
habitat
BY CHEF™

Mandi Walls

Technical Community Manager for EMEA
@lnxchk
mandi@chef.io
#habitatsh
http://slack.habitat.sh/
Ian Henry @Eeyun___ Habitat Community lead

How Do We Run Applications?

- On a computer
- With an OS
- And some libraries
- And some configuration
- And some way to start it and stop it

We've been moving complexity around rather than reducing it

Ugh.

```
case node['platform_family']
when 'freebsd'
  false
when 'arch', 'debian', 'rhel', 'fedora', 'amazon'
  true
when 'suse'
  node['platform_version'].to_f < 12.0 ? false : true
end
```

So. Habitat.

- Reduce snowflakeness
- Support microservices
- Manage container creep

https://www.bonanza.com/listings/Premier-Food-Storage-Containers-20-Piece-Set-Grey/443972348

Modern Applications Are Trending Toward

- Immutability
- Platform agnosticism
- Complexity reduction
- Scalability

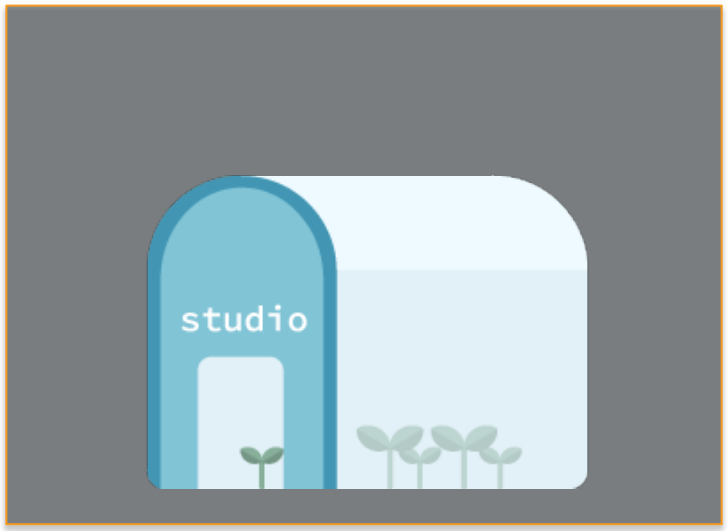
https://amazingmusthaves.com/products/steel-insulated-food-containers/

What Habitat Gets You

- Defer some decisions to runtime
- Do clean room builds
- Repeatable builds
- Distro agnostic packaging system
- Service runtime and discovery
- Configuration exposed via API
- Packages are signed by the system

Habitat Studio

- Provides a busy box clean room for your app
- Plus a set of tools for manipulating and running harts

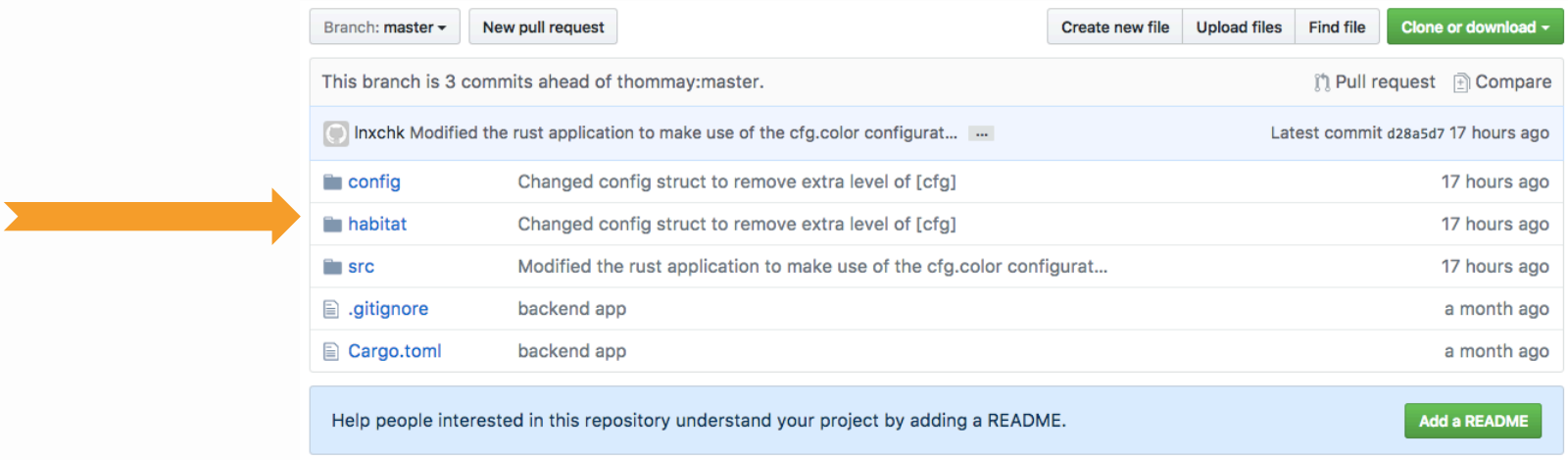


Why a Studio?

