workspace-tools Documentation

Release 0.2.20

Max Zheng

Apr 06, 2018

Contents

CHAPTER 1

workspace-tools

Tools to simplify working with multiple Python repositories by seamlessly integrating git and tox, where you can simply run one command instead of many native commands individually to do common tasks.

It is mostly a wrapper on top of existing tools with the end goal of providing a simple, seamless, and less repetive experience when working with one or more repositories. Feature support is mostly limited to what the author uses as, currently, it is foremost a personal tool to enhance the author's own productivity, but sharing it as others might find it useful.

CHAPTER 2

Overview

- One tool to seamlessly manage / integrate all workspace tools, from setup to publish.
- Simplified command execution for common workflow - just run one command, instead of many individual native ones.
- Command execution is also smart / optimized - i.e. test command auto detects requirement changes to redevelop.
- Path aware context commands that run across all checkouts - i.e. see status / diff for all repos.
- Get the most out of other products by easily update your dependencies to the latest
- Automatically install dependencies in editable mode for testing if configured
- Templates included to setup new product quickly
- Support for multiple branches with child/parent relationship.
- Cool and sensible shortcut aliases to help you do more by typing less - you will love “tv” [if you know ag]!

CHAPTER 3

Quick Start Tutorial

First, install it with:

```
$ pip install workspace-tools
```

Second, optionally setup environment with bash functions/aliases:

```
$ cd ~/workspace

$ wst setup --commands-with-aliases

[INFO] Added "ws" bash function with workspace directory set to ~/workspace
[INFO] Added bash functions: bump, checkout, clean, commit, log, publish, push,␣
↪status, test, update
[INFO] Added aliases: co=checkout, ci=commit, di=diff, st=status, up=update
[INFO] Added special aliases: a='source .tox/${PWD##*/}/bin/activate', d='deactivate',
↪ tv='open_files_from_last_command'  # from ag/ack/grep/find/which [t]o [v]im
[INFO] To use, run "source ~/.wstrc" or open a new shell.

$ source ~/.wstrc
```

To go to your workspace directory (if setup was run), run:

```
ws
```

To checkout a repo:

```
wst checkout https://github.com/maxzheng/workspace-tools.git

# Or checkout a group of repos as defined in workspace.cfg (using 'ws' alias from␣
↪setup)
# ws checkout mzheng-repos

# Or checkout a repo from GitHub (using 'co' alias from setup above, or use 'wst␣
↪checkout'):
# co workspace-tools                # Best match
# co maxzheng/workspace-tools        # Exact match
```

For more info about workspace.cfg, refer to Configuration doc.

The remaining tutorial will assume ‘wst setup’ was not run for the sake of clarity, though setup is recommended as there are many useful aliases provided.

To update all repos in your workspace concurrently:

```
wst update
```

Make a commit and create a new branch for it:

```
$ cd workspace-tools
# vi README.rst and make some changes

$ wst commit "Updated README.rst"

[updated-readme 0af8850] Updated README.rst
 1 file changed, 1 deletion(-)

# The commit created the branch 'updated-readme@master', added all files, and then␣
↪committed
# Notice the "@master" that indicates the parent branch. The parent branch will be␣
↪used
# during push with --merge and update. To specify a different branch without parent␣
↪relationship,
# use --branch option.
```

To install your test environment and test your change (with tox/py.test):

```
wst test

# To setup tox with test, style, and coverage environments, run:
# wst setup --product
#
# To check style or generate coverage report, run (using 'test' alias from setup):
# test style
# test cover
```

See status for all of your repos:

```
$ cd ~/workspace

$ wst status

[ bumper-lib ]
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   src/bumper/cars.py

no changes added to commit (use "git add" and/or "git commit -a")

[ clicast ]
```

```
# Branches: master display-changes@master fix-download@master

[ workspace-tools ]
# Branches: updated-readme@master master
```

See diff for all of your repos:

```
$ wst diff

[ bumper-lib ]
diff --git a/src/bumper/cars.py b/src/bumper/cars.py
index d552c2c..2d7bd12 100644
--- a/src/bumper/cars.py
+++ b/src/bumper/cars.py
@@ -281,7 +281,7 @@ class AbstractBumper(object):
   @classmethod
    def requirements_for_changes(self, changes):
       """
  -      Parse changes for requirements
  +      Parse changes for requirements.

         :param list changes:
       """
```

And finally amend the change and push:

```
$ cd workspace-tools
# vi README.rst and make more changes

$ wst commit --amend --push

[updated-readme@master 738f659] Updated README.rst
1 file changed, 2 insertions(+), 1 deletion(-)
Pushing updated-readme@master

# It will fail at push as you are not a committer, but the change was committed to␣
↪branch,
# and then merged into its parent branch (master).
```

Or simply push the change in your current branch:

```
wst push --merge

# This will update its parent branch (master), rebase branch with parent branch and␣
↪merge into
# parent branch if on child branch (child@parent) and then push.
# Upon success, it will remove the local and remote branch if pushing from child␣
↪branch.
```

If you have multiple upstream branches (defined by config [merge] branches) that you need to merge your change into, use auto merge:

```
# Assuming you are currently on 3.2.x branch and have these branches: 3.3.x, master
wst merge --all

[INFO] Merging 3.2.x into 3.3.x
[INFO] Pushing 3.3.x
```

```
[INFO] Merging 3.3.x into master
[INFO] Pushing master
```

