

Cross-Platform Testing

Кросс-платформенное тестирование

Peter Edwards
peter@dragonstaff.co.uk

Київ Перл Мова 2009
Perl Technical Talk
7th March 2009



Contents

Содержание

Background, CMS, Usage
Technology, Client-side, Server-side
Cross-Platform
Existing Testing, Problems

Add Windows Testing Under Unix
Test::MockObject
Running Unix unit tests under Windows
Future Plans For Testing
Summary and Links

Background Предыстория

London.pm MiltonKeynes.pm
Google Developer Day 2008 – London – Perl vs. PHP at chess



Cross-Platform Testing



Background Предыстория

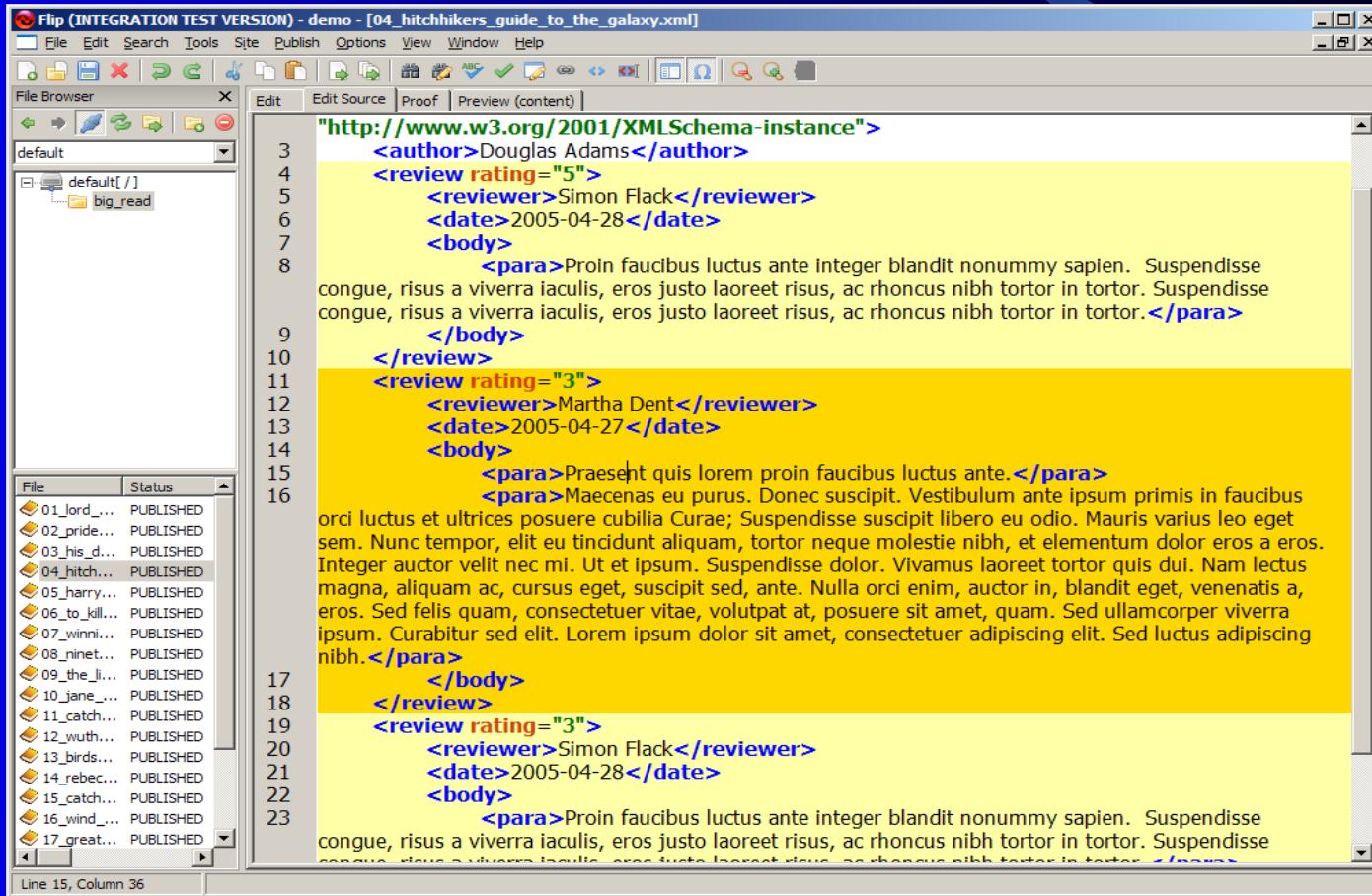


Coffee area | Кухня

Content Management System

Система управления содержимым

Used to enter XML documents transformed to make bbc.co.uk website



The screenshot shows the 'Flip (INTEGRATION TEST VERSION)' application window. The title bar reads 'Flip (INTEGRATION TEST VERSION) - demo - [04_hitchhikers_guide_to_the_galaxy.xml]'. The menu bar includes File, Edit, Search, Tools, Site, Publish, Options, View, Window, and Help. The toolbar contains icons for file operations like Open, Save, Print, and Publish. The left sidebar is a 'File Browser' showing a tree structure with 'default' selected, containing 'default[/]' and 'big_read'. The main area is an 'Edit Source' tab showing an XML document. The XML code is as follows:

```
<http://www.w3.org/2001/XMLSchema-instance>
  <author>Douglas Adams</author>
  <review rating="5">
    <reviewer>Simon Flack</reviewer>
    <date>2005-04-28</date>
    <body>
