JSON and Objects

JavaScript

Objects in Programming

Objects are often instantiated from a class (an object blueprint) which we can get more into later.

For now we will go over a basic object that can exist on its own to get familiar with the most essential pieces of an object.

In JavaScript, objects typically possess information (in the form of “properties”) and executable code blocks (in the form of “methods.”)

- Properties are much like variables, but inside an object!
- Methods are much like functions, but inside an object!

Objects in JavaScript

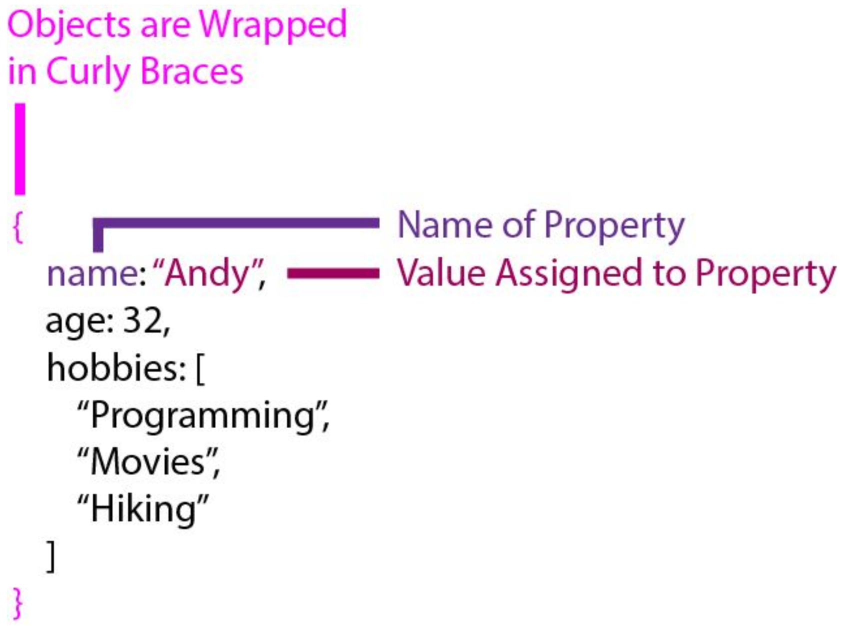
When declaring an object, we use curly braces to contain its data.

Data is broken up into property names and their values separated by commas.

The pattern you'll see with a JavaScript object is…

property: "value"

Properties or methods are followed by a colon and their value.


Objects are Wrapped in Curly Braces
Name of Property
Value Assigned to Property

```
{
  name: "Andy",
  age: 32,
  hobbies: [
    "Programming",
    "Movies",
    "Hiking"
  ]
}
```

