

# Web Programming

Pingmei Xu

# World Wide Web

- Wikipedia definition: a system of interlinked hypertext documents accessed via the Internet.



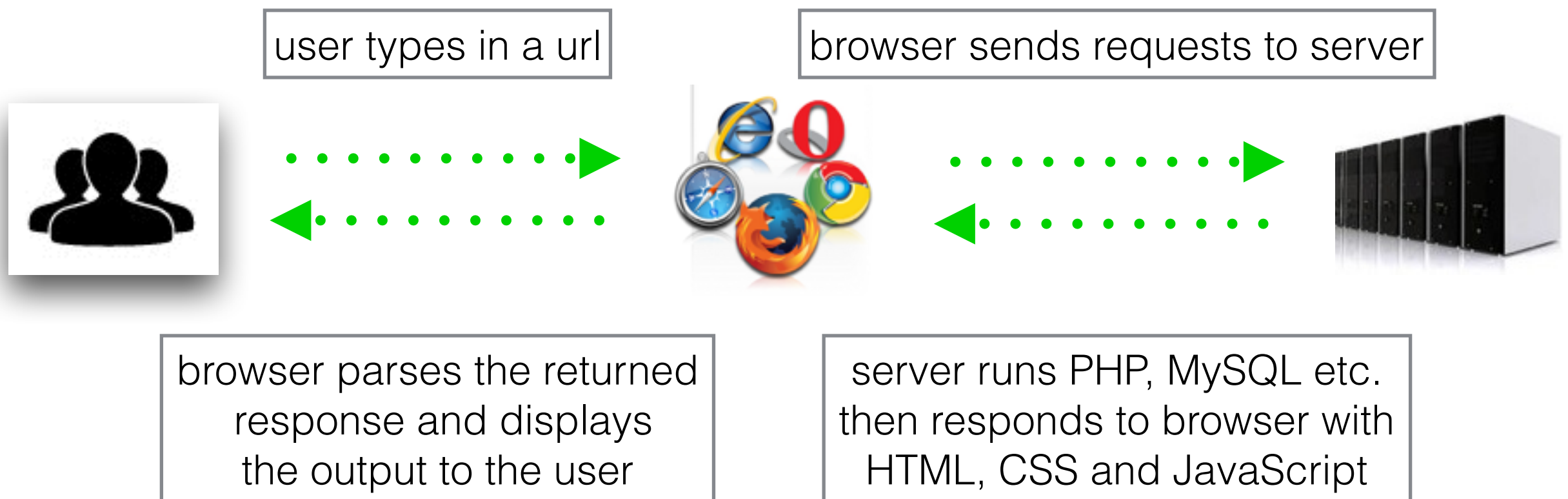
# Web $\neq$ Internet



**World Wide Web:** a collection of interlinked multimedia documents that are stored on the Internet and accessed using a common protocol (HTTP)

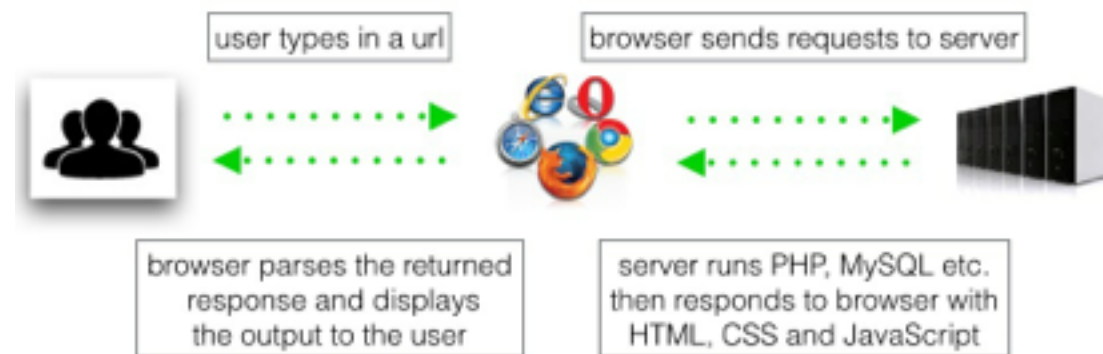
**Internet:** a physical network connecting millions of computers using the same protocols for sharing/transmitting information (TCP/IP)

# Web Programming





# Web Programming



## Static

- Web Document (HTML, CSS)

## Dynamic

- Client-side programming (JavaScript ...)  
can download program with Web page, execute on client machine
- Server-side programming (PHP, CGI, Perl ...)  
can store and execute program on Web server, link from Web page

HTML

# What is HTML?

HyperText Markup Language (HTML) is the core language of nearly all Web content.

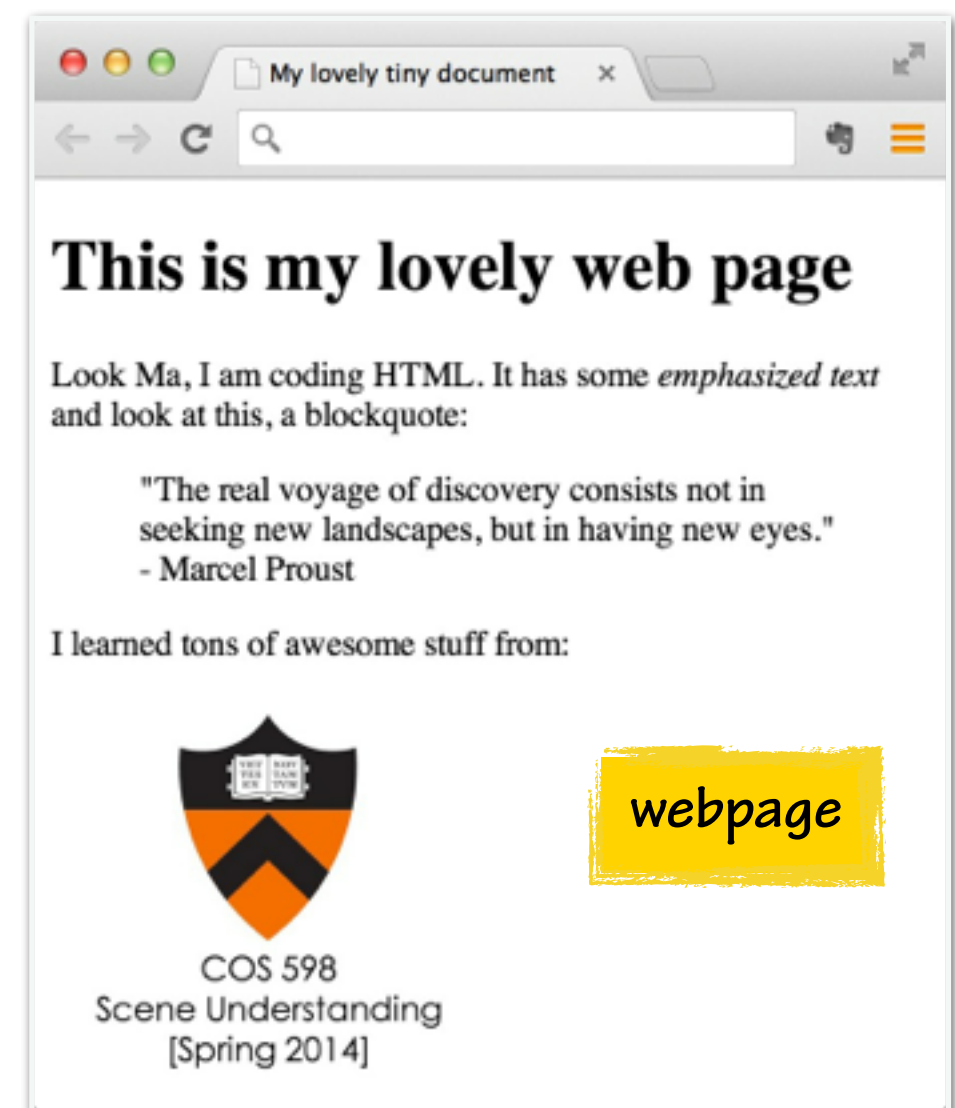
.html format



```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <title>My lovely tiny document</title>
5     <meta content="text/html; charset=UTF-8"/>
6   </head>
7   <body>
8     <!-- heading -->
9     <h1>This is my lovely web page</h1>
10    <p>Look Ma, I am coding HTML. It has some emphasized text
11    and look at this, a blockquote:
12    <blockquote>
13      <p>"The real voyage of discovery consists not in
14      seeking new landscapes, but in having new eyes."
15      - Marcel Proust</p>
16    </blockquote>
17    <p>I learned tons of awesome stuff from:
18    
19  </body>
20 </html>
```

Line 24, Column 1      Tab Size: 4      HTML

HTML code



webpage

# HTML: The Document Tree

This hierarchical structure is called the DOM:  
the Document Object Model.

head

```
<head>
  <title>My lovely tiny document</title>
  <meta content="text/html; charset=UTF-8"/>
</head>
```

body

```
<body>
  <!-- heading -->
  <h1>This is my lovely web page</h1>
  <p>
    Look Ma, I am coding <abbr title="Hyper Text Markup
    Language">HTML</abbr>. It has some <em> emphasized
    text</em> and look at this, a blockquote:
  </p>
  <blockquote>
    <p>"The real voyage of discovery consists not in
    seeking new landscapes, but in having new eyes."
    - Marcel Proust</p>
  </blockquote>
  <p> I learned tons of awesome stuff from:</p>
  
</body>
```

## This is my lovely web page

Look Ma, I am coding HTML. It has some *emphasized text* and look at this, a blockquote:

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."  
- Marcel Proust

I learned tons of awesome stuff from:



COS 598  
Scene Understanding  
[Spring 2014]



# HTML: Elements

Elements: the basic building blocks which define the semantic meaning of their content.

"<p>" element  
indicates a paragraph

the "<img>" element  
indicates an image

```
cos598demo.html
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <title>My lovely tiny document</title>
5      <meta content="text/html; charset=UTF-8" />
6    </head>
7    <body>
8      <!-- heading -->
9      <h1>This is my lovely web page</h1>
10     <p>
11       Look Ma, I am coding <abbr title="Hyper Text Markup
12         Language">HTML</abbr>. It has some <em>emphasized
13         text</em> and look at this, a blockquote:
14     </p>
15     <blockquote>
16       <p>"The real voyage of discovery consists not in
17         seeking new landscapes, but in having new eyes."
18         - Marcel Proust
19     </blockquote>
20     <p>I learned tons of awesome stuff from:</p>
21     
22   </body>
23 </html>
```


My lovely tiny document

## This is my lovely web page

Look Ma, I am coding HTML. It has some *emphasized text* and look at this, a blockquote:

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."  
- Marcel Proust

I learned tons of awesome stuff from:



COS 598  
Scene Understanding  
[Spring 2014]

Line 24, Column 1

Tab Size: 4

HTML

# HTML: Tags

cos598demo.html

UNREGISTERED

cos598demo.html

1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4 <h1>: opening tag  
5  
6  
7 </head>  
8 <body>  
9 <!-- heading -->  
10 <h1>This is my lovely web page</h1> <h1>: closing tag  
11 <p>  
12 Look Ma, I am coding <abbr title="Hyper Text Markup Language">HTML</abbr>. It has some <em>emphasized  
13 text</em> and look at this, a blockquote:  
14 </p>  
15  
16 empty elements like <img> doesn't need closing tag  
17  
18 seeking new landscapes, but in having new eyes."  
19 Marcel Proust</p>  
20 </blockquote>  
21 <p>I learned tons of awesome stuff from:</p>  
22   
23 </body>  
</html>

Line 24, Column 1

My lovely tiny document


< > ↺ 🔍 ☰

This is my lovely web page

Look Ma, I am coding HTML. It has some *emphasized text* and look at this, a blockquote:

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."  
- Marcel Proust

I learned tons of awesome stuff from:

  
COS 598  
Scene Understanding  
[Spring 2014]

Tab Size: 4 HTML

# HTML: Attributes

cos598demo.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <title>My lovely tiny document</title>
5     <meta content="text/html; charset=UTF-8"/>
6   </head>
7   <body>
8     <!-- heading -->
9     <h1>This is my lovely web page</h1>
10
11     <p>"The real voyage of discovery consists not in
12       seeking new landscapes, but in having new eyes."
13       - Marcel Proust</p>
14
15     </blockquote>
16     <p>I learned tons of awesome stuff from:</p>
17     
18
19   </body>
20 </html>
```

Line 24, Column 1

My lovely tiny document


← → ↻ 🔍 ☰

## This is my lovely web page

Look Ma, I am coding HTML. It has some *emphasized text* and look at this, a blockquote:

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."  
- Marcel Proust

I learned tons of awesome stuff from:



COS 598  
Scene Understanding  
[Spring 2014]

Tab Size: 4 HTML

Attributes usually consist of 2 parts:

An attribute name: e.g. width

An attribute value: e.g. 200

Text Markup  
> emphasized  
te:



# HTML: <img> Tag

cos598demo.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <title>My lovely tiny document</title>
5     <meta content="text/html; charset=UTF-8"/>
6   </head>
7   <body>
8     <!-- heading -->
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23 </body>
</html>
```

Line 24, Column 1

My lovely tiny document


← → ↻ 🔍 ☰

## This is my lovely web page

Look Ma, I am coding HTML. It has some *emphasized text* and look at this, a blockquote:

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."  
- Marcel Proust

I learned tons of awesome stuff from:

