

React Codelab Day 3

Styling War



@Vattanan.bu @Chalath.mo

Warning

This slide is not cover all best practice in CSS Programming.



Table Content

- Configuration typography
- Inline style sheet
- External style sheet
- Responsive
- Reference

Configuration typography



Configuration typography

- Same font-family
- Have font scale
- ETC.

Configuration typography

[Typography.js](#) is a powerful toolkit for building websites with beautiful design

Configuration typography

[Typography.js](#)

- Open source
- Easy to use
- Multi theme support

Configuration typography

See in code.

Inline style sheet

CSS



Inline

Inline style sheet

In normal HTML

```
18  
19  
20 <h1 style="color:blue;margin-left:30px; flex: 0">This is a heading</h1>  
21  
22  
23  
24
```

In JSX

```
34  
35  
36 <h1 style={{ color: 'blue', marginLeft: '30px', flex: 0 }}>This is a heading</h1>  
37  
38  
39
```

Inline style sheet

Use case

```
31
32 const headerStyle = {
33   color: 'blue',
34   marginLeft: '30px',
35   flex: 0
36 };
37
38
39
40 <h1 style={ headerStyle }>This is a heading</h1>
41
42
```

```
31
32 const headerStyle = {
33   color: 'blue',
34   marginLeft: '30px',
35   flex: 0
36 };
37
38 const fixHieght = '90px';
39
40
41
42 <h1 style={{ ...headerStyle, height: fixHieght }}>This is a heading</h1>
43
```

Inline style sheet

Use case

```
32  const headerStyle = {  
33    color: 'blue',  
34    marginLeft: '30px',  
35    flex: 0  
36  };  
37  
38  const scale = 2;  
39  
40  
41  
42  <h1 style={{ ...headerStyle, flex: 1 + scale }}>This is a heading</h1>  
43  
44
```

```
33  calculateFlex = (num) => {  
34    const scale = 2;  
35    return num + scale;  
36  }  
37  
38  
39  
40  <h1 style={{ ...headerStyle, flex: this.calculateFlex(1) }}>This is a heading</h1>  
41  
42
```

Inline style sheet

Use case

```
32  const headerStyle = {  
33    color: 'blue',  
34    marginLeft: '30px',  
35    flex: 0  
36  };  
37  
38  const scale = 2;  
39  
40  
41  
42  <h1 style={{ ...headerStyle, flex: 1 + scale }}>This is a heading</h1>  
43  
44
```

```
33  calculateFlex = (num) => {  
34    const scale = 2;  
35    return num + scale;  
36  }  
37  
38  
39  
40  <h1 style={{ ...headerStyle, flex: this.calculateFlex(1) }}>This is a heading</h1>  
41  
42
```

Inline style sheet

Use case

```
34  
35 |  
36 <h1 style={{ color: this.state.headerColor, height: this.props.headerHeight }}>This is a heading</h1>  
37  
38
```

Inline style sheet

Not Recommend

External style sheet

CSS



External

External style sheet

In normal HTML

Define

```
body {  
    background-color: linen;  
}  
.text-header {  
    color: maroon;  
    margin-left: 40px;  
}
```

Import (Import CSS file in html file)

```
<head>  
<link rel="stylesheet" type="text/css" href="mystyle.css">  
</head>
```

Use (class="...")

```
<h1 class="text-header">This is a heading</h1>  
<p>This is a paragraph.</p>
```

External style sheet

In JSX

Define

```
body {  
  background-color: linen;  
}  
.text-header {  
  color: maroon;  
  margin-left: 40px;  
}
```

Import (Import CSS file in JavaScript file)

```
import './mystyle.css';
```

Use (className="...")

```
<h1 className="text-header">This is a heading</h1>  
<p>This is a paragraph.</p>
```

Responsive



Responsive

- Viewport
- Responsive Image
- Media Query.

Responsive

Viewport

- **Viewport** is the user's visible area of a web page.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Responsive

Viewport

- **Problem**

Before tablets and mobile phones, web pages were designed only for computer screens, and it was common for web pages to have a static design and a fixed size.

- **Fix**

Then, when we started surfing the internet using tablets and mobile phones, fixed size web pages were too large to fit the viewport. To fix this, browsers on those devices scaled down the entire web page to fit the screen.

