Responsive Web Tables

Mirza Kabiljagic, Stefan Rajinovic, Aleksandar Stojicic, Inti Gabriel Mendoza Estrada

Institute of Interactive Systems and Data Science (ISDS),
Graz University of Technology
A-8010 Graz, Austria

05 Dec 2019

Abstract

Responsive techniques are techniques used in web design which enable better rendering of web pages, tables, and other elements. With better rendering, it is meant as a rescaling of the resolution of the device used to open the specific website. When using responsive techniques, it doesn't matter which device is used to look at a website, the website should look good.

It is mostly a combination of fluid grids, which enables the elements of a page to be sized in relative units, such as percentages, rather than using pixels or points, flexible images, same thing as in fluid grids, media queries, which enable using many CSS tricks or style rules and responsive layout which can automatically adapt, resize, and adjust to any device screen size, be it a laptop, mobile phone or tablet.

Responsive tables are tables which use the approach mentioned above, which allow them to be more flexible and complex, without compromising its usability while adding a better aesthetically pleasing look. In this survey report, we will provide a good overview of what responsive techniques, responsive tables, and good table designs are, and, finally, recommend the best technique combination, in our opinion, to use.

Contents

Contents i

List of Figures iii

List of Listings v

1 Introduction 1
- 1.1 How to achieve Responsive Web Design . . . 1
- 1.2 Media queries . . . 1
- 1.3 Fluid Grid Layouts . . . 2
- 1.4 Flexible images and media . . . 3
  - 1.4.1 Relative measurements . . . 3
  - 1.4.2 Cropping . . . 4

2 HTML Tables 7
- 2.1 Structure of Tables . . . 7

3 Good Table Design 9
- 3.1 Alternate row highlighting . . . 9
- 3.2 Current Row Highlighting . . . 9
- 3.3 Expandable Areas . . . 10
- 3.4 Pagination (With Sort and Search) . . . 12

4 Responsive Tables 15
- 4.1 Horizontal Scroll . . . 15
- 4.2 Fixed Header . . . 17
- 4.3 Flip Scroll . . . 20
- 4.4 User Resizeable Columns . . . 20
- 4.5 Long Two Column . . . 21

5 Recommendation 25
- 5.1 Recommendation . . . 25

Bibliography 27

List of Figures

1.1 Relative Image Dimensions . . . 3
1.2 Picture with original size . . . 4
1.3 Picture when cropped . . . 4
1.4 Image Cropping . . . 4

3.1 Alternate row highlighting . . . 10
3.2 Current Row Highlighting . . . 11
3.3 Expandable Areas . . . 11
3.4 Expandable Areas . . . 13
3.5 Pagination with sort option . . . 13
3.6 Pagination with search option . . . 14

4.1 Before reponsive . . . 16
4.2 After reposnsive . . . 17
4.3 Horizontal Scroll . . . 18
4.4 Fixed Header . . . 19
4.5 Flip Scroll . . . 21
4.6 User Resizeable Columns using only HTML and CSS . . . 21
4.7 User Resizeable Columns using JS (jQuery) . . . 22
4.8 Long Two Column . . . 24

5.1 Recommendation Techniques 1 . . . 26
5.2 Recommendation Techniques 2 . . . 26
5.3 Recommendation Techniques 3 . . . 26

List of Listings

Chapter 1

Introduction

Going back, at the starting point of web development, its creators and designers have not had the need to think about the look and design of the page, because back then, their users and clients would mostly access the site via the same device, that being a personal desktop computer which would mostly share the same resolution and display size. That means that they mostly needed to design and work around the same concept. Also, they didn't need to think about things like what if an user entered the site with this device or this, so they could basically use static and fixed positioning of elements.

However, todays' technologies, be it the mobile industry or technical industry in general, the devices have come a long way, and are being worked on really fast. What does that mean is that the programmers and designers need to consider that people will have different ways of using and browsing their pages, which require different approaches to design their web pages. Nowadays, the clients and users have a broad list of devices, which they can use to access a web page, be it a tablet, mobile phone, laptop and even TVs [ITU.int 2015].

Basically, the most important thing is to make the web pages' user interface as friendly as possible, so that the user does not lose much time navigating around. And with that being said, web programmers need to automatically think about making the web pages in that way, while they are being accessed from those different devices mentioned [Sharki and Fisher 2013].

Responsive Web Design is an approach, or more rather a technique, which is used to construct a self-adapting web page, meaning that it adapts, resizes, shrink and changes its' form in any way needed to perfectly fit the web-accessing device used to visit it [Negi 2018].

1.1 How to achieve Responsive Web Design

If a programmer wants his web site to be responsive, he has to use following techniques:

- Media queries
- Flexible images and media
- Fluid grid layouts

1.2 Media queries

Media queries are very important because with them it is possible to specify with what device the web page is being accessed. The query usually is made of a media type, and some conditions which can check if the media type has some special property [Mohamed 2015].

```
% An example of a media query, where one can say that for a screen of size 600px or
    less, the
% background color is olive:

@media screen and (max-width: 600px) {
  body   {
    background-color: olive;
  }
}
```

```
% An example of using display: none:
<!DOCTYPE html>
<html>
  <head>
    <style>
      h1.none {
        display: none;
      }
    </style>
  </head>

  <body>
    <h1 class="none">This will not be displayed</h1>
  </body>
</html>
```

Basically it allows defining specific CSS rules, depending on the device used. From the example in **??**, with the help of the parameter `screen` the media type was specified and using `max-width` the condition was determined which affects the device of the user. Now, there are many different conditions which can be used, and they are really helpful, because with them it is possible to adjust display settings for every device there is.

For example, if the user is entering the website with his phone, it can be automatically said that an image should have a lower resolution, thus adjusting it better for his screen and saving some bandwith.

Another example would be using `display:none` with which some content can completely be hidden that should not be displayed.

