
dulwich Documentation

Release 0.20.6

Jelmer Vernooij

Oct 23, 2020

Contents

1	Overview	1
1.1	Installation	1
1.2	Getting started	1
1.3	Further documentation	2
1.4	Help	2
1.5	Contributing	2
1.6	Supported versions of Python	2
2	Documentation	3
2.1	Possible areas for improvement	3
2.2	Git Server Protocol	3
2.3	Tutorial	4
2.4	Module reference	14
3	Changelog	15
4	Indices and tables	43

This is the Dulwich project.

It aims to provide an interface to git repos (both local and remote) that doesn't call out to git directly but instead uses pure Python.

Main website: <<https://www.dulwich.io/>>

License: Apache License, version 2 or GNU General Public License, version 2 or later.

The project is named after the part of London that Mr. and Mrs. Git live in in the particular Monty Python sketch.

1.1 Installation

By default, Dulwich' setup.py will attempt to build and install the optional C extensions. The reason for this is that they significantly improve the performance since some low-level operations that are executed often are much slower in CPython.

If you don't want to install the C bindings, specify the `--pure` argument to setup.py:

```
$ python setup.py --pure install
```

or if you are installing from pip:

```
$ pip install dulwich --global-option="--pure"
```

Note that you can also specify `--global-option` in a `requirements.txt` file, e.g. like this:

```
dulwich --global-option=--pure
```

1.2 Getting started

Dulwich comes with both a lower-level API and higher-level plumbing (“porcelain”).

For example, to use the lower level API to access the commit message of the last commit:

```
>>> from dulwich.repo import Repo
>>> r = Repo('.')
>>> r.head()
'57fbe010446356833a6ad1600059d80b1e731e15'
>>> c = r[r.head()]
>>> c
<Commit 015fc1267258458901a94d228e39f0a378370466>
>>> c.message
'Add note about encoding.\n'
```

And to print it using porcelain:

```
>>> from dulwich import porcelain
>>> porcelain.log('.', max_entries=1)
-----
commit: 57fbe010446356833a6ad1600059d80b1e731e15
Author: Jelmer Vernooij <jelmer@jelmer.uk>
Date: Sat Apr 29 2017 23:57:34 +0000

Add note about encoding.
```

1.3 Further documentation

The dulwich documentation can be found in docs/ and built by running `make doc`. It can also be found [on the web](#).

1.4 Help

There is a `#dulwich` IRC channel on the [Freenode](#), and `dulwich-announce` and `dulwich-discuss` mailing lists.

1.5 Contributing

For a full list of contributors, see the git logs or [AUTHORS](#).

If you'd like to contribute to Dulwich, see the [CONTRIBUTING](#) file and [list of open issues](#).

1.6 Supported versions of Python

At the moment, Dulwich supports (and is tested on) CPython 3.5, 3.6, 3.7, 3.8 and Pypy.

The latest release series to support Python 2.x was the 0.19 series. See the 0.19 branch in the Dulwich git repository.

2.1 Possible areas for improvement

Places for improvement, ordered by difficulty / effectiveness:

- `read_zlib()` should have a C equivalent (~ 4% overhead atm)
- `unpack_object()` should have a C equivalent

2.2 Git Server Protocol

2.2.1 Transport

The Git protocol operates over pipes or TCP/IP. When a client connects over TCP/IP, it sends a header that tells the server which program to run and what parameters to use. When invoked over SSH, git will run a program with the parameters as command line arguments.

2.2.2 Protocols

Basics

Git communicates with a server by piping data between a local program and a remote program.

A common way of sending a unit of information is a `pkt_line`. This is a 4 byte size as human encoded hex (i.e. totally underusing the 4 bytes. ...) that tells you the size of the payload, followed by the payload. The size includes the 4 bytes used by the size itself.

0009ABCDn

Git can also multiplex data using the sideband. As well as 4 bytes size, there would be a 1 byte channel number. This is in binary, so 1 will be `\x01`.

Typically Git will piggyback a list of capabilities on the first `pkt_line` it sends. It will also look for capabilities in the first `pkt_line` it receives. Git will degrade as much as possible when encountering a server or client with differing capabilities.

git-upload-pack

`git-upload-pack` is used by `git-ls-remote`, `git-clone`, `git-fetch` and `git-pull`. And i'm sure others. Typically a client will connect a local `git-fetch-pack` to a remote `git-upload-pack`.

Capabilities for this protocol include `multi_ack`, `thin-pack`, `ofs-delta`, `sideband` and `sideband-64k`. A thin pack can reference objects not in the current pack.

The server tells the client what refs it has. The client states which of those SHA1's it would like. It then starts to report which SHA1's it has. The server ACKs these allowing the client to work out when to stop sending SHA1's. This saves a lot of transfer because the client can make decisions like "well if it has this SHA, then it has all its parents so I don't need to care about those". When the client stops sending shas, the server can work out an optimal pack and then send it to the client.

git-receive-pack

`git-receive-pack` is used by `git push`. Typically a client connects a local `git-send-pack` to a remote `git-receive-pack`.

Capabilities include `report-status` and `delete-ref`.

2.3 Tutorial

2.3.1 Introduction

Like Git itself, Dulwich consists of two main layers; the so-called plumbing and the porcelain.

The plumbing is the lower layer and it deals with the Git object database and the nitty gritty internals. The porcelain is roughly what you would expect to be exposed to as a user of the `git` command-like tool.

Dulwich has a fairly complete plumbing implementation, and a more recently added porcelain implementation. The porcelain code lives in `dulwich.porcelain`.

For the large part, this tutorial introduces you to the internal concepts of Git and the main plumbing parts of Dulwich. The last chapter covers the porcelain.

2.3.2 Encoding

You will notice that all lower-level functions in Dulwich take byte strings rather than unicode strings. This is intentional.

Although C `git` recommends the use of UTF-8 for encoding, this is not strictly enforced and C `git` treats filenames as sequences of non-NUL bytes. There are repositories in the wild that use non-UTF-8 encoding for filenames and commit messages.

The library should be able to read *all* existing git repositories, irregardless of what encoding they use. This is the main reason why Dulwich does not convert paths to unicode strings.

A further consideration is that converting back and forth to unicode is an extra performance penalty. E.g. if you are just iterating over file contents, there is no need to consider encoded strings. Users of the library may have specific

assumptions they can make about the encoding - e.g. they could just decide that all their data is latin-1, or the default Python encoding.

Higher level functions, such as the porcelain in `dulwich.porcelain`, will automatically convert unicode strings to UTF-8 bytestrings.

2.3.3 Git File format

For a better understanding of Dulwich, we'll start by explaining most of the Git secrets.

Open the `.git` folder of any Git-managed repository. You'll find folders like `branches`, `hooks`... We're only interested in `objects` here. Open it.

You'll mostly see 2 hex-digits folders. Git identifies content by its SHA-1 digest. The 2 hex-digits plus the 38 hex-digits of files inside these folders form the 40 characters (or 20 bytes) id of Git objects you'll manage in Dulwich.

We'll first study the three main objects:

- The Commit;
- The Tree;
- The Blob.

The Commit

You're used to generate commits using Git. You have set up your name and e-mail, and you know how to see the history using `git log`.

A commit file looks like this:

```
commit <content length><NUL>tree <tree sha>
parent <parent sha>
[parent <parent sha> if several parents from merges]
author <author name> <author e-mail> <timestamp> <timezone>
committer <author name> <author e-mail> <timestamp> <timezone>

<commit message>
```

But where are the changes you committed? The commit contains a reference to a tree.

The Tree

A tree is a collection of file information, the state of a single directory at a given point in time.

A tree file looks like this:

```
tree <content length><NUL><file mode> <filename><NUL><item sha>...
```

And repeats for every file in the tree.

Note that the SHA-1 digest is in binary form here.

The file mode is like the octal argument you could give to the `chmod` command. Except it is in extended form to tell regular files from directories and other types.

We now know how our files are referenced but we haven't found their actual content yet. That's where the reference to a blob comes in.

The Blob

A blob is simply the content of files you are versioning.

A blob file looks like this:

```
blob <content length><NUL><content>
```

If you change a single line, another blob will be generated by Git at commit time. This is how Git can fastly checkout any version in time.

On the opposite, several identical files with different filenames generate only one blob. That's mostly how renames are so cheap and efficient in Git.

Dulwich Objects

Dulwich implements these three objects with an API to easily access the information you need, while abstracting some more secrets Git is using to accelerate operations and reduce space.

More About Git formats

These three objects make up most of the contents of a Git repository and are used for the history. They can either appear as simple files on disk (one file per object) or in a `pack` file, which is a container for a number of these objects.

There is also an index of the current state of the working copy in the repository as well as files to track the existing branches and tags.

For a more detailed explanation of object formats and SHA-1 digests, see: <http://www-cs-students.stanford.edu/~blynn/gitmagic/ch08.html>

Just note that recent versions of Git compress object files using `zlib`.

2.3.4 The repository

After this introduction, let's start directly with code:

```
>>> from dulwich.repo import Repo
```

The access to a repository is through the `Repo` object. You can open an existing repository or you can create a new one. There are two types of Git repositories:

Regular Repositories – They are the ones you create using `git init` and you daily use. They contain a `.git` folder.

Bare Repositories – There is no `.git` folder. The top-level folder contains itself the `“branches”`, `“hooks”`... folders. These are used for published repositories (mirrors). They do not have a working tree.

Creating a repository

Let's create a folder and turn it into a repository, like `git init` would:

```
>>> from os import mkdir
>>> import sys
>>> mkdir("myrepo")
>>> repo = Repo.init("myrepo")
>>> repo
<Repo at 'myrepo'>
```

You can already look at the structure of the “myrepo/.git” folder, though it is mostly empty for now.

Opening an existing repository

To reopen an existing repository, simply pass its path to the constructor of `Repo`:

```
>>> repo = Repo("myrepo")
>>> repo
<Repo at 'myrepo'>
```

Opening the index

The index is used as a staging area. Once you do a commit, the files tracked in the index will be recorded as the contents of the new commit. As mentioned earlier, only non-bare repositories have a working tree, so only non-bare repositories will have an index, too. To open the index, simply call:

```
>>> index = repo.open_index()
>>> print(index.path)
myrepo/.git/index
```

Since the repository was just created, the index will be empty:

```
>>> list(index)
[]
```

Staging new files

The repository allows “staging” files. Only files can be staged - directories aren’t tracked explicitly by git. Let’s create a simple text file and stage it:

```
>>> f = open('myrepo/foo', 'wb')
>>> _ = f.write(b"monty")
>>> f.close()

>>> repo.stage([b"foo"])
```

It will now show up in the index:

```
>>> print(", ".join([f.decode(sys.getfilesystemencoding()) for f in repo.open_
↳ index()]))
foo
```

Creating new commits

Now that we have staged a change, we can commit it. The easiest way to do this is by using `Repo.do_commit`. It is also possible to manipulate the lower-level objects involved in this, but we'll leave that for a separate chapter of the tutorial.