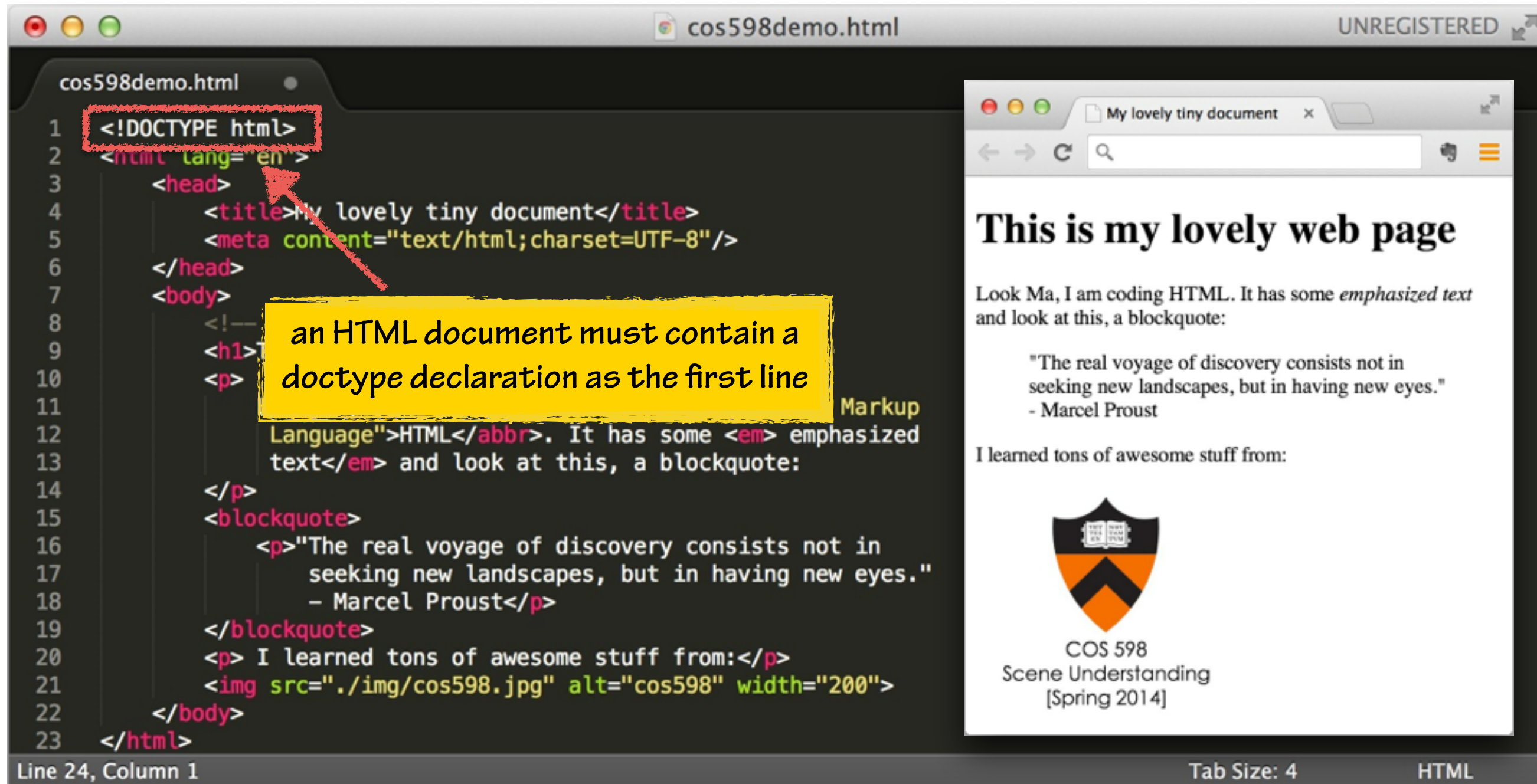


COS 598  
Scene Understanding  
[Spring 2014]

The <img> tag defines an image in an HTML page.  
It has two required attributes:  
src - specifies the URL of an image  
alt - specifies an alternate text for an image



# HTML: Doctype and Comments



# In-Class Exercise

- Task List

1. Change the content shown on the tab
2. Create a paragraph and write some stuff in that paragraph
3. Change the width of the image to 500px

- Cheat sheet

1. `<title> ... </title>`
2. `<p> ... </p>`
3. `<img ... width="...">`

# CSS

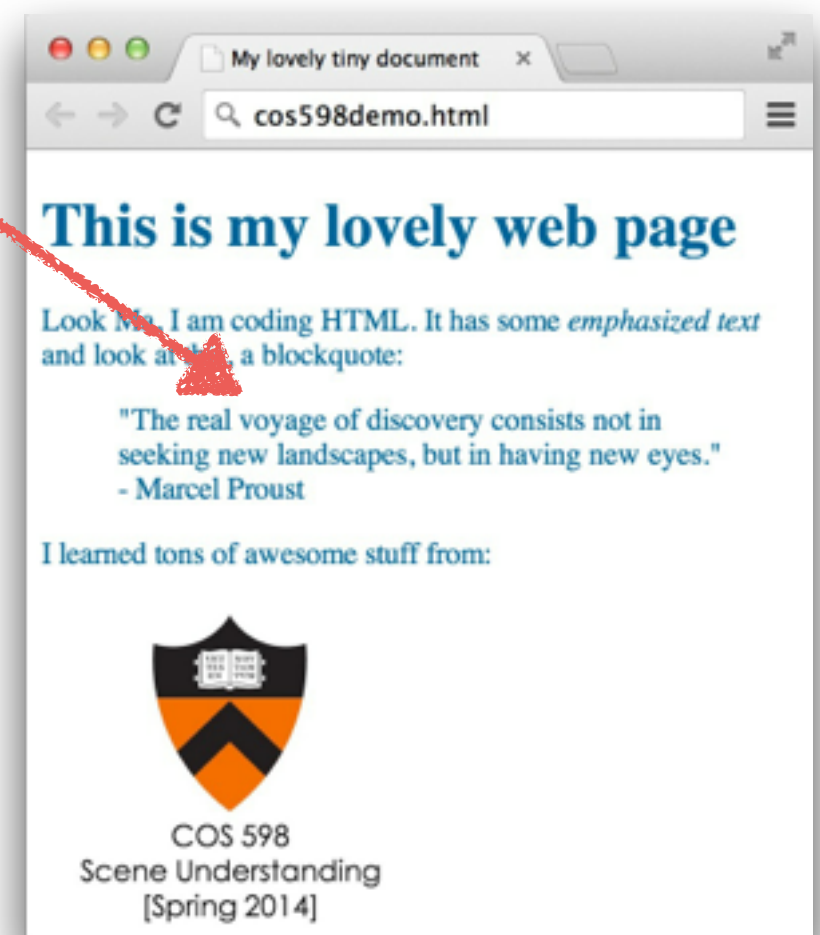
some examples borrowed from <https://developer.mozilla.org/>

# What is CSS?

Cascading Style Sheets (CSS) is a language for specifying how documents are presented to users.

```
<head>
  <title>My lovely tiny document</title>
  <meta content="text/html; charset=UTF-8" />
  <style type="text/css">
    body {color:#006699;}
  </style>
</head>
```

define the color of body



## CSS syntax

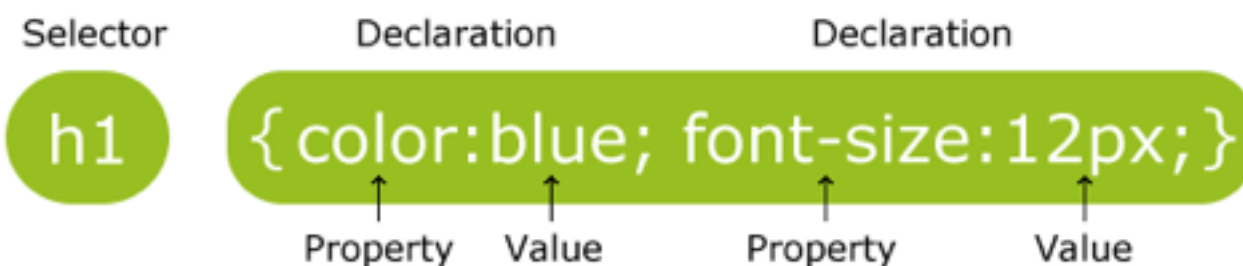


image from w3school

# Why Use CSS?

Keep the information content of a document separate from the details of how to display it (also know as 'style') so that you can:

- Avoid duplication
- Make maintenance easier
- Use the same content with different styles for different purposes



# How to Insert CSS?

- External Style Sheet

```
<head>  
  <link rel="stylesheet" type="text/css" href="mystyle.css">  
</head>
```

- Internal Style Sheet

```
<head>  
  <style type="text/css">  
    body {color:#006699;}  
  </style>  
</head>
```

- Inline Styles

```
<p style="color:sienna;margin-left:20px;">This is a paragraph.</p>
```



# CSS Box Model and Positioning

All HTML elements can be considered as boxes.



You can specify an element's position in four ways:  
***relative, fixed, absolute, static***

# In-Class Exercise

- Task List

1. Write CSS as **internal style sheet** and change the color of the body content to **'#FF0000'**
2. Write CSS as **inline style sheet** and change the font size of one paragraph to **'50px'**

- Cheat sheet

1. `<style> ... </style>`

```
<head>
  <style type="text/css">
    body {color:#006699;}
  </style>
</head>
```

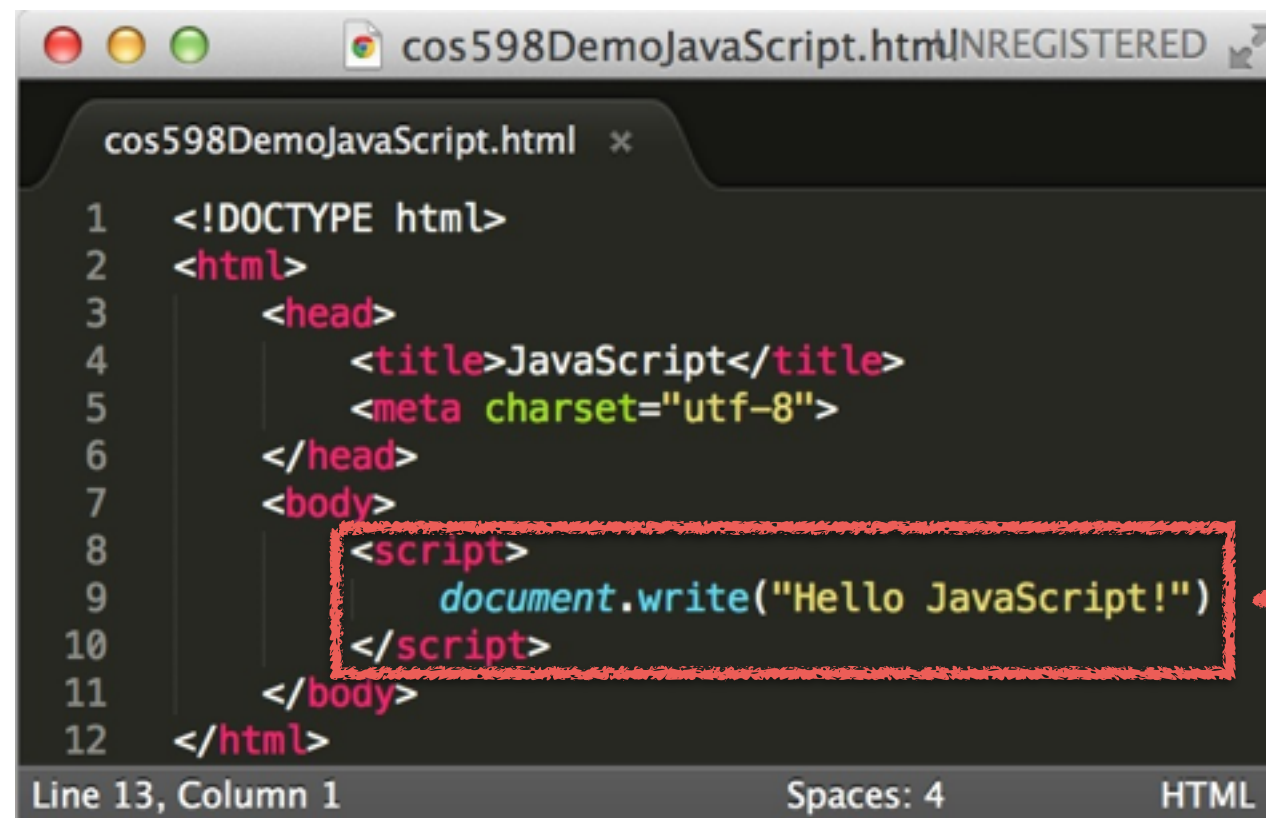
2. `<p style="font-size:...">`

```
<p style="color:sienna;margin-left:20px;">This is a paragraph.</p>
```

JavaScript

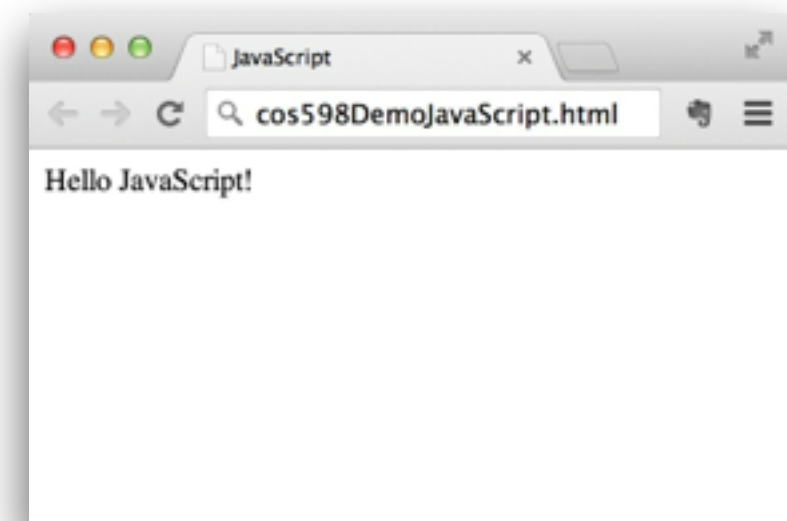
# What is JavaScript?

