Gathering Information from the Web with WWW::Mechanize

Tom Hukins

Screen Scraping

Web Services Where None Exist

WWW::Mechanize

- LWP
- HTML::Parser

LWP: Retrieve a Page

```
use LWP::Simple;
my $doc = get 'http://london.pm.org/lpw/signup.html';
defined($doc) or die 'Request Failed';
```

LWP: Retrieve a Page

```
use LWP::Simple;
my $doc = get 'http://london.pm.org/lpw/signup.html';
defined($doc) or die 'Request Failed';
```

LWP: Retrieve a Page

```
use LWP::Simple;
my $doc = get 'http://london.pm.org/lpw/signup.html';
defined($doc) or die 'Request Failed';
```

LWP: Retrieve a Page

```
use LWP::Simple;
my $doc = get 'http://london.pm.org/lpw/signup.html';
defined($doc) or die 'Request Failed';
```

LWP: Submit a Form

```
use HTTP::Request::Common qw(POST);
use LWP::UserAgent;

$ua = LWP::UserAgent->new;

my $req = POST 'http://london.pm.org/cgi-bin/signup.cgi',
    [
        name         => 'Guy Incognito',
        email        => 'guy@annoying.example.com',
        submit       => 'signup',
    ];

my $response = $ua->request($req);
```

LWP: Submit a Form

```
use HTTP::Request::Common qw(POST);
use LWP::UserAgent;

$ua = LWP::UserAgent->new;

my $req = POST 'http://london.pm.org/cgi-bin/signup.cgi',
    [
        name        => 'Guy Incognito',
        email       => 'guy@annoying.example.com',
        submit      => 'signup',
    ];

my $response = $ua->request($req);
```

LWP: Submit a Form

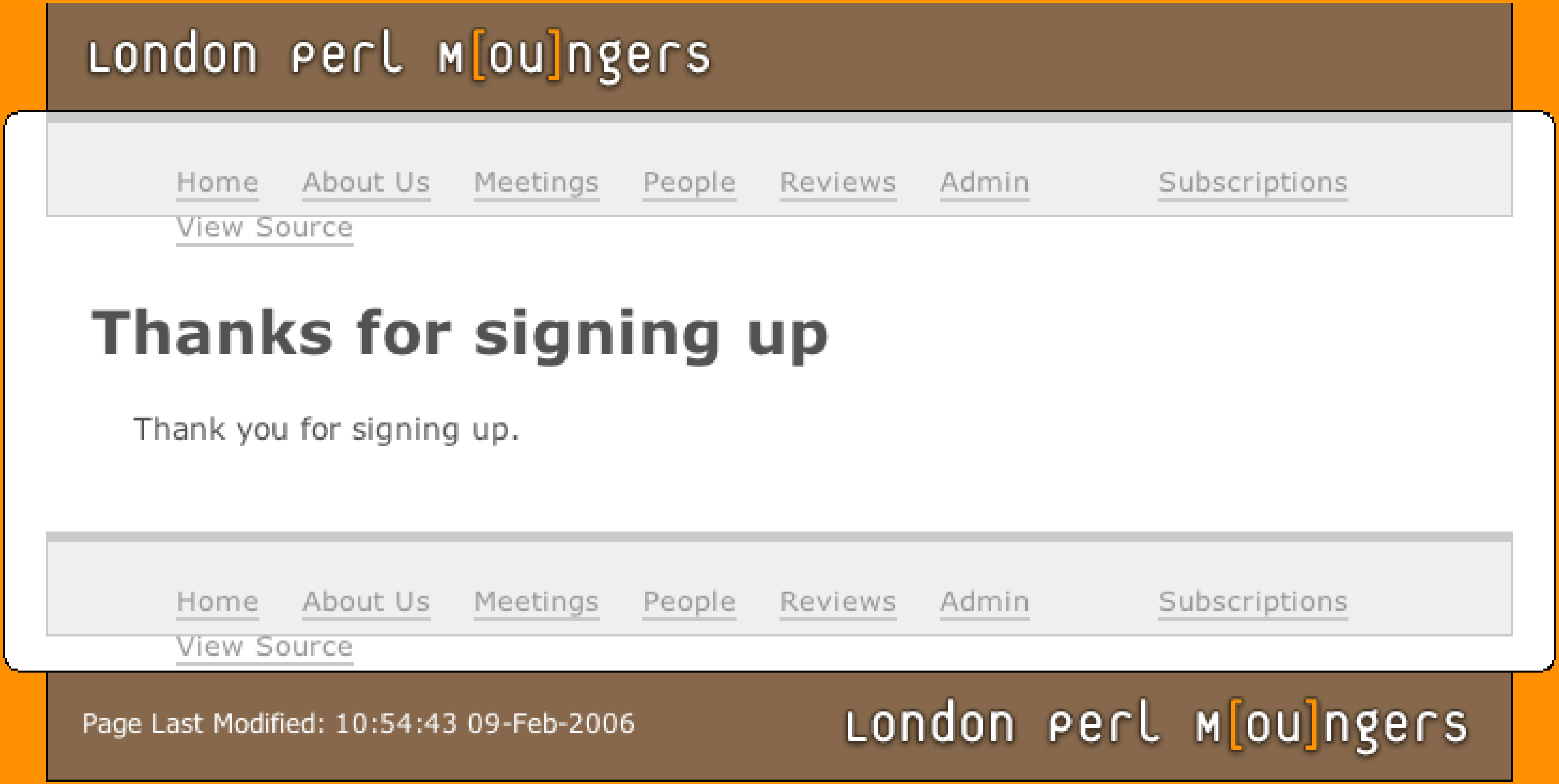
```
use HTTP::Request::Common qw(POST);
use LWP::UserAgent;

$ua = LWP::UserAgent->new;

my $req = POST 'http://london.pm.org/cgi-bin/signup.cgi',
    [
        name        => 'Guy Incognito',
        email       => 'guy@annoying.example.com',
        submit      => 'signup',
    ];

my $response = $ua->request($req);
```

