VIA University College

# Web Development 1

Responsiveness, Flex layout

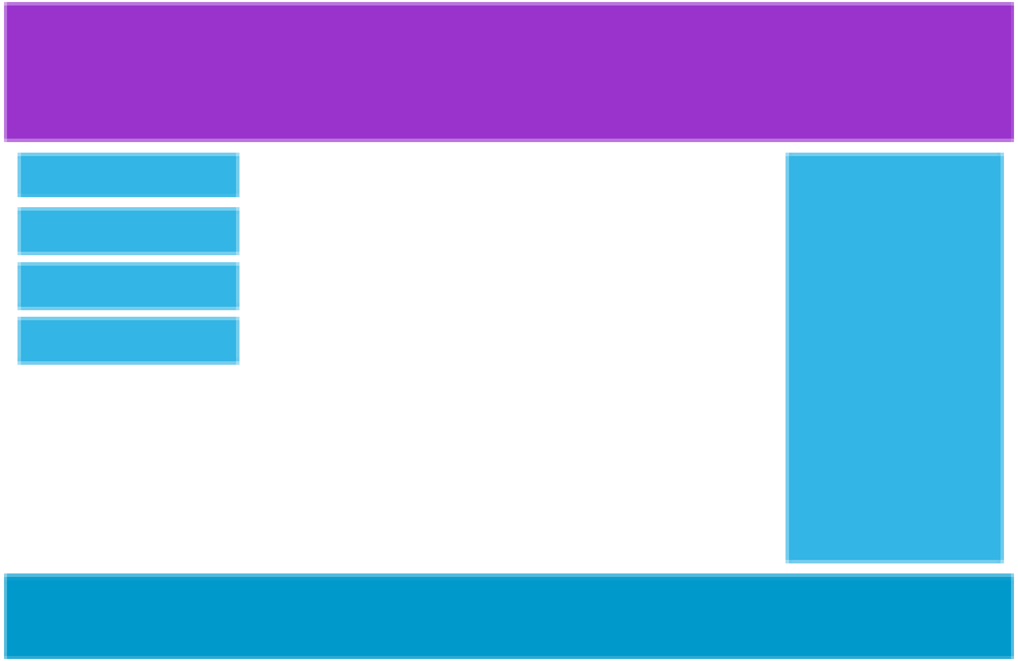
# Agenda

- Recap
- What is Responsive Web Design?
- Box-sizing
- Media Queries
- Flexbox
- Developer tools (in Chrome)

# What is Responsive Web Design

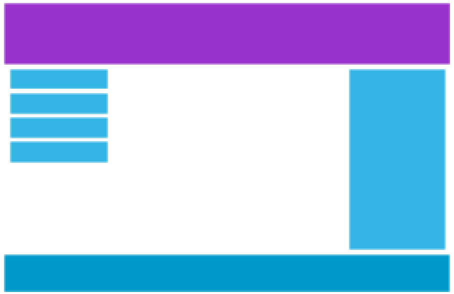
- Responsive web design makes your web page look good on all devices
- Responsive web design uses only HTML and CSS
- Responsive web design is not a program or a JavaScript
- The HTML code is the same; all layout changes are done in CSS

# What is Responsive Web Design

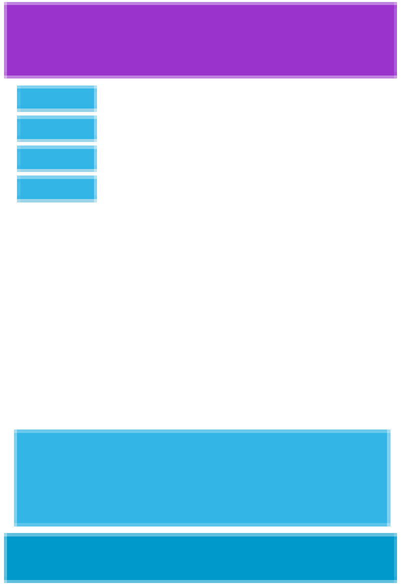


Desktop

# What is Responsive Web Design

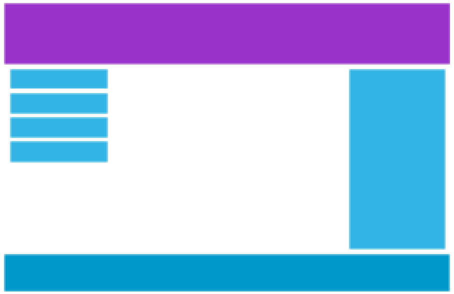


Desktop

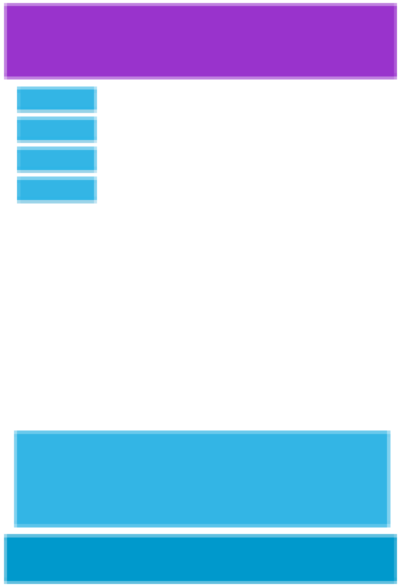


Tablet

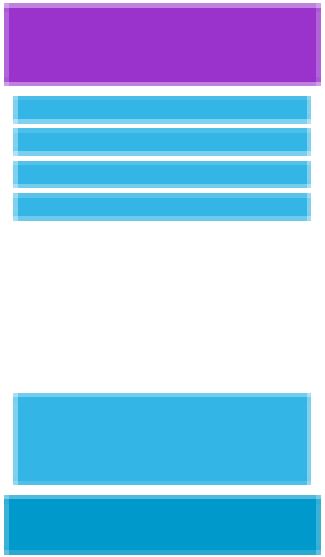
# What is Responsive Web Design



Desktop



Tablet



Phone

# What is Responsive Web Design

Your layout changes based on platform and viewport size

Check out this example: [W3Schools Tryit Editor]()

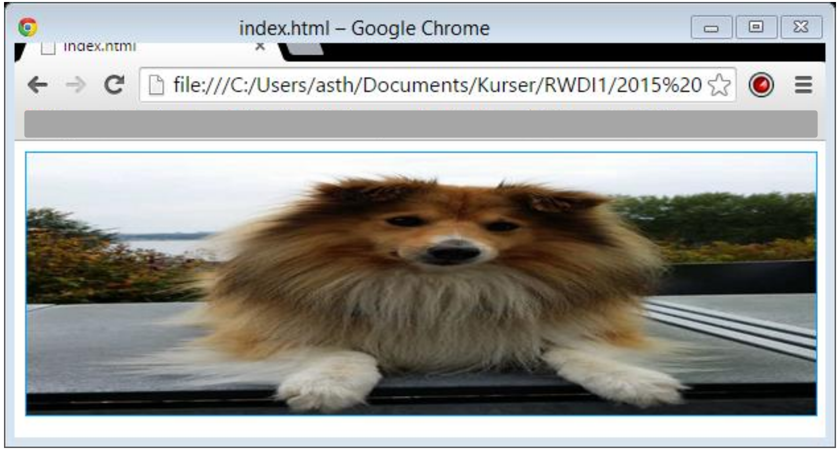
# Responsive Web Design - Images

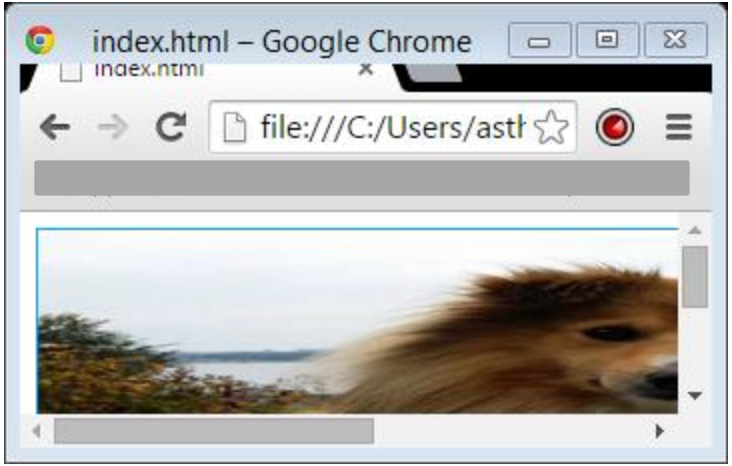
- We can control sizing of images
- Static (pixel dimensions) or dynamic (percentage dimensions)
- Size is generally always inherited outward to inward

# Image – Fixed Size

```
img {
        width: 600px;
        height: 200px;
}
```

Element size is set regardless of the size of parent and children elements.

