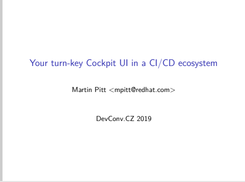
Your turn-key Cockpit UI in a CI/CD ecosystem

Martin Pitt <mpitt@redhat.com>

DevConv.CZ 2019

2019-01-20

Your turn-key Cockpit UI in a CI/CD ecosystem



Your turn-key Cockpit UI in a CI/CD ecosystem

Martin Pitt <mpitt@redhat.com>

DevConv.CZ 2019

IaaS

2019-01-20

Your turn-key Cockpit UI in a CI/CD ecosystem

1. 10-second history of cloud computing
2. Infrastructure aaS: my other computer is a data center

PaaS

2019-01-20

Your turn-key Cockpit UI in a CI/CD ecosystem

1. Platform aaS: Kubernetes

SaaS

1. Software aaS: we don't host our source repos any more, GitHub

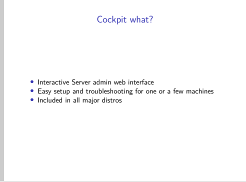
CoCICDaaS

Your turn-key Cockpit UI in a CI/CD ecosystem

1. undeniably the pinnacle of evolution: Cockpit Continuous Integration and Deployment aaS
2. that's what I introduce today

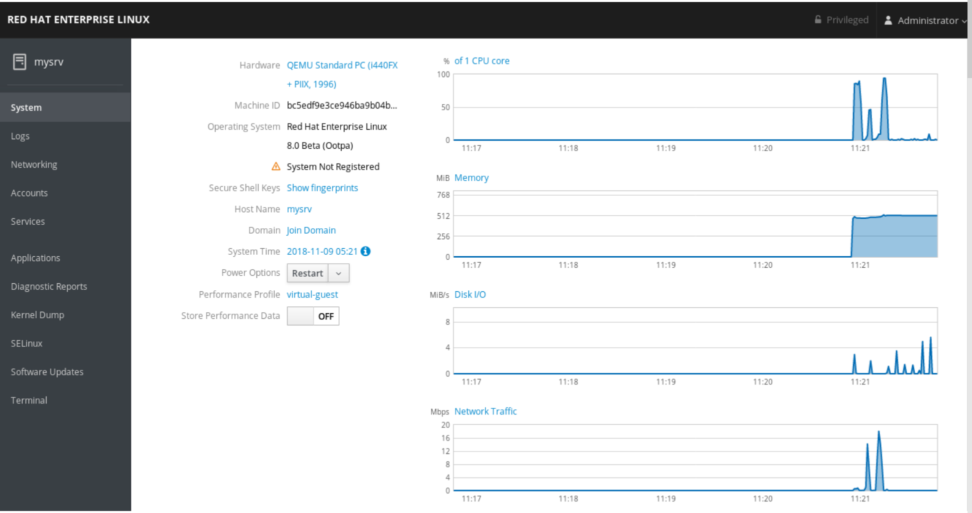
Cockpit what?

- Interactive Server admin web interface
- Easy setup and troubleshooting for one or a few machines
- Included in all major distros


Cockpit what?

- Interactive Server admin web interface
- Easy setup and troubleshooting for one or a few machines
- Included in all major distros

1. Conceptually: Linux session running in a web browser; technically very similar to ssh/VT/GNOME login
2. Aimed at admins who are new to Linux, e. g. coming from the Windows world and familiar with the concepts, but not Linux terminology
3. but also to experienced ones for infrequent tasks (set up RAID once a year, don’t remember all the commands); not just setup, but also investigating “what is wrong with this machine”
4. apt or yum install away in Fedora, Atomic, RHEL, Debian, Ubuntu, Arch


