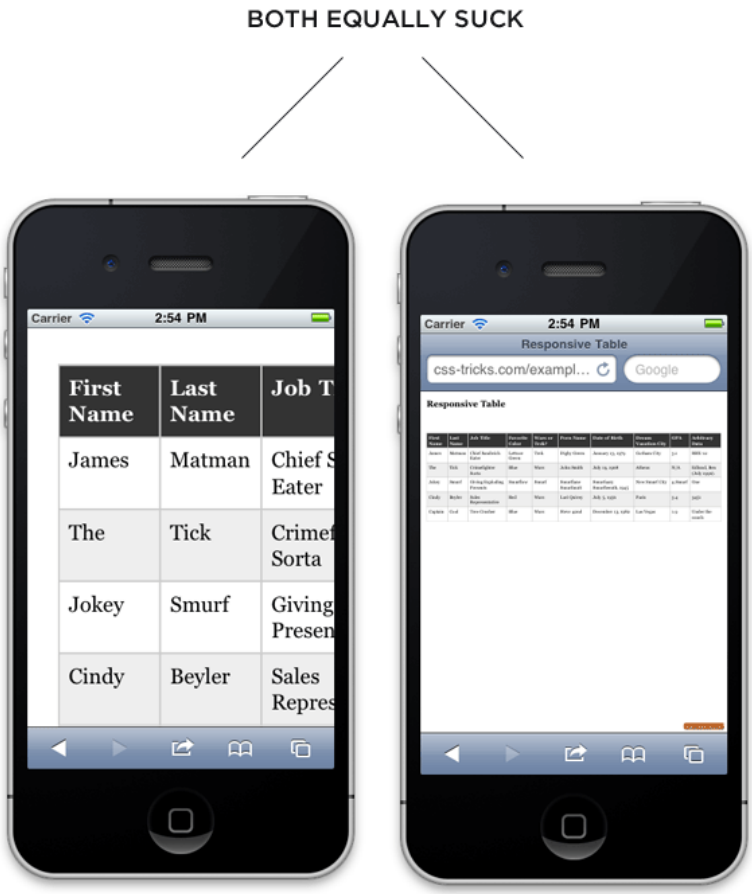
Responsive Data Tables

Garrett Dimon:

Data tables don't do so well with responsive design. Just sayin'.

He has a good point. Data tables can be quite wide, and necessarily so. A single row of data needs to be kept together to make any sense in a table. Tables can flex in width, but they can only get so narrow before they start wrapping cells contents uncomfortably or just plain can't get any narrower.

Responsive design is all about adjusting designs to accommodate screens of different sizes. So what happens when a screen is narrower than the minimum width of a data table? You can zoom out and see the whole table, but the text size will be too small to read. Or you can zoom in to the point of readability, but browsing the table will require both vertical and (sad face) horizontal scrolling.



So here's what we are gonna do...

We're going to use "responsive design" principles (CSS @media queries) to detect if the screen is smaller than the maximum squishitude of our table. If it is, we're going to reformat the table.

We're being good little developers and using Plain Ol' Semantic Markup here for our table. Bare bones example:

HTML

```
<table>
    <thead>
      <tr>
        <th>First Name</th>
        <th>Last Name</th>
        <th>Job Title</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>James</td>
        <td>Matman</td>
        <td>Chief Sandwich Eater</td>
      </tr>
      <tr>
        <td>The</td>
        <td>Tick</td>
        <td>Crimefighter Sorta</td>
      </tr>
    </tbody>
  </table>
```

Our regular CSS is nothing special:

CSS

```
/*
Generic Styling, for Desktops/Laptops
*/
table {
  width: 100%;
  border-collapse: collapse;
}
/* Zebra striping */
tr:nth-of-type(odd) {
  background: #eee;
}
th {
  background: #333;
  color: white;
  font-weight: bold;
}
td, th {
  padding: 6px;
  border: 1px solid #ccc;
  text-align: left;
}
```

The small-screen responsive stuff comes in now. We've already figured out our minimum table width is about 760px so we'll set up our media query to take effect when the narrower than that. Also, we'll target iPads as they are right in that zone.

The biggest change is that we are going to force the table to not behave like a table by setting every table-related element to be block-level. Then by keeping the zebra striping we originally added, it's kind of like each table row becomes a table in itself, but only as wide as the screen. No more horizontal scrolling! Then for each "cell", we'll use CSS generated content (:before) to apply the label, so we know what each bit of data means.

CSS

```
/* Max width before this PARTICULAR table gets nasty. This query will take effect for any screen smaller than 760px and also iPads specifically. */

@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: -9999px;
		left: -9999px;
	}

	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: 6px;
		left: 6px;
		width: 45%;
		padding-right: 10px;
		white-space: nowrap;
	}

	/* Label the data */
	td:nth-of-type(1):before { content: "First Name"; }
	td:nth-of-type(2):before { content: "Last Name"; }
	td:nth-of-type(3):before { content: "Job Title"; }
}
```

And so, desktops get the regular table experience, mobile (or otherwise small screens) get a reformatted and easier to explore table: