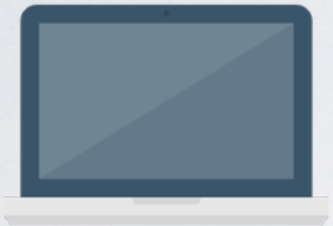
HTML & CSS

THE FOUNDATION

Anca Spătariu

WHAT HAPPENS WHEN YOU TYPE IN AN URL?

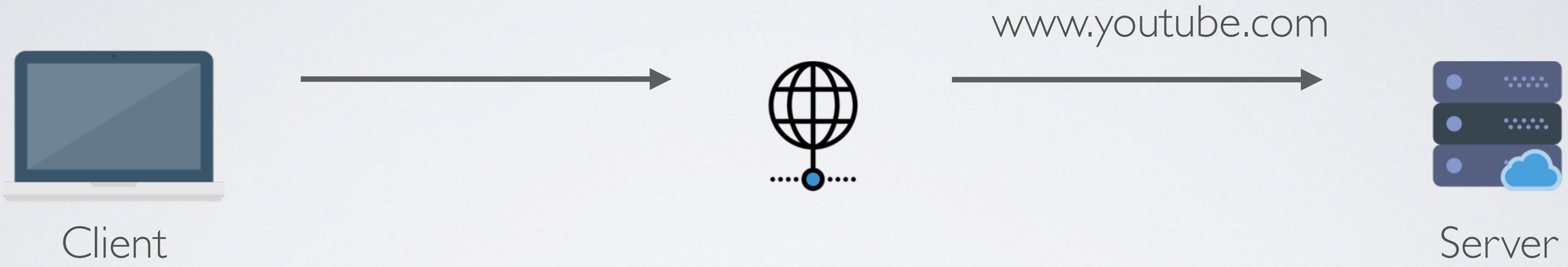
WHAT HAPPENS WHEN YOU TYPE IN AN URL?



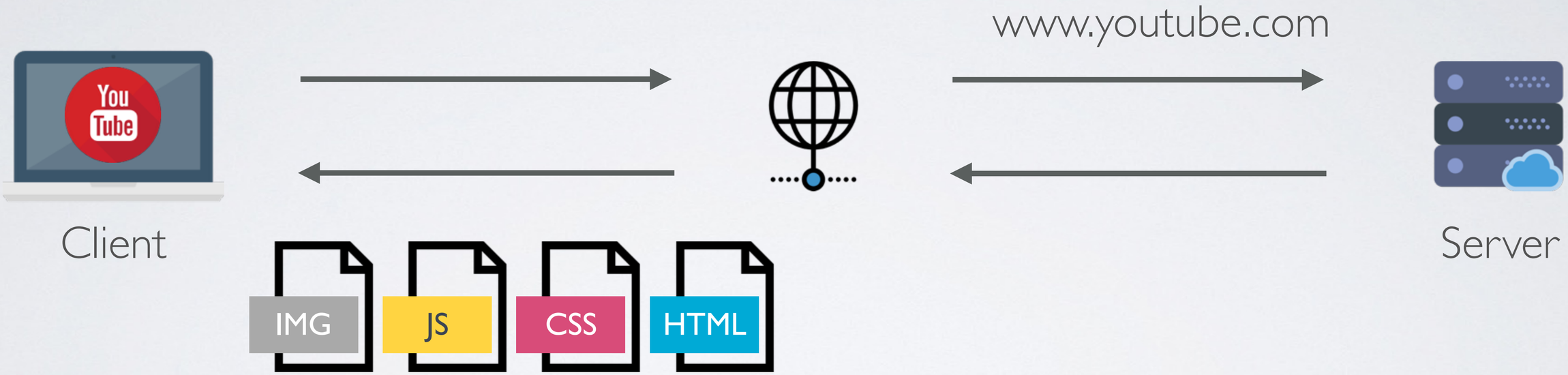
Client

Server

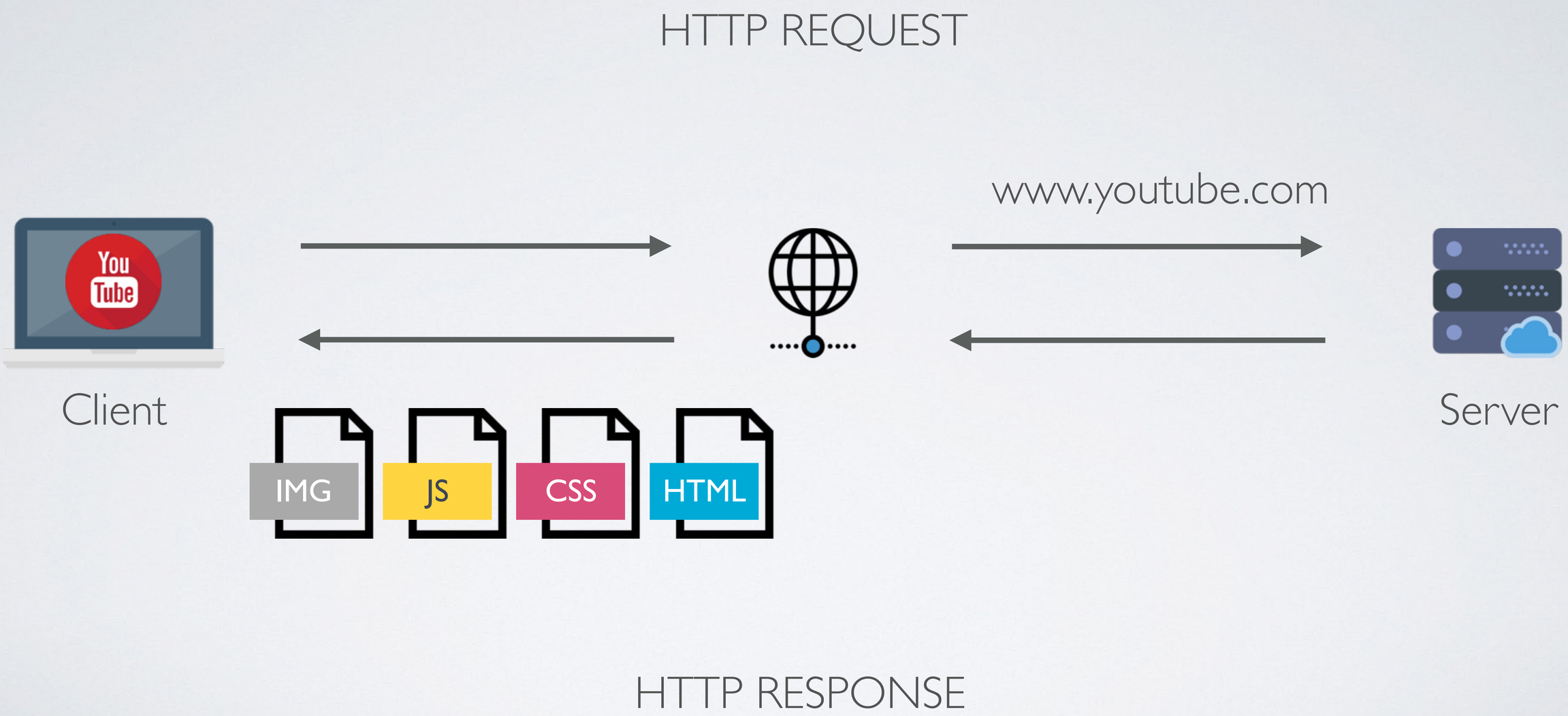
WHAT HAPPENS WHEN YOU TYPE IN AN URL?



