Open Source Software Project Development

Dr. T.Y. Wong

Weeks 12 - 13

Web Application Examples and Related Skills

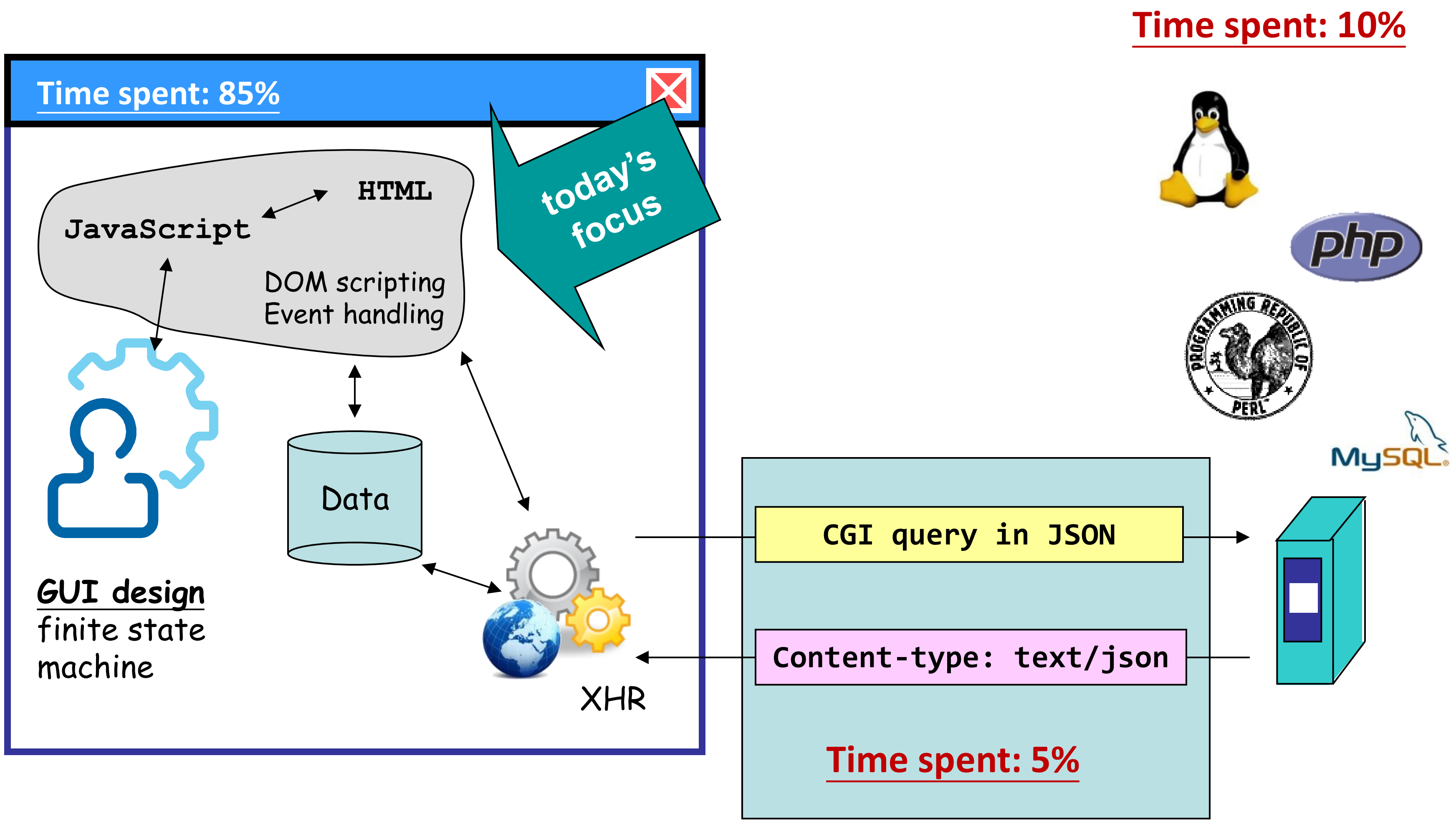
- Putting all we've learned together + new stuffs...

AJAX-based Notepad

- Integrating XHR, DOM Scripting, JSON, PHP together.

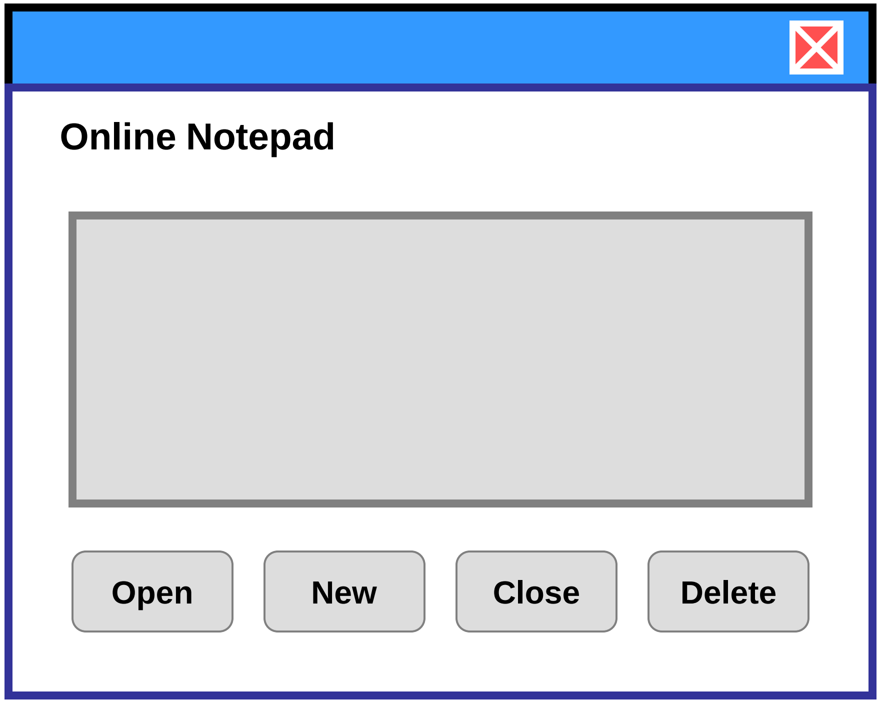
Examples: http://demo4140-tywong.rhcloud.com/15_notepad/

Our big picture...



Plan before you write...

- I'm not talking about the planning of the outlook of the UI...



Such an outlook won't cause you much time and effort...

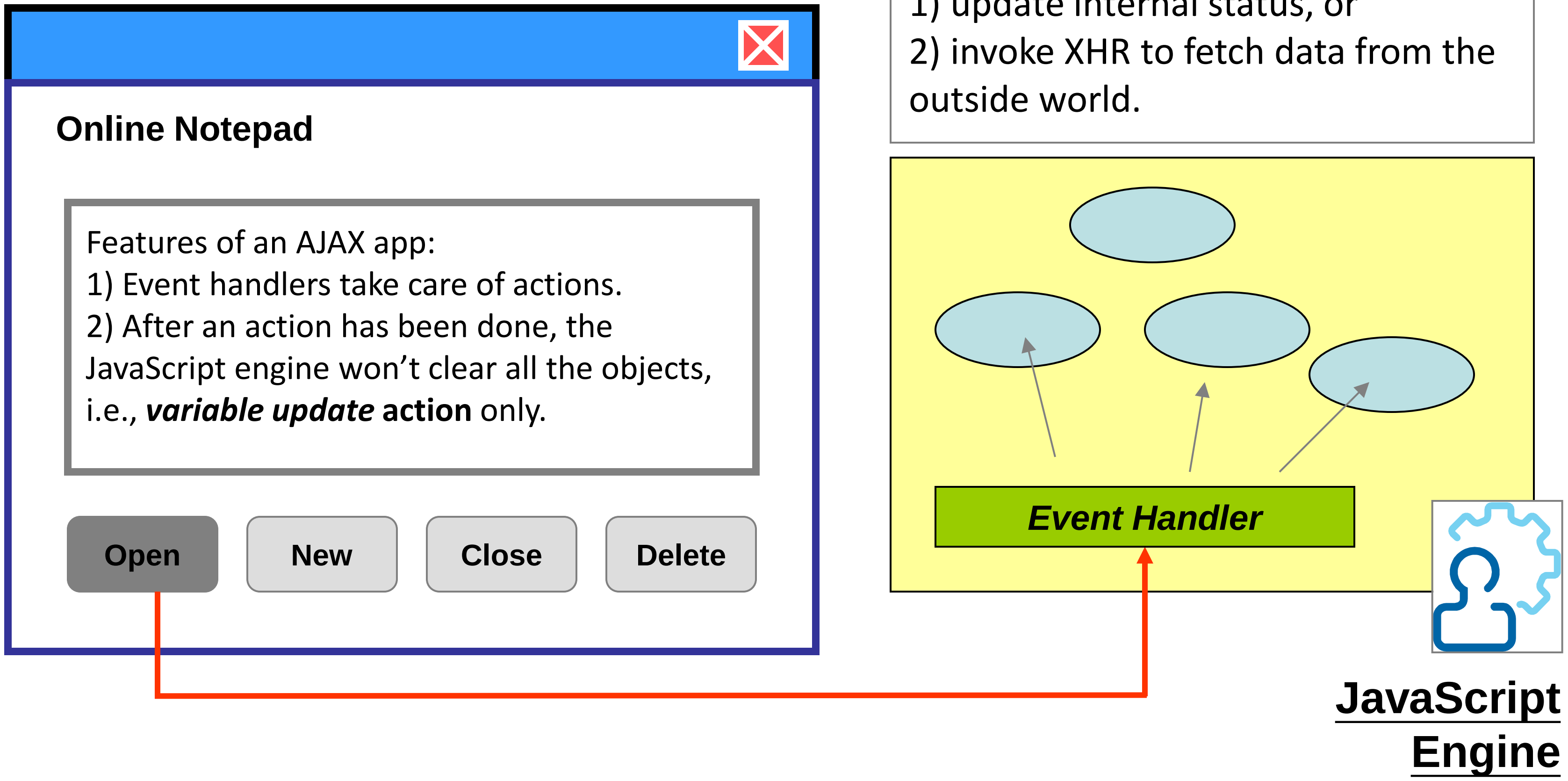
Our concern is:

You should plan ahead on the functions and the interactions among the objects before you write the JavaScript!

Plan before you write...

- Remember, an AJAX app seldom refreshes a page...

So, an event handler should either:
1) update internal status, or
2) invoke XHR to fetch data from the outside world.


Online Notepad

Features of an AJAX app:
1) Event handlers take care of actions.
2) After an action has been done, the JavaScript engine won't clear all the objects, i.e., ***variable update*** **action** only.

AJAX Application Design...

- It becomes GUI programming...

UI Layout
With or without the use of flowing DIVs;
With or without the use of absolute-position DIVs;

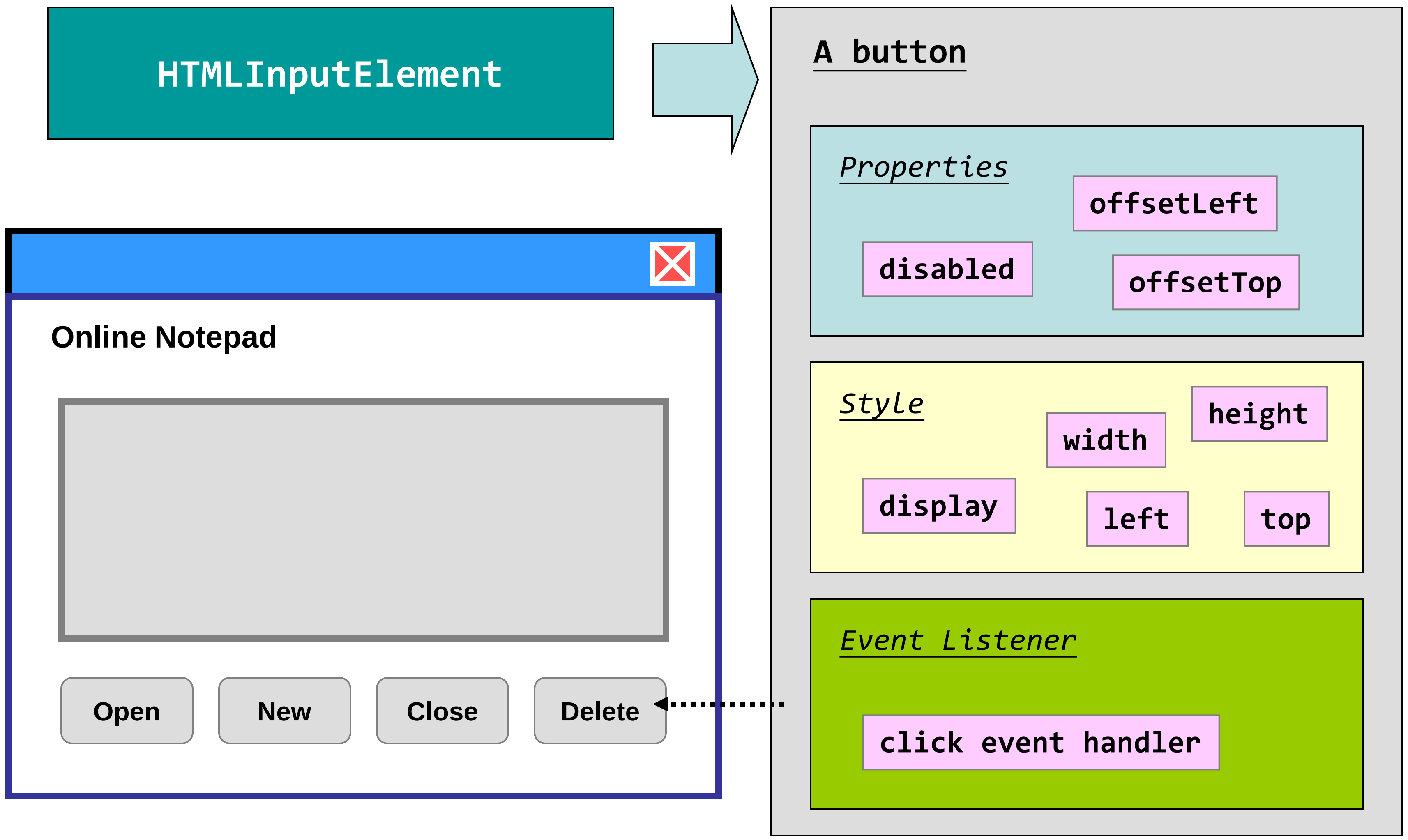
Structured Data
JavaScript-abstracted HTML objects, i.e., **the abstraction of the UI with DOM**;
JavaScript data or objects representing **internal status**.

AJAX Application Design...

- It becomes GUI programming...