If you have pinned your dependency requirements and want to update to latest version:

```
$ wst bump

[INFO] Updating workspace-tools
[INFO] Checking bumper-lib
...
[INFO] Checking requests
[bump ac06160] Require remoteconfig==0.2.4, requests==2.6.0
 1 file changed, 2 insertions(+), 2 deletions(-)

# Or bump a defined group of products as defined in workspace.cfg
# wst bump mzheng
#
# Or to a specific version (why not just vi? This validates the version for you and␣
↪pulls in the changelog)
# wst bump requests==2.5.1
```

Now you are ready to try out the other commands yourself:

```
usage: wst [-h] [-v] [--debug] <sub-command> ...

optional arguments:

  -h, --help            show this help message and exit
  -v, --version         show program's version number and exit
  --debug               Turn on debug mode

sub-commands:
  {bump,checkout,co,clean,commit,ci,diff,di,log,publish,push,setup,status,st,test,
↪update,up}
                        List of sub-commands
    bump                Bump dependency versions in requirements.txt,
                        pinned.txt, or any specified file.
    checkout (co)       Checkout products (repo urls) or revert files.
    clean               Clean workspace by removing build, dist, and .pyc
                        files
    commit (ci)         Commit all changes locally, including new files.
    diff (di)           Show diff on current product or all products in
                        workspace
    log                 Show commit logs
    merge               Merge changes from branch to current branch
    publish             Bumps version in setup.py (defaults to patch), writes
                        out changelog, builds a source distribution, and
                        uploads with twine.
    push                Push changes for branch
    setup               Optional (refer to setup --help). Setup workspace
                        environment. Run from primary workspace directory.
    status (st)         Show status on current product or all products in
                        workspace
    test                Run tests and manage test environments for product.
    update (up)         Update current product or all products in workspace
```

CHAPTER 4

Links & Contact Info

Documentation: http://workspace-tools.readthedocs.org

PyPI Package: https://pypi.python.org/pypi/workspace-tools
GitHub Source: https://github.com/maxzheng/workspace-tools
Report Issues/Bugs: https://github.com/maxzheng/workspace-tools/issues

Follow: https://twitter.com/MaxZhengX
Connect: https://www.linkedin.com/in/maxzheng
Contact: maxzheng.os @t gmail.com

CHAPTER 5

Command Reference

5.1 SCM Support

git is the only supported SCM. If you have a svn repo, check it out with git-svn.

5.2 Setting Up Your Workspace

It is optional to run ‘wst setup’, however it is recommended as you can type less to do what you want.

After running setup, bash functions and aliases are created for all wst commands. I.e. It creates “ci” alias for “wst commit” if the -a option is used (–commands-with-aliases). And for some commands (such as push), it also adds auto complete for branch name.

To setup, simply go to your workspace directory, and then run ‘wst setup’ with the apporpriate options (-a is recommended):

```
wst setup -a
```

For the sake of readability for the first time user, the remaining reference will use the full wst command instead of aliases from setup. But reference to the short version may be provided as a comment.

5.3 Checkout a Repository

To checkout a product, there are two ways:

1. Use an URL:

   ```
   wst checkout https://github.com/maxzheng/workspace-tools.git

   # Or after "wst setup -a": co https://github.com/maxzheng/workspace-tools.git
   ```

2. Use a product group (see Configuration for more info):

```
wst checkout mzheng-repos
```

5.4 Customize Commands

As simple as two steps:

1. Create your own controller by copying workspace/controller.py:main and add entrypoint to setup.py
2. Add your own commands or change existing in controller. See *workspace.commands* package for examples.

TBD for better docs.

5.5 Want More Docs?

Most commands have help info, so run the command with '-h'.

Bump is based on bumper, so read its doc to learn more.

TBD for more docs as I am not sure if anyone actually read this since there probably aren't that many users of workspace-tools. So If you would like to read the rest of the docs, let me know (maxzheng.os @t gmail.com), and I will take the time to finish this.

CHAPTER 6

API Documentation

6.1 Main Entry Point

class `workspace.controller.`**`Commander`**

Tools to simplify workspace / scm management when working with multiple repositories.

It helps you do more with less work by seamlessly integrating all workspace tooling into one where you can simply run one command instead of many native commands individually to do common tasks.

- **Optionally setup workspace environment/shortcuts, run "wst setup -h" for options.**
  - "wst setup -a" is recommended. :)
- To checkout a product, run: wst checkout <git repository url> [<url2> . . .]
- All commands are named appropriately for what they do, but see its –help for additional info.
- For more info, read the docs at http://workspace-tools.readthedocs.org

classmethod command(*name*)

Get command class for name

classmethod commands()

Map of command name to command classes. Override commands to replace any command name with another class to customize the command.

run(*name=None*, ***kwargs*)

Run the command by name with given args.

Parameters

- **name** (`str`) – Name of command to run. If not given, this calls self._run()
- **kwargs** – Args to pass to the command constructor

setup_parsers()

Sets up parsers for all commands

6.2 Commands

class `workspace.commands.bump.`**`Bump`**(**args*, ***kwargs*)

Bump dependency versions in requirements.txt, pinned.txt, or any specified file.

