DHTML: Object Model & Event Model

DHTML is about creating web pages that reacts to (user)events.

```
<body>
        <b>Mouse over the squares and the background color will change!</b>
        <table width="300" height="100">
            <tr>
                <td onmouseover="bgChange('red')"
                    onmouseout="bgChange('transparent')"
                    bgcolor="red">
                </td>
                <td onmouseover="bgChange('blue')"
                    onmouseout="bgChange('transparent')"
                    bgcolor="blue">
                </td>
                <td onmouseover="bgChange('green')"
                    onmouseout="bgChange('transparent')"
                    bgcolor="green">
                </td>
            </tr>
        </table>
    </body>
```

```
<head>
        <script type="text/javascript">
        function bgChange(bg)
        {
            document.body.style.background=bg;
        }
        </script>
    </head>
```

DHTML

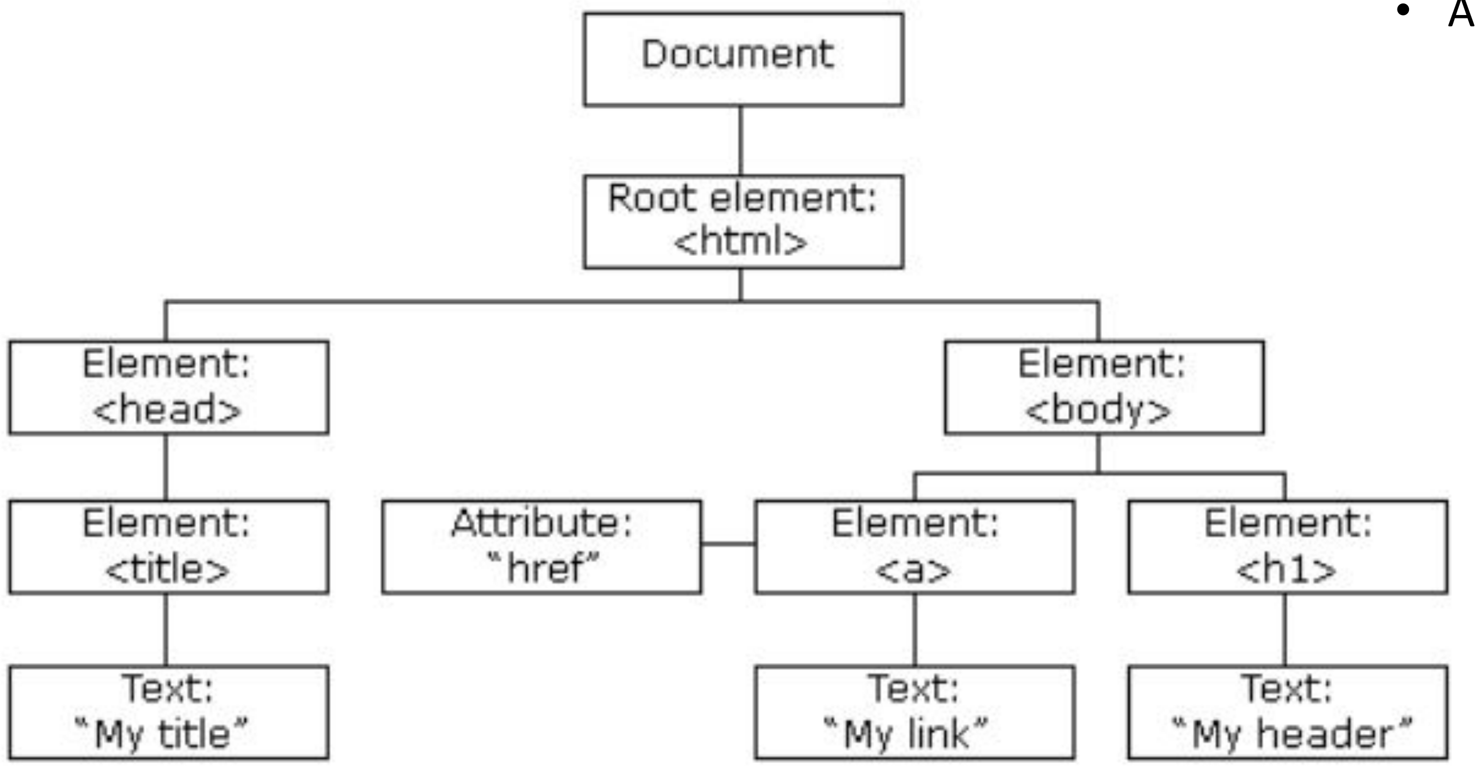
- DHTML is not a Language
- DHTML is a term describing the art of making dynamic and interactive web pages.
- DHTML combines HTML, JavaScript, DOM, and CSS.

According to the World Wide Web Consortium (W3C):
"Dynamic HTML is a term used by some vendors to describe the combination of HTML, style sheets and scripts that allows documents to be animated."

