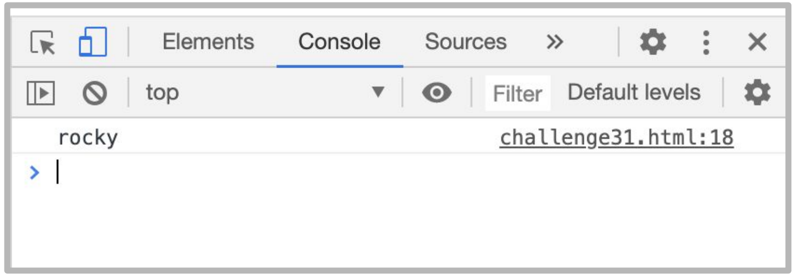
JavaScript Object Challenges

Challenge 1

Make a new HTML file and in that file create an object for a pet with two properties in it, one for the name of the pet and one for the species of the pet.

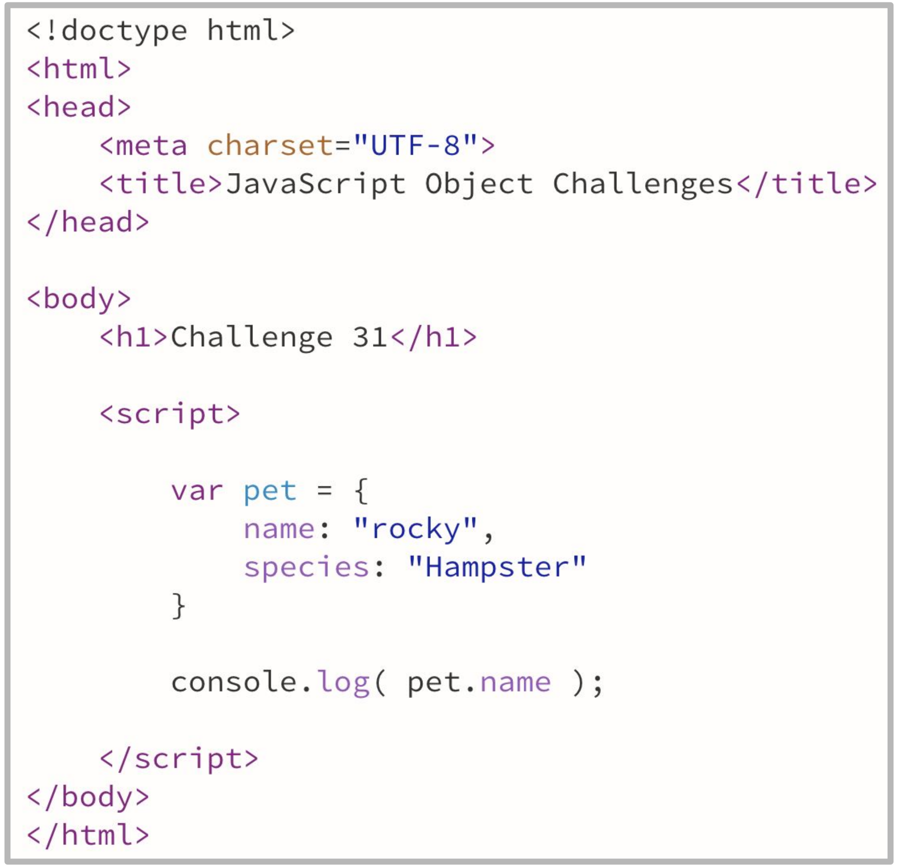
Give these two properties values and then use the console.log() method to print out the name of the pet.



```
rocky                                   challenge31.html:18
>
```

Challenge 1 - Answer

Hopefully you got something that looks kind of like this...

