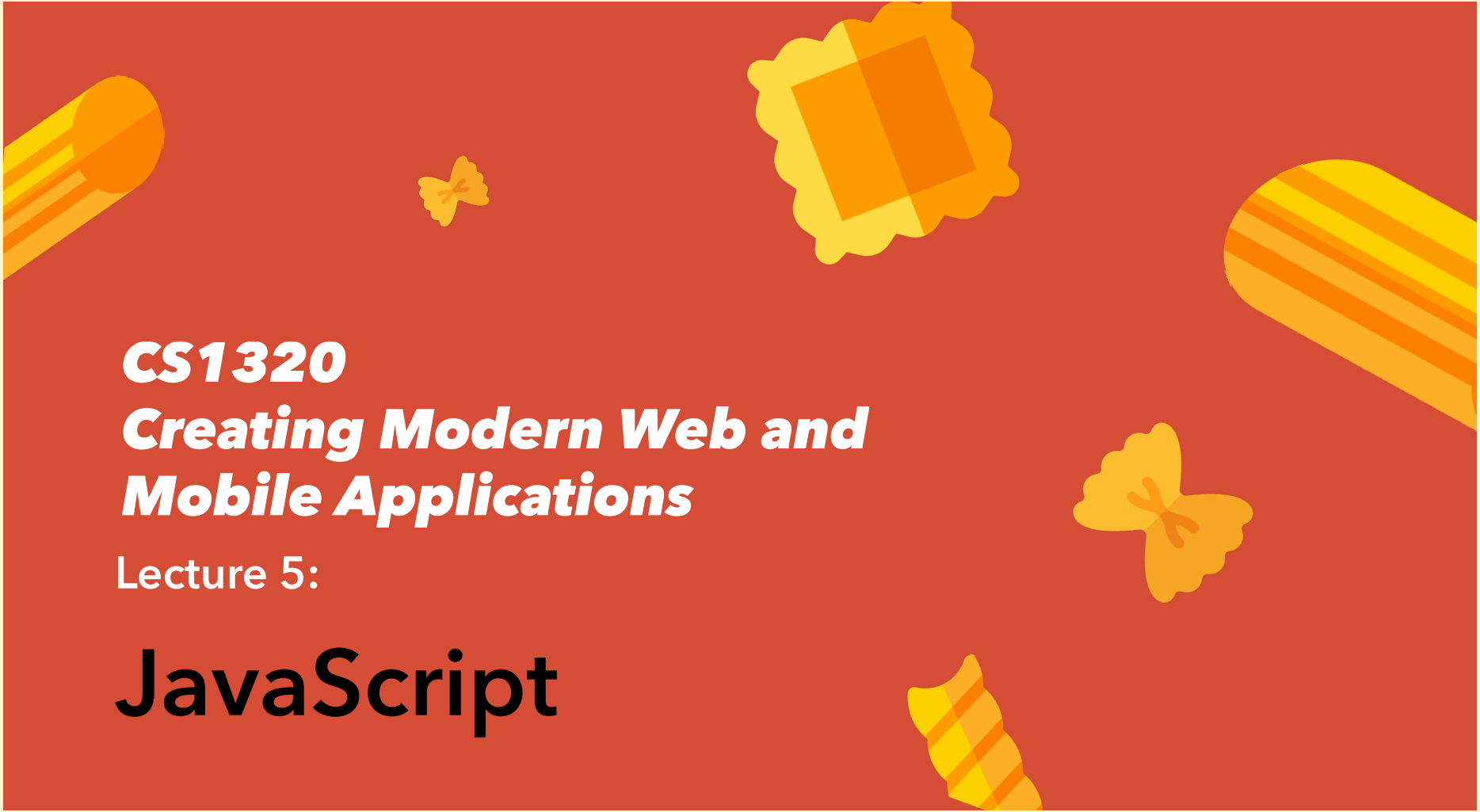


CS1320
Creating Modern Web and Mobile Applications

Lecture 5:

JavaScript

JavaScript & Tracks

- Designer
  - What JavaScript can be used for
  - When it should be used on a web page
  - How and when to incorporate interaction into your designs
    - What is possible, easy, …
- Concentrator
  - How to use JavaScript
  - Common errors
  - How to write and debug JavaScript code
  - What are the important features of the language

Static versus Dynamic Pages

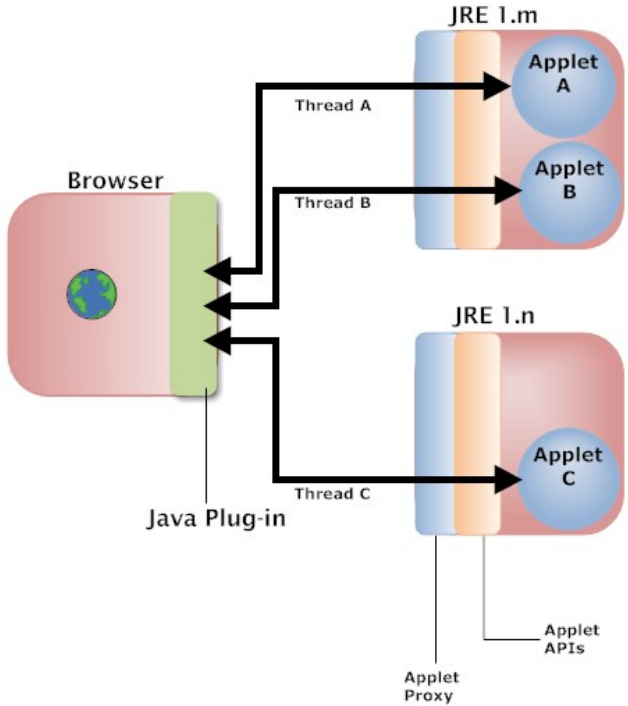
- What does dynamic mean
- Most good application user interfaces are dynamic
  - Examples
- Web Pages are inherently static
  - HTTP model: action replaces the whole pag
- Web applications require dynamics
  - Can these be done with pure HTML?

HTML is Basically Static

- Provides a description of the page, not what to do with it
  - Dynamics from built-in widgets (forms)
    - Clicking on submit causes a new page request, not an action on the page
    - With name-values pairs for the widgets as part of the URL or post data
    - Result is a page that **REPLACES** the current page
  - Limited dynamics from CSS tags
  - Limited interaction
- Is this sufficient for a web application?

How to Allow Dynamic Interactivity

- Plugins
  - Code (library) that is loaded into the browser
    - Using a somewhat standard API
    - Browser and platform specific
  - Introduce security and other problems
    - People are told not to install these
- Applets
  - Java programs downloaded and run by the browser
    - Using a standard interface
  - Introduce security and other problems
    - Java runs in a sandbox (you hope)
    - People are told not to enable these



How to Allow Dynamic Interactivity

- Extend HTML into a programming language
  - Historically different languages tried
    - VBScript, JavaScript, others
    - Hence the SCRIPT tag in HTML
  - CSS has some capabilities for interactivity
    - But it isn't a programming language
  - Eventually **JavaScript** won
    - Officially ECMAScript (standardized); ES6 is the latest incarnation
  - Available in almost all browsers today
    - Can be disabled

JavaScript Example

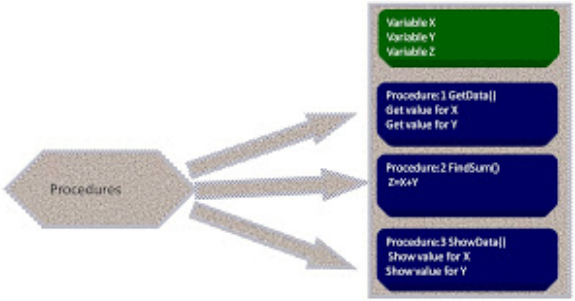
- Sumcompute.html
  - Show in operation
  - Lets look at how it works

