

Resurrecting dinosaurs, what can possibly go wrong?

How Containerised Apps could eat our users.

Richard Brown
openSUSE Chairman
rbrown@opensuse.org

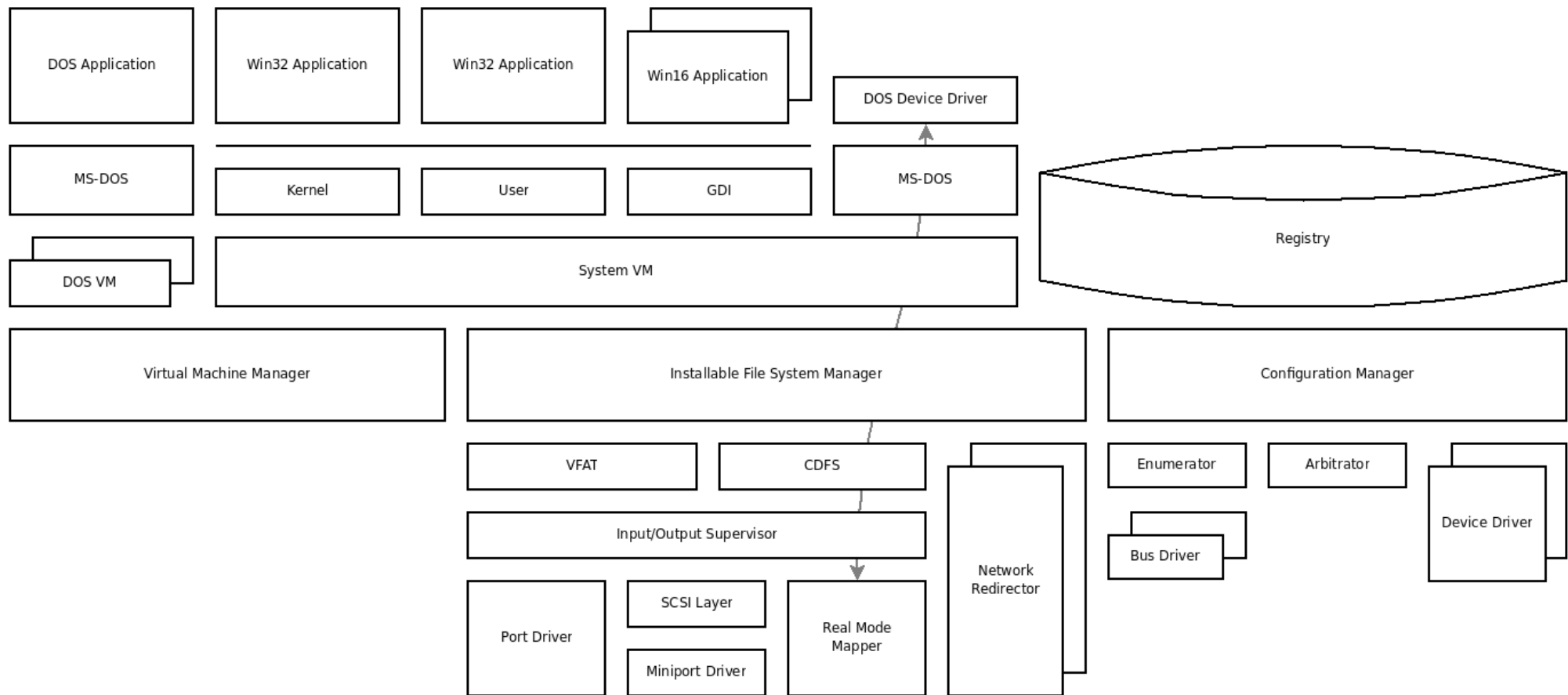
“Those who cannot remember the past are
condemned to repeat it”

- George Santayana



The background features abstract geometric shapes in two shades of green. On the left, a large teal shape with a pointed right side contains the text. To its right, a white diagonal line separates it from a bright green shape on the right side of the frame.

In the beginning



Windows 3.1/95 - DLL Hell

- No ABI backwards compatibility
- Most DLLs installed in C:\WINDOWS or C:\WINDOWS\SYSTEM
- Global COM Class IDs
- Service/Maintenance Nightmare



