



goMobi Traffic Switching Guide

Version 0.9, 28 September 2010

Mobile/Desktop Traffic Switching.....	3
Summary of Options	3
Single Site Solutions	4
Server-side Switching Code	4
JavaScript Switching Code.....	4
Multi-Site Switching Service	5
Redirect Service.....	5
Redirect/Proxy Service	6
DeviceAware	6
Appendix A – Code Samples	7
PHP Code	7
JSP Code	8
ASP.net Code	11
JavaScript Code.....	13

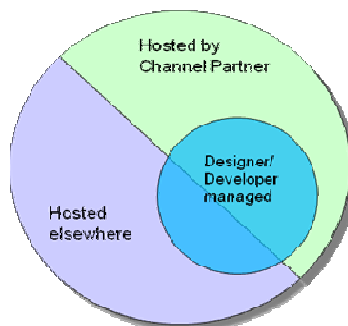
Mobile/Desktop Traffic Switching

goMobi allows you to create a mobile site for your business. In order to make the best use of this mobile-friendly site we recommend that you add functionality to your existing desktop web site to automatically redirect mobile devices to the mobile site. This ensures the mobile users get the best experience even if they type in the address of the existing desktop site.

This document summarizes the traffic switching options open to site owners, registrars and hosters.

Summary of Options

There are multiple different traffic switching options that are open to goMobi resellers, each one with its own benefits and issues. In particular, the best solution to deploy depends on whether the site is hosted by the channel partner or hosted elsewhere, and whether or not a designer/developer is involved:



In general, for traffic switching options to work well, you need access to the original site and server to implement an effective solution. The following table summarizes the options that are open. In all cases we are assuming that there are two web sites with URLs site.com and site.mobi, though the choice of TLD is not important.

Scenario	Environment	Switching Option	Notes
Single web site solution (individual developers)	Access to underlying server available	Embed server-side switching code in site using PHP / JSP or ASP.net.	Good solution, works on all devices
	No access to underlying server	Embed JavaScript switching code in site HTML	Not guaranteed, recommended only as last-ditch option
Multi-site switching service (switching service made available to)	DNS alias for desktop considered acceptable e.g. desktop.site.com	1) Redirect service using DeviceAware that redirects to desktop.site.com or site.mobi as appropriate	Simple solution but relies on an alias e.g. desktop.site.com DNS being in place, which may be a problem.
		2) Redirect/proxy service	Same reliance on

partners)		using DeviceAware that redirects to site.mobi or proxies traffic to desktop.site.com	desktop.site.com alias.
-----------	--	--	-------------------------

These options are described in the following sections.

Single Site Solutions

Server-side Switching Code

Server-side switching code is switching code that is operated co-resident with the existing site. Typically this takes the form of PHP, JSP or ASP code that is built into certain pages on the site.

The switcher code can be implemented as a new default page for the root URL that redirects users to a new home page on the desktop site or to the mobile-friendly URL. Alternatively, switching code can be added to a subset or all pages of the existing site to ensure that mobile users are redirected even if the initial page visited is not the home page.

The code is typically quite simple and merely redirects the user to the mobile-friendly URL if a mobile device is detected. The logic looks like this:

```
define('DA_URI', 'http://detect.deviceatlas.com/query');
$da_json = @file_get_contents(DA_URI . "?User-Agent=" .
urlencode($_SERVER["HTTP_USER_AGENT"]));
$da_results = (array)json_decode($da_json, true);
if ($da_results['mobileDevice']) {
    header('Location: http://mobile.mobi/');
}
```

This code relies on a web-service version of DeviceAtlas to detect the nature of the requesting device. In practice, to make this detection logic run acceptably quickly some local caching is required and is included in the provided code samples. Sample code is available in PHP, JSP and ASP.net. An Apache module version of the same API is also available and allows the switching logic to be implemented in any programming language that can access Apache environment variables.

This switching method results in a very good user experience and does not require DNS aliases such as desktop.site.com nor that the desktop site to respond to these aliases.

Clearly, installing such code requires that the site operator has the ability to gain access to the code that underpins the site in question. From the site owner point of view the difficulty of implementation depends on the nature of the site but some coding skills will be required.