What is JavaScript

- Type-less (dynamically typed) Scripting Language
  - Data is typed dynamically (at run time) rather than statically
  - Language is interpreted rather than compiled (in theory)
  - TypeScript – typed JavaScript is becoming more common
- Complete with libraries
  - Libraries providing basic functionality (strings,…)
  - Libraries providing access to browser capabilities
- Automatically invoked by the browser
  - Notion of events, on-conditions
  - Reactive language
- Can be embedded in HTML or in separate files for web pages
- Used for other purposes as well
  - Back end: Node.JS; Mobile Applications: NativeScript, React Native

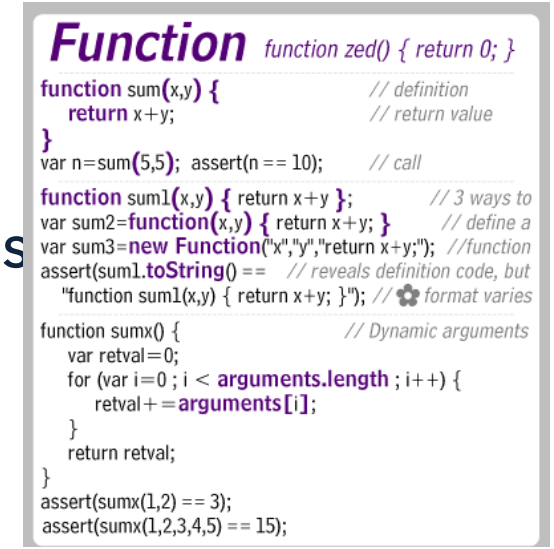
JavaScript is Procedural

- Standard control constructs
  - Loops, conditionals, …
- Functions and calls are the primary
  - User-defined functions
  - Functions called by browser
  - Library functions
- Modules provide high-level organization



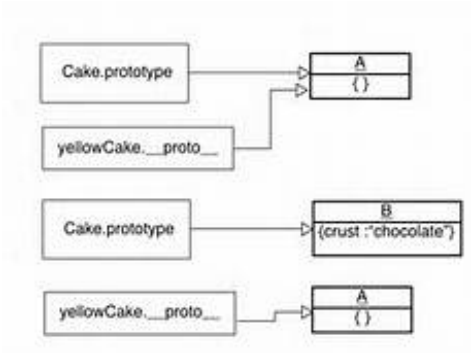
JavaScript is Functional

- Functions are first class objects
- Can be passed and used explicitly
- Lambda expressions and continuations



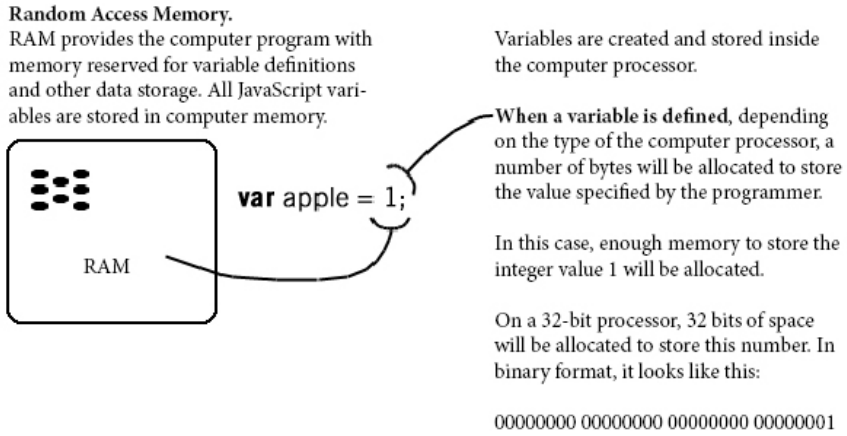
JavaScript is Object-Oriented

- Objects with fields and methods
- Prototype-based, not class-based
- new Object(), { }, { x : 1, y: 3+4 }
  - Dynamic set of fields and methods
- new Type()
  - Type is a function, not a class
  - Use of **this** inside its methods refers to the object
- Latest JavaScript has Java-like classes with inheritance, etc.
  - Syntactically, not internally