DHTML DOM

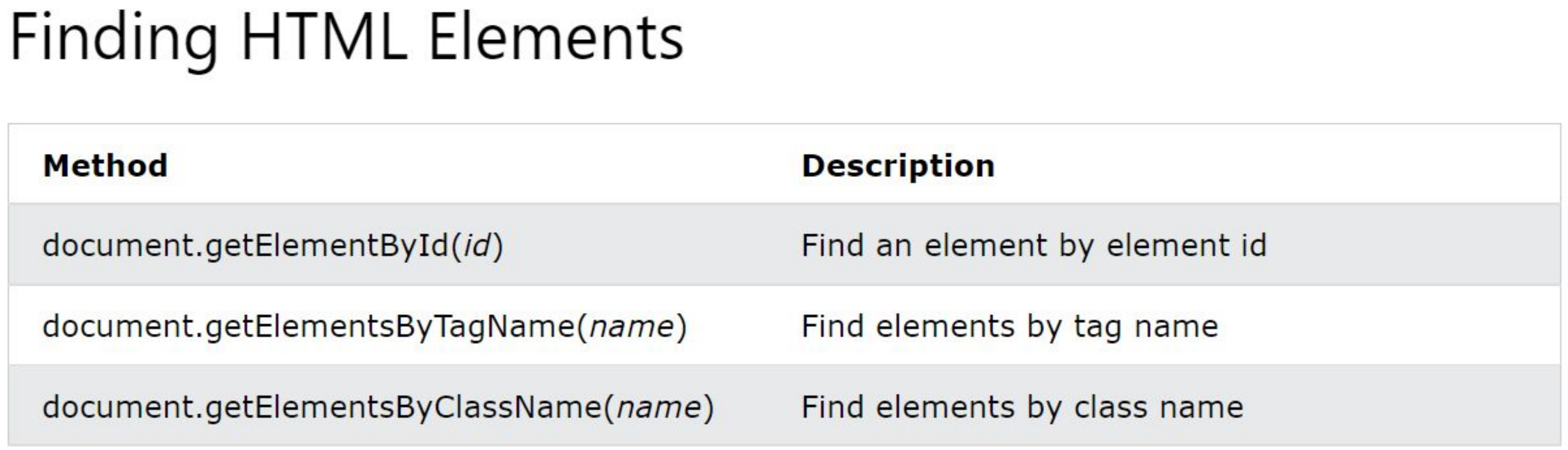
The HTML DOM:

- A Standard object model for HTML
- A standard programming interface for HTML
- Platform- and language-independent
- A W3C standard



DHTML DOM

- The document object represents your web page.



Finding HTML Elements

Method	Description
document.getElementById(*id*)	Find an element by element id
document.getElementsByTagName(*name*)	Find elements by tag name
document.getElementsByClassName(*name*)	Find elements by class name

Finding HTML Elements by CSS Selectors:

```
const x = document.querySelectorAll(“p.name”);
```

Changing HTML Elements

Property	**Description**
element.innerHTML = *new html content*	Change the inner HTML of an element
element.attribute = *new value*	Change the attribute value of an HTML element
element.style.*property* = *new style*	Change the style of an HTML element
Method	**Description**
element.setAttribute*(attribute, value)*	Change the attribute value of an HTML element

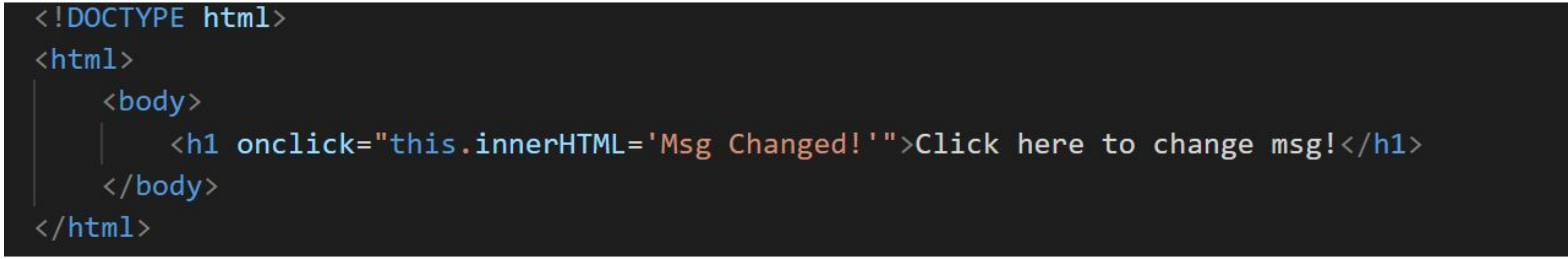
Adding and Deleting Elements

Method	**Description**
document.createElement(*element*)	Create an HTML element
document.removeChild(*element*)	Remove an HTML element
document.appendChild(*element*)	Add an HTML element
document.replaceChild(*new, old*)	Replace an HTML element
document.write(*text*)	Write into the HTML output stream