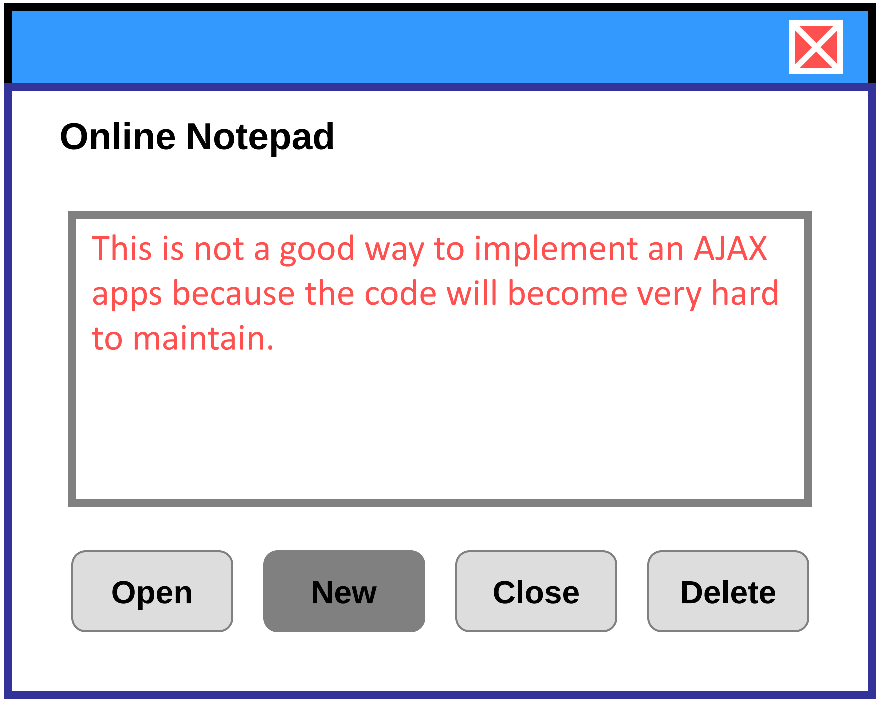
Events and the Handlers
JavaScript Engine are event-driven.
In most cases, an event handler will only **take your application to another state**, but seldom to another page (unless changing "**`window.location`**"). By the way...visiting another page means clearing up all the memory in the JavaScript Engine. Well...the "**Back**" button won't help neither. E.g., Facebook: "***See More***", "***View X more comments***".
Of course, an event handler can trigger XHR objects to fetch external data. Still, it is updating **internal status** in the JS engine.

AJAX Application Design...

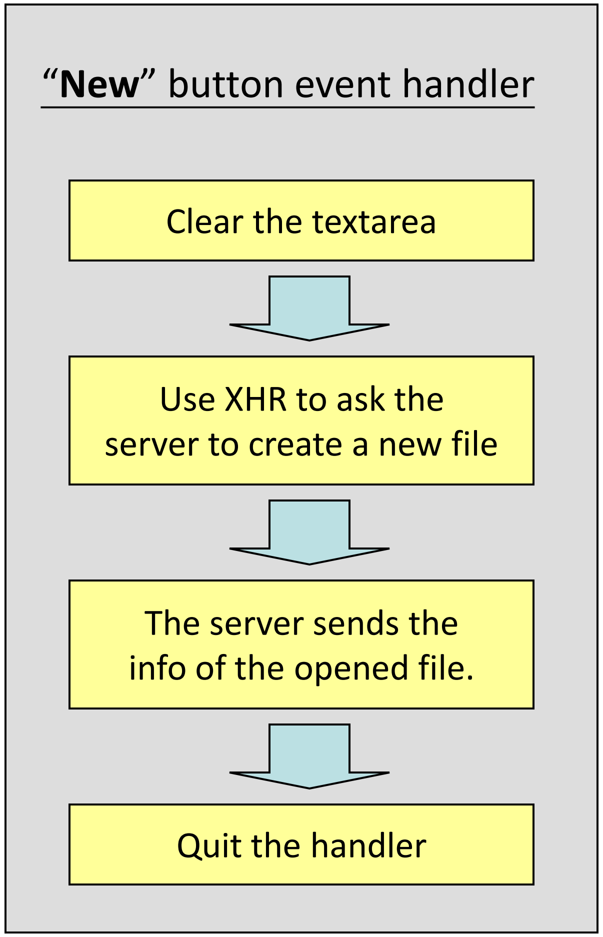


A way to implement ?

- Surely, you can implement each feature into a function…



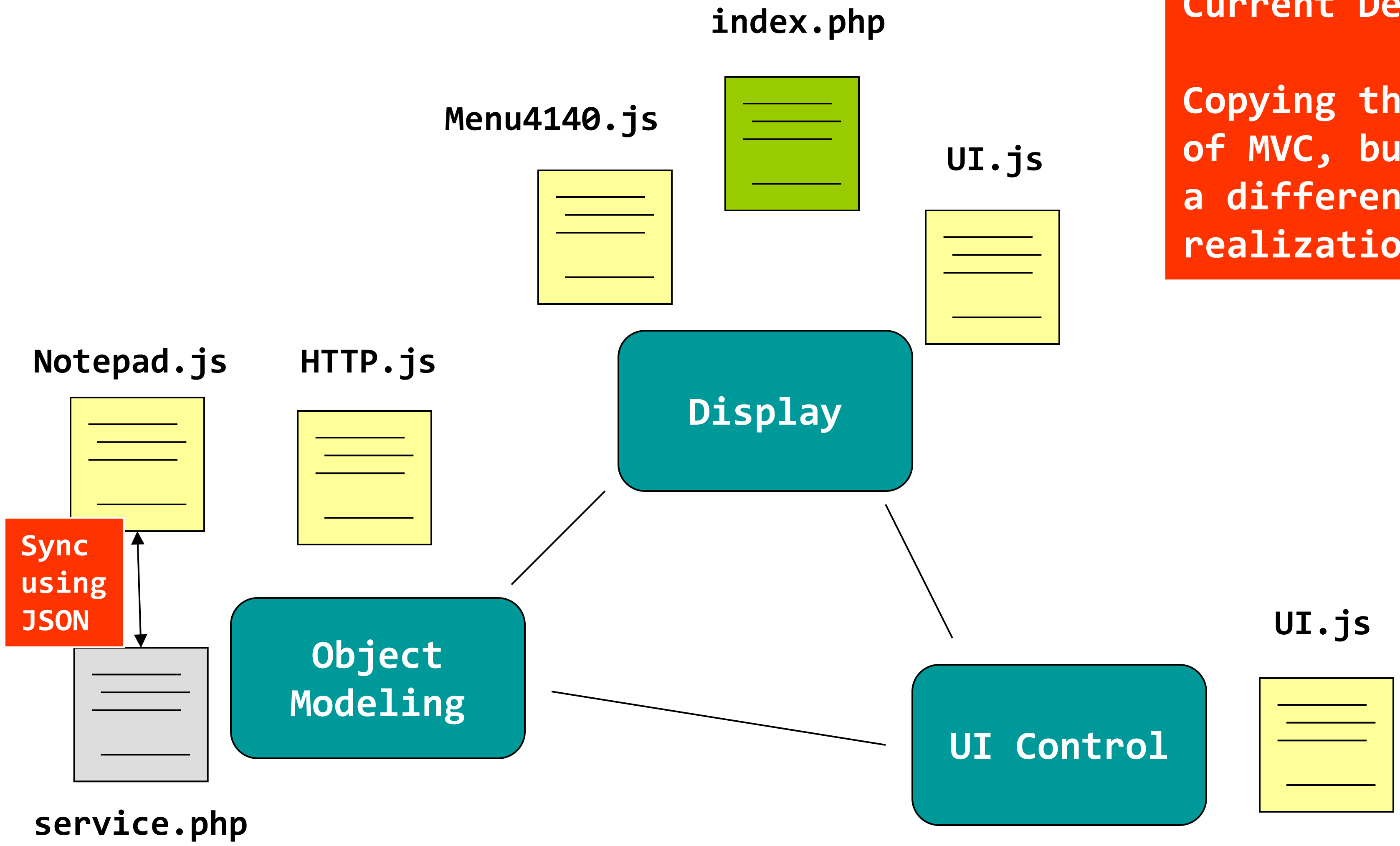
This is not a good way to implement an AJAX apps because the code will become very hard to maintain.


"**New**" button event handler
Clear the textarea
Use XHR to ask the server to create a new file
The server sends the info of the opened file.
Quit the handler

Design in the Online Notepad

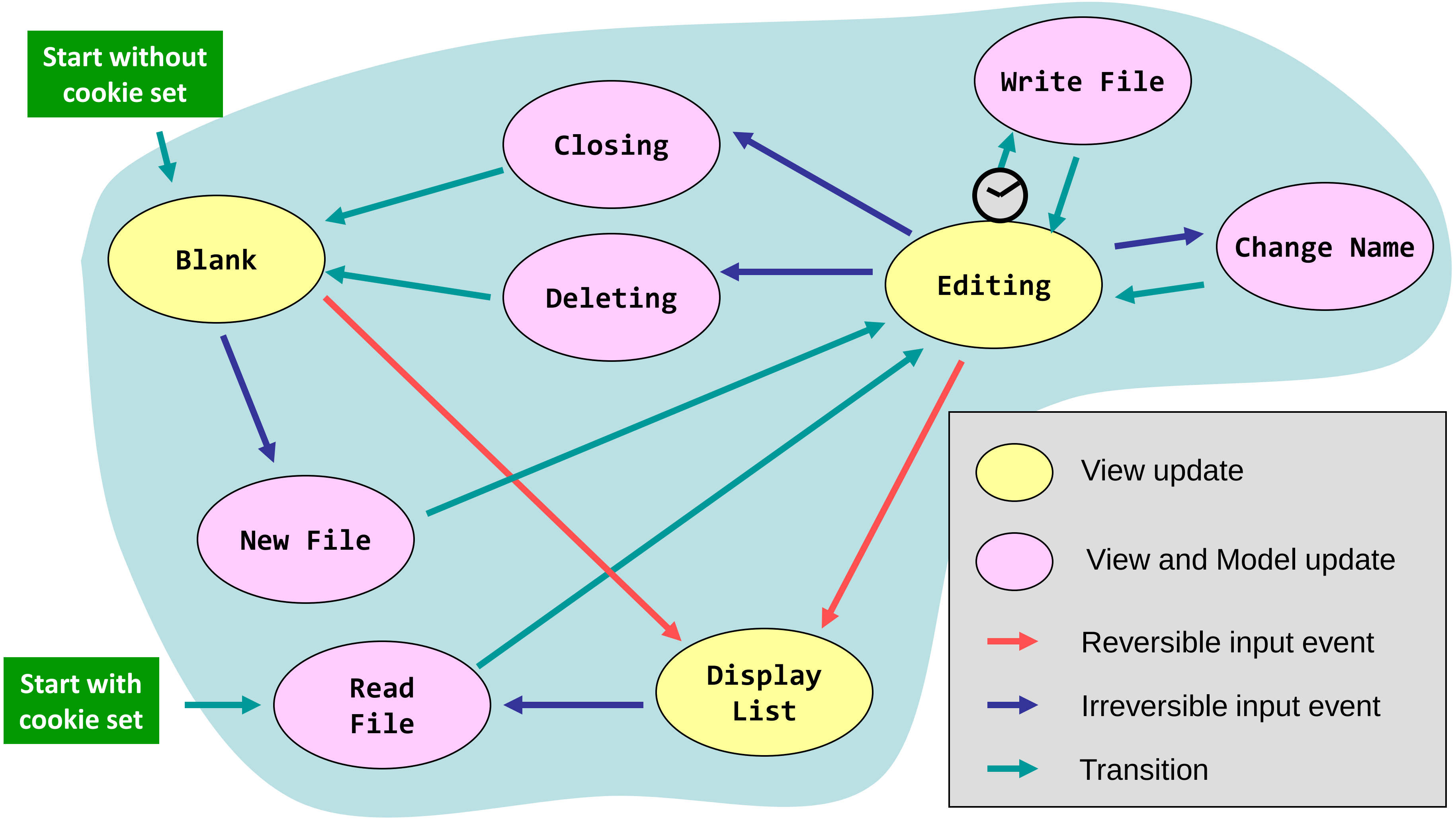
Current Design

Copying the idea of MVC, but with a different realization.



UI FSM

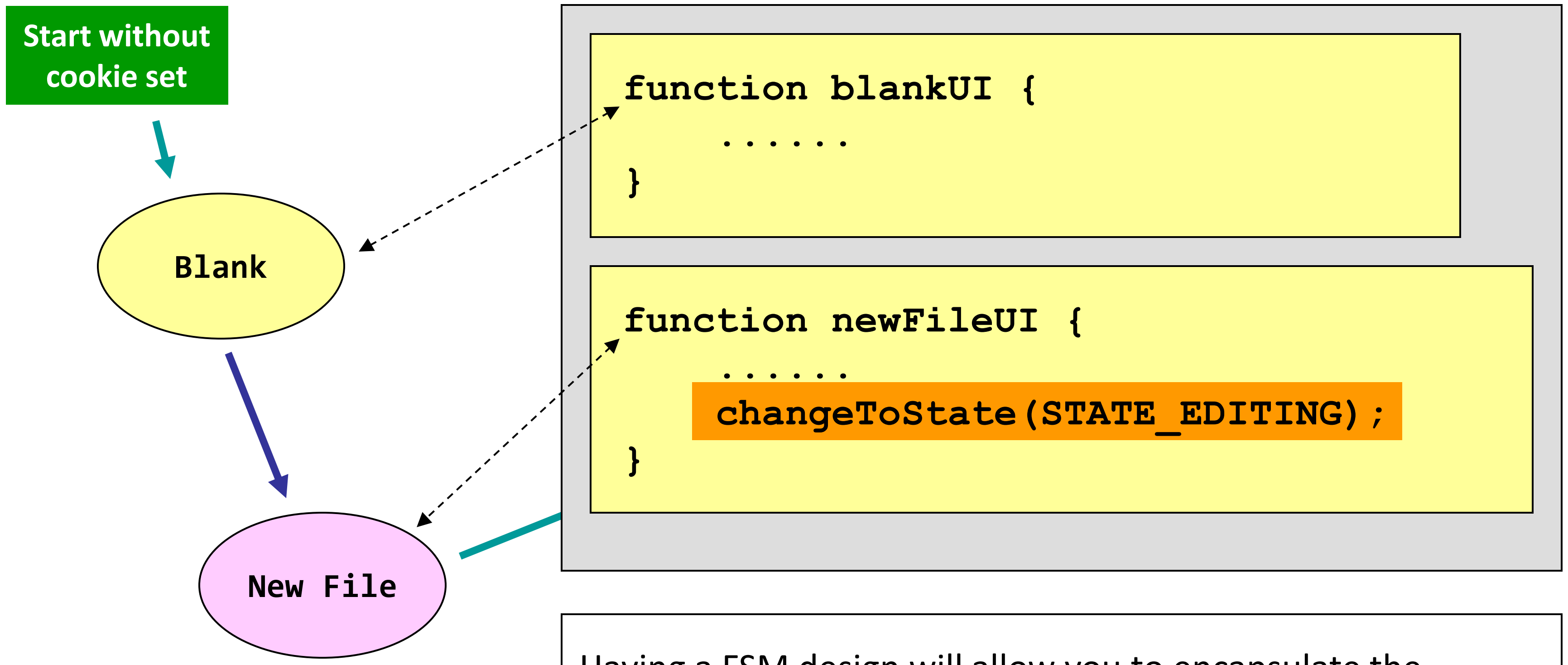
Note: *I really drew this on paper before I started coding!*



See "UI.js"

UI FSM

Note: *I really drew this on paper before I started coding!*



```
function blankUI {
      ......
}
```

```
function newFileUI {
      ......
    changeToState(STATE_EDITING);
}
```

Having a FSM design will allow you to encapsulate the information concerning a state.