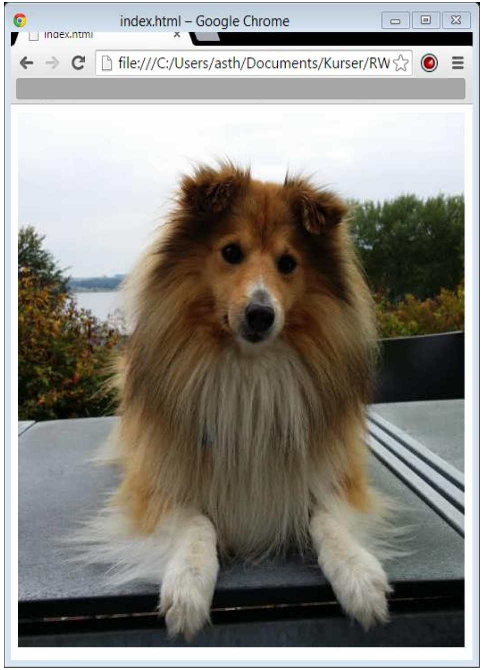

index.html – Google Chrome
index.html
file:///C:/Users/asth/Documents/Kurser/RWDI1/2015%20

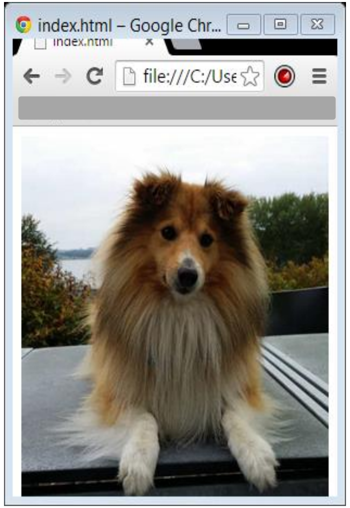


# Image – Relative Size

```
img {
          width: 600px;
          height: auto; //Default value
}
```

Element size is set relative to size of parent (in this case the browser window view/the <body> element).


index.html – Google Chrome
index.html
file:///C:/Users/asth/Documents/Kurser/RW

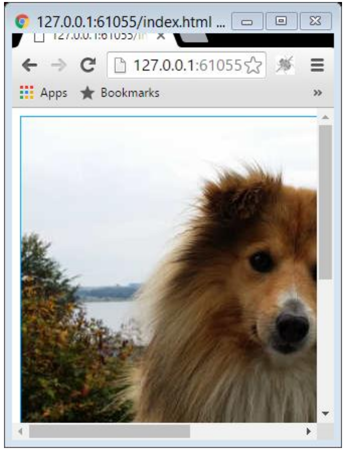


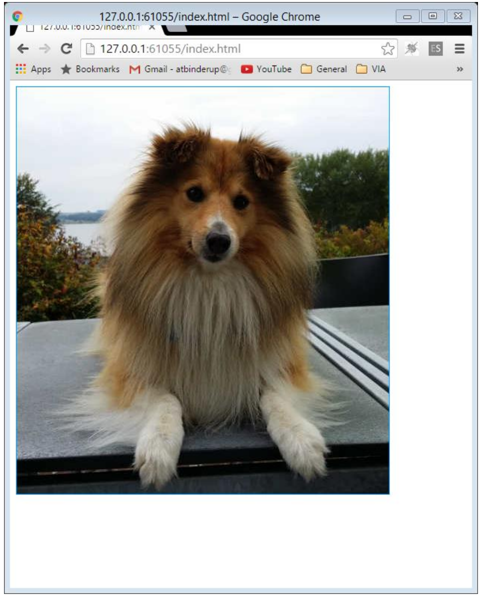
# Image – Relative Size

```
img {
          width: auto;
          height: 100%;
}
```

Does not work when the parent element is the browser view/the <body> element, but otherwise will.
What we see here is the default sizing.
To fix add:

```
html, body {
    height: 100%;
}
```



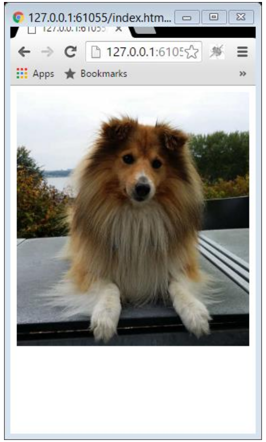

127.0.0.1:61055/index.html – Google Chrome
127.0.0.1:61055/index.html
Apps
Bookmarks
Gmail - atbinderup@
YouTube
General
VIA

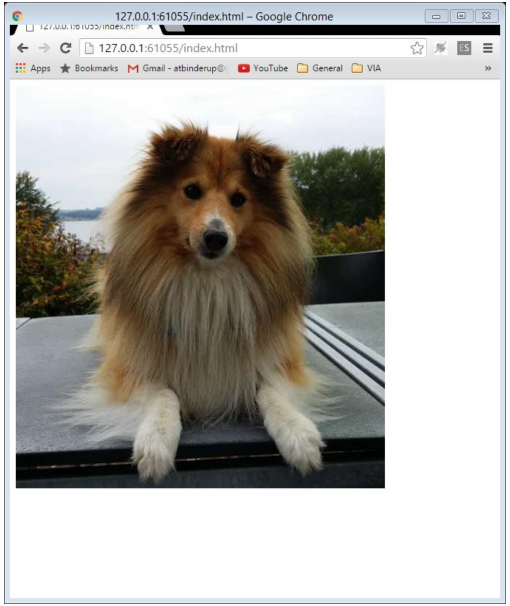
# Image – Relative Size

```
img {
        max-width: 100%;
        height: auto;
}
```

If the `max-width` property is set to `100%`, the image will scale down if it has to, but never scale up to be larger than its original size.

This only works on images though.



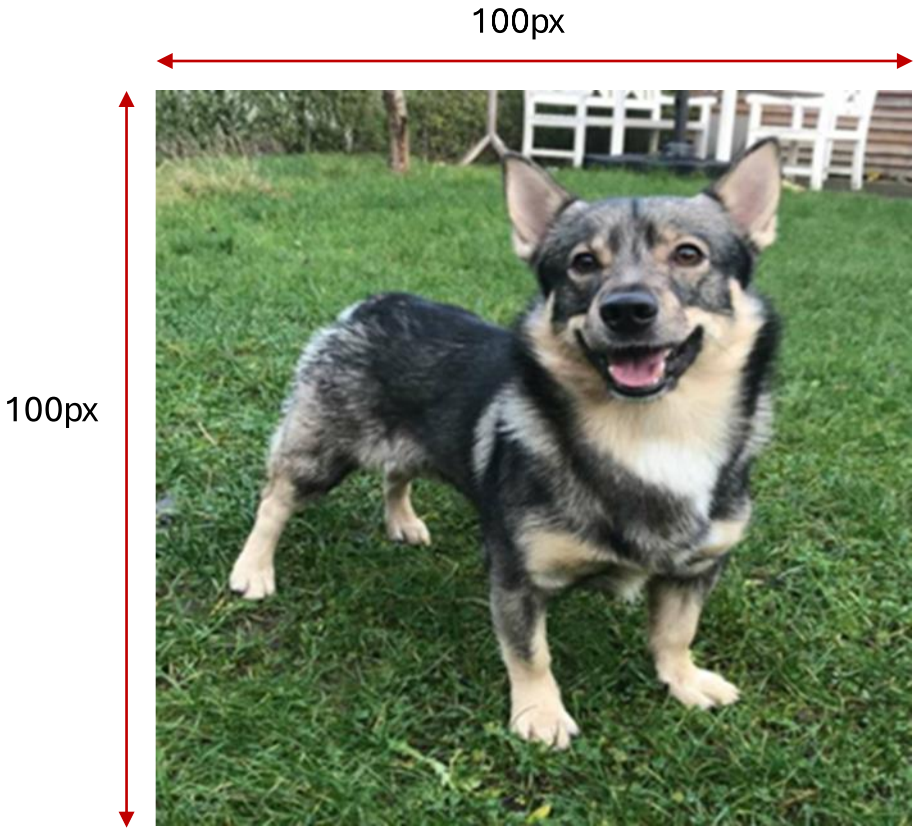

127.0.0.1:61055/index.html – Google Chrome
127.0.0.1:61055/index.html
Apps
Bookmarks
Gmail - atbinderup@
YouTube
General
VIA

# Box Sizing

- Remember the CSS box
- Has padding, border, margin
- All adds up when calculating size
- To make width and heigth include CSS box, use
  `box-sizing: border-box;`