Parameters

- **`names`** (*`str`*) – Only bump dependencies that match the name. Name can be a product group name defined in workspace.cfg. To bump to a specific version instead of latest, append version to name (i.e. requests==1.2.3 or 'requests>=1.2.3'). When > or < is used, be sure to quote.
- **`test`** (*`int`*) – Run tests.
- **`push`** (*`bool`*) – Push the change. Use with –test to test before pushing.
- **`add`** (*`bool`*) – Add the *names* to the requirements file if they don't exist.
- **`msg`** (*`str`*) – Summary commit message
- **`file`** (*`str/list`*) – Requirement file to bump. Defaults to requirements.txt or pinned.txt that are set by bump.requirement_files in workspace.cfg.
- **`bumper_models`** (*`dict`*) – List of classes that implements `bumper.cars.AbstractBumper` Defaults to `bumper.cars.RequirementsBumper`
- **`force`** (*`bool`*) – Force a bump even when certain bump requirements are not met.
- **`dry_run`** (*`bool`*) – Perform a dry run by printing out the changes only without making changes.
- **`kwargs`** (*`dict`*) – Additional args from argparse

`run`()

Returns Tuple with 3 elements: A map of file to bump message, commit message, and list of *`Bump`*

class `workspace.commands.checkout.`**`Checkout`**(***kwargs*)

Checkout products (repo urls) or branch, or revert files.

Parameters **`target`** (*`list`*) – List of products (git repository URLs) to checkout. When inside a git repo, checkout the branch or revert changes for file(s).

class `workspace.commands.clean.`**`Clean`**(***kwargs*)

Clean workspace by removing build, dist, and .pyc files

class `workspace.commands.commit.`**`Commit`**(***kwargs*)

Commit all changes locally, including new files.

Parameters

- **`msg`** (*`str`*) – Commit message. The first few words are used to create the branch name if branch isn't provided. That behavior can be configured with [commit] auto_branch_from_commit_words
- **`branch`** (*`str`*) – Use specified branch for commit instead of auto-computing the branch from commit msg.
- **`amend`** (*`bool`*) – Amend last commit with any new changes made
- **`test`** (*`bool`*) – Run tests. Repeat twice (-tt) to test dependents too.
- **`push`** (*`bool | int`*) – Push the current branch after commit. Repeat twice (-pp) to push to all remotes.

- **discard** (*int*) – Discard last commit, or branch (child only) if there are no more commits. Use multiple times to discard multiple commits. Other options are ignored. Any local changes may be discarded (hard reset)
- **move** (*str*) – Move last commit to branch. Other options are ignored.
- **test_command** (*callable*) – Alternative test command to run
- **skip_auto_branch** (*bool*) – Skip automatic branch creation from commit msg
- **files** (*list*) – List of files to add instead of all files.

class workspace.commands.diff.**Diff**(***kwargs*)

Show diff on current product or all products in workspace

Parameters

- **context** (*str*) – Show diff for context (i.e. branch or file)
- **parent** (*bool*) – Diff against the parent branch. If there is not parent, defaults to master.
- **name_only** (*bool*) – List file names only. Git only.

class workspace.commands.log.**Log**(***kwargs*)

Show commit logs.

Extra arguments are passed to the underlying SCM's log command.

Parameters

- **diff** (*bool*) – Generate patch / show diff
- **show** (*str*) – Show specific revision. This implies –diff and limit of 1
- **limit** (*int*) – Limit number of log entries
- **extra_args** (*list*) – Extra args to pass to the underlying SCM's log command

class workspace.commands.publish.**Publish**(***kwargs*)

Bumps version in setup.py (defaults to patch), writes out changelog, builds a source distribution, and uploads with twine.

Parameters

- **repo** (*str*) – Repository to publish to
- **minor** (*bool*) – Perform a minor publish by bumping the minor version
- **major** (*bool*) – Perform a major publish by bumping the major version

bump_version()

Bump the version (defaults to patch) in setup.py

update_changelog(*new_version*, *changes*, *skip_title_change=False*)

Parameters

- **new_version** (*str*) – New version
- **changes** (*list[str]*) – List of changes
- **skip_title_change** (*bool*) – Skip title change

Returns Path to changelog file

class workspace.commands.push.**Push**(***kwargs*)

Push changes for branch

Parameters

- **branch** (*str*) – The branch to push. Defaults to current branch.
- **all_remotes** (*bool*) – Push changes to all remotes
- **merge** (*bool*) – Merge the branch into its parent branch before push
- **force** (*bool*) – Force the push

class `workspace.commands.setup.`**`Setup`**(***kwargs*)

Sets up workspace or product environment.

Parameters

- **product_group** (*str*) – Setup product group by checking them out, developing them, and running any setup scripts and exports as defined by setup.cfg in each product.
- **product** (*bool*) – Initialize product by setting up tox with py27, style, and coverage test environments. Also create setup.py, README.rst, and test directories if they don't exist.
- **commands** (*bool*) – Add convenience bash function for certain commands, such as checkout to run "workspace checkout", or "ws" bash function that goes to your workspace directory when no argument is passed in, otherwise runs wst command.
- **commands_with_aliases** (*bool*) – Same as –commands plus add shortcut aliases, like "co" for checkout. This is for those developers that want to get as much done with the least key strokes - true efficienist! ;)
- **uninstall** (*bool*) – Uninstall all functions/aliases.

additional_commands = None

Dict for additional commands to setup

class `workspace.commands.status.`**`Status`**(***kwargs*)

Show status on current product or all products in workspace

class `workspace.commands.test.`**`Test`**(**args*, ***kwargs*)

Run tests and manage test environments for product.