To create a simple commit on the current branch, it is only necessary to specify the message. The committer and author will be retrieved from the repository configuration or global configuration if they are not specified:

```
>>> commit_id = repo.do_commit(
...     b"The first commit", committer=b"Jelmer Vernooij <jelmer@samba.org>")
```

`do_commit` returns the SHA1 of the commit. Since the commit was to the default branch, the repository's head will now be set to that commit:

```
>>> repo.head() == commit_id
True
```

2.3.5 The object store

The objects are stored in the `object` store of the repository.

```
>>> from dulwich.repo import Repo
>>> repo = Repo.init("myrepo", mkdir=True)
```

Initial commit

When you use Git, you generally add or modify content. As our repository is empty for now, we'll start by adding a new file:

```
>>> from dulwich.objects import Blob
>>> blob = Blob.from_string(b"My file content\n")
>>> print(blob.id.decode('ascii'))
c55063a4d5d37aa1af2b2dad3a70aa34dae54dc6
```

Of course you could create a blob from an existing file using `from_file` instead.

As said in the introduction, file content is separated from file name. Let's give this content a name:

```
>>> from dulwich.objects import Tree
>>> tree = Tree()
>>> tree.add(b"spam", 0o100644, blob.id)
```

Note that "0o100644" is the octal form for a regular file with common permissions. You can hardcode them or you can use the `stat` module.

The tree state of our repository still needs to be placed in time. That's the job of the commit:

```
>>> from dulwich.objects import Commit, parse_timezone
>>> from time import time
>>> commit = Commit()
>>> commit.tree = tree.id
>>> author = b"Your Name <your.email@example.com>"
>>> commit.author = commit.committer = author
>>> commit.commit_time = commit.author_time = int(time())
```

(continues on next page)

(continued from previous page)

```
>>> tz = parse_timezone(b'-0200')[0]
>>> commit.commit_timezone = commit.author_timezone = tz
>>> commit.encoding = b"UTF-8"
>>> commit.message = b"Initial commit"
```

Note that the initial commit has no parents.

At this point, the repository is still empty because all operations happen in memory. Let's "commit" it.

```
>>> object_store = repo.object_store
>>> object_store.add_object(blob)
```

Now the ".git/objects" folder contains a first SHA-1 file. Let's continue saving the changes:

```
>>> object_store.add_object(tree)
>>> object_store.add_object(commit)
```

Now the physical repository contains three objects but still has no branch. Let's create the master branch like Git would:

```
>>> repo.refs[b'refs/heads/master'] = commit.id
```

The master branch now has a commit where to start. When we commit to master, we are also moving HEAD, which is Git's currently checked out branch:

```
>>> head = repo.refs[b'HEAD']
>>> head == commit.id
True
>>> head == repo.refs[b'refs/heads/master']
True
```

How did that work? As it turns out, HEAD is a special kind of ref called a symbolic ref, and it points at master. Most functions on the refs container work transparently with symbolic refs, but we can also take a peek inside HEAD:

```
>>> import sys
>>> print(repo.refs.read_ref(b'HEAD').decode(sys.getfilesystemencoding()))
ref: refs/heads/master
```

Normally, you won't need to use `read_ref`. If you want to change what ref HEAD points to, in order to check out another branch, just use `set_symbolic_ref`.

Now our repository is officially tracking a branch named "master" referring to a single commit.

Playing again with Git

At this point you can come back to the shell, go into the "myrepo" folder and type `git status` to let Git confirm that this is a regular repository on branch "master".

Git will tell you that the file "spam" is deleted, which is normal because Git is comparing the repository state with the current working copy. And we have absolutely no working copy using Dulwich because we don't need it at all!

You can checkout the last state using `git checkout -f`. The force flag will prevent Git from complaining that there are uncommitted changes in the working copy.

The file `spam` appears and with no surprise contains the same bytes as the blob:

```
$ cat spam
My file content
```

Changing a File and Committing it

Now we have a first commit, the next one will show a difference.

As seen in the introduction, it's about making a path in a tree point to a new blob. The old blob will remain to compute the diff. The tree is altered and the new commit's task is to point to this new version.

Let's first build the blob:

```
>>> from dulwich.objects import Blob
>>> spam = Blob.from_string(b"My new file content\n")
>>> print(spam.id.decode('ascii'))
16ee2682887a962f854ebd25a61db16ef4efe49f
```

An alternative is to alter the previously constructed blob object:

```
>>> blob.data = b"My new file content\n"
>>> print(blob.id.decode('ascii'))
16ee2682887a962f854ebd25a61db16ef4efe49f
```

In any case, update the blob id known as "spam". You also have the opportunity of changing its mode:

```
>>> tree[b"spam"] = (0o100644, spam.id)
```

Now let's record the change:

```
>>> from dulwich.objects import Commit
>>> from time import time
>>> c2 = Commit()
>>> c2.tree = tree.id
>>> c2.parents = [commit.id]
>>> c2.author = c2.committer = b"John Doe <john@example.com>"
>>> c2.commit_time = c2.author_time = int(time())
>>> c2.commit_timezone = c2.author_timezone = 0
>>> c2.encoding = b"UTF-8"
>>> c2.message = b'Changing "spam"'
```

In this new commit we record the changed tree id, and most important, the previous commit as the parent. Parents are actually a list because a commit may happen to have several parents after merging branches.

Let's put the objects in the object store:

```
>>> repo.object_store.add_object(spam)
>>> repo.object_store.add_object(tree)
>>> repo.object_store.add_object(c2)
```

You can already ask git to introspect this commit using `git show` and the value of `c2.id` as an argument. You'll see the difference will the previous blob recorded as "spam".

The diff between the previous head and the new one can be printed using `write_tree_diff`:

```
>>> from dulwich.patch import write_tree_diff
>>> from io import BytesIO
>>> out = BytesIO()
```

(continues on next page)

(continued from previous page)

```
>>> write_tree_diff(out, repo.object_store, commit.tree, tree.id)
>>> import sys; _ = sys.stdout.write(out.getvalue().decode('ascii'))
diff --git a/spam b/spam
index c55063a..16ee268 100644
--- a/spam
+++ b/spam
@@ -1,1 @@
-My file content
+My new file content
```

You won't see it using git log because the head is still the previous commit. It's easy to remedy:

```
>>> repo.refs[b'refs/heads/master'] = c2.id
```

Now all git tools will work as expected. Most of the tests in this file require a Dulwich server, so let's start one:

```
>>> from dulwich.repo import Repo
>>> from dulwich.server import DictBackend, TCPGitServer
>>> import threading
>>> repo = Repo.init("remote", mkdir=True)
>>> cid = repo.do_commit(b"message", committer=b"Jelmer <jelmer@samba.org>")
>>> backend = DictBackend({b'/' : repo})
>>> dul_server = TCPGitServer(backend, b'localhost', 0)
>>> threading.Thread(target=dul_server.serve).start()
>>> server_address, server_port=dul_server.socket.getsockname()
```

2.3.6 Remote repositories

The interface for remote Git repositories is different from that for local repositories.

The Git smart server protocol provides three basic operations:

- upload-pack - provides a pack with objects requested by the client
- receive-pack - imports a pack with objects provided by the client
- upload-archive - provides a tarball with the contents of a specific revision

The smart server protocol can be accessed over either plain TCP (git://), SSH (git+ssh://) or tunneled over HTTP (http://).

Dulwich provides support for accessing remote repositories in `dulwich.client`. To create a new client, you can construct one manually:

```
>>> from dulwich.client import TCPGitClient
>>> client = TCPGitClient(server_address, server_port)
```

Retrieving raw pack files

The client object can then be used to retrieve a pack. The `fetch_pack` method takes a `determine_wants` callback argument, which allows the client to determine which objects it wants to end up with:

```
>>> def determine_wants(refs):
...     # retrieve all objects
...     return refs.values()
```

Another required object is a “graph walker”, which is used to determine which objects that the client already has should not be sent again by the server. Here in the tutorial we’ll just use a dummy graph walker which claims that the client doesn’t have any objects:

```
>>> class DummyGraphWalker(object):
...     def ack(self, sha): pass
...     def next(self): pass
...     def __next__(self): pass
```

With the `determine_wants` function in place, we can now fetch a pack, which we will write to a `BytesIO` object:

```
>>> from io import BytesIO
>>> f = BytesIO()
>>> result = client.fetch_pack(b"/", determine_wants,
...     DummyGraphWalker(), pack_data=f.write)
```

`f` will now contain a full pack file:

```
>>> print(f.getvalue()[:4].decode('ascii'))
PACK
```

Fetching objects into a local repository

It is also possible to fetch from a remote repository into a local repository, in which case Dulwich takes care of providing the right graph walker, and importing the received pack file into the local repository:

```
>>> from dulwich.repo import Repo
>>> local = Repo.init("local", mkdir=True)
>>> remote_refs = client.fetch(b"/", local)
```

Let’s shut down the server now that all tests have been run:

```
>>> dul_server.shutdown()
```

2.3.7 Tagging

This tutorial will demonstrate how to add a tag to a commit via dulwich.

First let’s initialize the repository:

```
>>> from dulwich.repo import Repo
>>> _repo = Repo("myrepo", mkdir=True)
```

Next we build the commit object and add it to the object store:

```
>>> from dulwich.objects import Blob, Tree, Commit, parse_timezone
>>> permissions = 0100644
>>> author = "John Smith"
>>> blob = Blob.from_string("empty")
>>> tree = Tree()
>>> tree.add(tag, permissions, blob.id)
>>> commit = Commit()
>>> commit.tree = tree.id
>>> commit.author = commit.committer = author
>>> commit.commit_time = commit.author_time = int(time())
```

(continues on next page)

(continued from previous page)

```
>>> tz = parse_timezone('-0200')[0]
>>> commit.commit_timezone = commit.author_timezone = tz
>>> commit.encoding = "UTF-8"
>>> commit.message = 'Tagging repo: ' + message
```

Add objects to the repo store instance:

```
>>> object_store = _repo.object_store
>>> object_store.add_object(blob)
>>> object_store.add_object(tree)
>>> object_store.add_object(commit)
>>> master_branch = 'master'
>>> _repo.refs['refs/heads/' + master_branch] = commit.id
```

Finally, add the tag top the repo:

```
>>> _repo['refs/tags/' + commit] = commit.id
```

Alternatively, we can use the tag object if we'd like to annotate the tag:

```
>>> from dulwich.objects import Blob, Tree, Commit, parse_timezone, Tag
>>> tag_message = "Tag Annotation"
>>> tag = Tag()
>>> tag.tagger = author
>>> tag.message = message
>>> tag.name = "v0.1"
>>> tag.object = (Commit, commit.id)
>>> tag.tag_time = commit.author_time
>>> tag.tag_timezone = tz
>>> object_store.add_object(tag)
>>> _repo['refs/tags/' + tag] = tag.id
```

2.3.8 Porcelain

The porcelain is the higher level interface, built on top of the lower level implementation covered in previous chapters of this tutorial. The `dulwich.porcelain` module in Dulwich is aimed to closely resemble the Git command-line API that you are familiar with.

