

Code For Jason Gessner's [Perl Blogging Techniques](#) talk, September 15, 2003

Source code and other materials for the talk are available at
<http://www.multiply.org/notebook/> (search for talk).

Basic bloggerAPI post using Net::Blogger

(with a movable type engine, but that doesn't make much of a difference for this example).

```
#!/usr/bin/perl -w

use strict;

use Net::Blogger;

use Data::Dumper;

use Carp;

my $mt = Net::Blogger->new(
    engine => "movabletype"
);

# set our proxy (the URL of our XML-RPC server)
$mt->Proxy("http://www.multiply.org/mt/mt-xmlrpc.cgi");

# replace with your username and password
$mt->Username("jason gessner");
$mt->Password("*****");

# ask the user what blog to use
$mt->BlogId(&prompt_for_blog_id());

# ask the user for their post
my $post = &prompt_for_entry_text();

# ok, now we have all the info we need to post this text
my $post_id = $mt->newPost(
    postbody => \$post,
    publish => 0
) || croak $mt->LastError();

print "\nThe post id is $post_id\n";

sub prompt_for_blog_id() {
    # get a list of the blogs for the specified user/pass
    my $bloglist = $mt->getUsersBlogs()
    || croak $mt->LastError();

    # print out the details of each blog with a prompt
    print "Please choose which blog you would like to post to:\n";
    print "-----\n";

    # make a hash of our user's blogs so we can verify the input
    # without a loop
    my %blogs;

    # since this returns an array ref of hashes, we need
    # to dereference our bloglist before cycling through it.
    for my $blog (@{ $bloglist }) {
        print $blog->{ 'blogid' }, " ", $blog->{ 'blogName' }, "\n",
            "\t", $blog->{ 'url' }, "\n";
        # stash this id
        $blogs{ $blog->{ 'blogid' } }++;
    }

    my $blog_id;

    chomp ($blog_id = <STDIN>);
    while (! $blogs{ $blog_id }) {
        print "Please choose a valid blog id (see list above)\n";
    }
}
```

```

        chomp ($blog_id = <STDIN>);
    }

    return $blog_id;
}

sub prompt_for_entry_text() {
    print "Please enter the text for your post:\n";
    print "\t--- end your post with period alone on a line\n";

    my $input;

    while (<>) {
        chomp ($_);
        $input .= "$_\n";

        last if ($_ eq ".");
    }

    return $input;
}

```

Post using the metaWeblog extended API.

This allows us to specify the title of the post (among other things).

The only difference is in the post call. I also added a `prompt_for_entry_title` function.

```

# ok, now we have all the info we need to post this text
my $post_id = $mt->metaWeblog()->newPost(
    title => $title,
    description => $post,
    publish => 1
) || croak $mt->LastError();

```

Post using the Movable Type extended API.

Basically, this is used to specify a category (or multiple categories) for a post. There are also more maintenance and management functions, but I am pretty much just dealing with posting for this talk.

```

# ask the user for the category to assign this post to
my $category_id = &prompt_for_category_id();
# TODO: add ability to prompt for multiple categories
my %category_hash;
$category_hash{categoryId} = $category_id;
$category_hash{isPrimary} = 1;
my @category_list = (\%category_hash);

# ok, now we have all the info we need to post this text
my $post_id = $mt->metaWeblog()->newPost(
    title => $title,
    description => $post,
    publish => 0
) || croak $mt->LastError();

print "\nThe post id is $post_id\n";

# now assign the category for the post
$mt->mt()->setPostCategories(
    postid => $post_id,
    categories => \@category_list
) || croak $mt->LastError();

$mt->mt()->publishPost($post_id);

...