According to my experience, drawing and then implementing a FSM **definitely improve your productivity**!

Design Concern – data abstraction

- Giving others a nicer programming experience....

```
<html>
<script type=text/javascript src=Menu4140.js></script>
<script type=text/javascript src=Notepad.js></script>
<script type=text/javascript src=HTTP.js></script>
<script type=text/javascript src=UI.js></script>

<script type=text/javascript >
   var notepadData = new Notepad(...);
   ......
</script>

</html>
```

The best practice is to implement each JS file as an independent library.

Similar to other modern prog. lang., previous statements *import* function implementations and variable declarations.

Design Concern – data abstraction

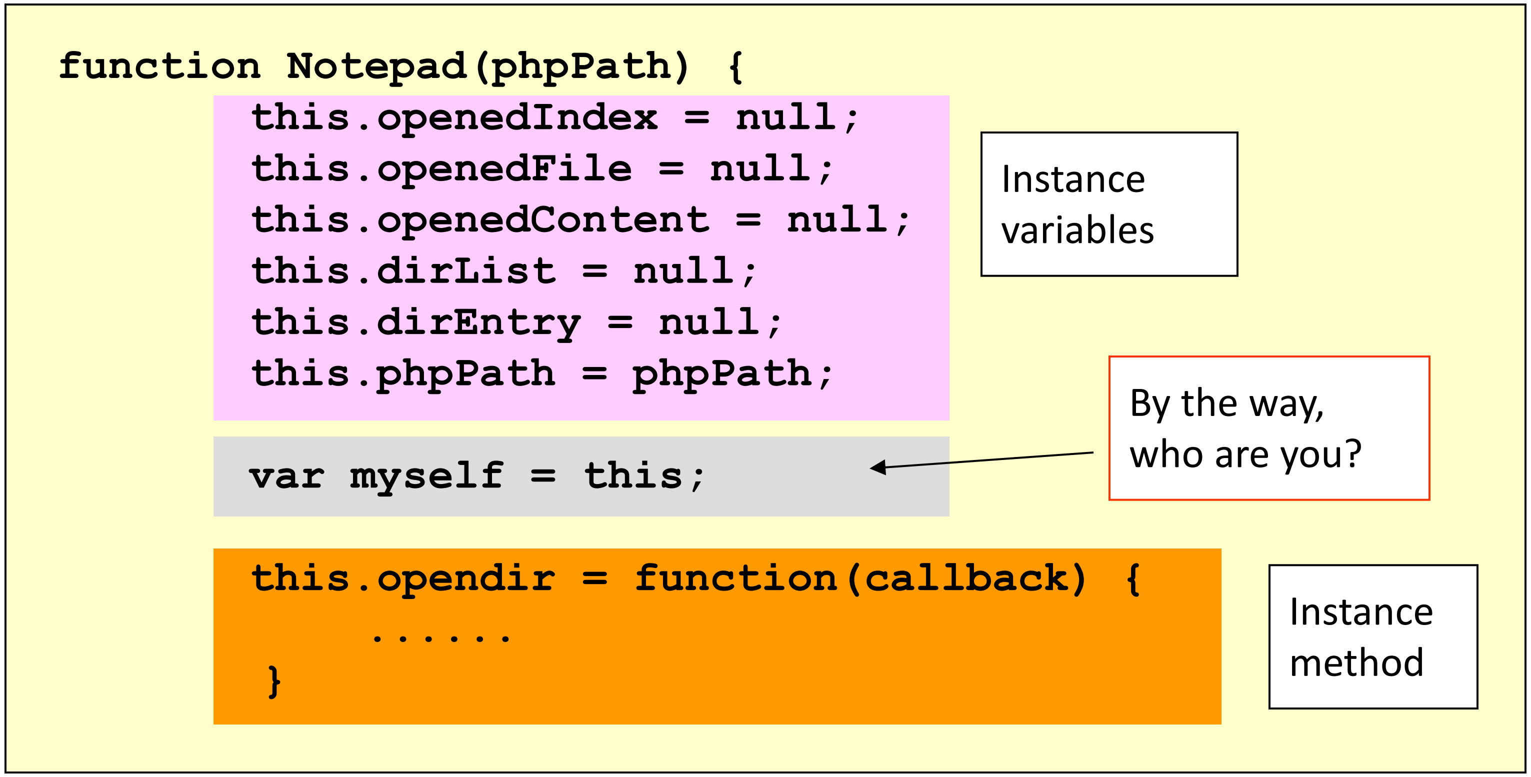
- Last but not least...**library initialization**...
  - POP QUIZ: how to initialize the library when it is loaded?

```
<html>
<script type=text/javascript src=Menu4140.js></script>
<script type=text/javascript src=Notepad.js></script>
<script type=text/javascript src=HTTP.js></script>
<script type=text/javascript src=UI.js></script>

<script type=text/javascript >
    var notepadData = new Notepad(...);
    ......
</script>

</html>
```

See “UI.js”

Design Concern – data abstraction

- By the way, there is a strange variable in the libraries…



```
function Notepad(phpPath) {
        this.openedIndex = null;
        this.openedFile = null;
        this.openedContent = null;
        this.dirList = null;
        this.dirEntry = null;
        this.phpPath = phpPath;

        var myself = this;

        this.opendir = function(callback) {
             ......
         }
```

See “Notepad.js”

Design Concern – data abstraction

- The keyword "**this**" is **context sensitive**!

```
function TestObject() {
    var myself = this;

    this.handler = function (e) {

        var output = "";
        output += "This is: " + this + "<br>\n";
        output += "Myself is: " + myself + "<br>\n";
        document.body.innerHTML = output;
    }
}

var testObj = new TestObject();
window.addEventListener("load", testObj.handler, false);
```

When resolving the "**this**" reference, the handler function is called as a member function of another object.

When resolving the "**myself**" reference, it is found that it is declared when "**this**" is equal to the "**TestObject**" object. So...

This registers the handler function becoming a member function of the "**window**" object.

See "myself_vs_this.html"

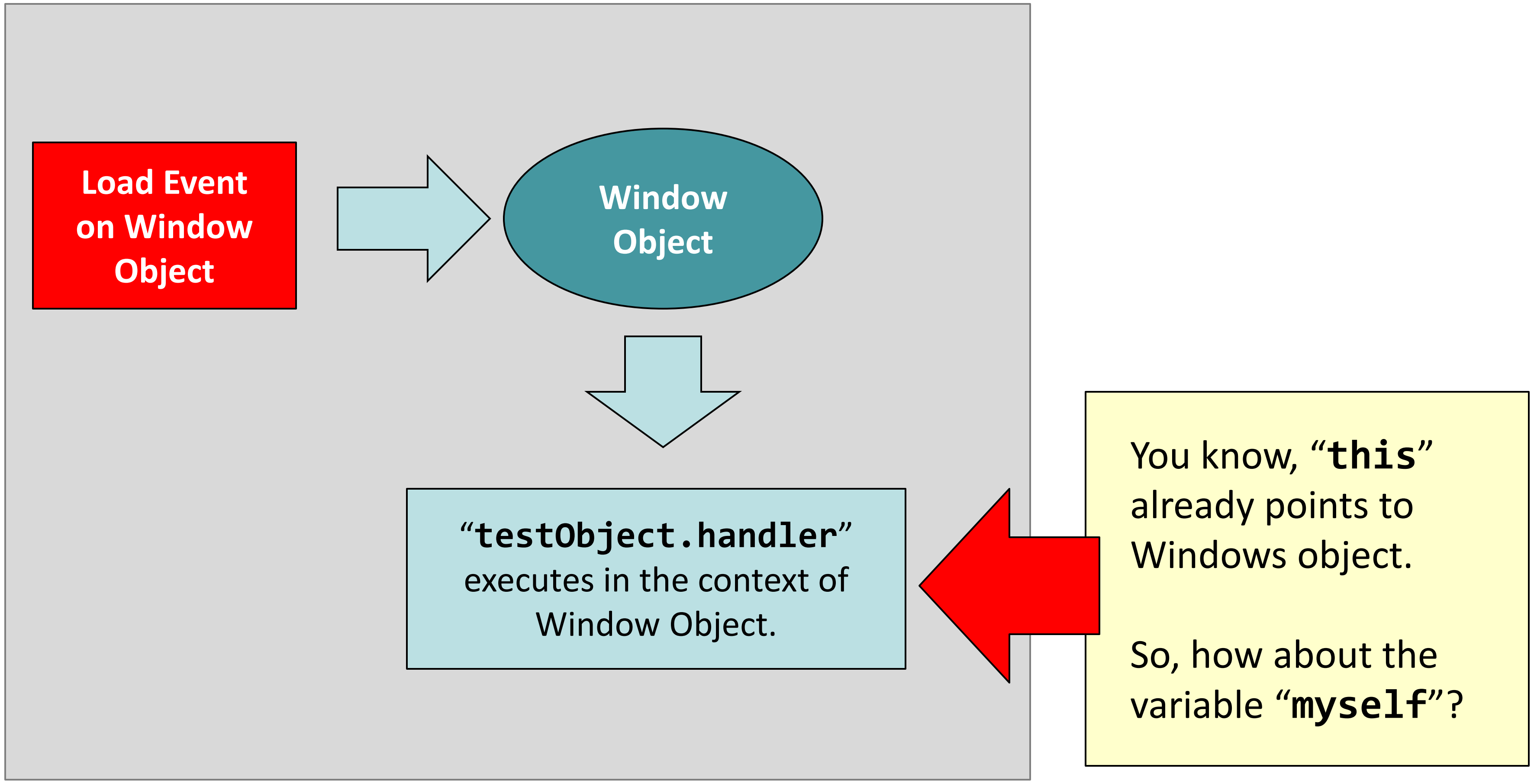
Sidetrack: Closure

An advanced “feature” of JavaScript…

Examples: http://demo4140-tywong.rhcloud.com/16_closure/

Back to previous example…

- The calling environment is changed!


Load Event on Window Object
Window Object
"**`testObject.handler`**" executes in the context of Window Object.

You know, "**`this`**" already points to Windows object.

So, how about the variable "**`myself`**"?

Functional Scoping VS Closure

- What are the differences between the following two functions?

```
function init() {
  var name = "Mozilla";
  function displayName() {
    alert(name);
  }
  displayName();
}
init();
```

Version 1

```
function makeFunc() {
  var name = "Mozilla";
  function displayName() {
    alert(name);
  }
  return displayName;
}

var myFunc = makeFunc();
myFunc();
```

Version 2

See “displayName_v1.html” & “displayName_v2.html”

Functional Scoping VS Closure

- What are the differences between the following two functions?

```
function init() {
  var name = "Mozilla";
  function displayName() {
    alert(name);
  }
  displayName();
}
init();
```

Version 1

The inner function “**displayName**” can access the variable “**name**” because of the functional scoping of JavaScript.

See “displayName_v1.html” & “displayName_v2.html”

Functional Scoping VS Closure

- What are the differences between the following two functions?

The return value of "**makeFunc()**" is a **closure**!

A closure is a special kind of object that combines two things:
-a function, and
-the environment in which that function was created.

```
function makeFunc() {
  var name = "Mozilla";
  function displayName() {
    alert(name);
  }
  return displayName;
}

var myFunc = makeFunc();
myFunc();
```

Environment of displayName().

as a return value.

Version 2

See "displayName_v1.html" & "displayName_v2.html"

Closure use case

- Setting up event listeners in loops!

```
function init(event) {
    var N = 4;
    var node = new Array();
    for(var i = 0; i < N; i++) {
        var tmp = document.createElement("div");
        ......
        tmp.innerHTML = i;
        tmp.addEventListener("click",
            function (e) {
                alert(i);
            },
        false);
        node.push(tmp);
        document.body.appendChild(tmp);
    }
}
window.addEventListener("load", init, false);
```

Environment of this closure is: ***variable i***, but not value of i. Therefore…

See "event_wrong.html"

Closure use case

- Setting up event listeners in loops!

```
function init(event) {
    var N = 4;
    var node = new Array();
    for(var i = 0; i < N; i++) {
        var tmp = document.createElement("div");
        ......
        tmp.innerHTML = i;
        tmp.addEventListener("click",
            function () {
                var cnt = i;
                return function (e) {
                    alert(cnt);
                }
            }(),
        false);
    }
}
```

This statement is executed when we create the closure.

Therefore, the closure binds to "cnt", but not "i".

In this example, **different iterations produce different closures**:

-Every closure binds to the variable named "cnt".

-But, "cnt" is re-created after each iteration, carrying a different value.