WHAT HAPPENS WHEN YOU TYPE IN AN URL?



WHAT HAPPENS WHEN YOU TYPE IN AN URL?



FRONT-END BUILDING BLOCKS

FRONT-END BUILDING BLOCKS

HTML

Building the structure and adding the content

FRONT-END BUILDING BLOCKS

HTML

Building the structure and adding the content

CSS

Making everything pretty

FRONT-END BUILDING BLOCKS

HTML

Building the structure and adding the content

CSS

Making everything pretty

JavaScript

Making it functional

HTML

WHAT DOES HTML MEAN?

WHAT DOES HTML MEAN?

Hyper Text Markup Language

WHAT DOES HTML MEAN?

Hyper Text Markup Language

Hyper Text — Means it's a text that's interactive

WHAT DOES HTML MEAN?

Hyper Text Markup Language

Hyper Text — Means it's a text that's interactive

Markup — Means you have the possibility to mark up sections of the page as tables, lists, paragraphs etc.

WHAT DOES HTML MEAN?

Hyper Text Markup Language

Hyper Text — Means it's a text that's interactive

Markup — Means you have the possibility to mark up sections of the page as tables, lists, paragraphs etc.

Language — Well… I guess it's a language

HOW AND WHERE DO WE WRITE HTML CODE?

HOW AND WHERE DO WE WRITE HTML CODE?

1. HTML code is written in a text files with the `.html` extension.

HOW AND WHERE DO WE WRITE HTML CODE?

1. HTML code is written in a text files with the `.html` extension.

2. The browser "reads" (or interprets) the HTML code and knows how to display it properly, based on the extension provided.

HOW AND WHERE DO WE WRITE HTML CODE?

1. HTML code is written in a text files with the `.html` extension.

2. The browser "reads" (or interprets) the HTML code and knows how to display it properly, based on the extension provided.

3. The file lives on the SERVER, but it's dowloaded to the CLIENT and interpreted locally

HTML TAGS

HTML TAGS

To markup a text, we use html tags. Usually, we have an opening tag and a closing tag, with the content going between these.

HTML TAGS

To markup a text, we use html tags. Usually, we have an opening tag and a closing tag, with the content going between these.

```
<h1> This is a heading </h1>
```

* tag that defines a heading

HTML TAGS

To markup a text, we use html tags. Usually, we have an opening tag and a closing tag, with the content going between these.

```
<h1> This is a heading </h1>
```

* tag that defines a heading

But we can also have individual tags (self closing).

HTML TAGS

To markup a text, we use html tags. Usually, we have an opening tag and a closing tag, with the content going between these.

```
<h1> This is a heading </h1>
```

* tag that defines a heading

But we can also have individual tags (self closing).

```
<br>
```

* tag that defines a single line break

HTML TAGS

While this will work in a browser, it's not standards compliant… and not a real HTML page.

```
<h1> This is a heading </h1>
```

HTML TAGS

While this will work in a browser, it's not standards compliant… and not a real HTML page.

```
<body>
   <h1> This is a heading </h1>
</body>
```

HTML TAGS

While this will work in a browser, it's not standards compliant… and not a real HTML page.

```
<head>
</head>
<body>
   <h1> This is a heading </h1>
</body>
```

As any body should, this one also needs a **head**.

Any code that needs to run before the page is displayed will go here.

HTML TAGS

While this will work in a browser, it's not standards compliant… and not a real HTML page.

```
<!DOCTYPE html>
<html>
   <head>
   </head>
   <body>
      <h1> This is a heading </h1>
   </body>
 </html>
```

And finally, we wrap everything in an `<html>` tag and add the html version tag.

By not specifying a version we tell the browser to use the latest.

HTML TAGS

Usual HTML tags:

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

`<a>` — Anchor or link

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

`<a>` — Anchor or link

`<ul>` — Unordered List

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

`<a>` — Anchor or link

`<ul>` — Unordered List

`<ol>` — Ordered List

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

`<a>` — Anchor or link

`<ul>` — Unordered List

`<ol>` — Ordered List

`<li>` — List item

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

`<a>` — Anchor or link

`<ul>` — Unordered List

`<ol>` — Ordered List

`<li>` — List item

`<img>` — Image

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

`<a>` — Anchor or link

`<ul>` — Unordered List

`<ol>` — Ordered List

`<li>` — List item

`<img>` — Image

`<form>` — Form for user input

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

`<a>` — Anchor or link

`<ul>` — Unordered List

`<ol>` — Ordered List

`<li>` — List item

`<img>` — Image

`<form>` — Form for user input

`<input>` — Input control

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

`<a>` — Anchor or link

`<ul>` — Unordered List

`<ol>` — Ordered List

`<li>` — List item

`<img>` — Image

`<form>` — Form for user input

`<input>` — Input control

`<button>` — Clickable button

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

`<a>` — Anchor or link

`<ul>` — Unordered List

`<ol>` — Ordered List

`<li>` — List item

`<img>` — Image

`<form>` — Form for user input

`<input>` — Input control

`<button>` — Clickable button

`<strong>` — bold text

HTML TAGS

Usual HTML tags:

`<div>` — Section in a document

`<span>` — Section in a document

`<p>` — Paragraph

`<h1>` ... `<h6>` — Headings

`<a>` — Anchor or link

`<ul>` — Unordered List

`<ol>` — Ordered List

`<li>` — List item

`<img>` — Image

`<form>` — Form for user input

`<input>` — Input control

`<button>` — Clickable button

`<strong>` — bold text

`<i>` — italic text

PRACTICE

CSS

WHAT DOES CSS MEAN?

WHAT DOES CSS MEAN?

Cascading Style Sheets

WHAT DOES CSS MEAN?

Cascading Style Sheets

CSS is a language used to describe how the HTML document should be displayed

WHAT DOES CSS MEAN?

Cascading Style Sheets

CSS is a language used to describe how the HTML document should be displayed

CSS not only defines how the HTML document will look in the browser, but also in other media (like print)

WHAT DOES CSS MEAN?

Cascading Style Sheets

CSS is a language used to describe how the HTML document should be displayed

CSS not only defines how the HTML document will look in the browser, but also in other media (like print)

Everything is possible with CSS.

HOW AND WHERE DO WE WRITE CSS?

HOW AND WHERE DO WE WRITE CSS?

In `.css` files — We write CSS code in text files with the `.css` extension. So we could have a total separation from the HTML.

HOW AND WHERE DO WE WRITE CSS?

In `.css` files — We write CSS code in text files with the `.css` extension. So we could have a total separation from the HTML.

In `.html` files, inside a `<style>` tag — We can also include a bunch of styling directly into the HTML files, wrapped inside a `<style>` tag.

HOW AND WHERE DO WE WRITE CSS?

In `.css` files — We write CSS code in text files with the `.css` extension. So we could have a total separation from the HTML.