```

```

sub prompt_for_category_id() {
    # get a list of the blogs for the specified user/pass
    my $categorylist = $mt->mt()->getCategoryList()
    || croak $mt->LastError();

    # print out the list of categories with a prompt
    print "Please choose which category to assign this post to:\n";
    print "\t(simply hit ENTER for no category)\n";
    print "-----\n";

    # make a hash of our user's blogs so we can verify the input
    # without a loop
    my %categories;

    # since this returns an array ref of hashes, we need
    # to dereference our bloglist before cycling through it.
    for my $category (@{ $categorylist } ) {
        print $category->{ 'categoryId' }, "\t", $category->{ 'categoryName' },
"\n";
        # stash this id
        $categories{ $category->{ 'categoryId' } }++;
    }

    my $category_id;

    chomp ($category_id = <STDIN>);
    while (! $categories{$category_id} ) {
        print "Please choose a valid category id (see list above)\n";
        chomp ($category_id = <STDIN>);
    }

    return $category_id;
}

```

Post Using a WWW::Mechanize hack

(based on the example script that comes with the distro).

```

use WWW::Mechanize;

# a small rewrite of the WWW::Mechanize Movable Type hack example
# this example prompts for the title and the body text

my $mech = WWW::Mechanize->new();
my %entry;
$entry->{title} = &prompt_for_entry_title();
$entry->{btext} = &prompt_for_entry_text();
my $server = qq|http://www.multiply.org|;
my $start = qq|/mt/mt.cgi|;

$mech->get($server . $start);
$mech->field('username','command line jason');
$mech->field('password','****');
$mech->submit(); # to get login cookie
$mech->get(qq|$start?__mode=view&_type=entry&blog_id=2|); # adjust as needed
$mech->form('entry_form');
$mech->field('title',$entry->{title});
$mech->field('category_id',7); # adjust as needed
$mech->field('text',$entry->{btext});
$mech->field('status',2); # 2 = publish, or 1 = draft

print "Thanks.\n-----\n";
print "Submitting entry...\n";
$results = $mech->submit();

# this is where we have to veer away from the initial example a bit.
# if you have chosen to publish this entry AND you have pings, then you
# will see 2 javascript redirects. That is the stinky thing about MT.

if ($mech->success()) {
    # travel the rebuild redirect

```

```

$results = $mech->get($results->{_headers}->{location});

if ($results->{_content} =~ /<body onLoad="([^\"]+)/is) {
    # do we have an onload redirect? yes
    my $js = $1;
    $js =~ /\\[^\[+\]\]/;
    # this returns something like:
    #     window.location='/mt/mt.cgi?__mode=rebuild&blog_id=2&type=entry-
214&next=0&offset=&limit=&total_entries=&is_bm=&entry_id=214&is_new=1&old_status=0'

    print "Rebuilding...\n";
    $results = $mech->get($server.$1);
    $mech->submit();

    if ($mech->success()) {
        if ($results->{_content} =~ /<body onLoad="([^\"]+)/is) {

            # follow the ping redirect
            my $js = $1;
            $js =~ /\\[^\[+\]\]/;
            # this returns something like:
            #
            window.location='/mt/mt.cgi?__mode=ping&blog_id=2&entry_id=218&is_new=1&old_status=0&is_bm='

            print "Sending out pings...\n";
            $results = $mech->get($server.$1);
            $mech->submit();
        }
    }
}

```

Creating a sendmail alias to run a script at an email address

Add a line like the following to /etc/aliases and then run the newaliases script. Note that your working directory will NOT be local to that script. Make sure that your script is not relying on local paths.

```
kpix:          "|/var/vhosts/www.kpix.jp/email_scripts/mailParse.pl"
```

Braindead Simple Homegrown Posting via Email

```

#!/usr/local/bin/perl -w

use MIME::Parser;

### Create parser, and set some parsing options:
my $parser = new MIME::Parser;

# set output directory
my $output_dir = "/var/vhosts/www.multiply.org/mobilesketches";

### Alter its filer:
my $filer = MIME::Parser::FileInto->new($output_dir);
$filer->ignore_filename(1);
$filer->output_prefix(time());
$parser->filer($filer);

### Parse input:
my $entity = $parser->parse(\*STDIN) or die "parse failed\n";