- Declare explicit dependencies
- Ship exactly what you need
- Sign your packages and store artifacts

Habitat Plans

- Plan files are where you put together your builds
- They are bash
- Live with the application



What's In A Plan?

```
pkg_name=container_sched_backend
pkg_origin=lnxchk
pkg_version="0.1.0"
pkg_build_deps=(core/rust)
pkg_deps=(core/glibc core/gcc core/gcc-libs)
pkg_bin_dirs=(bin)
bin="container_sched_backend"
pkg_exports=([out]=cfg.out)

do_build() {
  cargo build
}
do_install() {
  install -v -D "$PLAN_CONTEXT/../target/debug/$bin" \
    "$pkg_prefix/bin/$bin"
}
pkg_svc_run="$bin"
```

Examples at
https://github.com/habitat-sh/core-plans/

What Gets Built?

- Everything. Sort of.
- Build your own apps from source
- Decide if you want upstream binaries or source for things like runtime
  - You don't have to build Tomcat, but you can
- For COTS, use the binaries and skip steps

Configuration

- Can be manipulated at runtime
- Also travels with the app
- Provides variable substitution and templating using handlebars http://handlebarsjs.com/

Application Configuration File: TOML

```
[myconfig]
out = "{{cfg.out}}"
color = "{{cfg.color}}"
{{~# if svc.me.leader ~}}
leader = true
{{ else }}
leader = false
{{/if ~}}
```

Set Defaults in Habitat – default.toml

```
leader = false
out = "out"
color = "green”

[tomltable]
var = “val”
```

The Depot

- You can share plans with the Depot, and other hab users share theirs
- Has team namespacing
- The *core* plans are those built by the Habitat team
- https://app.habitat.sh/
- Private build services – Coming Soon!

  https://www.habitat.sh/blog/2017/05/Builder/

Caveat - Internet

- You can build your own stuff inside your own network, sort of, when it's all on one machine
- There will eventually be a private depot server
- For now, hab and its components need internet access

Build Output

- By default, it's a hart – a compressed tarball with some metadata and a signature
- You can export to other formats, like Docker containers
- The hart itself it runnable

Runtime

- The hab runtime includes management, service discovery, other features
- The habs in your application create a mesh so they can talk to each other
- You can even update your application via the mesh without restarting every application manually

Running a Hart

```
sudo hab start lnxchk/container_sched_backend
--peer 172.31.13.250 --topology leader
```

- The same hart runs on multiple distros — no need for other packages
- Once hooked together, the supervisors will have a leader election
- If instances move in or out of the mesh, a new election will occur after a timeout
- Updates are persistent and stored in metadata on the hosts

  /hab/data/services

Updating Configuration at Runtime

- Update all or part of the configuration while the apps are running
- Send the update to a member of the mesh and they will all update

```
sudo hab config apply container_sched_backend.default 2
newconfig.toml
```

Supervisor Web Interface

- http://ip.add.re.ss:9631/services
- http://ip.add.re.ss:9631/census

Additional Features

- *Healthchecks* – can be customized for your app, and travel in the hart

  https://www.habitat.sh/tutorials/sample-app/windows/add-health-check-hook/

- *Dynamic Updates* – when a new version is uploaded to the Depot in the "stable" channel, update running apps

  https://www.habitat.sh/tutorials/sample-app/windows/update-app/

Shortcut for common platforms: Scaffolding

- Default core-built dependencies for common runtimes
- Ruby and Node so far

```
pkg_name=MY_APP
pkg_origin=MY_ORIGIN
pkg_version=MY_VERSION
pkg_scaffolding=core/scaffolding-ruby
```

https://www.habitat.sh/docs/concepts-scaffolding/

Join Us!

- On Slack!

  http://slack.habitat.sh
- Online! With Tutorials!

  https://www.habitat.sh/
- On Github!

  https://github.com/habitat-sh
- The sample app in this talk

  https://github.com/lnxchk/container_sched_backend

Other References

- Summary on The New Stack

  https://thenewstack.io/chef-habitat-addresses-issues-moving-containers-production

- Our YouTube Channel

  https://www.youtube.com/user/getchef

October 10 – 11, 2017

etc.venues Fenchurch St London

https://chef.io/summits

mandi@chef.io