DLL Hell in Real Terms

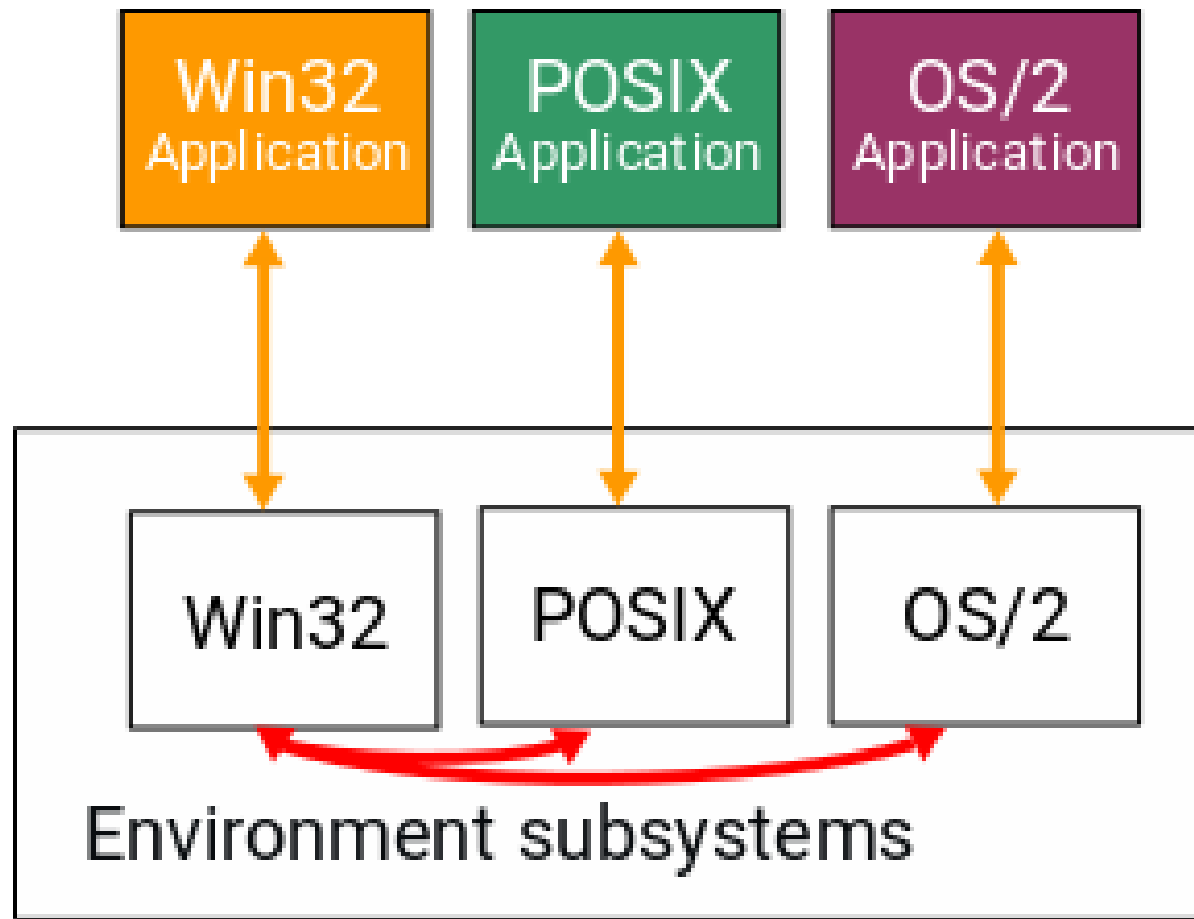
- Developers had to dev & test Apps on every possible DLL combination
- Then retest every App patch on every possible DLL combination
- AND test every DLL patch on every possible App & DLL combination
- Then cry when it all broke anyway



Windows 2000 to the Rescue

- Side-by-side (SxS) assembly – DLL “Containerisation”
 - Separate Memory Space for each App and its DLLs
 - ‘Private DLLs’ loaded from the Application Directory
- Windows File Protection (WFP) – Disk Isolation of System DLLs
- DLL Universal Problem Solver (DUPS) – Audit all the DLLs in use and help migrate ‘legacy’ applications to SxS bundles





Problem Solved? Right?

- Security nightmare
 - Security relevant DLLs lurking in countless application folders
- Maintenance nightmare
 - How are we going to update our app? Oh we'll ship an updater!
- Legal nightmare
 - Can we legally redistribute all the DLLs we need to?
- Storage vendor dream
 - More disk consumption, everyone buying bigger disks!



Meanwhile in Linuxland

The background features a large teal shape on the left and a green shape on the right, separated by a white diagonal line. The teal shape is a large, irregular polygon with a pointed top and a flat bottom. The green shape is a large, irregular polygon with a pointed top and a flat bottom, mirroring the teal shape's form. The white diagonal line runs from the top right towards the bottom left, creating a clear division between the two colors.



CC-BY-NC Dustin Jamison



Distributions – Solving Real Problems

- Security
 - Security Teams auditing packages, monitoring CVEs & embargoed lists
- Maintenance
 - Maintainers packaging applications & keeping them updated
- Legal
 - Lawyers auditing licenses and ensuring compatibility/compliance



In Defence of Shared Libraries/Dependencies

- Not just about using less space on disk
- Distributing fewer libraries have broad benefits
 - Fewer INSECURE libraries, more easily patched
 - Less manpower required to maintain/update
 - Easier to review/ensure legal compliance



Mission Accomplished?

- Compatibility
- Portability
- Pace of Change vs “It just works”



Windows 3.1/95 - DLL Hell

- No ABI backwards compatibility
- Most DLLs installed in C:\WINDOWS or C:\WINDOWS\SYSTEM
- Global COM Class IDs
- Service/Maintenance Nightmare



Compatibility

- Many distributions with many different libraries and apps
- Different apps require different libraries
- Application developers don't want to worry about what other application developers have chosen as their dependencies



Compatibility

- Many distributions with many different libraries and apps
- Different apps require different libraries
- Application developers don't want to worry about what other application developers have chosen as their dependencies

- But application developers don't (often) worry about this
- Distro Maintainers work on this for F/OSS licensed apps



Portability

- Many distributions with many different libraries and toolsets
- Application Developers don't want to learn dozens of toolsets, nor rebuild & retest their application on a dozen platforms



Portability

- Many distributions with many different libraries and toolsets
- Application Developers don't want to learn dozens of toolsets, nor rebuild & retest their application on a dozen platforms
- But application developers don't (often) worry about this
- Distro Maintainers solve the problem for F/OSS licensed apps



Pace of Change vs “It just works”

- Many distributions with fixed release schedules
- Distributions freeze package/library versions to aid ‘stability’
- Holds back new application versions from users



Pace of Change vs “It just works”

- Many distributions with fixed release schedules
- Distributions freeze package/library versions to aid ‘stability’
- Holds back new application versions from users

- But application developers don’t need to worry about this
- Rolling Distributions resolve this with increasing efficiency





Back to the Future!

