I have always wished that my computer would be as easy to use as my telephone. My wish has come true. I no longer know how to use my telephone.

Bjarne Stroustrup

Video Lectures • *Programming* 05 Aug 2007 12:25 pm

(**25** votes, average: **5** out of 5)

Loading ...

Learning JavaScript Programming Language through Video Lectures

I decided I wanted to learn JavaScript Programming language better. I had been programming in it now and then but I had never really developed any good skills in it.

If you have read about this blog page then you know that I also run Free Science Online blog which is all about free video lectures online. In my May's post I had found some really good programming video lectures, 13 of them being on JavaScript. Since I run this video lecture blog, I obviously have great interest in video lectures, so why not try to learn better JavaScript from these video lectures?

These lectures are given by Douglas Crockford who is a senior JavaScript Architect at Yahoo!. He is well known for his work in introducing JavaScript Object Notation (JSON).

First four lectures are on the basics of language:

- JavaScript Video Lecture Part I
- JavaScript Video Lecture Part II
- JavaScript Video Lecture Part III
- JavaScript Video Lecture Part IV

Sometimes Yahoo! Video gives this error: **Sorry! This video is no longer available on Yahoo! Video.** In this case refresh your browser a couple of times!

The next three lectures are on Advanced JavaScript:

- Advanced JS Part I
- Advanced JS Part II
- Advanced JS Part III

Then there are 6 more lectures on JavaScript which should probably be viewed only after you have viewed the ones just mentioned.

- Advancing JavaScript with Libraries - Part I and Part II
- Maintainable JavaScript
- An Inconvenient API: The Theory of the DOM - Part I, Part II and Part III

Viewing the YUI Theater I just found another JavaScript lecture which was published just recently:

- JavaScript - The Good Stuff

My approach to watching video lectures

I have been watching various video lectures for almost 4 years now. Mostly mathematics, physics and theory of computer science. My approach to getting most of the lectures is the following. When I watch the video lectures I take notes just as if I were in class.
Even better, when I do not understand any part of the lecture I can always rewind it back and see that fragment again. I can also pause the lecture, think for a while and then continue. But that's physics and maths.
Here is a photo of notes I have taken while watching MIT's 803: Vibrations and Waves (yes! it's available completely for free at MIT's Open Course Ware)

I am serious about physics and maths video lectures, as you can see in the image, all the main results are boxed in red, the results are fully derived (even if the professor does not do it on blackboard). Btw, one lecture perfectly fits on both sides of an A4 sheet.

Here is a close-up of Lecture 13: Electromagnetic Waves - Plane Wave Solutions to Maxwell's Equations - Polarization - Malus' Law.

(Sorry about the bad quality of the photos, I shot them with my Nokia N73 cell phone camera)

This approach probably does not work with programming languages and computer tools. Because mathematics and physics is mostly done on paper, watching these video lectures and taking notes is actually doing them. The process of taking notes develops the skills because you work with the new concepts/operators/theorems/whatnot. Not so in programming languages. You can take a book on a new programming language, read it, and the next moment you can't even write the hello world program because you have only got familiar with the subject and have not developed the skills. I have experienced this myself.

Here is my approach how I am going to learn JavaScript from these lectures. I might adjust this approach at any moment if I find it not working, I will update this post appropriately then.

I will definitely watch all 11 video lectures. I will start with the first four basic lectures, watch them one by one, take notes as with physics video lectures and experiment as I go.

I will be taking notes lightly to have my mind really think the information I am getting over. But no red boxes around constructs as with physics. That's the experimentation part to learning. I am going to try the new constructs as soon as I see them so they stuck in my mind better.
Update: I dropped the idea of taking any notes on paper, because I am blogging the key points from lectures here.

Also to make this article interesting, I will annotate each lecture if something really interesting catches my eye. As I mentioned, I have programmed JS before so I am not sure how much I will learn from the first four basic lectures.

In my previous blog post I used Windows Script Host to create a program in VBScript. The other language the same script host runs is JScript which conforms to the same standard as JavaScript. So I should be safe doing JavaScript experimentation in JScript.

