

**Front End**

# Module Overview (**HTML**)

- HTML in depth
- Main Elements - Heading, Paragraph, List, Anchor, Image.
- HTML5
- Block Vs. Inline elements - more elements (div, span, etc.)
- Additional Elements.
- HTML Entities
- Semantic Markup
- Tables
- Forms

# Module Overview (CSS)

- CSS in depth
- Rules (Syntax)
- Where to include CSS?
- Colors
- Color & Background-color
- Text properties
- CSS Units
- Selectors
- Inheritance
- The box model
- The display property
- Opacity
- Spacing and Positioning
- Fonts
- Responsiveness
- Flexbox
- Media Queries

# HTML in depth

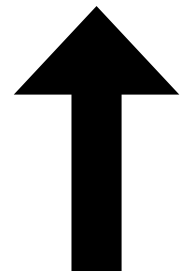
# HyperText Markup Language

# How to write HTML?

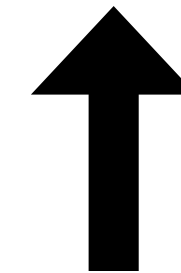
- We pick from a set of standard Elements that all browsers recognize, such as:
  - `<p>` element: marks text as a paragraph
  - `<h1>` element: marks text as a header
  - `<img>` element: embeds image
  - `<form>` element: represents forms

