Model of HTTP (Client)

- Location for HTTP spec:
  http://www.faqs.org/rfcs/rfc1945.html (HTTP/1.0)
  http://www.faqs.org/rfcs/rfc2616.html (HTTP/1.1)
  http://www.www9.org/w9cdrom/60/60.html (HTTP Next Generation)
- Clients:
  - Send requests to servers (or caches)
    - `GET http://java.csie HTTP/1.1`
      - `GET` can be replaced by `POST`
    - Headers … (e.g.)(date)
      - `Content-Type: text/html`
      - `Cache-Control: max-age=0`
    - (blank line)
    - Data...

Model of HTTP (Server)

- Servers:
  - Respond to the clients (or caches)
    - HTTP/1.1 200 OK (Status)
    - Headers … (e.g.)(date)
      - `Content-Type: text/html`
      - `If-Modified-Since:` (date)
    - (blank line)
    - Data...

Get and Post

- Get:

  `GET http://java.csie/test.cgi?a=b&c=d HTTP/1.1`

  – Parameters in URL.
  – No data content
  – The corresponding form of HTML files

  ```
  <form method="GET" action="test.cgi">
  ```

- Post:

  `POST http://java.csie/test.cgi HTTP/1.1`

  – Parameters `a=b&c=d` in Data Content.
  – The corresponding form of HTML files

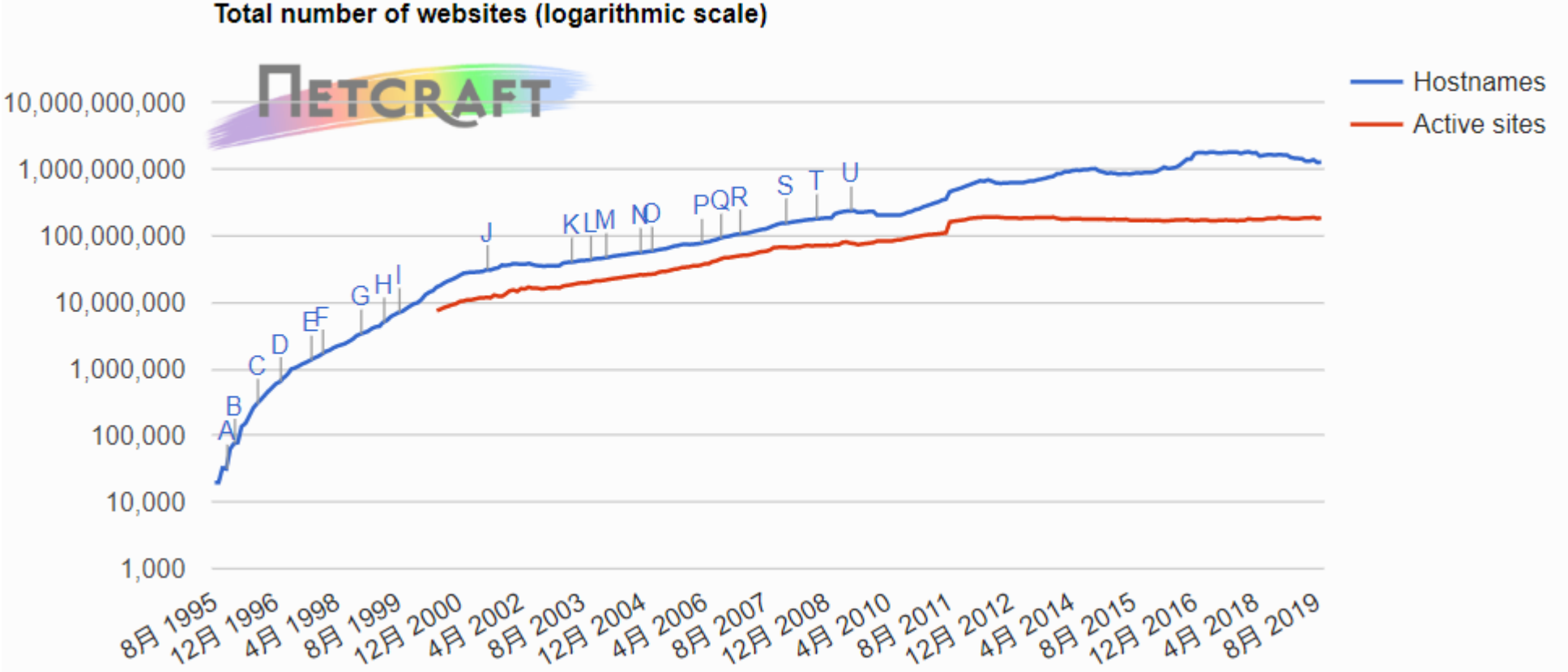
  ```
  <form method="POST" action="test.cgi">
  ```