RED HAT ENTERPRISE LINUX

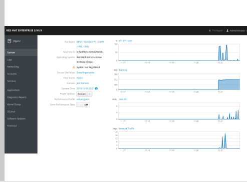
Privileged · Administrator

mysrv

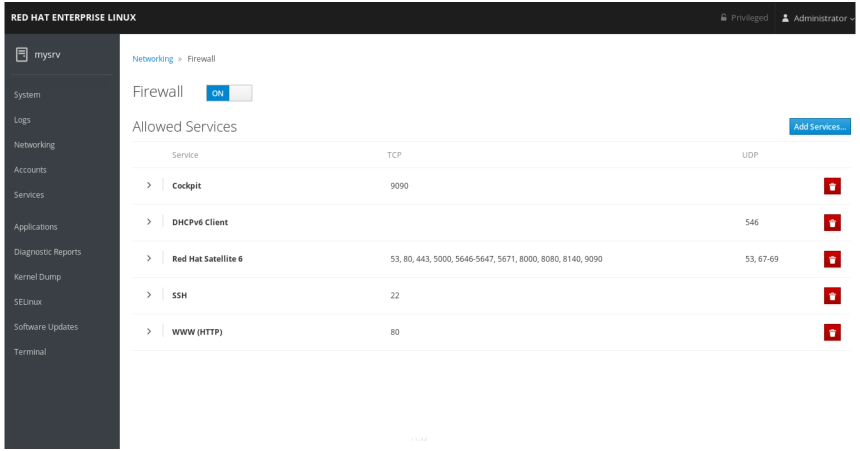
Hardware	QEMU Standard PC (i440FX + PIIX, 1996)
Machine ID	bc5edf9e3ce946ba9b04b...
Operating System	Red Hat Enterprise Linux 8.0 Beta (Ootpa)
	System Not Registered
Secure Shell Keys	Show fingerprints
Host Name	mysrv
Domain	Join Domain
System Time	2018-11-09 05:21
Power Options	Restart
Performance Profile	virtual-guest
Store Performance Data	OFF

2019-01-20

Your turn-key Cockpit UI in a CI/CD ecosystem



1. System page: Summary information about the machine and its current status
2. can drill down into more detailed graphs and information.
3. Menu on the left shows available administration pages for this machine, and can switch between multiple machines



RED HAT ENTERPRISE LINUX

Privileged | Administrator

mysrv

System
Logs
Networking
Accounts
Services
Applications
Diagnostic Reports
Kernel Dump
SELinux
Software Updates
Terminal

Networking » Firewall

Firewall ON

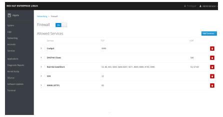
Allowed Services

Add Services...

Service	TCP	UDP
Cockpit	9090	
DHCPv6 Client		546
Red Hat Satellite 6	53, 80, 443, 5000, 5646-5647, 5671, 8000, 8080, 8140, 9090	53, 67-69
SSH	22	
WWW (HTTP)	80	

2019-01-20

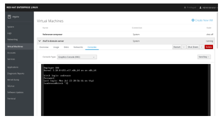
Your turn-key Cockpit UI in a CI/CD ecosystem

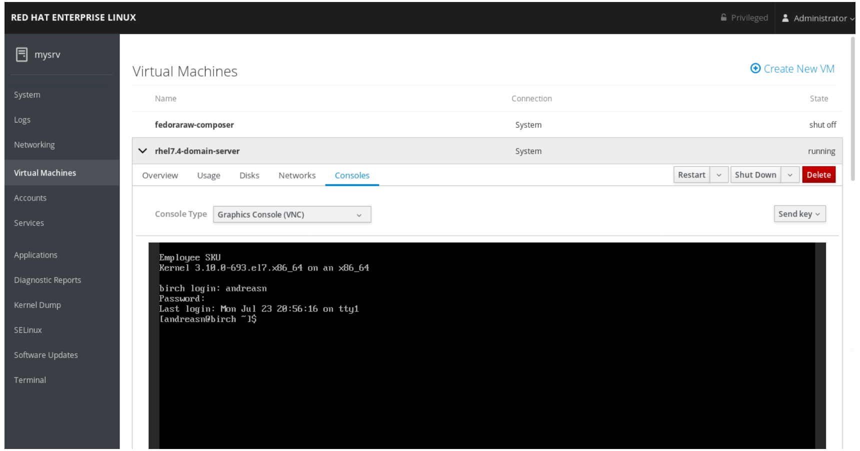


1. subpage of Networking is a UI for firewalld

2019-01-20

Your turn-key Cockpit UI in a CI/CD ecosystem





Name	Connection	State
fedoraraw-composer	System	shut off
rhel7.4-domain-server	System	running