Viewport

Responsive

Not have ViewPort



Have ViewPort



Remark: Small Screen Device

Responsive

Image Responsive

1. Auto scale
2. Crop Image
3. Use `<picture />`

Responsive

Image Responsive

1. Auto scale

you can **fix** one dimension and **auto scale** of other dimension

In CSS file

```
.responsive {  
    width: 100%;  
    max-width: 400px;  
    height: auto;  
}
```

In JavaScript file

```
<img src={imgNature} alt="Nature" className="responsive" />
```

Responsive

Image Responsive

2. Crop Image

you can create block for crop image with `overflow: hidden;`

In CSS file

```
.crop {  
  width: 100px;  
  height: 100px;  
  overflow: hidden;  
  display: flex;  
  justify-content: center;  
  align-items: center;  
}
```

In JavaScript file

```
<div className="crop">  
  <img src={imgNature} alt="Nature">  
</div>
```

Responsive

Image Responsive

3. Use <picture />

you can use tag picture for render **images** by **size screen**

In JavaScript file

```
<picture>
  <source media="(min-width: 650px)" srcSet={imgPinkFlowers}>
  <source media="(min-width: 465px)" srcSet={imgWhiteFlower}>
  <img src={imgOrangeFlowers} alt="Flowers">
</picture>
```

Responsive

Media Query

Media Query is a CSS technique introduced in CSS3.

It uses the `@media` rule to include a block of CSS properties only if a certain condition is true.

Responsive

Media Query

In CSS file

```
/* For mobile phones(is default): */
```

```
@media only screen and (min-width: 600px) {  
  /* For tablets: */  
}
```

```
@media only screen and (min-width: 768px) {  
  /* For desktop: */  
}
```

< 600px



< 768px



Other
(>= 768px)



Responsive

Media Query

In CSS file

```
/* For mobile phones(is default): */  
.header-color {  
  color: red;  
}  
  
@media only screen and (min-width: 600px) {  
  /* For tablets: */  
  .header-color {  
    color: green;  
  }  
}  
  
@media only screen and (min-width: 768px) {  
  /* For desktop: */  
  .header-color {  
    color: blue;  
  }  
}
```

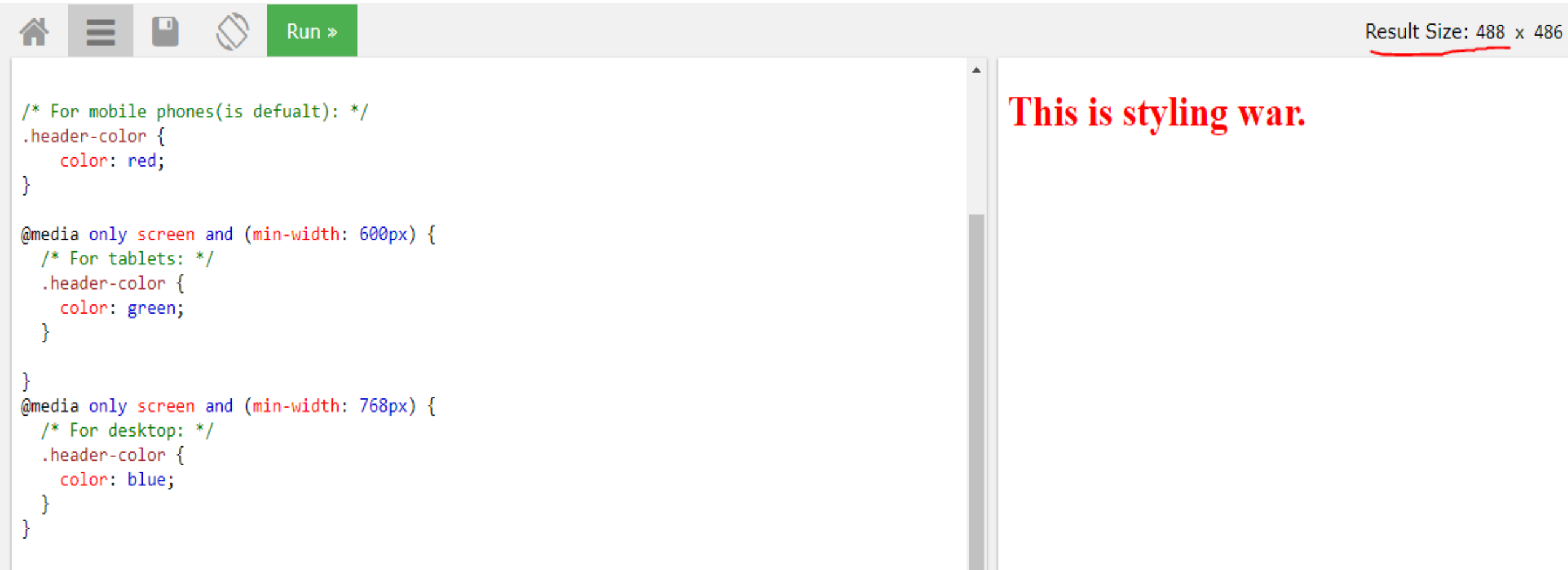
In JavaScript file

```
<h1 className="header-color">This is styling war.</h1>
```