Basic concepts

The porcelain operations are implemented as top-level functions in the `dulwich.porcelain` module. Most arguments can either be strings or more complex Dulwich objects; e.g. a repository argument will either take a string with a path to the repository or an instance of a `Repo` object.

Initializing a new repository

```
>>> from dulwich import porcelain
```

```
>>> repo = porcelain.init("myrepo")
```

Clone a repository

```
>>> porcelain.clone("git://github.com/jelmer/dulwich", "dulwich-clone")
```

Commit changes

```
>>> r = porcelain.init("testrepo")
>>> open("testrepo/testfile", "w").write("data")
>>> porcelain.add(r, "testfile")
>>> porcelain.commit(r, b"A sample commit")
```

Push changes

```
>>> tr = porcelain.init("targetrepo")
>>> r = porcelain.push("testrepo", "targetrepo", "master")
```

2.3.9 Conclusion

This tutorial currently only covers a small (but important) part of Dulwich. It still needs to be extended to cover packs, refs, reflogs and network communication.

Dulwich is abstracting much of the Git plumbing, so there would be more to see.

For now, that's all folks!

This is the API documentation for Dulwich.

2.4 Module reference

Indices:

- [modindex](#)
- [search](#)

0.27.7 UNRELEASED

- Check `core.repositoryformatversion`. (Jelmer Vernooij, #803)

0.20.6 2020-08-29

- Add a `RefsContainer.watch` interface. (Jelmer Vernooij, #751)
- Fix pushing of new branches from `porcelain.push`. (Jelmer Vernooij, #788)
- Honor shallows when pushing from a shallow clone. (Jelmer Vernooij, #794)
- Fix `porcelain.path_to_tree_path` for Python 3.5. (Boris Feld, #777)
- Add support for honor proxy environment variables for HTTP. (Aurélien Campéas, #797)

0.20.5 2020-06-22

- Print a clearer exception when `setup.py` is executed on Python < 3.5. (Jelmer Vernooij, #783)
- Send an empty pack to clients if they requested objects, even if they already have those objects. Thanks to Martijn Pieters for the detailed bug report. (Jelmer Vernooij, #781)
- `porcelain.pull`: Don't ask for objects that we already have. (Jelmer Vernooij, #782)
- Add LCA implementation. (Kevin Hendricks)
- Add functionality for finding the merge base. (Kevin Hendricks)
- Check for diverged branches during push. (Jelmer Vernooij, #494)
- Check for fast-forward during pull. (Jelmer Vernooij, #666)
- Return a `SendPackResult` object from `GitClient.send_pack()`. (Jelmer Vernooij)
- `GitClient.send_pack` now sets the `ref_status` attribute on its return value to a dictionary mapping ref names to error messages. Previously, it raised `UpdateRefsError` if any of the refs failed to update. (Jelmer Vernooij, #780)
- Add a `porcelain.Error` object that most errors in `porcelain` derive from. (Jelmer Vernooij)
- Fix argument parsing in `dulwich` command-line app. (Jelmer Vernooij, #784)

0.20.3 2020-06-14

- Add support for remembering remote refs after push/pull. (Jelmer Vernooij, #752)
- Support passing tree and output encoding to `dulwich.patch.unified_diff`. (Jelmer Vernooij, #763)
- Fix pushing of new refs over HTTP(S) when there are no new objects to be sent. (Jelmer Vernooij, #739)
- Raise new error `HTTPUnauthorized` when the server sends back a 401. The client can then retry with credentials. (Jelmer Vernooij, #691)
- Move the guts of `bin/dulwich` to `dulwich.cli`, so it is easier to test or import. (Jelmer Vernooij)
- Install `dulwich` script from `entry_points` when `setuptools` is available, making it slightly easier to use on Windows. (Jelmer Vernooij, #540)
- Set `python_requires>=3.5` in `setup.py`. (Manuel Jacob)

0.20.2 2020-06-01

- Brown bag release to fix uploads of Windows wheels.

0.20.1 2020-06-01

- Publish binary wheels for: Windows, Linux, Mac OS X. (Jelmer Vernooij, #711, #710, #629)

0.20.0 2020-06-01

- Drop support for Python 2. (Jelmer Vernooij)
- Only return files from the loose store that look like git objects. (Nicolas Dandrimont)
- Ignore `agent=` capability if sent by client. (Jelmer Vernooij)
- Don't break when encountering block devices. (Jelmer Vernooij)
- Decode URL paths in `HttpGitClient` using `utf-8` rather than file system encoding. (Manuel Jacob)
- Fix pushing from a shallow clone. (Brecht Machiels, #705)

0.19.16 2020-04-17

- Don't send "deepen None" to server if graph walker supports shallow. (Jelmer Vernooij, #747)
- Support tweaking the compression level for loose objects through the "core.looseCompression" and "core.compression" settings. (Jelmer Vernooij)
- Support tweaking the compression level for pack objects through the "core.packCompression" and "core.compression" settings. (Jelmer Vernooij)
- Add a "dulwich.contrib.diffstat" module. (Kevin Hendricks)

0.19.15 2020-01-26

- Properly handle files that are just executable for the current user. (Jelmer Vernooij, #734)
- Fix handling of stored encoding in `dulwich.porcelain.get_object_by_path` on Python 3. (Jelmer Vernooij)
- Support the `include_trees` and `rename_detector` arguments at the same time when diffing trees. (Jelmer Vernooij)

0.19.14 2019-11-30

- Strip superfluous `<>` around email. (monnerat)
- Stop checking for ref validity client-side. Users can still call `check_wants` manually. (Jelmer Vernooij)
- Switch over to Google-style docstrings. (Jelmer Vernooij)
- Add a `dulwich.porcelain.active_branch` function. (Jelmer Vernooij)

- Cleanup new directory if clone fails. (Jelmer Vernooij, #733)
- Expand “~” in global exclude path. (Jelmer Vernooij)

0.19.13 2019-08-19

BUG FIXES

- Avoid `PermissionError`, since it is Python3-specific. (Jelmer Vernooij)
- Fix regression that added a dependency on C git for the test suite. (Jelmer Vernooij, #720)
- Fix compatibility with Python 3.8 - mostly deprecation warnings. (Jelmer Vernooij)

0.19.12 2019-08-13

BUG FIXES

- Update directory detection for `get_unstaged_changes` for Python 3. (Boris Feld, #684)
- Add a basic `porcelain.clean`. (Lane Barlow, #398)
- Fix output format of `porcelain.diff` to match that of C Git. (Boris Feld)
- Return a 404 not found error when repository is not found.
- Mark `.git` directories as hidden on Windows. (Martin Packman, #585)
- Implement `RefsContainer.__iter__` (Jelmer Vernooij, #717)
- Don't trust modes if they can't be modified after a file has been created. (Jelmer Vernooij, #719)

0.19.11 2019-02-07

IMPROVEMENTS

- Use `fullname` from `gecos` field, if available. (Jelmer Vernooij)
- Support `GIT_AUTHOR_NAME / GIT_AUTHOR_EMAIL`. (Jelmer Vernooij)
- Add support for short ids in `parse_commit`. (Jelmer Vernooij)
- Add support for `prune` and `prune_tags` arguments to `porcelain.fetch`. (Jelmer Vernooij, #681)

BUG FIXES

- Fix handling of race conditions when new packs appear. (Jelmer Vernooij)

0.19.10 2018-01-15

IMPROVEMENTS

- Add `dulwich.porcelain.write_tree`. (Jelmer Vernooij)
- Support reading `MERGE_HEADS` in `Repo.do_commit`. (Jelmer Vernooij)
- Import from `collections.abc` rather than `collections` where applicable. Required for 3.8 compatibility. (Jelmer Vernooij)
- Support plain strings as `refspec` arguments to `dulwich.porcelain.push`. (Jelmer Vernooij)
- Add support for creating signed tags. (Jelmer Vernooij, #542)

BUG FIXES

- Handle invalid ref that pretends to be a sub-folder under a valid ref. (KS Chan)

0.19.9 2018-11-17

BUG FIXES

- Avoid fetching ghosts in `Repo.fetch`. (Jelmer Vernooij)
- Preserve port and username in parsed HTTP URLs. (Jelmer Vernooij)
- Add basic server side implementation of `git-upload-archive`. (Jelmer Vernooij)

0.19.8 2018-11-06

- Fix encoding when reading README file in `setup.py`. (egor <egor@sourced.tech>, #668)

0.19.7 2018-11-05

CHANGES

- Drop support for Python 3 < 3.4. This is because `pkg_resources` (which get used by `setuptools` and `mock`) no longer supports 3.3 and earlier. (Jelmer Vernooij)

IMPROVEMENTS

- Support `depth` argument to `GitClient.fetch_pack` and support fetching and updating shallow metadata. (Jelmer Vernooij, #240)

BUG FIXES

- Don't write to `stdout` and `stderr` when they are not available (such as is the case for `pythonw`). (Sylvia van Os, #652)
- Fix compatibility with newer versions of `git`, which expect `CONTENT_LENGTH` to be set to 0 for empty body requests. (Jelmer Vernooij, #657)
- Raise an exception client-side when a caller tries to request SHAs that are not directly referenced the servers' refs. (Jelmer Vernooij)
- Raise more informative errors when unable to connect to repository over SSH or subprocess. (Jelmer Vernooij)
- Handle commit identity fields with multiple ">" characters. (Nicolas Dandrimont)

IMPROVEMENTS

- `dulwich.porcelain.get_object_by_path` method for easily accessing a path in another tree. (Jelmer Vernooij)
- Support the `il8n.commitEncoding` setting in `config`. (Jelmer Vernooij)