      <para>Proin faucibus luctus ante integer blandit nonummy sapien. Suspendisse congue, risus a viverra iaculis, eros justo laoreet risus, ac rhoncus nibh tortor in tortor. Suspendisse congue, risus a viverra iaculis, eros justo laoreet risus, ac rhoncus nibh tortor in tortor.</para>
    </body>
  </review>
  <review rating="3">
    <reviewer>Martha Dent</reviewer>
    <date>2005-04-27</date>
    <body>
      <para>Praesent quis lorem proin faucibus luctus ante.</para>
      <para>Maecenas eu purus. Donec suscipit. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Suspendisse suscipit libero eu odio. Mauris varius leo eget sem. Nunc tempor, elit eu tincidunt aliquam, tortor neque molestie nibh, et elementum dolor eros a eros. Integer auctor velit nec mi. Ut et ipsum. Suspendisse dolor. Vivamus laoreet tortor quis dui. Nam lectus magna, aliquam ac, cursus eget, suscipit sed, ante. Nulla orci enim, auctor in, blandit eget, venenatis a, eros. Sed felis quam, consectetur vitae, volutpat at, posuere sit amet, quam. Sed ullamcorper viverra ipsum. Curabitur sed elit. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed luctus adipiscing nibh.</para>
    </body>
  </review>
  <review rating="3">
    <reviewer>Simon Flack</reviewer>
    <date>2005-04-28</date>
    <body>
      <para>Proin faucibus luctus ante integer blandit nonummy sapien. Suspendisse congue, risus a viverra iaculis, eros justo laoreet risus, ac rhoncus nibh tortor in tortor. Suspendisse congue, risus a viverra iaculis, eros justo laoreet risus, ac rhoncus nibh tortor in tortor.</para>
    </body>
  </review>
```

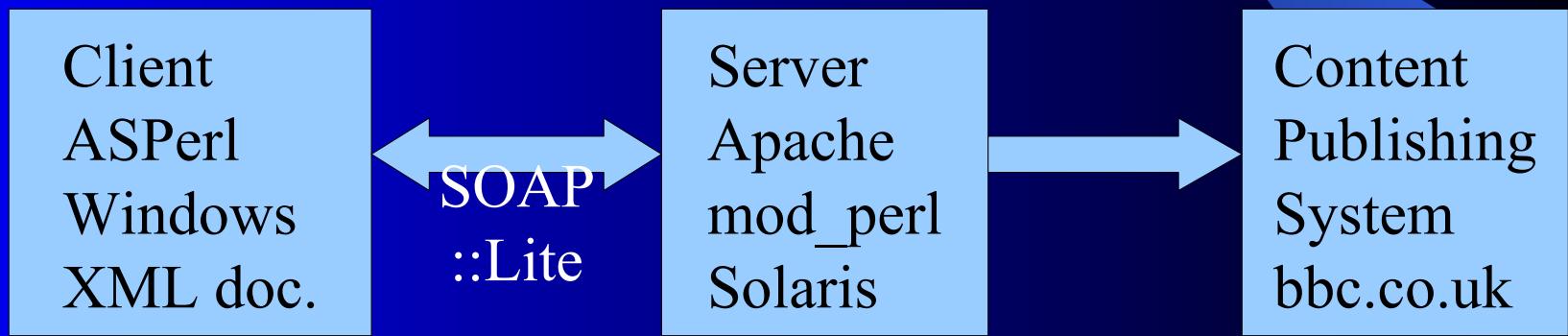
The status bar at the bottom left shows 'Line 15, Column 36'. The bottom right corner of the slide has the number '5'.

