The JavaScript Language

An introduction

What can you do with JavaScript

- A lot:
    - Process (limited) user input, get data from external source (databases and more), do any calculations, generate HTML (from previous slides)
- Most importantly, for our class:
    - Access and modify a web page (both content and HTML markup)
    - Do that as the user is viewing the page

1. ACCESS CONTENT

Select text, elements, or attributes on an HTML page

1. ACCESS CONTENT

Select text, elements, or attributes on an HTML page

2. MODIFY CONTENT

Add text, elements, or attributes to an HTML page (or remove them)

1. ACCESS CONTENT

Select text, elements, or attributes on an HTML page

2. MODIFY CONTENT

Add text, elements, or attributes to an HTML page (or remove them)

3. PROGRAM RULES

Specify a set of steps for the browser to follow (like a recipe)

1. ACCESS CONTENT

Select text, elements, or attributes on an HTML page

2. MODIFY CONTENT

Add text, elements, or attributes to an HTML page (or remove them)

3. PROGRAM RULES

Specify a set of steps for the browser to follow (like a recipe)

4. REACT TO EVENTS

Tell a script to run in response to some event that occurred

What you will need to learn

1. Programming concepts:
   - How to organize your ideas about sequences of tasks)

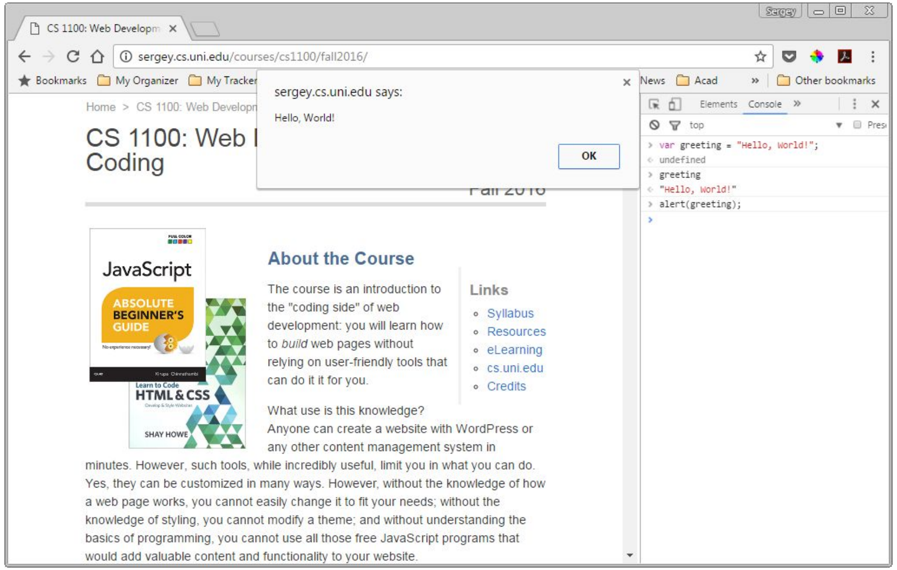
2. The JavaScript language:
   - How to express these ideas using the language vocabulary, syntax, and grammar

* You will also benefit greatly from looking at how this is done:

- Look at examples of code
- Study tutorials
- Use stackoverflow.com

Where to write your JavaScript code

- In your favorite text editor - just like you did with HTML and CSS
- You may use the browser console for trying out code



sergey.cs.uni.edu says:

Hello, World!

OK

```
> var greeting = "Hello, World!";
< undefined
> greeting
< "Hello, World!"
> alert(greeting);
>
```

Home > CS 1100: Web Developm

CS 1100: Web Coding

Fall 2016

About the Course

The course is an introduction to the "coding side" of web development: you will learn how to *build* web pages without relying on user-friendly tools that can do it it for you.

What use is this knowledge?
Anyone can create a website with WordPress or any other content management system in minutes. However, such tools, while incredibly useful, limit you in what you can do. Yes, they can be customized in many ways. However, without the knowledge of how a web page works, you cannot easily change it to fit your needs; without the knowledge of styling, you cannot modify a theme; and without understanding the basics of programming, you cannot use all those free JavaScript programs that would add valuable content and functionality to your website.

