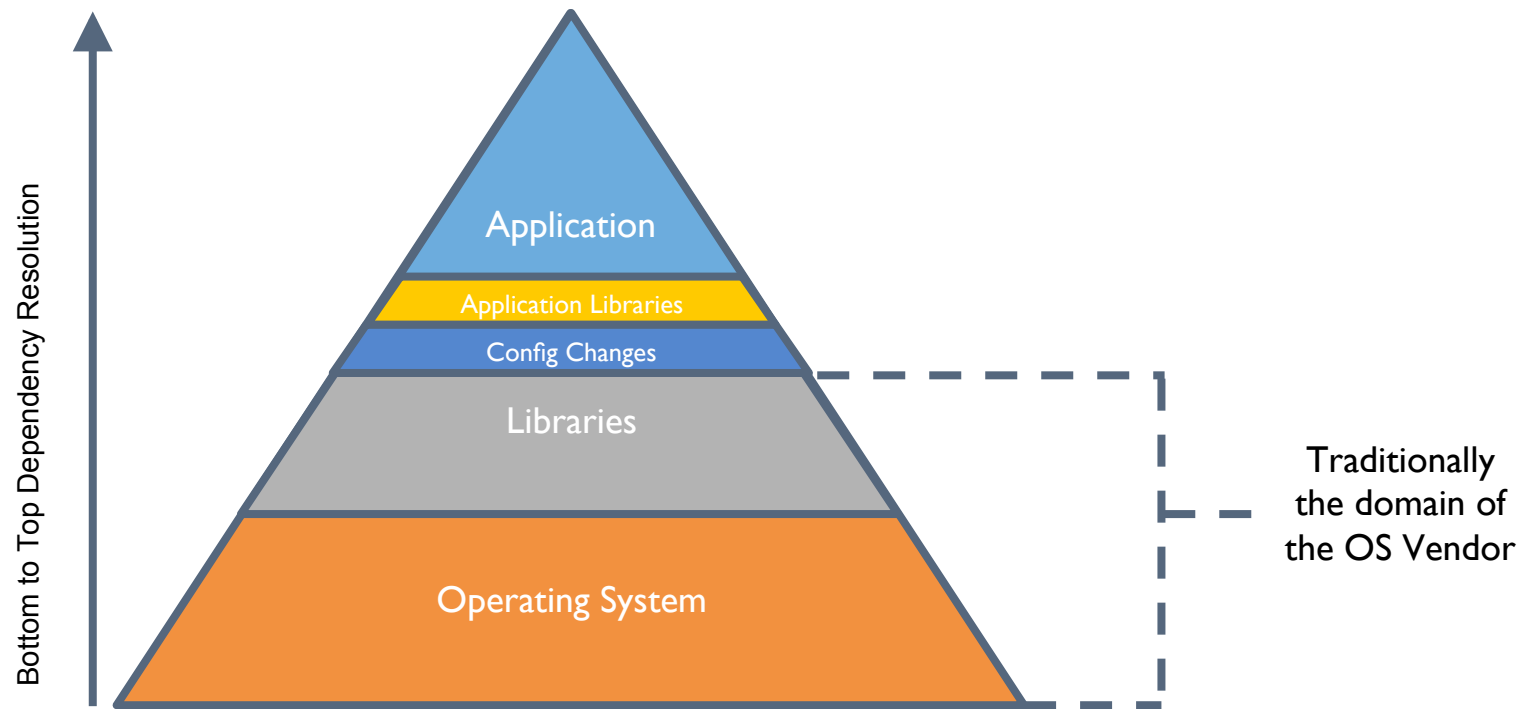
Description

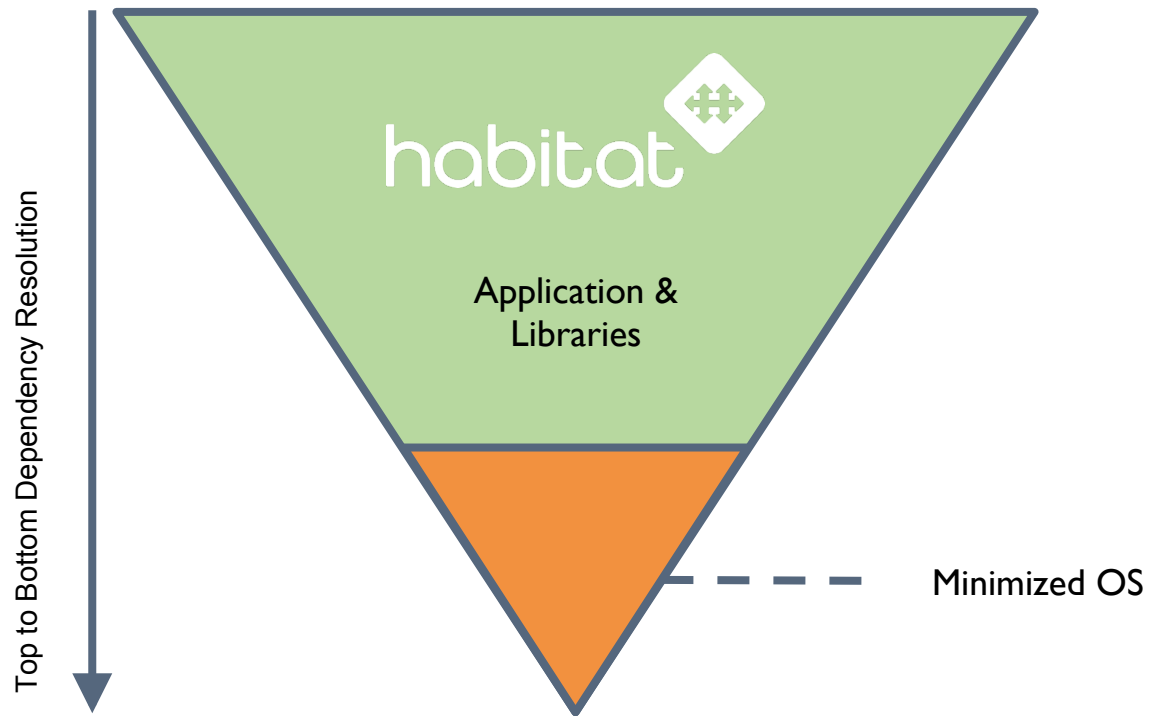
npm supports the "scripts" property of the package.json script, for the following scripts:

- **prepublish**: Run BEFORE the package is packed and published, as well as on local **npm install** without any arguments. (See below)
- **prepare**: Run both BEFORE the package is packed and published, and on local **npm install** without any arguments (See below). This is run AFTER **prepublish**, but BEFORE **prepublishOnly**.
- **prepublishOnly**: Run BEFORE the package is prepared and packed, ONLY on **npm publish**. (See below.)
- **prepack**: run BEFORE a tarball is packed (on **npm pack**, **npm publish**, and when installing git dependencies)
- **postpack**: Run AFTER the tarball has been generated and moved to its final destination.
- **publish**, **postpublish**: Run AFTER the package is published.
- **preinstall**: Run BEFORE the package is installed
- **install**, **postinstall**: Run AFTER the package is installed.
- **preuninstall**, **uninstall**: Run BEFORE the package is uninstalled.
- **postuninstall**: Run AFTER the package is uninstalled.
- **preversion**: Run BEFORE bumping the package version.
- **version**: Run AFTER bumping the package version, but BEFORE commit.
- **postversion**: Run AFTER bumping the package version, and AFTER commit.
- **pretest**, **test**, **posttest**: Run by the **npm test** command.
- **prestop**, **stop**, **poststop**: Run by the **npm stop** command.
- **prestart**, **start**, **poststart**: Run by the **npm start** command.
- **prerestart**, **restart**, **postrestart**: Run by the **npm restart** command. Note: **npm restart** will run the stop and start scripts if no **restart** script is provided.
- **pre shrinkwrap**, **shrinkwrap**, **postshrinkwrap**: Run by the **npm shrinkwrap** command.