DOM Events

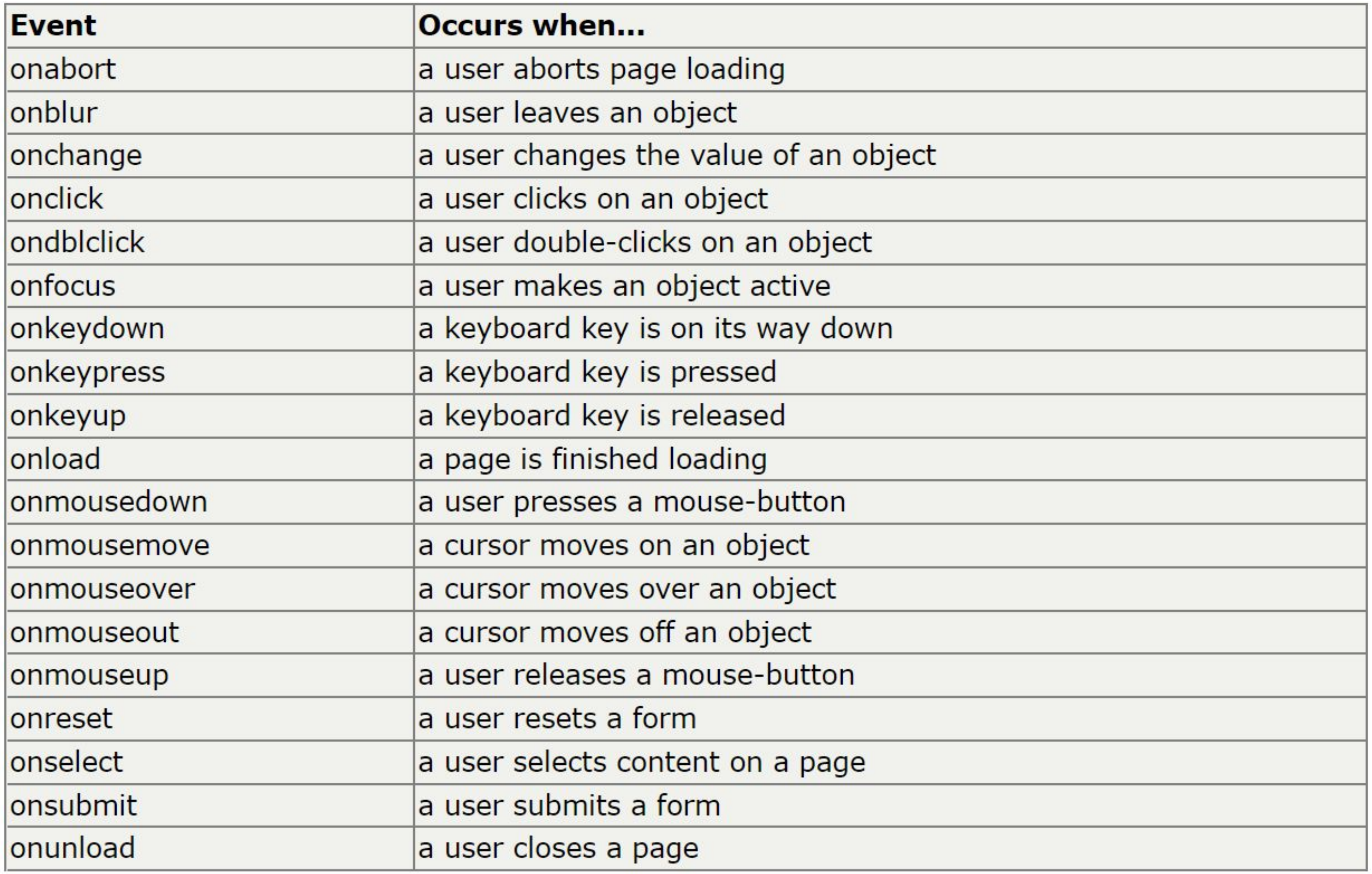
Examples of HTML events:

- When a user clicks the mouse
- When a web page has loaded
- When an image has been loaded
- When the mouse moves over an element
- When an input field is changed
- When an HTML form is submitted
- When a user strokes a key



```html
<!DOCTYPE html>
<html>
    <body>
        <h1 onclick="this.innerHTML='Msg Changed!'">Click here to change msg!</h1>
    </body>
</html>
```

List of Events

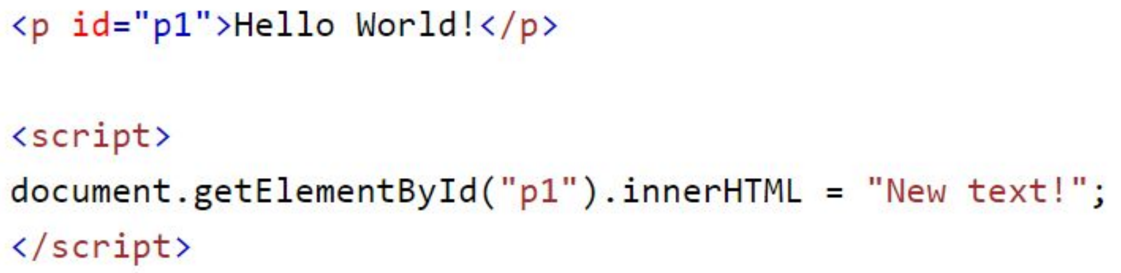


Event	Occurs when...
onabort	a user aborts page loading
onblur	a user leaves an object
onchange	a user changes the value of an object
onclick	a user clicks on an object
ondblclick	a user double-clicks on an object
onfocus	a user makes an object active
onkeydown	a keyboard key is on its way down
onkeypress	a keyboard key is pressed
onkeyup	a keyboard key is released
onload	a page is finished loading
onmousedown	a user presses a mouse-button
onmousemove	a cursor moves on an object
onmouseover	a cursor moves over an object
onmouseout	a cursor moves off an object
onmouseup	a user releases a mouse-button
onreset	a user resets a form
onselect	a user selects content on a page
onsubmit	a user submits a form
onunload	a user closes a page

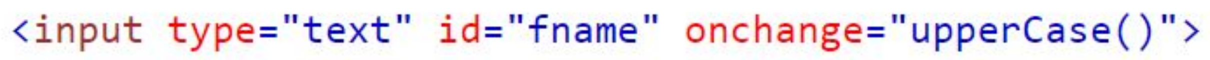
JavaScript

```
<script type="text/javascript">
    //javascript code
</script>
```