0.19.6 2018-08-11

BUG FIXES

- Fix support for custom transport arguments in `dulwich.porcelain.clone`. (Semyon Slepov)
- Fix compatibility with Python 3.8 (Jelmer Vernooij, Daniel M. Capella)
- Fix some corner cases in `path_to_tree_path`. (Romain Keramitas)
- Support paths as bytestrings in various places in `dulwich.index` (Jelmer Vernooij)
- Avoid `setup.cfg` for now, since it seems to break `pypi` metadata. (Jelmer Vernooij, #658)

0.19.5 2018-07-08

IMPROVEMENTS

- Add `porcelain.describe`. (Sylvia van Os)

BUG FIXES

- Fix regression in `dulwich.porcelain.clone` that prevented cloning of remote repositories. (Jelmer Vernooij, #639)
- Don't leave around empty parent directories for removed refs. (Damien Tournoud, #640)

0.19.4 2018-06-24

IMPROVEMENTS

- Add `porcelain.ls_files`. (Jelmer Vernooij)
- Add `Index.items`. (Jelmer Vernooij)

BUG FIXES

- Avoid unicode characters (e.g. the digraph `ijin` my surname) in `setup.cfg`, since `setuptools` doesn't deal well with them. See <https://github.com/pypa/setuptools/issues/1062>. (Jelmer Vernooij, #637)

0.19.3 2018-06-17

IMPROVEMENTS

- Add really basic `dulwich.porcelain.fsck` implementation. (Jelmer Vernooij)
- When the `DULWICH_PDB` environment variable is set, make `SIGQUIT` open `pdb` in the `'dulwich'` command.
- Add `checkout` argument to `Repo.clone`. (Jelmer Vernooij, #503)
- Add `Repo.get_shallow` method. (Jelmer Vernooij)
- Add basic `dulwich.stash` module. (Jelmer Vernooij)
- Support a `prefix` argument to `dulwich.archive.tar_stream`. (Jelmer Vernooij)

BUG FIXES

- Fix handling of encoding for tags. (Jelmer Vernooij, #608)
- Fix tutorial tests on Python 3. (Jelmer Vernooij, #573)
- Fix remote refs created by `porcelain.fetch`. (Daniel Andersson, #623)
- More robust pack creation on Windows. (Daniel Andersson)
- Fix recursive option for `porcelain.ls_tree`. (Romain Keramitas)

TESTS

- Some improvements to `paramiko` tests. (Filipp Frizzy)

0.19.2 2018-04-07

BUG FIXES

- Fix deprecated `Index.iterblobs` method. (Jelmer Vernooij)

0.19.1 2018-04-05

IMPROVEMENTS

- Add `'dulwich.mailmap'` file for reading mailmap files. (Jelmer Vernooij)
- Dulwich no longer depends on `urllib3[secure]`. Instead, `"dulwich[https]"` can be used to pull in the necessary dependencies for HTTPS support. (Jelmer Vernooij, #616)
- Support the `http.sslVerify` and `http.sslCAInfo` configuration options. (Jelmer Vernooij)
- Factor out `dulwich.client.parse_rsync_url` function. (Jelmer Vernooij)

- Fix repeat HTTP requests using the same smart HTTP client. (Jelmer Vernooij)
- New 'client.PLinkSSHVendor' for creating connections using PuTTY's plink.exe. (Adam Bradley, Philipp Frizzy)
- Only pass in *key_filename* and *password* to SSHVendor implementations if those parameters are set. (This helps with older SSHVendor implementations) (Jelmer Vernooij)

API CHANGES

- Index.iterblobs has been renamed to Index.iterobjects. (Jelmer Vernooij)

0.19.0 2018-03-10

BUG FIXES

- Make *dulwich.archive* set the gzip header file modification time so that archives created from the same Git tree are always identical. (#577, Jonas Haag)
- Allow comment characters (#, ;) within configuration file strings (Daniel Andersson, #579)
- Raise exception when passing in invalid author/commmitter values to Repo.do_commit(). (Jelmer Vernooij, #602)

IMPROVEMENTS

- Add a fastimport *extra*. (Jelmer Vernooij)
- Start writing reflog entries. (Jelmer Vernooij)
- Add ability to use password and keyfile ssh options with SSHVendor. (Filipp Kucheryavy)
- Add *change_type_same* flag to *tree_changes*. (Jelmer Vernooij)

API CHANGES

- `GitClient.send_pack` now accepts a `generate_pack_data` rather than a `generate_pack_contents` function for performance reasons. (Jelmer Vernooij)
- Dulwich now uses `urllib3` internally for HTTP requests. The *opener* argument to *dulwich.client.HttpGitClient* that took a *urllib2* opener instance has been replaced by a *pool_manager* argument that takes a *urllib3* pool manager instance. (Daniel Andersson)

0.18.6 2017-11-11

BUG FIXES

- Fix handling of empty repositories in `porcelain.clone`. (#570, Jelmer Vernooij)
- Raise an error when attempting to add paths that are not under the repository. (Jelmer Vernooij)
- Fix error message for missing trailing]. (Daniel Andersson)
- Raise `EmptyFileException` when corruption (in the form of an empty file) is detected. (Antoine R. Dumont, #582)

IMPROVEMENTS

- Enforce date field parsing consistency. This also add checks on those date fields for potential overflow. (Antoine R. Dumont, #567)

0.18.5 2017-10-29

BUG FIXES

- Fix cwd for hooks. (Fabian Grünbichler)

- Fix setting of origin in config when non-standard origin is passed into `Repo.clone`. (Kenneth Lareau, #565)
- Prevent setting SSH arguments from SSH URLs when using SSH through a subprocess. Note that Dulwich doesn't support cloning submodules. (CVE-2017-16228) (Jelmer Vernooij)

IMPROVEMENTS

- Silently ignored directories in `Repo.stage`. (Jelmer Vernooij, #564)

API CHANGES

- `GitFile` now raises `FileLocked` when encountering a lock rather than `OSError(EEXIST)`. (Jelmer Vernooij)

0.18.4 2017-10-01

BUG FIXES

- Make default User-Agent start with "git" because GitHub won't response to HTTP smart server requests otherwise (and reply with a 404). (Jelmer vernooij, #562)

0.18.3 2017-09-03

BUG FIXES

- Read config during porcelain operations that involve remotes. (Jelmer Vernooij, #545)
- Fix headers of empty chunks in unified diffs. (Taras Postument, #543)
- Properly follow redirects over HTTP. (Jelmer Vernooij, #117)

IMPROVEMENTS

- Add `dulwich.porcelain.update_head`. (Jelmer Vernooij, #439)
- `GitClient.fetch_pack` now returns symrefs. (Jelmer Vernooij, #485)
- The server now supports providing symrefs. (Jelmer Vernooij, #485)
- Add `dulwich.object_store.commit_tree_changes` to incrementally commit changes to a tree structure. (Jelmer Vernooij)
- Add basic `PackBasedObjectStore.repack` method. (Jelmer Vernooij, Earl Chew, #296, #549, #552)

0.18.2 2017-08-01

TEST FIXES

- Use constant timestamp so tests pass in all timezones, not just BST. (Jelmer Vernooij)

0.18.1 2017-07-31

BUG FIXES

- Fix syntax error in `dulwich.contrib.test_swift_smoke`. (Jelmer Vernooij)

0.18.0 2017-07-31

BUG FIXES

- Fix remaining tests on Windows. (Jelmer Vernooij, #493)
- Fix build of C extensions with Python 3 on Windows. (Jelmer Vernooij)
- Pass 'mkdir' argument onto `Repo.init_bare` in `Repo.clone`. (Jelmer Vernooij, #504)

- In `dulwich.porcelain.add`, if no files are specified, add from current working directory rather than repository root. (Jelmer Vernooij, #521)
- Properly deal with submodules in `'porcelain.status'`. (Jelmer Vernooij, #517)
- `dulwich.porcelain.remove` now actually removes files from disk, not just from the index. (Jelmer Vernooij, #488)
- Fix handling of “reset” command with markers and without “from”. (Antoine Pietri)
- Fix handling of “merge” command with markers. (Antoine Pietri)
- Support treeish argument to `porcelain.reset()`, rather than requiring a ref/commit id. (Jelmer Vernooij)
- Handle race condition when mtime doesn't change between writes/reads. (Jelmer Vernooij, #541)
- Fix `dulwich.porcelain.show` on commits with Python 3. (Jelmer Vernooij, #532)

IMPROVEMENTS

- Add basic support for reading ignore files in `dulwich.ignore`. `dulwich.porcelain.add` and `dulwich.porcelain.status` now honor ignores. (Jelmer Vernooij, Segev Finer, #524, #526)
- New `dulwich.porcelain.check_ignore` command. (Jelmer Vernooij)
- `dulwich.porcelain.status` now supports a `ignored` argument. (Jelmer Vernooij)

DOCUMENTATION

- Clarified docstrings for `Client.{send_pack,fetch_pack}` implementations. (Jelmer Vernooij, #523)

0.17.3 2017-03-20

PLATFORM SUPPORT

- List Python 3.3 as supported. (Jelmer Vernooij, #513)

BUG FIXES

- Fix compatibility with pypy 3. (Jelmer Vernooij)

0.17.2 2017-03-19

BUG FIXES

- Add workaround for https://bitbucket.org/pypy/pypy/issues/2499/cpyext-pystring_asstring-doesnt-work, fixing Dulwich when used with C extensions on pypy < 5.6. (Victor Stinner)
- Properly quote config values with a '#' character in them. (Jelmer Vernooij, #511)

0.17.1 2017-03-01

IMPROVEMENTS

- Add basic 'dulwich pull' command. (Jelmer Vernooij)

BUG FIXES

- Cope with existing submodules during pull. (Jelmer Vernooij, #505)

0.17.0 2017-03-01

TEST FIXES

- Skip test that requires sync to synchronize filesystems if `os.sync` is not available. (Koen Martens)

IMPROVEMENTS

- Implement `MemoryRepo.{set_description,get_description}`. (Jelmer Vernooij)
- Raise exception in `Repo.stage()` when absolute paths are passed in. Allow passing in relative paths to `porcelain.add()`. (Jelmer Vernooij)

BUG FIXES

- Handle multi-line quoted values in config files. (Jelmer Vernooij, #495)
- Allow `porcelain.clone` of repository without HEAD. (Jelmer Vernooij, #501)
- Support passing tag ids to `Walker()`'s `include` argument. (Jelmer Vernooij)
- Don't strip trailing newlines from extra headers. (Nicolas Dandrimont)
- Set `bufsize=0` for subprocess interaction with SSH client. Fixes hangs on Python 3. (René Stern, #434)
- Don't drop first slash for SSH paths, except for those starting with "~". (Jelmer Vernooij, René Stern, #463)
- Properly log off after retrieving just refs. (Jelmer Vernooij)

