Sir John Harrington- invented the flush toilet, was suppose to prevent disease(no more smell) queen did not like the noise which announced her taking a shit. She would rather have disease than have this new solution. Took another two centuries before new inventers came along and refined this invention.

Why is this important? Harrington called his invention Ajax- called his book the metamorphisis of ajax.. *laughs*

Jesse James Garret- user experience designer, trying to create effective web application, called his new invention Ajax.

Ajax Changed Everything-

- word processing- binary propriety, stand alon, shared logic, eventually became software that runs on pc
- what you see is what you get

Run Off- a language developed for cts time sharing for MIT, influenced UNix. Two kinds of lines, period= command line. Lines with no period would be raw text. Could have descriptice terms in commands.

- had influence, **GML**
    - take all display oriented stuff out, using colon, period to turn off
    - only talking about structure of document
    - used tags
        - :eol
        - ::ol
        -

Brian Reid- Scibe- element language, more evolution of language, have block of text with nested stuff after

- special forms could begin and end
- wouldnt have to worry about closing out of the contents
- elogant way of creating documents

Many people looked at WWW and thought it didnt have what it takes.

- web standards were grown out of these conditions
- it was not designed to do all of the Ajax stuff

HTML

- Huge improvment of SGML
- more simple
- more resilient. THe Dark Side-- browser makers got into competion- who could make sense out of the worse HTML. Enabled bad coding, nobody wanted a browser that

couldnt display a particular page. no enforcement of browsers, deveolpers made crap, vicious loop
- authors have virtually no control over presentation- sometimes we want pixel perfection of what gets displayed. set of tags too much, overloading. too many classes or ids
- too limited: classitis and iditis
- It did not anticipate applicaitons beyond simple document retrieval-- everything we do today is basically out of scope of what it was made to do.
- Web page is not a "page"
    - the thing that WWW calls a page isnt really a page at all. it is a scroll
    - the scroll is an ancient technology-- we have a site that calls itself a set of pages, but is really just a series of scrolls--this may not be of importance, just worth mentioning

CSS- cascading style sheets
- unhealthy seperation of structure and presentation- websites very often you have a group of people who are trying to work collaboratly, css was designed for paper printing, does not fit dynamic world we live in
- long, fragile lists of self-contradictory rules- error prone
- each rule has two parts:
    - selector
    - deselector
- difficult to understand. difficult to use

Five Problems with CSS
- Lack of modularity- independent content we want on the same page, but high likely hood that they will interfere with one another, overwritten rules; hard to detect. possible to impose modularity but difficult
- selector management is complicated-
- declarations are too weak for mocdern web apps- layouts are too complex today: want to be able to scale screen sizes, was not intended to do this (mobile)
- not intended for dynamic content
- it is unimplementable. its all about the quirks- browser makers find it hard to do what it says efficiently. not designed with the awareness of how browsers work and how apps run

Devs seem to be all for it-- "you just dont understand it" Crockford says it is awful because people get invested into, than are locked into it

if CSS was all there was the web would have been replaced by now

George F Colony said in 2000 that the internet would have been replaced by now. replaced with a new design that transports programs, which would hold the value "the exacutable internet" programs instead of static documents.

The reason it did not turn out the way that Colony said is because of **Java Script**

The Document Object Model

- the "dom"
- it is what most people hat when they say they hate JS
- the browser's API- it provides JS for manipulating documents
- Brendan Eich, Netscape- inventer of JS
    - influenced by a book on hyper card
- Scott Isaacs, Microsoft
    - In the original Nestscape model, not all elements were scriptable, microsoft- everything was scriptable

Scripted Browser FLOW--PAINT--SCRIPT--EVENT

document.write

- allows JS to produce HTML text
- before onload:insrts HTML text into the document
- after onload: uses HTML text to replace the current document
- causes security problems- not recommended, but alot of systems are dependent on this

<script></script>

- script files can have a big impact on page loading time- request response execute slows down, if at the bottom images load at the same time

1. place <script scr> tacs as close to the bottom of the body as possible. (also, place CSS <link> as high in the head as possible.)
2. minify and gzip script files.
3. reduce the number of script files as much as possible