Containerised Applications to the Rescue

- AppImage, FlatPak, Snappy
- Provides users with a “Bundle” containing App + Libraries
- Runs the App in some kind of Sandbox or Container

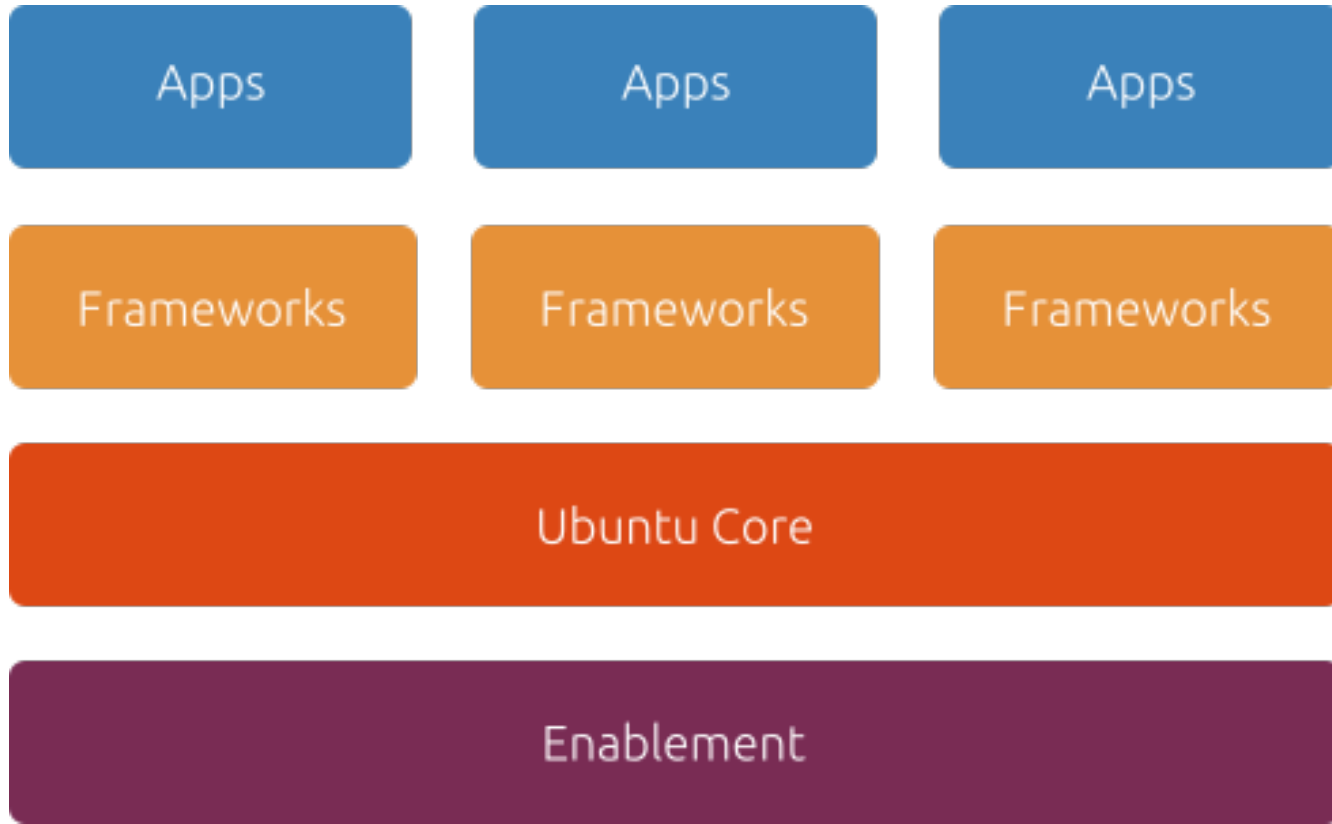


The Big Promises

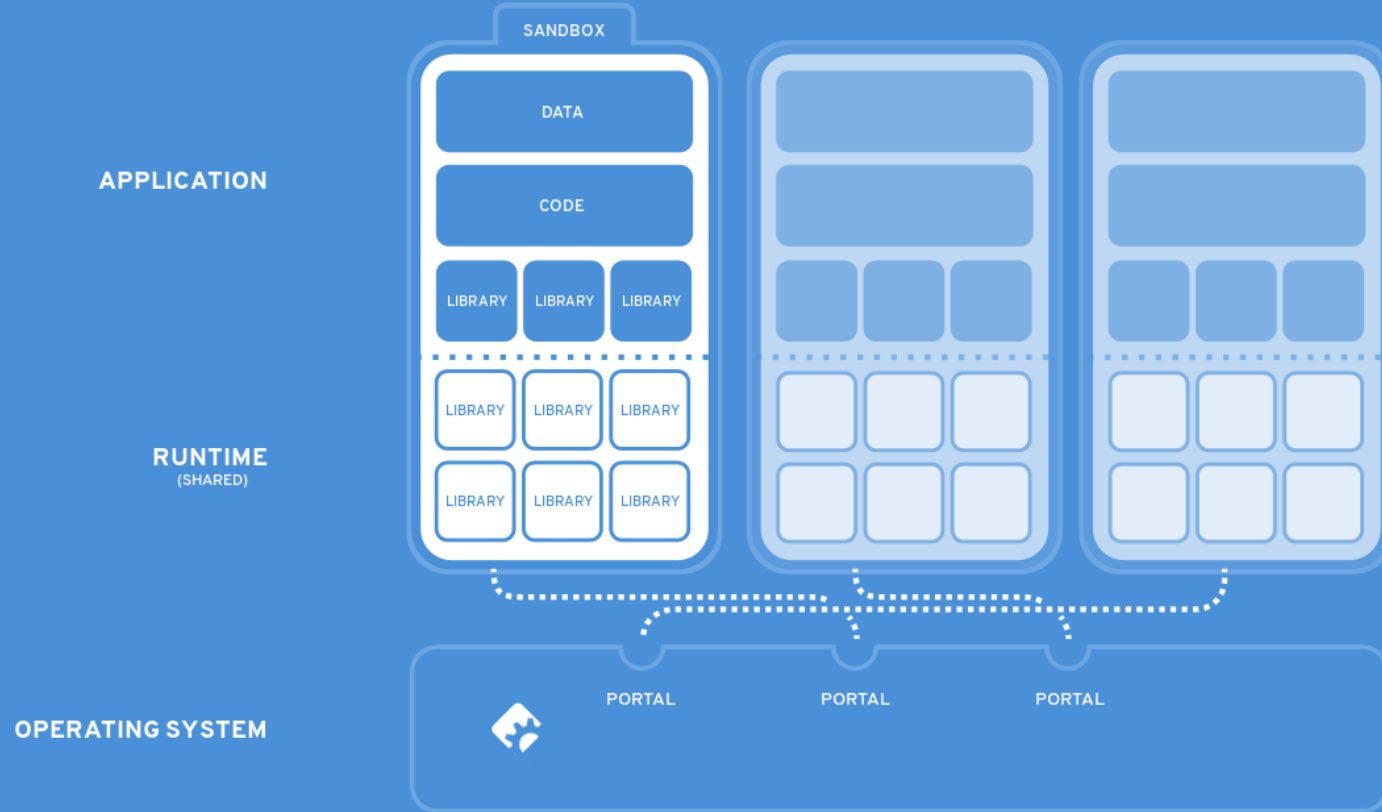
- Compatibility – SOLVED
 - Only compatible libraries in the bundle
- Portability – SOLVED
 - All dependencies in the bundle
- Pace of Change – SOLVED
 - App developers can distribute at their pace, not a distro pace
- “It just works” - SOLVED



