

## Client Pull and Server Push

Delivering Web content that changes over time.

---

---

---

---

---

---

---

---

## Documents that change at regular intervals

- Many possible applications:
  - stock tickers
  - web based chat
  - collaborative applications

---

---

---

---

---

---

---

---

## Alternatives

- Custom servers and clients
  - not just web server and browser
- Client Pull
  - browser asks the server for new copy of the document at regular intervals
- Server Push
  - server forces new versions of the document on the client at regular intervals.

---

---

---

---

---

---

---

---

## Server Push

- In the old days, you could do server push with Netscape and any server.
  - Server does not send a complete document
    - the TCP connection stays open.
  - The client must be told to render the part of the response it has received.
  - The client must be told to replace the old part with new content.

EIW - Pull and Push

4

---

---

---

---

---

---

---

---

## Multipart Document

- The server tells the browser the document contains many parts.
  - browser gets the first part (and knows when it has the entire *part*).
  - server waits a while, then sends the next part.
  - The next part can be a multipart document as well...

EIW - Pull and Push

5

---

---

---

---

---

---

---

---

## Server Push Now

- Clients don't like multipart documents
  - typically don't render anything until the entire document has been loaded.
- Push is now typically used for custom services (broadcasting, audio-feeds, etc.)
  - requires some special server-client interactions that are not just normal "HTTP".
  - MS has some products that will do this with IE.

EIW - Pull and Push

6

---

---

---

---

---

---

---

---

## Other approaches

- Applet creates connection to the server and can retrieve new messages, send new messages, etc.
- Java (applet) programmer can decide whether to keep connection open (or even whether to use TCP or UDP).
- Applet is responsible for drawing part of the web page.

---

---

---

---

---

---

---

---

## Client Pull

- Browser knows that document should be refreshed after some time interval.
- Each refresh is a new HTTP request.
- HTML META tag supports this.

---

---

---

---

---

---

---

---

## **META**

- Found in the HEAD of a document.
- Supports a variety of functions:
  - define *variables* that are used by search engines.
    - author
    - page generator
  - set document expiration date.
  - include embedded objects (sounds).

---

---

---

---

---

---

---

---

## <META HTTP-EQUIV=

- The **HTTP-EQUIV** attribute tells the *Web Server* to use the **CONTENT** attribute to generate HTTP response headers:

```
<META HTTP-EQUIV="Expires"  
  CONTENT="Tues, 29 Feb 2000  
  23:59:59 GMT">
```

---

---

---

---

---

---

---

---

## HTTP-EQUIV="refresh"

- Tells the browser to automatically replace the document with another document.
  - automatic forwarding
  - client pull

---

---

---

---

---

---

---

---

## Forwarding Example

```
<META HTTP-EQUIV="refresh"
```

```
  CONTENT="5; ← how long to wait (seconds)
```

```
  URL=http://www.foo.com">
```

↑  
New page to load (after waiting)

---

---

---

---

---

---

---

---

## Automatic Refresh Example

```
<META HTTP-EQUIV="refresh"
CONTENT="5">
```

- With no URL specified, the same document is reloaded.

---

---

---

---

---

---

---

---

## Client Pull

- CGI creates a dynamic document set to automatically refresh.
- CGI is run again (next request), this time the document may be different.
- If replacement also contains refresh instructions, the client keeps refreshing forever...

---

---

---

---

---

---

---

---

## Chat using Client Pull

- CGI program sends back a web page that includes refresh header.
  - web page also displays the latest stuff sent by others.
- Client refreshes the page at regular intervals (or whenever user wants).
- Will work, but potential for lots of unnecessary traffic, and response time is not so great...

---

---

---

---

---

---

---

---

## ASP and Client Pull

- To tell the client to refresh the document at regular intervals:

```
<META HTTP-EQUIV="refresh" CONTENT="5">
```



This is just HTML (can be in any document!)

---

---

---

---

---

---

---

---

## ASP Countdown Timer

- When user loads the page:
  - counts down from 10 to 0
  - when count reaches 0, loads a new page ([www.yahoo.com](http://www.yahoo.com))
- Client Pull is used to force the client to keep reloading the page until it gets a page without a refresh header.

---

---

---

---

---

---

---

---

```
<%If Session("count") >= 10 Then
  Session.abandon() %>
<META http-equiv="refresh" content="0;
URL=http://www.yahoo.com">
<% Else %>
<META http-equiv="refresh" content="1">
<% End If %>

<% Session("count") = Session("count")+1 %>
<P style="font-family:arial,sans-serif;
font-size:60pt;
font-weight: bold;">
<%=11-Session("count")%>
</P>
```

---

---

---

---

---

---

---

---

## Stock Ticker Using Client Pull

- Part of a document changes (just part).
- We could have the entire document reload at regular intervals, but this may take too long.
- IFRAME tag (inline frame)

---

---

---

---

---

---

---

---

## Stock Ticker in ASP

- ASP document that includes a refresh header (with META tag)
- The content of the document is just the current stock price.
  - some random number code just to simulate things.
- We can set up any HTML document with an IFRAME that holds this ASP document.
  - it will auto-refresh.

---

---

---

---

---

---

---

---

## ticker.asp

```
<META http-equiv="refresh" content="4">
<%
Randomize()
If Session("eiw")="" Then
  ' First time - start with a random number
  Session("eiw") = 100*Rnd()
Else
  ' generate new value
  If Rnd() < 0.5 Then
    Session("eiw") = Session("eiw") + 10*rnd()
  Else
    Session("eiw") = Session("eiw") - 10*rnd()
  End If
End If
Response.write( FormatNumber(Session("eiw"),2))
%>
```

---

---

---

---

---

---

---

---

## Using IFRAME

```
Stock Price:
<iframe src=ticker.asp
  frameborder=0
  align=top
  marginheight=1
  height=30
  scrolling=no>
iframe tag not supported
</iframe>
```

---

---

---

---

---

---

---

---

## Chat based on Client Pull

- Use Frames
  - top frame has form user can use to submit a line of text to the "chatroom"
  - bottom from shows everything submitted by all users.
    - » set to auto-refresh to the user doesn't need to keep reloading.

---

---

---

---

---

---

---

---

## Chat Main Document

```
<HTML>
<HEAD>
<TITLE>EIW Chat</TITLE>
</HEAD>
<FRAMESET ROWS="20%,*">
  <FRAME NAME=top SRC=chatform.html>
  <FRAME NAME=bottom SRC=chat.asp>
</FRAMESET>
</HTML>
```

Submission Form

set to auto-refresh

---

---

---

---

---

---

---

---

## Chat Submission Form

```
<HTML>
<HEAD>
<TITLE>EIW Chat</TITLE>
</HEAD>

<BODY>
<H3 ALIGN=center>EIW Chat</H3>
<FORM ACTION=chat.asp TARGET=bottom METHOD=POST>
<INPUT TYPE=TEXT WIDTH=40 NAME=new>
</FORM>
</BODY>
```

targets the other frame!

---

---

---

---

---

---

---

---

## chat.asp

```
<META http-equiv="refresh" content="4">
<%
If Request("new")<>" Then
  Application("chatlog") = Application("chatlog")_
    & Request("new") & "<BR>"
End If
%>
<DIV style="border:solid 1 black">
<% Response.Write(Application("chatlog")) %>
</DIV>
```

---

---

---

---

---

---

---

---