Get vs. Post

- Security issue:
  - Safer for POST (but not so significant)
- Size of parameters
  - The parameter size for GET is limited to a size, say 4k.
  - The parameter size for POST is unlimited.
- Prefetching (New semantics, due to Google Web Accelerator)
  - For POST, prefetching is not recommended.
    - Prefetching may do the “logout” operation.
  - For GET, prefetching is recommended.

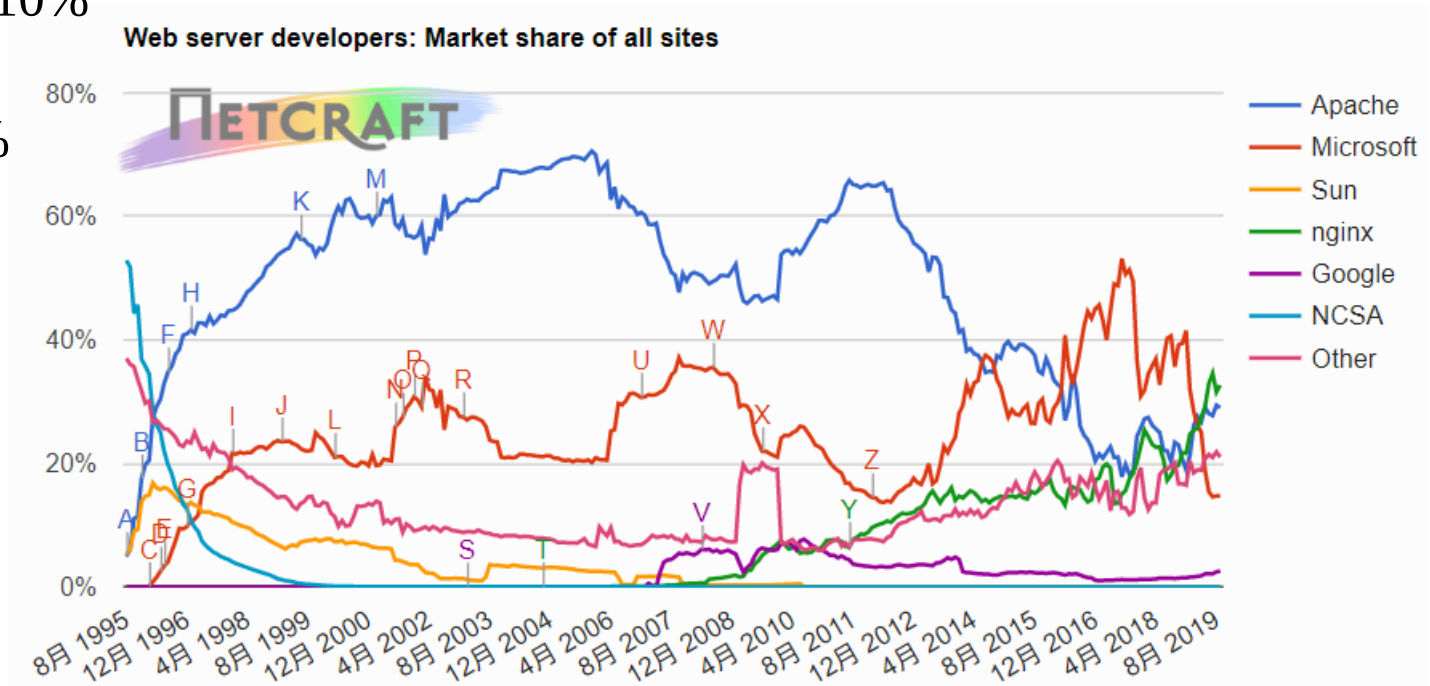
Total Number of Websites

Netcraft:
https://news.netcraft.com/archives/2019/09/27/september-2019-web-server-survey.html

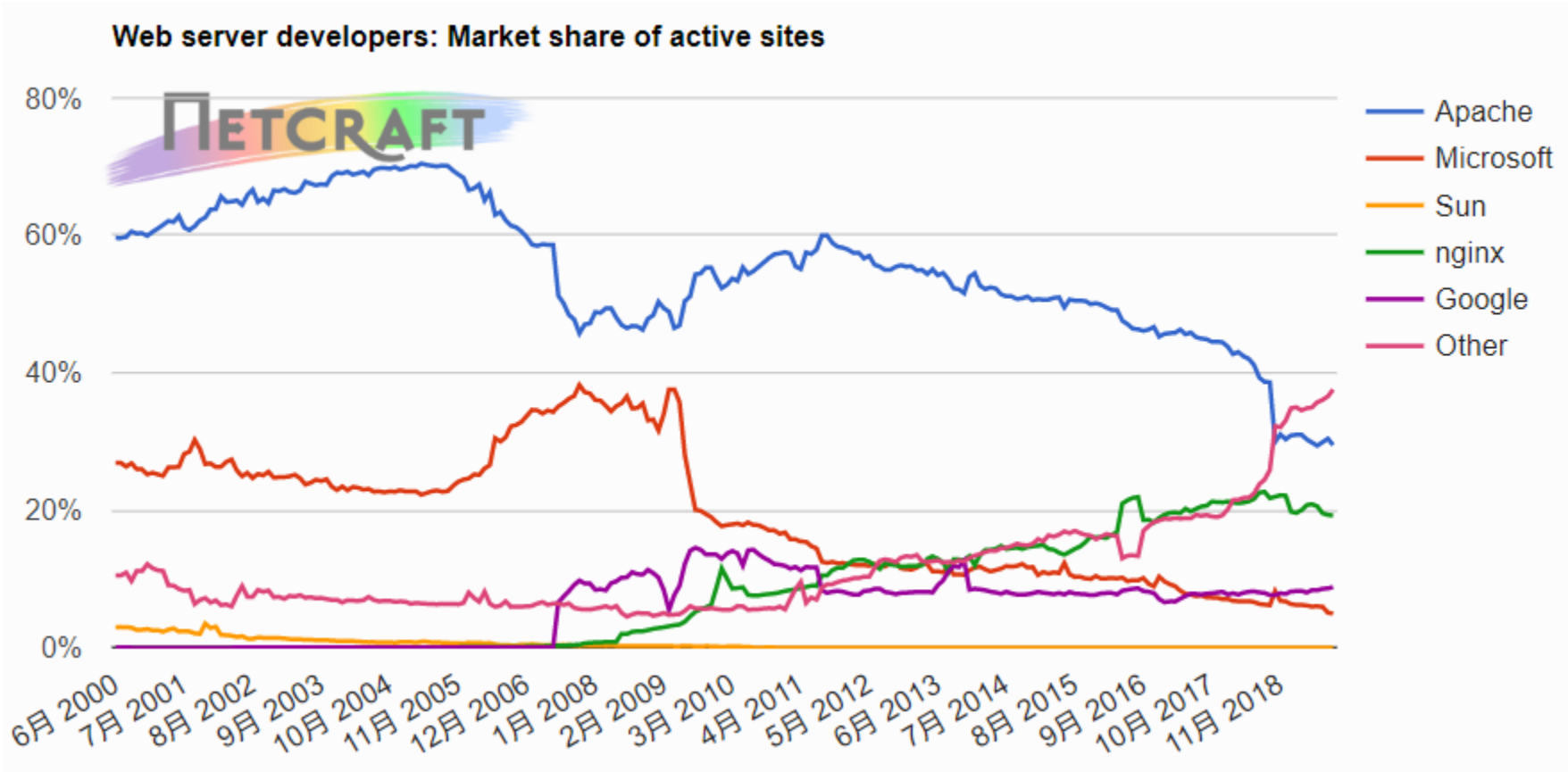


Apache

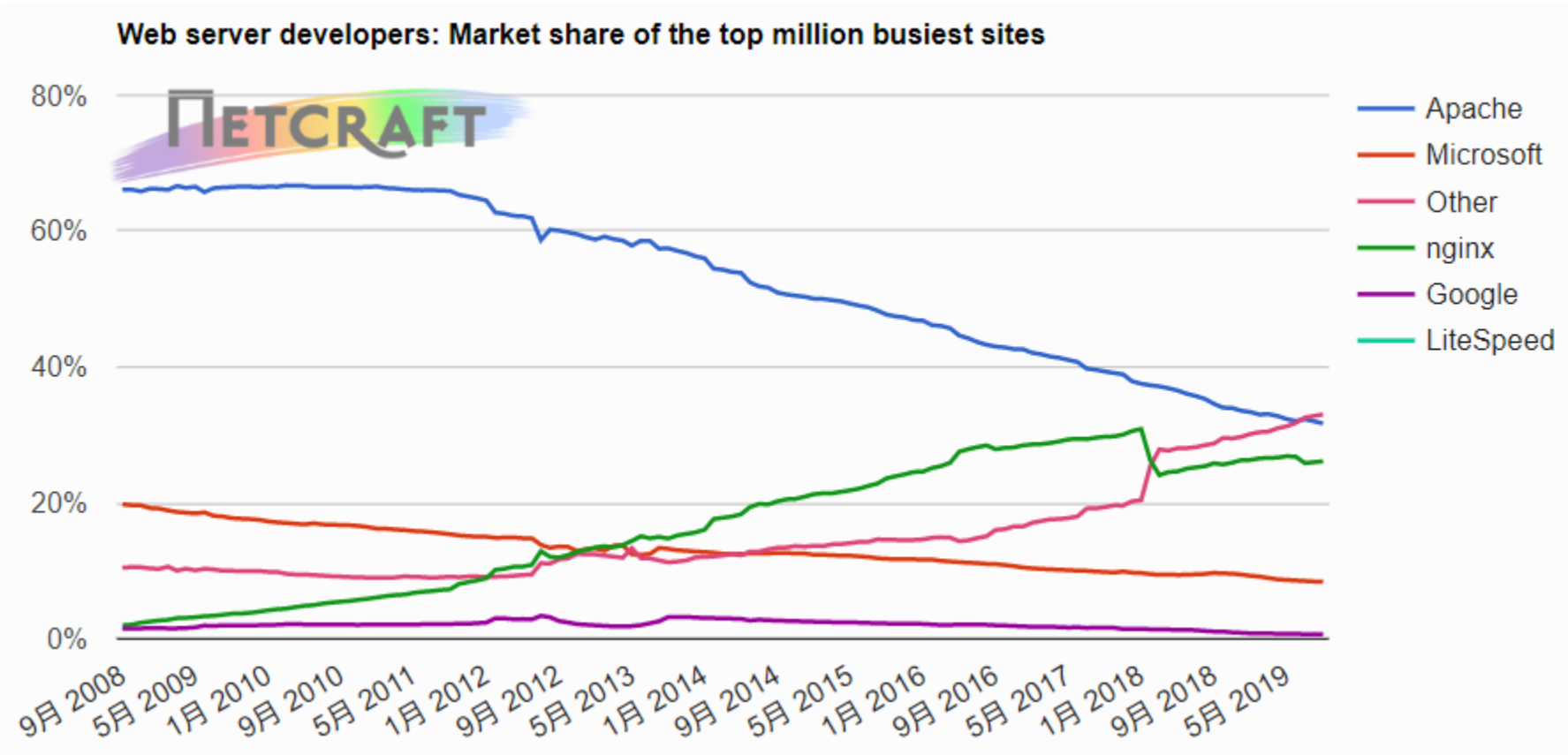
- Location: http://www.apache.org/
- The most popular Web servers. (news.netcraft.com, Feb. 2013)
  - Apache: 44.89%,
  - Microsoft: 23.10%
  - nginx: 16.05%
  - Google: 4.45%
- Free



Web Server Developers (Active Sites)



Web Server Developer (the top million)



CGI Model

On a request,

- If it is a page request, return the page.
- If it is a CGI, do the following
  - Fork a child process
  - Set its “stdin” from client and “stdout” to client.
  - Set environment variables
  - Exec the CGI.

CGI Model (cont.)

- Parameters from HTML:
  - keyword=value&keyword=value&…
    - ‘ ‘ ⇒ ‘+’ escape char ⇒ %xx
  - Using HTTP “GET” method
    - 從環境變數 QUERY_STRING 取出 input string
    - `char* queryString=getenv("QUERY_STRING");`
  - Using HTTP “POST” method
    - 從 STDIN 讀入 input string
    - 以環境變數 CONTENT_LENGTH 決定字數

Environment Variables

- CGI programs 透過環境變數與 http daemon 溝通
- 幾個重要的環境變數
  - QUERY_STRING
  - CONTENT_LENGTH
  - REQUEST_METHOD
    - REQUEST_METHOD="GET" or "POST"
  - SCRIPT_NAME
    - SCRIPT_NAME = "/~icwu/chat/cgi-bin/echo-cgi"
  - REMOTE_HOST
    - REMOTE_HOST="java.csie.nctu.edu.tw"
  - REMOTE_ADDR
    - REMOTE_ADDR="140.113.185.117"
  - AUTH_TYPE, REMOTE_USER, REMOTE_IDENT

Example: Simple I/O

- **Get data from input data**

```
char* length = getenv("CONTENT_LENGTH");
int leng = atoi(length);
fread(buffer, sizeof(char), leng, stdin);
```

- **Output a valid Web page**

```
main() {
  char* s = "Test CGI";
  printf("Content-type: text/html\n\n");
  printf("<html>");
  printf("<body>");
  printf("<h2>%s</h2>", s);
  printf("</body>");
  printf("</html>");
}
```

Example: Output GIF

- **Output a GIF file**

```
main() {
  printf(Content-Length: %d\n”, size);
  printf(Content-type: image/gif\n\n”);
  in = fopen(IMAGE_FILE, “rb”);
  while(1) {
    ch = getc(in);
    if(ch==EOF)    break;
    putc(ch, stdout);
  }
}
```

Caching in HTTP

- Motivation
  - Performance
  - Availability
  - Disconnected operations

==> Relax the goal of semantic transparency, i.e., Caching.

