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

5 Nov 2019

Abstract

Responsive is a technique used in web design which enables better rendering of web pages,tables and other things where it may be used. With better rendering, it is meant as a rescaling of the resolution of the device used to open the specific website. When using responsive, it doesn't matter from which device the user enters a website from, the web site should be resized accordingly to the device.

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, which allow them to be more flexible and complex, which results in more data being actually shown rather than comprising some. And, a better design and look of course. So with this paper, the reader will have a good overview of what responsive and responsive tables are, and which are the best techniques to achieve that.

Contents

List of Figures

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 [**1**].

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 [**2**].

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 [**3**].

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 [**4**].

Basically it allows defining specific CSS rules, depending on the device used.

```
% An example of a media query, where one can say that for a screen of size 850px
    or less, the background is black:

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

From this example above, 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.

```
% 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>
```

If this was ran in the browser, 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[**5**]. 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 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[**6**].

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.

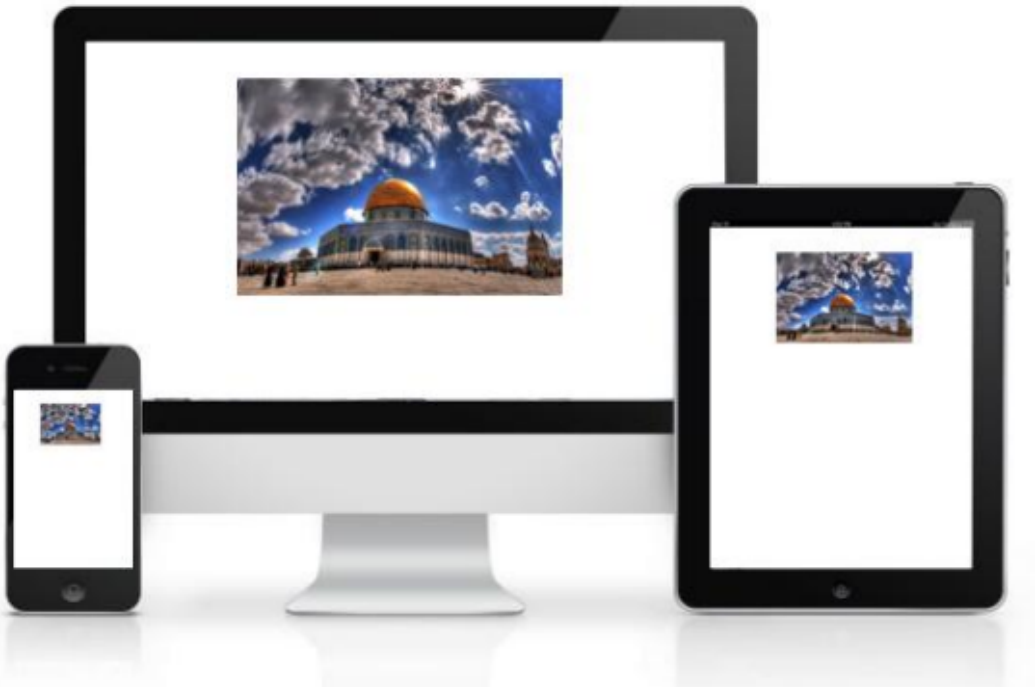

Figure 1.1: Relative Image Dimensions. [Taken from research paper.]

1.4.2 Cropping

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

(a) Picture with original size

(b) Picture when cropped

Figure 1.2: Image cropping. [https://alligator.io/css/cropping-images-object-fit/.]

In this Figure above, it can be obviously seen that the result of tweaking the image a bit. With just a few CSS commands, it is possible to crop the picture and if someone is browsing the web site with a smaller device, the image does not have to be hidden, just its' size is changed.

```
    % 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:300px; /*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*/
}
```

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

Chapter 2

HTML Tables

In case of having much data to display, or data which is better of in a grid, HTML tables are the best solution available. Before, tables have been used mostly for the layout of HTML web-sites and it is not a good practice to do so, mainly because of two things:

- 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 [7].