Extra optional boolean args (such as -s, -v, -vv, etc) are passed to py.test.

Parameters

- **env_or_file** (*list*) – The tox environment to act upon, or a file to pass to py.test (only used if file exists, we don't need to redevelop, and py.test is used as a command for the default environements). Defaults to the envlist in tox.
- **repo** (*str*) – Repo path to test instead of current repo
- **show_dependencies** (*bool*) – Show where product dependencies are installed from and their versions.
- **test_dependents** (*bool*) – Run tests in this product and in checked out products that depends on this product. This product must be installed as editable in its dependents for the results to be useful. Most args are ignored when this is used.
- **redevelop** (*bool*) – Redevelop the test environment by installing on top of existing one. This is implied if test environment does not exist, or whenever requirements.txt or pinned.txt is modified after the environment was last updated. Use -ro to do redevelop only without running tests. Use -rr to remove the test environment first before redevelop (recreate).
- **install_only** (*bool*) – Modifier for redevelop. Perform install only without running test.
- **match_test** (*bool*) – Only run tests with method name that matches pattern

- **return_output** (*bool*) – Return test output instead of printing to stdout
- **num_processes** (*str*) – Number of processes to use when running tests in parallel
- **tox_cmd** (*list*) – Alternative tox command to run. If env is passed in (from env_or_file), '-e env' will be appended as well.
- **tox_ini** (*str*) – Path to tox_ini file.
- **tox_commands** (*dict*) – Map of env to list of commands to override "[testenv:env] commands" setting for env. Only used when not developing.
- **args** (*list*) – Additional args to pass to py.test
- **silent** (*bool*) – Run tox/py.test silently. Only errors are printed and followed by exit.
- **debug** (*bool*) – Turn on debug logging
- **install_editable** (*list*) – List of products or product groups to install in editable mode.
- **extra_args** (*list*) – Extra args from argparse to be passed to py.test

Returns Dict of env to commands ran on success. If return_output is True, return a string output. If test_dependents is True, return a mapping of product name to the mentioned results.

classmethod summarize(*tests*, *include_no_tests=True*)
Summarize the test results

Parameters

- **tests** (*dict | str*) – Map of product name to test result, or the test result of the current prod.
- **include_no_tests** (*bool*) – Include "No tests" results when there are no tests found.

Returns A tuple of (success, list(summaries)) where success is True if all tests pass and summaries is a list of passed/failed summary of each test or just str if 'tests' param is str.

class workspace.commands.update.**Update**(**args*, ***kwargs*)
Update current product or all products in workspace

Parameters products (*list*) – When updating all products, filter by these products or product groups

6.3 Configuration

# Default config. Change these in your personal ~/.config/workspace.cfg

```
##################################################################################
↪#####################
# Define product groups to take action upon (such as wst checkout, develop, or bump)
##################################################################################
↪#####################
[product_groups]
mzheng-repos =
  git@github.com:maxzheng/workspace-tools.git
  git@github.com:maxzheng/clicast.git
  git@github.com:maxzheng/localconfig.git
  git@github.com:maxzheng/remoteconfig.git
mzheng = workspace-tools clicast localconfig remoteconfig
```

```
#######################################################################################
↪####################
# Settings for bump command
#######################################################################################
↪####################
[bump]

# List of requirement files to check / bump dependencies in
requirement_files = requirements.txt pinned.txt


#######################################################################################
↪####################
# Settings for checkout command
#######################################################################################
↪####################
[checkout]

# API used to find git repo for single word checkout (i.e. wst checkout workspace-
↪tools)
# It should accept a ?q=singleWord param
search_api_url = https://api.github.com/search/repositories

# URL to use when checking out a user repo reference (i.e. wst checkout maxzheng/
↪workspace-tools)
user_repo_url = git@github.com:%s.git

# User that is mapped to the origin remote. When set, checking out a repo that does␣
↪not belong to the user
# will use upstream remote. e.g. maxzheng
origin_user =

#######################################################################################
↪####################
# Settings for clean command
#######################################################################################
↪####################
[clean]

# Remove products that have not been modified since given days ago
remove_products_older_than_days =

# Remove all products except for these ones (product or group)
remove_all_products_except =

#######################################################################################
↪####################
# Settings for commit command
#######################################################################################
↪####################
[commit]

# Automatically create branch based on the first number of commit words. Set to 0 to␣
↪turn off.
auto_branch_from_commit_words = 2

# When auto branching from commit words, this is the indicator that will be used to
# separate the commit branch from the parent branch, like commit_branch@parent_branch.
commit_branch_indicator = @


#######################################################################################
↪#####################
# Settings for merge command
#######################################################################################
↪#####################
[merge]

# Branches to merge separated by space (i.e. 3.2.x 3.3.x master)
branches =
```

workspace.config.**product_groups**()
    Returns a dict with product group name mapped to products

6.4 SCM Utilities

exception workspace.scm.**SCMError**
    SCM command failed

workspace.scm.**all_branches**(*repo=None*, *remotes=False*, *verbose=False*)
    Returns all branches. The first element is the current branch.

workspace.scm.**all_remotes**(*repo=None*)
    Return all remotes with default remote as the 1st

workspace.scm.**checkout_branch**(*branch*, *repo_path=None*)
    Checks out the branch in the given or current repo. Raises on error.

