Responsive Design & Bootstrap

Contents

[1 Adaptive vs. Responsive Design 1](#_Toc155439240)

[2 Implementing Responsive Design 3](#_Toc155439241)

[2.1 Responsive Design 3](#_Toc155439242)

[2.2 Semantic Layout Elements 3](#_Toc155439243)

[2.3 Responsive CSS Properties 4](#_Toc155439244)

[3 CSS Grid & Flexbox 5](#_Toc155439245)

[4 Responsive Design Tutorials 6](#_Toc155439246)

[4.1 W3Schools 6](#_Toc155439247)

[4.2 freeCodeCamp 6](#_Toc155439248)

[Appendix 1 Responsive Design Resources 6](#_Toc155439249)

[Appendix 2 Firefox Tools 8](#_Toc155439250)

[Appendix 2.1 Responsive Design Mode 8](#_Toc155439251)

[Appendix 2.2 Developer Tools 9](#_Toc155439252)

# Adaptive vs. Responsive Design

Responsive Design:

A responsive website shows content based on the available browser space. If you open a responsive site on the desktop and then change the size of the browser window, the content will move dynamically to arrange itself (at least in theory) optimally for the browser window. On mobile phones, this process is automatic; the site checks for the available space and then presents itself in the ideal arrangement.

Responsive design is straightforward. Because it is fluid, it means that users can access your online world and enjoy as much of it on their handheld device as they would on a massive monitor. For this to be true, responsive design requires a very good conceptualization of the site and a deep knowledge of the needs and wants of the end users!

Adaptive Design:

Where responsive design relies on changing the design pattern to fit the real estate available to it, adaptive design has multiple fixed layout sizes. When the site detects the available space, it selects the layout most appropriate for the screen. So, when you open a browser on the desktop, the site chooses the best layout for that desktop screen; resizing the browser has no impact on the design.

Some sites have been quick to embrace adaptive design. Amazon, USA Today, Apple, and About.com configured themselves to be mobile-optimized websites[[1]](#footnote-1). The layout displayed on a mobile website using adaptive design may be different from the desktop’s version. However, this is because the designers have picked a different layout for the phone’s screen rather than leaving the design to try to rearrange itself.

In adaptive design, it’s normal to develop six designs for the six most common screen widths; 320, 480, 760, 960, 1200, and 1600 pixels.

Comparison:

Responsive Design is easier and takes less work to implement. … Responsive designers create a single design to be used on all screens and will generally start in the middle of the resolution and use media queries to determine what adjustments will be made for the lower and higher end of the resolution scale.

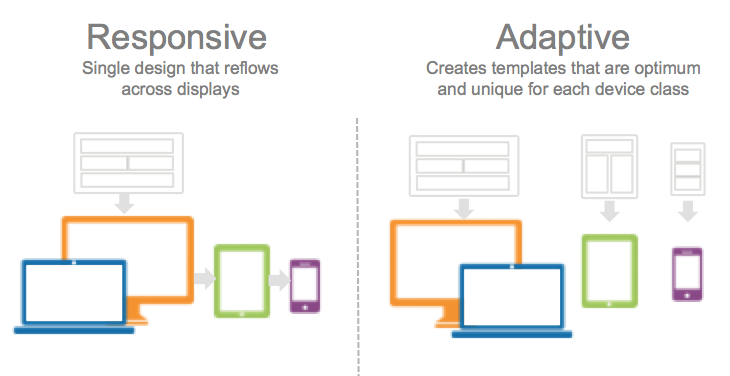
Adaptive design will (theoretically) ensure the best user experience according to whichever device the user is using to interface. Unlike responsive design, where a screen “flows” from desktop design into a smaller device’s, adaptive design offers tailor-made solutions…. Another advantage — research shows that a company with an adaptive website will often outperform, on speed tests, a company with a responsive website. This isn’t a small difference either; adaptive sites are often 2-3 times faster than responsive ones and give rather less data to the user in order to deliver the user experience

Responsive Design is the most popular, currently.

Source: <https://www.interaction-design.org/literature/article/adaptive-vs-responsive-design>

Other references:

* <https://www.interaction-design.org/literature/topics/responsive-design>
* <https://www.interaction-design.org/literature/topics/adaptive-design>
* <https://www.interaction-design.org/literature/topics/ui-design-patterns>
* <https://www.uxpin.com/studio/blog/responsive-vs-adaptive-design-whats-best-choice-designers/>



Source: <https://www.gomolearning.com/blog/elearning-technology/responsive-versus-adaptive-content-elearning/>

# Implementing Responsive Design

This section discusses some of the techniques need for designing responsive sites.