0.16.3 2016-01-14

TEST FIXES

- Remove racy check that relies on clock time changing between writes. (Jelmer Vernooij)

IMPROVEMENTS

- Add `porcelain.remote_add`. (Jelmer Vernooij)

0.16.2 2016-01-14

IMPROVEMENTS

- Fixed failing test-cases on windows. (Koen Martens)

API CHANGES

- `Repo` is now a context manager, so that it can be easily closed using a `with` statement. (Søren Løvborg)

TEST FIXES

- Only run `worktree list` compat tests against git 2.7.0, when 'git worktree list' was introduced. (Jelmer Vernooij)

BUG FIXES

- Ignore `filemode` when building index when `core.filemode` is false. (Koen Martens)
- Initialize `core.filemode` configuration setting by probing the filesystem for trustable permissions. (Koen Martens)
- Fix `porcelain.reset` to respect the `comittish` argument. (Koen Martens)
- Fix `dulwich.porcelain.ls_remote()` on Python 3. (#471, Jelmer Vernooij)
- Allow both unicode and byte strings for host paths in `dulwich.client`. (#435, Jelmer Vernooij)
- Add remote from `porcelain.clone`. (#466, Jelmer Vernooij)
- Fix unquoting of credentials before passing to `urllib2`. (#475, Volodymyr Holovko)
- Cope with submodules in `build_index_from_tree`. (#477, Jelmer Vernooij)

- Handle deleted files in *get_unstaged_changes*. (#483, Doug Hellmann)
- Don't overwrite files when they haven't changed in *build_file_from_blob*. (#479, Benoît HERVIER)
- Check for existence of index file before opening pack. Fixes a race when new packs are being added. (#482, wme)

0.16.1 2016-12-25

BUG FIXES

- Fix python3 compatibility for *dulwich.contrib.release_robot*. (Jelmer Vernooij)

0.16.0 2016-12-24

IMPROVEMENTS

- Add support for worktrees. See *git-worktree(1)* and *gitrepository-layout(5)*. (Laurent Rineau)
- Add support for *commondir* file in Git control directories. (Laurent Rineau)
- Add support for passwords in HTTP URLs. (Jon Bain, Mika Mäenpää)
- Add *release_robot* script to contrib, allowing easy finding of current version based on Git tags. (Mark Mikofski)
- Add `Blob.splitlines` method. (Jelmer Vernooij)

BUG FIXES

- Fix handling of `Commit.tree` being set to an actual tree object rather than a tree id. (Jelmer Vernooij)
- Return remote refs from `LocalGitClient.fetch_pack()`, consistent with the documentation for that method. (#461, Jelmer Vernooij)
- Fix handling of unknown URL schemes in *get_transport_and_path*. (#465, Jelmer Vernooij)

0.15.0 2016-10-09

BUG FIXES

- Allow missing trailing LF when reading service name from HTTP servers. (Jelmer Vernooij, Andrew Shadura, #442)
- Fix *dulwich.porcelain.pull()* on Python3. (Jelmer Vernooij, #451)
- Properly pull in tags during *dulwich.porcelain.clone*. (Jelmer Vernooij, #408)

CHANGES

- Changed license from “GNU General Public License, version 2.0 or later” to “Apache License, version 2.0 or later or GNU General Public License, version 2.0 or later”. (#153)

IMPROVEMENTS

- Add *dulwich.porcelain.ls_tree* implementation. (Jelmer Vernooij)

0.14.1 2016-07-05

BUG FIXES

- Fix regression removing untouched refs when pushing over SSH. (Jelmer Vernooij#441)
- Skip Python3 tests for SWIFT contrib module, as it has not yet been ported.

0.14.0 2016-07-03

BUG FIXES

- Fix `ShaFile.id` after modification of a copied `ShaFile`. (Félix Matrat, Jelmer Vernooij)
- Support removing refs from `porcelain.push`. (Jelmer Vernooij, #437)
- Stop magic protocol ref `capabilities^{}` from leaking out to clients. (Jelmer Vernooij, #254)

IMPROVEMENTS

- Add `dulwich.config.parse_submodules` function.
- Add `RefsContainer.follow` method. (#438)

0.13.0 2016-04-24

IMPROVEMENTS

- Support `ssh://` URLs in `get_transport_and_path_from_url()`. (Jelmer Vernooij, #402)
- Support missing empty line after headers in Git commits and tags. (Nicolas Dandrimont, #413)
- Fix `dulwich.porcelain.status` when used in empty trees. (Jelmer Vernooij, #415)
- Return copies of objects in `MemoryObjectStore` rather than references, making the behaviour more consistent with that of `DiskObjectStore`. (Félix Matrat, Jelmer Vernooij)
- Fix `dulwich.web` on Python3. (#295, Jonas Haag)

CHANGES

- Drop support for Python 2.6.
- Fix python3 client web support. (Jelmer Vernooij)

BUG FIXES

- Fix hang on Gzip decompression. (Jonas Haag)
- Don't rely on working `tell()` and `seek()` methods on `wsgi.input`. (Jonas Haag)
- Support `fastexport/fastimport` functionality on python3 with newer versions of `fastimport` ($\geq 0.9.5$). (Jelmer Vernooij, Félix Matrat)

0.12.0 2015-12-13

IMPROVEMENTS

- Add a `dulwich.archive` module that can create tarballs. Based on code from Jonas Haag in klaus.
- Add a `dulwich.reflog` module for reading and writing reflogs. (Jelmer Vernooij)
- Fix handling of ambiguous refs in `parse_ref` to make it match the behaviour described in <https://git-scm.com/docs/gitrevisions>. (Chris Bunney)
- Support Python3 in C modules. (Lele Gaifax)

BUG FIXES

- Simplify handling of SSH command invocation. Fixes quoting of paths. Thanks, Thomas Liebetraut. (#384)
- Fix inconsistent handling of trailing slashes for `DictRefsContainer`. (#383)
- Add hack to support thin packs during `fetch()`, albeit while requiring the entire pack file to be loaded into memory. (jsbain)

CHANGES

- This will be the last release to support Python 2.6.

0.11.2 2015-09-18

IMPROVEMENTS

- Add support for agent= capability. (Jelmer Vernooij, #298)
- Add support for quiet capability. (Jelmer Vernooij)

CHANGES

- The ParamikoSSHVendor class has been moved to
- dulwich.contrib.paramiko_vendor, as it's currently untested. (Jelmer Vernooij, #364)

0.11.1 2015-09-13

Fix-up release to exclude broken blame.py file.

0.11.0 2015-09-13

IMPROVEMENTS

- Extended Python3 support to most of the codebase. (Gary van der Merwe, Jelmer Vernooij)
- The *Repo* object has a new *close* method that can be called to close any open resources. (Gary van der Merwe)
- Support 'git.bat' in SubprocessGitClient on Windows. (Stefan Zimmermann)
- Advertise 'ofs-delta' capability in receive-pack server side capabilities. (Jelmer Vernooij)
- Switched *default_local_git_client_cls* to *LocalGitClient*. (Gary van der Merwe)
- Add *porcelain.ls_remote* and *GitClient.get_refs*. (Michael Edgar)
- Add *Repo.discover* method. (B. M. Corser)
- Add *dulwich.objectspec.parse_refspect*. (Jelmer Vernooij)
- Add *porcelain.pack_objects* and *porcelain.repack*. (Jelmer Vernooij)

BUG FIXES

- Fix handling of 'done' in graph walker and implement the 'no-done' capability. (Tommy Yu, #88)
- Avoid recursion limit issues resolving deltas. (William Grant, #81)
- Allow arguments in local client binary path overrides. (Jelmer Vernooij)
- Fix handling of commands with arguments in paramiko SSH client. (Andreas Klöckner, Jelmer Vernooij, #363)
- Fix parsing of quoted strings in configs. (Jelmer Vernooij, #305)

0.10.1 2015-03-25

BUG FIXES

- Return *ApplyDeltaError* when encountering delta errors in both C extensions and native delta application code. (Jelmer Vernooij, #259)

0.10.0 2015-03-22

BUG FIXES

- In *dulwich.index.build_index_from_tree*, by default refuse to create entries that start with *.git/*.
- Fix running of testsuite when installed. (Jelmer Vernooij, #223)
- Use a block cache in *_find_content_rename_candidates()*, improving performance. (Mike Williams)
- Add support for *core.protectNTFS* setting. (Jelmer Vernooij)

- Fix TypeError when fetching empty updates. (Hwee Miin Koh)
- Resolve delta refs when pulling into a MemoryRepo. (Max Shawabkeh, #256)
- Fix handling of tags of non-commits in missing object finder. (Augie Fackler, #211)
- Explicitly disable mmap on plan9 where it doesn't work. (Jeff Sickel)

IMPROVEMENTS

- New public method *Repo.reset_index*. (Jelmer Vernooij)
- Prevent duplicate parsing of loose files in objects directory when reading. Thanks to David Keijsers for the report. (Jelmer Vernooij, #231)

0.9.9 2015-03-20

SECURITY BUG FIXES

- Fix buffer overflow in C implementation of *pack_apply_delta()*. (CVE-2015-0838)
Thanks to Ivan Fratric of the Google Security Team for reporting this issue. (Jelmer Vernooij)

0.9.8 2014-11-30

BUG FIXES

- Various fixes to improve test suite running on Windows. (Gary van der Merwe)
- Limit delta copy length to 64K in v2 pack files. (Robert Brown)
- Strip newline from final ACKed SHA while fetching packs. (Michael Edgar)
- Remove assignment to *PyList_SIZE()* that was causing segfaults on pypy. (Jelmer Vernooij, #196)

IMPROVEMENTS

- Add porcelain 'receive-pack' and 'upload-pack'. (Jelmer Vernooij)
- Handle SIGINT signals in bin/dulwich. (Jelmer Vernooij)
- Add 'status' support to bin/dulwich. (Jelmer Vernooij)
- Add 'branch_create', 'branch_list', 'branch_delete' porcelain. (Jelmer Vernooij)
- Add 'fetch' porcelain. (Jelmer Vernooij)
- Add 'tag_delete' porcelain. (Jelmer Vernooij)
- Add support for serializing/deserializing 'gpgsig' attributes in Commit. (Jelmer Vernooij)

CHANGES

- dul-web is now available as 'dulwich web-daemon'. (Jelmer Vernooij)
- *dulwich.porcelain.tag* has been renamed to *tag_create*. *dulwich.porcelain.list_tags* has been renamed to *tag_list*. (Jelmer Vernooij)

