Node.js

Lesson 07—Working with Events and Buffers

Lesson Overview

In this lesson, you will be able to understand about custom events and buffers.

Learning Objectives

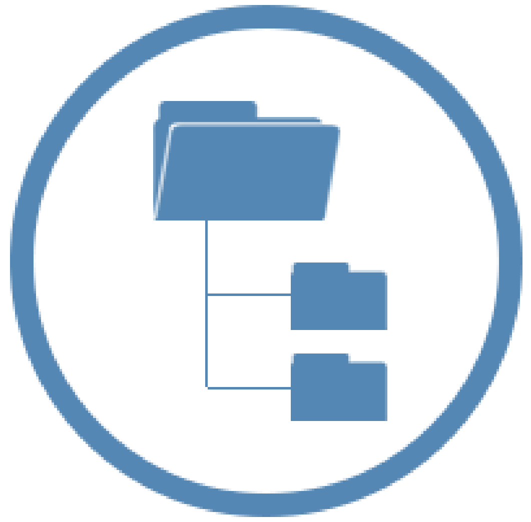
- ✓ Working with Custom Events
- ✓ Working with Buffer in Node.js

Working with Events and Buffers

Topic 1—Working with Custom Events

Node.js Event System

INTRODUCTION



- Node.js is very fast because it was developed around events.
- Instead of loading all the resources with each request, node.js starts the server, initiates most of the data variables and functions, and waits for an event to execute.
- Node.js comes with preloaded useful events (i.e., requests). It would be useful to develop custom events that can be triggered when needed.
- *Node.js events* behave like observer design patterns, whereas *user events* behave like subjects. All functions connected to events act as observers.

Node.js Event System

EVENT EMITTER INTRODUCTION

EventEmitter allows developers to listen for custom "events" and associate actions to execute when events occur.

JavaScript developers will know about standard mouse and keyboard events that occur when users interact.

EventEmitter behaves similarly, except that we can call events when we want to, not based on user interaction.

EventEmitter system is based on publish/subscribe model, as we subscribe to the events and then publish them.

There are many JavaScript-based libraries with pub/sub support, but Node comes with built-in support.

Node.js Event System

USE OF EVENT EMITTER

Import events module: `var myevents = require("events");`

This events object comes with single property, the EventEmitter class itself.