Compatibility & Portability



Compatibility & Portability



Compatibility & Portability

- Containerised Apps at some point make assumptions of a common standard base provided by the Distribution
- No such common base exists in a practical sense



Compatibility & Portability

not required.

2. **Gather suitable binaries of all dependencies** that are not part of the base operating systems you are targeting. For example, if you are targeting Ubuntu, Fedora, and openSUSE, then you need to gather all libraries and other dependencies that your app requires to run that are not part of Ubuntu, Fedora, and openSUSE.
3. **Create a working AppDir** from your binaries. A working AppImage runs your app when you execute its AppRun file.



Compatibility & Portability

- For a Containerised App to be portable, it must contain ALL compatible dependencies which MIGHT not be provided by ANY distribution
- If not, expect crashes



So it's hopeless?

If everything is still liable to break, what is the point?

- Frameworks/Runtimes attempt to mitigate by providing curated 'Middledistros' to build Applications for
- The "Real" Solution: A well defined Linux Standard Base?



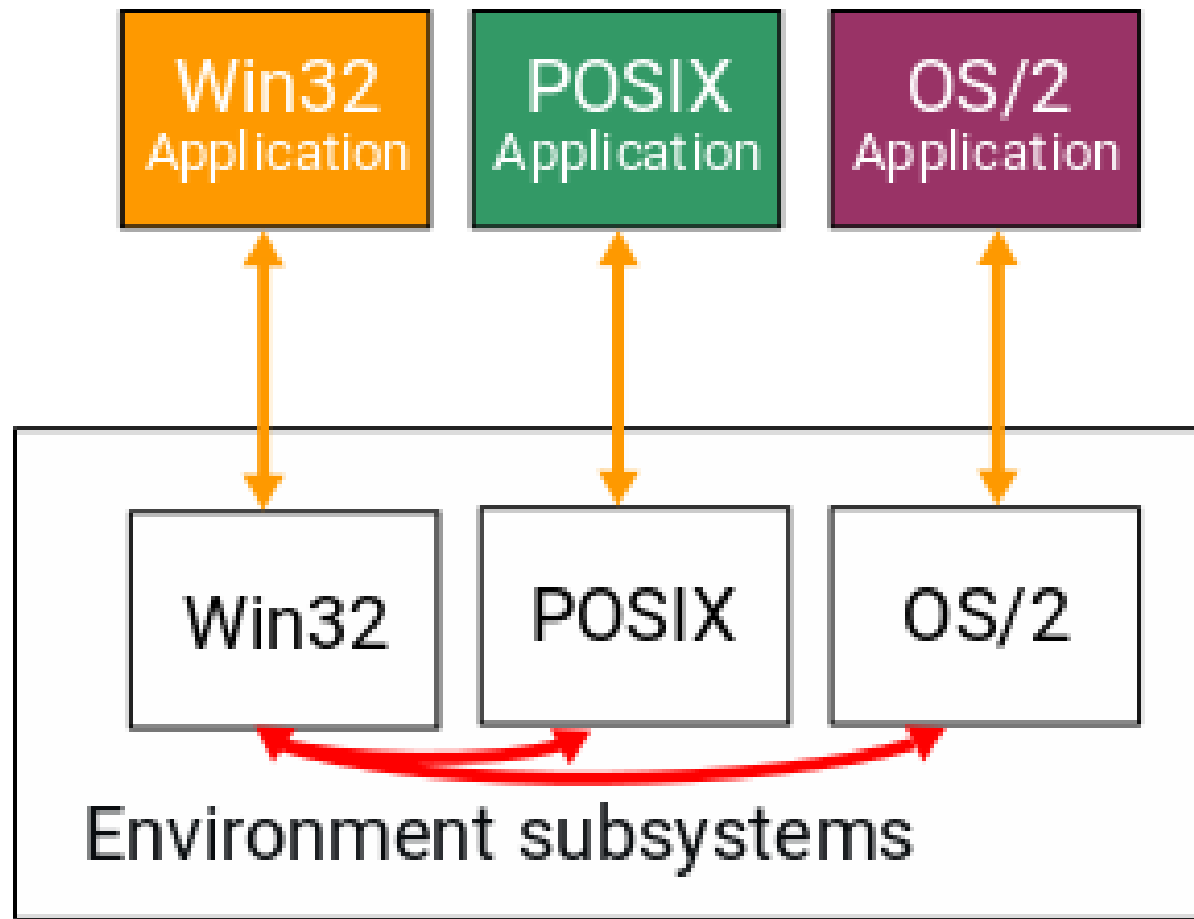
The Big Promises - Reality

- Compatibility – ~~SOLVED~~
 - Only compatible libraries in the bundle
- Portability – ~~SOLVED~~
 - All dependencies in the bundle
- Pace of Change – SOLVED
 - App developers can distribute at their pace, not a distro pace
- “It just works” - ?