If this Listing **??** was ran in the browser, the header would not be seen, because it was decided to not be shown. If `display:none` was changed to `display:block`, our header would be visible.

Include a list of `all` the relevant papers and resources you have found and mark those you have chosen to focus on. Make sure `all` the papers and resources you found or were given appear in the bibliography.

1.3 Fluid Grid Layouts

Normally, websites are constructed in a layout which is grid-built, because they are easier to handle in different kind of devices. Also what is specific for that kind of layout is that it is built using divs, HTML tables and so on. Now, on top of that responsive web design comes in play[Frain 2015]. With it, it is possible to use percentage-based element sizing which is easier than using pixels.

According by (Abdulrehman,2015) a flexible grid-layout is one of the foundations of responsive design. Using the term "grid" does not mean necessarily that a grid framework needs to be used. What it means

Figure 1.1: Relative Image Dimensions. [Taken from research paper.]

is that specific CSS tricks are used for positioning. Also, he suggest that using pixels for the measurement unit should be stopped, because pixels can vary.

For example, on one device they can be one point or dot, on another device it can be a few more, and therefore unreliable. So, what needs to be done is that, if pixels are used, they should be converted using a formula stated by (Marcotte, 2010) in Dan Cederholm's book named "Handcrafted CSS":

$$Target \div Context = Result \tag{1.1}$$

It is asserted by (Pettit, 2012) that, to use this formula, the context of the element needs to be divided by the target element.

1.4 Flexible images and media

With this approach, any type of media,images depending on the screen resolution will resize,collapse,crop and adjust accordingly to it, and to achieve this it is possible to use the same thing mentioned in the previous section, relative units, if the screen becomes too small then hide the image completely or by cropping some parts of the image, again in the case if the screen becomes too small[Tutorial no date].

1.4.1 Relative measurements

Instead of using pixels,relative dimensions can be used, percentage based. So instead of specifying the view and dimensions in pixels, one can specify for example 60%, which would result in resizing or reshrinking of the image based on the resolution of the device accordingly to fill 60% of the page.

Figure 1.2: Picture with original size

Figure 1.3: Picture when cropped

Figure 1.4: Image cropping. [https://alligator.io/css/cropping-images-object-fit/.]

1.4.2 Cropping

Another option is cropping, which crops the picture to a certain width with the help of CSS.

In Figure 1.4, we see the result of tweaking the image a bit with just a few CSS commands. It enables us to crop the picture. If someone is browsing the website with a smaller device, the image does not have to be hidden, just its size is changed.

Taken from [https://medium.com/@elad/how-to-crop-images-with-css-b8471d402b16.]

```
% An example of a media query, where one can say that for a screen
% of size 850px or less, the background is black:
figure{
  width:18.75rem; /*container-width*/
  overflow:hidden; /*hide bounds of image */
  margin:0;   /*reset margin of figure tag*/
}

figure img{
  display:block; /*remove inline-block spaces*/
  width:100%; /*make image streatch*/
}
```

Chapter 2

HTML Tables

In case of having too much data to display, or data which is inherently better in a grid, HTML tables are the best solution available. Before, tables have been used mostly for the layout of HTML websites (and is now an example of bad website development pracices) mainly due to two reasons:

- Semantically, it is wrong.
- Tables aren't as adaptable and flexible as divs'

2.1 Structure of Tables

The basic structure of an HTML table starts with the `<table>` tag. It is the starting point for constructing a table. Now, HTML tables consist of columns and rows, like normal tables. For rows, the tag `<tr>` is used, whereas for table header `<th>` tag is used. Normally, table headers are positioned in the center and are bold. For table cells `<td>` tag is used [Wood no date].

Now, there exist more tags which can be used for HTML5 tables:

- `<thead>` - Table header, it is used to point out single or multiple rows of a table, which do not contain table data but column labels [Coyier 2018].
- `<tbody>` - Table body, it is used to point out `<tr>` elements. Position this tag always after `<thead>`, but it can also come after or before `<tfoot>` [Coyier 2018].
- `<tfoot>` - Table footer, it is used to point out single or multiple `<tr>` elements where those elements are presenting an overview of the data in the table [Coyier 2018].
- `<caption>` - Table caption, as the name already says, can be used to specify table caption. Can be put on the bottom of the CSS document.
- `<col>` - While using col and some other keyword, for example, align, it is possible direct the alignment of text in the table. There are other keywords whom can be used to adjust colors, width and many other things of table columns.

An example HTML table code can be seen in Listing **??**.

```
% An example of an HTML Table which demonstrates information about cars:
<!DOCTYPE html>
<html>
<head>
  <title>Best Cars 2019</title>
</head>
<body>
<table class="table table-bordered table-hover table-condensed">
  <thead>
    <tr>
      <th title="Field #1">Car</th>
      <th title="Field #2">Manufacturer</th>
      <th title="Field #3">Engine Size</th>
      <th title="Field #4">Cylinders</th>
      <th title="Field #5">Horsepower</th>
      <th title="Field #6">Torque</th>
      <th title="Field #7">Compresion Ratio</th>
      <th title="Field #8">Miles per gallon</th>
      <th title="Field #9">Price</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>2019 Acura RDX</td>
      <td>Acura</td>
      <td>2.00L</td>
      <td align="right">4</td>
      <td align="right">272</td>
      <td align="right">280</td>
      <td>9.8:1</td>
      <td align="right">28</td>
      <td>€33,600.00</td>
    </tr>
    <tr>
      <td>2019 Ford Ranger</td>
      <td>Ford</td>
      <td>2.30L</td>
      <td align="right">4</td>
      <td align="right">270</td>
      <td align="right">310</td>
      <td>10.0:1</td>
      <td align="right">21</td>
      <td>€21,800.00</td>
    </tr>
  </tbody>
</table>

</body>
</html>
```

Chapter 3

Good Table Design

We will discuss certain conventions that are widely regarded as “Good Table Design”. These conventions are generally neat little CSS tricks which serve to aesthetically improve the look of a table and in most cases also add to the effectiveness and efficiency of tables. All tables must incorporate at least one of the following conventions.