Console Type: Graphics Console (VNC)

```
Employee SKU
Kernel 3.10.0-693.el7.x86_64 on an x86_64

birch login: andreasn
Password:
Last login: Mon Jul 23 20:56:16 on tty1
[andreasn@birch ~]$
```

1. See and interact with your local libvirt or ovirt VMs
2. Cockpit team maintains pages seen on the screenshots

Imagine your own page here!

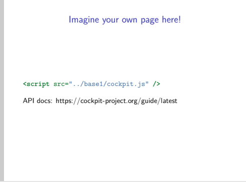
```
<script src="../base1/cockpit.js" />
```

API docs: https://cockpit-project.org/guide/latest

2019-01-20

Your turn-key Cockpit UI in a CI/CD ecosystem

└ Imagine your own page here!


Imagine your own page here!

```
<script src="../base1/cockpit.js" />
```

API docs: https://cockpit-project.org/guide/latest

1. there will always be things missing for your use cases, so designed from the ground up to be easily extensible
2. offers JS API to interact with connected target machine: programs, D-Bus, files, sockets, etc.

```
<table>
    <tr>
        <td><label for="address">Address</label></td>
        <td><input id="address" value="8.8.8.8"></td>
    </tr>
    <tr>
        <td><button id="ping">Ping</button></td>
        <td><span id="result"></span></td>
    </tr>
</table>

<p> <pre id="output"></pre> </p>
```

2019-01-20

Your turn-key Cockpit UI in a CI/CD ecosystem

```
<table>
    <tr>
        <td><label for="address">Address</label></td>
        <td><input id="address" value="8.8.8.8"></td>
    </tr>
    <tr>
        <td><button id="ping">Ping</button></td>
        <td><span id="result"></span></td>
    </tr>
</table>

<p> <pre id="output"></pre> </p>
```

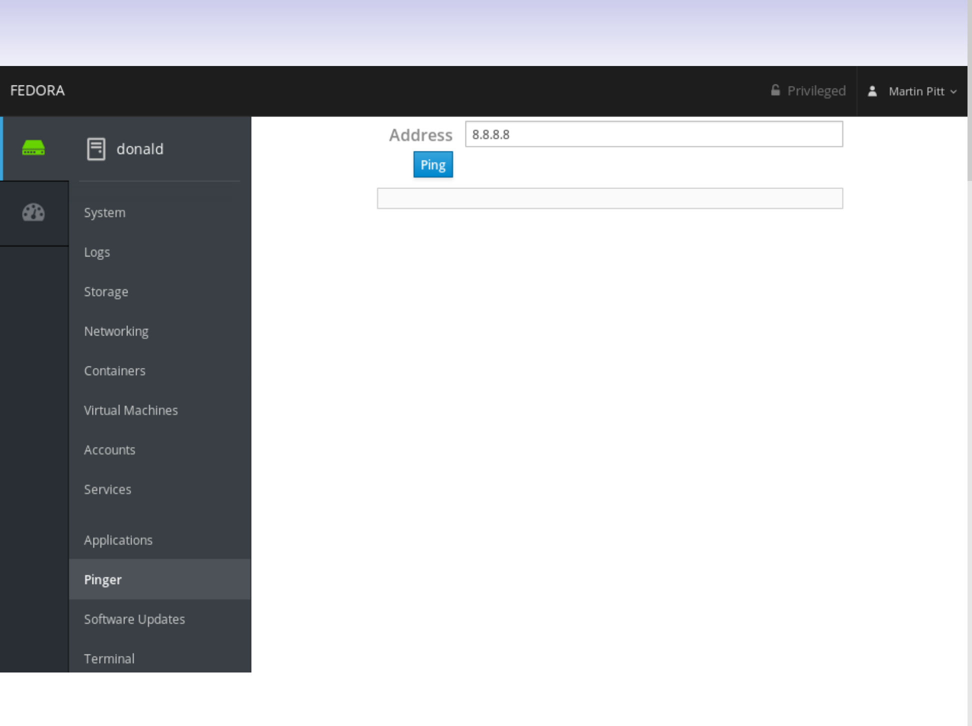
1. little example: create a UI for ping
2. input for address, button to start, and pre for output

```
const button = document.getElementById("ping");
const address = document.getElementById("address");
const result = document.getElementById("result");
const output = document.getElementById("output");

button.addEventListener("click", () => {
  cockpit.spawn(["ping", "-c", "4", address.value])
    .stream(data => output.append(
                        document.createTextNode(data))
    .done(() => {
        result.innerHTML = "success";
        result.style.color = "green";
    });
});
```

