LegiScan API Client Downloads

File	Size
legiscan-current.tar.gz (legiscan-current.tar.gz)	79.78KB
legiscan-1.4.1.tar.gz (legiscan-1.4.1.tar.gz)	79.78KB
LegiScan_API_User_Manual.pdf (LegiScan_API_User_Manual.pdf)	667.08KB
Database_ERD.png (Database_ERD.png)	161.45KB
ChangeLog (ChangeLog)	7.70KB

LegiScan API Client

About this Documentation

This is the installation documentation for installing the LegiScan API Client on Linux. This guide is written for installation from source on a Debian platform and any Linux variant should work with little modifications. There is nothing intrinsically Linux specific to the code, so Windows based servers with access to PHP should work with as well, though not officially tested with MSSQL.

Tested On

- PHP 5.6
- MariaDB 10.1 w/Percona-XtraDB
- Debian 8 (Jessie)
- PHP 7.1

- PostgreSQL 9.5
- Ubuntu LTS 16.04 (Xenial)

Requirements

- PHP 5.4 or later
- MySQL 5.6 or PostgreSQL compatible database
- A large filesystem, if storing documents locally

Further Reading

> API Client source documentation is available at: LegiScan API Client Documentation (https://api.legiscan.com/docs/index.html)
>
> Additional documentation and data dictionary is available at: LegiScan API Manual (https://legiscan.com/gaits/documentation/legiscan)
>
> For information about support options and upgrading to Push API please email us at info@legiscan.com (mailto:info@legiscan.com?subject=LegiScan%20API%20Inquiry), or call 800-618-2750 x401 (tel:8006182750,,401)

Files Included

The following files are included in the LegiScan API archive:

- LegiScan.php - LegiScan core classes
- config.dist.php - INI style configuration file settings & options
- legiscand.php - Pull Daemon script to keep local database updated
- legiscan-bulk.php - Bulk import utility for JSON dataset archives
- legiscan-cli.php - Command line tool to generate API requests
- legiscan-push.php - Listener endpoint for Push API
- legiscan-ui.php - Simple web page demo interface to LegiScan API
- common.php - utility and helper functions
- schema-mysql.sql - MySQL database schema with static table data
- schema-pgsql.sql - PostgreSQL database schema with static table data
- upgrade.php - A script to apply database changes when upgrading

Installation

In this example we will install the package under `/opt`, this could also be your root web server folder such as `/var/www`.

Unpack the archive that you downloaded from legiscan.com (https://legiscan.com/). This will create a new directory `legiscan` that has all the files necessary to interface with the LegiScan API.

```
cd /opt
wget https://api.legiscan.com/dl/legiscan-current.tar.gz
tar zxvf legiscan-current.tar.gz
cd legiscan
```

You will need to have a valid API key, if you don't already have one you can get one for free at LegiScan API Registration (https://legiscan.com/legiscan/)

Upgrade

When upgrading from a previous version unpack the archive over the existing installation, verify if there are any config.php updates that need to be applied, then run the `upgrade.php` script to make any schema and data modifications.

```
php upgrade.php
```

Database MySQL

Create the `legiscan_api` database, add a user `legiscan_api` with password `<your password>`, grant rights on the database to the new user and reload privleges.

```
mysql -u root -p
CREATE DATABASE legiscan_api DEFAULT CHARACTER SET utf8;
GRANT USAGE ON *.* TO 'legiscan_api'@'localhost' IDENTIFIED BY '<your pas
sword>';
GRANT CREATE, ALTER, SELECT, INSERT, UPDATE, DELETE ON `legiscan_api`.* T
O 'legiscan_api'@'localhost';
FLUSH PRIVILEGES;
exit
```

Next load the schema and static lookup tables.

```
mysql -D legiscan_api -u root -p < schema-mysql.sql
```

Database PostgreSQL

Create the `legiscan_api` user with password `<your password>` since in PostgreSQL the assumption is a database will exist with the same name as the role being used to login.

```
sudo -u postgres createuser -S -D -R -P legiscan_api
Enter password for new role: <your password>
Enter it again: <your password>
```

Next create the actual database `legiscan_api` owned by `legiscan_api`.

```
sudo -u postgres createdb -E UTF8 -O legiscan_api legiscan_api
```

Finally load the schema and static lookup tables.

```
sudo -u postgres psql -U legiscan_api -f schema-pgsql.sql
```

Database Diagram

See: Entity Relationship Diagram (https://api.legiscan.com/dl/Database_ERD.png) for a schema overview and table relations.

