MozTrap Connect Documentation

Release 1.1

Cameron Dawson

Jul 19, 2017

Contents

Contents:

CHAPTER 1

Installation

Create a virtualenv to execute in, and then install the requirements:

```
pip install -r requirements/req.txt
```

CHAPTER 2

API

Connect

class mtconnect.connect.**Connect**(*protocol*, *host*, *username*, *api_key*, *limit=100*, *DEBUG=False*)

Connect to MozTrap to run tests and submit results.

You can either fetch tests for an existing run and submit results for it, or you can get tests for a productversion and create a new run with the new results.

get_product_cases(*productversion_id*, *environment_id=None*)

Return a list of test cases for the specified product version and environment (optional).

get_product_environments(*productversion_id*)

Return a list of environments for the specified product version.

get_products(*name=None*)

Return a list of Products with their productversions.

name - Filter by Product name

get_run_cases(*run_id*, *environment_id*)

Return a list of TestCase objects. Pass/Fail/Invalid is set on each object in the list. The list of environments is filtered by the environment, just as it would be in the UI of MozTrap.

get_run_environments(*run_id*, *name=None*)

Return a list of environments for the specified test run.

::Args:: - run_id - run id - name - environment name

get_runs(*product=None*, *version=None*, *productversion=None*, *name=None*, *run_id=None*)

Return a list of test runs.

::Args:: product - Filter by product name version - Filter by product version name productversion - Filter by product version id name - Filter by run name run_id - Filter by run id

::Raises:: InvalidFilterParamsException if neither run_id nor productversion_id nor product and version are provided.

submit_results(*run_id*, *testresults*)
Submit the results for an existing run.

Pass in a TestResults object with test results for case ids that match cases in MozTrap. This object can contain results for multiple environments. So you could run the cases over several environments and build a whole test run that shows coverage for all of that.

Results are not required for any of the tests.

submit_run(*name*, *description*, *productversion_id*, *testresults*)
Create a new, active run with results.

Pass in a TestResults object with test results for case ids that match cases in MozTrap. This object can contain results for multiple environments. So you could run the cases over several environments and build a whole test run that shows coverage for all of that.

Results are not required for any of the tests.

TestResults

class mtconnect.connect.**TestResults**
A holder for results of all tests that will be submitted.

addfail(*case_id*, *environment_id*, *comment*, *stepnumber=0*, *bug=None*)
Submit a failing result for a test case.

addinvalid(*case_id*, *environment_id*, *comment*)
Submit a result for a test case that is invalid or unclear.

addpass(*case_id*, *environment_id*)
Submit a passing result for a test case.

CHAPTER 3

Examples

New Run

Create a new Run with results

Commonly, an automated test will know the test case IDs that each test applies to, and will want to execute those tests, and create a run on the fly for those results.

Example:

```
from mtconnect.connect import Connect, TestResults
import json

connect = Connect("http", "localhost:8000", "camd", "abc123", DEBUG=True)

# get the product of "Zurago" and its product versions

products = connect.get_products(name="Zurago")
pv_id = products[0]["productversions"][0]["id"]

envs = connect.get_product_environments(productversion_id=pv_id)

# get the environment ids for the envs we care about

env_id = envs[0]["id"]

# get the cases for each env for the product version

cases = connect.get_product_cases(
    productversion_id=pv_id, environment_id=env_id)

# repository for the results we will be collecting

results = TestResults()
```

```
# submit tests for each case / env.  It's possible to
# submit results for the same case for multiple environments
# with the same results object.

results.addpass(case_id=243, environment_id=env_id)
results.addfail(
        case_id=244,
        environment_id=env_id,
        comment="dang thing..."
        stepnumber=3,
        bug="https://bugzilla.mycompany.com"
        )
results.addinvalid(
        case_id=245,
        environment_id=env_id,
        comment="what the?"
        )

# submit those results back to MozTrap

res = connect.submit_run(
    "Smoke tests for build: {0}".format(build_id),
    "The awesome smoketests",
    productversion_id=pv_id,
    testresults=results,
    )
```

Existing Run

Submit results for an existing Run.

If a test run already exists that you would like to submit results for, then this example is for you.

Example:

```
from mtconnect.connect import Connect, TestResults
import json

connect = Connect("http", "localhost:8000", "camd", "abc123", DEBUG=True)
runs = connect.get_runs()

# run you want
run_id=runs[0]["id"]

envs = connect.get_run_environments(run_id=run_id)

# env you want
env_id=envs[22]["id"]

tests = connect.get_run_cases(run_id=run_id, environment_id=env_id)
print jstr(tests)

# the object to accumulate all your test results

results = TestResults()
```

```
results.addpass(case_id=243, environment_id=env_id)
results.addfail(
        case_id=244,
        environment_id=env_id,
        comment="dang thing..."
        stepnumber=3,
        bug="https://bugzilla.mycompany.com"
        )
results.addinvalid(
        case_id=245,
        environment_id=env_id,
        comment="what the?"
        )

r = connect.submit_results(run_id=run_id, testresults=results)
```

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Index