```

More Complicated Homegrown Posting via Email

This script adds some protection in the form of a ban list (via email address) and a list of approved mail gateways. The script also checks for a valid combination of a text message and an attached jpeg.

```
#!/usr/local/bin/perl -w

use MIME::Parser;
use File::Copy;

# for parsing out the address in the From: header
use Mail::Address;

# our DB functions
use KPDB;

# chuck's thumbnail function
use ThumbnailGen;

# for generating temp file names
use Fcntl;
use POSIX qw(tmpnam);

# making sure our text stays all legit, like
use MIME::Words qw(:all);

# dump_entities skipped for space.

#-----
#
# main
#
my @files;

sub main {
    print STDERR "(reading from stdin)\n" if (-t STDIN);

    # Create a new MIME parser:
    my $parser = new MIME::Parser;

    # Create and set the output directory:
    (-d "/tmp/kpix-mimedump-tmp") or mkdir "/tmp/kpix-mimedump-tmp",0777 or die "mkdir: $!";
    (-w "/tmp/kpix-mimedump-tmp") or die "can't write to directory";

    # control our output format
    $parser->output_dir("/tmp/kpix-mimedump-tmp");
    $parser->filer->ignore_filename(1);
    $parser->output_prefix(time());

    # Read the MIME message:
    $entity = $parser->read(\*STDIN) or die "couldn't parse MIME stream";

    # check for from address
    my $from = $entity->head->get('from');
    chomp($from);

    my $from_address;
    my @addrs = Mail::Address->parse($from);
    foreach $addr (@addrs) {
        $from_address = $addr->user . "@" . $addr->host;
    }

    # add some error checking for email addresses

    # connect to our database
    KPDB->connect();

    if (!KPDB->is_banned($from_address)) {
        # valid sender
        print "Sender not banned!\n";

        # check for valid gateway
```

```

my $lineNum = 1;
foreach $line ($entity->head->get('received')) {
    if ($lineNum == 1) {
        $gateway .= $line;
    }
    $lineNum++;
}

# pull in the subject for later use
my $subject = decode_mimewords($entity->head->get('subject'));

print $subject, "\n";

chomp($gateway);
# print $gateway, "\n";

my ($ehelo,$validname,$validip) = parseReceived($gateway);

# print "$ehelo\t$t$validname\t$t$validip\n";

if (KPDB->event_check_gateway($validname)) {

    # add log4perl stuff
    # print "Gateway is valid.\n";
    my @newfiles = dump_entity($entity);
    # only accept messages that have a .txt part and a .jpg part
    if ($#newfiles == 1) {
        print "We have the proper number of files\n";

        my $pic = "";
        my $txt = "";
        foreach $file (@newfiles) {
            # print "\t$file\n";
            # determine what kind of file we have
            # copy files to the destination

            # read text file contents into var
            if ($file =~ m/txt$/i) {
                $txt = decode_mimewords(returnFileContents($file));
            } elsif ($file =~ m/jpg$/i) {
                $pic = $file;
            }
        } # foreach

        # insert record into DB & get the ID of the new record
        my $photo_id = KPDB->new_event_photo(1, $from_address, $subject, $txt, 1);

        # close up our database connection
        KPDB->disconnect();

        # copy picture over to the dest with the new ID
        $dest_base = "/var/vhosts/www.kpix.jp/eventphotos/";
        $dest = "$photo_id.jpg";
        $dest =~ s#(^.*\\)(##;
        # print "\t\t$dest\n";
        copy($pic, "$dest_base$dest");

        # generate our thumbnails
        ThumbnailGen->generateThumb("$dest_base$dest", .9, "$dest_base${photo_id}
_1.jpg");
        ThumbnailGen->generateThumb("$dest_base$dest", .4, "$dest_base${photo_id}
_2.jpg");
        ThumbnailGen->generateThumb("$dest_base$dest", .2, "$dest_base${photo_id}
_3.jpg");
        ThumbnailGen->generateThumb("$dest_base$dest", .1, "$dest_base${photo_id}
_4.jpg");

    } # proper number of files
} else {
    # add a listing to our new_gateway table
    KPDB->event_add_new_gateway($validname, $from_address);

    # invalid gateway, so exit
    print "Mail from unauthorized gateway.\n";
}