A scripting language. JavaScript code can be inserted into any HTML page, and it can be executed by all types of web browsers.



```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>JavaScript</title>
5     <meta charset="utf-8">
6   </head>
7   <body>
8     <script>
9       document.write("Hello JavaScript!")
10    </script>
11  </body>
12 </html>
```

Line 13, Column 1      Spaces: 4      HTML



put all your JavaScript  
code in <script> block

# JavaScript: Syntax

- Comment: `//` or `/* */`
- Variables: `var x = 0`
- Data types: `undefined`, `null`, `number`, `string`, `boolean` ...
- Operators: `+`, `-`, `*`, `/`, `%`, `>`, `<` ...
- Control structure: `if... else`, `for`, `while`, `switch` ...
- Native objects: `array`, `date`, `error`, `math`

# JavaScript: Object

```
> var course = new Object();  
course.coursename = "Scene Understanding";  
course.year = 2014;  
course.instructor = "Jianxiong Xiao";
```

create an  
JavaScript object

```
> course.instructor  
"Jianxiong Xiao"
```

accessing object properties:  
objectName.propertyName

```
> course.instructor.length  
14
```

```
> course.instructor.toUpperCase()  
"JIANXIONG XIAO"
```

accessing object methods:  
objectName.methodName()

# JavaScript: Array

For Matlab users,  
note: index starts from zero!

```
> var course = new Array();  
course[0] = "Scene Understanding";  
course[1] = 2014;  
course[2] = "Jianxiong Xiao";
```

create an array: you  
can have different  
objects in on array

```
> course  
["Scene Understanding", 2014, "Jianxiong Xiao"]  
> course[0]  
"Scene Understanding"
```

access data

```
> course[0] = "Computer Vision";  
> course  
["Computer Vision", 2014, "Jianxiong Xiao"]
```

modify data



# JavaScript: Math

- Constants:
  - `Math.PI`
- Methods:
  - `Math.round()`, `Math.floor()`, `Math.ceil()`
  - `Math.sin()`, `Math.cos()`
  - `Math.abs()`
  - `Math.sqrt()`, `Math.pow()`, `Math.exp()`
  - `Math.max()`, `Math.min()`
  - `Math.random()`

# JavaScript: Function

- A simple example

The diagram illustrates a JavaScript function and its usage. It features a dark-themed code editor with the following code:

```
> function addTwoNumbers(a,b)
{
    return a+b;
}

> var a = 1, b = 2;
> var c = addTwoNumbers(a,b);
> console.log(c);
3
```

Annotations with yellow boxes and red arrows point to specific parts of the code:

- function name**: Points to `addTwoNumbers` in the function definition.
- arguments**: Points to `a,b` in the function definition.
- call a function**: Points to `addTwoNumbers(a,b)` in the function call.

- Important concept: callback functions

# JavaScript: Library

- To use a JavaScript framework library in your web pages, just include the library in a `<script>` tag:

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>
```

- Computer Vision library:

**jsfeat** GitHub

<http://inspirit.github.io/jsfeat/>

# In-Class Exercise

- Write a function to compute the multiplication of two numbers
- Define two variables num1=5, num2=7
- Define another variable num3, set the value as the multiplication of num1 and num2 using the function that you just created, and check the value of num3

```
> function addTwoNumbers(a,b)
{
  return a+b;
}

> var a = 1, b = 2;
> var c = addTwoNumbers(a,b);

> console.log(c);
3
```

# jQuery

some examples borrowed from <http://www.w3schools.com/>

# What is jQuery?

- jQuery is a lightweight JavaScript library to make it much easier to use JavaScript on your website.
- Install jQuery
  - Download the jQuery library from [jquery.com](http://jquery.com)

```
<head>  
<script src="jquery-1.11.0.min.js"></script>  
</head>  
  
<\u009d>
```

- Include jQuery from a CDN, like Google

```
<head>  
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>  
</head>  
  
<\u009d>
```

# jQuery Syntax

- `$(selector).action()`

\$ sign to  
access jQuery

A (selector) to "query  
(or find)" HTML elements

A jQuery action() to be  
performed on the element(s)

```
$("#p").click(function(){  
    // action goes here!!  
    $(this).hide(); // hide the current <p> element  
});
```

- Some commonly used DOM events:

Mouse Events	Keyboard Events	Form Events	Document/Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload



# jQuery Basics

```
<!-- not use jquery -->
<input type="button" id="mybutton" onclick="alertMsg()" value="cos598button"/>
<script type="text/javascript">
    function alertMsg(){
        alert('someone clicked me!');
    }
    window.onload=function(){
        document.getElementById('mybutton').click();
    };
</script>
```

```
<!-- use jquery -->
<input type="button" id="mybutton" value="cos598button"/>
<script type="text/javascript">
    $(document).ready(function(){
        $('#mybutton').click(function(){
            alert('someone clicked me!');
        });
        $('#mybutton').click();
    });
</script>
```

It is good practice to wait for the document to be fully loaded and ready before working with it.

# jQuery Tablesorter

Tablesorter is a jQuery plugin for turning a standard HTML table with THEAD and TBODY tags into a sortable table without page refreshes: <http://tablesorter.com/>



My Lovely Classmates

COS 598  
Scene Understanding  
[Spring 2014]

Last Name	First Name	Due
Sema	Berkiten	\$15.00
Huiwen	Chang	\$100.00
Xinyi	Fan	\$20.00
Maciej	Halber	\$50.00
Zhirong	Wu	\$-5.00
Yinda	Zhang	\$10.00

sort by first name in descending order



My Lovely Classmates

COS 598  
Scene Understanding  
[Spring 2014]

Last Name	First Name	Due
Huiwen	Chang	\$100.00
Maciej	Halber	\$50.00
Xinyi	Fan	\$20.00
Sema	Berkiten	\$15.00
Yinda	Zhang	\$10.00
Zhirong	Wu	\$-5.00

sort by money in ascending order

# jQuery Tablesorter

Step 1: create a table in HTML

```
<table id="mycos598Table" class="tablesorter">
<thead>
<tr>
<th><span>Last Name</span></th>
<th><span>First Name</span></th>
<th><span>Due</span></th>
</tr>
</thead>
<tbody>
<tr>
<td>Huiwen</td>
<td>Chang</td>
<td>$100.00</td>
</tr>
<tr>
<td>Xinyi</td>
<td>Fan</td>
<td>$20.00</td>
</tr>
</tbody>
</table>
```

Step 2: include a single line of code

```
<script type="text/javascript">
$(document).ready(function() {
    $("#mycos598Table").tablesorter();
});
</script>
```

Step 3: sit there and collect money from your classmates



Last Name	First Name	Due ↕
Huiwen	Chang	\$100.00
Maciej	Halber	\$50.00
Xinyi	Fan	\$20.00
Sema	Berkiten	\$15.00
Yinda	Zhang	\$10.00
Zhirong	Wu	\$-5.00



# JavaScript Minification

- Minification is the process of removing all unnecessary characters from source code without changing its functionality.

## what's in jquery.min.js:

```

/*! jQuery v1.9.1 | (c) 2005, 2012 jQuery Foundation, Inc. | jquery.org/license
**/ @ sourceMappingURL=jquery.min.map
;(function(e,t){var n,r,i=typeof t,o=e.document,a=e.location,s=e.jQuery,u=e.$,l={},c=[],p="1.9.1",f=c.concat,d=c.push,h=c.slice,g=c.indexOf,m=l.toString,v=l.hasOwn,w=l.noop,y=p.trim,b=function(e,t){t{return new b.fn.init(e,t,r)},x=[+!/?:\d\.\-|?[\d\.\-]?[\d+\-|\.source,w/\S+/g,T="/^\s\s\uFFFF\xAB]+\s\s\uFFFF\xAB]+$/g,E=/^?:<(\w\W)+>[>]#?([ \w-]*)$/,C=/^(<\w+)>\s+\/>?:<\/\>|\$|$/,k=/^\[/,z:{\$}\$/g,E=/^?:^|:|,)?:(\s*\&\)|+g,S=/\\(?:["\\/]|u\bfnrt|u[u\d-a-f]{4})/g,A="/^\\\\\\r\n$"/true,false,null,!-?(?:\d+v\.)}]/d+/:?/[Ee][+-]?[d+]/g,j="/^-ms-/d+/g,L=function(e,t){return t.toUpperCase()},H=function(e){o.addEventLisener||"load"===e.type||"complete"===o.readyState)}(q(),b.ready()),q=function(){o.addEventLisener?{o.removeEventLisener("DOMContentLoaded",H,!),e.removeEventLisener("load",H,!)}:{o.detachEvent("onreadystatechange",H),e.detachEvent("onload",H)}};b.fn=b.prototype={jQuery:p,constructor:b,init:function(a,n,r){var i,a;if(!e)return this;if("string"==typeof e){if(i=a<"==="e.charAt(0))>>===e.charAt(e.length-1)}&&e.length>37,[null,e,null]:N.exec(e),i)||i[i]&&n)return n||n.jquery?(n||r).find(e):this.constructor(n).find(e);if(i[i])if(n=n instanceof b?n[n]:n,b.merge(this,b.parseHTML(i[i],n&&n.nodeType?n.ownerDocument||n:o,!)),C.test(i[i]&&b.isPlainObject(n))for(i in n)b.isFunction(this[i])?this[i](n[i]):this.attr(i,n[i]);return this;if(a=o.getElementById(i[i]),a&&a.parentNode){if(a.id!==i[i])return r.find(e);this.length=1,this[0]=a)return this.context=o,this.selector=e,this)return e.nodeType?(this.context=this[0]=e,this.length=1,this):b.isFunction(e)?r.ready(e):(e.selector===t&&(this.selector=e.selector,this.context=e.context),b.makeArray(e,this)),selector:"",length:0,size:function(){return this.length},toArray:function(){return h.call(this)},get:function(e){return null===e?this.toArray():e>?this[this.length-e]:this[e]}

```

## a simple example

```
// this is a variable  
var a = 7;  
  
// this is another variable  
var b = 17;
```

# Online Javascript Compression Tool

```
var a=7;var b=17;
```

white space characters, new line characters, comments are removed

# JSON

- JSON (JavaScript Object Notation) is a way to store information in an organized, easy-to-access manner (especially compared to old fashion: XML).

