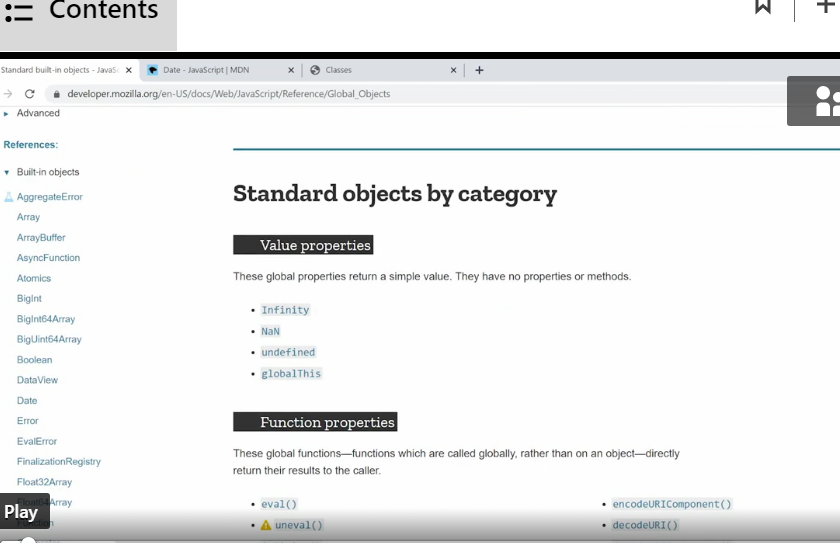
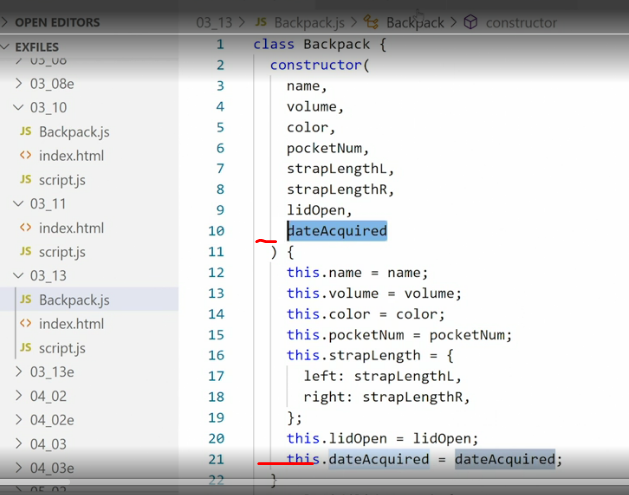
* - [Instructor] In addition to the objects you build yourself, either directly or through a class or a constructor, the browser has a long list of default objects you can use for a variety of different purposes .
* A full list of all of these global objects it's available and extensively documented in the MDN Web Docs .



* These global objects work exactly like the object we've seen so far except, we don't need to define them, they're just available through the browser, and for the most part we interact with them through their methods .
* Let me give you an example of how these global objects are used, with the help of the date object that you see here .



* I want to extend my Backpack class an object type with a new property called dateAcquired, so I know when I got the backpack, and I can do some math to figure out how old that backpack is .
* Adding the new property is straightforward now that we know how the object and the class works .
* We first add a new parameter to the class constructor, then we set up a new property, this .
* dateAcquired and populated with that parameter with the value, and then finally we go through the script and just add in the new value, and here you see I'm passing human readable string, so this is not a number this is just texts that I'm passing into that value .
* Now that we've added the property we can output it in the browser using the same method we've been using so far, so down here I'm console logging out Date acquired, everydayPackpack .

Graphical user interface, text, application

Description automatically generated

* dateAcquired so that is the property in question .

Graphical user interface, text, application, email

Description automatically generated

* If I go back to my browser, you'll see down here we have the Date acquired, and this is just that string .
* Alright now I want to convert this text string into something meaningful, so that I can compare that date with the current date and figure out how many days have gone between them, and figuring out the age of my backpack .
* To do that, I'll seek the help of the **date object .**
* The date object can take pretty much any string that describes a date and time and convert it into something meaningful .
* What it does **is it converts it into the milliseconds** that have passed since the 1st of January, 1970 UTC, and then it applies methods on top of that to convert that huge number of milliseconds into something meaningful for us humans, that's done using one of the myriad of instant methods that come with the date object, and as you can see you can do so much with a date object you can take any time or any date and turn it into milliseconds, or minutes, or months, or the current date, or just a year there's a ton of stuff we can do here, let me just give you a brief preview of this, if we go into the console and I say I wonder what date and time it is right now where I am .
* I can set up a new constant, call it rightNow, and set it equal to new Date .
* Now you'll notice I am using a constructor here, just like we've done in the previous movies, only this time using a constructor that already exists in the browser .

Graphical user interface, text, application

Description automatically generated

* Now I can output rightNow, and what I get is the current date and time, so October six, 2020 at 2:30, in the Pacific Daylight Time Zone, and it's Tuesday .
* Cool, but I want to convert this into a shorter string I don't need all of that information, for that I can use a method, so in this case am going to use a method called toDateString, so if I search here toDate, you'll see here, Date prototype toDateString, returns the date portion of the date as a human readable string, Thursday April 12, 2018 .

Text

Description automatically generated with medium confidence

* Cool I want to use this, now just looking at this example I can see this as an object prototype so that's what we've been creating and then we used a method on it, so let's try to see if I can do exactly that I'll say rightNow .

Graphical user interface, text

Description automatically generated

* toDateString yes and I want to run this as a function so I add the parentheses at the end, and then we get Tuesday October six, 2020 .
* Okay so the method works on the date object, that means I can now use the date object to do some pretty magical things I can take any date and convert it to anything else, which is exactly what I want to do .
* If we scroll a little further down in the Backpack class, you'll see I've added a new method here, it's called backpackAge, this new method use of the date object and a couple of different ways to do the math we need to figure out how many days it spans since I first got the backpack .

Graphical user interface, text, application, email

Description automatically generated

* So first I set up a new let so a let is just a different type of variable, I set a new let called now and set it to a new Date, so that's what we just did in the browser, that gives us the current date and time right now .
* Then I set up another let called acquired, and here I use new Date again, only this time I'm passing in the dateAcquired string, so I'm taking the string in the property and putting it inside the new Date, what will happen now is the date object will give me that same output except the output will be for the date passed to it that string of texts, which in our case is December 5th, 2018 at three o'clock PST .

Text, website

Description automatically generated with medium confidence

* Now I have two new objects and these are milliseconds, and these represent the milliseconds since, the 1st of January, 1970 UTC for these two different dates, the current date and the date originally acquired the back .
* These are now just single large numbers and I can do regular math on them so in the let elapsed, I say take the now Date and subtract the acquired date so that gives me the elapsed time in milliseconds .
* Then finally I set up one last let, called daysSinceAcquired, here I used a math floor method to do some math on it to just take away all the decimals .
* And I say take elapsed value which are all those milliseconds, and then first divided it by 1000 to turn it into seconds, then divided by 3,600 to turn it into hours, and then divided by 24 which is the number of hours in a day, so this will return to us the number of days since I first got this backpack, alright? Then I return the days since acquired values, so that would be the number of days, to wherever the backpackAge method was called .
* Then I need to use that method and I do that just like I've used every other method so far, console .
* log, Days since acquired, I called everydayPack object, and then I called for this new method called backpackAge .

Graphical user interface, text

Description automatically generated

* I've wanted to run it as a method so I'll put parentheses at the end, save this, go back in the browser, and now I get this up, Days since acquired 670, so I've taken a string of text, use the date object to turn it into a meaningful date, then done some math on that date to figure out how many days ago that was, and return that back so that I can see it in the browser .

Graphical user interface

Description automatically generated with medium confidence

* This is a simple example of how powerful these built-in objects are, and like I said, there are a ton of these objects and they do a ton of different things .
* So whenever you want to do something, and you think it's something maybe other people also want to do there's a good chance, there's an object for it in the standard built-in objects that you can use to do math, to convert dates into other types of dates, and do a whole bunch of other things .
* So take your time and read through this document, to figure out what's available, and what you can use it for .