LWP: Submit a Form

```
use HTTP::Request::Common qw(POST);
use LWP::UserAgent;

$ua = LWP::UserAgent->new;

my $req = POST 'http://london.pm.org/cgi-bin/signup.cgi',
    [
        name         => 'Guy Incognito',
        email        => 'guy@annoying.example.com',
        submit       => 'signup',
    ];

my $response = $ua->request($req);
```

LWP: Submit a Form

```
use HTTP::Request::Common qw(POST);
use LWP::UserAgent;

$ua = LWP::UserAgent->new;

my $req = POST 'http://london.pm.org/cgi-bin/signup.cgi',
    [
        name        => 'Guy Incognito',
        email       => 'guy@annoying.example.com',
        submit      => 'signup',
    ];

my $response = $ua->request($req);
```



london perl m[ou]ngers

Home About Us Meetings People Reviews Admin Subscriptions View Source

Thanks for signing up

Thank you for signing up.

Home About Us Meetings People Reviews Admin Subscriptions View Source

Page Last Modified: 10:54:43 09-Feb-2006

london perl m[ou]ngers

HTML::Parser

```
use HTML::TokeParser;
my $p = HTML::TokeParser->new('bad.html');

while (my $token = $p->get_tag("a")) {
    my $url = $token->[1]{href};
    print "$url\n";
}
```

HTML::Parser

```
use HTML::TokeParser;
my $p = HTML::TokeParser->new('bad.html');

while (my $token = $p->get_tag("a")) {
    my $url = $token->[1]{href};
    print "$url\n";
}
```

HTML::Parser

```
use HTML::TokeParser;
my $p = HTML::TokeParser->new('bad.html');

while (my $token = $p->get_tag("a")) {
    my $url = $token->[1]{href};
    print "$url\n";
}
```

HTML::Parser

```
use HTML::TokeParser;
my $p = HTML::TokeParser->new('bad.html');

while (my $token = $p->get_tag("a")) {
    my $url = $token->[1]{href};
    print "$url\n";
}
```

See Also
HTML::TokeParser::Simple

Parsing Bad HTML

```
<html>
  <title>A title belongs in the <head></title>
  <BODY>
    Here is <a href="some" title="example">some</a>
    <a href="other" title="example2">text</a>.
    <p>
    The end.
  </body>
<html>
```

Parsing Bad HTML

```
<html>
  <title>A title belongs in the <head></title>
  <BODY>
    Here is <a href="some" title="example">some</a>
    <a href="other" title="example2">text</a>.
    <p>
    The end.
  </body>
<html>
```

Parsing Bad HTML

```
<html>
  <title>A title belongs in the <head></title>
  <BODY>
    Here is <a href="some" title="example">some</a>
    <a href="other" title="example2">text</a>.
    <p>
    The end.
  </body>
<html>
```

Parsing Bad HTML

```
<html>
  <title>A title belongs in the <head></title>
  <BODY>
    Here is <a href="some" title="example">some</a>
    <a href="other" title="example2">text</a>.
    <p>
    The end.
  </body>
<html>
```

Parsing Bad HTML

```html
<html>
  <title>A title belongs in the <head></title>
  <BODY>
    Here is <a href="some" title="example">some</a>
    <a href="other" title="example2">text</a>.
    <p>
    The end.
  </body>
<html>
```

WWW::Mechanize

Acts like a browser:

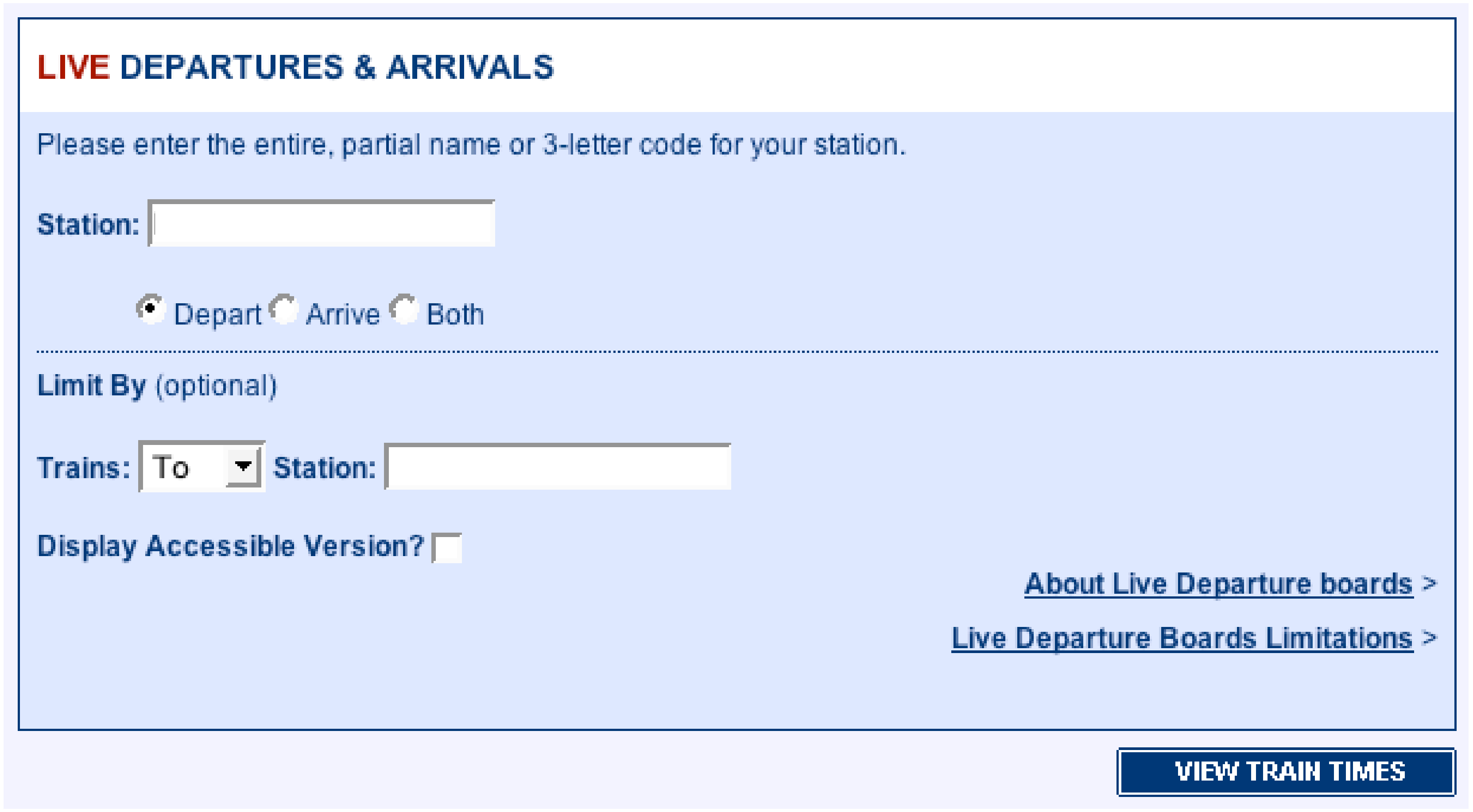
- Hidden Form Fields
- Cookies

WWW::Mechanize

Not like a browser:

- JavaScript
- Flash
- Java Applets

Analysing Web Forms


LIVE DEPARTURES & ARRIVALS

Please enter the entire, partial name or 3-letter code for your station.

Station:

[✓ Depart] Arrive Both

Limit By (optional)

Trains: To Station:

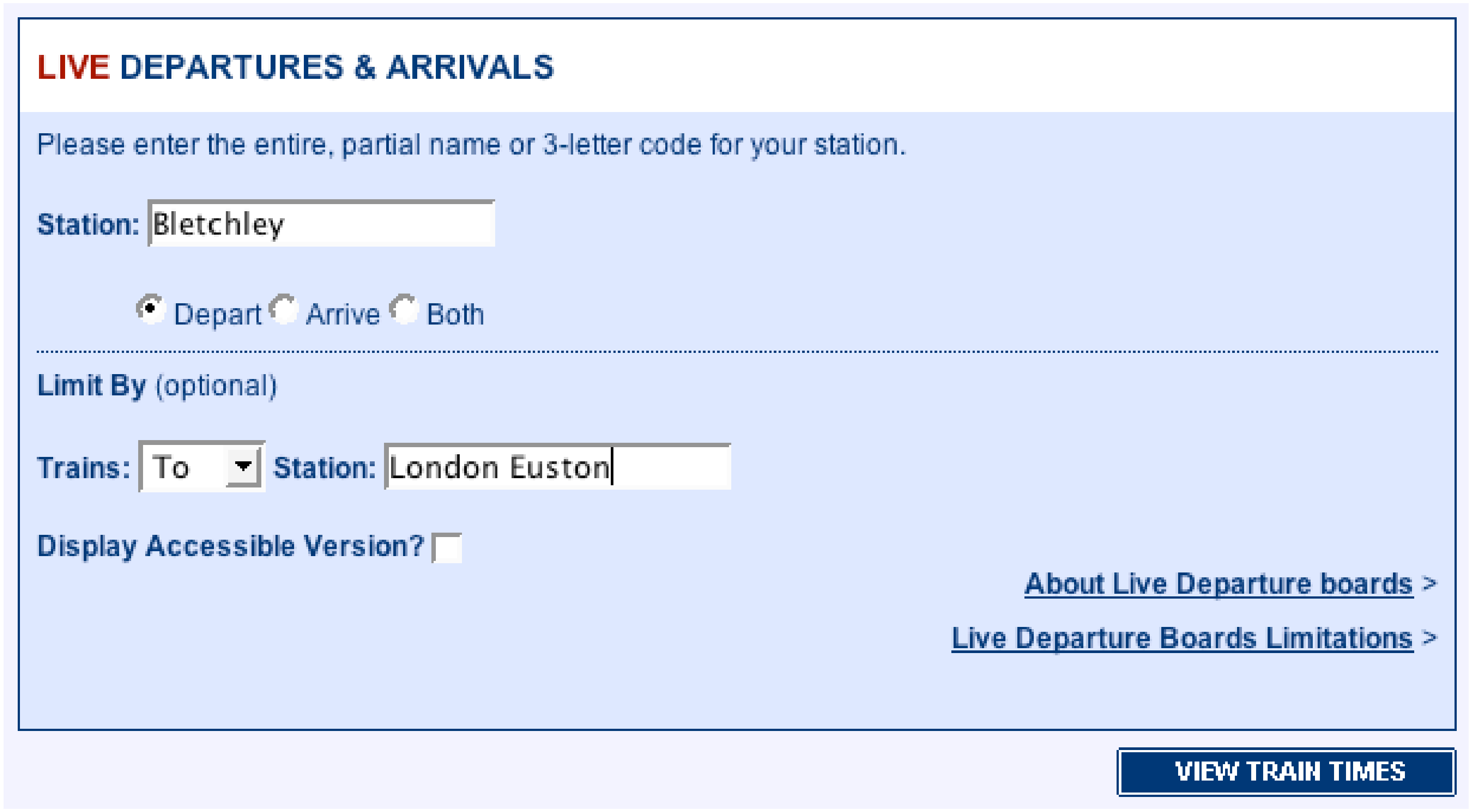
Display Accessible Version?

About Live Departure boards >

Live Departure Boards Limitations >

VIEW TRAIN TIMES

Analysing Web Forms



LIVE DEPARTURES & ARRIVALS

Please enter the entire, partial name or 3-letter code for your station.

