Span and Div

- and are tags that let you select a group of elements and apply styles to them
- is an inline tag
  - no breaks are added before or after
- is a block tag
  - a break is usually added by the browser before and after the tags

Example of Span and Div

```
<html>
<head><title>Span and Div</title>
<style type="text/css">
.red {color:red; font-family: Georgia; font-weight:bold;}
</style>
</head><body>
<p>This will also appear as normal paragraph text except <span class="red">here
    because I made the text red,bold, and Georgia font without breaking up the
    paragraph.</span> Now I am back to normal... </p>
<p>
I start off as normal paragraph text here as well. However, when I use a div tag to apply a
    style, my paragraph is broken like <div class="red">this. You can see that the
    browser sets off this text from the text before and </div> after it. Now I am back to
    normal again.
</p>
</body></html>
```

Block Level Elements:

- If no width is set, will expand naturally to fill its parent container.
- Can have margins and/or padding
- If no height is set, will expand naturally to fit its child elements (assuming they are not floated or positioned)
- By default, will be placed below previous elements in the markup (assuming no floats or positioning on surrounding elements)

Inline Elements:

- Flows along with text content, thus
- Is subject to white-space settings in CSS
- Will ignore top and bottom margin settings, but will apply left and right margins, and any padding
- Will ignore the width and height properties

- Block-level elements in HTML:
  - Heading tags, e.g., <H1>, <H2>
  - <p> , <pre>
  - List tags, e.g., <ul>, <ol>, <dl>
  - , <body>
  - <hr>, <img>
  - <fieldset> , <form>
  - <table>
- Inline Elements:
  - b, big, i, small
  - abbr, strong
  - a, bdo, br, img, map, span, sub, sup
  - button, input, label, select, textarea.

Formatting Width & Height of Block-Level Boxes

- To set the width of a block-level element, use:
  - width:*value*
  - height:*value*
  - where *value* can be a length value, a percentage, or auto
- E.g., textarea {width:225px; height:100px}

BOX MODEL

- In CSS, the term "box model" is used when talking about design and layout.
- The CSS box model is essentially a box that wraps around HTML elements, and it consists of: margins, borders, padding, and the actual content.
- The box model allows us to place a border around elements and space elements in relation to other elements.

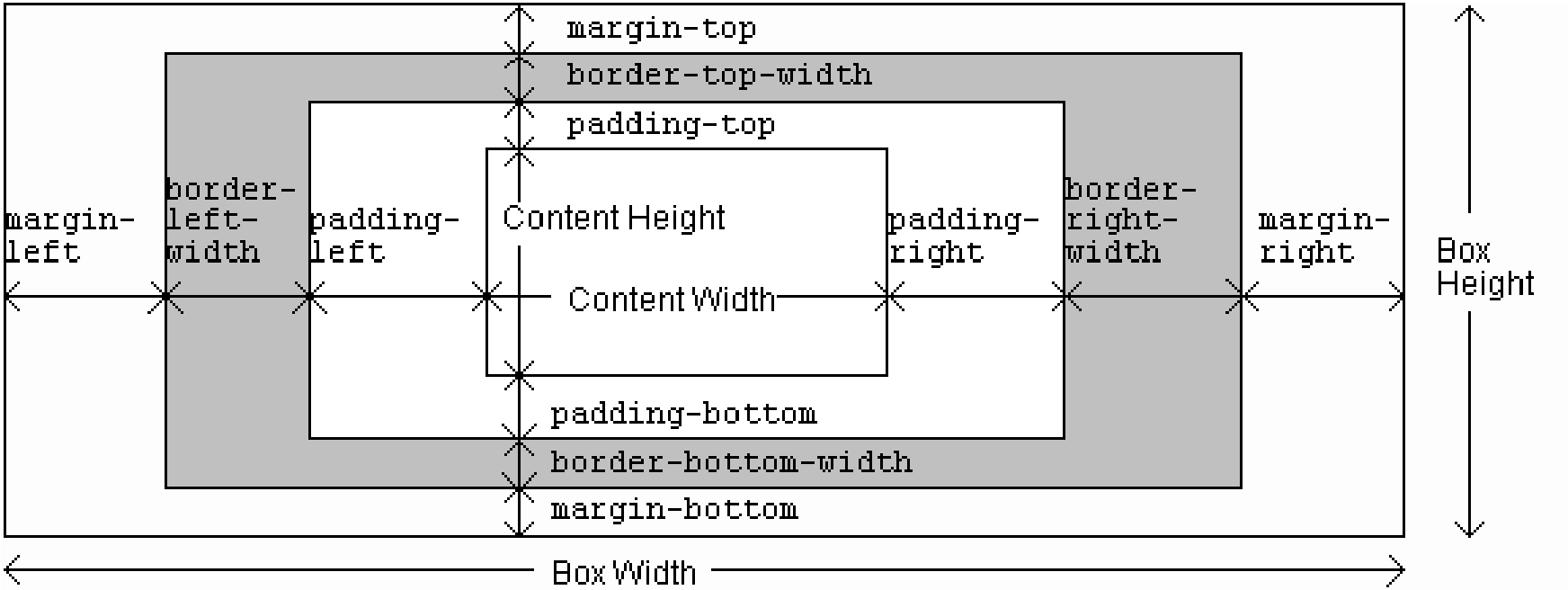
CSS Box Model

- Every rendered element occupies a box:

Explanation of the different parts:

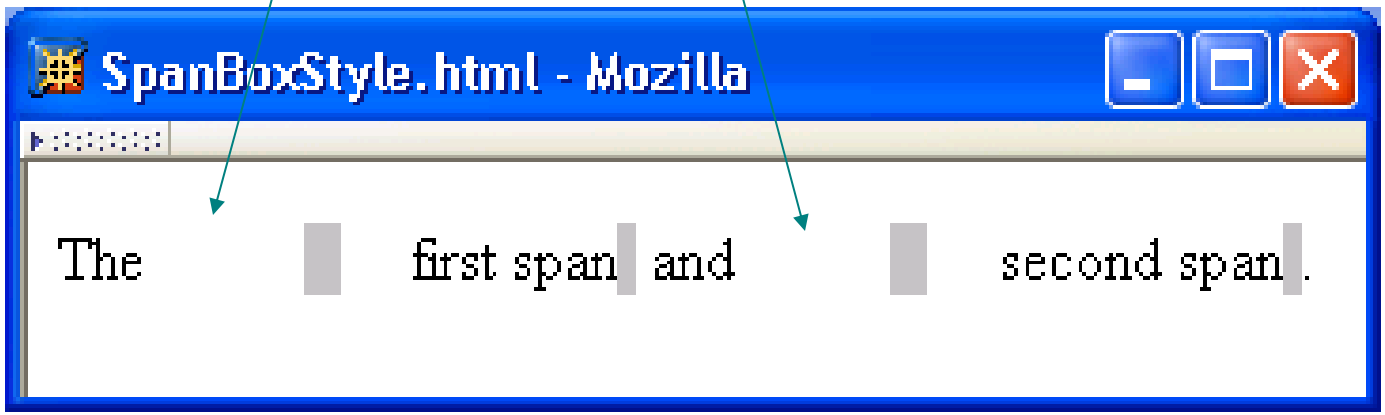
- **Margin** - Clears an area around the border. The margin does not have a background color, it is completely transparent
- **Border** - A border that goes around the padding and content. The border is inherited from the color property of the box
- **Padding** - Clears an area around the content. The padding is affected by the background color of the box
- **Content** - The content of the box, where text and images appear

CSS Box Model



CSS Box Model

```
span { margin-left: 1cm;
       border-left-width: 10px;
       border-left-color: silver;
       border-left-style: solid;
       padding-left: 0.5cm;
       border-right-width: 5px;
       border-right-color: silver;
       border-right-style: solid }
```



Margins & Padding

- **Margins**
- Margins define the space around elements outside the border
- Margin properties can have negative values in order to deliberately overlap content
- Margin properties will affect the position of background elements (graphics and/or colours) in relation to the edges of the containing block element
- Margin properties can be defined independently on top, right, bottom and left, or all-at-once using CSS shorthand

The margin property can have from one to four values.

margin:25px 50px 75px 100px;

top margin is 25px
right margin is 50px
bottom margin is 75px
left margin is 100px

margin:25px 50px 75px;

top margin is 25px
right and left margins are 50px
bottom margin is 75px

margin:25px 50px;

top and bottom margins are 25px
right and left margins are 50px

margin:25px;

all four margins are 25px

Margins & Padding

- **Padding**
- Padding defines the space around elements inside the border; i.e between the border and the content itself
- Padding properties cannot have negative values
- Padding properties do not affect the position of background elements (graphics and/or colours) in the containing block element; only the position of content.
- Padding properties can be defined independently on top, right, bottom and left, or all-at-once using CSS shorthand

The padding property can have from one to four values.

padding:25px 50px 75px 100px;

top padding is 25px
right padding is 50px
bottom padding is 75px
left padding is 100px

padding:25px 50px 75px;

top padding is 25px
right and left paddings are 50px
bottom padding is 75px

padding:25px 50px;

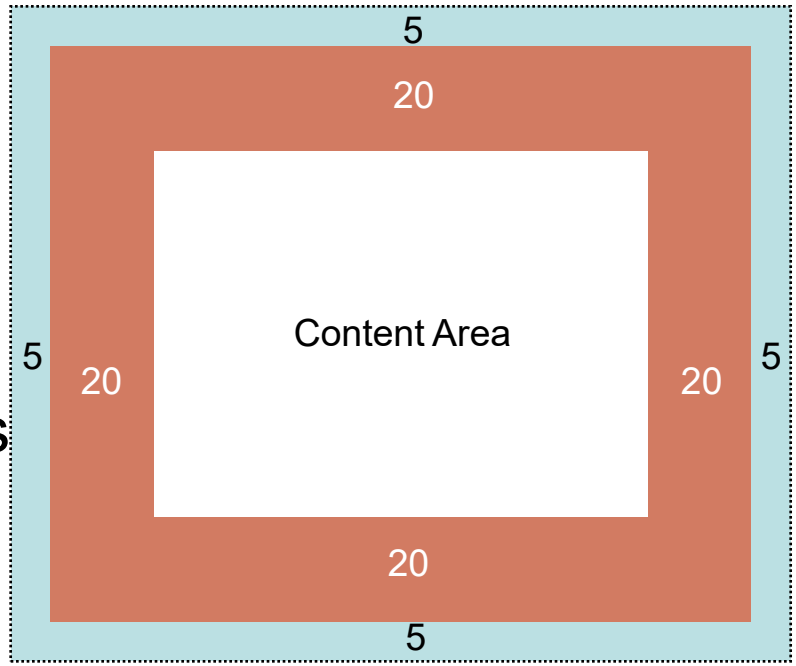
top and bottom paddings are 25px
right and left paddings are 50px

padding:25px;

all four paddings are 25px

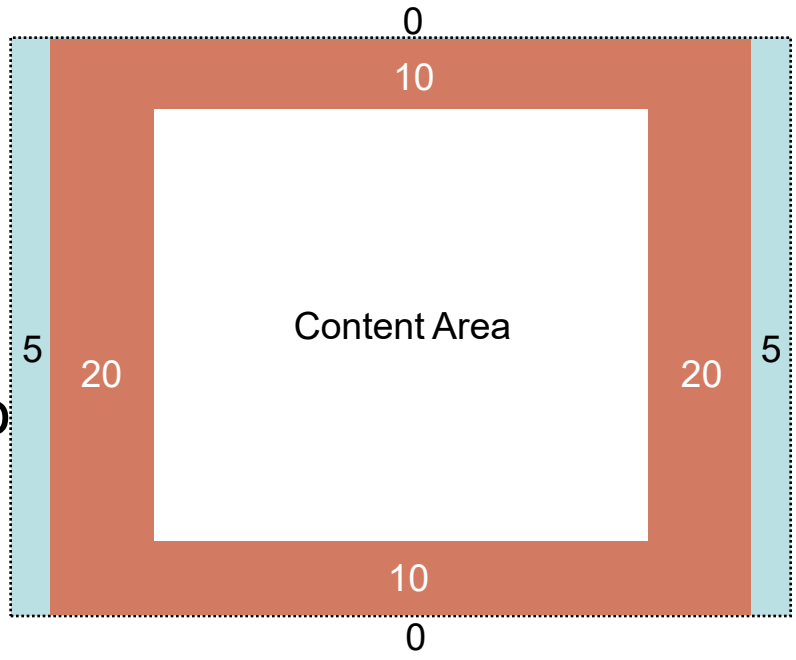
CSS Shorthand: Margin and Padding

- You can also apply just one value, example:
- ```
  #container {
        padding: 20px;
        margin: 5px;
  }
  ```
- Which will apply the value specified equally on all 4 sides


  ```

CSS Shorthand: Margin and Padding

- And you can apply two values, example:
- #container {
  padding: 10px 20px;
  margin: 0px 5px;
  }
- The first value is applied to the top and bottom
- The second value is applied to the left and right



Borders

- Borders can be applied to any block element
- Borders always come outside the padding area, but inside the margin area.
- Border properties cannot have negative values
- If a border is not specified, the default is no-border (i.e. no border appears and no space between any margin and padding is allocated for a border)
- Border properties can be defined independently on top, right, bottom and left, or all-at-once using CSS shorthand

Borders

- The core border properties are:
  - **Width:** px, in, cm, or ‘thin’, ‘medium’, ‘thick’
  - **Style:** dotted, dashed, solid, double, groove, ridge, inset, outset, hidden, etc
  - **Color:** ‘blue’, ‘red’, #FF9900, etc
- You can also create the effect of a border by using a graphic image in a CSS background property, instead of the border property

Border - Individual sides

In CSS it is possible to specify different borders for different sides:

```
p
{
border-top-style:dotted;
border-right-style:solid;
border-bottom-style:dotted;
border-left-style:solid;
}
```

The border-style property can have from one to four values.

- **border-style:dotted solid double dashed;**
  - top border is dotted
  - right border is solid
  - bottom border is double
  - left border is dashed
- **border-style:dotted solid double;**
  - top border is dotted
  - right and left borders are solid
  - bottom border is double
- **border-style:dotted solid;**
  - top and bottom borders are dotted
  - right and left borders are solid
- **border-style:dotted;**
  - all four borders are dotted

Property: Display

- Display property will change your block level element to inline element and vice-versa.
- Three values of display properties are:
  - Block
  - Inline
  - None

Display Examples

```
span.block
{
  display: block;
  width: 100%;
}
```

Making a behave like a

```
div.inline
{
  display: inline;
}
```

Making a behave like a

```
img.hidden
{
  display: none;
}
```

Hiding any images of the class ‘hidden’

Using the Float and Clear property

- To float (wrap) a block-level element, use:
  - float:*margin*
  - Where *margin* = right, left, none

property	description
float	side to hover on; can be left, right, or none (default)

- To prevent an element from wrapping, use:
  - Clear:margin
  - Where margin=right, left, both, none

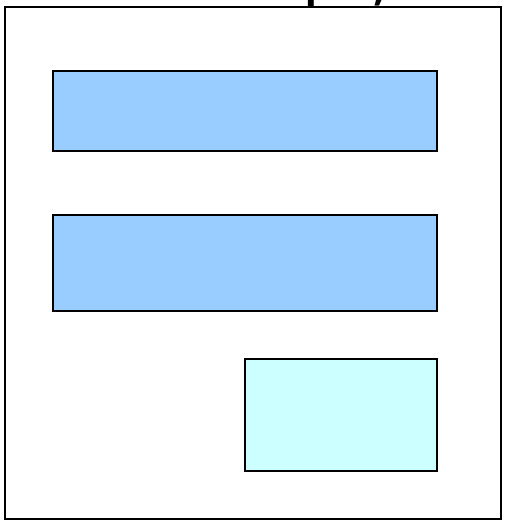
property	description
clear	disallows floating elements from overlapping this element; can be left, right, or none (default)

Float:

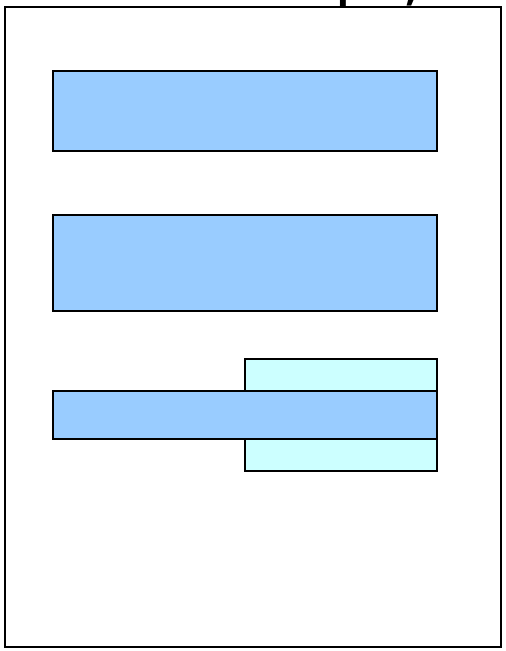
- Elements are floated horizontally, this means that an element can only be floated left or right, not up or down.
- A floated element will move as far to the left or right as it can. Usually this means all the way to the left or right of the containing element.
- The elements after the floating element will flow around it.
- The elements before the floating element will not be affected.
- Eg: If you place several floating elements after each other, they will float next to each other if there is room.
- Eg: Elements after the floating element will flow around it. To avoid this, use the clear property. The clear property specifies which sides of an element other floating elements are not allowed.

Using the Float Attribute

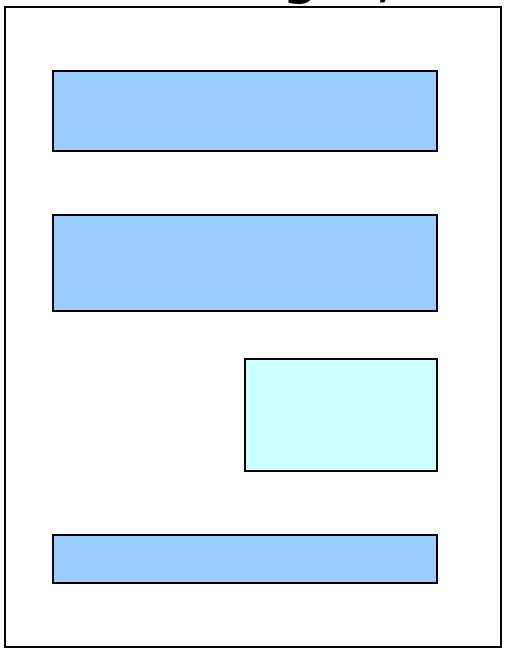
3 boxes
float:right;
width:50px;



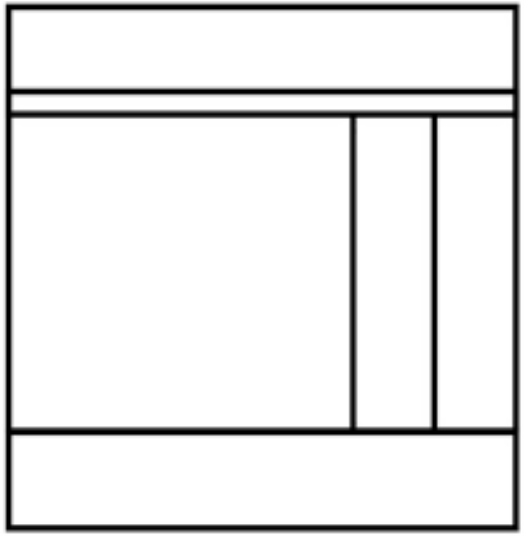
4 boxes
float:right;
width:50px;

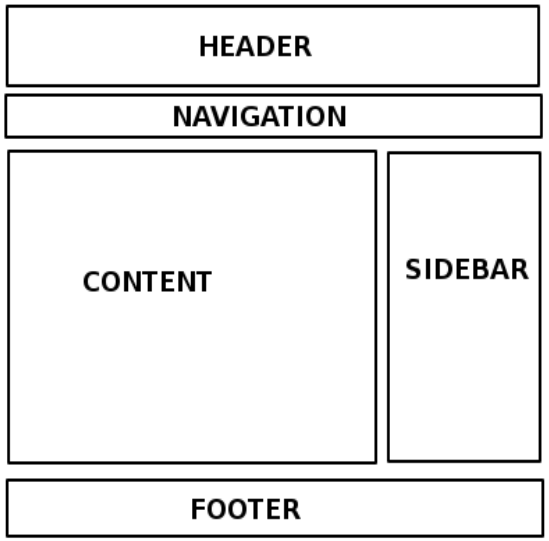


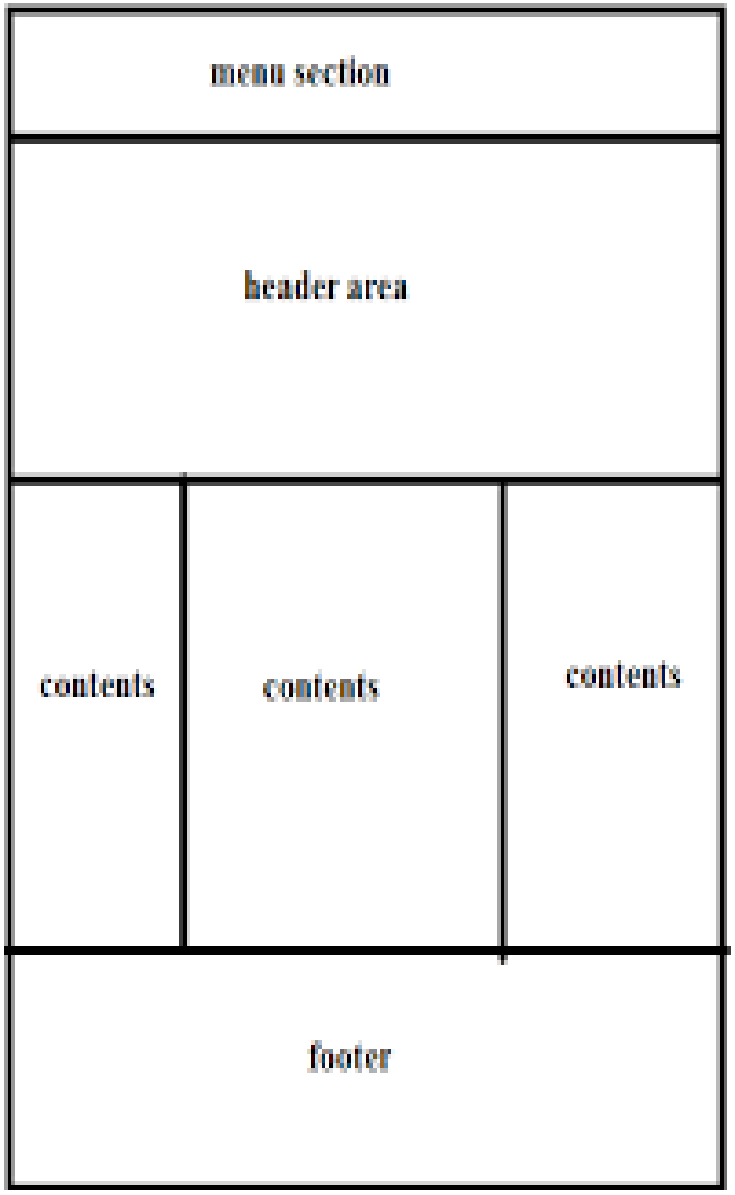
4 boxes
float:right;
width:50px;
clear:right;

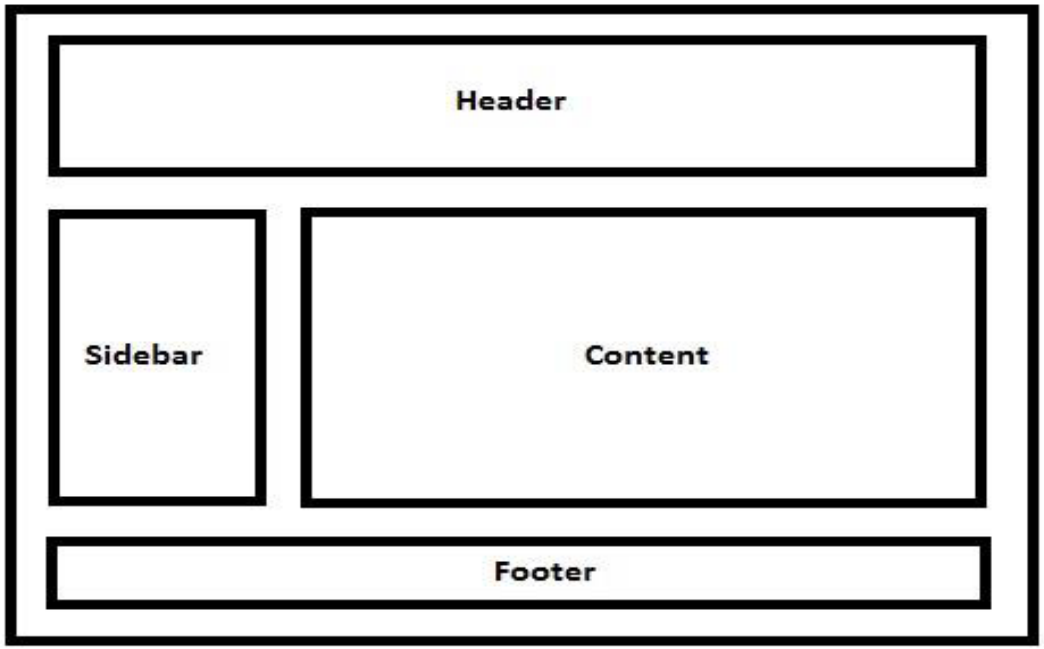


Different Layouts with float and clear







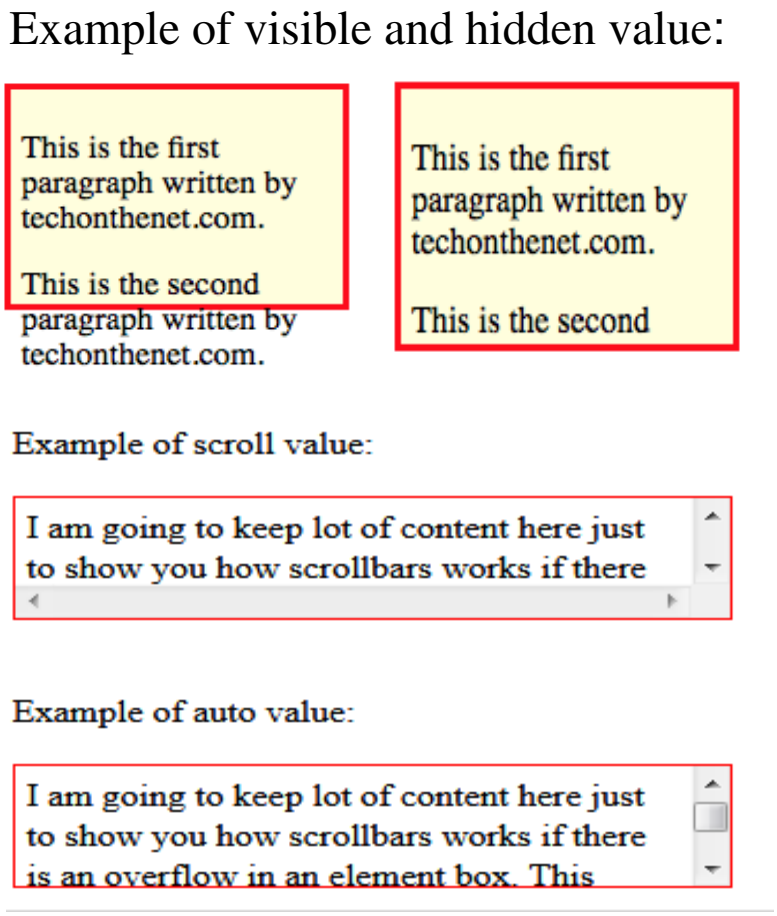


The `overflow` property (cont.)

property	description
overflow	specifies what to do if an element's content is too large; can be auto, visible, hidden, or scroll

Values of overflow property

- visible: content is not clipped when it proceeds outside its box. This is the default value of the property.
- hidden: overflowing content will be hidden.
- scroll: similar to hidden except users will be able to scroll through the hidden content.
- auto: if the content proceeds outside its box then that content will be hidden whilst a scroll bar should be visible for users to read the rest of the content.



- The overflow property specifies what should happen if content overflows an element's box.
- This property specifies whether to clip content or to add scrollbars when an element's content is too big to fit in a specified area.
- **Note:** The overflow property only works for block elements with a specified height.

Value	Description
visible	The overflow is not clipped. It renders outside the element's box. This is default
hidden	The overflow is clipped, and the rest of the content will be invisible
scroll	The overflow is clipped, but a scroll-bar is added to see the rest of the content
auto	If overflow is clipped, a scroll-bar should be added to see the rest of the content

Positioning

- The position property specifies the type of positioning method used for an element (static, relative, fixed or absolute).
- There are four different position values:
  - static
  - relative
  - fixed
  - absolute

property	value	description
position	static	default position
	relative	offset from its normal static position
	absolute	a fixed position within its containing element
	fixed	a fixed position within the browser window
top, bottom, left, right	positions of box's corners	