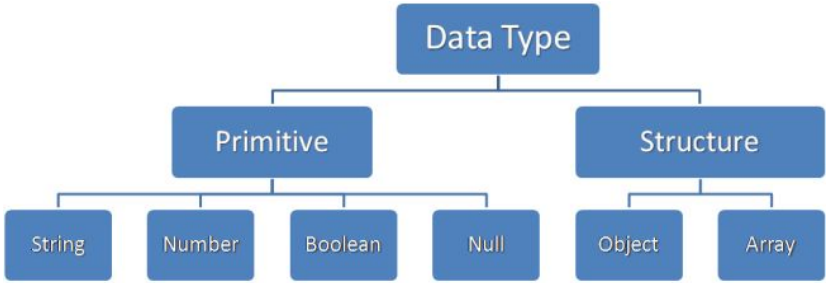
JavaScript Declarations

- No types => No declarations
- Except you have scopes
  - Global scope
  - Function scope
  - Local scopes (sometimes)
  - But function scopes nest
- Variables are global by default
  - Except for parameters
  - Except for variables declared using **var** or **let** or **const** in a function
    - Good practice: declare all variables
    - Good practice: use **let** or **const,** not **var**



JavaScript Data Types

- Numeric types (int, double)
- Booleans (true, false) {0,NaN,"",null,undefined}
- null, undefined
- Arrays
  - Indexed (Arrays)
  - Associative arrays (Objects)
- Strings ("..", '...')
- Templated Strings (`...${expr}...`)
- Regular Expressions (/pattern/g)
- Functions
- Objects
  - Field-value pairs with some inheritance
  - Values can be functions (methods)
  - Associative arrays



JavaScript Strings

- Can use single or double quotes
  - Backslash escapes
- JavaScript is designed somewhat for string processing
- String equality (s == “hello”)
- String concatenation (“hello” + “ ” + “world”)
- Other string functions
  - indexOf, split, substring, charAt, toUpperCase, toLowerCase
  - endsWith, startsWith, contains

JavaScript Templated Strings

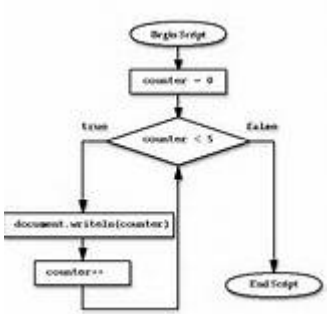
- Use backquote (`) as delimiter
- Can span multiple lines (new line included in the string)
- Can have embedded expressions
  - ${expression}
  - Replaced with the string value of the expression
  - Replaces concatenation operations and complex expressions

JavaScript Regular Expressions

- Regular expressions are useful in web applications
  - Checking formats, values
  - Advanced string processing (find/replace)
- let x = /pattern/mods
  - let re = /ab+c/;
  - let re = /\bt[a-z]*\b/i;
- Regular expression are a basic prir

