



bookmarklet will figure this out for the page as well as for all the frames (and their frames) recursively. Enjoy!

#### Install

Drag this to your bookmarklets/favorites or right click and add to

favorites:

```
0){response+="\n\n";for(var i=0;iquirks or !quirks
```

#### Sample output

Here's a sample output, produced when used in my Wordpress backend when

writing this post:

As you can see the page has two frames (probably iframes, doesn't matter), one of them is rendered in Standards Compliant Mode (CSS1Compat) the other one is in Quirks Mode (BackCompat). The overall document is CSS1Compat as well. For the frames, if they were named, you would see the name of the frame before the URL brackets.

And this is GMail, wow, lotsa frames, none compliant

#### The code

The code is pretty simple, just accessing the compatMode of the the document object. Here it is in more human readable form (not one long line like bookmark code).

```
var response = 'Document mode: ' + document.compatMode;
function checkFrames(w) {
if(w.frames && w.frames.length>0){
response+="\n\n";
for(var i=0;i<w.frames.length;i++){
var fr=w.frames[i];
try {
response+=fr.name +
'(+fr.document.location+) - '+
fr.document.compatMode+"\n";
} catch (e) {
response+='Could not access this frame\n';
}
}
```

```
checkFrames(fr);  
}  
}  
}  
checkFrames(window);  
alert(response);
```

0 points September 21, 2006 by Stoyan at phpied.com

post #

[–]

#### ABAP code syntax highlighting

Just finished and eager to share – I added a new syntax definition to

the Text\_Highlighter PEAR package (see also here). It's for

highlighting code written in the SAP's own ABAP programming language.

A live demo is available at the hiliteme.com site, just pick ABAP from

the drop-down of programming languages. Any feedback is appreciated,

because it's a brand new thing and may have bugs or incompletenesses

(is that a word?). So feel free to highlight your ABAP code and post it

to blogs or forums.

The implementation wasn't hard, the Text\_Highlighter package is made to

be extended and even provides the tools for that. All you need to do is

create an XML file that contains keywords and other syntax rules, such

as formats of the comments and so on. Then there is a command line tool

that takes the XML file and generates a class out of it. The class is

later on used when highlighting. Here's the XML file in case you want

to improve on it and generate your own ABAP.php class:

```
abap.xml  
ABAP.php source file generated  
0 points September 20, 2006 by Stoyan at phpied.com post #
```

[–]

AJAX MVC (so to speak)

This is sort of a framework thing to create AJAX applications, based on

the MVC design pattern. Yep, I have a lot of buzzwords here, I admit,

but this shouldn't be taken too seriously. I was doing a bunch of small

projects lately and I found myself using something like this little

framework, without even thinking about it. Then I thought about it and

I found that the scripts and the organization of them may resemble MVC

a bit. So how does MVC fit when you mix things like thin and fatter

client, HTML, JavaScript, XMLHttpRequest, PHP and CSS?

Usual AJAX app flow

What usually happens in an AJAX application is:

you have an HTML page, styled with CSS

you click on something

JS sends request to the server (to a PHP script)

JS updates the original HTML page

Mapping to the MVC pattern

OK, so what part of this process can be associated with a View, or a

Model or a Controller? The Model is easy, it's the business logic,

writing to a database and so on. This is the PHP script. The View?

Obviously this is the HTML page and the CSS. But I'd like to think also

about the JS that updates the page as part of the View. I mean it makes

sense, it's updating the presentation part. Sometimes you even use

innerHTML in the JS, but even if you use DOM, it becomes part of the

HTML anyway. How about the Controller? Well, we have two controllers

here. One that is on the server side, a PHP script that receives

requests and "asks" the Model for the response. The other controller is

on the client side, this is the JavaScript that decides what happens on a click of a button and sends an appropriate AJAX request to the PHP controller. Therefore I would consider any behavioural JS as part of the Controller, including attaching events as well as sending HTTP requests.

Here's an illustration:

In action (example)

I went ahead and implemented a very simple application to prove the concept. It's just a blank styled HTML page with a button. The HTML page includes two JavaScripts responsible for behaviours (Controller) and page updates (View). The page also includes a few unrelated helper javascripts, in my case I'm using the YUI library. The JS Controller attaches an event to the button. Then when you click the button, the JS Controller sends a request to the PHP controller. The PHP controller (just a simple switch) figures out what was requested and calls the appropriate object of the business model. In my simplistic case, the abovementioned "model object" is just a simple function, but this can be easily built upon. The Model returns (JSON-encoded) response, in this case it's a list of installed PHP extensions. Now the response is received by the View JS and it updates the page. After that the View calls another function from the JS controller that attaches new events to the new content. (Yep, a little glitch here, maybe it would have been better if the Model's response is handled by the JS controller which in turn calls the JS view updater, but anyway this is easy to fix)

Directory layout

Here's the directory structure:

One might argue that it's better if you don't mix .js, .css and .php

files in the same directory but the whole idea is open to

interpretations anyway, it's just an illustration of the idea.

The code for the example

We get to the fun part, the actual implementation. So we start with a

simple .html page, the initial part of the view.

This is index.html

```
<?xml version="1.1" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml xml:lang="en">
<head>
<title>Welcome</title>
<link rel="stylesheet" href="../view/styles.css" type="text/css"
media="all" title="Default styles" />
<script language="javascript" type="text/javascript"
src="../ extras/yui/build/yahoo/yahoo-min.js"></script>
<script language="javascript" type="text/javascript"
src="../ extras/yui/build/event/event-min.js"></script>
<script language="javascript" type="text/javascript"
src="../ extras/yui/build/connection/connection-min.js"></script>
<script language="javascript" type="text/javascript"
src="../view/updates.js"></script>
<script language="javascript" type="text/javascript"
src="../controller/behaviours.js"></script>
</head>
<body>

Welcome to my app!
<br />
<form action="" method="post">
<input type="button" name="b" id="thebutton" value="I'm a button."/>
```

```
click me!" />  
</form>  
<div id="content">&nbsp;  </div>
```

```
</body>  
</html>As you can see, nothing special, simply including the CSS
```

styles, the YUI "extras" and two other javascripts – one part of the  
View and one that is part of the Controller.

