What does LOVEFiLM do?

- DVDs & games in the post
- Video on demand via PC, Sony & Samsung TVs
- Other devices coming soon...
- Use their website to view film reviews, create a rental list, play trailers, play films etc etc
API

- Exposes lots of the functionality
- Can write apps to manage your rental list
- Search for films, actors, directors...
- Can serve up images for films, links to play trailers, synopsis of each film links for each actor etc etc
WWW::Lovefilm::API

- I created it to help me learn the API
- I wanted to write apps but before I could do that I needed some lower level code
- Based on another module in CPAN... thanks to David Westbrook for his blessing
OAuth

- Open standard
- Used by Twitter, Myspace, Yahoo, Google etc
- Open protocol to allow secure API authorisation
- Provides 2 levels of access
- A website can be given permission to access a user's account on the LOVEFiLM site without being given the user's password
2 Legged

- Oauth 2 legged is the simplest and most common
- Also called ”tokenless”
- No access to users data
- Quite a lot of the calls do not require anything more than signed requests
Signing

- Send your application key
- Generate a signature using the shared secret
- No need for HTTPS

GET /catalog/title?expand=actors%2Csynopsis&term=batman
HTTP/1.1
Server: openapi.lovefilm.com
Authorization: Oauth
oauth_consumer_key="2blu7svnhkwkzw29zg7cwydn",
oauth_nonce="b2254b2bd8bf62423c73f1",
oauth_signature="h7aMKIgTuE3FvnUloNWNNuM42Gw%3D",
oauth_signature_method="HMAC-SHA1",
oauth_timestamp="1256822344",
oauth_version="1.0"
3 Legged

- More of a "dance"
- A user moves from your website/app to the LOVEFiLM web site and you are asked if you wish to give permission to your app.
- You grant permission
- You are redirected back to your app
- Your site can now control aspects of the users account
REST

- Representational State Transfer
- HTTP:
  - GET – list or retrieve
  - POST – create a new
  - PUT – replace or update
  - DELETE - delete
Examples

my %auth = Your::Custom::getAuthFromCache();

my $lovefilm = WWW::Lovefilm::API->new({
    consumer_key    => $auth{consumer_key},
    consumer_secret => $auth{consumer_secret},
    access_token    => $auth{access_token},
    access_secret   => $auth{access_secret},
    user_id         => $auth{user_id},

    # Optional
    content_filter => sub { decode_json(shift) },
});
use WWW::Lovefilm::API;
use XML::Simple;
my $lovefilm = WWW::Lovefilm::API->new({
    consumer_key
    consumer_secret
    content_filter => sub { XMLin(@_) },
});
$lovefilm->REST->Catalog->Title;
$lovefilm->Get( term => 'zzyzx' );
printf "%d Results.\n",
    $lovefilm->content->{total_results};
printf "Title: %s\n", @{$title}->{clean}
    for values %{
        $lovefilm->content->{catalog_title} };
# Basic info on this user
$lovefilm->REST->Users;
$lovefilm->Get() or die $lovefilm->content_error;
print Dumper $lovefilm->content;

# What discs do they have at home
$lovefilm->REST->Users->at_home();
$lovefilm->Get() or die $lovefilm->content_error;
print Dumper $lovefilm->content;
OAuth Limitations

- Your consumer key is embedded in the app, along with your secret
- Not a big deal but can easily dig them out of an app
- This is ok if it is website only.
- Knowing this does not get you free films :(
- OAuth v2 may well get around this problem
OAuth + REST

- Several sites use this (Open Social)
- Could take WWW::Lovefilm::API and call it WWW::OpenSocial::API
- Change a few lines
- It would just work
- Not much (at the moment) LOVEFiLM specific
Thanks

- LOVEFiLM for making their API open
- David Westbrook for letting me copy and modify his work
Links

- http://github.com/pmooney
- http://search.cpan.org/~pmooney/