CSS
MEDIA QUERIES

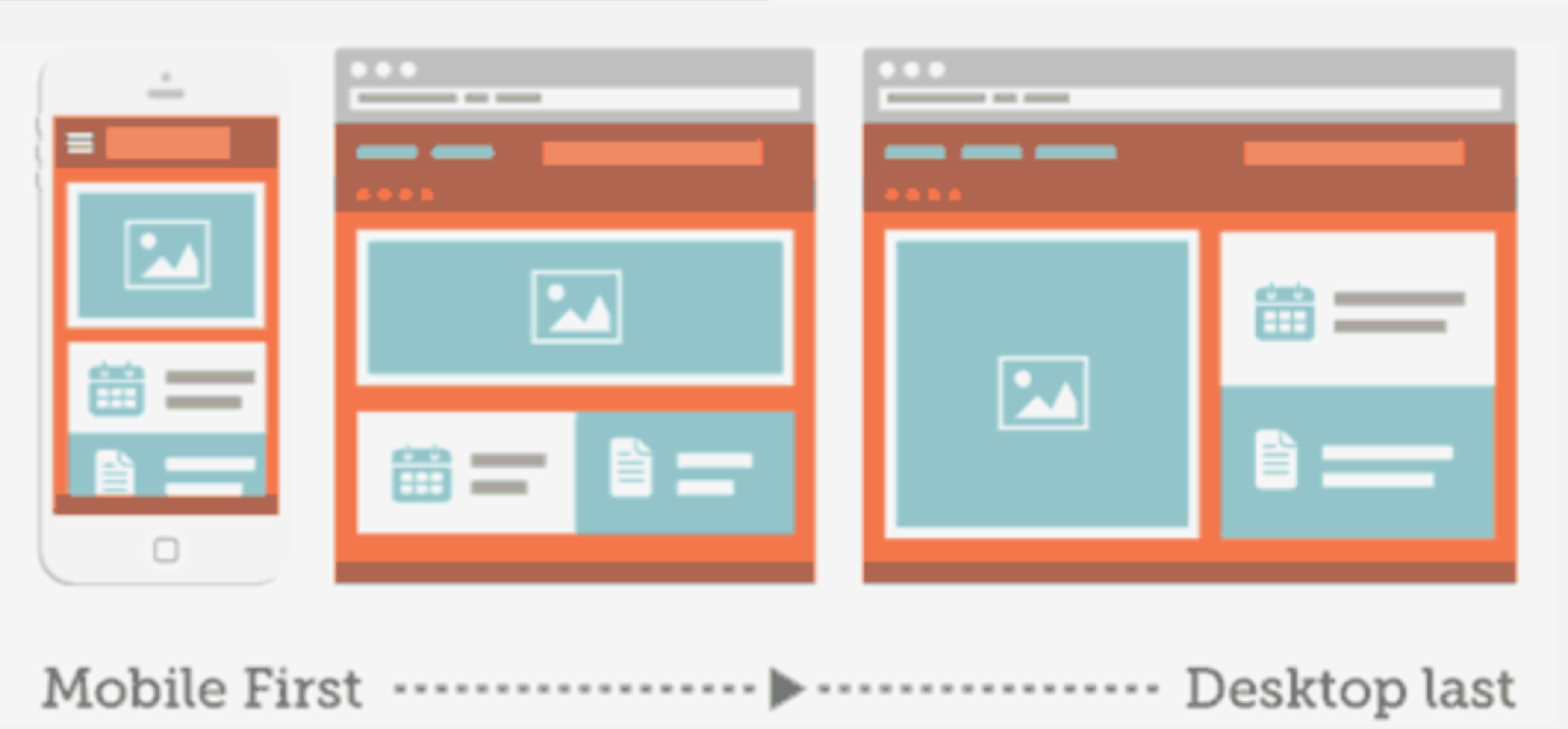
Andrew Sheehan

Boston University
Metropolitan College

WHAT ARE MEDIA QUERIES

A **media query** is method to change the way your webapp looks on different devices.

