# Layouts
## Floats+Position

ART326

# Float + Position

# Webpage Layout

Unlike print design, where elements can be placed anywhere, web designers use HTML + CSS.

By default, elements flow from top to bottom of browser window in one column.

How do we control placement of elements?

CSS

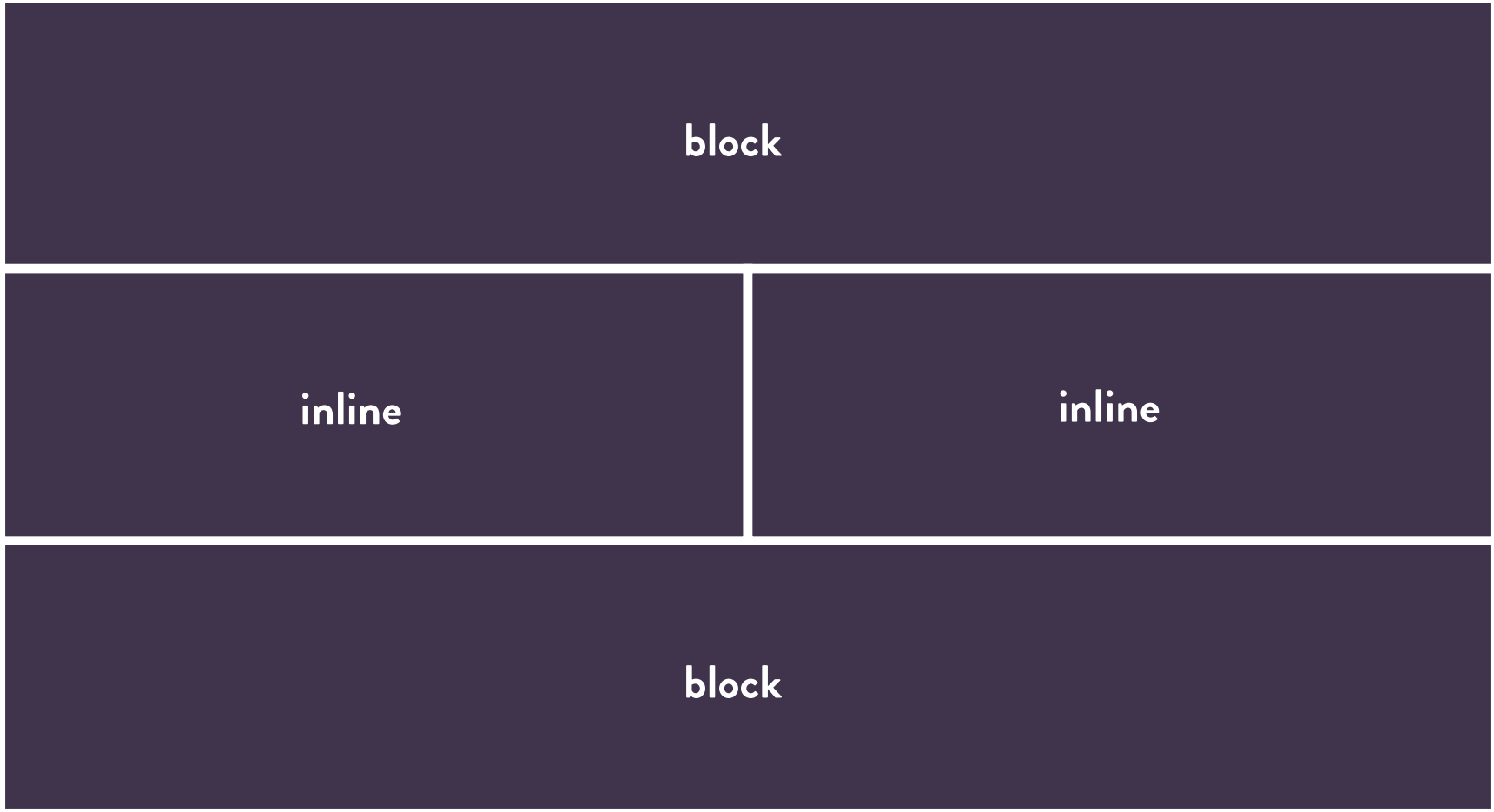
# Normal Flow

Text elements appear on page in the same order as your markup.

**Block elements** (p, div, h1) fill up the entire width of either the browser window, or containing element. Begin on new line.

**Inline elements** (img, strong, em, span) line up next to one another to fill up the block elements. Begin on same line.