## Responsive Design

There are two approaches to build responsive or adaptive sites:

1. Manually – Apply CSS to HTML in such a way that the site is responsive. This involves techniques we consider next: semantic HTML elements (next [Section 2.2](#_Layout_Elements_&)) and/or *divs*; and styling them properly using CSS *display*, *float*, *clear*, *box-sizing,* and others; Media Queries; CSS Grid; and CSS Flexbox. We consider these in [Section 2.3](#_Responsive_CSS_Properties).
2. Framework - Use a CSS framework such as [W3.CSS](https://www.w3schools.com/w3css/defaulT.asp) or [Bootstrap](https://www.w3schools.com/bootstrap5/index.php) to build responsive designs. These are built on top of the CSS concepts in the preceding paragraph. We consider bootstrap in the next lesson. They usually also rely on JavaScript.

“In essence, a CSS framework comprises several CSS stylesheets ready for use by web developers and designers. The stylesheets are prepped for standard web design functions: setting colors, layout, fonts, navbars, etc. Generally, stylesheets are supported and expanded by other scripting technologies like SASS and JavaScript.”

“With a CSS framework, the user has a completed CSS stylesheet, and they only have to code the HTML with accurate classes, structure, and IDs to set up a web page. The framework already has classes built-in for common website elements – footer, slider, navigation bar, hamburger menu, column-based layouts, etc.”

Source: <https://www.browserstack.com/guide/top-css-frameworks>

## Semantic Layout Elements

|  |  |  |
| --- | --- | --- |
| HTML has several semantic elements that define the different parts of a web page:   |  |  | | --- | --- | | HTML5 Semantic Elements | * <header> - Defines a header for a document or a section * <nav> - Defines a set of navigation links * <section> - Defines a section in a document * <article> - Defines an independent, self-contained content * <aside> - Defines content aside from the content (like a sidebar) * <footer> - Defines a footer for a document or a section * <details> - Defines additional details that the user can open and close on demand * <summary> - Defines a heading for the <details> element |   Source: <https://www.w3schools.com/html/html_layout.asp> |

For more detail, see [HTML Semantics](https://www.w3schools.com/html/html5_semantic_elements.asp).

Using an approach like this makes the HTML more understandable, and easier to apply CSS. Of course, you could do it all with *div* tags.

## Responsive CSS Properties

This section has a few more CSS topics that are useful for responsive design and for an understanding of the bootstrap framework we consider in the next lesson.

1. The *display* property is the most important CSS property for controlling layout.

|  |  |
| --- | --- |
| Reference | Comments |
| [CSS display Property](https://www.w3schools.com/css/css_display_visibility.asp) | [W3schools] |
| [display](https://developer.mozilla.org/en-US/docs/Web/CSS/display) | In second section, *Definition & Usage*, choose the “Show demo” button. [W3schools] |
| [CSS Display: One-Stop Tutorial](https://www.simplilearn.com/tutorials/css-tutorial/css-display) | [simplilearn] |

1. The CSS *overflow* property controls what happens to content that is too big to fit into an area.

|  |  |
| --- | --- |
| Reference | Comments |
| [CSS Layout - Overflow](https://www.w3schools.com/css/css_overflow.asp) | [W3schools] |
| [overflow](https://developer.mozilla.org/en-US/docs/Web/CSS/overflow) | [developer.mozilla] |
| [Flow layout and overflow](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_flow_layout/Flow_layout_and_overflow) | [developer.mozilla] |

1. The CSS *float* property specifies how an element should float (left, right, none, inherit). Examples: float an image to the left or right around text.

|  |  |
| --- | --- |
| Reference | Comments |
| [CSS Layout - float and clear](https://www.w3schools.com/css/css_float.asp) | [W3schools] |
| [float](https://developer.mozilla.org/en-US/docs/Web/CSS/float) | [developer.mozilla] |

1. The CSS *clear* property specifies what should happen to an element when it is next to a floating element. In other words, if you have some floated elements and you want the next element to be below, you use the *clear* property. The authors recommend: *clearfix::after* to handle an element (*e.g.* an image) that overflows its container.

|  |  |
| --- | --- |
| Reference | Comments |
| [CSS Layout - clear and clearfix](https://www.w3schools.com/css/css_float_clear.asp) | [W3schools] |
| [clear](https://developer.mozilla.org/en-US/docs/Web/CSS/clear) | [developer.mozilla] |

1. The CSS *box-sizing* property allows us to include the padding and border in an element’s total width and height. By default, the actual width of an element is: width + padding + border. In other words, if you have a *div* with a width of 100 and add 10-pixel padding, then the *div* will appear bigger than you have set. In this case, the width of the *div* would be 120 pixels. By applying: box-sizing: border-box, to an element, this will force the element’s actual width to include the padding and border. A common rule defines that all elements on the page (“\*” is the universal selector, it selects everything on the page) will have this feature applied to them.