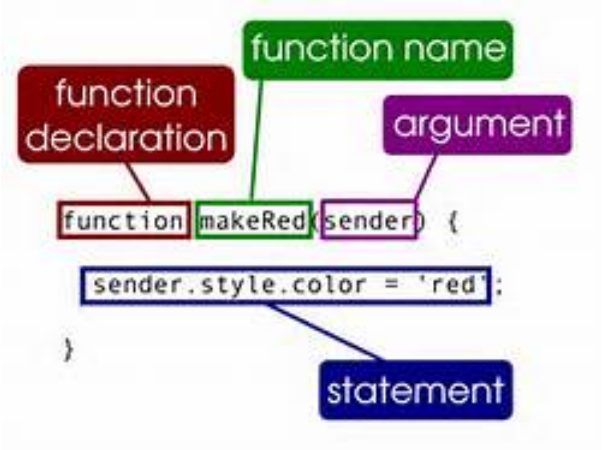
JavaScript Control Constructs

- if (test) { … } else { … }
- while (test) { … }
- switch (expr) { case: … }
- break, continue
- for (init; test; update) { … }
- for (let x in expr) { … }
  - Expr is an object
  - x is the fields (indices) of the object
  - Can use const or let
- for (const x of expr) { … }
  - Expr is an Iterable (Array, String, Map, Set, NodeList, …)
- try … catch

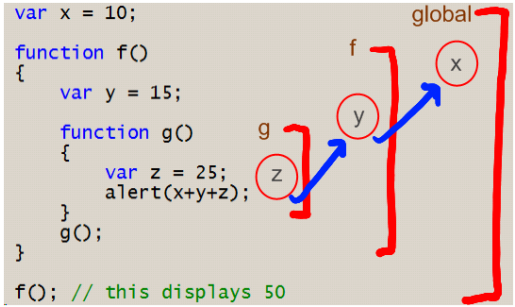


JavaScript Functions

- function name(arg,arg,…) { … }
  - No argument matching (type or number)
  - return value
  - name = function(args) { … }
  - Default argument values: arg = value
  - Varargs (arg,arg, …rest)
- (arg,arg) => expression
  - (arg,arg) => { statements }
- Functions are first-class objects
  - Can be assigned to variables
  - Can be passed as arguments
  - Can be used as values



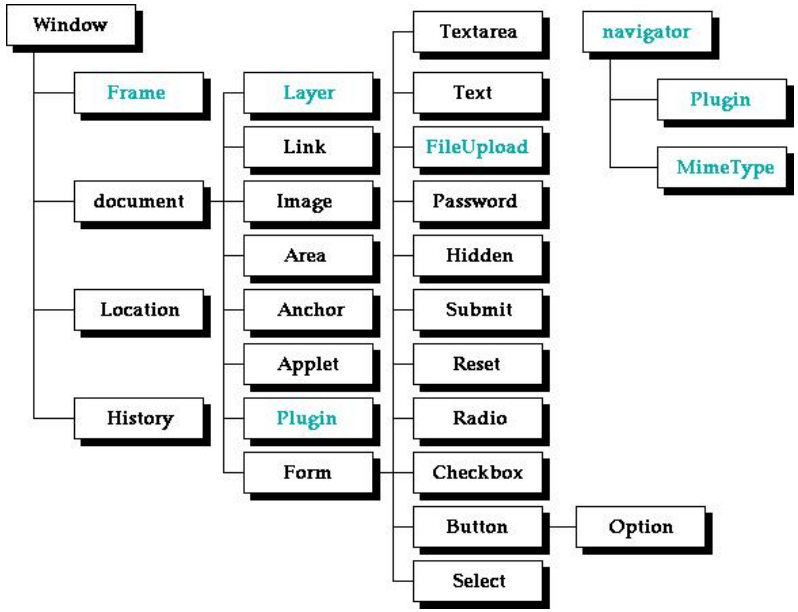
JavaScript Scopes



- Global, function, and local scopes
- Variables are global unless otherwise stated
  - **var x;** declares a variable to be local if in function
    - Can occur anywhere in the scope (& multiple times)
  - **let x;** declares a variable to be in local scope
  - **const x;** declares a non-changeable variable in the local scope
  - parameters are local
  - Function scopes can nest with nested functions
- Many JavaScript problems are scope errors