In `.html` files, inside a `<style>` tag — We can also include a bunch of styling directly into the HTML files, wrapped inside a `<style>` tag.

In `.html` files, inline — For old times sake, we can also inline the CSS code, in the `style` attribute of an element.

HOW AND WHERE DO WE WRITE CSS?

In `.css` files

CSS

```
.first-class {
  color: red;
}
.second-class {
  color: red;
}
div {
  color: red;
}
```

HOW AND WHERE DO WE WRITE CSS?

Inside the HTML, in a `<style>` tag

HTML

```
<style>
 .first-class {
   color: red;
 }
</style>

 <h1 class="first-class"> Hello World! </h1>
```

Inline

HTML

```
 <h1 style="color: red;"> Hello World! </h1>
```

SELECTORS: ELEMENTS

SELECTORS: ELEMENTS

How do we apply a style on a particular tag? We use an element selector.

SELECTORS: ELEMENTS

How do we apply a style on a particular tag? We use an element selector.

HTML

```
<h1>
  Hello World!
</h1>
```

SELECTORS: ELEMENTS

CSS

How do we apply a style on a particular tag? We use an element selector.

HTML

```html
<h1>
  Hello World!
</h1>
```

CSS

```css
h1 {
  color: blue;
}
```

SELECTORS: ELEMENTS

How do we apply a style on a particular tag? We use an element selector.

HTML

```html
<h1>
  Hello World!
</h1>
```

CSS

```css
h1 {
  color: blue;
}
```

BROWSER

Hello World!

SELECTORS: ELEMENTS

Descendent selectors:

SELECTORS: ELEMENTS

Descendent selectors:

HTML

```
<h1>
  Hello World!
</h1>

<div>
  <p> First </p>
</div>

<p> Second </p>
```

Descendent selectors:

HTML

```html
<h1>
  Hello World!
</h1>

<div>
  <p> First </p>
</div>

<p> Second </p>
```

CSS

```css
p {
  color: magenta;
}
```

Descendent selectors:

HTML

```html
<h1>
  Hello World!
</h1>

<div>
  <p> First </p>
</div>

<p> Second </p>
```

CSS

```css
p {
  color: magenta;
}
```

CSS

```css
div p {
  color: magenta;
}
```

SELECTORS: ELEMENTS

Note the difference:

BROWSER

First

Second

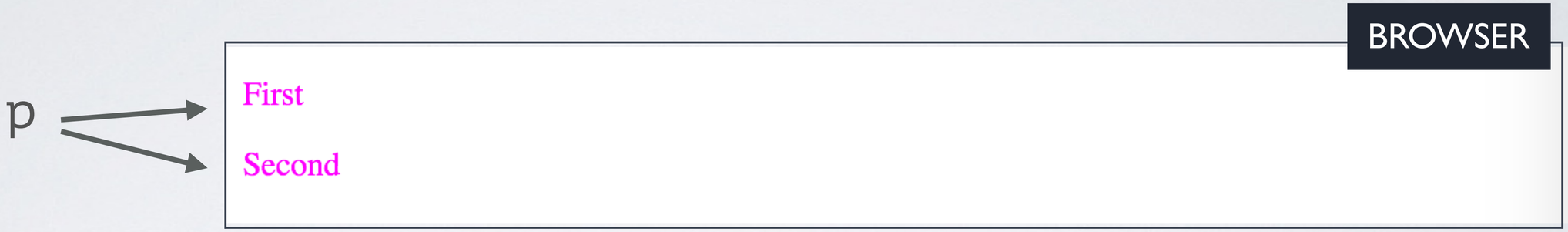
BROWSER

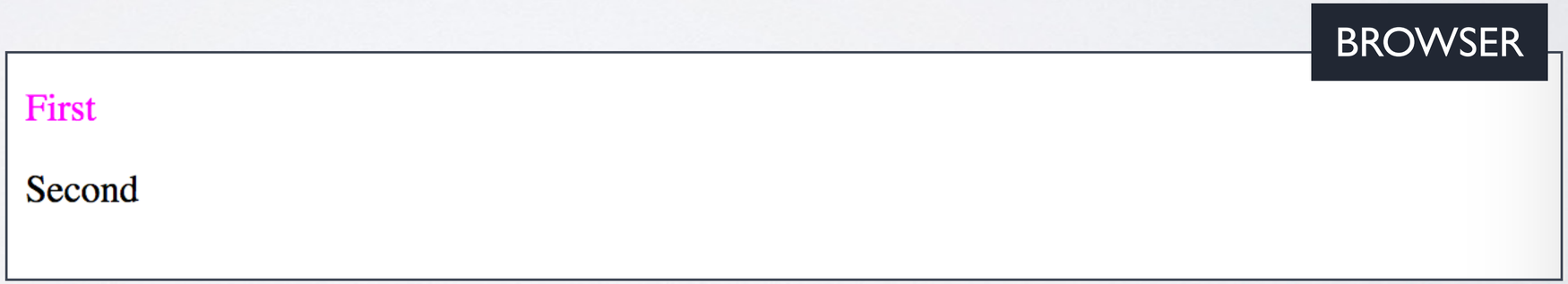
First

Second

SELECTORS: ELEMENTS

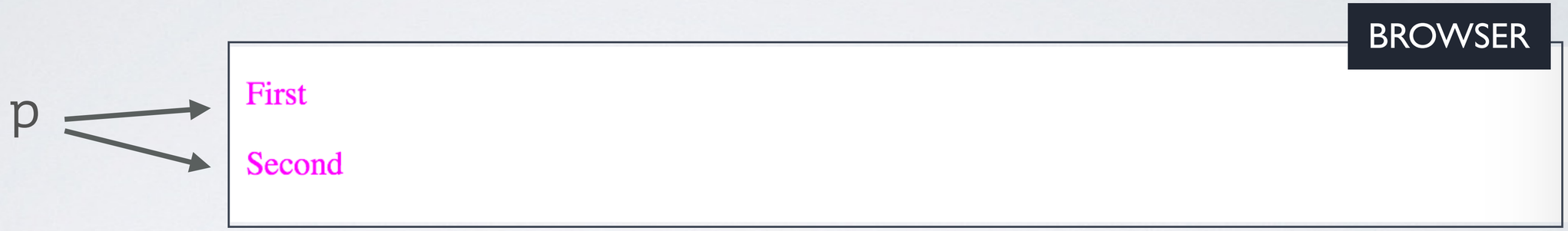
Note the difference:

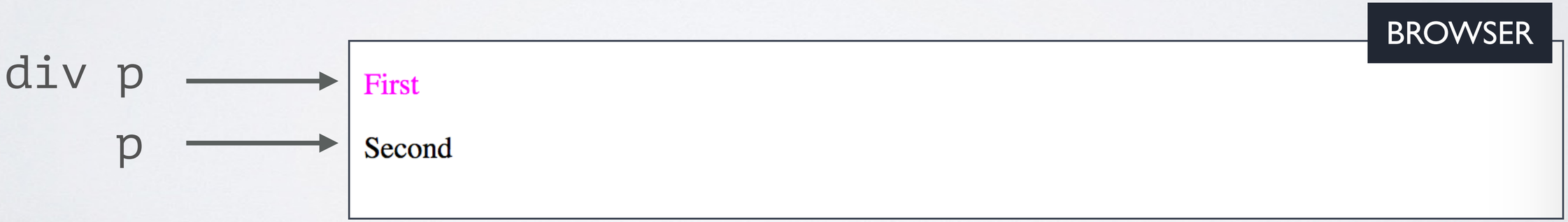




SELECTORS: ELEMENTS

Note the difference:





SELECTORS: CLASSES

SELECTORS: CLASSES

The `.class` selector selects elements with a specific class attribute.