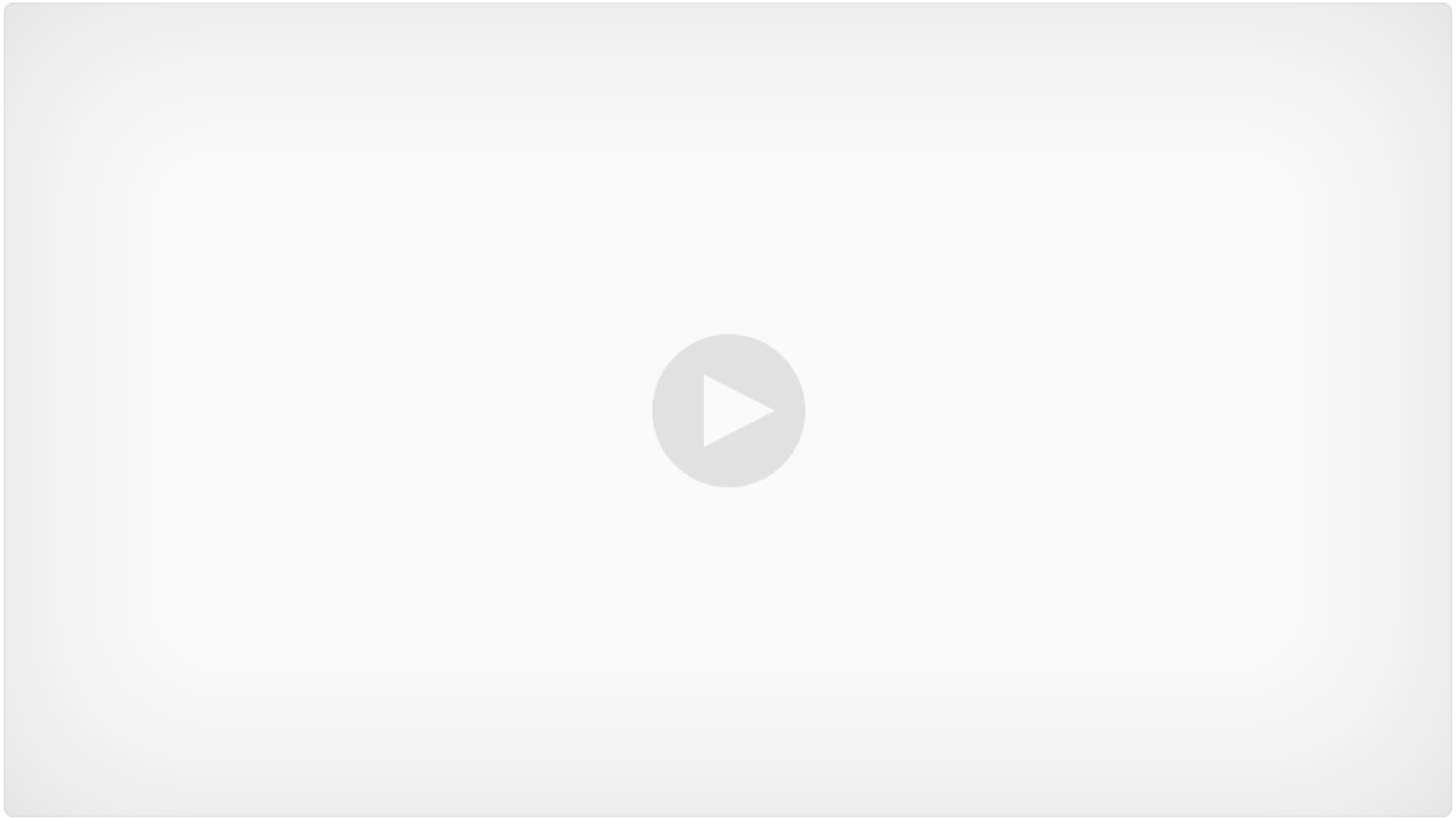
```
var em = require("events").EventEmitter;

var eem2 = new EventEmitter();
eem2.on("CustEvent", function () {
 console.log("custom event called");
});
eem2.emit(" CustEvent ");
```

This Object comes with two main methods.

1. `On:` This method takes two params (name of the event and function to be called when event occurs)
2. `Emit`: It fires the event name to the EventEmitter instance.