- Eliminate some requests
  - Use the expiration mechanism
- Eliminate some full requests
  - Use the validation mechanism

Ref: [RFC 2068]

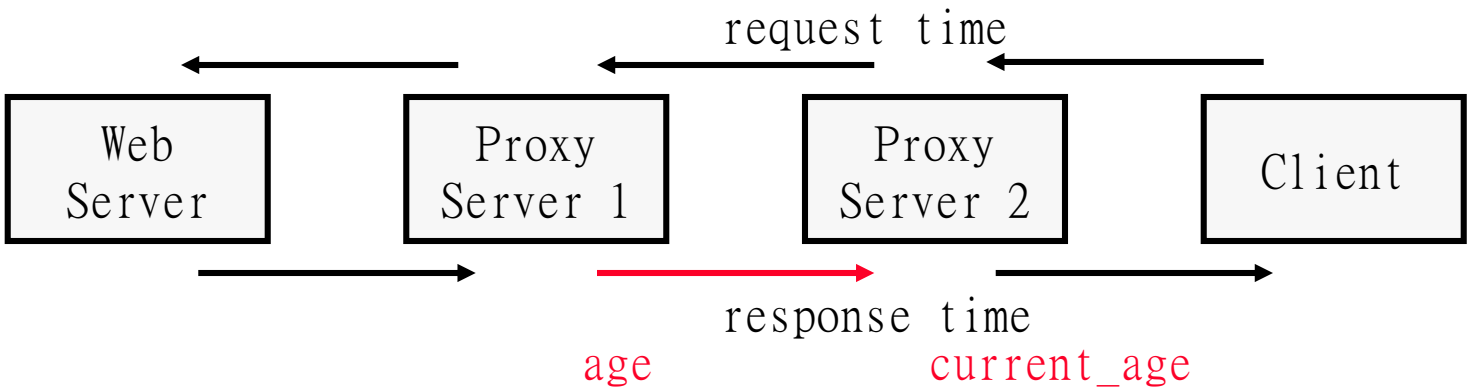
Expiration Model

- The cache can return the page without contacting the server before expiration.
- Server-Specified Expiration
  - Specify when to expire
    - Expires:
  - Problem: the clocks are not the same.
- Heuristic Expiration
  - Use heuristic method based on
    - Last_modified:
  - Problem: the clocks are not the same.
- Expiration Calculation (next page)
  - Age Calculation

Expiration Calculation

- Algorithm
  - freshness_lifetime = max_age_value (from server header), or
  - freshness_lifetime = expires_value (from server header) - date_value (from server header)
  - is fresh? ==> freshness_lifetime > current_age (see next page)
- Others:
  - Cache-Control: max-age=0
    ==> force to validate the object again
  - Cache-Control: no-cache
    ==> force to obtain a new copy

Age



- Age: the time from server to now
  - Current_age = age + propagation_time + resident_time
- What is the problem?
  - Each server has different timer.
- Principle:
  - Be as CONSERVATIVE as possible

Age Calculation

- Age calculation algorithm in cache:
  - corrected_received_age = max (age_value (from server header), response_time - date_value (from server header))
  - response_delay = response_time - request_time
  - corrected_initial_age = corrected_received_age + response_delay
  - resident_time = now - response_time;
  - current_age = corrected_initial_age + resident_time
- Note: if date_time > request_time, probably not first-hand

Validation Model

- If validator is matched, return 304 (not modified) and no entity-body.

Validator

- Last-modified dates (quite common)
  - Last-Modified (from server header) and If-Modified-Since (from client header)
  - For example, RSS (Really Simple Syndication) tool.
- Entity Tag Cache Validators
  - ETag: … (used when the date is inconvenient.)
- Weak and Strong Validators
  - Strong Validator: the entity must be the same.
    - For sub-ranges, must use this.
  - Weak Validator: the entity is “semantically” the same. E.g.,
    - A counter won’t be changed in days or weeks.
    - A file should not be changed within one second.

Keep Alive

- For a homepage,
  - Open/Close a connection for each small file (like jpg, gif, etc)
- Problem:
  - Inefficient
- Solution
  - Allow many file requests to use one connection.

- More techniques:
  - Mix with pipeline. (Access more gif files while reading HTML files.)