JavaScript Switching Code

This is a variant of the server-side switching code that does not require access to the source code underpinning the site in question. Instead, a JavaScript switcher is invoked by referencing a dynamically-generated JavaScript file hosted by dotMobi within the <head> element of pages on the site. Once the initial page HTML has loaded, the browser will then be redirected to the alternate address, a mobile-friendly URL. From the site owner point of view integration of the

switcher code is straight-forward as the drop-in JavaScript solution is a familiar model. The drop-in code looks like this:

```
<script type="text/javascript"
src="http://detect.deviceatlas.com/redirect.js?d=http://desktop.com&m=http://mobile.mobi">
</script>
```

where `http://desktop.com` is the URL of your desktop site and `http://mobile.mobi` is the URL of your mobile site.

This switching method has the advantage that only HTML changes are required to the site—no changes to the underlying logic are required. The downside is that JavaScript redirection is not guaranteed to work in all cases, especially on older mobile devices.

Multi-Site Switching Service

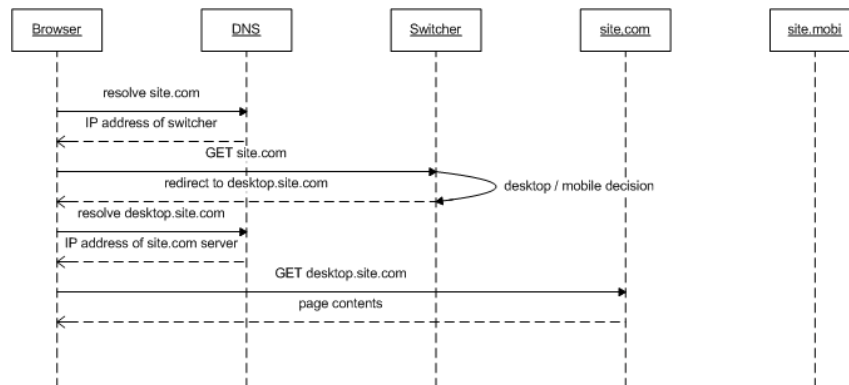
Redirect Service

For hosters and registrars a multi-site switching option is desirable because it enables multiple desktop/mobile site pairs to utilize a single traffic switching solution.

The easiest way to accomplish this a combination of a redirecting service in conjunction with a DNS alias for the existing desktop website. For example, if the existing desktop site is `site.com` and the newly created mobile site is `site.mobi`, the following DNS changes are required:

- modify the A record of `site.com` to point at the switching service IP address
- add an A record for `desktop.site.com` that points at the existing `site.com` web server machine

The following sequence diagram shows what happens when a user with a desktop browser tries to visit `site.com` by typing the URL `http://site.com` into their browser.



In the case where a user enters the URL for the mobile site directly there is no requirement to go via the switcher—the IP address of the `site.mobi` site can be pointed directly at the servers for the mobile site.

There are three issues with this solution:

- a DNS entry must be added for the chosen third-level alias e.g. `desktop.site.com`

- the desktop site (site.com) must be configured to respond to requests for desktop.site.com
- users and stats packages may have an issue with the desktop.site.com URL

Redirect/Proxy Service

A variant of the redirect service is one that, in the case of a desktop browser visiting site.com, proxies the onward traffic to desktop.site.com instead of redirecting. This solution may work better in a hosting environment where the switcher machine and the server for site.com are in fact the same machine i.e. each web server in a farm also has device detection and proxying capabilities built in.

DeviceAware

The switch solutions described in the previous sections rely on dotMobi's DeviceAware software component. DeviceAware is not a packaged solution—it is a software component that is designed to be integrated into existing infrastructure to enable switching applications. DeviceAware combines a database of mobile and desktop device properties with a query API to enable an intelligent switching service to be built that bases its decisions on the nature of the device in question.

The DeviceAware API is available for Java, PHP, .net and C++. In each case the API provides a query method that takes an HTTP User Agent header from an HTTP request and returns a Boolean property indicating of the device is mobile or not. This property can then be used to decide the best course of action to take, which typically means redirecting the browser to an alternate, mobile-friendly, address.