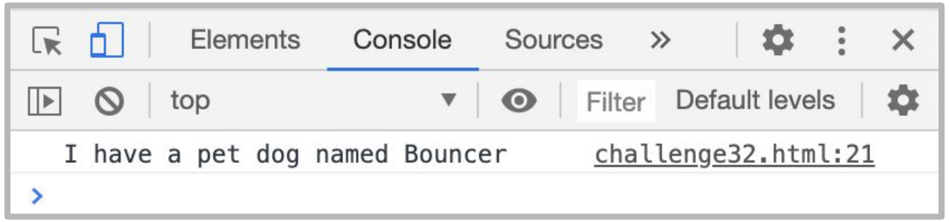


```
<!doctype html>
<html>
<head>
    <meta charset="UTF-8">
    <title>JavaScript Object Challenges</title>
</head>

<body>
    <h1>Challenge 31</h1>

    <script>

        var pet = {
            name: "rocky",
            species: "Hampster"
        }

        console.log( pet.name );

    </script>
</body>
</html>
```

Challenge 2

Imagine you traded this pet for a different one. Change the name and the species in the object to a different name and a different species.

Then console log out “I have a pet **X** named **Y**” where the X and Y are the species and name of the pet.



```
I have a pet dog named Bouncer        challenge32.html:21
```

Challenge 2 - Answer

Hopefully, you got something like this:

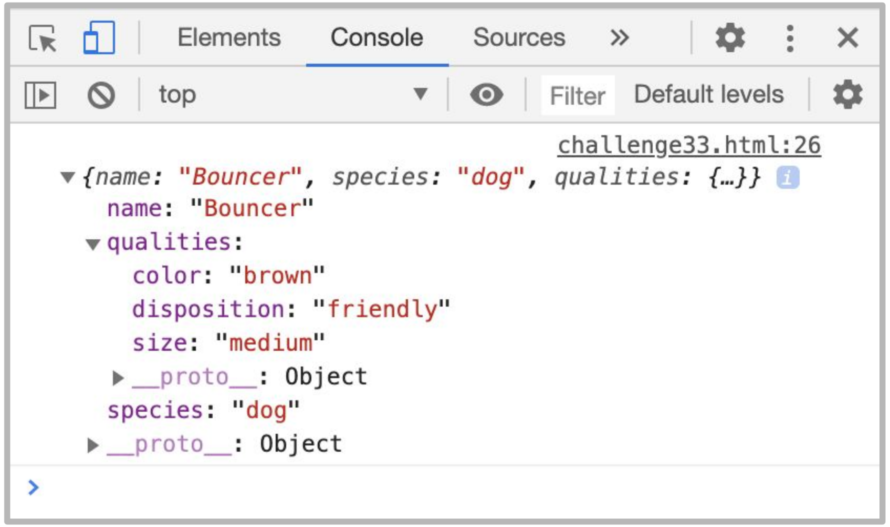
```
<script>

    var pet = {
        name: "Rocky",
        species: "hampster"
    }

    pet.name = "Bouncer";
    pet.species = "dog";

    console.log( `I have a pet ${pet.species} named ${pet.name}` );

</script>
```

Challenge 3

Add a new property to the pet object called “qualities”. This property will be an object that has three properties with corresponding values:

For example:
color, disposition, and size

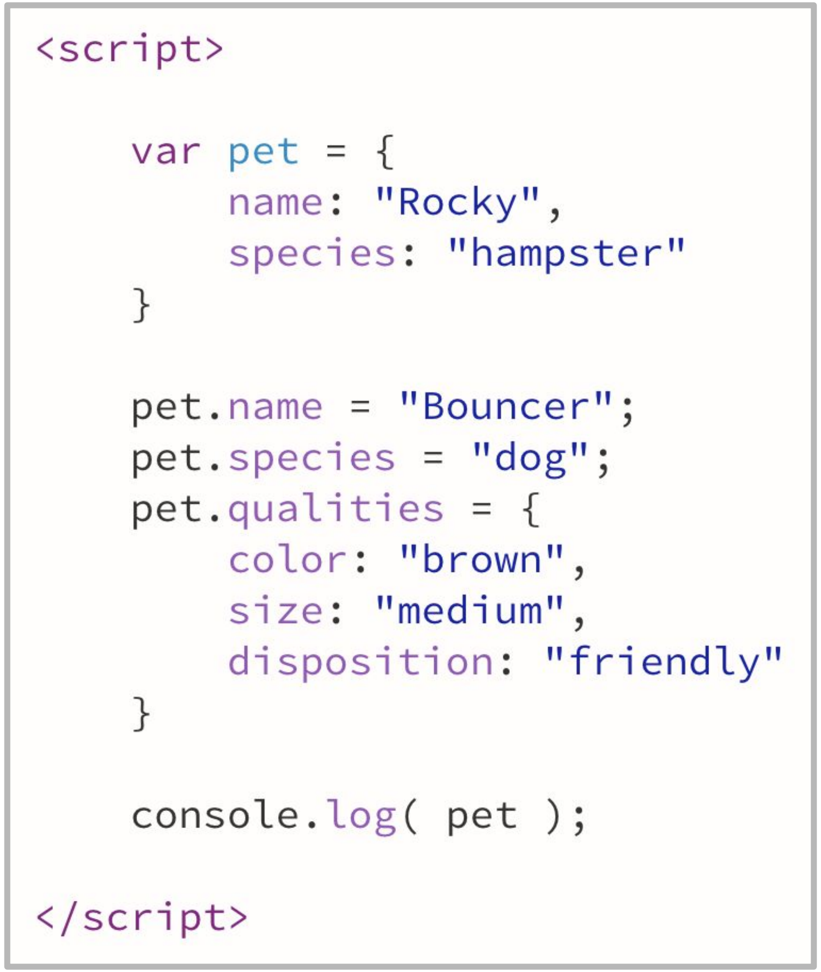
Console log out the whole object, so you can see the whole thing. Example on the right.



```
                                          challenge33.html:26
▼{name: "Bouncer", species: "dog", qualities: {…}}
   name: "Bouncer"
  ▼qualities:
     color: "brown"
     disposition: "friendly"
     size: "medium"
   ▶__proto__: Object
   species: "dog"
  ▶__proto__: Object
>
```

Challenge 3 - Answer

Did you get this?



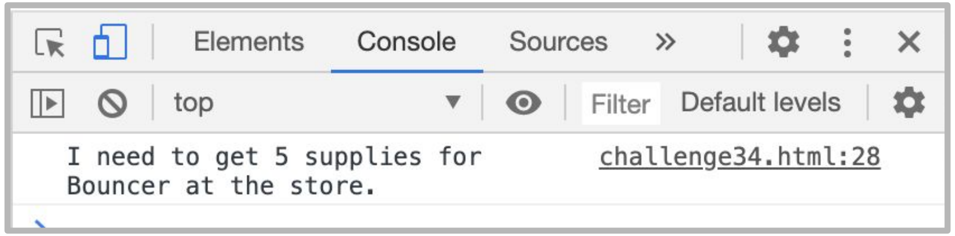
```
<script>

    var pet = {
        name: "Rocky",
        species: "hampster"
    }

    pet.name = "Bouncer";
    pet.species = "dog";
    pet.qualities = {
        color: "brown",
        size: "medium",
        disposition: "friendly"
    }

    console.log( pet );

</script>
```

Challenge 4

Add an additional property into the qualities properties, which is a list of supplies (an array) you need for your pet.

Then console log out a statement that says "I need to get **X** amount of supplies for **Y** at the store" Where **X** is the number of supplies in the list and **Y** is the name of the pet.