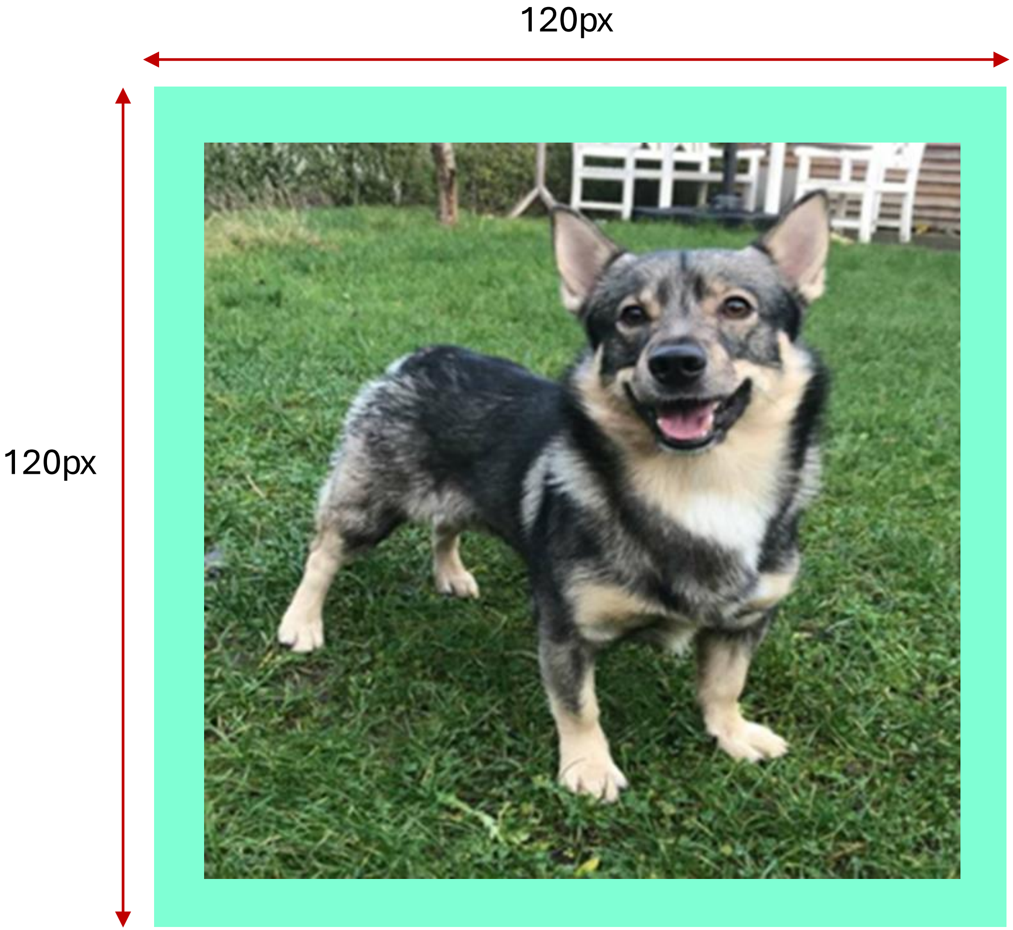
# Box Sizing - Example

```
div {
width: 100px;
height: 100px;
background-color:
aquamarine; //not
visible
```



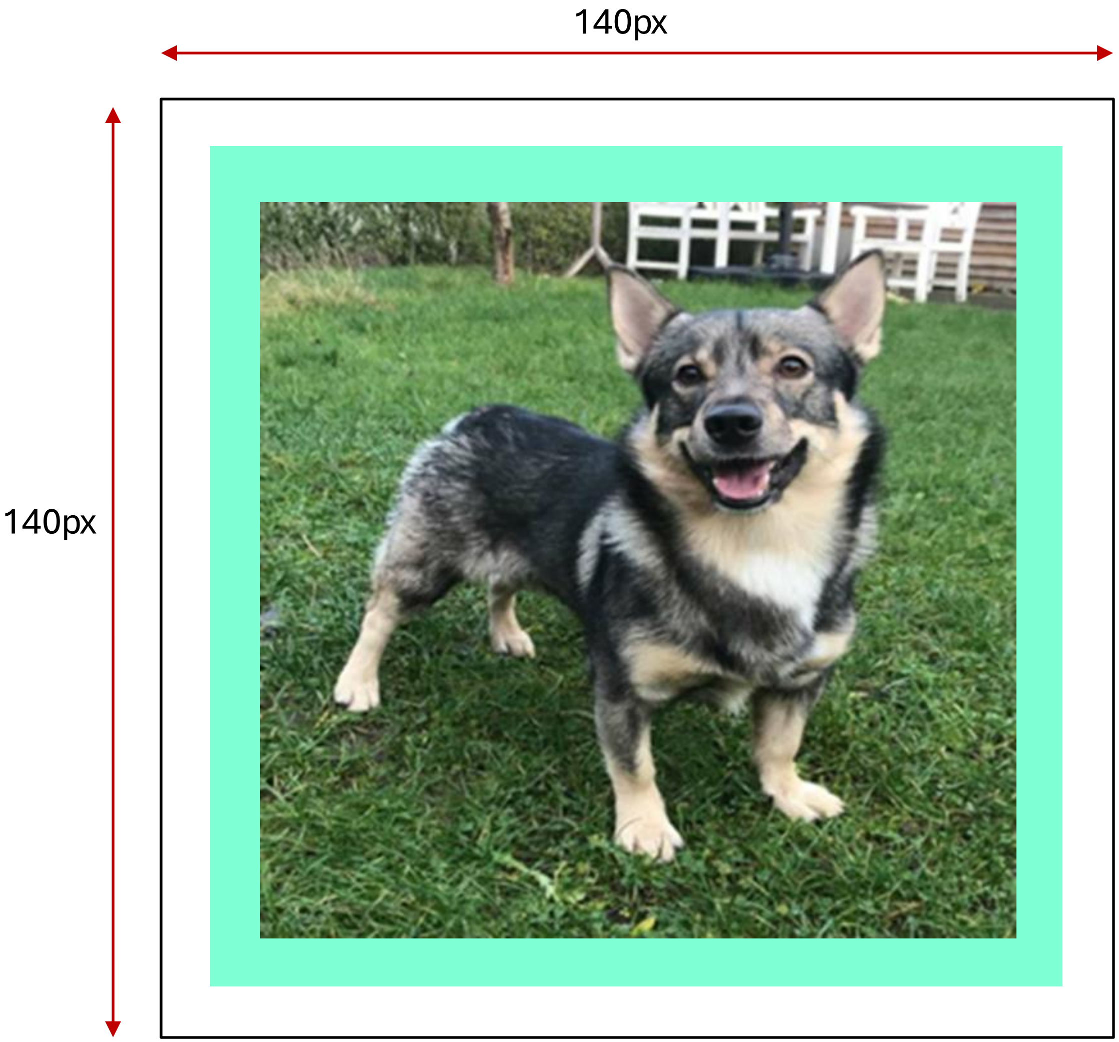
# Box Sizing - Example

```
div {
width: 100px;
height: 100px;
background-color:
aquamarine;
padding: 10px;
```



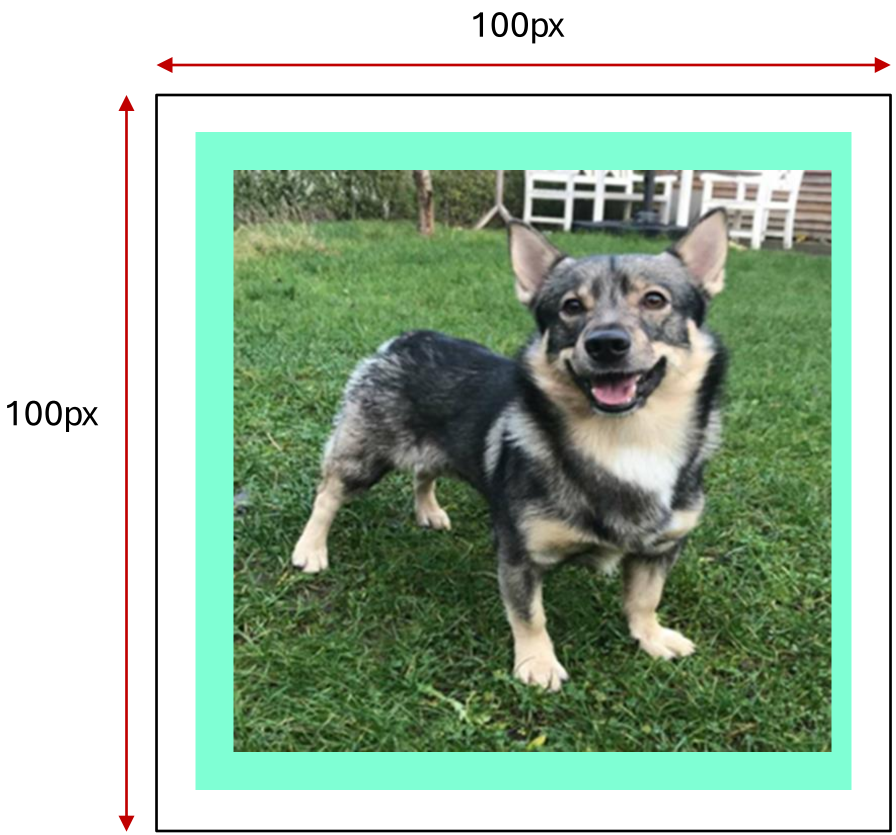
# Box Sizing - Example

```
div {
width: 100px;
height: 100px;
background-color:
aquamarine;
padding: 10px;
marging: 10px;
```