See "event_right.html"

Closure use case

- Setting up event listeners in loops!

```
function init(event) {
    var N = 4;
    var object = { "value" : 0 };
    var node = new Array();
    for(var i = 0; i < N; i++) {
        var tmp = document.createElement("div");
        ......
        object.value = i;
        tmp.addEventListener("click",
            function () {
                var cnt = object;
                return function (e) {
                    alert(cnt.value);
                }
            }(),
        false);
    }
}
```

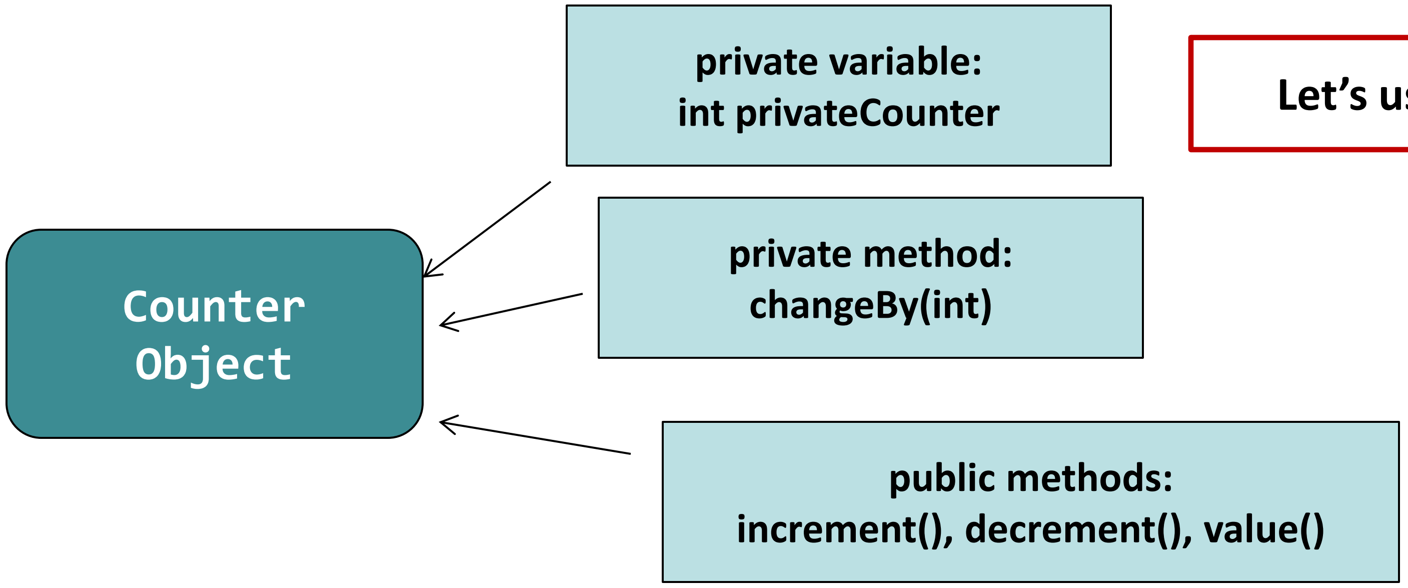
Pop Quiz!

What is wrong?

See “event_wrong_v2.html”

Closure use case: OOP

- Do you still remember the differences between the keywords "**public**" and "**private**".
  - They exist in C++ and Java.
  - However, they are missing in JavaScript.


Counter Object
private variable: int privateCounter
private method: changeBy(int)
public methods: increment(), decrement(), value()

Let's use closure!

Closure use case: OOP

```
var Counter = (function() {
  var privateCounter = 0;
  function changeBy(val) {
    privateCounter += val;
  }
  return {
    increment: function() {
1     changeBy(1);
    },
    decrement: function() {
2     changeBy(-1);
    },
    value: function() {
3     return privateCounter;
    }
  }
})();
```

All three closures are sharing the **same environment** when they are created.

Can you see that?

-**changeBy()** and **privateCounter** are private to every closure.

-Since the environments of the 3 closures are shared, the public functions are **using the same set of private variables and private methods**.

See "private_v1.html"

Closure use case: OOP

```
var Counter = (function() {
  var privateCounter = 0;
  function changeBy(val) {
    privateCounter += val;
  }
  return {
1   increment: function() {
      changeBy(1);
    },
2   decrement: function() {
      changeBy(-1);
    },
3   value: function() {
      return privateCounter;
    }
  }
})();
```

```
alert(Counter.value());
Counter.increment();
Counter.increment();
alert(Counter.value());
Counter.decrement();
alert(Counter.value());

alert(Counter.privateCounter);
```

See how much did you get

Can you print the value of "privateCounter"?

See "private_v1.html"

Closure use case: OOP

```
var Counter = (function() {
  var privateCounter = 0;
  function changeBy(val) {
    privateCounter += val;
  }
  return {
1   increment: function() {
      changeBy(1);
    },
2   decrement: function() {
      changeBy(-1);
    },
3   value: function() {
      return privateCounter;
    }
  }
})();
```

```
Counter.increment();

alert( Counter.privateCounter );
Counter.privateCounter = 100;
alert( Counter.privateCounter );
alert( Counter.value() );
```

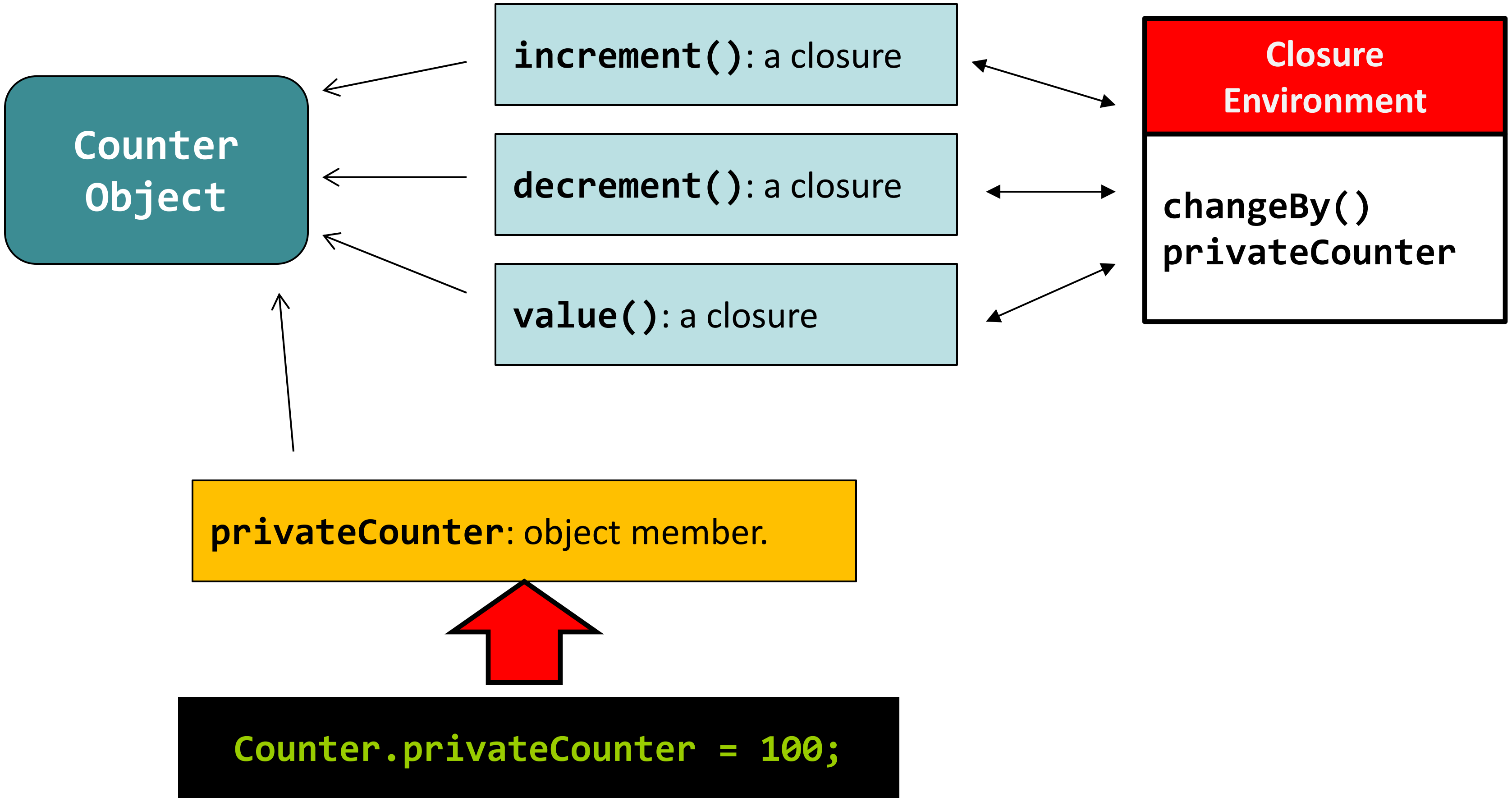
Let's have a tougher example

What will you get?

See "private_v2.html"

Closure use case: OOP

- Explaining "**`private_v2.html`**":



```
Counter.privateCounter = 100;
```

Closure use case: OOP

- Creating objects without constructors:

"Constructor" code: 1 of 2

```
var makeCounter = function(name) {
  var privateCounter = 0;
  function changeBy(val) {
    privateCounter += val;
  }
```

(To-be) Closure Environment

```
name, privateCounter,
changeBy()
```

"Constructor" code: 2 of 2

```
  return {
    toString: function() {
      return name;
    },
    increment: function() {
      changeBy(1);
    },
    decrement: function() {
      changeBy(-1);
    },
    value: function() {
      return privateCounter;
    }
  }
};
```

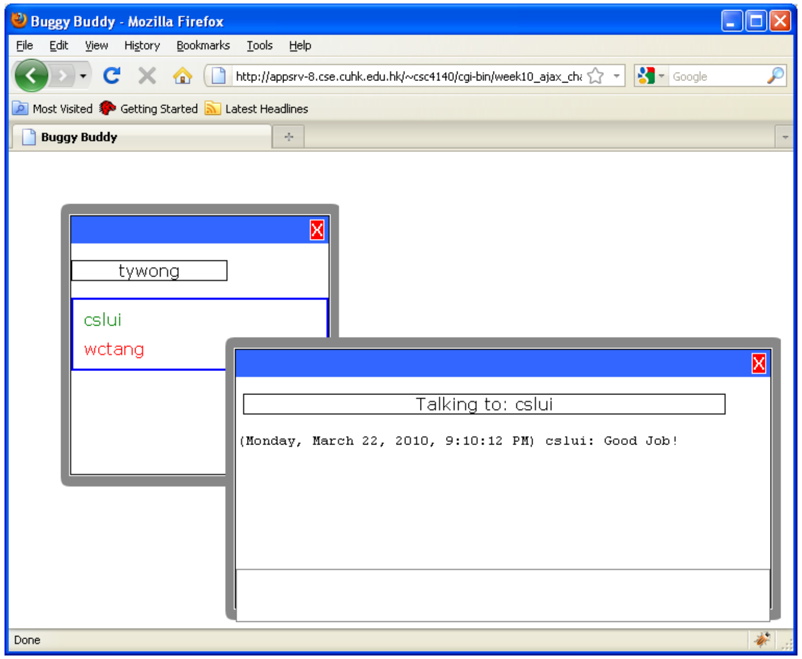
See "private_v3.html"

A component used in chatroom

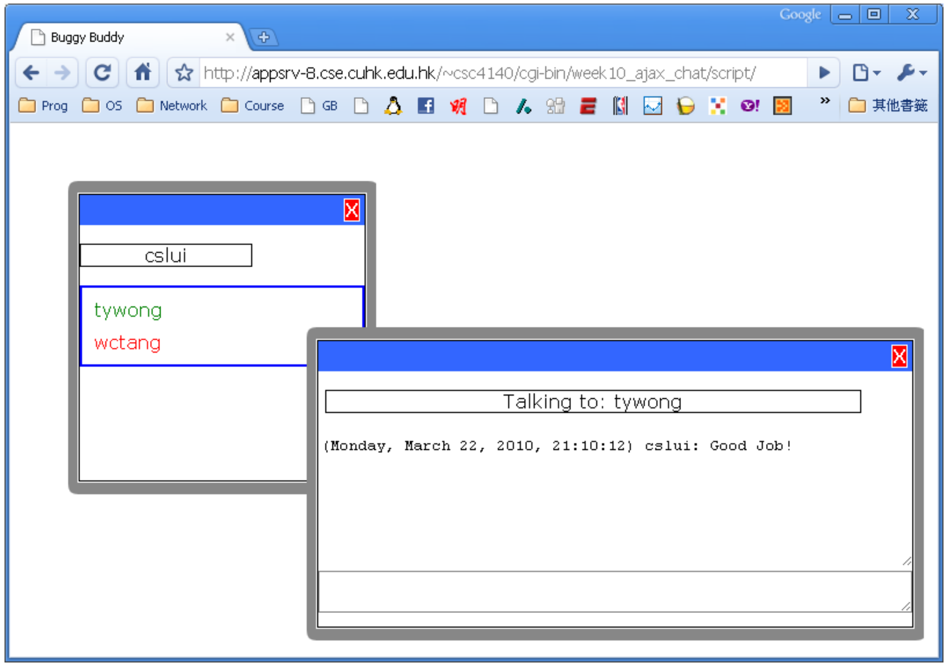
Comet: enabling browsers talking to each other...

Examples: http://demo4140-tywong.rhcloud.com/17_comet/

How to write this application?


Buggy Buddy - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://appsrv-8.cse.cuhk.edu.hk/~csc4140/cgi-bin/week10_ajax_cha
Most Visited Getting Started Latest Headlines
Buggy Buddy
tywong
cslui
wctang
Talking to: cslui
Done

```
(Monday, March 22, 2010, 9:10:12 PM) cslui: Good Job!
```

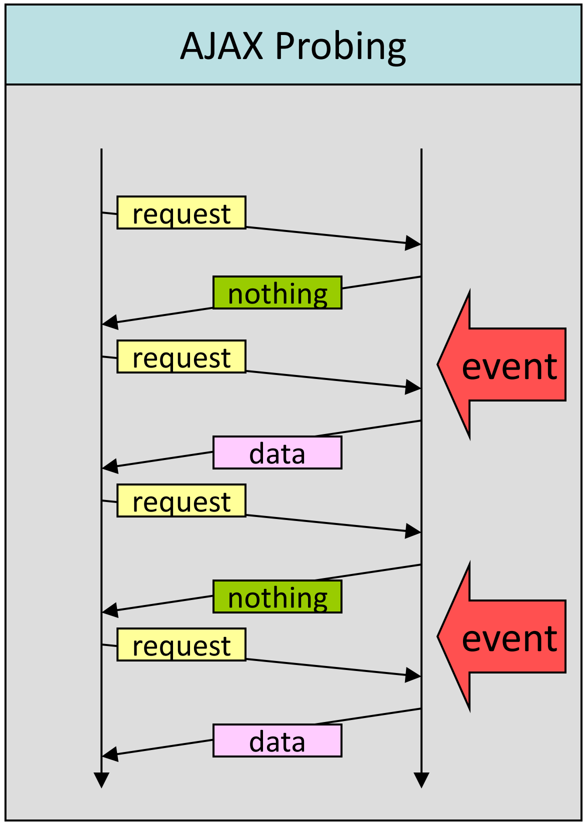

Buggy Buddy
http://appsrv-8.cse.cuhk.edu.hk/~csc4140/cgi-bin/week10_ajax_chat/script/
Prog OS Network Course GB 其他書籤
cslui
tywong
wctang
Talking to: tywong

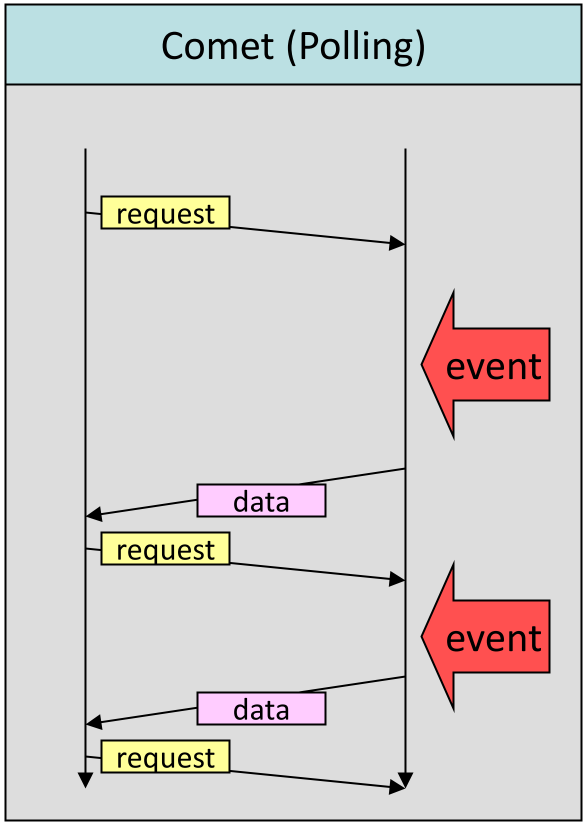
```
(Monday, March 22, 2010, 21:10:12) cslui: Good Job!
```

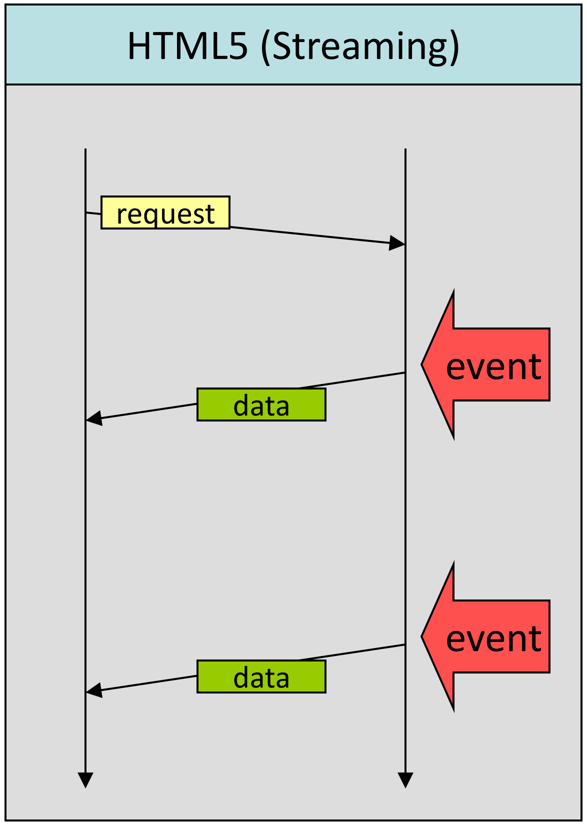
Techniques involved.

HTTP cookie, DOM scripting, PHP scripting, XHR (XMLHttpRequest) and **PHP Comet**.

Techniques?

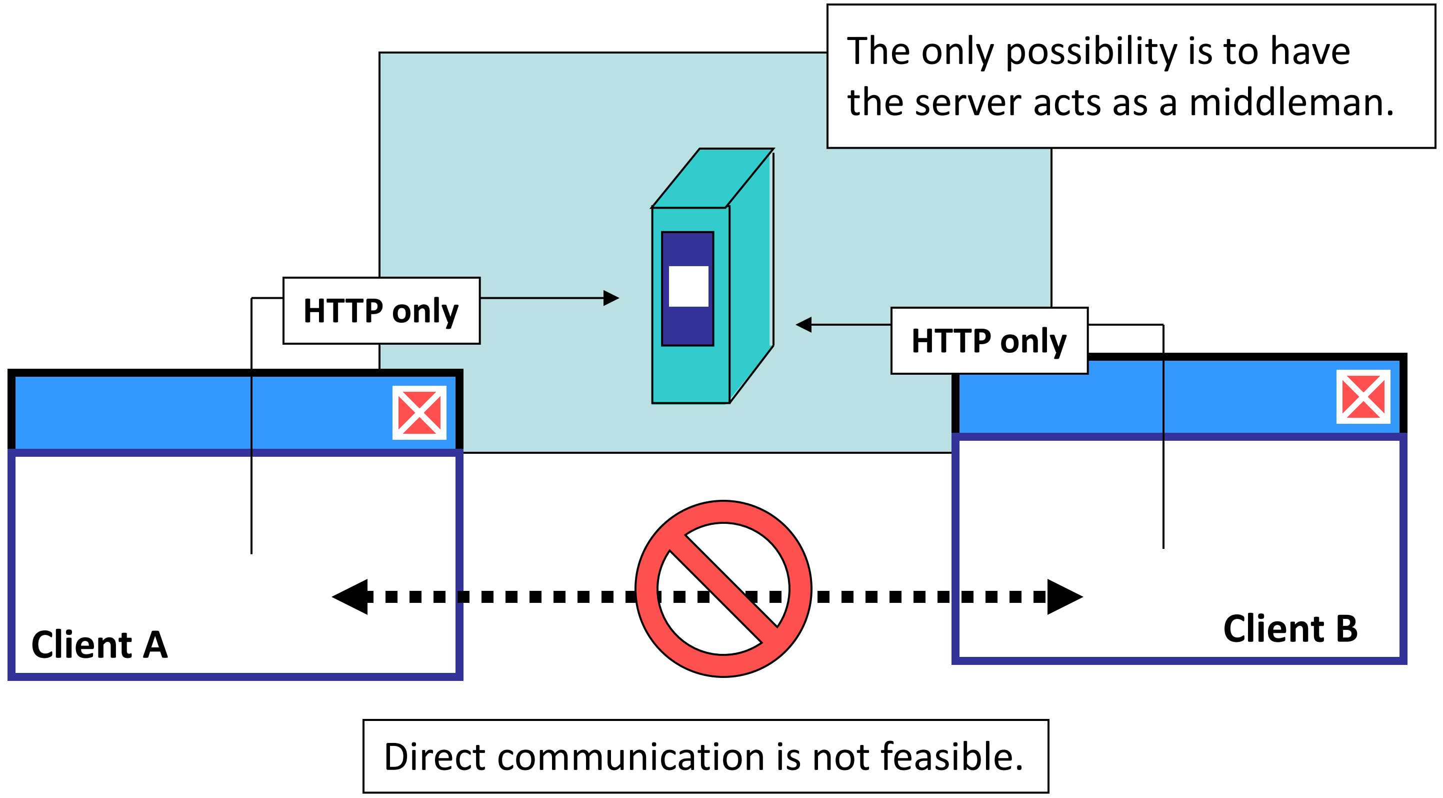

AJAX Probing
request
nothing
request
data
request
nothing
request
data
event
event


Comet (Polling)
request
data
request
data
request
event
event

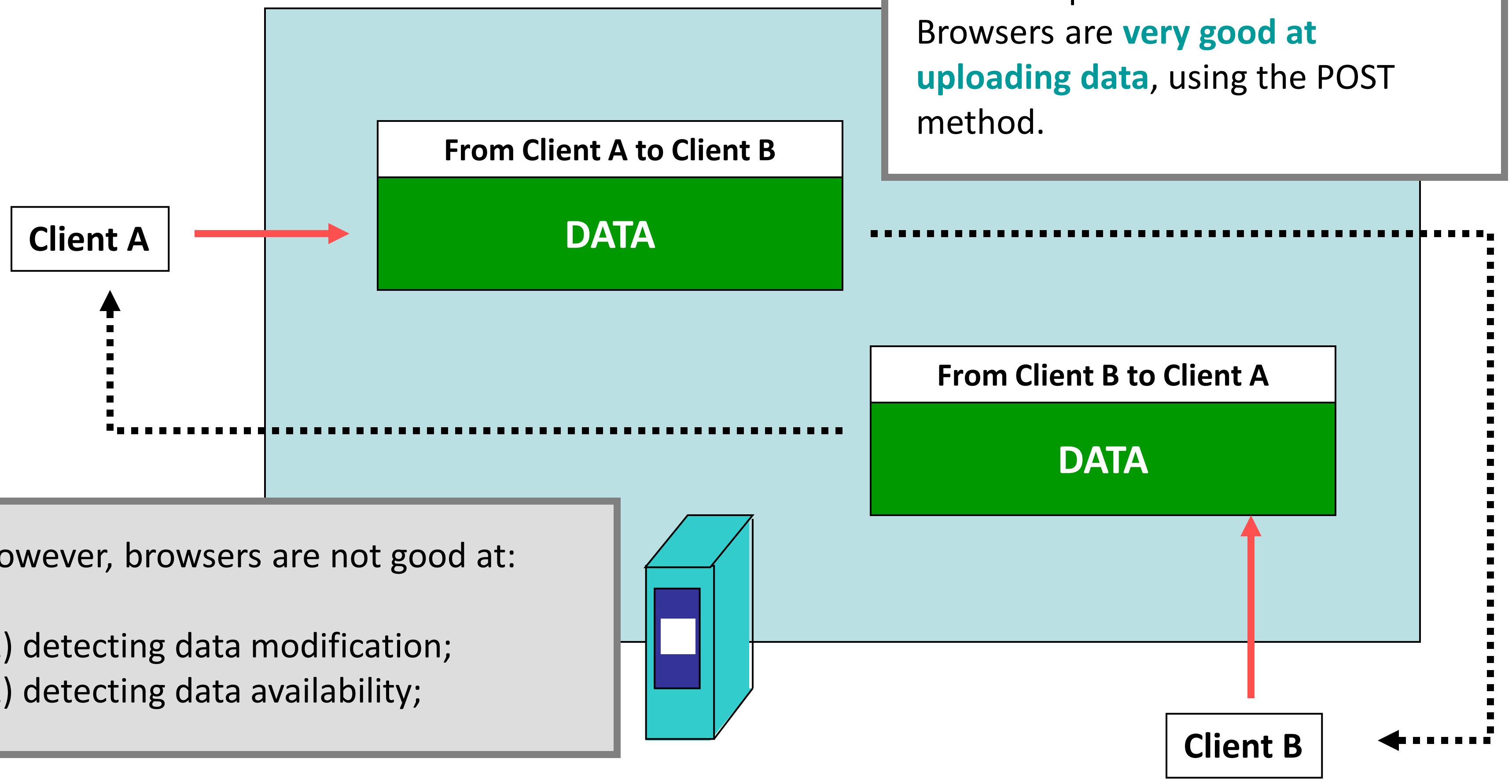

HTML5 (Streaming)
request
data
data
event
event

Sending message?

- Traditionally, we use socket programming.
  - But, we don't have sockets for browsers unless we have a **full HTML5** support.


The only possibility is to have the server acts as a middleman.
HTTP only
HTTP only
Client A
Client B
Direct communication is not feasible.

Server – the middleman

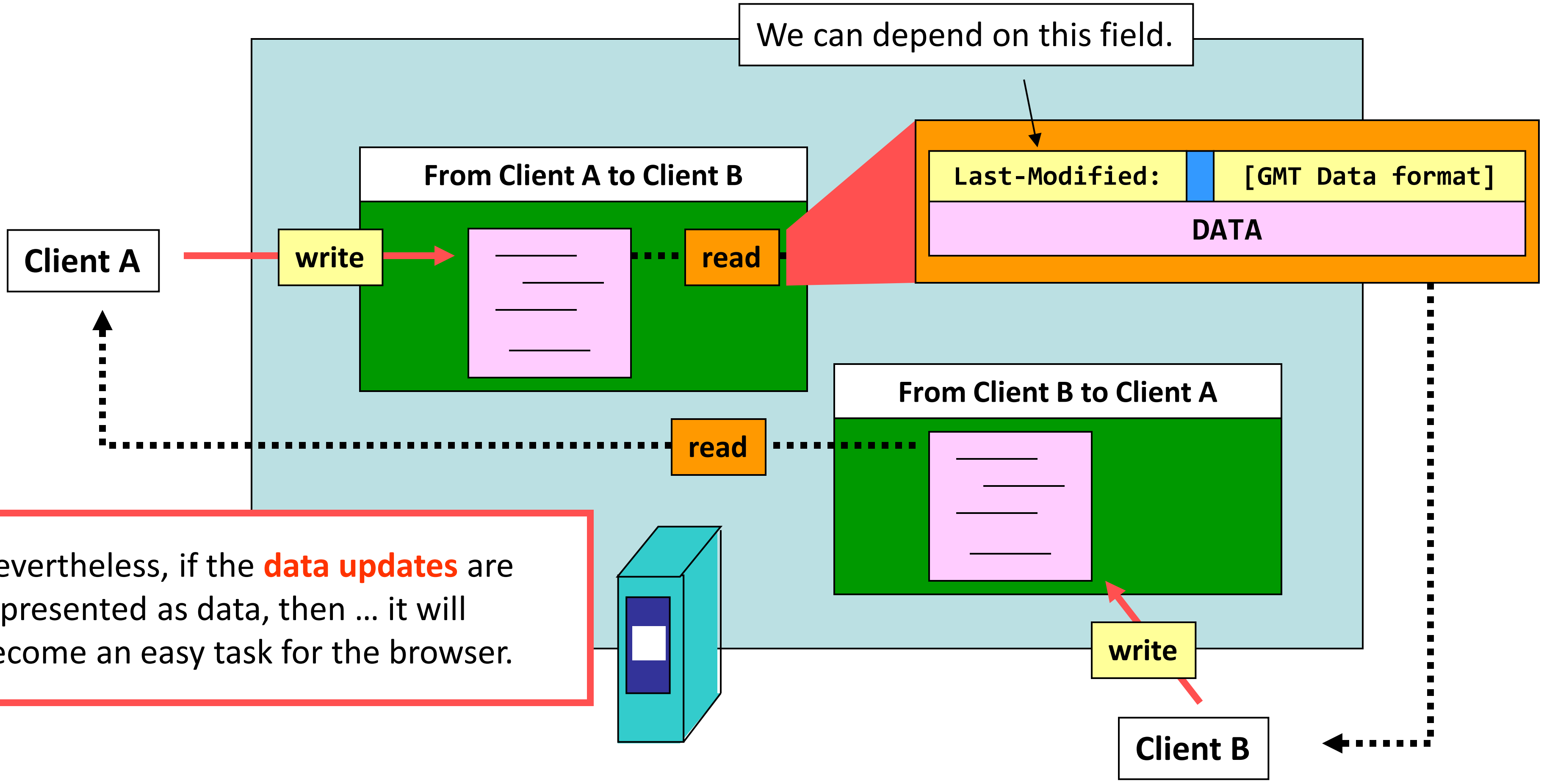


There should be no problem for a client to upload data to the server. Browsers are **very good at uploading data**, using the POST method.

However, browsers are not good at:

(1) detecting data modification;
(2) detecting data availability;

Server – the middleman


We can depend on this field.
From Client A to Client B
Client A
write
read
Last-Modified:
[GMT Data format]
DATA
From Client B to Client A
read
write
Client B

Nevertheless, if the **data updates** are represented as data, then … it will become an easy task for the browser.

Using "**Last-Modified**" …

- The client can do active probing…

```
var http = new XMLHttpRequest();
http.open("GET", "input.txt", false);
http.send(null);
alert( http.getResponseHeader("Last-Modified") );
```

Loop until the field is updated.

Helpful Functions

```
window.setInterval() &
window.setTimeout()
```

Disadvantages
Many HTTP requests.
May cause a high CPU loading on the client side.

See "probing.html", then modify "input.txt" using "modify_txt.php"

Server takes an active role?

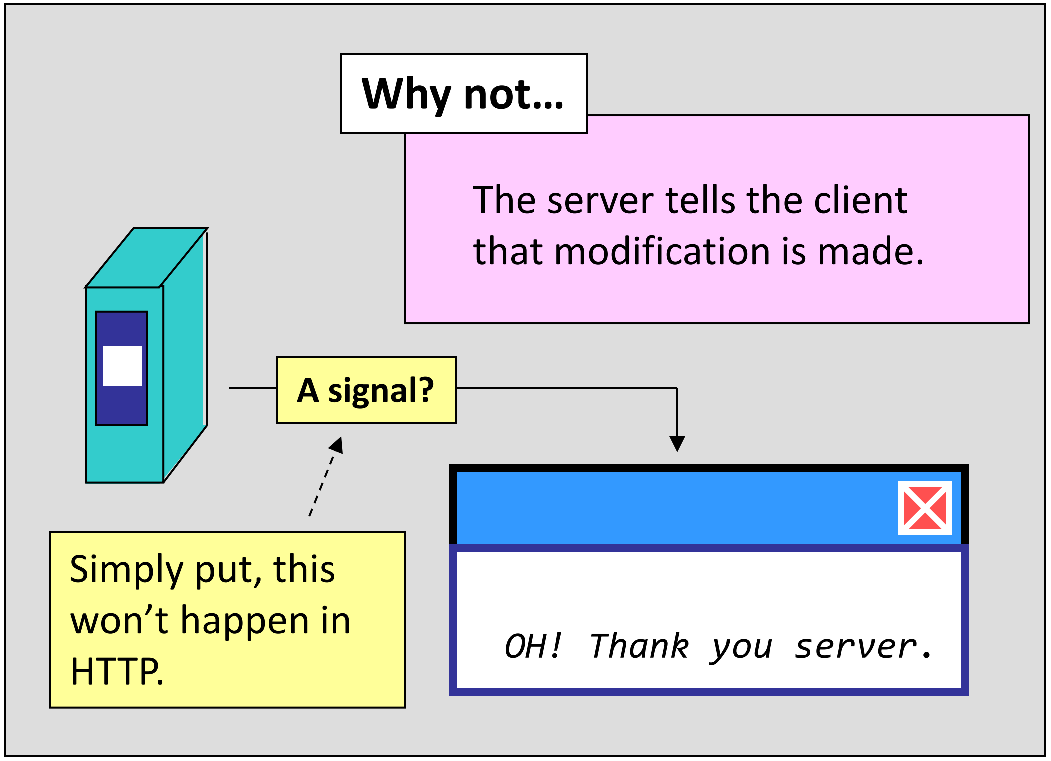
- This is about how to **game the system**…

Remember

HTTP protocol is a **request-response protocol**.

Without a request, there won’t be any response.

Plus, a server will always a server; it won’t send out HTTP requests.


Why not…
The server tells the client that modification is made.
A signal?
OH! Thank you server.
Simply put, this won’t happen in HTTP.

Server takes a sleeping role?

- Understand **the trick** to game the system…

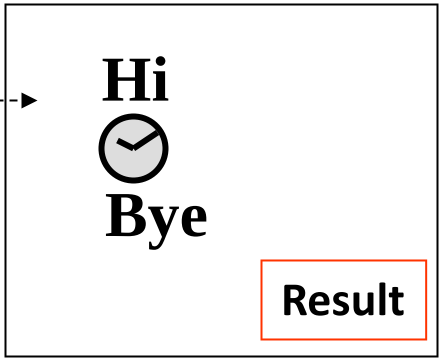
```
<html>

<h1>Hi</h1>
<?
    flush();
    sleep(5);
?>
<h1>Bye</h1>

</html>
```

flush the partial output.

The browser is willing to wait.

So…what if this statement is to determine whether a piece of data is modified or not?

appsrv-8 is a server without gzip enabled.



http://appsrv-8.cse.cuhk.edu.hk/~tywong/csci4140/comet/script/sleep_simple.php

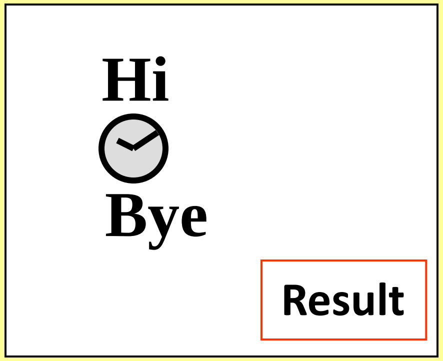
`Try in Chrome and Safari`

Server takes a sleeping role?

- Issue #1: buffering before rendering.

Padding space characters on the left of "**<h1>Hi</h1>**" until the total output is of 2048 bytes. Why using space? **You tell me...**

```
<html>
<?
    echo str_pad("<h1>Hi</h1>", 2048, " ", STR_PAD_LEFT);
    flush();
    sleep(5);
?>
<h1>Bye</h1>

</html>
```

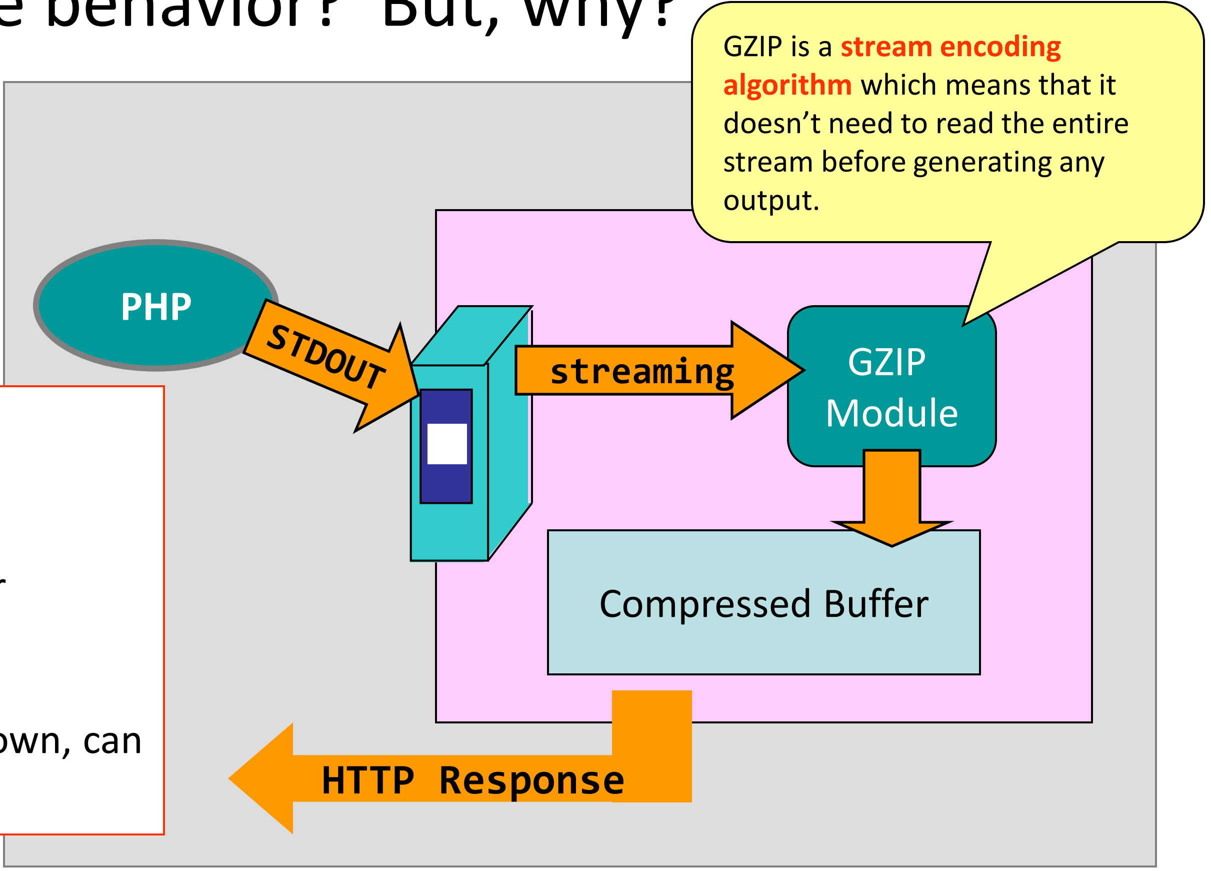


WebKit browsers buffer data before rendering the page. It is "*rumored*" that the buffer size is 2048 bytes.

http://appsrv-8.cse.cuhk.edu.hk/~tywong/csci4140/comet/script/sleep_padding.php
Try in Safari

Server takes a sleeping role?

- Issue #2: gzip compression in Apache.
  - Isn't it a strange behavior? But, why?


GZIP is a **stream encoding algorithm** which means that it doesn't need to read the entire stream before generating any output.
PHP
STDOUT
streaming
GZIP Module
Compressed Buffer
HTTP Response

The truth.

Response is sent when either:
(1) the compressed buffer is full, or
(2) the STDOUT stream is closed.

Discussion: Having the reasons known, can you think of any workaround?

`http://demo4140-tywong.rhcloud.com/17_comet/script/sleep_padding.php`
`Try in Chrome and Safari`

Server takes a sleeping role?

- Issue #2: gzip compression in Apache.

```
<?
   //// Once...
        $n = 100000;
        $str = "<!--";
        for($i = 0; $i < $n; $i++)
           $str .= chr(rand( ord('a'), ord('z') ));
        $str .= "-->";

    //// Twice...
        $str .= "<!--";
        for($i = 0; $i < $n; $i++)
            $str .= chr(rand( ord('a'), ord('z') ));
        $str .= "-->";

    //// Main stuff
        echo "<h1>Hi</h1>\n";
        echo "$str\n";
        flush();
        sleep(5);

        echo "<h1>Bye</h1>";
?>
```

http://demo4140-tywong.rhcloud.com/17_comet/script/sleep_bigpad.php

Try in Chrome and Safari

Server takes a sleeping role?

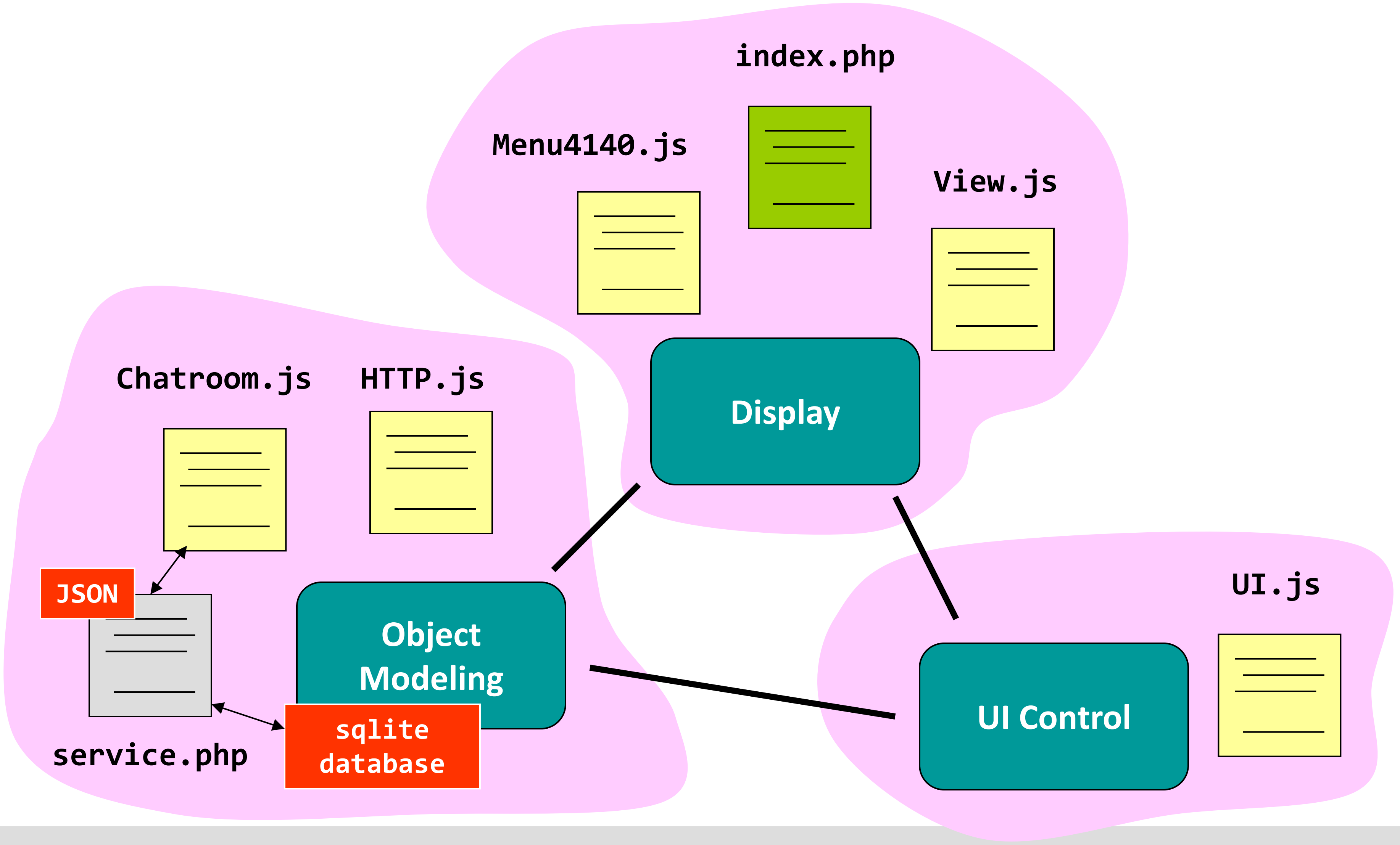
- How do the issues affect you?
  - Issue #1: you need to insert data before the useful content if you want to support Chrome and Safari.
    - A hello/handshake message may be a good idea.
  - Issue #2: you need to insert tons of bogus data after a useful output.
    - It is a high price to pay.
    - For your project, you can choose to **disable the use of compression** in Apache.
    - Well, “appsrv-8” is a machine having the compression feature disabled!

Design of the Buggy-Buddy Chatroom

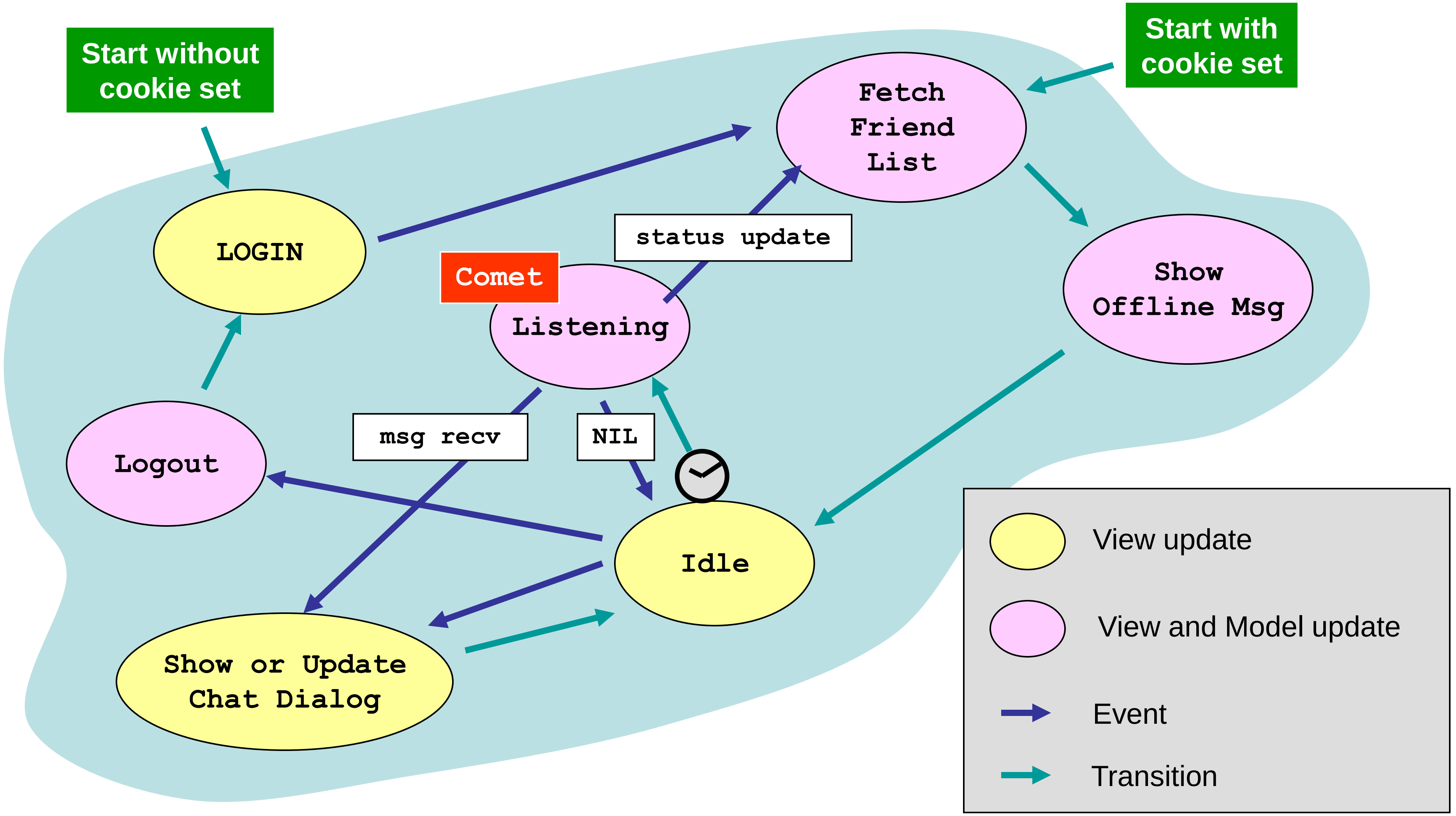
- featuring PHP comet

Examples: http://demo4140-tywong.rhcloud.com/18_chatroom/

Code design

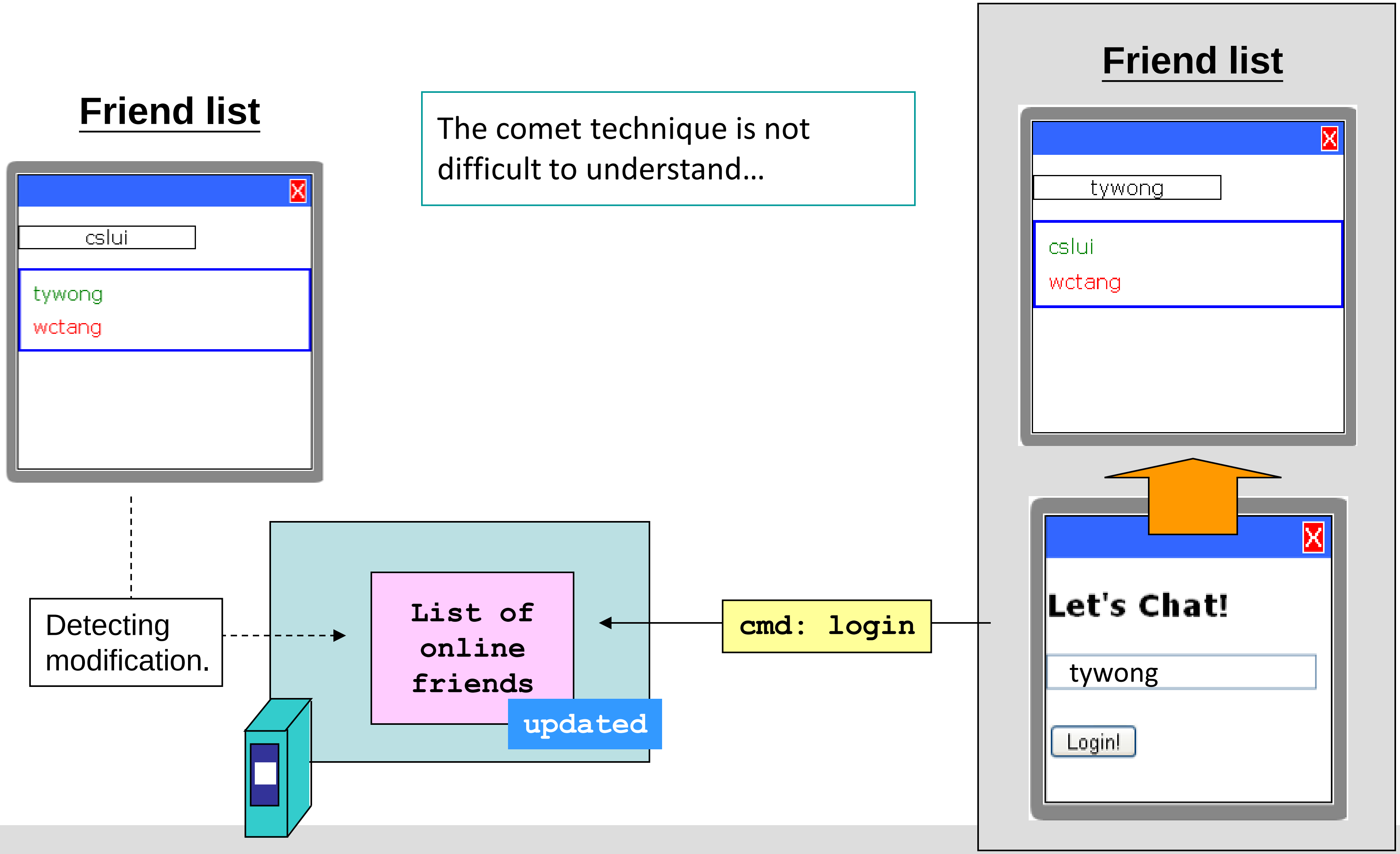

index.php
Menu4140.js
View.js
Display
Chatroom.js
HTTP.js
JSON
Object Modeling
UI.js
UI Control
sqlite database
service.php

Controller FSM

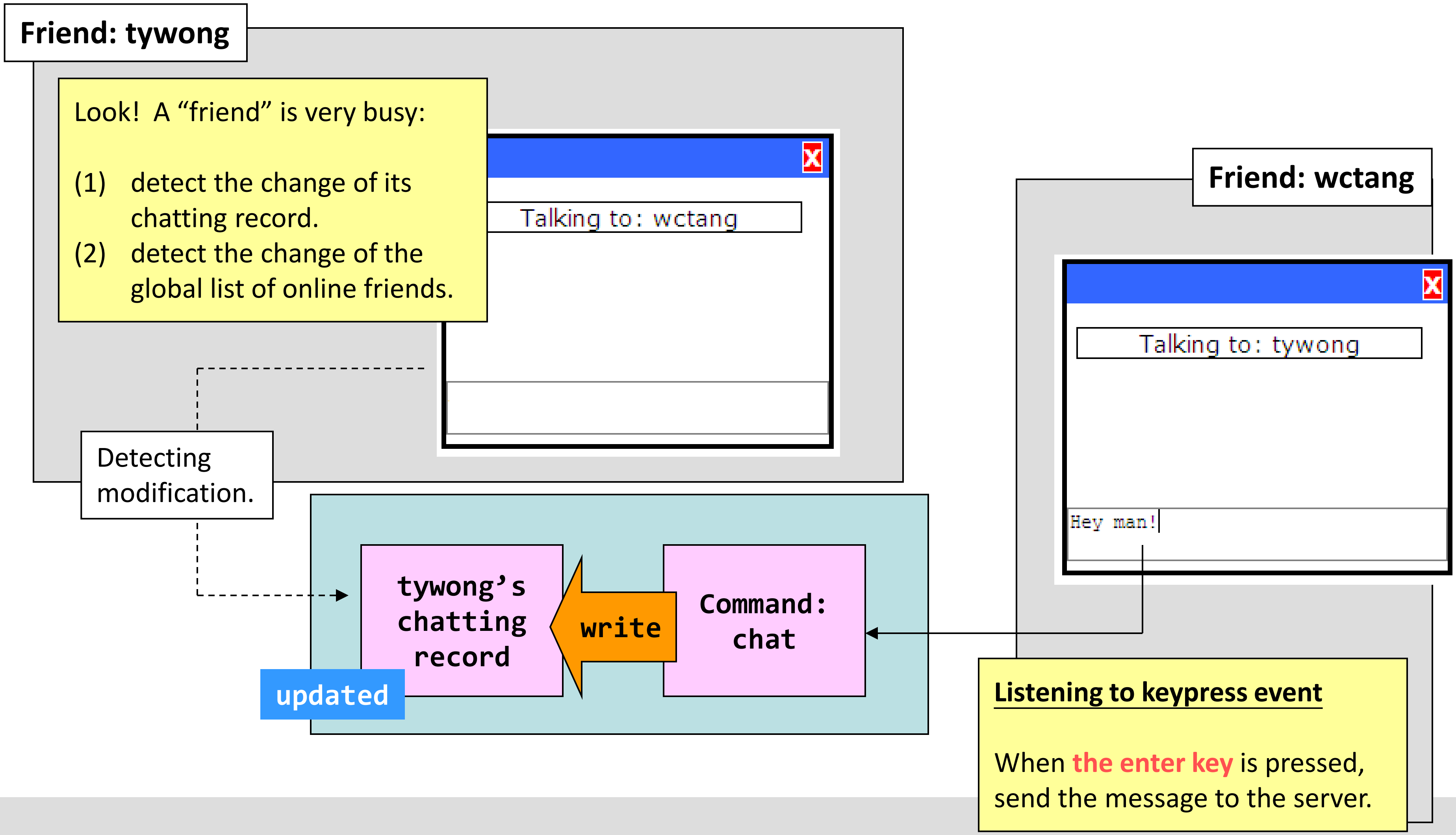


The Flow...

The comet technique is not difficult to understand...



The Flow...


Friend: tywong
Look! A "friend" is very busy:
(1) detect the change of its chatting record.
(2) detect the change of the global list of online friends.
Talking to: wctang
Detecting modification.
tywong's chatting record
updated
write
Command: chat
Friend: wctang
Talking to: tywong
Hey man!
Listening to keypress event
When the enter key is pressed, send the message to the server.

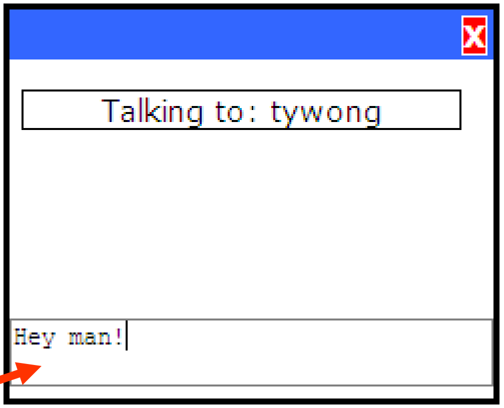
Sidetrack: **keypress** handler...

What is the difference between "**event.keyCode**" and "**event.charCode**"?

Well...different browsers has different answers.

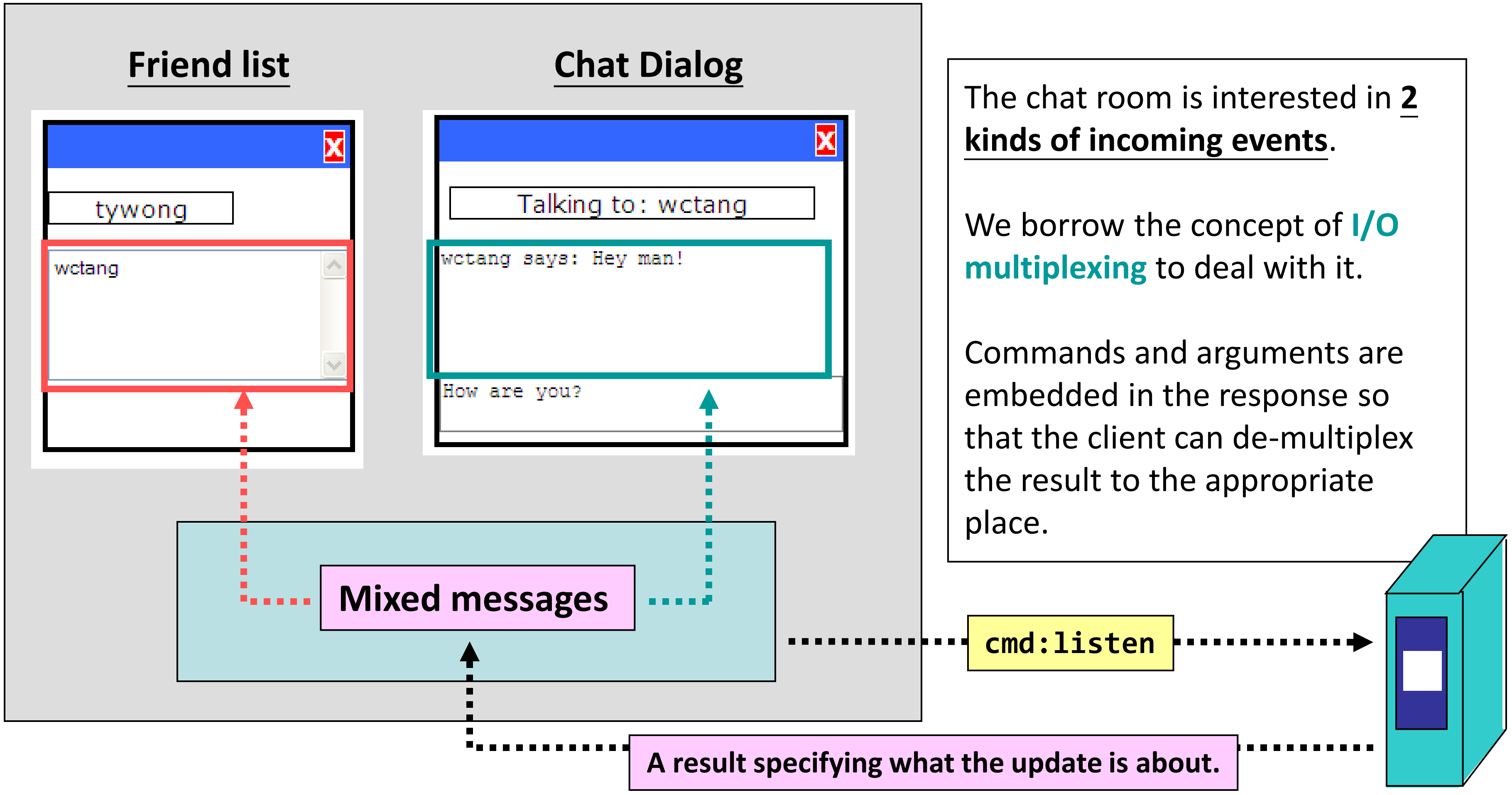
By the way, how can we handle the "Esc" key?

Question: how to handle the "**enter key**" only?



See "keypress.html" & "keyupdown.html" with different browsers!

The Flow…

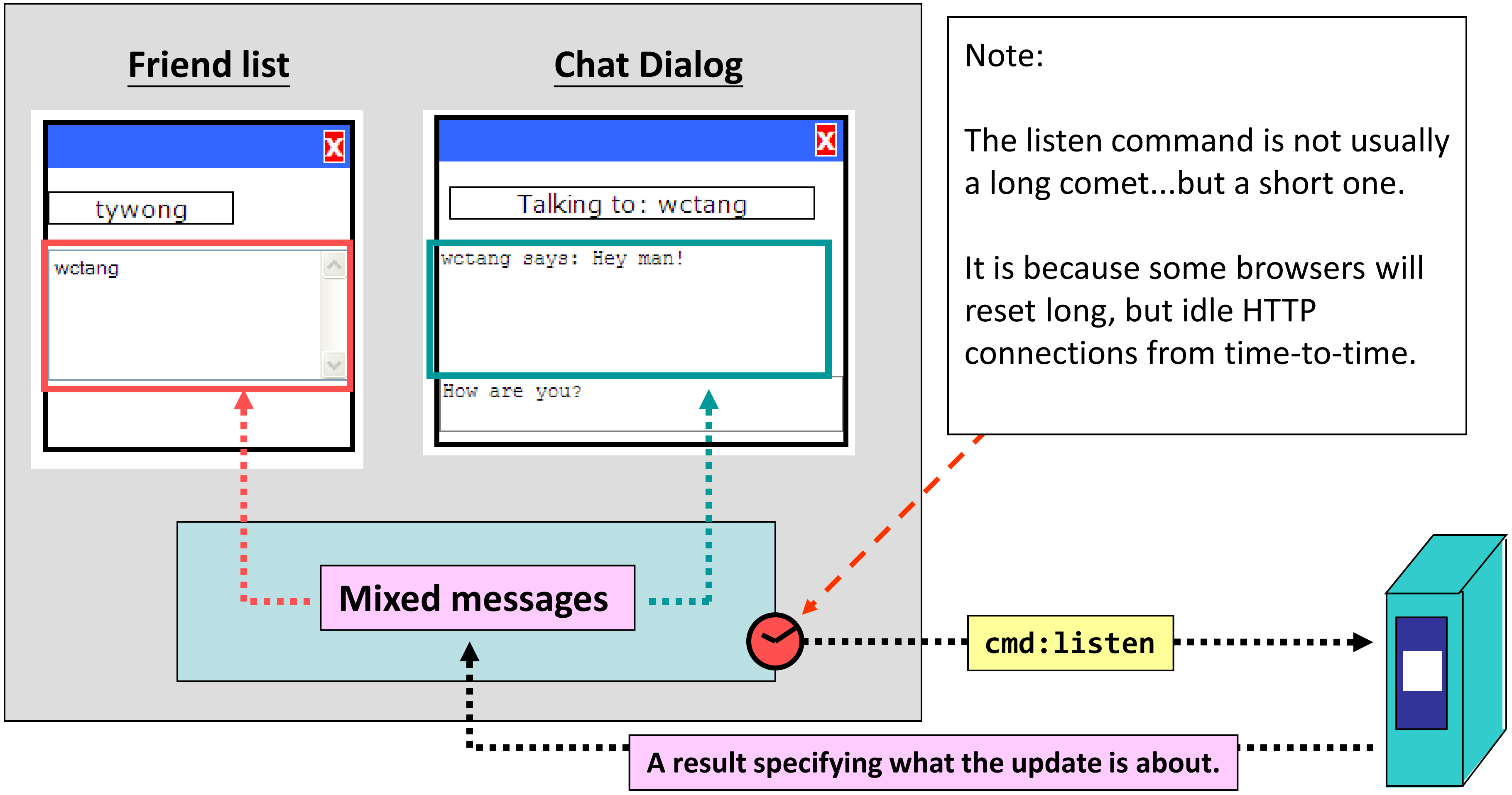


The chat room is interested in **2 kinds of incoming events**.

We borrow the concept of **I/O multiplexing** to deal with it.

Commands and arguments are embedded in the response so that the client can de-multiplex the result to the appropriate place.

The Flow…

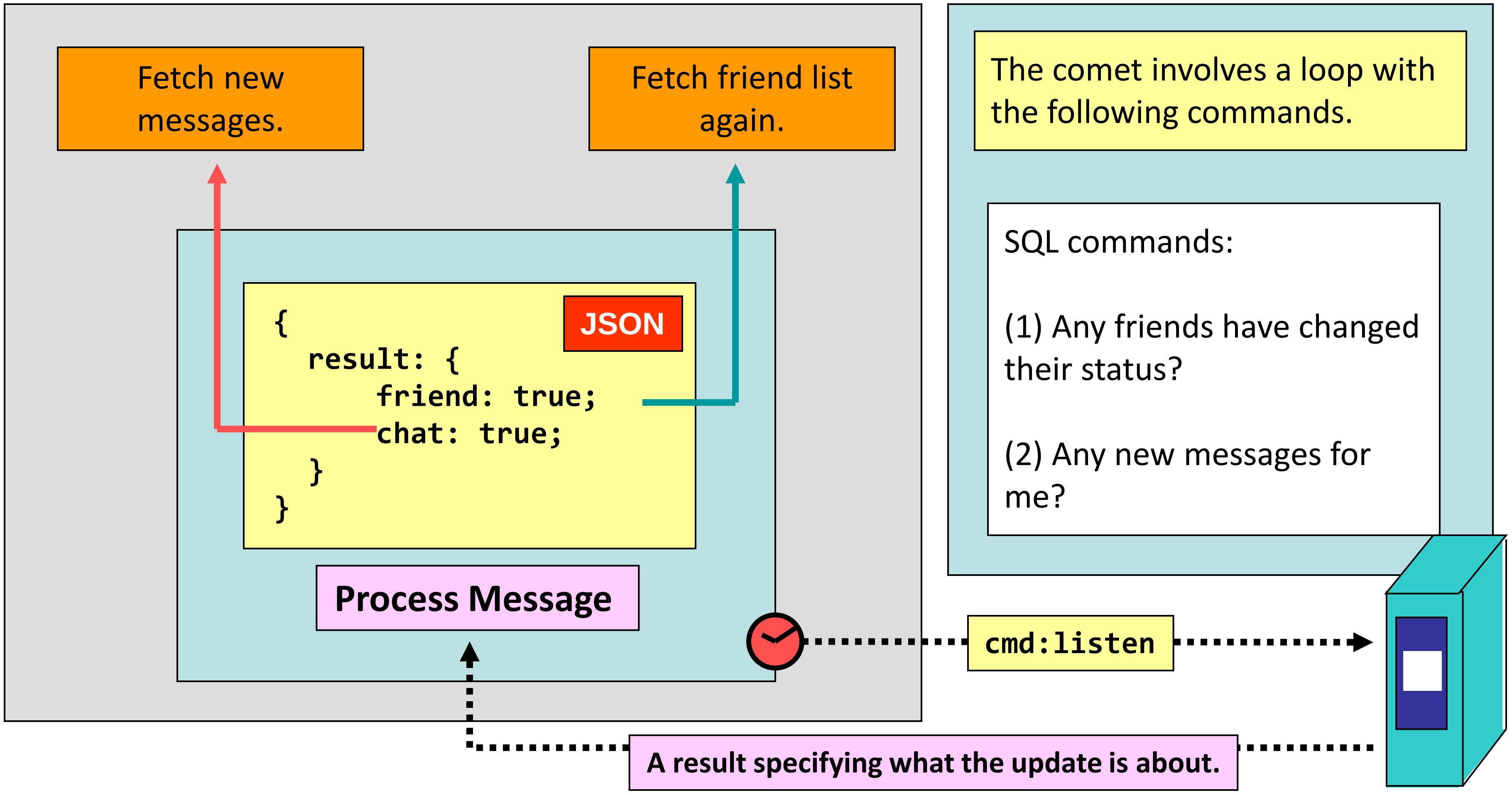


Note:

The listen command is not usually a long comet…but a short one.

It is because some browsers will reset long, but idle HTTP connections from time-to-time.

The Flow…



```
{
  result: {
      friend: true;
      chat: true;
  }
}
```

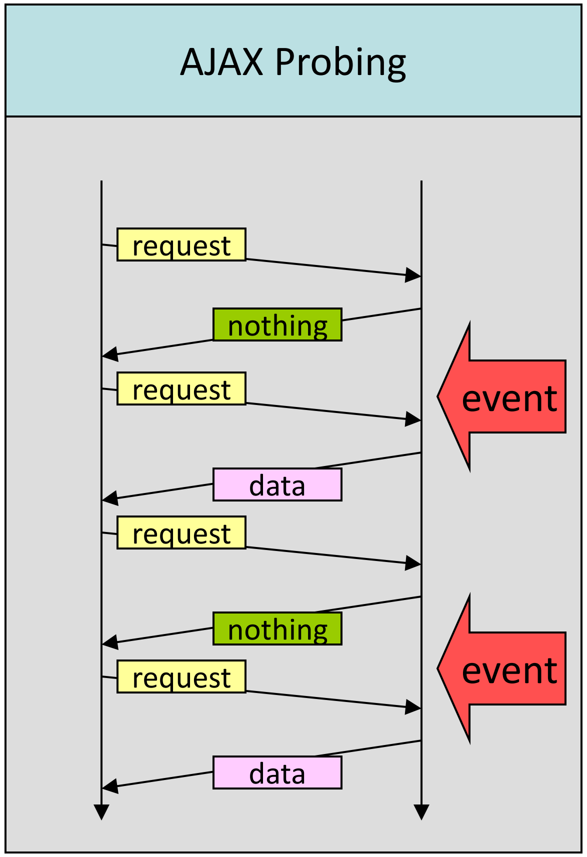
The comet involves a loop with the following commands.

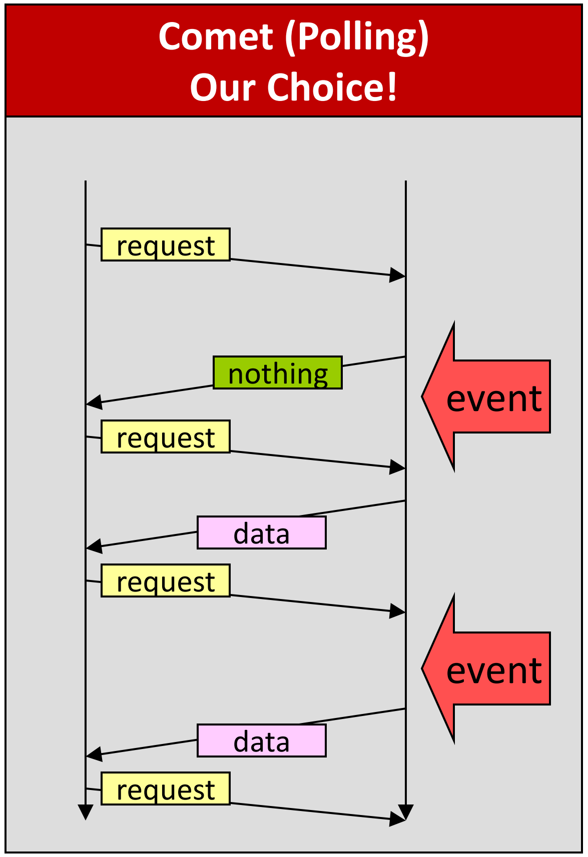
SQL commands:

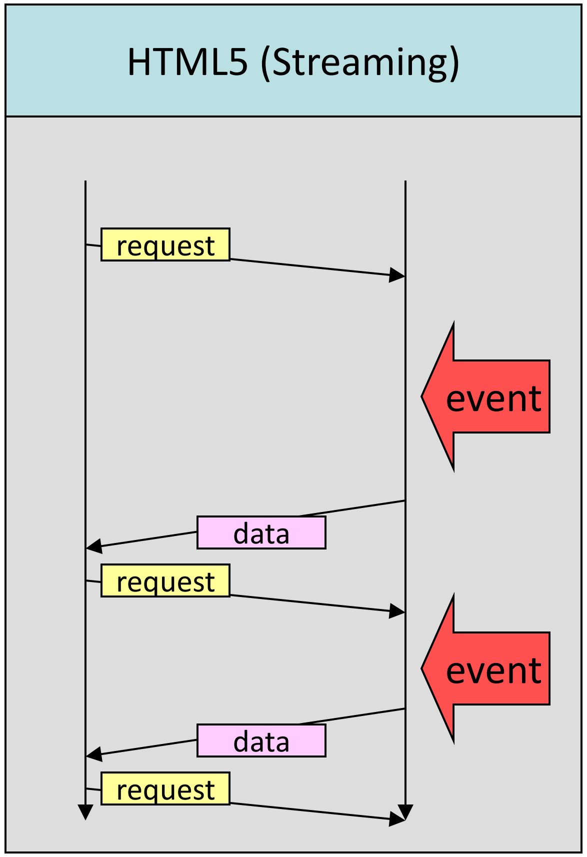
(1) Any friends have changed their status?

(2) Any new messages for me?

The Flow…







Unlike AJAX probing, short polling will wait or sleep for some time before replying.