Introduction

CSS Foundations

Hi, I'm Emma

- Engineering Manager @ Spotify
- Previously Software Engineer @ GoToMeeting + IBM
- From Upstate New York but live in Stockholm
- New mom to Freja
- FEM Instructor, LinkedIn Learning Instructor, Ladybug Podcast co-host

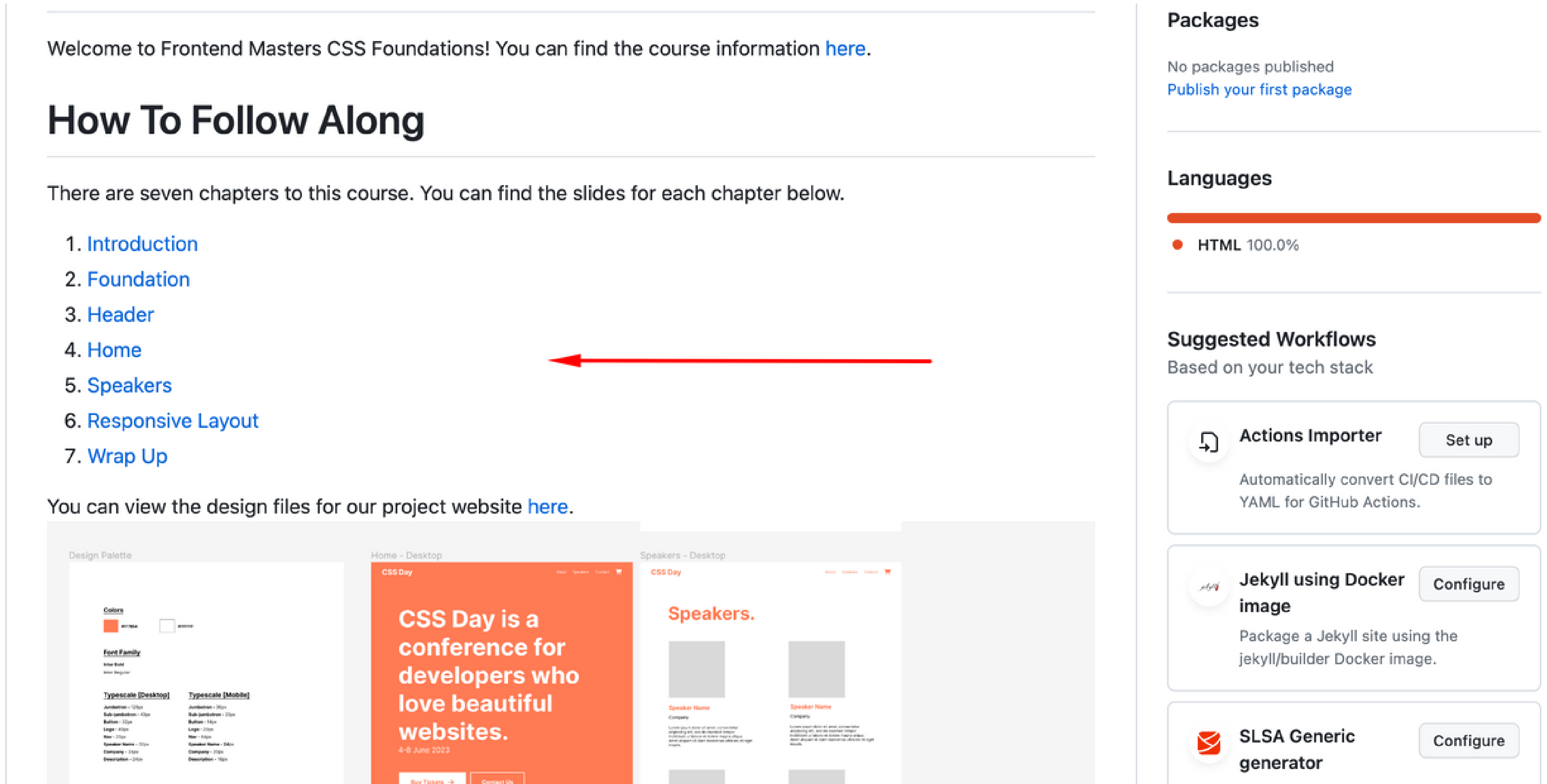
How This Course Is Structured

There are seven chapters:

- Introduction (now)
- Foundation
- Header
- Home
- Speakers
- Responsive Layout
- Wrap Up

How To Follow Along

https://github.com/emmabostian/fem-css-foundations



Welcome to Frontend Masters CSS Foundations! You can find the course information here.

How To Follow Along

There are seven chapters to this course. You can find the slides for each chapter below.

1. Introduction
2. Foundation
3. Header
4. Home
5. Speakers
6. Responsive Layout
7. Wrap Up

You can view the design files for our project website here.

Packages

No packages published
Publish your first package

Languages

● HTML 100.0%

Suggested Workflows
Based on your tech stack

Actions Importer — Set up
Automatically convert CI/CD files to YAML for GitHub Actions.

Jekyll using Docker image — Configure
Package a Jekyll site using the jekyll/builder Docker image.

SLSA Generic generator — Configure

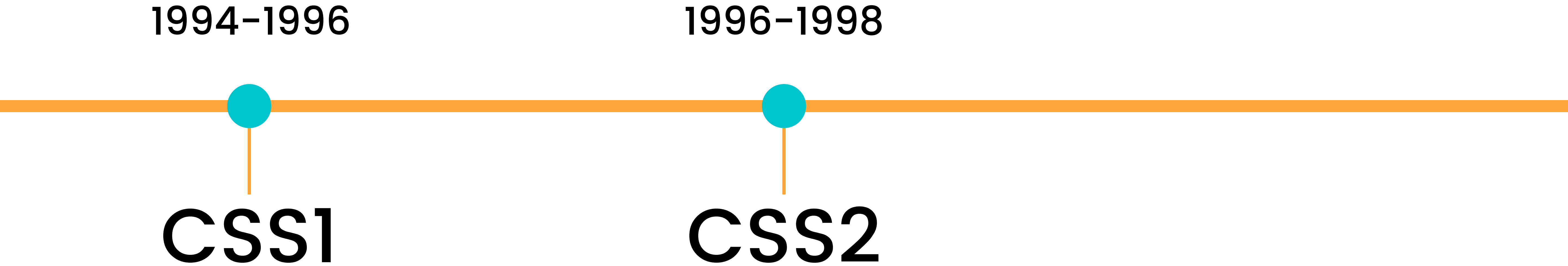
Design Palette
Home - Desktop
Speakers - Desktop
CSS Day is a conference for developers who love beautiful websites.
Speakers.

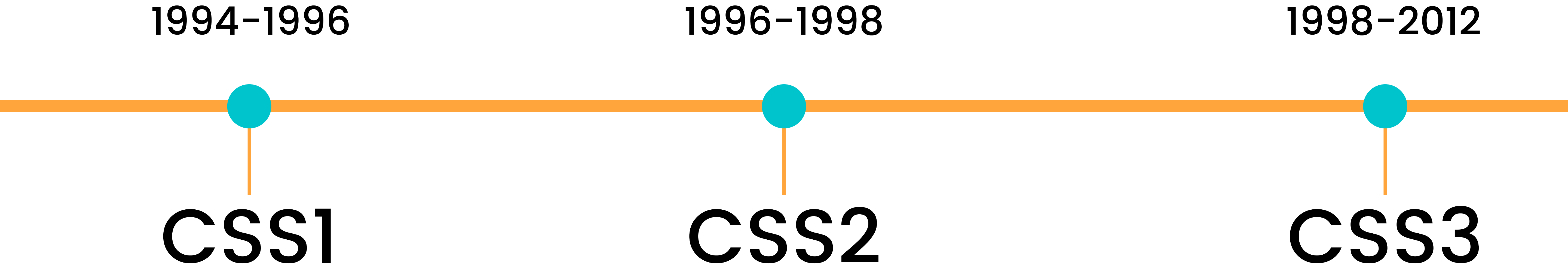
What Is CSS?

"CSS, Cascading Stylesheets, allows you to create great-looking web pages."

MDN

1994–1996
CSS1


1994-1996
CSS1
1996-1998
CSS2


1994-1996
CSS1
1996-1998
CSS2
1998-2012
CSS3

How CSS Is Rendered

How CSS Is Rendered

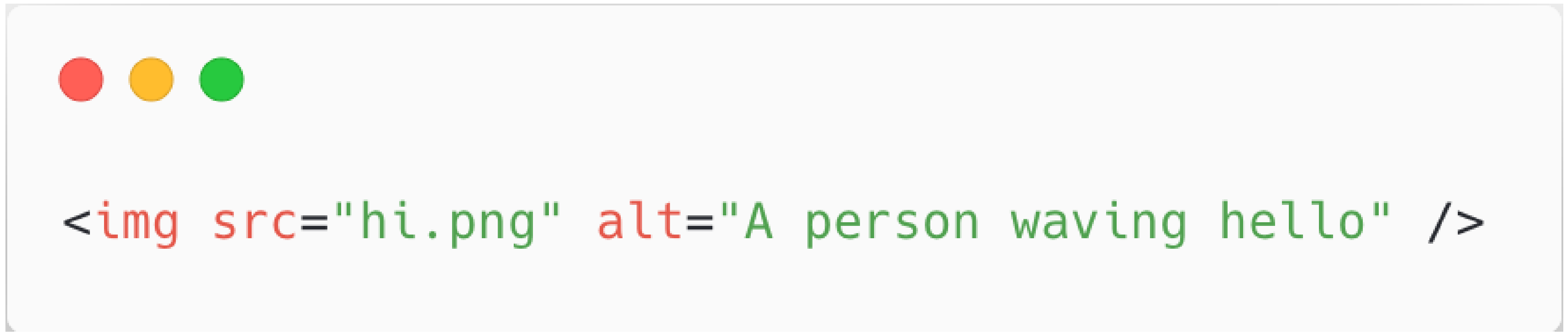
1. Browser loads HTML
2. Converts HTML to the DOM
3. Fetches linked resources
4. Browser parses CSS
5. Render tree is laid out
6. UI is painted

Terminology + Core Concepts

Elements

- Replaced
- Non-Replaced

Replaced Elements are elements where the content is replaced by something not directly represented in the document content.



```html
<img src="hi.png" alt="A person waving hello" />
```

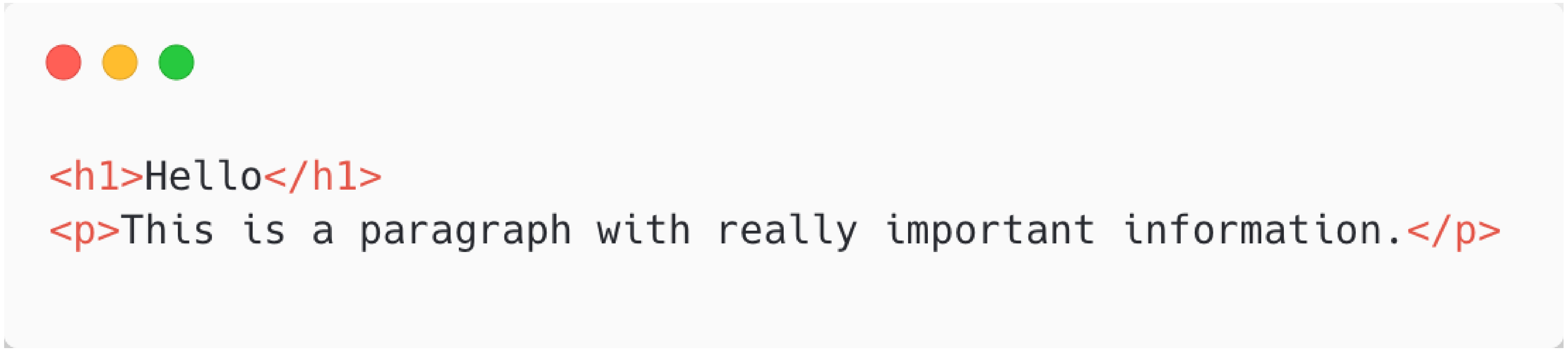
Non-replaced Elements are elements where the content is presented by the user agent (generally a browser) inside a box generated by the element itself.

```
<h1>Hello</h1>
```

Elements

- Block
- Inline

Block Elements generate an element box that fills its parent element's content area and cannot have other elements beside it.