Points that caught my attention in JavaScript Video Lecture Part I

- (00:45) World's most misunderstood programming language - has "Java" in its name and "Script". It has nothing to do with Java programming language and it's a real programming language not some tiny scripting language
- (02:38) There are generally no books available to learn JS from - all are bad and full of nasty examples
- (02:56) The only book recommended is JavaScript: The Definitive Guide, 5th Edition
- by David Flanagan - the least bad book
- (03:37) JavaScript is a functional language
- (08:12) Microsoft reverse engineered original implementation of JavaScript and called it JScript to avoid trademark issues
- (09:49) During standardization, the original bugs were left as is without fixing to prevent already written programs from breaking (Douglas slips and says "Sun" but he actually means "Microsoft")
- (12:16) One of the key ideas of the language is prototypal inheritance where objects inherit from objects and there are no classes
- (12:45) One other key idea is that functions are first-class objects
- (13:36) There are no integers in the language, everything is represented as 64-bit floating point numbers
- (14:30) *NaN* (Not a Number) is not equal to antyhing, including *NaN*. Which means *NaN* == *NaN* is false
- (15:22) Type of *NaN* is *Number*
- (15:52) + prefix operator does the same thing as *Number* function
- (16:08) Generally always specify radix argument in *parseInt* function because if the first character is '0′ and there is no radix argument provided, it will assume that '0′ to be an octal constant
- (17:15) Each character takes 16-bits of memory
- (17:56) There is no separate character type in JS, characters are represented as strings with a length of 1
- (19:55) *undefined* is the default value for uninitialized variables and parameters
- (20:38) Falsy values are: false, null, undefined, "", 0, NaN. All other values (including all objects are truthy). Everything else in the language are objects
- (22:15) Object members can be accessed with dot notation Object.member or subscript notation Object["member"]
- (22:59) The language is loosely typed but not "untyped"
- (25:19) Reserved words are overused, they can't be used in dot notation as method names
- (27:25) Operators == and != can do type coercion, so it's better to use === and !=== which do no coercion
- (28:24) Operator && is also called the 'guard operator', Operator || is also called the 'default operator'
- (30:16) The bitwise operators convert the operand to a 32-bit signed integer, perform the operation and then turn the result back into 64-bit floating point. Don't use bitwise operators in cases like multiplying by 4 using << 2. It will not.

After having watched the first lecture I decided that there was no point in taking notes on paper because I am blogging the key points here and trying various examples I can come up with with JScript, taking notes just wastes time.

Points that caught my attention in JavaScript Video Lecture Part II

- (00:20) Break statements can have labels
- (00:41) Iterating over all the members of an object with for (var name in object) { } syntax, will also iterate over inherited members
- (06:10) There is only global and function scope in JavaScript, there is no block scope
- (07:40) If there is no expression in a *return* statement, the value returned is *undefined*. Except for constructors, whose default return value is *this*
- (08:29) An object is an unordered collection of name/value pairs
- (21:10) All objects are linked directly or indirectly to Object.prototype
- (23:42) Array indexes are converted to strings and used as names for retrieving values

Points that caught my attention in JavaScript Video Lecture Part III

- (00:29) Functions inherit from Object and can store name/value pairs
- (02:00) The function statement is just a short-hand for a var statement with a function value
- (02:35) Functions can be defined inside of other functions
- (03:08) JavaScript has closures
- (07:29) There are four ways to call a function: function form, method form, constructor form and apply form
- (10:00) *this* is an extra parameter. Its value depends on the calling form
- (10:30) When a function is invoked, in addition to its parameters, it also gets a special parameter called arguments
- (11:53) Built-in types can be augmented through (Object|Array|Function|Number|String|Boolean).prototype
- (14:09) The *typeof* prefix operator returns 'object' for *Array* and *null* types
- (15:23) *eval*() is the most misused feature of the language
- (21:57) In web browsers, the global objects is the window object
- (22:51) Use of the global namespace must be minimized
- (23:11) Any variable which is not properly declared is assumed to be global by default
- (23:45) JSLint is a tool which helps identify weaknesses

- (24:21) Every object is a separate namespace, use an object to organize your variables and functions
- (27:15) Function scope can create an encapsulation

Points that caught my attention in JavaScript Video Lecture Part IV

- (03:30) The language definition is neutral on threads
- (11:20) When the compiler sees an error, it attempts to replace a nearby linefeed with a semicolon and try again
- (12:51) Do not use extra commas in array literals. Netscape will tell you that length of [1,2,3,] is 3 while IE will tell it's 4
- (18:48) Key ideas in JavaScript - Load and go delivery, loose typing, objects as general containers, prototypal inheritance, lambda, linkage through global variables

These four lectures gave me much better theoretical understanding of JavaScript but just a little better practical skills. I should do a project entirely in JavaScript to become more skillful.

I can't wait to see the Advanced JavaScript lectures. At the end of the 4th lecture Douglas said that they will continue with theory of DOM which I will follow and only then continue with Advanced JS.