\* {  
  box-sizing: border-box;  
}

|  |  |
| --- | --- |
| Reference | Comments |
| [CSS Box Sizing](https://www.w3schools.com/css/css3_box-sizing.asp) | [W3schools] |
| [box-sizing](https://developer.mozilla.org/en-US/docs/Web/CSS/box-sizing) | [developer.mozilla] |

1. High-level overview of CSS Grid & Flexbox. Examples of each of these techniques is found in the code download (*ResponsiveDesign.zip*).

Source for all: <https://www.simplilearn.com/tutorials/css-tutorial/css-grid-vs-flexbox>

CSS Grid:

* Two-dimensional. Defined with rows and columns, which can be merged.
* Uses: *display: grid* in the container.
* Useful for creating more complex and organized layouts.

CSS Flexbox:

* One-dimensional, useful in allocating space among the flex items in the flex container.
* Uses: *display: flex* in the container.
* Can model a row or column (grids work with both).

“Another major difference between the two is that Flexbox takes a basis in the content while Grid takes a basis in the layout.”

1. CSS Grid layout is used to create a grid of elements. It utilizes the: *display: grid*, *grid-template-columns*, and *gird-column* properties.

|  |  |
| --- | --- |
| Reference | Comments |
| [CSS Grid Layout Module](https://www.w3schools.com/css/css_grid.asp) | [W3schools] |
| [CSS grid layout](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_grid_layout) | There are many “Guides” towards the bottom of the page. [developer.mozilla] |
| [Basic concepts of grid layout](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_grid_layout/Basic_concepts_of_grid_layout) | [developer.mozilla] |
| [CSS Grid Layout](https://www.simplilearn.com/tutorials/css-tutorial/css-grid-layout) | [simplilearn] |
| [CSS Grid Crash Course](https://www.youtube.com/watch?v=0xMQfnTU6oo) | Excellent video (54 min) [Traversy Media] |

1. CSS Flexbox is used to arrange elements. It utilizes the: *display: flex*, *justify-content* (*center, flex-start, flex-end*), and *align-content*. Other sources: [CSS Flexbox](https://www.simplilearn.com/tutorials/css-tutorial/css-flexbox). [CSS Grid vs. Flexbox](https://www.simplilearn.com/tutorials/css-tutorial/css-grid-vs-flexbox).

|  |  |
| --- | --- |
| Reference | Comments |
| [CSS Flexbox](https://www.w3schools.com/css/css3_flexbox.asp) | [W3schools] |
| [Basic concepts of flexbox](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_flexible_box_layout/Basic_concepts_of_flexbox) | Other related topics in left menu. [developer.mozilla] |
| [CSS Flexbox: he Best Guide](https://www.simplilearn.com/tutorials/css-tutorial/css-flexbox) | [simplilearn] |
| [CSS Grid Vs. Flexbox: A Tutorial](https://www.simplilearn.com/tutorials/css-tutorial/css-grid-vs-flexbox) | [simplilearn] |
| [Flexbox Crash Course 2022](https://www.youtube.com/watch?v=3YW65K6LcIA) | Excellent video (47 min) [Traversy Media] |

1. [Media Queries](https://www.w3schools.com/css/css3_mediaqueries.asp) are a way to conditionally apply a CSS rule, depending on the screen size. Also see [Examples](https://www.w3schools.com/css/css3_mediaqueries_ex.asp).

|  |  |
| --- | --- |
| Reference | Comments |
| [Intro](https://www.w3schools.com/css/css3_flexbox.asp) | [W3schools] |
| [Beginner’s guide to media queries](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_flexible_box_layout/Basic_concepts_of_flexbox) | Other related topics in left menu. [developer.mozilla] |
| [Using media queries](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_media_queries/Using_media_queries) | More detailed [developer.mozilla] |
| [Printing](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_media_queries/Printing) | [developer.mozilla] |

