PREP

WEEK 14

CS 105

CS @ Illinois

WEEK 14

ANNOUNCEMENTS

15 04.24.2017	Monday • Due: MP 7/Lab 11 (3:00 pm) • Due: Activity 12 (3:00 pm) • Out: MP 8	Javascript IV • JavaScript cheatsheet	Labs: Mon-Wed. • Out: LAB 12	Friday • TBD:
16 05.01.2017	Monday • Out: MP 9	Javascript V	Labs: Mon-Wed. • Out: LAB 13	Friday • All MPS, LABS due 8:00 PM (SUNDAY, MAY 7th) • All extra credit due 8:00 PM (SUNDAY, MAY 7th)
Finals Week 05.08.2017	Monday		**Final Wednesday: 05.10.2017** • Foellinger Auditorium • 1:30 - 4:30 pm	

FINAL
SCHEDULE

Course	Section	CRN	Date	Day	Start Time	End Time	Room
CS 105	AL2	31128	05/10/2017	W	1:30 PM	4:30 PM	AUD Foellinger Auditorium

CONFLICT:
TUESDAY 8:00 AM
PLACE: TBD
[NO MORE THAN 2 FINALS]
SEND EMAIL WITH OTHER CLASSES

ASCII "codes"

0	NUL	16	DLE	32	SP	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(	56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41	)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[	107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	|
13	CR	29	GS	45	-	61	=	77	M	93	]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL

ありがとうございました

"♠♡♢♣"

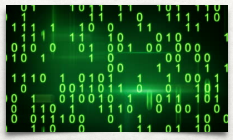
0	NUL	16	DLE	32	SP	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(	56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41	)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[	107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	|
13	CR	29	GS	45	-	61	=	77	M	93	]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^				
15	SI	31	US	47	/	63	?	79	O	95	_				

謝謝

ENCODING

encoder:
ASCII, UNICODE,
MPEG, MP3, proprietary
(rules or standards)

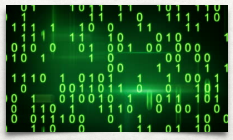
23,43,56,67,90



DECODING

decoder:
ASCII, UNICODE,
MPEG, MP3, proprietary
(rules or standards)

23,43,56,67,90



DECIMAL

base 10 (10 digits 0-9)

0-9	0-9	0-9	0-9	0-9
2	8	7	1	5
10^4	10^3	10^2	10^1	10^0

```
20000 (2 * 10000) +
 8000 (8 *  1000) +
  700 (7 *   100) +
   10 (1 *    10) +
    5 (5 *     1)
------------------
28715
```

BINARY

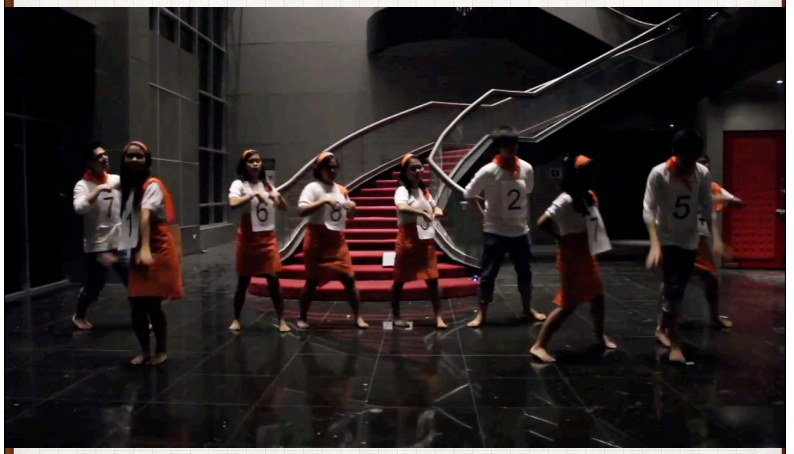
base 2 (2 digits 0-1)

0-1	0-1	0-1	0-1	0-1
1	1	0	1	0
2^4	2^3	2^2	2^1	2^0

```
16 (1 * 16) +
 8 (1 *  8) +
 0 (0 *  4) +
 2 (1 *  2) +
 0 (0 *  1)
------------
26
```

WHO AM I

piazza please



SORTING VISUALLY

SORTING SOLVED

SORTING
OPERATIONS

- compare two items
  - ability to provide a custom compare function
- swap two items

```
var numbers = [10,23,1,2,100];
numbers.sort();
// YIKES: 1,10,100,2,23
```

DEMO TIME

```
function numberCompare(n1, n2) {
  if (n1 > n2) {
    return 1;
  }
  if (n2 > n1) {
    return -1;
  }
  return 0; // equal
};

var numbers = [10,23,1,2,100];
numbers.sort( numberCompare );
```

SWAPPING ITEMS

```
var items = [42, 23, 10];
// how to swap items[0] with items[1]?
// after: [23, 42, 10];
// write the JavaScript code

var items = [42, 23, 10];

  var tmp  = items[0]; // tmp == 42
  items[0] = items[1]; //[23, 23, 10]
  items[1] = tmp;      //[42, 23, 10]
```