Points that caught my attention in The Theory of the DOM Part I

- (03:31) A scripted web browser, Netscape Navigator 2, was first introduced in 1995
- Best thing happened to have standards work was Mozilla abandoning the Netscape layer model in favor of the W3C model
- (10:03) List of browsers Yahoo! wants their JavaScript library to run on are FireFox 1.5, FireFox 2.0, Safari 2, Internet Explorer 6, IE 7, Opera 9
- (12:11) The <script> tag first appeared in Netscape Navigator 2
- (13:05) <!-- –> comment around script was a Netscape 2 hack for Mosaic and Navigator 1.0
- (14:51) W3C deprecated *language*=javascript attribute in script tags, don't put it there anymore
- (18:48) If you call document.write before onload it inserts data into the document, if you call it after, it replaces the document with the new stuff
- (20:25) *name*= is used to identify values in form data and to identify a window or frame
- (20:45) *id*= is used to uniquely identify an element so that you could get access to it
- (20:59) Microsoft introduced document.all as a super-collection of all elements with name or id
- (21:39) W3C instead said use document.getElementById(id) and document.getElementsByName(name)
- (23:41) Document tree structure is different for IE than for other browsers because Microsoft decided to depart from W3C standard and not to include whitespaces as text nodes in the tree
- (25:02) *document.body* gets you to body node of the tree, *document.documentElement* gets you to the html root tag of the tree

After watching this lecture I decided to add time where each point that caught my attention happened so that if anyone is interested in any of the points he/she could just fast forward to that place in the video. Eventually I will go through the videos up to this one once more and add timestamps.

Points that caught my attention in The Theory of the DOM Part II

- (04:32) The guy designing CSS chose a not so appealing names to a programmer for CSS style names. JavaScript guys converted those to camel case in JavaScript which is probably the least compatible with CSS style names
- (08:31) Replacing a child is done with "java oriented, model based, nothing in common with reality sort of api" through old.parentNode.replaceChild(new, old) where you specify *old* twice
- (09:03) It is important to remove any event handlers from the object before you delete it
- (10:10) Microsoft and their Internet Explorer were the first to realize that it is convenient to provide access to HTML parser and provided innerHTML property which can be assigned a string containing HTML directly
- (10:50) There is no standard describing innerHTML property
- (12:12) The browser has an event-driven, single-threaded, asynchronous programming model
- (12:55) There are three ways to adding event handlers - classic mode (node["on" + type] = func), Microsoft mode (node.attachEvent("on" + type, func)) and W3C mode (node.addEventListener(type, func, bool))
- (14:50) Microsoft does not send an event parameter, they use the global event object instead.
- (15:58) There are two ways how events are handled - trickling and bubbling
- (17:23) The extra bool parameter in W3C mode of adding event handlers node.addEventListener(type, func, bool) tells whether the events are processed bottom up (bubbling) or top down (trickling)

Points that caught my attention in The Theory of the DOM Part III

- (01:26) Hugest memory leaks happen in IE 6
- (01:33) Because of that you must explicitly remove event handlers from nodes before deleting or replacing them
- (06:49) *self*, *parent* and *top* are aliases of window object
- (08:10) A script can access another window if and only if document.domain === otherwindow.document.domain
- (10:10) There are three ways to get cross browser compatibility - browser detection, feature detection, platform libraries
- (11:20) Internet Explorer 1.0 identified itself as "Internet Explorer" but many sites refused to serve the contents complaining that it was not "Mozilla" so in version 1.5 IE identifies itself as "Mozilla"
- (12:16) Browser detection cross compatibility is the least recommended way
- (15:37) Platform library cross compatibility is the most recommended way
- (15:35) Platform library cross compatibility is the most recommended way
- (18:48) No browser completely implements the standards and much of the DOM is not in any standards. If there was a 100% standards compliant browser, it would not work!
- (19:19) When programming DOM: 1) do what works; 2) do what's common; 3) do what's standard

Okay, I watched the whole Theory of DOM course and have gained good theoretical knowledge but basically no practical skills. To get better with the JavaScript and DOM will require me to do some interesting practical projects with both of these guys.

Now I am off to watch Advanced JavaScript lectures and then the remaining.

Points that caught my attention in Advanced JavaScript Part I

- (01:20) In prototypal inheritance objects inherit directly from objects, there are no classes
- (01:30) An objects contains a “secret link” to another object. Mozilla calls it __proto__
- (03:36) If looking for a member fails, the last object searched is Object.prototype
- (07:50) When functions are designed to be used with *new*, they are called constructors
- (08:13) new Constructor() returns a new object with a link to Constructor.prototype
- (09:40) Always have your constructors named with a capital letter so you at least develop reflex for putting a *new* in front of it
- (09:48) Forgetting the *new* still makes code work but it does not construct a new object. This is considered one of the language design errors
- (10:00) When a new function object is created, it is always given a prototype member
- (10:49) “Differential inheritance” is a special form of inheritance where you specify just the changes from one generation of objects to the next
- (17:24) JavaScript doesn’t have an operator which makes a new object using an existing object as its prototype, so we have to write our own function
- (18:50) A “public method” is a function that uses this to access its object
- (21:55) Functions can be used to create module containers
- (25:01) “Privileged methods” are functions that have access to “secret” information and they are constructed through closures
- (28:05) “Parasitic inheritance” is a way of creating an object of an augmented version an existing object.

