TUTORIAL DE NETTUTS.COM

So How Do I Use This Thing?

There are three things you need to do in your index.php file before being able to use your new paginator class.

1. First, include the paginator class in the page where you want to use it. I like to use require_once because it ensures that the class will only be included once and if it can’t be found, will cause a fatal error.
2. Next, make your database connections.
3. Finally, query your database to get the total number of records that you'll be displaying.

Step three is necessary so that the paginator can figure out how many records it has to deal with. Typically the query can be as simple as SELECT COUNT(*) FROM table WHERE blah blah blah.

You're almost there. Now it's time to create a new paginator object, call a few of its methods, and set some options. Once you have your total record count from step three above you can add the following code to index.php:

```
1. $pages = new Paginator;
2. $pages->items_total = $num_rows[0];
3. $pages->mid_range = 9;
4. $pages->paginate();
5. echo $pages->display_pages();
```

Let's break it down...

- The first line gives us a shiny new paginator object to play with and initializes the default values behind the scenes.
- The second line uses the query we did to get the total number of records and assigns it to our paginator's items_total property. $num_rows is an array containing the result of our count query (you could also use PHP's mysql_num_rows function to retrieve a similar count if you like).
- The third line tells the paginator the number of page links to display. This number should be odd and greater than three so that the display is symmetrical. For example if the mid range is set to seven, then when browsing page 50 of 100, the mid range will generate links to pages 47, 48, 49, 50, 51, 52, and 53. The mid range moves in relation to the selected page. If the user is at either the low or high end of the list of pages, it will slide the range toward the other side to accommodate the position. For example, if the user visits page 99 of 100, the mid range will generate links for pages 94, 95, 96, 97, 98, 99, and 100.
- The fourth line tell the paginator to get to work and paginate and finally the fifth line displays our page numbers.

If you decide to give your visitors the option of changing the number of items per page or jumping to a specific page, you can add this code to index.php:

```
1. echo "<span style="\"margin-left:25px\""> ". $pages->display_jump_menu()
2. . $pages->display_items_per_page() . "</span>";
```

If you stopped here and viewed your page without adding anything else, you'd see your page numbers but no records. Aha! We haven't yet told PHP to display the specific subset of records we want. To do that you create a SQL query that includes a LIMIT statement that the paginator creates for you. For example, your query could look like:

```
1. SELECT title FROM articles WHERE title != '' ORDER BY title ASC $pages->limit
```

$pages->limit is critical in making everything work and allows our paginator object to tell the query to fetch only the limited number of records that we need. For example, if we wanted to see page seven of our data, and we're viewing 25 items per page, then $pages->limit would be the same as LIMIT 150,25 in SQL.

Once you execute your query you can display the records however you like. If you want to display the page numbers again at the bottom of your page, just use the display_pages method again:

```
1. echo $pages->display_pages();
```

More Features

As an added bonus, the paginator class adds "Previous", "Next", and "All" buttons around your page links. It even disables them when you're on the first and last pages respectively when there are no previous or next pages. If you like, you can also tell your visitors that they're viewing page x of y by using the code:

```
1. echo "Page $pages->current_page of $pages->num_pages";
```

Styling

Finally, you're free to customize the look of your pagination buttons as much as you want. With the exception of the current page button, all other page buttons have the CSS class "paginate" applied to them while the current page button has, can you guess, the "current" class applied to it. The "Previous" and "Next" buttons will also have the class "inactive" applied to them automatically when they're not needed so you can style them specifically. Using these three classes along with other CSS gives you tremendous flexibility to come up with a variety of styling choices.

Styled:

« Previous 1 … 59 60 61 62 63 64 65 66 67 … 82 Next » All

Unstyled:

« Previous 1 … 5 6 7 8 9 10 11 12 13 … 164 Next » All

Wrapping Up

As you've seen with the pagination class and just a few lines of code you can tame those giant database lists into manageable, easy to navigate, paginated lists. Not only are they easier on the eyes but they're faster not only to load but to browse. Feel free to checkout a couple of demos at example 1 and example 2.