GitHub::Crud - Create, Read, Update, Delete files, commits, issues, and web hooks on GitHub.

Synopsis

Create, Read, Update, Delete files, commits, issues, and web hooks on GitHub as described at:

```
https://developer.github.com/v3/repos/contents/#update-a-f
```

Commit a folder to GitHub then read and check some of the uploaded content:

```
use GitHub::Crud;
use Data::Table::Text qw(:all);

my $f  = temporaryFolder;
my $c  = dateTimeStamp;
my $if = q(/home/phil/.face);

writeFile(fpe($f, q(data), $_, qw(txt)), $c) for 1..3;
copyBinaryFile $if, my $If = fpe $f, qw(face jpg);

my $g = GitHub::Crud::new
  (userid           => q(philiprbrenan),
   repository       => q(aaa),
   branch           => q(test),
   confessOnFailure => 1);

$g->loadPersonalAccessToken;
$g->writeCommit($f);

my $C = $g->read(q(data/1.txt));
my $I = $g->read(q(face.jpg));
my $i = readBinaryFile $if;

confess "Date stamp failed" unless $C eq $c;
confess "Image failed"      unless $i eq $I;
confess "Write commit succeeded";
```

Prerequisites

```
sudo apt-get install curl
```

Description

Create, Read, Update, Delete files, commits, issues, and web hooks on GitHub.

Version 20201030.

The following sections describe the methods in each functional area of this module. For an alphabetic listing of all methods by name see Index.

Constructor

Create a GitHub object with the specified attributes describing the interface with GitHub.

new(%attributes)

Create a new GitHub object with attributes as described at: "GitHub::Crud Definition".

```
   Parameter    Description
1  %attributes  Attribute values
```

Example:

```
my $f  = temporaryFolder;
my $c  = dateTimeStamp;
my $if = q(/home/phil/.face);

writeFile(fpe($f, q(data), $_, qw(txt)), $c) for 1..3;
copyBinaryFile $if, my $If = fpe $f, qw(face jpg);


my $g = GitHub::Crud::new

  (userid           => q(philiprbrenan),
   repository       => q(aaa),
   branch           => q(test),
   confessOnFailure => 1);

$g->loadPersonalAccessToken;
$g->writeCommit($f);

my $C = $g->read(q(data/1.txt));
my $I = $g->read(q(face.jpg));
my $i = readBinaryFile $if;

confess "Date stamp failed" unless $C eq $c;
confess "Image failed"      unless $i eq $I;
success "Write commit succeeded";
```

Files

File actions on the contents of GitHubrepositories.

list($gitHub)

List all the files contained in a GitHubrepository or all the files below a specified folder in the repository.

Required attributes: userid, repository.

Optional attributes: gitFolder, refOrBranch, nonRecursive, patKey.

Use the gitFolder parameter to specify the folder to start the list from, by default, the listing will start at the root folder of your repository.

Use the nonRecursive option if you require only the files in the start folder as otherwise all the folders in the start folder will be listed as well which might take some time.

If the list operation is successful, failed is set to false and fileList is set to refer to an array of the file names found.

If the list operation fails then failed is set to true and fileList is set to refer to an empty array.

Returns the list of file names found or empty list if no files were found.

	Parameter	Description
1	$gitHub	GitHub

Example:

```
  success "list:", gitHub->list;  # Example


# list: alpha.data .github/workflows/test.yaml images/aaa.t
```

specialFileData($d)

Do not encode or decode data with a known file signature

	Parameter	Description
1	$d	String to check

read($gitHub, $File)

Read data from a file on GitHub.

Required attributes: userid, repository.

Optional attributes: gitFile = the file to read, refOrBranch, patKey.

If the read operation is successful, failed is set to false and readData is set to the data read from the file.

If the read operation fails then failed is set to true and readData is set to **undef**.

Returns the data read or **undef** if no file was found.

	Parameter	Description
1	$gitHub	GitHub
2	$File	File o read if not specified in gitFile

Example:

```
my $g = gitHub;
$g->gitFile = my $f = q(z'2  'z"z.data);
my $d = q(αβγ);
$g->write($d);

confess "read FAILED" unless $g->read eq $d;  # Example

success "Read passed";
```

write($gitHub, $data, $File)

Write utf8 data into a GitHub file.

Required attributes: userid, repository, patKey. Either specify the target file on:<github> using the gitFile attribute or supply it as the third parameter. Returns **true** on success else undef.