3.1 Alternate row highlighting

When presented a table with a lot of entries, it can be hard to look at. Scrolling through numerous rows can be frustrating. With this CSS trick, it can be a bit easier, at least for the eyes, if nothing else. The idea is to color every even row, while leaving the odd ones intact. Implementing this techniques requires only two lines of CSS, and is objectively useful. This can be seen in Listing **??**.

An example HTML table with this technique is shown in Figure 3.1.

3.2 Current Row Highlighting

Just like in Alternate Row Highlighting, ‘walking’ through a big table one may easily become lost and wouldn’t know in which row he is at the moment, which can be pretty stressful and overall decrease the efficiency of the table.

With the help of some CSS, the lives of the users are made easier. This code can be seen in Listing 3.2

```
% An example of using simple CSS to highlight table rows:
table {
  overflow: hidden;
}

tr:hover {
  background-color: #ffa;
}

td, th {
  position: relative;
}

td:hover::after, th:hover::after {
  content: "";
  position: absolute;
  background-color: #ffa;
  left: 0;
  width: 100%;
  z-index: -1;
}
```

An example HTML table with this technique is shown in Figure 3.2.

```
% An example of using simple CSS to color table rows:
table.alt tr:nth-child(even) {
  background: #CCC
}

table.alt tr:nth-child(odd) {
  background: #FFF
}
```

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compresion Ratio	Miles per gallon	Price(€)
2019 Acura RDX	Acura	2.0	4	272	280	9.8:1	28	33,600.00
2019 Ford Ranger	Ford	2.3	4	270	310	10.0:1	21	21,800.00
2019 Genesis G70	Genesis	2.0	4	252	260	10.0:1	22	31,300.00
2019 GMC Sierra 1500	GMC	4.0	6	285	305	11.0:1	16	26,600.00
2019 Honda Passport	Honda	3.5	6	280	262	11.5:1	21	28,700.00
2019 Jaguar I-Pace	Jaguar	0.0	0	394	513	0.0:1	0	62,400.00
2020 Mercedes-Benz GLE 450	Mercedes-Benz	3.0	4	362	369	10.5:1	22	54,900.00
2019 Nissan Altima	Nissan	2.5	4	188	180	8.0:1	32	21,500.00
2019 Ram 1500	Ram	5.7	8	395	410	11.3:1	23	28,600.00

Figure 3.1:
[Screenshot taken by the author.]

3.3 Expandable Areas

Should we have any further information available about a table's object, making the row clickable and showing the extra information is a better alternative than trying to cram everything inside the table. In the code snippet shown in Listing **??** we see how additional data in an HTML table can be implemented [Alligator 2019].

To implement this table feature the combination of JavaScript and CSS needs to be implemented. In the snippet codes in Listings 3.3 and **??**, the JavaScript and CSS code can be reviewed.

```
<script type="text/javascript">
$(document).ready(function(){

  $("#report tr:odd").addClass("odd");
  $("#report tr:not(.odd)").hide();
  $("#report tr:first-child").show();

  $("#report tr.odd").click(function () {
    var trToToggle = $(this).next("tr");
    $("#report tr:not(.odd)").not(trToToggle).hide();
    $("#report tr:first-child").show();
    $(trToToggle).toggle();
    $(this).find(".arrow").toggleClass("up");
  });
})
</script>
```

An example HTML table with this technique before a row is clicked is shown in Figure 3.3.

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compresion Ratio	Miles per gallon	Price(€)
2019 Acura RDX	Acura	2.0	4	272	280	9.8:1	28	33,600.00
2019 Ford Ranger	Ford	2.3	4	270	310	10.0:1	21	21,800.00
2019 Genesis G70	Genesis	2.0	4	252	260	10.0:1	22	31,300.00
2019 GMC Sierra 1500	GMC	4.0	6	285	305	11.0:1	16	26,600.00

Figure 3.2:
[Screenshot taken by the author.]

```
<tr>
  <td>2019 Acura RDX</td>
  <td>Acura</td>
  <td style="text-align: right">2.0</td>
  <td style="text-align: right">4</td>
  <td style="text-align: right">272</td>
  <td style="text-align: right">280</td>
  <td style="text-align: right">9.8:1</td>
  <td style="text-align: right">28</td>
  <td style="text-align: right">33,600.00</td>
</tr>
<tr>
  <td colspan="5">
    <h4>Additional information about the car</h4>

    <ul>
      <li><a href="https://en.wikipedia.org/wiki/Acura_RDX">Acura RDX</a>
      </li>
      <li><a href="https://www.acura.ca/rdx">Acura RDX official webpage</a>
      </li>
    </ul>
  </td>
</tr>
<tr>
```

Expandable Areas

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compresion Ratio	Miles per gallon	Price(€)
2019 Acura RDX	Acura	2.0	4	272	280	9.8:1	28	33,600.00
2019 Genesis G70	Genesis	2.0	4	252	260	10.0:1	22	31,300.00
2019 Honda Passport	Honda	3.5	6	280	262	11.5:1	21	28,700.00
2020 Mercedes-Benz GLE 450	Mercedes-Benz	3.0	4	362	369	10.5:1	22	54,900.00
2019 Ram 1500	Ram	5.7	8	395	410	11.3:1	23	28,600.00
2019 Toyota Rav4	Toyota	2.5	4	219	184	13.0:1	39	22,900.00
2019 Ford Ranger	Ford	2.3	4	270	310	10.0:1	21	21,800.00

Figure 3.3:
[Screenshot taken by the author.]

```
#report {
  border-collapse:collapse;
}
#report h4 {
  margin:0rem;
  padding:0rem;
}
#report img {
  float:right;
}
#report ul {
  margin:0.625rem 0 0.625rem 2.5rem;
  padding:0rem;
}
#report th {
  background:#7CB8E2  repeat-x scroll center left;
  color:#fff;
  padding:0.4375rem 0.9375rem;
  text-align:left;
}
#report td {
  background:#C7DDEE none repeat-x scroll center left;
  color:#000;
  padding:0.4375rem 0.9375rem;
}
#report tr.odd td {
  cursor:pointer;
}
#report div.arrow {
  background:transparent url(arrows.png) no-repeat scroll 0rem -1rem;
  width:1rem;
  height:1rem;
  display:block;
}
#report div.up {
  background-position:0rem 0rem;
}
```

An example HTML table with this technique after a row is clicked is shown in Figure 3.4.

