Screen Scraping Your Way Into RSS

Introduction

RSS is one the hottest technologies at the moment, and even big web publishers (such as the New York Times) are getting into RSS as well. However, there are still a lot of websites that do not have RSS feeds.

If you still want to be able to check those websites in your favourite aggregator, you need to create your own RSS feed for those websites. This can be done automatically with PHP, using a method called screen scrapping. Screen scrapping is usually frowned upon, as it's mostly used to steal content from other websites.

I personally believe that in this case, to automatically generate a RSS feed, screen scrapping is not a bad thing. Now, on to the code!

Getting the content

For this article, we'll use PHPit as an example, despite the fact that PHPit already has RSS feeds (http://www.phpit.net/syndication/).

We'll want to generate a RSS feed from the content listed on the frontpage (http://www.phpit.net). The first step in screen scraping is getting the complete page. In PHP this can be done very easily, by using implode(file("", " the url here ")); IF your web host allows it. If you can't use file() you'll have to use a different method of getting the page, e.g. using the CURL library (http://www.php.net/curl).

Now that we have the content available, we can parse it for the content using some regular expressions. The key to screen scraping is looking for patterns that match the content, e.g. are all the content items wrapped in 's or something else? If you can successfully discover a pattern, then you can use preg match all() to get all the content items.

For PHPit, the pattern that match the content is Content Here . You can verify this yourself by going to the main page of PHPit, and viewing the source.

Now that we have a match we can get all the content items. The next step is to retrieve the individual information, i.e. url, title, author, text. This can be done by using some more regular expression and str replace() on the each content items.

By now we have the following code;

```
<?php

// Get page
 url = "http://www.phpit.net/";
 data = implode("", file( url));

// Get content items
preg match all ("/<div class= "contentitem ">(     ?)< /div>/",  data,  matches);
```

Like I said, the next step is to retrieve the individual information, but first let's make a beginning on our feed, by setting the appropriate header (text/xml) and printing the channel information, etc.

```
// Begin feed
header ("Content-Type: text/xml; charset=ISO-8859-1");
echo "<?xml version= "1.0 " encoding= "ISO-8859-1 " ?> n";
?>
```

```
<rss version="2.0"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:content="http://purl.org/rss/1.0/modules/content/"
  xmlns:admin="http://webns.net/mvcb/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns ">
 <channel>
  <title>PHPit Latest Content</title>
  <description>The latest content from PHPit (http://www.phpit.net), screen scraped!</description>
  <link>http://www.phpit.net</link>
  <language>en-us</language>


<?
```

Now it's time to loop through the items, and print their RSS XML. We first loop through each item, and get all the information we get, by using more regular expressions and preg match(). After that the RSS for the item is printed.

```
<?php
// Loop through each content item
foreach ( matches 0  as  match) {
 // First, get title
 preg match ("/ ">(     ?)< /a>< /h3>/",  match,  temp);
  title =  temp '1' ;
  title = strip tags( title);
  title = trim( title);

 // Second, get url
 preg match ("/<a href= "(     ?) ">/",  match,  temp);
  url =  temp '1' ;
  url = trim( url);

 // Third, get text
 preg match ("/<p>(     ?)<span class= "byline ">/",  match,  temp);
  text =  temp '1' ;
  text = trim( text);

 // Fourth, and finally, get author
 preg match ("/<span class= "byline ">By (     ?)< /span>/",  match,  temp);
  author =  temp '1' ;
  author = trim( author);

 // Echo RSS XML
 echo "<item> n";
  echo " t t t<title>" . strip tags( title) . "</title> n";
  echo " t t t<link>http://www.phpit.net" . strip tags( url) . "</link> n";
  echo " t t t<description>" . strip tags( text) . "</description> n";
  echo " t t t<content:encoded><! CDATA   n";
  echo  text . " n";
  echo "   ></content:encoded> n";
  echo " t t t<dc:creator>" . strip tags( author) . "</dc:creator> n";
 echo " t t</item> n";
}
?>
```

And finally, the RSS file is closed off.

</channel>
</rss>

That's all. If you put all the code together, like in the demo script, then you'll have a perfect RSS feed.

Conclusion

In this tutorial I have shown you how to create a RSS feed from a website that does not have a RSS feed themselves yet. Though the regular expression is different for each website, the principle is exactly the same.

One thing I should mention is that you shouldn't immediately screen scrape a website's content. E-mail them first about a RSS feed. Who knows, they might set one up themselves, and that would be even better.

Download sample script at http://www.phpit.net/viewsource.php?url=/demo/screenscrape 20rss/example.php

Short note about the author

Dennis Pallett is a young tech writer, with much experience in ASP, PHP and other web technologies. He enjoys writing, and has written several articles and tutorials. To find more of his work, look at his websites at http://www.phpit.net, http://www.aspit.net and http://www.ezfaqs.com

---