DeviceAware is also available in the form of an Apache module. Enabling this module on an Apache server adds an environment variable to every request received by the server. This environment variable can then be acted upon by subsequently executing modules or Apache configuration rules.

The DeviceAware data file is updated daily by dotMobi and can be downloaded as frequently as desired by customers.

Appendix A – Code Samples

PHP Code

```

<?php
define('MOBILE_SITE', 'http://live.com');
define('DESKTOP_SITE', 'http://google.com');

if($_REQUEST['redirect']=='false') {
    header('Location: ' . DESKTOP_SITE );
    exit;
}

define('DA_USE_COOKIES', true);
define('DA_USE_CACHE', true);
define('DA_CACHE_DIR', sys_get_temp_dir() . DIRECTORY_SEPARATOR . 'DeviceAtlasCache' .
DIRECTORY_SEPARATOR);
define('DA_URI', 'http://detect.deviceatlas.com/query');
$da_results = array('_source' => 'none');

if (DA_USE_COOKIES && isset($_COOKIE['Mobi_Mtld_DA_Properties'])) {
    $da_results = (array)json_decode($_COOKIE['Mobi_Mtld_DA_Properties'], true);
    $da_results['_source'] = 'cookie';
}
if (DA_USE_CACHE && $da_results['_source'] === 'none') {
    $da_cache_file = md5($_SERVER["HTTP_USER_AGENT"]) . '.json';
    if (!file_exists(DA_CACHE_DIR) && !mkdir(DA_CACHE_DIR)) {
        $da_results['_error'] = "Unable to create cache directory: " . DA_CACHE_DIR . "\n";
    } else {
        $da_json = @file_get_contents(DA_CACHE_DIR . $da_cache_file);
        if ($da_json !== false) {
            $da_results = (array)json_decode($da_json, true);
            $da_results['_source'] = 'cache';
            if (DA_USE_COOKIES) {
                setcookie('Mobi_Mtld_DA_Properties', $da_json);
            }
        }
    }
}
if ($da_results['_source'] === 'none') {
    $da_json = @file_get_contents(DA_URI . "?User-Agent=" .
urlencode($_SERVER["HTTP_USER_AGENT"]));
    if ($da_json !== false) {
        $da_results = array_merge(json_decode($da_json, true), $da_results);
        $da_results['_source'] = 'webservice';
        if (DA_USE_COOKIES) {
            setcookie('Mobi_Mtld_DA_Properties', $da_json);
        }
        if (DA_USE_CACHE) {
            if (@file_put_contents(DA_CACHE_DIR . $da_cache_file, $da_json) === false) {
                $da_results['_error'] .= "Unable to write cache file " . DA_CACHE_DIR .
$da_cache_file . "\n";
            }
        }
    } else {
        $da_results['_error'] .= "Error fetching DeviceAtlas data from webservice.\n";
    }
}

if($da_results['mobileDevice']) header('Location: ' . MOBILE_SITE);
else header('Location: ' . DESKTOP_SITE);

?>

```

JSP Code

```

<%@ page import="java.net.*, java.io.*, java.security.*" %>
<%

final String mobileSite = "http://live.mobi";
final String desktopSite = "http://google.com";

if(request.getParameter("redirect")!=null && request.getParameter("redirect")=="false") {
    response.sendRedirect(desktopSite);
}

final String DA_URI = "http://detect.deviceatlas.com/query";
String source = "none";
boolean useCookies = true;
boolean useCache = true;
String error = "";
String daJson = null;
final String DA_CACHE_DIR = System.getProperty("java.io.tmpdir")+
System.getProperty("file.separator")+"DeviceAtlasCache"+System.getProperty("file.separator");
String userAgent = request.getHeader("user-agent");

//Look up cookies
if(useCookies) {
    daJson = getCookieValue("Mobi_Mtld_DA_Properties", request.getCookies());
    if(daJson != null && !daJson.equals("null")) {
        source = "cookie";
    }
}

//Look up cache
if(useCache && source.equals("none")) {
    String filename = DA_CACHE_DIR + md5(userAgent) + ".json";

    File dir = new File(DA_CACHE_DIR);
    if(!dir.exists() && !dir.mkdirs()) {
        error = "Unable to create cache directory: " + DA_CACHE_DIR + "\n";
    }
    else {
        //Read the cache file if it exists
        daJson = readFileAsString(filename);
        if(daJson!=null && !daJson.trim().equals("null") && !daJson.trim().equals("")) source =
"cache";
    }
}

//Look up DA webservice if we need to
if(source.equals("none")) {
    URL DA = new URL("http://detect.deviceatlas.com/query?User-Agent=" +
URLEncoder.encode(userAgent));
    URLConnection conn = DA.openConnection();
    BufferedReader br = new BufferedReader(new InputStreamReader(conn.getInputStream()));
    String line = null;
    line = br.readLine();
    while (line != null) {
        daJson += line;
        line = br.readLine();
    }
    br.close();

    if(null!=daJson) {
        source = "webservice";

        if(useCookies) {
            Cookie cookie = new Cookie("Mobi_Mtld_DA_Properties", daJson);
            response.addCookie(cookie);
        }

        if(useCache) {
            String filename = DA_CACHE_DIR + md5(userAgent) + ".json";