API CHANGES

- Restore support for Python 2.6. (Jelmer Vernooij, Gary van der Merwe)

0.9.7 2014-06-08

BUG FIXES

- Fix tests dependent on hash ordering. (Michael Edgar)
- Support staging symbolic links in *Repo.stage*. (Robert Brown)
- Ensure that all files object are closed when running the test suite. (Gary van der Merwe)

- When writing OFS_DELTA pack entries, write correct offset. (Augie Fackler)
- Fix handler of larger copy operations in packs. (Augie Fackler)
- Various fixes to improve test suite running on Windows. (Gary van der Merwe)
- Fix logic for extra adds of identical files in rename detector. (Robert Brown)

IMPROVEMENTS

- Add porcelain 'status'. (Ryan Faulkner)
- Add porcelain 'daemon'. (Jelmer Vernooij)
- Add *dulwich.greenthreads* module which provides support for concurrency of some object store operations. (Fabien Boucher)
- Various changes to improve compatibility with Python 3. (Gary van der Merwe, Hannu Valtonen, michael-k)
- Add OpenStack Swift backed repository implementation in *dulwich.contrib*. See *README.swift* for details. (Fabien Boucher)

API CHANGES

- An optional close function can be passed to the Protocol class. This will be called by its close method. (Gary van der Merwe)
- All classes with close methods are now context managers, so that they can be easily closed using a *with* statement. (Gary van der Merwe)
- Remove deprecated *num_objects* argument to *write_pack* methods. (Jelmer Vernooij)

OTHER CHANGES

- The 'dul-daemon' script has been removed. The same functionality is now available as 'dulwich daemon'. (Jelmer Vernooij)

0.9.6 2014-04-23

IMPROVEMENTS

- Add support for recursive add in 'git add'. (Ryan Faulkner, Jelmer Vernooij)
- Add porcelain 'list_tags'. (Ryan Faulkner)
- Add porcelain 'push'. (Ryan Faulkner)
- Add porcelain 'pull'. (Ryan Faulkner)
- Support 'http.proxy' in *HttpGitClient*. (Jelmer Vernooij, #1096030)
- Support 'http.useragent' in *HttpGitClient*. (Jelmer Vernooij)
- In server, wait for clients to send empty list of wants when talking to empty repository. (Damien Tournoud)
- Various changes to improve compatibility with Python 3. (Gary van der Merwe)

BUG FIXES

- Support unseekable 'wsgi.input' streams. (Jonas Haag)
- Raise *TypeError* when passing *unicode()* object to *Repo.__getitem__*. (Jonas Haag)
- Fix handling of *reset* command in *dulwich.fastexport*. (Jelmer Vernooij, #1249029)
- In client, don't wait for server to close connection first. Fixes hang when used against GitHub server implementation. (Siddharth Agarwal)

- DeltaChainIterator: fix a corner case where an object is inflated as an object already in the repository. (Damien Tournoud, #135)
- Stop leaking file handles during pack reload. (Damien Tournoud)
- Avoid reopening packs during pack cache reload. (Jelmer Vernooij)

API CHANGES

- Drop support for Python 2.6. (Jelmer Vernooij)

0.9.5 2014-02-23

IMPROVEMENTS

- Add porcelain 'tag'. (Ryan Faulkner)
- New module *dulwich.objectspec* for parsing strings referencing objects and commit ranges. (Jelmer Vernooij)
- Add shallow branch support. (milki)
- Allow passing urllib2 *opener* into *HttpGitClient*. (Dov Feldstern, #909037)

CHANGES

- Drop support for Python 2.4 and 2.5. (Jelmer Vernooij)

API CHANGES

- Remove long deprecated *Repo.commit*, *Repo.get_blob*, *Repo.tree* and *Repo.tag*. (Jelmer Vernooij)
- Remove long deprecated *Repo.revision_history* and *Repo.ref*. (Jelmer Vernooij)
- Remove long deprecated *Tree.entries*. (Jelmer Vernooij)

BUG FIXES

- Raise *KeyError* rather than *TypeError* when passing in unicode object of length 20 or 40 to *Repo.__getitem__*. (Jelmer Vernooij)
- Use 'rm' rather than 'unlink' in tests, since the latter does not exist on OpenBSD and other platforms. (Dmitrij D. Czarkoff)

0.9.4 2013-11-30

IMPROVEMENTS

- Add *ssh_kwargs* attribute to *ParamikoSSHVendor*. (milki)
- Add *Repo.set_description()*. (Vidur Valberg Guðmundsson)
- Add a basic *dulwich.porcelain* module. (Jelmer Vernooij, Marcin Kuzminski)
- Various performance improvements for object access. (Jelmer Vernooij)
- New function *get_transport_and_path_from_url*, similar to *get_transport_and_path* but only supports URLs. (Jelmer Vernooij)
- Add support for *file://* URLs in *get_transport_and_path_from_url*. (Jelmer Vernooij)
- Add *LocalGitClient* implementation. (Jelmer Vernooij)

BUG FIXES

- Support filesystems with 64bit inode and device numbers. (André Roth)

CHANGES

- Ref handling has been moved to `dulwich.refs`. (Jelmer Vernooij)

API CHANGES

- Remove long deprecated `RefsContainer.set_ref()`. (Jelmer Vernooij)
- `Repo.ref()` is now deprecated in favour of `Repo.refs[]`. (Jelmer Vernooij)

FEATURES

- Add support for graftpoints. (milki)

0.9.3 2013-09-27

BUG FIXES

- Fix path for `stdint.h` in `MANIFEST.in`. (Jelmer Vernooij)

0.9.2 2013-09-26

BUG FIXES

- Include `stdint.h` in `MANIFEST.in` (Mark Mikofski)

0.9.1 2013-09-22

BUG FIXES

- Support lookups of 40-character refs in `BaseRepo.__getitem__`. (Chow Loong Jin, Jelmer Vernooij)
- Fix fetching packs with `side-band-64k` capability disabled. (David Keijser, Jelmer Vernooij)
- Several fixes in `send-pack` protocol behaviour - handling of empty pack files and deletes. (milki, #1063087)
- Fix capability negotiation when fetching packs over HTTP. (#1072461, William Grant)
- Enforce `determine_wants` returning an empty list rather than `None`. (Fabien Boucher, Jelmer Vernooij)
- In the server, support pushes just removing refs. (Fabien Boucher, Jelmer Vernooij)

IMPROVEMENTS

- Support passing a single revision to `BaseRepo.get_walker()` rather than a list of revisions. (Alberto Ruiz)
- Add `Repo.get_description` method. (Jelmer Vernooij)
- Support thin packs in `Pack.iterobjects()` and `Pack.get_raw()`. (William Grant)
- Add `MemoryObjectStore.add_pack` and `MemoryObjectStore.add_thin_pack` methods. (David Bennett)
- Add paramiko-based SSH vendor. (Aaron O'Mullan)
- Support running `'dulwich.server'` and `'dulwich.web'` using `'python -m'`. (Jelmer Vernooij)
- Add `ObjectStore.close()`. (Jelmer Vernooij)
- Raise appropriate `NotImplementedError` when encountering dumb HTTP servers. (Jelmer Vernooij)

API CHANGES

- `SSHVendor.connect_ssh` has been renamed to `SSHVendor.run_command`. (Jelmer Vernooij)
- `ObjectStore.add_pack()` now returns a 3-tuple. The last element will be an `abort()` method that can be used to cancel the pack operation. (Jelmer Vernooij)

0.9.0 2013-05-31

BUG FIXES

- Push efficiency - report missing objects only. (#562676, Artem Tikhomirov)
- Use indentation consistent with C Git in config files. (#1031356, Curt Moore, Jelmer Vernooij)
- Recognize and skip binary files in diff function. (Takeshi Kanemoto)
- Fix handling of relative paths in `dulwich.client.get_transport_and_path`. (Brian Visel, #1169368)
- Preserve ordering of entries in configuration. (Benjamin Pollack)
- Support `~` expansion in SSH client paths. (milki, #1083439)
- Support relative paths in alternate paths. (milki, Michel Lespinasse, #1175007)
- Log all error messages from wsgiref server to the logging module. This makes the test suit quiet again. (Gary van der Merwe)
- Support passing None for empty tree in `changes_from_tree`. (Kevin Watters)
- Support fetching empty repository in client. (milki, #1060462)

IMPROVEMENTS:

- Add optional `honor_filemode` flag to `build_index_from_tree`. (Mark Mikofski)
- Support `core/filemode` setting when building trees. (Jelmer Vernooij)
- Add chapter on tags in tutorial. (Ryan Faulkner)

FEATURES

- Add support for mergetags. (milki, #963525)
- Add support for posix shell hooks. (milki)

0.8.7 2012-11-27

BUG FIXES

- Fix use of alternates in `DiskObjectStore.__contains__, __iter__`. (Dmitriy)
- Fix compatibility with Python 2.4. (David Carr)

0.8.6 2012-11-09

API CHANGES

- `dulwich.__init__` no longer imports client, protocol, repo and server modules. (Jelmer Vernooij)

FEATURES

- `ConfigDict` now behaves more like a dictionary. (Adam 'Cezar' Jenkins, issue #58)
- `HTTPGitApplication` now takes an optional `fallback_app` argument. (Jonas Haag, issue #67)
- Support for large pack index files. (Jameson Nash)

TESTING

- Make index entry tests a little bit less strict, to cope with slightly different behaviour on various platforms. (Jelmer Vernooij)
- `setup.py test` (available when `setuptools` is installed) now runs all tests, not just the basic unit tests. (Jelmer Vernooij)

BUG FIXES

- `Commit._deserialize` now actually deserializes the current state rather than the previous one. (Yifan Zhang, issue #59)
- Handle `None` elements in lists of `TreeChange` objects. (Alex Holmes)
- Support cloning repositories without `HEAD` set. (D-Key, Jelmer Vernooij, issue #69)
- Support `MemoryRepo.get_config`. (Jelmer Vernooij)
- In `get_transport_and_path`, pass extra keyword arguments on to `HttpGitClient`. (Jelmer Vernooij)

0.8.5 2012-03-29

BUG FIXES

- Avoid use of 'with' in `dulwich.index`. (Jelmer Vernooij)
- Be a little bit strict about OS behaviour in index tests. Should fix the tests on Debian GNU/kFreeBSD. (Jelmer Vernooij)