Source: *Learning Web Design, Jennifer Niederst Robbins*




The normal flow (above) can be changed or organized in different ways using floating + positioning.

# Floating + Positioning

CSS Methods for breaking out of the normal flow and arranging elements on a page.

Floating moves an element to the left or right, and the following elements wrap around it.

Positioning is a way to specify the **exact** location of an element on page.

Source: *Learning Web Design, Jennifer Niederst Robbins*

# Floating

Moves an element to the far left or right of the element that contains it.

Elements after the floated element wrap around it.

Used to create multicolumn layouts, navigations from list items, and floated images.

**Example 1:** http://htmlandcssbook.com/code-samples/chapter-15/float.html

**Example 2:** http://htmlandcssbook.com/code-samples/chapter-15/using-float.html

# Points to Remember

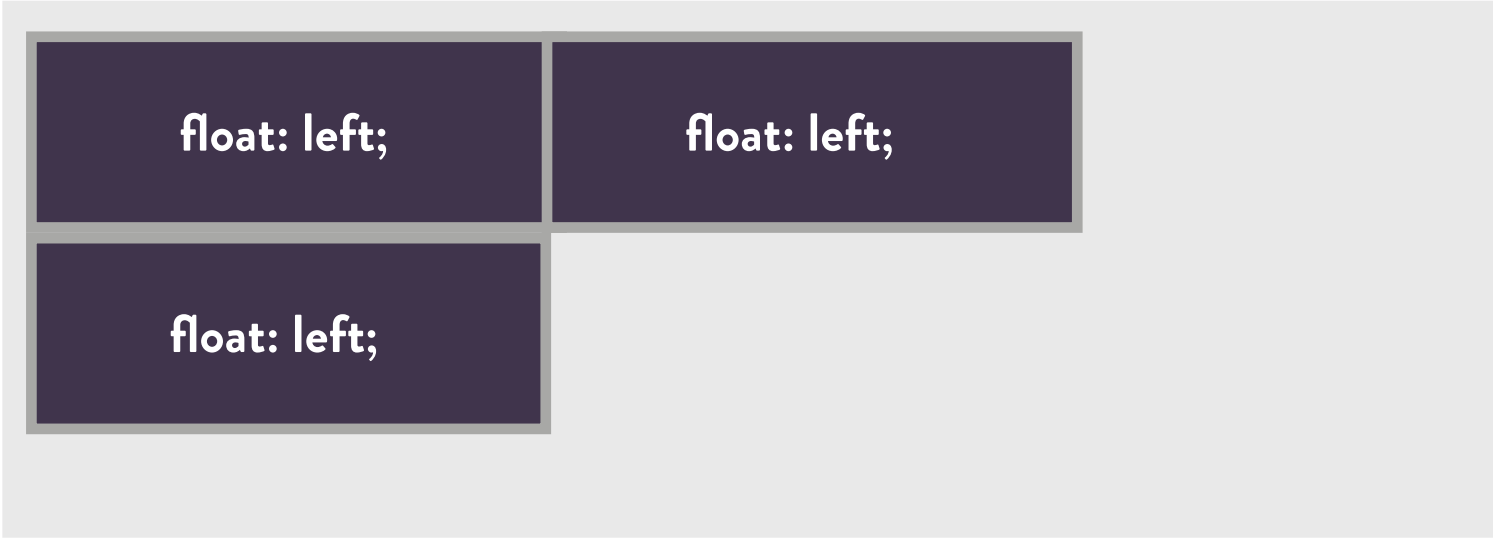
01. Always provide a width for floated elements.
02. Elements do not float higher in the markup than where you inserted them.

Source: *Learning Web Design, Jennifer Niederst Robbins*

# Floating Multiple Elements

Elements floated to the same side line-up.

If one cannot fit, it moves to the next line.



# Clearing

To turn off the subsequent floating, and return the layout to "business as usual," clearing is used.

The clear property forces an element to ignore the float and start on the next available line.

clear: left;

clear: right;

clear: both;

**Example:** http://htmlandcssbook.com/code-samples/chapter-15/clear.html

Source: *Learning Web Design, Jennifer Niederst Robbins*

# Float Border/Background Fix

One issue to be aware of when using floats, is that borders and background colors on parent containers, won't stretch to fill the entire parent container.

**Example:** http://htmlandcssbook.com/code-samples/chapter-15/float-problem.html

To fix this issue, "**overflow: auto;**" and "**width: 100%;**" can be added to the parent containing element to stretch it.