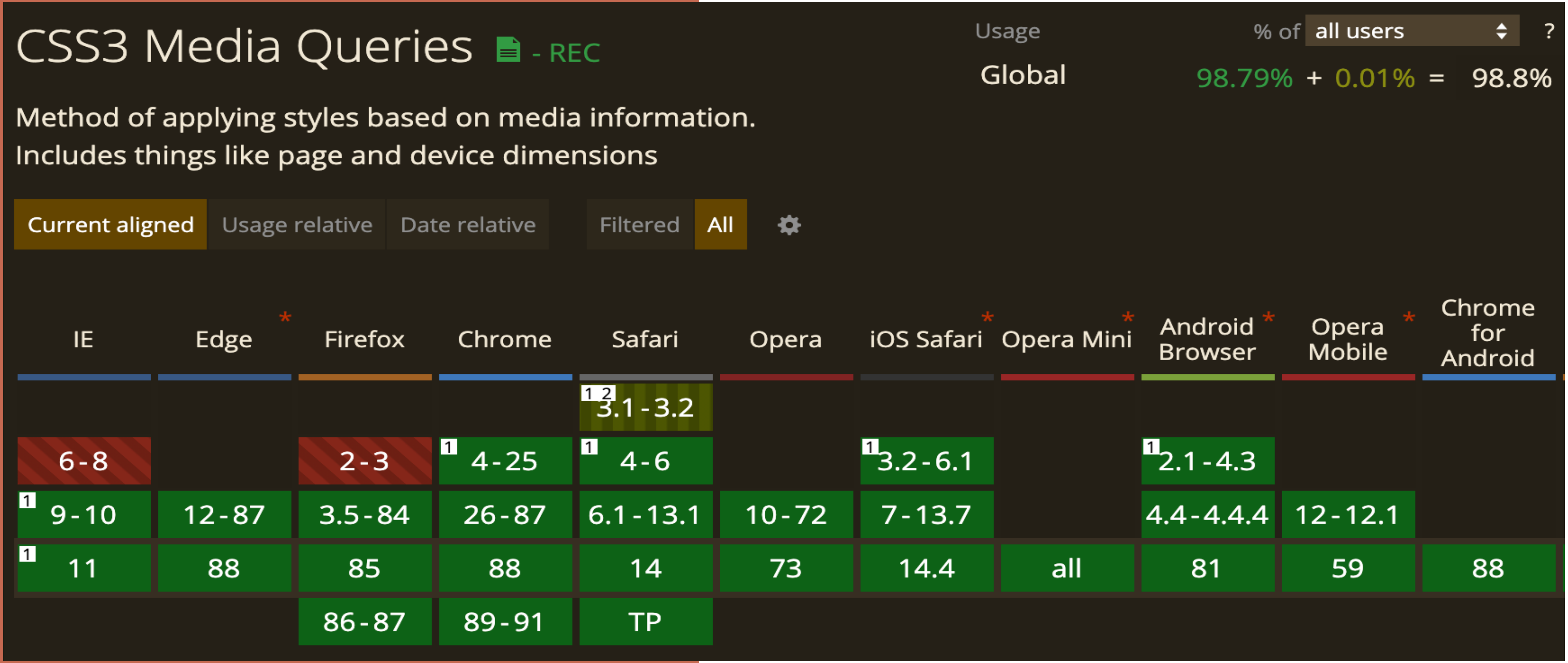
MOBILE FIRST > THEN OTHERS



CONTENT CAN ADAPT TO THE DEVICE

- Based on the device width & height.
- The orientation of the viewport
- Resolution (pixel density)

CANIUSE.COM
MEDIA QUERIES



CSS3 Media Queries - REC

Usage % of all users ?

Global 98.79% + 0.01% = 98.8%

Method of applying styles based on media information. Includes things like page and device dimensions

Current aligned | Usage relative | Date relative | Filtered | All

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android
				[1] [2] 3.1 - 3.2						
6 - 8		2 - 3	[1] 4 - 25	[1] 4 - 6		[1] 3.2 - 6.1		[1] 2.1 - 4.3		
[1] 9 - 10	12 - 87	3.5 - 84	26 - 87	6.1 - 13.1	10 - 72	7 - 13.7		4.4 - 4.4.4	12 - 12.1	
[1] 11	88	85	88	14	73	14.4	all	81	59	88
		86 - 87	89 - 91	TP						

BASIC PRINCIPALS

Media type:

(all, print, screen and speech)

Matching expression:

to trigger the media change

MEDIA QUERIES
DEFAULT

When no media type indicated

“all” will be used (all media types)

MEDIA EXPRESSIONS
MUST EVALUATE TO TRUE

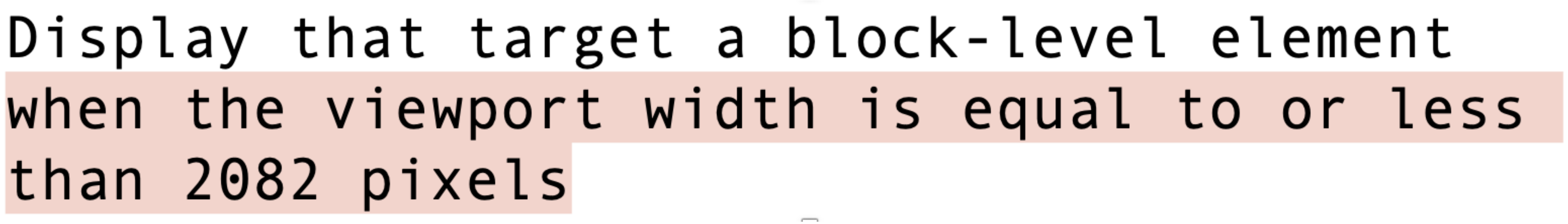
i. The media type of the media query matches the media type of the device.