	Parameter	Description
1	$gitHub	GitHub object
2	$data	Data to be written
3	$File	Optionally the name of the file on github

Example:

```
my $g = gitHub;
$g->gitFile = "zzz.data";

my $d = dateTimeStamp.q( αβγ);

if (1)
 {my $t = time();

  $g->write($d);  # Example


  lll "First write time: ", time() -  $t;  # Example

 }

my $r = $g->read;
lll "Write bbb: $r";
if (1)
 {my $t = time();

  $g->write($d);  # Example


  lll "Second write time: ", time() -  $t;  # Example

 }

confess "write FAILED" unless $g->exists;  # Example

success "Write passed";
```

readBlob($gitHub, $sha)

Read a blobfrom GitHub.

Required attributes: userid, repository, patKey. Returns the content of the blobidentified by the specified sha.

	Parameter	Description
1	$gitHub	GitHub object
2	$sha	Data to be written

Example:

```
my $g = gitHub;
$g->gitFile = "face.jpg";
my $d = readBinaryFile(q(/home/phil/.face));
my $s = $g->writeBlob($d);
my $S = q(4a2df549febb701ba651aae46e041923e9550cb8);
confess q(Write blob FAILED) unless $s eq $S;


my $D = $g->readBlob($s);  # Example

confess q(Write/Read blob FAILED) unless $d eq $D;
success q(Write/Read blob passed);
```

writeBlob($gitHub, $data)

Write data into a GitHub as a blob that can be referenced by future commits.

Required attributes: userid, repository, patKey. Returns the sha of the created blob or undef in a failure occurred.

	Parameter	Description
1	$gitHub	GitHub object
2	$data	Data to be written

Example:

```
my $g = gitHub;
$g->gitFile = "face.jpg";
my $d = readBinaryFile(q(/home/phil/.face));

my $s = $g->writeBlob($d);  # Example

my $S = q(4a2df549febb701ba651aae46e041923e9550cb8);
confess q(Write blob FAILED) unless $s eq $S;

my $D = $g->readBlob($s);
confess q(Write/Read blob FAILED) unless $d eq $D;
success q(Write/Read blob passed);
```

copy($gitHub, $target)

Copy a source file from one location to another target location in your GitHub repository, overwriting the target file if it already exists.

Required attributes: userid, repository, patKey, gitFile = the file to be copied.

Optional attributes: refOrBranch.

If the write operation is successful, failed is set to false otherwise it is set to true.

Returns **updated** if the write updated the file, **created** if the write created the file else **undef** if the write failed.

```
   Parameter  Description
1  $gitHub    GitHub object
2  $target    The name of the file to be created
```

Example:

```
my ($f1, $f2) = ("zzz.data", "zzz2.data");
my $g = gitHub;
$g->gitFile   = $f2; $g->delete;
$g->gitFile   = $f1;
my $d = dateTimeStamp;
my $w = $g->write($d);

my $r = $g->copy($f2);  # Example

lll "Copy created: $r";
$g->gitFile   = $f2;
my $D = $g->read;
lll "Read     ccc: $D";

confess "copy FAILED" unless $d eq $D;  # Example

success "Copy passed"
```

exists($gitHub)

Test whether a file exists on GitHub or not and returns an object including the **sha** and **size** fields if it does else undef.

Required attributes: userid, repository, gitFile file to test.

Optional attributes: refOrBranch, patKey.

```
   Parameter  Description
1  $gitHub    GitHub object
```

Example:

```
my $g = gitHub;
$g->gitFile    = "test4.html";
my $d = dateTimeStamp;
$g->write($d);

confess "exists FAILED" unless $g->read eq $d;  # Example

$g->delete;

confess "exists FAILED" if $g->read eq $d;  # Example

success "Exists passed";
```

rename($gitHub, $target)

Rename a source file on GitHub if the target file name is not already in use.

Required attributes: userid, repository, patKey, gitFile = the file to be renamed.

Optional attributes: refOrBranch.

Returns the new name of the file **renamed** if the rename was successful else **undef** if the rename failed.

```
   Parameter  Description
1  $gitHub    GitHub object
2  $target    The new name of the file
```

Example:

```
my ($f1, $f2) = qw(zzz.data zzz2.data);
my $g = gitHub;
   $g->gitFile = $f2; $g->delete;

my $d = dateTimeStamp;
$g->gitFile  = $f1;
$g->write($d);

confess "rename FAILED" unless $g->read eq $d;  # Example



$g->rename($f2);  # Example


confess "rename FAILED" if $g->exists;  # Example


$g->gitFile  = $f2;

confess "rename FAILED" if $g->read eq $d;  # Example

success "Rename passed";
```

delete($gitHub)

Delete a file from GitHub.