The Controller JS is responsible for attaching an event listener to the

button.

This is an excerpt from the behaviours.js

```
// the behaviour class  
var behaviours = {  
  
phpcontroller: "../controller/switch.php?request=",  
  
// more behaviour.methods....  
};  
  
// initial page load, attach onload event(s)  
YAHOO.util.Event.addListener(  
'thebutton', 'click', behaviours.theButtonClick);Now when the user  
clicks the button, the method behaviours.theButtonClick() is executed.  
  
It fires a request to the PHP controller switch and says that the  
request type is "loadSomething":  
  
theButtonClick: function(e) {  
alert('Ouch! nnOK, I'll make a request for ya, buddy!');  
YAHOO.util.Connect.asyncRequest(  
'GET',  
behaviours.phpcontroller + 'loadSomething',  
{success: updates.writeContent}  
});  
}.The PHP controller (controller/switch.php) receives the request, does
```

a simple switch to validate the request type and then calls the  
appropriate (in my case just a simple) function from the business  
model. Here's the full switch.php code:

```
<?php
// is this a request?
if (empty($_GET['request'])) {
die();
}
// get the business logic
include_once '../model/business.php';

// figure out the request
// and call the business logic object
switch ($_GET['request'])
{
case 'loadSomething':
echo loadSomething();
break;
case 'loadSomeMore': // not used, example
echo loadSomeMore();
break;
}
?>
```

The function loadSomething() from the PHP model gets a list of

installed PHP extensions, encodes them into JSON and sends them back.

This is a full listing of the ../model/business.php

```
<?php
function loadSomething() {
$extensions = get_loaded_extensions();
return '[' . implode(", ", $extensions) . "']";
}
?>
```

If you go back and look at the AJAX request, you'll see that on

success, I call the updates.writeContent() method. The

../view/updates.js script contains stuff that updates the HTML of the

original page, so its place is in the View part of the app.

writeContent simply creates an HTML table with the results (the list of

PHP extensions). Then I wanted to attach event listeners to this table

just to change color, but it can be more than that. Attaching events is

a job for the JS Controller, therefore a method of its class is called.

Here's a full listing of updates.js:

```
var updates = {  
  
writeContent: function (xmlhttp) {  
if (!xmlhttp.responseText) {  
alert("I got nothing from the server");  
}  
var data = eval(xmlhttp.responseText);  
var write_to = document.getElementById('content');  
write_to.innerHTML = "// yeah, I know  
  
var html2dom_root = write_to;  
var table = document.createElement("table");  
var table_1_tbody = document.createElement("tbody");  
for (var i in data) {  
table_1_tbody_2_tr = document.createElement("tr");  
table_1_tbody_2_tr_1_td = document.createElement("td");  
num = 1 + parseInt(i);  
table_1_tbody_2_tr_1_td_1_text = document.createTextNode(num);  
  
table_1_tbody_2_tr_1_td.appendChild(table_1_tbody_2_tr_1_td_1_text);  
table_1_tbody_2_tr.appendChild(table_1_tbody_2_tr_1_td);  
table_1_tbody_2_tr_2_td = document.createElement("td");  
table_1_tbody_2_tr_2_td_1_text =  
  
document.createTextNode(data[i]);  
  
table_1_tbody_2_tr_2_td.appendChild(table_1_tbody_2_tr_2_td_1_text);  
table_1_tbody_2_tr.appendChild(table_1_tbody_2_tr_2_td);  
table_1_tbody.appendChild(table_1_tbody_2_tr);  
}  
table.appendChild(table_1_tbody);  
html2dom_root.appendChild(table);  
  
behaviours.updateTableBehaviour();  
}  
}(BTW, for the DOM part I'm used the help from my little tool html2dom  
to make my life a bit easier)
```

And finally here's the rest of the JS controller (behaviours.js), the  
method behaviours.updateTableBehaviour() that adds an event listener to  
the new table and the trClick() that handles clicks on this table. On  
click, it justs changes the color of the underlying row.

```
trClick: function (e) {  
var target = (e.srcElement) ?  
e.srcElement.parentNode : e.target.parentNode;  
if (target.tagName == 'TR') {  
if (target.className == 'tr-on') {  
target.className = '';  
} else {  
target.className = 'tr-on';  
}  
}  
}
```

```
updateTableBehaviour: function () {  
var el = document.getElementById('content').firstChild;  
YAHOO.util.Event.addListener(  
el, 'click', behaviours.trClick);  
}Demo and downloads  
Demo – the live example  
Zipped demo – all the source code for the example  
Template – the source code for the example but with the example part
```

commented, so you can use it as a template for your next AJAX project.

The only thing you need to do is to drop the YUI in the extras/yui

folder.

Thank you for reading, any comments welcome!

0 points September 20, 2006 by Stoyan at phpied.com post #

[–]

### HTML2DOM

Here's this HTML-2-DOM service – <http://www.html2dom.com> What it does

is pretty simple – you paste some HTML code and the output is JS script

code that uses DOM functions to produce the same result. Could be

useful when you're working on an AJAX-style app that generates new

content using JavaScript.

I build this simple script, inspired by this great book I was reading –

"Build Your Own AJAX Web Applications". In the book, the author

sometimes starts with writing up what is the HTML code for the result

you want to achieve, and then goes ahead with giving the DOM code.

Because, you know, DOM code can be quite verbose and sometimes a bit

hard to follow. So I thought, why not write up a simple tool to

automate this HTML to DOM transition.

The code is not complicated at all, it just takes the HTML, treats it

as an XML document, then loops through all the elements of the XML doc

and all the attributes for each element. The script is here, hopefully

reusable and you can grab it for your own projects if you wish. You can

check the source of [html2dom.com](http://html2dom.com)'s index page for an example how to use

the [html-2-dom](#) class.

Some limitations of the script (that I know of) are result of the fact

that I'm treating the HTML as XML document. So you might get some

errors if the HTML you paste is not well-formed, with all closed tags

and so on. Also you cannot use `&nbsp;` and other entities, because XML

doesn't know about them. What XML knows is only the pre-defined 5. And

last, out of the different node types, my script understands only three

- element, attribute and comment. I think it's enough for the practical

purposes I was aiming at, even the comment type is a bit of a stretch.