Configuration

Copy the sample configuration file to its proper location.

```
cp config.dist.php config.php
```

Using your favorite text editor, open the `config.php` file and follow the instructions included in the file for setting up configuration options.

```
vim config.php
```

Of particular note enter your `api_key` and change `dsn`, `db_pass` to the value set in the above step.

If setting up for pull, `update_type` and the `states` list and/or `searches` lists should be populated. You may also want to tune the global `relevance` cutoff for searches.

Local document storage is disabled by default, change `want_bill_text`, `want_amendment`, `want_supplement` if needed.

Directories and Permissions

In general the LegiScan Client expects to be ran under a single non-privledged account, often `www-data` or similar account when running from a web server, or perhaps a custom user/group. Whatever the case, mixing users later on will result in permission conflicts within cache.

There are several directories specified in `config.php` that need to be writable by the user running the scripts. By default these are located within the `legiscan` directory, if you select alternate locations please ensure proper permissions and access.

```
chown -R www-data cache/api cache/doc log signal
```

NOTE: Be consistent when running from command line e.g., `sudo -u www-data php legiscan-cli.php` .

NOTE: If `middleware_signal` is not used or is table based, the `signal` directory can be ignored.

Bulk Import

Starting here is **STRONGLY** encouraged as a few hundred files are the equivalent of approximately **2 million** individual API calls.

Using the getDataset API hooks, a manually downloaded copy of a weekly Public Dataset (https://legiscan.com/datasets), or a custom subscriber onboarding snapshot, this script will extract the contents of the archives and import/update data as needed.

When using getDataset, the script can operate in two modes, `--scan` and `--bulk` , both of which import new and updated datasets.

The `--scan` mode acts like a interactive search / browser with command line filters to import eligible datasets. While in `--bulk` mode, the `states[]` and `years[]` variables in `config.php` will be used to select which datasets will be synchronized. The latter of which is most appropriate for scheduling automated updates.

Show the command help

```
php legiscan-bulk.php --help
```

Use `config.php` settings to import datasets and answer yes to any prompts.

```
php legiscan-bulk.php --bulk --import --yes
```

Show a listing of all available datasets in California and their status

```
php legiscan-bulk.php --scan --state CA
```

Import all new/changed datasets in 2020 with verbose output

```
php legiscan-bulk.php --scan --year 2020 --import --verbose
```

Import all new/changed 2016 special sessions in North Carolina

```
php legiscan-bulk.php --scan --state NC --special --year 2016 --import
```

Process `example.zip` but do not import, only show what would have been done

```
php legiscan-bulk.php --file example.zip --dry-run --debug
```

NOTE: This will not pull local copies of documents unless they are included in the archive, though appropriate stub records will be created.

Pull Daemon

The `legiscand.php` script provides a daemon process that can use four different methods of keeping a local database synchronized via LegiScan Pull API.

Monitor

This mode will use the `ls_monitor` table to keep a specific `bill_id` list updated. This list can be managed with `legiscan-cli.php` , though determining the `bill_id` is an exercise for the reader and would require additional scripting, though a likely source would be through the search engine.

```
php legiscan-cli.php --monitor 823882
php legiscan-cli.php --unmonitor 823882
```

State

This mode will synchronize the entire master list from one or more states. To set the state list edit `config.php` and add each state to the `states[]` setting.

For example to track all legislation in California and US Congress:

```
states[] = CA
states[] = US
```

NOTE: We *HIGHLY* recommend pre-loading the current state dataset with `legiscan-bulk.php` prior to the first run to minimize the on-boarding queries.

Search (National)

This mode will synchronize the results of searches ran against the national database. To specify the searches edit the `config.php` and add each search to the `searches[]` setting.

The searches will also be filtered by the global `relevance` cutoff setting, which can be overridden on a per search basis by prepending a different score and the pipe `|` character. In addition a `state` abbreviations can also be prefixed to override either national or state search. When used with a `relevance` override the `state` should appear first separated by a comma `,`.

Also notice that the entire search string should be quoted, and any internal quotes should be escaped as `\"`.

```
searches[] = "gender AND bathroom"
searches[] = "\"national popular vote\""
searches[] = "42|hemp OR cannabis OR marijuana"
searches[] = "NY|charter ADJ schools"
searches[] = "CA,60|vaccination AND status:passed"
```

State Search

This mode combines both of the other methods such that the `searches[]` are only ran against the `states[]` list, unless a search specific `state` is used.

