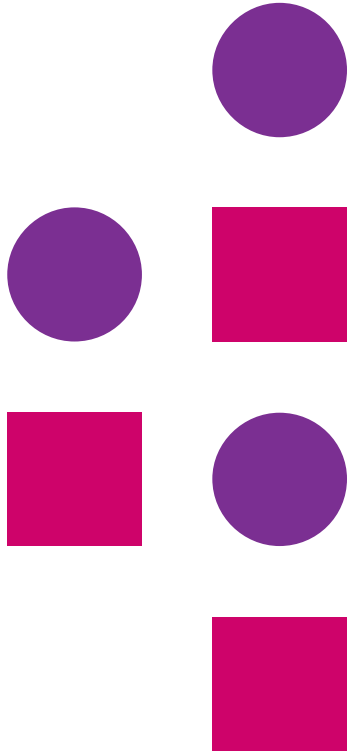
Positioning

Cascading Style Sheets (CSS3)

Table of Content

- Positioning



Positioning

(Cascading Style Sheets 3)

Positioning

- Positioning allows you to take elements out of the normal document layout flow, and make them behave differently
- Example – elements sitting on top of one another, or always remaining in the same place inside the browser viewport

Positioning
(Document flow)

- The block level elements are laid out vertically in the viewport
- Each block level element will appear on a new line below the previous one
- They will be separated by any margin that is set on them

Positioning
(Document flow)

- Inline elements don't appear on new lines
- They are laid out in same line and sit next to each other inside the width of the parent block level element (as long as there is space for them)
- If there isn't space, then the overflowing text or elements will move down to a new line

Positioning
(Why?)

- Do you want to slightly alter the position of some boxes inside a layout from their default layout flow position, to give a slightly quirky, distressed feel?

Positioning
(Why?)

- Or, Do you want to create a UI element that floats over the top of other parts of the page, and/or always sits in the same place inside the browser window no matter how much the page is scrolled?

Positioning
(Why?)

- Positioning makes such layout work possible by allowing us to override the basic document flow behaviour described in earlier slides to produce interesting effects

Positioning
(Types and property)

- There are a number of different types of positioning that applied on HTML elements
  - Static
  - Relative
  - Absolute
  - Fixed
- The position property is used apply specific type of positioning on an element

Positioning
(Static)

- Static positioning is the default for every element
- It just means "place the element into normal flow of the document”
- Nothing special about it

```
.static-pos {
  position: static;
  background: yellow;
}
```

Positioning
(Relative)

- Elements that are relatively positioned remain in the normal flow of the document
- The final position of relative elements can be modified
- Such elements can overlap other elements on the page

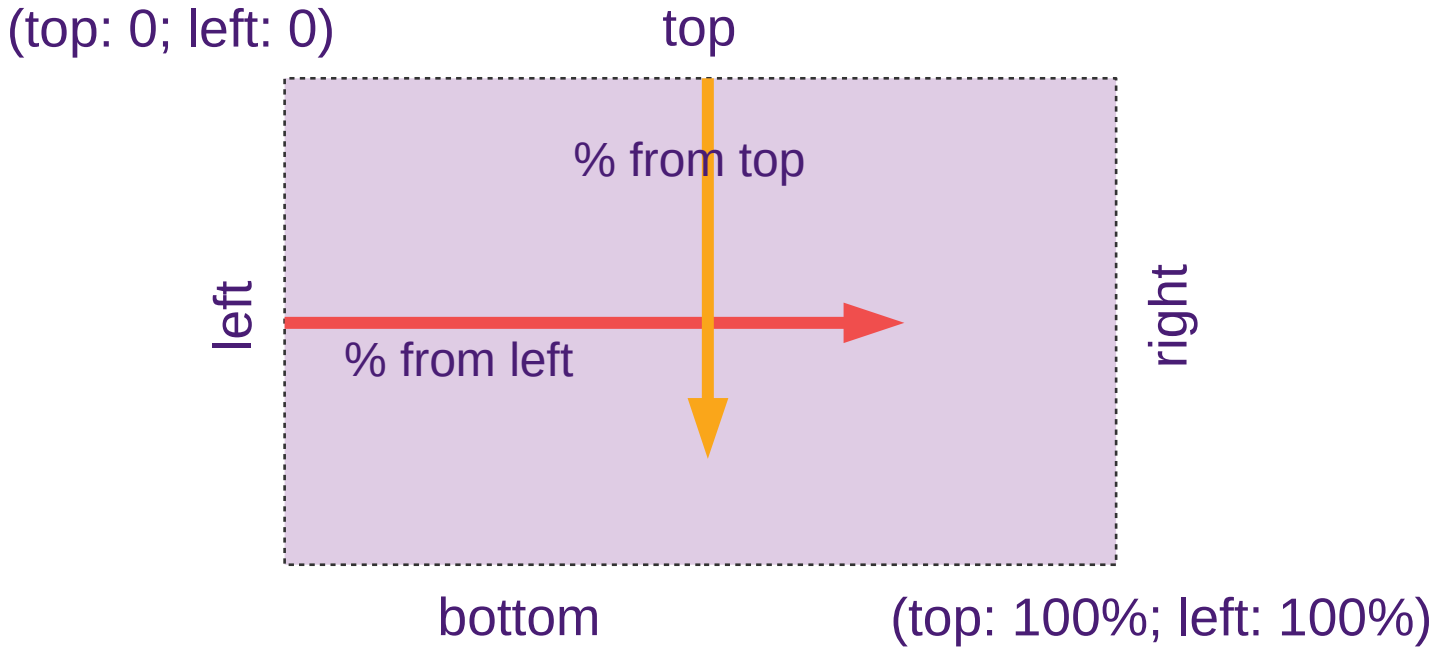
```
.relative-pos {
  position: relative;
  background: skyblue;
}
```

