M O N G O

D A T A V A L I D A T I O N

Mongo - Data Validation

Validation logic is executed by Mongoose prior to saving a document to the database.

You can also trigger it manually by calling the validate() method

Built-in validators

-Strings: minlength, maxlength, match, enum

-Numbers: min, max

-Dates: min, max

-All types: required

Built-in validators

```
// Defining a schema
const courseSchema = new mongoose.Schema({
    name: {
        type: String,
        required: true,
        minlength: 5,
        maxlength: 255,
        //match: /pattern/
    },
    category: {
        type: String,
        required: true,
        enum: ['web', 'mobile', 'network']
    },
    author: [String],
    tags: [String],
    date: { type: Date, default: Date.now },
    isPublished: Boolean,
    price: {
        type: Number,
        required: function() { return this.isPublished; },
        min: 10,
        max: 200
    }
});
```

Custom Validators

index.js

```
// Defining a schema
const courseSchema = new mongoose.Schema({
    name: {
...
    },
    category: {
        type: String,
        required: true,
        enum: ['web', 'mobile', 'network']
    },
    author: [String],
    tags: {
        type: Array,
        validate: {
            validator: function(v) {
                return v && v.length > 0;
            },
            message: 'A course should have at least one tag.'
        }
    },
    date: { type: Date, default: Date.now },
    isPublished: Boolean,
    price: {
        type: Number,
        required: function() { return this.isPublished; },
        min: 10,
        max: 200
    }
});
```

Validation Errors

```
async function createCourse(){
    const course = new Course({
        name: 'Node Course',
        category: '-',
        author: 'MarizzaMil',
        tags: null,
        isPublished: true,
        price: 20
    });

    try{
        const result = await course.save();
        console.log(result);
    }
    catch (ex){
        for (field in ex.errors)
            console.log(ex.errors[field].message)
    }
}
```

SchemaType Options

index.js

```
// Defining a schema
const courseSchema = new mongoose.Schema({
    name: {
...
    },
    category: {
        type: String,
        required: true,
        enum: ['web', 'mobile', 'network'],
        lovercase: true,
        trim: true
    },
    author: [String],
    tags: {
...
    },
    date: { type: Date, default: Date.now },
    isPublished: Boolean,
    price: {
        type: Number,
        required: function() { return this.isPublished; },
        min: 10,
        max: 200,
        get: v => Math.round(v),
        set: v => Math.round(v)
    }
});
```