Wait a second...





History Repeating?

- Security nightmare?
 - Security relevant libs lurking in countless application bundles
- Maintenance nightmare?
 - How are we going to update our app and every single lib?
- Legal nightmare?
 - Can we legally redistribute all the libs we need to?
- Storage vendor dream
 - More disk consumption, everyone buying bigger disks!





“With Great Power...”

“... Comes Great Responsibilities”

- AppImage/FlatPak/Snappy are tools that enable App Developers to directly distribute software without the ‘need’ for Distributions
- Therefore, they must adopt the responsibilities which come with being a distributor of software



Compatibility & Portability

Consider everything an App needs that isn't in the Bundle

- Can this break my App if the ABI changes?
 - If YES, then move it to the Bundle
- Can I rely on it being there on ALL systems?
 - If NO, then move it to the Bundle



Compatibility & Portability in Real Teams

Application Developers will still need to

- Dev & test Apps on every possible distro
- Then retest every App patch on every possible distro
- Then cry when it all breaks anyway



Broader Responsibilities

- Security – Monitor & rapidly react to CVEs. Audit libraries. Do not assume sandboxing is enough.
- Maintenance – Update all bundled dependencies in a timely manner
- Legal – Review licences of all bundled dependencies and ensure compliance & compatibility



Distributions can be part of the solution

- Distributions should like the promise of Containerised Applications
- Less work & responsibility for us is always good
- Should not be fearful of the transfer of responsibility, but should not encourage it blindly either



Distributions can be part of the solution

- A Common Base (“LSB for the Container Age”) must be considered
 - Without one, the portability promise is unachievable
- Distributions have decades of tools and talent for dealing with the broader issues. USE THEM
- Don’t reinvent every wheel just because we can



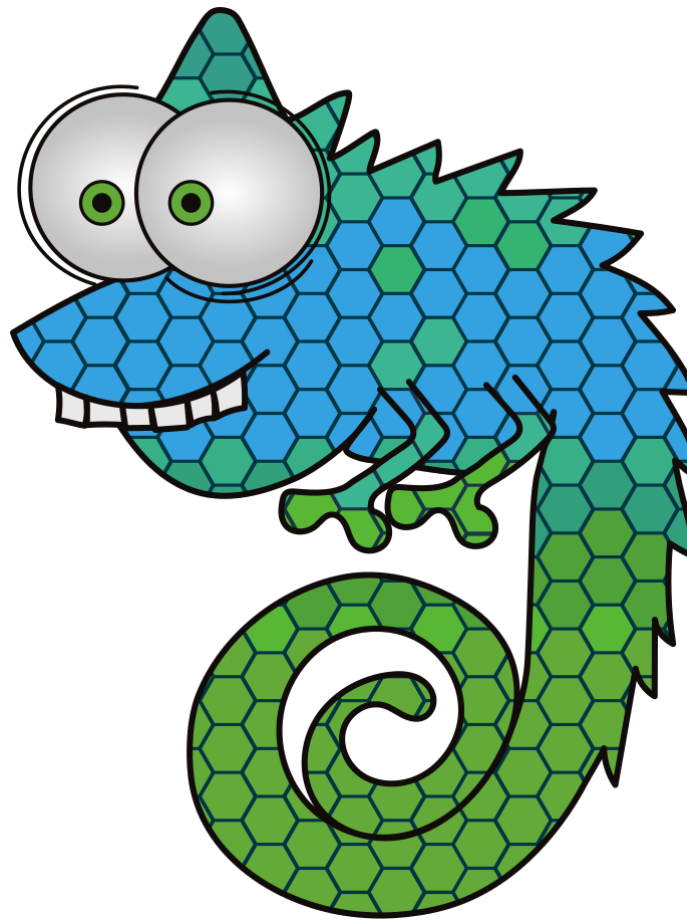


One more thing

Rolling Releases for Everyone?

- To get Applications in the hands of users fast, what model beats a rolling distribution?
- Users can be guaranteed an integrated “built together” experience
- Security/Maintenance burdens less broadly distributed, fewer points of failure, Devs don’t need to be security engineers
- “It just works” can be reached with good tools – OBS & openQA





Join Us at www.opensuse.org



License

This slide deck is licensed under the Creative Commons Attribution-ShareAlike 4.0 International license. It can be shared and adapted for any purpose (even commercially) as long as Attribution is given and any derivative work is distributed under the same license.

Details can be found at <https://creativecommons.org/licenses/by-sa/4.0/>

General Disclaimer

This document is not to be construed as a promise by any participating organisation to develop, deliver, or market a product. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. openSUSE makes no representations or warranties with respect to the contents of this document, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. The development, release, and timing of features or functionality described for openSUSE products remains at the sole discretion of openSUSE. Further, openSUSE reserves the right to revise this document and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes. All openSUSE marks referenced in this presentation are trademarks or registered trademarks of SUSE LLC, in the United States and other countries. All third-party trademarks are the property of their respective owners.