Demo for Event Emitter



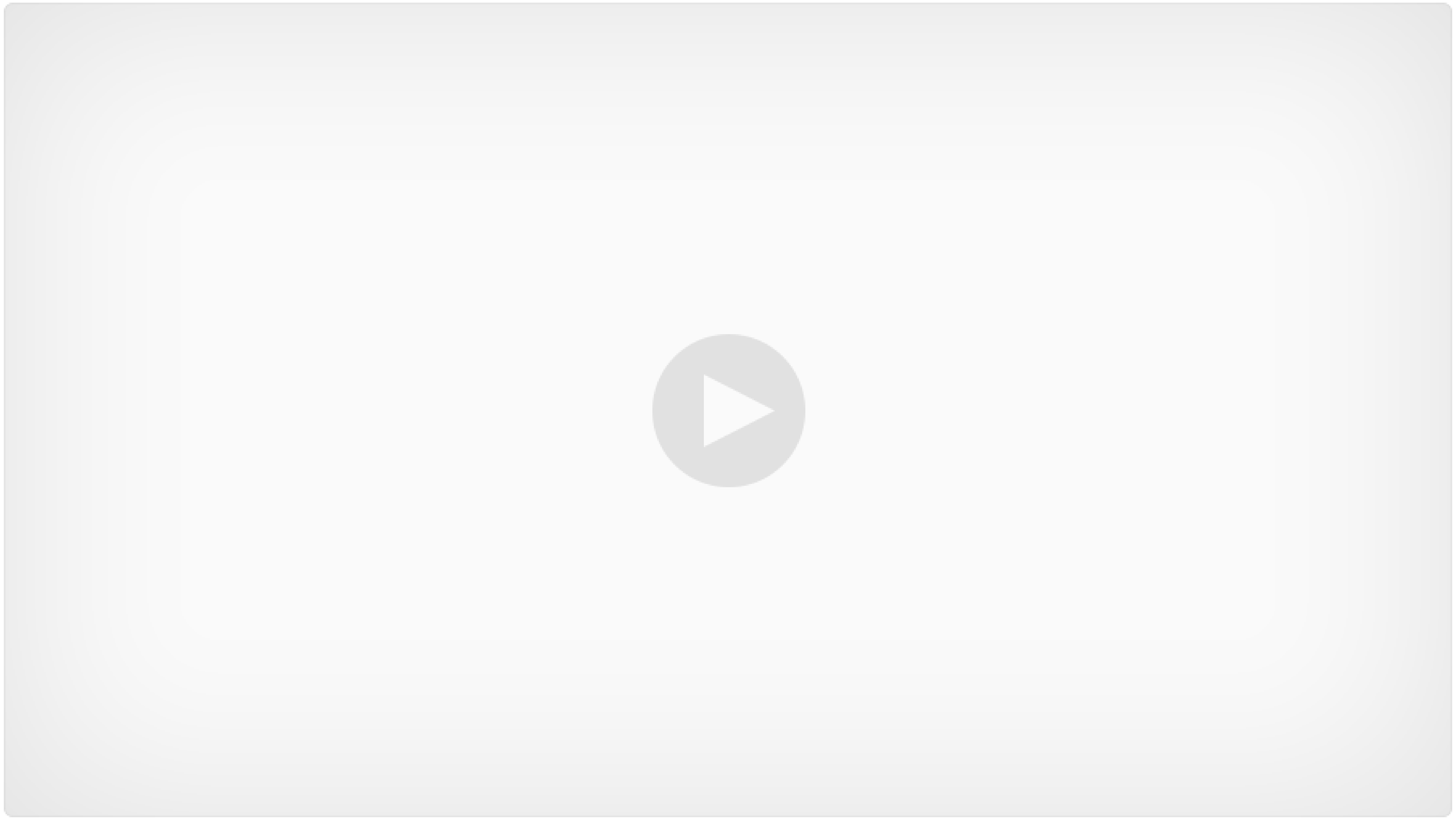
Node.js Event System

USEFUL EVENTEMITTER METHODS

These examples show how to read a physical file in memory synchronously and asynchronously.

`setMaxListeners(n)`	Node by default allows up to ten listeners at a time on one event . If you require more listeners, call this function by passing the numbers.
`Once()`	If you execute this, you will only see the message once. The second occurrence of the event will not be picked up by any listeners as listeners are removed after being used once.
`removeListener()`	If you intend to remove a single listener, two params must be used: event name and listener function.
`em.removeAllListeners(" lname")`	If you want to remove all listeners, use this command.