Required attributes: userid, repository, patKey, gitFile = the file to be deleted.

Optional attributes: refOrBranch.

If the delete operation is successful, failed is set to false otherwise it is set to true.

Returns true if the delete was successful else false.

```
   Parameter  Description
1  $gitHub    GitHub object
```

Example:

```
my $g = gitHub;
my $d = dateTimeStamp;
$g->gitFile = "zzz.data";
$g->write($d);


confess "delete FAILED" unless $g->read eq $d;  # Example


if (1)
 {my $t = time();

  my $d = $g->delete;  # Example

  lll "Delete   1: ", $d;

  lll "First delete: ", time() -  $t;  # Example


  confess "delete FAILED" if $g->exists;  # Example

 }

if (1)
 {my $t = time();

  my $d = $g->delete;  # Example

  lll "Delete   1: ", $d;

  lll "Second delete: ", time() -  $t;  # Example


  confess "delete FAILED" if $g->exists;  # Example

 }
success "Delete passed";
```

Repositories

Perform actions on GitHub repositories.

listCommits($gitHub)

List all the commits in a GitHub repository.

Required attributes: userid, repository.

```
   Parameter  Description
1  $gitHub    GitHub object
```

Example:

```
    my $c = gitHub->listCommits;  # Example

    my %s = listCommitShas $c;
    lll "Commits
",     dump $c;
    lll "Commit shas
", dump \%s;
    success "ListCommits passed";
```

listCommitShas($commits)

Create {commit name => sha} from the results of listCommits.

```
   Parameter  Description
1  $commits   Commits from listCommits
```

Example:

```
    my $c = gitHub->listCommits;

    my %s = listCommitShas $c;  # Example

    lll "Commits
",     dump $c;
    lll "Commit shas
", dump \%s;
    success "ListCommits passed";
```

writeCommit($gitHub, $folder, @files)

Write all the files in a **$folder** (or just the the named files) into a GitHub repository in parallel as a commit on the specified branch.

Required attributes: userid, repository, refOrBranch.

```
   Parameter  Description
1  $gitHub    GitHub object
2  $folder    File prefix to remove
3  @files     Files to write
```

Example:

```
my $f  = temporaryFolder;
my $c  = dateTimeStamp;
my $if = q(/home/phil/.face);

writeFile(fpe($f, q(data), $_, qw(txt)), $c) for 1..3;
copyBinaryFile $if, my $If = fpe $f, qw(face jpg);

my $g = GitHub::Crud::new
  (userid           => q(philiprbrenan),
   repository       => q(aaa),
   branch           => q(test),
   confessOnFailure => 1);

$g->loadPersonalAccessToken;

$g->writeCommit($f);


my $C = $g->read(q(data/1.txt));
my $I = $g->read(q(face.jpg));
my $i = readBinaryFile $if;

confess "Date stamp failed" unless $C eq $c;
confess "Image failed"      unless $i eq $I;
success "Write commit succeeded";
```

listWebHooks($gitHub)

List web hooks associated with your GitHubrepository.

Required: userid, repository, patKey.

If the list operation is successful, failed is set to false otherwise it is set to true.

Returns true if the list operation was successful else false.

```
   Parameter  Description
1  $gitHub    GitHub object
```

Example:

```
success join ' ', q(Webhooks:), dump(gitHub->listWebHooks);
```

createPushWebHook($gitHub)

Create a web hook for your GitHub userid.

Required: userid, repository, url, patKey.

Optional: secret.

If the create operation is successful, failed is set to false otherwise it is set to true.

Returns true if the web hook was created successfully else false.

```
   Parameter  Description
1  $gitHub    GitHub object
```

Example:

```
my $g = gitHub;

my $d = $g->createPushWebHook;  # Example

success join ' ', "Create web hook:", dump($d);
```

listRepositories($gitHub)

List the repositories accessible to a user on GitHub.

Required: userid.

Returns details of the repositories.

```
   Parameter  Description
1  $gitHub    GitHub object
```

Example:

```
success "List repositories: ", dump(gitHub()->listReposito
```

createRepository($gitHub)

Create a repository on GitHub.

Required: userid, repository.