So enjoy and as always, any feedback is welcome!

0 points September 15, 2006 by Stoyan at [phpied.com](http://phpied.com) post #

[ - ]

The PEAR book is on it's way

Here's the link to publisher's page dedicated to the PHP Programming

with PEAR. Guess who wrote the chapter for MDB2?

It's an honour to me to be in the company of the other authors, people

who have done a lot for the PEAR community:

Stephan Schmidt

Aaron Wormus

Carsten Lucke

Here's what the book is about:

Accessing databases with MDB2

Displaying data in a range of formats (HTML, Excel spreadsheet, PDF)

Creating and parsing XML documents

Serializing PHP objects into XML, and unserializing XML documents to

PHP objects

Consuming and offering web services

Accessing Web APIs including Google, Yahoo, Amazon, and Technorati

250 pages of good stuff Get your copy or just spread the word!

Update: In the rush to share the news I forgot to say a big "thank

you!" to Lukas Smith, the man behind MDB2, who was responsive as always

and was kind enough to review my chapter.

0 points September 12, 2006 by Stoyan at phpied.com post #

[-]

HiLiteMe.com updated

Following from here, I'm proud to announce an update to HiLiteMe.com.

With two custom renderers for Text Highlighter, the service now offers

you:

- BB code for your source, so you can post beautiful code to bulletin

boards and others that use BB code for formatting

- simple HTML code - the formatting is using only the tags b, i and u.

this is for devices such as iPod that can understand only tiny amount

of HTML code. So you highlight some code and take it on the road.

sweet.

In addition to that there is Simple HTML preview, CSS styles that are

used for the "rich" HTML highlighting and also the code for the rich

HTML formatting, so no need to view->source->copy

Other additions – tabsize setting (how many spaces for a tab, if you

paste code with tabs in it) and line numbers setting (yes/no) to

specify whether you want lines to be numbered in your code.

Test drive?

Note on the BB code highlighting – I tested on:

– vBulletin – works beautifully and

– phpBB – the current out-of-the-box version will not work, because

it's not allowing [color] BB tags inside [code] tags. Buuut, there is a

MOD to enable this

0 points September 9, 2006 by Stoyan at phpied.com post #

[-]

Greasemonkey – execute custom javascripts on any page

If you want to try executing custom local scripts on any page you

visit, try Greasemonkey. Here's a 10 seconds tut.

The task is to create a custom script and to make sure it's executed

every time you leave a page on phpied.com

(prerequisite) Get Firefox!

Install Gresemonkey from here

Create a file phpied.user.js (all your custom scripts must end in

..user.js) with the following

var start = new Date();

window.addEventListener("unload", function(e) {

var end = new Date();

var diff = Math.floor((end – start) / 1000);

alert("Man, I spent " + diff +

" of my precious seconds on this guy's page!" +

" Now that's called investment!"

);

}, false);Open phpied.user.js in the browser. You'll see a message from

Greasemonkey inviting you to install.

Click Install... and you're done. Now this script will execute on every

single page you hit. To change it so that it executes only when you

visit [phpied.com](http://phpied.com) do:

Right-click the monkey icon in the bottom-right of the browser screen.

Select Manage User Scripts.

Click the \* in the Included Pages list, then hit Edit. Type

"<http://www.phpied.com/>\* This means "execute this script on every page

on [phpied.com](http://phpied.com)". Click OK.

Reload this page to see the script in action.

N.B. To modify a user script, do not modify the original file where you

had it initially on your file system, won't work, I tried it Instead.

modify the copy that GM has stored. Right-click the monkey icon ->

Manage -> select your script in the listing on the left and click Edit.

I've read before about the Greasemonkey Firefox extension, but never

tried it before yesterday. Never tried probably because of a comment on

the [sitepoint.com](http://sitepoint.com)'s article about Greasemokey. The comment I found so

funny, yet true, was "I just don't like the idea of having to spend

time on another person's website when I barely get enough time to spend

on my own." Well, sometimes one might like to try custom scripts on

his/her own site, for example to test some stuff on the production

server without the risk of breaking something.

More resources:

- [Home page](#)

- [Tutorial @ sitepoint](#)

- [Free ebook](#)

- [User scripts](#)

0 points September 7, 2006 by Stoyan at [phpied.com](http://phpied.com) post #

[ - ]

[HiLiteMe.com](#)

In case someone is wondering how do I highlight the code I post on this

blog ... well, the lazy way. I don't. Some time ago I setup a free

service, hiliteme.com to do it for me, then I just copy/paste the

generated code. It's far from the best solution, but it's definitely

the laziest, without any spend-time-to-save-time effort on my end

So if you ever need to highlight source code – for a blog, word doc or

whatever, you can use this free service. I'll be happy. HiLiteMe.com is

using the PEAR Text Highlighter package (I talk about it here)

BTW, I think in general it's a good idea is to use JavaScript to do the

code highlighting. After all, it's just presentation and if it can be

done on the client, why loading you server with this task. I know at

least two free scripts that do that, there's probably more. The one

that looks very good is this one.

0 points September 4, 2006 by Stoyan at phpied.com post #

[–]

Parent's styles in an iframe

Here's a JavaScript that let's you style an iframe just like its top

parent. The script is basically just a proof of concept I did after

talking to a friend about similar problem he has had in the past, so

feel free to modify and use if you like it.

So I have a page, called big.html and a stylesheet for this page,

called big.css. On the page there is an iframe that loads small.html.

small.html has its own style, called small.css. What my little

Javascript function does is:

Getting all top parent's <link> tags

Looping through the tags and checking if the rel attribute has value

stylesheet

For all stylesheets found, makes a copy of the link tag and all its

attributes and adds it to the head of the iframed page

Here's the code:

function styleMe() {

if(window.top && window.top.location.href != document.location.href)

{

// I'm small but I'm not alone

// all parent's <link>s

var linkrels = window.top.document.getElementsByTagName('link');

// my head

var small\_head = document.getElementsByTagName('head').item(0);

// loop through parent's links

for (var i = 0, max = linkrels.length; i < max; i++) {

// are they stylesheets

if (linkrels[i].rel && linkrels[i].rel == 'stylesheet') {

// create new element and copy all attributes

var thestyle = document.createElement('link');

var attrib = linkrels[i].attributes;

for (var j = 0, attribmax = attrib.length; j < attribmax; j++)

{

thestyle.setAttribute(attrib[j].nodeName,

attrib[j].nodeValue);

}

// add the newly created element to the head

small\_head.appendChild(thestyle);

}

}

// maybe, only maybe, here we should remove the kid's own styles...

