pcfinancial

Warren Wilkinson

January 9, 2013

Contents

Overview

The pcfinancial package uses Drakma to download CSV data from the PC Financial website.

```
(setf *unparsed-csv* (pcf:fetch-and-logout "your-account" "date-or-nil" "your-password" "your-card-hash"))
```

-or-

```
(pcf:login "your-password" "cardNumber-from-your-cookie")
(setf *unparsed-csv* (pcf:download-transactions "your-account" (encode-universal-time 0 0 0 day month year
(pcf:logoff)
```

You need 3 magic values to make it work. You can get most of them through creative web browsing.

- Your Password in plain text (the transfer is done using SSH though, so it's secure I guess).
- cardNumber as found in a cookie from your webbrowser. It's a hashed value (I think). If you have many cards, get a unique cookie for each one by using incognito browser mode and PCFinancials 'add card' option on the login page.
- your-account is a string in the form "##,#####,##############". Log in with a web browser and inspect the 'Account' select options on the download page (https://www.txn.banking.pcfinancial.ca/a/banking/accounts/downloadTransactions1.ams).

Features

- Login, Logoff and download transactions.
- Omitting the 'since' date from download transactions will cause it to download all since last download.

Limitations

- It doesn't parse the CSV.
- Can't help you find the magic numbers.
- It's a scrapper, so if the PCfinancial site changes, this will break..
- There are no tests.

Installation

Quick Lisp

Install Quick Lisp and then run:

```
(ql:quickload 'cl-pcfinancial)
```

If you have problems, see the support section, and you may want to run the tests.

Gentoo

As root,

```
emerge cl-pcfinancial
```

Once the emerge is finished, the package can be loaded using ASDF:

```
(asdf:operate 'asdf:load-op :cl-pcfinancial)
```

If you have problems, see the support section.

Ubunto

```
sudo apt-get install cl-pcfinancial
```

Once the installation is finished, the package is loadable using ASDF:

```
(asdf:operate 'asdf:load-op :cl-pcfinancial)
```

If you have problems, see the support section.

Manual Installation

In summary: Untar the .tar package and then symlink the .asd files into a place where ASDF can find them.

1. Untar the files where you want them to be. On windows download the .zip and unzip it instead, it's the same files.
2. ASDF could be looking anywhere – it depends on your setup. Run this in your lisp repl to get a clue as to where ASDF is seeking libraries[1]:

   ```
   (mapcan #'funcall asdf:*default-source-registries*)
   ```

3. Symlink the .asd files to the source directory. If you use windows, these instructions on symlink alternatives apply to you.

[1] you might need to (require 'asdf) before running this example

Once the files are in place, the package can be loaded with ASDF by:

```
(asdf:operate 'asdf:load-op :cl-pcfinancial)
```

If you have problems, see the support section.

Getting Support

You can find support on this libraries website and/or github repository. Or you can email Warren Wilkinson.

Implementation

This library is a thin layer on top of Drakma. It provides nearly no features, and that is my intention. Scrapping is volatile because it relies on unreliable things.

1. The source website
2. Your internet connection

3. Magic numbers, cookies, etc.

This tiny library may break occasionally, but we’ll know the error is somewhere in it’s 85 loc. If it had features, debugging would be hard: is it broken or has the underlying website changed?

Logging In

A cookie-jar is created/bound and it has your cardNumber. Then we log in with your password and cardNumber 0.

```
(defun login (password cardNumber)
  (if *cookie-jar*
      (error "Already logged in...")
      (setf *cookie-jar*
            (make-instance ’drakma:cookie-jar
                           :cookies (list
                                     (make-instance ’drakma:cookie
                                                    :name "cardNumber"
                                                    :value cardNumber
                                                    :domain "www.txn.banking.pcfinancial.ca")))))
  (submit-to "https://www.txn.banking.pcfinancial.ca/a/authentication/signOn.ams"
             (cons "cardNumberSaved" "0")
             (cons "password" password)))
```

Downloading Transactions

Once logged in, we can just post the required variables!

```
(defun download-transactions (account &optional since)
  (multiple-value-bind (sec min hour day month year) (decode-universal-time (or since (get-universal-time)
    (declare (ignore sec min hour))
    (setf month (princ-to-string (decf month))
          year (princ-to-string year)
          day (princ-to-string day))
    (submit-to "https://www.txn.banking.pcfinancial.ca/a/banking/accounts/downloadTransactions2.ams"
               (cons "fromAccount" account)
               (cons "sinceLastDownload" (if since "false" "true"))
               (cons "previewDownload" "false")
               (cons "pfmSoftware" "other")
               (cons "fromDate__YEAR" year)
               (cons "fromDate__MONTH" month)
               (cons "fromDate__DAY" day))))
```

Logging Out

Go to the signOff page and set our cookie-jar to nil.

```
(defun logout ()
  (multiple-value-prog1 (open-url "https://www.txn.banking.pcfinancial.ca/a/authentication/signOff.ams")
    (setf *cookie-jar* nil)))
```

Other Utilities

For reading the CSV data, I recommend read-csvfn: A lisp library for parsing CSV, also written by me..

Within this package are some utilities to split PC Finance's date format (mm/dd/yyyy) into Y, M, D.

```
(defun string-date-to-ymd (date)
  (values (parse-integer (subseq date 6 10))
          (parse-integer (subseq date 0 2))
          (parse-integer (subseq date 3 5))))

(defun string-date-to-universal-time (date)
  (multiple-value-bind (y m d) (pc-financial-date-to-ymd date)
    (encode-universal-time 0 0 0 d m y)))
```

The fetch-and-logout function will fetch the transactions and logout safely.

```
(defun fetch-and-logout (account date password cardhash)
  (login password cardhash)
  (unwind-protect (download-transactions account date)
    (logout)))
```

License

cl-pcfinancial is distributed under the MIT license.