# Video – Responsive Web Development Tips

[Responsive Web Development Tips That Everyone Should Know](https://www.youtube.com/watch?v=vZB1s8J6dhY) (len: 9:40)

|  |  |
| --- | --- |
| **Time** | **Description** |
| 0:00 | Responsive Design |
| 7:02 | SASS Preprocessor |
| 7:40 | BEM (Block Element Modifier) |
| 8:36 | What is next for CSS |

# Responsive Design Tutorials

## W3Schools

Go through this in class: <https://www.w3schools.com/html/html_responsive.asp>

* What is responsive design?
* Setting the viewport (another source: [Viewport meta tag](https://developer.mozilla.org/en-US/docs/Web/HTML/Viewport_meta_tag))
* Responsive images – max-width property, picture property (scan/ignore)
* Responsive text size – use the “vw” unit for text sizes (viewport width)
* Media queries – Different styles for different browser sizes. ([Another source: Using media queries](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_media_queries/Using_media_queries))
* Full Example
* Frameworks:
  + w3.css – smaller, alternative to bootstrap, independent of jQuery and any other JS library. Shows example with no explanation, and provides link to tutorial: <https://www.w3schools.com/w3css/default.asp>. We will NOT cover this
  + Bootstrap - uses jQuery, most popular of frameworks. Shows example with no explanation, and provides link to tutorial: <https://www.w3schools.com/bootstrap/default.asp>. We will cover this later.

## freeCodeCamp

Go through this in class: <https://www.freecodecamp.org/news/learn-responsive-web-design-in-5-minutes/>

Appendix

1. Responsive Design Resources

Responsive Design, UI Design, RD-Let Device to the Work. More detail than w3schools. High-level. Useful.

* <https://www.interaction-design.org/literature/topics/responsive-design>
* <https://www.interaction-design.org/literature/topics/ui-design-patterns>
* <https://www.interaction-design.org/literature/article/responsive-design-let-the-device-do-the-work>
* <https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Responsive_Design> - First bit is useful, detailed later

The Encyclopedia of Human-Computer Interaction, 2nd Ed. 4000 pages, 53 chapters.

* <https://www.interaction-design.org/literature>

HTML 5

* <https://www.tutorialspoint.com/html5/html5_new_tags.htm> – New tags in HTML 5 (2014)
* <https://www.techonthenet.com/html/index.php> – Index on right of all tags, simple example of each
* Header sizes in em: <https://html.spec.whatwg.org/multipage/rendering.html#sections-and-headings>
* HTML spec: <https://html.spec.whatwg.org/multipage/rendering.html>

CSS

The is the W3School tutorial on CSS. The items in blue were (mostly) not considered in class

|  |  |
| --- | --- |
| [CSS Tutorial](http://www.w3schools.com/css/default.asp)   1. CSS HOME – Already covered 2. CSS Introduction – Already covered 3. CSS Syntax – Already covered 4. CSS Selectors – Already covered 5. CSS How To – Already covered 6. CSS Colors – Already covered 7. CSS Backgrounds – Already covered 8. CSS Borders – Already covered 9. CSS Margins – Already covered 10. CSS Padding – Already covered 11. CSS Height/Width – Already covered 12. CSS Box Model – Already covered 13. CSS Outline – Already covered 14. CSS Text – Already covered 15. CSS Fonts – Already covered 16. CSS Icons - Omit 17. CSS Links – Scan quickly 18. CSS Lists – Already covered 19. CSS Tables – Already covered | 1. CSS Display – Scan 2. CSS Max-width - Scan 3. CSS Position – Read carefully 4. CSS Overflow - Scan 5. CSS Float – Read carefully 6. CSS Inline-block - Read carefully 7. CSS Align – Read carefully 8. CSS Combinators – Already covered 9. CSS Pseudo-class – Read carefully 10. CSS Pseudo-element – Scan 11. CSS Opacity – Omit 12. CSS Navigation Bar – Scan 13. CSS Dropdowns – Scan 14. CSS Image Gallery – Omit 15. CSS Image Sprites – Omit 16. CSS Attr Selectors – Scan Quickly 17. CSS Forms – Read carefully 18. CSS Counters – Scan Quickly 19. CSS Website Layout – Read very carefully 20. CSS Units – Scan quickly 21. CSS Specificity – Scan quickly |

