**Mastering WP\_Query: Using the Loop**

掌握WP\_Query：教你使用Loop循环

As I outlined in the [introduction](http://code.tutsplus.com/tutorials/mastering-wp_query-an-introduction--cms-23023) to this series, the [WP\_Query](http://codex.wordpress.org/Class_Reference/WP_Query) class has four main elements:

* the arguments for the query, using parameters which will be covered in detail in this series
* the query itself
* the loop, which will output post content, titles or whatever you want to display
* finishing off: closing if and while tags and resetting post data

In this tutorial I'll show you how to use the loop with WP\_Query, including the two main ways to structure your loop and how to use multiple loops.

正如我们在入门指引里面提到的， [WP\_Query](http://codex.wordpress.org/Class_Reference/WP_Query)类包括四个部分：

* 需要查询的参数，或者参数集合。
* 开始查询。
* 循环输出——这将输出文章内容、标题或任何你想显示的内容。
* 查询结束——通过标签判断、重置请求数据。

在这篇博文里面，我们将要学习如何使用Loop循环，包括：两种主要的Loop循环结构和多重的Loop循环结构。

## Where the Loop Fits

Without a [loop](http://codex.wordpress.org/The_Loop), nothing will be displayed on your page. After WordPress has run the query, using the arguments you've defined, it then needs to be told what to output from the data that it's fetched. This is where the loop comes in.

So the loop comes after your query, and it uses three tags:

* if( $query->have\_posts() ) checks if there are any posts.
* while( $query->have\_posts() ) repeats the loop for each post as long as there are posts to retrieve.
* $query->the\_post() accesses that specific post.

哪里适合使用Loop循环。

如果没有Loop循环，页面上不会显示任何东西。当Wordpress运行了查询机制之后（当然啦，在这之前，需要设定好参数），接着需要告诉查询程序哪些数据是需要展现的。这里就是需要Loop循环的地方。

所以，在查询之后，循环就来了（Bang，孩儿们，俺老孙来也！猴年，祝大家新年快乐）。Loop循环会使用如下三个标签：

* if( $query->have\_posts() ) 检查是否有文章。
* while( $query->have\_posts() ) 循环往复的执行Loop循环，去检索是否有文章。
* $query->the\_post() 获取到特定的文章。

So this is where the loop fits in the WP\_Query class:

以下是Loop循环在WP\_Query类中的位置：

<?php

$args = array(

    // 给予Query查询定义的参数

);

// 定义一个新的查询

$query = new WP\_Query( $args );

// 判断我们所查询到的结果.

if ( $query->have\_posts() ) {

    // 开始循环往复的查询结果.

    while ( $query->have\_posts() ) {

        $query->the\_post();

        // 查询到的文章内容.

    }

}

// 重置请求数据.

wp\_reset\_postdata();

?>

After running the loop, all that's left to do is tidy things up using wp\_reset\_postdata().

小伙伴们别忘记了，当循环结束后，要使用wp\_reset\_postdata().清理程序（这个过程类似360的释放内存）

## Taking the Loop Further: Checking For Content

Loop循环进阶教程：检查文章内容

But sometimes you might want to add a heading before your list of posts, or you might want to enclose them all in a containing element. If you simply added this before your loop, it would be output regardless of whether the query actually returned any data, meaning you could have a heading with nothing beneath it, or some unnecessary markup.

有的时候，可能需要给所有的文章加一个标题清单，或者是把它们都附到一个容器元素里面。如果只是简单的使用上面的Loop循环代码，它可能不管查询得到的任何数据——意思是，在标题的后面看不到任何数据，或者是不需要的乱码。

This is very easy to get around by putting the enclosing element or the heading inside your if tag:

简单来做是：把需要附到Loop循环里的代码和标题代码放到if 标签里面去。

<?php

$args = array(

    //给予Query查询定义的参数

);

//定义一个新的查询

$query = new WP\_Query( $args );

//判断我们所查询到的结果.

if ( $query->have\_posts() ) {

    echo '<section class="clear">';

        echo '<h2>' . \_\_( 'Heading', 'tutsplus' ) . '</h2>';

        //开始循环往复的查询结果.

        while ( $query->have\_posts() ) {

            $query->the\_post();

            ?>

            <article id="post-<?php the\_ID(); ?>" <?php post\_class( 'left' ); ?>>

                <a href="<?php the\_permalink(); ?>" title="<?php the\_title\_attribute(); ?>">

                    <?php post\_thumbnail( 'thumbnail' );?>

                </a>

                <a href="<?php the\_permalink(); ?>" title="<?php the\_title\_attribute(); ?>">

                    <?php the\_title(); ?>

                </a>

                <?php the\_excerpt(); ?>

            </article>

            <?php

        }

    echo '</section>';

}

//重置请求数据.

wp\_reset\_postdata();

?>

Here you can see I've checked if there are any posts retrieved by my query, and if there are I've opened a containing element and added a heading.