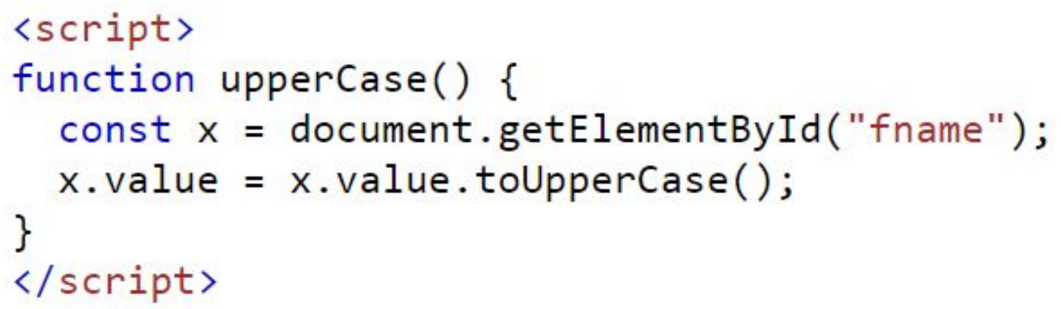
```
<script type="text/javascript">
    function fun_name(list_of_parameters){
        //body of the function;
    }
</script>
```



```
<p id="p1">Hello World!</p>

<script>
document.getElementById("p1").innerHTML = "New text!";
</script>
```

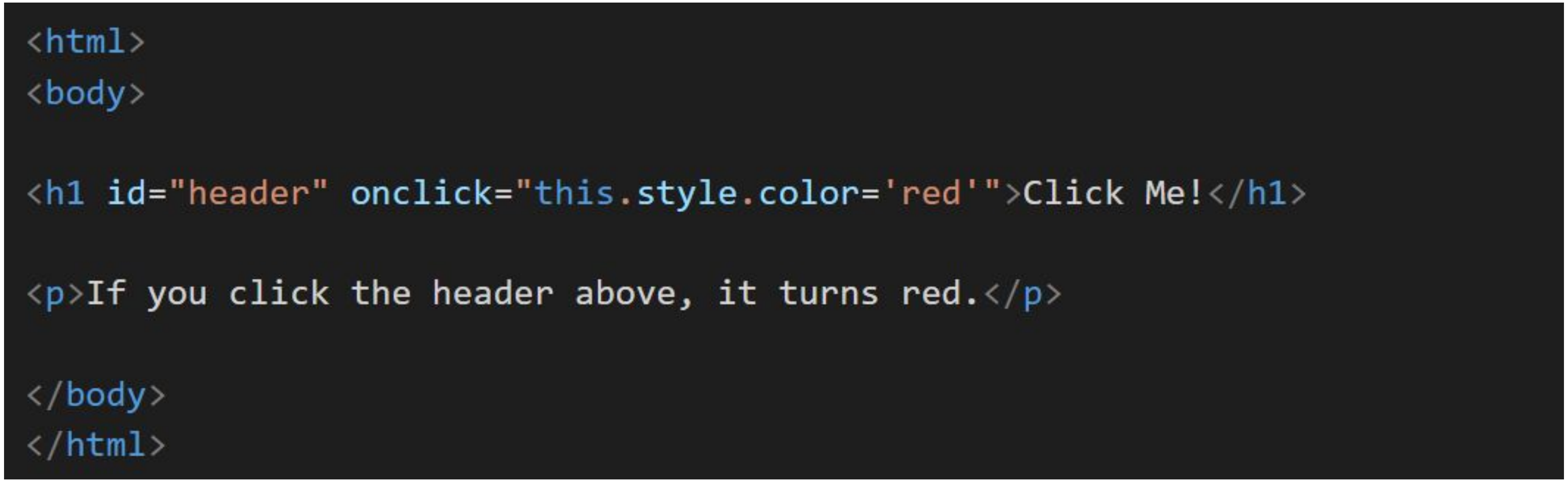


```
<input type="text" id="fname" onchange="upperCase()">
```

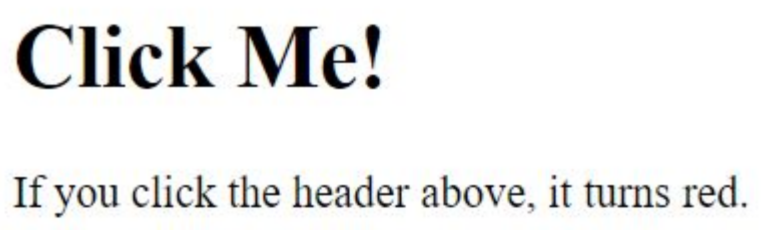


```
<script>
function upperCase() {
  const x = document.getElementById("fname");
  x.value = x.value.toUpperCase();
}
</script>
```

DHTML CSS

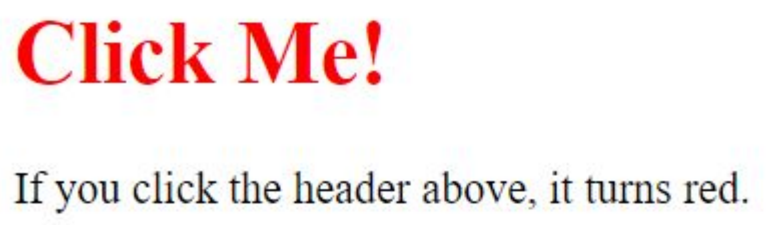


```html
<html>
<body>

<h1 id="header" onclick="this.style.color='red'">Click Me!</h1>

<p>If you click the header above, it turns red.</p>

</body>
</html>
```



Click Me!

If you click the header above, it turns red.