Demo for Event Emitter Methods



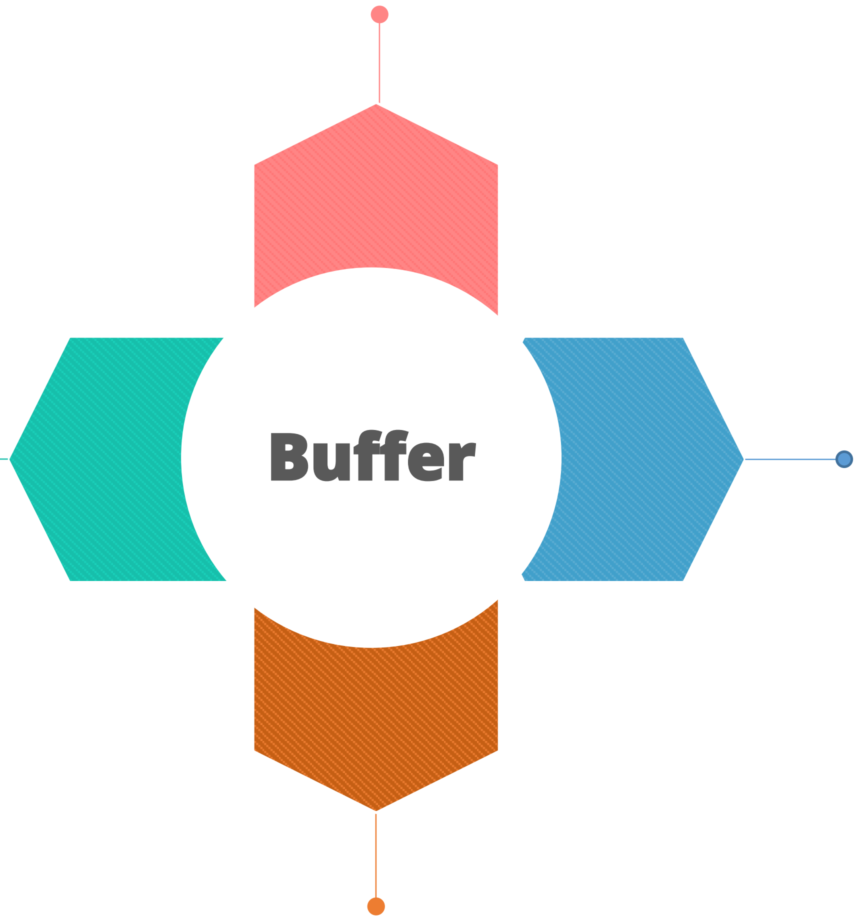
Working with Events and Buffers

Topic 2—Node.js Buffer System

Node.js Buffer System

WHY BUFFER?

JavaScript works well with Unicode string but does not handle binary data well. This is fine in a browser-based environment, where most of the data is in strings, but not in binary format.



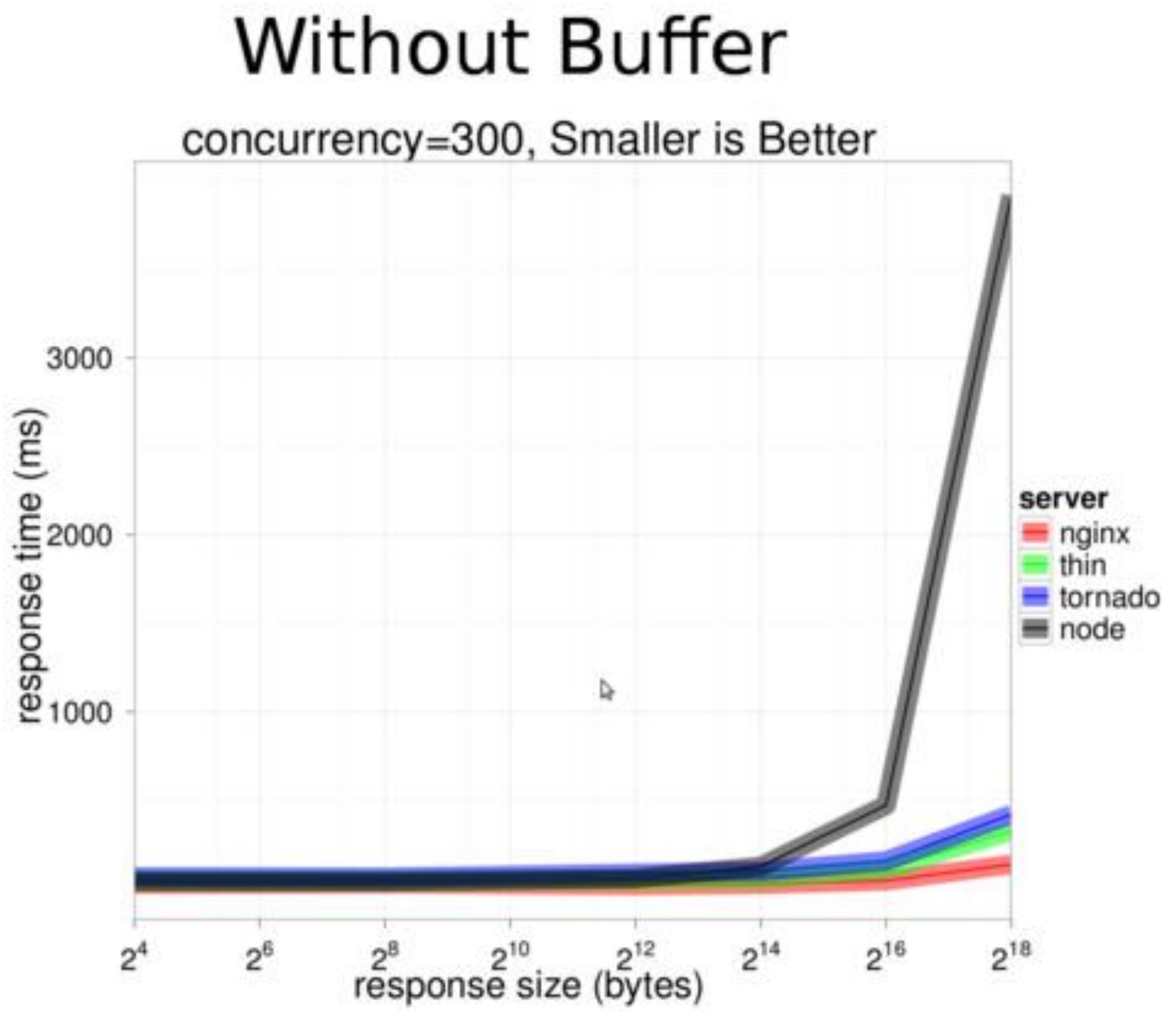
One way to solve this problem is to just use strings. However, this approach is extremely problematic and slow, as you have to deal with strings and not binary data; strings have a tendency to break in outrageous ways, so don't use strings. Use buffers instead!