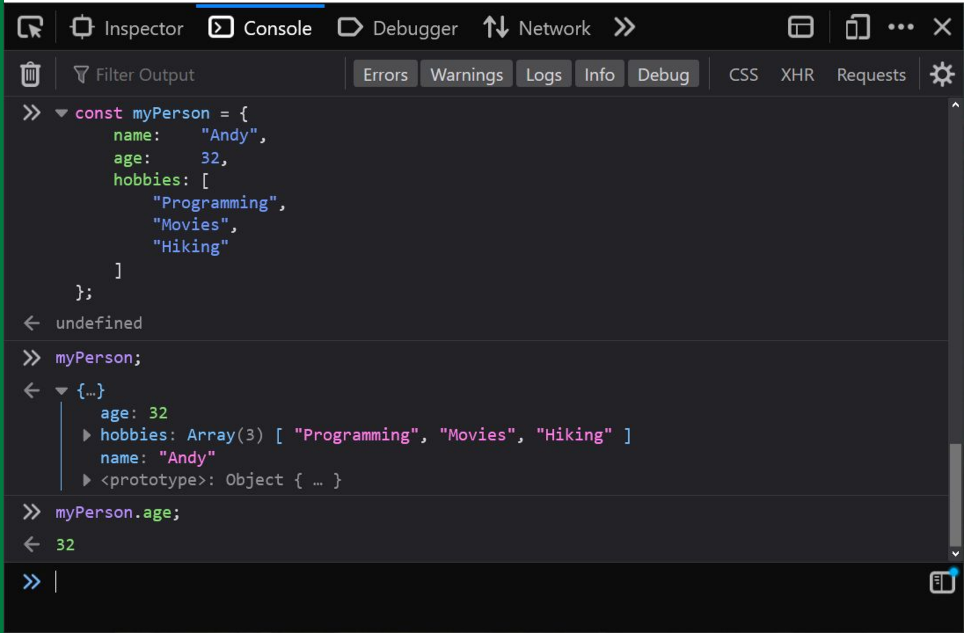
Creating and Accessing an Object

Using the syntax we saw in the previous slide, we can create objects and easily access their labeled information (properties.)

We can assign variables an object as a value, as you may have expected.

We can access individual properties by proceeding the name of the object with a period followed by the name of the property.

See the example below where we create an object, and access the "age" property!



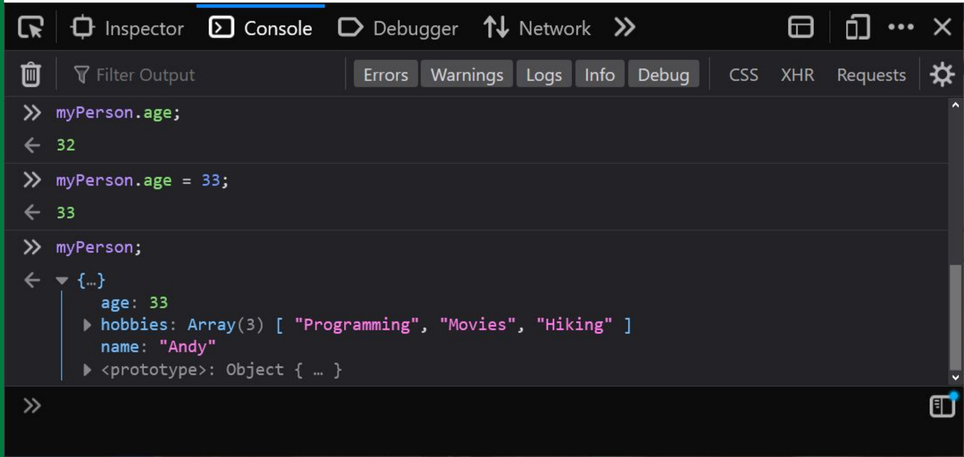
```
>> const myPerson = {
       name:    "Andy",
       age:     32,
       hobbies: [
           "Programming",
           "Movies",
           "Hiking"
       ]
   };
<- undefined
>> myPerson;
<- {…}
     age: 32
     hobbies: Array(3) [ "Programming", "Movies", "Hiking" ]
     name: "Andy"
     <prototype>: Object { … }
>> myPerson.age;
<- 32
>>
```

Updating a Property

When accessing a property of a simple object like the one we've made, you are able to update its value via a reassignment.

You can assign properties for simple objects like this much like we did with variables. Simply access the property—follow up with the assignment operator and a value.

See below for a quick example!



```
>> myPerson.age;
← 32
>> myPerson.age = 33;
← 33
>> myPerson;
← ▼ {…}
      age: 33
    ▶ hobbies: Array(3) [ "Programming", "Movies", "Hiking" ]
      name: "Andy"
    ▶ <prototype>: Object { … }
>>
```

JSON

JSON stands for:
JavaScript Object Notation.

JSON is one of the most common formats for transmitting data between programs.

As APIs on the web continue to grow in popularity and front-end frameworks continue to encourage live page updates to content, easy to use and read formats like JSON become—not only more popular, but—important.

JSON is very similar to what we saw with regular JavaScript objects, but this standard is a bit more restrictive.

JSON Syntax

JSON helps us serialize objects based on JavaScript syntax, but it isn't exactly the same.