Example result on the right.



```
I need to get 5 supplies for        challenge34.html:28
Bouncer at the store.
```

Challenge 4 Answer

Starting with the code for Challenge 3, you should have these two lines added at the bottom of the script.

```
pet.qualities.supplies = ["dog food", "dog bed", "treats", "collar", "leash"];

console.log( `I need to get ${pet.qualities.supplies.length} supplies for ${pet.name} at the store.` );
```

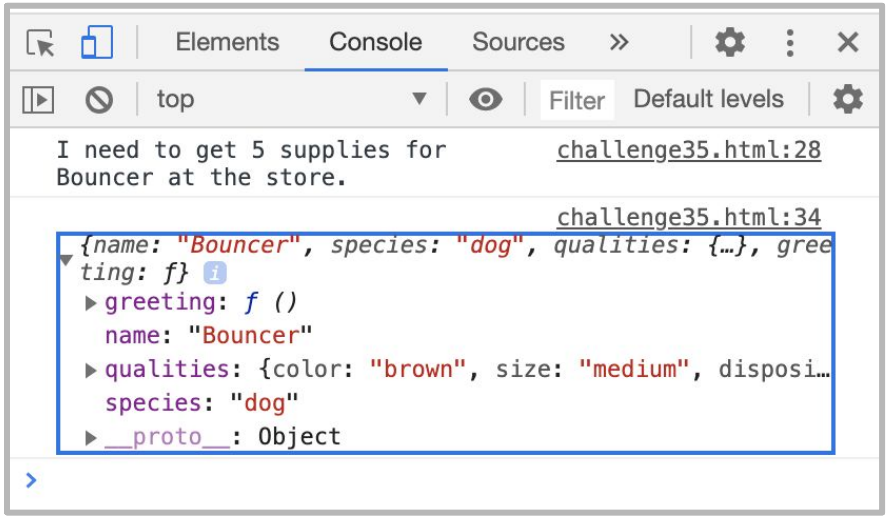
Challenge 5

Add an additional property to the pet object called “greeting” that contains a function as its value.

The function should alert the greeting that the pet says or does when you arrive home.

Remember that when an object has a function, it is called a method.

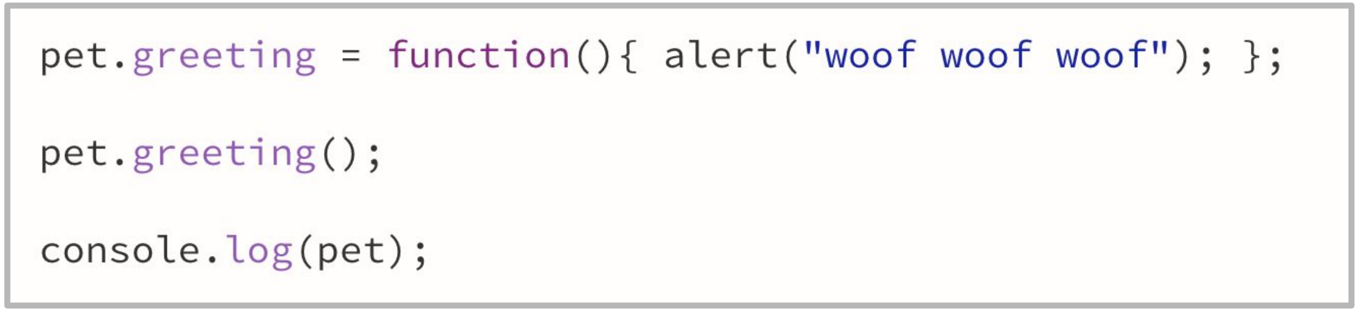
Make a call to the pet greeting method and also console log the whole pet object so you can see everything you have added to it.



```
I need to get 5 supplies for                challenge35.html:28
Bouncer at the store.
                                            challenge35.html:34
▼{name: "Bouncer", species: "dog", qualities: {…}, gree
 ting: f}
  ▶greeting: f ()
   name: "Bouncer"
  ▶qualities: {color: "brown", size: "medium", disposi…
   species: "dog"
  ▶__proto__: Object
>
```

Challenge 5 Answer

In addition to the challenge 4 script, you should have something that looks like the code below.



```
pet.greeting = function(){ alert("woof woof woof"); };

pet.greeting();

console.log(pet);
```

Challenge 6

Starting with the challenge 5 code, comment out or remove all the script so far. Instead add a constructor function for pets that creates an object with the following properties:

name, species and greeting.