Responsive

Media Query

Example 1



The screenshot shows a web browser interface. The top bar includes a home icon, a menu icon, a save icon, a refresh icon, and a green 'Run »' button. On the right side of the top bar, the text 'Result Size: 488 x 486' is displayed, with '488 x 486' underlined in red. The main content area is split into two panels. The left panel contains CSS code for a responsive design. The right panel shows the rendered output of the code, which is the text 'This is styling war.' in red.

```
/* For mobile phones(is default): */
.header-color {
  color: red;
}

@media only screen and (min-width: 600px) {
  /* For tablets: */
  .header-color {
    color: green;
  }
}

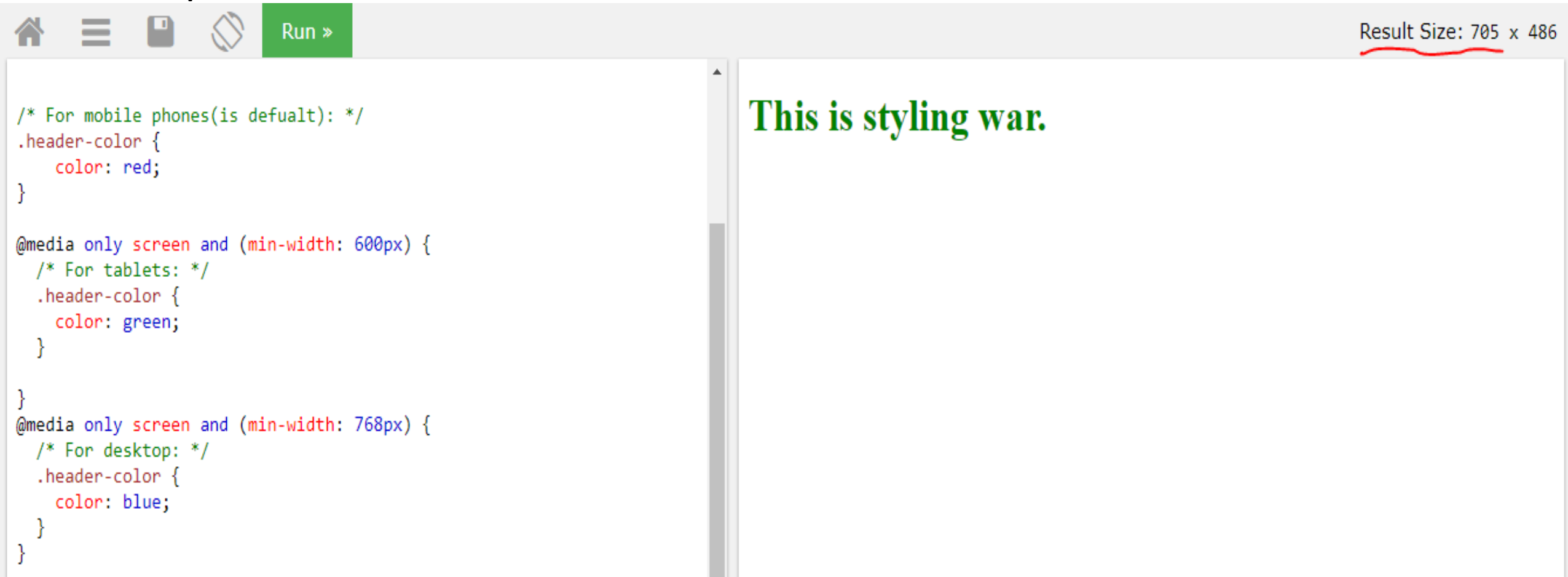
@media only screen and (min-width: 768px) {
  /* For desktop: */
  .header-color {
    color: blue;
  }
}
```

This is styling war.