Links
- Syllabus
- Resources
- eLearning
- cs.uni.edu
- Credits

Where to place your JavaScript code

a. Inside your html:
    - inside <script> </script> tags
b. In a separate file (preferred approach):
    - use any name with a ".js" extension
    - do NOT include the <script> tags in your .js file
    - link to your HTML using this tag: <script src="path-to-your-.js-file">
c. Both (only if you really need to)

* Make sure to place your <script> tag close to the bottom of your HTML file

JavaScript statements

- Each “step” in a program is a statement
- A statement is code that performs a task
- Statements are terminated with a semicolon:
  - var message;
    message = “Hello, World!”;
  - var x; x = 42;
  - **recommended style: one line = one statement (or partial statement)**
- There are different types of statements; today we take a closer look at assignment statements

Variables

1. Declaring a variable

2. Naming a variable

3. Assigning a value to a variable (assignment statement)

1. Declaring a variable

- We must declare a variable before using it in our program

- We should use the var keyword*

***If we don't use var, the variable will be given global scope, which is not what we intend. We'll discuss this later in the semester*

```
var quantity;
```

$$\underbrace{\texttt{var}}_{\text{KEYWORD}}\ \texttt{quantity;}$$

$$\texttt{var}\ \underbrace{\texttt{quantity}}_{\textbf{VARIABLE NAME}}\texttt{;}$$

2. Naming a variable

- JavaScript is case-sensitive: message <> Message <> MESSAGE
- Naming variables:
  - start with a character, “$”, or underscore
  - do not use spaces
  - do not use keywords (var, for, if, else, function, etc…)
  - use camelCasing when name consists of more than one word
  - use descriptive names,
    - that are easy to interpret (bad name: n / good name: items OR numberOfItems)
    - that are short (bad name: buttonHasBeenClicked / good name: isClicked)

3. Assigning a value to a variable

```
quantity = 3;
```

$$\underbrace{\texttt{quantity}}_{\text{VARIABLE NAME}}\ \texttt{= 3;}$$

```
quantity = 3;
```

ASSIGNMENT OPERATOR

```
quantity = 3;
```

VALUE

Assignment statement

We copy the value of the expression on the right into the variable on the left:

```
[variable] = [expression]
```

Assignment operator IS NOT the same as equality operator:

- **a = b** *does not mean* **a equals b**
- **a = b** means `take the value of b and copy it into a`

Assignment statement

- Declaring and assigning variables:
  - We can do this in 2 statements:

    ```
    var message;
    message = “hello!”;
    ```

  - or we can do this in one statement:

    ```
    var message = “hello!”;
    ```

Assignment statement practice

statement	value of **a**	value of **b**
var a = 1;	1	undefined

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;		

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;		

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;		

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	2

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	2
a = b;		

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	2
a = b;	2	

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	2
a = b;	2	2

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	2
a = b;	2	2
b = b + 3		

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	2
a = b;	2	2
b = b + 3	2	

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	2
a = b;	2	2
b = b + 3	2	5

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	2
a = b;	2	2
b = b + 3	2	5
a = a - b		

Reassigning a variable

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	2
a = b;	2	2
b = b + 3	2	5
a = a - b	-3	

Assignment statement practice

statement	value of **a**	value of **b**
var a = 1;	1	undefined
a = a + 1;	2	undefined
var b = a;	2	2
a = a + 1;	3	2
a = b;	2	2
b = b + 3	2	5
a = a - b	-3	5

Browser as context for JavaScript

As you recall, JavaScript can be used to access and modify your HTML

BUT. Before we can do anything meaningful in the browser, we need to understand: ***exactly what*** we are accessing and modifying.

Then, we can understand ***how*** we access and modify that.

For this, let's look at data types.

