Describe an instance of prototypal inheritance in JavaScript?

- Contrary to many object oriented languages, JavaScript is classless and does not support classical inheritance. Because there are no classes, each object is a prototype of another object, and inherits the properties defined in the prototype.
- In the example below, you can see how 'dog' is the prototype for 'beagle', which in turn is the prototype for 'spot'. Any properties defined in dog, such as number of legs, will be inherited by Spot the dog.

```
1  var dog = { legs: 4;}
2  var beagle = Object.create(dog);
3  var spot = Object.create(beagle);
```

What are closures and how are they used?

- This is a mouth full to try and explain, and you'll probably need to whiteboard an example if this question comes up. The simplest definition I could find comes from Jibbering.com blog:
    - "A "closure" is an expression (typically a function) that can have free variables together with an environment that binds those variables (that "closes" the expression)."
- If you're like most people and that definition means nothing to you, MDN has an excellent section on closures that is chalk full of examples. Even if you've never recognized them as closures, you've probably seen or even used these patterns before.

Can you use x === "object" to test if x is an object?

- In short, yes, but you must take into account the fact that null is considered an object in JavaScript. Even if x is null, 'console.log(typeof x === "object")' will log true instead of false.

- To account for this, you must also test whether or not x is null by including the following:
    - console.log((x !== null) && (typeof x === "object"));

What happens when you don't declare a variable in Javascript?

- If you don’t explicitly declare a variable, you risk creating an implied global variable. This is a problem, as you will be unable to create multiple instances of that object, as each new instance will overwrite the data from the last.
- In general, global variables should be used only in very specific situations and are typically not recommended, as they can lead to a lot of odd side effects that may be difficult to track down.

What is the difference between == and === ?

- ‘==’ evaluates equality of the value, while ‘===’ evaluates equality of type and value.

What datatypes are supported in Javascript?

- Undefined
- Number
- String
- Boolean
- Object
- Function
- Null

What would “1”+2+3 and 1+2+“3” evaluate to, respectively?

- When the first character is a string, the remaining characters will be converted into a single string, rendering 123.
- If the string is at the end, the initial characters will perform normal mathematical functions before converting into a string, resulting in 33.

What is the difference between window.onload and the jQuery $(document).ready() method?

- The window.onload method occurs after all the page elements have loaded(HTML, CSS, images), which can result in a delay.
- The $(document).ready() method begins to run code as soon as the Document Object Model (DOM) is loaded, which should be faster and less prone to loading errors across different browsers.

Explain the concept of unobtrusive Javascript?

- Unobtrusive JavaScript is basically a JavaScript methodology that seeks to overcome browser inconsistencies by separating page functionality from structure. The basic premise of unobtrusive JavaScript is that page functionality should be maintained even when JavaScript is unavailable on a user's browser.

What is the difference between a null value and an undefined value?

- Null is used to assign an empty value to a variable and needs to be assigned manually.
- Undefined values result when you declare a variable without assigning it a value. Undefined will be the default whenever you don't explicitly assign a value.

Which conditional statements will JavaScript support?

- if statement
- if – else statement
- if – else if – else statement
- switch statement

What is NaN?

- Nan is literally "Not-a-Number". NaN usually results when either the result or one of the values in an operation is non-numeric. Even though NaN is not a number, 'console.log(typeof NaN === "number");' logs true, while NaN compared to anything else (including NaN) logs false. The only real way to test if a value is equal to NaN is with the function 'isNaN()'.

What types of pop-up boxes you can create in JavaScript and how can you create them?

- Alert, prompt, and confirm boxes are the three main pop-up boxes that can be created in JavaScript.
- Alert boxes are used to alert the user to important info regarding the site and require the user to click 'OK' in order to proceed.
- Alert boxes are created using the following syntax:
    - window.alert("Text");
- Prompt boxes prompt the user to input some form of data before proceeding.

- Prompt boxes are created using the following syntax:
    - window.prompt(“Prompt Text”,”Default value”);
- Confirm boxes are used when for verification, such as agreeing to a terms of service, rendering a TRUE value when the user clicks ‘OK’.
- Confirm boxes are created with the following syntax:
    - window.confirm(“Confirm text here.”);

Which boolean operators are in JavaScript?

- ‘and’ operator: &&
- ‘or’ operator: ||
- ‘not’ operator: !