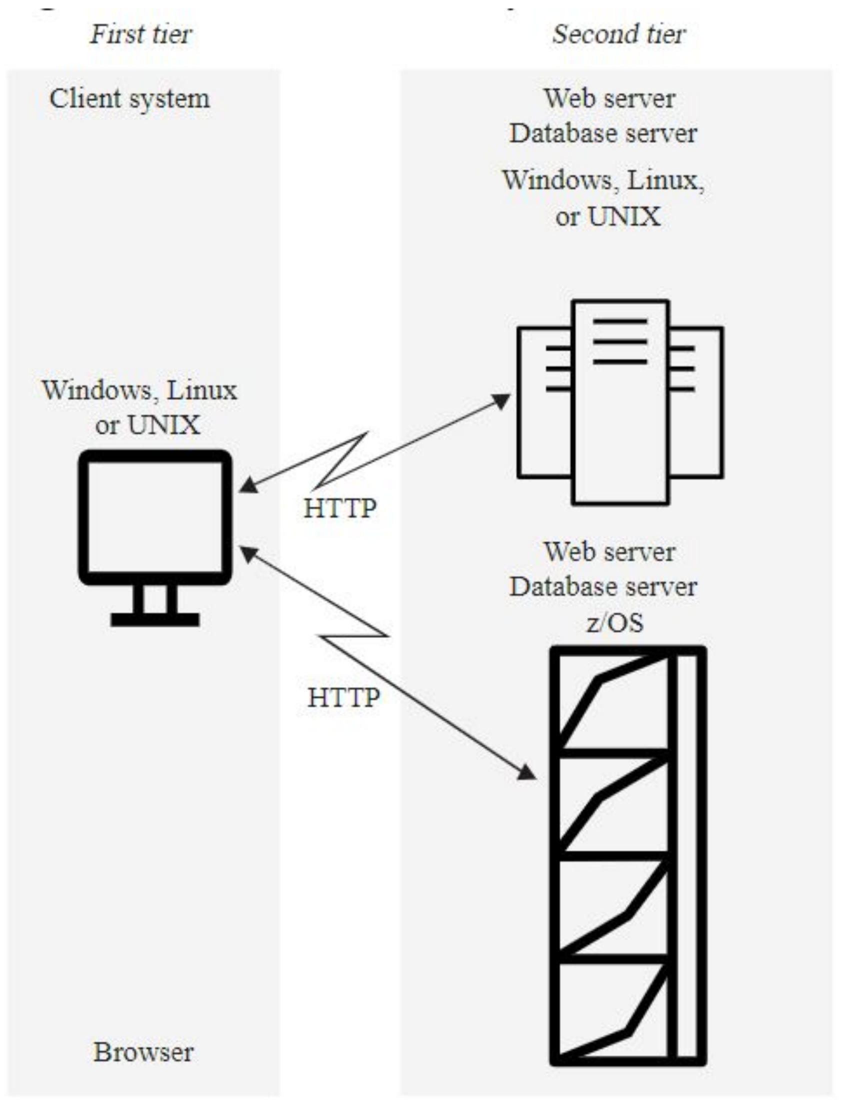


Click Me!

If you click the header above, it turns red.

Web Architectures: (Two-tier architecture)

- Client is on first-tier
- DB Server and Web Application Server is on Second-tier
- Second tier serves the Data and Business logic for the web application.
- The second tier is responsible for *availability, scalability*, and *performance* characteristics for the organization's web environment