ii. All expressions in the media query eval to true.

iii. If the media query list is empty (i.e. the declaration is the empty string or consists solely of whitespace) it evaluates to true.

MEDIA QUERIES
EXAMPLE

```
@media (max-width:  2082px) {
    #main_content_area {
            display: block;
    }
}
```



Display that target a block-level element when the viewport width is equal to or less than 2082 pixels

MEDIA TYPES: PRINT

When a user wants to print, they *~probably~* do not want to waste their color ink on graphics or unnecessary content.

```
@media print {
    #header, #left-side-ads, #menu {
            display: none;
    }
}
```

```
<link rel="stylesheet" type="text/css”
    media="screen" href="sans-serif.css">
```

```
<link rel="stylesheet" type="text/css"
    media="print" href="serif.css">
```

EXAMPLE
USING THE LINK ELEMENT

EXAMPLE COLOR SCREENS

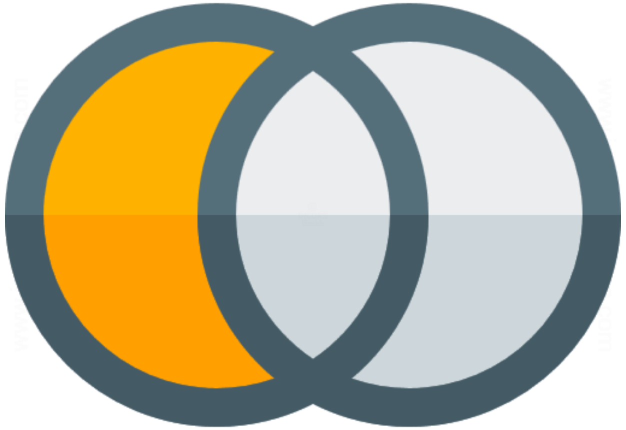
```
<link rel="stylesheet" href=”styles.css"
                     media="screen and (color)" />
```

This style sheet applies to devices of a certain media type ‘screen’, with a certain feature (it must be a color screen).

Here the same media query written in an @import-rule

```
@import url(color.css) screen and (color);
```

LOGICAL
NOT (NEGATION)



It will negate (true/false) the entire expression you are testing.

```
<link rel="stylesheet” href="example.css"
                    media="not screen and (color)" />
```

TAKE NOTICE

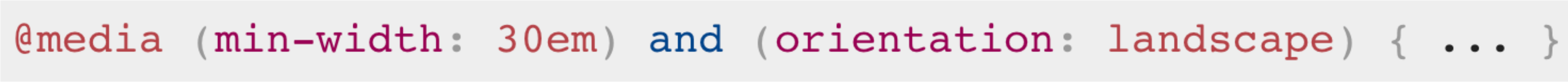
The not keyword can't be used to negate an individual feature query, **only an entire media query**

STANDARD AND EXPERIMENTAL ELEMENTS

Misspalling your media type or having an invalid format or logic will result in **false**

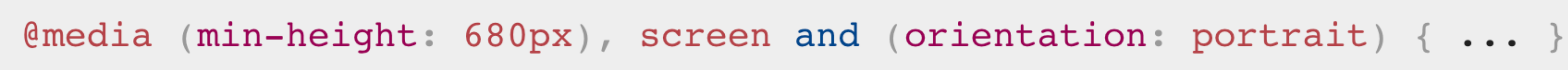
(the media query will be be applied or used)

COMPLEX EXPRESSIONS



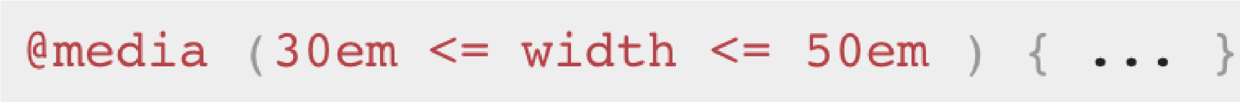
```
@media (min-width: 30em) and (orientation: landscape) { ... }
```

Viewport set to landscape mode with a width of at least 30em’s



```
@media (min-height: 680px), screen and (orientation: portrait) { ... }
```