Points that caught my attention in Advanced JavaScript Part II

- (06:30) Pseudoclassical patterns are less effective than prototypal patterns or parasitic patterns
- (09:06) Inner functions do not have access to *this*
- (09:32) In JavaScript 1.0 there were no arrays
- (10:31) When arrays got added, *arguments* object was forgot to be converted to Array object and it continues to be an array like object
- (15:47) There are debuggers for IE - Microsoft Script Debugger, which is bad and two debuggers built in Visual Studio and Office 2003. Mozilla has Venkman and Firebug. Safari has Drosera
- (17:30) *funny instruction on how to get debugger working with Office 2003*
- (24:29) All implementations of JavaScript have non-standard *debugger* statement which cause a breakpoint if there is a dubgger present

Points that caught my attention in Advanced JavaScript Part III

- (04:20) Array join() method is much faster for concatenating large set of strings than using operator +
- (07:03) Just have the server gzip the JavaScript source file to minimize the load times, avoid tools which can introduce bugs such as minificators and obfuscators
- (07:19) JSON

The advanced JavaScript lectures provided lots of idioms and patterns used in the language. I did not do much experimentation and have really grasped just the concepts and overall structure of these advanced concepts.

Now I am going to watch “Advancing JavaScript with Libraries” by John Resig, creator of the JQuery JavaScript library and author of Pro JavaScript Techniques

, is a Mozilla technologist focused on the relationship between Mozilla and the world of JavaScript libraries.

Interesting points from Advancing JavaScript with Libraries Part I

- (08:20) In IE7 basically the only change to JavaScript was to XmlHTTPRequest object
- (25:14) There are two standards for querying the DOM document - XPath and CSS 3 selectors
- (26:13) IE doesn’t have very good CSS selector support, because of that users have been using the very minimum of CSS selectors which almost equates CSS 1
- (27:30) jQuery allows selecting elements from the DOM by using CSS 3 selectors which is done in one line of jQuery code instead of 20 - 25 lines of just JavaScript DOM code

Interesting points from Advancing JavaScript with Libraries Part II

- (05:15) Users are expecting the DOM selectors to behave more like CSS, that is like when a new CSS selector is added, it propagates to all the elements affected. The users expect the same to happen when a chunk of HTML is added to the DOM, that the handlers get added to them without re-running any code
- (12:30) Object Relational Mappings
- (14:50) Libraries create new patterns on top of existing APIs

There were not that many points that got me interested because it was pretty obvious stuff. It was just interesting to see that the DOM is not perfect and there are many bugs which one or the other browser fails, why they fail and how to solve these DOM related problems. Also typical programming meta-problems were discussed such as JavaScript trying to get elements before browser has loaded the DOM, how the white spaces are treated and what methods to use for navigating the DOM. Later the lecture went on how to query the DOM tree using jQuery and mix of XPath and CSS 3. Then it is discussed how injecting HTML in an existing document is done, what’s tricky about it and what problems can arise. Finally the lecture continues with FUEL and object relational mappings.

The other lecture I watched was “Maintainable JavaScript” by Nicholas Zakas. He is an engineer on the team that brings you My Yahoo!, one of the most popular personalized portals on the web. He is also the author of two books on frontend engineering, including “Professional JavaScript for Web Developers

,” one of the best tomes of its kind.

Interesting points from Maintainable JavaScript

- (01:15) It is estimated that as much as 80% of time is spent maintaining existing code
- (04:30) Maintainable code is understandable, intuitive, adaptive, extendable and debuggable

- (16:20) There are three layers on the client side - JavaScript for behavior, CSS for presentation and HTML for structure
- (23:42) Programming practices
- (26:30) Namespace your objects
- (32:10) Avoid null comparison, use *instanceof* or *typeof* operators
- (37:15) Write code in separate JavaScript files and use a build process to combine them
- (40:47) Summary of writing maintainable code - code conventions, loose coupling, programming practices and build process

There are not that many interesting points anymore because most of the stuff has been covered in the previous lectures. Apart from that these lecture did not teach me much new because this stuff was pretty obvious. The only point to watch this lecture is to refresh all these obvious suggestions - agree on indentation and naming conventions in your team, comment difficult algorithms and large sections of code, don't write obvious comments, comment hacks, loose coupling, careful use of complex code and design patterns, etc.