Habitat Defines the Build Lifecycle

- Default implementation is C/C++ Build Lifecycle
- Other default implementations provided by Scaffolding
 - core/scaffolding-ruby
 - core/scaffolding-node
- Build dependencies explicitly declared
- Runtime dependencies explicitly declared.







Post-process packages

Docker

ACI

Mesosphere

What do applications need to run?

What do applications need to run?

- Lifecycle events
 - Start, Stop, Reconfigure, etc
- Environment specific configuration
- Knowledge of peers
- Knowledge of dependent services

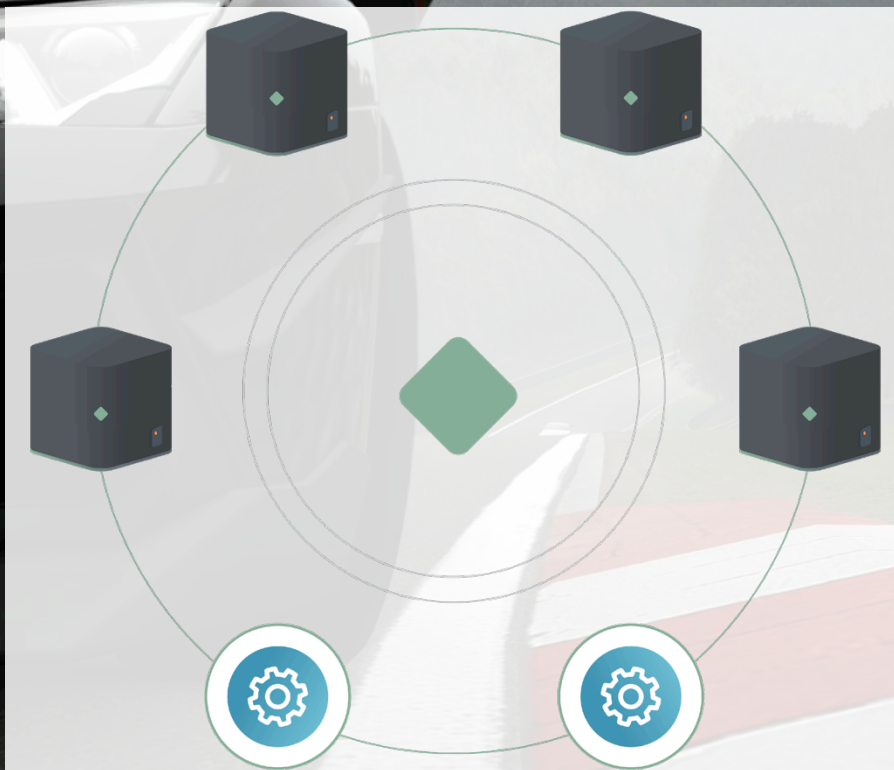
Lifecycle Hooks

- init
- run
- health_check
- file_updated
- reload
- reconfigure
- post-run
- suitability

Configuration

- Config files stored in ./config
- Default values provided by
 - Scaffolding
 - default.toml
- Override defaults through
 - ENV variable
 - user.toml
 - Over the network

Supervisors form a ring



Peers

Service Groups

Gossip

Availability
increases with
scale

What do applications need to run?