Important differences between JSON and JavaScript object syntax:

- All property names in JSON must be in double quotations.
- You may not include trailing commas in JSON.
- JSON does not support…
  - NaN
  - Infinity
  - Leading zeros in numbers

JSON data-types include:

- Strings
- Numbers
- Arrays
- Objects
- Booleans (true, false)
- Null (null)

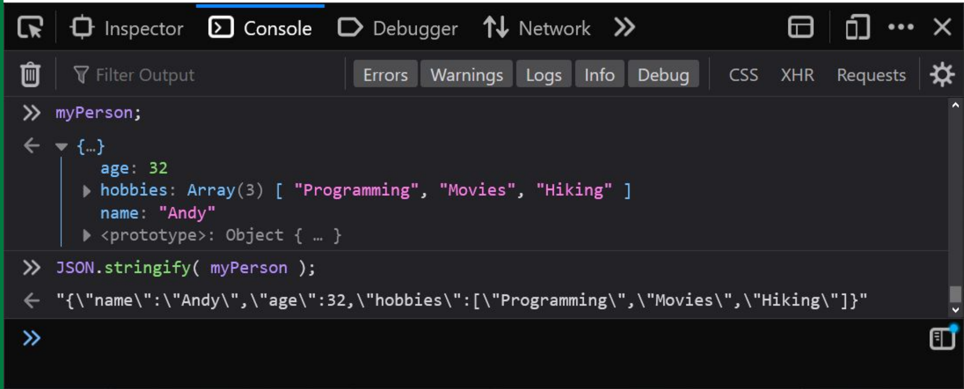
JavaScript to JSON

How do we convert the JavaScript object we created earlier into a JSON object?

The fastest and easiest way, if you already have a JavaScript object, is to utilize JavaScript's JSON class.

Passing a JavaScript object into the JSON.stringify method will return to you a converted valid JSON string representation of that object.

Backslashes are escape sequence characters, allowing for double quotes inside of double quotes without exiting or terminating a string.



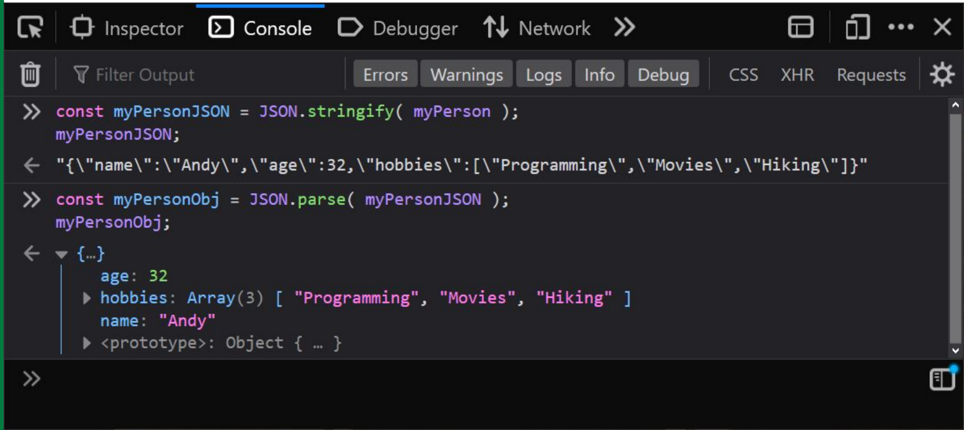
```
>> myPerson;
← ▼ {…}
      age: 32
    ▶ hobbies: Array(3) [ "Programming", "Movies", "Hiking" ]
      name: "Andy"
    ▶ <prototype>: Object { … }
>> JSON.stringify( myPerson );
← "{\"name\":\"Andy\",\"age\":32,\"hobbies\":[\"Programming\",\"Movies\",\"Hiking\"]}"
>>
```

JSON to JavaScript

How do we convert a JSON string into a valid JavaScript object?

The fastest and easiest way, if you already have a JSON string, is to utilize JavaScript's JSON class.

Passing a JSON string into the JSON.parse method will return to you a valid JavaScript object representation of the provided string.



```
>> const myPersonJSON = JSON.stringify( myPerson );
   myPersonJSON;
<- "{\"name\":\"Andy\",\"age\":32,\"hobbies\":[\"Programming\",\"Movies\",\"Hiking\"]}"
>> const myPersonObj = JSON.parse( myPersonJSON );
   myPersonObj;
<- ▼ {…}
       age: 32
     ▶ hobbies: Array(3) [ "Programming", "Movies", "Hiking" ]
       name: "Andy"
     ▶ <prototype>: Object { … }
>>
```