Usage Использование



- 100s of users
- Time-critical publishing
- After software release “failure is not an option”
- Answer: test coverage, and lots of it

Technology Устройство



Client-side

Клиентская часть

Client-side

- MS Windows
- GUI using WxPerl (wxWidgets)
- WYSIWYG editing
- Talks SOAP over HTTP to server
- Runs under ActiveState Perl



Server-side Серверная часть

Server-side

- Handles SOAP requests
- Stores document blobs in filesystem
- Stores indexes, metadata in Oracle database
- Runs under Solaris Perl
- Publishes to BBC Central Production Servers



Cross-Platform Кросс-платформенность

CMS code running on Windows and Solaris

Solaris perl 5.8.8

```
$ perl -V
Summary of my perl5 (revision 5 version 8
subversion 8) configuration:
Platform:
    osname=solaris, osvers=2.10, archname=sun4-
solaris
```

Windows ASPerl 5.8

```
C:\WINNT>perl -V
Summary of my perl5 (revision 5 version 8 subversion 6) configuration:
Platform:
    osname=MSWin32, osvers=4.0, archname=MSWin32-x86-multi-thread
```

Existing Testing

Существующие тесты

Unit tests for development

Automated overnight smoke testing of unit tests

Dev / Staging Test / Live environments

Manual release test on staging test area using
Windows application

Problems With Testing

Проблемы тестирования

Lots of tests for server side code

But few for client side because difficult to run
WxWidgets code on Unix in batch (no X display)

Existing tests run on Unix, fail on Windows

Add Windows Testing Under Unix

Добавление Windows-тестирования под Unix

Need to write client-side tests for

- GUI WxPerl -> Gtk+ under Solaris
 - ‘Use Wx’ was failing because no X display
 - Problems with font sizing and window alignment
 - Windows-specific components, e.g. ActiveX Altova editor
- Installation
 - Shortcuts, registry Win32::OLE, unzipping archives to Windows Apps dir etc.
- Test coverage – Devel::Cover

Add Windows Testing Under Unix

Добавление Windows-тестирования под Unix

Solutions

- GUI – can use Xvfb (X windows virtual framebuffer)

```
$ alias runxvfb='Xvfb :10 -dev vfb  
screen 0 1152x900x8 > /dev/null  
2>&1 &'
```

Lets you check code compiles and can call many routines

But how do you test the UI rendered properly?

- interpreting the virtual screen bitmaps is too hard!

Add Windows Testing Under Unix

Добавление Windows-тестирования под Unix

Solutions

- Sand-boxing and mocking
 - Mock required Win32 functions and check for correct call order, parameters
 - Make them do file I/O to a sandbox area
 - Test::MockObject** - Perl extension for emulating troublesome interfaces
 - Test::MockModule** - Override subroutines in a module for unit testing

Test::MockObject

Mocking

```
my $class = 'Win32::OLE';
my $ole_obj = Test::MockObject->new;
$ole_obj->fake_module($class);
$ole_obj->fake_new($class);

$ole_obj->mock( 'AppData' ,
  sub {
    return catdir(qw(data win32) ,
      'Application Data');
  } );
...
.

# unit test my routine that uses
# Win32::OLE -> my mocked object
my_routine();
```

Test::MockObject

Helper

```
sub make_mock_obj_in_class {
    my $class = shift;
    my $obj = Test::MockObject->new;
    $obj->fake_module($class);
    $obj->fake_new($class);
    return $obj;
}

my $ole_obj = make_mock_
    obj_in_class('Win32::OLE');
$ole_obj->mock('AppData', sub {...});
```

Test::MockObject

Mocking installer's WxWidgets progress bar

```
my $wx = make_mock_obj_in_class( 'Wx' );  
my $mock_WxPerlSplashProgress =  
    make_mock_obj_in_class(  
        'Wx::Perl::SplashProgress' );  
  
$mock_WxPerlSplashProgress->set_true(qw(  
    SetLabelColour  
    SetIcon  
    Show  
    SetValue  
    Update  
    Destroy ));  
  
$mock_WxPerlSplashProgress->mock(  
    SetLabel => sub { diag 'SetLabel: '.  
        $_[1] } );
```