Responsive

Media Query

Example 1



The screenshot shows a web browser interface. The top bar includes a home icon, a menu icon, a save icon, a refresh icon, and a green 'Run »' button. On the right side of the top bar, the text 'Result Size: 705 x 486' is displayed, with '705 x 486' underlined in red. The main content area is split into two panels. The left panel contains CSS code for a responsive design. The right panel shows the rendered output of the code, which is the text 'This is styling war.' in a green serif font.

```
/* For mobile phones(is default): */
.header-color {
  color: red;
}

@media only screen and (min-width: 600px) {
  /* For tablets: */
  .header-color {
    color: green;
  }
}

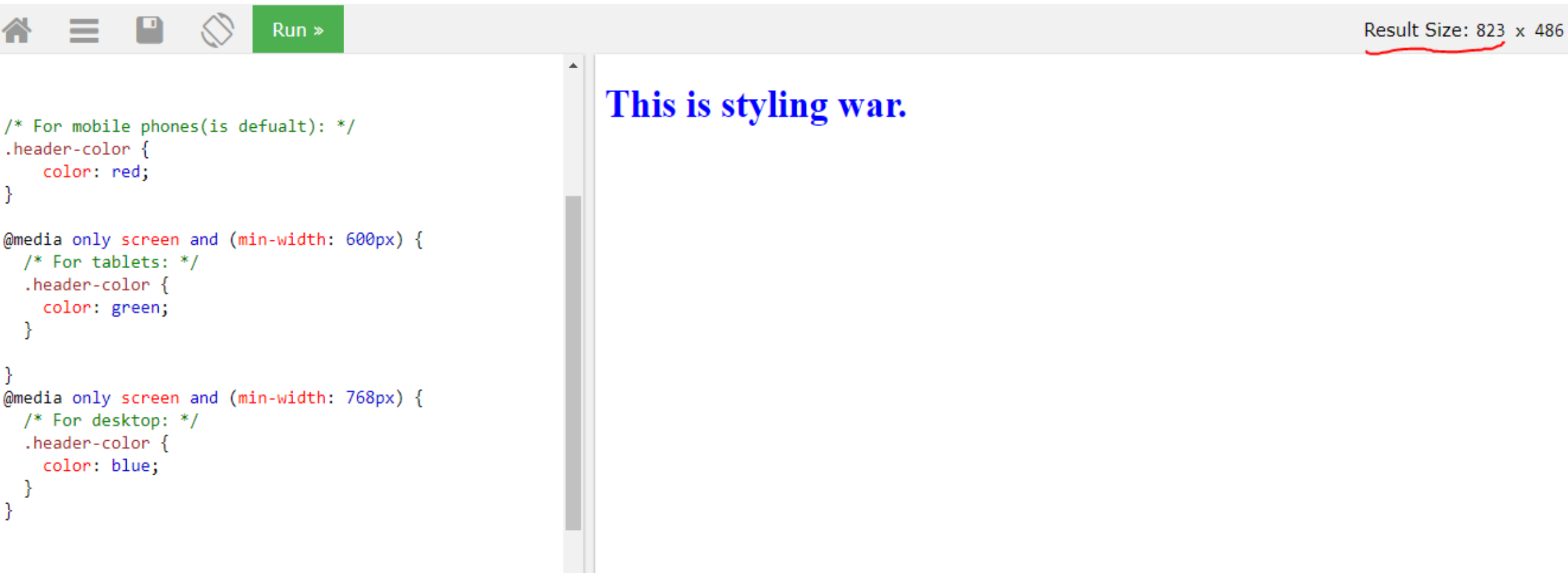
@media only screen and (min-width: 768px) {
  /* For desktop: */
  .header-color {
    color: blue;
  }
}
```

This is styling war.