Positioning
(Relative)

- “top”, “bottom”, “left”, and “right” properties are used along with position to specify exact position of element
- Unit of above property values could be pixels, mm, rems, % etc

```
.relative-pos {
    left: 100px;
    top: 100px;
}
```

Positioning
(Relative)



Positioning
(Absolute)

- An absolutely positioned element no longer exists in normal document layout flow
- The absolutely positioned element is positioned relative to its nearest positioned ancestor
- If a positioned ancestor doesn't exist, the initial container (by default HTML element) is used
- Absolutely positioned elements are displayed in separate layer called stacking context

Stacking Context

- Stacking context is a three-dimensional conceptualization of HTML elements along an imaginary z-axis relative to the user, who is assumed to be facing the viewport or the webpage
- HTML elements occupy this space in priority order based on element attributes

Stacking Context

- Stacking context contains stack of layers
- This can be a root stacking context, as created by the html element
- Or it can be a local stacking context, as created by specific properties and values

Stacking Context

- Children of a stacking context are painted from bottom to top in the following order

Stacking Context

- Stacking contexts can be contained in other stacking contexts, and together create a hierarchy of stacking contexts
- Each stacking context is completely independent from its siblings
  - Only descendant elements are considered when stacking is processed
- Each stacking context is self-contained
  - After the element's contents are stacked, the whole element is considered in the stacking order of the parent stacking context

Positioning
(Absolute)

```
.absolute-pos {
    position: absolute;
    background: lightgreen;
}
```

Positioning
(Absolute)

- Since, absolutely positioned elements are displayed in stacking context, they can be used to create isolated UI features that don't interfere with the position of other elements on the page
- Example
  - Popup information boxes
  - Control menus
  - Rollover panels

Positioning Context

- Though, absolutely positioned element is nested inside the <body> in the HTML source
- In final layout, it is positioned N pixels away from the top and left of the edge of the page, which is the <html> element
- This is more accurately called the element's positioning context

Positioning Context

- The positioning context can be modified for absolutely positioned element
- This is done by setting relative positioning on parent (or ancestor) element

```
.parent {
  position: relative;
}
.child {
  position: absolute;
}
```

Positioning
(Fixed)

- The element is removed from the normal document flow; no space is created for the element in the page layout
- It is positioned relative to the screen's viewport and doesn't move when scrolled
- Its final position is determined by the values of top, right, bottom, and left

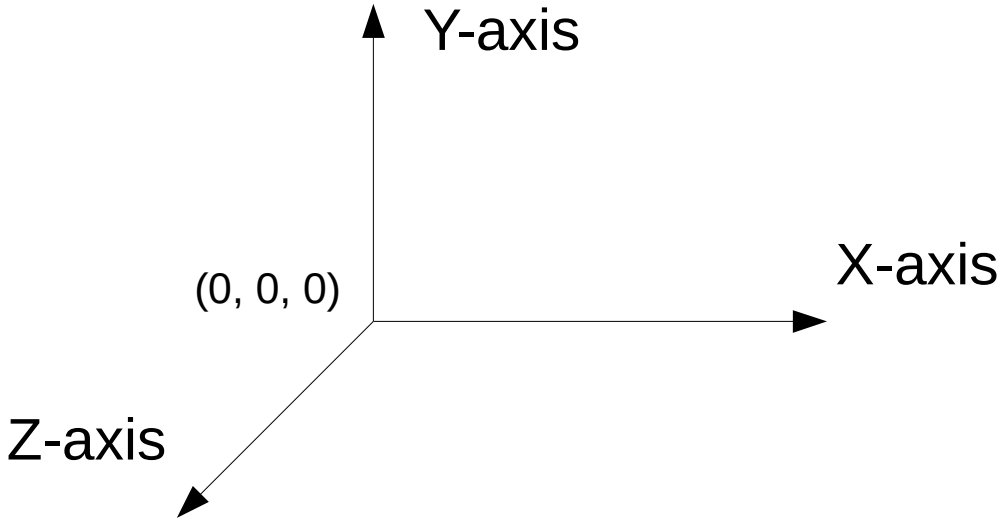
Positioning
(Fixed)

- New stacking context is created for fixed elements
- When an ancestor has the transform, perspective, or filter property set to something other than none, that ancestor is used as the container instead of the viewport

```
.box {
  position: fixed;
}
```

z-index

- "z-index" is a reference to the z-axis
- Z-axis is an imaginary line that runs from the surface of your screen towards your face



z-index

- When elements start getting overlap, z-index determines which elements appear on top of which other elements
- z-index values affect the position of elements on z-axis
- Positive (larger) z-index move elements higher up the stack and negative (smaller) values move them lower down the stack
- By default, all elements have a z-index of auto, which is effectively 0

z-index

- z-index only accepts unitless index values
- Elements with negative z-index are not displayed

```
.front {
  z-index: 3;
}
.middle {
  z-index: 2;
}
.back {
  z-index: 1;
}
```

WSA | Forward looking IT finishing school

Web Stack Academy (P) Ltd

#83, Farah Towers,

1st floor,MG Road,

Bangalore – 560001

M: +91-80-4128 9576

T: +91-98862 69112

E: info@www.webstackacademy.com