Your turn-key Cockpit UI in a CI/CD ecosystem

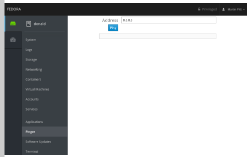
```
const button = document.getElementById("ping");
const address = document.getElementById("address");
const result = document.getElementById("result");
const output = document.getElementById("output");

button.addEventListener("click", () => {
  cockpit.spawn(["ping", "-c", "4", address.value])
    .stream(data => output.append(
                        document.createTextNode(data))
    .done(() => {
        result.innerHTML = "success";
        result.style.color = "green";
    });
});
```

1. wire cockpit API for running a process - ping in this case to this UI
2. whenever something new on stdout → append to output field for live streaming
3. slightly simplified, e. g. no error handling, but this is the gist
4. similar structure for a D-Bus call, or files

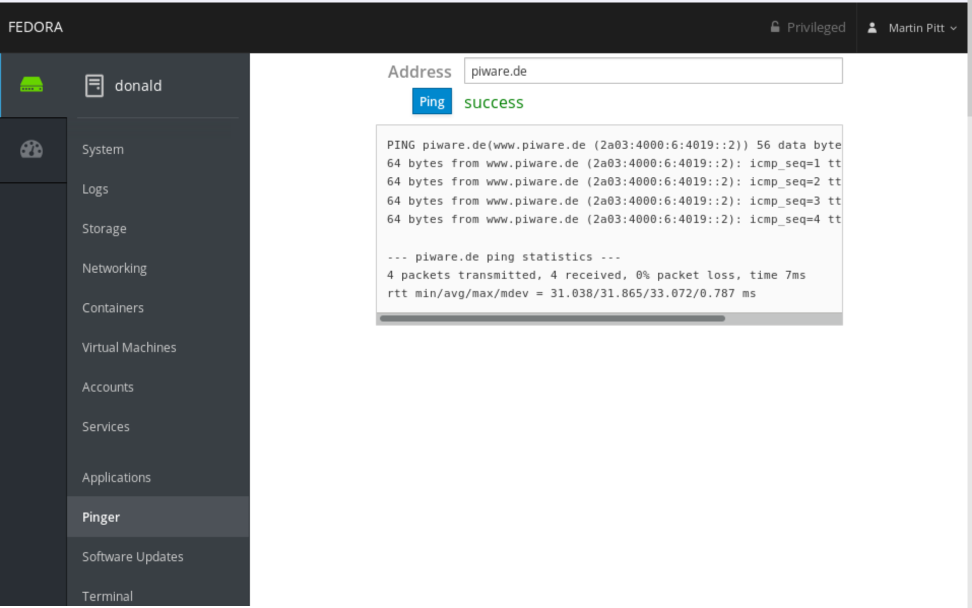


2019-01-20

Your turn-key Cockpit UI in a CI/CD ecosystem



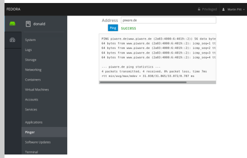
1. initially looks like this; enter address, press button



```
PING piware.de(www.piware.de (2a03:4000:6:4019::2)) 56 data byte
64 bytes from www.piware.de (2a03:4000:6:4019::2): icmp_seq=1 tt
64 bytes from www.piware.de (2a03:4000:6:4019::2): icmp_seq=2 tt
64 bytes from www.piware.de (2a03:4000:6:4019::2): icmp_seq=3 tt
64 bytes from www.piware.de (2a03:4000:6:4019::2): icmp_seq=4 tt

--- piware.de ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 7ms
rtt min/avg/max/mdev = 31.038/31.865/33.072/0.787 ms
```

2019-01-20

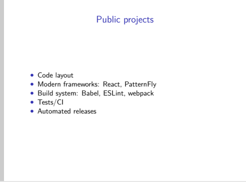
Your turn-key Cockpit UI in a CI/CD ecosystem



1. and you see the result