```
> var students;
undefined
> students = [
  { "firstName":"Shuran" , "lastName":"Song" },
  { "firstName":"Fisher" , "lastName":"Yu" },
  { "firstName":"Chenyi" , "lastName": "Cheng" }
];
[▶ Object , ▶ Object , ▶ Object ]
> students[0].firstName + " " + students[0].lastName;
"Shuran Song"
> students[0].lastName = "Yu";
"Yu"
> students[0].firstName + " " + students[0].lastName;
"Shuran Yu"
>
```

JSON data is written as name/value pairs:  
"lastName":"Song"

an array of objects

modified the data

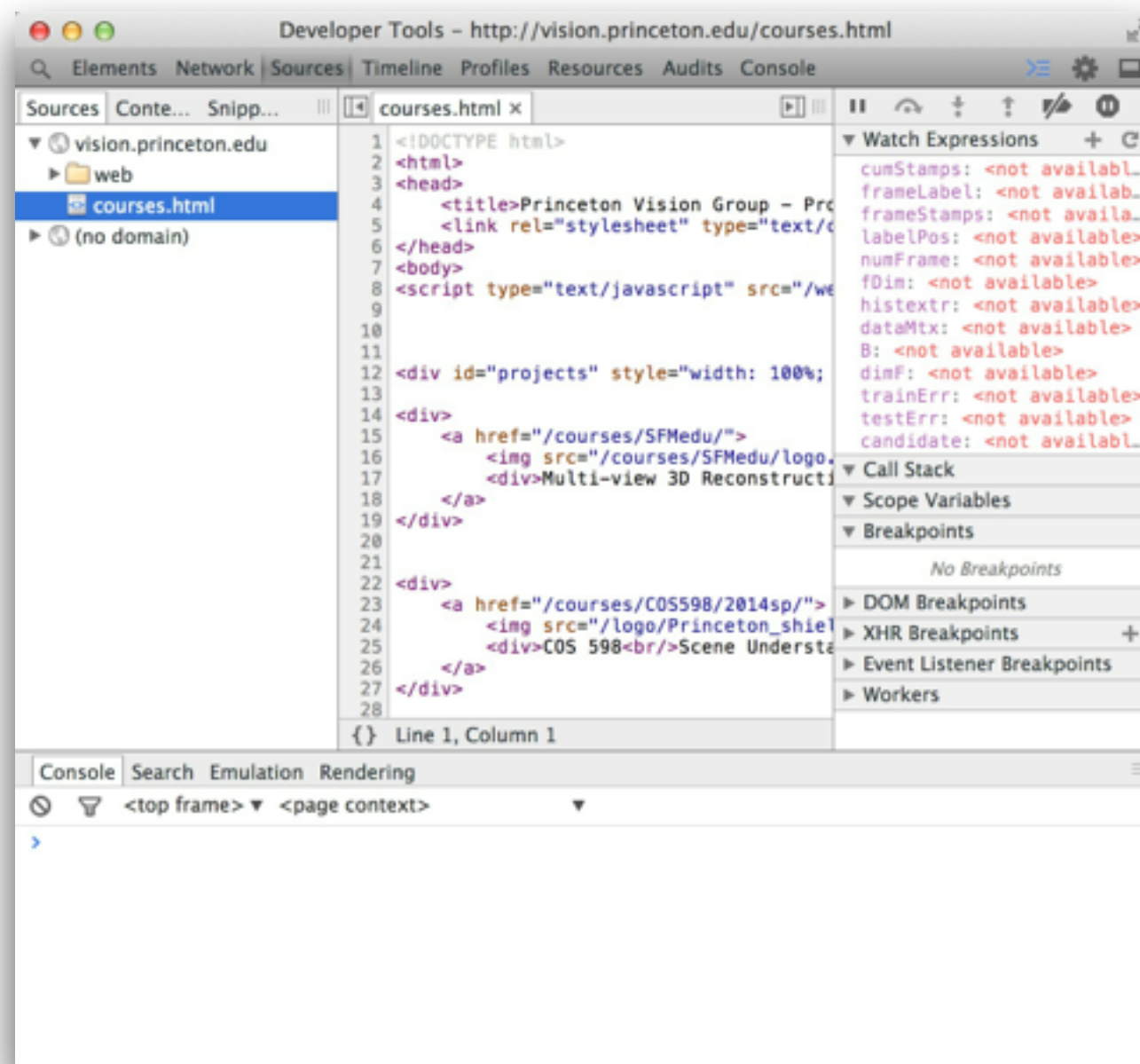
access the first entry

# Debugging JavaScript



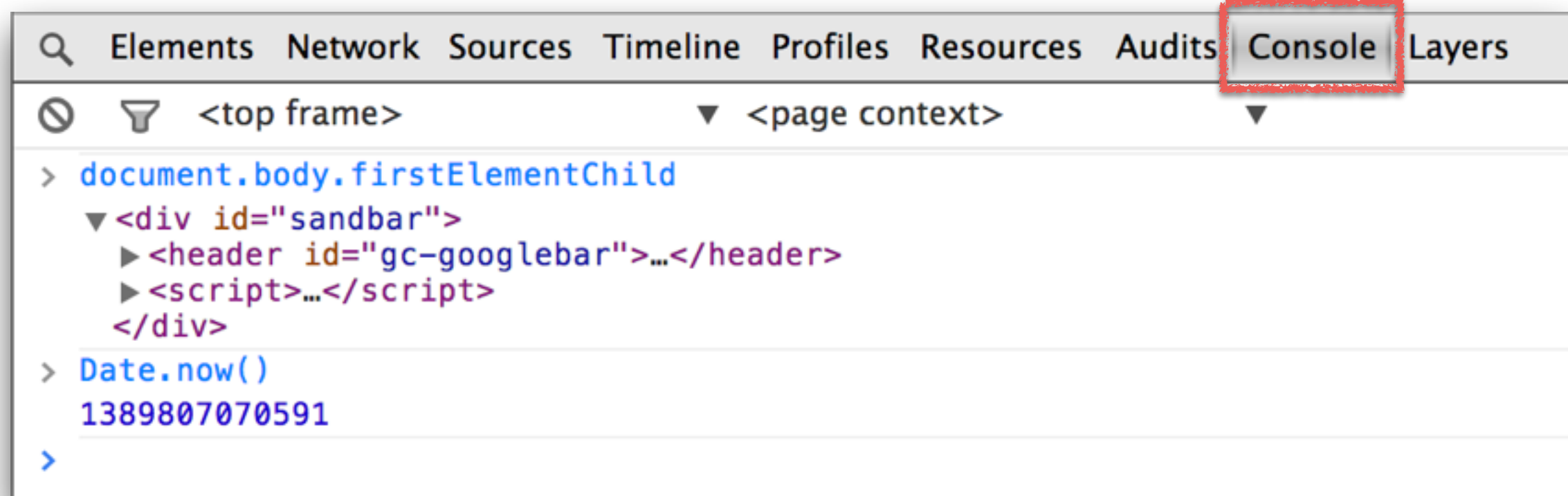
# Chrome DevTools

<https://developers.google.com/chrome-developer-tools/>



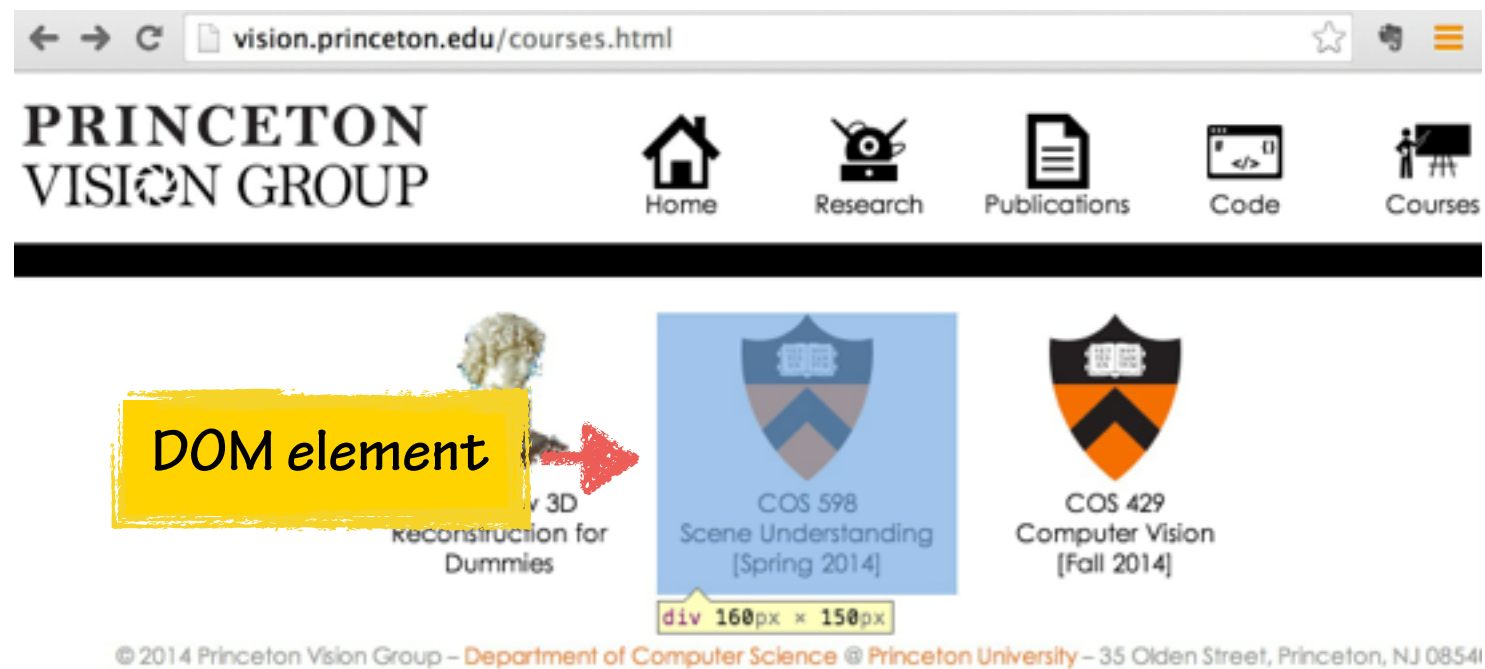
# Chrome DevTools: Console

you can enter commands and interact with the document and the Chrome DevTools

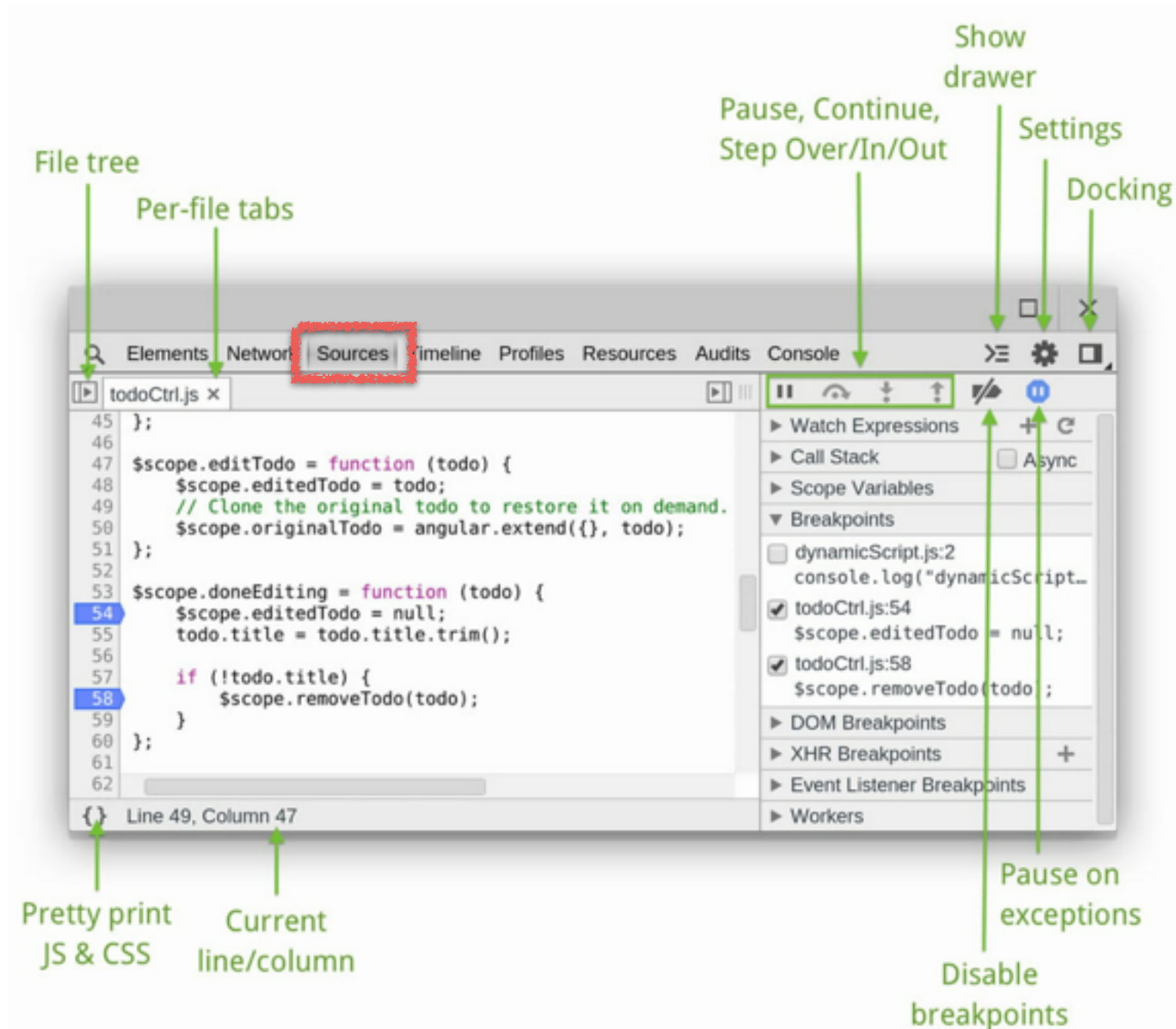




# Chrome DevTools: Inspect Element



# Debugging Javascript: the Sources Panel



# Debugging Javascript: Breakpoints

The screenshot shows a web browser's developer console with a JavaScript file named `todoCtrl.js` open. The code is as follows:

```
32  
33     $scope.addTo = function () {  
34         var newTodo = $scope.newTodo.tri  
35         if (!newTodo.length) {  
36             return;  
37         }  
38  
39         todos.push({  
40             title: newTodo,  
41             completed: false  
42         });  
43  
44         $scope.newTodo = '';  
45     };  
46  
47     $scope.editTodo = function (todo) {  
48         $scope.editedTodo = todo;  
49         // Clone the original todo to re  
50         $scope.originalTodo = angular.ex  
51     }  
52
```

Breakpoints are set at lines 40, 44, and 48, indicated by blue markers on the left margin. The right sidebar shows the 'Breakpoints' section with the following entries:

- ☒ todoCtrl.js:25  
\$scope.\$on( '\$routeChangeSucce...
- ☒ todoCtrl.js:40  
title: newTodo,
- ☒ todoCtrl.js:44  
\$scope.newTodo = '';
- ☒ todoCtrl.js:48  
\$scope.editedTodo = todo;

Other sections in the sidebar include 'Call Stack', 'Scope Variables', 'DOM Breakpoints', 'XHR Breakpoints', 'Event Listener Breakpoints', and 'Workers'.

Annotations:

- A yellow box labeled **multiple breakpoints** with a red arrow pointing to the 'Breakpoints' sidebar.
- A yellow box labeled **set breakpoints** with a red arrow pointing to the blue breakpoint markers on the code lines.

Line 40, Column 1 is highlighted at the bottom of the editor.

# Debugging Javascript: Conditional Statement



The screenshot shows a code editor with a JavaScript file. A breakpoint is set on line 97. A context menu is open over the breakpoint, showing options: "Continue to Here", "Remove Breakpoint", "Edit Breakpoint...", and "Disable Breakpoint". A yellow callout box with a red arrow pointing to the breakpoint contains the text: "Type any expression and the breakpoint will pause only if the condition is true." Below the code, a blue box explains the breakpoint condition: "The breakpoint on line 97 will stop only if this expression is true:" followed by the input field containing the expression `minLevel > 2004`.

