Apache Distill Documentation

Release 1.0 alpha

Michelle Beard

Jun 22, 2016

CONTENTS

Apache Distill is an analytical framework and backend server to support Apache Tap requests. It can also be used standalone as a RESTful service.

CHAPTER

ONE

USER'S GUIDE

1.1 Installation Guide

1.1.1 Installing Distill in an Development Environment

The first step is to install Distill. First, checkout the latest version of Distill.

```
$ git clone https://github.com/draperlaboratory/distill.git
```

Distill is a python project, so it can be installed like any other python library. Several operating systems (Mac OS X, Major Versions of Linux/BSD) have Python pre-installed, so you should just have to run

```
$ easy_install distill
```

or

```
$ pip install distill
```

Advanced users can install Distill in a virtualenv if they wish. Instructions to setup an virtual environment will be explained below.

Note: When the package is installed via `easy_install` or `pip` this function will be bound to the `distill` executable in the Python installation's `bin` directory (on Windows - the `Scripts` directory).

1.1.2 Installing Distill in an Virtual Environment

TODO

1.1.3 Development and Testing

To build the source code and run all unit tests.

```
$ python setup.py develop test
```

To start up a local web server, running on localhost:8090.

```
$ dev
```

1.1.4 Deployment

I will describe a setup with nginx as a web server on Ubuntu. A web server cannot communicate directly with a Flask application such as Distill, that's why gunicorn will be used to act as a medium between the web server and Distill. Gunicorn is like an application web server that will be running behind nginx, and it is WSGI compatible. It can communicate with applications that support WSGI – Flask, Django, etc.

Install requirements.

```
$ sudo apt-get update
$ sudo apt-get install -y python python-pip nginx gunicorn
```

Create a directory to store the project.

```
$ sudo mkdir /home/pubic_html && cd /home/public_html
```

Download the project from the GitHub repository and copy the application to the `/home/public_html` directory.

```
$ git clone https://github.com/draperlaboratory/distill.git /home/public_html
```

Install Distill's requirements.

```
$ pip install -r requirements.txt
```

Distill has provided an nginx configuration file located in `distill/deploy/nginx.conf`.

Gunicorn will use port 8000 and handle the incoming HTTP requests.

Restart nginx to load the configuration changes.

```
$ sudo /etc/init.d/nginx restart
```

Run gunicorn on port 8000.

```
$ gunicorn --workers 4 --bind unix:distill.sock -m 007 deploy/run_server:app
```

Start a new browser instance and navigate to http://localhost.

1.1.5 Installing Documentation

To save yourself the trouble, all up to date documentation is available at https://draperlaboratory.github.io/distill/.

However, if you want to manully build the documentation, the instructions are below.

To build Distill's documentation, create a directory at the root level of `/distill` called distill-docs.

```
$ mkdir distill-docs & cd distill/docs
```

Execute build command:

```
# Inside top-level docs/ directory.
$ make html
```

This should build the documentation in your shell, and output HTML. At then end, it should say something about documents being ready in `distill-docs/html`. You can now open them in your browser by typing

```
$ open distill-docs/html/index.html
```

1.2 Quickstart Guide

1.2.1 Usage

Using `curl`:

```
$ curl -XGET 'http://localhost:8090/app/register' -d '{
        "application_name" : "my_app",
        "version" : "0.1",
        "application_description" : "my test app"
}'
```

CHAPTER

TWO

API REFERENCE

This entire section is mainly for Developers of Distill. This section was automatically generated by Sphinx and apidoc.

2.1 API Documentation

2.1.1 Distill RESTful API

`distill.app.`**`create`**`(`*app_id*`)`

Registers an application in Distill.

```
$ curl -XPOST https://localhost:8000/xdata_v3
```

Parameters **`app_id`** – Application name

Returns Newly created application's status as JSON blob

`distill.app.`**`delete`**`(`*app_id*`)`

Deletes an application permentantly from Distill

```
$ curl -XDELETE https://localhost:8000/xdata_v3
```

Parameters **`app_id`** – Application name

Returns Boolean response message as JSON blob

`distill.app.`**`denoise`**`(`*app_id*`)`

Bootstrap script to cleanup the raw logs. A document type called "parsed" will be stored with new log created unless specified in the request. Have option to save parsed results back to data store. These parsed logs can be intergrated with STOUT results by running the stout bootstrap script.

```
$ curl -XGET https://localhost:8000/denoise/xdata_v3?save=true&type=parsed
```

Parameters **`app_id`** – Application name

Returns JSON blob of status

`distill.app.`**`index`**`()`

Show Distill version information, connection status, and all registered applications.

```
$ curl -XGET https://localhost:8000

{
        "author" : "Michelle Beard",
        "email" : "mbeard@draper.com",
        "name": "Distill",
        "status" : true,
        "version" : "1.0",
        "applications" : {
                "xdata_v3" : {
                        testing: 205,
                        parsed: 500,
                },
                "test_app" : {
                        logs: 500,
                        parsed: 100,
                }
        }
}
```

Returns Distill's status information as JSON blob

`distill.app.`**`merge_stout`**`()`

Bootstrap script to aggregate user ale logs to stout master answer table This will save the merged results back to ES instance at new index stout OR denoise data first, then merge with the stout index... If STOUT is enabled, the select method expects a stout index to exist or otherwise it will return an error message.

```
$ curl -XGET https://locahost:8000/stout/xdata_v3
```

Returns Status message

`distill.app.`**`page_not_found`**`(`*error*`)`

Generic Error Message

`distill.app.`**`search`**`(`*app_id*, *app_type*`)`

Search against an application on various fields.

```
$ curl -XGET https://[hostname]:[port]/search/xdata_v3?q=session_id:A1234&size=100&scroll=false&
```