JAVASCRIPT OBJECTS

```
var suits = "♠♡♢♣";

var spade   = suits[0];
var heart   = suits[1];
var diamond = suits[2];
var club    = suits[3];

// how to represent a deck
// of 52 playing cards ?
```

building a deck of cards

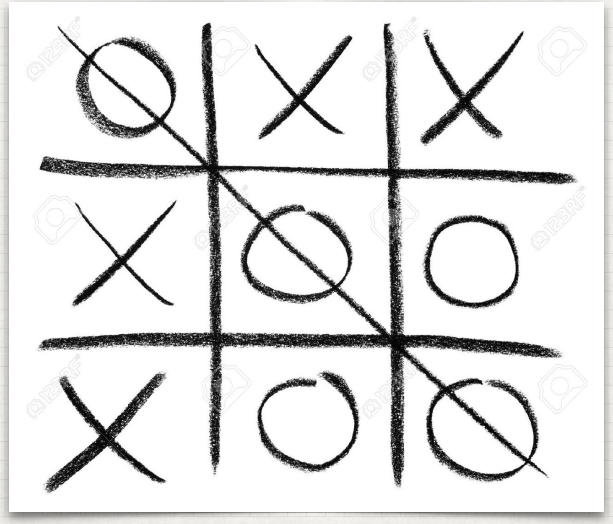
```
suits = "♠♡◇♣";
rank: 1 - 13
var aceOfSpades = {
  suit: spade,
  rank: 1
}
var queenOfHearts = {
  suit: heart,
  rank: 12
}
```

```
var suits = "♠♡◇♣";
var deck = [];
for (var i = 0; i < suits.length; i++) {
  for (var j = 1; j <= 13; j++) {
    var card = {
       suit:  suits[i],
       rank: j
     };
     deck.push(card); // NEW
  }
}
// how many cards will be created
// how to access the first card's rank in deck
```

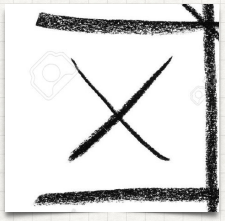
Dealing a hand (5 cards)

```
var hand = [];
deck.shuffle(); // you would build this
for (var i = 0; i < 5; i++) {
  hand[i] = deck[i];
}
```

```
var suits = "♠♡♢♣"

function isFlush(hand)
{
  // all the cards have the same suit


}
```



```
var slot = {
  glyph:     undefined, // 'X', 'O'
  moveNumber: -1,       // when was it occupied
  // wow a function
  isEmpty: function() {
     return typeof(this.glyph) === 'undefined';
  }
}
```



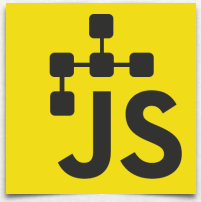
```
var board = [];
for (var row = 0; row < 3; row++) {
  var row = [];
  for (var col = 0; col < 3; col++) {
     row.push({ moveNumber:0 });
  }
  board.push(row);
}

// how would you describe board ?
```

```
var board = [
  [ {}, {}, {} ], // row 0
  [ {}, {}, {} ], // row 1
  [ {}, {}, {} ]  // row 2
];

// Array of Array of Objects

// Where on the board is this
board[1][1].glyph = 'X';
```

WEB DEVELOPER

WEB TECH

DONEC QUIS NUNC



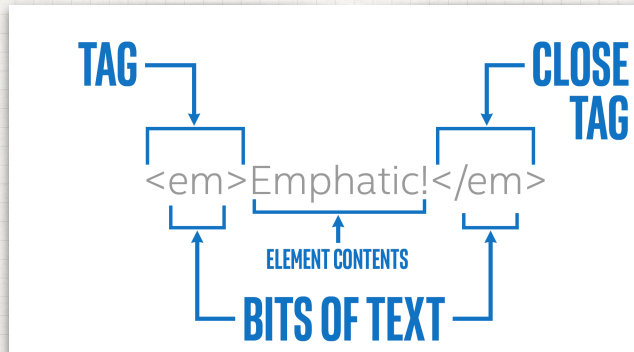
HTML

```
<!DOCTYPE html>

<html>

  <head>
  </head>

  <body>
  </body>

</html>
```

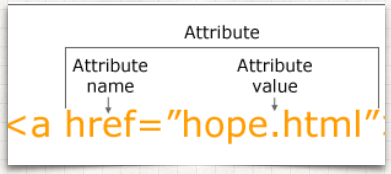
HTML

STRUCTURE AND CONTENT

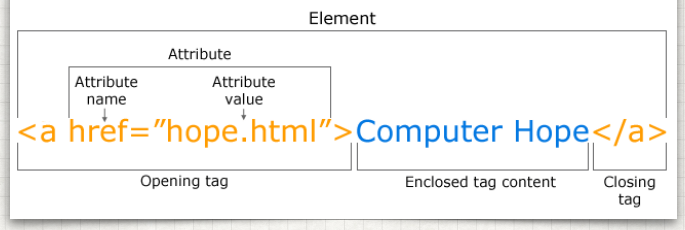
- Describes structure of web page
  - semantics (heading, links, sidebar, navigation)
  - content
- Elements
  - start tag + content + end tag
- Tag: reserved word + attributes




```
<p>
This is the content of the paragraph element.
</p>
```