Parameters

- **branch** (`str`) – Branch to checkout. It can be a branch name or remote/branch combo. if remote is provided, it will always set upstream to `upstream_remote()` regardless of the downstream remote, as we always want to track downstream changes to the upstream remote.
- **repo_path** (`str`) – Path to repo to run checkout in. Defaults to current.

workspace.scm.**checkout_files**(*files*, *repo_path=None*)
    Checks out the given list of files. Raises on error.

workspace.scm.**checkout_product**(*product_url*, *checkout_path*)
    Checks out the product from url. Raises on error

workspace.scm.**commit_changes**(*msg*)
    Commits any modified or new files with given message. Raises on error

workspace.scm.**create_branch**(*branch*, *from_branch=None*)
    Creates a branch from the current branch. Raises on error

workspace.scm.**default_remote**(*repo=None*, *remotes=None*)
    Default remote to take action against, such as push

workspace.scm.**extract_commit_msgs**(*output*, *is_git=True*)
    Returns a list of commit msgs from the given output.

`workspace.scm.`**`is_project`**`(`*path=None*`)`
Check if we are inside of a project.

`workspace.scm.`**`is_repo`**`(`*path=None*`)`
Check if we are inside of a git repo.

`workspace.scm.`**`parent_branch`**`(`*branch*`)`
Returns the parent branch if available, otherwise None

`workspace.scm.`**`product_repos`**`()`
Product repos for the current workspace.

`workspace.scm.`**`project_path`**`(`*path=None*`)`
Return the project path with tox.ini by looking at the current dir and its parent dirs.

`workspace.scm.`**`remove_branch`**`(`*branch*, *raises=False*, *remote=False*, *force=False*`)`
Removes branch

`workspace.scm.`**`repo_path`**`(`*path=None*`)`
Return the git repo path with .git by looking at current dir and its parent dirs.

`workspace.scm.`**`repo_url`**`(`*path=None*, *name='origin'*, *action='push'*`)`

Parameters

- **`path`** (*`str`*) – Local repo path
- **`source`** (*`str`*) – Remote name
- **`action`** (*`str`*) – Action for the corresponding URL

Returns Remote url for repo or None if not found

`workspace.scm.`**`repos`**`(`*dir=None*`)`
Returns a list of repos either for the given directory or current directory or in sub-directories.

`workspace.scm.`**`update_repo`**`(`*path=None*, *quiet=False*`)`
Updates given or current repo to HEAD

`workspace.scm.`**`upstream_remote`**`(`*repo=None*, *remotes=None*`)`
Upstream remote to track against

`workspace.scm.`**`workspace_path`**`()`
Guess the workspace path based on if we are in a repo or not.

6.5 General Utiltities

`workspace.utils.`**`background_processes`**`()`
List of background processes from *run_in_background*

`workspace.utils.`**`log_exception`**`(`*title=None*, *exit=False*, *call=None*, *stack=False*`)`
Context generator that logs exceptions as error

Parameters

- **`title`** (*`str`*) – Title to log before the exception
- **`exit`** (*`bool`*) – Do sys.exit when exception occurs
- **`call`** (*`callable`*) – Call to make before exit
- **`stack`** (*`bool`*) – Log stacktrace

`workspace.utils.`**`parallel_call`**(*call*, *args*, *callback=None*, *workers=10*, *show_progress=None*, *progress_title='Progress'*)

Call a callable in parallel for each arg

Parameters

- **`call`** (*`callable`*) – Callable to call
- **`args`** (*`list (iterable|non-iterable)`*) – List of args to call. One call per args.
- **`callback`** (*`callable`*) – Callable to call for each result.
- **`workers`** (*`int`*) – Number of workers to use.
- **`bool/str/callable`** – Show progress. If callable, it should accept two lists: completed args and all args and return progress string.

Return dict Map of args to their results on completion

`workspace.utils.`**`parent_path_with`**(*check*, *path=None*)

Find parent that satisfies check with content.

Parameters

- **`check`** (*`str`*) – Callable that accepts current path returns True if path should be returned
- **`path`** (*`str`*) – Initial path to look from. Defaults to current working directory.

Returns Parent path that contains the directory

Return type str on success or False on failure

`workspace.utils.`**`parent_path_with_dir`**(*directory*, *path=None*)

Find parent that contains the given directory.

Parameters

- **`directory`** (*`str`*) – Directory to look for
- **`path`** (*`str`*) – Initial path to look from. Defaults to current working directory.

Returns Parent path that contains the directory

Return type str on success or False on failure

`workspace.utils.`**`parent_path_with_file`**(*name*, *path=None*)

Find parent that contains the given directory.

Parameters

- **`name`** (*`str`*) – File name to look for
- **`path`** (*`str`*) – Initial path to look from. Defaults to current working directory.