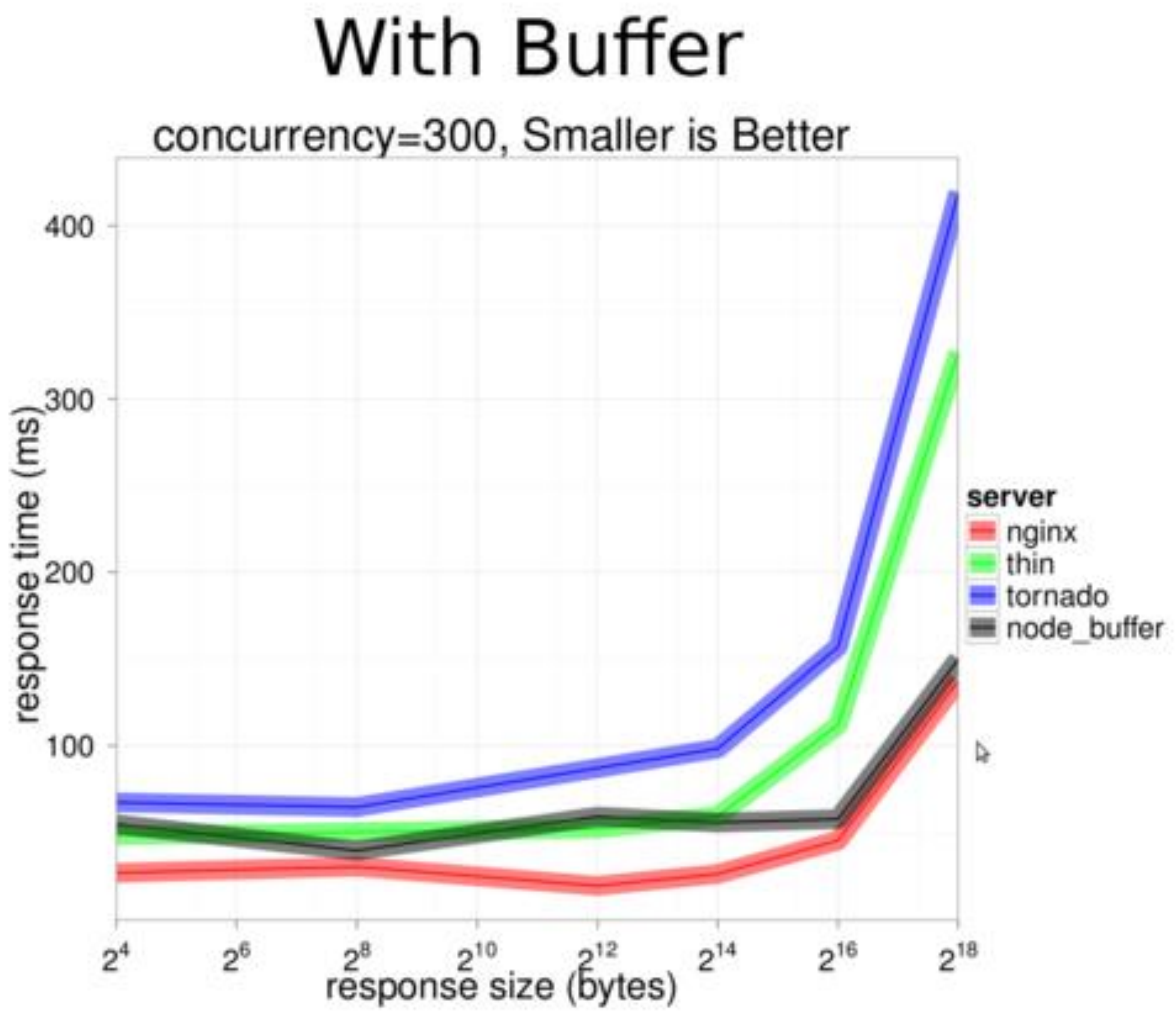
Node.js servers have to manage TCP streams and read and write with filesystems. This makes it essential to deal with purely binary data.