3.4 Pagination (With Sort and Search)

The amount of data increases every second, over 2.5 quintillion bytes of data are made every day and there is also estimation that in 2020 will be created 1.7MB of data in every second for each person in the world[Ahmad 2018]. Tables are one of the best mediums for displaying large sets of data efficiently. Features can be added to tables to increase efficiency even more. For example, if a table is too long, it can be divided into ‘pages’. Each page houses a certain amount of rows. The user then can ‘flip’ through these pages as page navigation is also implemented.

Furthermore, it is possible to sort the table by column (in ascending/descending and alphabetical order), and search for an element in the table. This solution works for following browsers:

- Google Chrome
- Mozzila Firefox
- Internet Explorer
- Opera
- Microsoft Edge

For the implementation plug-in for the jQuery Javascript library called DataTables is used [no date]. This can be seen in Listing 3.4.

Expandable Areas

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compresion Ratio	Miles per gallon	Price(€)
2019 Acura RDX	Acura	2.0	4	272	280	9.8:1	28	33,600.00
Additional information about the car • Acura RDX • Acura RDX official webpage								
2019 Genesis G70	Genesis	2.0	4	252	260	10.0:1	22	31,300.00
2019 Honda Passport	Honda	3.5	6	280	262	11.5:1	21	28,700.00
2020 Mercedes-Benz GLE 450	Mercedes-Benz	3.0	4	362	369	10.5:1	22	54,900.00

Figure 3.4:
[Screenshot taken by the author.]

Show 10 entries Search:

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compresion Ratio	Miles per gallon	Price(€)
2019 Nissan Altima	Nissan	2.5	4	188	180	8.0:1	32	21,500.00
2019 Ford Ranger	Ford	2.3	4	270	310	10.0:1	21	21,800.00
2019 Toyota Rav4	Toyota	2.5	4	219	184	13.0:1	39	22,900.00
2019 GMC Sierra 1500	GMC	4.0	6	285	305	11.0:1	16	26,600.00
2019 Ram 1500	Ram	5.7	8	395	410	11.3:1	23	28,600.00
2019 Honda Passport	Honda	3.5	6	280	262	11.5:1	21	28,700.00
2019 Subaru Ascent	Subaru	2.4	4	260	277	10.6:1	22	28,700.00
2019 Volvo XC40	Volvo	2.0	4	248	258	10.8:1	26	30,300.00
2019 Genesis G70	Genesis	2.0	4	252	260	10.0:1	22	31,300.00
2019 Acura RDX	Acura	2.0	4	272	280	9.8:1	28	33,600.00

Showing 1 to 10 of 12 entries Previous 1 2 Next

Figure 3.5:
[Screenshot taken by the author.]

```
% An example of using Javascript plugin DataTable():
%Include these two files in order to include additional advanced features to any
    HTML table
%cdn.datatables.net/1.10.20/css/jquery.dataTables.min.css
%cdn.datatables.net/1.10.20/js/jquery.dataTables.min.js
$(document).ready(function(){
    $('#myTable').dataTable(); //this plugin provides searching, sorting and
        pagination
 });
```

Table is initilaized with "myTable" id and this id is used in ready function() to assign dataTable funcionality to our HTML table instance[Tompsett 2018]. An example HTML table with this technique when using the sort feature on the `Price` column is shown in Figure 3.5

An example HTML table with this technique when using the search feature is shown in Figure 3.6.

Show 10 entries Search: Acura

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compresion Ratio	Miles per gallon	Price(€)
2019 Acura RDX	Acura	2.0	4	272	280	9.8:1	28	33,600.00

Showing 1 to 1 of 1 entries (filtered from 12 total entries) Previous 1 Next

Figure 3.6:
[Screenshot taken by the author.]

Chapter 4

Responsive Tables

Due to the increasing amount of screens and their varying shapes, sizes, and developers' space allocation when designing and implementing tables, responsive table techniques have been 'developed' by manipulating the table's columns and rows to provide an optimal experience for users across most mediums. It means the row and columns can be repositioned, resized, collapsed, minimized, etc..

Data tables can contain many information, which makes displaying that data quite messy and hard to look at. So by using responsive design, a big favor is done to the clients, by adjusting the table according to their devices. One idea would be to minimize the table, but if the user is looking at the table from his mobile device, he would have to zoom in, which is not that useful to him, because then again he would need to scroll to view the whole table [Alligator 2019].

As seen in Figure 4.1, both options do not really look nor perform well. So, by using some simple CSS it is possible to fix that problem. With the help of the mentioned media queries, it is possible to specify for which device sizes, which settings should be used.

And now, the end result is seen in the Figure 4.2.

Some of the most popular responsive techniques are:

- Hidden Columns - Selected by User
- Horizontal Scroll
- Fixed Header
- Flip Scroll
- User Resizeable Columns
- Long Two Column

In the following sections we will try to give a brief but detailed description of what exactly each technique consists of, as well as the reason to include this techniques on tables, and the code required to implement such techniques.