# Remember!

`<p>We create elements by writing tags.</p>`



Opening Tag



Closing Tag

**Mozilla DN**

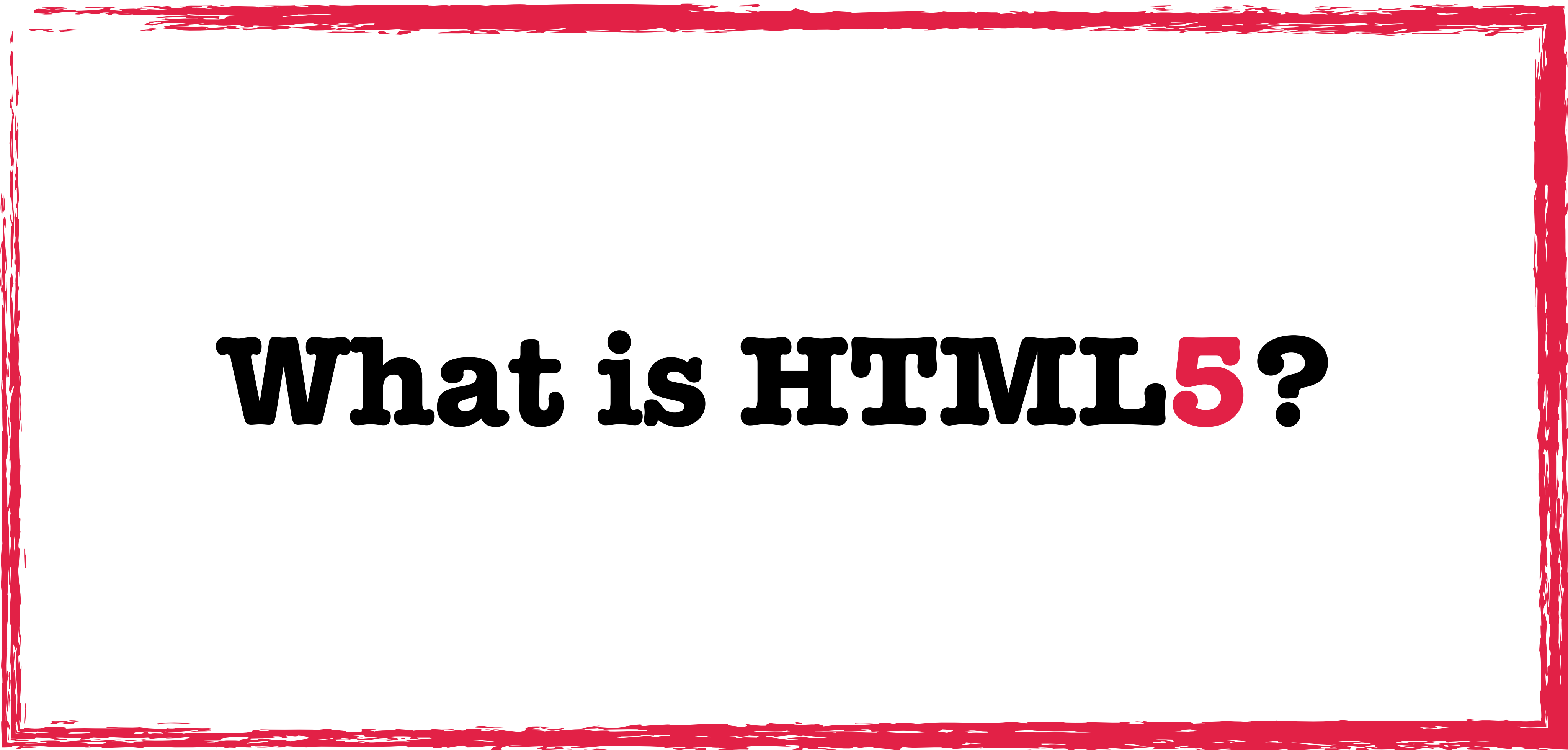


**Let's Start  
Coding!**

# HTML Skeleton

```
<!DOCTYPE html>
<html>
  <head>
    <title>I love PHP!</title>
  </head>
  <body>
    <h1>Hello Mr. Mike</h1>
  </body>
</html>
```





**What is HTML5?**

# **Block VS Inline Elements**

**<div> VS <span>**

A thick, red, hand-drawn rectangular border with a rough, textured appearance, framing the central text.

# **Additional Elements**

# Entity Codes

# **Semantic Markup**

(relating to meaning)



**Semantic HTML = Writing meaningful HTML**

**Semantic HTML = What purpose or role does that HTML part have?**

**Semantic = SEO Booster**

# **Live Homework**

# Tables

<b><u>&lt;table&gt;</u></b>	Defines a table
<b><u>&lt;th&gt;</u></b>	Defines a header cell in a table
<b><u>&lt;tr&gt;</u></b>	Defines a row in a table
<b><u>&lt;td&gt;</u></b>	Defines a cell in a table
<b><u>&lt;caption&gt;</u></b>	Defines a table caption
<b><u>&lt;colgroup&gt;</u></b>	Specifies a group of one or more columns in a table for formatting
<b><u>&lt;col&gt;</u></b>	Specifies column properties for each column within a <colgroup> element
<b><u>&lt;thead&gt;</u></b>	Groups the header content in a table
<b><u>&lt;tbody&gt;</u></b>	Groups the body content in a table
<b><u>&lt;tfoot&gt;</u></b>	Groups the footer content in a table

# Forms

**<form>**

**<input>**

**<label>**



**<button>**

# **The “Name” attribute**

**checkbox**

**radio**

**<select>**

**range**

**number**

**<textarea>**



# **Form Validation**

# **Live Homework**



CSS

# Get deep into CSS

- CSS stands for Cascading Style Sheets.
- It is a stylesheet language used to describe the presentation of a document written in HTML.
- CSS describes how elements should be **rendered** on screen, on paper, or on other media.
- CSS is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents.
- CSS can be intimidating due to the number of properties we can manipulate.

# CSS Syntax

```
selector {  
    property: value;  
}
```

# Make all <p> elements red

```
selector {  
    property: value;  
}
```

```
P {  
    Color: red;  
}
```

# How to include CSS

- Inline
- Internal
- External



Colors



**color &  
background-color**

**text-align**  
**font-weight**  
**text-decoration**  
**line-height**  
**letter-spacing**

# **Text Properties**

**font-size**

**font-family**

# Selectors

# Universal Selector

```
* {  
  color: rgb(100,100,100);  
}
```

# Element Selector

```
p {  
  color: rgb(100,100,100);  
}
```

# Selector List

```
h1, p, a, span {  
    color: rgb(100,100,100);  
}
```



# Class Selector

```
.alerts {  
    color: rgb(100,100,100);  
}
```

# Descendant Selector

```
li a {  
    color: rgb(100,100,100);  
}
```

# Adjacent Selector

```
img + p {  
    color: rgb(100,100,100);  
}
```

# Direct Child

```
p > span {  
  color: rgb(100,100,100);  
}
```

# Attribute Selector

```
input[name="username"] {  
    color: rgb(100,100,100);  
}
```

# Pseudo Classes

- :active
- :checked
- :focus
- :first-child
- :last-child
- :hover
- :not()

