Working With Dates

Dates and times are very important in any programming language. JavaScript has a `Date` object that represents a point in time and let's you do some basic operations on it.

We can instantiate a `Date` object with the `new` keyword.

```
let d;
d = new Date(); // Fri Jul 22 2022 08:35:10 GMT-0400
```

If do not include any arguments, the Date object will be set to the current date and time and you will also get the timezone information. Sometimes you will need to specify the timezone because it can cause some weird issues.

Converting to a string

If we look at the type of the variable, it will show **object**. You can always convert it to a string using the `toString()` method.

```
console.log(typeof d); // object

d.toString(); // "Fri Jul 22 2022 08:35:10 GMT-0400"

console.log(typeof d); // string
```

Specific dates

If you want a specific date and time, you can pass in the year, month, day, hour, minute, second, and millisecond.

One caveat is that the month is 0-indexed, so January is 0 and December is 11.

```
d = new Date(2022, 6, 10); // Fri Jul 10 2022 00:00:00 GMT-0400
```

If you want to add a time, you can. Let's make it 12:30:00.

```
d = new Date(2022, 6, 10, 12, 30, 0); // Fri Jul 10 2022 12:30:00 GMT-0400
```

You can also pass in different date/time strings. You would use the actual month number here.

```
d = new Date('2022-07-10T12:30:00'); // July 10, 2022 12:30:00
d = new Date('07/10/2022 12:30:00'); // July 10, 2022 12:30:00
d = new Date('2022-07-10 12:30:00'); // July 10, 2022 12:30:00
```

You can pass a date without time as well.

Note that if you use the **YYYY-MM-DD** format with hyphens, it may be one day off due to timezones.

```
d = new Date('2022-07-10'); // May be July 09, 2022
```

If you use the **MM-DD-YYYY** format, it should not have this issue.

```
d = new Date('07-10-2022'); // July 10, 2022

// Backslashes will also work
d = new Date('07/10/2022'); // July 10, 2022
```

Timestamps

The `UNIX timestamp` is a system for describing a point in time. It is an integer that represents the number of seconds elapsed since January 1 1970 (An arbitrary date).

The timestamp in JavaScript is expressed in milliseconds.

You can get the current timestamp using:

```
Date.now(); // 1658497991311
```

To get the timestamp of a specific date, you can use the `getTime()` or the `valueOf` method.

```
d = new Date('07/22/2022'); // July 10, 2022
d.getTime(); // 1658497991311
d.valueOf(); // 1658497991311
```

You can also create date objects from a timestamp.

```
d = new Date(1658497991311); // Fri Jul 22 2022 08:35:10 GMT-0400
```

The JavaScript timestamp is expressed in milliseconds. To convert it to seconds:

```
Math.floor(Date.now() / 1000); // 1658498058
```