```

```

    } else {
        # invalid sender
        print $from, " is not allowed to send me mail.\n";
    }

    # clean up our mess
    $parser->filer->purge;
}

sub replace_filename() {
    # replace our base filename, but leave the extension intact
    my ($filename, $new_base) = @_;

    $filename =~ s/(.*)($$new_base$2/i;

    return $filename;
}

sub parseReceived() {
    my $line = shift;

    # "normal" -- from HELO (REAL [IP])
    if ($line =~ m/from\s+(\w\S+)\s*\((\S+)\s*\[(\d+\.\d+\.\d+\.\d+)\])/){
        ($helo,$validname,$validip) = ($1,$2, $3);
    }
    # can't reverse resolve -- from HELO ([IP])
    elsif ($line =~ m/from\s+(\w\S+)\s*\((\S+)\s*\[(\d+\.\d+\.\d+\.\d+)\])/){
        ($helo,$validname,$validip) = ($1,undef, $2);
    }
    # exim -- from [IP] (helo=[HELO IP])
    elsif ($line =~ m/from\s*\[(\d+\.\d+\.\d+\.\d+)\]\s*\((helo=\[(\d+\.\d+\.\d+\.\d+)\])/){
        ($validip,$helo,$validname) = ($1,$2, undef);
    }
    # Sun Internet Mail Server -- from [IP] by HELO
    elsif ($line =~ m/from\s*\[(\d+\.\d+\.\d+\.\d+)\]\s*by\s+(\S+)/){
        ($validip,$helo,$validname) = ($1,$2, undef);
    }
    # Microsoft SMTPSVC -- from HELO - (IP)
    elsif ($line =~ m/from\s+(\S+)\s+-\s+(\d+\.\d+\.\d+\.\d+)\s+/){
        ($helo,$validname,$validip) = ($1,$2, $3);
    }
    else { # punt!
        $helo = $validname = $validip = undef;
    }


    return ($helo,$validname,$validip);
}

```

Adding a script to your CVS repository

First, check out the CVSROOT and add a variation of the following to the logininfo file. STDIN is the summarized commit message and %{} is the module and filename. The first argument is the module name. ALL matches all modules, otherwise you can use a regexp to specify which module you want. Like the sendmail alias, your working directory will NOT be local to that script. Make sure that your script is not relying on local paths.

```
ALL /home/jason/cvs-logininfo-posttoMT.pl "[Multiply-Code CVS] %{}"
```



NOTEBOOK

a place to keep my mental garbage, a repository for ideas that should (or should not) be forgotten, a reminder to keep thinking and working. Yeah. Something like that.

LISTENING - FIRST ITUNES MUSIC STORE PURCHASE

I registered for the iTunes music store to enter their iPod a day drawing, but had no real intention of using it. Then I saw that there was a re-released 2 disc version of Sonicy Youth's Dirty. A couple of quick price checks later and i had purchased it from iTunes.

The files are the protected AAC files, but the sound quality is really nice and pretty much only listen to stuff on my computer or iPod anyway. Even through the stereo, I hook the iPod up to it.

I am worried at how easy it is to buy albums. Then i looked at the prices on some of them. \$ for some albums, \$13 for others. I thought there was supposed to be a savings. Oh wait. That is for the labels. The artists don't really make any substantial amount from this service, so I guess that is like the status quo.

Anyway, it is a great SY album. ;)

Posted by command line jason, on September 13, 2003 at 09:50 PM | Comments (0) | TrackBack (0)

LOOKING - MATCHSTICK MEN

bill, tracey and I went to see Matchstick Men (salon review, ebert review) last night. It was not great, primarily due to the ending, but it had a great performance by Nicolas Cage (who I loved in Adaptation, as well). The movie reminded me of Peter Bogdonavich's Paper Moon, but it had less heart. The ending assumes the audience is too dumb to realize what just happened over the last 2 hours.

Posted by instant messaging jason, on September 13, 2003 at 05:49 PM | Comments (0) | TrackBack (0)

CODE - APOLOGIES AND ANNOUNCEMENTS

Thanks for tolerating a few weeks of random, incoherent posts (most of which have been deleted). In the process of wrapping up the code for my perl blogging techniques talk, i made lots of test posts which should have simply gone to a test blog. Ha! I am not that smart. ;)

Anyway, new content to expect here soon:

- Zelda: The Wind Waker extensive review
- Golden Sun: The Lost Age review
- Big Questions review
- Local gallery observations
- Photo galleries

Also, in an effort to organize my coding projects, I have set up changeblog. Changeblog will detail what I am coding. Exciting, eh? Anyway, check it out, hopefully it will be chock full of interesting goodies.