Test::MockObject

Unit test installer's use of WxWidgets progress bar

```
use Test::More tests => 3;

# test _install_loginscript()
my $i = FLIPClient::Installation->new(...);
$mock_WxPerlSplashProgress->clear();
is( $i->_install_loginscript, 1,
    '_install_loginscript' );

# order of subroutine calling
$mock_WxPerlSplashProgress->
    called_pos_ok( 4, 'SetLabel', 'called
Wx::Perl::SplashProgress->SetLabel' );

# arguments passed to subroutine
$mock_WxPerlSplashProgress->
    called_args_pos_is( 4, 2, 'Checking login
script' );
```

Test::MockObject

Helper to see order of calls made and arguments

```
sub dump_mock_calls {
    my $mockobj = shift;
    my $i = 1;
    while (my $name = $mockobj->call_pos($i)) {
        diag "call $i: $name";
        my @args = $mockobj->call_args($i);
        for (0 .. $#args) {
            diag 'arg ' . ($_.+1) . ': ' ;
            diag Dumper($args[$_]);
        }
        $i++;
    }
}

# then in test script
dump_mock_calls($mock_WxPerlSplashProgress);
```

Test::MockObject

Direct mocking of variable and subroutines

Wx has > 3000 subroutines/variables – only mock the ones used

```
BEGIN {
    $mock_Wx = make_mock_obj_in_class( 'Wx' );
    no strict 'refs';
    *{ 'Wx::wxICON_EXCLAMATION' } = sub { 1 };
    *{ 'Wx::wxOK' } = sub { 1 };
    *{ 'Wx::Yield' } = sub { 1 };
    *{ 'Wx::MessageBox' } = sub {
        my ($msg, $caption, $style, $window, $x,
        $y) = @_;
        $msg =~ s/\n/\n# /g;
        DUMP sprintf('Wx::MessageBox() caption = <<<%s>>>, style = %d, msg = <<<%s%s>>>', $caption, $style, "\r\n#", $msg);
    };
    @{ 'Wx::EXPORT_OK' } = qw( id );
    %{ 'Wx::EXPORT_TAGS' } = ( id => [ 'id' ],
        everything => \@{ 'Wx::EXPORT_OK' } );
}
```

Running Unix unit tests under Windows

Запуск Unix unit-тестов под Windows

Perl libraries shared between Unix and Windows not being tested properly client-side

Perl Portability

- "perldoc perlport" <http://perldoc.perl.org/5.8.8/perlport.html>
"When the code will run on only two or three operating systems, you may need to consider only the differences of those particular systems. The important thing is to decide where the code will run and to be deliberate in your decision."
- Only worrying about Windows and Unix; OpenVMS is hard
 - binmode and chomp - binmode saves headaches on Windows like EOF ^Z; watch out for CR-LF
 - use File::Spec::Functions rather than Unix paths

YES : `my $path = rel2abs(catdir(qw(data local cache file.txt)));`

NO : `my $path = './data/local/cache/file.txt';`

Perl functions handle '/', but what about calls to Win32 API?

Running Unix unit tests under Windows

Запуск Unix unit-тестов под Windows

Generic configuration interface with platform-specific subclasses

System.pm

```
|-- System/Win32.pm  
|-- System/Unix.pm
```

```
using File::Spec::Functions for paths
```

Changed tests from path strings to regexes using a quote path separator

```
my $script = $i->startup('remote');  
NO:  
is( $script, 'scripts/FLIP_real.PL',  
  '$i->startup("remote") script'  
YES:  
$ps = ($^O eq 'MSWin32') ? "\\" : '/';  
$qps = quotemeta $ps;  
like( $script, qr{  
    scripts[$qps] FLIP_real.pl \z  
}xms, '$i->startup("remote") script' );
```

(“Perl Best Practices” style regex)

Need to actually run the tests on multiple platforms

Future Plans For Testing

Планы на будущее

Automate application release test under Windows

- Win32::GuiTest (or pay for “WinRunner”)

Summary and Links

ИТОГИ И ССЫЛКИ

Summary

- "perldoc perlport"
- Write cross-platform tests from the outset; convert old ones
- Mock platform-specific GUI or system library calls
- Automate tests (life is short) and get as much coverage as possible

Links

- WxPerl <http://wxperl.sourceforge.net/>
- WxWidgets <http://docs.wxwidgets.org/trunk/>
- "Perl Testing: A Developer's Notebook" Ian Langworth & chromatic, O'Reilly Media, Inc., 2005
<http://preview.tinyurl.com/5k6wnc>

Thank you. Any Questions?