Station: Bletchley

[✓ Depart] Arrive Both

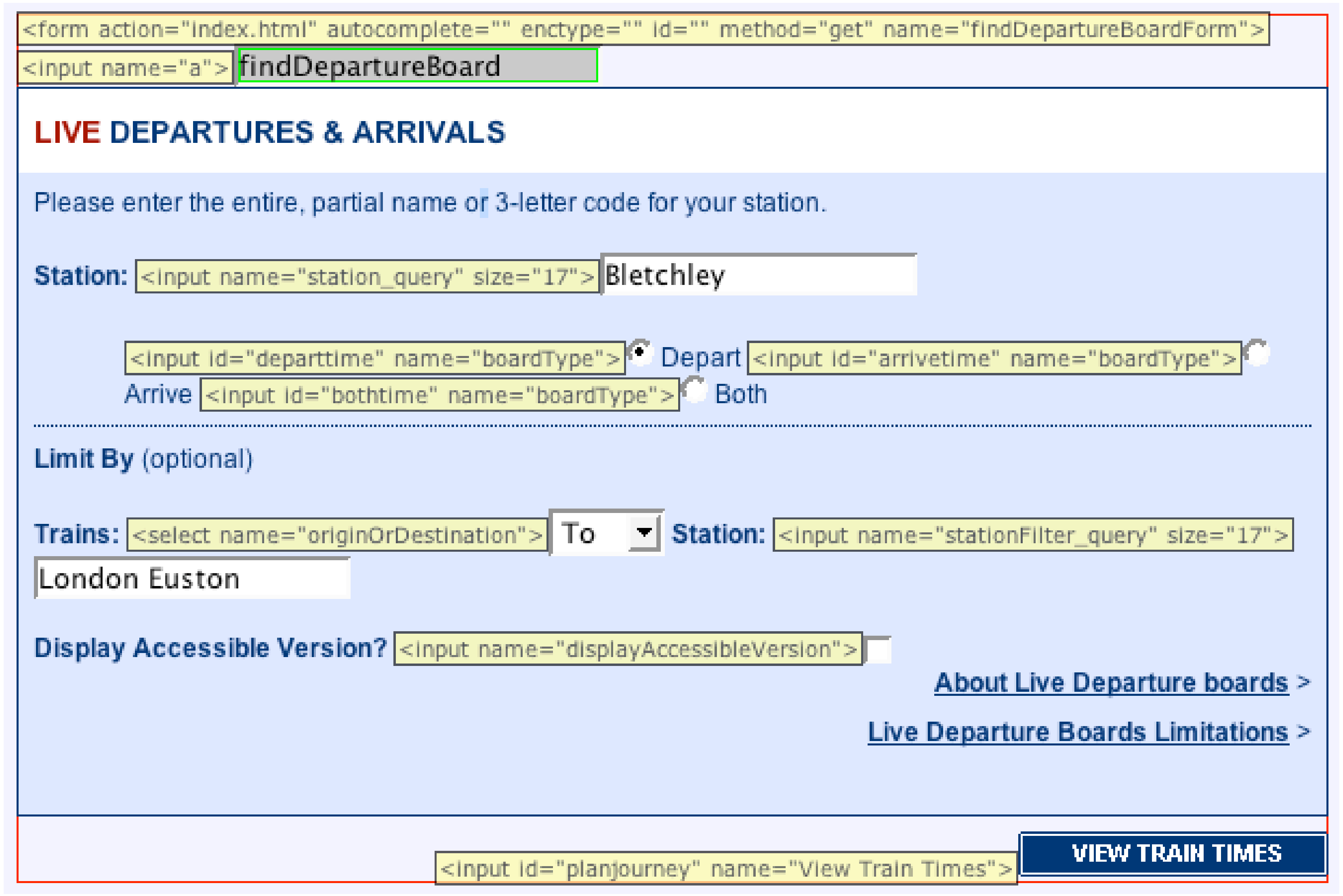
Limit By (optional)

Trains: To Station: London Euston

Display Accessible Version? ☐

About Live Departure boards >
Live Departure Boards Limitations >
VIEW TRAIN TIMES

Analysing Web Forms



<form action="index.html" autocomplete="" enctype="" id="" method="get" name="findDepartureBoardForm">
<input name="a"> findDepartureBoard

LIVE DEPARTURES & ARRIVALS

Please enter the entire, partial name or 3-letter code for your station.

Station: <input name="station_query" size="17"> Bletchley

<input id="departtime" name="boardType"> ● Depart <input id="arrivetime" name="boardType"> ○ Arrive <input id="bothtime" name="boardType"> ○ Both

Limit By (optional)

Trains: <select name="originOrDestination"> To Station: <input name="stationFilter_query" size="17"> London Euston

Display Accessible Version? <input name="displayAccessibleVersion"> ☐

About Live Departure boards >

Live Departure Boards Limitations >

<input id="planjourney" name="View Train Times"> VIEW TRAIN TIMES

Writing a Script

```
use strict;
use warnings;

use WWW::Mechanize;
my $mech = WWW::Mechanize->new( autocheck => 1 );

$mech->get('http://www.nationalrail.co.uk/ldb/');

$mech->form('findDepartureBoardForm');
$mech->field('station_query'              => 'Bletchley');
$mech->field('stationFilter_query'        => 'London Euston');
$mech->submit;
```

Writing a Script

```
use strict;
use warnings;

use WWW::Mechanize;
my $mech = WWW::Mechanize->new( autocheck => 1 );

$mech->get('http://www.nationalrail.co.uk/ldb/');

$mech->form('findDepartureBoardForm');
$mech->field('station_query'           => 'Bletchley');
$mech->field('stationFilter_query'     => 'London Euston');
$mech->submit;
```