4.1 Horizontal Scroll

When the allocated space is too small - horizontally, instead of hiding it, a horizontal scroll bar (just for the table) is created. The user can then scroll away.

Horizontal Scrolling represents a technique that resizes the table into columns at small screen resoultion [Bushell 2016]. This is different from viewport scrolling as we scroll only through the table.

It is very useful technique when presenting large data sets with identifiers in the first column. Then is very easy for every user to compare data content with multiple identifiers[Coyle 2017].

This solution works for following browsers:

- Google Chrome
- Mozzila Firefox
- Internet Explorer
- Opera

Figure 4.1:
[https://css-tricks.com/responsive-data-tables/.]

```
% An example of using simple CSS with media queries on how to
% achieve Responsive Design:

@media
only screen and (max-width: 760px),
(min-device-width: 768px) and (max-device-width: 1024px)  {

  /* Force table to not be like tables anymore */
  table, thead, tbody, th, td, tr {
    display: block;
  }

  /* Hide table headers (but not display: none;, for accessibility) */
  thead tr {
    position: absolute;
    top: -624.9375rem;
    left: -624.9375rem;
  }

  tr { border: 0.0625rem solid #ccc; }

  td {
    /* Behave  like a "row" */
    border: none;
    border-bottom: 0.0625rem solid #eee;
    position: relative;
    padding-left: 50%;
  }

  td:before {
    /* Now like a table header */
    position: absolute;
    /* Top/left values mimic padding */
    top: 0;
    left: 0.375rem;
    width: 45%;
    padding-right: 0.625rem;
    white-space: nowrap;
  }
}
```

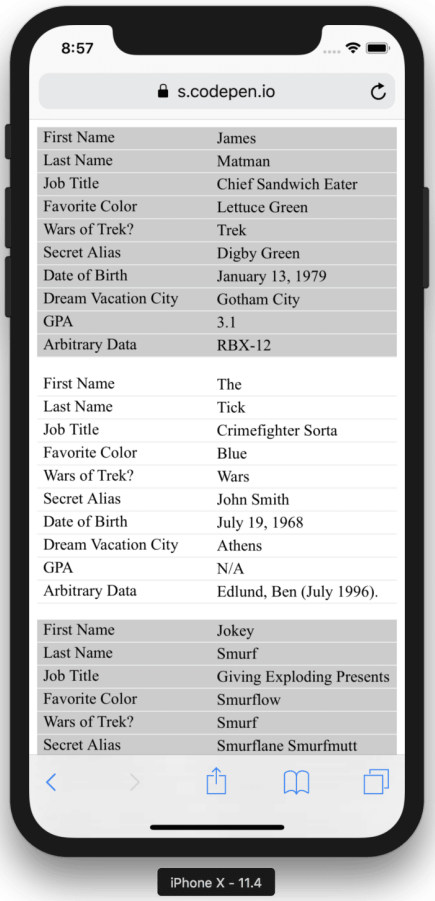


Figure 4.2:
[https://css-tricks.com/responsive-data-tables/.]

– Microsoft Edge

It is impossible to find one size that fits all solution. Data comparing is very difficult on the small screens. There are a lot of possible workarounds for this issue, but no one can solve this problem [Bushell 2016]. Our implementation of horizontal scrolling creates table elements scrollable, but also can not solve the issue to the end[Bushell 2016].

The code required to implement this technique on a table is shown in Listing **??**.

This CSS code represents CSS class `.rtable` used in the case when container is not resized. With display: inline-block all items are listed horizontally instead of vertically. `vertical-align` property maintains how elements set next to each other in a line. The `overflow` property determines whether to crop content or to add scroll bars (along the x-axis) when a table's content is too big to satisfy some screen resolution. We can use `white-space: nowrap` property as optional for small cell values. CSS border properties are used to control borders into a table[Bushell 2016].

An example HTML table with this technique is shown in Figure 4.3.

4.2 Fixed Header

When the table is vertically long, the table's header is fixed to the top of the table's view. As we scroll down, the table's header is kept always visible. It is very useful to have the fixed header (first row containing the header cells) fixed on the top of the table, when presenting tables with a particularly larage data set. This helps users to quickly determine what every column identify rather than need to scroll back to the top of the table every time.

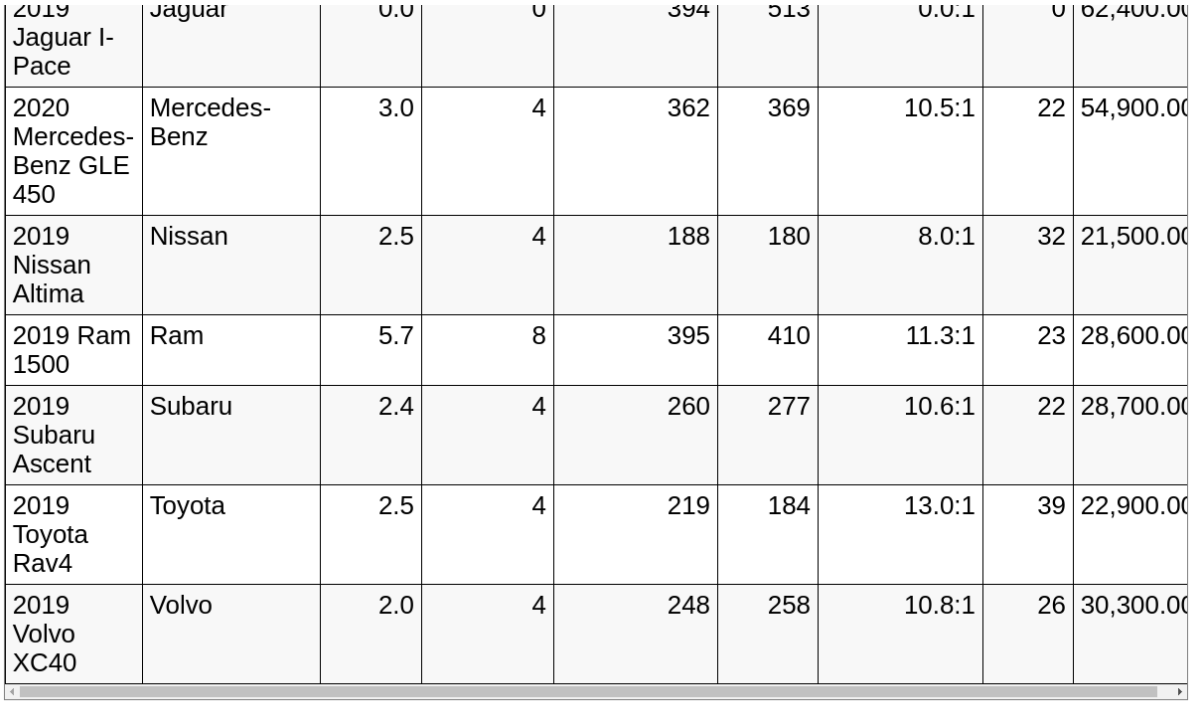
Fixed Header provides information that allows the user to always know in what column the cell the user is looking at is in. This is very effective feature that makes our life easier [Coyle 2017]. An implementation can be seen in Listing **??**. This solution works for following browsers:

```
.rtable {
  display: inline-block;
  vertical-align: top;
  max-width: 100%;

  overflow-x: auto;

  // optional - looks better for small cell values
  white-space: nowrap;

  border-collapse: collapse;
  border-spacing: 0;
}
```



