Handling multiple base currencies with Magento

Magento has the ability to handle distinct base currencies for multiple shops running off one code base. This offers great flexibility in pricing your products for different territories.

Using a well known online e-commerce store as an example (Apple.com) this article will demonstrate the benefits of choosing Magento commerce as your online shopping platform.

This example assumes that you serve customers in 3 global locations. You serve:

1. U.S. customers in U.S. Dollars (USD)
2. UK customers in Sterling (GBP)
3. European customers in Euros (Euro)

When it comes to approaching the architecture of an international store, some firms will have a large array of international domain extensions at their disposal such as .co.uk, .us, eu, .com, etc. However, this approach is not always possible. This could be due to a lack of available domain extensions or maybe you just don't want to force your customers remember different domain names for different regions.

Utilizing a single domain name could be the answer to your problems.

To this end, my demo store will use the domain example.com as the main website serving the most important customer base - U.S. customers and all other websites will be mapped to a folder within the root of store.com. So the three websites we are using are:

1. example.com - This website uses USD as its base currency. Products are displayed in USD and conversions are mapped to USD.
2. example.com/uk - This website uses GBP as its base currency. Products are displayed in GBP and conversions are mapped to GBP.
3. example.com/eu - This website uses Euro as its base currency. Products are displayed in EURO and conversions are mapped to EURO.

To achieve this kind of architecture within Magento we first need to set-up 3 websites under system->manage stores. Next, assign stores and store views to the websites for each locale. You should end up with something like:

Website Name	Store Name	Store View Name
Example North America	USD Store	USD Store 001
Example Europe	Euro Store	Euro Store 001
Example United Kingdom	GBP Store	GBP Store 01

Next go to: system->configuration->catalogue and click on the tab labelled: 'Price'. At this point you will want to set the price scope to 'Website'. This will allow you to use individual prices and currencies per website.

Next go to: system->configuration->currency set-up and assign the base currency and display currency you want to use as the default current option. Please note products will use the default base currency unless you provide alternatives. Next configure the individual websites to use distinct base currencies. To do this simply change the 'configuration scope' drop down menu to a specific website such as

'Example Europe' you will then need to set a new base currency and display currency for your store - set both options to Euro, remembering to uncheck the 'use default' check box. Next do the same for the UK website but remember to use British Pound Sterling for both the base and display currencies. Please note: you can choose which currencies to accept by using the control key and scrolling through the list of accepted currencies - it's a b little finicky but you will get there eventually.

The final step of setting up your currency options will be to load up some currency exchange values. Go to: system->manage currency rates and click the import button at the top of the screen. Hopefully, if you have installed Magento correctly you should see a list of your default base currencies with your accepted currencies listed as numerical exchange values.

Next go to: system->configuration->web and use the configuration scope drop down to select 'Example Europe'. This will allow you to set a base URL for the website. So go ahead and set the base URL for 'Example Europe' to http://example.com/eu/ click save and then do the same for the UK website (/uk/).

Setting up the server

This section of the *Handling multiple base currencies with Magento* tutorial can be extended to many different scenarios and is not limited to handling base currencies but can also be used as a guide to setting up a staging (development) / live environment that allows you to switch between versions of Magento. This is great for seamlessly upgrading Magento on the fly, or testing a new Beta version of Magento without affecting your live site.

Symlinks

The most important part of the Magento multi website server set up is using Symlinks (symbolic references) to hook into the Magento core folders and files.

The following server directory structure uses the multi-currency example.com website to illustrate the idea:

/server-root/public_html/magento-live – This folder contains the 'live' version of Magento core.
/server-root/public_html/example.com/ - Example.com is pointed at this folder.
/server-root/public_html/example.com/uk/ - example.com/uk is the root folder for your UK site.

Firstly, you need to create symlinks within the example.com/ folder and the example.com/uk/ folder to point to the Magento core folders.