Buffer class provides instances in which we can store the data in the same way we deal with arrays. However, the raw data storage compares to a raw memory location, which is defined outside the Google V8 heap.

Node.js Buffer System

STRING vs. BUFFER COMPARISION





Node.js Buffer System

WHAT IS A BUFFER CLASS?

Buffer class is a global member in Node.js and can be accessed anywhere in a project without including the buffer module.

In Node.js, buffer is an instance of the buffer class; it is designed to deal with raw binary data.

Creating Buffer:

This buffer is uninitialized and contains 8 bytes.

```
var buffer = new Buffer(8);
```

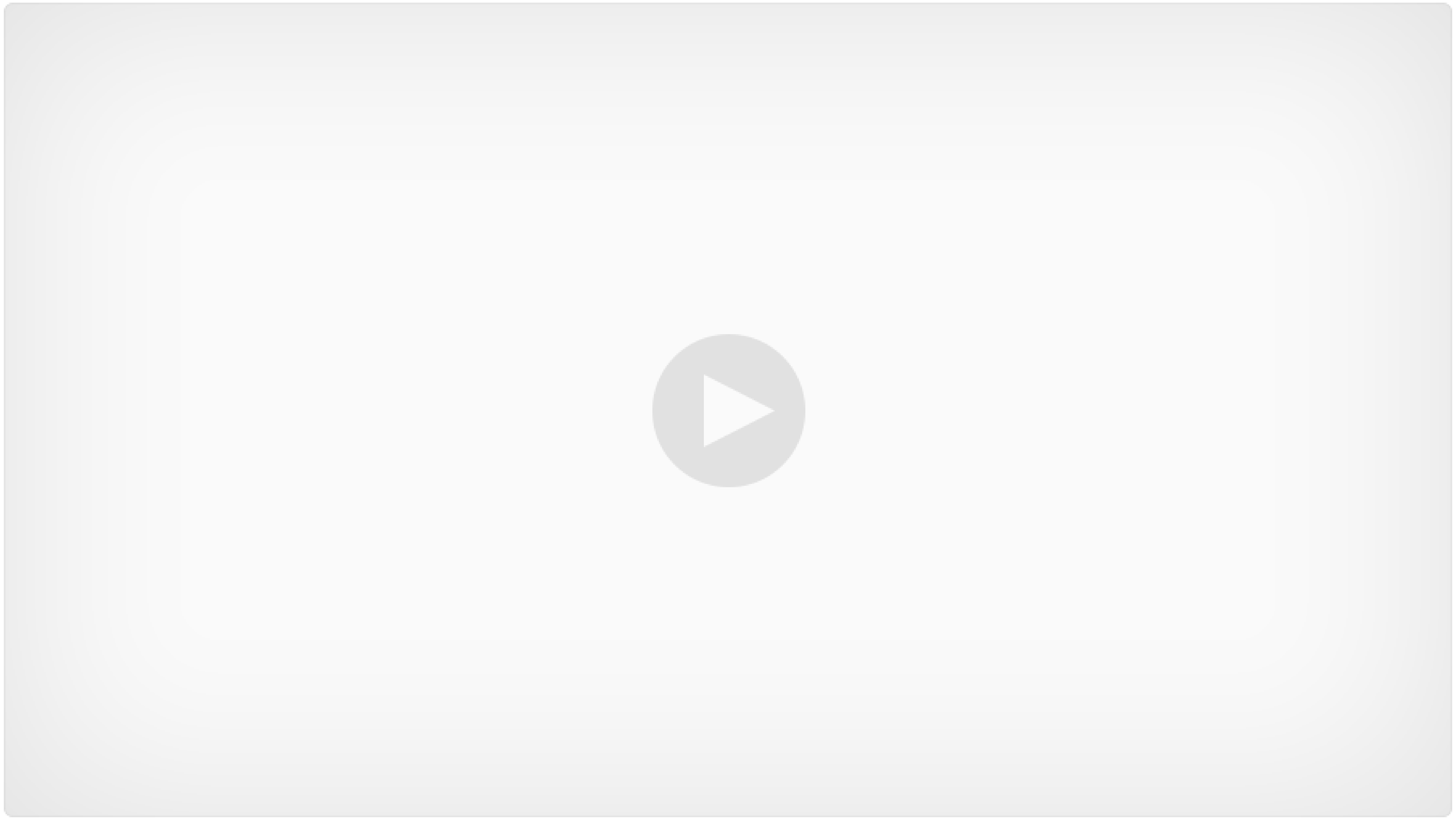
This will initialize the buffer of array. Note that the contents of the arrays represent bytes.

```
var buffer = new Buffer([ 8, 6, 7, 5, 3, 0, 9]);
```