JavaScript Objects

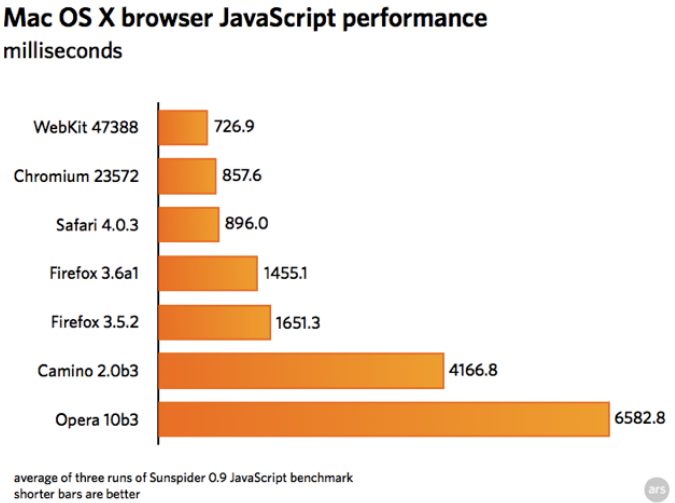
- An object is a dynamic mapping of fields to values
  - let x = new Object(); let x = { }
  - x.y = 5, x["y"] = 5
  - x.plusone = function() { return x.y + 1; }
    - x.plusone() == 6
  - for (let x in object) { print x,object[x]; }
- Objects can be defined explicitly
  - function Type(a) {
    - this.field = a;
    - this.method = function() { return this.field+a; }; }
  - var x = new Type(5);
- Objects can be defined incrementally
  - Type.prototype.method = function() …



JavaScript Arrays

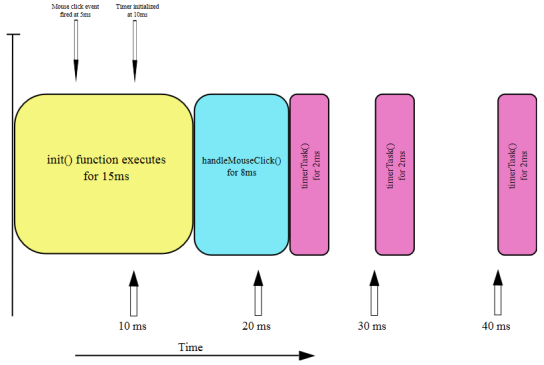
- Arrays are indexed (numerically)
  - x = [1,2,"a",5]
  - x[0] = 1; x[1] = 2;
  - x[2]= "a"; x[3] = 5;
- Should be used as such
  - Not associative
    - A["hello"] works but not the way you expect
    - Why?
  - Do **not** use *for(let x in ARRAY)*
    - Might not do what you expect
    - Missing elements, extra elements
  - You can use *for(let i = 0; I < ARRAY.length; ++i)* or *for(let x of ARRAY)*

JavaScript in the Browser



- Designed for interaction
  - JavaScript code is typically not running all the time
    - Invoked when something happens
    - What might that something be?
- Event Types
  - onLoad, onUnload (of page, frame, …)
  - On widget value change; form submit; mouse over; …
  - On almost any possible event you may want to trigger on
- As part of HTML, can specify the event handler
  - onXXX='expression'
  - Expression is typically a function or a call to a user-defined function

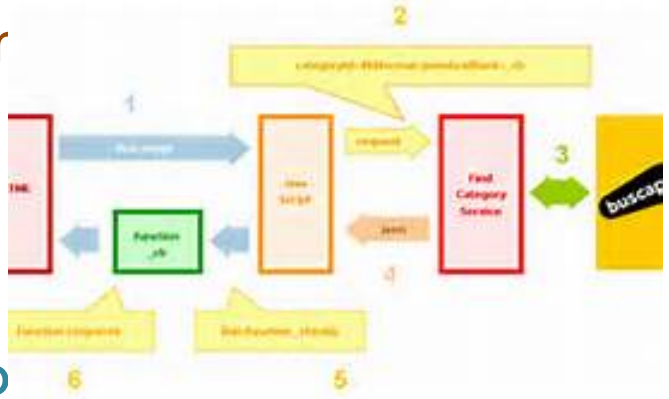
JavaScript and Threads



- JavaScript is single-threaded
  - Runs as a single thread in the browser too
    - Can hold up other things
  - Considered part of page loading or UI handling
    - The browser isn't doing something else when your JavaScript is running
- Promises provide language-level background processing
  - We'll cover these next week