} else {

alert('I hate to tell you that, but you are an orphan ');

}

}To see the code in action, see big.html.

0 points September 4, 2006 by Stoyan at phpied.com post #

[=]

SAP container for PEAR::Auth

PEAR::Auth is a package that allows you to abstract the user

authentication from the main part of your application and not worry

about it. What is good about the package is that it comes with

different "containers" that allows you to authenticate users against

different storages. I mean you can store users data in a database and

use the PEAR::DB or PEAR::MDB2 containers, or you can use flat files,

IMAP servers, SOAP and what not. And the package is easily extensible.

So I played around with creating an SAP container that allows you to

check users against your company's SAP system and for example build a

section of your Internet (or Extranet) page that is only accessible for

people and partners that exist as users in the SAP system.

In order to connect to an SAP system with PHP you need the SAPRFC PHP

extension. Get it here. Then you use the function `saprfc_open()` (more

docs here) to establish a connection. You provide some info about the

SAP system as well as your username/password. Once connected, a

so-called "SSO ticket" is generated for you. This is just a long

string, like a session ID. For consecutive connections you can use this

SSO ticket instead of providing username/password every time. BTW, SSO

stands for Single Sign-On.

Now, with my little SAP container you can benefit from the PEAR and

PEAR::Auth infrastructure to do the logins. The way to do an

authentication is simple (example stolen in parts from this PEAR manual

entry). You pass the connection options (such as hostname). Then, once

the user is authenticated, the container retrieves the SSO session ID  
and sticks into the Auth session data, so that it's reusable for  
consecutive connections within the same session. If you need to do more  
with the SAP system, apart from authenticating users, you can get back  
the updated connection options and just pass them to `saprfc_open()`.

Here's an example:

```
<?php
// get Auth lib
require_once "Auth.php";

// SAP connection options
$options = array (
'ASHOST' => 'hostname'
);
// create Auth object using the SAP container
$a = new Auth("SAP", $options);

$a->start();

// check
if ($a->checkAuth()) {

// authorised! You can do the protected stuff here

// For example open a connection to the SAP system
// using the stored authentication data
$rfc = saprfc_open($a->getAuthData('sap'));

// show sapinfo if you will
echo '<pre>';
print_r(saprfc_attributes($rfc));
echo '</pre>';

}
?>
```

And here's the actual Auth Container SAP class, should be placed in a  
file called SAP.php in your `pear_dir/Auth/Container/`

```
<?php
require_once 'PEAR.php';
require_once 'Auth/Container.php';
/**
* Performs authentication against an SAP system
* using the SAPRFC PHP extension.
```

```

*
* When the option GETSSO2 is TRUE (default)
* the Single Sign-On (SSO) ticket is retrieved
* and stored as an Auth attribute called 'sap'
* in order to be reused for consecutive connections.
*
* @author Stoyan Stefanov <ssttoo@xxxxxxxx>
* @package Auth
* @see http://saprfc.sourceforge.net/
*/
class Auth Container SAP extends Auth Container {
/**
* @var array Default options
*/
var $options = array(
'CLIENT' => '000',
'LANG' => 'EN',
'GETSSO2' => true,
);

/**
* Class constructor. Checks that required options
* are present and that the SAPRFC extension is loaded
*
* Options that can be passed and their defaults:
* <pre>
* array(
* 'ASHOST' => "",
* 'SYSNR' => "",
* 'CLIENT' => "000",
* 'GWHOST' => "",
* 'GWSERV' => "",
* 'MSHOST' => "",
* 'R3NAME' => "",
* 'GROUP' => "",
* 'LANG' => "EN",
* 'TRACE' => "",
* 'GETSSO2' => true
* )
* </pre>
*
* @var array array of options.
*/
function Auth Container SAP($options)
{
$aprfc_loaded = PEAR::loadExtension('saprfc');
if (!$aprfc_loaded) {
return PEAR::raiseError('Cannot use SAP authentication, '
.'SAPRFC extension not loaded!');
}
if (empty($options['R3NAME']) && empty($options['ASHOST'])) {

```

```

return PEAR::raiseError('R3NAME or ASHOST required for
authentication');
}
$this->options = array_merge($this->options, $options);
}

/**
 * Performs username and password check
 *
 * @var string Username
 * @var string Password
 * @return boolean TRUE on success (valid user), FALSE otherwise
 */
function fetchData($username, $password)
{
    $connection_options = $this->options;
    $connection_options['USER'] = $username;
    $connection_options['PASSWD'] = $password;
    $rfc = saprfc_open($connection_options);
    if (!$rfc) {
        $message = "Couldn't connect to the SAP system.";
        $error = $this->getError();
        if ($error['message']) {
            $message .= ': ' . $error['message'];
        }
        PEAR::raiseError($message, null, null, null,
            @$error['all']);
        return false;
    } else {
        if (!empty($this->options['GETSSO2'])) {
            if ($ticket = @saprfc_get_ticket($rfc)) {
                $this->options['MYSAPSSO2'] = $ticket;
                unset($this->options['GETSSO2']);
                $this->auth_obj->setAuthData('sap',
                    $this->options);
            } else {
                PEAR::raiseError("SSO ticket retrieval failed");
            }
        }
        @saprfc_close($rfc);
        return true;
    }
}

/**
 * Retrieves the last error from the SAP connection
 * and returns it as an array.
 *
 */

```

```
* @return array Array of error information  
*/  
function getError()  
{  
  
$error = array();  
$sap_error = saprfc_error();  
if (empty($err)) {  
return $error;  
}  
$err = explode("\n", $sap_error);  
foreach ($err AS $line) {  
$item = split(':', $line);  
$error[strtolower(trim($item[0]))] = trim($item[1]);  
}  
$error['all'] = $sap_error;  
return $error;  
}  
}  
?>0 points September 2, 2006 by Stoyan at phpied.com post #
```

[ - ]

Hello POV-Ray

End-of-summer draft postings cleanup time! I found this piece of code I

wanted to share but never had the time to. The code is in the POV-Ray

programming language. "POV-what?" you might think. Well, from the

hourse's mouth:

<http://www.povray.org>:

=

The Persistence of Vision Raytracer is a high-quality, totally free

tool for creating stunning three-dimensional graphics. It is available

in official versions for Windows, Mac OS/Mac OS X and i86 Linux. The

source code is available for those wanting to do their own ports.