I use SSH for this using the MAC OSX terminal. Here are the commands I would use to link the folders that need to be linked. an example symlink:

```
cd /server-root/public_html/example.com/

ln -s /server-root/public_html/magento-live/app ./app
ln -s /server-root/public_html/magento-live/includes ./includes
ln -s /server-root/public_html/magento-live/js ./js
ln -s /server-root/public_html/magento-live/lib ./lib
ln -s /server-root/public_html/magento-live/media ./media
ln -s /server-root/public_html/magento-live/report ./report
ln -s /server-root/public_html/magento-live/skin ./skin
ln -s /server-root/public_html/magento-live/var ./var
```

This will create linked folders that are similar to 'shortcuts' on a desktop PC that reference the Magento core folders.

You will need to create symlinks for each folder / website you use with Magento.

Now you have created symlinks to Magento core you will need to make copies of the Magento core .htaccess and index.php files and place them in the example.com and example.com/uk/ folders. This will instruct Magento to treat your newly created sub-directories as stand alone websites.

.htaccess

Open up your FTP client and navigate to the Magento core folder (/server-root/public_html/magento-live) and download both the .htaccess and index.php files.

Server Rewrites

Next, edit the .htaccess file by making the following edits:

Uncomment and change the line that reads: #RewriteBase / to RewriteBase / - delete the hash. This will mean that all requests to / will be treated as the base URL. This edit applies to example.com.

For the uk sub-folder you would change this line to:

```
RewriteBase /uk/
```

Site-maps

If you are using site-maps you can provide a simple way to include them from your core Magento installation like so:

```
RewriteRule ^sitemap.xml/?$ /pathtoyourmagentofolder/sitemaps/us/sitemap.xml
```

This means that all requests to example.com/sitemap.xml will be re-written to the /pathtoyourmagentofolder/sitemaps/us/sitemap.xml path.

Fooman Speedster

If you are using Fooman speedster to minify your CSS and JS files (highly recommended) then you can include something like the following to include the Fooman minification scripts:

```
############################################ ## Compress, Combine and Cache Javascript/CSS
```

```
RewriteRule ^(index.php/)?minify/([^/]+)(/.*.(js|css))$ pathtoyourmagentofolder/lib/minify/m.php?
f=$3&d=$2
```

Done editing .htaccess

You are now done editing the .htaccess file so upload it to the /server-root/public_html/example.com/ and /server-root/public_html/example.com/uk/ folders. Remembering to use the correct RewriteBase rule for each folder.

Index.PHP

Now you have created the symlinks and edited the ./htaccess file for each website / folder you can edit the index.php file that is the Magento bootstrap – the file that initializes Magento.

You will need an index.php file for each domain and 2 simple edits need to be completed to include Magento core and your new currency specific website.

Find the lines:

```
$mageRunCode = isset($_SERVER['MAGE_RUN_CODE']) ? $_SERVER['MAGE_RUN_CODE'] : '';
$mageRunType = isset($_SERVER['MAGE_RUN_TYPE']) ? $_SERVER['MAGE_RUN_TYPE'] : 'store';
```

And edit these to something like the following:

```
$mageRunCode = isset($_SERVER['MAGE_RUN_CODE']) ? $_SERVER['MAGE_RUN_CODE'] :
'websitecode';
$mageRunType = isset($_SERVER['MAGE_RUN_TYPE']) ? $_SERVER['MAGE_RUN_TYPE'] : 'website';
```

In the above example, websitecode would be the code used to define your website as set within the Magento administration interface under system->manage stores.

Conclusion

After you have successfully completed the above steps you are now ready to add some products with locale based pricing. So go ahead and create a product and when you have finished adding some details such as a product description simply click into the prices tab on the product edit screen and firstly add a price in USD, click and save and edit, next change the top left drop down menu labelled 'choose your store view' to another website's store view such as the GBP Store 001 and enter a price in GBP next click save and edit again. Magento may display the price with a label of [USD], don't pay attention to this as it is a bug that has been reported by myself so hopefully this screen will be more intuitive to use in a forthcoming release. Repeat this step for the EURO store view and click save and edit. Finally revert back to the default store view and assign the product to all of your websites and include the product in a category.

If everything went as planned you should now have three separate websites accessed using example.com, example.com/eu and example.com/uk with separate base currencies for each website.