

Find the Parent Post ID and Children ID's of Multiple Nested Posts or Pages – WordPress Tip

- http://technerdia.com/573_post-ancestors.html

Find the Parent Post ID from a Child Page, when that Child Page is multiple nested levels deep.

Google is filled with countless Wordpress users asking *How to find the Parent Post ID from a Child Page*, when that Child Page is multiple nested levels deep. And like those users, I recently found myself asking this same question.

Unfortunately this is one question that has more answers, solutions, and headaches than even Google can understand. All the solutions I did find did not recursively go through endless children posts, the rest simply didn't work.

So I kept looking and I'm glad that I did....

Even though this is still a popular topic today, it appears Wordpress solved this problem way back in version 2.5, with the function [get_post_ancestors](#).

The **get_post_ancestors** function creates an array of Post ID's, starting on the first Child Page. So if you're 50 nested pages and on the 50th page, this function would display every post id, for every page, up to the parent page.

Note: This function does not display the Parent Post ID IF you are on the Parent Post Page itself - You must be at least one level deep for it to work.

Examples of get_post_ancestors

```
/* Example use of get_post_ancestors */
/*Outside Loop: like the sidebar, footer, etc. */
$ancestors = get_post_ancestors( $wp_query->post );
```

```
/*Within Loop: Pages template. */
$postid = get_the_ID();
$ancestors = get_post_ancestors( $postid );
```

The *results of get_post_ancestors* is now in an Array named, \$ancestors. For development purposes you may want to view the results of the array.

```
/* Display $ancestors */
echo "<pre>";
print_r($ancestors);
echo "</pre>";
```

```
/* Output of $ancestors */
Array
(
    [0] => 824
    [1] => 822
    [2] => 796
)
```

In the example above, the Parent Post ID would be: 796 - Or the last value in the array.

You can access the unique value by calling it directly: This would only work if you are two levels deep from the parent.

```
echo $ancestors[2];
```

PHP Function in_array

In the example below, we use the PHP function [in_array](#) to look at all the values returned in the Array, rather than calling each value, like above.

```
/* http://php.net/manual/en/function.in-array.php */
in_array( "796", $ancestors );
```

Example Use: In the example below, if ID 796 is found in the \$ancestors Array, the if statement would echo: Found In Array

```
/* Full example use of get_post_ancestors */
$ancestors = get_post_ancestors( $wp_query->post );
if( in_array( "796", $ancestors ) ){ echo "Found In Array"; }
```

Practical Use

I use the `get_post_ancestors` function within the sidebar.php file of a few product related websites. When the parent page or child page of that parent is loaded, a unique sidebar & widget is called based on the ID. The widget contains a unique, yet typical list style menu that contains nested items within the list.

Example Nested List:

```
<ul>
  <li><a href="#">Parent Page</a>
  <ul>
    <li><a href="#">Child Page 1</a>
    <ul>
      <li><a href="#">Sub-Child Page 2</a>
      <ul>
        <li><a href="#">Sub-Sub-Child Page 3</a></li>
      </ul></li>
    </ul></li>
    <li><a href="#">Sub-Child Page 4</a></li>
  </ul></li>
</ul></li>
</ul>
```

Example Use In Sidebar

This will display the custom sidebar-widget menu on the Parent Page and all Child Pages of that Parent.

```
/* get_post_ancestors in the sidebar */
if( function_exists('dynamic_sidebar') ) {
  $ancestors = get_post_ancestors( $wp_query->post ); /* Build Ancestors Array*/?>
  <div id="sidebar">
  <?php if( is_page(796) || in_array( "796", $ancestors ) ){ dynamic_sidebar(Sidebar1); }?>
  </div>
```

The `is_page` function: This function checks to see if the Parent Page is being shown. If you only wanted the custom menu to display on the Child pages, remove `is_page()` from the above if statement.

Custom sidebar.php file using get_post_ancestors

A more complete example of using the `get_post_ancestors` function within the sidebar.php file.

```
/* get_post_ancestors in the sidebar */
<?php if( function_exists('dynamic_sidebar') ) {
$ancestors = get_post_ancestors( $wp_query->post ); /* Build Ancestors Array*/
?>
<div id="sidebar">
<?php /* sidebars */
if( is_home() ){
dynamic_sidebar(Sidebar1); /* home menu */
}else{
if( is_page(796) || in_array( "796", $ancestors ) ){ dynamic_sidebar(CustomMenu); } /* widget */
// if( is_page(###) || in_array( "###", $ancestors ) ){ dynamic_sidebar(Side-Bar-Name); } /* example widget */
}

if( !is_home() ){ dynamic_sidebar(Sidebar2); } /* non-home mneu */
?>
</div>
} /* end dynamic_sidebar */
?>
```

It's that simple... this method can be used throughout a theme, simply make sure you're passing the *post id* into `get_post_ancestors`, and everything will work perfectly!

[Follow techNerdia on Google+](#) | [Follow Chris on Google+](#)
[Subscribe to techNerdia's RSS Feed](#)

Copyright 2012 © [techNerdia](#) - All Rights Reserved
<http://techNerdia.com/>