Q) Can JavaScript be used on Server side?

A)

There's the project [Phobos](https://phobos.dev.java.net/), which is a server side Javascript framework.

Back In The Day, the Netscape web server offered server-side javascript as well.

In both of these cases, Javascript is used just like you'd use any language on the server. Typically to handle HTTP requests and generate content.

[Rhino](http://www.mozilla.org/rhino/), which is Mozillas Javascript system for Java, compiles Javascript in to Java byte codes, which the JVM can choose to JIT. Other systems use other means for executing javascript, even to the point that some are JIT compiling their javascript internal codes.

I forsee that there will be more and more Javascript on the server. When you're writing "thick" applications in Javascript on the client, then you may as well be able to write your logic in Javascript on the server in order to not have to make the cognitive leaps from one language to another. The environments will be different, but much of your code and knowledge will be sharable.

Finally, Javascript is probably the singular language that has the most money pointing at it right now in terms of implementations. From Apple, Mozilla, Google, and even Microsoft as well as the efforts to make it an even more advanced language (i.e. basically a Scheme with Algol syntax sans macros).

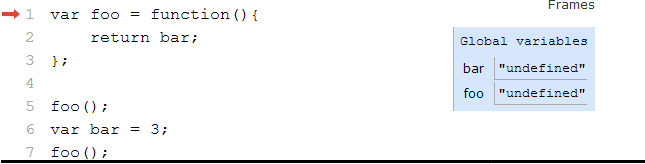
Most of those implementation are buried in the browser, but that's not to say that there's no value on the server side as well.

The tooling is the biggest place where Javascript is lacking, especially on the server side, but if you consider something like Phobos, where you can debug your server side Javascript in the IDE, that's a great advancement.

Personally, I'm tossing Javascript around in my applications like white paint. It offers cheap extensibility for very little cost and is a great enabler.

-----------------------------------------------------------------------------------------------------------------------------------------

Q) How does JavaScript Interpreter know all the variables existing in the code, before it starts execution?



A)

There is something called **hoisting**, where interpreter perform name bindings for all the variables in the code, before the interpreter starts execution. All the default values of variables will be **undefined**.

That's something called [**hoisting**](https://stackoverflow.com/search?q=%5Bjavascript%5D%20hoisting). Before the engine executes any line of code (of a function), it looks for all variable and function declarations and creates a binding in the current environment ([§10.5, steps 5 and 8](http://es5.github.io/#x10.5)). In case of variable declarations, the value is undefined because the assignment has not taken place yet.

Apparently Python doesn't work that way

-----------------------------------------------------------------------------------------------------------------------------

Q) What are objects in JavaScript as per below code?

var foo = {};

foo.bar = function(){

this.ack=3;

};

foo.bar();

A) In JavaScript, when you say

var foo = {};

you are creating an object and when you assign something like

foo.bar = function(){

this.ack=3;

};

it first looks for bar in foo. If it doesn't it, it creates a new attribute and stores function object in it. Here, key is bar and the value is the function object.

The this binding happens very lately. When you invoke ack, it checks the object on which ack was invoked. Since it is on foo object, this will be set to foo. So, you are creating a new attribute ack on foo indirectly after calling foo.bar();.

If you are looking for a similar behaviour in Python, use objects like this

class TempClass(object):

pass

foo = TempClass()

def function():

ack = 3

foo.f = function

--------------------------------------------------------------------------------------------------------------------------------------

Q) What are the differences between \_\_proto\_\_ and prototype properties?

var b = new Foo(20);

var c = new Foo(30);