0.8.4 2012-03-28

BUG FIXES

- Options on the same line as sections in config files are now supported. (Jelmer Vernooij, #920553)
- Only negotiate capabilities that are also supported by the server. (Rod Cloutier, Risto Kankkunen)
- Fix parsing of invalid timezone offsets with two minus signs. (Jason R. Coombs, #697828)
- Reset environment variables during tests, to avoid test isolation leaks reading `~/.gitconfig`. (Risto Kankkunen)

TESTS

- `$HOME` is now explicitly specified for tests that use it to read `~/.gitconfig`, to prevent test isolation issues. (Jelmer Vernooij, #920330)

FEATURES

- Additional arguments to `get_transport_and_path` are now passed on to the constructor of the transport. (Sam Vilain)
- The WSGI server now transparently handles when a git client submits data using Content-Encoding: `gzip`. (David Blewett, Jelmer Vernooij)
- Add `dulwich.index.build_index_from_tree()`. (milki)

0.8.3 2012-01-21

FEATURES

- The config parser now supports the `git-config` file format as described in `git-config(1)` and can write git config files. (Jelmer Vernooij, #531092, #768687)
- `Repo.do_commit` will now use the user identity from `.git/config` or `~/.gitconfig` if none was explicitly specified. (Jelmer Vernooij)

BUG FIXES

- Allow `determine_wants` methods to include the zero sha in their return value. (Jelmer Vernooij)

0.8.2 2011-12-18

BUG FIXES

- Cope with different `zlib` buffer sizes in sha1 file parser. (Jelmer Vernooij)

- Fix `get_transport_and_path` for HTTP/HTTPS URLs. (Bruno Renié)
- Avoid calling `free_objects()` on NULL in error cases. (Chris Eberle)
- Fix use `-bare` argument to `'dulwich init'`. (Chris Eberle)
- Properly abort connections when the `determine_wants` function raises an exception. (Jelmer Vernooij, #856769)
- Tweak `xcodebuild` hack to deal with more error output. (Jelmer Vernooij, #903840)

FEATURES

- Add support for retrieving tarballs from remote servers. (Jelmer Vernooij, #379087)
- New method `update_server_info` which generates data for dumb server access. (Jelmer Vernooij, #731235)

0.8.1 2011-10-31

FEATURES

- `Repo.do_commit` has a new argument `'ref'`.
- `Repo.do_commit` has a new argument `'merge_heads'`. (Jelmer Vernooij)
- New `Repo.get_walker` method. (Jelmer Vernooij)
- New `Repo.clone` method. (Jelmer Vernooij, #725369)
- `GitClient.send_pack` now supports the `'side-band-64k'` capability. (Jelmer Vernooij)
- `HttpGitClient` which supports the smart server protocol over HTTP. “dumb” access is not yet supported. (Jelmer Vernooij, #373688)
- Add basic support for alternates. (Jelmer Vernooij, #810429)

CHANGES

- `unittest2` or `python >= 2.7` is now required for the test suite. `testtools` is no longer supported. (Jelmer Vernooij, #830713)

BUG FIXES

- Fix compilation with older versions of MSVC. (Martin gz)
- Special case `'refs/stash'` as a valid ref. (Jelmer Vernooij, #695577)
- Smart protocol clients can now change refs even if they are not uploading new data. (Jelmer Vernooij, #855993)
- Don't compile C extensions when running in pypy. (Ronny Pfannschmidt, #881546)
- Use different name for `strnlen` replacement function to avoid clashing with system `strnlen`. (Jelmer Vernooij, #880362)

API CHANGES

- `Repo.revision_history` is now deprecated in favor of `Repo.get_walker`. (Jelmer Vernooij)

0.8.0 2011-08-07

FEATURES

- New `DeltaChainIterator` abstract class for quickly iterating all objects in a pack, with implementations for pack indexing and inflation. (Dave Borowitz)
- New `walk` module with a `Walker` class for customizable commit walking. (Dave Borowitz)

- New `tree_changes_for_merge` function in `diff_tree`. (Dave Borowitz)
- Easy rename detection in `RenameDetector` even without `find_copies_harder`. (Dave Borowitz)

BUG FIXES

- Avoid storing all objects in memory when writing pack. (Jelmer Vernooij, #813268)
- Support IPv6 for `git://` connections. (Jelmer Vernooij, #801543)
- Improve performance of `Repo.revision_history()`. (Timo Schmid, #535118)
- Fix use of `SubprocessWrapper` on Windows. (Paulo Madeira, #670035)
- Fix compilation on newer versions of Mac OS X (Lion and up). (Ryan McKern, #794543)
- Prevent raising `ValueError` for correct refs in `RefContainer.__delitem__`.
- Correctly return a tuple from `MemoryObjectStore.get_raw`. (Dave Borowitz)
- Fix a bug in reading the pack checksum when there are fewer than 20 bytes left in the buffer. (Dave Borowitz)
- Support `~` in `git://` URL paths. (Jelmer Vernooij, #813555)
- Make `ShaFile.__eq__` work when other is not a `ShaFile`. (Dave Borowitz)
- `ObjectStore.get_graph_walker()` now no longer yields the same revision more than once. This has a significant improvement for performance when wide revision graphs are involved. (Jelmer Vernooij, #818168)
- Teach `ReceivePackHandler` how to read empty packs. (Dave Borowitz)
- Don't send a pack with duplicates of the same object. (Dave Borowitz)
- Teach the server how to serve a clone of an empty repo. (Dave Borowitz)
- Correctly advertise capabilities during `receive-pack`. (Dave Borowitz)
- Fix `add/add` and `add/rename` conflicts in `tree_changes_for_merge`. (Dave Borowitz)
- Use correct MIME types in web server. (Dave Borowitz)

API CHANGES

- `write_pack` no longer takes the `num_objects` argument and requires an object to be passed in that is iterable (rather than an iterator) and that provides `__len__`. (Jelmer Vernooij)
- `write_pack_data` has been renamed to `write_pack_objects` and no longer takes a `num_objects` argument. (Jelmer Vernooij)
- `take_msb_bytes`, `read_zlib_chunks`, `unpack_objects`, and `PackStreamReader.read_objects` now take an additional argument indicating a `crc32` to compute. (Dave Borowitz)
- `PackObjectIterator` was removed; its functionality is still exposed by `PackData.iterobjects`. (Dave Borowitz)
- Add a `sha` arg to `write_pack_object` to incrementally compute a SHA. (Dave Borowitz)
- Include `offset` in `PackStreamReader` results. (Dave Borowitz)
- Move `PackStreamReader` from `server` to `pack`. (Dave Borowitz)
- Extract a `check_length_and_checksum`, `compute_file_sha`, and `pack_object_header` pack helper functions. (Dave Borowitz)
- Extract a `compute_file_sha` function. (Dave Borowitz)

- Remove `move_in_thin_pack` as a separate method; `add_thin_pack` now completes the thin pack and moves it in in one step. Remove `ThinPackData` as well. (Dave Borowitz)
- Custom buffer size in `read_zlib_chunks`. (Dave Borowitz)
- New `UnpackedObject` data class that replaces ad-hoc tuples in the return value of `unpack_object` and various `DeltaChainIterator` methods. (Dave Borowitz)
- Add a `lookup_path` convenience method to `Tree`. (Dave Borowitz)
- Optionally create `RenameDetectors` without passing in tree SHAs. (Dave Borowitz)
- Optionally include unchanged entries in `RenameDetectors`. (Dave Borowitz)
- Optionally pass a `RenameDetector` to `tree_changes`. (Dave Borowitz)
- Optionally pass a request object through to server handlers. (Dave Borowitz)

TEST CHANGES

- If `setuptools` is installed, “`python setup.py test`” will now run the testsuite. (Jelmer Vernooij)
- Add a new `build_pack` test utility for building packs from a simple spec. (Dave Borowitz)
- Add a new `build_commit_graph` test utility for building commits from a simple spec. (Dave Borowitz)

0.7.1 2011-04-12

BUG FIXES

- Fix double `decref` in `_diff_tree.c`. (Ted Horst, #715528)
- Fix the build on Windows. (Pascal Quantin)
- Fix `get_transport_and_path` compatibility with pre-2.6.5 versions of Python. (Max Bowsher, #707438)
- `BaseObjectStore.determine_wants_all` no longer breaks on zero SHAs. (Jelmer Vernooij)
- `write_tree_diff()` now supports submodules. (Jelmer Vernooij)
- Fix compilation for XCode 4 and older versions of `distutils.sysconfig`. (Daniele Sluijters)

IMPROVEMENTS

- Sphinxified documentation. (Lukasz Balcerzak)
- Add `Pack.keep`. (Marc Brinkmann)

API CHANGES

- The order of the parameters to `Tree.add(name, mode, sha)` has changed, and is now consistent with the rest of Dulwich. Existing code will still work but print a `DeprecationWarning`. (Jelmer Vernooij, #663550)
- `Tree.entries()` is now deprecated in favour of `Tree.items()` and `Tree.iteritems()`. (Jelmer Vernooij)

0.7.0 2011-01-21

FEATURES

- New `dulwich.diff_tree` module for simple content-based rename detection. (Dave Borowitz)
- Add `Tree.items()`. (Jelmer Vernooij)
- Add `eof()` and `unread_pkt_line()` methods to `Protocol`. (Dave Borowitz)
- Add `write_tree_diff()`. (Jelmer Vernooij)

- Add `serve_command` function for git server commands as executables. (Jelmer Vernooij)
- `dulwich.client.get_transport_and_path` now supports rsync-style repository URLs. (Dave Borowitz, #568493)

BUG FIXES

- Correct short-circuiting operation for no-op fetches in the server. (Dave Borowitz)
- Support parsing git mbox patches without a version tail, as generated by Mercurial. (Jelmer Vernooij)
- Fix `dul-receive-pack` and `dul-upload-pack`. (Jelmer Vernooij)
- Zero-padded file modes in Tree objects no longer trigger an exception but the check code warns about them. (Augie Fackler, #581064)
- `Repo.init()` now honors the `mkdir` flag. (#671159)
- The ref format is now checked when setting a ref rather than when reading it back. (Dave Borowitz, #653527)
- Make sure pack files are closed correctly. (Tay Ray Chuan)

DOCUMENTATION

- Run the tutorial inside the test suite. (Jelmer Vernooij)
- **Reorganized and updated the tutorial.** (Jelmer Vernooij, Dave Borowitz, #610550, #610540)

0.6.2 2010-10-16

BUG FIXES

- HTTP server correctly handles empty `CONTENT_LENGTH`. (Dave Borowitz)
- Don't error when creating `GitFiles` with the default mode. (Dave Borowitz)
- `ThinPackData.from_file` now works with `resolve_ext_ref` callback. (Dave Borowitz)
- Provide `strlen()` on `mingw32` which doesn't have it. (Hans Kolek)
- Set `bare=true` in the configuratin for bare repositories. (Dirk Neumann)