What do you think about these lectures?

Post tags: education, javascript, js, jscript, learning, programming, video lectures, yui theater
Bookmark with del.icio.us:
Post to Reddit:
Digg it!
Stumble it!
Email Post
| Print Post
| Permalink
| Trackback

(Popularity: 100%) 62,027 Views

Did you like this post? Subscribe here:

Related Posts

- Learning Python Design Patterns Through Video Lectures
- Learning Python Programming Language Through Video Lectures
- Video Lecture on Best Practices in JavaScript Library Design
- Searching and Mining Open Source Code from the Web
- Video Lecture: From Nand to Tetris in 12 Steps
- Revealing Reddit Score for Just Posted Links with FireFox and GreaseMonkey
- Bjarne Stroustrup's Video Lecture on C++0x Standard
- How Reddit Top and Hacker Top Programs Were Made
- How to Read Reddit the Fanatic Programmer Way
- Follow Reddit from the Console

45 Responses

1. foobar15 says:
   August 10th, 2007 at 8:01 am

   1. Very useful information. Great blog.

2. uchitha says:
   August 10th, 2007 at 12:36 pm

   2. Hi,

   2. Is there a way to save these video lectures? I have a bad connection here in Sri Lanka and its not possible to see the video real time.

3. Andreas Kompanez says:
   August 10th, 2007 at 12:48 pm

   3. Thanks a lot, i'm in the same situation. Using javascript all the time, but knowing nearly nothing about this language. Hope to change it, after this movie session;) Your notes are also very interesting.

4. apprendre javascript par la video par Douglas Crockford | Webmaster-dZ says:
   August 10th, 2007 at 4:17 pm

   4. […] source: catonmat […]

5. Peteris Krumins says:
   August 10th, 2007 at 11:28 pm

   5. uchitha, yes, it's possible to save the video lectures.

   5. Go to: YUI Theater and you will see "M4V download" links next to each lecture. Right click and choose "save as" to save the lectures!

   5. To play M4V files you will need ffdshow video codec. Once installed your player will play the lectures!

6. links for 2007-08-11 « Simply… A User says:
   August 11th, 2007 at 3:31 am

6. […] Learning JavaScript Programming Language through Video Lectures - good coders code, great reuse (tags: class free google hacks howto interesting java tutorials toread education lectures tutorial programming video javascript **) […]

7. Carsten's Log » Learning JavaScript through Video Lectures says:
August 11th, 2007 at 12:23 pm

7. […] this article Peteris Krumins explains how he is learning the finer points of JavaScript programming by watching […]

8. A few good links « Silent Walker says:
August 13th, 2007 at 9:48 am

8. […] don't feel like reading it (like always, CSS … naah! very bad, I know). Well someone here has posted few links to video lectures of Java script. What I found pretty amusing is, the author […]

9. andy says:
August 17th, 2007 at 7:09 am

9. Thank you peter.
I am also using video lessons at
http://freevideolectures.com/webdesign.html recently.

10. Scott says:
August 17th, 2007 at 11:34 pm

10. Awesome site. Keep up the great work!!

11. Eric Johnson says:
August 28th, 2007 at 7:42 pm

11. Peteris,
Thank you very much for publishing these links, both here and on your video lectures blog. I too have worked my way through a few of the OCW courses and really enjoyed them. I find it's much easier to learn from a lecturer than from reading alone. I don't have anything to add; I just wanted to thank you for all doing all of the "legwork" in finding and sharing these links.

12. binary says:
September 1st, 2007 at 8:27 pm

12. Hi there

12. Nice article. Although I'm not a big fan of video lectures yet (to be honest, I think I haven't seen any video lecture yet; shame on me ;D), I think I'm going to watch these today and/or tomorrow. Seems like a good material.

12. Before watching those lectures, I decided to read all your conspects about them (tx for conspects, they're quite valuable). While reading, I found one thing that I can't agree with:

12. "(06:49) *self*, *parent* and *top* are aliases of window object"

12. OK, all those 4 objects have the same type, but they are *not* the same object, therefore they can't be called aliases. Not in documents with frames…

12. As far as I understand, these were invented for working with multi-frame documents (frameset/frame, iframe), where they mean what they're called. "self" is the same window where the script runs, "parent" is parent window of a window where the script runs, and "top" is a top window. You may think of document with frames as of window with multiple child windows. If you have a document with *iframe*, you will have a window within a window. If there is another *iframe* in that first iframe, you will have a window within a window, that is within another window. If you need to get to a parent window while working in child window, you have to use parent or top instead of window. That's the purpose of those variables.

