Last week I published two articles using custom loops. The first was about [how to create a custom loop](http://weblogtoolscollection.com/archives/2008/04/13/define-your-own-wordpress-loop-using-wp_query/). The second was [how to retrieve posts based on custom fields](http://weblogtoolscollection.com/archives/2008/04/13/how-to-only-retrieve-posts-with-custom-fields/). In both articles, several readers commented that they would like to see paging and have it explained.

I’d like to thank Aaron Harun from [Anthology of Ideas](http://anthologyoi.com/) for supplying the code used in this post.

**Paging, and why it doesn’t work with WP\_Query**

The paging magic happens in a file called ‘**link-template.php**‘ in the ‘**wp-includes**‘ folder.

Most themes have basic paging built in, with the help of two functions: **next\_posts\_link** and **previous\_posts\_link**.

These functions, as well as several others, make use of a global variable called **$wp\_query**. This variable is actually an instance of the **WP\_Query** object. However, when we create our own instantiation of **WP\_Query**, we don’t have this global variable **$wp\_query**, which is why paging will not work.

**How to make paging work with a custom loop**

As Aaron mentioned, the paging functions rely on a global variable called **$wp\_query**. The “fix” is to trick WordPress into using the global **$wp\_query** variable when using our own custom loops.

In the example below, I’ll show how to display the last five recent articles with our own custom loop with paging intact.

<h3>Recent Articles</h3> <ul> <?php $temp = $wp\_query; $wp\_query= null; $wp\_query = new WP\_Query(); $wp\_query->query('showposts=5'.'&paged='.$paged); ?>

The first thing the code does (shown above) is create a temporary variable to hold a reference to global variable **$wp\_query**. The **wp\_query** variable is set to null, and we instantiate a new **WP\_Query** object.

Notice in the query above that we have these parameters:

$wp\_query->query('showposts=5'.'&paged='.$paged);

The paged portion of the query is critical in order to have paging.

From there, we can now do our own query and begin our custom loop.

<?php while ($wp\_query->have\_posts()) : $wp\_query->the\_post(); ?> <li><a href="<?php the\_permalink() ?>" rel="bookmark"><?php the\_title(); ?></a></li> <?php endwhile; ?> </ul>

Notice in the above code that the loop is initiated using the **$wp\_query** variable. Since we are using **$wp\_query**, paging should work.

Now it’s time to create our navigation links for paging:

<div class="navigation"> <div class="alignleft"><?php previous\_posts\_link('&laquo; Previous') ?></div> <div class="alignright"><?php next\_posts\_link('More &raquo;') ?></div> </div>

And finally, we assign the variable **$wp\_query** back its original value using the **$temp** variable we set earlier.

<?php $wp\_query = null; $wp\_query = $temp;?>

**The Full Code**

<h3>Recent Articles</h3> <ul> <?php $temp = $wp\_query; $wp\_query= null; $wp\_query = new WP\_Query(); $wp\_query->query('showposts=5'.'&paged='.$paged); ?> <?php while ($wp\_query->have\_posts()) : $wp\_query->the\_post(); ?> <li><a href="<?php the\_permalink() ?>" rel="bookmark"><?php the\_title(); ?></a></li> <?php endwhile; ?> </ul> <div class="navigation"> <div class="alignleft"><?php previous\_posts\_link('&laquo; Previous') ?></div> <div class="alignright"><?php next\_posts\_link('More &raquo;') ?></div> </div> <?php $wp\_query = null; $wp\_query = $temp;?>