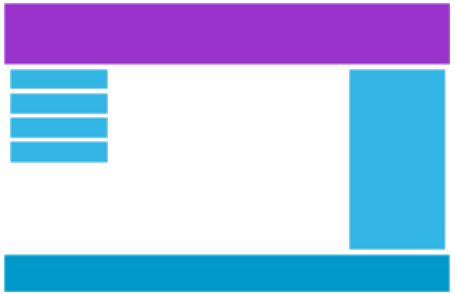


# Box Sizing - Example

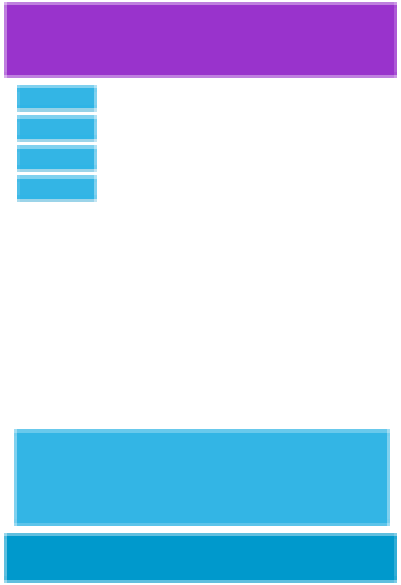
```
div {
width: 100px;
height: 100px;
background-color:
aquamarine;
padding: 10px;
marging: 10px;
box-sizing: border-box;
```



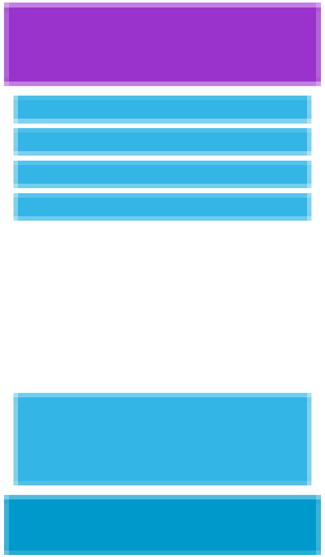
# Media Queries



Desktop

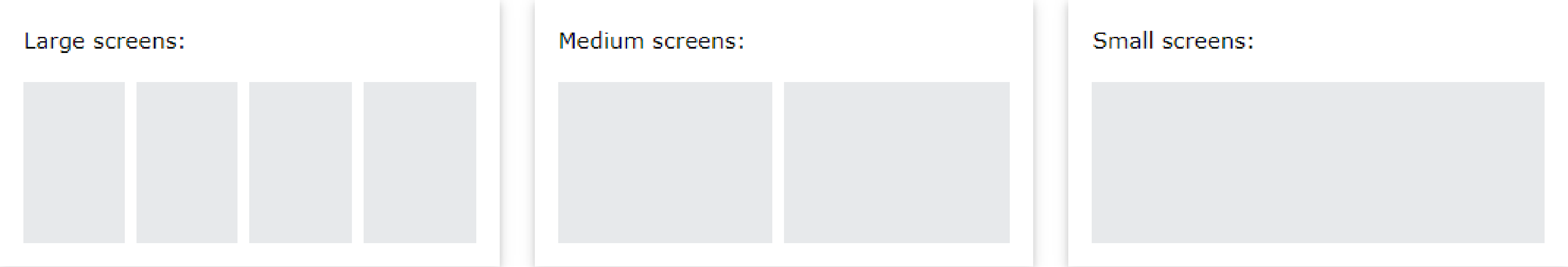


Tablet



Phone

# Media Queries



Large screens:

Medium screens:

Small screens:

# Media Queries

- Change which CSS styling is applied, based on client (screen resolution/window size)

- Select which stylesheet to apply:

```
<link media="mediatype and (media feature)“
      href="myStylesheet.css"
      rel="stylesheet"
      type="text/css">
```

- Or manipulate the CSS Styles directly:

```
@media mediatype and (media feature) {
  CSS-Code;
}
```

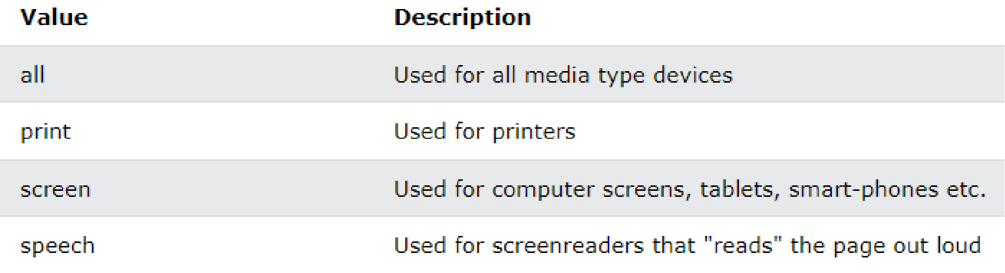
# Media Queries

- Media type and feature = “Breakpoints” to define different behaviour

- Relates to device (type) and size (feature)

- When `mediatype` is `screen` and `media feature` is `max-width/min-width`, you are comparing against browser window size (CSS pixels) and not the actual device size (to do that, use `max-device-width/min-device-width` as `media feature` instead).

- Generally, `mediatype` is `screen`



| Value | Description |
|---|---|
| all | Used for all media type devices |
| print | Used for printers |
| screen | Used for computer screens, tablets, smart-phones etc. |
| speech | Used for screenreaders that "reads" the page out loud |

# Media Queries

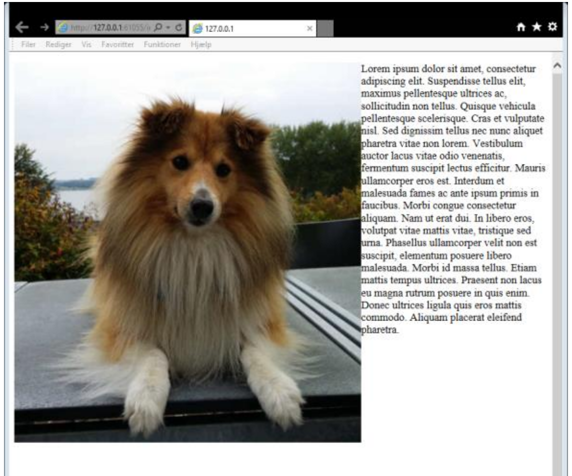
- Select which stylesheet to apply:

```
<head>
  <link rel="stylesheet" media="screen and (max-width: 480px)"
  href="css/smallStylesheet.css" type="text/css">

  <link rel="stylesheet" media="screen and (min-width: 481px)"
  href="css/largeStylesheet.css" type="text/css">
  …
</head>
```

# Media Queries

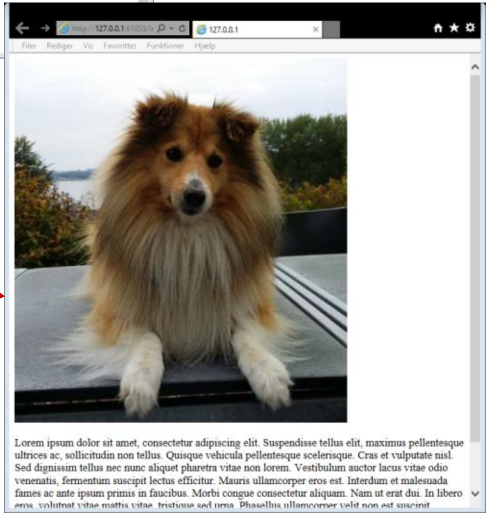
- In CSS

```css
img {
        width: 500px;
}

@media screen and (min-width: 800px) {
        img {
                float: left;
        }
}
```



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse tellus elit, maximus pellentesque ultrices ac, sollicitudin non tellus. Quisque vehicula pellentesque scelerisque. Cras et vulputate nisl. Sed dignissim tellus nec nunc aliquet pharetra vitae non lorem. Vestibulum auctor lacus vitae odio venenatis, fermentum suscipit lectus efficitur. Mauris ullamcorper eros est. Interdum et malesuada fames ac ante ipsum primis in faucibus. Morbi congue consectetur aliquam. Nam ut erat dui. In libero eros, volutpat vitae mattis vitae, tristique sed urna. Phasellus ullamcorper velit non est suscipit, elementum posuere libero malesuada. Morbi id massa tellus. Etiam mattis tempus ultrices. Praesent non lacus eu magna rutrum posuere in quis enim. Donec ultrices ligula quis eros mattis commodo. Aliquam placerat eleifend pharetra.