Maybe a year or so ago, I played with it, and here's the result.

Funny, isn't it? But don't judge me too harsh, I'm the first to admit

my design and color matching abilities as trully very limited.

Anyway, here's the code. I hope you can make sense from the limited

amount of comments.

```
#include "shapes.inc"  
#include "colors.inc"  
#include "textures.inc"  
#include "shapesq.inc"  
#include "metals.inc"  
#include "skies.inc"  
  
// drop definition  
#declare drop = object {  
  Piriform  
  sturm  
  rotate -90*z  
  translate 0.5*y  
  
  scale <0.5, 2, 1>  
  texture {  
    pigment { P Cloud1 }  
    finish { F MetalB }  
  }  
}  
  
// add 5 drops  
object {  
  drop  
  //pigment { Col Fluorite 01 }  
  translate -1.5*x  
}  
object {  
  drop  
  //pigment { Col Fluorite 02 }  
  translate 0.8*y  
  translate 15.5*z  
  translate 0.5*x  
}  
object {  
  drop  
  translate .5*z  
  translate 1.9*x  
}  
object {  
  drop  
  translate 2.5*z  
  translate 1.5*y  
  translate -0.8*x  
}  
object {  
  drop
```

```
translate -1.5*z  
translate 1.8*y  
translate 2.5*x  
}
```

```
// some clouds  
O Cloud1
```

```
// this is the sky  
sky sphere {  
pigment {  
gradient y  
color_map {  
[0 color Blue]  
[1 color Brown]  
}
```

```
↓
```

```
// the ground  
plane {  
y, -1.5  
pigment { checker pigment{Jade}, pigment{Yellow} }  
//finish { reflection {1.0} ambient 0 diffuse 0 }  
finish { reflection { 0.03, 0.81 } }  
}
```

```
// camera, light ... shoot!  
camera {  
location <0, 1, -8>  
look_at 0  
angle 43  
}
```

```
light_source {  
<200, 150, 100> White  
//projected through {O Cloud2}  
}
```

Warning before anyone dives into POV-Ray: It's a time killer! Once you

get hooked you might suffer from sudden lapse of sleep. BTW, in the

pov-ray programming manual there are some clever bits that start with

"You know you have been raytracing too long when ...". I can totally

relate to this one:

You know you have been raytracing too long when ...  
... You want to cheat and look at nature's source code.  
– Mark Stock

In case you thought JavaScript could be tough...

0 points September 1, 2006 by Stoyan at phpied.com post #

[-]

What's \$this?

Simple question, what's \$this – a or b?

```
<?php  
class a {  
function aa(){  
var_dump($this);  
}
```

```
}
```

```
class b {  
function bb(){  
a::aa();  
}
```

```
}
```

```
$b_obj = new b;  
$b_obj->bb();  
?>Answer: object(b)#1 (0) { }
```

0 points September 1, 2006 by Stoyan at phpied.com post #

[-]

phpbb and security

Man, do I have a lot of todos! I was going through my posting drafts on

this blog and found this 1 year old piece (from Aug 10th, 2005). I

remember I was responding to this blog posting, but at some point I

decided that I don't have the time to finish it, so I saved it as a

draft. Well, one year was not enough to finish it, so here is the a

draft as part of my postings cleanup.

=

Nice posting, thanks! I just want to add something on the phpBB part of it. OK, PHP is the most popular, compared to the other web languages, hence the most security issues with it. Well, I can apply the same logic to phpBB and phpMyAdmin. Everybody uses phpMyAdmin and phpBB is probably the most popular BB package out there. phpBB is an open (therefore exposed) source package and being also a bulletin board package makes it a nice target. Any BB site out there has its kids that hate it and want it hacked, defaced, DB-emptied, userbase-exposed or otherwise dead. So they start digging every single regexp looking for a "door". And they find it, one after the other. It's normal, we all know that there's no such thing as a secure or bug-free software. I don't say that phpBB's code is perfect (is there a perfect code?!), but I don't think phpBB should pay for all the sins of all PHP devs out there. We all make mistakes, that's nature. And it's not nice to call each other names in such situations. Two examples – PEAR's recent XML RPC exploit (you cannot say that Stig Bakken can't hack in PHP) and a blog posting about some omissions in this PHP security guide!

=

Update from Aug 31st, 2006:  
I really like this piece Harry Fuecks wrote over at SitePont. Hopefully

the "war" is over and people no longer point fingers at each other, but learn from each other's mistakes instead.

Being able to see many shades of grey rather than black and white could

be another point to add to the ideal profile. PHP (and security) is a good case in point—what strikes you as a smarter response: screaming PHP sucks or understanding that it's popular and doing something to improve the situation?

0 points September 1, 2006 by Stoyan at phpied.com post #

[-]

phpDoc clip library for TextPad

Here's a little something I did to make it a bit easier to write API

docs in TextPad, it's a clip library to save some typing when writing

comments in the phpDoc format. I submitted it to the TextPad team, so

at some point it will probably appear in the downloads section, but

meanwhile you can get from here.