SELECTORS: CLASSES

The `.class` selector selects elements with a specific class attribute.

HTML

```
<h1 class="title">
  Hello World!
</h1>
```

SELECTORS: CLASSES

CSS

The `.class` selector selects elements with a specific class attribute.

HTML

```
<h1 class="title">
  Hello World!
</h1>
```

CSS

```
.title {
  color: blue;
}
```

SELECTORS: CLASSES

CSS

The `.class` selector selects elements with a specific class attribute.

HTML

```html
<h1 class="title">
  Hello World!
</h1>
```

CSS

```css
.title {
  color: blue;
}
```

BROWSER

Hello World!

SELECTORS: ELEMENTS

Descendent class selectors:

Descendent class selectors:

HTML

```
<h1>
  Hello World!
</h1>

<div class="wrapper">
  <p class="text"> First </p>
</div>

<p> Second </p>
```

Descendent class selectors:

HTML

```
<h1>
  Hello World!
</h1>

<div class="wrapper">
  <p class="text"> First </p>
</div>

<p> Second </p>
```

CSS

```
.wrapper .text {
  color: magenta;
}
```

Descendent class selectors:

HTML

```html
<h1>
  Hello World!
</h1>

<div class="wrapper">
  <p class="text"> First </p>
</div>

<p> Second </p>
```

CSS

```css
.wrapper .text {
  color: magenta;
}
```

CSS

```css
.text {
  color: magenta;
}
```

Descendent class selectors:

The space is very important!

HTML

```
<h1>
  Hello World!
</h1>

<div class="wrapper">
  <p class="text"> First </p>
</div>

<p> Second </p>
```

CSS

```
.wrapper .text {
  color: magenta;
}
```

CSS

```
.text {
  color: magenta;
}
```

SELECTORS: ELEMENTS

With space: selects an element with the class .text, that's a child of an element with the class .wrapper

```
.wrapper .text {
  color: magenta;
}
```

With space: selects an element with the class .text, that's a child of an element with the class .wrapper

```
.wrapper .text {
  color: magenta;
}
```

Without space: selects an element that has both the classes .wrapper and .text

```
.wrapper.text {
  color: magenta;
}
```

SELECTORS: ID

SELECTORS: ID

The `#id` selector selects the element with a specified id.

SELECTORS: ID

The `#id` selector selects the element with a specified id.

HTML

```
<h1 id="hero">
  Hello World!
</h1>
```

SELECTORS: ID

CSS

The `#id` selector selects the element with a specified id.

HTML

```html
<h1 id="hero">
  Hello World!
</h1>
```

CSS

```css
#hero {
  color: blue;
}
```

SELECTORS: ID

The `#id` selector selects the element with a specified id.

HTML

```html
<h1 id="hero">
  Hello World!
</h1>
```

CSS

```css
#hero {
  color: blue;
}
```

BROWSER

Hello World!

SELECTORS: PSEUDO-CLASSES

SELECTORS: PSEUDO-CLASSES

A keyword added to a selector that acts as a modifier. It specifies a special state of the element.

SELECTORS: PSEUDO-CLASSES

A keyword added to a selector that acts as a modifier. It specifies a special state of the element.

HTML

```
<a>
  Click me
</a>
```

SELECTORS: PSEUDO-CLASSES

CSS

A keyword added to a selector that acts as a modifier. It specifies a special state of the element.

HTML

```
<a>
  Click me
</a>
```

CSS

```
a {
  color: magenta;
}
```

SELECTORS: PSEUDO-CLASSES

CSS

A keyword added to a selector that acts as a modifier. It specifies a special state of the element.

HTML

```
<a>
  Click me
</a>
```

CSS

```
a {
  color: magenta;
}
a:hover {
  color: green;
}
```

SELECTORS: PSEUDO-CLASSES

Other popular pseudo-classes.

SELECTORS: PSEUDO-CLASSES

Other popular pseudo-classes.

Dynamic:

`:hover`

`:active`

`:focus`

`:visited`

SELECTORS: PSEUDO-CLASSES

Other popular pseudo-classes.

Dynamic:

`:hover`

`:active`

`:focus`

`:visited`

Structural:

`:first-child`

`:nth-child(n)`

`:nth-of-type(n)`

`:empty`

THE BOX MODEL

THE BOX MODEL

Think of an element as a box:

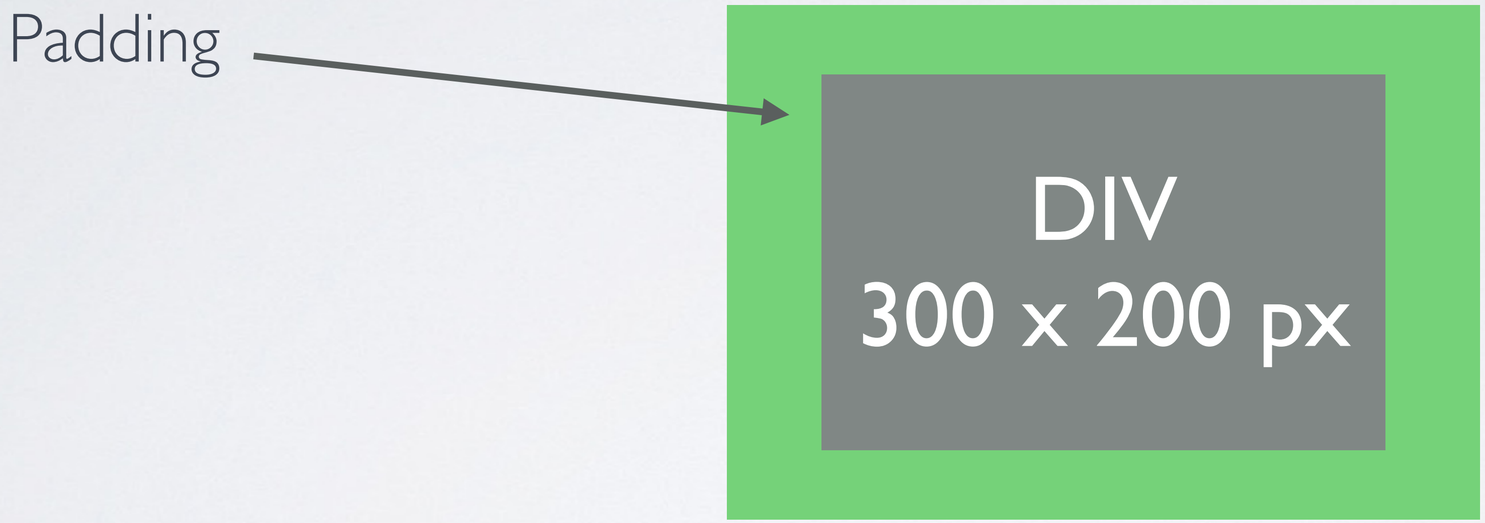
THE BOX MODEL

Think of an element as a box:

DIV
300 x 200 px

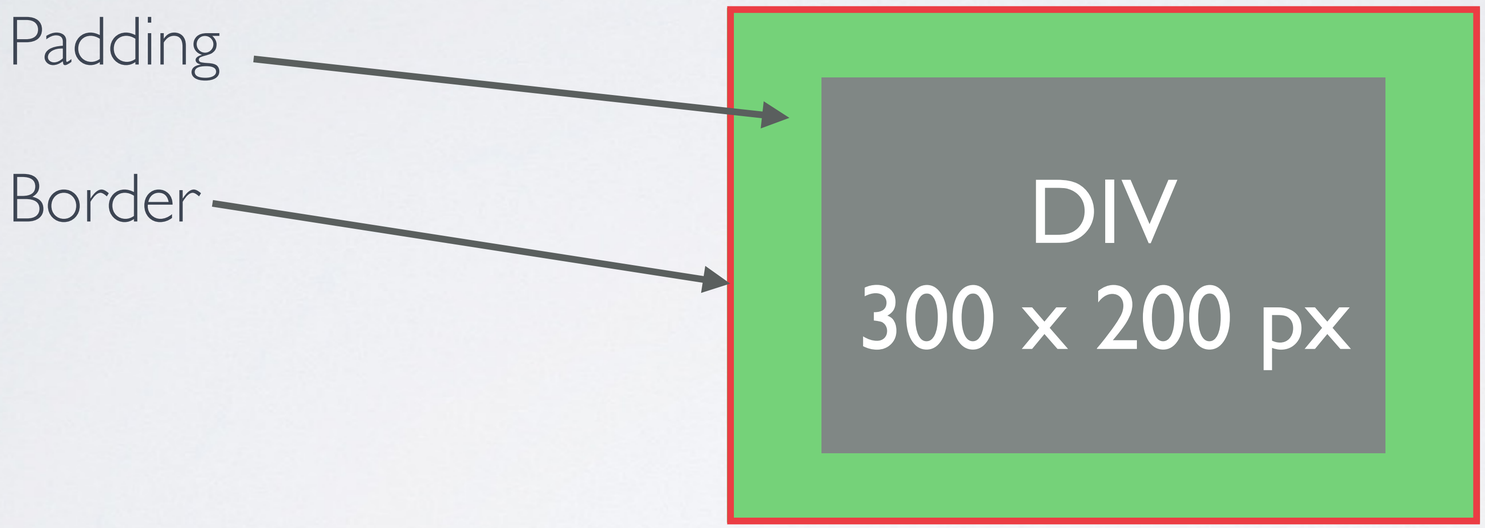
THE BOX MODEL

Think of an element as a box:



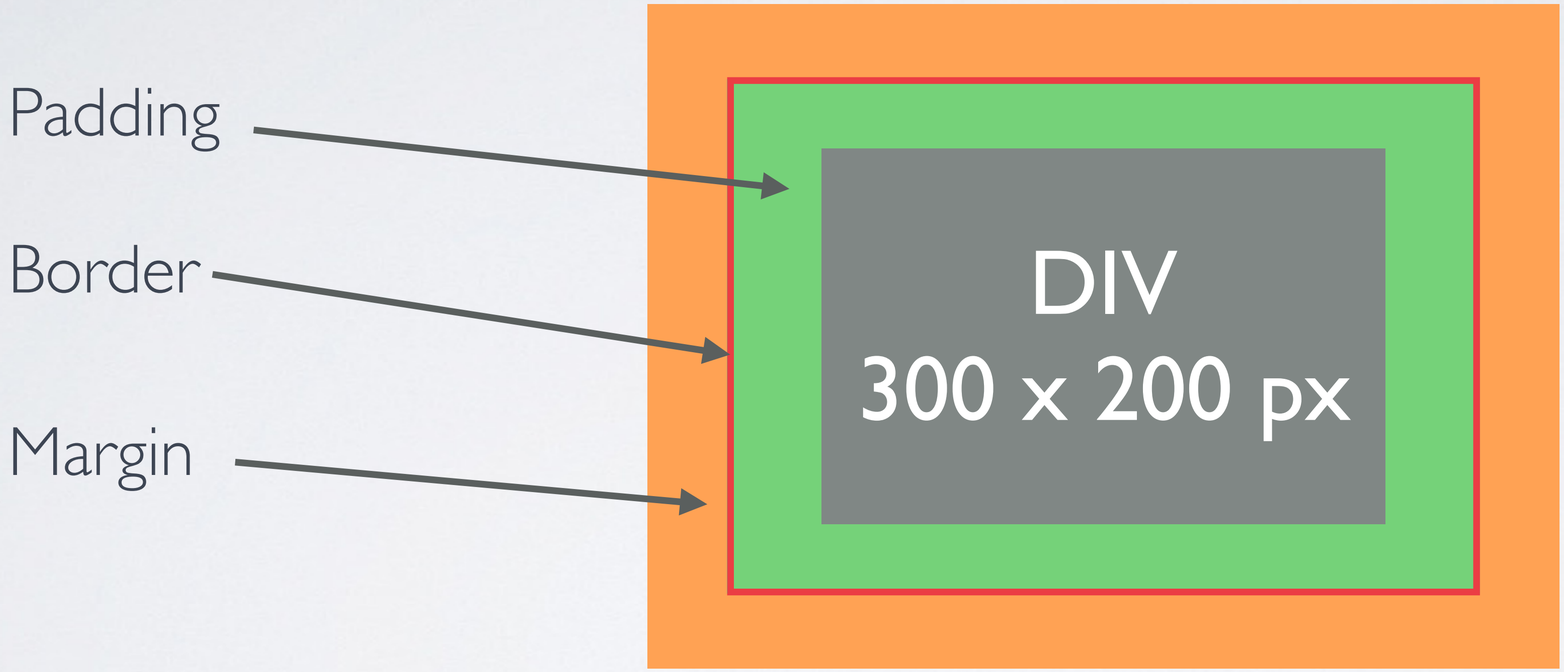
THE BOX MODEL

Think of an element as a box:



THE BOX MODEL

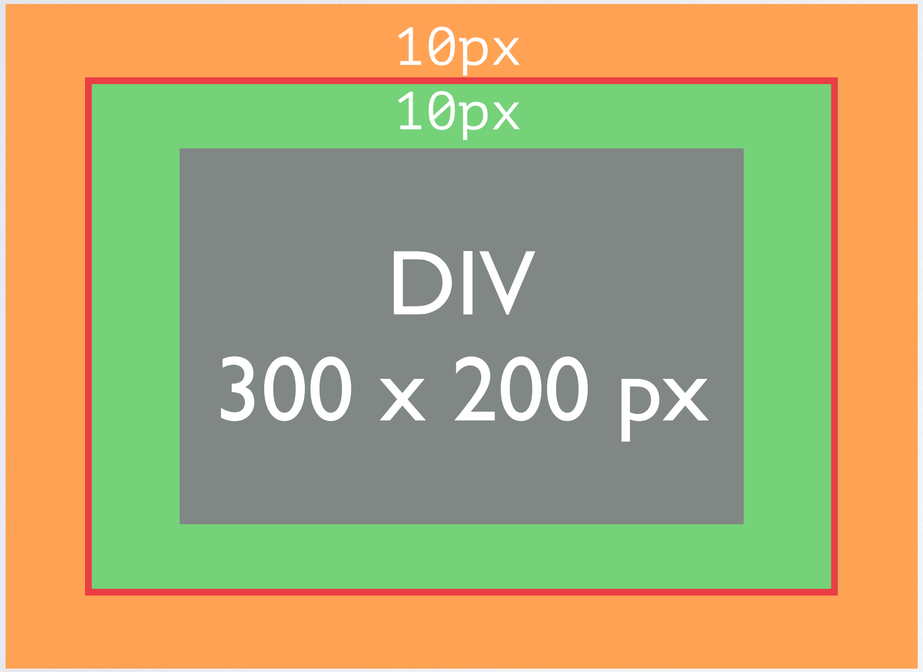
Think of an element as a box:



THE BOX MODEL

CSS

Think of an element as a box:

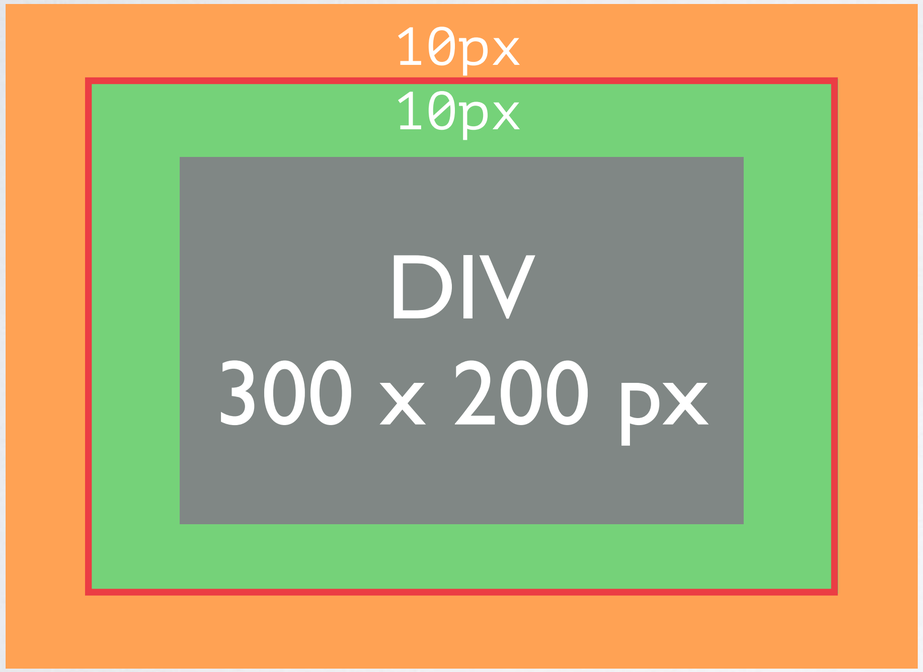


CSS

```
div {
  width: 300px;
  height: 200px;
  padding: 10px;
  margin: 10px;
  border: 2px solid red;
}
```

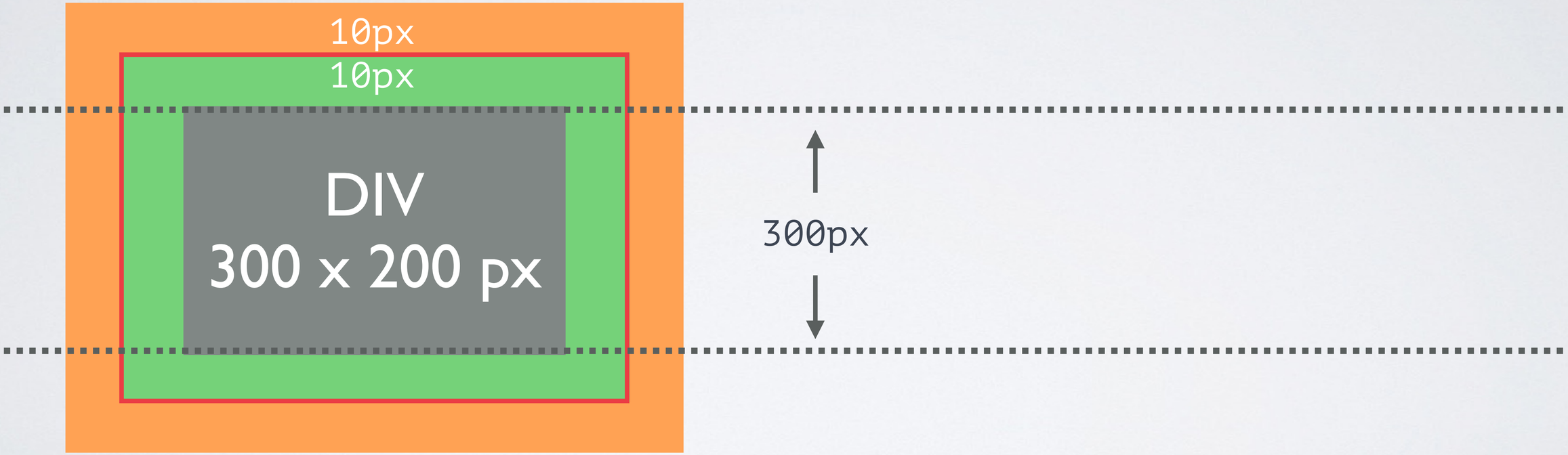
THE BOX MODEL

Think of an element as a box:



THE BOX MODEL

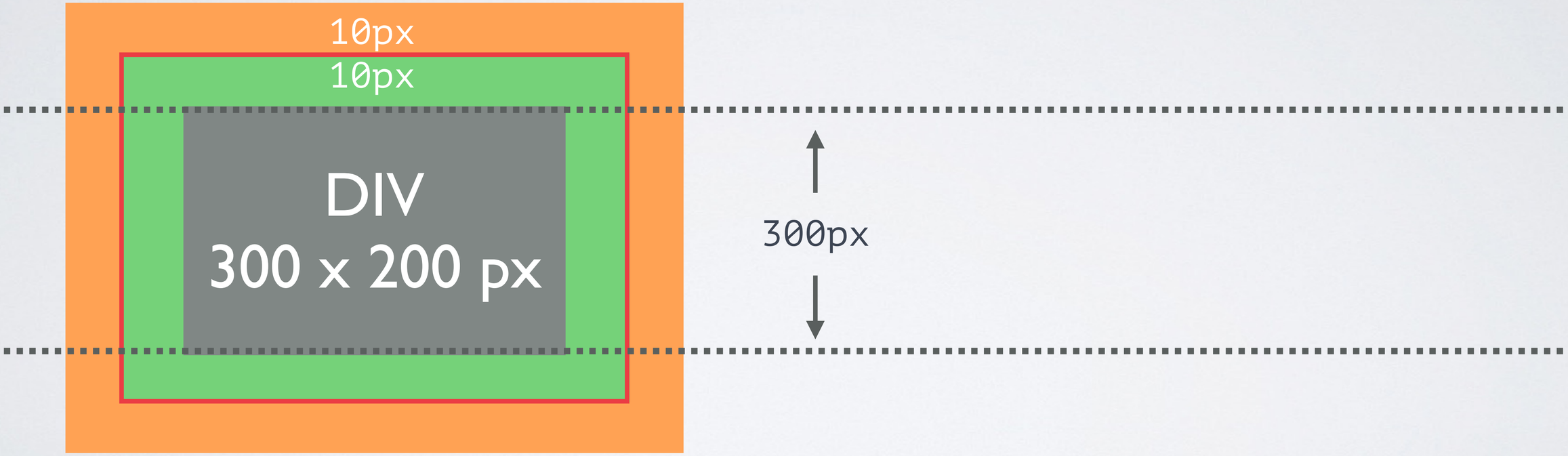
Think of an element as a box:



THE BOX MODEL

Think of an element as a box:

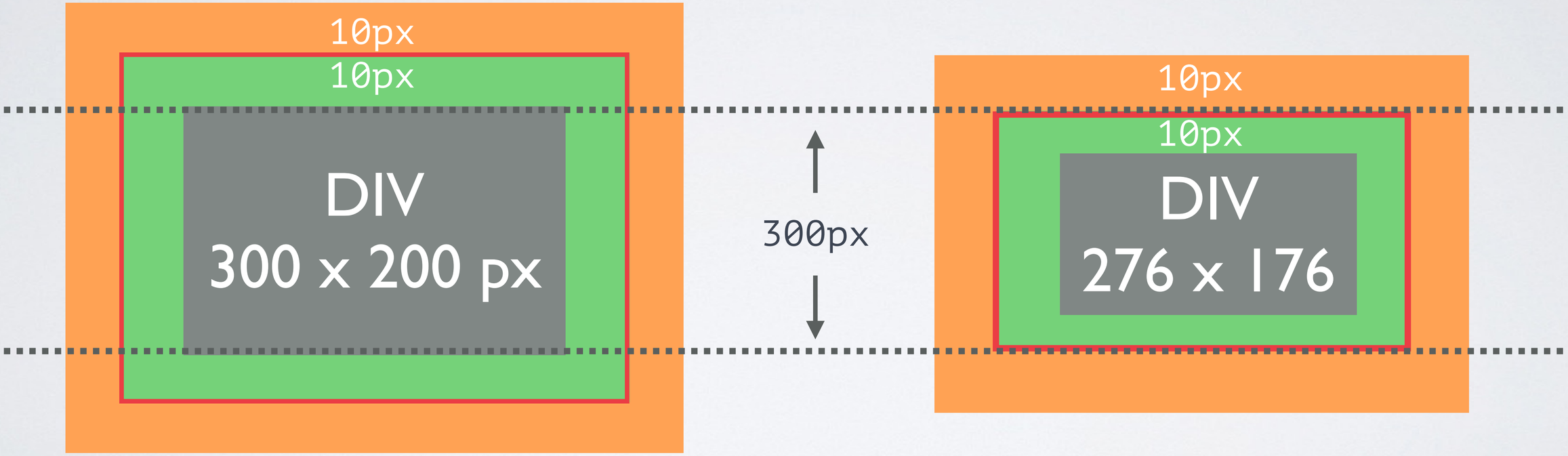
```
box-sizing: content-box;
```



THE BOX MODEL

Think of an element as a box:

```
box-sizing: content-box;
```

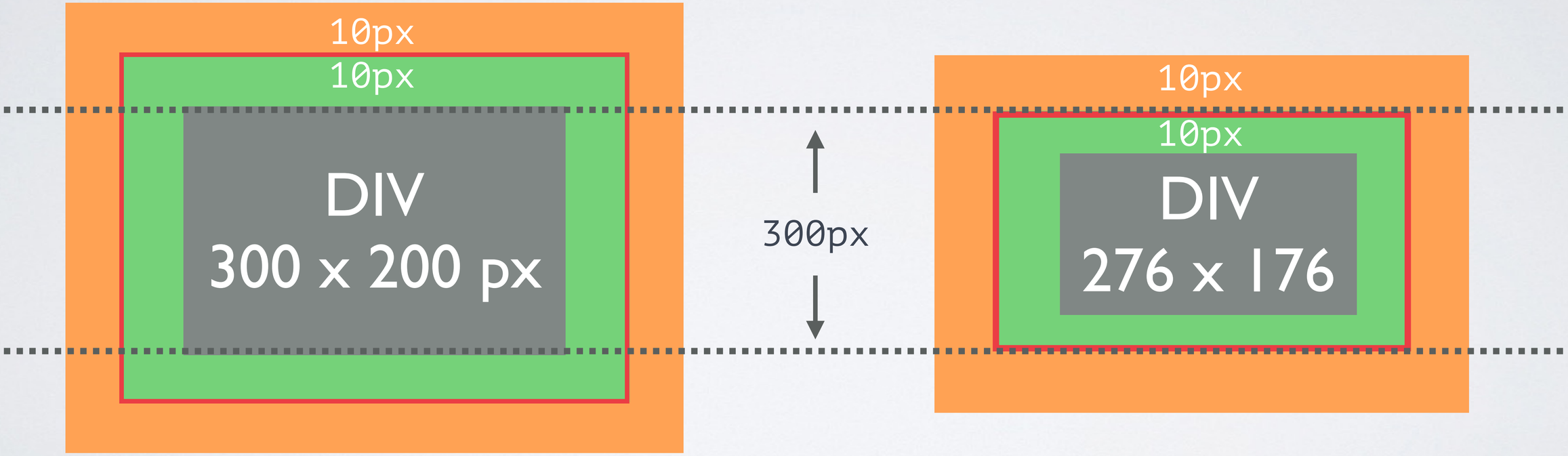


THE BOX MODEL

Think of an element as a box:

```
box-sizing: content-box;
```

```
box-sizing: border-box;
```



DISPLAY

DISPLAY

The `display` property specifies the type of box used for an HTML element.

The most common `display` property values are:

DISPLAY

The `display` property specifies the type of box used for an HTML element.