Returns Parent path that contains the file name

Return type str on success or False on failure

`workspace.utils.`**`prompt_with_editor`**(*instruction*)

Prompt user with instruction in $EDITOR and return the response

`workspace.utils.`**`run`**(*cmd*, *cwd=None*, *silent=None*, *return_output=False*, *raises=True*, ***subprocess_args*)

Runs a CLI command.

Parameters

- **`cmd`** (*`list/str`*) – Command with args to run.

- **cwd** (*str*) – Change directory to cwd before running
- **silent** (*bool/int*) – Suppress stdout/stderr. If True, completely silent. If 2, print cmd output on error.
- **return_output** (*bool*) – Return the command output. Defaults silent=True. Set silent=False to see output. If True, always return output. If set to 2, return a tuple of (output, success) where output is the output of the command and success is exit code 0. When used, it is guaranteed to always return output / other options are ignored (like raises).
- **raises** (*bool*) – Raise an exception if command exits with an error code.
- **subprocess_args** (*dict*) – Additional args to pass to subprocess

Returns Output or boolean of success depending on option selected

Raises **RunError** – if the command exits with an error code and raises=True

`workspace.utils.`**`run_in_background`**(*title*, *repo=None*, *info_suffix='[To check*, *run: {prog} wait]'*, *log_file=None*)

Run any code after this point in the background. This call is idempotent as subsequent calls won't do anything except change the title.

Parameters

- **title** (*str*) – Title to set the running process. This should be informative to the user on what is being run in the background and will happen.
- **repo** (*str*) – Name of repo the task is being acted on. Defaults to os.getcwd()
- **info_suffix** (*str*) – Informational suffix to append when showing the title before forking. {prog} will be replaced by the running program's name.
- **log_file** (*str*) – Full path to log file to save output/error. Saves to temp dir by default. Set to "/dev/null" to discard output.

`workspace.utils.`**`shortest_id`**(*name*, *names*)

Return shortest name that isn't a duplicate in names

`workspace.utils.`**`show_status`**(*message*)

Parameters **message** (*str*) – Status message to show. If not, then status bar will be cleared.

`workspace.utils.`**`silent_run`**(**args*, ***kwargs*)

Same as run with slient=True

CHAPTER 7

Change Log

7.1 Version 3.1.0

- Use regex to better match update error
- Checkout using upstream remote and add origin remote for user when checkout.origin_user is set
- Indicate tracking remote for branch status
- Always track upstream branch and pull from all remotes
- Better support to checkout remote/branch combo
- Use tox.envdir instead of tox.workdir to check if a product is in editable mode or not
- Remove test venv foo

7.2 Version 3.0.28

- Fix envvar expansion
- Use ~/.virtualenvs as the envdir for tox
- Support venv name for activate

7.2.1 Version 3.0.27

- Support activate for ~/.virtualenvs

7.2.2 Version 3.0.26

- Set min code coverage to 80

- Bump min Python to 3.6

7.2.3 Version 3.0.25

- Ignore .eggs in flake8

7.2.4 Version 3.0.24

- Remove commit checking as we only merge when there are stuff to be merged

7.2.5 Version 3.0.23

- Skip style check when pushing a merge

7.2.6 Version 3.0.22

- Add quiet option to merge
- Update source branch before merging

7.2.7 Version 3.0.21

- Add –allow-commits option for merge

7.2.8 Version 3.0.20

- Add strategy option to merge

7.2.9 Version 3.0.19

- Show commits that will be merged
- Include ls for tv
- Show error when updating without remote checking and do –ff-only for update

7.2.10 Version 3.0.18

- Set tracking to upstream remote
- Require origin/upstream remotes when there are more than 1 remote
- Show remotes in status
- Show only child branches at summary view
- Show when there is just 1 child branch
- No need to echo deleted branch as git already does that
- Fix bug to display all branches when there is only 1 repo

- Show status for child branches only when listing all repos

7.2.11 Version 3.0.17

- Add skip update flag for merge

7.2.12 Version 3.0.16

- Add dry run option to merge
- Support checking out remote branches

7.2.13 Version 3.0.15

- Skip style check during publish

7.2.14 Version 3.0.14

- Limit publish to commit setup.py/changelog files only

7.2.15 Version 3.0.13

- Fix repo title
- Support multiple repositories in publish
- Use multiple push flags to indicate pushing to all remotes during commit
- Use git checkout path for git.Repo so it works from child dirs
- Set default max-line-length to 140
- Update keywords

7.2.16 Version 3.0.12

- Merge branch 'master' of github.com:maxzheng/workspace-tools
- Use proper email format for author

7.2.17 Version 3.0.11

- Check code style before pushing
- Change setup.py template to require Python 3.5+
- Remove requirements.txt from tox.ini
- Create example test in "tests" folder
- Move tests to "tests" folder

7.2.18 Version 3.0.10

- Skip printing about merging to downstream branches

7.2.19 Version 3.0.9

- Show parent branch when merging during push
- Show rebase message only if verbose

7.2.20 Version 3.0.8

- Show branch and remotes being pulled from

7.2.21 Version 3.0.7

- Check for any merge changes before pushing

7.2.22 Version 3.0.6

- Change option name to merge –downstreams and add more validation

7.2.23 Version 3.0.5

- Switch to use click.echo instead of log.info
- Revert "Split config lists early"

  This reverts commit 1b2867dc2c5c33ecdc2c5c6e70e8a8f874e6ced1.
- Fix indent for dependency script

7.2.24 Version 3.0.4

- Split config lists early
- Add more info on merge.branch config

7.2.25 Version 3.0.3

- Add merge doc

7.2.26 Version 3.0.2

- Set upstream or remote but not both when pushing
- Add merge command with option to merge to a list of user configured branches
- Add push –all-remotes option
- Reindent to use 4 spaces

- Some minor changes

7.2.27 Version 3.0.1

- Add follow link

7.2.28 Version 3.0.0

- Fix tests and bugs
- Remove review and wait commands.

