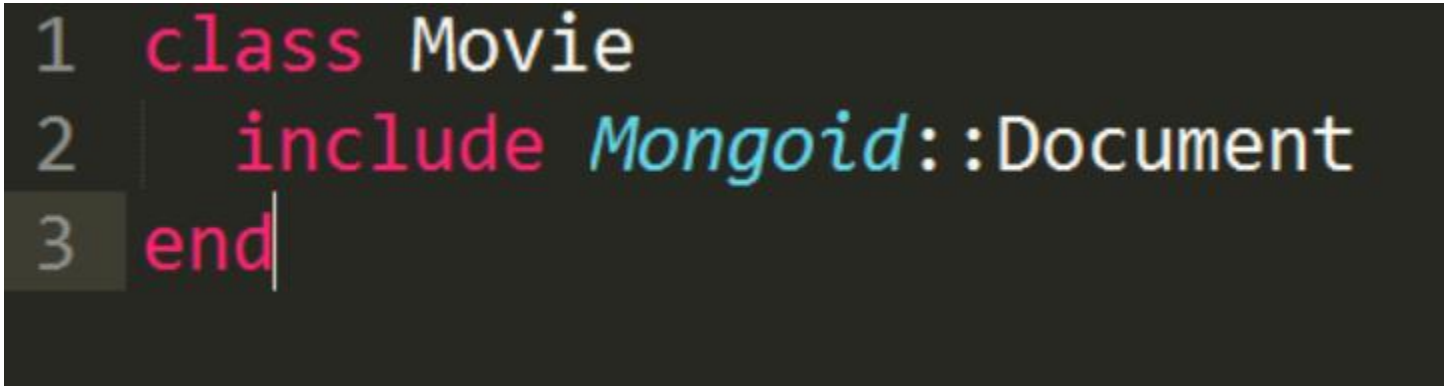
In this lecture, we will discuss…

- ✧ Document Class
- ✧ Fields
- ✧ Field Types
- ✧ Field Aliases
- ✧ Custom Fields

Document

- Documents are the core objects in Mongoid
  - `Mongoid::Document`
- Documents can be stored in a collection or embedded in other documents



Fields

- ✧ Fields are attributes
  - `field`
  - `type` - String by default
  - `rails g model`

```
1 class Movie
2   include Mongoid::Document
3   field :title, type: String
4   field :type, type: String
5   field :rated, type: String
6   field :year, type: Integer
7 end
```

`rails g model` - command

```
$ rails g model Movie title type rated year:integer release_date:date \
    runtime:Measurement votes:integer countries:array languages:array \
    genres:array filming_locations:array metascore simple_plot:text \
    plot:text url_imdb url_poster directors:array actors:array
```

```
$ rails g model Actor name birth_name data_of_birth:Date height:Measurement bio:text
```

Field Types

Array	Boolean	DateTime	Hash
BigDecimal	Date	Float	Integer
BSON	Range	Regexp	String
Symbol	Time	TimeWithZone	

Timestamps

- Timestamp information is not added by default in Mongoid -- as it is within ActiveRecord.
- `created_at` and `updated_at` fields done via `Mongoid::Timestamps` mixin.
- `touch` - Will update the document's `updated_at` timestamp.

Field Aliases

```ruby
class Actor
  include Mongoid::Document
  include Mongoid::Timestamps

  field :name, type: String
  field :birthName, as: :birth_name, type: String
  field :data_of_birth, type: Date
  field :height, type: Measurement
  field :bio, type: String
```

- birthName in document → mapped to birth_name in model
- Comply with rails naming convention
- Helps during compression

Custom Fields

- You can define custom types in Mongoid and determine how they are serialized and deserialized
- 5 methods in total
  - initialize
  - mongoize (instance method)
  - mongoize, demongoize, evolve (class methods)
- Example: `Measurement`

```
:runtime=>{:amount=>60, :units=>"min"}
```

store_in

```
1 class Location
2     include Mongoid::Document
3     store_in collection: "places"
4     field :city, type: String
5     field :state, type: String
6     field :country, type: String
7 end
```

- Application type to Document type mapping
- `Location` gets stored in to “`places`” collection

Summary

- ✧ Good data type support
- ✧ Aliases, Timestamps
- ✧ Document class and Custom Fields

What's Next?

- ✧ CRUD