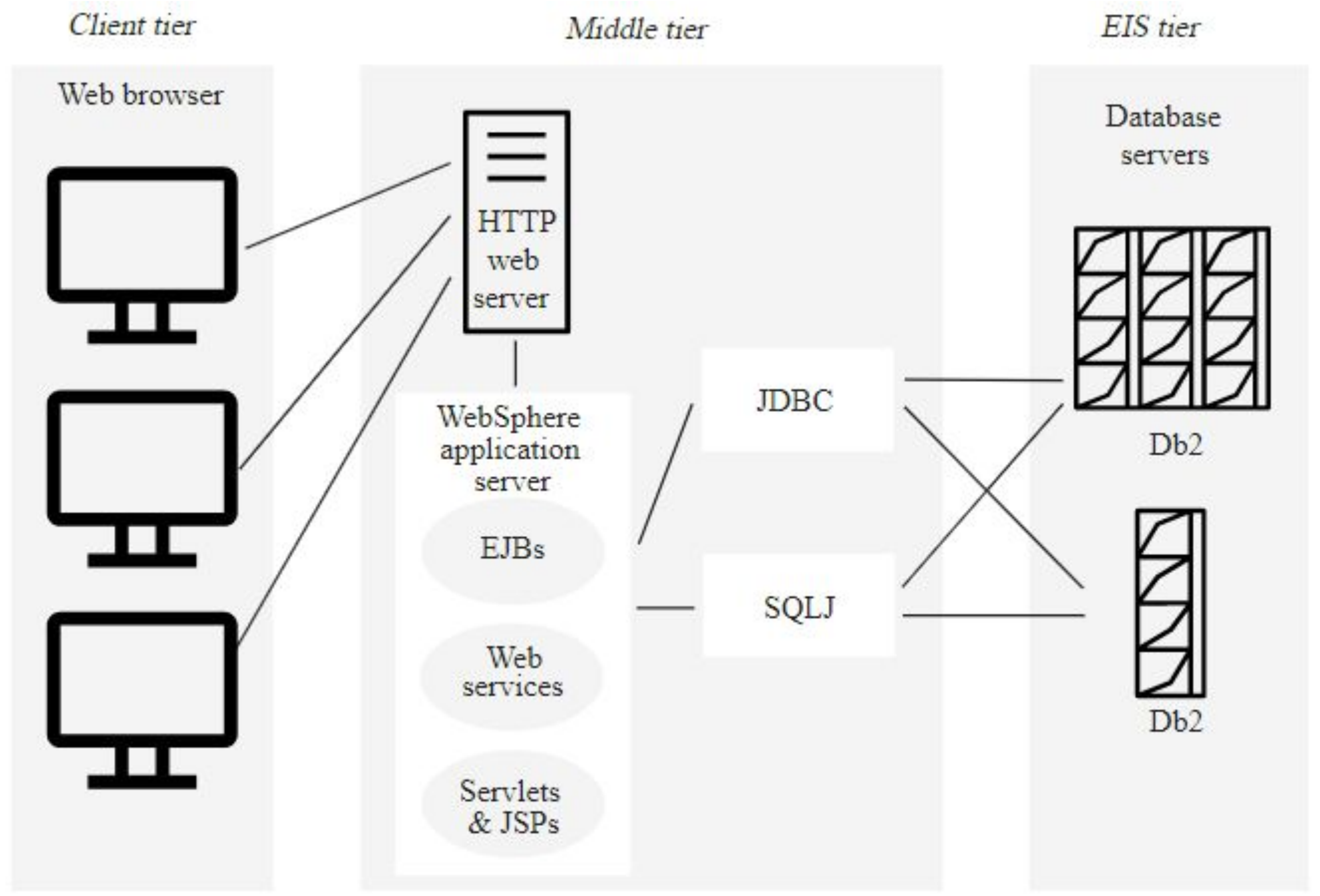


Web Architectures: (Three-tier architecture)

In three-tier architecture:

- *First-tier*: Client
- *Second-tier*: Application server
- *Third-tier*: Database Server

In this approach, hardware and software components of the second and third tiers share responsibility for the availability, scalability, and performance characteristics of the web environment.



Hypertext Transfer Protocol (HTTP)

- The Hypertext Transfer Protocol (HTTP)is used to request and serve web content.
- HTTP is plaintext protocol that runs on port 80
- To increase the security of internet many websites pushes to use HTTPS, which encrypts traffic using TLS and serves it over port 443.
- communication can be handled in the form
  - HTTP Request
  - HTTP Response
- Various tools to analyze HTTP Requests
  - Default ***Developer tools*** in web browsers
  - ***Wireshark*** is network protocol analyzer
  - …

HTTP Request

Start line

Header field(s)
(one or more)

Blank line

Message body
(optional)

Request Method

Request URI Portion of web address

HTTP Version (HTTP/1.1)

GET
POST
HEAD
OPTIONS
PUT
DELETE
TRACE

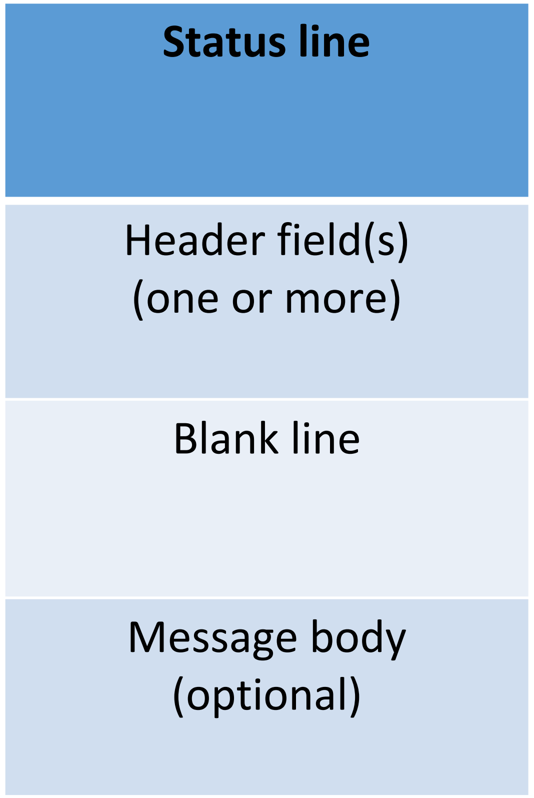
`https://secure.example.org/sec.txt`

`urn:ISBN:0-1404-4417-3`

```
POST /servlet/EchoHttpRequest HTTP/1.1
host: www.example.org:56789
user-agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.4)
  Gecko/20030624
accept: text/xml,application/xml,application/xhtml+xml,
  text/html;q=0.9,text/plain;q=0.8,video/x-mng,image/png,image/jpeg
  image/gif;q=0.2,*/*;q=0.1
accept-language: en-us,en;q=0.5
accept-encoding: gzip,deflate
accept-charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
connection: keep-alive
keep-alive: 300
content-type: application/x-www-form-urlencoded
content-length: 13
```

MIME Type
text/html
image/gif
image/jpeg
text/plain
application/octet-stream
application/x-www-form-urlencoded