Why would you include ‘use strict’ at the beginning of a JavaScript source file?

- Using strict mode enforces stricter error handling when running your code. It essentially raises errors that would have otherwise failed silently. Using strict mode can help you avoid simple mistakes like creating accidental globals, undefined values, or duplicate property names. This is typically a good idea when writing JavaScript, as such errors can create a lot of frustrating side effects and be difficult to track down.

Explain the meaning of the keyword ‘this’ in JavaScript functions

- The keyword ‘this’ in JavaScript refers to the object that a function is a method of. If it’s not specified, it will default to the global object, the window.
- In the example below, you can see that ‘this’ refers to the box object when it is applied to the width function. When no object is passed in, it will default to the browser window.

```
var box = { outerWidth: 50 };

function width() {
    alert(this.outerWidth);
};

width.apply(box);// Returns 50, the outerWidth of the box object

width();// Returns the outerWidth of the browser window
```

What does a timer do and how would you implement one?

- Setting timers allows you to execute your code at predefined times or intervals.
- This can be achieved through two main methods: setInterval(); and setTimeout();
- setInterval() accepts a function and a specified number of milliseconds.
    - ex) setInterval(function(){alert("Hello, World!"),10000) will alert the "Hello, World!" function every 10 seconds.
- setTimeout() also accepts a function, followed by milliseconds. setTimeout() will only execute the function once after the specified amount of time, and will not reoccur in intervals.

Is there automatic type conversion in JavaScript?

- Yes, JavaScript supports automatic type conversion. In the example below, JavaScript is expecting a string. When it receives a fixnum as an operand, it's automatically converted to a string.
    - ex)
        - 5 + " cats" = "5 cats"

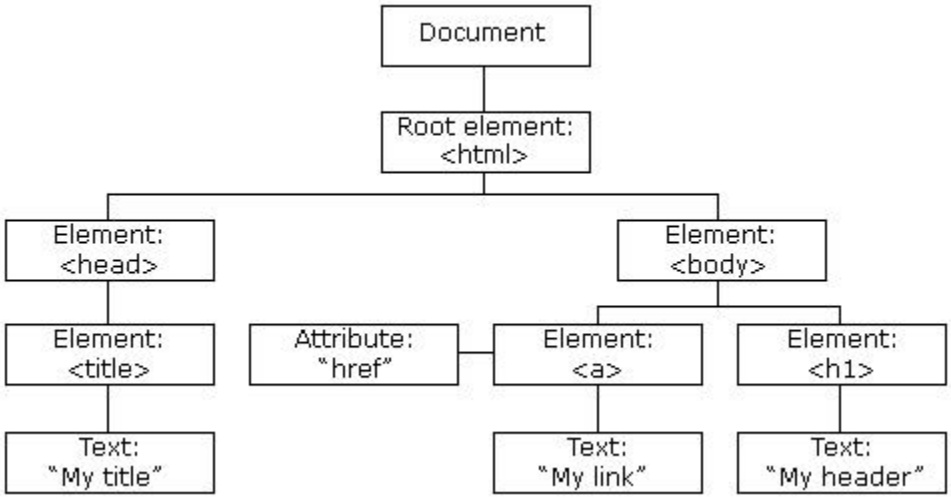
Explain the HTML DOM and its relevance to JavaScript

- The Document Object Model allows JavaScript to obtain information about a page and manipulate it by adding or removing HTML elements, or by altering their attributes. The DOM can be viewed as a programming interface for HTML, whereby you can access and alter its structure, attributes, styles, and events. According to Mozilla Developer Network (MDN), "The DOM provides a representation of the document as a structured group of nodes and objects that

have properties and methods. Essentially, it connects web pages to scripts or programming languages."

- Using the below example from MDN, the code would return all <p> elements within the HTML document.

```
paragraphs = documtent.getElementsByTagName("P");
//paragraphs[0] is the first <p> element
//paragraphs[1] is the second <p> element, etc.
alert(paragraphs[0].nodeName);
```



A DOM tree structure, from MDN

What are your favorite JavaScript libraries?

- This is one of those general open-ended questions interviewers love to throw out. The only correct answer here is one that shows you're familiar with and have an opinion on the various JavaScript libraries. You could discuss the pros and cons of using raw JavaScript vs. jQuery, or describe an instance in which you'd use one library over another. Visit jsdb.io for a complete rundown.