```
    % 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>
```

Now, there exist some other 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 [**8**].

• <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> [**8**].

• <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 [**8**].

• <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.

2.2 Good Table Design

Now, there are certain guidelines which can help a developer, or if that person can be called that way, table maintainer, make a table and its design better. By that is meant that there are some interesting ways where a little of simple CSS can be used to your advantage to make your table stand out.

2.2.1 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, atleast for the eyes if nothing else. The idea is to color every even row, while leaving the odd ones in tact. As said, it is pretty simple, and requires only two lines of CSS, but also pretty useful.

```
1    % An example of using simple CSS to color table rows:
2
3  table.alt tr:nth-child(even) {background: #CCC}
4  table.alt tr:nth-child(odd) {background: #FFF}
```

And now, at the end, this is the final result:

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 2.1:
[Screenshot taken by the author.]

2.2.2 Current Row Highlighting

Now again, talking about a big table, and going through it, one may easily becomes lost and wouldn't know in which row he is at the moment, which can be pretty stressful. Again with the help of some CSS, the lives of the users' is made easier.

```
    % 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;
}
```

And again, it is shown how a small amount of CSS can be helpful. The result:

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 2.2:
[Screenshot taken by the author.]

2.2.3 Expandable Areas

Should we have any further information available about a table's object, making the row clickable and showing the extra information then, is a better alternative than trying to cram everything inside the table. In the following code sipped we see how addtion data is in html code implemented. [**10**]

```
<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>
```

To implement this table feature the combination of javascript and css need to be implemented. In the further codes sipped the javascript and css code can be reviewed. JavaScript Code

```
<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>
```

```
#report {
    border-collapse:collapse;
}
#report h4 {
    margin:0px;
    padding:0px;
}
#report img {
    float:right;
}
#report ul {
    margin:10px 0 10px 40px;
    padding:0px;
}
#report th {
    background:#7CB8E2  repeat-x scroll center left;
    color:#fff;
    padding:7px 15px;
    text-align:left;
}
#report td {
    background:#C7DDEE none repeat-x scroll center left;
    color:#000;
    padding:7px 15px;
}
#report tr.odd td {
    cursor:pointer;
}
#report div.arrow {
    background:transparent url(arrows.png) no-repeat scroll 0px -16px;
    width:16px;
    height:16px;
    display:block;
}
#report div.up {
    background-position:0px 0px;
}
```

Now we will see how it look like in partice on html page. It is compiled localy. In the figure 2.3 we can see how our table look when we see it in html page.

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 2.3:
[Screenshot taken by the author.]

Pressing to some of cell in row or colums we get the additional infomrmation, that we defined in html code. This is shown in the figure 2.4.

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 2.4:
[Screenshot taken by the author.]

2.2.4 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[**PG_2**].
Tables are sometimes one of the possible places for saving large sets of data. This type of tables should consists of different types of features that enables users easier operations of maintaining.
Were the table be too long, it can be divided into 'pages', and then certain amount of rows is viewed at a time.

An example of good table design with different features is called pagination, because the most important feature of this technique is pagination. With this feature is possible to determine the number of rows per page, previous and next page

navigation are also available. Every user has also possibility to filter results by text search and in this way find the desired row. There is also possibility to sort table content in an descending or ascending order.

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[**PG_1**].

```
    % 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[**PG**].

Following image represents html table with sorted Price column and 10/12 entries:

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 2.5:
[Screenshot taken by the author.]

Following image represents html table with searched term:

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 2.6:
[Screenshot taken by the author.]

2.3 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.. Here are some techniques we would like to decsribe in the following chapter:

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

In literatury the mostly is find online("on internet") there are a lot of techniques i.e. there are a lot of names for very similar or the same techniques. We tried to take the best ones, to name it on most resonable way and to describe them.

2.4 Responsive Tables Techniques

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 [**9**].