Breakdown of an HTML Tag



CSS

CSS

PRESENTATION (DESIGN, LAYOUT)

- cascading style sheets
- enable presentation (style)
  - design and layout
  - text style, color, size, position, appearance, animation
- rules (css selectors) on how to apply style

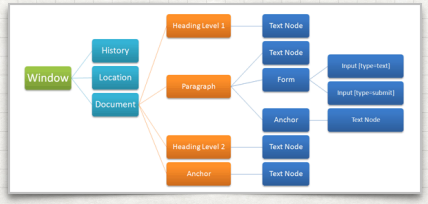
JAVASCRIPT

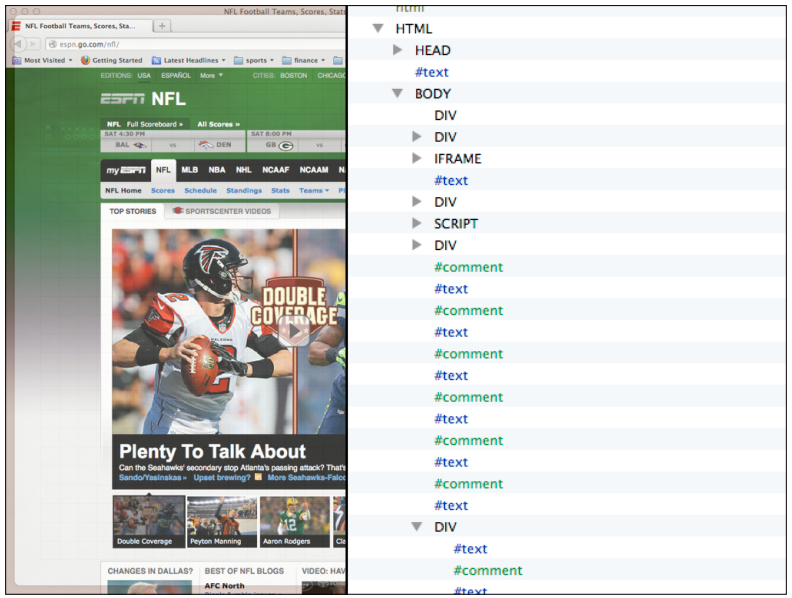
JAVASCRIPT

DYNAMIC INTERACTION

- event based interaction
  - <button onClick="processClick();">press me </button>
- window events (onload, onresize, on focus)
- mouse events (onclick, onmousedown, onmouseover, etc)
- keyboard events (onkeypress)
- input control (onchange, onsubmit)
- media events (onload, onplay, onpause)

DOM





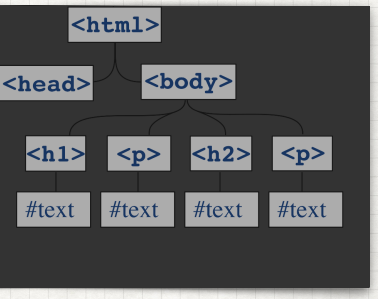
DOCUMENT OBJECT MODEL

API

- DOM: Document Object Model
- a data structure (i.e. **tree**) that represents the HTML elements that can be accessed and manipulated via an API using javascript
- change node location (update, delete, insert)
- change node attributes (class, width, x, y)

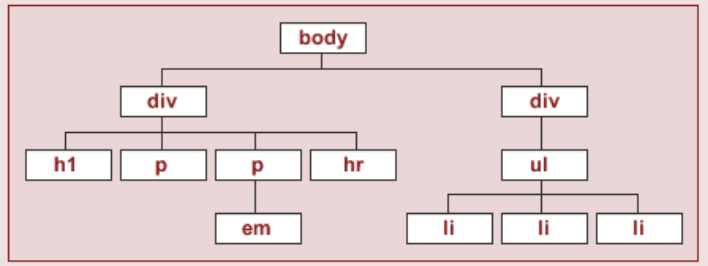
EXAMPLE.COM

```
<html>
  <head></head>
  <body>
    <h1>Heading 1</h1>
    <p>Paragraph.</p>
    <h2>Heading 2</h2>
    <p>Paragraph.</p>
  </body>
</html>
```



```
<body>
 <div id="content">
  <h1>Heading here</h1>
  <p>Lorem ipsum dolor sit amet.</p>
  <p>Lorem ipsum
  <em>sit</em> amet.</p>
  <hr>
 </div>
 <div id="nav">
  <ul>
    <li>item 1</li>
    <li>item 2</li>
    <li>item 3</li>
  </ul>
 </div>
</body>
```

DRAW ME



DEMO IT

DOM - JavaScript Bridge

DOM API

```
document.getElementById('id');
 Retrieves the element with the given id as an object
 css: #myId { }

document.getElementsByClassName("classname");
 Retrieves all elements with the class name classname
 and stores them in an array- like list
 css: p.myclass {  }

document.getElementsByTagName('tagname'):
 Retrieves all elements with the tag name tagname and
 stores them in an array- like list
 css: p { }
```

DOM navigation