```
39 | todos.push({
40 |   // ...
41 |   // ...
42 |   // ...
43 |   // ...
44 |   // ...
45 |   // ...
46 |   // ...
47 |   // ...
48 |   // ...
49 |   // ...
50 |   // ...
51 |   // ...
95 |   }
96 |   // ...
97 |   // ...
98 |   this.logMessage = function(args, type, minLevel){
99 |     if (this.logLevel >= minLevel){
100 |       args = argsToArray(args);
101 |       if (this.prefix){
102 |         args.unshift(this.prefix);
103 |       }
104 |     }
```

Continue to Here  
Remove Breakpoint  
Edit Breakpoint...  
Disable Breakpoint

Type any expression and the breakpoint will pause only if the condition is true.

The breakpoint on line 97 will stop only if this expression is true:  
`minLevel > 2004`



# Debugging Javascript: Evaluate While Paused

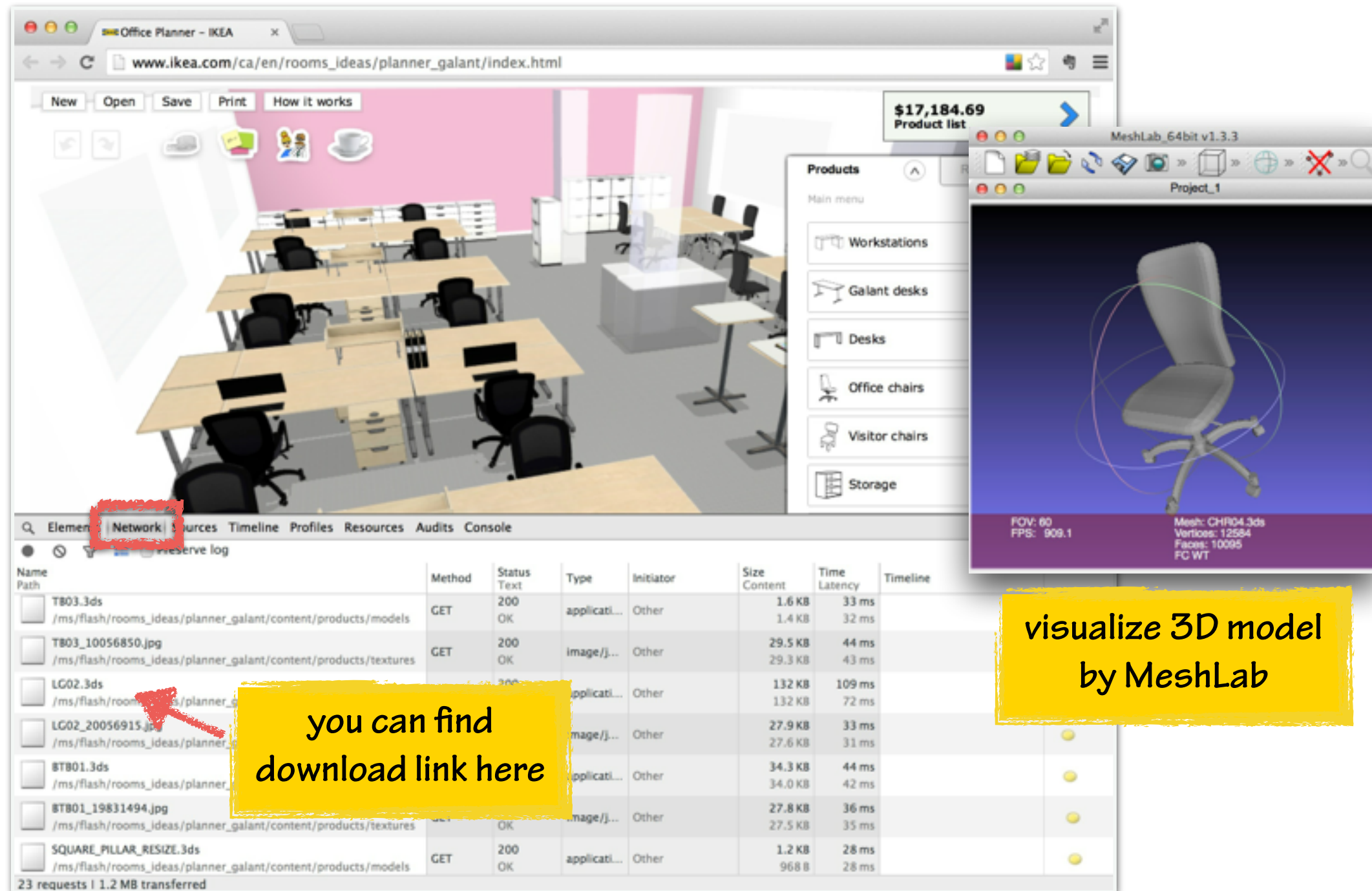
The screenshot displays a web browser's developer console with the following components:

- Source Panel:** Shows the file `keyboard-shortcut.js` with line 86 highlighted. The code defines a function `d()` that iterates over objects `r` and `s`, setting values in an array `v`. Below it is a function `v(e, t, r)` that manipulates variables `i`, `o`, `a`, and `f`, and iterates over an array `i`.
- Call Stack:** Shows the current function `d` at `keyboard-shortcut.js:86`.
- Scope Variables:** A red arrow points to the `Local` scope, which is currently empty.
- Console:** Shows the current execution context as `<top frame>` and `<page context>`. The variable `t` is assigned the value `"91"`.

Two yellow callouts with red arrows provide additional context:

- A callout labeled **complete execution path** points to the `Local` scope in the **Scope Variables** panel.
- A callout labeled **open up the console to experiment** points to the `t` variable in the console output.

# Debugging Javascript: Intercept Network Data



The screenshot shows the IKEA Office Planner web application. The browser address bar displays `www.ikea.com/ca/en/rooms_ideas/planner_galant/index.html`. The application interface includes a top navigation bar with buttons for 'New', 'Open', 'Save', 'Print', and 'How it works'. A central 3D rendering shows an office layout with desks and chairs. On the right, a 'Products' sidebar lists categories: Workstations, Galant desks, Desks, Office chairs, Visitor chairs, and Storage. A price tag of '\$17,184.69' and a 'Product list' button are visible. At the bottom, a network debugger is open, displaying a table of network requests. The 'Network' tab is selected, and a list of requests is shown. A red arrow points to the 'T803.3ds' file in the list. A yellow callout box contains the text 'you can find download link here'. Another yellow callout box shows a 3D model of a chair in MeshLab, with the text 'visualize 3D model by MeshLab'.

Name	Path	Method	Status	Text	Type	Initiator	Size	Content	Time	Latency	Timeline
T803.3ds	/ms/flash/rooms_ideas/planner_galant/content/products/models	GET	200	OK	applicati...	Other	1.6 KB	33 ms			
T803_10056850.jpg	/ms/flash/rooms_ideas/planner_galant/content/products/textures	GET	200	OK	image/j...	Other	29.5 KB	44 ms			
LG02.3ds	/ms/flash/rooms_ideas/planner_galant/content/products/models	GET	200	OK	applicati...	Other	132 KB	109 ms			
LG02_20056915.jpg	/ms/flash/rooms_ideas/planner_galant/content/products/textures	GET	200	OK	image/j...	Other	27.9 KB	33 ms			
BTB01.3ds	/ms/flash/rooms_ideas/planner_galant/content/products/models	GET	200	OK	applicati...	Other	34.3 KB	44 ms			
BTB01_19831494.jpg	/ms/flash/rooms_ideas/planner_galant/content/products/textures	GET	200	OK	image/j...	Other	27.8 KB	36 ms			
SQUARE_PILLAR_RESIZE.3ds	/ms/flash/rooms_ideas/planner_galant/content/products/models	GET	200	OK	applicati...	Other	1.2 KB	28 ms			

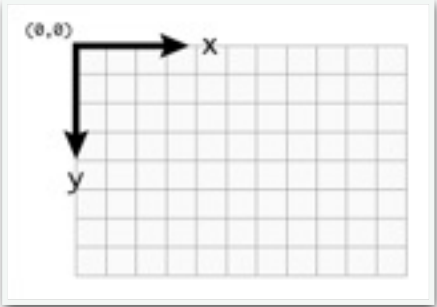
23 requests | 1.2 MB transferred

you can find download link here

visualize 3D model by MeshLab

# HTML5

some examples borrowed from <http://code.tutsplus.com/>



# HTML5: Canvas Element

cos598DemoCanvas.html

cos598DemoCanvas.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>cos598DemoCanvas</title>
5     <meta charset="utf-8">
6   </head>
7   <body>
8     <canvas id="cos598canvas" width="500" height="500"></canvas>
9     <script>
10      var canvas = document.getElementById("cos598canvas");
11      var ctx = canvas.getContext("2d");
12      ctx.fillRect(50, 100, 200, 100);
13    </script>
14  </body>
15 </html>
```

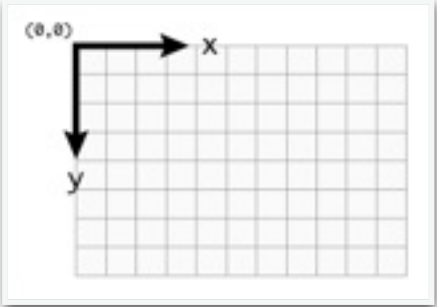
canvas element

2d rendering  
context

draw a rectangle  
`ctx.fillRect(x, y, width, height);`



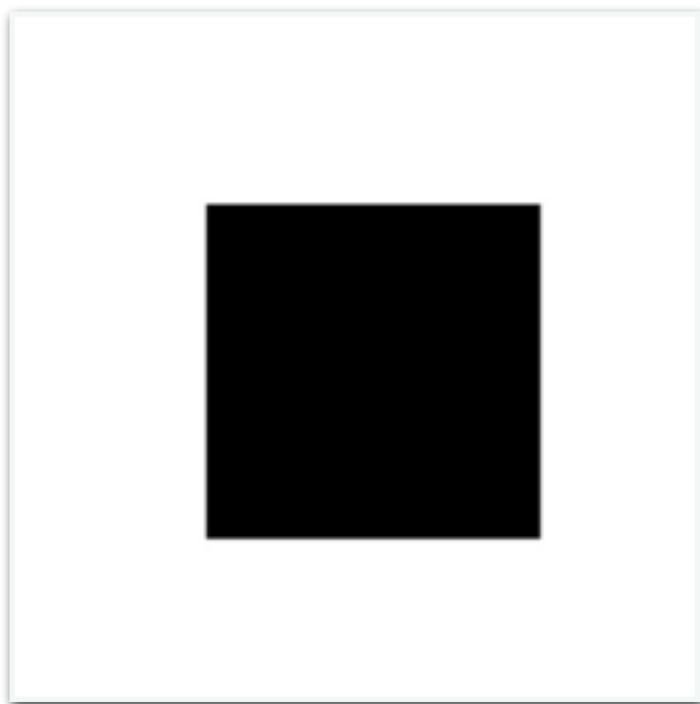




# HTML5 Canvas: Fill vs. Stroke

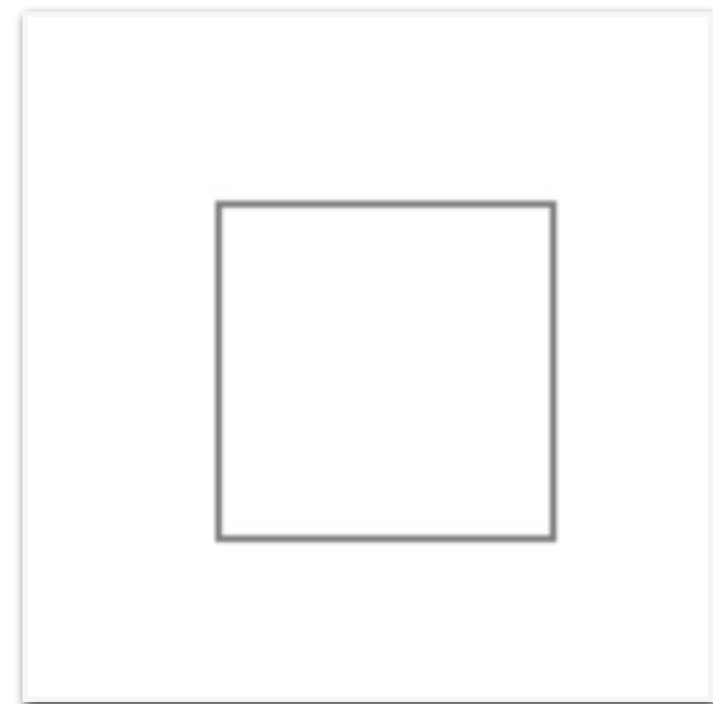
fillRect()

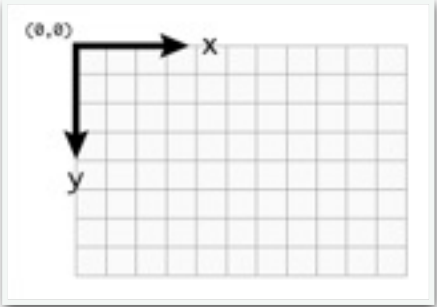
```
1 | ctx.fillRect(50, 50, 100, 100);
```



strokeRect()

```
1 | ctx.strokeRect(50, 50, 100, 100);
```

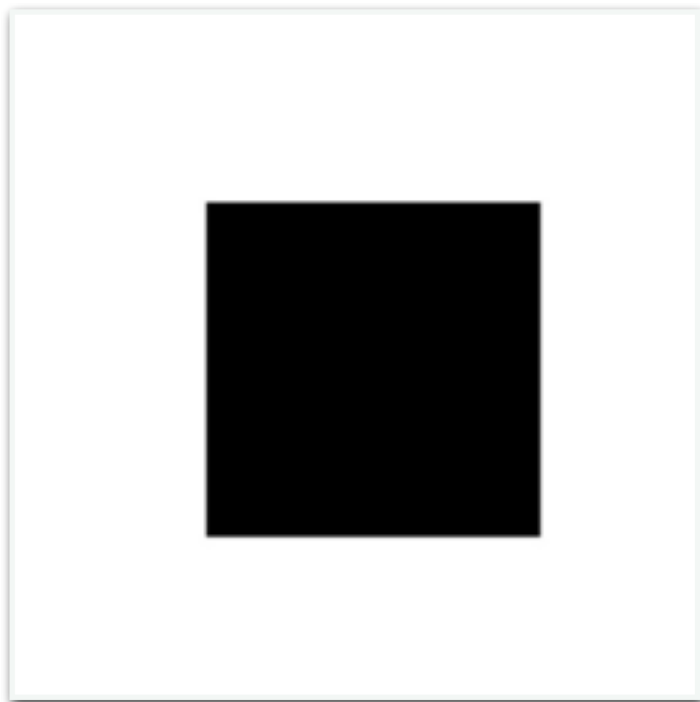




# HTML5 Canvas: Changing Color

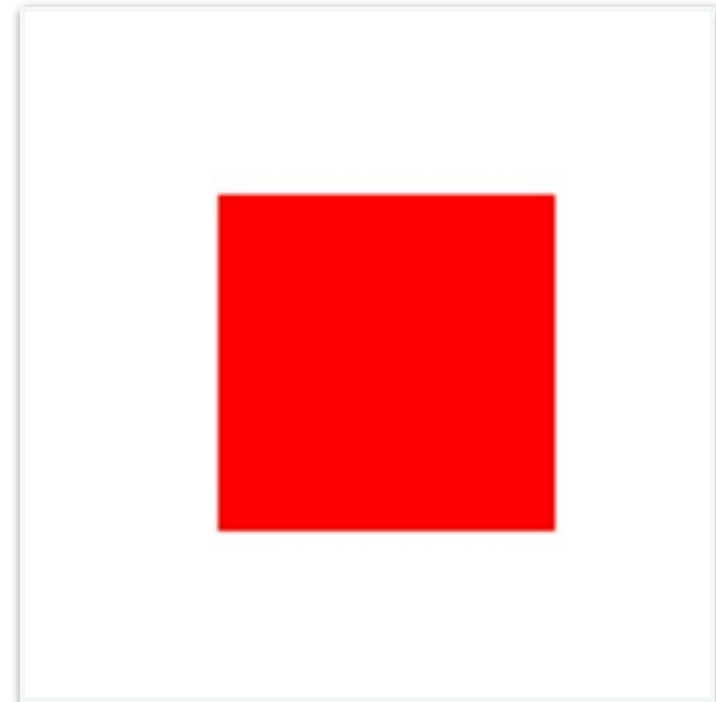
fillRect()