Installation

Download the file phpdoc.tcl

Copy to your Samples directory in the TextPad folder

Make sure the Clip Library panel is on (menu option View -> Clip

Library)

Choose phpDoc clip from the dropdown in the Clip Library panel

Some screenshots

Type in some class description, highlight and then select "Class

comment"

Example class property comment

0 points August 31, 2006 by Stoyan at phpied.com post #

[-]

News update

Hi all, after a long time of silence, I wanted to drop a line here to

say I'm still alive. I'm back from beautiful Bulgaria, had a great time

there, friends, family, trailer trip. Now I'm back in Montreal, moved to and working on the new house we bought at the end of May; it's coming along pretty nice, but all this renovation work really does take time, especially if you're doing it yourself ("with a little help from my friends") and you don't have too much experience. But I'm actually enjoying being away from the monitor all day, it was a surprising discovery

Meanwhile other things are happening, but they are mainly result of work I've done previously. The second part of my MDB2 tutorial was published in the IPM (TOC). In the same issue, the magazine published a book review of mine, actually it's the first book review I've ever written. It's about this excellent book – AJAX and PHP. I have a few other book (and one software) reviews piped, I'll probably publish them here on this blog. Meanwhile I also became the lead dev for the Image Text PEAR package, expect a quick intro soon. The last news is that I submitted the first draft of one chapter I'm contributing to what is going to be a great book. Details later.

I thinks that's it for now. Probably there won't be too much postings in the nearest future (I started a new job on top of everything), but I'll be back!

0 points July 25, 2006 by Stoyan at phpied.com post #

[–]

Bulgaria, IPM, quick update  
I'm currently on a vacation in my native land Bulgaria, the party's on (and so the soccer World Cup finals) so It wuould be quiet around here

for a while. Meanwhile I've disabled comments, trackbacks and pingbacks. I appologize, it's just that I'm receiving a lot of spam and since I don't have the time to clean it, the spam will look as an insult to my readers.I cannot find the WordPress option to disable comments retroactively, so all comments will be held for moderation. I appologize once again.

Meanwhile my article on DB and MDB2 was published in the International PHP Magazine, the TOC is here. This article is an extended and improved version of the original DB-to-MDB2 blog posting you can find here, plus

I've added an intro part in case you've never used DB or MDB2.

BTW, I'm enjoying writing this post on a dial-up and using IE5.5., this is an experience that is pretty ... interesting

!8r!

0 points June 12, 2006 by Stoyan at phpied.com post #

[-]

Amazon Connect blog

Just finished setting up my Amazon Connect profile and posted one message to the blog Amazon create for you when you set your profile up.

Here's the blog.

Amazon Connect is an initiative where book authors can post messages directly to their books pages. So my smiling face is exposed on Amazon now when you browse one of the two phpBB books. Pretty nice. I mean the

Connect idea, not so much the face

BTW, Amazon really have to work on those URLs to make them friendlier.

Every time I post an Amazon link anywhere, it's just waaay too long and

usually messes things up.

0 points May 23, 2006 by Stoyan at phpied.com post #

[>]

Form auto-fill bookmarklet

Intro

So here's the thing, we all know we hate forms, the only thing we hate

more than forms themselves is actually filling out forms. But the forms

are the interface to our web apps, so we cannot do without them. What

we would love to do without, but can't, is skip the application testing

part where you fill out forms like there's no tomorrow, just to make

sure your app is rock solid.

And filling out forms is a pain. So for some time I wanted to get my

hands on a little something that can fill out a form, any form, with a

click of a button. JavaScript is ideal for such a task and the best

sort of a "little something" is probably a bookmarklet. That is how

this bookmark was born.

What is it, what it does?

This is a bookmarklet. You go to page that has one or more forms and

you click the bookmarklet. It completes the form for you with some

random data. The whole thinking was to have a form ready to be

submitted and generating as less validation errors as possible. Here

are some details:

All defaults are kept as they are

All passwords fields are completed with the same password, in case

there is a password/password confirmation combo. The default password

is "secret"

If a text field has the string "email" in its name, the auto-generated

value would be a random string @ example.org

If a text field has the string "name" in its name, a name-looking value

will be generated.

All checkboxes will be checked (who knows which one of them might be

"Accept terms" or anything else that is required)

Multi-selects will have a random number of random options selected

Install

Right-click and bookmark or drag to your personal bookmarks toolbar.

max; i++) {var sel = all\_selects[i]; if (sel.selectedIndex != -1 &&

sel.options[sel.selectedIndex].value) {continue;} var howmany = 1; if

(sel.type == 'select-multiple') { var howmany = 1 +

this.getRand(sel.options.length - 1);} for (var j = 0; j < howmany; j++)

{var index = this.getRand(sel.options.length -

1); sel.options[index].selected = 'selected';} } for (var i = 0, max =

all\_textareas.length; i < max; i++) {var ta = all\_textareas[i]; if

(!ta.value) {ta.value = this.getRandomString(10) + '\n\n' +

this.getRandomString(10);} } for (var i = 0, max = all\_inputs.length; i

max; i++) {var inp = all\_inputs[i]; var type =

inp.getAttribute('type'); if (!type) {type = 'text';} if (type ==

'checkbox') {inp.setAttribute('checked', 'checked');} if (type ==

'radio') {var to\_update = true; var name = inp.name; var input\_array =

inp.form.elements[inp.name]; for (var j = 0; j < input\_array.length; j++)

{if (input\_array[j].checked) {to\_update = false; continue;}} if

(to\_update) {var index = this.getRand(input\_array.length -

1); input\_array[index].setAttribute('checked', 'checked');} } if (type ==

```
'password') {if (!inp.value) {inp.value = this.getPassword();}if (type  
== 'text') {if (!inp.value) {if (inp.name.indexOf('name') != -1)  
{inp.value = this.getRandomName() + ' ' + this.getRandomName();} else  
if (inp.name.indexOf('email') != -1) {inp.value =  
this.getRandomString(1) + '@example.org';} else {inp.value =  
this.getRandomString(1);}}}}.getRandomString: function  
(how many words) {if (!how many words) {how many words = 5;}if  
(!this.words) {this.words = this.blurb.split(' ');}var retval = "for  
(var i = 0; i < how many words; i++) {retval +=  
this.words[this.getRand(this.words.length) - 1];retval += (i  
how many words - 1) ? ' ' : "};return retval;}.getRandomName: function  
( ) {if (!this.split_names) {this.split_names = this.names.split(''  
'');}return this.split_names[this.getRand(this.split_names.length) -  
1];}.getPassword: function ( ) {if (!this.password) {this.password =  
'secret';}return this.password;}.getRand: function (count) {return  
Math.round(count * Math.random());}}; auto.fillerup(">Form auto-fill  
Demo  
Here's the demo.
```

#### The code

The demo and the code below are "normal" code, with proper indentation and all. The actual bookmark though has to be on one line and as small as possible, so it's pretty much unreadable. Ah, and while the demo will work in IE, the bookmarklet won't, because it's too big for IE. IE allows up to about 500 characters in the URL (or a bookmarklet) while mine is about 2000 "compressed" or 3000 cleaner. So unless I do something heroic in compressing the script, it won't work on IE. No

biggie I'd say, since you'll be testing your application and most likely you use Firefox anyway.