  They are not easy to implement and does not provide that much value. Maybe later.
- Only delete child branches
- Many improvements for working with multiple branches
- Use autostash when doing update (git pull)
- Migrate to Python 3.x and add support for multiple projects per repo.

  And remove support for svn, git-svn.
- Add .eggs to .gitignore
- Sync / update

7.3 Version 1.0.11

- Log wait command output and allow them to be viewed with –log option

7.3.1 Version 1.0.10

- Add –install-editable option to “ws test” and remove config.test.editable_products
- Sync changes from downstream
- Remove use of –download-cache option
- Set testpaths to “test”
- Add –name-only option and fix some bugs

7.3.2 Version 1.0.9

- Add –rb to bump to be consistent with other commands and various test fixes
- Do sys.exit(1) if any repo failed to update instead of existing silently.

  Also check if package exists before including it in version display.

7.3.3 Version 1.0.8

- Scope not implemented exception to base Wait class for review/publish event

7.3.4 Version 1.0.7

- Run wait chaining actions in background
- Sort task view by repo/task

7.3.5 Version 1.0.6

- Prompt user for commit msg if not given
- Ensure branch is assigned before use

7.3.6 Version 1.0.5

- Add –push/–bump-in chaining options to wait command

7.3.7 Version 1.0.4

- Support running tasks in background
- Detect if .pypirc has necessary info and prompt as needed. require=localconfig

7.3.8 Version 1.0.3

- Fall back to use build results if there is no test result
- Skip style check if there is no style env
- Suppress stacktrace when getting ^C

7.3.9 Version 1.0.2

- Display chaining options separately in help
- Run style check when running tests for commit
- Quote args to tv alias

7.3.10 Version 1.0.1

- Centralize test result summary / evaluation logic

7.3.11 Version 1.0.0

- Switch to class-based command architecture to simplify downstream customization

7.4 Version 0.8.19

- Check for branches before removing repo when cleaning
- Create config dir if not exists

7.4.1 Version 0.8.18

- Check another directory for setup.cfg

7.4.2 Version 0.8.17

- Add repo_url method to get remote repo url
- Redirect STDERR to STDOUT when running command with silent/return_output option

7.4.3 Version 0.8.16

- Amend commit before running tests as tests might run long

7.4.4 Version 0.8.15

- Exit early if test failed before commit
- Update doc

7.4.5 Version 0.8.14

- Add install-only modifier for redevelop/recreate
- Update activate alias to work in different situations

7.4.6 Version 0.8.13

- Revert removing //build dir during clean
- Add –test option to run tests before committing
- Use auto branch when bumping to support multiple bumps
- Add remove_all_products_except option for clean command
- Ensure dummy commit msg starts with “Empty commit”

7.4.7 Version 0.8.12

- Use pip to list installed dependencies instead of pkg_resources

7.4.8 Version 0.8.11

- Use existing msg field for dummy msg

7.4.9 Version 0.8.10

- Allow dummy commit msg to be changed

7.4.10 Version 0.8.9

- Add filter option for showing installed dependencies

7.4.11 Version 0.8.8

- Use setup.cfg instead of setup.ws

7.4.12 Version 0.8.7

- Remove test code

7.4.13 Version 0.8.6

- Support custom product setup with setup.ws

7.4.14 Version 0.8.5

- Simplify product group bootstrap with setup command

7.4.15 Version 0.8.4

- Show progress for dependent tests

7.4.16 Version 0.8.3

- Run dependent tests in parallel

7.4.17 Version 0.8.2

- When bumping, only add/commit files updated by bump
- Only run transitive tests if current product is in editable_products list

7.4.18 Version 0.8.1

- Update README

7.4.19 Version 0.8.0

- Add skip_editable_install internal arg for test command
- Deprecate [test] scope_transitive_test_products with editable_products
- Deprecate [test] editable_product_dependencies with editable_products that is also used for scoping products that will install editables

7.5 Version 0.7.24

- Fix “-n 0” option for test command

7.5.1 Version 0.7.23

- Fix repo detection in nested repos
- Skip auto branch for commit when already on a branch

7.5.2 Version 0.7.22

- Better checking for clean repo that works for older git

7.5.3 Version 0.7.21

- Perform product update in parallel
- Add remove_products_older_than_days option for clean command
- Add scope_transitive_test_products config option to scope transitive products to test

7.5.4 Version 0.7.20

- Flush streamed test output

7.5.5 Version 0.7.19

- Do not count one/two letter words when creating branch from commit msg

7.5.6 Version 0.7.18

- Append error from subprocess to output

7.5.7 Version 0.7.17

- Stream test output when returning output

7.5.8 Version 0.7.16

- Return bumps made for bump()

