Lecture Summary

Keywords:

• const

• function

• this

• that

• javascript

Summary:

• Here's a function

declaration: function

getData() { // do

something } A

function can be run any time

you want by invoking it, like

this: getData() A

function can have one or more

argument: function

getData() { //do

something }

function getData(color) {

//do something }

function getData(color, age)

{ //do something

} When we can pass an

argument, we invoke the

function passing

parameters: function

getData(color, age) {

//do something }

getData('green', 24)

getData('black')29 Note

that in the second invocation

I passed the black string

parameter as the color

argument, but no age

• */ callback(result)

//do things const

result = /* }The

main problem with this

approach is that if we need to

use the result ofthis

function in the rest of our

code, all our code must be

nested inside thecallback,

and if we have to do 2-3

callbacks we enter in what is

usuallydefined "callback

hell" with many levels of

functions indented into

otherfunctions:46

doSomething((result) => {

doSomethingElse((anotherResult)

=> {

doSomethingElseAgain((yetAnothe

rResult) => {

console

• 02 ;('something')

true false this

//the current scope

undefined i //where i is

a variable or a constant

Arithmetic expressions are

expressions that take a

variable and an operator

(more on operators soon), and

result into a number:

1 / 2 i++ i -= 2

i * 2 String expressions

are expressions that result

into a string: 'A ' +

'string' Logical

expressions make use of

logical operators and resolve

to a boolean value:

a && b a || b

Links:

• https://www.google.com/search?q

=const

• https://www.google.com/search?q

=function

• https://www.google.com/search?q

=this

• https://www.google.com/search?q

=that

• https://www.google.com/search?q

=javascript