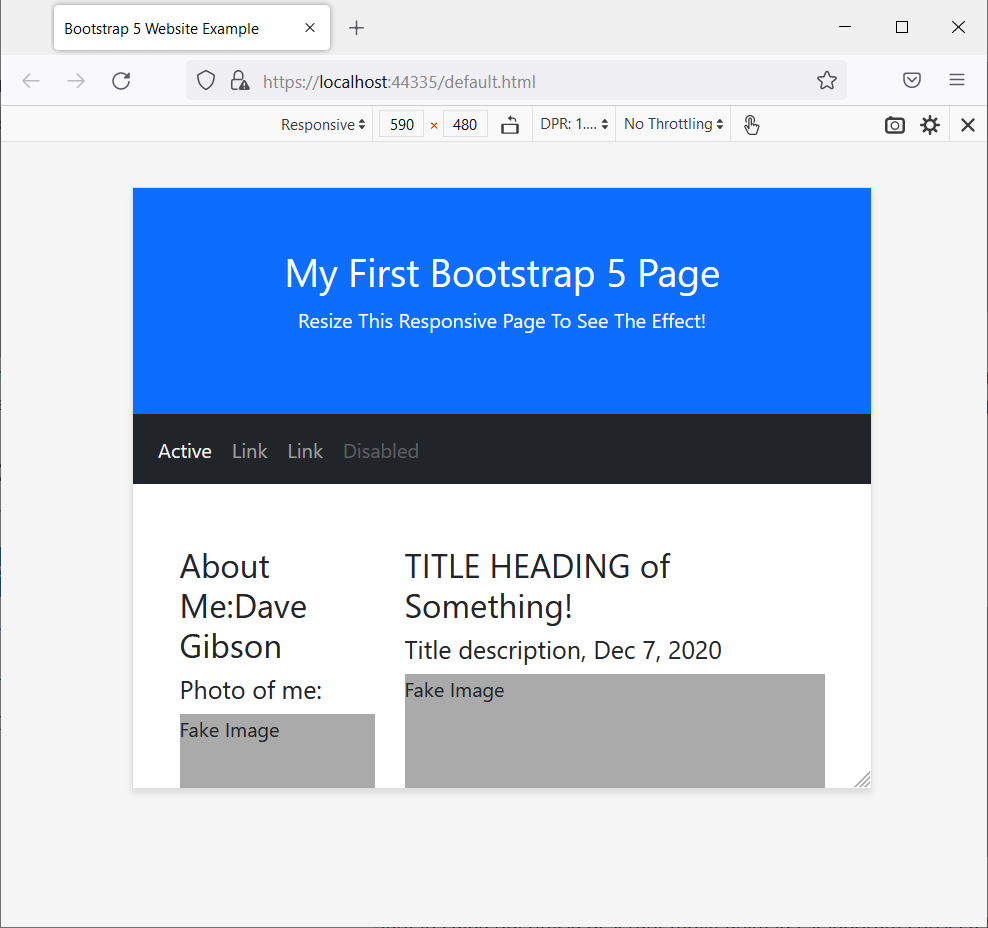
The “Advanced” section at the bottom of the tutorial also has some useful topics:

|  |  |
| --- | --- |
| [CSS Advanced Tutorial](http://www.w3schools.com/css/default.asp)   1. CSS Rounded Corners – Scan quickly 2. CSS Border Images – Omit 3. CSS Backgrounds – Omit 4. CSS Colors – Scan quickly, only first section: RGBA Colors 5. CSS Gradients – Omit 6. CSS Shadows – Omit 7. CSS Text Effects – Scan quickly, only first section: CSS Word Wrapping 8. CSS Web Fonts – Omit 9. CSS 2D Transforms – Omit 10. CSS 3D Transforms – Omit 11. CSS Transitions – Omit 12. CSS Animations – Omit | 1. CSS Tooltips – Scan quickly 2. CSS Style Images – Omit 3. CSS object-fit – Omit 4. CSS Buttons – Scan 5. CSS Pagination – Scan quickly 6. CSS Multiple Columns – Scan quickly 7. CSS User Interface – Scan quickly, only the second section: CSS Resizing 8. CSS Variables – Scan quickly 9. CSS Box Sizing – Read carefully 10. CSS Media Queries – Read carefully 11. CSS MQ Examples – Scan quickly, just the first 3 examples. 12. CSS Flexbox 13. CSS Flexbox – Read carefully 14. CSS Flex Container – Scan 15. CSS Flex Items – Scan quickly, just the first 2 sections: Child Elements & The order Property 16. CSS Flex Responsive – Scan |

1. Firefox Tools
   1. Responsive Design Mode

Firefox has a tool to allow you to see what you page looks like with different screen sizes, without having to resize the browser. It allows you to resize the web page, inside the browser, without changing the size of the browser itself.