2. appears in the menu via a little declaration file called manifest; not shown here
3. above good enough for your own personal environment/company specific pages
4. ex: cheap monitoring/control of services or house automation
5. cockpit more popular, more extension projects which are public, get packaged and team-maintained
6. ex: UI for podman, building installable OS images, IPA server, Fleet Commander
7. proposed: NFS server, SSL certificate management
8. then tossing the above into a single HTML file is not good enough

Public projects

- Code layout
- Modern frameworks: React, PatternFly
- Build system: Babel, ESLint, webpack
- Tests/CI
- Automated releases

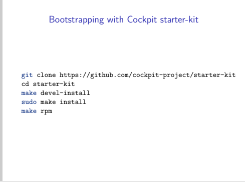

Public projects

- Code layout
- Modern frameworks: React, PatternFly
- Build system: Babel, ESLint, webpack
- Tests/CI
- Automated releases

1. Separation of HTML, CSS, and JavaScript into lots of little files for maintainability
2. Don't do UI by hand like in pinger, integrate React and PatternFly
3. JavaScript toolchain to compile all your files into a blob the browser can understand
4. complex build system, integrate static code checks
5. create automated browser tests, run them in PRs
6. test on various operating systems, maintain VM images for these
7. release very often to GitHub, various distros, COPR, dockerhub, update your project page, etc.
8. putting this together is a daunting task

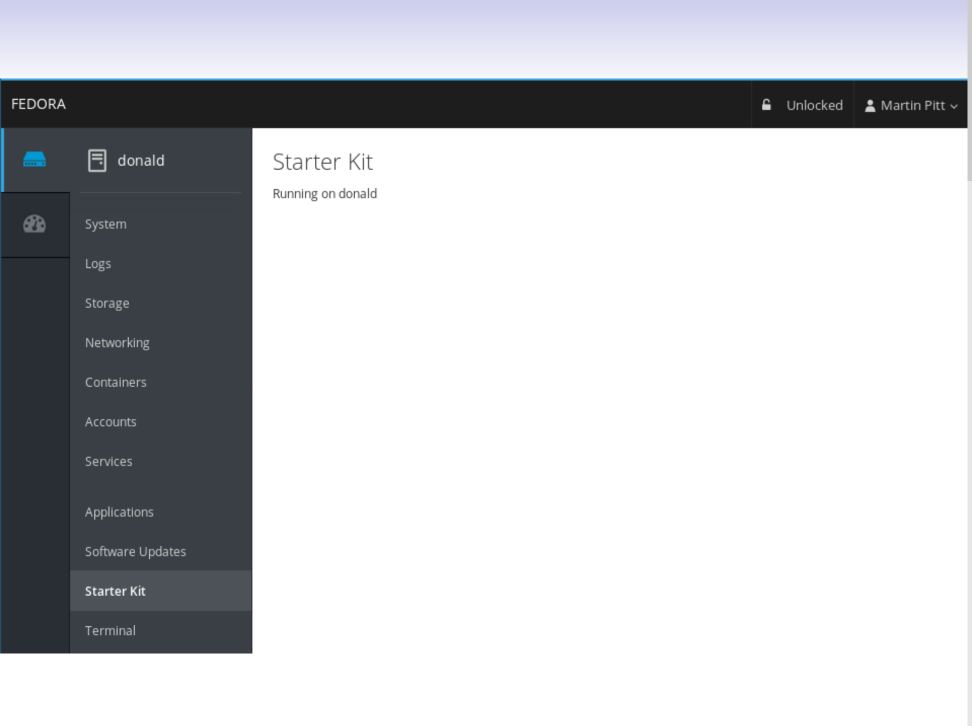
Bootstrapping with Cockpit starter-kit

```
git clone https://github.com/cockpit-project/starter-kit
cd starter-kit
make devel-install
sudo make install
make rpm
```


Bootstrapping with Cockpit starter-kit

```
git clone https://github.com/cockpit-project/starter-kit
cd starter-kit
make devel-install
sudo make install
make rpm
```

1. we put together the Cockpit starter kit, does all that for you
2. best practices for a Cockpit project
3. example UI with all the glory I mentioned before
4. devel-install: run straight out of your build tree; install: /usr/local/, build rpm

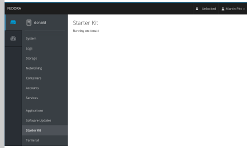


Starter Kit

Running on donald

2019-01-20

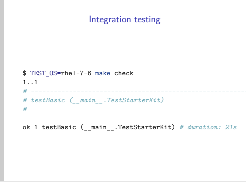
Your turn-key Cockpit UI in a CI/CD ecosystem



1. looks unspectacular, but demonstrates cockpit API (reading hostname) and LESS/CSS
2. point is to be a simple React component which you can directly hack on without worrying about all the boilerplate

Integration testing

```
$ TEST_OS=rhel-7-6 make check
1..1
# ----------------------------------------------------------
# testBasic (__main__.TestStarterKit)
#

ok 1 testBasic (__main__.TestStarterKit) # duration: 21s
```


Integration testing

```
$ TEST_OS=rhel-7-6 make check
1..1
# ----------------------------------------------------------
# testBasic (__main__.TestStarterKit)
#

ok 1 testBasic (__main__.TestStarterKit) # duration: 21s
```

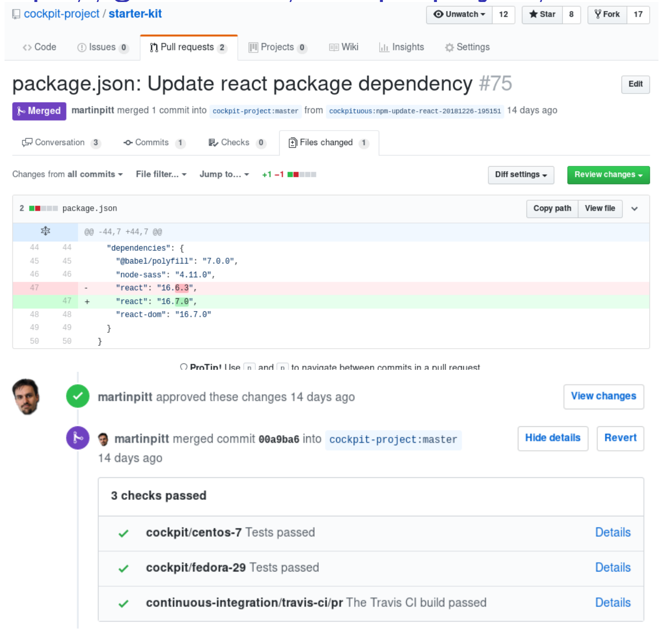
1. RPM build, integration test
2. test looks simple, but does a lot of stuff for you
3. download appropriate Cockpit VM image (lots of OSes), builds code, installs it into the VM, starts headless Chromium, runs your test on it
4. re-uses VMs of Cockpit team, half-time job to maintain them
5. integrate into CI: webhook, ask Cockpit team to whitelist to run on our infra