Data types

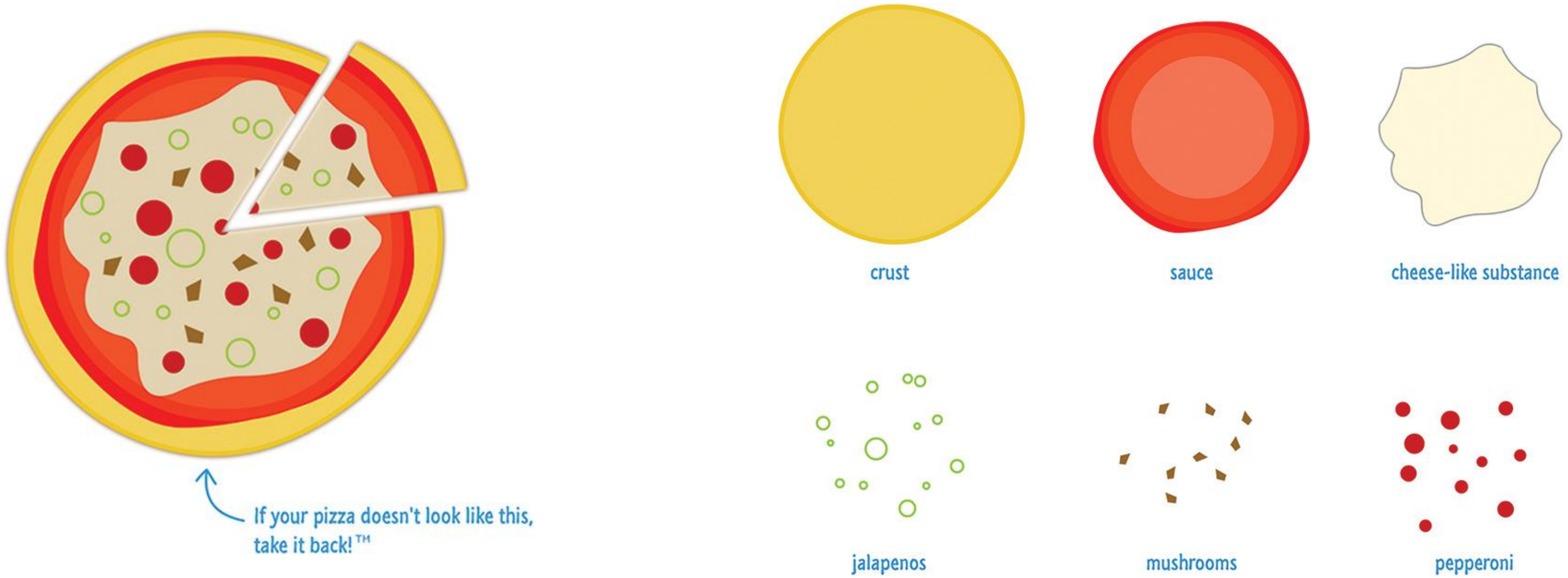
What ***types*** of values can we have?

1. `42` (this is a number)
2. `"Don't panic!"` (this is a string)
3. `true` (this is a boolean)
4. ...and more...

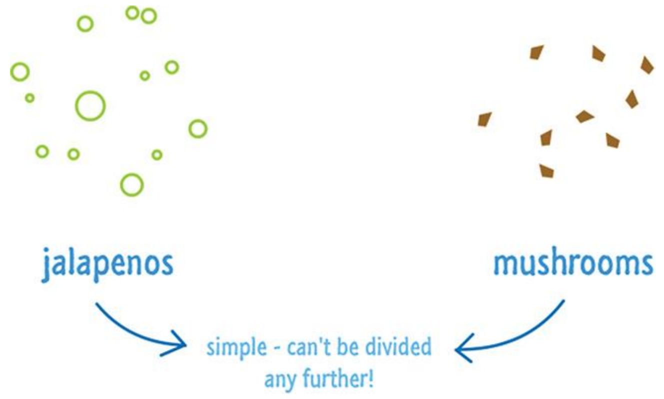
We refer to types of values as ***data types***

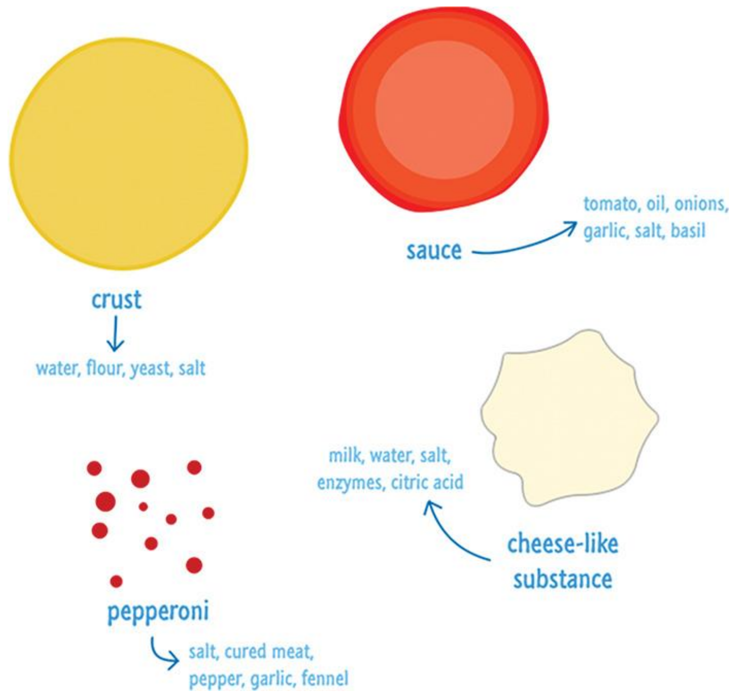
As for "...and more..." - let's look at your textbook's pizza analogy!