Posted by jason gessner, on September 11, 2003 at 10:28 PM | Comments (0) | TrackBack (0)

SEARCH

Search this site:

ITUNES, THEREFORE, I AM

Sonic Youth - Brother James
Sonic Youth - New White Kross (Rehearsal Tapes Version)
Sonic Youth - Hendrix Necro
Sonic Youth - Swimsuit Issue
Sonic Youth - Expressway to Yr. Skull
Sonic Youth - Shadow of a Doubt
Sonic Youth - Genetic
Sonic Youth - Trilogy
Sonic Youth - Kissability
Sonic Youth - Rain King

ARCHIVES

September 2003
August 2003
July 2003
June 2003
May 2003
April 2003
March 2003
February 2003

RECENT ENTRIES

First iTunes Music Store Purchase
matchstick men
Apologies and announcements
[SushiBubblegum CVS] CVSROOT
[SushiBubblegum CVS] chatmonkey
asdf
A few regexps later and
Hi!This is "Instant Messaging Jason"
Well, I have added a
Basic Scripts completed for Perl Blogging Techniques

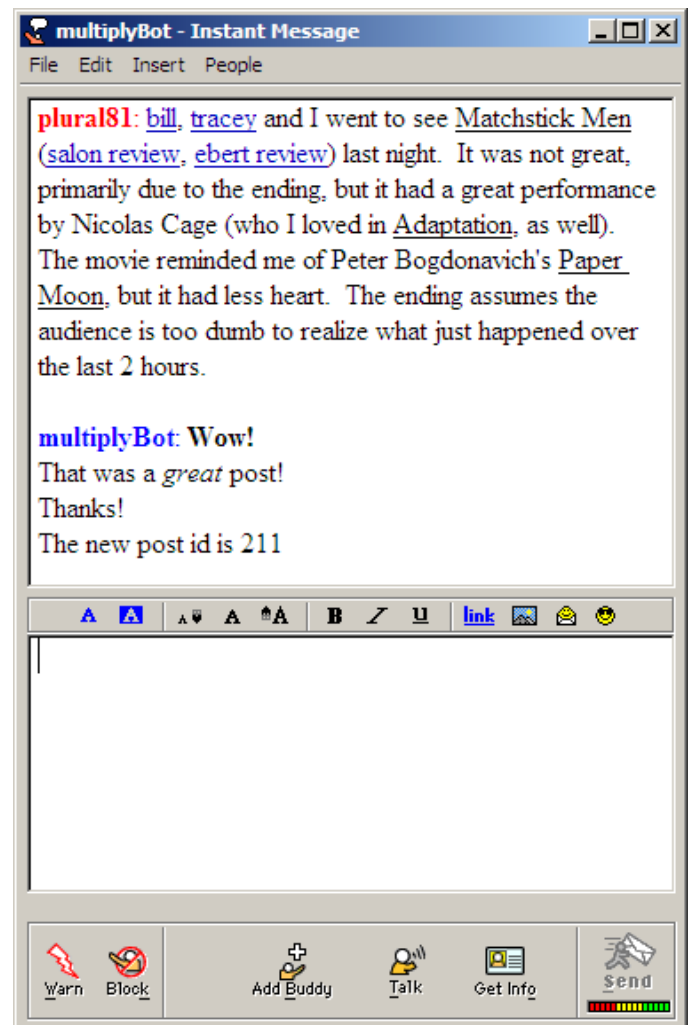
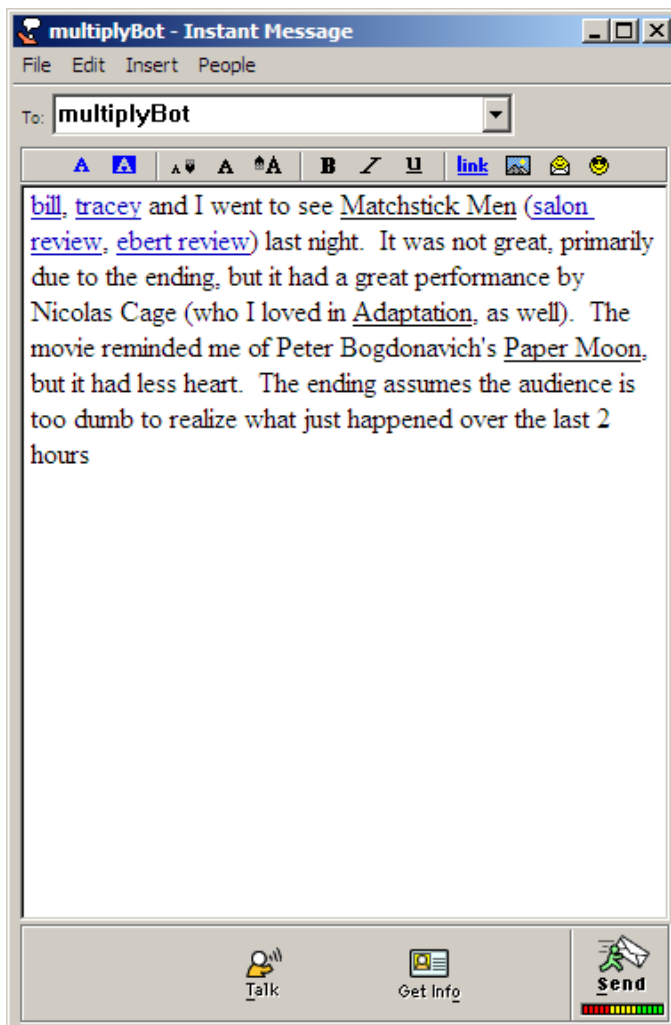
RECENT READING