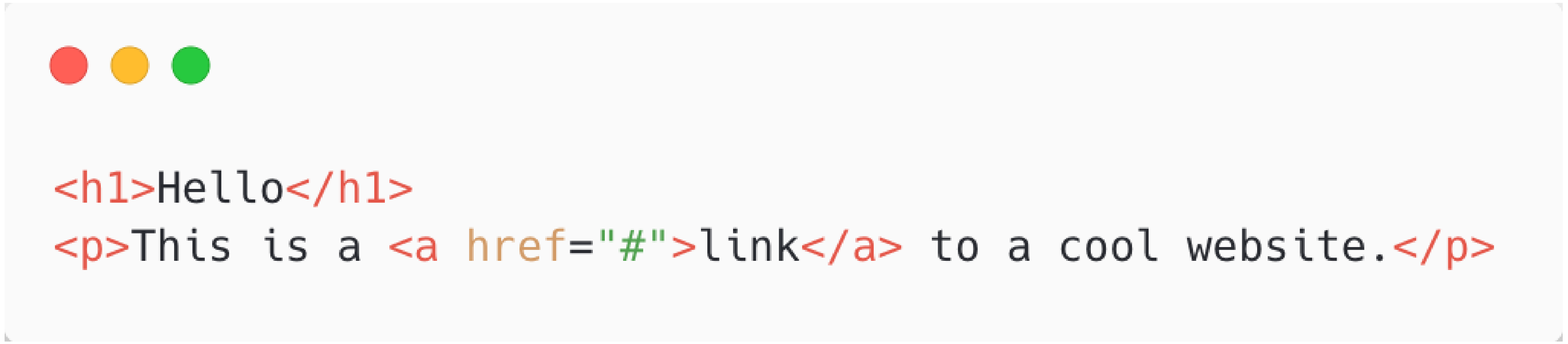


```html
<h1>Hello</h1>
<p>This is a paragraph with really important information.</p>
```

Hello

This is a paragraph with really important information.

Inline Elements generate an element box within a line of text and do not break up the flow of that line.



```
<h1>Hello</h1>
<p>This is a <a href="#">link</a> to a cool website.</p>
```

Hello

This is a [link](#) to a cool website.

Documents have a structure which is different from the visual structure.

Selectors

Selects the HTML element or elements you want to apply some styles to.

HTML

```html
<h1>Hello</h1>
<p>This is a <a href="#">link</a> to a cool website.</p>
```

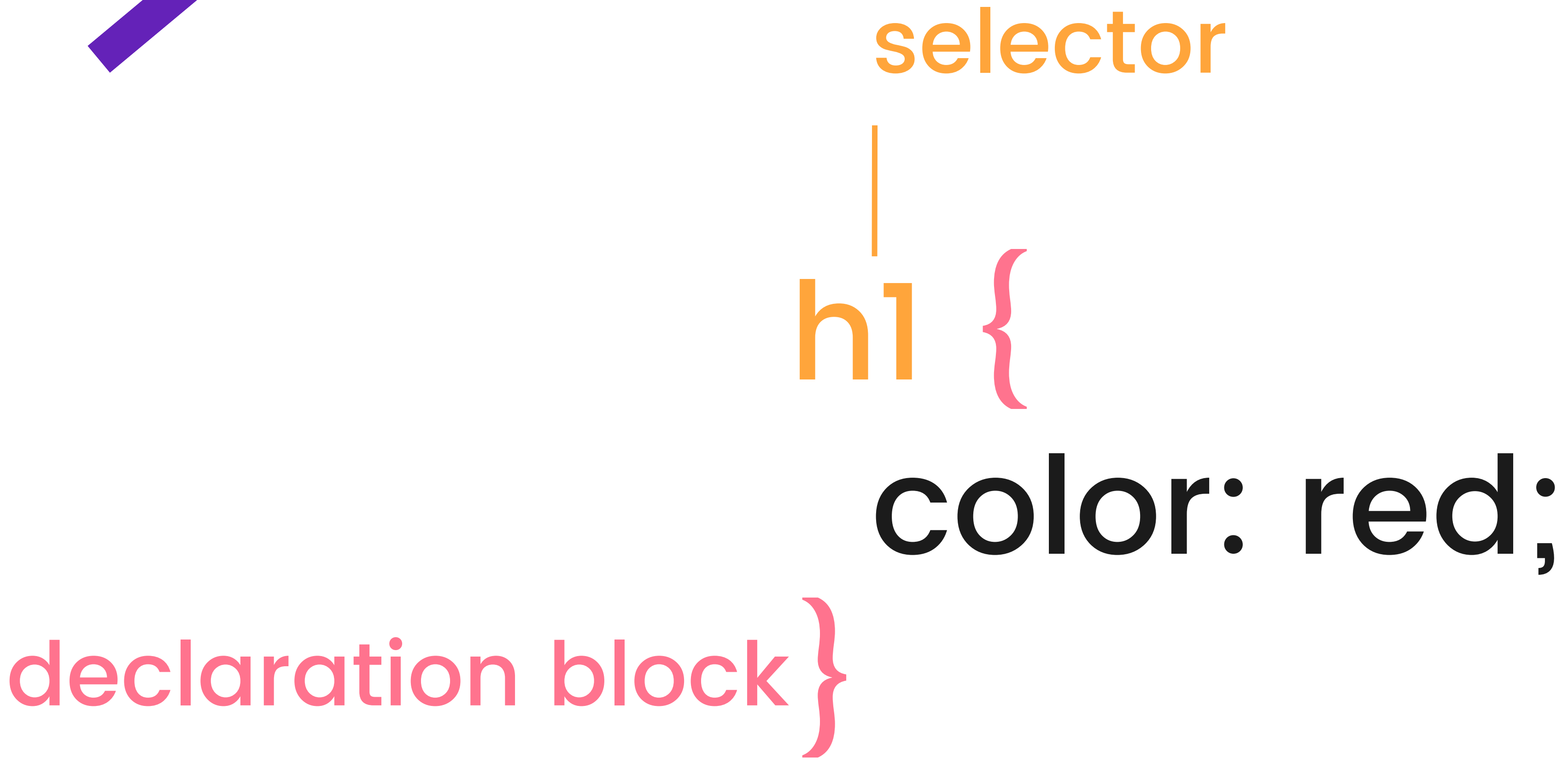
Hello

This is a [link](#) to a cool website.

CSS

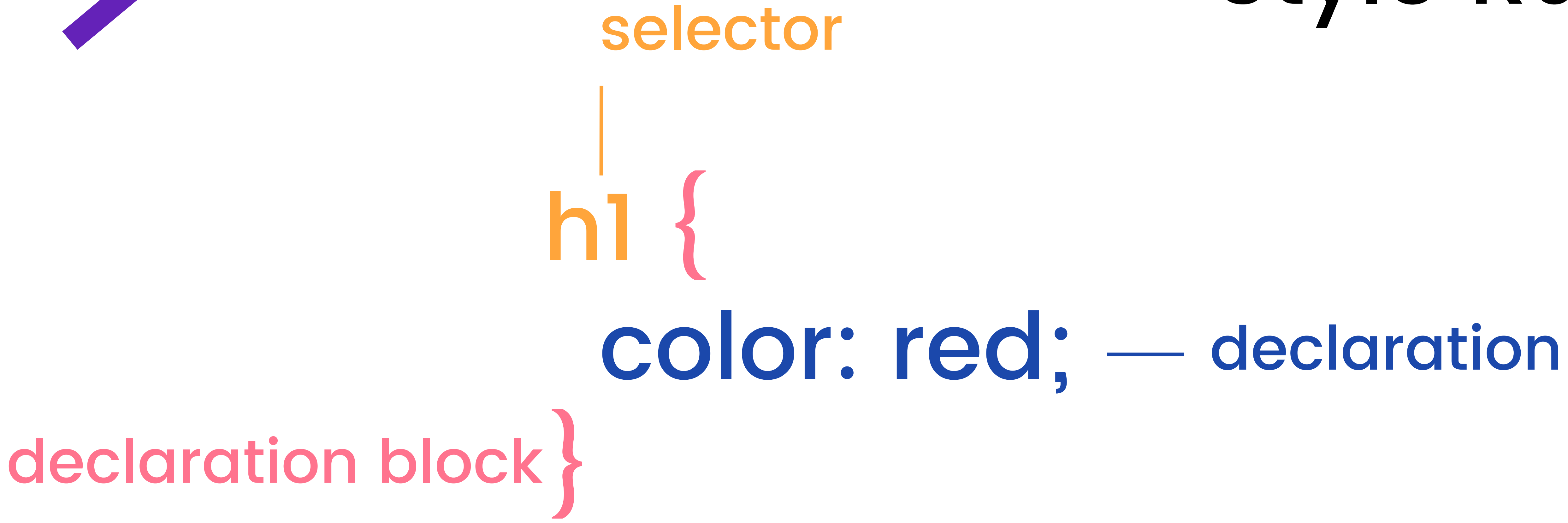
```css
h1 {
  color: red;
}
```

Style Rule



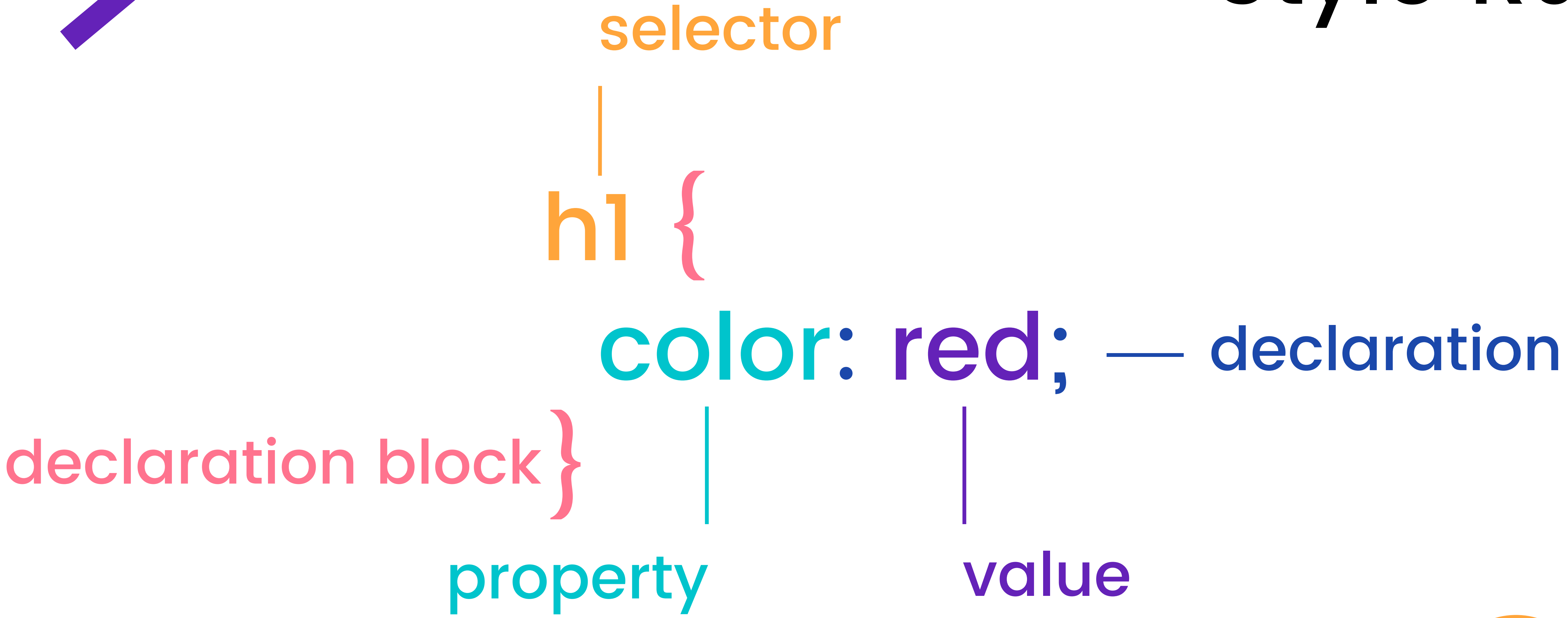
```
h1 {
   color: red;
}
```

Style Rule



```
h1 {
  color: red;
}
```

Style Rule



```
h1 {
  color: red;
}
```

Selectors

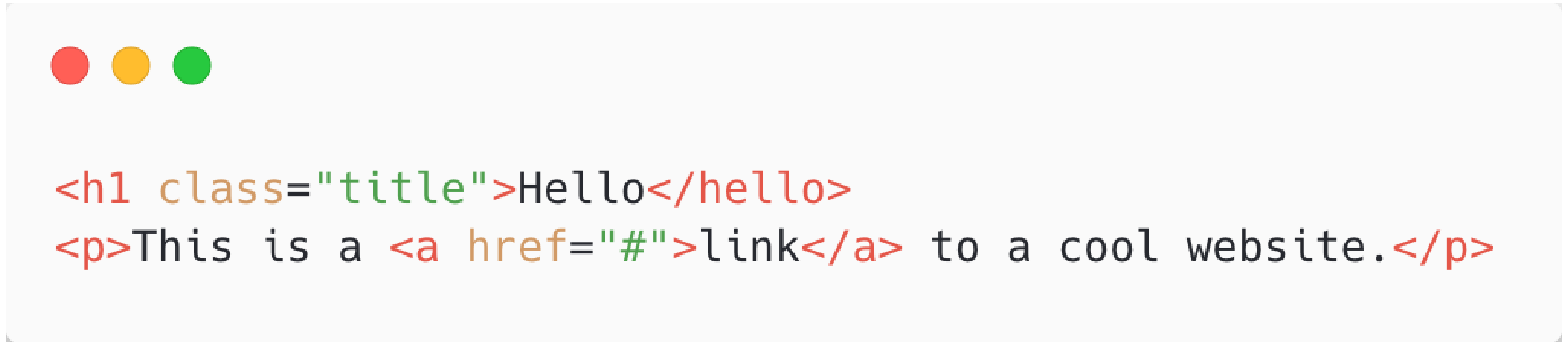
- Type selectors

Selectors

- Type selectors
- Class selectors

Class Selector

HTML

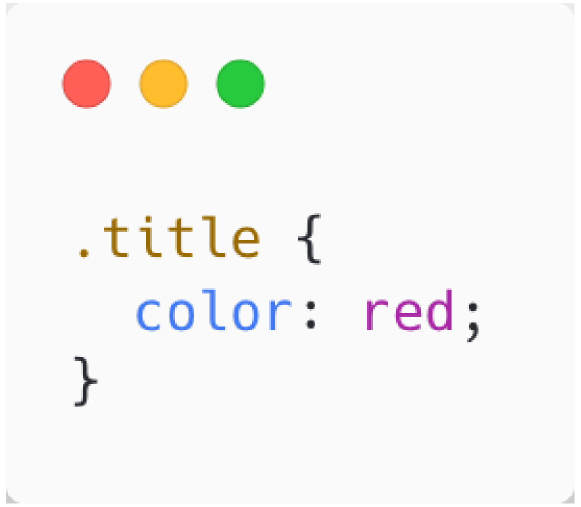


```html
<h1 class="title">Hello</hello>
<p>This is a <a href="#">link</a> to a cool website.</p>
```

Hello

This is a link to a cool website.