# Pseudo Elements

- `::after` (`:after`)
- `::before` (`:before`)
- `::selection`
- `::first-letter`



style.css

```
p {  
  font-size: 20px;  
}
```

```
p {  
  font-size: 18px;  
}
```

The  
Cascade

**Conflict!**





# **Specificity**

ID > Class > element

ID > Class > element

Inline Style

ID > Class > element

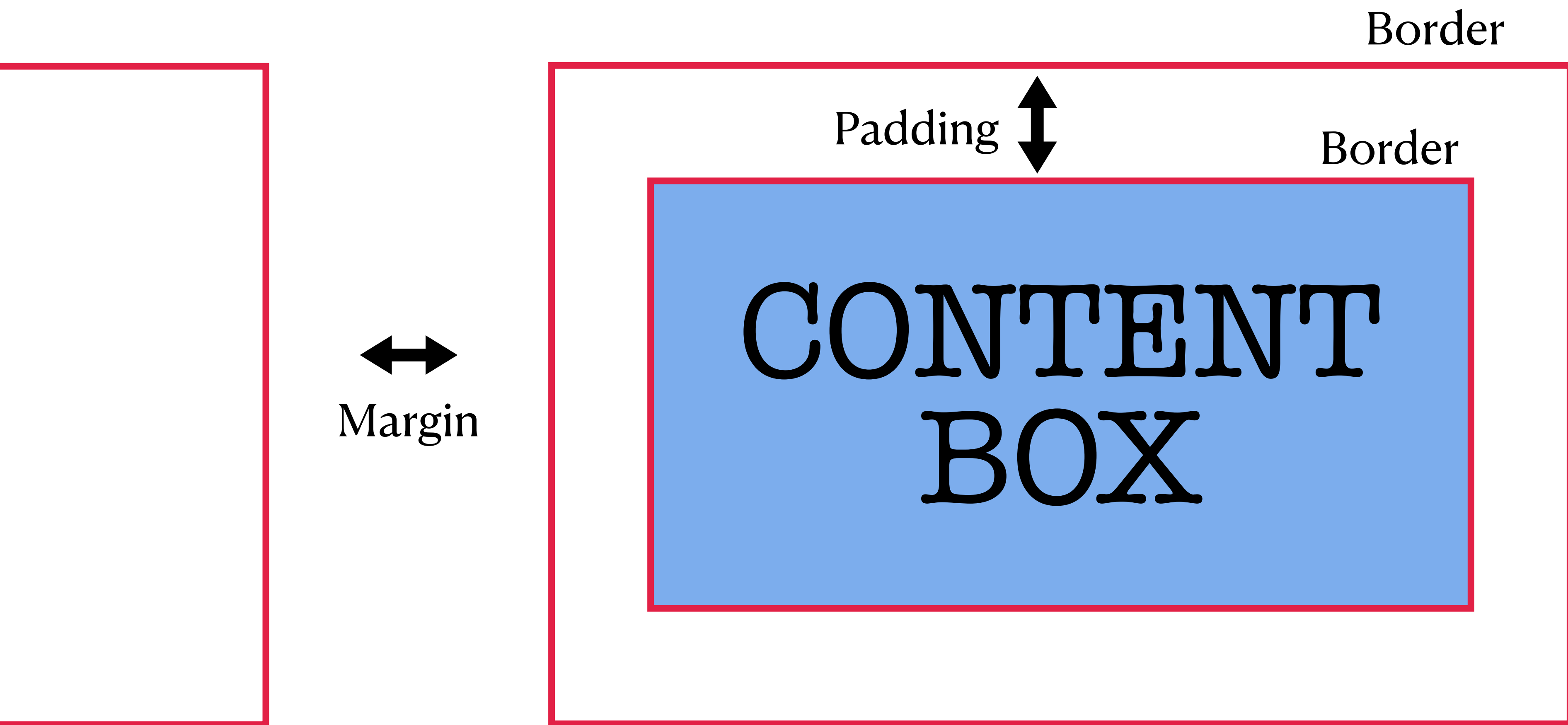
Inline Style  
!important

# **!important**

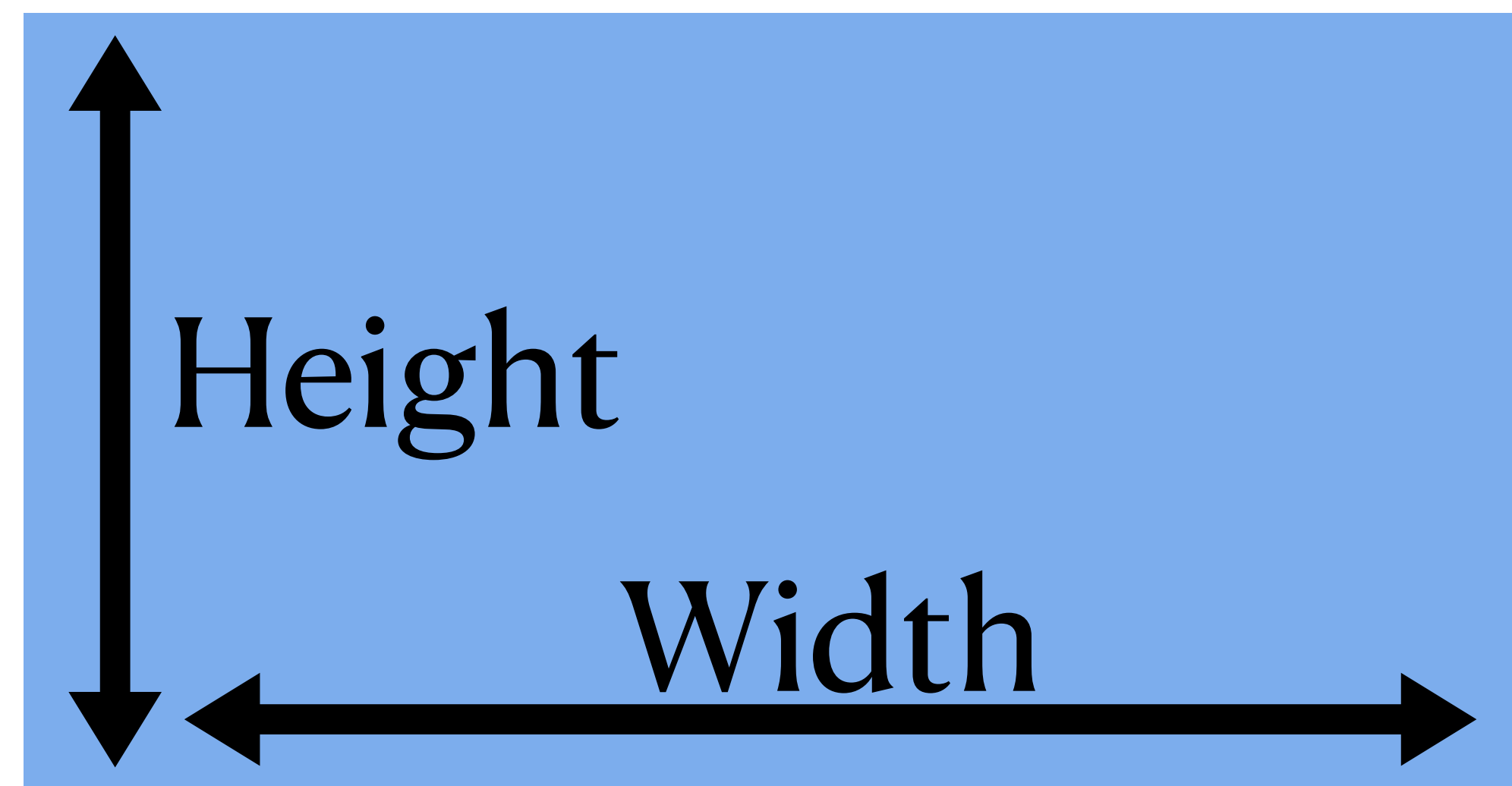
- When you write CSS, forget that it exists.
- If you can't override an existed CSS, then you can use it.