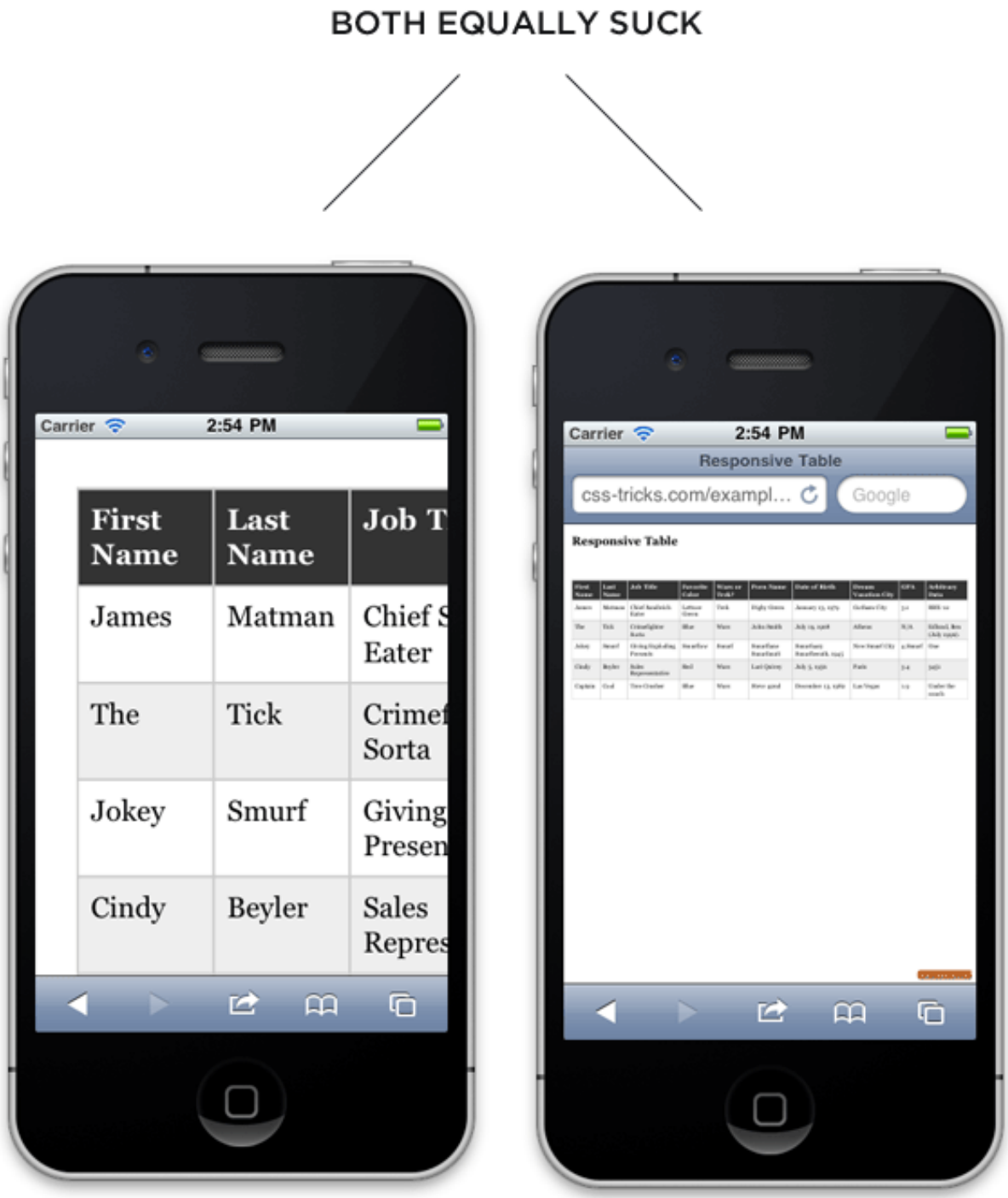

Figure 2.7:
[https://css-tricks.com/responsive-data-tables/.]

