Responsive example Class control

You can tell Responsive what columns to want to be visible on different devices through the use of class names on the columns. The breakpoints are horizontal screen resolutions and the defaults are set for common devices:

* desktop x >= 1024px
* tablet-l (landscape) 768 <= x < 1024
* tablet-p (portrait) 480 <= x < 768
* mobile-l (landscape) 320 <= x < 480
* mobile-p (portrait) x < 320

You may leave the -[lp] option from the end if you wish to just target all tablet or mobile devices. Additionally to may add min-, max- or not- as a prefix to the class name to perform logic operations. For example not-mobile would cause a column to appear as visible on desktop and tablet devices, while min-tablet-l would require at least a horizontal width of 768 for the browser window to be shown, and be shown at all sizes larger.

Additionally, there are three special class names:

* all - Always display
* none - Don't display as a column, but show in the child row
* never - Never display
* control - Used for the column [responsive.details.typeR](http://datatables.net/extensions/responsive/reference/option/responsive.details.type) option.

Please [refer to the Responsive manual](http://datatables.net/extensions/responsive/) for further details of these options.

This example shows the salary column visible on a desktop only - office and age require a tablet, while the position column requires a phone in landscape or larger. The name column is always visible and the start date is never visible.

This can be useful if you wish to change the format of the data shown on different devices, for example using a combination of mobile and not-mobile on two different columns would allow information to be formatted suitable for each device type.

| Name | Position | Office | Age | Start date | Salary | Extn. |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Position | Office | Age | Start date | Salary | Extn. |

* Javascript
* HTML
* CSS
* Ajax
* Server-side script

The Javascript shown below is used to initialise the table shown in this example:

$(document).ready(function() { $('#example').DataTable( { "ajax": "../../../../examples/ajax/data/objects.txt", "columns": [ { "data": "name" }, { "data": "position" }, { "data": "office" }, { "data": "age" }, { "data": "start\_date" }, { "data": "salary" }, { "data": "extn" } ] } ); } );

In addition to the above code, the following Javascript library files are loaded for use in this example:

* [../../../../media/js/jquery.js](http://docs.google.com/media/js/jquery.js)
* [../../../../media/js/jquery.dataTables.js](http://docs.google.com/media/js/jquery.dataTables.js)
* [../../js/dataTables.responsive.js](http://docs.google.com/js/dataTables.responsive.js)

The HTML shown below is the raw HTML table element, before it has been enhanced by DataTables:

This example uses a little bit of additional CSS beyond what is loaded from the library files (below), in order to correctly display the table. The additional CSS used is shown below:

The following CSS library files are loaded for use in this example to provide the styling of the table:

* [../../../../media/css/jquery.dataTables.css](http://docs.google.com/media/css/jquery.dataTables.css)
* [../../css/dataTables.responsive.css](http://docs.google.com/css/dataTables.responsive.css)

This table loads data by Ajax. The latest data that has been loaded is shown below. This data will update automatically as any additional data is loaded.

The script used to perform the server-side processing for this table is shown below. Please note that this is just an example script using PHP. Server-side processing scripts can be written in any language, using [the protocol described in the DataTables documentation](http://datatables.net/manual/server-side).

Other examples

[**Basic initialisation**](http://docs.google.com/initialisation/index.html)

* [Class name](http://docs.google.com/initialisation/className.html)
* [Configuration option](http://docs.google.com/initialisation/option.html)
* [`new` constructor](http://docs.google.com/initialisation/new.html)
* [Ajax data](http://docs.google.com/initialisation/ajax.html)
* [Default initialisation](http://docs.google.com/initialisation/default.html)

[**Styling**](http://docs.google.com/styling/index.html)

* [Bootstrap styling](http://docs.google.com/styling/bootstrap.html)
* [Foundation styling](http://docs.google.com/styling/foundation.html)
* [Vertical scrolling](http://docs.google.com/styling/scrolling.html)
* [Compact styling](http://docs.google.com/styling/compact.html)

[**Display control**](http://docs.google.com/index.html)

* [Automatic column hiding](http://docs.google.com/auto.html)
* [Class control](http://docs.google.com/classes.html)
* [Assigned class control](http://docs.google.com/init-classes.html)
* [With FixedHeader](http://docs.google.com/fixedHeader.html)
* [Complex headers (rowspan / colspan)](http://docs.google.com/complexHeader.html)

[**Child rows**](http://docs.google.com/child-rows/index.html)

* [Disable child rows](http://docs.google.com/child-rows/disable-child-rows.html)
* [Column controlled child rows](http://docs.google.com/child-rows/column-control.html)
* [Column control - right](http://docs.google.com/child-rows/right-column.html)
* [Whole row child row control](http://docs.google.com/child-rows/whole-row-control.html)
* [Custom child row renderer](http://docs.google.com/child-rows/custom-renderer.html)

Please refer to the [DataTables documentation](http://www.datatables.net) for full information about its API properties and methods.

Additionally, there are a wide range of [extras](http://www.datatables.net/extras) and [plug-ins](http://www.datatables.net/plug-ins) which extend the capabilities of DataTables.

DataTables designed and created by [SpryMedia Ltd](http://www.sprymedia.co.uk) © 2007-2015

DataTables is licensed under the [MIT license](http://www.datatables.net/mit).