Responsive

Media Query

Example 1



The screenshot shows a web browser interface. The top bar includes icons for home, menu, save, and print, followed by a green 'Run >>' button. On the right, the text 'Result Size: 823 x 486' is displayed, with '823' underlined in red. The main content area is split into two panels. The left panel contains CSS code for a .header-color class, using media queries to set different colors (red, green, blue) based on screen width. The right panel shows the rendered output: the text 'This is styling war.' in blue.

```
/* For mobile phones(is default): */
.header-color {
  color: red;
}

@media only screen and (min-width: 600px) {
  /* For tablets: */
  .header-color {
    color: green;
  }
}

@media only screen and (min-width: 768px) {
  /* For desktop: */
  .header-color {
    color: blue;
  }
}
```

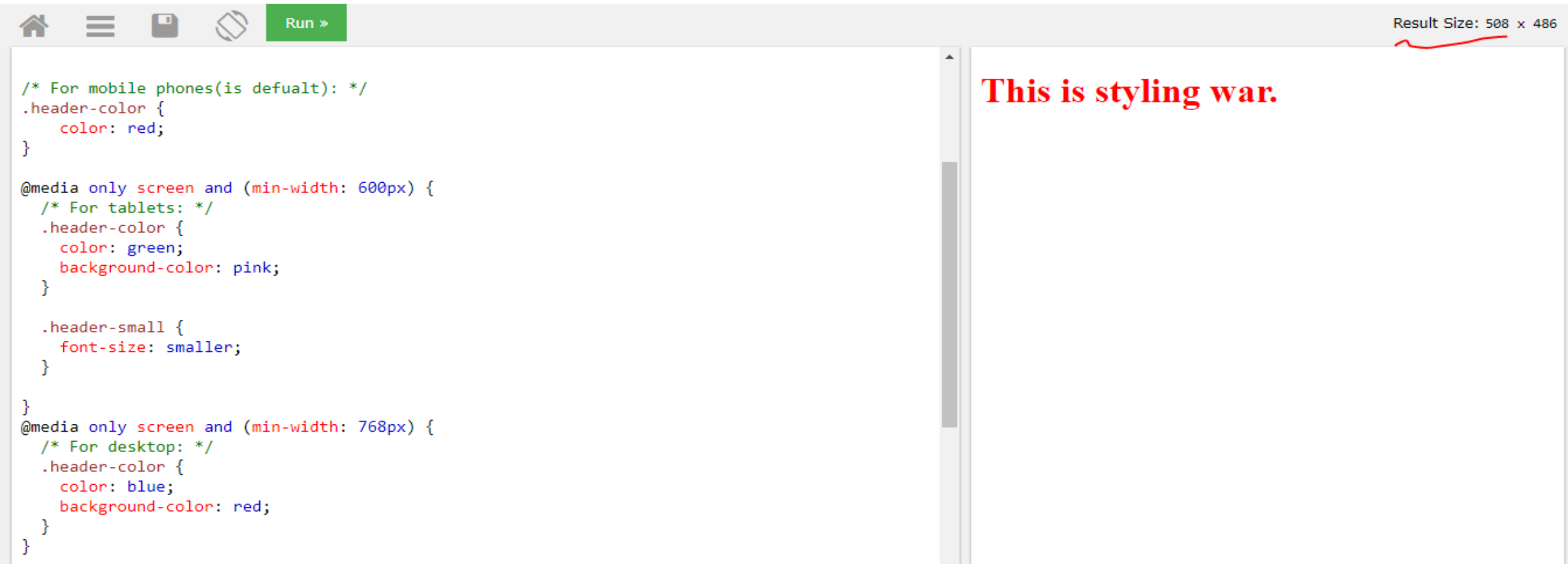
This is styling war.

