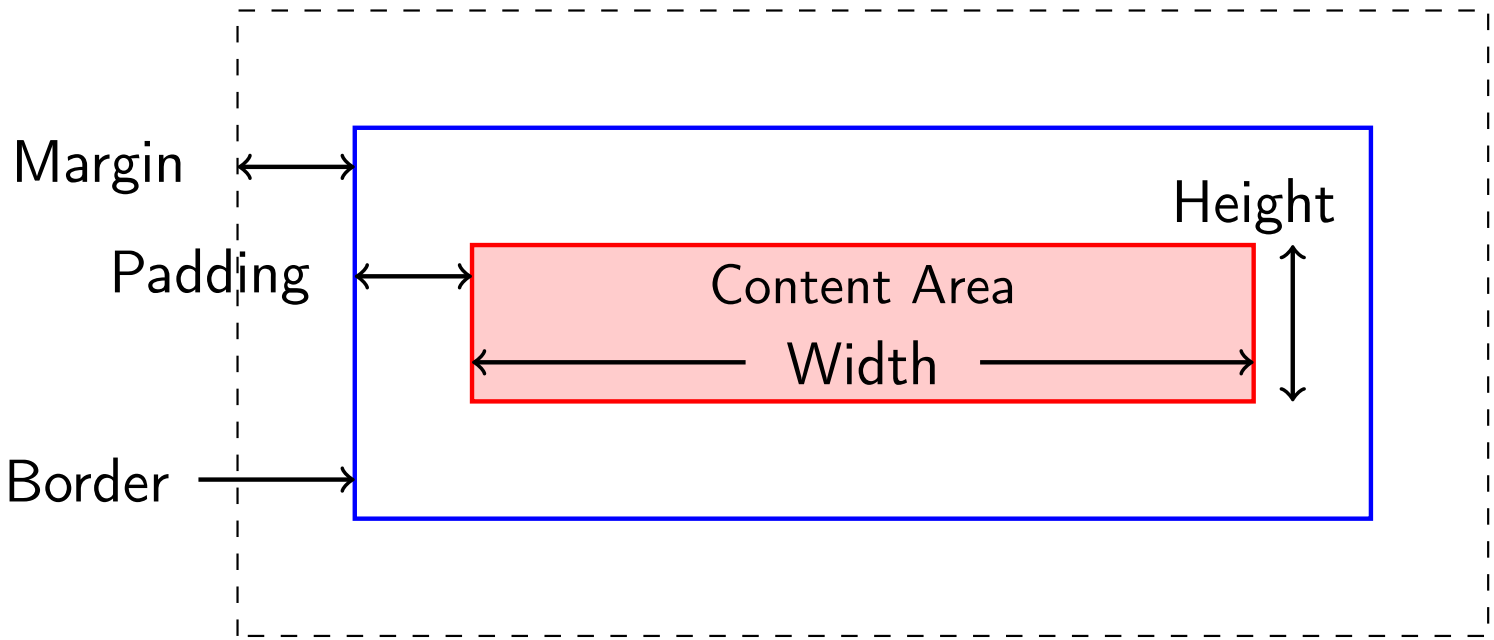
# Layout

Class 5

# The CSS Box Model

- a web page is made up of rectangular boxes
- boxes can contain other boxes
- the collection of all boxes is the page shown in the browser
- every box has a content area with width and height
- in addition, every box has padding, border, and margin, in that order from inside to out



# Borders

- the border is between padding and margin
- CSS styles affect the border:
  - border-color
  - border-width
  - border-style
  - border-radius
  - border-collapse
  - box-shadow (not actually a border, but looks like one)

# Borders

- most of these can apply either to the entire border: `border-width: 5px;`
- or to just one of the four sides of the border: `border-left-width: 5px;`