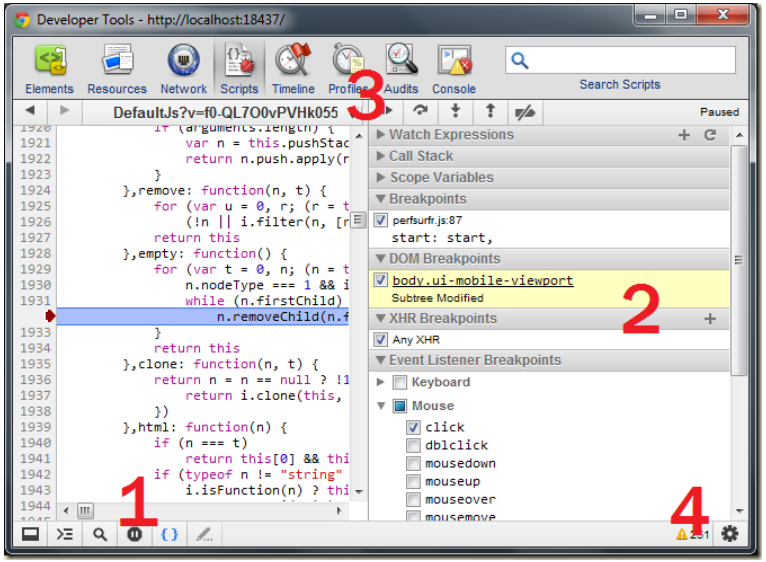
What Dynamics Can JavaScript Do?

- Check the values of fields before submission
- Compute new values as part of subm
  - E.g. encrypt a password
- Edit values in various ways
- Add values to other fields
- Establish timers and simple animatio
- Draw on HTML5 canvases
- Modify parts of the page



JavaScript Debugging

- It is useful to be able to debug your JavaScript
- Most browsers today include a JavaScript debugger
  - Firebug as part of firefox
  - IE developer tools
  - Chrome developer tools
  - Safari developer tools
- Facilities
  - Set breakpoints, examine values
- Learn to use it
  - Before you ask TAs what's wrong



JavaScript Example

- Sumcompute.html
  - What was the problem
  - Lets look at how it works

JavaScript has its Quirks

- Objects are very different from Java/C++
  - Newer versions have Java-like classes however
- Functions are first class objects
- Function scopes are defined based on var staten
  - Globally, not in-order,
- Automatic type conversion
- Strict versus non-strict equality testing
- eval function
- Semicolons are optional if unambiguous
- Timeouts (setTimeout and setInterval)
- Read up on the language (prelab)

What Else Would You Like to Do

- Change the page, animate things
- This can require extensive computatior
- Next Time

Next Time

- DOM manipulation
- Assignment 1 is out
- Lab 1 is due before next lecture
- Project preferences due by midnight
- Homework:
  - PreLab 2: to familiarize yourself with JavaScript

JavaScript Demo

- Basic types

JavaScript Demo

- Objects and Arrays

Question

A browser plug-in is:

A. A JavaScript file that can be downloaded as part of a page.
B. A library that becomes part of the browser once downloaded and accepted by the user.
C. A technique that allows you to play a video directly in the browser.
D. A Java program that is run inside the browser.
E. A file that is loaded into the web server to handle special access for a web application.

Questions

The JavaScript language is

I. Procedural
II. Functional
III. Object-oriented

A. I
B. III
C. I and II
D. I and III
E. I, II and III

Experience Reports

- What page did you look at and what types of things did you find that were dynamic?
- What did you find that you think should be dynamic?