13. links for 2007-09-02 at DeStructUred Blog says:
September 2nd, 2007 at 5:19 am

13. […] Learning JavaScript Programming Language through Video Lectures - good coders code, great reuse (tags: Javascript Development Video Tutorial Blog) […]

14. Peteris Krumins says:
September 6th, 2007 at 5:38 pm

14. Hey, binary! Sorry that it took me so long to answer to your comment.

14. You are right! I had my wording wrong and what I actually meant was that they may be aliases of the same object. Thanks!

15. mohammedd abdul tawwab says:
October 7th, 2007 at 9:54 am

15. hi, m from india i was seaarchin the sites and found urs for video lectures .i jz wanna tell u to put topics on pharmacy subjects especially pharmacology , biochemistry ,clinical pharmacy pharmacetical chemistry etc in an easy approach

16. ZESHAN ALI says:
November 30th, 2007 at 2:04 am

16. HI Peteris Krumins

16. i am ali from pakistan .i am intresting in java language.you can help me in java .

16. i am student and i want to become the expert programmer in java can you tell me the link of vedio lectures of java of different universities
to improve my knowledge about java.

16. Thanks

17. Peteris Krumins says:
December 1st, 2007 at 5:21 am

17. ZESHAN ALI, thanks for your comment! I just posted a new collection of video lectures to my Free Science Online blog. Check out this post:
Computer Science and Programming Video Lectures for two courses on Java:
Java Programming and
Applications Development and Java

17. Have fun!

18. Tommy V says:
January 19th, 2008 at 2:48 pm

18. I totally agree that learning a language by video is the way forward. People come to http://www.languagebyvideo.com to learn Spanish.

19. Sushant says:
January 22nd, 2008 at 4:28 pm

19. Hi!
Thanks a lot for your comment on my blog, softer-copy.
Great work, I have updated by blog linking your post.
Thanks

20. Learning Python Programming Language Through Video Lectures - good coders code, great reuse says:
March 7th, 2008 at 5:43 am

20. […] blog post) is going to be written entirely in Python. I have a good understanding of Python but, same as I had with JavaScript, I have little experience doing projects from the ground up in […]

21. Ido says:
March 25th, 2008 at 8:57 am

21. Very good post… I like the 'show notes' that you taking with the video

As I attend some of these lectures in person it was nice to see your notes and compare them with the ones I've taken.
Keep up the good work.

21. –
http://mybuywatcher.com

22. Javascript ■■■■■ ■■ ■■■■ « webmaster cartoons israel says:
April 13th, 2008 at 1:04 am

22. […] http://www.catonmat.net/blog/learning-javascript-programming-language-through-video-lectures/ […]

23. Aaron says:
April 26th, 2008 at 11:03 am

23. I'm glad I found your website, particularly this page.

23. I am also going through the same process as you in learning Javascript. I've had David Flanagan's book for a couple of years, but never had the time to read it until now. I'm glad to learn that it's the "least bad" book. I do find most javascript tutorials to be very superficial.

23. I have the following learning plan.

23. 1. Read through the book
2. Create a mindmap for each chapter
3. Use a shell editor for testing simple scripts, or just use the javascript: protocol in the url
4. Create a set of scripts to test each part of the language in isolation. Use this as a reference for future programming
4.5 Re-inforce understanding by watching video lectures
5. Go to a script site, and pull down some of the highest rated scripts. Study each one in detail so I can pickup good design habits.
6. Visit forums to chat to people about concepts I'm unsure of.
7. Modify existing scripts
8. Create my own scripts from scratch, including bookmarklets
9. Create a few simple video lectures
10. Move onto AJAX

24. blub says:
May 15th, 2008 at 1:27 am

24. great!

25. Glen Moriarty says:
May 15th, 2008 at 2:05 am

25. Great stuff. You are one motivated guy! We are actually working on a startup that will hopefully help other people learn w/out having to try so hard. One of our main goals is to help further unlock OCW, so that it can be used in a meaningful way to really share with one another. As your tagline suggests, good learners learn, great reuse :). We are going to be launching our closed beta soon. If you are interested in teaching online, or just finding other coders that share your interest, then drop me an email and I'll be sure you get a login.