CSS



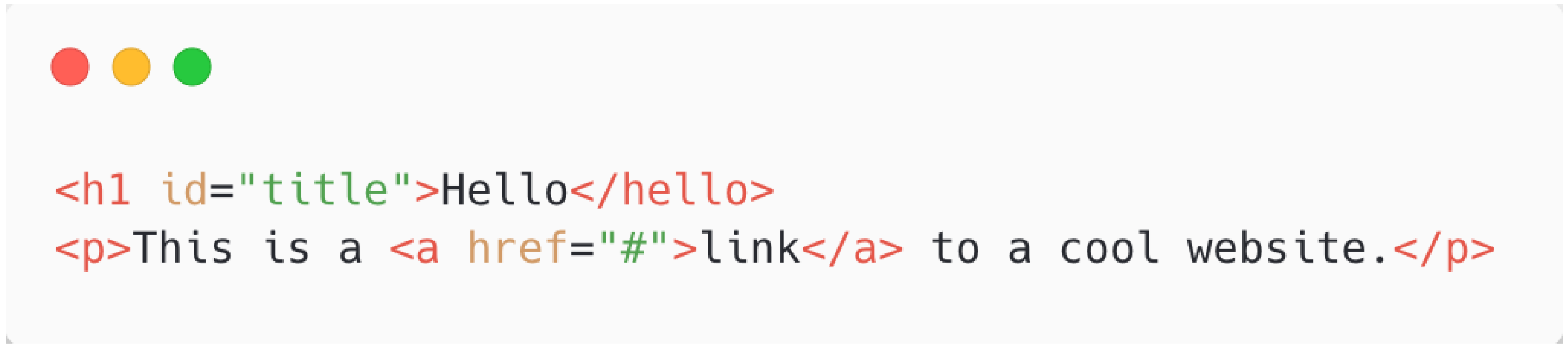
```css
.title {
  color: red;
}
```

Selectors

- Type selectors
- Class selectors
- ID selectors

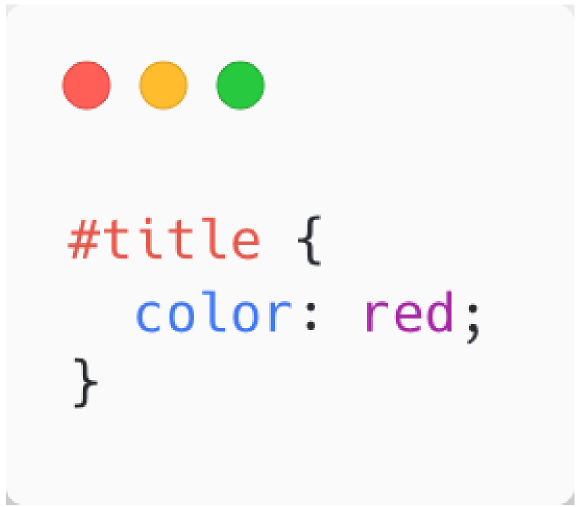
ID Selector

HTML

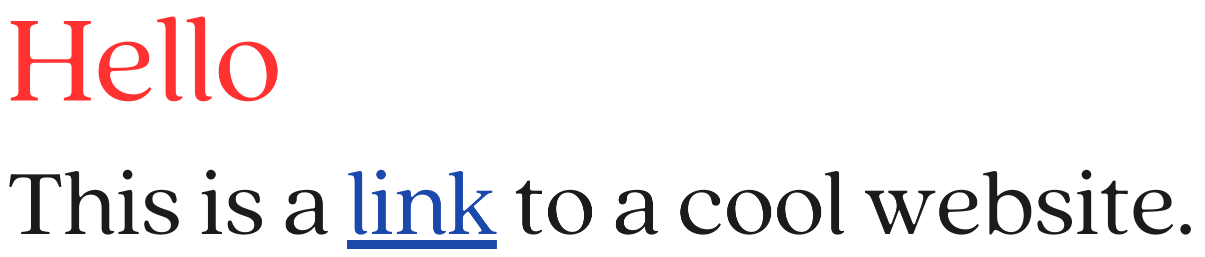


```html
<h1 id="title">Hello</hello>
<p>This is a <a href="#">link</a> to a cool website.</p>
```

CSS



```css
#title {
  color: red;
}
```



Hello

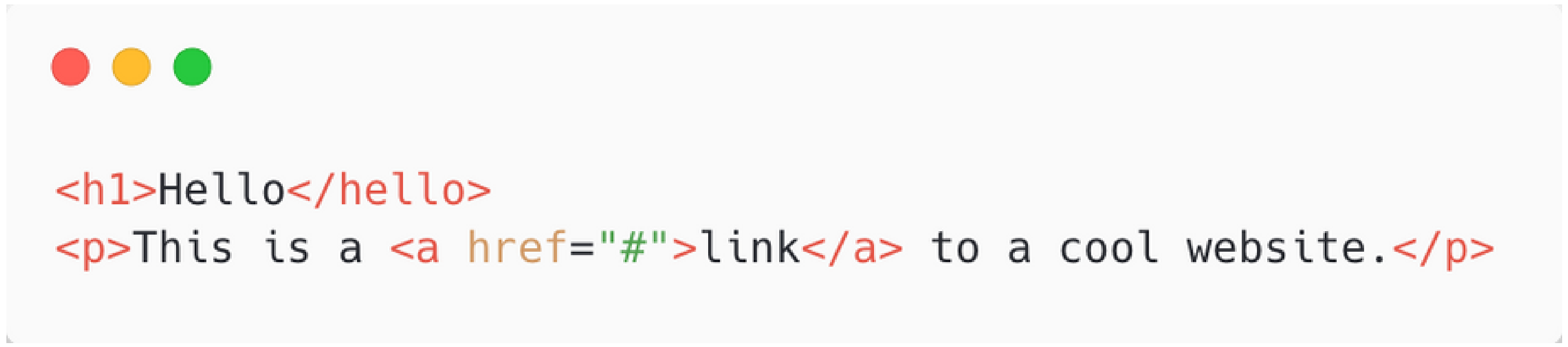
This is a link to a cool website.

Selectors

- Type selectors
- Class selectors
- ID selectors
- Universal selector

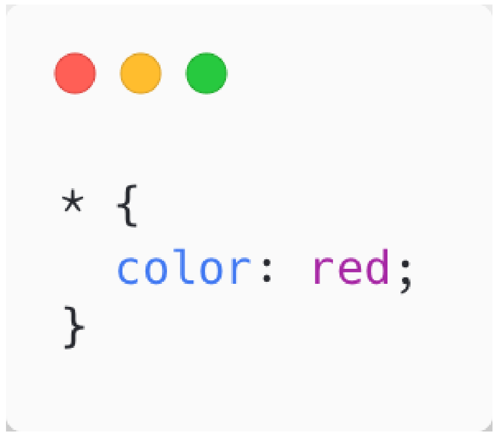
Universal Selector

HTML



```html
<h1>Hello</hello>
<p>This is a <a href="#">link</a> to a cool website.</p>
```

CSS



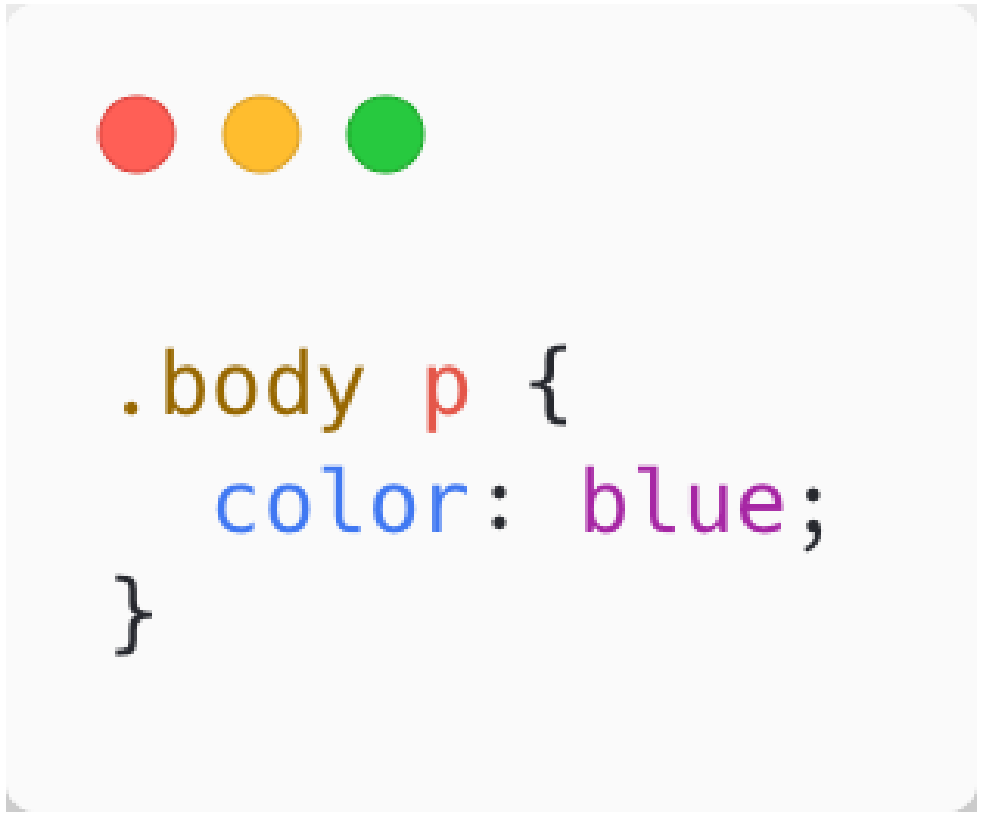
```css
* {
  color: red;
}
```

Hello

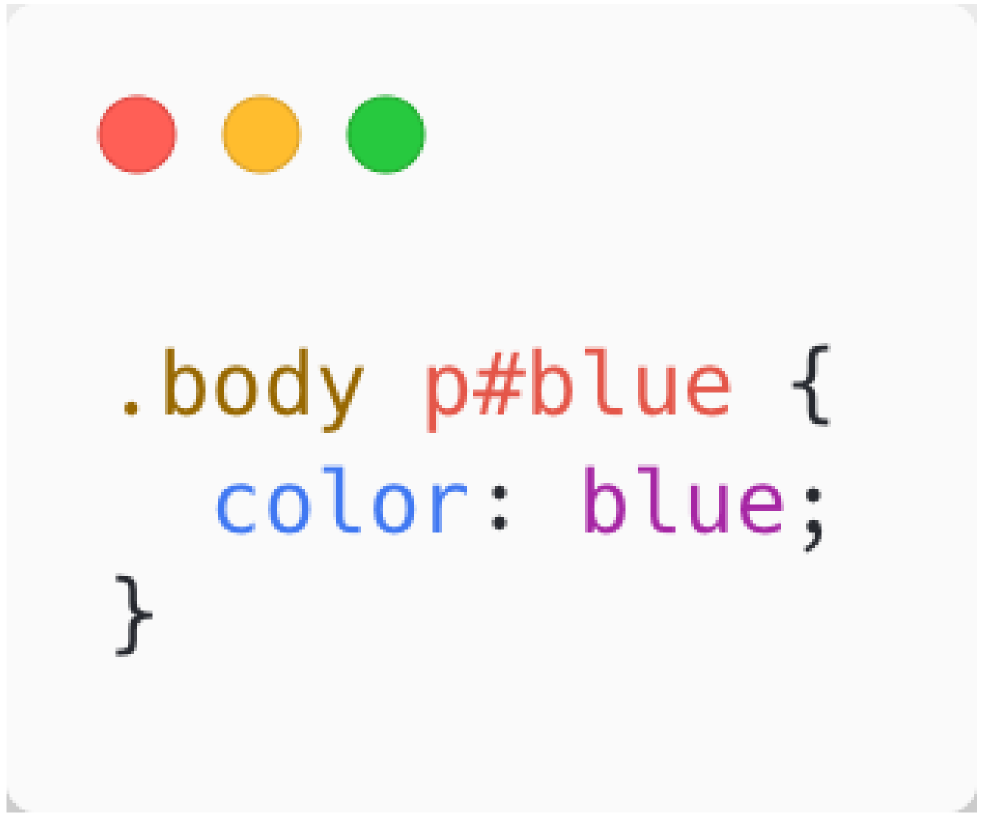
This is a link to a cool website.