Parameters

- **`app_id`** – Application name
- **`app_type`** – Optional document type to filter against
- **`q`** – Main search query. To return all documents, pass in q=*:*
- **`size`** – Maximum number of documents to return in request
- **`scroll`** – Scroll id if the number of documents exceeds 10,000
- **`fl`** – List of fields to restrict the result set

Returns JSON blob of result set

`distill.app.`**`stat`**`(`*app_id*, *app_type*`)`

Warning: Not implemented/available

Generic histogram counts for a single registered application filtered optionally by document type.

```
$ curl -XGET https://localhost:8000/xdata_v3/testing/?elem=signup&event=click
```

Parameters

- **`app_id`** – Application name
- **`app_type`** – Application type

Returns JSON blob of result set

`distill.app.`**`status`**`(`*app_id*`)`

Presents meta information about an registered application, including field names and document types.

```
$ curl -XGET https://localhost:8000/status/xdata_v3

{
  "application": "xdata_v3",
  "health": "green",
  "num_docs": "433",
  "status": "open"
}
```

Parameters **`app_id`** – Application name

Returns Registered applications meta data as JSON blob

`distill.app.`**`update`**`(`*app_id*`)`

Renames a specific application

```
$ curl -XPOST https://localhost:8000/update/xdata_v3?name="xdata_v4"
```

Parameters **`app_id`** – Application name

Returns Boolean response message as JSON blob

2.1.2 Distill Exceptions

exception `distill.exceptions.`**`Error`**

Bases: `exceptions.Exception`

Base class for exceptions.

exception `distill.exceptions.`**`ValidationError`**`(`*url, msg*`)`

Bases: *`distill.exceptions.Error`*

Exceptions raised for errors in validated a url.

2.1.3 Distill Validation Library

`distill.validation.`**`str2bool`**`(`*v*`)`

Convert string expression to boolean

Parameters **`v`** – Input value

Returns Converted message as boolean type

Return type bool

`distill.validation.`**`validate_request`**(*q*)

Parse out request message and validate inputs

Parameters **`q`** – Url query string

Raises **`ValidationError`** – if the query is missing required parameters

2.1.4 Distill Analytics

Graph Analytics

class `distill.algorithms.graphs.graph.`**`GraphAnalytics`**

Bases: `object`

Distill's graph analytics package. Apply graph algorithms to User Ale log data segmented with Stout.

static **`foo`**()

Statistics Package

class `distill.algorithms.stats.stats.`**`Stats`**

Bases: `object`

Distill's statistics package. Apply statistical algorithms to User Ale log data segmented with Stout.

static **`foo`**()

2.1.5 Distill Models

Stout Interface

distill: This package contains a flask app RESTful api for distill

This flask app exposes some restful api endpoints for querying User-ALE. Very similar to Lucene syntax for basic query operations.

Copyright 2016, The Charles Stark Draper Laboratory Licensed under Apache Software License.

class `distill.models.stout.`**`Stout`**

Bases: `object`

static **`ingest`**()

class `distill.models.stout.`**`StoutDoc`**(*meta=None*, ***kwargs*)

Bases: `elasticsearch_dsl.document.DocType`

`get_model_obj`()

`save`(**args*, ***kwargs*)

classmethod **`sync`**(*stout*)

`distill.models.stout.`**`parse`**()

UserAle Interface

class `distill.models.userale.`**`UserAle`**
Bases: `object`

Generic class supporting basic CRUD operations

static **`create`**(*app*)

static **`delete`**(*app*)

static **`denoise`**(*app*, *doc_type='parsed'*, *save=False*)

static **`getApps`**()
Fetch all the registered applications for an Elasticsearch instance.

Note: Privated indexes starting with a period are not included in the result set.

Returns A

Return type dict

static **`getStatus`**()
Fetch the status of an Elasticsearch instance.

Returns True/False if connection to Elasticsearch instance has been established.

Return type bool

static **`read`**(*app*)

static **`select`**(*app*, *app_type=None*, *params=None*)

static **`update`**(*app*)

class `distill.models.userale.`**`UserAleParsedDoc`**(*meta=None*, ***kwargs*)
Bases: `elasticsearch_dsl.document.DocType`

`save`(***kwargs*)

`distill.models.userale.`**`get_all_fields`**(*app*, *app_type=None*)

`distill.models.userale.`**`get_cluster_status`**(*app*)

`distill.models.userale.`**`merge_dicts`**(*lst*)

`distill.models.userale.`**`parse_mappings`**(*app*, *app_type=None*)

`distill.models.userale.`**`parse_query_parameters`**(*indx*, *app_type=None*, *request_args={}*)

2.1.6 Distill Deploy

Run Server

`distill.deploy.run_server.`**`dev_server`**()

2.1.7 Distill Utilities

Utilities Library

CHAPTER

THREE

ADDITIONAL NOTES

Design notes, legal information and changelog are here.

3.1 Authors

Distill is written and maintained by Michelle Beard and various contributors:

3.1.1 Development Lead

- Michelle Beard <mbeard@draper.com>

3.1.2 Additional Staff

- Laura Mariano <lmariano@draper.com>
- Dr. Joshua Poore <jpoore@draper.com>
- Clay Gimenez <cgimenez@draper.com>
- Steven York <syork@draper.com>

3.2 Distill Changelog

Here you can see the full list of changes between each Distill release.

3.2.1 Version 1.0

- Initial version.

3.3 License

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search