Returns true if the issue was created successfully else false.

```
   Parameter  Description
1  $gitHub    GitHub object
```

Example:

```
gitHub(repository => q(ccc))->createRepository;  # Example

success "Create repository succeeded";
```

createRepositoryFromSavedToken($userid, $repository, $private, $accessFolderOrToken)

Create a repository on GitHub using an access token either as supplied or saved in a file using savePersonalAccessToken.

Returns true if the issue was created successfully else false.

	Parameter	Description
1	$userid	Userid on GitHub
2	$repository	The repository name
3	$private	True if the repo is private
4	$accessFolderOrToken	Location of access token.

Example:

```
createRepositoryFromSavedToken(q(philiprbrenan), q(ddd));

success "Create repository succeeded";
```

createIssue($gitHub)

Create an issue on GitHub.

Required: userid, repository, body, title.

If the operation is successful, failed is set to false otherwise it is set to true.

Returns true if the issue was created successfully else false.

	Parameter	Description
1	$gitHub	GitHub object

Example:

```
gitHub(title=>q(Hello), body=>q(World))->createIssue;  # Ex

success "Create issue succeeded";
```

createIssueFromSavedToken($userid, $repository, $title, $body, $accessFolderOrToken)

Create an issue on GitHub using an access token as supplied or saved in a file using savePersonalAccessToken.

Returns true if the issue was created successfully else false.

```
   Parameter             Description
1  $userid               Userid on GitHub
2  $repository           Repository name
3  $title                Issue title
4  $body                 Issue body
5  $accessFolderOrToken  Location of access token.
```

Example:

```
&createIssueFromSavedToken(qw(philiprbrenan ddd hello Worl

success "Create issue succeeded";
```

writeFileUsingSavedToken($userid, $repository, $file, $content, $accessFolderOrToken)

Write to a file on GitHub using a personal access token as supplied or saved in a file. Return **1** on success or confess to any failure.

	Parameter	Description
1	$userid	Userid on GitHub
2	$repository	Repository name
3	$file	File name on github
4	$content	File content
5	$accessFolderOrToken	Location of access token.

Example:

```
my $s = q(HelloWorld);

&writeFileUsingSavedToken(qw(philiprbrenan ddd hello.txt),

my $S = gitHub(repository=>q(ddd), gitFile=>q(hello.txt))->

confess "Write file using saved token FAILED" unless $s eq
success "Write file using saved token succeeded";
```

writeFileFromFileUsingSavedToken($userid, $repository, $file, $localFile, $accessFolderOrToken)

Copy a file to GitHub using a personal access token as supplied or saved in a file. Return **1** on success or confess to any failure.

```
   Parameter             Description
1  $userid               Userid on GitHub
2  $repository           Repository name
3  $file                 File name on github
4  $localFile            File content
5  $accessFolderOrToken  Location of access token.
```

Example:

```
    my $f = writeFile(undef, my $s = "World
");

    &writeFileFromFileUsingSavedToken(qw(philiprbrenan ddd

    my $S = gitHub(repository=>q(ddd), gitFile=>q(hello.txt
    confess "Write file from file using saved token FAILED'
    success "Write file from file using saved token succeec
```

readFileUsingSavedToken($userid, $repository, $file, $accessFolderOrToken)

Read from a file on GitHub using a personal access token as supplied or saved in a file. Return the content of the file on success or confess to any failure.

```
   Parameter             Description
1  $userid               Userid on GitHub
2  $repository           Repository name
3  $file                 File name on github
4  $accessFolderOrToken  Location of access token.
```

Example:

```
my $s = q(Hello to the World);
        &writeFileUsingSavedToken(qw(philiprbrenan ddd hell

my $S = &readFileUsingSavedToken (qw(philiprbrenan ddd hell


confess "Read file using saved token FAILED" unless $s eq $
success "Read file using saved token succeeded"
```

writeFolderUsingSavedToken($userid, $repository, $targetFolder, $localFolder, $accessFolderOrToken)

Write all the files in a local folder to a target folder on a named GitHub repository using a personal access token as supplied or saved in a file.

```
   Parameter             Description
1  $userid               Userid on GitHub
2  $repository           Repository name
3  $targetFolder         Target folder on github
4  $localFolder          Local folder name
5  $accessFolderOrToken  Location of access token.
```

Access tokens

Load and save access tokens. Some GitHubrequets must be signed with an oauth access token. These methods allow you to store and reuse such tokens.

savePersonalAccessToken($gitHub)

Save a GitHub personal access token by userid in folder personalAccessTokenFolder.

```
   Parameter  Description
1  $gitHub    GitHub object
```