Make the greeting property contain a function.

Then make two variables for two different pets.

Then run the greeting method for each pet.

Challenge 6 - Answer

Did you get something like this?

```
function pet( name, species, greeting){
    this.name = name;
    this.species = species;
    this.greeting = function(){ alert(greeting)};
}

var myDog = new pet("Bouncer", "dog", "woof woof woof");
var myCat = new pet("Mouser", "cat", "meow!");

myDog.greeting();
myCat.greeting();
```

Challenge 7

```html
<form id="addpet">
      <p><label>Name <input type="text" id="name"></label></p>
      <p><label>Species <input type="text" id="species"></label></p>
      <p><label>Greeting <input type="text" id="greeting"></label></p>
      <p><input type="submit" value="add pet"></p>
</form>
```

For this challenge, add the form above to the HTML and then add an event handler that will listen for the submission of the form.

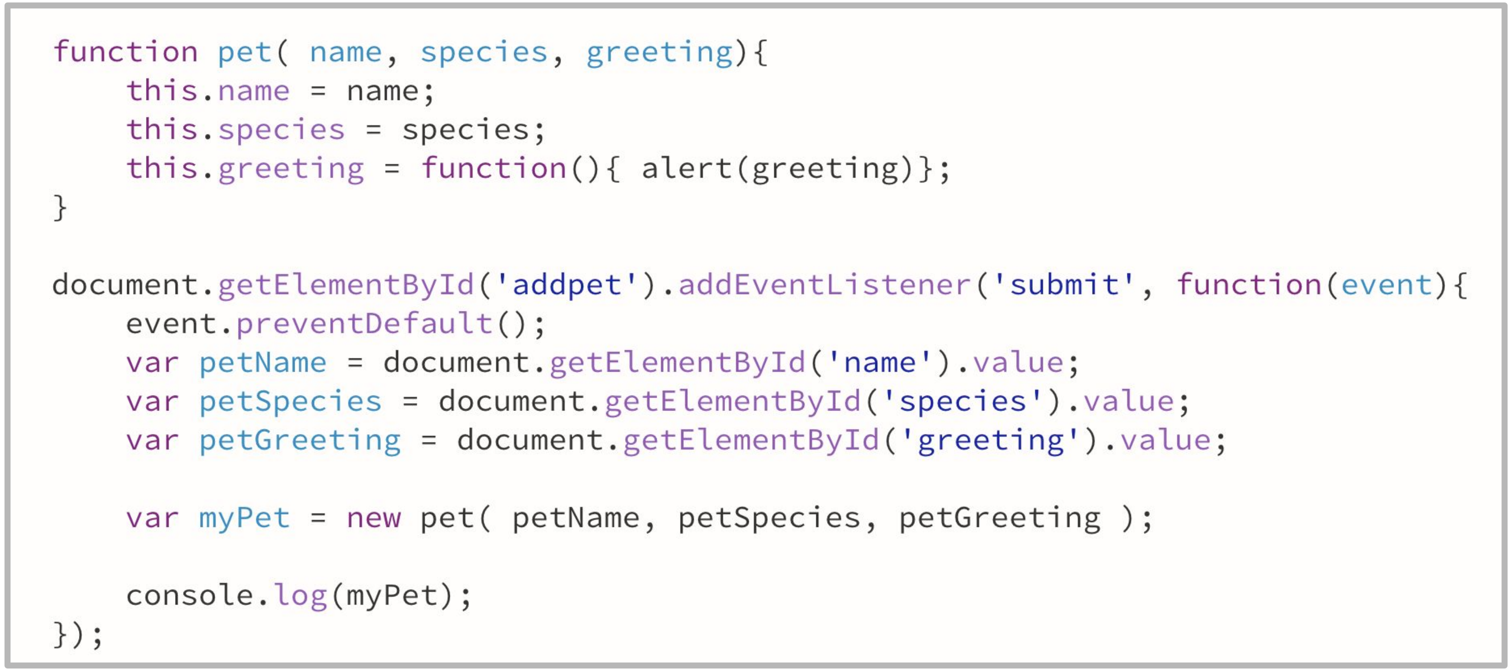
Take the data from the form submission and create a new pet object and give it the values from the form data as values.