Automated releases

```
$ cat ./cockpituous-release
RELEASE_SOURCE="_release/source"
RELEASE_SPEC="cockpit-starter-kit.spec"
RELEASE_SRPM="_release/srpm"

job release-source
job release-srpm

# job release-koji -k master
# job release-koji f29
# job release-bodhi F29
# job release-github
# job release-copr @myorg/myrepo
```

1. release process: push a signed git tag with a summary of changes
2. our cockpituous infra then builds release tarballs, srpms, pushes them to github, Fedora, dockerhub, copr, etc.
3. real file has lots of comments
4. just like with CI, ask Cockpit team
5. that part is relatively easy to self-host: container with a bunch of credentials; or run on your laptop

https://github.com/cockpit-project/starter-kit/pull/75


cockpit-project / **starter-kit**

Unwatch 12 Star 8 Fork 17

Code | Issues 0 | Pull requests 2 | Projects 0 | Wiki | Insights | Settings

package.json: Update react package dependency #75

Edit

Merged **martinpitt** merged 1 commit into `cockpit-project:master` from `cockpituous:npm-update-react-20181226-195151` 14 days ago

Conversation 3 | Commits 1 | Checks 0 | Files changed 1

Changes from **all commits** ▾ File filter... ▾ Jump to... ▾ +1 −1

Diff settings ▾ Review changes ▾

2 package.json Copy path View file

```
@@ -44,7 +44,7 @@
      "dependencies": {
        "@babel/polyfill": "7.0.0",
        "node-sass": "4.11.0",
-       "react": "16.6.3",
+       "react": "16.7.0",
        "react-dom": "16.7.0"
      }
    }
```

ProTip! Use n and p to navigate between commits in a pull request.

martinpitt approved these changes 14 days ago View changes

martinpitt merged commit **0a9ba6** into `cockpit-project:master` 14 days ago Hide details Revert

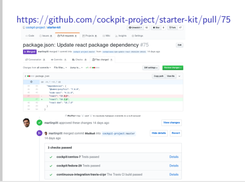
3 checks passed

✓	**cockpit/centos-7** Tests passed	Details
✓	**cockpit/fedora-29** Tests passed	Details
✓	**continuous-integration/travis-ci/pr** The Travis CI build passed	Details

Your turn-key Cockpit UI in a CI/CD ecosystem

2019-01-20

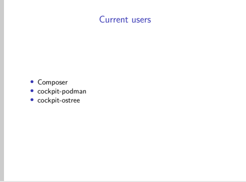
https://github.com/cockpit-project/starter-kit/pull/75



1. routine maintenance tasks: latest NPM dependencies, uploading translation templates to Zanata, download translations
2. bots for code maintenance; example for NPM update
3. proposes a PR for updating to latest React, tests pass; human can sign off and presses the button

Current users

- Composer
- cockpit-podman
- cockpit-ostree

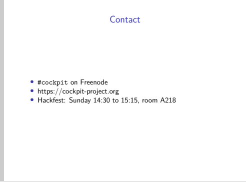

Current users

- Composer
- cockpit-podman
- cockpit-ostree

1. these projects are real-life, thus this is not a pipe dream; let's add your's
2. Our team wants to scale from "we build UIs for everything" to "we support your team with building your UI"
3. we work a lot on providing CI infrastructure, cross-project testing and maintenance

Contact

- `#cockpit` on Freenode
- https://cockpit-project.org
- Hackfest: Sunday 14:30 to 15:15, room A218


Contact

- `#cockpit` on Freenode
- https://cockpit-project.org
- Hackfest: Sunday 14:30 to 15:15, room A218

1. Home page leads to mailing lists, documentation
2. Join us on the hackfest on Sunday
3. thanks for your attention; Q+A