Combining Selectors

You can combine selectors to be more specific about which element you want to select.

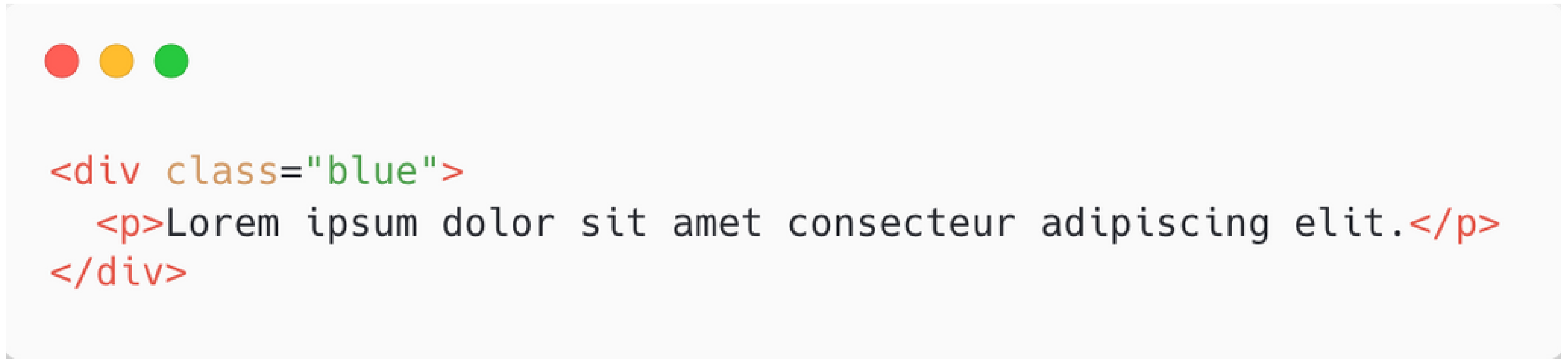


```css
.body p {
  color: blue;
}
```



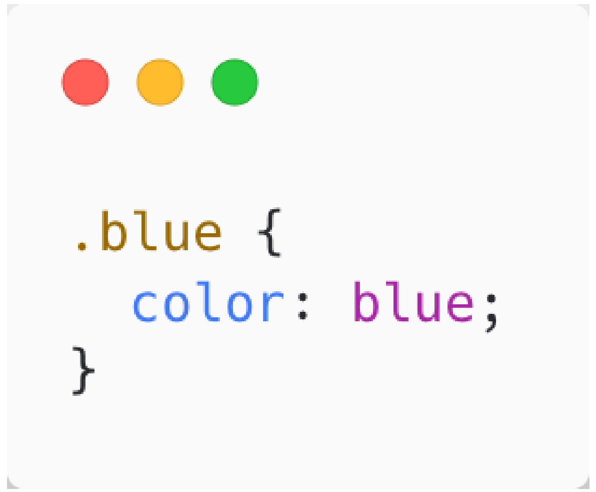
```css
.body p#blue {
  color: blue;
}
```

HTML



```html
<div class="blue">
  <p>Lorem ipsum dolor sit amet consecteur adipiscing elit.</p>
</div>
```

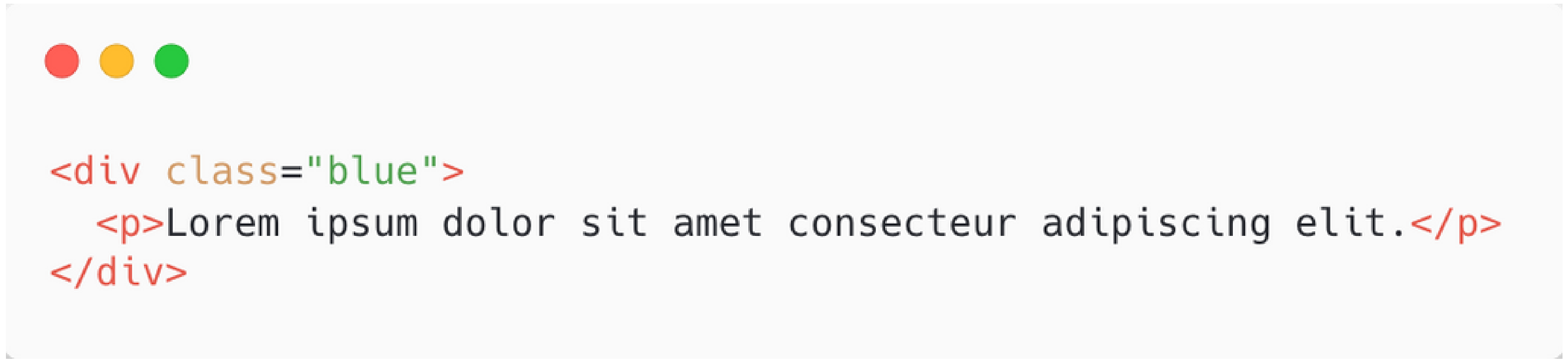
CSS



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

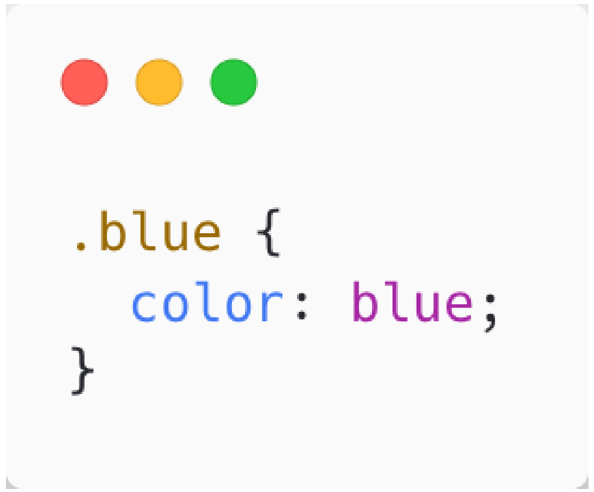
Lorem ipsum dolor sit amet consectetur adipisicing elit.

HTML



```html
<div class="blue">
  <p>Lorem ipsum dolor sit amet consecteur adipiscing elit.</p>
</div>
```

CSS



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

Lorem ipsum dolor sit amet consectetur adipisicing elit.

Inheritance

Occurs when an inheritable CSS property (i.e. color) is not set directly on an element, the parent chain is traversed until a value for that property is found.

HTML

```html
<div class="blue">
  <p>Lorem ipsum dolor sit amet consecteur adipiscing elit.</p>
</div>
```

CSS

```css
p {
  color: red;
}

.blue {
  color: blue;
}
```

Lorem ipsum dolor sit amet consectetur adipisicing elit.

HTML

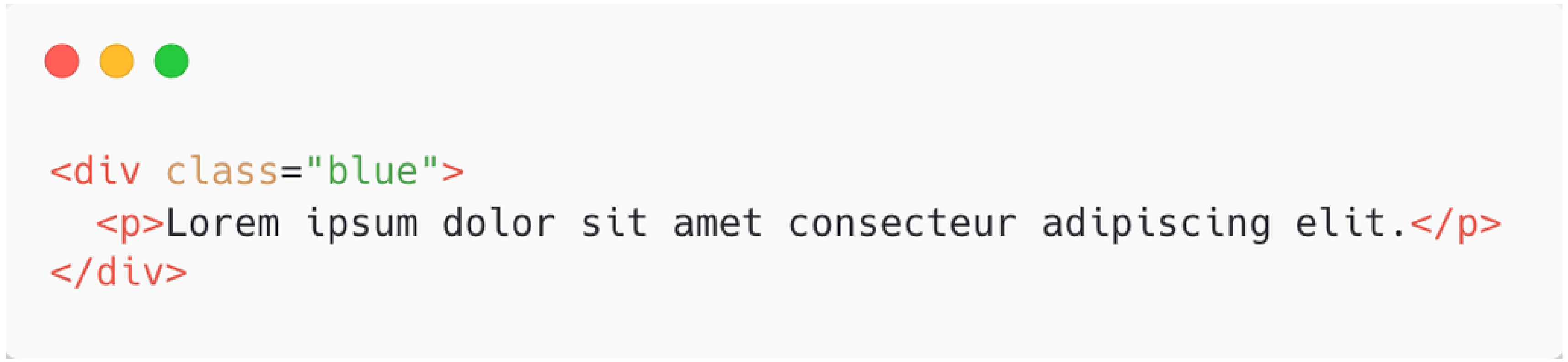
```html
<div class="blue">
  <p>Lorem ipsum dolor sit amet consecteur adipiscing elit.</p>
</div>
```

CSS

```css
p {
  color: red;
}

.blue {
  color: blue;
}
```

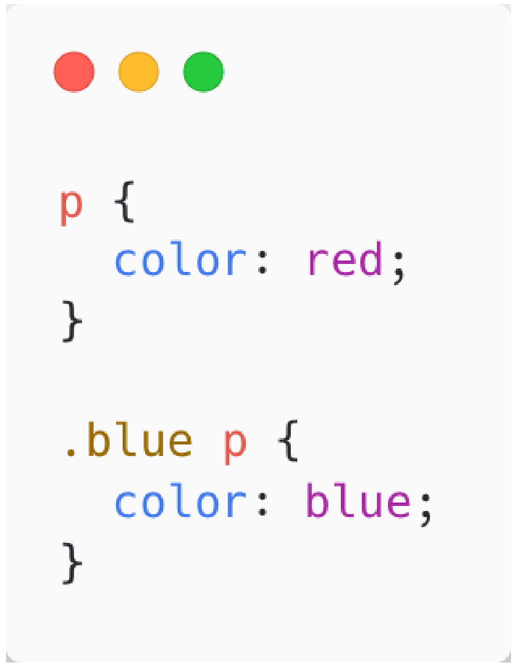
Lorem ipsum dolor sit amet consectetur adipisicing elit.

HTML



```html
<div class="blue">
  <p>Lorem ipsum dolor sit amet consecteur adipiscing elit.</p>
</div>
```

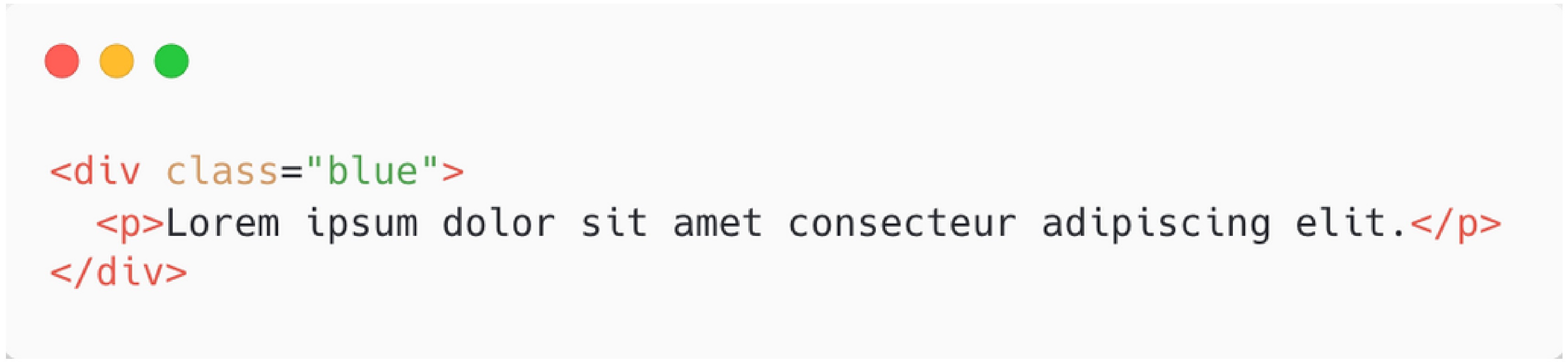
CSS



```css
p {
  color: red;
}

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

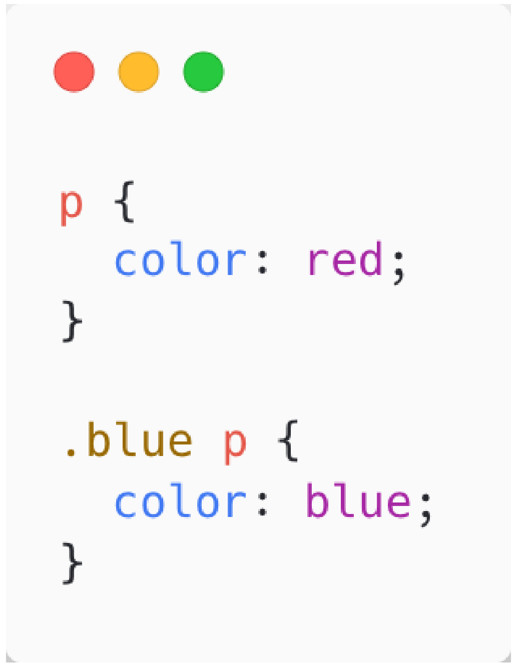
Lorem ipsum dolor sit amet consectetur adipisicing elit.

HTML



```html
<div class="blue">
  <p>Lorem ipsum dolor sit amet consecteur adipiscing elit.</p>
</div>
```

CSS



```css
p {
  color: red;
}

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

Lorem ipsum dolor sit amet consectetur adipisicing elit.

Specificity

The algorithm used by browsers to determine which CSS declaration should be applied.

Each selector has a calculated weight. The most specific weight wins.

Specificity

ID Selector:	1-0-0
Class Selector:	0-1-0
Type Selector:	0-0-1

ID	Class	Type

```
.body p{
   ...
}
```

```
.body p{
  ...
}
```

ID	Class	Type
	1	1

0-1-1

```
.body .text p{
  ...
}
```

```
.body .text p{
  ...
}
```

ID	Class	Type
	2	1

0-2-1

```
.body #title{
   ...
}
```

```
.body #title{
  ...
}
```

ID	Class	Type
1	1	

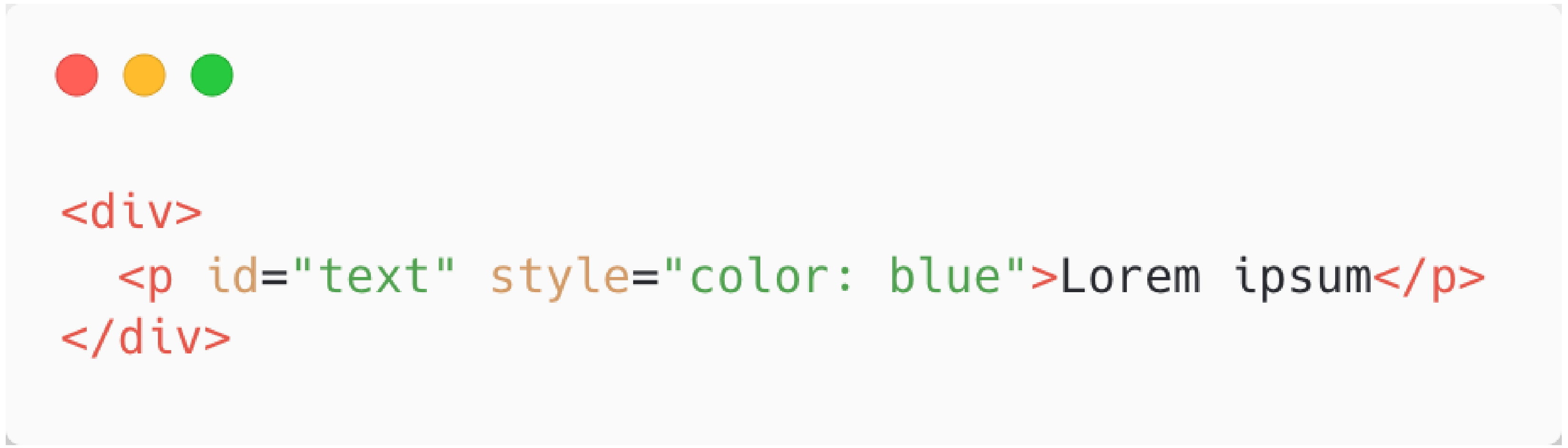
1-1-0

Inline Styles

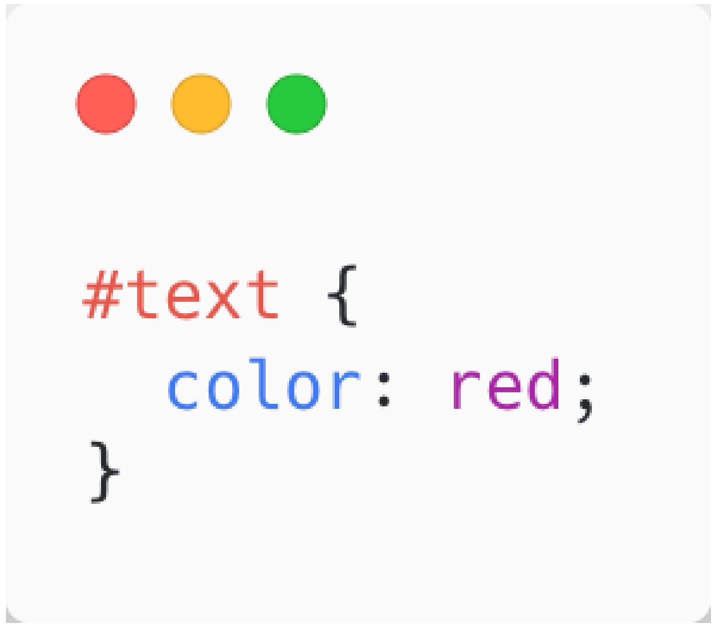
Inline styles have a higher specificity than ID selectors.

HTML

CSS



```html
<div>
  <p id="text" style="color: blue">Lorem ipsum</p>
</div>
```



```css
#text {
  color: red;
}
```

Lorem ipsum

!important

Marks a style rule as important; overrides all other styles.

HTML

```html
<div>
  <p id="text" style="color: blue">Lorem ipsum</p>
</div>
```

CSS

```css
#text {
  color: red !important;
}
```

Lorem ipsum

!important

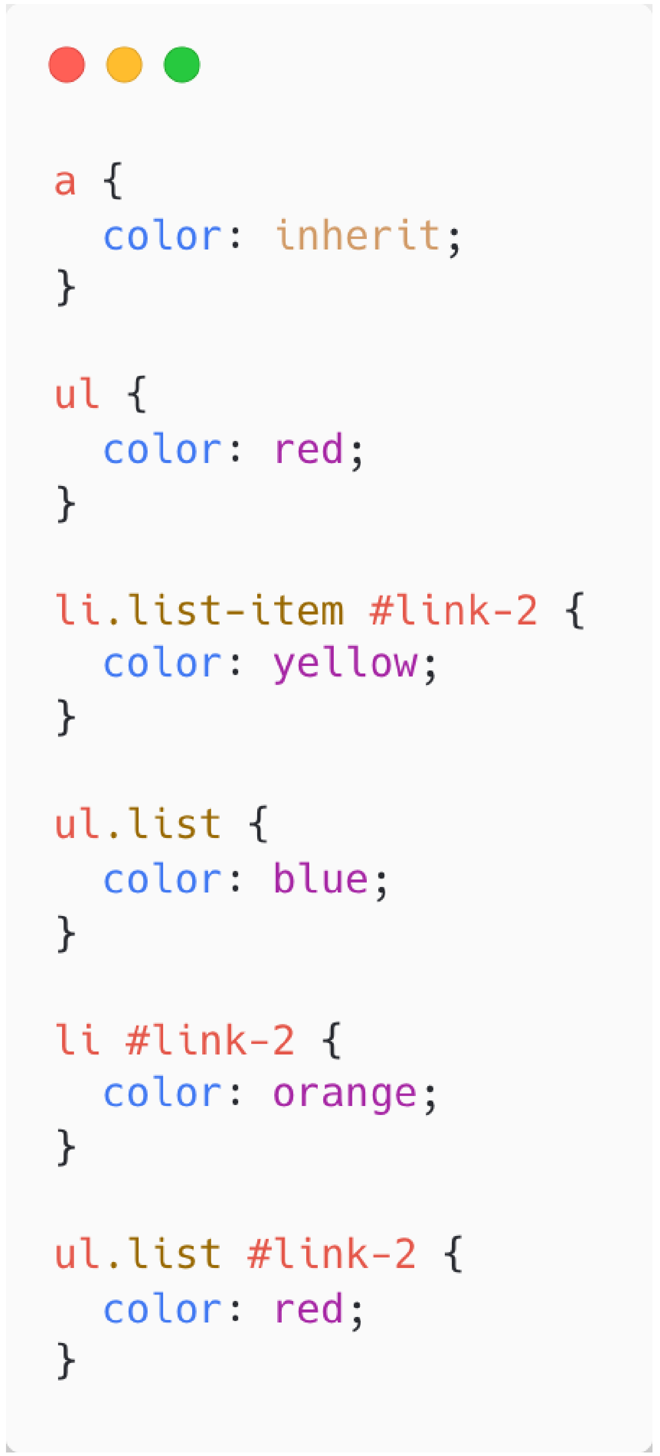
Not recommended practice

Specificity

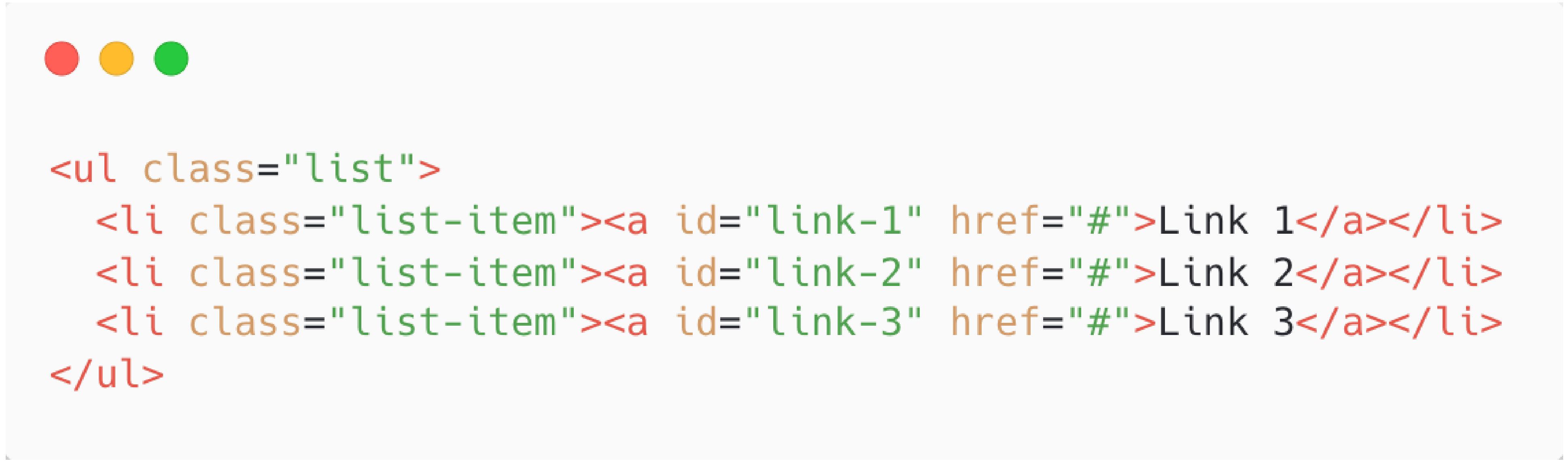
Specificity calculations come into play when multiple selectors are trying to style the same element.

Specificity

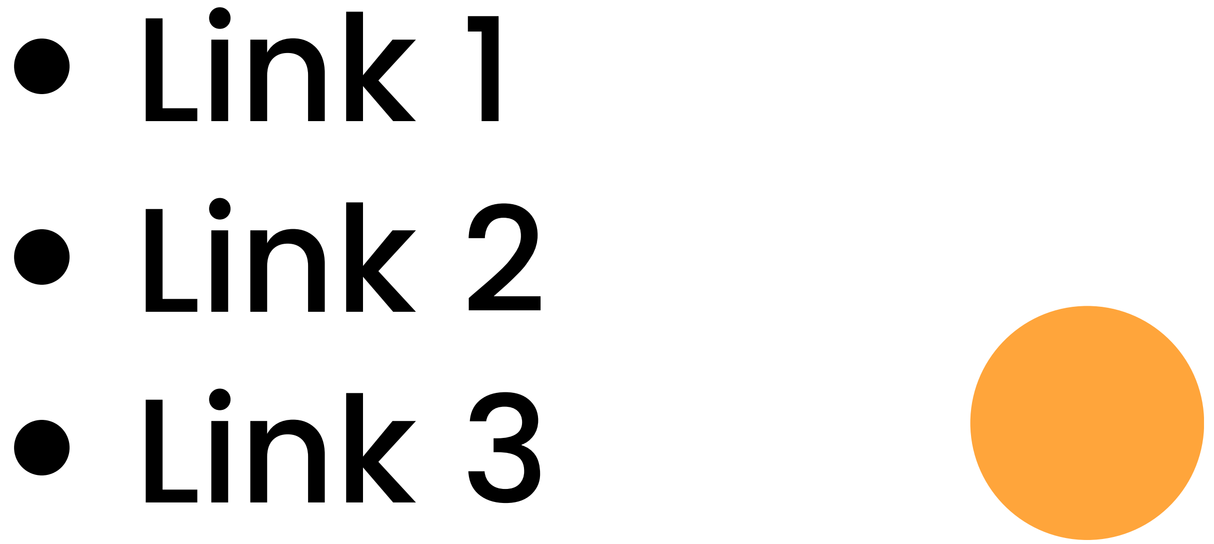
If there are two or more declarations providing different property values for the same element, the declaration with the most specific selector wins.



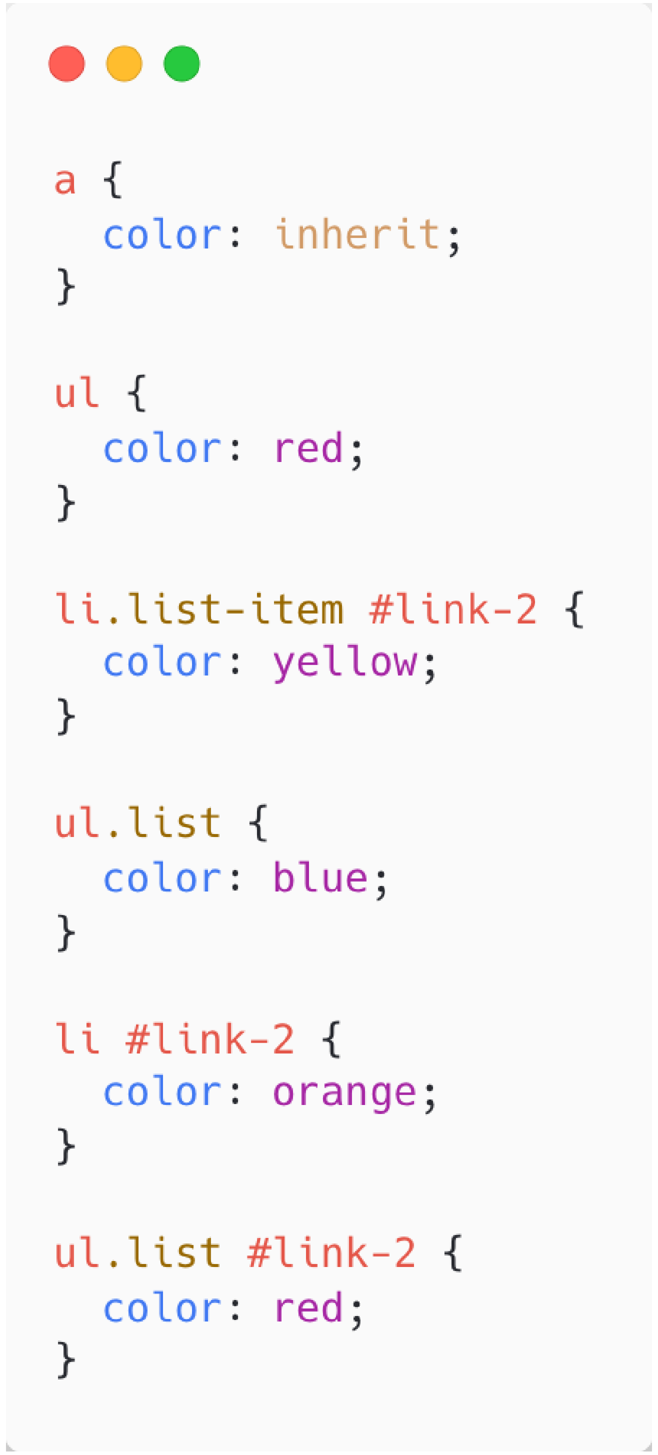
```css
a {
  color: inherit;
}

ul {
  color: red;
}

li.list-item #link-2 {
  color: yellow;
}

ul.list {
  color: blue;
}

li #link-2 {
  color: orange;
}

ul.list #link-2 {
  color: red;
}
```



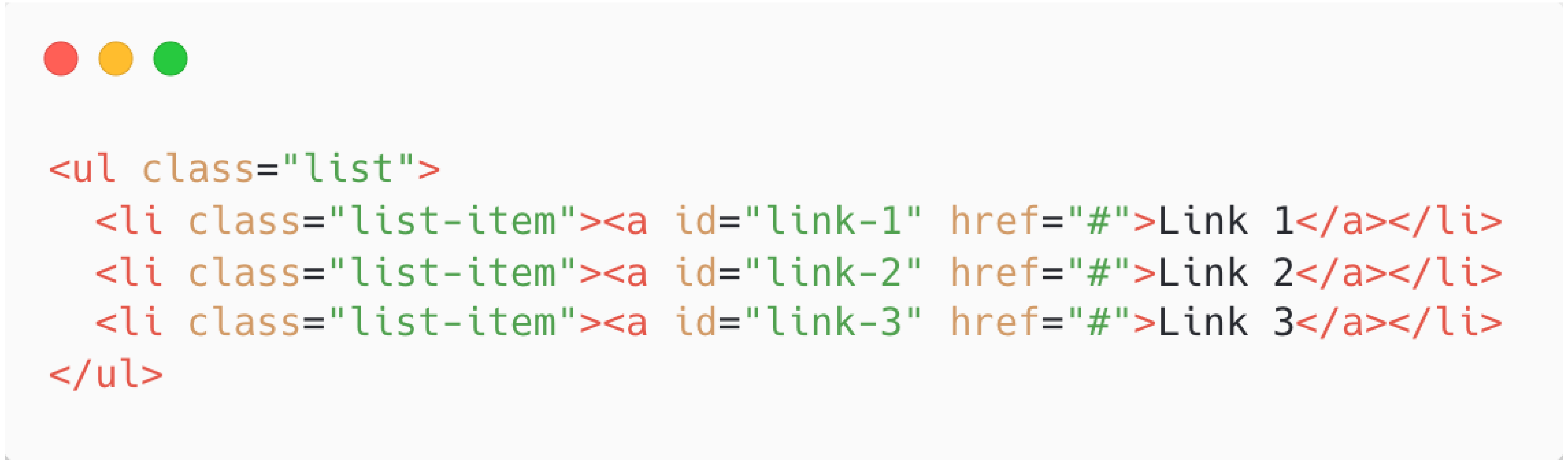
```html
<ul class="list">
  <li class="list-item"><a id="link-1" href="#">Link 1</a></li>
  <li class="list-item"><a id="link-2" href="#">Link 2</a></li>
  <li class="list-item"><a id="link-3" href="#">Link 3</a></li>
</ul>
```



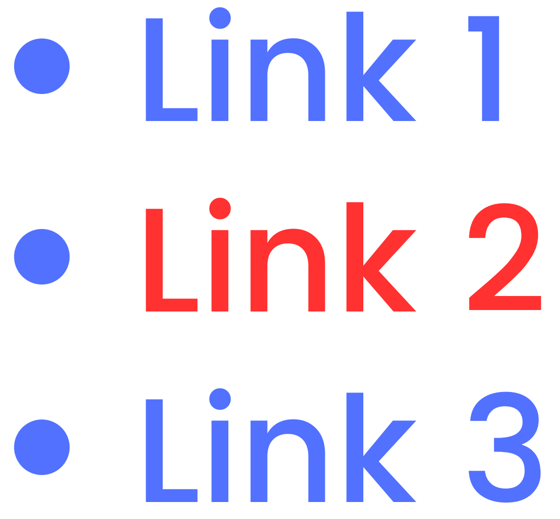
- Link 1
- Link 2
- Link 3



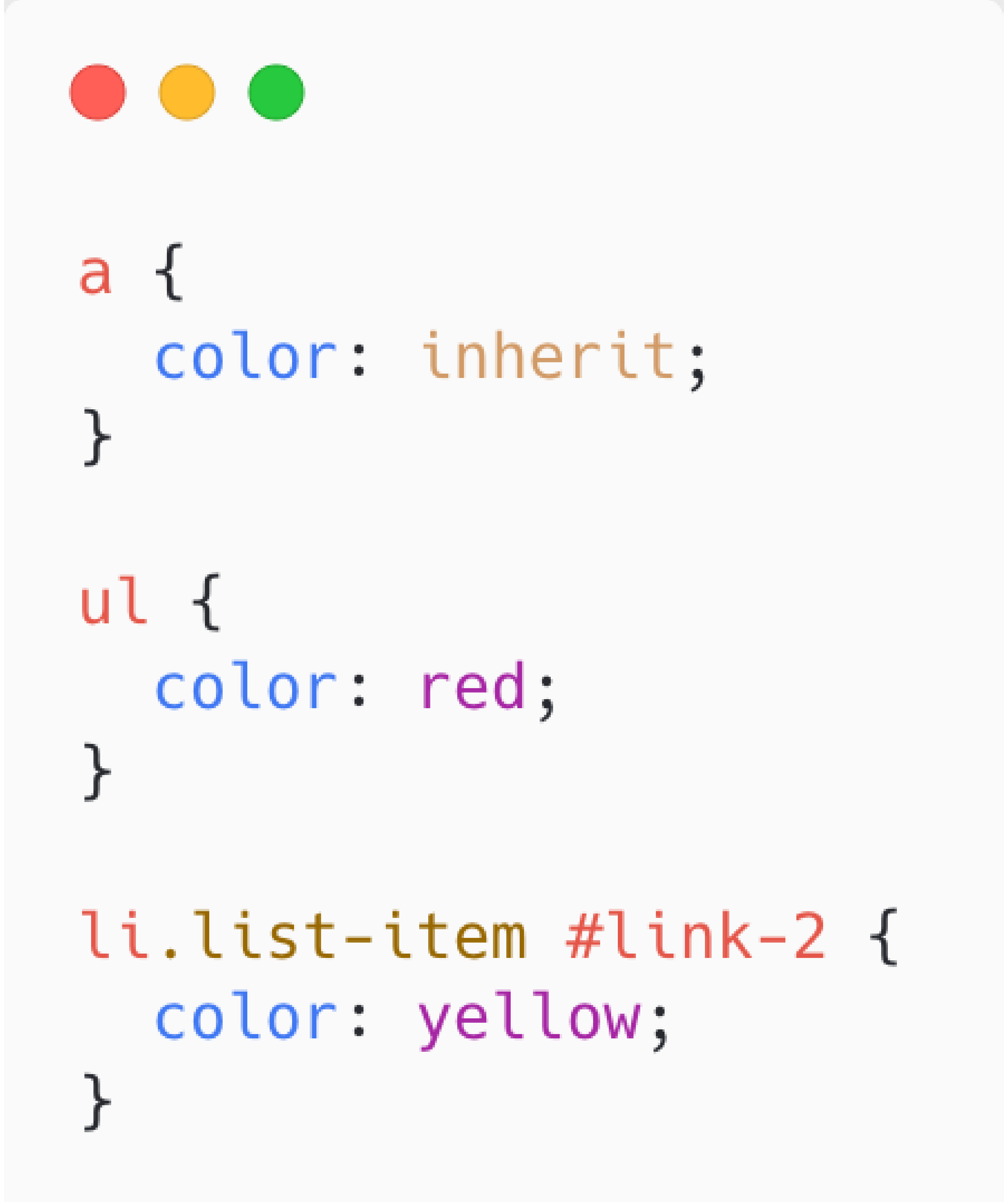
```css
a {
  color: inherit;
}

ul {
  color: red;
}

li.list-item #link-2 {
  color: yellow;
}

ul.list {
  color: blue;
}

li #link-2 {
  color: orange;
}

ul.list #link-2 {
  color: red;
}
```



```html
<ul class="list">
  <li class="list-item"><a id="link-1" href="#">Link 1</a></li>
  <li class="list-item"><a id="link-2" href="#">Link 2</a></li>
  <li class="list-item"><a id="link-3" href="#">Link 3</a></li>
</ul>
```



- Link 1
- Link 2
- Link 3



```css
a {
  color: inherit;
}

ul {
  color: red;
}

li.list-item #link-2 {
  color: yellow;
}
```

0-0-1

0-0-1

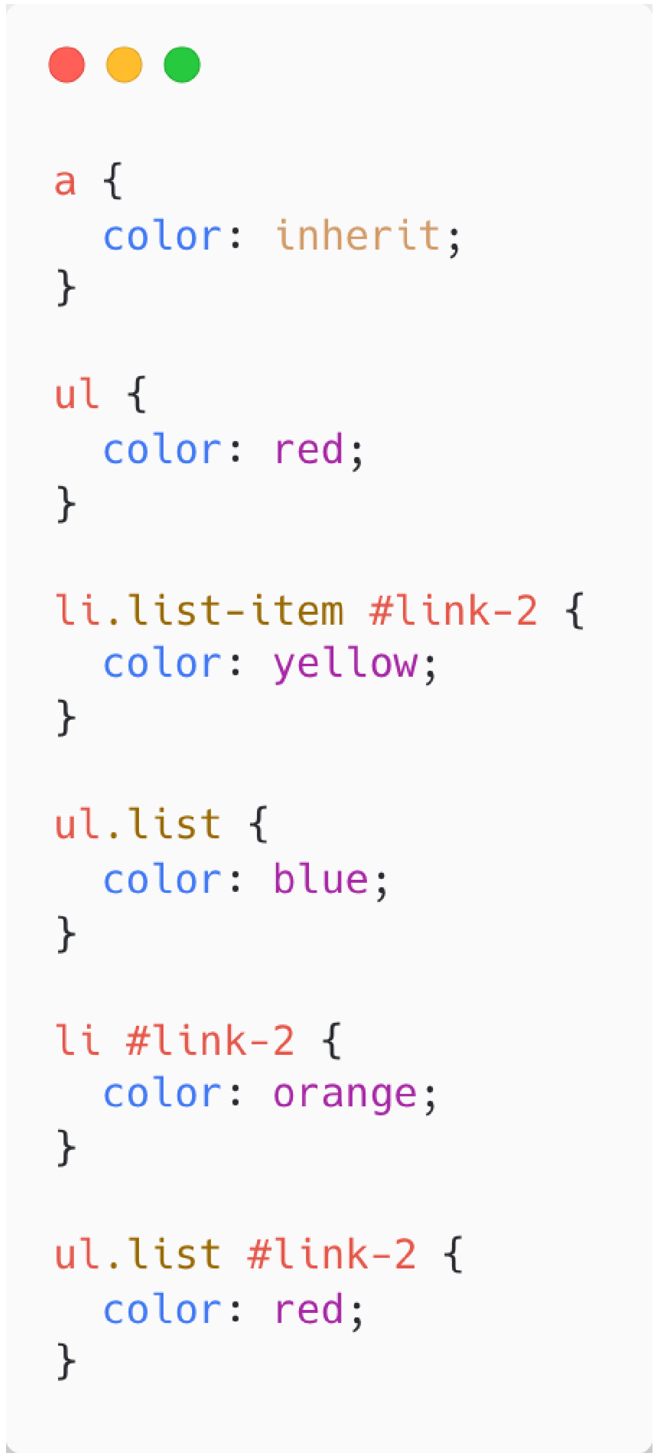
1-1-1

```
}

ul.list {
  color: blue;
}

li #link-2 {
  color: orange;
}

ul.list #link-2 {
  color: red;
}
```

0-1-1

1-0-1

1-1-1



0-0-1

```css
a {
  color: inherit;
}
```

0-0-1

```css
ul {
  color: red;
}
```

1-1-1

```css
li.list-item #link-2 {
  color: yellow;
}
```

0-1-1

```css
ul.list {
  color: blue;
}
```

1-0-1

```css
li #link-2 {
  color: orange;
}
```

1-1-1

```css
ul.list #link-2 {
  color: red;
}
```

- Link 1
- Link 2
- Link 3

0-0-1

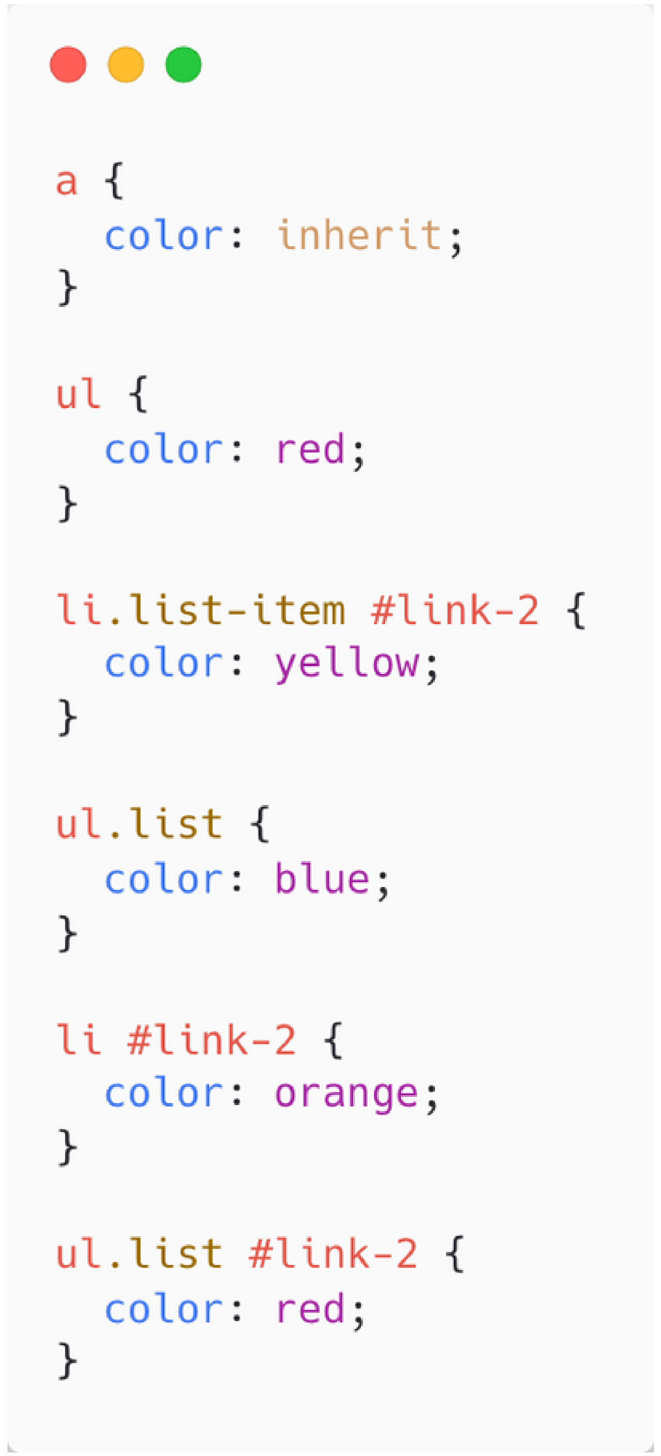
0-0-1

1-1-1

0-1-1

1-1-0

1-1-1



```css
a {
  color: inherit;
}

ul {
  color: red;
}

li.list-item #link-2 {
  color: yellow;
}

ul.list {
  color: blue;
}

li #link-2 {
  color: orange;
}

ul.list #link-2 {
  color: red;
}
```