# Border Collapse

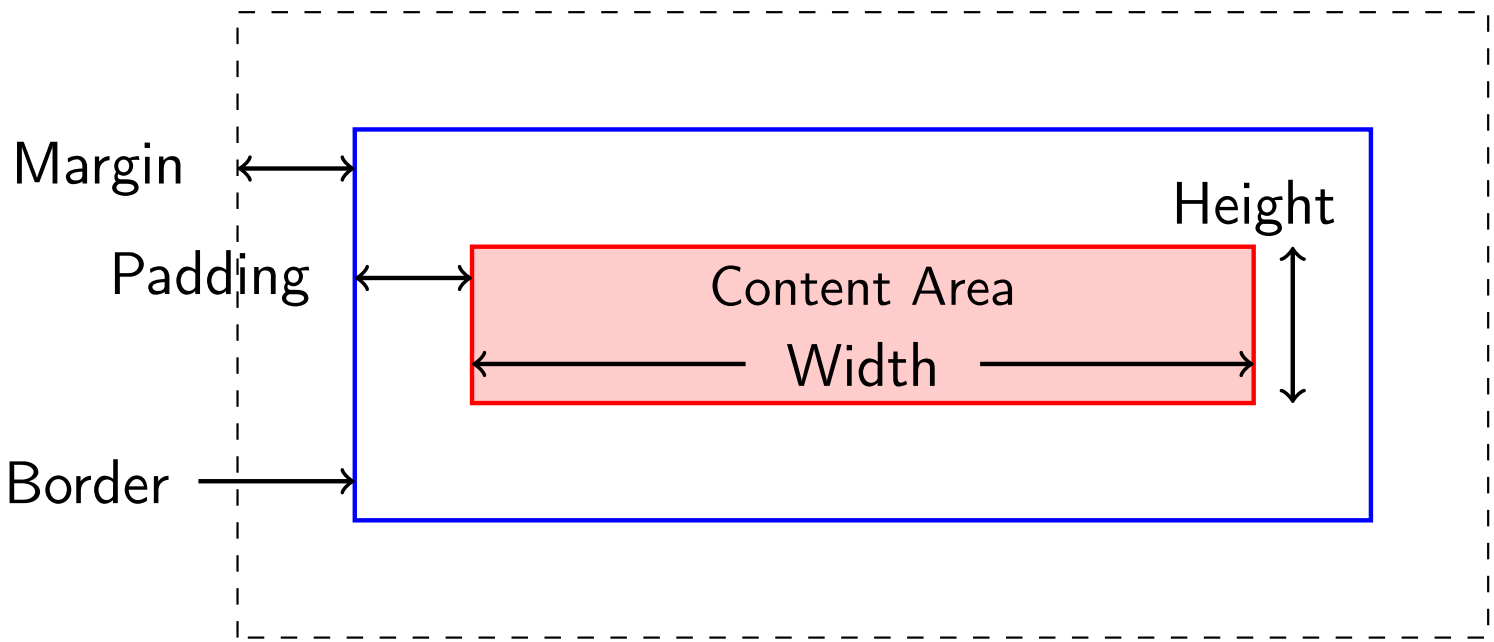
- every box has a border, but most are not drawn
- borders are frequently used in tables
- the border-collapse property applies only to table td elements
  - `border-collapse: separate;` borders are distinct, with the distance between cells controlled by the border-spacing property
  - `border-collapse: collapse;` collapses adjacent borders and the margin between them

| Item | Fat | Sugar |
|---|---|---|
| Big Mac | 27 | 9 |
| Quarter Pounder w/ Cheese | 26 | 100 |
| Bacon Clubhouse Burger | 40 | 14 |

| Item | Fat | Sugar |
|---|---|---|
| Big Mac | 27 | 9 |
| Quarter Pounder w/ Cheese | 26 | 100 |
| Bacon Clubhouse Burger | 40 | 14 |

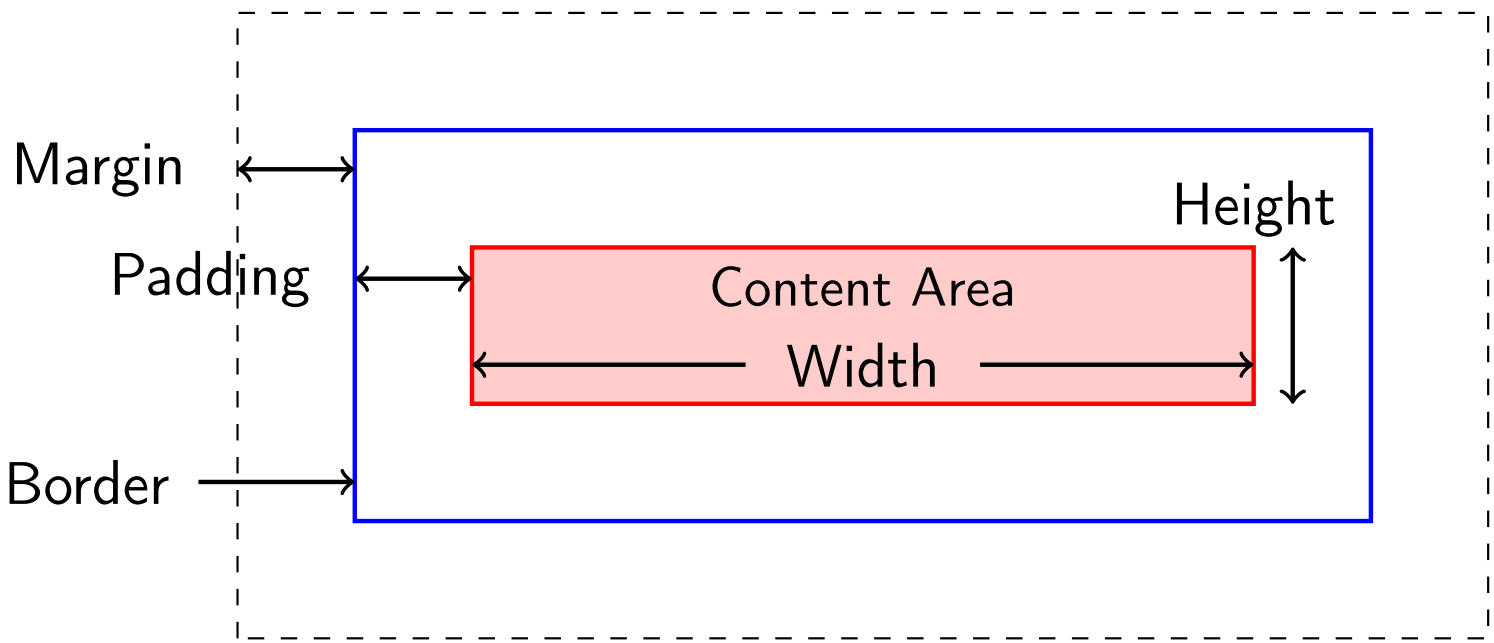
# Padding

- padding is the space between the content of a box and its border
- padding exists whether the border is visible or not
- padding itself is transparent and has no color



# Margin

- margin is the space outside a box's border
- also transparent and without color
- vertical margins between two block elements are automatically collapsed
- the final margin is the larger of the two



# Margin, Padding, and Background

- background includes the padding and border
- background does not include the margin
- if present, a background image will block the background unless the image has a transparent background

background example

# Default Styles

| Tag | Style |
|---|---|
| p, ul, ol, dl, form | margin: 1.12em |
| blockquote | margin-left & margin-right: 40px |
| h1 – h6 | font-weight: bold |
| h1 | font-size: 2em; margin: 0.67em |
| h6 | font-size: 0.75em; margin: 1.67em |
| cite, em, var, address | font-style: italic |
| pre, code, tt, kbd, samp | font-family: monospace |
| ol, ul, dd | margin-left: 40px |
| table | border-spacing: 2px |
| th | font-weight: bold; text-align: center |

Do not try to memorize these, just know they exist.

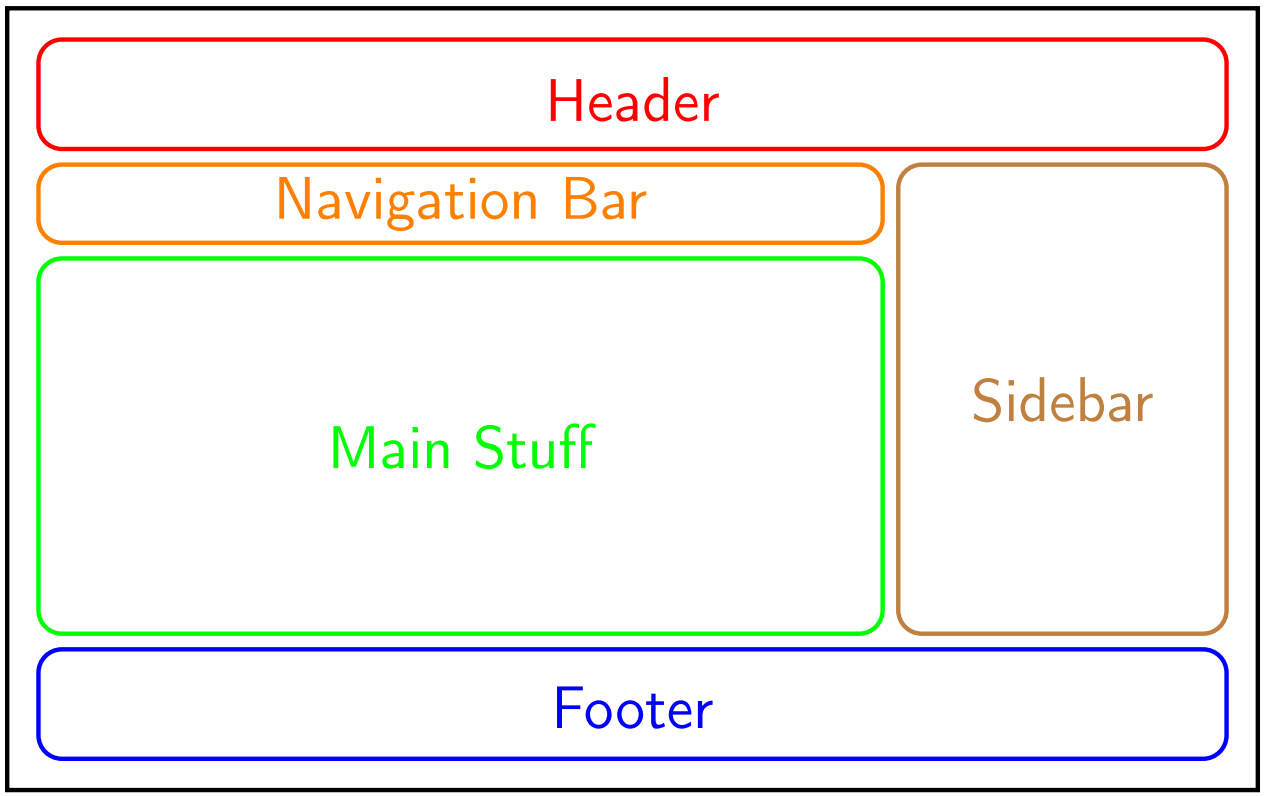
# Debugging Layout

- inspect element
  - chrome
  - firefox

example: look at background.html with element inspector

# Document Layout

- a webpage is typically not a monolithic block of text
- a typical arrangement

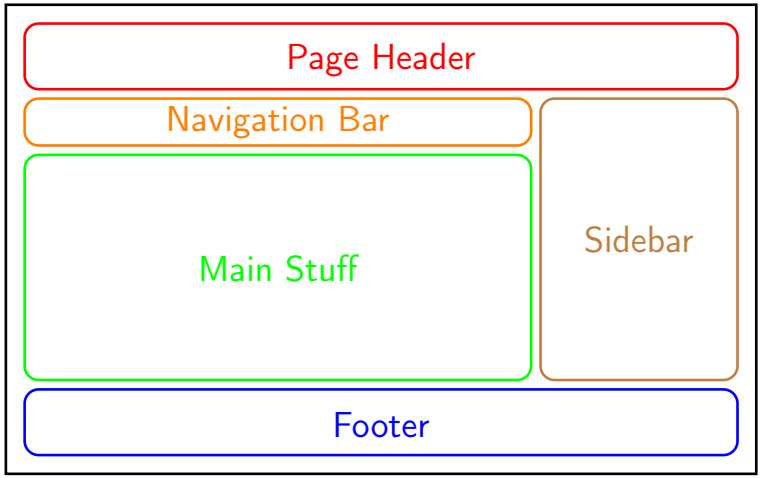


# HTML Elements for Document Layout

- HTML has block elements designed to facilitate layout
  - header (cf. head)
  - nav
  - article
  - section
  - aside
  - footer

  *and*

  - div




- these elements have no default style
- they are used to denote the semantic and logical structure of the page

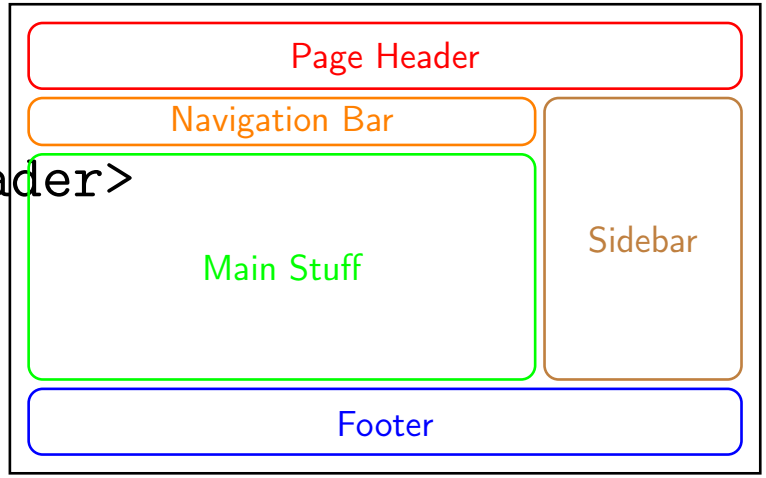
# Logical Elements

- all the semantic elements are self-explanatory except article and div

  article independent, self-contained content, should make sense on its own independent from the rest of the site, e.g., blog post, news story

  div a generic block-level element, used when nothing else is appropriate
  also used for grid containers

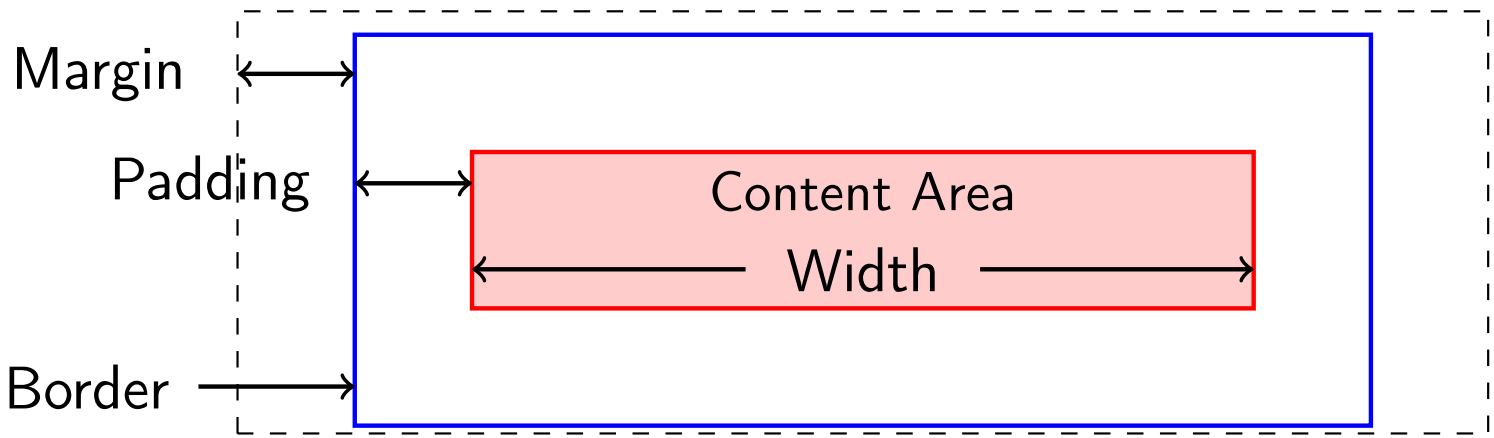
# Semantic Layout




```
<header> ... PetStorz ... </header>

<nav> ... </nav>

<section id="cats">
  <p> ... stuff about cats ... </p>
  <p> ... more stuff about cats ... </p>
</section>

<section id="dogs">
  <p> ... stuff about dogs ... </p>
  <p> ... more stuff about dogs ... </p>
</section>
```

# Width

- the width property only applies to block (and img) elements
- you can only specify the width of the content area
- the horizontal size of the entire box is the sum of:
  - left margin width
  - left border width
  - left padding width
  - content width
  - right padding width
  - right border width
  - right margin width



# Text-Align

- text-align applies only to the content area, not the entire box

```
#chai, #brew
{
  border: 2px solid;
  width: 12em;
}
```

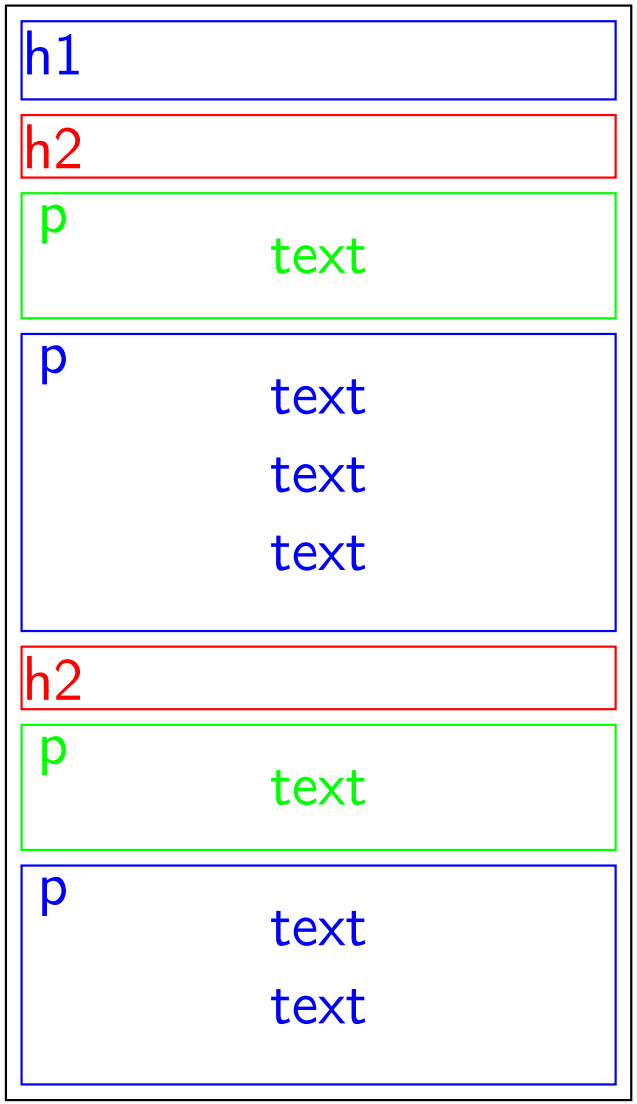
Not your traditional elixir, our Chai Chiller mixes maté with chai spices and adds an extra chocolate kick.

Want to boost your memory? Try our Black Brain Brew made with oolong tea and just a touch of espresso. Your brain will thank you for the boost!

```
#brew
{
  text-align: right;
}
```

# Flow

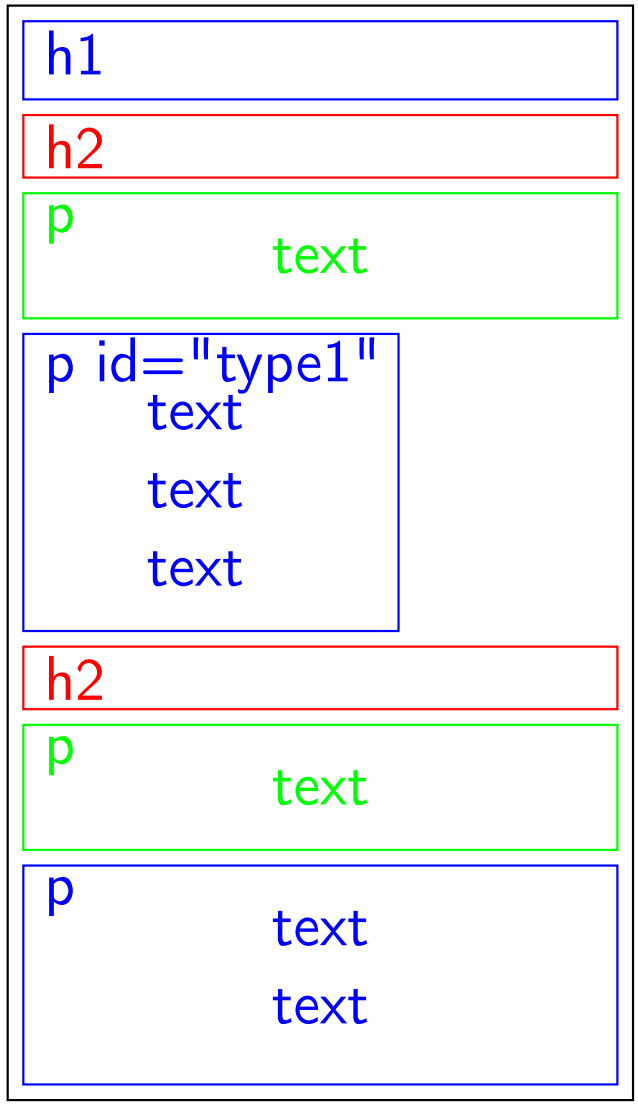
- all block elements flow vertically down the page
- one block after another
- all on the left edge of the page
- collapsed margins ("line breaks") between them



# Flow

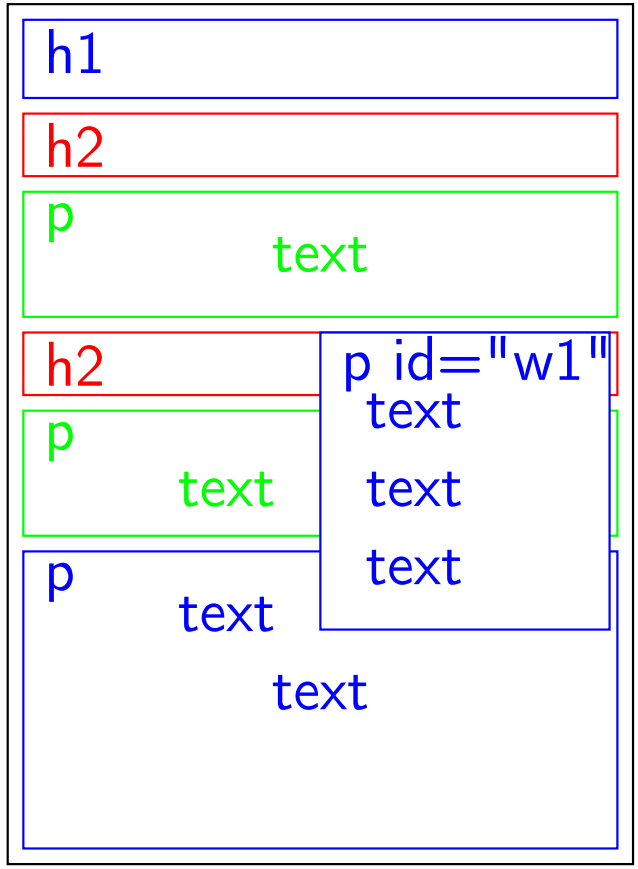
- we can set a width on a block element
- flow is still strictly vertical

```
#type1
{
  width: 60%;
}
```



# Float

- a block can be floated
- the block is removed from the flow
- flow of other elements is vertical
- their inline elements respect the floated element

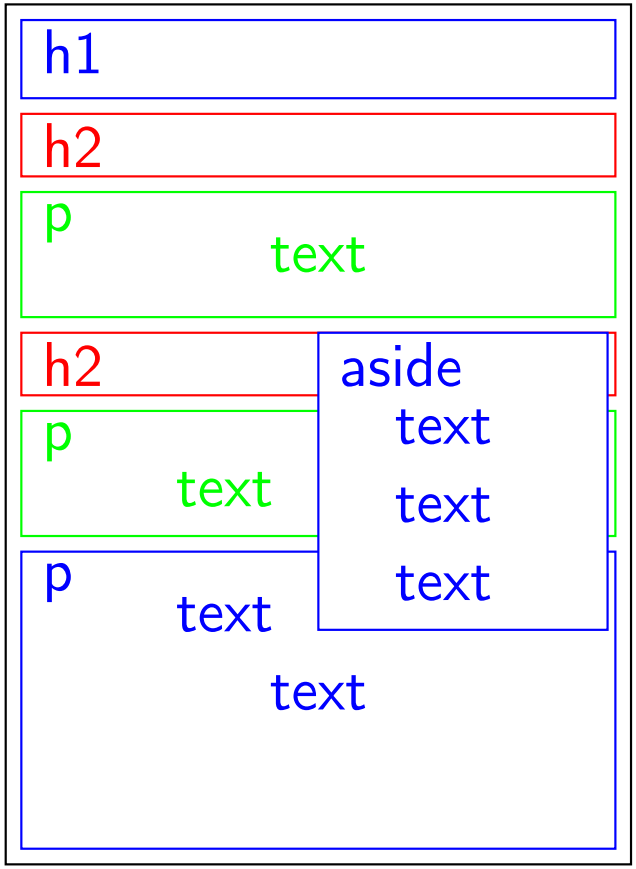



```
#w1
{
  width: 60%;
  float: right;
}
```

# Aside

- this makes a sidebar
- this is the purpose of the aside element

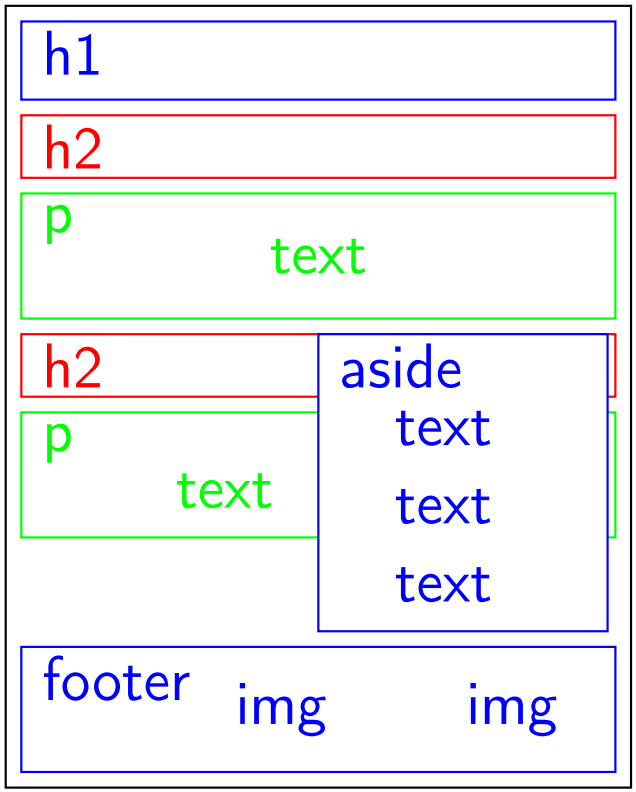
```
aside
{
  width: 60%;
  float: right;
}
```



# Clear

- clear is used to stop the float

```
aside
{
  width: 60%;
  float: right;
}

footer
{
  clear: right;
}
```



# Size and Position

Summary: every element has a default position and size

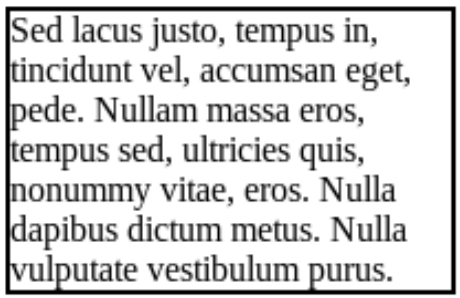
- block:
  - default position: stacked vertically down the page
  - default horizontal size: 100% of the page, sum of width, borders, margins, and paddings
  - default height: the combined height of its inline content
- inline:
  - default position: a left to right flow from top to bottom within the enclosing block element
  - default size: just large enough to fit its contents

# Horizontally Center Block Element

A block element can be centered within its context by

1. giving it a width
2. setting both left and right margins to auto

```
p
{
  margin-left: auto;
  margin-right: auto;
  width: 200px;
}
```



# Vertically Align Inline Elements

The vertical-align property specifies how an element is vertically aligned with respect to other content on the same line within its containing block.

The values are:

- baseline (the default)
- top
- middle
- bottom
- text-top
- text-bottom

# Absolute Positioning

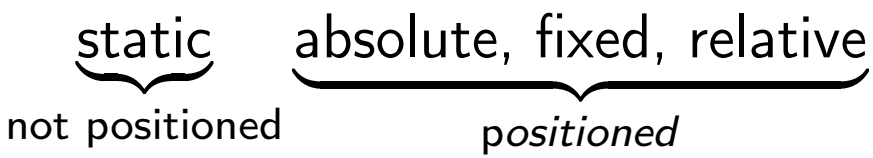
- block elements normally flow vertically down the page
- this is called static positioning, which just means "normal"
- block elements can be floated, pulled up out of the flow
- they are still affected by everything above them

Sometimes we want more control

- block items can be given an absolute position

# Absolute Positioning

- it's a little more complicated
  - absolute positioning is relative to the nearest containing element that is positioned




absolute example

# Fixed Positioning

- absolute positioning is relative to the nearest containing positioned element
- fixed positioning is relative to the browser window

fixed example

# Float vs Fixed and Absolute

- all three are removed from the page's flow
- float still affects inline elements — they flow around it
- fixed and absolute are ignored by inline elements — one is above the other

# Display — Block and Inline

- the display property has two radically different uses
- the first is block vs inline
  - display: block causes this element to be treated as a block element even if it's really an inline element (caution!)
  - display: inline causes this element to be treated as inline even if it's really a block element (for lists)
- has no effect on HTML validation, often causing great confusion

inline example