2019 Jaguar I-Pace	Jaguar	0.0	0	394	513	0.0:1	0	62,400.0(
2020 Mercedes-Benz GLE 450	Mercedes-Benz	3.0	4	362	369	10.5:1	22	54,900.0(
2019 Nissan Altima	Nissan	2.5	4	188	180	8.0:1	32	21,500.0(
2019 Ram 1500	Ram	5.7	8	395	410	11.3:1	23	28,600.0(
2019 Subaru Ascent	Subaru	2.4	4	260	277	10.6:1	22	28,700.0(
2019 Toyota Rav4	Toyota	2.5	4	219	184	13.0:1	39	22,900.0(
2019 Volvo XC40	Volvo	2.0	4	248	258	10.8:1	26	30,300.0(

Figure 4.3:
[Screenshot taken by the author.]

- Google Chrome
- Mozzila Firefox
- Internet Explorer
- Opera
- Microsoft Edge

`<tbody>` element is determined with the type of rendering box. With `overflow: auto`, a scrollbar will appear along y-axis. All `<thead>` and `<tbody>` rows will behave as table elements. Fixed table layout algorithm is used to control table and column widths. Width for almost every table element is static, just the header has non-static width [Frederic no date]. The width of header is determined with the `calc()` function to allow responsiveness [w3schools.com no date(a)].

An example HTML table with this technique is shown in Figure 4.4.

```
.fixedHeader tbody {
  display: block;
  overflow: auto;
  width: 100%;
}

.fixedHeader thead tr {
  display: table;
  width: 100%;
  table-layout: fixed;
}

.fixedHeader thead, .fixedHeader tbody tr {
  display: table;
  width: 100%;
  table-layout: fixed;
}

.fixedHeader thead {
  width: calc( 100% - 0.6em )
}

.fixedHeader td {
  width: 100%;
}
```

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compression Ratio	Miles per gallon	Price(€)
2019 GMC Sierra 1500	GMC	4.0	6	285	305	11.0:1	16	26,600.00
2019 Honda Passport	Honda	3.5	6	280	262	11.5:1	21	28,700.00
2019 Jaguar I-Pace	Jaguar	0.0	0	394	513	0.0:1	0	62,400.00
2020 Mercedes-Benz GLE 450	Mercedes-Benz	3.0	4	362	369	10.5:1	22	54,900.00

Figure 4.4:
[Screenshot taken by the author.]

```
table.flipscroll{
  width: 100%;
  overflow-x: auto;
}

table.flipscroll  tbody tr {
  display: inline-block;
  vertical-align: top;
}

table.flipscroll  tbody {
  display: block;
  width: auto;
  position: relative;
  overflow-x: auto;
  white-space: nowrap;
}
```

4.3 Flip Scroll

This technique is useful for an object-based table; one where each row is populated by a single 'object' where each column represents a different feature of the object. If all features – columns – cannot be seen at once, it is sometimes more desireable to be able to show all features of an object rather than a good amount of (incomplete) features for a good amount of objects.

Rather than having the user scroll through features, the user is now able to scroll through objects horizontally instead. This is the result of transposing the table. The first column will now house the (previously column) headers and each column will be devoted to an object, rather than each row. Flip scroll also 'promises' to keep the table in the allocated space – by allowing table horizontal scrolling.

An implementation can bee seen in Listing **??**. This solution works for following browsers:

- Google Chrome
- Mozzila Firefox
- Internet Explorer
- Opera
- Microsoft Edge

Setting `width: 100%` and `overflow-x: auto` makes the table horizontally scrollable and stick to the allocated space. The trick is setting the attribute `display: inline-block` on the `tbody tr` HTML tag and `white-space: nowrap` on the `tbody` HTML tag [Elvery no date].

Figure 4.5 represents an HTML table with Flip Scroll.

4.4 User Resizeable Columns

This technique empowers the user quite significantly. As the title suggests, the user is able to manipulate the size of a table's columns.

This technique is useful when the user is immune to space limitation penalties the table might have on most other users. For example, if a user has a screen big enough that space between columns is wasted to keep column sizes consistent, being able to resize this column to bring other columns into view is highly desireable.