```

```

        writeFile(filename, daJson);
    }
}

String isMobile = parseJson(daJson, "mobileDevice");

//Do the redirect
if(null!=isMobile && isMobile.equals("true")) response.sendRedirect(mobileSite);
else response.sendRedirect(desktopSite);

%>
<%!

//Parse json for property - really basic & assumes flat json file as returned by DA service
static final String parseJson(String json, String property) {
    String[] tokens = json.split(",\\s\"|\\\":|\\{\\}|\\s|,\\s|\\|");
    for(int i=0;i<tokens.length;i++) {
        if(tokens[i].trim().equals(property)) {
            return tokens[i+1].trim();
        }
    }
    return null;
}

//Get MD5 of string
static final String md5(String clearText) {
    byte[] buffer = clearText.getBytes();
    String resultHash = null;
    try {
        MessageDigest md5 = MessageDigest.getInstance("MD5");

        byte[] result = new byte[md5.getDigestLength()];
        md5.reset();
        md5.update(buffer);
        result = md5.digest();

        StringBuffer buf = new StringBuffer(result.length * 2);

        for (int i = 0; i < result.length; i++) {
            int intVal = result[i] & 0xff;
            if (intVal < 0x10) {
                buf.append("0");
            }
            buf.append(Integer.toHexString(intVal));
        }

        resultHash = buf.toString();
    } catch (NoSuchAlgorithmException e) {
    }
    return resultHash;
}

//Read value of a cookie
static final String getCookieValue(String cookieName, Cookie[] cookies) {
    if(null==cookies) return null;
    for(int i=0; i<cookies.length; i++) {
        Cookie cookie = cookies[i];
        if (cookieName.equals(cookie.getName()))
            return(cookie.getValue());
    }
    return null;
}

//Read file as string
static final String readFileAsString(String filePath) {
    byte[] buffer = new byte[(int) new File(filePath).length()];
    BufferedInputStream f = null;
    try {
        f = new BufferedInputStream(new FileInputStream(filePath));
        f.read(buffer);
    }
    catch (Exception e) {}
    finally {
        if (f != null) try { f.close(); } catch (IOException ignored) { }
    }
}

```

```
    return new String(buffer);
}

//Write to file
static final void writeFile(String filePath, String data) {
    Writer writer = null;

    try{
        File file = new File(filePath);
        writer = new BufferedWriter(new FileWriter(file));
        writer.write(data);
    }
    catch (FileNotFoundException e) {}
    catch (IOException e) {}
    finally {
        try {
            if (writer != null) {
                writer.close();
            }
        } catch (IOException e) {}
    }
}

%>
```

