edX

Mongoose Model Methods

A list of the static Mongoose model methods (invoked on a capitalized object, e.g., `Book`) along with their meaning follows:

- `Model.create(data, [callback (error, doc)])`: creates a new Mongoose document and saves it to the database

- `Model.remove(query, [callback(error)])`: removes documents from the collection that match the query; when finished, calls `callback` with `error`

- `Model.find(query, [fields], [options], [callback(error, docs)])`: finds documents that match the query (as a JSON object); possible to select fields and use options

- `Model.update(query, update, [options], [callback(error, affectedCount, raw)])`: updates documents, similar to native `update`

- `Model.populate(docs, options, [callback(error, doc)])`: populates documents using references to other collections; an alternative to another approach described in the next section

- `Model.findOne(query, [fields], [options], [callback(error, doc)])`: finds the first document that matches the query

- `Model.findById(id, [fields], [options], [callback(error, doc)])`: finds the first element for which `_id` equals the `id` argument (cast based on the schema)

- `Model.findOneAndUpdate([query], [update], [options], [callback(error, doc)])`: finds the first document that matches the query (if present) and updates it, returning the document; uses `findAndModify`

- `Model.findOneAndRemove(query, [options], [callback(error, doc)])`: finds the first document that matches the query and removes it when returning the document

- `Model.findByIdAndUpdate(id, [update], [options], [callback(error, doc)])`: similar to `findOneAndUpdate` using only the ID

- `Model.findByIdAndRemove(id, [options], [callback(error, doc)])`: similar to `findOneAndRemove` using only the ID

Warning Not all the Mongoose model methods trigger hooks. Some of them are executed directly. For example, calling `Model.remove()` does not trigger the `remove` hook, because no Mongoose documents are involved (instances of `Model` that use lowercase literals, e.g., `practicalNodeBook`).

The complete list of the methods is extensive; therefore, refer to the official Mongoose API documentation. The most used instance methods are as follows:

- `doc.model(name)`: returns another Mongoose model

- `doc.remove([callback(error, doc)])`: removes this document

- `doc.save([callback(error, doc, affectedCount)])`: saves this document

- `doc.update(doc, [options], [callback(error, affectedCount, raw)])`: updates the document with `doc` properties, and `options` parameters, and then upon completion fires a callback with `error`, number of `affectedCount` and the database output

- `doc.toJSON([option])`: converts a Mongoose document to JSON (options are listed later)

- `doc.toObject([option])`: converts a Mongoose document to a plain JavaScript object (options are listed later)

- `isModified([path])`: true/false, respectively, if some parts (or the specific path) of the document are or are not modified

- `markModified(path)`: marks a path manually as modified which is useful for mixed (`Schema.Types.Mixed`) data types because they don't trigger the modified flag automatically

- `doc.isNew`: true/false, respectively, whether the document is new or not new

- `doc.id`: returns the document ID

- `doc.set(path, value, [type], [options])`: sets `value` at a `path`

- `doc.validate(callback(error))`: checks validation manually (triggered automatically before `save()`)

Options for `toObject()` and `toJSON()` are as follows:

- `getters`: true/false, calls all `getters` including path and virtual types

- `virtuals`: true/false, includes virtual `getters` and can override the `getters` option

- `minimize`: true/false, removes empty properties/objects (defaults to true)

- `transform`: transforms the function called right before returning the object

For more methods, visit the Mongoose document API.