26. Videos para aprender a programar Javascript | Kabytes says:
May 15th, 2008 at 6:22 pm

26. […] visitar en Catonmat el índice completo del curso y el respectivo enlace a cada lección. Por el momento vi completo el […]

27. Abhirama says:
May 15th, 2008 at 7:32 pm

27. Awesome stuff!!!!!!

28. Reggie Drake says:
May 15th, 2008 at 7:42 pm

28. The web is full of great JS resources nowadays. Oh happy day.

29. Need to learn JavaScript? « Later On says:
May 15th, 2008 at 8:11 pm

29. […] in Education, Technology at 11:01 am by LeisureGuy Here's how: a series of video lectures together with notes. An example of the notes: Points that caught my attention in JavaScript Video Lecture Part […]

30. {Gregorio | Espadas} .com.mx » Blog Archive » Recursos para Programadores Web says:
May 15th, 2008 at 9:15 pm

30. […] Si quieres aprender JavaScript, o bien, pretendes volverte experto en este lenguaje, aquí les dejo unos video tutoriales de Douglas Crockford, uno de los gurús de JavaScript del mismísimo Yahoo!. Vía: Catonmat. […]

31. links for 2008-05-16 « Mike Does Tech says:
May 16th, 2008 at 2:47 am

31. […] Learning JavaScript Programming Language through Video Lectures - good coders code, great reuse (tags: javascript video tutorial) […]

32. links for 2008-05-16 at DeStructUred Blog says:
May 16th, 2008 at 4:40 am

32. […] Learning JavaScript Programming Language through Video Lectures - good coders code, great reuse (tags: Javascript programming Video Learning) […]

33. links for 2008-05-16 « Mandarine says:
May 16th, 2008 at 6:46 am

33. […] Learning JavaScript Programming Language through Video Lectures These lectures are given by Douglas Crockford who is a senior JavaScript Architect at Yahoo!. He is well known for his work in introducing JavaScript Object Notation (JSON). (tags: education javascript learning lectures video webdev) […]

34. Weekly linkdump #126 - max - ■■■■■ ■■■■■■■■■■■■■■■ says:
May 16th, 2008 at 10:46 am

34. […] ■■■■■■■■■■■■■■■■■■ ■ ■■■■■■■■■■■■ (■■■■■■■■■■■■) ■ ■■■■■■■■■ Javascript ■■■■■■ ■ ■■■■■■■■ ■■■■■■-■■■■■■■, ■■■■■ ■■■■■■■■■■■, Learning JavaScript Programming Language through Video Lectures […]

35. links for 2008-05-17 « Can you believe I have another blog? says:
May 17th, 2008 at 2:41 am

35. […] Learning JavaScript Programming Language through Video Lectures - good coders code, great reuse Videos on learing Javascript. (tags: javascript programming video tutorials learning lectures) […]

36. What is Javascript and some must watch tutorial videos « Technofriends says:
May 17th, 2008 at 5:47 pm

36. […] Contribution : Learning JavaScript Programming Language through Video Lectures (Thanks to Peteris for sharing this wonderful […]

37. Terry says:
May 21st, 2008 at 4:57 pm

37. Points that caught my attention in your blog title:
Javascript != Programming…

37. Didn't read the rest, starting with nonsense normally means the rest is full of it.

38. Peteris Krumins says:
May 24th, 2008 at 6:17 pm

38. Surely you are joking, Mr. Terry.

39. Soney says:
July 11th, 2008 at 7:13 am

39. Very very useful consolidation thank you very much for sharing

40. Daniel says:
July 21st, 2008 at 9:42 pm

40. Hi.. First Congratulations for your site.. a great resource. Second.. don't become too nerd.. have a life, friends and gf.. is good too

:-)

40. See if you can help me. I am still didn't find the best way to learn a programming…I have a lot of "half" knowledge about some languages.. but nothing deep. The "problem" is that if you wanna learn JS.. and start google… you will receive so many information that make you confuse.. and don't know where start.

40. Any tips are welcome.

40. Bests,
Daniel

41. Peteris Krumins says:
July 22nd, 2008 at 12:14 am

41. Daniel, my advice is to work on projects. Whatever you can think of. For example, make a JavaScript library which creates various HTML form elements automatically.
The more programming you do, the better you will become! Start with simple projects, then move to more complicated ones.

42. A Year of Blogging - good coders code, great reuse says:
July 22nd, 2008 at 5:41 am

42. […] Learning JavaScript through Video Lectures (44′548 views) […]

43. Learn JavaScript programming with these video lectures | Askarize.com says:
July 22nd, 2008 at 7:06 pm

43. […] [via CatOnMat] […]

44. Bookmarks for April 29th through August 4th | just another 45 miles says:
August 5th, 2008 at 10:49 am

44. […] Learning JavaScript Programming Language through Video Lectures - good coders code, great reuse - […]

45. MIT's Introduction to Algorithms: Lectures 1 and 2 - good coders code, great reuse says:
August 20th, 2008 at 2:32 am

45. […] I wrote earlier, I am very serious about watching video lectures. If they are math-intensive, I usually take notes […]

Leave a Comment

Name

Mail (will not be published)

Website

XHTML: You can use these tags: <a href="" title=""> <abbr title=""> <acronym title=""> <b> <blockquote cite=""> <pre> <i> <strike>

To leave **code snippets** use the <pre> tag!