Writing a Script

```
use strict;
use warnings;

use WWW::Mechanize;
my $mech = WWW::Mechanize->new( autocheck => 1 );

$mech->get('http://www.nationalrail.co.uk/ldb/');

$mech->form('findDepartureBoardForm');
$mech->field('station_query'          => 'Bletchley');
$mech->field('stationFilter_query'    => 'London Euston');
$mech->submit;
```

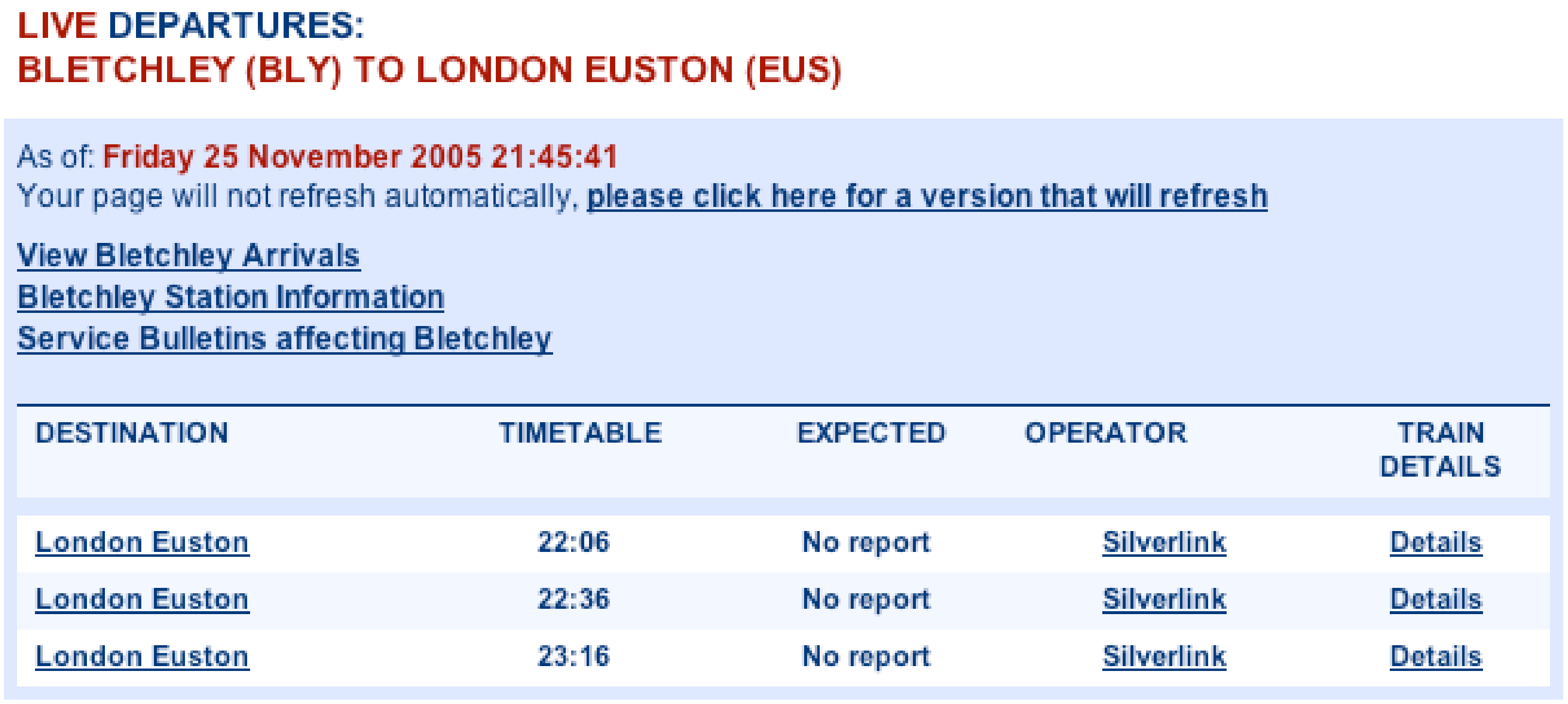
Writing a Script

```
use strict;
use warnings;

use WWW::Mechanize;
my $mech = WWW::Mechanize->new( autocheck => 1 );

$mech->get('http://www.nationalrail.co.uk/ldb/');

$mech->form('findDepartureBoardForm');
$mech->field('station_query'          => 'Bletchley');
$mech->field('stationFilter_query'    => 'London Euston');
$mech->submit;
```

Writing a Script

```
use strict;
use warnings;

use WWW::Mechanize;
my $mech = WWW::Mechanize->new( autocheck => 1 );

$mech->get('http://www.nationalrail.co.uk/ldb/');

$mech->form('findDepartureBoardForm');
$mech->field('station_query'           => 'Bletchley');
$mech->field('stationFilter_query'     => 'London Euston');
$mech->submit;
```

Analysing Results



LIVE DEPARTURES:
BLETCHLEY (BLY) TO LONDON EUSTON (EUS)

As of: **Friday 25 November 2005 21:45:41**
Your page will not refresh automatically, **please click here for a version that will refresh**

View Bletchley Arrivals
Bletchley Station Information
Service Bulletins affecting Bletchley

DESTINATION	TIMETABLE	EXPECTED	OPERATOR	TRAIN DETAILS
London Euston	22:06	No report	Silverlink	Details
London Euston	22:36	No report	Silverlink	Details
London Euston	23:16	No report	Silverlink	Details

Analysing Results

```
my $results_html = $mech->content;
my @time;
while ($results_html =~ m{<td >(\d\d:\d\d)</td>}g) {
    push @time, $1;
}

# @time contains ('22:06', '22:36', '23:16')
```

Analysing Results

```
my $results_html = $mech->content;
my @time;
while ($results_html =~ m{<td >(\d\d:\d\d)</td>}g) {
    push @time, $1;
}

# @time contains ('22:06', '22:36', '23:16')
```