ASP.net Code

```

<%@ Page Language="C#" %>
<%@ Import Namespace="System" %>
<%@ Import Namespace="System.IO" %>
<%@ Import Namespace="System.Security.Cryptography" %>
<%@ Import Namespace="System.Net" %>

<%
string mobileSite = "http://live.mobi";
string desktopSite = "http://google.com";

if(Request["redirect"]!=null && Request["redirect"]=="false") Response.Redirect(desktopSite);

string source = "none";
bool useCookies = true;
bool useCache = true;
string daJson = null;
string DA_CACHE_DIR = System.IO.Path.GetTempPath() + "DeviceAtlasCache\\";
string userAgent = Request.UserAgent.ToString();

if(useCookies) {
    HttpCookie cookie = Request.Cookies["Mobi_Mtld_DA_Properties"];
    if(cookie!=null) {
        daJson = cookie.Value.ToString();
        source = "cookie";
    }
}

//Look up cache
if(useCache && source=="none") {
    string filename = DA_CACHE_DIR + md5(userAgent) + ".json";
    if(!Directory.Exists(DA_CACHE_DIR)) Directory.CreateDirectory(DA_CACHE_DIR);
    else {
        //Read the cache file if it exists (assumes just one line of text)
        if(File.Exists(filename)) {
            TextReader tr = new StreamReader(filename);
            daJson = tr.ReadLine();
            tr.Close();
            source="cache";
        }
    }
}

//Look up DA webservice
if(source=="none") {
    string url = "http://detect.deviceatlas.com/query?User-Agent=" +
HttpUtility.UrlEncode(userAgent);
    WebClient client = new WebClient();
    daJson = client.DownloadString( url );

    if(daJson!=null && daJson!="null") {
        if(useCookies) {
            HttpCookie cookie = new HttpCookie("Mobi_Mtld_DA_Properties");
            cookie.Value = daJson;
            Response.Cookies.Add(cookie);
        }

        if(useCache) {
            string filename = DA_CACHE_DIR + md5(userAgent) + ".json";
            TextWriter tw = new StreamWriter(filename);
            tw.WriteLine(daJson);
            tw.Close();
        }
    }
}

string isMobile = parseJson(daJson, "mobileDevice");
if(isMobile=="true") Response.Redirect(mobileSite);
else Response.Redirect(desktopSite);

%>

```

```
<script runat="server" language="C#" >

//Parse json for property - really basic & assumes flat json file as returned by DA service
string parseJson(String json, String property) {
    string[] tokens = Regex.Split(json, "\\s\"|\"|\\:|\\\\{|\\}|\\s|,\\s|\\\\");
    for(int i=0;i<tokens.Length;i++) {
        if(tokens[i].Trim()==property) {
            return tokens[i+1].Trim();
        }
    }
    return null;
}

//Function to create md5 hash of filename
string md5(string clearText) {
    MD5 md5 = MD5.Create();
    byte[] inputBytes = System.Text.Encoding.ASCII.GetBytes(clearText);
    byte[] hash = md5.ComputeHash(inputBytes);

    StringBuilder sb = new StringBuilder();
    for (int i=0;i<hash.Length; i++) {
        sb.Append(hash[i].ToString("X2"));
    }
    return sb.ToString();
}
</script>
```

JavaScript Code

If you do not have access to your server there is one further option that can be used. This option uses JavaScript and hence can be pasted into the <head> section of your pages without having to alter the underlying server code.

Note: this option is not recommended for a few reasons:

- It is not guaranteed to work on all devices, since many mobile phones do not execute JavaScript.
- Depending on the device, the entire HTML and linked resources in the page may be downloaded before the redirect happens, removing much of the benefit of switching to a mobile-friendly page.
- By necessity, the JavaScript must be generated each time, per device. This means that there will always be a delay while the device fetches the JavaScript file from the DeviceAtlas servers.

To use this approach, add the following line to the <head> section of the relevant pages of your site.

```
<script type="text/javascript"
src="http://detect.deviceatlas.com/redirect.js?d=http://desktop.com&m=http://mobile.mobi">
</script>
```

where <http://desktop.com> is the URL of your desktop site and <http://mobile.mobi> is the URL of your mobile site. The resulting JavaScript will then redirect mobile devices to the mobile site but leave desktop browsers at the current URL. This solution has been tested on the following browsers and device ranges:

- Opera Mini 4 and later
- Nokia Series 40 Series 3 and later
- Nokia Series 60 3rd Edition and later
- iPhone (all types)
- Android 2.1 and later

This solution has not been tested on the following classes of devices:

- OpenWave browsers
- Obigo browsers