Microsoft vs Netscape- mistakes were made. W3C attemted to moderate- for developers it was like living with divorced parents, trying to code for both to make them happy. they were working in different directions

HTML- a language for describing trees-- Document Tree Sctucture

- root- document node

- h1 vs h2 are both simple containers, just separate containers

- each node contains pointers

- no direct pointers for the middle children, but they have sibling pointers to previous siblings

Using recursion, follow the firstChild node, and then the nextSibling nodes

```
function walkTheDOM(node, func) {
  func(node);
  node = node.firstChild;
  while (node) {
    walkTheDOM(node, func);
    node = node.nextSibling;
  }
}
```

- passes node and function
- first will go to first child than all siblings of that child
- for each node we find recursive call walkTheDom to get to its children and its siblings

Retrieving Nodes- these return node lists, live querries on the dom structure

- document.getElementById(id)
- document.getElementsByName(name)
- node.getElementsByTagName(tagName)

CSS chose to use hyphens even tho program languages typically use hypons for subtraction. CSS used hypons, JS used camel case. Crockford says damn them! The lease compatible way to do it

innerHTML

- the W3C standard does not provide access to the HTML parser.
- all A browsers implement Microsoft's innHTML property
- Security hazard

Events

- the browser has an eventdriven single threaded asynchonous p[rogramminmodel
- targeted to particular node

Bubbling- means that the event is given to the target, and then its parent, and then its parent, and so on until the event is cancled

Why Bubble?

- suppose you have 100 draggable objects

- you could attach 100 sets of event handlers to those object
- or you could attach one set of event handlers to the container of the 100 objects

Events

- the browser has an eventdriven single threaded asynchonous p[rogramminmodel
- targeted to particular node

Bubbling- means that the event is given to the target, and then its parent, and then its parent, and so on until the event is cancled

Performance

- touching a node has a cost
- styling can have a big cost
- reflow can have big cost
- repaint can have bif cost
- random things like nodelist can have big cost
- *in mostt apps JS has small cost*
- speed tracer on chrome is a great tool that lets you see what you are doing

All these concepts were being created in great competition. Then netscape self destructs. Microsoft abandons Internet Explorer in favor of Avalon.

- microsoft IE team is pulled, they begin developing .net, there new net platform

Then AJAX happens- the web blindsides Microsoft again

Microsoft's Ajax

- JScript- high fidelity clone of JavaScript
- generalized document model-all elements are scriptable
- XMLHttpRequest

Microsoft declared victory and left the field- the browser war is over

JS would have died with Netscape, the Ajax revolution succeeded because of the goodness of JS

2000- WHy did Ajax take five years?

- the web was full of bugs and unstable- once you ship a buggy browser you can't expect everyone to update, bug list grows, takes a long time to fix
- most people won't install new browsers
- they replace their software when they replace their hardware
- some companies lock their machines down

- it took years to flush Netscape 4 and IE4 and IE5 out of the market
- It took years to stabilize the web after the disruption of the browser wars

In the browser wars browser was a driver of innovation- now it is a frustration to innovation

Web time used to mean things change really fast- as we get more mainstream, early adopters become less sig, every mistake ever made is still out there because people still use old software

With Ajax, innovation shifted from browser to makers of web apps, JS can transform the DOM (one of the worst APIS) into something pleasant. Ajax libraries are fun and easy to make

Ajax libray

- portability- hard to write a program that writes on all browsers, this provides correction of that, api that works consistently on all browsers.
- correction
- model- new object, inheritance, processing model. may be new or inherit from other language
- wdgets-- ready made bundles of functionality instantly do good things

Crockford predicted

- the market would reduce the set of viable libraries to 1 or 2
- didnt happen
- ajax app market is large enough to support a large collection of ajax libraries- worlds biggest dev market
- the shcakeout will not happen until Mashups become viable

JSLint only objective standard for JS code quality

Ajax is no longer new or special, just the way we make web apps now- dominant app platform

The Ajax community has mined all the laten potential of the 20th century browser platform- things the makers did not put into initially. We are now at the limit of innovation

IE6 problem

- for many years IE6 was the best browser in the world
- more recently became worst

HTML5- step in wrong direction, too complicated says crockford