Create a Responsive HTML/CSS Framework

Contents

Responsive Frameworks

A responsive framework is a set of tools consisting of standardized files and code that can be used as the basis of a responsive website.

A framework is a set of standardized tools or concepts used to address certain types of projects. With web development in particular, frameworks consist of standardized files and code that can be used as the basis of a website. Frameworks may be positioned to provide a responsive site structure.

Responsive frameworks may include an HTML skeleton document, responsive CSS classes and elements like grids or image containers, basic styling and setup, and other files like JavaScripts that may add some extra functionality or behavior to the site.

There are many responsive frameworks available that can be used to structure the beginning of a website. You can also create your own basic framework if you'd like to customize websites that you build in the future. You may want to use a framework to structure your website so that you don't have to start from scratch each time you build a new site. This manual includes how to build a framework containing a CSS grid and header sizing scheme.

Create a Basic Responsive CSS Grid

These steps will guide you through the process of creating a basic two-column responsive CSS grid.

1. Open your HTML document and create a basic div-based structure:

```
<div class="container">
    <div class="header"></div>
    <div class="main"></div>
    <div class="sidebar"></div>
    <div class="footer"></div>
</div>
```

2. Open the corresponding CSS document and style the container. The width you set to the container will determine the max width of all content. Use a percent-based width in order for the container to be resizable.

```
.container {
    width: 90%;
    margin: 0 auto; /* to center the container */
}
```

3. Style your main and sidebar divs. The widths you set here will determine how the columns fit inside the container. The float left lines the columns up side by side.

```
.main {
    width: 75%;
    float: left;
}

.sidebar {
    width: 25%;
    float: left;
}
```

4. Add the clear:both property to your footer. This ensures that the footer is pushed the bottom of the layout since the float property takes the main and sidebar divs out of the normal document flow.

```
.footer {
    clear: both;
}
```

5. Determine the breakpoint(s) at which your content should reflow. This breakpoint may be sized in pixels or in ems. For this example, we will use 568px
6. Add a media query to your CSS document at your predetermined breakpoint and redefine your main and sidebard divs to full width:

```
@media screen (max-width: 568px) {

    .main {
        width: 100%;
    }

    .sidebar {
        width: 100%;
    }

}
```

Responsive CSS Grids

A responsive CSS grid is a set of CSS classes that allow designers to create fluid grid-based layouts for web pages that adapt to various screen sizes and dimensions

Responsive CSS grids allow you to create web page layouts that will adapt to various screen sizes and dimensions. These grids are set to deform and collapse at predetermined pixel-based breakpoints so that content fits fluidly to smaller screens.

Responsive grids can be set up in several different ways, but the most practical for our use is by using percentage-based column widths and implementing media queries to define breakpoints. Grids are generally divided into rows, and in each row a set number of columns can be defined. These columns and their widths and properties are defined using CSS classes.

The advantage to using responsive grids is that your content will flow seamlessly between devices and you can design for larger screens while also taking into consideration how your content will appear on smaller devices.

Related tasks

Set the Viewport Scale on page 5
Sometimes, if the viewport is not properly set, your responsive layouts will not collapse properly on mobile devices. To ensure proper scaling and zoom on mobile browsers, make sure to set the viewport meta tag in the head of each of your HTML documents.

Related reference

Common Device Screen Dimensions on page 5
There are a large variety of different device screen dimensions. This table provides some common sizes to consider when determining breakpoints.

Common Device Screen Dimensions

There are a large variety of different device screen dimensions. This table provides some common sizes to consider when determining breakpoints.

Device	Screen Width	Screen Height
Laptop	1440px	900px
Tablet	768px	1024px
Smartphone	320px	568px

Related concepts

Responsive CSS Grids on page 4
A responsive CSS grid is a set of CSS classes that allow designers to create fluid grid-based layouts for web pages that adapt to various screen sizes and dimensions

Related tasks

Set the Viewport Scale on page 5
Sometimes, if the viewport is not properly set, your responsive layouts will not collapse properly on mobile devices. To ensure proper scaling and zoom on mobile browsers, make sure to set the viewport meta tag in the head of each of your HTML documents.

Set the Viewport Scale

Sometimes, if the viewport is not properly set, your responsive layouts will not collapse properly on mobile devices. To ensure proper scaling and zoom on mobile browsers, make sure to set the viewport meta tag in the head of each of your HTML documents.

1. Open your HTML document.
2. Add the following code to your document head. Setting the width to determine the device-width ensures the optimized display of content, while the initial scale to 1 sets the proper zoom level.

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

3. Save and close your HTML document.
4. Repeat for all documents.

Related concepts

Responsive CSS Grids on page 4
A responsive CSS grid is a set of CSS classes that allow designers to create fluid grid-based layouts for web pages that adapt to various screen sizes and dimensions

Related reference

Common Device Screen Dimensions on page 5
There are a large variety of different device screen dimensions. This table provides some common sizes to consider when determining breakpoints.

Create a Header Font Scheme

Fonts and headers should follow a general sizing scheme. Use CSS to size your headers appropriately. To

1. Open your CSS document and add rules for each header you wish to syle and size it accordingly:

```
h1 {
    font-size: 36px;
}
```

2. If you would like the header to resize at a smaller screen width, add a media query containing the header(s) in question with an appropriate size:

```
@media screen (max-width: 568px) {

    h1 {
        font-size: 24px;
    }

}
```

Common Header Sizes

Headers may be sized in px or em. This table provides some standard relative header sizes. These sizes may be adjusted based on your design needs.

Header	font-size
h1	36px
h2	24px
h3	14px
h4	18px