由上可以看出，在每次查询到新的文章的时候，会添加一个开放标签的容器（<section class="clear">）和一个标题（<h2>' . \_\_( 'Heading', 'tutsplus' ) . '</h2>）。

This is also useful if you want to output the results of your query as a list. Let's say I want to create a list of all posts in a given category. The ul element isn't inside my loop as it doesn't relate to one specific post, but I only want to output it if there are posts. So I use this:

这种方法同样适用，通过查询展示文章列表。如果某同学说，我想在一个分类中创建所有文章的列表。  ul 元素不在Loop循环中，也不在文章中，但是需要在有文章的地方去输出 ul 。那么可以如下做：

<?php

$args = array(

    'category\_name' => 'category-slug',

    'post\_type' => 'post'

);

//定义一个新的查询

$query = new WP\_Query( $args );

//判断我们所查询到的结果.

if ( $query->have\_posts() ) {

    echo '<ul class="category posts">';

        //开始循环往复的查询结果.

        while ( $query->have\_posts() ) {

            $query->the\_post();

            ?>

            <li <?php post\_class( 'left' ); ?>>

                <a href="<?php the\_permalink(); ?>" title="<?php the\_title\_attribute(); ?>">

                    <?php the\_title(); ?>

                </a>

            </li>

            <?php

        }

    echo '</ul>';

}

//重置请求数据.

wp\_reset\_postdata();

?>

This checks if the query has fetched any posts, and if so, it opens the ul element and then runs the loop.

当查询检索到任何文章的时候，它会添加 ul 元素，并开始Loop循环检索，添加，检索，添加…..

## Running Extra Loops

It's important to be aware that while you can use WP\_Query to run more than one loop, you have to reset the post data and start a second instance of WP\_Query to do this. This is because each of your loops will be outputting data based on different arguments.

This example displays the excerpt and featured image for the first post and then just the title of each subsequent post:

运行附加的Loop循环

可以很清楚的看到使用 WP\_Query 可以创建多个Loop循环，必须每次重置请求数据，并重新创建一个新的 WP\_Query 实例。这是因为每个Loop循环都会由不同的参数来输出数据。

以下的例子展示了头文章使用引用和特色图片，但后面的文章只展示标题：

<?php

// 定义查询1的参数

$args1 = array(

    'post\_type' => 'post',

    'posts\_per\_page' => '1'

);

// 自定义查询1

$query1 = new WP\_Query( $args1 );

// 判断是否有文章

if ( $query1->have\_posts() ) {

    //开始循环往复的查询结果.

    while ( $query1->have\_posts() ) {

        $query1->the\_post();

        ?>

        <article id="post-<?php the\_ID(); ?>" <?php post\_class(); ?>>

            <a href="<?php the\_permalink(); ?>" title="<?php the\_title\_attribute(); ?>">

                <?php post\_thumbnail( 'thumbnail' );?>

            </a>

            <a href="<?php the\_permalink(); ?>" title="<?php the\_title\_attribute(); ?>">

                <?php the\_title(); ?>

            </a>

            <?php the\_excerpt(); ?>

        </article>

        <?php

    }

}

// 重置请求数据

wp\_reset\_postdata();

// 定义查询2的参数

$args2 = array(

    'offset' => '1',

    'post\_type' => 'post'

);

// 自定义查询2

$query2 = new WP\_Query( $args2 );

//判断是否有文章

if ( $query2->have\_posts() ) {

    echo '<ul class="more-posts">';

        //开始循环往复的查询结果.

        while ( $query2->have\_posts() ) {

            $query2->the\_post();

            ?>

            <li <?php post\_class(); ?>>

                <a href="<?php the\_permalink(); ?>" title="<?php the\_title\_attribute(); ?>">

                    <?php the\_title(); ?>

                </a>

            </li>

            <?php

        }

    echo '</ul>';

}

// 重置请求数据

wp\_reset\_postdata();

?>

I've used two key arguments here:

* 'posts\_per\_page' => '1', used with the first query, outputs just the most recent post.
* 'offset' = '1', used with the second query, skips the first post, ensuring it's not repeated in the list below.
* As you can see from the code above, the loop is slightly different for each query. The first one outputs the featured image, title and excerpt, while the second checks if the query has posts and if so, opens a ul element and encloses each post title in a li element and a link to its page.

You'll also notice that I used wp\_reset\_postdata() after both loops. If I hadn't done this, the second loop would still output data from the first.

这里使用到的两个参数：

* 'posts\_per\_page' => '1',第一个查询所用的是为了输出最新的一篇文章。
* 'offset' = '1',第二查询中，是为了跳过第一篇文章，确保在列表中不会重复。
* 上面的代码中，loop循环之间有细微的差别。第一个循环输出特色图片，标题和引用，而第二个循环会判断是否有文章，并在有文章的时候，添加一个ul元素，在每个篇文章li元素和链接后添加一个闭合标签。

注意到第一个循环后使用的wp\_reset\_postdata()，那是为了避免第二个循环输出重复的数据（与第一个循环重复）。

## Summary

Without a loop, WP\_Query doesn't really do very much. The loop is the code you use to display the data that WordPress has fetched from the database based on your query arguments.

As I've demonstrated, there are a few variations on the loop. A simple loop will just output all posts in the order you've specified in your query arguments (or by date in descending order by default). If you separate if( $query->have\_posts() ) and while( $query->have\_posts() ), you can insert additional markup outside your loop, but only if your query has returned data. And finally, by specifying alternative arguments and using wp\_reset\_postdata() after each loop, you can use WP\_Query more than once to create multiple loops on your page.

小结

在循环中，wp\_query做的事情不多。Loop循环是用来展示数据的——你来定义需要查询的参数，并由Wordpress查询数据库，最后Loop循环展示。

正如例子代码，可以有很多种变化的Loop循环。一个简单的Loop循环只是按序输出了文章，如果分离 if( $query->have\_posts() ) 和 while( $query->have\_posts() ),那可以插入附加的标签到Loop循环（当然啦，前题是Query查询到数据了）。最后需要注意的是，Loop循环传递不同的参数的时候使用wp\_reset\_postdata()，这样就可以在同一个页面上新建多个Loop循环（多重循环）。

译者：痴空见观 链接：wayearn.com