User Resizeable Columns allows the user to tailor the table's experience to his/her own preferences. There are two possible solutions. The solution using only CSS can be seen in Listings **??** and the solution using JQuery can be seen in Listing **??**. Both work for following browsers:

- Google Chrome
- Mozzila Firefox

Car	2019 Acura RDX	2019 Ford Ranger	2019 Genesis G70
Manufacturer	Acura	Ford	Genesis
Engine Size(L)	2.0	2.3	2.0
Cylinders	4	4	4
Horsepower	272	270	252
Torque	280	310	260
Compresion Ratio	9.8:1	10.0:1	10.0:1
Miles per gallon	28	21	22
Price(€)	33,600.00	21,800.00	31,300.00

Figure 4.5:
[Screenshot taken by the author.]

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compresion Ratio	Miles per gallon	Price(€)
2019 Acura RDX	Acura	2.0	4	272	280	9.8:1	28	33,600.00
2019 Ford Ranger	Ford	2.3	4	270	310	10.0:1	21	21,800.00
2019 Genesis G70	Genesis	2.0	4	252	260	10.0:1	22	31,300.00
2019 GMC Sierra 1500	GMC	4.0	6	285	305	11.0:1	16	26,600.00
2019 Honda Passport	Honda	3.5	6	280	262	11.5:1	21	28,700.00

Figure 4.6:
[Screenshot taken by the author.]

- Internet Explorer
- Opera
- Microsoft Edge

Encapsulating every header cell (or every single cell) with a `<div>` tag allows CSS to enable resizeability by setting the attribute `resize: horizontal` to these divs [w3schools.com no date(b)].

Figure 4.6 represents an HTML table with User Resizeable Columns using only HTML and CSS.

The trick to use this technique using JQuery is calling the `resizeableColumns()` on the table of your choice [Jo-Geek 2019]. The resulting image can be seen in Figure 4.7.

4.5 Long Two Column

Just like Flip Scroll, this technique is useful for an object-based table; one where each row is populated by a single 'object' where each column represents a different feature of the object.

Rather than having the user scroll through features, the user is now able to scroll through objects vertically instead. This is the result of transposing the table. The first column will now house the (previously column) headers and the second column will house the data previously housed in the first row. This creates a 'minitable' for the (previously) first row. For the next object

```
<thead>
    <tr>
        <th title="Car">
            <div class = "resizecell">Car</div>
        </th>
        <th title="Manufacturer">
            <div class = "resizecell">Manufacturer</div>
        </th>
        <th title="Engine Size(L)">
            <div class = "resizecell">Engine Size(L)</div>
        </th>
        [...]
    </tr>

</thead>
  .resizecell {
    resize: horizontal;
    overflow: auto;
    width: 100%;
    hyphens: auto;
  }
<style>

</style>
```

```
~/js/jQuery.resizableColumns.js

$(function(){
  $('.rcl').resizeableColumns();
});
```

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compr
2019 Acura RDX	Acura	2.0	4	272	280	
2019 Ford Ranger	Ford	2.3	4	270	310	
2019 Genesis G70	Genesis	2.0	4	252	260	
2019 GMC Sierra 1500	GMC	4.0	6	285	305	
2019 Honda Passport	Honda	3.5	6	280	262	
2019 Jaguar I-Pace	Jaguar	0.0	0	394	513	
2020 Mercedes-Benz GLE 450	Mercedes-Benz	3.0	4	362	369	
2019 Nissan Altima	Nissan	2.5	4	188	180	
2019 Ram 1500	Ram	5.7	8	395	410	
2019 Subaru Ascent	Subaru	2.4	4	260	277	
2019 Toyota Rav4	Toyota	2.5	4	219	184	
2019 Volvo XC40	Volvo	2.0	4	248	258	

Figure 4.7:
[Screenshot taken by the author.]

```
table, thead, tbody, th, td, tr {
  display: block;
}

td {
  /* Behave  like a "row" */
  border: none;
  border-bottom: 0.0625rem solid #eee;
  position: relative;
  padding-left: 50%;
}

td:before {
  /* Now like a table header */
  position: absolute;
  /* Top/left values mimic padding */
  top: 0;
  left: 0.375rem;
  width: 45%;
  padding-right: 0.625rem;
  white-space: nowrap;
}
```

(and row), another ‘minitable’ is created like with the first one and then appended below. This approach creates a ‘minitable’ for each object and appends it to the previous object.

As the name suggest, you end up with a long table of 2 columns, the first of which is ’always the same’. Unlike Flip scroll, it does not keep the table in the allocated space, unless you set vertical scrolling. This solution works for following browsers:

- Google Chrome
- Mozzila Firefox
- Internet Explorer
- Opera
- Microsoft Edge

As seen in Listing **??**, trick is setting the table’s elements to `display: block` and the <td> tag’s attribute to `position: relative` [CSS-Tricks no date]. Figure 4.8 shows an HTML table with Long Two Column technique applied to it.

Car	2019 Acura RDX
Manufacturer	Acura
Engine Size(L)	2.0
Cylinders	4
Horsepower	272
Torque	280
Compresion Ratio	9.8:1
Miles per gallon	28
Price(€)	33,600.00
Car	2019 Ford Ranger
Manufacturer	Ford
Engine Size(L)	2.3
Cylinders	4
Horsepower	270
Torque	310
Compresion Ratio	10.0:1
Miles per gallon	21
Price(€)	21,800.00
Car	2019 Jaguar I-Pace
Manufacturer	Jaguar
Engine Size(L)	0.0
Cylinders	

Figure 4.8:
[Screenshot taken by the author.]

Chapter 5

Recommendation

5.1 Recommendation

Responsive tables must be able to efficiently and effectively display data. It is only through a combination of good table design and responsive techniques one is able to achieve this.

For good table design, it is imperative that you include Alternate Row Highlighting. This is a very simple technique that works on every browser and is not only helpful to the user but to the developer as it adds positively to the aesthetics.

