

Deleting Records

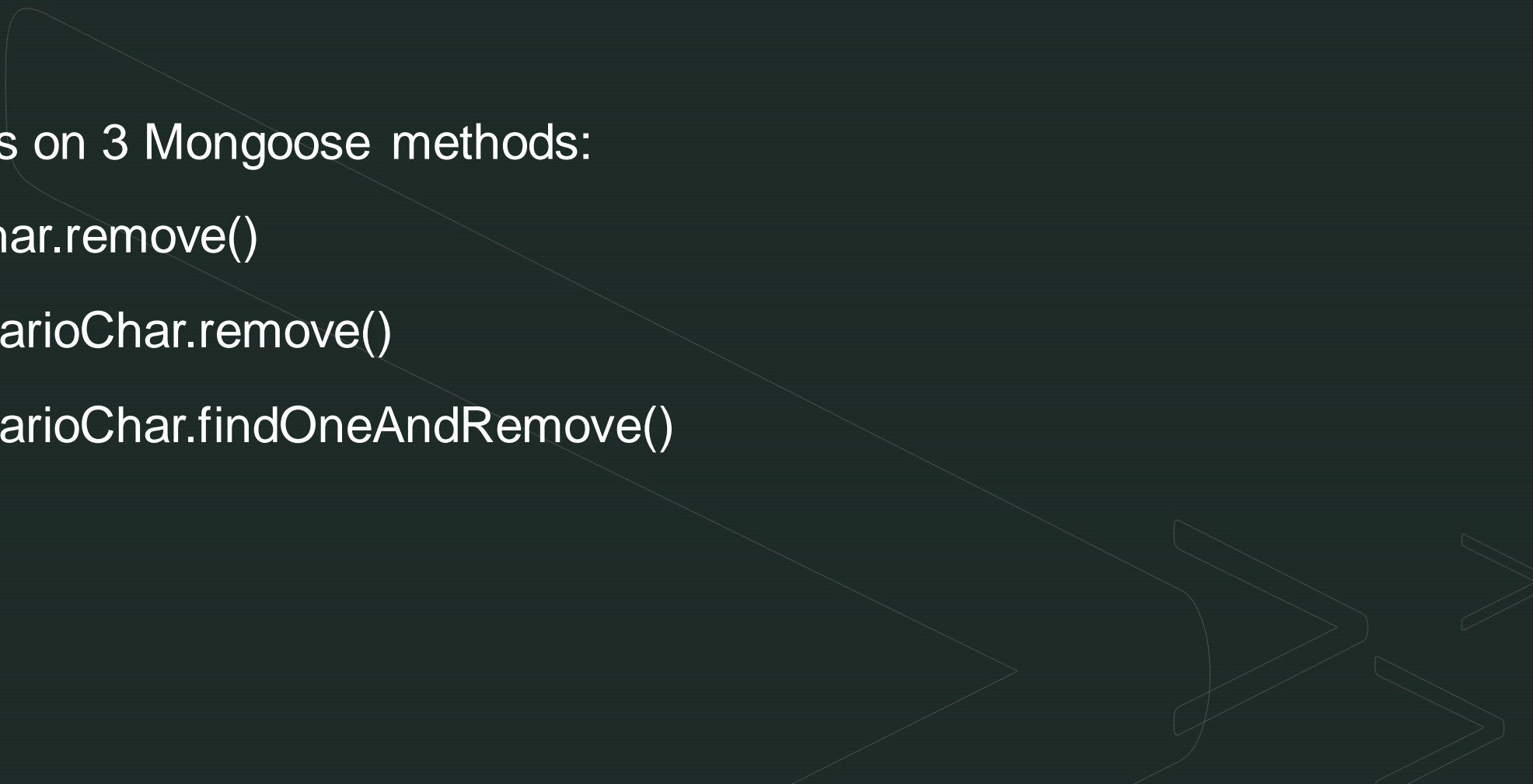




How we can delete records from the database.

We can do this a number of ways, but we'll be looking at the `findOneAndRemove()` method in this tutorial.

Deleting Records

- Focus on 3 Mongoose methods:
 - `char.remove()`
 - `MarioChar.remove()`
 - `MarioChar.findOneAndRemove()`
- 

The process...

- Create and save a new record to the DB
- Use `findOneAndRemove()` to remove the record
- Try to `findOne` in the DB (the one we just removed)
- Assert that the result is null

```
const assert = require('assert');
const MarioChar = require('../models/mariochar');

// Describe our tests
describe('Deleting records', function(){
  var char;
  // Add a character to the db before each tests
  beforeEach(function(done){
    char = new MarioChar({
      name: 'Mario'
    });
    char.save().then(function(){
      done();
    });
  });

  // Create tests
  it('Deletes a record from the database', function(done){
    MarioChar.findOneAndRemove({name: 'Mario'}).then(function(){
      MarioChar.findOne({name: 'Mario'}).then(function(result){
        assert(result === null);
        done();
      });
    });
  });
});
```

Insert
Code

```
const mongoose = require('mongoose');
const url = 'mongodb://localhost/testaroo'

// ES6 Promises
mongoose.Promise = global.Promise;

// Connect to db before tests run
before(function(done){

  // Connect to mongodb
  mongoose.connect(url, { useNewUrlParser: true, useUnifiedTopology: true,
    useFindAndModify: false });
  mongoose.connection.once('open', function(){
    console.log('Connection has been made, now make fireworks...');
    done();
  }).on('error', function(error){
    console.log('Connection error:', error);
  });
});

// Drop the characters collection before each test
beforeEach(function(done){
  // Drop the collection
  mongoose.connection.collections.mariochars.drop(function(){
    done();
  });
});
```