The Pentagon Papers as published by the New York times by Neil Sheehan
The Korean War by William Stueck
Free as in Freedom: Richard Stallman's Crusade for Free Software by Sam Williams
V. by Thomas Pynchon
The Transmigration of Timothy Archer by Philip K. Dick
Divine Invasion by Philip K. Dick
Coraline by Neil Gaiman, Dave McKean
Valis by Philip K. Dick
Alec: After The Snooter by Eddie Campbell, Eddie Campbell

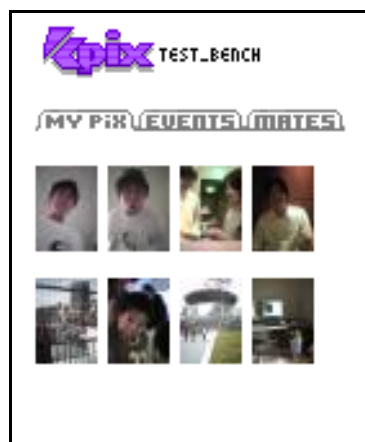
RECENT LINKS

Brain Cloud
tracey's space
Salon.com Arts & Entertainment | "Matchstick Men"
MATCHSTICK MEN / **** (PG-13)
changeblog
POE: Perl Object Environment
Letter App
search.cpan.org: WWW::Mechanize::Examples - Sample

My blog home page. A typical blog structure has entries organized by date and/or categories, plus one or many side bars containing links, lists or other fun things. I have a script that posts a trackback to a certain category each time iTunes switches tracks on my G4 (iTunes, Therefore, I Am).



Posting to a Movable Type blog via AOL Instant Messenger. This would be possible using IRC, Jabber or any other IM service that you could run a bot on.



Screenshots from the development version of Kpix. Kpix is/will be a moblogging service used for personal galleries and interactive events, primarily in Japan. The site is a home-made blog, using several of the same ideas as a normal blog, and featuring an email gateway for use from camera-enabled cell phones. The site also has a version of the site that is viewable from an internet-enabled phone.



guy reading newspaper
next to me



juliana puts stickers on her
legs for fun



plane out the window



flight overbooked & 2
hours late (part 1)



waiting for the west line



i talked to a guy on the
train this morning with a
sony-ericsson p800.
Yummy



cigar & bourbon on the
rocks



chicken & ka bobs



bbg in the rain

Screenshot of my Sketch Blog (SKOG!). This is a collection of drawings done on my palm based cell phone and emailed to a perl program waiting as a sendmail alias. This example is very similar to the Kpix shots, but for a single user.



September 13, 2003

[MULTIPLY-CODE CVS] PERL_BLOGGING_TECHNIQUES/IMAGES

Update of /usr/local/multiply-code/perl_blogging_techniques/images
In directory tetsuo.mengelt.com:/tmp/cvs-serv24046/images

Added Files:

AIMblogbot-Results.png AIMblogbot1.png AIMblogbot2.png
kpix_001.png kpix_002.png kpix_003.png

Log Message:

added screenshots for the examples.
The shots added were from the instant messenger bot post and some kpix screens.