**inheritance**

# **The Box Model**







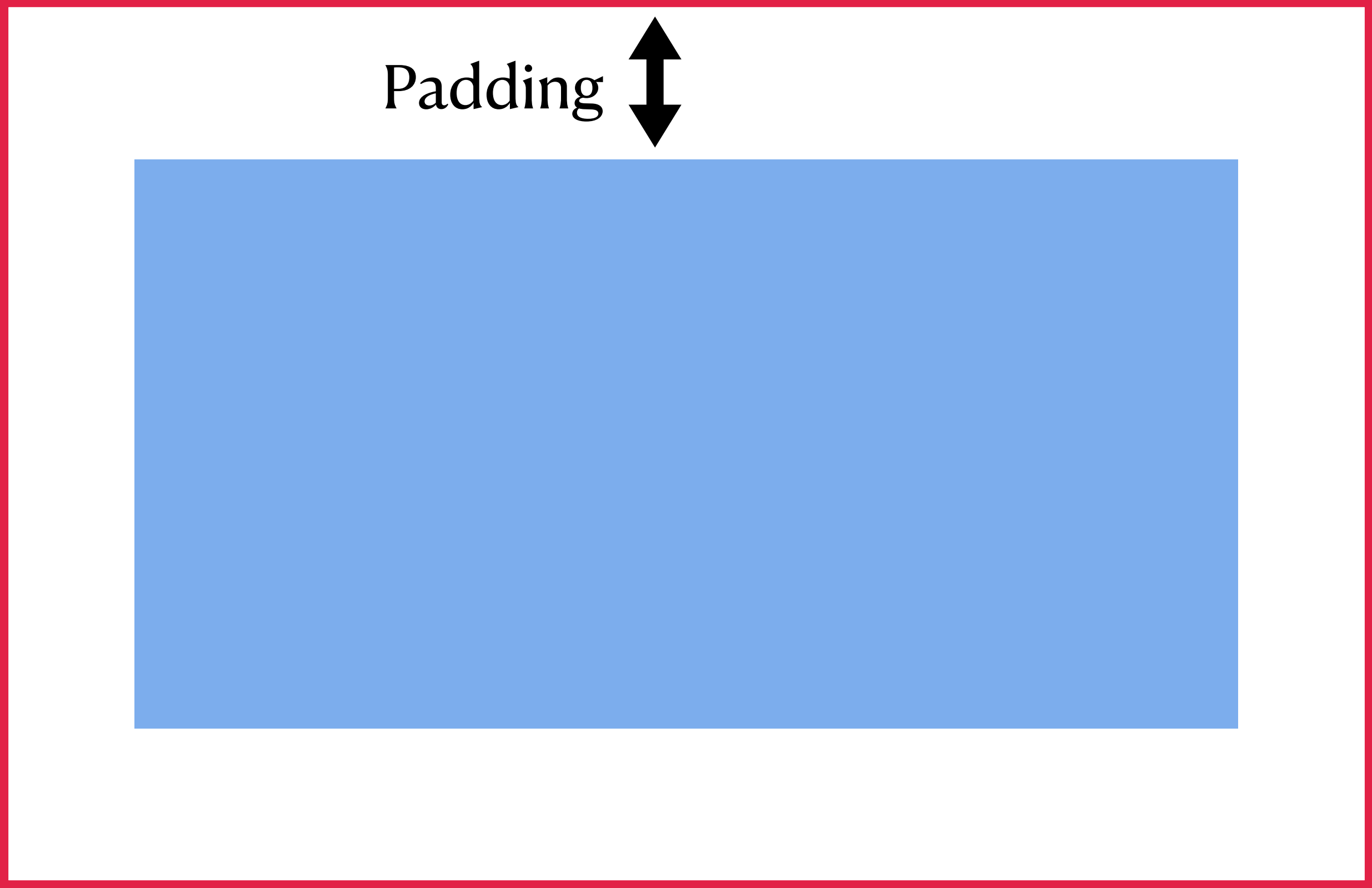
# Border



border-width

border-color

border-style

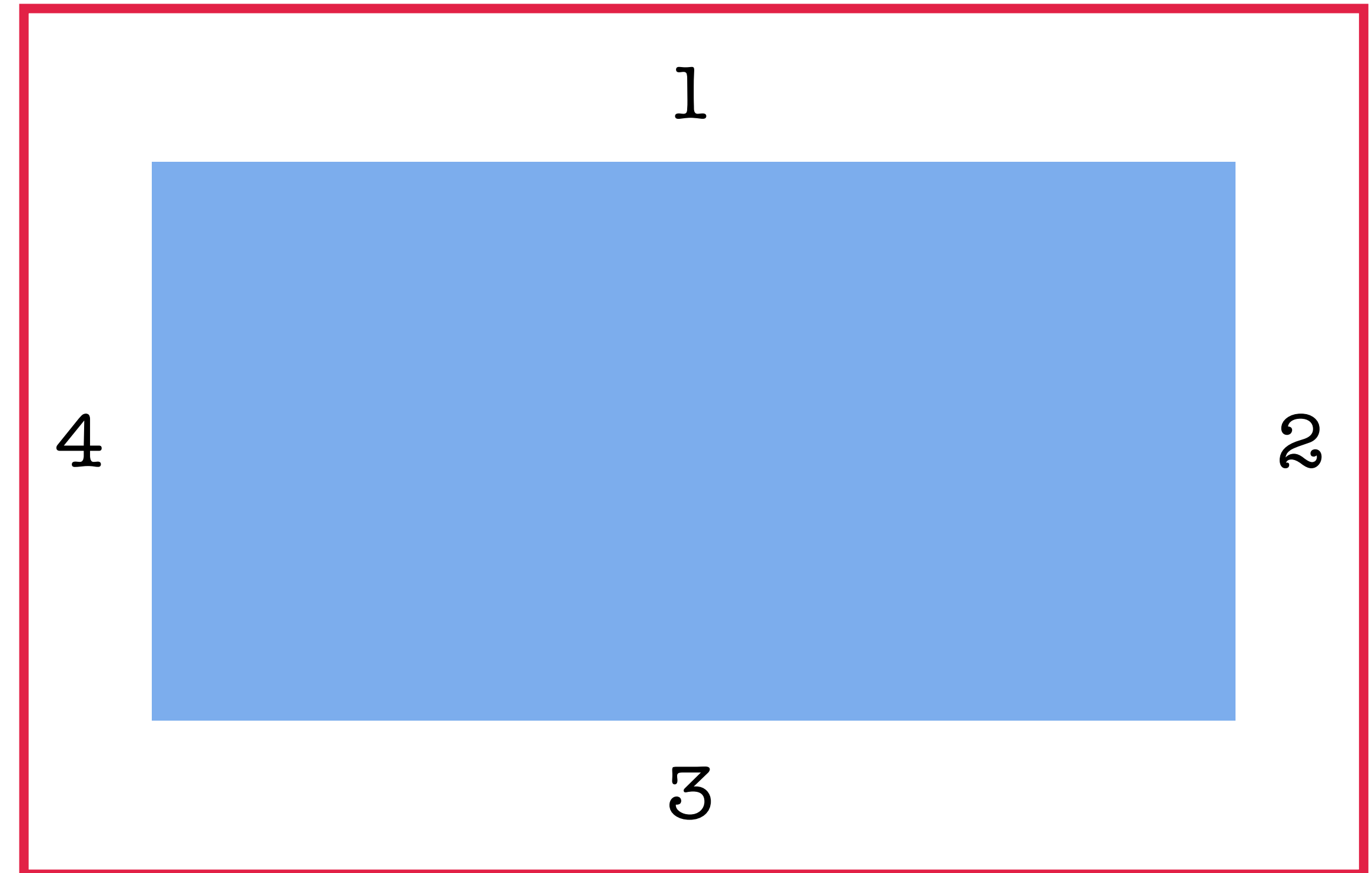


# Individual Properties

```
.box {  
    padding-top: 10px;  
    padding-right: 10px;  
    padding-bottom: 10px;  
    padding-left: 10px;  
}
```

# Shorthand Property

```
.box {  
  padding: 10px;  
  padding: 5px 10px;  
  padding: 5px 10px 3px;  
  padding: 5px 10px 3px 15px;  
}
```