Credits

Template

Richard Brown
rbrown@opensuse.org

Design & Inspiration

openSUSE Design Team

<http://opensuse.github.io/branding-guidelines/>

[Code](#)

Issues 30

Pull requests 8

Pulse

Graphs

Tree: 6e26fd... [wordpress](#) / [apache](#) / Dockerfile

Find file

Copy path

 tianon Update to 4.5

6e26fd1 Apr 13, 2016

6 contributors



38 lines (29 sloc) | 1.28 KB

Raw

Blame

History



```
1 FROM php:5.6-apache
2
3 RUN a2enmod rewrite expires
4
5 # install the PHP extensions we need
6 RUN apt-get update && apt-get install -y libpng12-dev libjpeg-dev && rm -rf /var/lib/apt/lists/* \
7     && docker-php-ext-configure gd --with-png-dir=/usr --with-jpeg-dir=/usr \
8     && docker-php-ext-install gd mysqli opcache
9
10 # set recommended PHP.ini settings
11 # see https://secure.php.net/manual/en/opcache.installation.php
12 RUN [ \
13     echo 'opcache.memory_consumption=128'; \
14     echo 'opcache.interned_strings_buffer=8'; \
15     echo 'opcache.max_accelerated_files=4000'; \
16     echo 'opcache.revalidate_freq=60'; \
17     echo 'opcache.fast_shutdown=1'; \
18     echo 'opcache.enable_cli=1'; \
19 ] > /usr/local/etc/php/conf.d/opcache-recommended.ini
20
21 VOLUME /var/www/html
22
23 ENV WORDPRESS_VERSION 4.5
24 ENV WORDPRESS_SHA1 439f09e7a948f02f00e952211a22b8bb0502e2e2
25
26 # upstream tarballs include ./wordpress/ so this gives us /usr/src/wordpress
27 RUN curl -o wordpress.tar.gz -SL https://wordpress.org/wordpress-${WORDPRESS_VERSION}.tar.gz \
28     && echo "$WORDPRESS_SHA1 *wordpress.tar.gz" | shasum -c - \
29     && tar -xzf wordpress.tar.gz -C /usr/src/ \
30     && rm wordpress.tar.gz \
31     && chown -R www-data:www-data /usr/src/wordpress
32
33 COPY docker-entrypoint.sh /entrypoint.sh
34
35 # grr, ENTRYPOINT resets CMD now
36 ENTRYPOINT ["/entrypoint.sh"]
37 CMD ["apache2-foreground"]
```

docker-library / wordpress

Watch 24 Star 195 Fork 144

Code Issues 30 Pull requests 8 Pulse Graphs

Tree: 6e26fd... wordpress / apache / Dockerfile Find file Copy path

tianon Update to 4.5 6e26fd1 Apr 13, 2016

6 contributors

38 lines (29 sloc) 1.28 KB Raw Blame History

```
1 FROM php:5.6-apache
2
3 RUN a2enmod rewrite expires
4
5 # install the PHP extensions we need
6 RUN apt-get update && apt-get install -y libpng12-dev libjpeg-dev && rm -rf /var/lib/apt/lists/* \
7     && docker-php-ext-configure gd --with-png-dir=/usr --with-jpeg-dir=/usr \
8     && docker-php-ext-install gd mysqli opcache
9
10 # set recommended PHP.ini settings
11 # see https://secure.php.net/manual/en/opcache.installation.php
12 RUN [ \
13     echo 'opcache.memory_consumption=128'; \
14     echo 'opcache.interned_strings_buffer=8'; \
15     echo 'opcache.max_accelerated_files=4000'; \
16     echo 'opcache.revalidate_freq=60'; \
17     echo 'opcache.fast_shutdown=1'; \
18     echo 'opcache.enable_cli=1'; \
19 ] > /usr/local/etc/php/conf.d/opcache-recommended.ini
20
21 VOLUME /var/www/html
22
23 ENV WORDPRESS_VERSION 4.5
24 ENV WORDPRESS_SHA1 439f09e7a948f02f00e952211a22b8bb0502e2e2
25
26 # upstream tarballs include ./wordpress/ so this gives us /usr/src/wordpress
27
28
29
30
31
32
33 COPY docker-entrypoint.sh /entrypoint.sh
34
35 # grr, ENTRYPOINT resets CMD now
36 ENTRYPOINT ["/entrypoint.sh"]
37 CMD ["apache2-foreground"]
```

RUN curl -o wordpress.tar.gz

-SL https://wordpress.org/wordpress-\$WORDPRESS_VERSION}.tar.gz



<> Code

Issues 30

Pull requests 8

Pulse

Graphs

Tree: 6e26fd...

