Published June 1, 2021 | Version v2

Dataset Open

Software Environments in Binder Containers

Shaffer, Tim[1]; Chard, Kyle[2]; Thain, Douglas[1]

Show affiliations

Binder is a publicly accessible online service for executing interactive notebooks based on Git repositories. Binder dynamically builds and deploys containers following a recipe stored in the repository, then gives the user a browser-based notebook interface. The Binder group periodically releases a log of container launches from the public Binder service. Archives of launch records are available here. These records do not include identifiable information like IP addresses, but do give the source repo being launched along with some other metadata. The main content of this dataset is in the `binder.sqlite` file. This SQLite database includes launch records from 2018-11-03 to 2021-06-06 in the `events` table, which has the following schema.

```
CREATE TABLE events(
    version INTEGER,
    timestamp TEXT,
    provider TEXT,
    spec TEXT,
    origin TEXT,
    ref TEXT,
    guessed_ref TEXT
);
CREATE INDEX idx_timestamp ON events(timestamp);
```

- `version` indicates the version of the record as assigned by Binder. The `origin` field became available with version 3, and the `ref` field with version 4. Older records where this information was not recorded will have the corresponding fields set to null.
- `timestamp` is the ISO timestamp of the launch
- `provider` gives the type of source repo being launched ("GitHub" is by far the most common). The rest of the explanations assume GitHub, other providers may differ.
- `spec` gives the particular branch/release/commit being built. It consists of `<github-id>/<repo>/<branch>`.
- `origin` indicates which backend was used. Each has its own storage, compute, etc. so this info might be important for evaluating caching and performance. Note that only recent records include this field. May be null.
- `ref` specifies the git commit that was actually used, rather than the named branch referenced by `spec`. Note that this was not recorded from the beginning, so only the more recent entries include it. May be null.
- For records where `ref` is not available, we attempted to clone the named reference given by `spec` rather than the specific commit (see below). The `guessed_ref` field records the commit found at the time of cloning. If the branch was updated since the container was launched, this will not be the exact version that was used, and instead will refer to whatever was available at the time (early 2021). Depending on the application, this might still be useful

information. Selecting only records with version 4 (or non-null `ref`) will exclude these guessed commits. May be null.

The Binder launch dataset identifies the source repos that were used, but doesn't give any indication of their contents. We crawled GitHub to get the actual specification files in the repos which were fed into repo2docker when preparing the notebook environments, as well as filesystem metadata of the repos. Some repos were deleted/made private at some point, and were thus skipped. This is indicated by the absence of any row for the given commit (or absence of both `ref` and `guessed_ref` in the `events` table). The schema is as follows.

```
CREATE TABLE spec_files (
    ref TEXT NOT NULL PRIMARY KEY,
    ls TEXT,
    runtime BLOB,
    apt BLOB,
    conda BLOB,
    pip BLOB,
    pipfile BLOB,
    julia BLOB,
    r BLOB,
    nix BLOB,
    docker BLOB,
    setup BLOB,
    postbuild BLOB,
    start BLOB
);
```

Here `ref` corresponds to `ref` and/or `guessed_ref` from the `events` table. For each repo, we collected spec files into the following fields (see the repo2docker docs for details on what these are). The records in the database are simply the verbatim file contents, with no parsing or further processing performed.

- `runtime`: `runtime.txt`
- `apt`: `apt.txt`
- `conda`: `environment.yml`
- `pip`: `requirements.txt`
- `pipfile`: `Pipfile.lock` or `Pipfile`
- `julia`: `Project.toml` or `REQUIRE`
- `r`: `install.R`
- `nix`: `default.nix`
- `docker`: `Dockerfile`
- `setup`: `setup.py`
- `postbuild`: `postBuild`
- `start`: `start`

The `ls` field gives a metadata listing of the repo contents (excluding the `.git` directory). This field is JSON encoded with the following structure based on JSON types:

- Object: filesystem directory. Keys are file names within it. Values are the contents, which can be regular files, symlinks, or subdirectories.
- String: symlink. The string value gives the link target.
- Number: regular file. The number value gives the file size in bytes.

```
CREATE TABLE clean_specs (
    ref TEXT NOT NULL PRIMARY KEY,
    conda_channels TEXT,
    conda_packages TEXT,
    pip_packages TEXT,
    apt_packages TEXT
);
```

The `clean_specs` table provides parsed and validated specifications for some of the specification files (currently Pip, Conda, and APT packages). Each column gives either a JSON encoded list of package requirements, or null. APT packages have been validated using a regex adapted from the repo2docker source. Pip packages have been parsed and normalized using the Requirement class from the pkg_resources package of setuptools. Conda packages have been parsed and normalized using the `conda.models.match_spec.MatchSpec` class included with the library form of Conda (distinct from the command line tool). Users might want to use these parsers when working with the package data, as the specifications can become fairly complex.

The `missing` table gives the repos that were not accessible, and `event_logs` records which log files have already been added. These tables are used for updating the dataset and should not be of interest to users.

Files

Files (5.0 GB)

binder.sqlite

md5:230c727e8893ae4d4ce88db04995c33f

5.0 GB

Download

Citations

Show only:

Search for citation ... Search

Literature (0) Dataset (0) Software (0)

Unknown (0) Citations To This Version

No citations found

805 116

VIEWS DOWNLOADS

Show more details

Versions

Version v2 Jun 1, 2021
10.5281/zenodo.4915858

Version v1 Jun 1, 2021
10.5281/zenodo.4891791

View all 2 versions

Cite all versions? You can cite all versions by using the DOI 10.5281/zenodo.4891790. This DOI represents all versions, and will always resolve to the latest one. Read more.

External resources

Indexed in

OpenAIRE

Details

DOI

DOI 10.5281/zenodo.4915858

Resource type
Dataset

Publisher
Zenodo

Rights

License

Citation

Shaffer, T., Chard, K., & Thain, D. (2021). Software Environments in Binder Containers [Data set]. Zenodo. https://doi.org/10.5281/zenodo.4915858

Style APA

Export

JSON

Export

Technical metadata

Created June 9, 2021

Modified September 5, 2021

Jump up