↔  
Margin

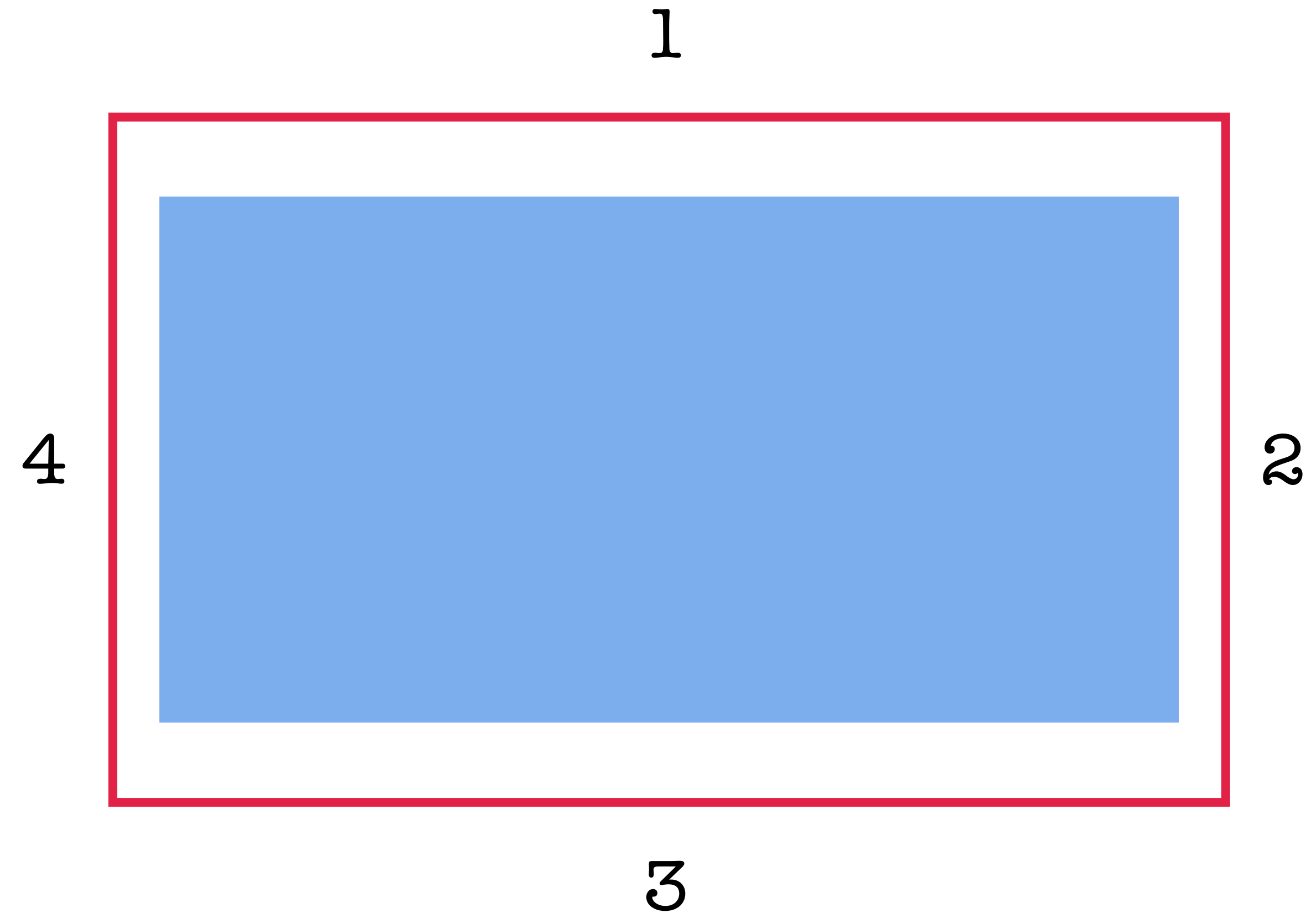


# Individual Properties

```
.box {  
    margin-top: 10px;  
    margin-right: 10px;  
    margin-bottom: 10px;  
    margin-left: 10px;  
}
```

# Shorthand Property

```
.box {  
  margin: 10px;  
  margin: 5px 10px;  
  margin: 5px 10px 3px;  
  margin: 5px 10px 3px 15px;  
}
```







# **Display Property**

- Inline
- Block

# CSS Units

# **Relative VS Absolute**

- EM
- REM
- VH
- VW
- %
- AND MORE!