Please note: Comment moderation is enabled and may delay your comment. There is no need to resubmit your comment.

About the Site:

- Peteris Krumins' blog about programming, hacking, software reuse, software ideas, computer security, google and technology.
- more about this blog...
- Peter can be reached at:

Subscribe to Posts

- Through a feed:

  Through email:

- Enter your email address:

  Delivered by FeedBurner

Server Sponsors

- I am being sponsored by ZigZap - we are tech! These guys gave me an amazingly great dedicated server. If you are looking for hosting, I recommend ZigZap guys!

Recent Posts

- MIT's Introduction to Algorithms, Lectures 4 and 5: Sorting
- Musical Geek Friday #14: Alice and Bob
- MIT's Introduction to Algorithms, Lecture 3: Divide and Conquer
- MIT's Introduction to Algorithms: Lectures 1 and 2
- Musical Geek Friday #13: Song For Hackers and Crackers
- Traffic Accounting with Linux IPTables
- Python Yesterday, Today and Tomorrow

- Musical Geek Friday #12: Every OS Sucks
- Performance Tuning Best Practices for MySQL
- How Reddit Top and Hacker Top Programs Were Made

Popular Posts

- Learning JavaScript Programming Language through Video Lectures
- The Definitive Guide to Bash Command Line History
- Working Productively in Bash's Vi Command Line Editing Mode (with Cheat Sheet)
- How to Extract Audio Tracks from YouTube Videos
- Downloading YouTube videos with a Perl one-liner
- Designing Digg Picture Website in a Matter of Hours
- Learning Python Programming Language Through Video Lectures
- Sed - UNIX Stream Editor - Cheat Sheet
- Bash Emacs Editing Mode Cheat Sheet
- Screen VT100/ANSI Terminal Emulator Cheat Sheet

Most Downloads

- sed stream editor cheat sheet (.pdf) (13592)
- perl's pack/unpack and printf cheat sheet (.pdf) (12302)
- awk cheat sheet (.pdf) (11919)
- screen cheat sheet (.pdf) (8460)
- awk cheat sheet (.txt) (7213)
- sed stream editor cheat sheet (.txt) (6617)
- bash vi editing mode cheat sheet (.pdf) (6457)
- perl's special variable cheat sheet (.pdf) (6058)
- bash history cheat sheet (.pdf) (6036)
- crypt-o.mp3 (musical geek friday #1) (5791)
- model-view-controller song.mp3 (musical geek friday #4) (5225)
- readline emacs cheat sheet (.pdf) (5097)
- gawk youtube video downloader (5058)
- ed text editor cheat sheet (.pdf) (4581)
- vbscript youtube video downloader (3839)

Recent Comments

- Ralph Corderov: Hi, Are you familiar with the `sort pump'? It's O(n). This Powerpoint file has some...
- wildan: Nice blog.. nice to meet you..
- Timmy Jose: Hi z0ltan!
- Timmy Jose: Nice blogs d00d but I have to agree with Ignasi. Seems to be a very unhealthy lifestyle. Kinda scary too...
- dont ver: excellent post, well researched

Categories:

- Cheat Sheets (10)
- Hacker's Approach (7)
- Howto (6)
- Introduction to Algorithms (3)
- Misc (5)
- Musical Geek Friday (14)
- Programming (25)
- Security (3)
- Tools (4)
- Video Lectures (16)

Archives:

- August 2008 (6)
- July 2008 (14)
- June 2008 (4)
- May 2008 (5)
- April 2008 (8)
- March 2008 (3)
- February 2008 (1)
- January 2008 (1)
- November 2007 (1)
- October 2007 (3)
- September 2007 (5)
- August 2007 (10)
- July 2007 (9)
- All Posts

My other sites

- Free University Video Lectures
- Free Science Videos

My Blog Log

Advertisement

Designed by Peteris Krumins by *reusing* Exquisite WordPress theme.
<img class="statcounter"
src="http://c.statcounter.com/3189873/0/dc7255a7/0/" alt="web site analytic" />
  <img class="statcounter"
src="http://c18.statcounter.com/counter.php?sc_project=1938274&java=0&security=25ba7878&invisible=1" alt="web statistic" />