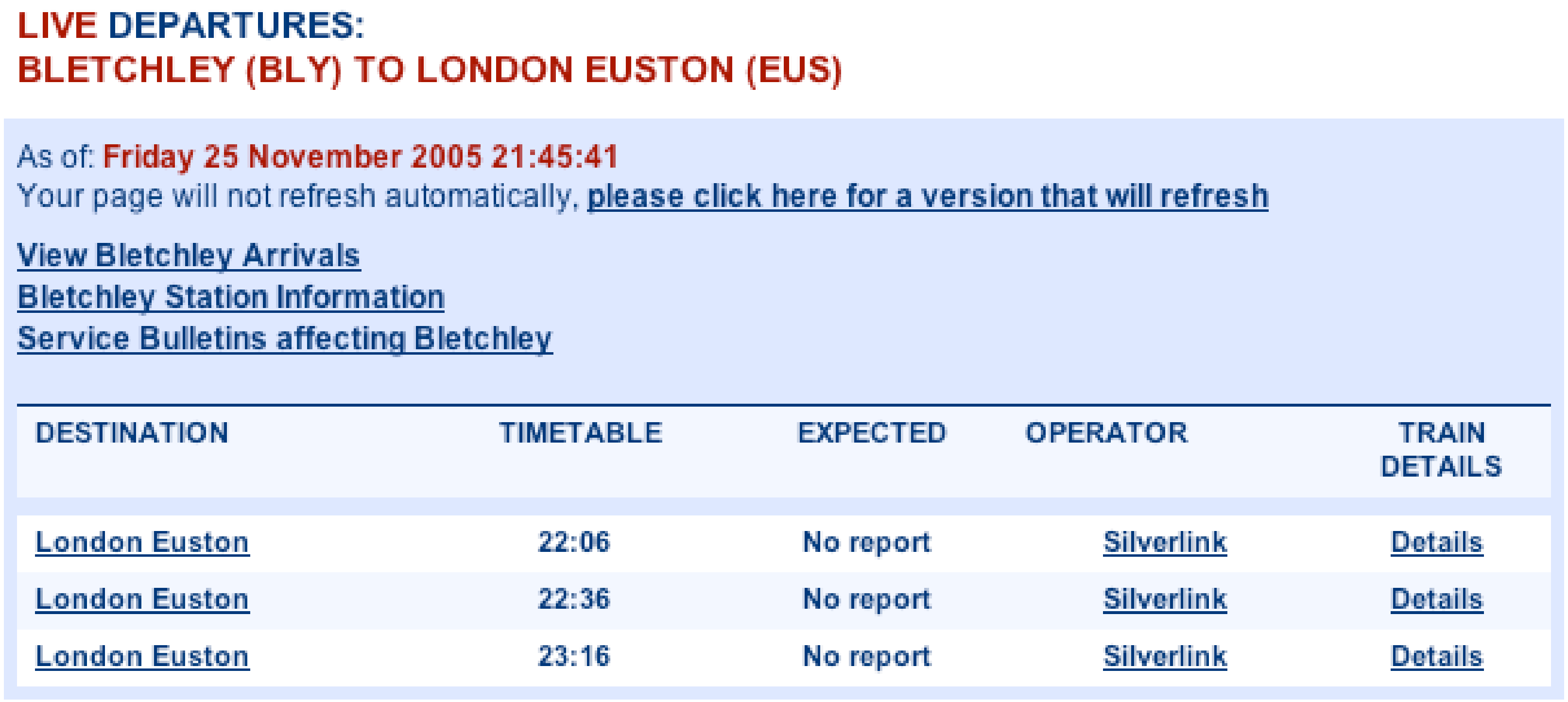
Analysing Results

```
my $results_html = $mech->content;
my @time;
while ($results_html =~ m{<td >(\d\d:\d\d)</td>}g) {
    push @time, $1;
}

# @time contains ('22:06', '22:36', '23:16')
```

Analysing Results

```
my $results_html = $mech->content;
my @time;
while ($results_html =~ m{<td >(\d\d:\d\d)</td>}g) {
    push @time, $1;
}

# @time contains ('22:06', '22:36', '23:16')
```