- Link 1
- Link 2
- Link 3

0-0-1

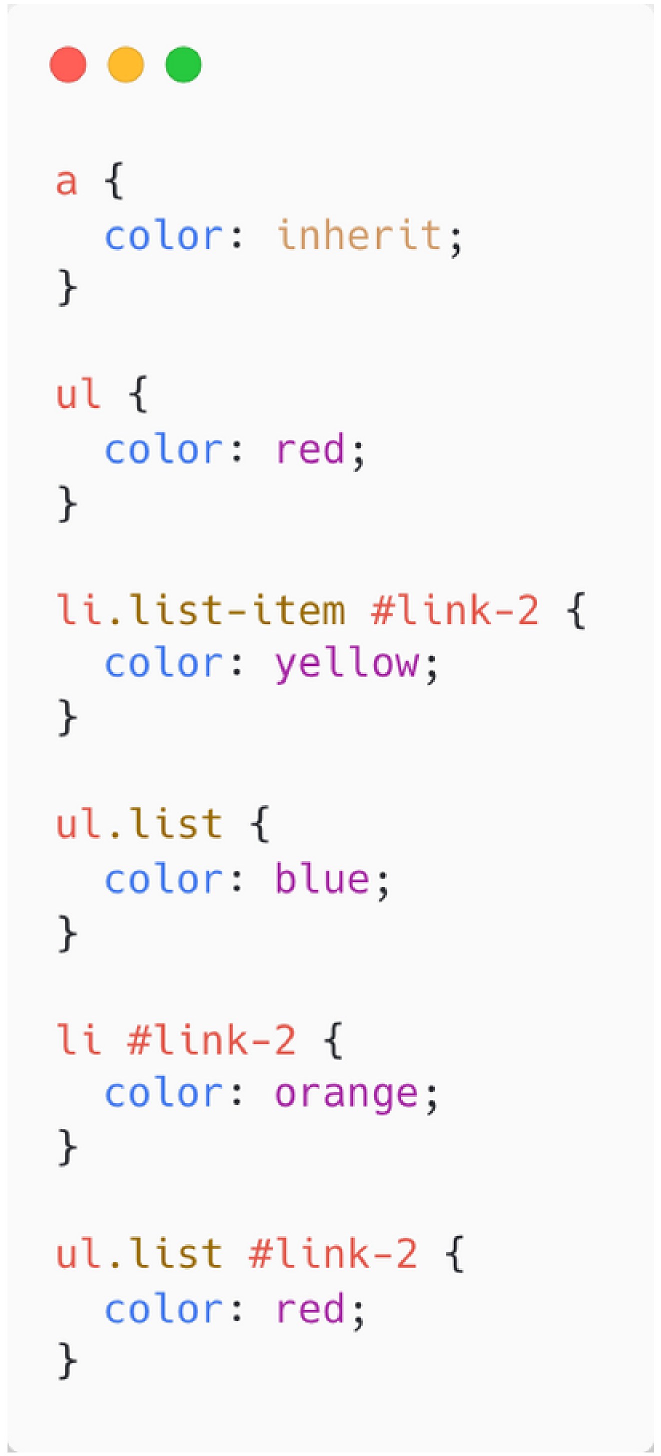
0-0-1

1-1-1

0-1-1

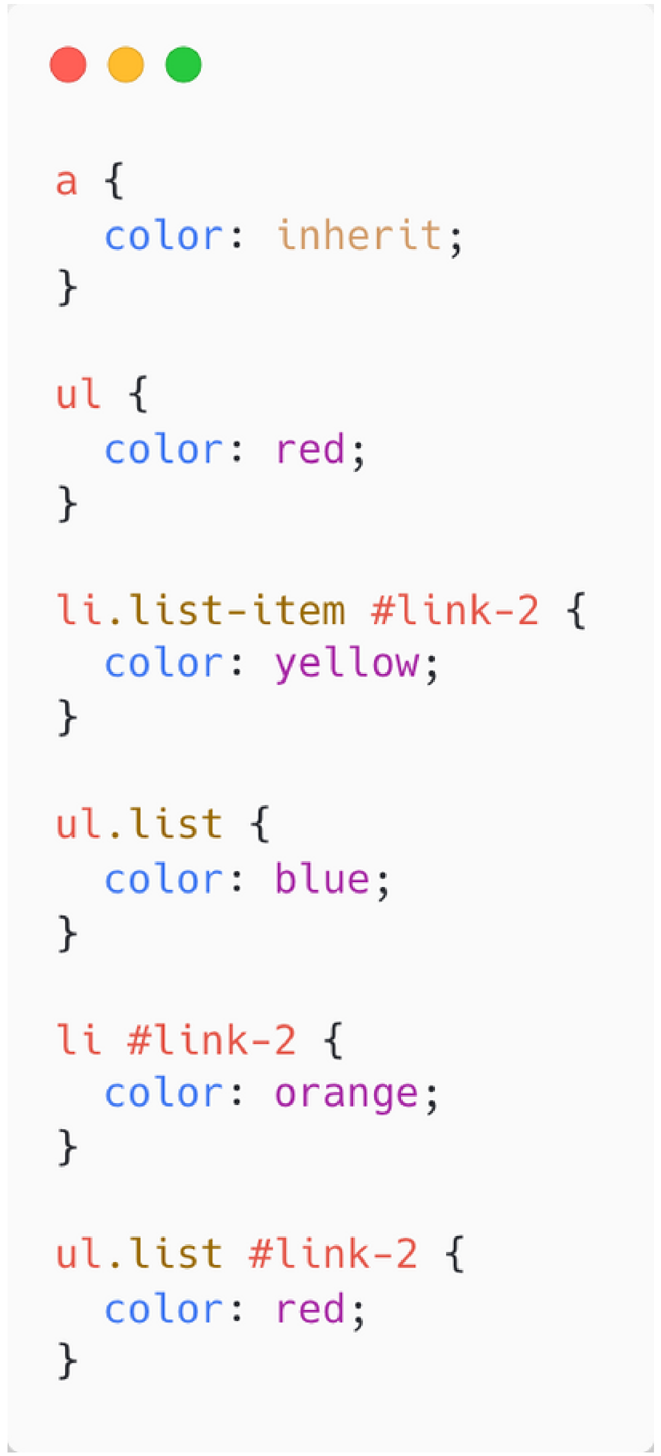
1-1-0

1-1-1



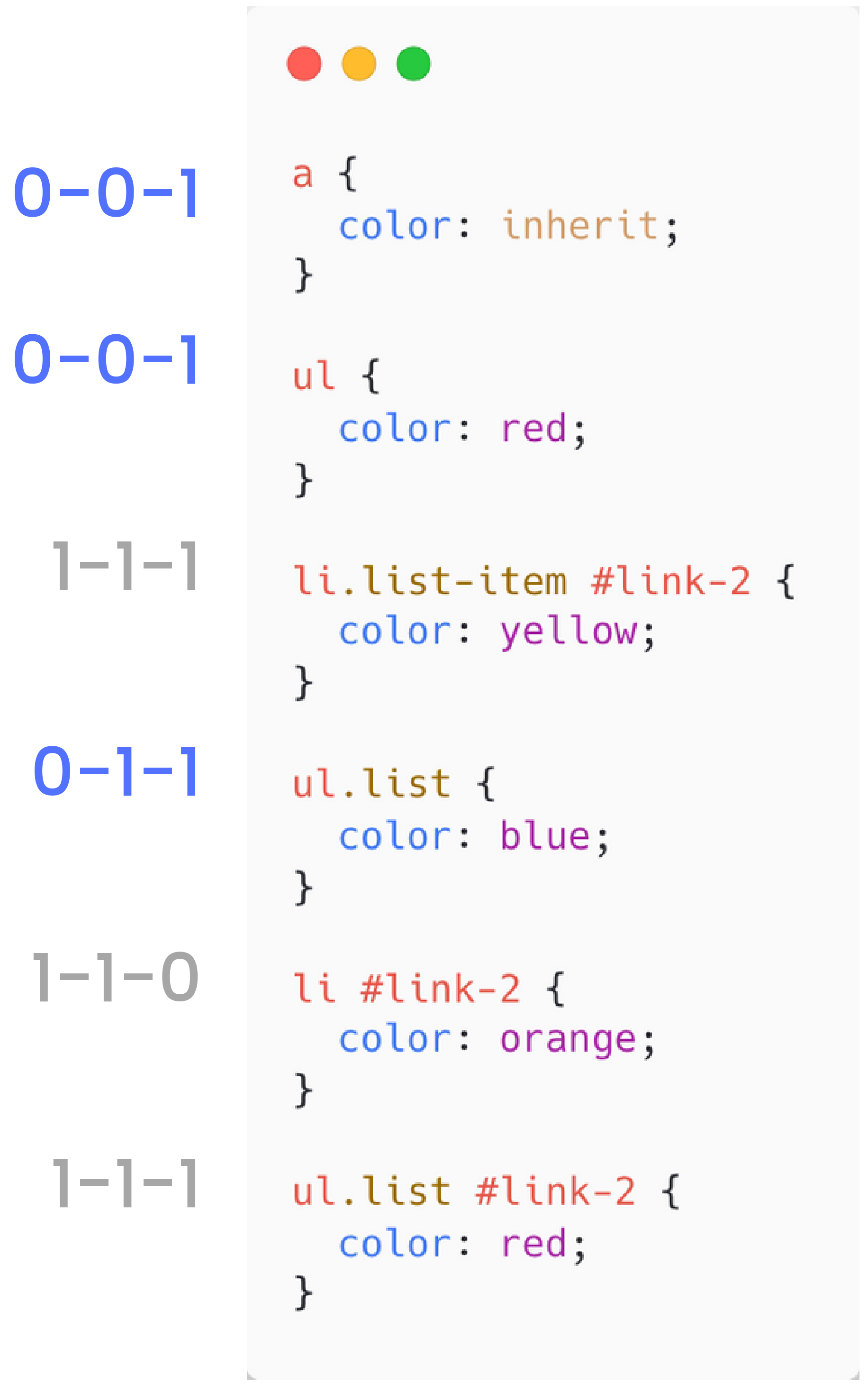
```css
a {
  color: inherit;
}

ul {
  color: red;
}

li.list-item #link-2 {
  color: yellow;
}

ul.list {
  color: blue;
}

li #link-2 {
  color: orange;
}

ul.list #link-2 {
  color: red;
}
```

- Link 1
- Link 2
- Link 3



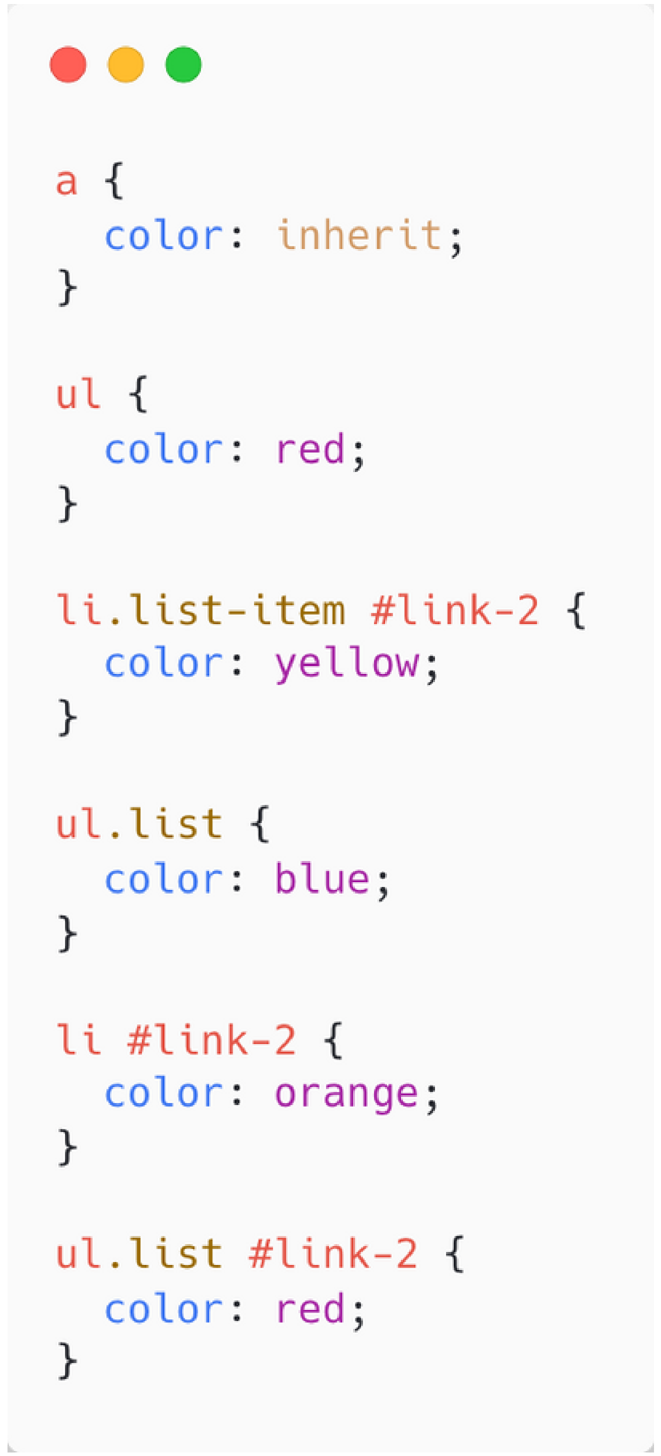
```
0-0-1   a {
          color: inherit;
        }

0-0-1   ul {
          color: red;
        }

1-1-1   li.list-item #link-2 {
          color: yellow;
        }

0-1-1   ul.list {
          color: blue;
        }

1-1-0   li #link-2 {
          color: orange;
        }

1-1-1   ul.list #link-2 {
          color: red;
        }
```

- Link 1
- Link 2
- Link 3



```
0-0-1    a {
           color: inherit;
         }

0-0-1    ul {
           color: red;
         }

1-1-1    li.list-item #link-2 {
           color: yellow;
         }

0-1-1    ul.list {
           color: blue;
         }

1-1-0    li #link-2 {
           color: orange;
         }

1-1-1    ul.list #link-2 {
           color: red;
         }
```

- Link 1
- Link 2
- Link 3



0-0-1

```
a {
  color: inherit;
}
```

0-0-1

```
ul {
  color: red;
}
```

1-1-1

```
li.list-item #link-2 {
  color: yellow;
}
```

0-1-1

```
ul.list {
  color: blue;
}
```

1-1-0

```
li #link-2 {
  color: orange;
}
```

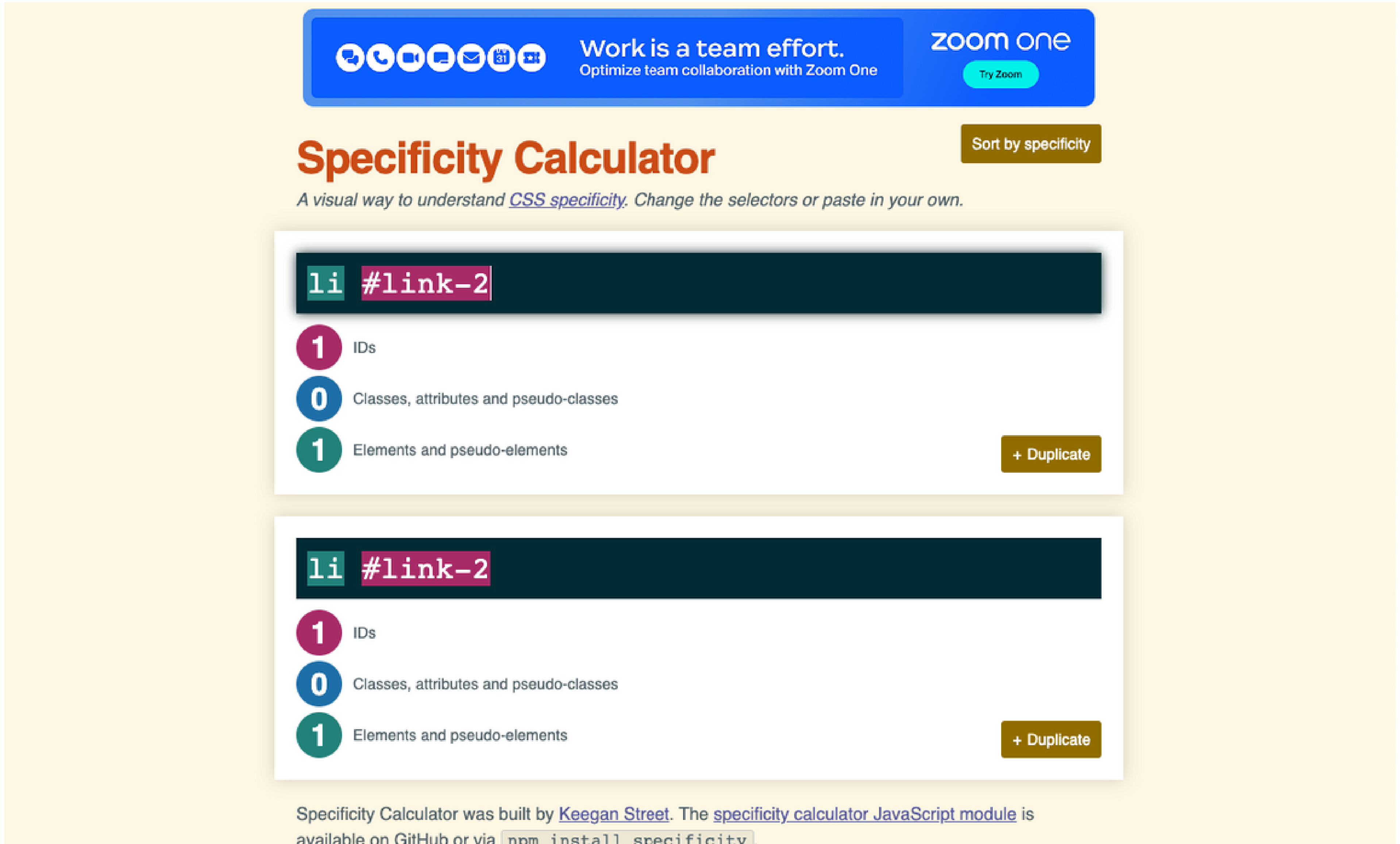
1-1-1

```
ul.list #link-2 {
  color: red;
}
```

- Link 1
- Link 2
- Link 3

Specificity Calculator

https://specificity.keegan.st/



Specificity Calculator

A visual way to understand CSS specificity. Change the selectors or paste in your own.

```
li #link-2
```

1 IDs

0 Classes, attributes and pseudo-classes

1 Elements and pseudo-elements

```
li #link-2
```

1 IDs

0 Classes, attributes and pseudo-classes

1 Elements and pseudo-elements

Specificity Calculator was built by Keegan Street. The specificity calculator JavaScript module is available on GitHub or via `npm install specificity`

What we've learned

- History of CSS
- Elements
- Selectors (replaced, non-replaced, block, inline)
- Specificity
- Inheritance

Next Up

We'll begin building our project and learning some new skills along the way.