HTTP Response

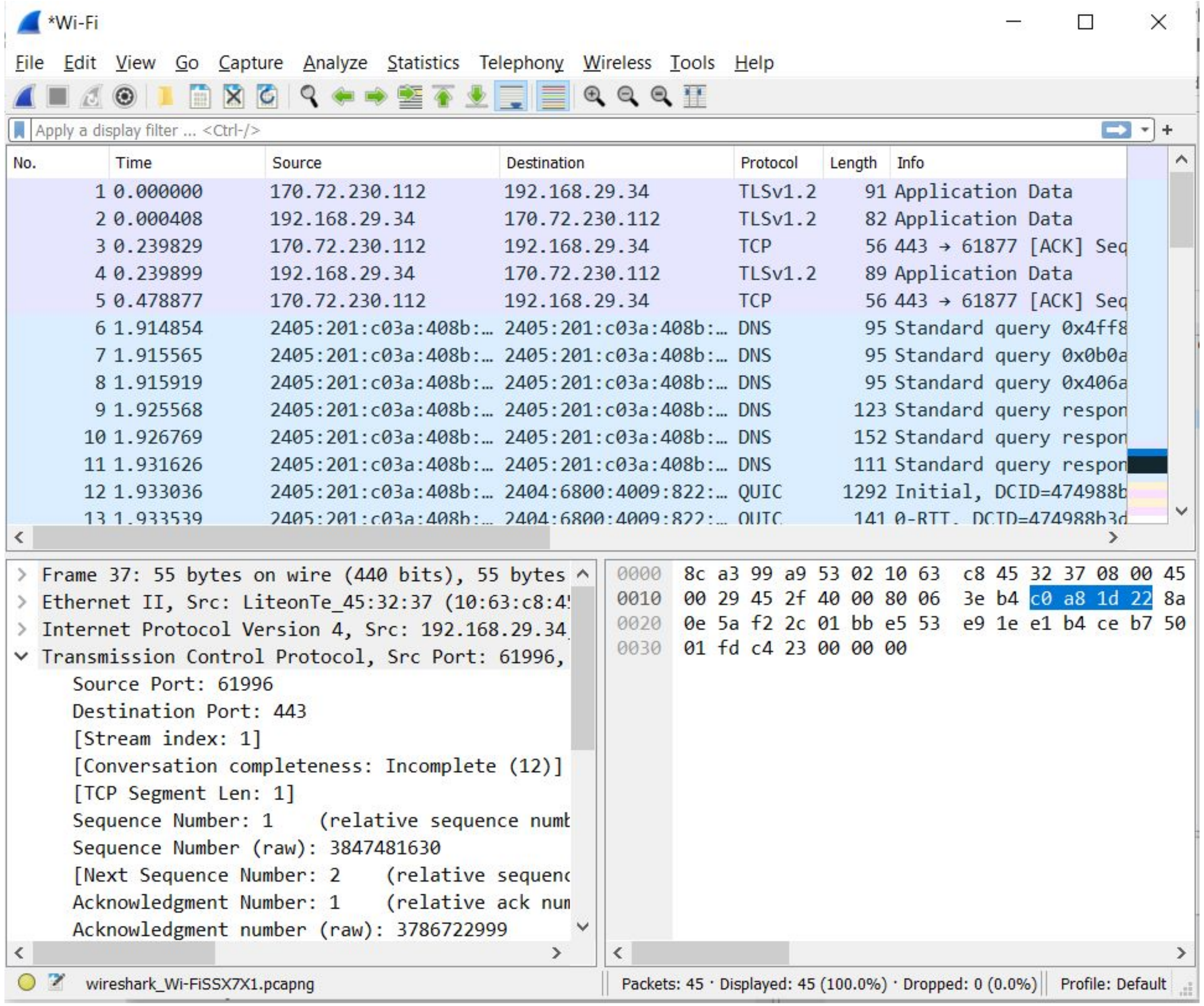


```
HTTP/1.1 200 OK
```

Digit	Class
1	Informational
2	Success
3	Redirection
4	Client Error
5	Server Error

Status Code	Recommended Reason Phrase
200	OK
301	Moved Permanently
307	Temporary Redirect
401	Unauthorized
403	Forbidden
404	Not Found
500	Internal Server Error