The most common `display` property values are:

```
display: inline;

display: block;

display: inline-block;
```

DISPLAY

```
display: block;
```

DISPLAY

CSS

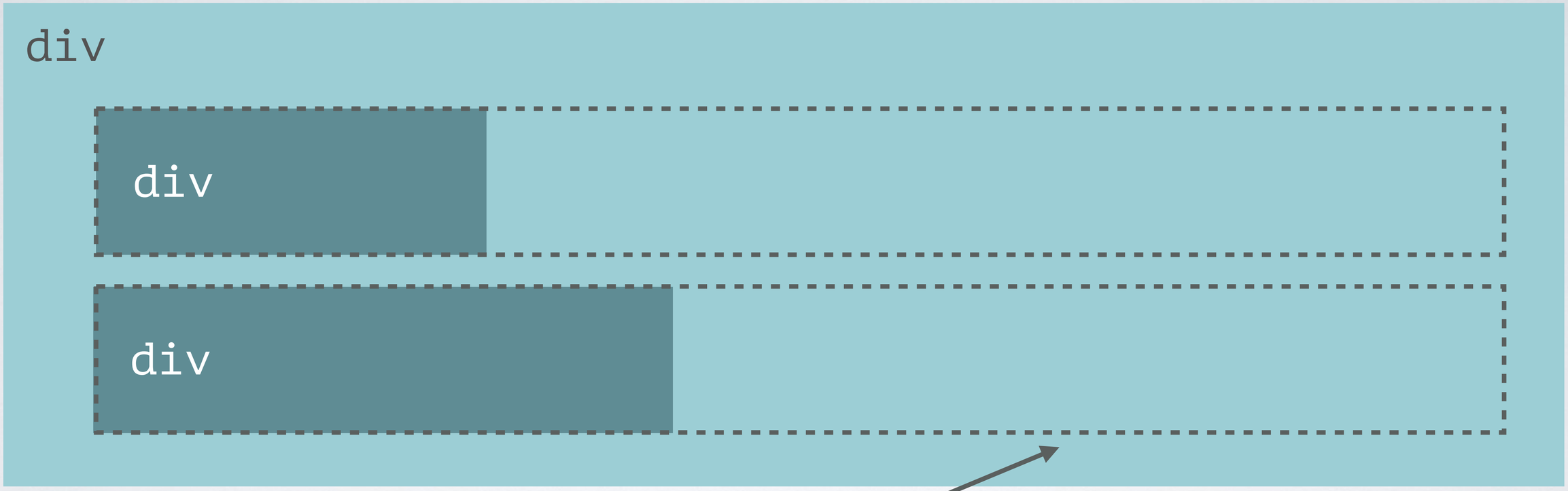
```
display: block;
```

div

```
display: block;
```

```
display: block;
```

```
display: block;
```



If the width is not specifies, the display block div will take 100% of the parent.

DISPLAY

```
display: inline;
```

DISPLAY

```
display: inline;
```

div

```
display: inline;
```

```
display: inline;
```

DISPLAY

```
display: inline-block;
```

```
display: inline-block;
```

```
display: inline-block;
```

```
display: inline-block;
```

FLOAT

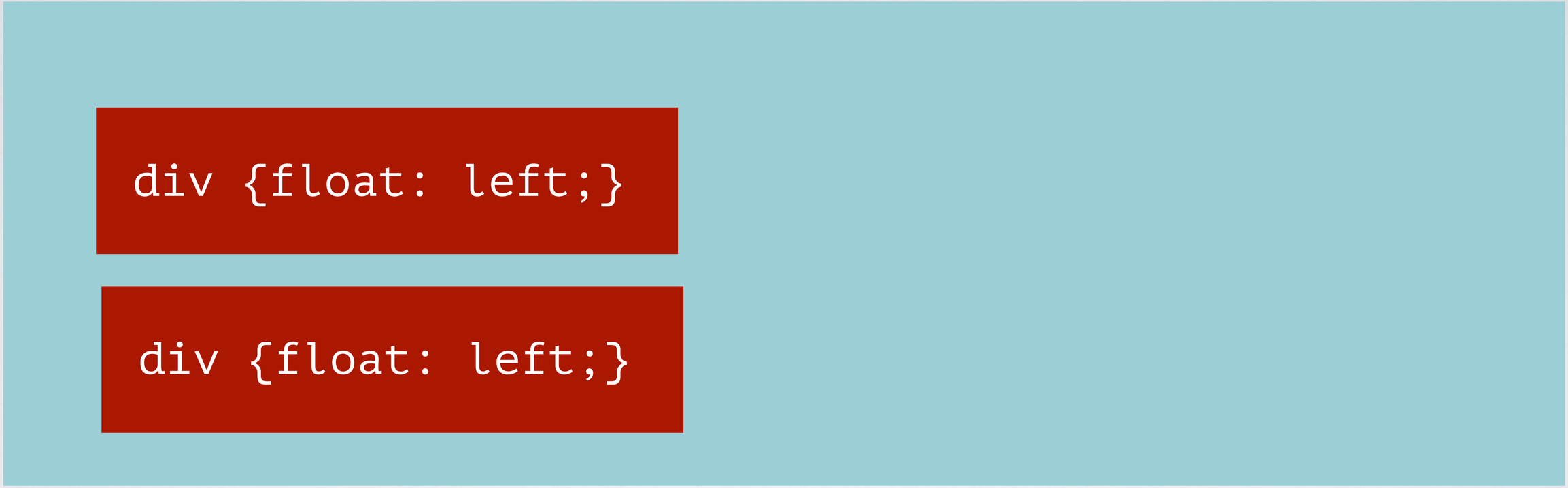
`float: left;` and `float: rigth;`

`float: left;` and `float: rigth;`

`float: left;` and `float: rigth;`

`float: left;` and `float: rigth;`

```
float: left;
```



```
float: left;
```

div

```
div {float: left;}
```

```
div {float: left;}
```

```
float: left;
```

```
div {float: left;}
```

```
div {float: left;}
```

```
float: left;
```

div

```
div {float: left;}
```

```
div {float: left;}
```

FLOAT

```
float: left;
```

```
float: left;
```

div

```
div {float: left;}
```

```
div {float: left;}
```

```
float: rigth;
```

```
div {float: left;}
```

```
div {float: right;}
```

```
float: rigth;
```

```
div
```

```
div {float: left;}
```

```
div {float: right;}
```

```
float: rigth;
```

div

```
div {float: left;}
```

```
div {float: right;}
```

POSITIONING

POSITIONING

The `position` property specifies the type of positioning method used for an element. It can be set to:

POSITIONING

The `position` property specifies the type of positioning method used for an element. It can be set to:

```
position: static;
```

POSITIONING

The `position` property specifies the type of positioning method used for an element. It can be set to:

```
position: static;

position: absolute;
```

POSITIONING

The `position` property specifies the type of positioning method used for an element. It can be set to:

```
position: static;

position: absolute;

position: relative;
```

POSITIONING

The `position` property specifies the type of positioning method used for an element. It can be set to:

```
position: static;

position: absolute;

position: relative;

position: fixed;
```

```
position: static;
```

position: static;

```
div {position: static;}
```

```
position: static;
```

```
div {position: static;}
```

```
div {position: static;}
```

```
position: static;
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
position: static;
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
position: relative;
```

```
div {position: relative;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
position: absolute;
```

```
div {position: relative;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: absolute;}
```

```
position: absolute;
```

```
div {position: relative;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: absolute; top: 0;}
```

```
position: absolute;
```

```
div
```

```
div {position: absolute; top: 0;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
position: fixed;
```

```
position: fixed;
```

```
div {position: static;}
```

```
position: fixed;
```

```
div {position: static;}
```

```
div {position: static;}
```

```
position: fixed;
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
position: fixed;
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
position: fixed;
```

```
div {position: static;}
```

```
div {position: fixed;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: static;}
```

```
div {position: static;}
div {position: fixed;}
div {position: static;}
```

```
position: fixed;
```

```
div {position: static;}
```

```
div {position: fixed;}
```

```
div {position: static;}
```

```
div {position: static;}
```

PRACTICE