Result Size: 823 x 486

Responsive

Media Query

Example 2



The screenshot shows a web browser interface with a toolbar at the top containing icons for home, menu, save, and print, along with a green 'Run' button. The browser window displays a web page with the text 'This is styling war.' in red. The browser's status bar at the bottom right indicates 'Result Size: 508 x 486'. On the left side of the browser window, a code editor displays CSS media queries. The code defines three styles for the '.header-color' class based on screen width: default (mobile) is red, tablets (600px+) are green with a pink background, and desktops (768px+) are blue with a red background. The '.header-small' class is also defined with a smaller font size.

```
/* For mobile phones(is default): */
.header-color {
  color: red;
}

@media only screen and (min-width: 600px) {
  /* For tablets: */
  .header-color {
    color: green;
    background-color: pink;
  }

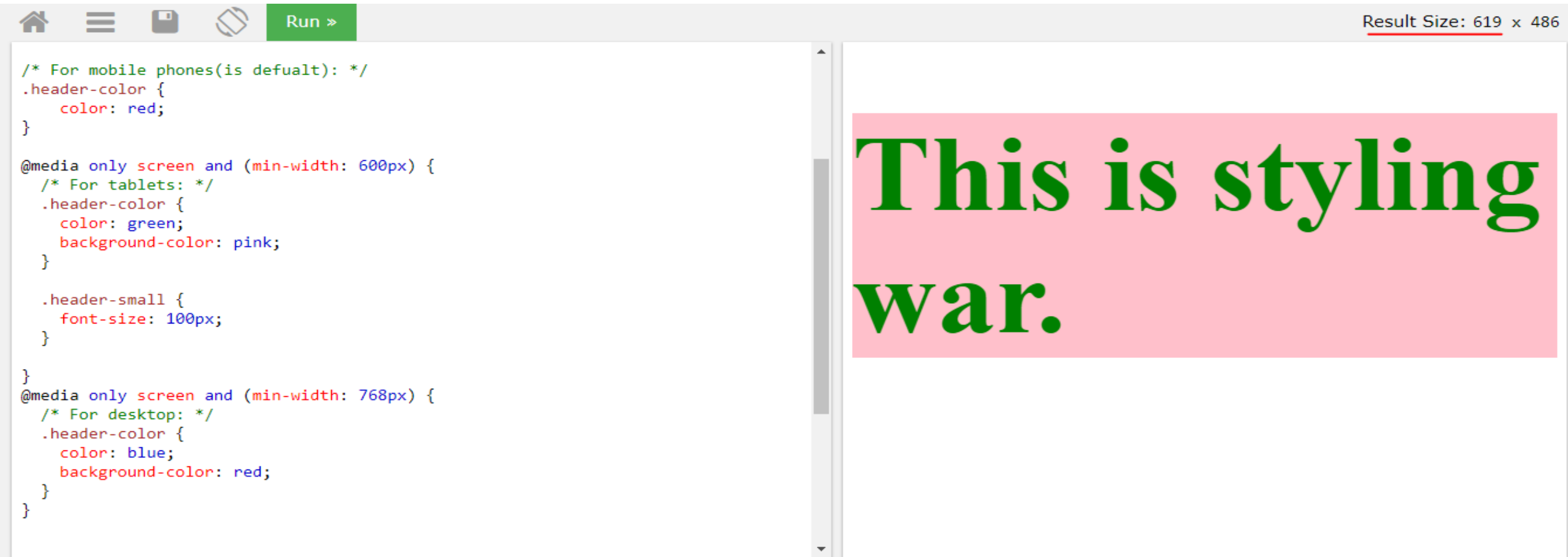
  .header-small {
    font-size: smaller;
  }
}

@media only screen and (min-width: 768px) {
  /* For desktop: */
  .header-color {
    color: blue;
    background-color: red;
  }
}
```

Responsive

Media Query

Example 2



The image shows a web browser interface with a code editor on the left and a preview window on the right. The code editor contains CSS media queries for different screen widths. The preview window shows the result of these queries: a pink rectangular box containing the text "This is styling war." in a large, green, serif font.