Supervisors support topologies



Dynamic
configuration

Service group
level

Uses the ring

Self Organizing Applications

- Habitat applications can self organize
- Typical pattern
 - Bake intent into the container
 - Redis-master container image
 - Redis-follower container image
 - Introduces image sprawl

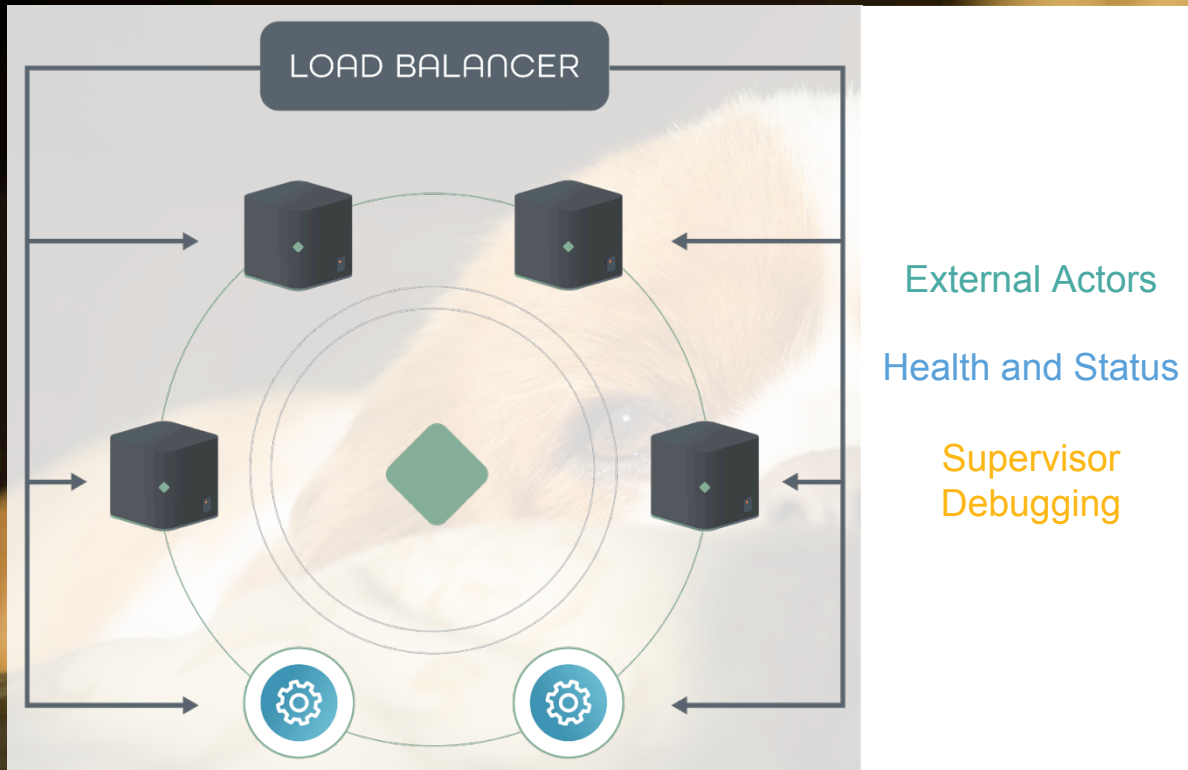
Self Organizing Applications

- Habitat pattern
 - One container image
 - Launch images with knowledge of peers
 - Application self organizes

Self Organizing Applications

- Habitat config files, hooks, and labeling
 - Config files are generating based on self organization
 - Hooks will be fired after the application self-organizes
 - Hooks can communicate with other services (Kubernetes) to inform services of changes

Supervisors provide a REST API



What Habitat brings to Containers:

- Build containers from the Application down
- Export containers in a variety of formats
- Automatically export containers with:
 - Service Discovery
 - Configuration Management
 - Supervisor API
 - Clustering Topology Support



It's all open source

Apache License

Questions

- Github repo with this demo- https://github.com/mfdii/hab_on_k8s
- Join the Habitat Slack Team - <http://slack.habitat.sh/>
- Work through the tutorial at <https://www.habitat.sh/tutorials/>
- Explore Habitat packages on the depot - <https://app.habitat.sh/>
- Explore the Habitat projects - <https://github.com/habitat-sh>
- Read Habitat Blog posts - <https://blog.chef.io/?s=habitat>
- Join the Habitat Forums - <https://forums.habitat.sh/>