Following Links



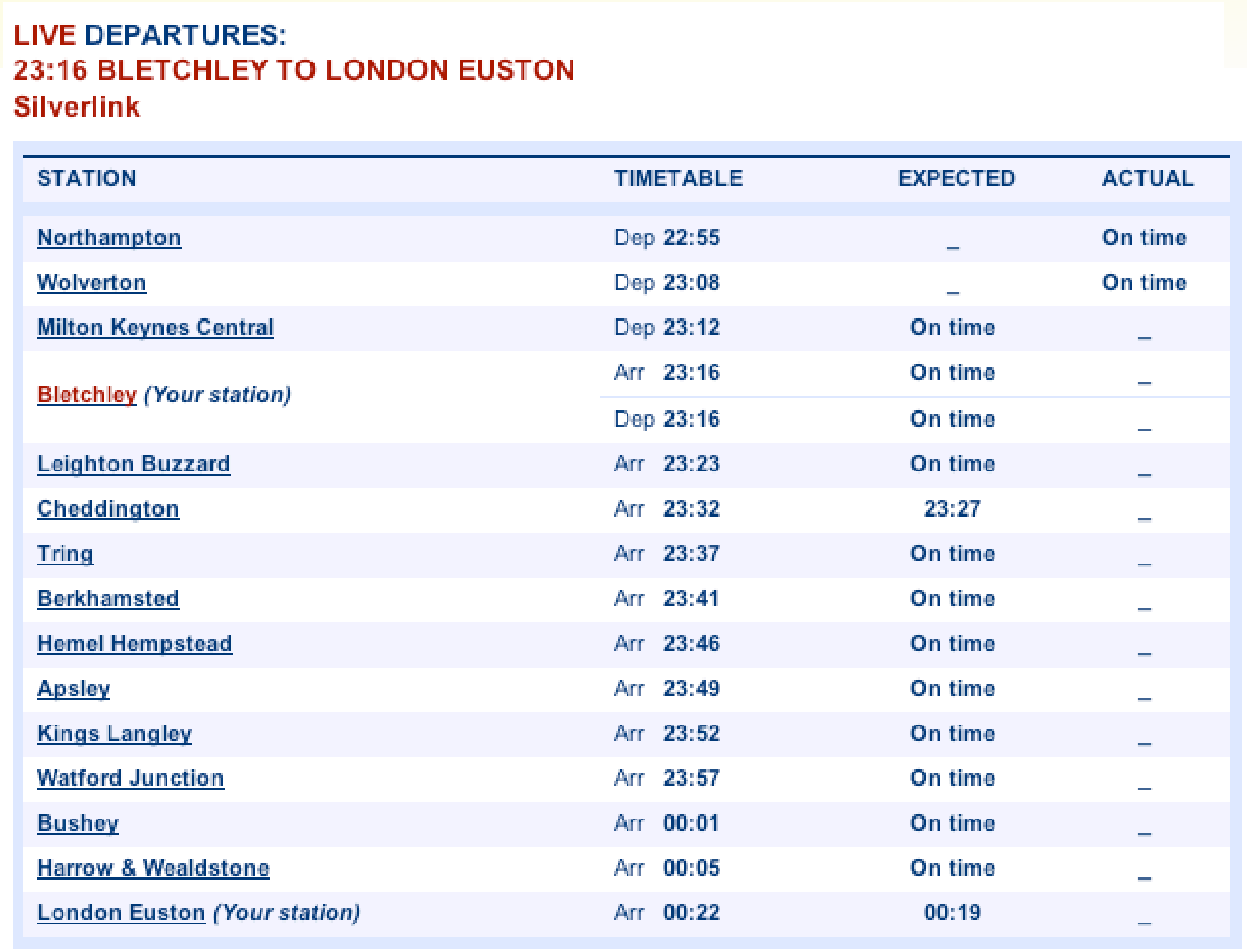
LIVE DEPARTURES:
BLETCHLEY (BLY) TO LONDON EUSTON (EUS)

As of: Friday 25 November 2005 21:45:41
Your page will not refresh automatically, please click here for a version that will refresh

View Bletchley Arrivals
Bletchley Station Information
Service Bulletins affecting Bletchley

DESTINATION	TIMETABLE	EXPECTED	OPERATOR	TRAIN DETAILS
London Euston	22:06	No report	Silverlink	Details
London Euston	22:36	No report	Silverlink	Details
London Euston	23:16	No report	Silverlink	Details

Following Links



LIVE DEPARTURES:
23:16 BLETCHLEY TO LONDON EUSTON
Silverlink

STATION	TIMETABLE	EXPECTED	ACTUAL
Northampton	Dep 22:55	_	On time
Wolverton	Dep 23:08	_	On time
Milton Keynes Central	Dep 23:12	On time	_
Bletchley *(Your station)*	Arr 23:16	On time	_
	Dep 23:16	On time	_
Leighton Buzzard	Arr 23:23	On time	_
Cheddington	Arr 23:32	23:27	_
Tring	Arr 23:37	On time	_
Berkhamsted	Arr 23:41	On time	_
Hemel Hempstead	Arr 23:46	On time	_
Apsley	Arr 23:49	On time	_
Kings Langley	Arr 23:52	On time	_
Watford Junction	Arr 23:57	On time	_
Bushey	Arr 00:01	On time	_
Harrow & Wealdstone	Arr 00:05	On time	_
London Euston *(Your station)*	Arr 00:22	00:19	_

Following Links

```perl
my @journey = $mech->find_all_links( text => 'Details' );
foreach (@journey) {
    my $content = $mech->get($_)->content;
    next unless $content =~ m{Cheddington};

    $content =~ m{<title>.*?(\d\d:\d\d)};
    print "The $1 stops at Cheddington\n";
}
```

Following Links

```
my @journey = $mech->find_all_links( text => 'Details' );
foreach (@journey) {
    my $content = $mech->get($_)->content;
    next unless $content =~ m{Cheddington};

    $content =~ m{<title>.*?(\d\d:\d\d)};
    print "The $1 stops at Cheddington\n";
}
```

Following Links

```
my @journey = $mech->find_all_links( text => 'Details' );
foreach (@journey) {
    my $content = $mech->get($_)->content;
    next unless $content =~ m{Cheddington};

    $content =~ m{<title>.*?(\d\d:\d\d)};
    print "The $1 stops at Cheddington\n";
}
```

Following Links

```
my @journey = $mech->find_all_links( text => 'Details' );
foreach (@journey) {
    my $content = $mech->get($_)->content;
    next unless $content =~ m{Cheddington};

    $content =~ m{<title>.*?(\d\d:\d\d)};
    print "The $1 stops at Cheddington\n";
}
```

Following Links

```
my @journey = $mech->find_all_links( text => 'Details' );
foreach (@journey) {
    my $content = $mech->get($_)->content;
    next unless $content =~ m{Cheddington};

    $content =~ m{<title>.*?(\d\d:\d\d)};
    print "The $1 stops at Cheddington\n";
}
```

Following Links

```
my @journey = $mech->find_all_links( text => 'Details' );
foreach (@journey) {
    my $content = $mech->get($_)->content;
    next unless $content =~ m{Cheddington};

    $content =~ m{<title>.*?(\d\d:\d\d)};
    print "The $1 stops at Cheddington\n";
}
```

XPath

- XML Parsing Language
- W3C Specification
- Support in many languages (including Perl)

XPath Example

```
<?xml version="1.0"?>
<html>
  <head>
    <title>A title belongs in the</title>
  </head>
  <body>
    <p>
      Here is <a href="some" title="example">some</a>
      <a href="other" title="example2">text</a>.
    </p>
  </body>
</html>
```