When viewport width larger than 800px



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse tellus elit, maximus pellentesque ultrices ac, sollicitudin non tellus. Quisque vehicula pellentesque scelerisque. Cras et vulputate nisl. Sed dignissim tellus nec nunc aliquet pharetra vitae non lorem. Vestibulum auctor lacus vitae odio venenatis, fermentum suscipit lectus efficitur. Mauris ullamcorper eros est. Interdum et malesuada fames ac ante ipsum primis in faucibus. Morbi congue consectetur aliquam. Nam ut erat dui. In libero eros, volutpat vitae mattis vitae, tristique sed urna. Phasellus ullamcorper velit non est suscipit

Default view

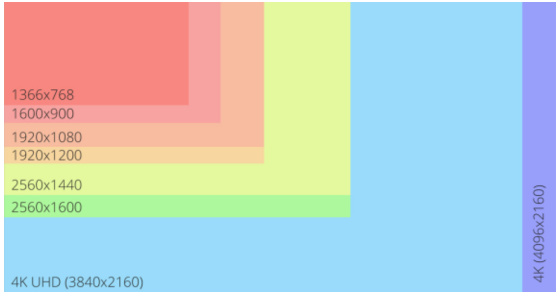
# Media Queries

```
@media screen and (max-width: 400px) {
  …
}

@media screen and (min-width: 401px) and (max-width: 1024px) {
  …
}

@media screen and (min-width: 1025px) {
  …
}
```

# Media Queries

- How to determine ”breakpoints”?
- Common screen resolutions?
  - Hard to maintain (new resolutions to accomodate)
  - Differences in devices (smartphones in new form factors)

✗



- Manual testing
  - Time consuming
  - Leads to best results

✓

# Flexbox

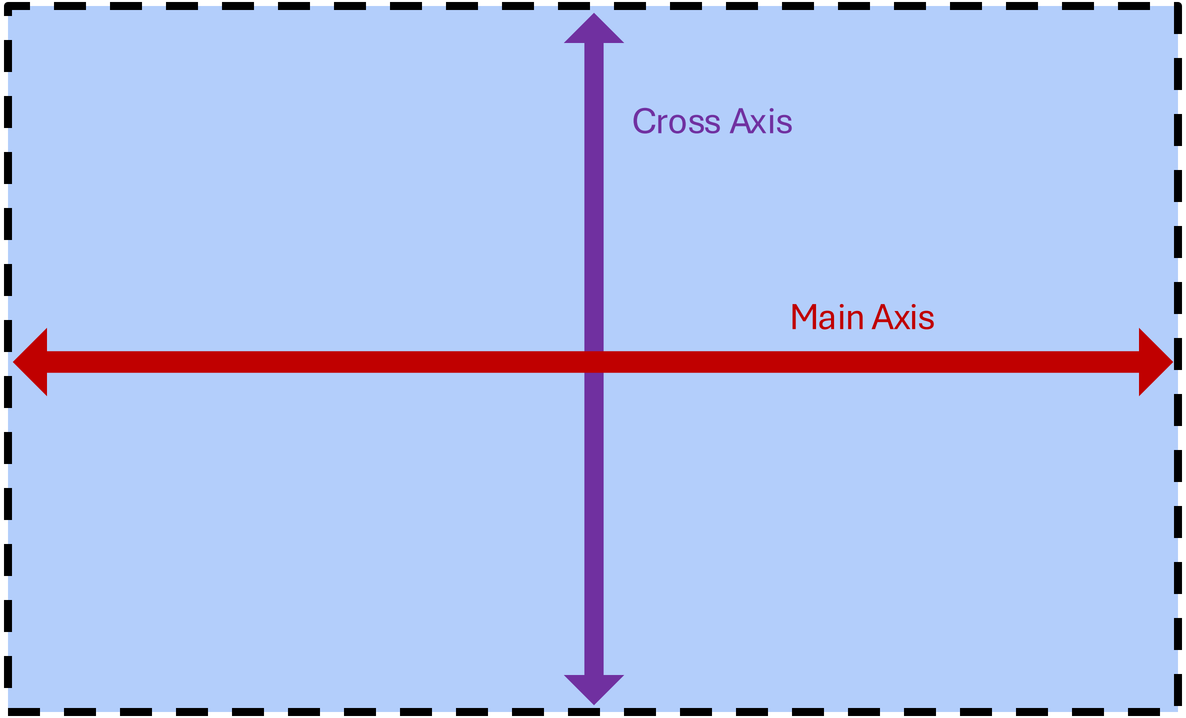
- A layout model
- Like `float`, but more powerful
- When applied to a container, will control layout of nested elements

# Flexbox

- A layout model
- Like `float`, but more powerful
- When applied to a container, will control layout of nested elements
- Two axes
  - Main axis: `justify-content`
  - Cross axis: `align-items`
- Can be used in many ways – try it out

# Flexbox

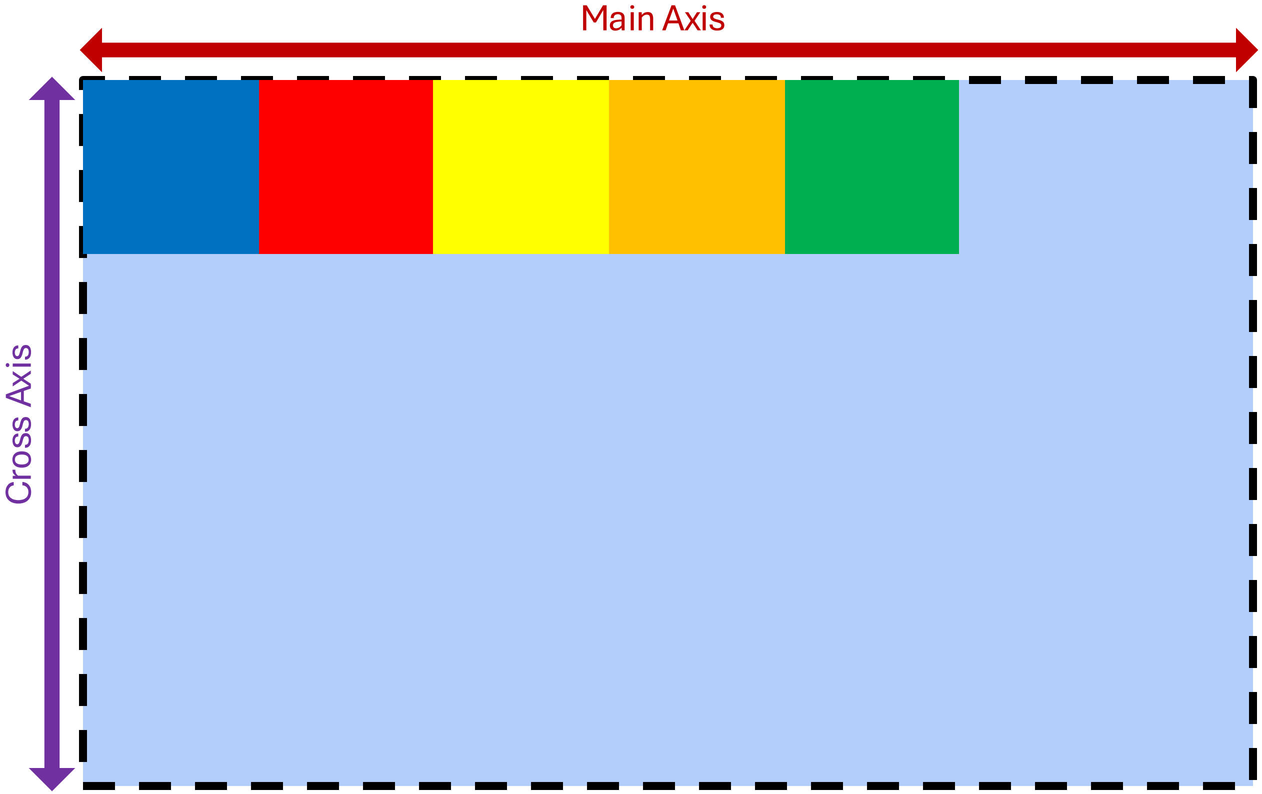
```
<body>
    <div class="box">
        <div class="container">
            <div></div>
            <div></div>
            <div></div>
            <div></div>
            <div></div>
        </div>
    </div>
</body>
```