7.5.9 Version 0.7.15

- Update usage for commit
- Add –test-dependent option to run tests in dependent products
- Add option to return test output

7.5.10 Version 0.7.14

- Ignore DRAFT: prefix when creating branch from commit msg

7.5.11 Version 0.7.13

- Add links to bumper

7.5.12 Version 0.7.12

- Change auto branch commit words to 2 and add more ignored words
- Change –discard to count to allow deleting of multiple commits
- Add skip auto branch option for commit
- Automatically create a branch from commit msg
- Redevelop if tox.ini has been modified
- Fix tests

7.5.13 Version 0.7.11

- Better composed commit message / revert on failed commit
- Remove extra line between changes when generating changelog

7.5.14 Version 0.7.10

- Ignore "Update changelog" commits when publishing
- Update setup.py template
- Add url and summary info

7.5.15 Version 0.7.1

- Add -D alias for –discard in commit

7.5.16 Version 0.7.0

- Refactor to use bumper-lib

7.6 Version 0.6.10

- Add re constant for user repo reference

7.6.1 Version 0.6.9

- Make -1, -2, etc limit work for svn log
- Pass unknown args for log to underlying SCM / better args

7.6.2 Version 0.6.8

- Allow arbitrary boolean optional args to be passed to py.test from test command

7.6.3 Version 0.6.7

- Support which command in tv alias

7.6.4 Version 0.6.6

- Add -n pass thru option for py.test
- Only install editable dependencies in [tox] envlist environments

7.6.5 Version 0.6.5

- Support checking out from github using product name or user/name format

7.6.6 Version 0.6.4

- Remove checking of setup.py for test as that is affected by version bumps. Add pinned.txt to be checked

7.6.7 Version 0.6.3

- Faster clean for *.pyc files

7.6.8 Version 0.6.2

- Only use first line when showing what changed for svn during bump

7.6.9 Version 0.6.1

- Update checkout usage

7.6.10 Version 0.6.0

- Commit multiple file bumps as a single commit and use –msg as the summary (prepended)
- Improved tv alias

7.7 Version 0.5.11

- Skip editable mode change if there are no dependencies

7.7.1 Version 0.5.10

- Support silent run that outputs on error and use on test command

7.7.2 Version 0.5.9

- Return commands ran per env for test command

7.7.3 Version 0.5.8

- Add tv alias to open files from ag in vim. Add env auto complete for test command
- Add doc link to usage

7.7.4 Version 0.5.7

- Add install_command with -U to ensure latest versions are installed and without {opts} to always install dependencies

7.7.5 Version 0.5.6

- Better exception handling/output for test

7.7.6 Version 0.5.5

- Better support for customizing test command

7.7.7 Version 0.5.4

- Rename dependencies to show_dependencies for test arg and update test usage
- Add example to setup tox and run style/coverage

7.7.8 Version 0.5.3

- Skip install dependencies in editable mode if already in editable mode
- Add test for status
- Add test.editable_product_dependencies option to auto install dependencies in editable mode
- Support multiple environments when showing product dependencies
- Refactor tox ini code into ToxIni class
- Auto-detect requirement files change to re-develop environment

7.7.9 Version 0.5.2

- Activate environment before running py.test
- Use spaces instead of tabs in tox template

7.7.10 Version 0.5.1

- Add tests and support -k / -s options from py.test in test command

7.7.11 Version 0.5.0

- Support multiple test environments and use optimized test run
- Update tox template
- Skip creating requirements.txt if setup.py already exists
- Fix import issues with setup –product
- Deprecate/break develop into test and setup command
- Update usage in README
- Remove remote doc config as that was checked in accidentally

7.8 Version 0.4.11

- Skip bump branch check when doing dry run

7.8.1 Version 0.4.7

- Fix bump doc
- Update doc
- Update doc

7.8.2 Version 0.4.6

- Add doc for bump / start but not finish Command Reference
- Add tests for bump and remove use of memozie
- Remove ln whitelist from tox

7.8.3 Version 0.4.5

- Strip version spec from entry scripts in dev env

7.8.4 Version 0.4.4

- Allow downstream package to show its version with -v

7.8.5 Version 0.4.3

- Support custom file processing for bump and do not use squash merge for push

7.8.6 Version 0.4.2

- Add bump bash shortcut

7.8.7 Version 0.4.1

- Fix product name computation for url ends with /trunk
- Update changelog

7.8.8 Version 0.4.0

- Add example on setting up / using product group
- Add bump command to bump dependency versions

7.9 Version 0.3.1

- Skip checking for user config file existence as that is done in RemoteConfig now
- Add -U to pip install

7.9.1 Version 0.3.0

- Refactor to use remoteconfig
- Remove activate soft linking in –init

7.10 Version 0.2.40

- Retain latest major/minor release title in changelog

7.10.1 Version 0.2.39

- Use bullet list for changes in CHANGELOG

7.10.2 Version 0.2.38

- Add changelog to index by listing the latest version only

7.10.3 Version 0.2.37

- Exit early / without changing version when there are no changes when publishing. Better ‘a’ alias to avoid having to do symlink in tox.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index