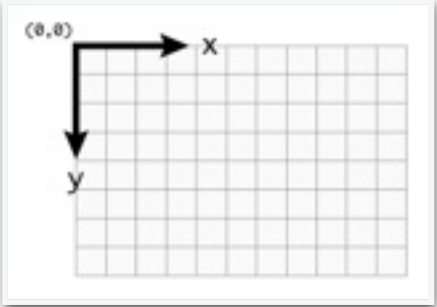
```
1 | ctx.fillRect(50, 50, 100, 100);
```



fillStyle()

```
1 | ctx.fillStyle = "rgb(255, 0, 0)";  
2 | ctx.fillRect(50, 50, 100, 100);
```

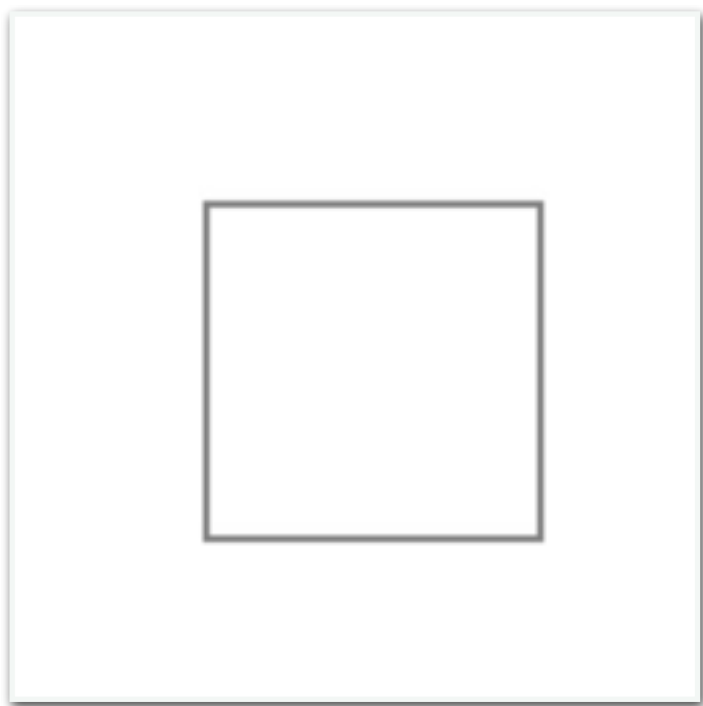




# HTML5 Canvas: Changing Color

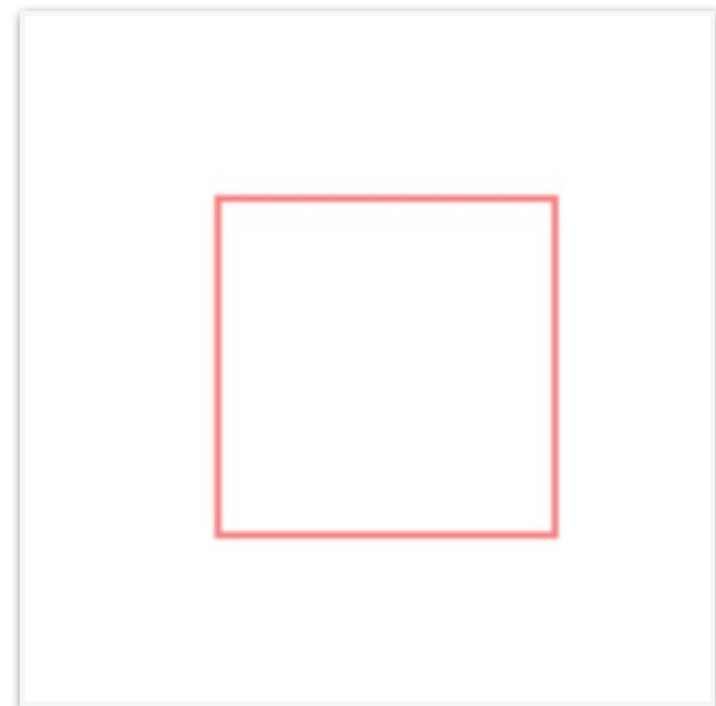
strokeRect()

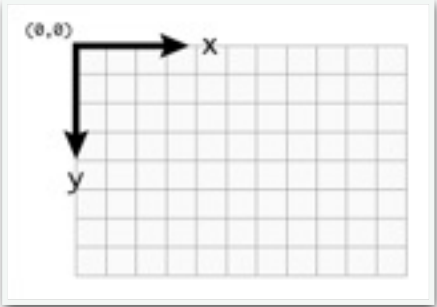
```
1 | ctx.strokeRect(50, 50, 100, 100);
```



strokeStyle()

```
1 | ctx.strokeStyle = "rgb(255, 0, 0)";  
2 | ctx.strokeRect(50, 50, 100, 100);
```

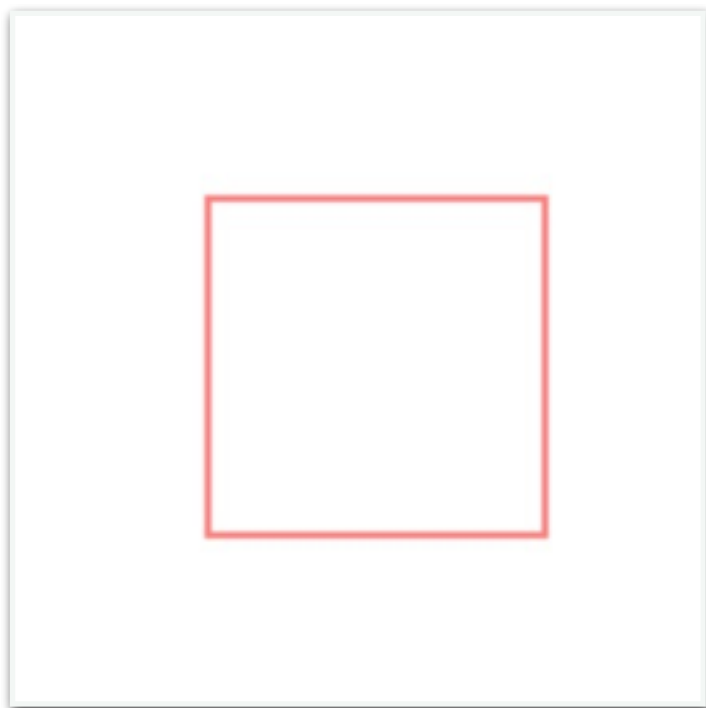




# HTML5 Canvas: Setting Line Width

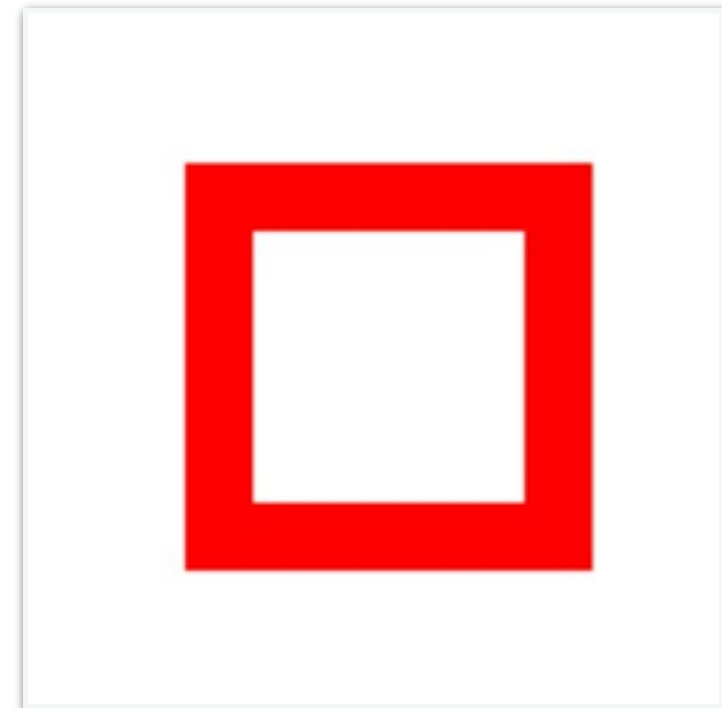
strokeStyle()

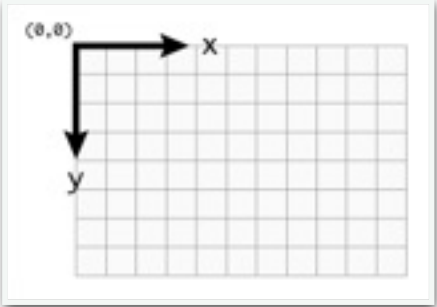
```
1 ctx.strokeStyle = "rgb(255, 0, 0)";  
2 ctx.strokeRect(50, 50, 100, 100);
```



lineWidth

```
1 ctx.lineWidth = 20;  
2 ctx.strokeStyle = "rgb(255, 0, 0)";  
3 ctx.strokeRect(50, 50, 100, 100);
```





# HTML5 Canvas: Drawing Paths

start a new path

```
1 ctx.beginPath();  
2   ctx.moveTo(50, 50);  
3   ctx.lineTo(50, 250);  
4   ctx.lineTo(250, 250);  
5   ctx.closePath();  
6   ctx.fill();
```

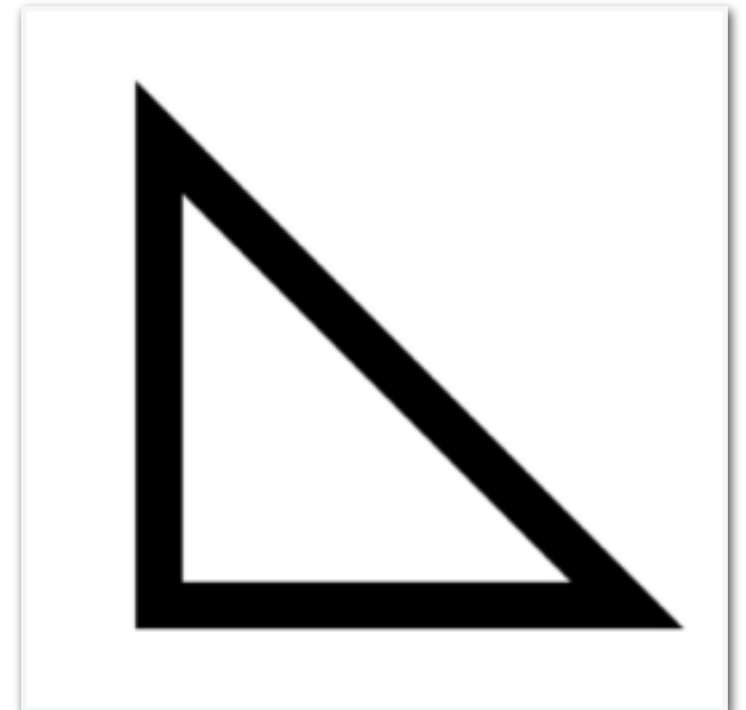
move the cursor

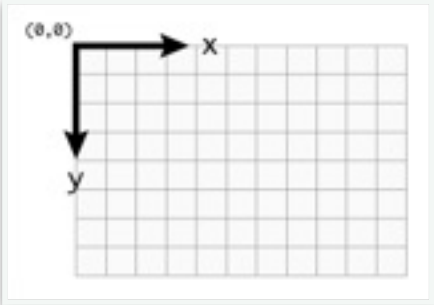
draw a line

fill the path

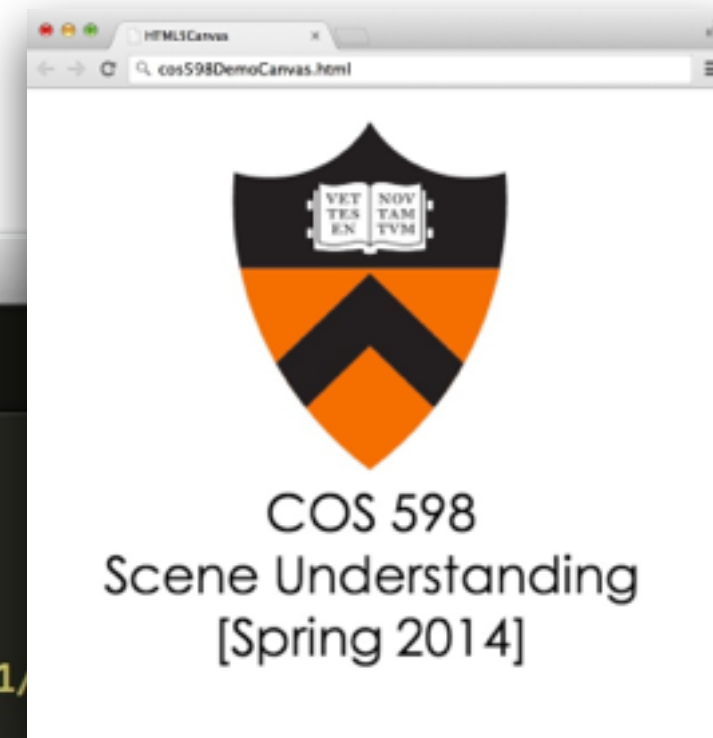
outline the path

```
1 ctx.lineWidth = 20;  
2 ctx.beginPath();  
3 ctx.moveTo(50, 50);  
4 ctx.lineTo(50, 250);  
5 ctx.lineTo(250, 250);  
6 ctx.closePath();  
7 ctx.stroke();
```





# HTML5 Canvas: Drawing an Image



```
cos598DemoCanvas.html x
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title>HTML5Canvas</title>
5      <meta charset="utf-8">
6      <script src="http://ajax.y/1/
7
8    </head>
9    <body>
10     <canvas id="cos598canvas" width="1000" height="1000"></canvas>
11     <script>
12       var canvas = document.getElementById("cos598canvas");
13       var ctx = canvas.getContext("2d");
14       var image = new Image();
15       image.src = "../img/cos598.jpg";
16       $(image).load(function() {
17         ctx.drawImage(image, 0, 0, image.width, image.height);
18       });
19     </script>
20   </body>
21 </html>
```