# Flexbox

```
<body>
    <div class="box">
        <div
class="container">
            <div></div>
            <div></div>
            <div></div>
            <div></div>
            <div></div>
        </div>
    </div>
</body>
```

```
.container>div {
    width: 50px;
    height: 50px;
}

.container>div:nth-child(1) {
    background-color: blue;
}

.container>div:nth-child(2) {
    background-color: red;
}

.container>div:nth-child(3) {
    background-color: yellow;
}

.container>div:nth-child(4) {
    background-color: orange;
}

.container>div:nth-child(5) {
    background-color: green;
}
```

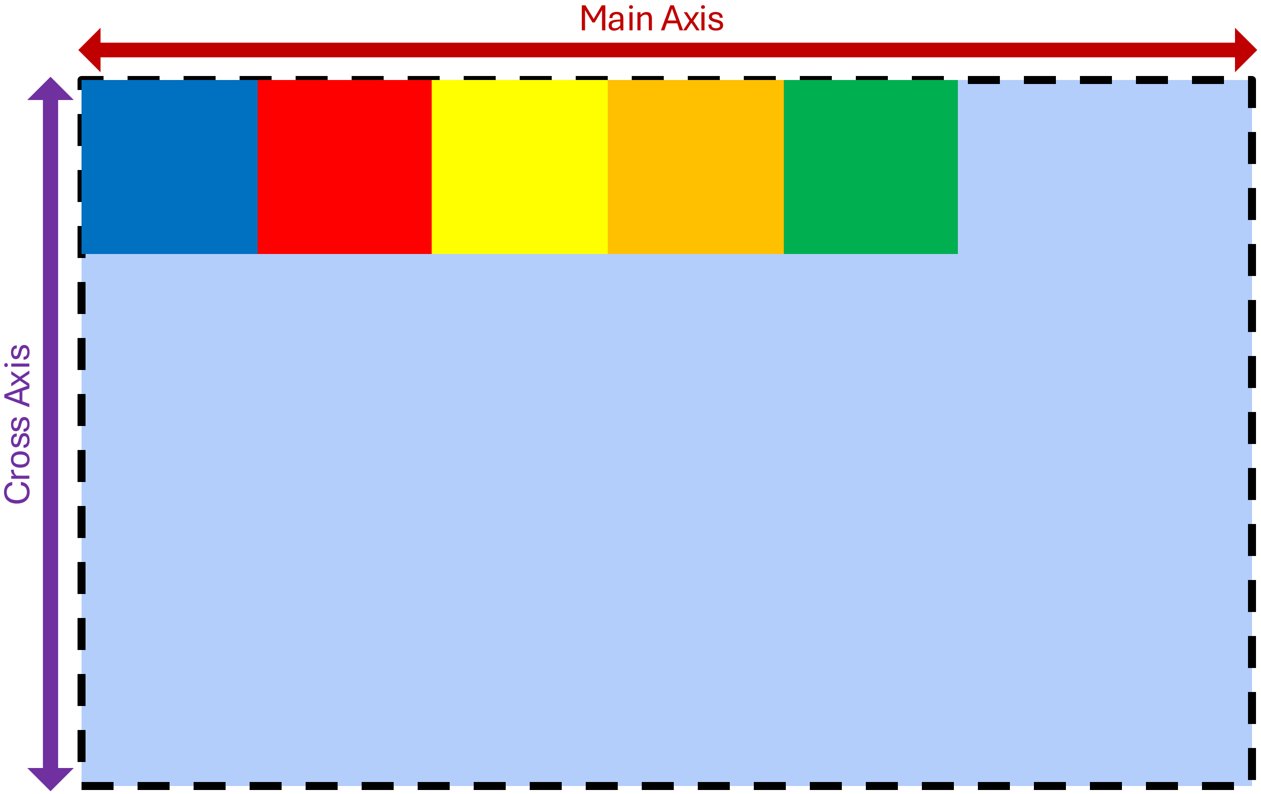


# Flexbox

```
<body>
    <div class="box">
        <div
class="container">
            <div></div>
            <div></div>
            <div></div>
            <div></div>
            <div></div>
        </div>
    </div>
</body>
```

```
.container {
    width: 400px;
    height: 200px;
    border: 5px dashed black;
}
```

```
.container {
    display: flex;
    justify-content: start;
```

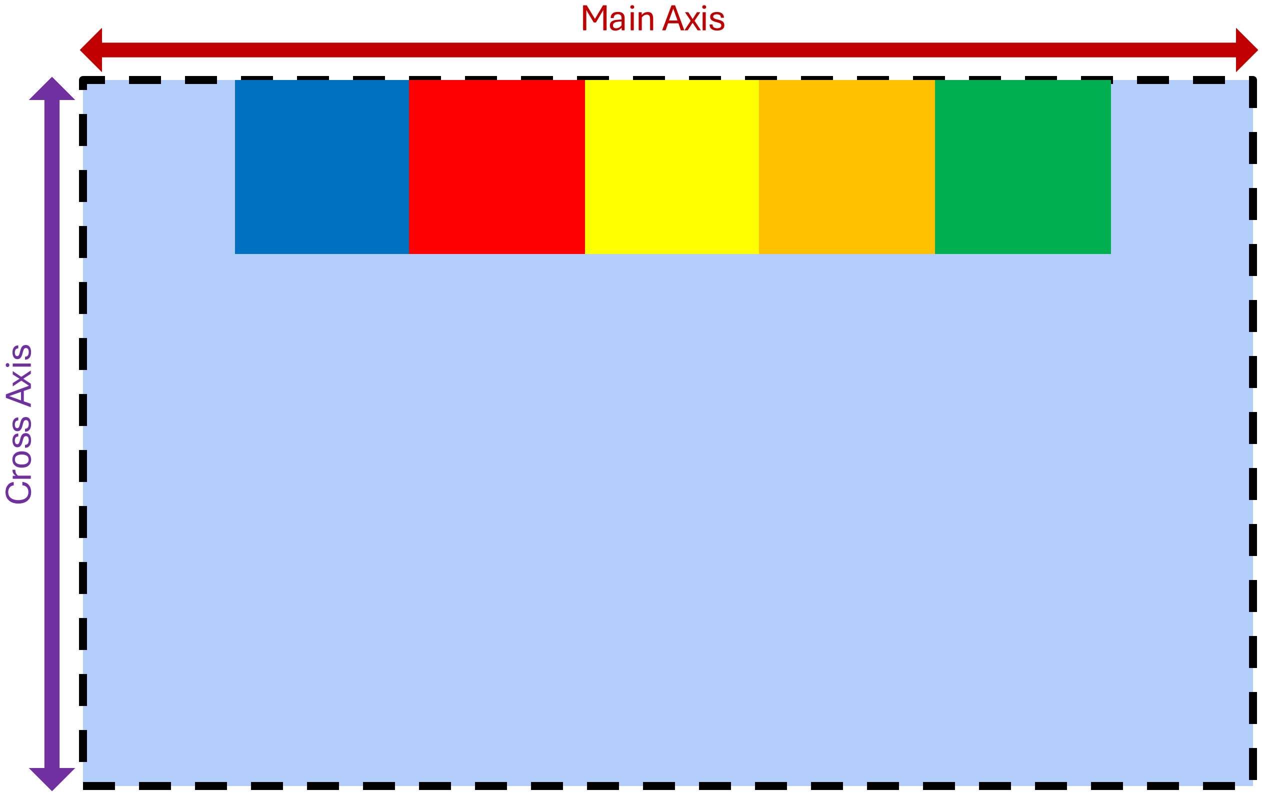


# Flexbox

```
<body>
    <div class="box">
        <div
class="container">
            <div></div>
            <div></div>
            <div></div>
            <div></div>
            <div></div>
        </div>
    </div>
</body>
```

```
.container {
    width: 400px;
    height: 200px;
    border: 5px dashed black;
}
```

```
.container {
    display: flex;
    justify-content: center;
```