XPath Example

/html

```
<?xml version="1.0"?>
<html>
  <head>
    <title>A title belongs in the</title>
  </head>
  <body>
    <p>
      Here is <a href="some" title="example">some</a>
      <a href="other" title="example2">text</a>.
    </p>
  </body>
</html>
```

XPath Example

/html/body

```
<?xml version="1.0"?>
<html>
  <head>
    <title>A title belongs in the</title>
  </head>
  <body>
    <p>
      Here is <a href="some" title="example">some</a>
      <a href="other" title="example2">text</a>.
    </p>
  </body>
</html>
```

XPath Example

/html/body//a

```
<?xml version="1.0"?>
<html>
  <head>
    <title>A title belongs in the</title>
  </head>
  <body>
    <p>
      Here is <a href="some" title="example">some</a>
      <a href="other" title="example2">text</a>.
    </p>
  </body>
</html>
```

XPath Example

```
/html/body//a/@href
```

```
<?xml version="1.0"?>
<html>
  <head>
    <title>A title belongs in the</title>
  </head>
  <body>
    <p>
      Here is <a href="some" title="example">some</a>
      <a href="other" title="example2">text</a>.
    </p>
  </body>
</html>
```

XPath Example

```
/html/body//a[@title="example2"]/@href
```

```
<?xml version="1.0"?>
<html>
  <head>
    <title>A title belongs in the</title>
  </head>
  <body>
    <p>
      Here is <a href="some" title="example">some</a>
      <a href="other" title="example2">text</a>.
    </p>
  </body>
</html>
```

Mechanize and XPath

```
use WWW::Mechanize ();
use XML::LibXML ();

my $mech = WWW::Mechanize->new();
$mech->get('http://annoyme.scrubhole.org/average.html');

my $parser = XML::LibXML->new();
$parser->recover(1);

my $doc = $parser->parse_html_string( $mech->content );
print $doc->findvalue(
    '/html/body//a[@title="example2"]/@href'
), "\n";
```

Mechanize and XPath

```
use WWW::Mechanize ();
use XML::LibXML ();

my $mech = WWW::Mechanize->new();
$mech->get('http://annoyme.scrubhole.org/average.html');

my $parser = XML::LibXML->new();
$parser->recover(1);

my $doc = $parser->parse_html_string( $mech->content );
print $doc->findvalue(
    '/html/body//a[@title="example2"]/@href'
), "\n";
```

Mechanize and XPath

```
use WWW::Mechanize ();
use XML::LibXML ();

my $mech = WWW::Mechanize->new();
$mech->get('http://annoyme.scrubhole.org/average.html');

my $parser = XML::LibXML->new();
$parser->recover(1);

my $doc = $parser->parse_html_string( $mech->content );
print $doc->findvalue(
    '/html/body//a[@title="example2"]/@href'
), "\n";
```

Mechanize and XPath

```
use WWW::Mechanize ();
use XML::LibXML ();

my $mech = WWW::Mechanize->new();
$mech->get('http://annoyme.scrubhole.org/average.html');

my $parser = XML::LibXML->new();
$parser->recover(1);

my $doc = $parser->parse_html_string( $mech->content );
print $doc->findvalue(
    '/html/body//a[@title="example2"]/@href'
), "\n";
```

Cleaning up Bad HTML

```
# The new() method takes arguments from the tidy.cfg file
my $tidy        = HTML::Tidy->new();
my $clean_html  = $tidy->clean($raw_html);
```

xsh: The XML Shell

```
xsh scratch:/> recovering 1
xsh scratch:/> open HTML doc=average.html
parsing average.html
done.
xsh doc:/> cd /html/body
xsh doc:/html/body> ls
<body><p>...</p><p>...</p></body>

Found 1 node(s).
```

xsh: The XML Shell

```
xsh scratch:/> recovering 1
xsh scratch:/> open HTML doc=average.html
parsing average.html
done.
xsh doc:/> cd /html/body
xsh doc:/html/body> ls
<body><p>...</p><p>...</p></body>

Found 1 node(s).
```

xsh: The XML Shell

```
xsh scratch:/> recovering 1
xsh scratch:/> open HTML doc=average.html
parsing average.html
done.
xsh doc:/> cd /html/body
xsh doc:/html/body> ls
<body><p>...</p><p>...</p></body>

Found 1 node(s).
```

xsh: The XML Shell

```
xsh scratch:/> recovering 1
xsh scratch:/> open HTML doc=average.html
parsing average.html
done.
xsh doc:/> cd /html/body
xsh doc:/html/body> ls
<body><p>...</p><p>...</p></body>

Found 1 node(s).
```

xsh: The XML Shell

```
xsh scratch:/> recovering 1
xsh scratch:/> open HTML doc=average.html
parsing average.html
done.
xsh doc:/> cd /html/body
xsh doc:/html/body> ls
<body><p>...</p><p>...</p></body>

Found 1 node(s).
```

xsh: The XML Shell

```
xsh doc:/html/body> ls *
<p>
    Here is <a href="some" title="example">some</a>
    <a href="other" title="example2">text</a>.
    </p>
<p>
    The end.
  </p>

Found 2 node(s).
```

Play Nicely!

- Robots Exclusion Protocol
  http://www.robotstxt.org/
- Site's Terms and Conditions
- `sleep` between requests
  WWW::Mechanize::Sleepy

Diagnosing Problems

```
use LWP::Debug qw(+);
LWP::Debug::debug qw(url);
```

Diagnosing Problems

- Web Developer Toolbar
- Live HTTP Headers
- DOM Inspector
- View Selection Source

Related Modules

- WWW::Mechanize::Pluggable
- Win32::IE::Mechanize & Mozilla::Mechanize
- HTTP::Recorder & WWW::Mechanize::Shell

References

- LWP (libwww-perl on CPAN)
- HTML::Parser (on CPAN)
- XPath (http://www.w3c.org/)
- tidy (http://tidy.sourceforge.net/)
- xsh (XML::XSH or XML::XSH2 on CPAN)