**Example:** http://htmlandcssbook.com/code-samples/chapter-15/float-solution.html

**More information:** http://www.quirksmode.org/css/clearing.html

# Positioning

Elements can also be positioned:

Relative: **relative** to where they would normally appear.

Absolute: or removed from the flow and placed at an exact, **absolute** location.

Possible values for the position property include “static, relative, absolute, fixed, inherit”

Source: *Learning Web Design, Jennifer Niederst Robbins*

# Position Values

**Static:** default; normal positioning.

**Relative:** Moves relative to normal position. The space the element would have occupied is preserved.

**Absolute:** Removed from normal flow and positioned relative to the containing element.

**Fixed:** Removed from normal flow and stays in one position even when the document scrolls

Source: *Learning Web Design, Jennifer Niederst Robbins*

# Position Properties

**Offset properties:** top, right, bottom, left

Defines the distance the element should be moved away from the respective edge.

Negative values are also used to move the element in the opposite direction.

```
.topright {
    position: absolute;
    top: 8px;
    right: 16px;
    font-size: 18px;
}
```


Sources: w3schools.com, *Learning Web Design, Jennifer Niederst Robbins*

# Relative Positioning

Moves relative to normal position.

The original space is preserved.

Overlap happens.

---

```
em { position: relative; top: 15px; color: blue; }

<p>I want one word a little <em>lower</em> than the rest.</p>
```

I want one word a little lower than the rest.

**Example:** http://htmlandcssbook.com/code-samples/chapter-15/position-relative.html

# Absolute Positioning

Removed from normal flow and positioned relative to the containing element.

**No influence on surrounding elements.**

---

```
em { position: absolute; top: 25px; color: blue}

<p>I want one word a little <em>lower</em> than the rest.</p>
```

I want one word a little than the rest.
lower

Example: http://htmlandcssbook.com/code-samples/chapter-15/position-absolute.html

# The “Containing Box”

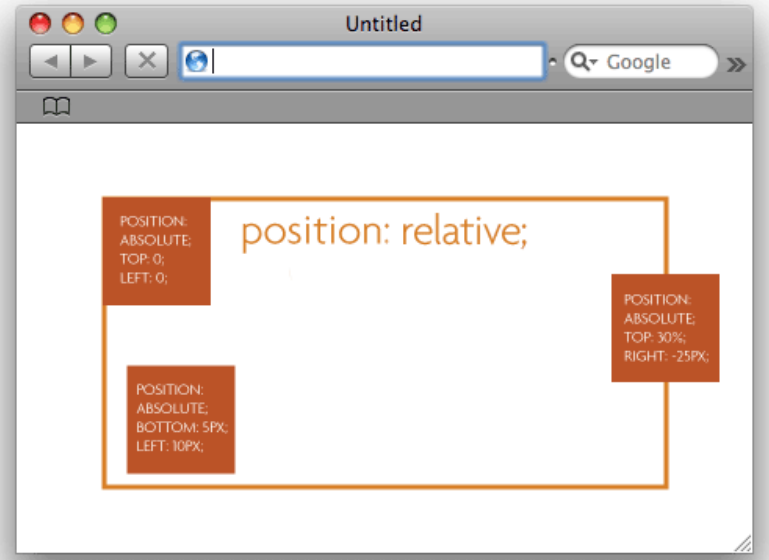
The box the element is being positioned to.

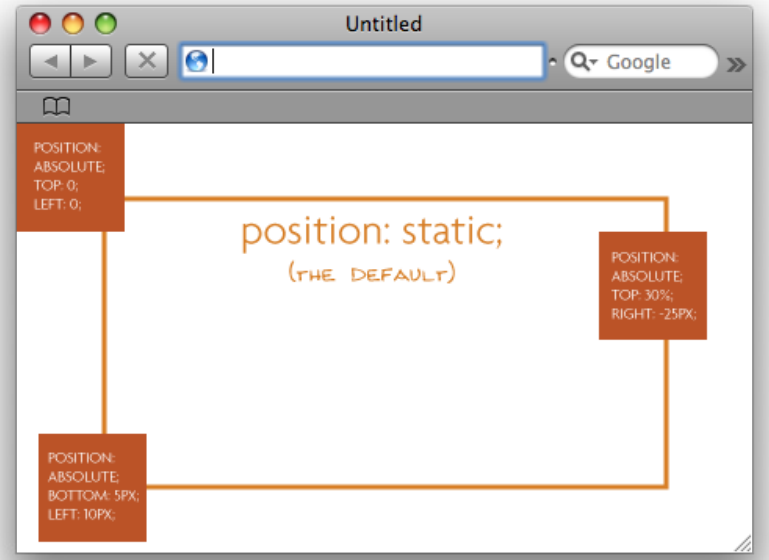
If the positioned element is contained in another positioned element it will use that as the container.