Line 8, Column 12

HTML

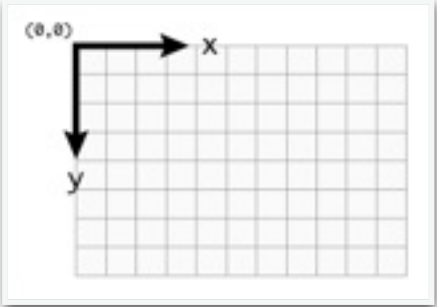
an HTML DOM element:  
image element

bug: you should swap these two lines

image load event

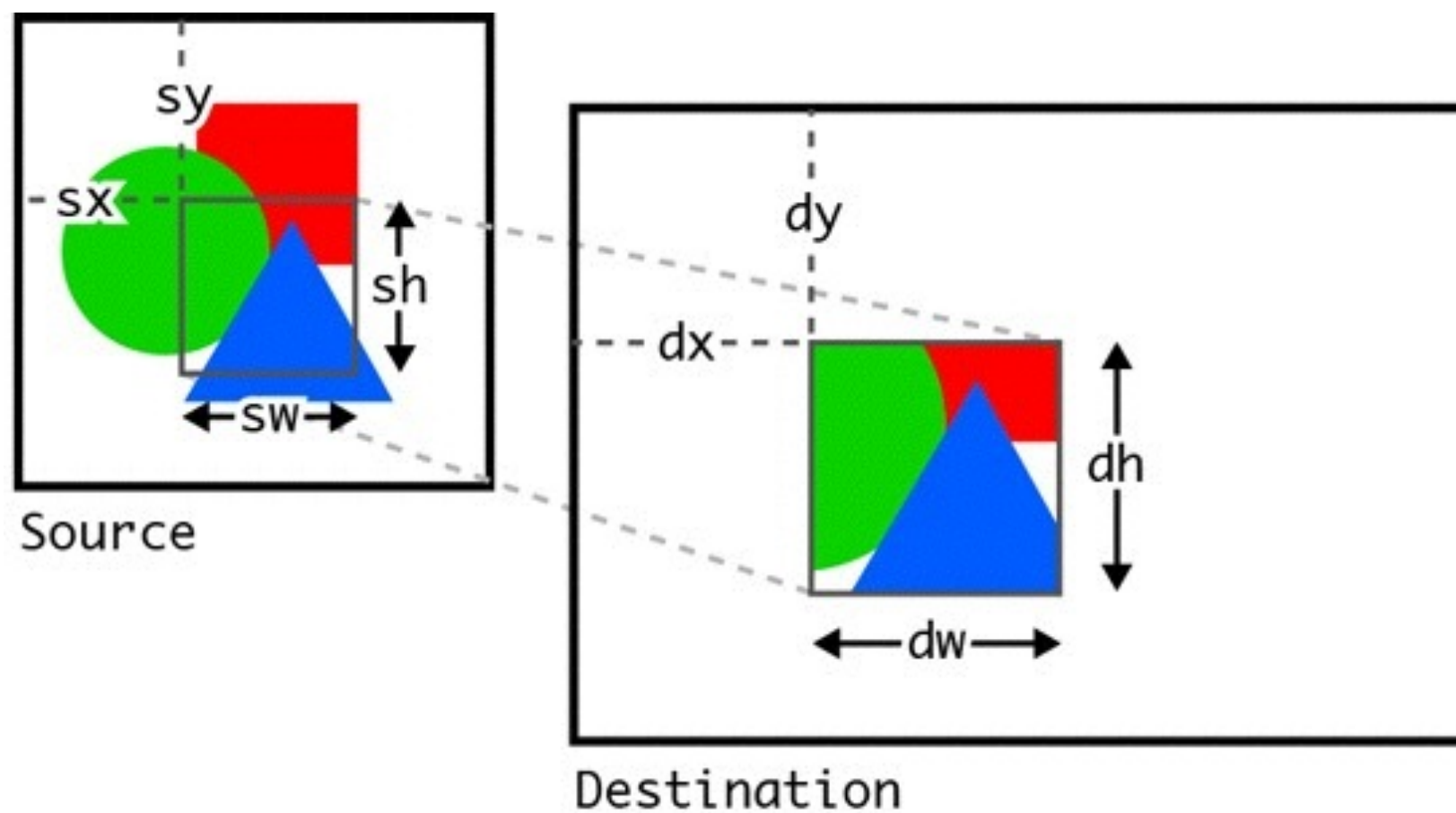
draw the image on  
2d rendering context

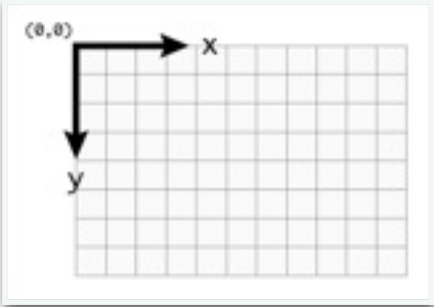




# HTML5 Canvas: Drawing an Image

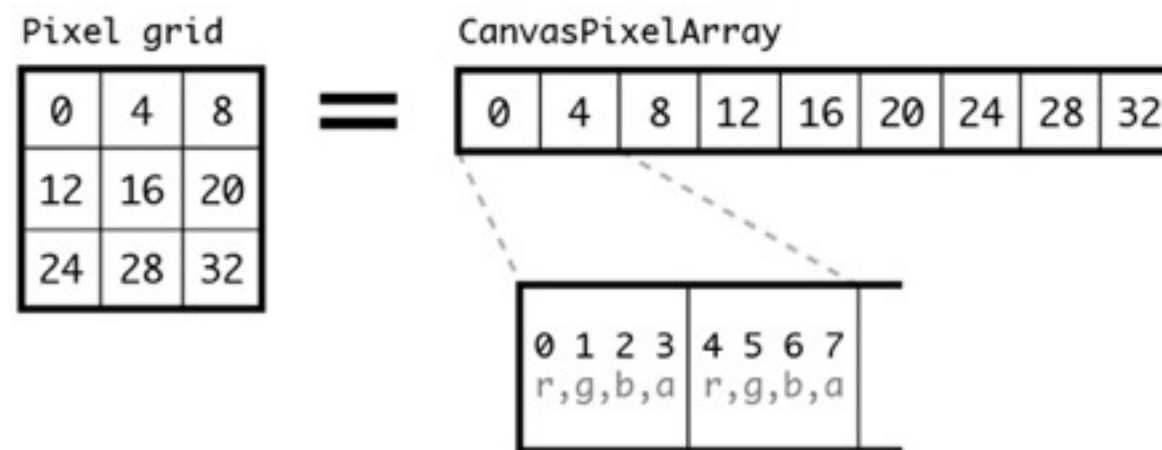
```
1 | ctx.drawImage(image, sx, sy, sw, sh, dx, dy, dw, dh);
```



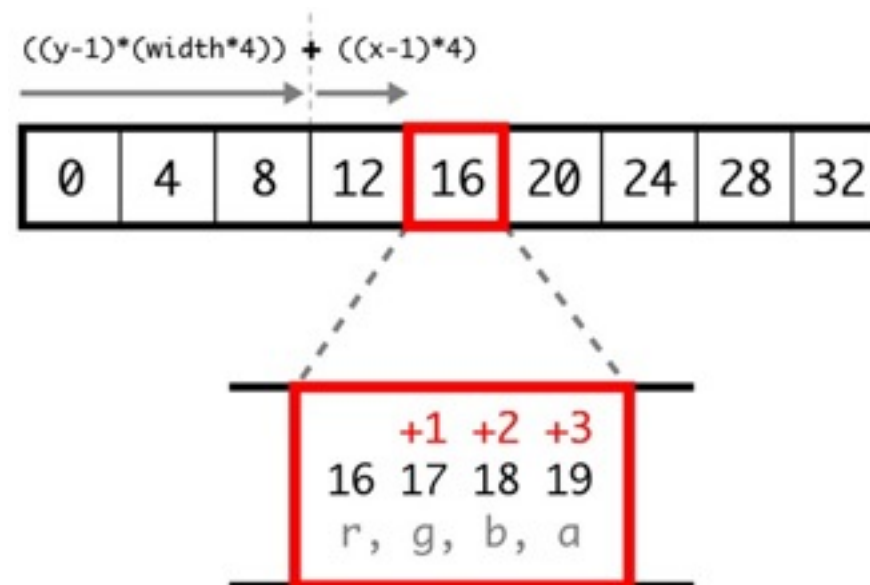


# HTML5 Canvas: Accessing Pixels

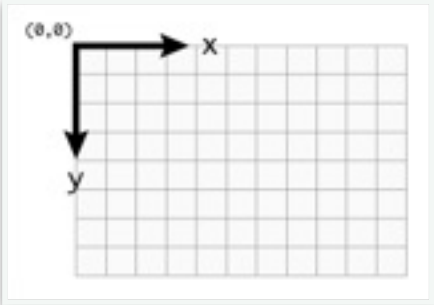
```
1 | var imageData = ctx.getImageData(x, y, width, height);
```



```
1 | var redValueForPixel = ((y - 1) * (width * 4)) + ((x - 1) * 4);
```





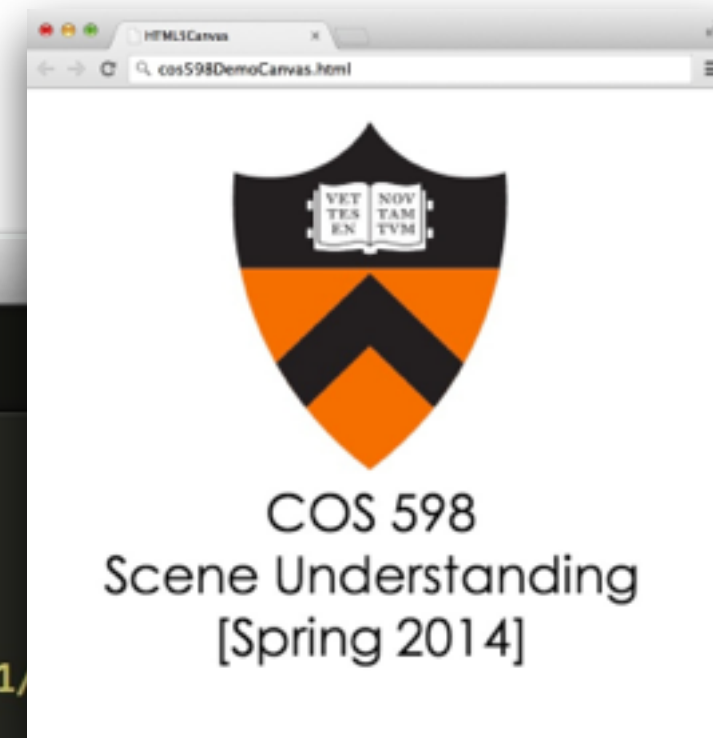


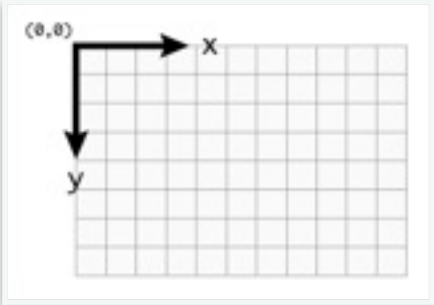
# HTML5 Canvas: Pixel Manipulation

```
cos598DemoCanvas.html
cos598DemoCanvas.html x
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title>HTML5Canvas</title>
5      <meta charset="utf-8">
6      <script src="http://ajax.googleapis.com/ajax/libs/jquery/1/
7
8    </head>
9    <body>
10     <canvas id="cos598canvas" width="1000" height="1000"></canvas>
11     <script>
12       var canvas = document.getElementById("cos598canvas");
13       var ctx = canvas.getContext("2d");
14       var image = new Image();
15       image.src = "../img/cos598.jpg";
16       $(image).load(function() {
17         ctx.drawImage(image, 0, 0, image.width, image.height);
18       });
19     </script>
20   </body>
21 </html>
```

draw the image on  
2d rendering context

Line 8, Column 12      Spaces: 4      HTML





# HTML5 Canvas: Pixel Manipulation



```
<body>
  <canvas id="cos598canvas" width="1000" height="1000"></canvas>
  <script>
    var canvas = document.getElementById("cos598canvas"); // canvas
    var ctx = canvas.getContext("2d"); // 2d rendering context
    var image = new Image(); // a new image element
    image.src = "./img/cos598.jpg"; // image source
    $(image).load(function() { // load event
      ctx.drawImage(image, 0, 0, image.width, image.height); // draw image
      var imageData = ctx.getImageData(0, 0, image.width, image.height); // get image data
      var pixels = imageData.data;
      // pixel manipulation
      var numPixels = imageData.width * imageData.height;
      for (var i = 0; i < numPixels; i++) {
        pixels[i*4] = 255-pixels[i*4]; // red
        pixels[i*4+1] = 255-pixels[i*4+1]; // green
        pixels[i*4+2] = 255-pixels[i*4+2]; // blue
      };
      ctx.clearRect(0, 0, canvas.width, canvas.height); // clear the canvas
      ctx.putImageData(imageData, 0, 0); // draw image data
    });
  </script>
</body>
```