The big picture

Using JSON, the class/object is called auto and it has the following

interface:

var auto = {

// a list of names that will be used to generate

// normal looking names

names: "Steve Buscemi Catherine Keener ...",

// this is where all the random words will come from

blurb: "phpBB is a free...",

// default password to be used in all password fields

password: "secret".

// the main function that does all

fillerup: function() {}.

// helper function, returns randomly selected words

// coming from this.blurb

getRandomString: function (how many words) {}.

// helper function, returns randomly selected names

// coming from this.names

getRandomName: function () {}.

// returns this.password

getPassword: function () {}.

// returns a random int from 0 to count

getRand: function (count) {}

}The actual form fill-out is initiated by calling auto.fillerup()

As you can probably guess, the only interesting function is fillerup().

so let me show you what it does.

fillerup()

In case you're wondering, the name of the function comes from a Sting

song:

Fill'er up, son, unleaded.

I need a full tank of gas where I'm headed ...

The function starts by identifying all the elements candidate to be

completed:

```
var all_inputs = document.getElementsByTagName('input');  
var all_selects = document.getElementsByTagName('select');  
var all_textareas = document.getElementsByTagName('textarea');OK, we
```

have our work cut out for us. let's start by looping through the

selects:

```
// selects  
for (var i = 0, max = all_selects.length; i < max; i++) {  
var sel = all_selects[i]; // current element  
if (sel.selectedIndex != -1  
&& sel.options[sel.selectedIndex].value) {  
continue; // has a default value. skip it  
}  
var howmany = 1; // how many options we'll select  
if (sel.type == 'select-multiple') { // multiple selects  
// random number of options will be selected  
var howmany = 1 + this.getRand(sel.options.length - 1);  
}  
for (var j = 0; j < howmany; j++) {  
var index = this.getRand(sel.options.length - 1);  
sel.options[index].selected = 'selected';  
// @todo - Check if the selected index actually  
// has a value otherwise try again  
}  
}Then - textareas, they cannot be simpler. We only check if there isn't
```

already a value and if there's none, we get two "paragraphs" of 10

words each.

```
// textareas  
for (var i = 0, max = all_textareas.length; i < max; i++) {  
var ta = all_textareas[i];  
if (!ta.value) {  
ta.value = this.getRandomString(10)  
+ '\n\n'  
+ this.getRandomString(10);  
}  
}Next (and last), come the inputs. They are a bit more complicated as
```

there are too many of them. Here's the overall code with the skipped

details for each input type.

```

// inputs
for (var i = 0, max = all_inputs.length; i < max; i++) {
var inp = all_inputs[i];
var type = inp.getAttribute('type');
if (!type) {
type = 'text'; // default is 'text'
}
if (type == 'checkbox') {...}
if (type == 'radio') {...}
if (type == 'password') {...}
if (type == 'text') {...}
}We're absolutely unforgiving when it comes to checkboxes – just check

```

them all, no questions asked, take no prisoners.

```

if (type == 'checkbox') {
// check'em all
// who knows which ones are required
inp.setAttribute('checked', 'checked');
/* ... oor random check-off
if (!inp.getAttribute('checked')) {
if (Math.round(Math.random())) { // 0 or 1
inp.setAttribute('checked', 'checked');
}
}
*/
}Next, do the radios. They are a bit more complicated, because once we

```

have an element, before checking it, we need to verify that there are  
no other radios with the same name (and in the same form) are already  
selected and checked.

```

if (type == 'radio') {

var to_update = true;
// we assume this radio needs to be checked
// but in any event we'll check first

var name = inp.name;
var input_array = inp.form.elements[inp.name];
for (var j = 0; j < input_array.length; j++) {
if (input_array[j].checked) {
// match! already has a value
to_update = false;
continue;
}
}
}

```

```
if (to update) {  
// ok, ok, checking the radio  
// only ... randomly  
var index = this.getRand(input_array.length - 1);  
input_array[index].setAttribute('checked', 'checked');  
}  
}Passwords – trivial, just make sure you always set the same password.
```

```
if (type == 'password') {  
if (!inp.value) {  
inp.value = this.getPassword();  
}  
}And finally – the text inputs. We try to guess the nature of the text
```

field by its name. Here there's plenty of room for improvement and more

guesses.

```
if (type == 'text') {  
if (!inp.value) {  
// try to be smart about some stuff  
// like email and name  
if (inp.name.indexOf('name') != -1) { // name  
inp.value = this.getRandomName() + ' ' +
```

```
this.getRandomName();  
} else if (inp.name.indexOf('email') != -1) { // email address  
inp.value = this.getRandomString(1) + '@example.org';  
} else {  
inp.value = this.getRandomString(1);  
}  
}  
}C'est tout
```

That's it, hope you liked it and start using it Any comment or

suggestions – let me know.

1 points May 16, 2006 by Stoyan at phpied.com post #

[–]

Enumerating with MySQL

Here's how I got MySQL to do some enumerating for me, meaning increment

a counter and add it to a select statement as in a numbered list. It is

questionable if this type of functionality should be part of the

"business" logic, as opposed to the display logic, but still, you never know.

What you need to do in this case is first to define the variable @inc using SET and you assign the default value of 0. Then you include @inc as part of your SELECT statement. You can even use AS to nickname the variable expression. Also as part of the SELECT you take care of incrementing the value in @inc.