Add a call to the console.log() method within the event handler function to print the object to the console.

Don’t forget to pass in the event object and prevent the default behavior of the form submission.

Challenge 7 - Answer

This is one possible answer:



```
function pet( name, species, greeting){
    this.name = name;
    this.species = species;
    this.greeting = function(){ alert(greeting)};
}

document.getElementById('addpet').addEventListener('submit', function(event){
    event.preventDefault();
    var petName = document.getElementById('name').value;
    var petSpecies = document.getElementById('species').value;
    var petGreeting = document.getElementById('greeting').value;

    var myPet = new pet( petName, petSpecies, petGreeting );

    console.log(myPet);
});
```

Challenge 8

The answer to challenge 7 could be a little more elegant. There is a property on HTML forms called “elements”. You can use this property to get all the elements of a form when it is submitted all at once like this:

```
event.target.elements
```

If you assign that to a variable, that variable will contain an array of all of those elements. Like this:

```
var formData = event.target.elements;
```

For this challenge, can you update the function to use this method to get the form data, instead of using getElementById for each field.

Note that formData will be an array of form elements. You will still need to get the values out of each field.

Challenge 8 - Answer

Did you get something like this?

```
document.getElementById('addpet').addEventListener('submit', function(event){
    event.preventDefault();
    var formData = event.target.elements;
    var myPet = new pet( formData[0].value, formData[1].value, formData[2].value );

    console.log(myPet);
    myPet.greeting();
});
```

Challenge 9

For this challenge, add a little basic form validation to the function. If any of the fields are empty, have an alert pop up that says, “Fill in the Fields”.

If all the fields are filled in, create the object and call the greeting method.

A few hints…

1. Change the formData variable to using the querySelectorAll method to get all the input tags.
2. Use a variable set to zero, and use the forEach method to loop through all the values to see if any are empty. If they are, increment the variable
3. If the variable at the end is any higher than zero, you know the form was not completely filled out.

Challenge 9 - Answer

One possible answer for this challenge

```
document.getElementById('addpet').addEventListener('submit', function(event){
    event.preventDefault();
    var formData = document.querySelectorAll('input');
    var errors = 0;

    formData.forEach( function(eachField){
        if(eachField.value == ""){ errors++; }
    } );

    if( errors === 0 ){
        var myPet = new pet( formData[0].value, formData[1].value, formData[2].value );
        myPet.greeting();
    }
    else {
        alert("fill in the fields!");
    }
});
```

Challenge 10

Add an empty div after the form on the challenge 9 file.

Then make it so that when the user fills out the form, the div in the HTML gets filled with two paragraphs. The first says “I have a pet X named Y”.

The second paragraph contains a button that says “click for a greeting from Y”.

When the user clicks the button, the greeting method on the pet object runs.

Hint: innerHTML is your friend.

Hint: Make a function and pass in the myPet variable as data to work with within that function.

Challenge 10 - Answer

You might have added something like this to the script you already had...

```
if( errors === 0 ){
    var myPet = new pet( formData[0].value, formData[1].value, formData[2].value );
    addToPetsDiv(myPet);
}
```

And added a function like this...

```
var petsDiv = document.querySelector('div');

function addToPetsDiv(thisPet){
    var newPet = `<p>I have a pet ${thisPet.species} named ${thisPet.name}</p>`;
    newPet += `<p><button id="sayhi"> Click for a greeting from ${thisPet.name}</button></p>`;
    petsDiv.innerHTML = newPet;
    document.getElementById('sayhi').addEventListener('click', thisPet.greeting);
}
```