**VS**

- PX
- PT
- CM
- IN
- MM

# Opacity & Alpha Channel

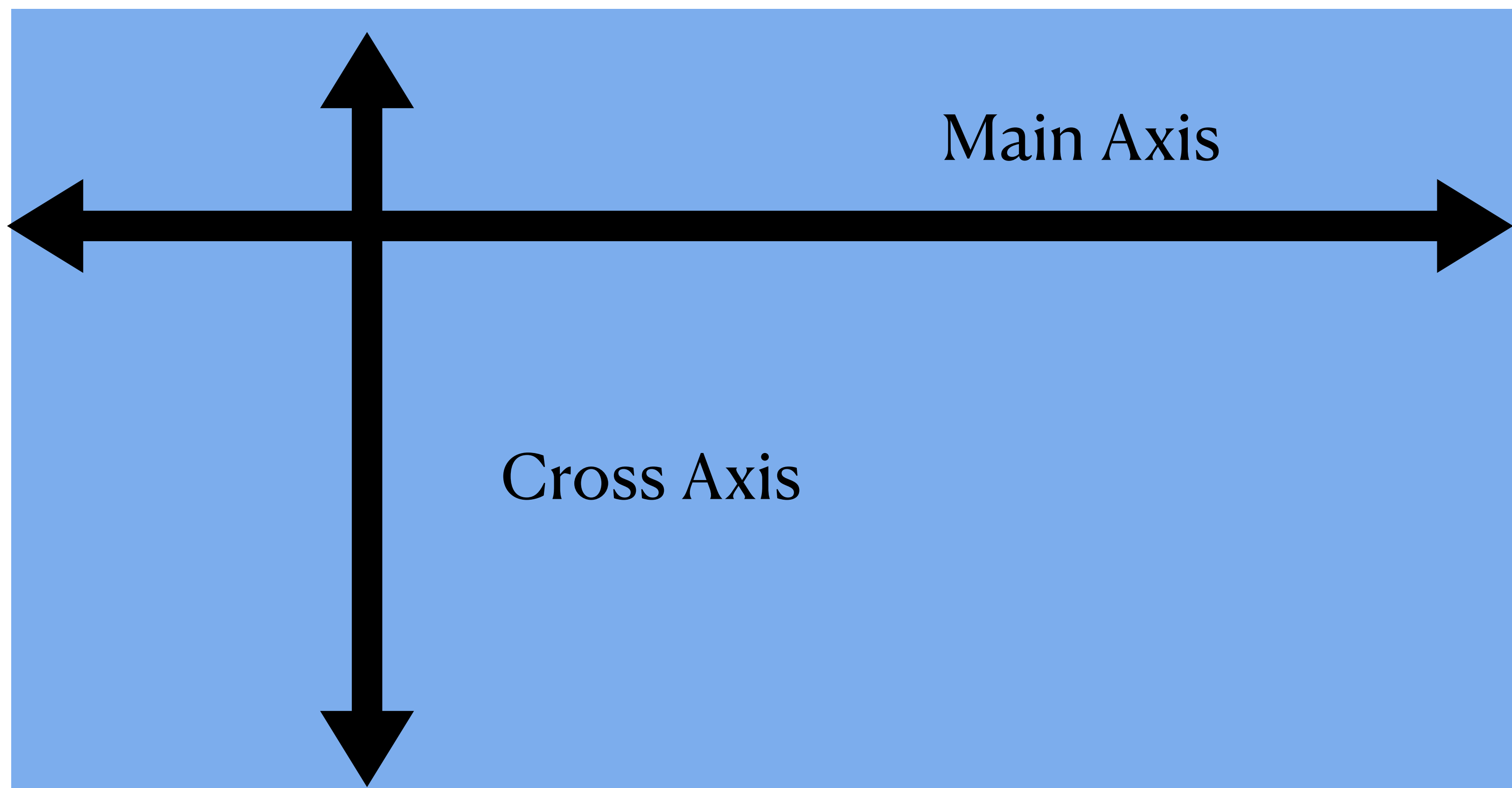
# **Position**

**Background**



**Fonts**

# CSS Flexbox



flex-direction

flex-wrap

justify-content

align-items

flex-basis  
flex-grow  
flex-shrink



# **Responsiveness & Media Queries**