Example:

```
my $d = temporaryFolder;
my $t = join '', 1..20;

my $g = gitHub
 (userid                    => q(philiprbrenan),
  personalAccessToken       => $t,
  personalAccessTokenFolder => $d,
 );


        $g->savePersonalAccessToken;  # Example

my $T = $g->loadPersonalAccessToken;

confess "Load/Save token FAILED" unless $t eq $T;
success "Load/Save token succeeded"
```

loadPersonalAccessToken($gitHub)

Load a personal access token by userid from folder personalAccessTokenFolder.

```
   Parameter  Description
1  $gitHub    GitHub object
```

Example:

```
my $d = temporaryFolder;
my $t = join '', 1..20;

my $g = gitHub
 (userid                    => q(philiprbrenan),
  personalAccessToken       => $t,
  personalAccessTokenFolder => $d,
 );

        $g->savePersonalAccessToken;

my $T = $g->loadPersonalAccessToken;  # Example


confess "Load/Save token FAILED" unless $t eq $T;
success "Load/Save token succeeded"
```

GitHub::Crud Definition

Attributes describing the interface with GitHub.

Input fields

body

The body of an issue.

branch

Branch name (you should create this branch first) or omit it for the default branch which is usually 'master'.

confessOnFailure

Confess to any failures

gitFile

File name on GitHub - this name can contain '/'. This is the file to be read from, written to, copied from, checked for existence or deleted.

gitFolder

Folder name on GitHub - this name can contain '/'.

message

Optional commit message

nonRecursive

Fetch only one level of files with list.

personalAccessToken

A personal access token with scope "public_repo" as generated on page: https://github.com/settings/tokens.

personalAccessTokenFolder

The folder into which to save personal access tokens. Set to q(/etc/GitHubCrudPersonalAccessToken) by default.

private

Whether the repository being created should be private or not.

repository

The name of the repository to be worked on minus the userid - you should create this repository first manually.

secret

The secret for a web hook - this is created by the creator of the web hook and remembered by GitHub,

title

The title of an issue.

userid

Userid on GitHub of the repository to be worked on.

webHookUrl

The url for a web hook.

Output fields

failed

Defined if the last request to GitHub failed else **undef**.

fileList

Reference to an array of files produced by list.

readData

Data produced by read.

response

A reference to GitHub's response to the latest request.

GitHub::Crud::Response Definition

Attributes describing a response from GitHub.

Output fields

content

The actual content of the file from GitHub.

data

The data received from GitHub, normally in Json format.

status

Our version of Status.

Index

8 exists - Test whether a file exists on GitHubor not and returns an object including the **sha** and **size** fields if it does else undef.

9 list - List all the files contained in a GitHubrepository or all the files below a specified folder in the repository.

10 listCommits - List all the commits in a GitHub repository.

11 listCommitShas - Create {commit name => sha} from the results of listCommits.

12 listRepositories - List the repositories accessible to a user on GitHub.

13 listWebHooks - List web hooks associated with your GitHub repository.

14 loadPersonalAccessToken - Load a personal access token by userid from folder personalAccessTokenFolder.

15 new - Create a new GitHub object with attributes as described at: "GitHub::Crud Definition".

16 read - Read data from a file on GitHub.

17 readBlob - Read a blob from GitHub.

18 readFileUsingSavedToken - Read from a file on GitHub using a personal access token as supplied or saved in a file.

19 rename - Rename a source file on GitHub if the target file name is not already in use.

20 savePersonalAccessToken - Save a GitHubpersonal access token by userid in folder personalAccessTokenFolder.

21 specialFileData - Do not encode or decode data with a known file signature

22 write - Write utf8 data into a GitHub file.

23 writeBlob - Write data into a GitHub as a blob that can be referenced by future commits.

24 writeCommit - Write all the files in a **$folder** (or just the the named files) into a GitHub repository in parallel as a commit on the specified branch.

25 writeFileFromFileUsingSavedToken - Copy a file to GitHub using a personal access token as supplied or saved in a file.

26 writeFileUsingSavedToken - Write to a file on GitHub using a personal access token as supplied or saved in a file.

27 writeFolderUsingSavedToken - Write all the files in a local folder to a target folder on a named GitHubrepository using a personal access token as supplied or saved in a file.

Installation

This module is written in 100% Pure Perl and, thus, it is easy to read, comprehend, use, modify and install via **cpan**:

```
sudo cpan install GitHub::Crud
```