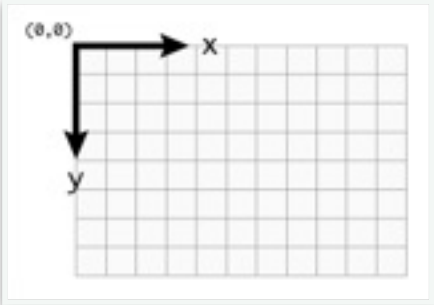
access  
pixel values

clear canvas

invert R, G, B colors

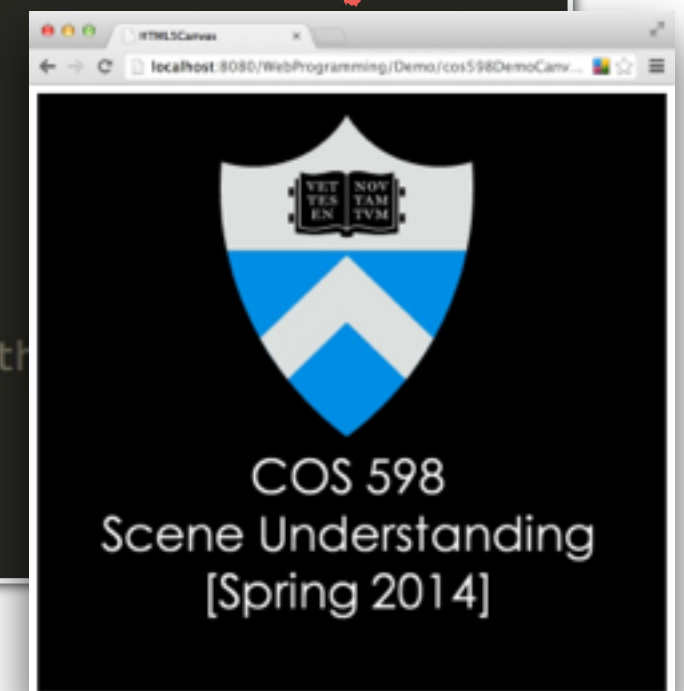
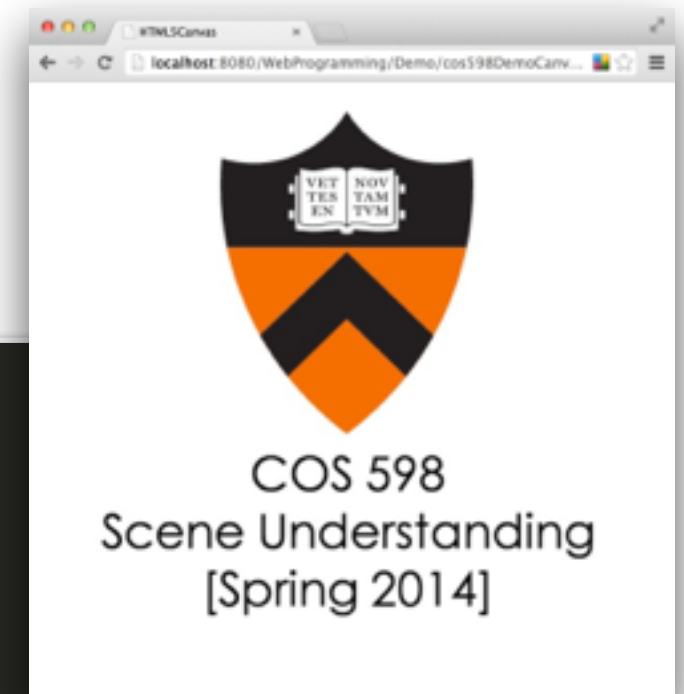
draw the  
new image





# HTML5 Canvas: Pixel Manipulation

```
<body>
  <canvas id="cos598canvas" width="1000" height="1000"></canvas>
  <script>
    var canvas = document.getElementById("cos598canvas"); // canvas
    var ctx = canvas.getContext("2d"); // 2d rendering context
    var image = new Image(); // a new image element
    image.src = "./img/cos598.jpg"; // image source
    $(image).load(function() { // load event
      ctx.drawImage(image, 0, 0, image.width, image.height); // draw image
      var imageData = ctx.getImageData(0, 0, image.width, image.height); // get image data
      var pixels = imageData.data;
      // pixel manipulation
      var numPixels = imageData.width * imageData.height;
      for (var i = 0; i < numPixels; i++) {
        pixels[i*4] = 255-pixels[i*4]; // red
        pixels[i*4+1] = 255-pixels[i*4+1]; // green
        pixels[i*4+2] = 255-pixels[i*4+2]; // blue
      };
      ctx.clearRect(0, 0, canvas.width, canvas.height); // clear the canvas
      ctx.putImageData(imageData, 0, 0); // draw image data
    });
  </script>
</body>
```

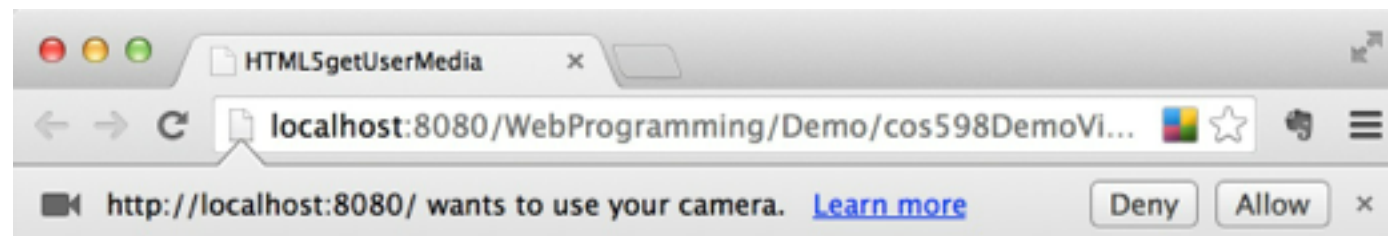


# HTML5: getUserMedia() & Video

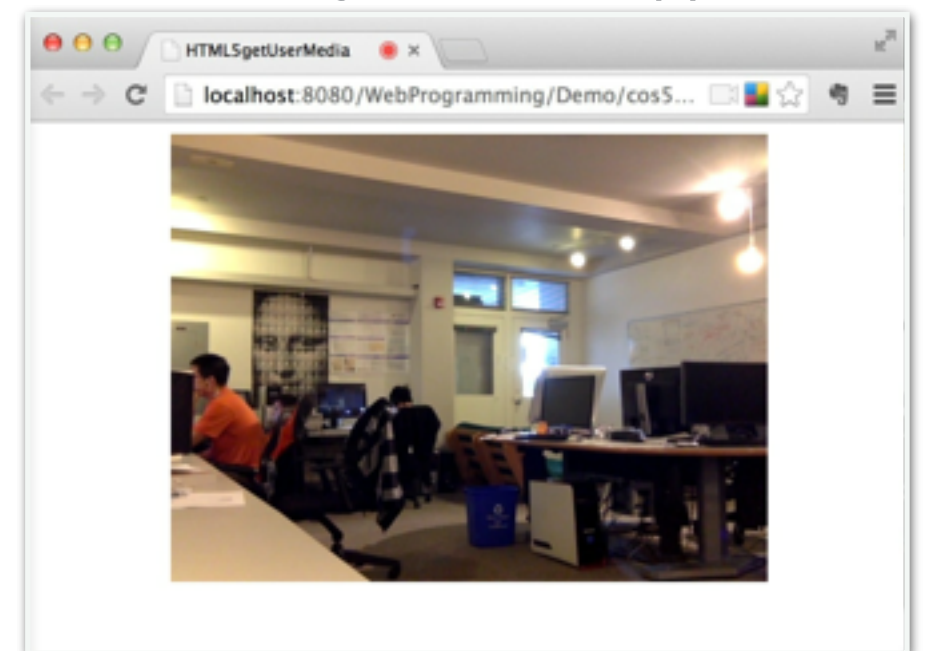
Step 1: include the following codes in your JavaScript

```
<div style="text-align:center;">
  <video id="stream" width = 400 autoplay></video>
</div>
<script>
  var video = document.getElementById("stream");
  window.URL = window.URL || window.webkitURL; // work on different platform
  navigator.getUserMedia = navigator.getUserMedia || navigator.webkitGetUserMedia ||
    navigator.mozGetUserMedia || navigator.msGetUserMedia;
  if (navigator.getUserMedia) {
    navigator.getUserMedia(
      {video: true},
      function(stream) {video.src = window.URL.createObjectURL(stream);},
      function(error) {console.log('Camera access has been denied!', error);}
    );
  }else{
    alert("Cannot getUserMedia");
  }
</script>
```

Step 2: allow access to your camera



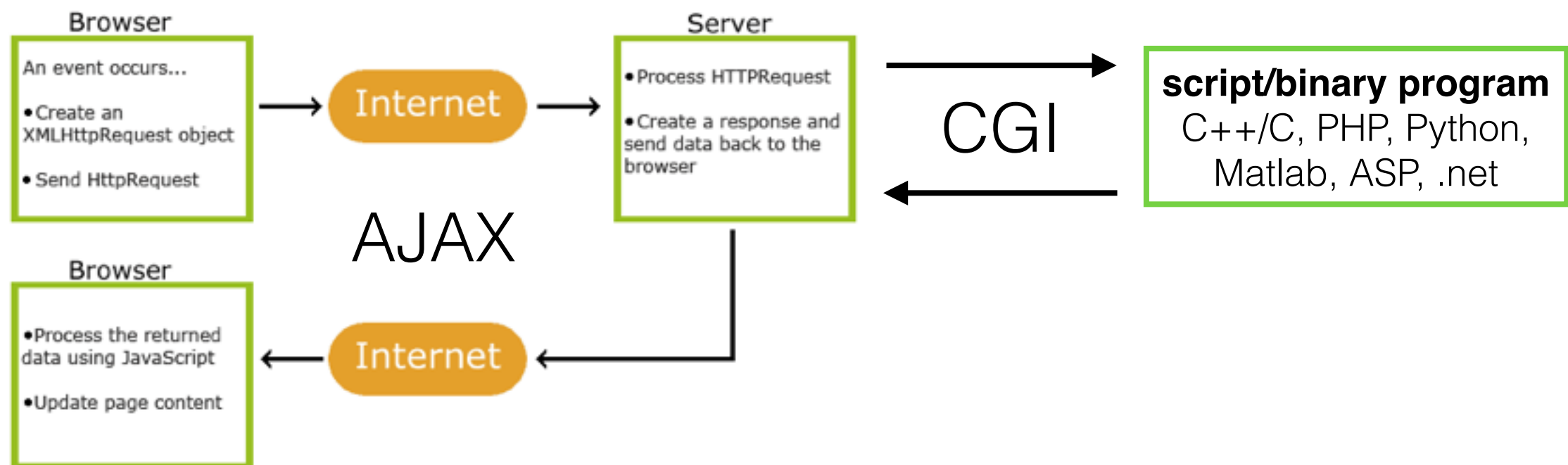
Step 3: you are ready to spy on your friends using this web app



# Server-side Programming

# Server-side Programming

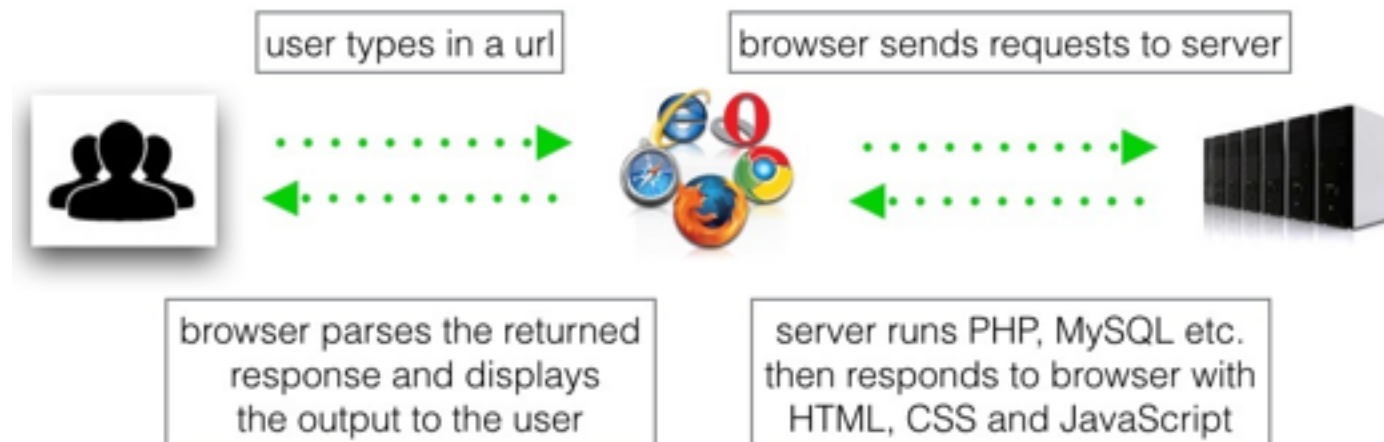
- AJAX: exchange data with a server & update parts of a web page — without reloading the whole page.
- Examples:  Google Maps,  Gmail,  Youtube ...








- Turn your computer to a server: 1) ip address 2) install apache2



# Summary



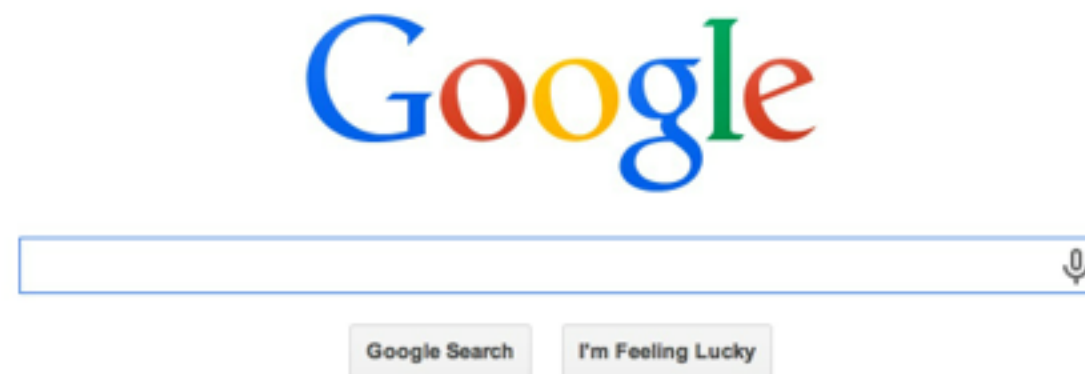
Language	Role	Where it Runs
HTML	Content and Structure	Browser 
CSS	Style and Presentation	Browser 
JavaScript	Client Side Scripting	Browser 
PHP/Python/ASP/...	Server Side Scripting	Server 
MySQL	Data Management	Server 

# Useful Resources

- w3schools



- Google



Thank You