Website Construction
Position Is Everything

Frédéric Haziza <daz@it.uu.se>

Department of Computer Systems
Uppsala University

Summer 2009
First, the online demo
Normal flow = Tree order

### HTML

```html
<div id="header"></div>
<div id="mainpage"></div>
<div id="footer"></div>
```

### CSS

```css
body { background-color: black; padding: 10px; }
#header { background-color: yellow; height: 100px; }
#mainpage { background-color: green; height: 300px; }
#footer { background-color: blue; height: 50px; }
```
Normal flow
Floating an element

**HTML**

```html
<div id='header'></div>
<div id='mainpage'>
  <div id='sidebar'></div>
  <div id='content'></div>
</div>
<div id='footer'></div>
```

**CSS**

```css
#sidebar { float:right; margin:20px; width:200px; height:200px; }
#content { background-color: white; height: 50px; }
```
Floating an element

Removes from the flow and pushes to the parent edges.

CSS

**float**: right, left, none, inherit
Clearing an element

**HTML**

```html
<div id='header'></div>
<div id='mainpage'>
  <div id='sidebar'></div>
  <div id='content'></div>
</div>
<div id='footer'></div>
```

**CSS**

```css
#sidebar { float:right; margin:20px; width:200px; height:200px; }
#content { clear:right; background-color:white; height:50px; }
```
Clearing an element

Aligns the element under what was floated ahead
(follows the tree order and normal flow)

CSS

clear: right, left, both, inherit
No, not as a layout mechanism

- CSS can be cached, table markup cannot
- Accessibility issues
- Presentation is meshed within content
- Load time
- Semantically wrong: it’s not tabular data
Showing/Hiding

- **visibility**: visible (default), hidden
- **display**: block, inline, none, ...

Content doesn’t fit the box?

- **overflow**: visible, hidden, scroll, auto, inherit
  - the content might be clipped or not
  - the scroll bars might be visible or not
Positioning

**position: relative**
- *shifted* display (by **top**, **bottom**, **right** or **left**)
- still takes the same original place in the flow
- limits the scope for other absolute positioned descendants

**position: absolute**
- absolutely positioned
  - w.r.t the closest relative-positioned ancestors
  - w.r.t the browser window otherwise

**position: fixed**
- absolutely positioned with respect to the browser window
Higher **z-index** means higher position in the z-axis. Can be any integer (positive and negative).

From the W3C

All positioned descendants with `z-index: auto` or `z-index: 0`, in tree order.

For those with `z-index: auto`, treat the element as if it created a new stacking context, but any positioned descendants and descendants which actually create a new stacking context should be considered part of the parent stacking context, not this new one. For those with `z-index: 0`, treat the stacking context generated atomically.