As seen in the figure above, both options do not really look nor do as any good. 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.

```
    % 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: 1px solid #ccc; }

  td {
    /* Behave  like a "row" */
    border: none;
    border-bottom: 1px 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 now, the end result would be the following one:

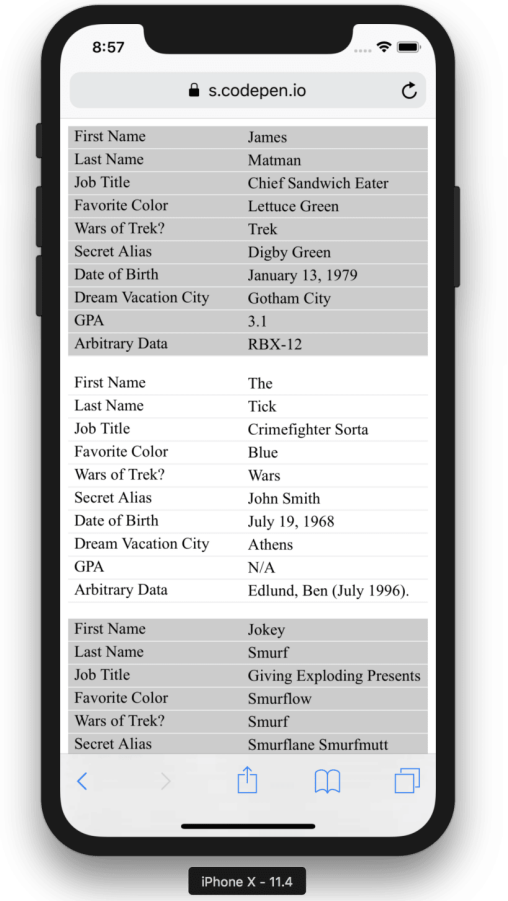

Figure 2.8:
[https://css-tricks.com/responsive-data-tables/.]

2.5 Horizontal Scroll

When the allocated space is too small - horizontally, instead of hiding it, horizontal scroll bar just for the table is created and let the user scroll away.

Horizontal Scrolling represents a technique that resizes the table into columns at small screen resoultion [**HS_1**]. The rows can be scrolled from left to right with fixed first row. 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[**HS**].

This solution works for following browsers:
• Google Chrome
• Mozzila Firefox
• Internet Explorer
• Opera
• 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 [**HS_1**]. Our implementation of horizontal scrolling creates table elements scrollable, but also can not solve the issue to the end[**HS_1**]. All important css-properties that creates a responsive table with horizontal scrolling are explained.

```
.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;
}
```

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[**HS_1**].

Following image represents html table with horizontal scroll:

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 2.9:
[Screenshot taken by the author.]

2.6 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 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 background on what column the user is on. This is very effective feature that makes our life easier [**HS**].

This solution works for following browsers:
- Google Chrome
- Mozzila Firefox
- Internet Explorer
- Opera
- Microsoft Edge

```
.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%;
}
```

Tbody element is determined with the type of rendering box. With overflow: auto 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 [**FH_1**]. The width of header is determined with calc() function and in this way is solved the main issue in this responsive technique[**FH**].

Following image represents an HTML table with fixed header:

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 2.10:
[Screenshot taken by the author.]

2.7 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 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.

This solution works for following browsers:
- Google Chrome
- Mozzila Firefox
- Internet Explorer
- Opera
- Microsoft Edge

```
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;
}
```

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.

Following image represents an HTML table with Flip Scroll:

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 2.11:
[Screenshot taken by the author.]

2.8 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. Both work for following browsers:
- Google Chrome
- Mozzila Firefox
- Internet Explorer
- Opera
- Microsoft Edge

Solution using only CSS and HTML:

```
<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;
}
```

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.

Following image represents an HTML table with User Resizeable Columns using only HTML and CSS:

urc_css.png

Figure 2.12:
[Screenshot taken by the author.]

Solution using JavaScript (JQuery):

```
<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;
}
```

Chapter 3

Recommendation

3.1 Recommendation

For responsive design and techniques you should employ on your tables, we suggest: Alternate Row Highlighting, Long Two Column, and Fixed Header. As well the developer i.e. designer show take consist different techniques to different screen size. For example like it is on Figure 1.1 the three different screen sizes. For the biggest one you define the height and weight of yout table to 100% of container. For Tablets we recomand the fixed header, and for small screen size like mobile devices we recommand the Long Two Column technique. This is done in order to cover all posible screen(container) sizes. As well to be much easier to find wanted data in table, please use Alternate Row Highlighting. To take a look how each of this techniques is working take a look on previous chapters. How our recommentation table look like, you can see in following figure.

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 3.1:
[Screenshot taken by the author.]