This will initialize the buffer with Encoding type.

```
var buffer = new Buffer("I'm a string!", "utf-8");
```

Demo for Path and Directory



Node.js Buffer System

READING AND WRITING USING BUFFER

In node.js, the syntax below will create a buffer of 100 octets.

```
var buffer = new Buffer(100);

var string = buffer.write("World!");

console.log("String Length = "+ string);
```

toString () is the standard method to read buffers in node.js:

```
var buffer = new Buffer(26);
for (var index = 0 ; index < 26 ; index++) {
     buffer[index] = index + 97;
}
 console.log( buffer.toString('ascii'));
console.log( buffer.toString('ascii',23,52));
console.log( buffer.toString('utf8',0,11));
console.log( buffer.toString('utf16le',0,11));
console.log( buffer.toString('ucs2',0,11));
console.log( buffer.toString('base64',0,11));
```

Demo for Reading and Writing using Buffer

Quiz

QUIZ 1

Transform stream is a type of duplex stream.

a. True

b. False

QUIZ

1

Transform stream is a type of duplex stream.

a. True

b. False

The correct answer is **a. True.**

Transform stream is a type of duplex stream.

QUIZ

2

Which of the following is true about EventEmitter.on property?

a. On property is used to fire an event.

b. On property is used to bind a function with an event.

c. On property is used to locate an event handler.

d. None of the above.

QUIZ

2

Which of the following is true about EventEmitter.on property?

a. On property is used to fire an event.

b. On property is used to bind a function with an event.

c. On property is used to locate an event handler.

d. None of the above.

The correct answer is **b. On property is used to bind a function with an event.**

It is true that on property is used to bind a function with an event

QUIZ

3

Which of the following is true about piping streams?

a. Piping is a mechanism where we provide output of one stream as the input to another stream.

b. Piping is normally used to get data from one stream and to pass output of that stream to another stream.

c. There is no limit on piping operations.

d. All of the above.

QUIZ

3

Which of the following is true about piping streams?

a. Piping is a mechanism where we provide output of one stream as the input to another stream.

b. Piping is normally used to get data from one stream and to pass output of that stream to another stream.

c. There is no limit on piping operations.

d. All of the above.

The correct answer is **d. All of the above.**

All of the following are true about piping streams.

QUIZ

4

Which of the following is true about process global object?

a. The process object is an instance of EventEmitter.

b. Process emits exit event when process is about to exit.

c. Process emits uncaughtException when an exception bubbles all the way back to the event loop.

d. All of the above.

QUIZ
4

Which of the following is true about process global object?

a. The process object is an instance of EventEmitter.

b. Process emits exit event when process is about to exit.

c. Process emits uncaughtException when an exception bubbles all the way back to the event loop.

d. All of the above.

The correct answer is **d. All of the above.**

All of the following are true about global objects.

Key Takeaways

✓ Node.js is very important because it is developed around events.

✓ In Node.js, a buffer is an instance of the buffer class, which is designed to deal with raw binary data.

Thank You