Posted by command line jason at 07:10 PM | [Comments \(0\)](#)

[MULTIPLY-CODE CVS] PERL_BLOGGING_TECHNIQUES/IMAGES

Update of /usr/local/multiply-code/perl_blogging_techniques/images
In directory tetsuo.mengelt.com:/tmp/cvs-serv24017/images

Log Message:

Directory /usr/local/multiply-code/perl_blogging_techniques/images added to the repository

Posted by command line jason at 07:09 PM | [Comments \(0\)](#)

September 11, 2003

[MULTIPLY-CODE CVS] PERL_BLOGGING_TECHNIQUES

Update of /usr/local/multiply-code/perl_blogging_techniques
In directory tetsuo.mengelt.com:/tmp/cvs-serv5199

Added Files:

presentation.sxw

Log Message:

The presentation outline and speaking notes for the talk.

Posted by command line jason at 11:16 PM | [Comments \(0\)](#)

[MULTIPLY-CODE CVS] PERL_BLOGGING_TECHNIQUES

Update of /usr/local/multiply-code/perl_blogging_techniques
In directory tetsuo.mengelt.com:/tmp/cvs-serv5154

Log Message:

Import of the current code for the Perl Blogging Techniques presentation.

Status:

Vendor Tag: multiply

Release Tags: multiply-import

N perl_blogging_techniques/code/AIM-stripHTML.pl
N perl_blogging_techniques/code/ CVS-logininfo_test.pl
N perl_blogging_techniques/code/AIMblogBot.pl
N perl_blogging_techniques/code/bloggerAPI.pl

September 2003

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

Search

Search this site:

Search

Archives

September 2003

Recent Entries

[Multiply-Code CVS] perl_blogging_techniques/images
[Multiply-Code CVS] perl_blogging_techniques/images
[Multiply-Code CVS] perl_blogging_techniques
[Multiply-Code CVS] perl_blogging_techniques
[SushiBubblegum CVS] chatmonkey
Welcome to changeblog!

Links

Add Your Links Here

Syndicate this site (XML)

POWERED BY
MOVABLE TYPE 2.6.4

Screenshot of my "ChangeBlog" - a blog that is tied to a CVS repository (or multiple, in this case). This could easily be extended to suit the specific needs of a workgroup. In the case of a Movable Type based blog, plugins could alter the entries to provide links to a web based CVS tree viewer, show graphs of commits on any given day (using something like the blogtimes plugin), or any number of things.

Links for Jason Gessner's Perl Blogging Techniques talk, September 15, 2003

- Blog Software
 - Hosted Services
 - Blogger
 - <http://www.blogger.com/>
 - LiveJournal
 - <http://www.livejournal.com>
 - TypePad
 - <http://www.typepad.com/>
 - use.perl.org journals
 - <http://use.perl.org/>
 - AOL Journals (aol subscribers only).
 - <http://hometown.aol.com/>
 - Radio Userland
 - <http://www.userland.com/>
- Services that can be installed on your own server
 - Bloxom
 - <http://www.bloxom.com/>
 - Movable Type
 - <http://www.movabletype.org>
 - GreyMatter
 - <http://www.noahgrey.com/greysoft/>
 - Slash
 - <http://slashcode.com/>
 - Wordpress
 - <http://www.wordpress.org>
- Articles/Info on coding for blogs
 - Net::Blogger documentation
 - <http://search.cpan.org/author/ASCOPE/Net-Blogger-0.86/lib/Net/Blogger.pm>
 - Developing Movable Type Plug-ins, by Timothy Appnel
 - <http://www.oreillynet.com/pub/a/javascript/2003/03/18/movabletype.html>
 - Purple Numbers integration for Movable Type
 - <http://www.burningchrome.com:8000/~cdent/mt/archives/000034.html>
 - MT Plugin Directory
 - <http://www.mt-plugins.org/>
 - Bloxom Plugin Directory
 - <http://www.bloxom.com/plugins/>
 - Bloxom Plugin details
 - <http://www.bloxom.com/documentation/users/plugins.html>
 - Movable Type external API documentation
 - This is a great intro to the differences in the APIs.
 - http://www.movabletype.org/docs/mtmanual_programmatic.html#programmatic%20interfaces