The pizza analogy

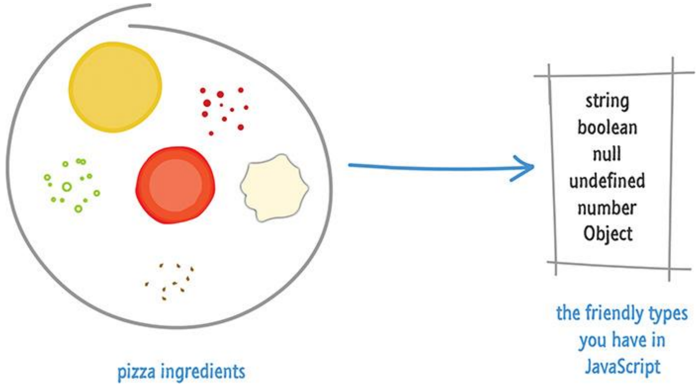


Simple and complex ingredients




crust
water, flour, yeast, salt
sauce
tomato, oil, onions, garlic, salt, basil
pepperoni
salt, cured meat, pepper, garlic, fennel
cheese-like substance
milk, water, salt, enzymes, citric acid

Simple = primitives / Complex = objects

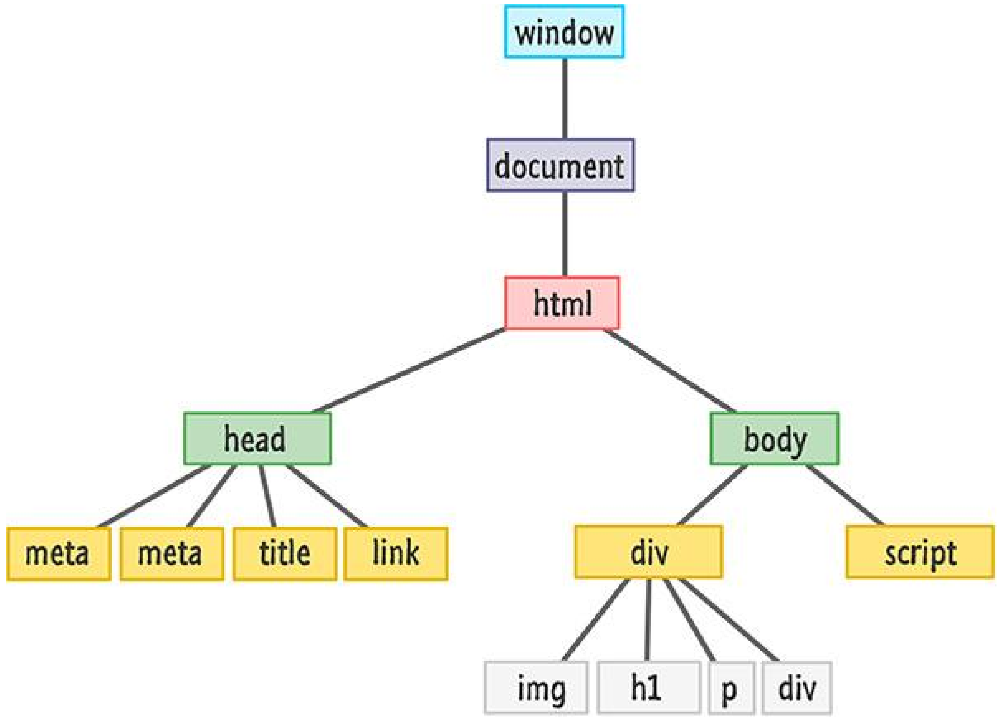


Type	What It Does
`String`	The basic structure for working with text
`Number`	As you can guess, it allows you to work with numbers
`Boolean`	Comes alive when you are using true and false
`Null`	Represents the digital equivalent of nothing…or **moo** :P
`Undefined`	While sorta similar to null, this is returned when a value should exist but doesn’t…like when you declare a variable but don’t assign anything to it
`Object`	Acts as a shell for other types including other objects

Objects

- An object is an abstraction:
  - ball
- An object can have properties and methods
- We can access and modify an object's **properties**:
  - ball.color *accesses the ball's color*
  - ball.color = "red" *changes the ball color to red*
- We can call an object's **methods**
  - ball.roll() *the ball rolls*
- We can construct our own objects (but we don't have to right now)
- We can use objects that have been constructed for us
- One such object gives us access to the content of our HTML document!

The Document Object Model: a preview



Practice (if time permits)

- Use any simple web page (cs.uni.edu us one option)
- Use the console in your browser to write your code
- use document.querySelector(“**your-selector**”) to access elements in your HTML
- assign these elements to variables:
  - `var heading = document.querySelector(“h1”)`
- use **your-variable**.innerHTML to access an element's HTML
- use **your-variable**.innerHTML = “new text” to modify that element's HTML