wordpress / apache / Dockerfile

Find file

Copy path

tlanon Update to 4.5

6e26fd1 Apr 13, 2016

6 contributors



FROM php:5.6-apache

Raw

Blame

History



```
2
3  RUN a2enmod rewrite expires
4
5  # install the PHP extensions we need
6  RUN apt-get update && apt-get install -y libpng12-dev libjpeg-dev && rm -rf /var/lib/apt/lists/* \
7      && docker-php-ext-configure gd --with-png-dir=/usr --with-jpeg-dir=/usr \
8      && docker-php-ext-install gd mysqli opcache
9
10 # set recommended PHP.ini settings
11 # see https://secure.php.net/manual/en/opcache.installation.php
12 RUN [ \
13     echo 'opcache.memory_consumption=128'; \
14     echo 'opcache.interned_strings_buffer=8'; \
15     echo 'opcache.max_accelerated_files=4000'; \
16     echo 'opcache.revalidate_freq=60'; \
17     echo 'opcache.fast_shutdown=1'; \
18     echo 'opcache.enable_cli=1'; \
19 ] > /usr/local/etc/php/conf.d/opcache-recommended.ini
20
21 VOLUME /var/www/html
22
23 ENV WORDPRESS_VERSION 4.5
24 ENV WORDPRESS_SHA1 439f09e7a948f02f00e952211a22b8bb0502e2e2
25
26 # upstream tarballs include ./wordpress/ so this gives us /usr/src/wordpress
27 RUN curl -o wordpress.tar.gz -SL https://wordpress.org/wordpress-${WORDPRESS_VERSION}.tar.gz \
28     && echo "$WORDPRESS_SHA1 *wordpress.tar.gz" | shasum -c - \
29     && tar -xzf wordpress.tar.gz -C /usr/src/ \
30     && rm wordpress.tar.gz \
31     && chown -R www-data:www-data /usr/src/wordpress
32
33 COPY docker-entrypoint.sh /entrypoint.sh
34
35 # grr, ENTRYPOINT resets CMD now
36 ENTRYPOINT ["/entrypoint.sh"]
37 CMD ["apache2-foreground"]
```



```
docker-library / php
Watch - 16 ★ Star 334 Fork 305
Code Issues 44 Pull requests 7 Pulse Clippy
Branch: master - php / 5.6 / apache / Dockerfile Find file Copy path
Hanan update to 7.0.5, 5.6.20, 5.5.34 4077ca1 Mar 31, 2016
16 contributors
101 lines (86 sloc) 3.5 kB Raw Blame History
1 FROM debian:jessie
2
3 # $@:2:4 $@ps
4 RUN apt-get update && apt-get install -y \
5     autoconf \
6     file \
7     g++ \
8     gcc \
9     libc-dev \
10    make \
11    pkg-config \
12    runc \
13    --no-install-recommends && rm -f /var/lib/apt/lists/*
14
15 # persistent / run:1:4 $@ps
16 RUN apt-get update && apt-get install -y \
17     ca-certificates \
18     curl \
19     libbz2 \
20     libbz2-dev \
21     libbz2 \
22    --no-install-recommends && rm -f /var/lib/apt/lists/*
23
24 RUN PHP_DIR=/usr/local/etc/php
25 RUN mkdir -p $PHP_DIR/conf.d
26
27 #enable-opcache
28 RUN apt-get update && apt-get install -y apache2-bin apache2.2-common --no-install-recommends && rm -rf /var/lib/apt/lists/*
29
30 RUN rm -rf /var/aaa/html && mkdir -p /var/lock/apache2 /var/run/apache2 /var/log/apache2 /var/aaa/html && chown -R www-data:www-data
31
32 # Apache + PHP requires preferring apache for best results
33 RUN additoid opm_event && addmod opm_prefork
34
35 RUN mv /etc/apache2/apache2.conf /etc/apache2/apache2.conf.dist && rm /etc/apache2/conf-enabled/* /etc/apache2/sites-enabled/*
36 COPY apache2.conf /etc/apache2/apache2.conf
37 # it's better to copy the conf file until the end of the dockerfile, but its contents are checked by PHP during compile
38
39 ENV PHP_EXTRA_BUILD_DEPS apache2-dev
40 ENV PHP_EXTRA_CONFIGURE_OPTS --with-apache
41 #enable-opcache
42
43 ENV PHP_VERSION 5.6.20
44 ENV PHP_FILE_NAME php-5.6.tar.gz
45 ENV PHP_SHA256 208762321112149157a35686c0f03ac807c1c0844032b1f4447a
46
47 RUN set -e \
48     && buildDeps=" \
49         $PHP_EXTRA_BUILD_DEPS \
50         libcurl-openssl-dev \
51         libbz2-dev \
52         libbz2-dev \
53         libbz2 \
54         libbz2-dev \
55         libbz2 \
56         xz-utils \
57     " \
58     && apt-get update && apt-get install -y $buildDeps --no-install-recommends && rm -rf /var/lib/apt/lists/* \
59     && curl -fsSL "http://php.net/get/$PHP_FILE_NAME/from/this/mirror" -o "$PHP_FILE_NAME" \
60     && echo "$PHP_SHA256 $PHP_FILE_NAME" | sha256sum -c - \
61     && curl -fsSL "http://php.net/get/$PHP_FILE_NAME.asc/from/this/mirror" -o "$PHP_FILE_NAME.asc" \
62     && echo "$(cat "$PHP_FILE_NAME.asc" | tr -d '\n')" | sha256sum -c - \
63     && for key in $PHP_VERSION; do \
64         gpg --keyserver ha.pool.sks-keyservers.net --recv-keys "$key"; \
65     done \
66     && gpg --batch --verify "$PHP_FILE_NAME.asc" "$PHP_FILE_NAME" \
67     && rm -f "$PHP_FILE_NAME.asc" \
68     && mkdir -p /usr/src/php \
69     && tar -xzf "$PHP_FILE_NAME" -C /usr/src/php --strip-components=1 \
70     && rm "$PHP_FILE_NAME" \
71     && cd /usr/src/php \
72     && ./configure \
73         --with-config-file-path="$PHP_DIR" \
74         --with-config-file-path="$PHP_DIR/conf.d" \
75         $PHP_EXTRA_CONFIGURE_OPTS \
76         --enable-cgi \
77     # --enable-embed is included here because it's harder to compile w/other the fact than extensions are (since it's a plugin for some
78     # --enable-embed is included here because otherwise there's no way to get perl to use it properly (see https://github.com/docker