FEATURES

- Use slots for core objects to save up on memory. (Jelmer Vernooij)
- Web server supports streaming progress/pack output. (Dave Borowitz)
- New public function `dulwich.pack.write_pack_header`. (Dave Borowitz)
- Distinguish between missing files and read errors in HTTP server. (Dave Borowitz)
- Initial work on support for fastimport using `python-fastimport`. (Jelmer Vernooij)
- New `dulwich.pack.MemoryPackIndex` class. (Jelmer Vernooij)
- Delegate SHA peeling to the object store. (Dave Borowitz)

TESTS

- Use `GitFile` when modifying packed-refs in tests. (Dave Borowitz)
- New tests in `test_web` with better coverage and fewer ad-hoc mocks. (Dave Borowitz)
- Standardize quote delimiters in `test_protocol`. (Dave Borowitz)
- Fix use when `testtools` is installed. (Jelmer Vernooij)

- Add trivial test for write_pack_header. (Jelmer Vernooij)
- Refactor some of dulwich.tests.compat.server_utils. (Dave Borowitz)
- Allow overwriting id property of objects in test utils. (Dave Borowitz)
- Use real in-memory objects rather than stubs for server tests. (Dave Borowitz)
- Clean up MissingObjectFinder. (Dave Borowitz)

API CHANGES

- ObjectStore.iter_tree_contents now walks contents in depth-first, sorted order. (Dave Borowitz)
- ObjectStore.iter_tree_contents can optionally yield tree objects as well. (Dave Borowitz).
- Add side-band-64k support to ReceivePackHandler. (Dave Borowitz)
- Change server capabilities methods to classmethods. (Dave Borowitz)
- Tweak server handler injection. (Dave Borowitz)
- PackIndex1 and PackIndex2 now subclass FilePackIndex, which is itself a subclass of PackIndex. (Jelmer Vernooij)

DOCUMENTATION

- Add docstrings for various functions in dulwich.objects. (Jelmer Vernooij)
- Clean up docstrings in dulwich.protocol. (Dave Borowitz)
- Explicitly specify allowed protocol commands to ProtocolGraphWalker.read_proto_line. (Dave Borowitz)
- Add utility functions to DictRefsContainer. (Dave Borowitz)

0.6.1 2010-07-22

BUG FIXES

- Fix memory leak in C implementation of sorted_tree_items. (Dave Borowitz)
- Use correct path separators for named repo files. (Dave Borowitz)
- python > 2.7 and testtools-based test runners will now also pick up skipped tests correctly. (Jelmer Vernooij)

FEATURES

- Move named file initialization to BaseRepo. (Dave Borowitz)
- Add logging utilities and git/HTTP server logging. (Dave Borowitz)
- The GitClient interface has been cleaned up and instances are now reusable. (Augie Fackler)
- Allow overriding paths to executables in GitSSHClient. (Ross Light, Jelmer Vernooij, #585204)
- Add PackBasedObjectStore.pack_loose_objects(). (Jelmer Vernooij)

TESTS

- Add tests for sorted_tree_items and C implementation. (Dave Borowitz)
- Add a MemoryRepo that stores everything in memory. (Dave Borowitz)
- Quiet logging output from web tests. (Dave Borowitz)
- More flexible version checking for compat tests. (Dave Borowitz)
- Compat tests for servers with and without side-band-64k. (Dave Borowitz)

CLEANUP

- Clean up file headers. (Dave Borowitz)

TESTS

- Use GitFile when modifying packed-refs in tests. (Dave Borowitz)

API CHANGES

- `dulwich.pack.write_pack_index_v{1,2}` now take a file-like object rather than a filename. (Jelmer Vernooij)
- Make `dul-daemon/dul-web` trivial wrappers around server functionality. (Dave Borowitz)
- Move reference WSGI handler to `web.py`. (Dave Borowitz)
- Factor out `_report_status` in `ReceivePackHandler`. (Dave Borowitz)
- Factor out a function to convert a line to a `pkt-line`. (Dave Borowitz)

0.6.0 2010-05-22

note: This list is most likely incomplete for 0.6.0.

BUG FIXES

- Fix `ReceivePackHandler` to disallow removing refs without `delete-refs`. (Dave Borowitz)
- Deal with capabilities required by the client, even if they can not be disabled in the server. (Dave Borowitz)
- Fix trailing newlines in generated patch files. (Jelmer Vernooij)
- Implement `RefsContainer.__contains__`. (Jelmer Vernooij)
- Cope with `r` in ref files on Windows. (<http://github.com/jelmer/dulwich/issues/#issue/13>, Jelmer Vernooij)
- Fix `GitFile` breakage on Windows. (Anatoly Techtonik, #557585)
- Support packed ref deletion with no peeled refs. (Augie Fackler)
- Fix send pack when there is nothing to fetch. (Augie Fackler)
- Fix fetch if no progress function is specified. (Augie Fackler)
- Allow double-staging of files that are deleted in the index. (Dave Borowitz)
- Fix `RefsContainer.add_if_new` to support dangling symrefs. (Dave Borowitz)
- Non-existent index files in non-bare repositories are now treated as empty. (Dave Borowitz)
- Always update `ShaFile.id` when the contents of the object get changed. (Jelmer Vernooij)
- Various Python2.4-compatibility fixes. (Dave Borowitz)
- Fix thin pack handling. (Dave Borowitz)

FEATURES

- Add `include-tag` capability to server. (Dave Borowitz)
- New `dulwich.fastexport` module that can generate fastexport streams. (Jelmer Vernooij)
- Implemented `BaseRepo.__contains__`. (Jelmer Vernooij)
- Add `__setitem__` to `DictRefsContainer`. (Dave Borowitz)
- Overall improvements checking Git objects. (Dave Borowitz)

- Packs are now verified while they are received. (Dave Borowitz)

TESTS

- Add framework for testing compatibility with C Git. (Dave Borowitz)
- Add various tests for the use of non-bare repositories. (Dave Borowitz)
- Cope with diffstat not being available on all platforms. (Tay Ray Chuan, Jelmer Vernooij)
- Add `make_object` and `make_commit` convenience functions to test utils. (Dave Borowitz)

API BREAKAGES

- The `'committer'` and `'message'` arguments to `Repo.do_commit()` have been swapped. `'committer'` is now optional. (Jelmer Vernooij)
- `Repo.get_blob`, `Repo.commit`, `Repo.tag` and `Repo.tree` are now deprecated. (Jelmer Vernooij)
- `RefsContainer.set_ref()` was renamed to `RefsContainer.set_symbolic_ref()`, for clarity. (Jelmer Vernooij)

API CHANGES

- The primary serialization APIs in `dulwich.objects` now work with chunks of strings rather than with full-text strings. (Jelmer Vernooij)

0.5.02010-03-03

BUG FIXES

- Support custom fields in commits (readonly). (Jelmer Vernooij)
- Improved ref handling. (Dave Borowitz)
- Rework server protocol to be smarter and interoperate with `cgit` client. (Dave Borowitz)
- Add a `GitFile` class that uses the same locking protocol for writes as `cgit`. (Dave Borowitz)
- Cope with forward slashes correctly in the index on Windows. (Jelmer Vernooij, #526793)

FEATURES

- `-pure` option to `setup.py` to allow building/installing without the C extensions. (Hal Wine, Anatoly Techtonik, Jelmer Vernooij, #434326)
- Implement `Repo.get_config()`. (Jelmer Vernooij, Augie Fackler)
- HTTP dumb and smart server. (Dave Borowitz)
- Add abstract baseclass for `Repo` that does not require file system operations. (Dave Borowitz)

0.4.1 2010-01-03

FEATURES

- Add `ObjectStore.iter_tree_contents()`. (Jelmer Vernooij)
- Add `Index.changes_from_tree()`. (Jelmer Vernooij)
- Add `ObjectStore.tree_changes()`. (Jelmer Vernooij)
- Add functionality for writing patches in `dulwich.patch`. (Jelmer Vernooij)

0.4.0 2009-10-07

DOCUMENTATION

- Added tutorial.

API CHANGES

- `dulwich.object_store.tree_lookup_path` will now return the mode and sha of the object found rather than the object itself.

BUG FIXES

- Use `binascii.hexlify` / `binascii.unhexlify` for better performance.
- Cope with extra unknown data in index files by ignoring it (for now).
- Add proper error message when server unexpectedly hangs up. (#415843)
- Correctly write opcode for equal in `create_delta`.

0.3.3 2009-07-23

FEATURES

- Implement `ShaFile.__hash__()`.
- Implement `Tree.__len__()`

BUG FIXES

- Check for 'objects' and 'refs' directories when looking for a Git repository. (#380818)

0.3.2 2009-05-20

BUG FIXES

- Support the encoding field in Commits.
- Some Windows compatibility fixes.
- Fixed several issues in commit support.

FEATURES

- Basic support for handling submodules.

0.3.1 2009-05-13

FEATURES

- Implemented `Repo.__getitem__`, `Repo.__setitem__` and `Repo.__delitem__` to access content.

API CHANGES

- Removed `Repo.set_ref`, `Repo.remove_ref`, `Repo.tags`, `Repo.get_refs` and `Repo.heads` in favor of `Repo.refs`, a dictionary-like object for accessing refs.

BUG FIXES

- Removed import of 'sha' module in `objects.py`, which was causing deprecation warnings on Python 2.6.

0.3.0 2009-05-10

FEATURES

- A new function 'commit_tree' has been added that can commit a tree based on an index.

BUG FIXES

- The memory usage when generating indexes has been significantly reduced.
- A memory leak in the C implementation of `parse_tree` has been fixed.
- The send-pack smart server command now works. (Thanks Scott Chacon)

- The handling of short timestamps (less than 10 digits) has been fixed.
- The handling of timezones has been fixed.

0.2.1 2009-04-30

BUG FIXES

- Fix compatibility with Python2.4.

0.2.0 2009-04-30

FEATURES

- Support for activity reporting in smart protocol client.
- Optional C extensions for better performance in a couple of places that are performance-critical.

0.1.1 2009-03-13

BUG FIXES

- Fixed regression in Repo.find_missing_objects()
- Don't fetch ^{ } objects from remote hosts, as requesting them causes a hangup.
- Always write pack to disk completely before calculating checksum.

FEATURES

- Allow disabling thin packs when talking to remote hosts.

0.1.0 2009-01-24

- Initial release.

CHAPTER 4

Indices and tables

- `genindex`
- `modindex`
- `search`