Wireshark


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

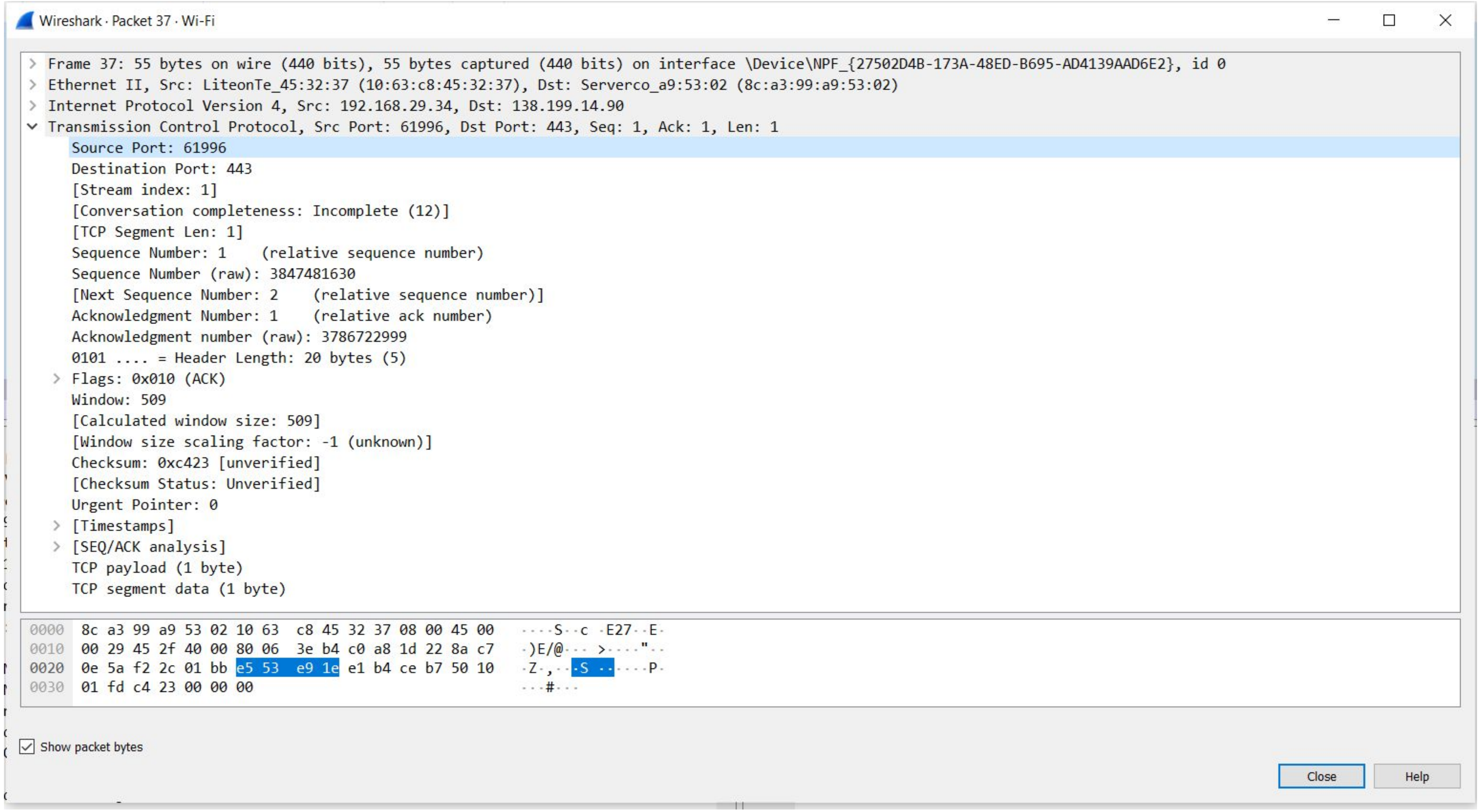
```
No.   Time       Source                 Destination            Protocol  Length Info
    1 0.000000   170.72.230.112         192.168.29.34          TLSv1.2     91 Application Data
    2 0.000408   192.168.29.34          170.72.230.112         TLSv1.2     82 Application Data
    3 0.239829   170.72.230.112         192.168.29.34          TCP         56 443 → 61877 [ACK] Seq
    4 0.239899   192.168.29.34          170.72.230.112         TLSv1.2     89 Application Data
    5 0.478877   170.72.230.112         192.168.29.34          TCP         56 443 → 61877 [ACK] Seq
    6 1.914854   2405:201:c03a:408b:…   2405:201:c03a:408b:…   DNS         95 Standard query 0x4ff8
    7 1.915565   2405:201:c03a:408b:…   2405:201:c03a:408b:…   DNS         95 Standard query 0x0b0a
    8 1.915919   2405:201:c03a:408b:…   2405:201:c03a:408b:…   DNS         95 Standard query 0x406a
    9 1.925568   2405:201:c03a:408b:…   2405:201:c03a:408b:…   DNS        123 Standard query respon
   10 1.926769   2405:201:c03a:408b:…   2405:201:c03a:408b:…   DNS        152 Standard query respon
   11 1.931626   2405:201:c03a:408b:…   2405:201:c03a:408b:…   DNS        111 Standard query respon
   12 1.933036   2405:201:c03a:408b:…   2404:6800:4009:822:…   QUIC      1292 Initial, DCID=474988b
   13 1.933539   2405:201:c03a:408b:…   2404:6800:4009:822:…   QUIC       141 0-RTT, DCID=474988b3d
```

```
> Frame 37: 55 bytes on wire (440 bits), 55 bytes
> Ethernet II, Src: LiteonTe_45:32:37 (10:63:c8:4
> Internet Protocol Version 4, Src: 192.168.29.34
v Transmission Control Protocol, Src Port: 61996,
    Source Port: 61996
    Destination Port: 443
    [Stream index: 1]
    [Conversation completeness: Incomplete (12)]
    [TCP Segment Len: 1]
    Sequence Number: 1    (relative sequence numb
    Sequence Number (raw): 3847481630
    [Next Sequence Number: 2    (relative sequenc
    Acknowledgment Number: 1    (relative ack num
    Acknowledgment number (raw): 3786722999
```

```
0000  8c a3 99 a9 53 02 10 63  c8 45 32 37 08 00 45
0010  00 29 45 2f 40 00 80 06  3e b4 c0 a8 1d 22 8a
0020  0e 5a f2 2c 01 bb e5 53  e9 1e e1 b4 ce b7 50
0030  01 fd c4 23 00 00 00
```

wireshark_Wi-FiSSX7X1.pcapng | Packets: 45 · Displayed: 45 (100.0%) · Dropped: 0 (0.0%) | Profile: Default

Wireshark



```
> Frame 37: 55 bytes on wire (440 bits), 55 bytes captured (440 bits) on interface \Device\NPF_{27502D4B-173A-48ED-B695-AD4139AAD6E2}, id 0
> Ethernet II, Src: LiteonTe_45:32:37 (10:63:c8:45:32:37), Dst: Serverco_a9:53:02 (8c:a3:99:a9:53:02)
> Internet Protocol Version 4, Src: 192.168.29.34, Dst: 138.199.14.90
v Transmission Control Protocol, Src Port: 61996, Dst Port: 443, Seq: 1, Ack: 1, Len: 1
    Source Port: 61996
    Destination Port: 443
    [Stream index: 1]
    [Conversation completeness: Incomplete (12)]
    [TCP Segment Len: 1]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 3847481630
    [Next Sequence Number: 2    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 3786722999
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x010 (ACK)
    Window: 509
    [Calculated window size: 509]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0xc423 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > [Timestamps]
  > [SEQ/ACK analysis]
    TCP payload (1 byte)
    TCP segment data (1 byte)
```

```
0000  8c a3 99 a9 53 02 10 63  c8 45 32 37 08 00 45 00   ····S··c ·E27··E·
0010  00 29 45 2f 40 00 80 06  3e b4 c0 a8 1d 22 8a c7   ·)E/@··· >····"··
0020  0e 5a f2 2c 01 bb e5 53  e9 1e e1 b4 ce b7 50 10   ·Z·,···S ·····P·
0030  01 fd c4 23 00 00 00                               ···#···
```