For a "Jack of all trades" approach, we suggest the following responsive techniques: Long Two Column, and Fixed Header. You are able to apply this for the 3 main screen sizes: PC, tablet, and phone. Being able to apply Long Two Column to phone-sized screens (or tablets in vertical mode) and leave Fixed Header on the remaining two sizes (PC and tablets in horizontal mode) automatically eliminates one of the 3 sizes you have to take care of.

Another thing to keep in mind is to attach the vertical scrollable feature of Fixed Header to the Long Two Column. This ensures that the space the table is taking up is exactly the one allocated to it. It goes without saying that if the table is too wide (horizontally), it should be horizontally scrollable.

Figure 5.1 shows these techniques working together.

As you scroll through the table, notice the Fixed Header Technique in Figure 5.2.

For small-sized screens, the technique that would "kick in" is shown in Figure 5.3. Notice the 'creeping' second 'minitable' in the bottom part.

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compresion Ratio	Miles per gallon	Price(€)
2019 Acura RDX	Acura	2.0	4	272	280	9.8:1	28	33,600.00
2019 Ford Ranger	Ford	2.3	4	270	310	10.0:1	21	21,800.00
2019 Jaguar I-Pace	Jaguar	0.0	0	394	513	0.0:1	0	62,400.00
2020 Mercedes-Benz GLE 450	Mercedes-Benz	3.0	4	362	369	10.5:1	22	54,900.00
2019 Nissan Altima	Nissan	2.5	4	188	180	8.0:1	32	21,500.00
2019 Ram	Ram	5.7	8	395	410	11.3:1	23	28,600.00

Figure 5.1:
[Screenshot taken by the author.]

Car	Manufacturer	Engine Size(L)	Cylinders	Horsepower	Torque	Compresion Ratio	Miles per gallon	Price(€)
2019 Nissan Altima	Nissan	2.5	4	188	180	8.0:1	32	21,500.00
2019 Ram 1500	Ram	5.7	8	395	410	11.3:1	23	28,600.00
2019 Subaru Ascent	Subaru	2.4	4	260	277	10.6:1	22	28,700.00
2019 Toyota Rav4	Toyota	2.5	4	219	184	13.0:1	39	22,900.00
2019	Volvo	2.0	4	248	258	10.8:1	26	30,300

Figure 5.2:
[Screenshot taken by the author.]

Car	2019 Acura RDX
Manufacturer	Acura
Engine Size(L)	2.0
Cylinders	4
Horsepower	272
Torque	280
Compresion Ratio	9.8:1
Miles per gallon	28
Price(€)	33,600.00
Car	2019 Ford Ranger

Figure 5.3:
[Screenshot taken by the author.]

Bibliography

Ahmad, Irfan [2018]. *Fixed header and scrollable body in table in html angular*. 15 Jun 2018. https://www.socialmediatoday.com/news/how-much-data-is-generated-every-minute-infographic-1/525692/ (cited on page 12).

Alligator [2019]. *Implementing A Pure CSS Collapsible*. 2019. https://alligator.io/css/collapsible/ (cited on pages 10, 15).

Bushell, David [2016]. *CSS only Responsive Tables*. 04 Mar 2016. https://dbushell.com/2016/03/04/css-only-responsive-tables/ (cited on pages 15, 17).

Coyier, Chris [2018]. *Responsive Data Tables*. Jun 2018. https://css-tricks.com/responsive-data-tables/ (cited on page 7).

Coyle, Andrew [2017]. *Design better data tables*. 07 May 2017. https://uxdesign.cc/design-better-data-tables-4ecc99d23356 (cited on pages 15, 17).

CSS-Tricks [no date]. *Responsive Table Demo*. https://codepen.io/team/css-tricks/pen/wXgJww?editors=1100 (cited on page 23).

[No date]. *DataTables*. https://datatables.net/ (cited on page 12).

Elvery, Simon [no date]. *Responsive Tables Demo*. https://elvery.net/demo/responsive-tables/ (cited on page 20).

Frain, B. [2015]. *Responsive web design with HTML5 and CSS3*. 2015 (cited on page 2).

Frederic [no date]. *Fixed header and scrollable body in table in html angular*. https://mdbootstrap.com/support/angular/fixed-header-and-scrollable-body-in-table-in-html-angular/ (cited on page 18).

Jo-Geek [2019]. *jQuery Plugin For Resizeable Table Columns - ResizeableColumns*. 15 Nov 2019. https://www.jqueryscript.net/table/jQuery-Plugin-Resizable-Table-Columns.html (cited on page 21).

ITU.int [2015]. *Individuals using the Internet 2005 to 2014*. International Telecommunication Union, May 2015. https://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx (cited on page 1).

Mohamed, A. A. [2015]. *An Enhanced Approach to Responsive Web Design Influid Grid Concept*. 2015 (cited on page 1).

Negi, Himanshu [2018]. *Using MediaQueries*. May 2018. https://medium.com/beginners-guide-to-mobile-web-development/media-queries-54a1a463356f (cited on page 1).

Sharki, C. and Fisher [2013]. *Jump Start Responsive Web Design*. Apr 2013 (cited on page 1).

Tompsett, Brian [2018]. *How to use paging in a table in html*. 02 Feb 2018. https://stackoverflow.com/questions/48966595/how-to-use-paging-in-a-table-in-html/48966882 (cited on page 13).

Tutorial, HTML5 [no date]. *Basics of Tables*. https://html5-tutorial.net/tables/basics-of-tables/ (cited on page 3).

w3schools.com [no date(a)]. *CSS calc() Function*. https://www.w3schools.com/cssref/func_calc.asp (cited on page 18).

w3schools.com [no date(b)]. *CSS resize Property*. https://www.w3schools.com/cssref/css3_pr_resize.asp (cited on page 21).

Wood, Adam [no date]. *HTML Tags*. https://html.com/tags/tbody/ (cited on page 7).