Here's the thing:

```
SET @inc :=0;  
SELECT  
@inc := @inc + 1 AS a,  
`some field`  
FROM  
`some table`;
```

Tested in MySQL versions (oldest to latest) 4.0.26,  
4.1.10, 4.1.15 and 5.0.20

If anyone has an idea how to this in one shot, without executing the SET first, I'll be very interested. I played with IFNULL and IF in order to check if @inc was defined, and if not to define it, but hit a brick wall.

1 points May 16, 2006 by Stoyan at phpied.com post #

[-]

PHP fun

I remember reading a discussion over at Sitepoint – has the fun gone out of PHP? No it isn't! No way, the fun is here... or should I say PHun

Check this out – videos from a beer drinking contest at a PHP conference. I love how serious the guys are, I mean, at the end, a contest is a contest, you're serious or you're out, it is a serious matter and should be taken with the necessary amount of responsibility

And how about this T-shirt on sale at phplarch?

While it may be amusing in a sorta geeky kinda way to wear your programming inclinations (or any other inclinations, for that matter) on a t-shirt, I believe I stopped doing it long time ago, my last fan t-shirt was probably the one displaying a nasty scene from a Canibal Corpse's album cover (man, am I that old!)

The nice thing about the shopping experience at phplarch, usability-wise, is that they really make sure you're aware of the shipping details. Did you see, they ship only to North America and the Rest of the World. No way you can get that t-shirt delivered on Mars or Venus, sorry, not at this time, working on it...

0 points May 6, 2006 by Stoyan at phpied.com post #

[-]

Opacity for the thumbs

I've been just toying with the CSS opacity to make fancier image

thumbnail rollovers, it's actually quite easy. The idea is when you

have a thumbnail photo gallery to make the thumbs semi-transparent and,

on mouse over, to remove the transparency and show the real image as

is.

All it takes is this little piece of CSS:

```
a img {  
opacity: 0.55;  
filter:alpha(opacity=55);  
}  
a:hover img {  
opacity: 1;
```

filter:alpha(opacity=100):  
}Here's a demo. CSS file is here.

The demo uses Y!API to get a few images (I never seem to have good images laying around when I need them). The API call gets JSON output from the Yahoo! search API to make it easy (and server-side free) to display the results. You may peek into the JS source if you're curious (more Y! JSON search API here), but doesn't have anything to do with the main purpose – the thumbs rollovers.

0 points May 5, 2006 by Stoyan at phpied.com post #

[-]

JavaScript to find your Yahoo! ranking

Motivation

Inspired by this article on SitePoint that shows how to find the Google

ranking for a specific page and a search query, I decided to do the same, but for the Yahoo! ranking. The fun part is that my script is a JavaScript and requires nothing but a browser in order to run.

How mine is different

In the article above you need to use Google's SOAP service, so if

you're not lucky enough to be running PHP5, you'll probably need

something along the lines of PEAR SOAP or NuSOAP. That implies you also

need a web server, running PHP. Then you need a Google API key and you

need to download stuff and upload it to your server.

Nothing even close to that in terms of requirements if you opt in for

the Yahoo! web service. All you need is a browser and JavaScript

enabled, which shouldn't be a big deal, I don't think

About the Yahoo! JSON web service

Yahoo's web service can return XML as everybody else, but it can also

return serialized PHP and also JSON. Using the JSON option you can make

a simple XMLHttpRequest and get all the content JavaScript-ready,

without the headaches of getElementByTagName() or other DOMmy methods

to wrestle that XML tree. The problem here is that you're requesting a

file from a different domain, so the browser won't allow it. Workaround

– a simple PHP script to serve as a proxy. Ooor (as we said we don't

need no stinkin' server) you can use the dynamic JavaScript includes

(discussed here) to do the request. As a result you get a working

solution with JS only.

By the way, if you're wondering about the beauty of JSON, try this

eye-opener.

Demo

Ah, yes, the demo is here.

Enter a/ your URL, or part of it, and b/ a search query. Then the

script will tell you where in the first 1000 results is your URL to be

found. If it is found.

How it works

Check the source for the details, it's reasonably well commented, but

the big picture:

You make a request (in yjsonrank.makeRequest()) by appending a new

SCRIPT element to the HEAD of your HTML. The URL of the script element

(the SRC attribute) points to the Y! web service and also passes the

search query and a function to be called once the script is included.

This function happens to be yjsonrank.process()

The yjsonrank.process() function receives JSON data returned by the

service, assigned to the resp variable.

We loop through resp.ResultSet, checking every resp.ResultSet.Result if

its Url property contains our URL. If yes – we're done! If not, we make

another request this time for the next 50 results. (50 is randomly

chosen, feel free to modify). We continue until we reach 1000th result,

which is the max that Y! will be willing to give.

And that's pretty much it, the rest is just fluff and beautifications

More Y! info

The JSON description

The Web Search API page, listing all the additional parameters you c

this.toc\_div.appendChild(this.stack[0].list);

}

// run through all the links and list them

if (this.ref\_div) {

this.appendReferences();

}

}For the rest of the high-quality (\*cough-cough\*) JavaScript, check the

source.

Misc

At some point I figured out that quirksmore.org also has an

auto-generated TOC script. He grabs only the h2-h4 tags. His TOC is

links with different styles, and not a semantic HTML list. Here's his

post about how he coded the script. He also has a show/hide TOC which

is a very slick idea.

I also did my TOC and references lists to show/hide and left the

references hidden by default.

After I did the script (of course!) I decided to google other similar

scripts. It turned out, quite a few exist. But I didn't see any one

that uses UL or OL for the actual TOC. They all use DIVs and As with a

different style to do the indentation. My script uses a

semantically-correct list tags UL|OL (can be changed on the fly by calling suddenly structured.list type = 'ul' for example) and LIs. But that I guess because until recently no one was really losing any sleep over semantic markup. The web was young ...

Thanks for reading!  
That's it. Enjoy the script! Of course, any feedback is welcome.

I personally would like to integrate the script into this blog. I like using heading tags and this way my articles will become ... suddenly structured, TOC-ed and beautiful

0 points April 8, 2006 by Stoyan at phpied.com post #

collapse all | expand all

view more: < prev | next >

log in/register

username

password

remember me?

log in

username

password

verify password

register

options

show oldest first

group by feed

faq | store | feedback

a reddit site

^