Bs_Browscap:
browser and user capturing

Andrej Arn
Sam Blume

blueshoes

WHICH OS AND BROWSER VERSION DOES THE VISITOR USE? DOES HE HAVE COOKIES AND JAVA ENABLED? WHAT SCREEN RESOLUTION DOES HE HAVE, WHAT LANGUAGE DOES HE PREFER, AND WHAT'S HIS TIMEZONE DIFFERENCE TO GMT? ARE FLASH AND ACROBAT INSTALLED? BS_BROWSCAP CAN TELL YOU THIS AND A LOT MORE.

Version:
1.0, updated 2003-10-18 for bs-4.5

Source Location:

class:	core/net/http/Bs_Browscap.class.php
examples:	core/net/http/examples/browscap_simple.php
	core/net/http/examples/browscap_js.php
	core/net/http/examples/browscap_function.php

Finding out the OS, browser vendor and browser version is easy. Parsing the HTTP_USER_AGENT string does this job. For most of the other user information, like finding out the user's time zone, we need to execute JavaScript on the user's machine. To read in all this information, we use a temporary web page on the first hit to our server, store the information in the session, and then redirect to the originally requested page.

This redirect has drawbacks too. So if you don't need the extra information, you're better off with the simple version.

Simple Example:

To see this in action, call http://www.blueshoes.org/_bsCore/net/http/examples/browscap_simple.php

```php
<?php
//include the class:
require_once($APP['path']['core'] . 'net/http/Bs_Browscap.class.php');

//compute all information we can:
$GLOBALS['Bs_Browscap']->compute();

//then spit it out to see what we have:
dump($GLOBALS['Bs_Browscap']->data);
?>
```

Listing 1: also see core/net/http/examples/browscap_simple.php

On blueshoes.org we use the more complex version. These steps are required:

Procedure

1. The user requests mysite.com/foo.html
2. We receive the request, see that we don't have an active session, and thus start a new one.
3. If the requesting "user" is a friendly web spider, for example from google, we send the original (most-backward-compatible) web page directly. If it is an unfriendly spider, for example an email spider, we send it to hell. Otherwise we go on.
4. If there is post data ($_POST) then we store that temporarily.
5. We show the temporary browser capture html page with lots of JavaScript code (Bs_Browscap->runTest ()), this page redirects to the server again automatically.
6. We receive our own automatic redirect from the browser capture page.
7. We have an active session. We let Bs_Browscap work on the user information (Bs_Browscap->compute()), and store that into the session.
8. If we have temporary stored post data, we load that now.
9. Then we process the request as if it was the first one to /foo.html and show the page.

Simplified Complex Example:

This is only here to show you how it works in theory. In real life you need a session, otherwise the browser capturing kicks in on every page request, which is very annoying. Other things described in the procedure above are missing as well.

To see this in action, call http://www.blueshoes.org/_bsCore/net/http/examples/browscap_js.php

```php
<?php
// include the class:
require_once($APP['path']['core'] . 'net/http/Bs_Browscap.class.php');

if (@$_GET['bcRun']) {
    $GLOBALS['Bs_Browscap']->compute();
    dump($GLOBALS['Bs_Browscap']->data);
} else {
    $GLOBALS['Bs_Browscap']->runTest();
}
?>
```

Listing 2: also see core/net/http/examples/browscap_js.php

The global blueshoes function bsSessionBrowscap() from Bs_Misc.lib.php considers all details described in the procedure above. I recommend you call it directly at the end of your global.conf.php file, after having the session started.

Complex Example using bsSessionBrowscap():

To see this in action, call http://www.blueshoes.org/_bsCore/net/http/examples/browscap_function.php

```
<?php
//require dependencies
$GLOBAL_CONF_TINY = TRUE;
require_once($_SERVER['DOCUMENT_ROOT'] . '../global.conf.php');
if (!isSet($GLOBALS['bsSession'])) {
  $APP['sess']['maxStandbyTime'] = 60; //minutes
    $APP['sess']['path']           = $_SERVER['DOCUMENT_ROOT']  . '../session/';
    require_once($APP['path']['core'] .
      'net/http/session/Bs_SimpleSession.class.php');
    $GLOBALS['bsSession'] =& $GLOBALS['Bs_SimpleSession'];
    $GLOBALS['bsSession']->start();
}

bsSessionBrowscap();
?>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head><title>BlueShoes Browscap Example</title></head>
<body>

<a href="<?php echo $_SERVER['PHP_SELF'];?>">link to this page.</a><br>
If you click this link (again and again) you can see that the redirection is not
done again since the session already exists.
<br><br><hr><br>

<?php
dump($GLOBALS['Bs_Browscap']->data);
?>

</body>
</html>
```

Listing 3: also see core/net/http/examples/browscap_function.php

Using this code the whole browser detection works smoothly and transparent. Get and post data is not lost because of the redirect.

To detect web crawlers we use a database from the BlueShoes Knowledge Base.
DB table name: BsKb.CloakList

Even if the KB is not installed, the most important spiders like google are detected. It is important that no browscap redirection is made, otherwise your website won't be correctly indexed with search engines.