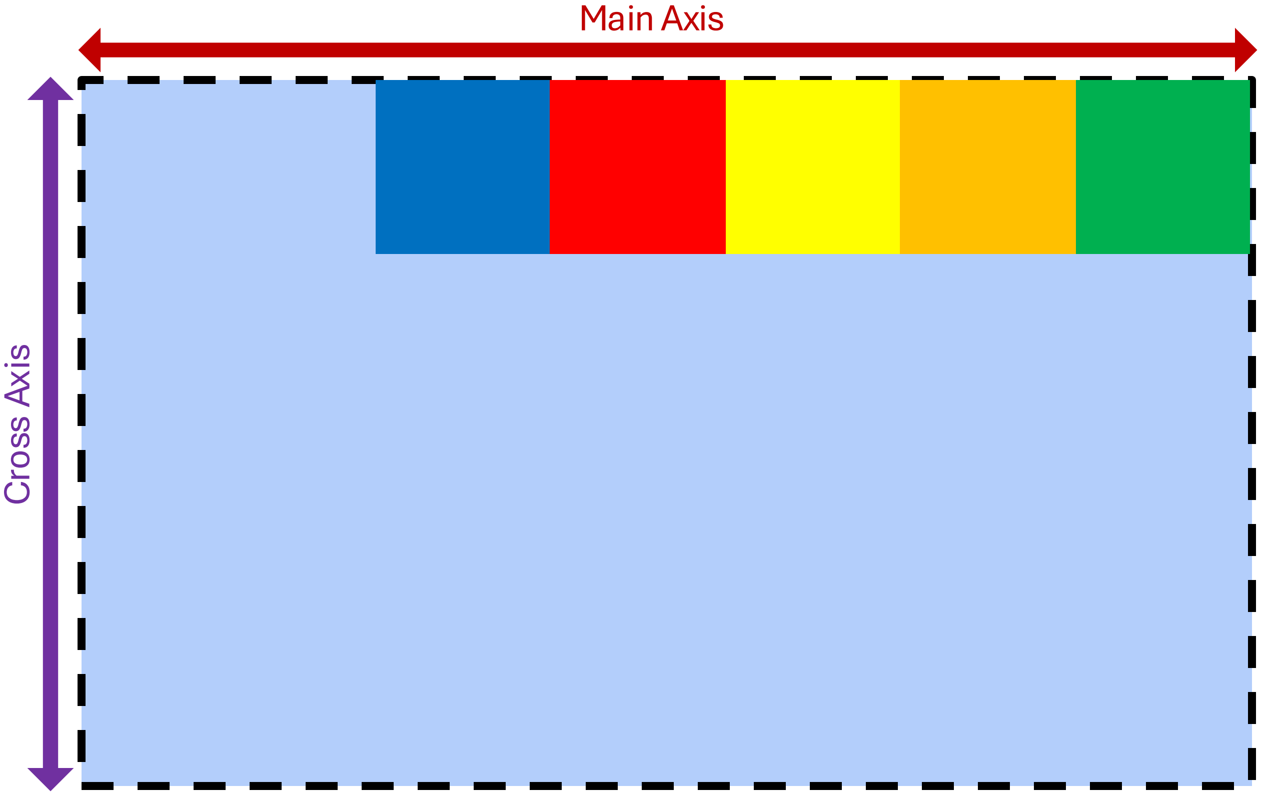


# Flexbox

```
<body>
    <div class="box">
        <div
class="container">
            <div></div>
            <div></div>
            <div></div>
            <div></div>
            <div></div>
        </div>
    </div>
</body>
```

```
.container {
    width: 400px;
    height: 200px;
    border: 5px dashed black;
}
```

```
.container {
    display: flex;
    justify-content: end;
```

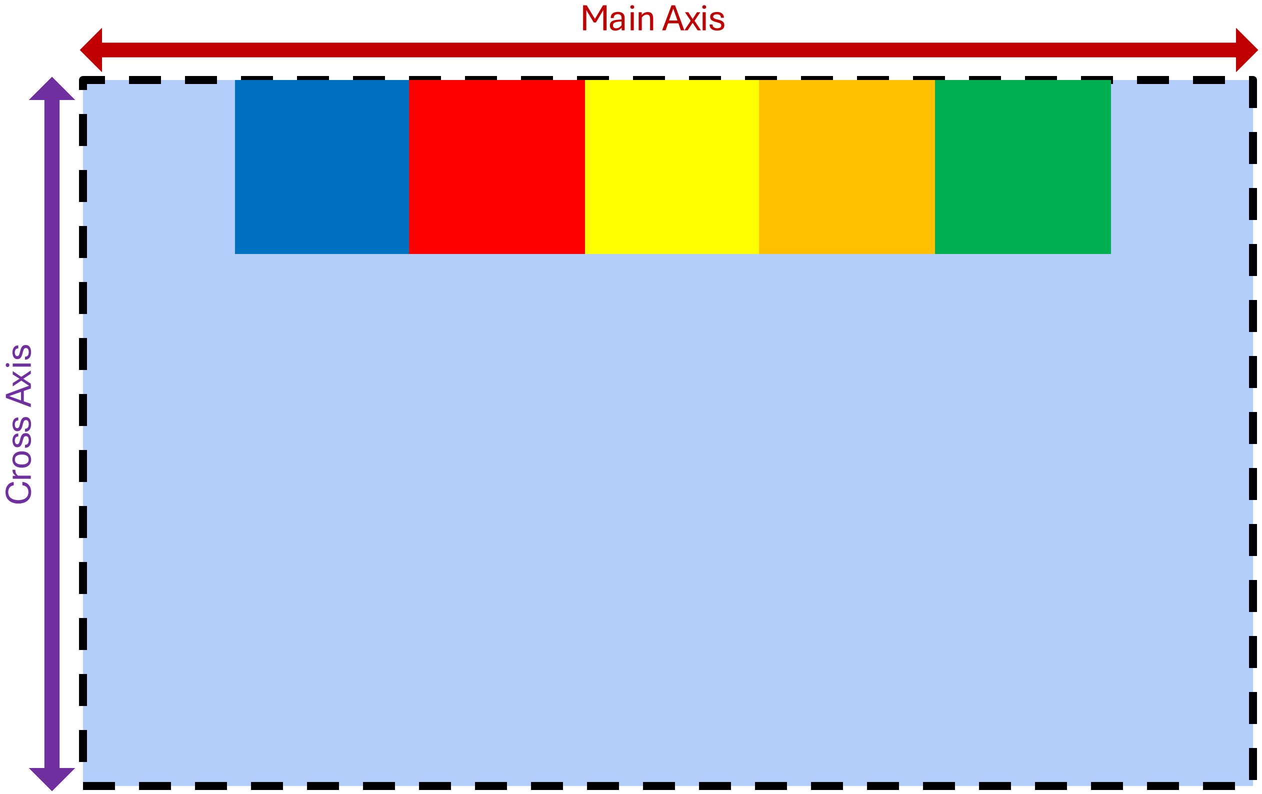


# Flexbox

```
<body>
    <div class="box">
        <div 
class="container">
            <div></div>
            <div></div>
            <div></div>
            <div></div>
            <div></div>
        </div>
    </div>
</body>
```

```
.container {
    width: 400px;
    height: 200px;
    border: 5px dashed black;
}
```

```
.container {
    display: flex;
    justify-content: center;
    align-items: flex-start;
}
```