Viewport with at least a height of 680 pixels or using a screen display in portrait mode

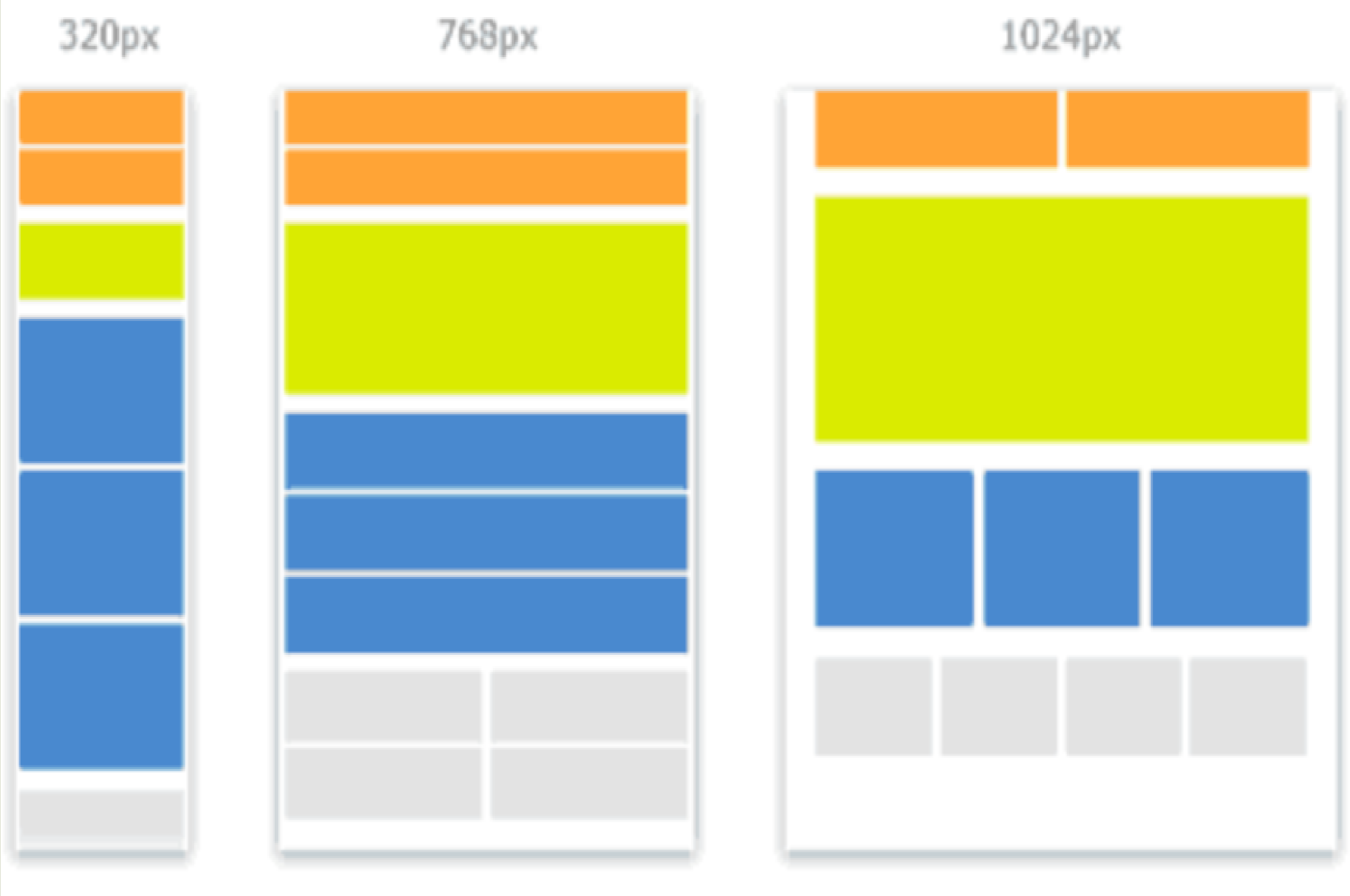


```
@media (30em <= width <= 50em ) { ... }
```

Viewport min-width and max-width between 30em & 50em
(Example using new, Level 4 media query syntax)

GENERAL PRINCIPAL

Shift content (or remove) to be the width of your client's device



```
@media screen and (max-width:  520px) {
  /* CSS rules go inside brackets */
}
@media screen and (max-width:  980px) {
  /* CSS rules go inside brackets */
}

@media screen and (max-width:  1440px) {
  /* CSS rules go inside brackets */
}
```

STANDARD USE OR EXAMPLE