Push Endpoint

The `legiscan-push.php` script serves as the endpoint listener for LegiScan Push API services. This will receive payloads from the LegiScan Push server and process them into the provided database. The LegiScan API Client should be installed in a location that is accessible by your web server and externally by LegiScan servers.

For additional authentication you may also set `api_auth_key` which can be found in the LegiScan API control panel at legiscan.com (https://legiscan.com/legiscan). This will validate against the HTTP Authorization header sent with incoming payloads and reject those with invalid tokens.

And if your API key is set to receive cooked application/x-www-form-urlencoded payloads, be sure to change the `push_form_var` setting to the form variable name specified in the API control panel.

NOTE: To utilize this script you will need a Push API subscription.

Command Line Interface

The CLI interface allow for importing specific objects from the command line along with manipulating some internal controls. This could be used for testing purposes or for automation with other custom scripting.

Show the command help

```
php legiscan-cli.php --help
```

Import the current GAITS monitoring list associated with the API key and sync with the local `ls_monitor` list

```
php legiscan-cli.php --monitorlist --import all --sync
```

Get the master list for the most recent session in California, but do not actually import and dump the response payload

```
php legiscan-cli.php --dry-run --debug --masterlist CA
```

Request a specific `bill_id` be imported locally

```
php legiscan-cli.php --bill 823201
```

Run a national search and import new bills with a relevance score of 75% or higher

```
php legiscan-cli.php --search "citizens ADJ united" --state ALL --import
new --score 75
```

Add a `bill_id` to the monitor list with support stance and synchronize with GAITS

```
php legiscan-cli.php --monitor 843038,908829,817390 --stance support --sy
nc
```

Add a `bill_id` to the ignore list

```
php legiscan-cli.php --ignore 898381
```

Clean the API cache of stale entries

```
php legiscan-cli.php --clean --verbose
```

Web Interface

The web interface is primarily for demonstration purposes to generate single API requests to examine payload structure. The LegiScan API client and `legiscan-ui.php` should be accessible by your web server. Then point your brower to the URL of interface script (e.g., http://example.com/legiscan/legiscan-ui.php).

Enter your API key in the space provided, select the type of payload request you would like and enter the appropriate ID, then click `Run Test` .

The system will then output the decoded payload response.

Middleware Signaling

The LegiScan API Client supports signaling a third party application of bill and related data changes so that additional processing can be done. The use case for this would be when the `legiscan_api` database is used as staging for another custom production system already in place.

This is controlled by the `middleware_signal` setting and can be either `table` or `directory` based.

Table Signaling

When a bill first appears or later updated, or a new text, amendment, supplement, roll call or person record is created a record will be made in the `ls_signal` table within the `legiscan_api` database.

This record will contain the `object_type` , its internal `object_id` and a `processed` flag.

The third party application should scan the table for `processed = 0` , take any necessary actions, then update the record with `processed = 1` .

Directory Signaling

When a bill first appears or later updated, or a new text, amendment, supplement, roll call or person record is created a file will be created in the `signal` directory.

This file will be named in the form of `object_type` . `object_id` and will contain a JSON object containing the same field along with the `updated` time stamp.

The third party application should scan the directory for files, take any necessary actions, then delete the file from the `signal` directory.

Configure init.d and cron job

To install the LegiScan Pull Daemon so that it runs on startup and can be controlled by `service`. This file presumes the install location is `/opt/legiscan`, if this is not the case edit the `legiscan.service` file and change `LEGISCAN_PATH`.

```
cp legiscan.service /etc/init.d/legiscan
chmod +x /etc/init.d/legiscan
update-rc.d legiscan defaults
service legiscan start
```

For scheduled bulk updates, the following will add a weekly cron job to pull dataset updates as needed.

```
tee /etc/cron.weekly/legiscan-bulk <<EOF
#!/bin/sh
php -q /opt/legiscan/legiscan-bulk.php --bulk --import --yes
EOF
chmod +x /etc/cron.weekly/legiscan-bulk
```

If running via the Pull interface, the API cache directory will grow over time. This will create an hourly cron job to clean stale entries.

```
tee /etc/cron.hourly/legiscan-clean <<EOF
#!/bin/sh
php -q /opt/legiscan/legiscan-cli.php --clean
EOF
chmod +x /etc/cron.hourly/legiscan-clean
```