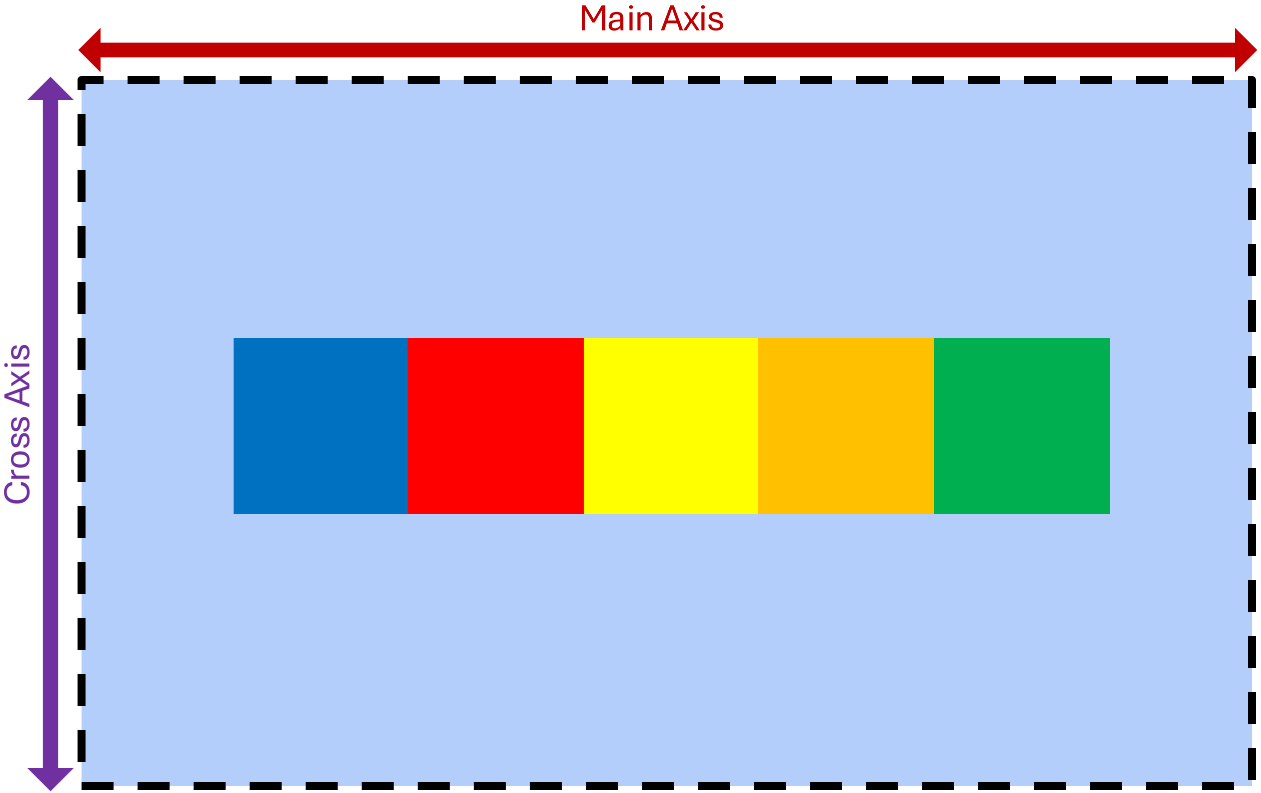


# Flexbox

```
<body>
    <div class="box">
        <div
class="container">
            <div></div>
            <div></div>
            <div></div>
            <div></div>
            <div></div>
        </div>
    </div>
</body>
```

```
.container {
    width: 400px;
    height: 200px;
    border: 5px dashed black;
}
```

```
.container {
    display: flex;
    justify-content: center;
    align-items: center;
```

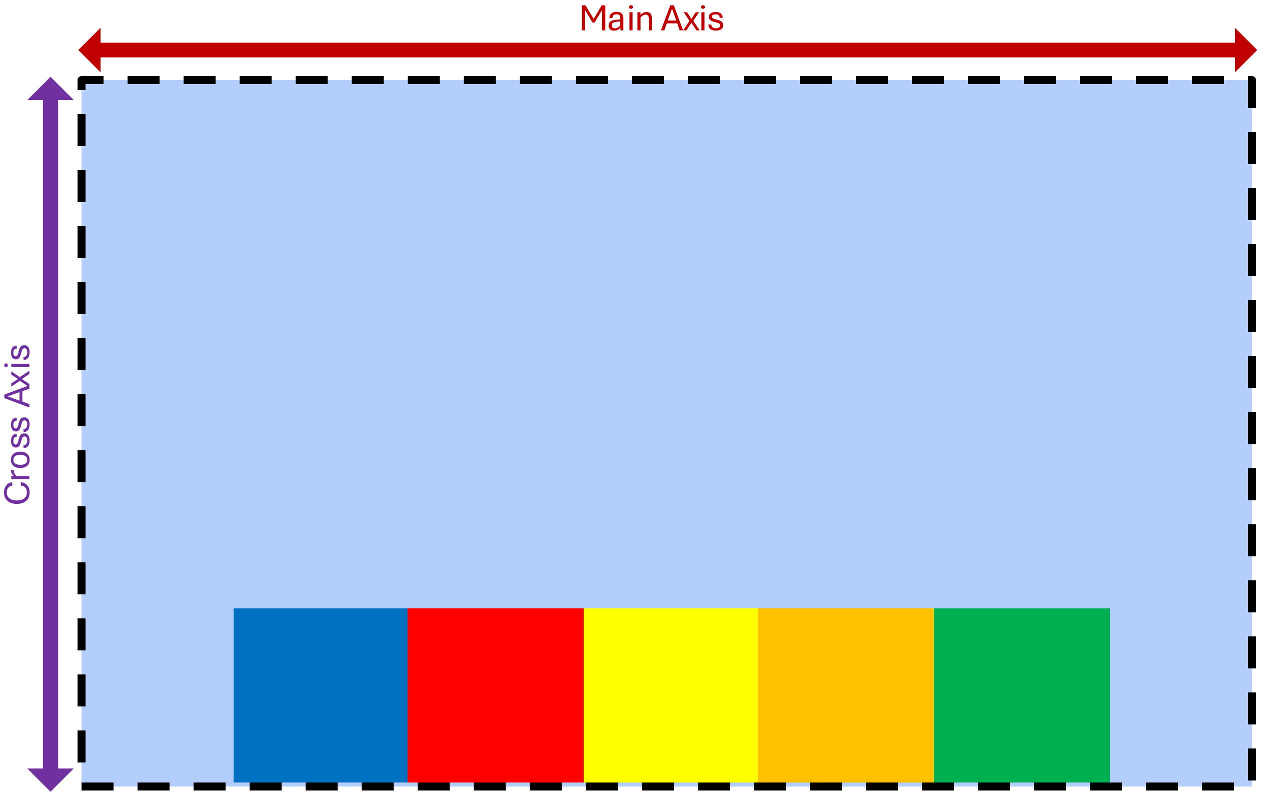


# Flexbox

```
<body>
    <div class="box">
        <div
class="container">
            <div></div>
            <div></div>
            <div></div>
            <div></div>
            <div></div>
        </div>
    </div>
</body>
```

```
.container {
    width: 400px;
    height: 200px;
    border: 5px dashed black;
}
```

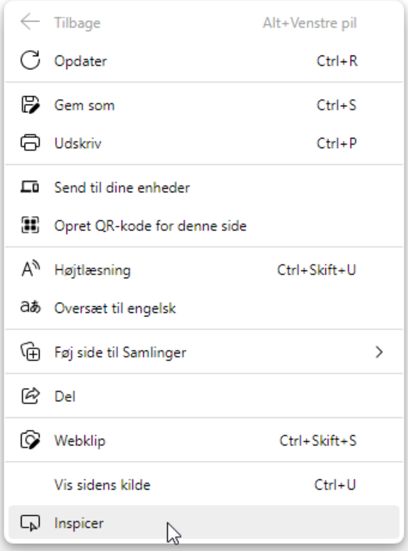
```
.container {
    display: flex;
    justify-content: center;
    align-items: flex-end;
```



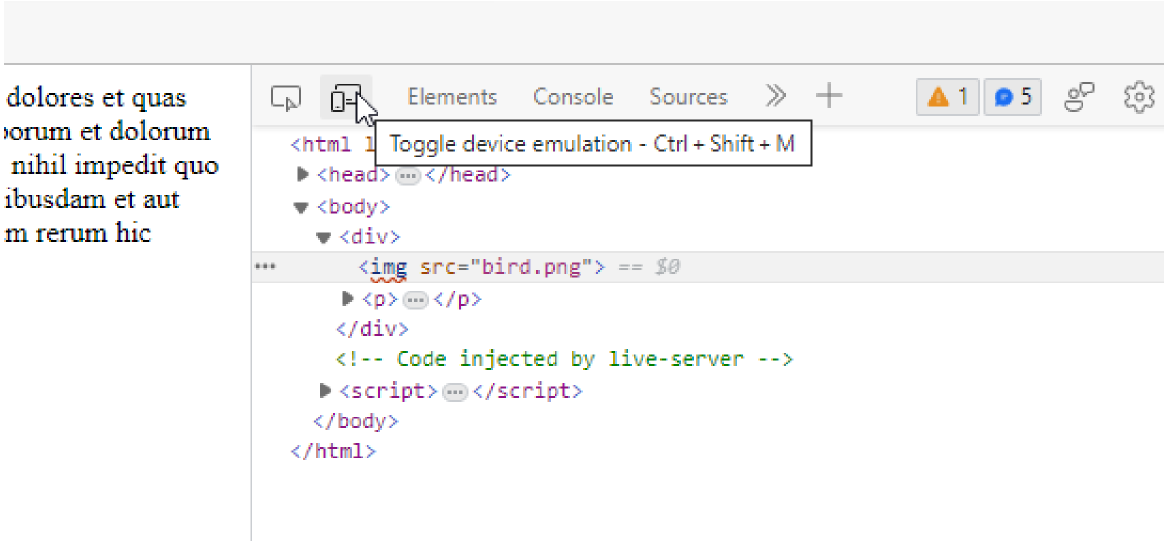
# Developer tools

- Test out responsiveness without actual devices
- Simulate screen size of different devices
- Can also resize window manually to test

# Developer tools



Right click anywhere



```
<html l
 ▶ <head> ⋯ </head>
 ▼ <body>
   ▼ <div>
•••     <img src="bird.png"> == $0
     ▶ <p> ⋯ </p>
     </div>
     <!-- Code injected by live-server -->
   ▶ <script> ⋯ </script>
   </body>
</html>
```

You can see the HTML, highlighted line is what you rightclicked.
The ”Toggle device emulation” is great for finding media breakpoints for tablets/mobiles.