79     --enable-embed \
80     --enable-embed=shared \
81     --enable-embed=shared \
82     --with-curl \
83     --with-libbz2 \
84     --with-openssl \
85     --with-zlib \
86     && make -j "$(nproc)" \
87     && make install \
88     && [ $(find /usr/local/bin /usr/local/sbin -type f -executable -exec strip --strip-all '{}' +) ] || true; \
89     && make clean \
90     && apt-get purge -y --auto-remove -o APT::AutoRemove::RecommendsImportant=false -o APT::AutoRemove::SuggestsImportant=false
91
92 COPY docker-php-ext.* /usr/local/bin/
93 #enable-opcache
94 COPY apache2-foreground /usr/local/bin/
95 WORKDIR /var/aaa/html
96
97 EXPOSE 80
98 CMD ["apache2-foreground"]
99 #enable-opcache
```



```
docker-library / php
Watch - 16 Star 234 Fork 265
Code Issues 44 Pull requests 7 Pulse Clippy
Branch: master - php / 5.6 / apache / Dockerfile Find file Copy path
FROM debian:jessie
2
3 # Build2x deps
4 RUN apt-get update && apt-get install -y \
5     autoconf \
6     file \
7     gcc \
8     gcc \
9     libc-dev \
10    make \
11    pkg-config \
12    runc \
13    --no-install-recommends && rm -f /var/lib/apt/lists/*
14
15 # persistent / runtime deps
16 RUN apt-get update && apt-get install -y \
17     ca-certificates \
18     curl \
19     libbz2-dev \
20     libbz2-dev \
21     libbz2 \
22     --no-install-recommends && rm -f /var/lib/apt/lists/*
23
24 RUN PHP_DIR=/usr/local/etc/php
25 RUN mkdir -p $PHP_DIR/conf.d
26
27 #enableopcache
28 RUN apt-get update && apt-get install -y apache2-bin apache2.2-common --no-install-recommends && rm -rf /var/lib/apt/lists/*
29
30 RUN rm -rf /var/aaa/html && mkdir -p /var/lock/apache2 /var/run/apache2 /var/log/apache2 /var/aaa/html && chown -R www-data:www-data
31
32 # Apache + PHP requires preferring apache for best results
33 RUN additoid opm_event && addmod ssl_module
34
35 RUN mv /etc/apache2/apache2.conf /etc/apache2/apache2.conf.dist && rm /etc/apache2/conf-enabled/* /etc/apache2/sites-enabled/*
36 COPY apache2.conf /etc/apache2/apache2.conf
37 # it's best to copy if we could not COPY apache2.conf until the end of the dockerfile, but its contents are checked by PHP during compile
38
39 ENV PHP_EXTRA_BUILD_DEPS apache2-dev
40 ENV PHP_EXTRA_CONFIGURE_OPTS --with-apache
41 #enableopcache
42
43 ENV PHP_VERSION 5.6.28
44 ENV PHP_FILE_NAME php-5.6.tar.gz
45 ENV PHP_SHA256 289746223211224745157a35686dc0f03ac807c731c80844032b1f4447a
46
47 RUN set -e \
48     && builddeps=" \
49         $PHP_EXTRA_BUILD_DEPS \
50         libcurl4-openssl-dev \
51         libbz2-dev \
52         libbz2-dev \
53         libbz2 \
54         libbz2-dev \
55         libbz2 \
56         xz-utils \
57     " \
58     && apt-get update && apt-get install -y $buildDeps --no-install-recommends && rm -rf /var/lib/apt/lists/* \
59     && curl -fsSL "http://php.net/get/$PHP_FILE_NAME/from/this/mirror" -o "$PHP_FILE_NAME" \
60     && echo "$PHP_SHA256 $PHP_FILE_NAME" | sha256sum -c - \
61     && curl -fsSL "http://php.net/get/$PHP_FILE_NAME.asc/from/this/mirror" -o "$PHP_FILE_NAME.asc" \
62     && echo "$(cat "$PHP_FILE_NAME.asc")" | sha256sum -c - \
63     && for key in $PHP_VERSION; do \
64         gpg --keyserver ha.pool.sks-keyservers.net --recv-keys "$key"; \
65     done \
66     && gpg --batch --verify "$PHP_FILE_NAME.asc" "$PHP_FILE_NAME" \
67     && rm -f "$PHP_FILE_NAME.asc" \
68     && mkdir -p /usr/src/php \
69     && tar -xzf "$PHP_FILE_NAME" -C /usr/src/php --strip-components=1 \
70     && rm "$PHP_FILE_NAME" \
71     && cd /usr/src/php \
72     && ./configure \
73     --with-config-file-path="$PHP_DIR" \
74     --with-config-file-path="$PHP_DIR/conf.d" \
75     $PHP_EXTRA_CONFIGURE_OPTS \
76     --enable-cgi \
77     --enable-embed=1 \
78     --enable-embed=1 \
79     --enable-embed=1 \
80     --enable-embed=1 \
81     --enable-embed=1 \
82     --enable-embed=1 \
83     --enable-embed=1 \
84     --enable-embed=1 \
85     --enable-embed=1 \
86     --enable-embed=1 \
87     --enable-embed=1 \
88     --enable-embed=1 \
89     --enable-embed=1 \
90     --enable-embed=1 \
91     --enable-embed=1 \
92     --enable-embed=1 \
93     --enable-embed=1 \
94     --enable-embed=1 \
95     --enable-embed=1 \
96     --enable-embed=1 \
97     --enable-embed=1 \
98     --enable-embed=1 \
99     --enable-embed=1 \
100    --enable-embed=1
101 #enableopcache
102 COPY docker-php-ext.* /usr/local/bin/
103 #enableopcache
104 COPY apache2-foreground /usr/local/sbin/
105 WORKDIR /var/aaa/html
106 EXPOSE 80
107 CMD ["apache2-foreground"]
108 #enableopcache
```

tianon / **docker-brew-debian**

Watch 18

Star 64

Fork 38

Code

Issues 3

Pull requests 0

Wiki

Pulse

Graphs

Tree: 04fb8b...

docker-brew-debian / **jessie** / **Dockerfile**

Find file

Copy path

 **tianon** 2016-03-01 debootstraps (CVE-2016-0800, CVE-2016-0705, CVE-2016-0798,...)

d431f09 Mar 1, 2016

1 contributor

4 lines (3 sloc) | 51 Bytes

Raw

Blame

History



```
1 FROM scratch
2 ADD rootfs.tar.xz /
3 CMD ["/bin/bash"]
```