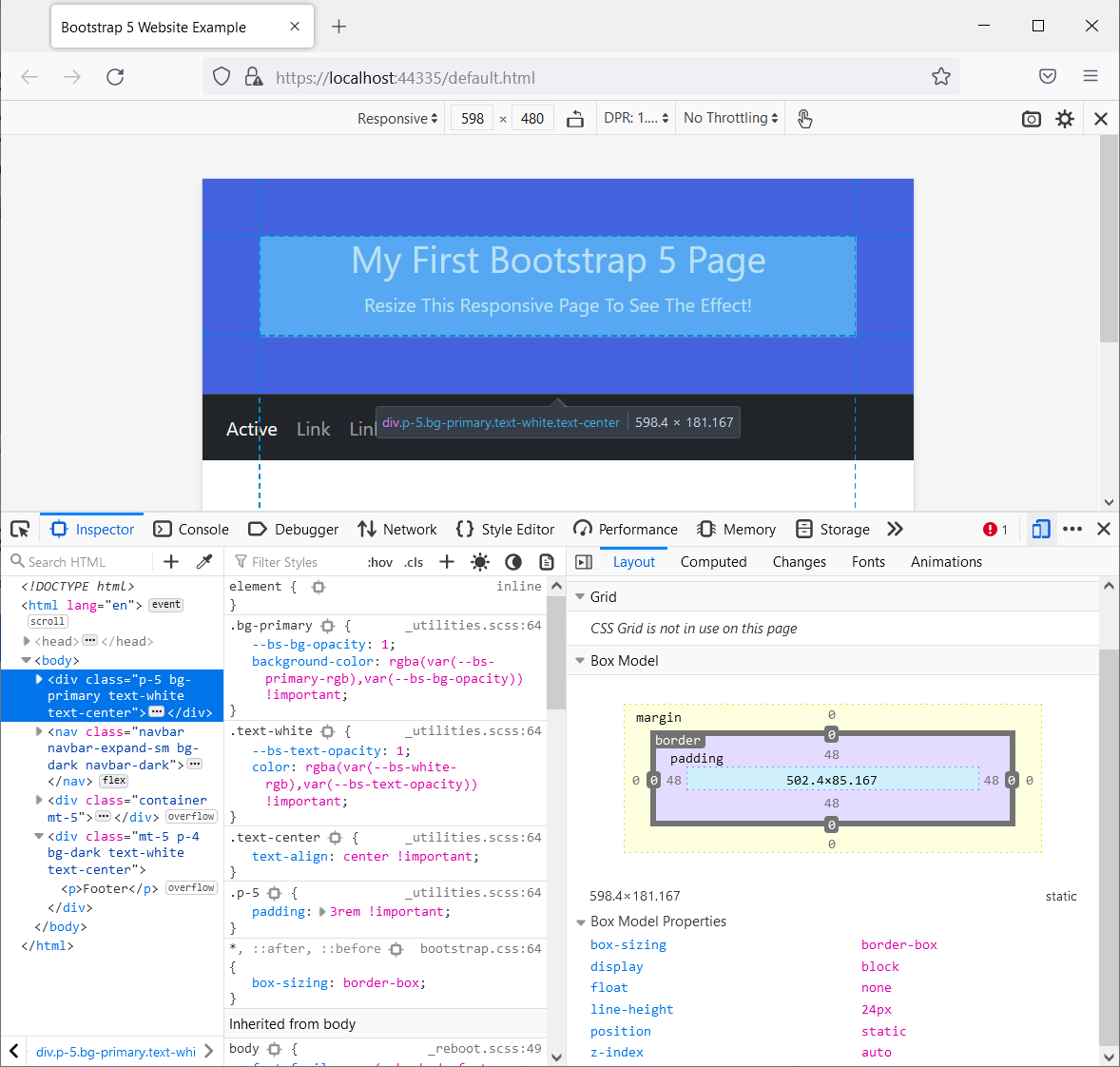
Choose: Application Menu (Hamburger menu), More Tools, Responsive Design Mode.



* 1. Developer Tools

You can also add the Developer Tools. Under Inspector, can select a tag and see what CSS is used and the size of the element including borders, padding, margins.

Choose: Application Menu (Hamburger menu), More Tools, Responsive Design Mode.



1. <https://medium.muz.li/what-is-awd-5-great-adaptive-web-design-examples-for-inspiration-3218e9f624ea> [↑](#footnote-ref-1)