Otherwise, it places the items relative to the initial contained block, the html element.

# Absolute Positioned Layouts

To absolute position elements to a containing box, “position: relative” must also be put on containing element.”




Source: *csstricks.com*

# Z-Index

Because absolutely positioned elements can overlap one another, z-index in needed.

Items stack in the order they appear in the markup.

The z-index is a number you can use to bring an item to the front. The higher the number, the higher the element will appear.

# Fixed Positioning

Works similar to absolute positioning.

The main difference is that the offset values are always relative to the browser window.

What this means it that the positioned element will always stay in one position even when the page scrolls.

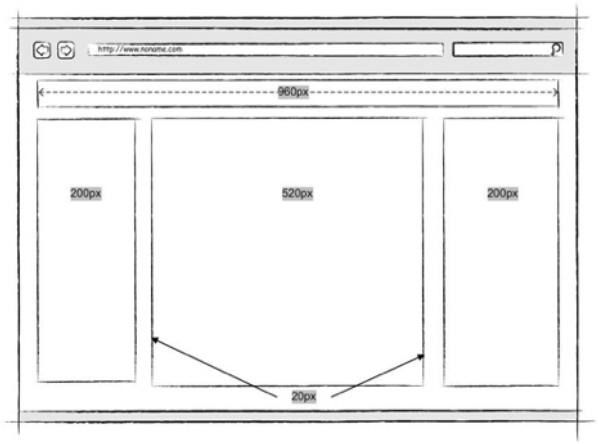
**Example:** http://htmlandcssbook.com/code-samples/chapter-15/position-fixed.html

# Layout

# Fixed-Width Layouts

Keep the content of a particular width, measured in pixels, regardless of the window size.

Has a wrapper or container that everything fits inside.




**Example:** http://htmlandcssbook.com/code-samples/chapter-15/fixed-width-layout.html

Source: *Learning Web Design, Jennifer Niederst Robbins*

Image Source: *http://www.cs.unh.edu*

# Fixed-Width Layouts

## Pros:

Gives the designer the most control over sizes/positioning

Pixel values are most accurate

Control the lengths of measures

Size of images will remain the same relative to the rest of page

## Cons:

Can create big gaps around the edge of the page. Higher res screens can cause pages to look smaller

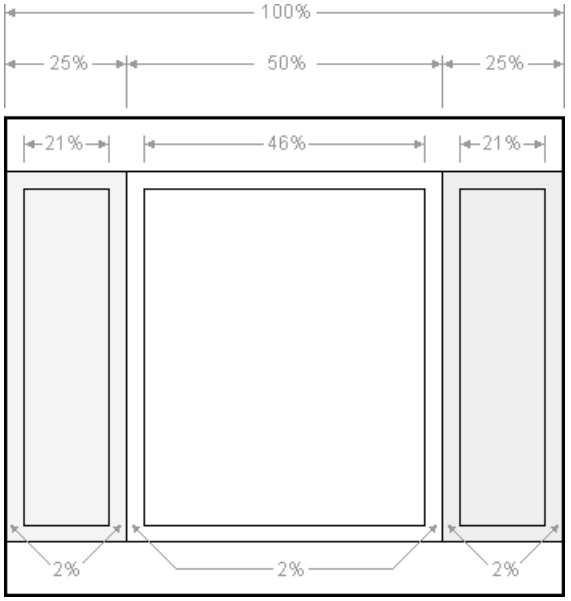
If a user increases text size, text might not fit into space

Design tends to take up more vertical space than liquid layouts

# Liquid/Fluid Layouts

Resizes and adapt to the changing window size.

Known as "**Responsive Web Design**."




**Pro:** Most effective use of screen size.

**Con:** More challenging to design.

**Example:** http://htmlandcssbook.com/code-samples/chapter-15/liquid-layout.html

Source: *Learning Web Design, Jennifer Niederst Robbins*

Image Source: *http://www.cs.unh.edu*

# Liquid Layouts

### Pros:

Pages expand to fill entire browser window; no spaces around edges of page

Small screen sizes contract the design to fit without horizontal scrolling

Design is tolerant of users setting larger font sizes

### Cons:

Without width control, the page can look very different across different devices

Larger windows can result in larger measures

Narrow windows can result in narrow measures