

CSS: Cascading Style Sheets

Lecture 10

Evolution of CSS

- MIME type: text/css
- CSS 1 ('96): early recognition of value
- CSS 2 ('98): improvements in language
 - Adding media types (screen vs print)
 - Inconsistent support by browsers
- CSS 2.1 ('11)
 - In practice since '04
 - Took forever to standardize
- CSS 3 (informal name)
 - Breaks standard into many (50?) modules
 - Modules developed, adopted independently
 - <https://www.w3.org/Style/CSS/current-work>
 - <https://caniuse.com/?cats=CSS>

Key Idea

- Separate content and style
 - Different languages (syntax): HTML vs CSS
 - Different documents
- Goal: Single-point-of-control-over-change
 - Change font of every word in paragraph?
 - Change font of every `` element in document?
 - Change font of every `` element in every document on a site?
 - Change font of every `` element which is part of instructions, but not finalized, on site?

CSS Syntax

- CSS is *declarative* (not *procedural*)
 - Describe a thing, not how to do compute it
 - Example: RE matching
- CSS = list of *rules* (order can matter)
- Rule = a **location** & the **style** to use there
- Basic syntax of a rule

```
selector {  
    property1: style1;  
    property2: style2;  
    . . .  
}    /* comments always help */
```

Example CSS

```
h2 {  
  /* draconian OSU visual identity */  
  color: darkred;  
  background: gray;  
  /* additional gratuitous styling */  
  font-style: italic;  
}
```

Many Available Properties

- Background
 - `background-color`, `background-image`
- Text, font
 - `line-height`, `text-align`, `color`
 - `font-family`, `font-style`, `font-size`
- Border, margin, padding
 - `border-left-width`, `border-bottom-color`
- Positioning
 - `clear`, `display`, `float`
- Dimension
- List, table
 - `list-style-type`
 - `border-collapse`, `caption-side`
- Generated content and other fancy stuff
- See: developer.mozilla.org/Web/CSS/Reference

Shorthand Properties

- Example: Margins have 4 sides

```
margin-top: 3px;  
margin-right: 5px;  
margin-bottom: 7px;  
margin-left: 9px;
```

- Shorthand property: margin

```
margin: 3px 5px 7px 9px; /* TRBL */  
margin: 7px 9px; /* TB sides */  
margin: 2px 6px 8px; /* T sides B */
```

- Mnemonic: always "TRouBLE"

- Missing values filled in with provided value(s)

- Other shorthand properties:

- Padding, border-width, font, border, background...

Including CSS: Mechanics

- Embed directly in element

```
<p style="color: red; background: gray">
```

- Place in style element in head

```
<head>
```

```
  <style media="screen">
```

```
    p {color: red; background: gray;}
```

```
  </style>
```

```
</head>
```

- Link to separate CSS file in head

```
<head>
```

```
  <link rel="stylesheet"
```

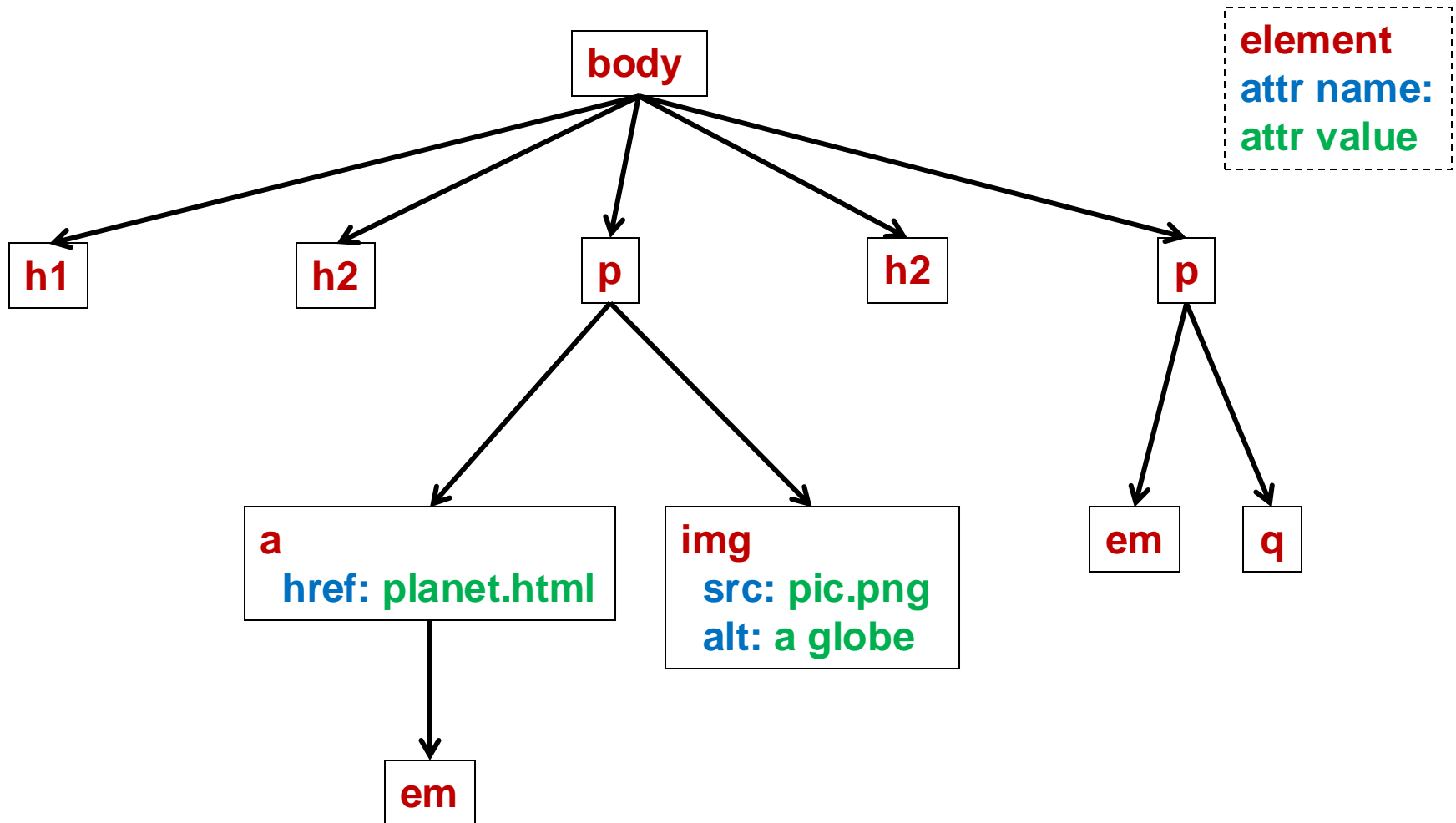
```
    href="3901Style.css" media="screen" />
```

```
</head>
```

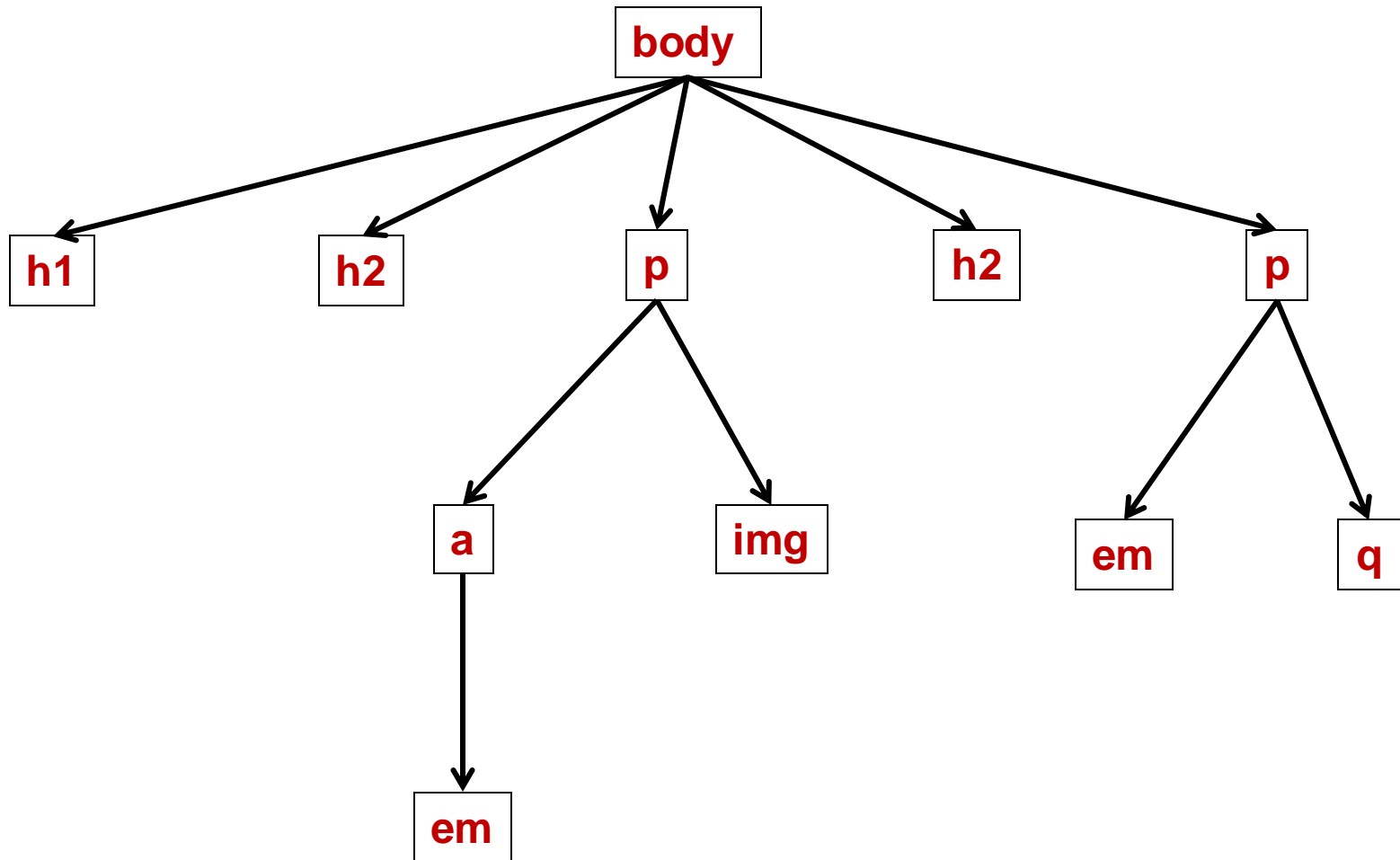

Example CSS

```
h2 {  
    color: darkred;  
    background: gray;  
    font-style: italic;  
}  
  
em {  
    font-style: normal;  
    font-weight: bold;  
}
```

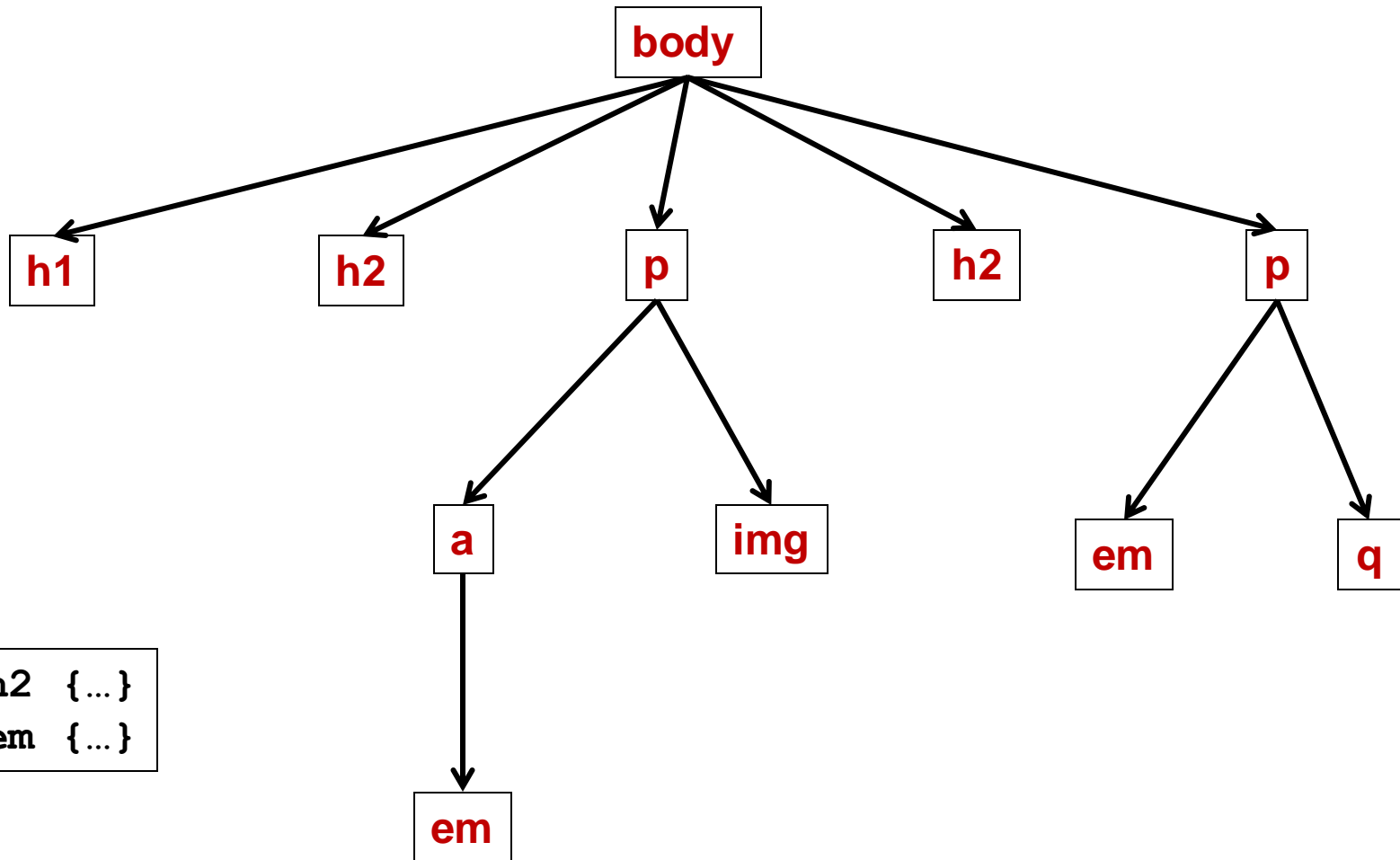
Tree (Rooted at Body)



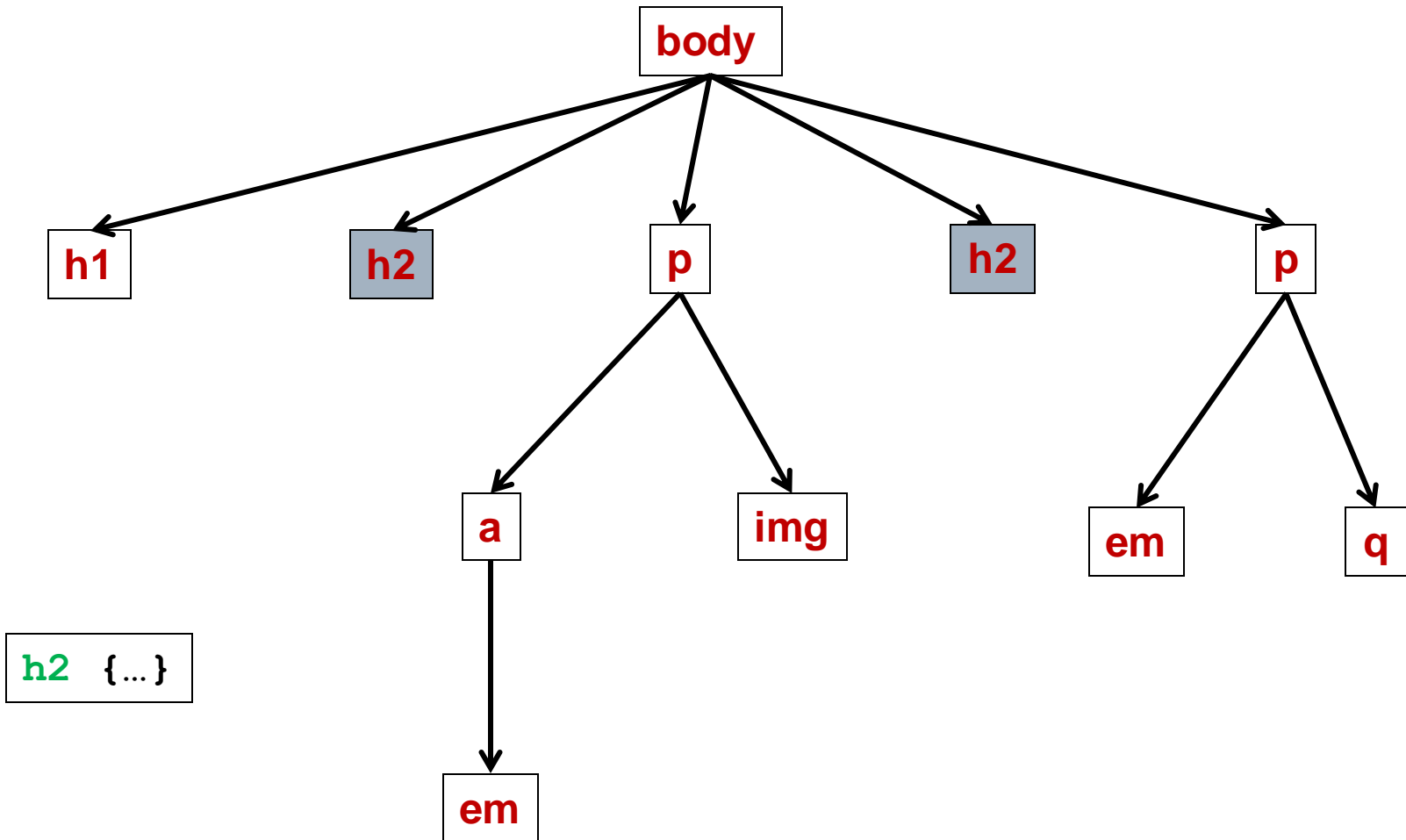
Tree (sans Attributes)



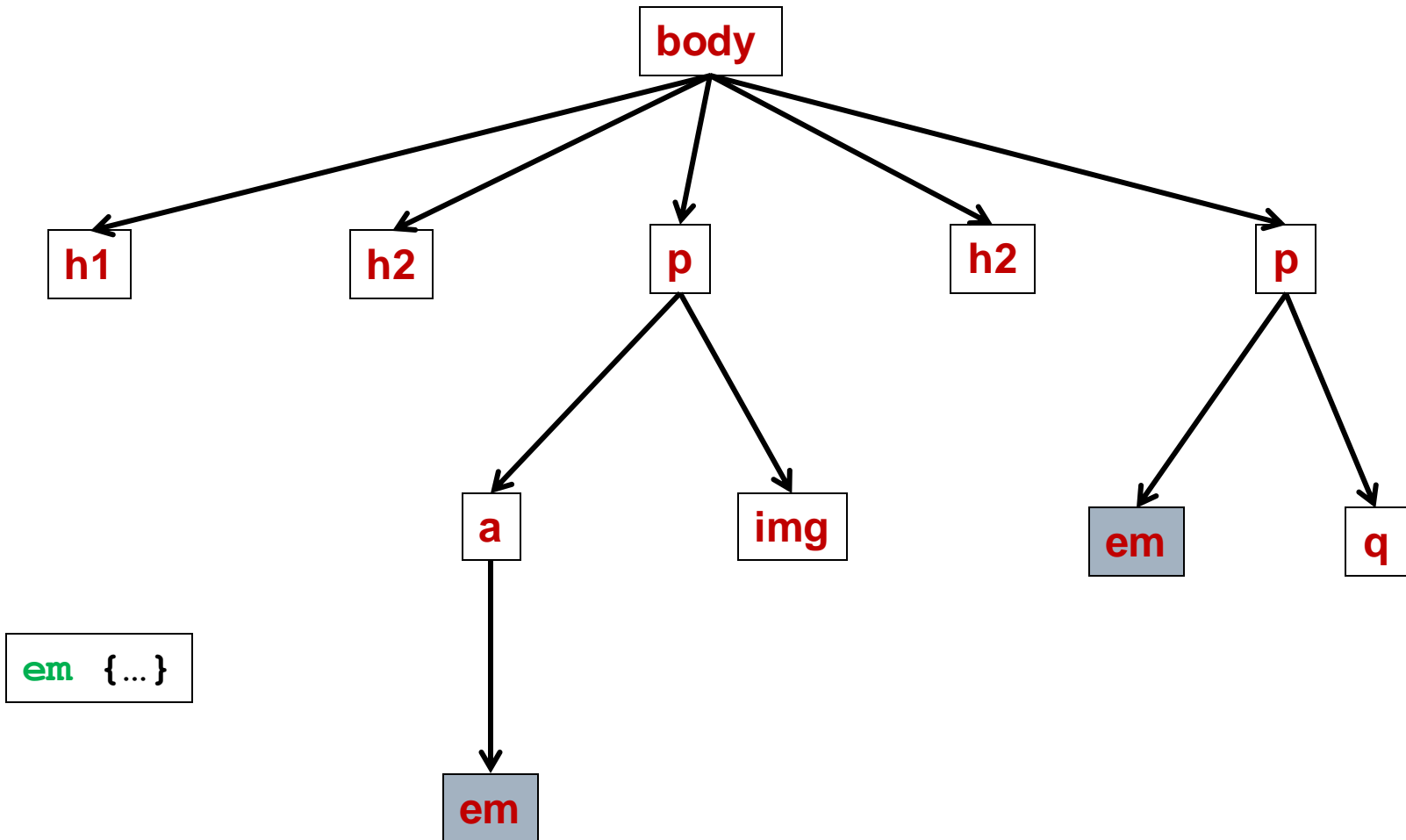
Tree (sans Attributes)



Selectors Applied to Tree



Selectors Applied to Tree



Multiple Selectors

```
h1 {  
    color: darkred;  
    background: gray;  
    font-style: italic;  
    border-bottom-style: solid;  
}  
h2 {  
    color: darkred;  
    background: gray;  
    font-style: italic;  
}
```

Multiple Selectors: SPOCOC

```
h1, h2 {  
    color: darkred;  
    background: gray;  
    font-style: italic;  
}
```

```
h1 {  
    border-bottom-style: solid;  
}
```


Inheritance for SPOCOC

- A child inherits many properties from parent by default
 - Font weight, color, family, etc
 - Can be overridden in child
- Set global styles in root

```
body {  
    font-family: sans-serif;  
}
```

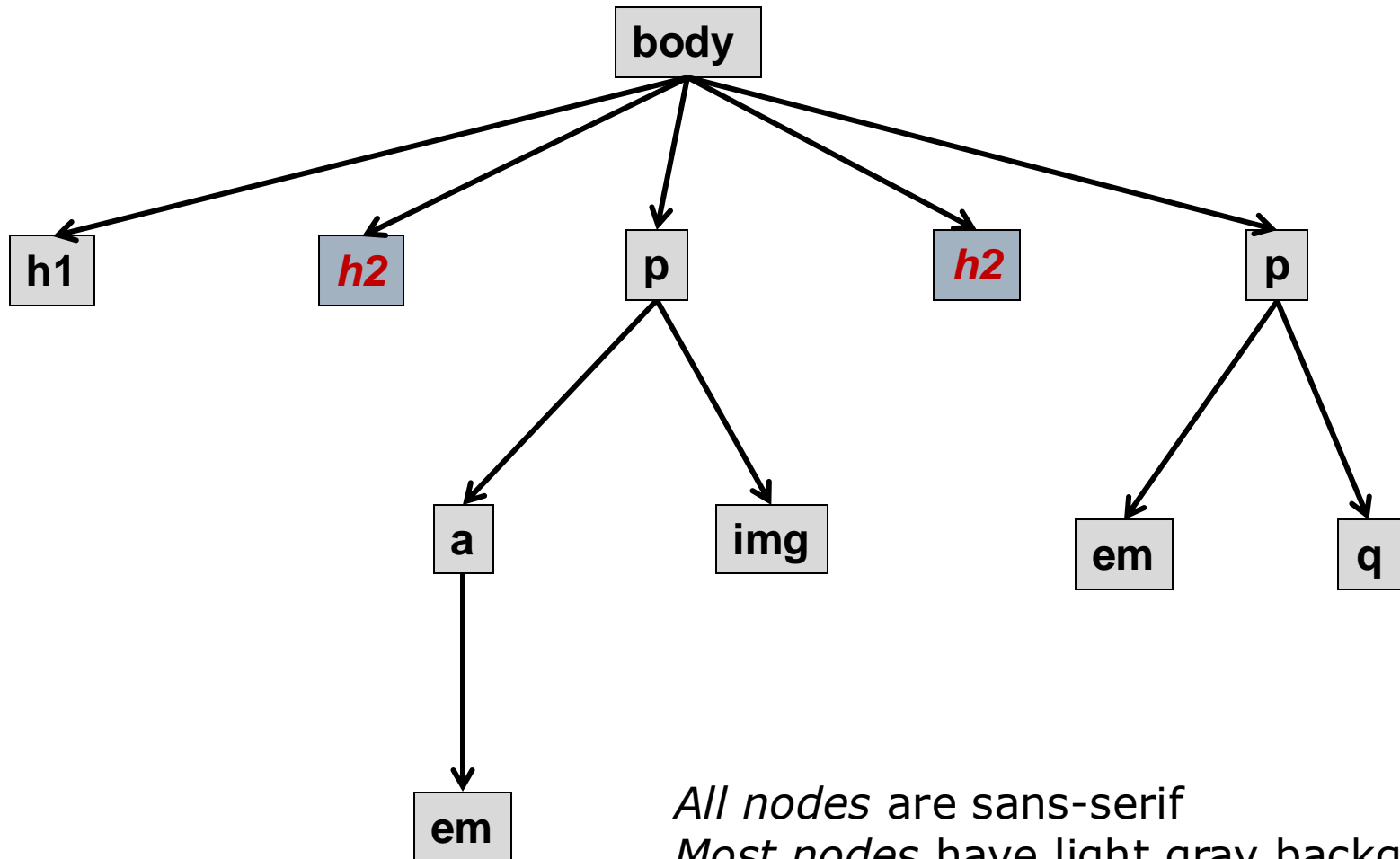
 - Contrast this with having to set property in all possible elements!
- Generally, text properties (eg color) are inherited, box-related (eg border) are not

Example Inheritance

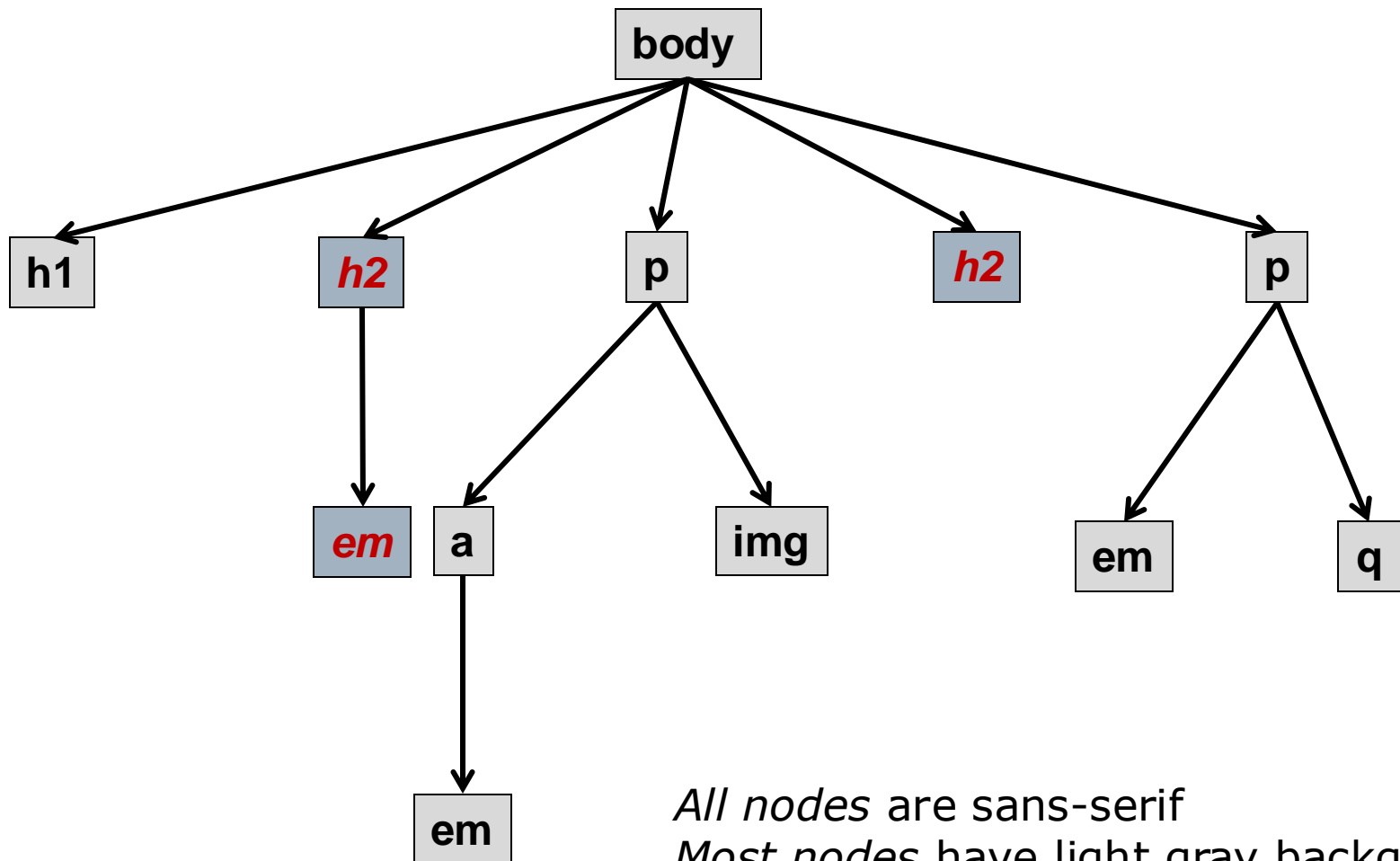
```
body {  
    font-family: sans-serif;  
    background: lightgray;  
}
```

```
h2 {  
    /* inherits font-family, backgrd */  
    color: darkred;  
    background: gray; /* new backgrd */  
    font-style: italic;  
}
```

Inherited Properties



Inherited Properties



Demo: Chrome Dev. Tools

The image shows a browser window with a CodePen page titled "CSS example" and the Chrome DevTools interface overlaid. The browser address bar shows "cdpn.io/pen/debug/eVdMXR?authentication_h...". The page content includes a red "CSS example" header, a blue "h2" element with dimensions "669 x 28", and a section titled "Things to Note".

DevTools is open to the "Elements" tab, showing the following HTML structure:

```
<!DOCTYPE html>
<html lang="en">
  <head>...</head>
  <body translate="no">
    <h1> CSS example</h1>
    <p>This page is full of goodness!</p>
    <h2>Things to Note</h2>
    ...
  </body>
</html>
```

The "Computed" tab is active, displaying a box model diagram with the following values:

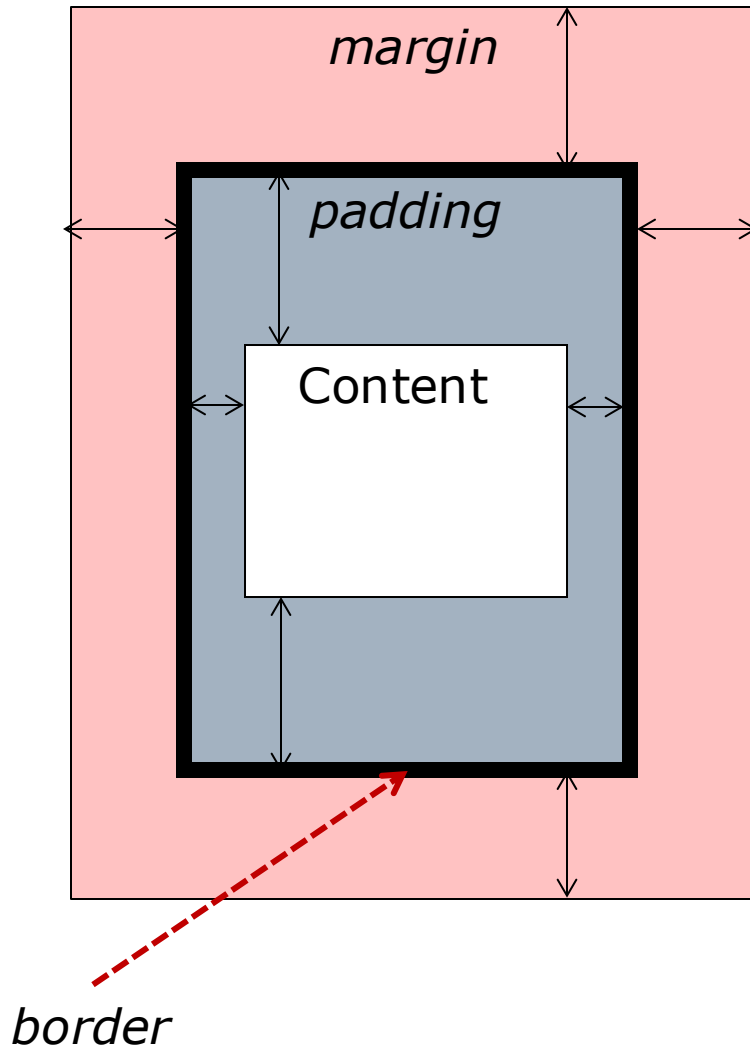
- margin: 19.920
- border: -
- padding: -
- 669x28 (content dimensions)
- 19.920 (bottom margin)

Below the diagram, the "Filter" section shows the following computed styles:

- display: block
- font-family: sans-serif
- font-size: 24px
- font-weight: 700
- height: 28px

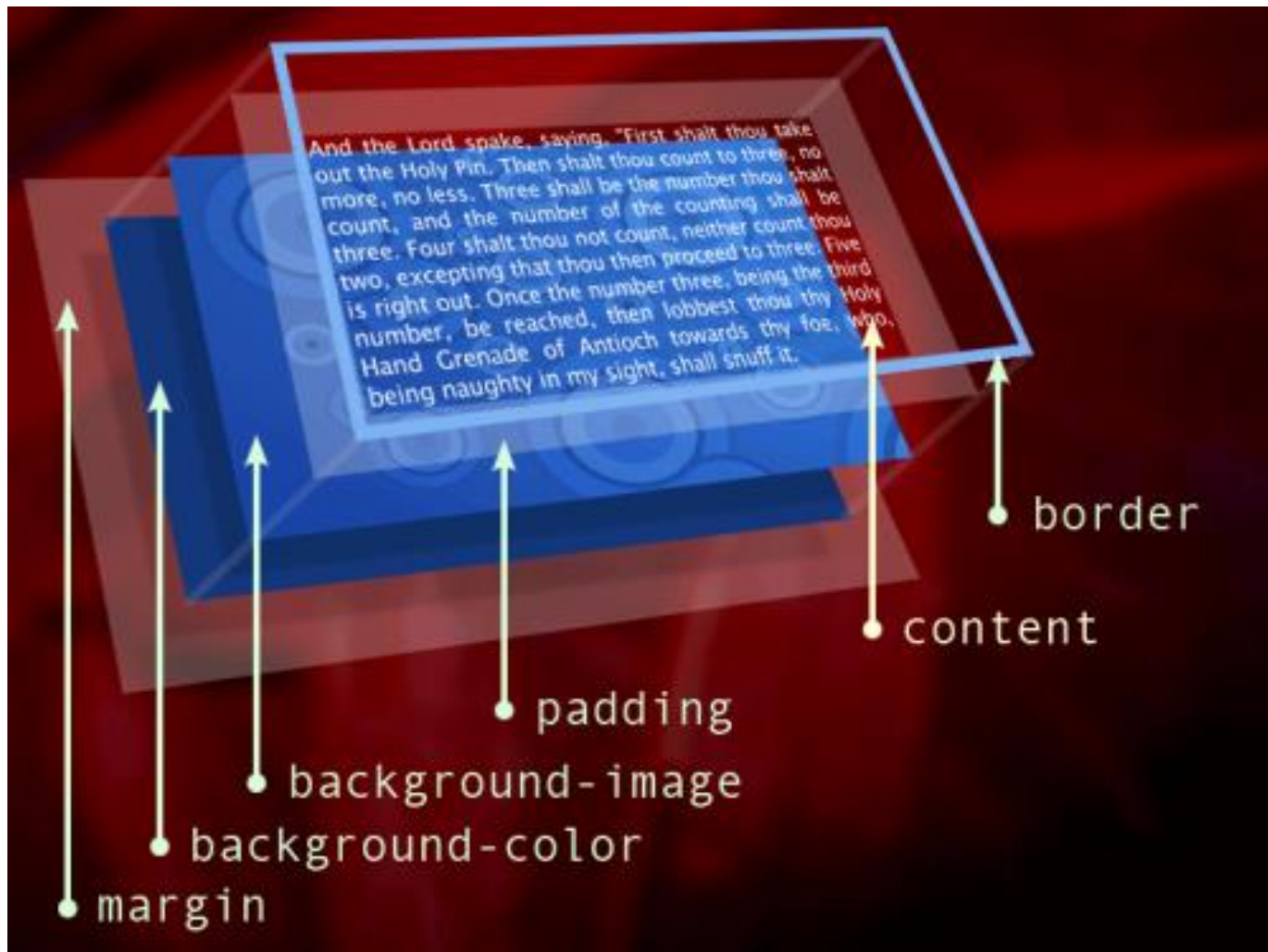
Red arrows point from the "h2" element in the browser to the "h2" tag in the Elements panel, and from the "h2" tag to the "html", "body", and "h2" breadcrumb at the bottom of the Elements panel. The "Elements" and "Computed" tabs in DevTools are also circled in red.

Layout: The Box Model

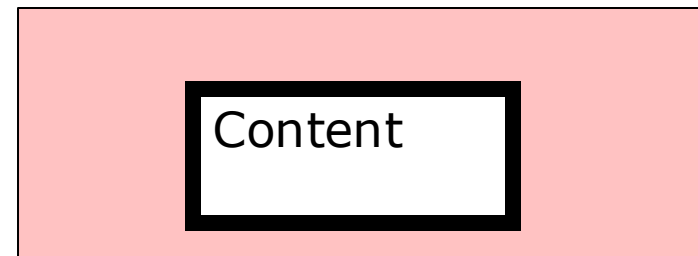
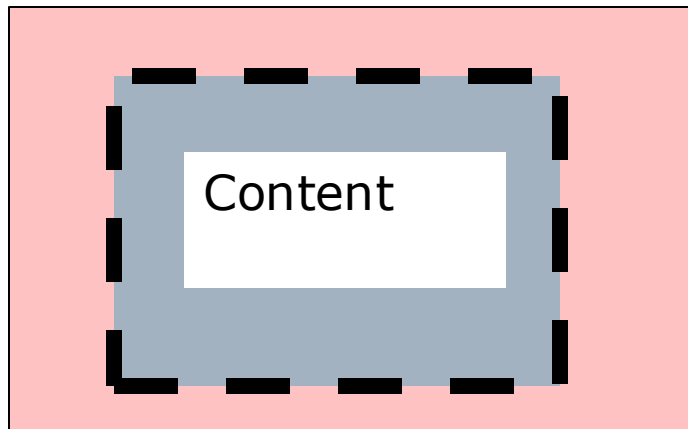
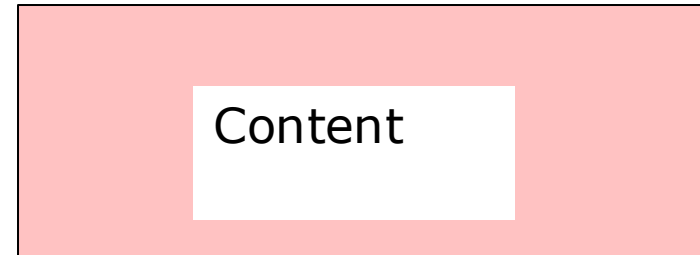
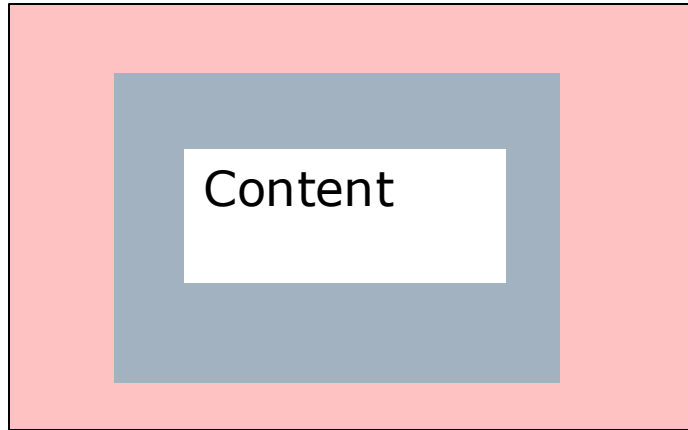


- Both block & inline
 - Minor differences
- Border appearance
 - Style, width, color, radius
- Margins & padding
 - Transparent
 - 4 independent sides
- Padding is *part* of it
 - *Content* background shows through
- Margins gives space
 - Some adjacent margins "collapse"

The Box Model As Layers

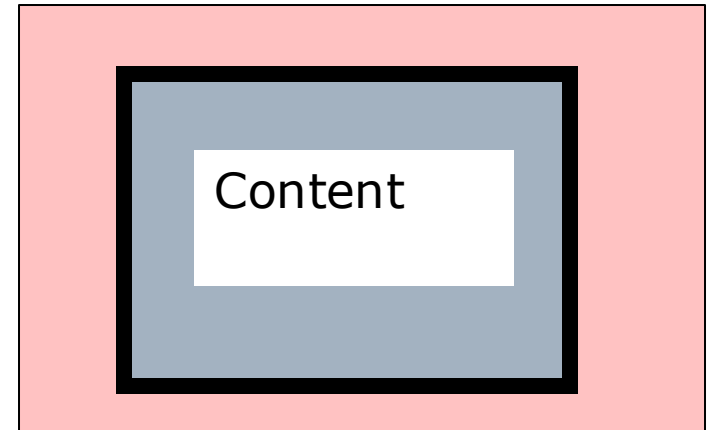


Examples



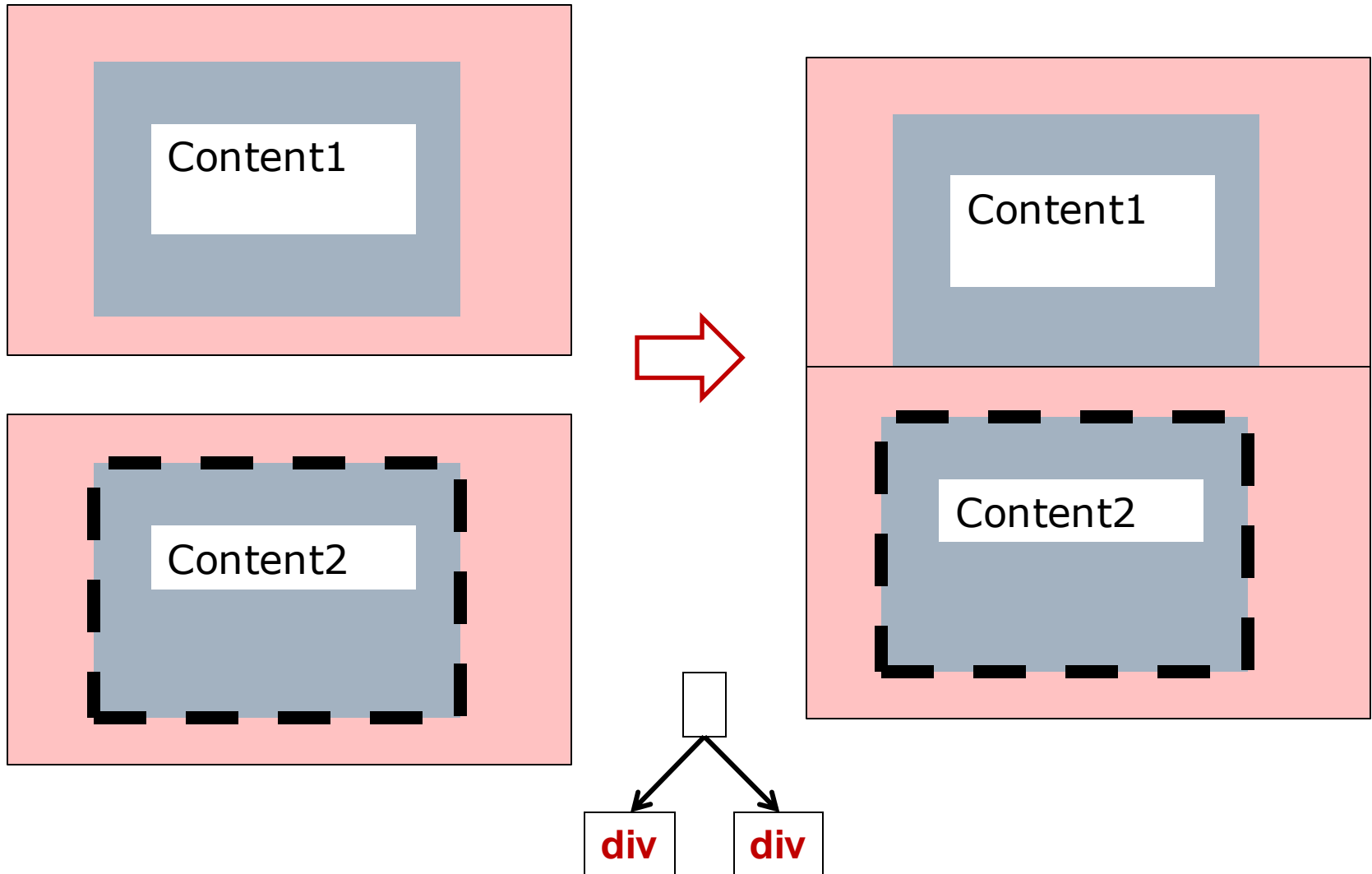
Box Sizing

```
p {  
  margin: 10px 100px 10px 10px;  
  border-width: 5px 1px 5px;  
  width: 200px;  
  padding: 2px;  
}
```

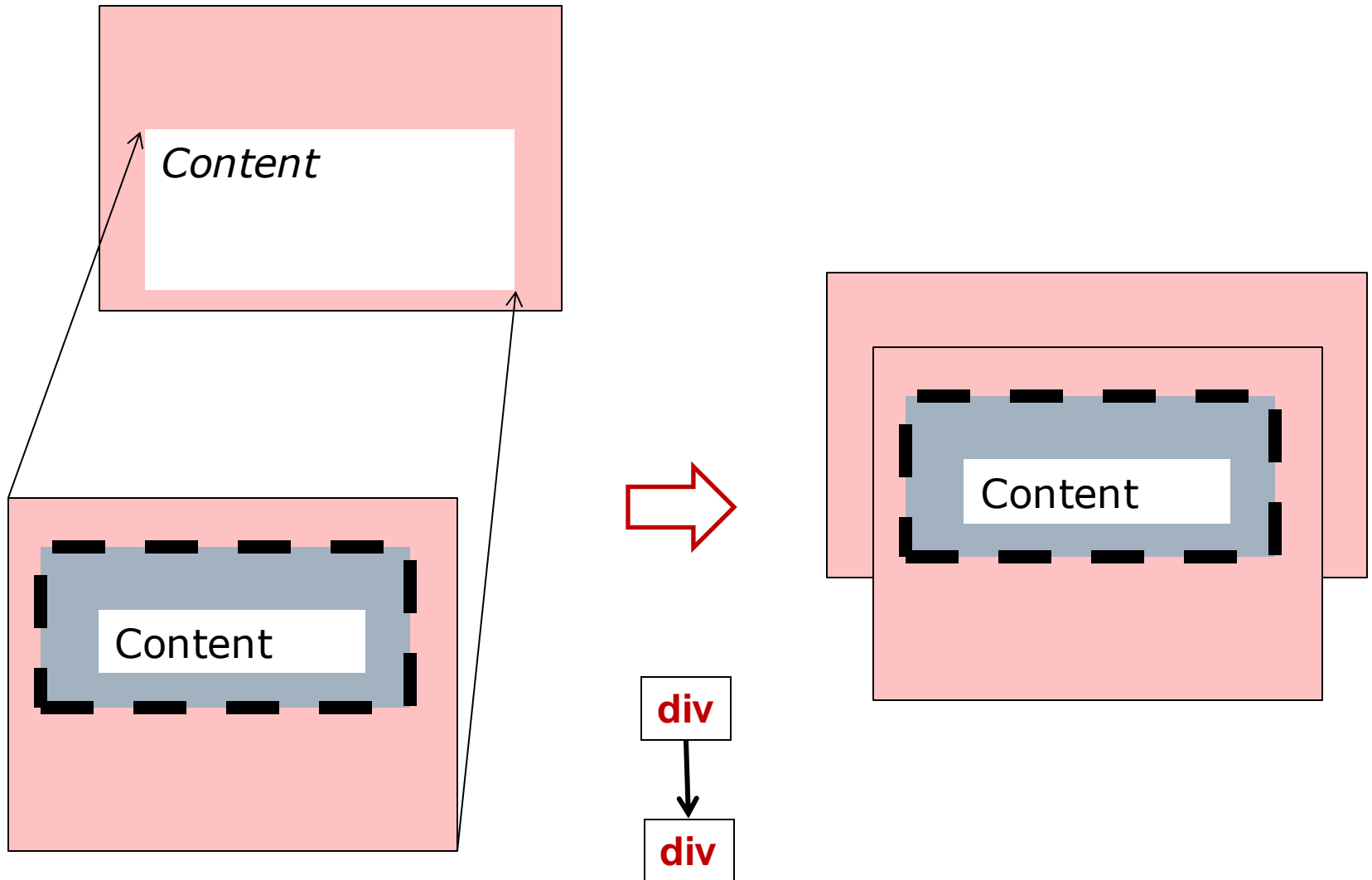


- ❑ Total width = ?
- ❑ CSS3 adds box-sizing
 - content-box (width sizes content only)
 - border-box (width includes border & padding)

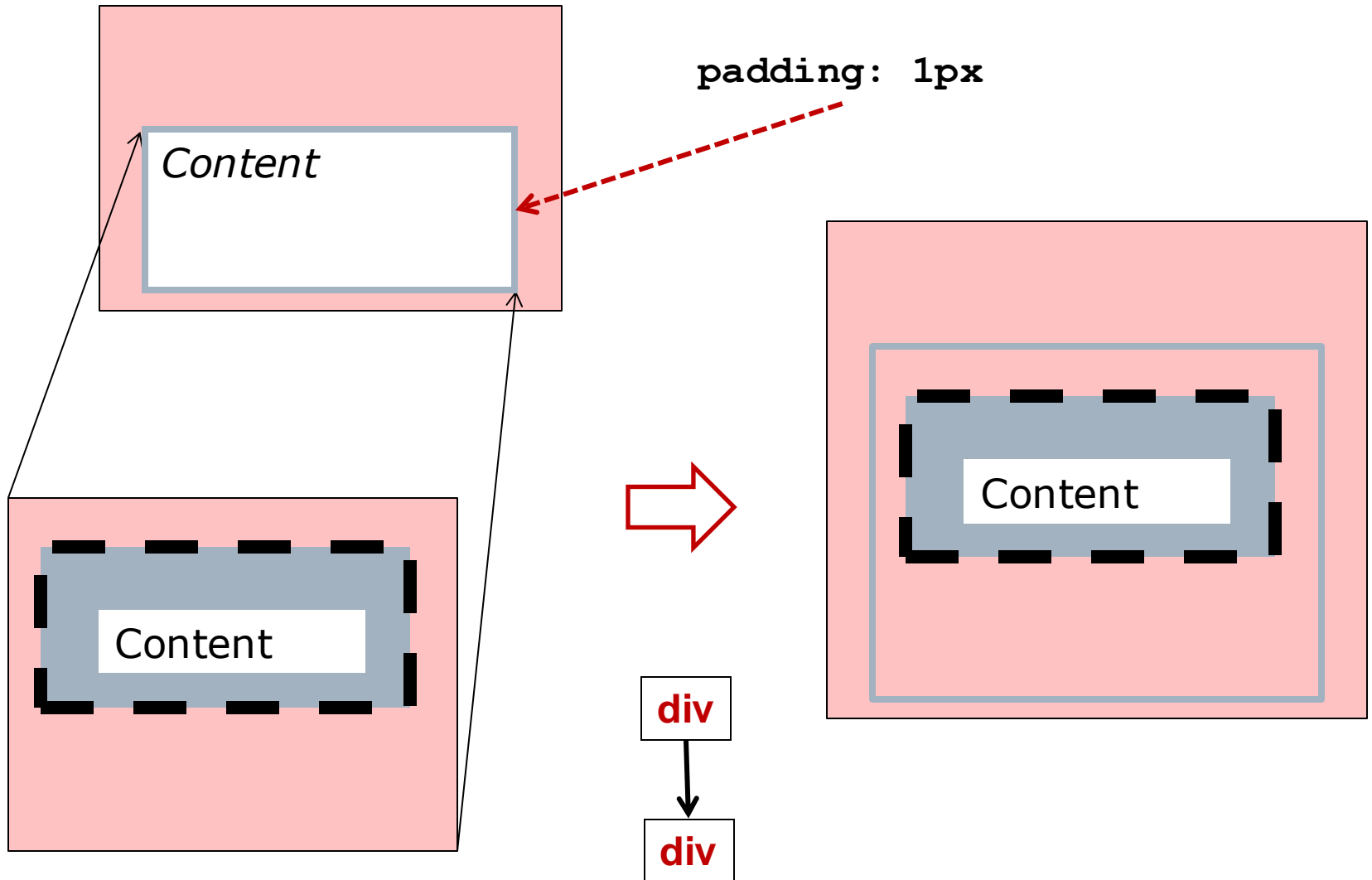
Collapsing Vertical Margins



Collapsing Nested Margins



Preventing Margin Collapse



Demo: Chrome Dev. Tools

The image shows a Chrome browser window displaying a CodePen page titled "CSS example". The page content includes an

CSS example

 and a paragraph "This page is full of goodness!". A red arrow points to an

Things to Note

 element. The Chrome DevTools interface is open, showing the Elements panel with the

Things to Note

 element selected. The Computed panel shows the following styles:

Property	Value
display	block
font-family	sans-serif
font-size	24px
font-weight	700
height	28px

The diagram in the Computed panel illustrates the box model for the

Things to Note

 element. It shows a blue box representing the element's content with dimensions 669x28. This is surrounded by a green box representing padding, an orange box representing border, and a dashed orange box representing the total margin, which is 19.920px on the top and bottom.

Inheritance and Box Sizing

- Property inheritance:
 - Text properties (eg color) **are** inherited
 - Box-related (eg border) **are not**
- Box sizing of (content) *width*: top-down
 - Set by **parent**, child “fits” inside
 - Relative, absolute
- Box sizing of (content) *height*: bottom-up
 - Set by **child**, parent “fits” around
 - Relative, absolute
- Parent and child's (vertical) margins collapse (if they touch)






Summary

- CSS separates style from structure
 - Syntax: Rules with selectors, properties
 - Link to CSS file from HTML document
- Selectors for picking elements in tree
- Box Model
 - Content, padding, border, margin
 - Margins can collapse when overlapping
- Inheritance
 - Parent passes (font) properties to child
 - Box-related properties aren't inherited

To Ponder

Class Meeting Schedule

Note: Information that appears **in this font**, below, is *not yet* officially posted. While a draft version of the material might be available, it is subject to change before its official posting.

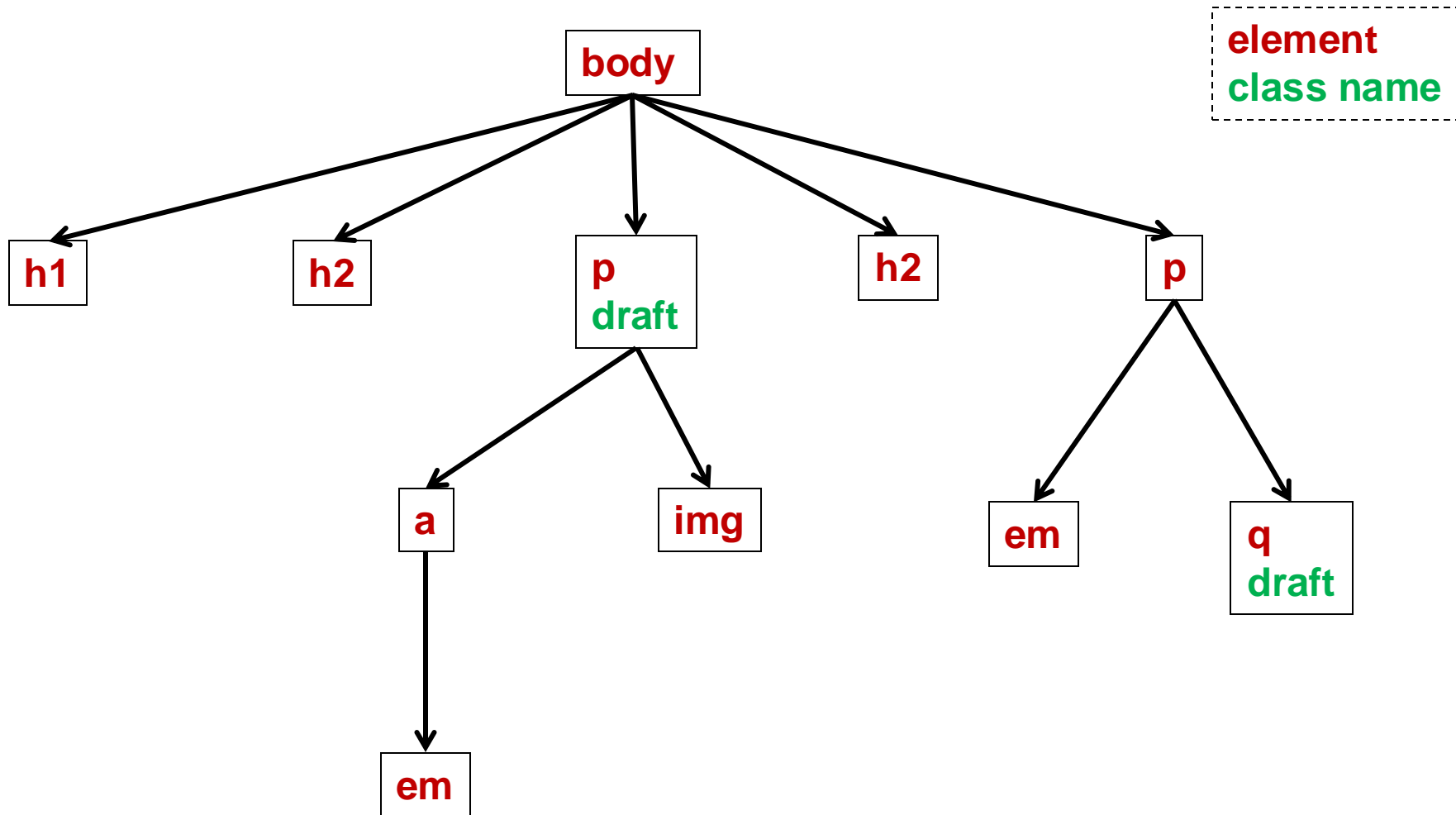
Meeting	Day	Date	Topic	Other
1	W	Aug 24	Architecture 	
2	F	Aug 26	Git: Version Control 	
3	M	Aug 29	Git: Distributed VC 	
4	W	Aug 31	Git: Extensions 	
5	F	Sep 2	Ruby: Basics 	Textbook (RoR Tutorial), 4.2–4.5

CSS Cont'd: Cascading Style Sheets

Classes

- Not all paragraphs created equally
 - Some paragraphs are not finalized (draft), so want them styled differently
- Solution: **class** attribute
 - `<p class="draft">... </p>`
- CSS syntax for selector: *elt.class*
 - `p.draft { color: gray; }`
- Wildcard (any element): *.class*
 - `.draft { font-style: italic; }`
- An element can be in multiple classes
 - Recall: attributes are a map, ie names unique
 - `<p class="draft even">... </p>`

Classes Add to Tree Structure



Notes on Classes

- When an element belongs to multiple classes, which style gets applied?
 - Different properties are combined (union)
 - Conflicts on same property need to be resolved (more later)
- Classes should reflect semantics or structure, not visual formatting
 - Bad class name: green
 - Good class name: draft
- Example: [css-classes](#)

Problem

- Multiple block elements that need to be styled together
 - Example: Header and paragraph(s) are both part of the same warning

```
<h2 class="warning">...</h2>
<p class="warning"> ... </p>
```
- This approach is awkward
 - Every block element in group needs to be decorated in this way
 - Difficult to style the entire unit (*e.g.*, add a border around the whole warning)

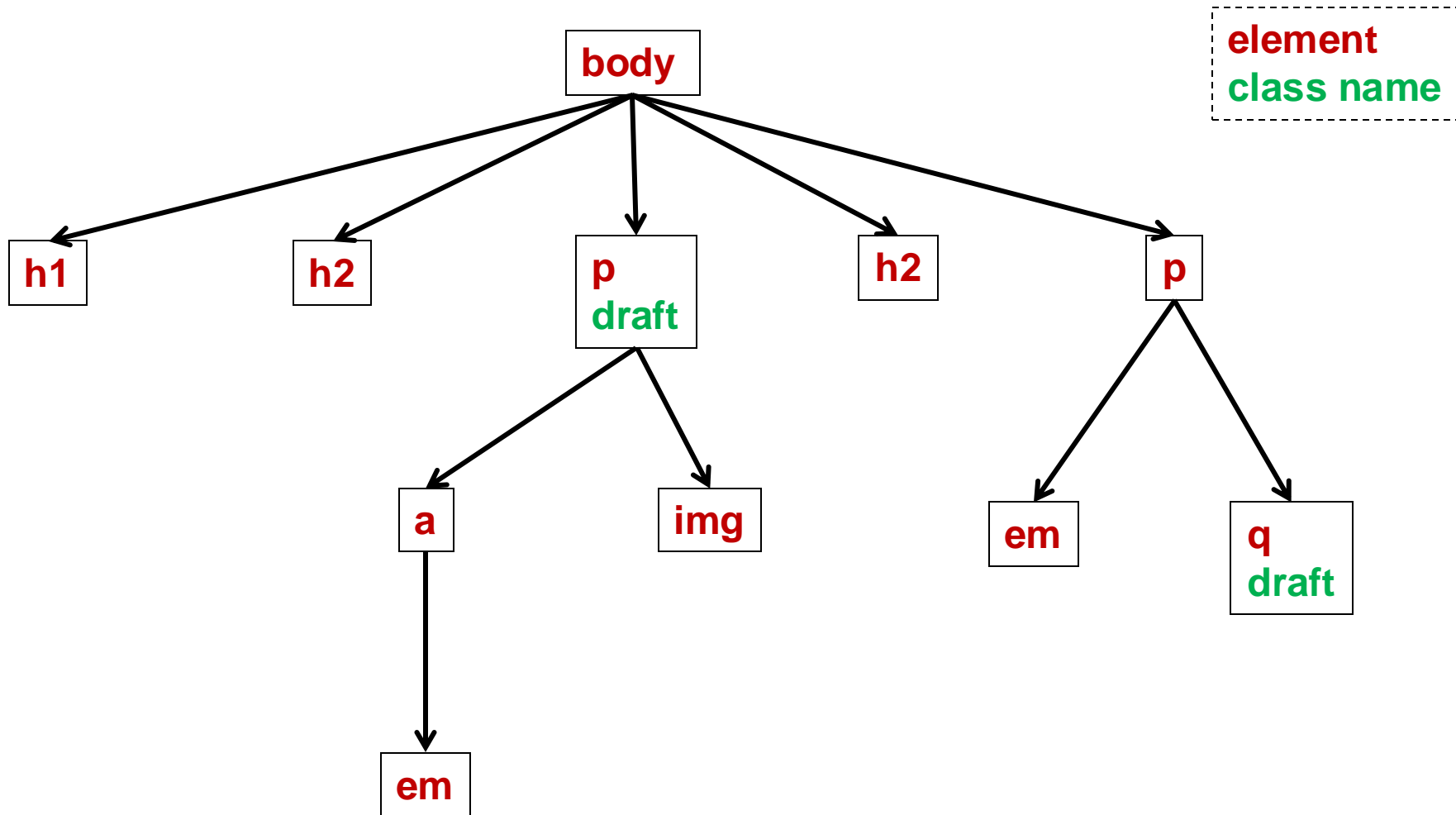
Solution: Div Element

- `div` gives a *logical* block element
- Can be styled just like any other block element
 - Font, dimension, border, margin, etc

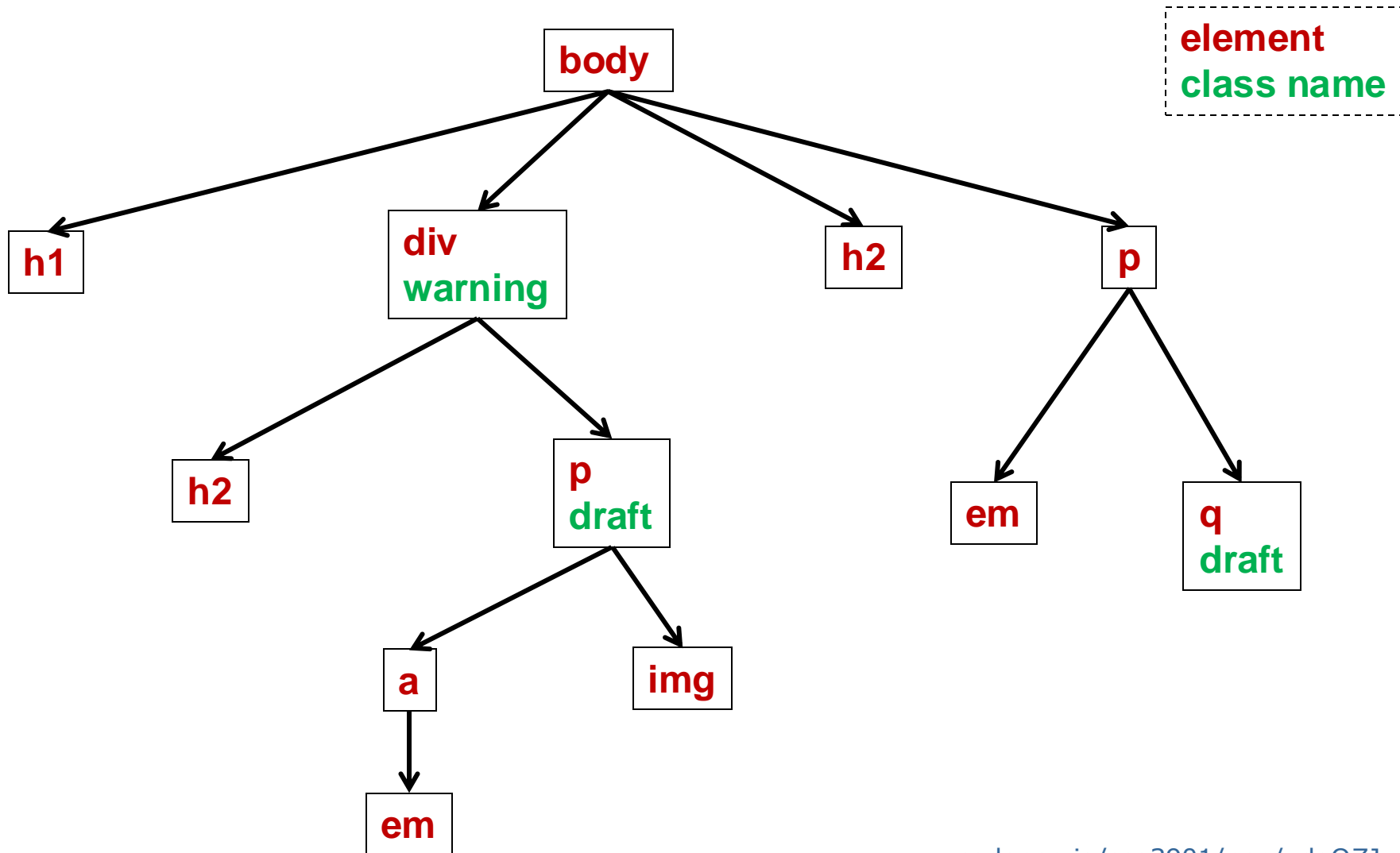
```
.warning { border: thick; }
```
- Can have block elements as children
 - Style inherited by children

```
<div class="warning">  
  <h2> ... </h2>  
  <p> ... </p>  
</div>
```

Original Tree



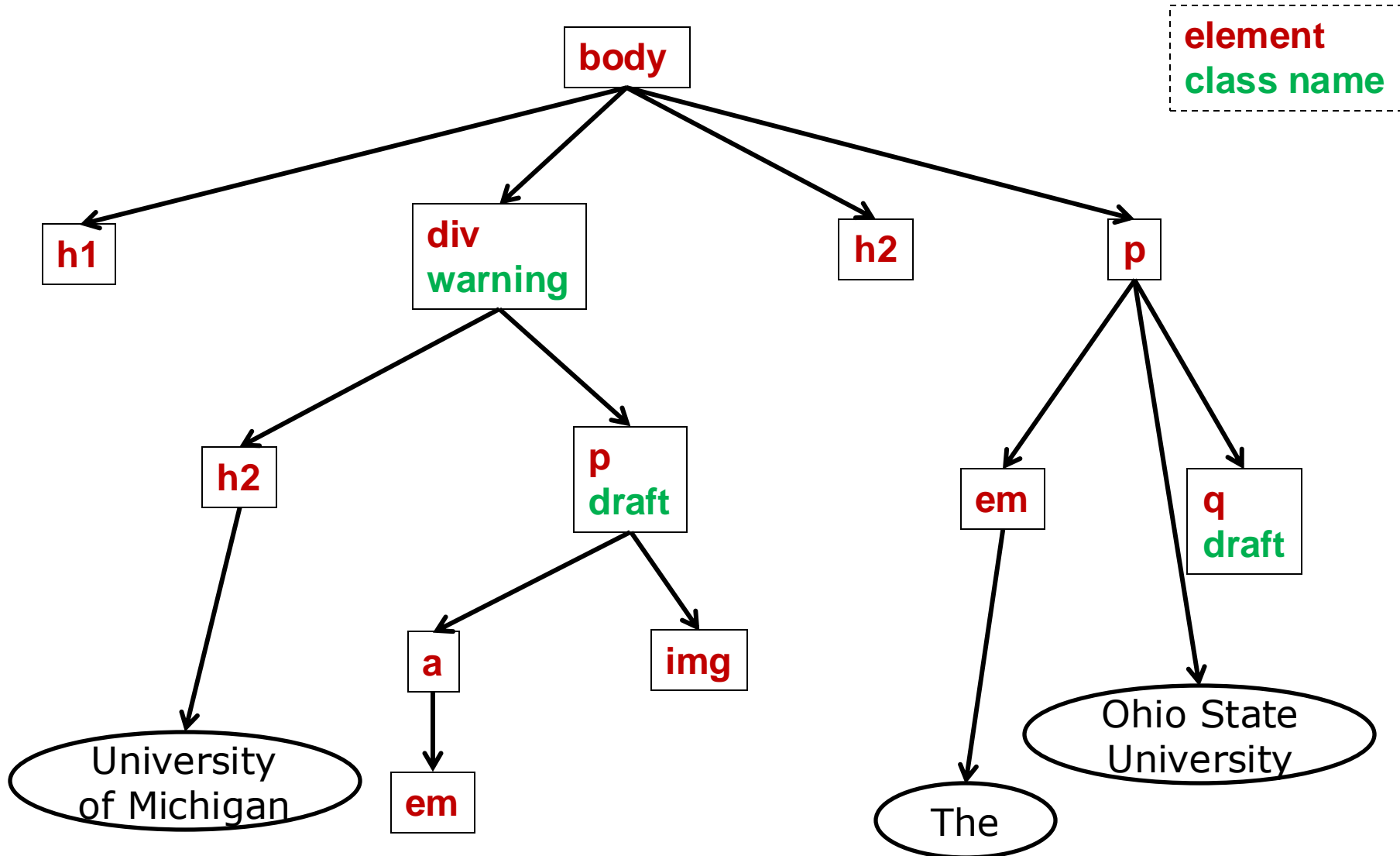
Divs in the Tree



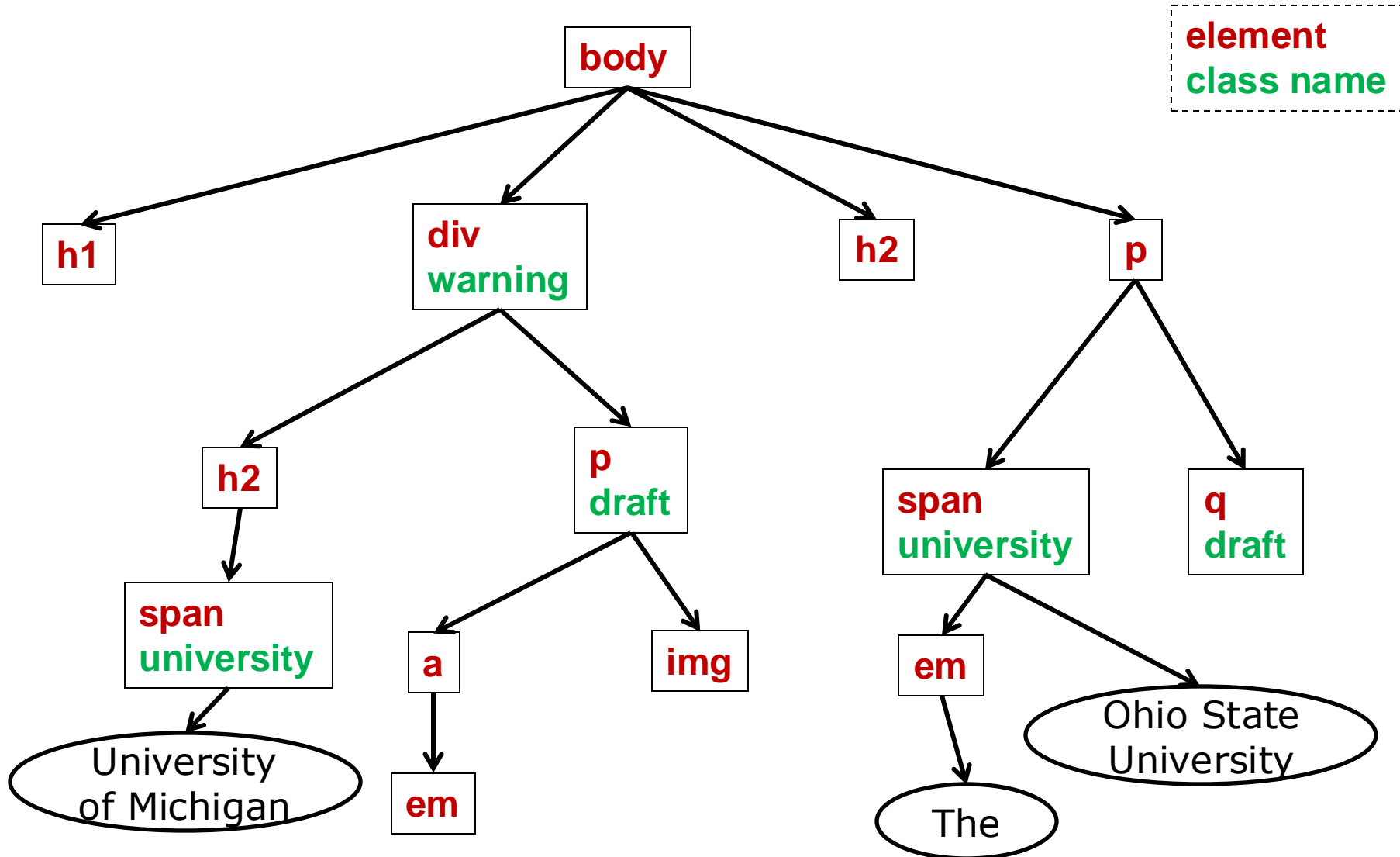
Span Element

- `div` is a (logical) block level element
 - Gives line breaks
- Sometimes styling/semantics belongs to *inline* elements
 - Text discussing different textbooks, where titles appear here and there
- Solution: `span` tag
 - `<p> One book to consider is the`
`Book of Ruby, ...`
- Now all book titles can be styled consistently
- Like `div`, `span` is often used with classes

Original Tree



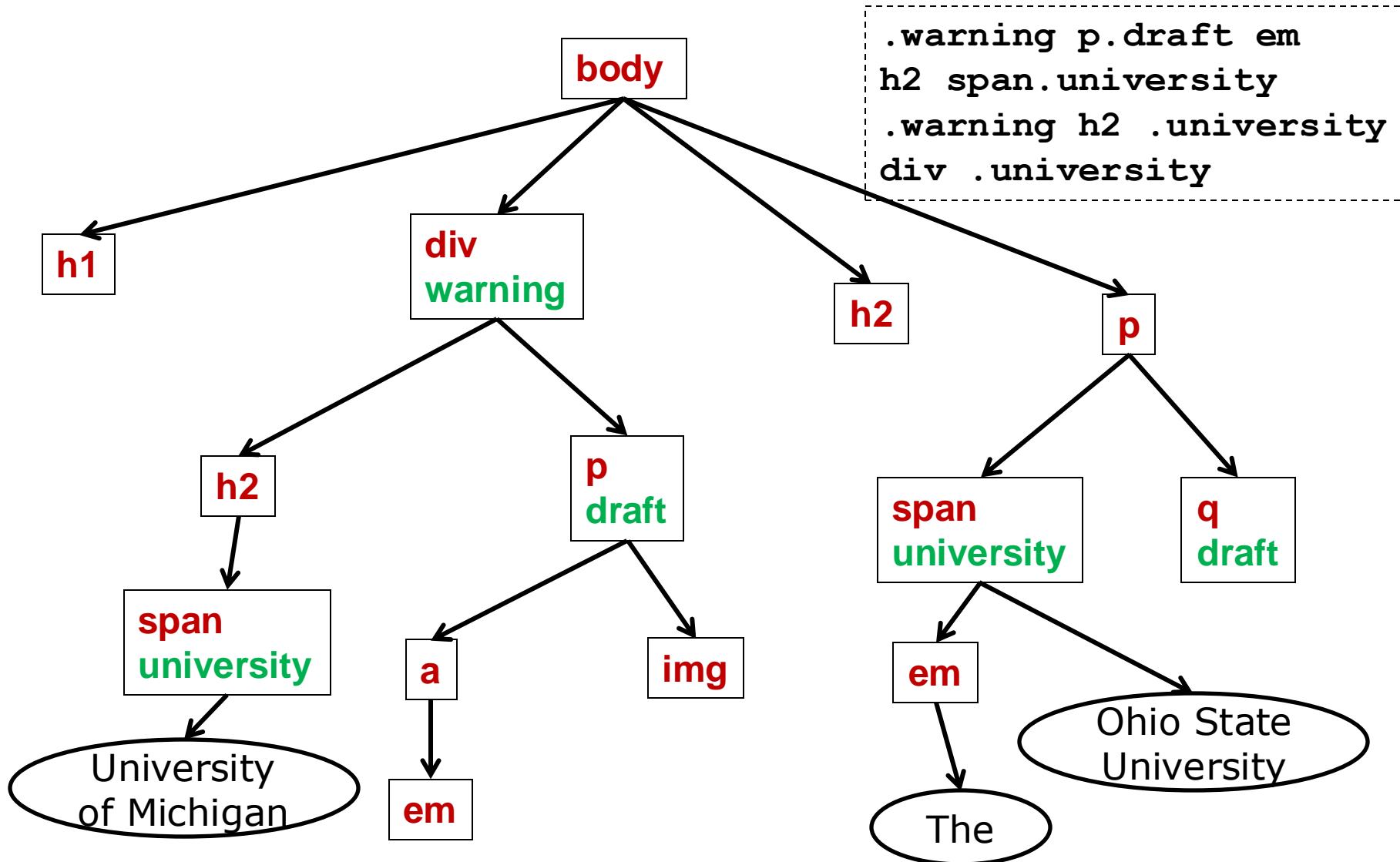
Adding Spans to the Tree



Ancestors in Selectors

- Sometimes you care about *where* in the tree an element occurs
 - University names appearing *somewhere inside* warnings need a different styling
- CSS syntax: **ancestor ancestor.. elt**
`.warning .university`
- Note: *big* difference between
`.warning em .university`
`.warning em, .university`
`.warning, em .university`

Your Turn



More Exotic Paths in Selectors

□ Child: >

```
.warning > p
```

```
.warning li > em
```

□ Adjacent sibling: +

```
h1 + p /* only first p after h1 */
```

□ General sibling: ~

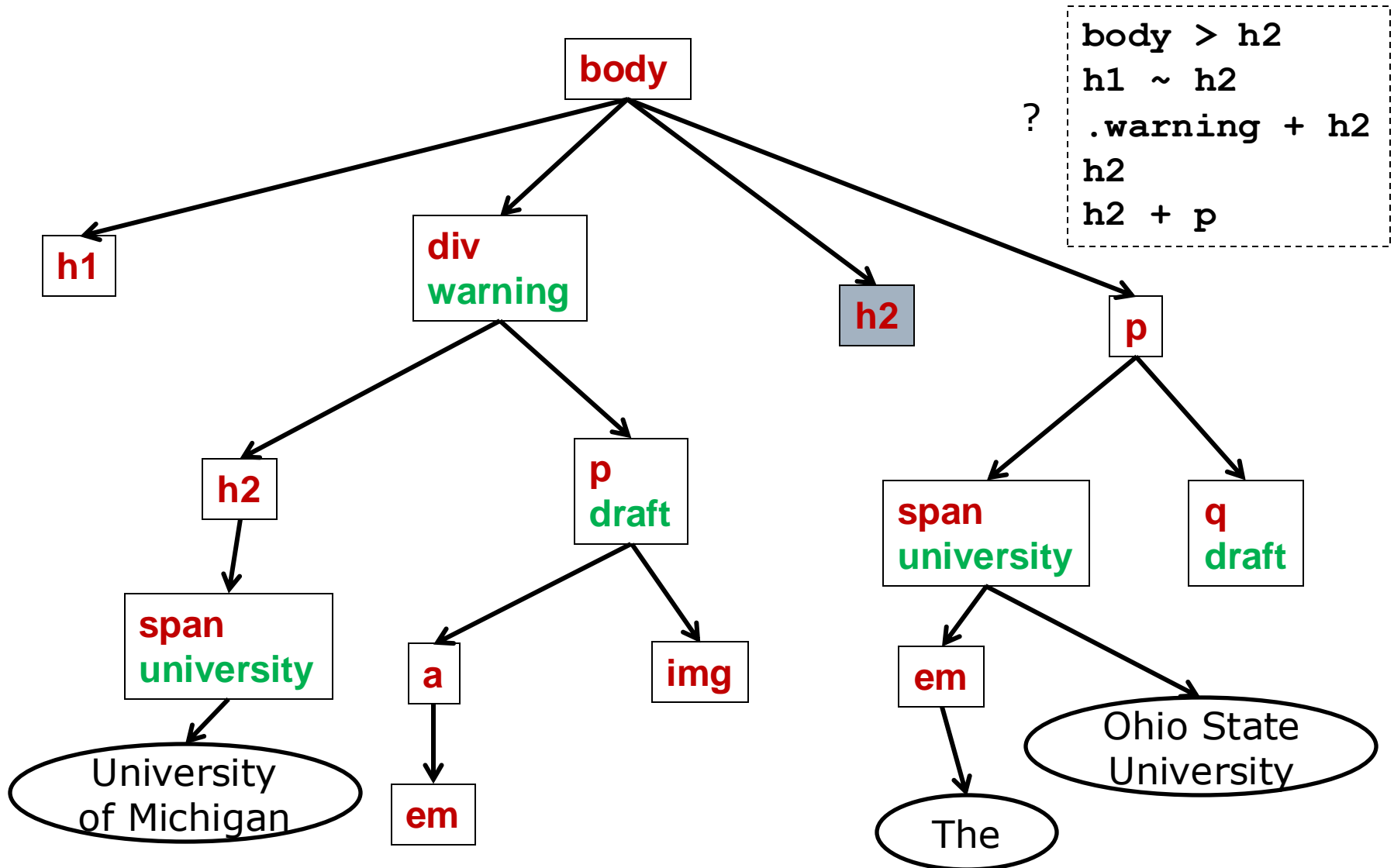
```
h1 ~ p /* sibling p's after h1 */
```

□ Attributes: [attr="value"], *=, \$=

```
input[type="button"]
```

```
a[href$=".pdf"] /* class website */
```

Your Turn: Select Shaded Node



Id = Class Plus Two Invariants

- Some classes are meant to be unique
 - *At most one* such element per page

```
<div class="sponsors">
```
- Solution: **id** attribute

```
<div id="sponsors">
```
- CSS syntax for selector: *elt#id*

```
p#sponsors { color: red; }
```
- Wildcard (any element): *#id*

```
#headline { box-style: thin; }
```
- An element can have *at most one* id

Scraping With Selectors

- Nokogiri: A Ruby gem for parsing and scraping HTML
 - Given CSS selector, returns matching elements in page
 - Returns a NodeSet, which acts like an array

```
agent = Mechanize.new
page = agent.get
      'http://www.cse.osu.edu/news '
news = page.css 'h2.content-headline '
news.class ==> Nokogiri::XML::NodeSet
news.size ==> 11
news.each { |title| puts title.text }
```

Summary

- Classes and Ids
 - Class gives an extra dimension to tree
 - ID is unique: at most one per page
 - CSS selector syntax (. vs #)
- Divs and Spans
 - Div is a logical block element
 - Span is a logical inline element
 - Often used together with classes/ids
- Selectors with ancestors, siblings
 - CSS selector syntax (space, >, +, ~)

To Ponder

- What color is the li font?

```
.draft div .warning li { }  
.draft div #main li { !important; }  
div #main ul li { }  
.draft .warning ul li { }
```