Result Size: 619 x 486

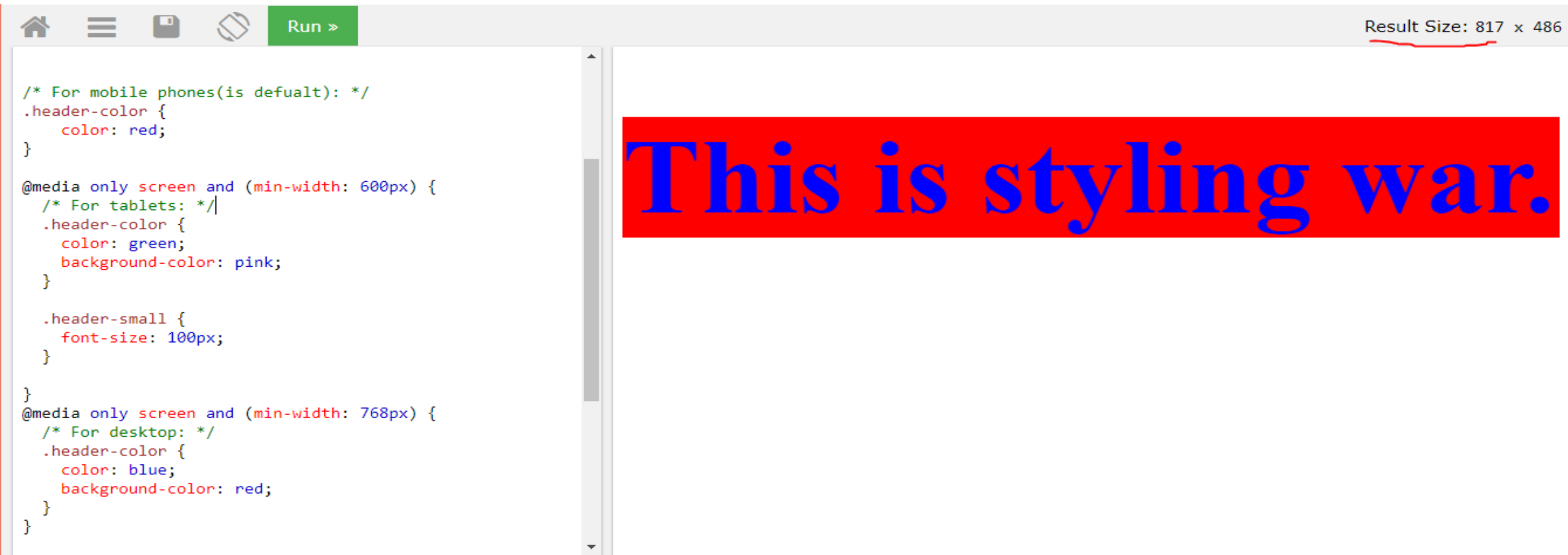
```
/* For mobile phones(is default): */  
.header-color {  
  color: red;  
}  
  
@media only screen and (min-width: 600px) {  
  /* For tablets: */  
  .header-color {  
    color: green;  
    background-color: pink;  
  }  
  
  .header-small {  
    font-size: 100px;  
  }  
}  
  
@media only screen and (min-width: 768px) {  
  /* For desktop: */  
  .header-color {  
    color: blue;  
    background-color: red;  
  }  
}
```

This is styling
war.

Responsive

Media Query

Example 2



The screenshot shows a web browser window with a red banner containing the text "This is styling war." in blue. The browser's developer tools are open, displaying CSS media queries for different screen widths. The top of the browser window shows a toolbar with icons for home, menu, save, and print, along with a "Run" button. The right side of the browser window displays the "Result Size: 817 x 486".

```
/* For mobile phones(is default): */
.header-color {
  color: red;
}

@media only screen and (min-width: 600px) {
  /* For tablets: */
  .header-color {
    color: green;
    background-color: pink;
  }

  .header-small {
    font-size: 100px;
  }
}

@media only screen and (min-width: 768px) {
  /* For desktop: */
  .header-color {
    color: blue;
    background-color: red;
  }
}
```

Reference

- Typography.js:
<https://github.com/KyleAMathews/typography.js>
- CSS:
<https://www.w3schools.com/css/default.asp>
- Style in JSX:
<https://reactjs.org/docs/dom-elements.html>
- Viewport:
https://www.w3schools.com/css/css_rwd_viewport.asp

Reference

- Responsive Image:
https://www.w3schools.com/howto/howto_css_image_responsive.asp
- Tag picture:
https://www.w3schools.com/tags/tag_picture.asp
- Media Query:
https://www.w3schools.com/css/css_rwd_mediaqueries.asp

Thank You

