SAILS.JS

AN INTRODUCTION

Nic Jansma | nicj.net | @NicJ

SAILS.JS IS:

- A MVC backend web framework for Node.js
- Built on top of Express
- Inspired by Ruby on Rails / Symfony / Zend
- Convention over configuration
- Ideal for chat, realtime dashboards and multiplayer games

SAILS.JS CORE FEATURES

- 100% JavaScript
- Database agnostic (includes custom ORM)
- Auto-generated REST APIs
- Easy WebSocket support and integration with REST
- Reusable security policies
- Front-end agnostic
- Flexible asset pipeline (builds)

GETTING STARTED

```
npm install -g sails
sails new [project path]
cd [project path]
```

RUNNING YOUR NEW WEB APP

```
sails lift
```

```
info: Starting app...

info:
info:
info:    Sails               <|
info:    v0.10.2              |\
info:                        /|.\
info:                       / || \
info:                     ,'  |'  \
info:                  .-'.-==|/_--'
info:                  `--'-------'
info:    __---___--___---___--___---___--___
info:  ____---___--___---___--___---___--___-__
info:
info: Server lifted in `example1`
info: To see your app, visit http://localhost:1337
info: To shut down Sails, press <CTRL> + C at any time.

debug: -------------------------------------------------------
debug: :: Thu Aug 07 2014 06:43:55 GMT-0400 (Eastern Daylight Time)

debug: Environment : development
debug: Port        : 1337
debug: -------------------------------------------------------
```

HTTP://LOCALHOST:1337

1 Generate a REST API.

Run `sails generate api user`. This will create two files: a model and a controller.

2 Lift your app.

Run `sails lift` to start up your app server. If you visit `http://localhost:1337/user` in your browser, you'll see a WebSocket-compatible user API.

3 Dive in.

Blueprints are just the beginning. You'll probably also want to learn how to customize your app's routes, set up security policies, configure your data sources, and build custom controller actions. For more help getting started, check out the links on this page.

Docs

- App Structure
- Reference
- Supported Databases

Tutorials

- Sails 101

Community

- StackOverFlow
- GitHub
- Google Group
- IRC (#sailsjs on freenode)

GENERATE MODEL AND REST API

Let's create a new `beer` model and REST API

```
sails generate api beer
```

This creates skeleton files:
`api\controllers\BeerController.js` and
`api\model\Beer.js`

TRY IT OUT

http://localhost:1337/beer

```
[]
```

http://localhost:1337/beer/create?name=Centennial IPA&brewery=Founders&have=10

```
{
    "name": "All Day IPA",
    "brewery": "Founders",
    "createdAt": "2014-08-07T13:11:10.536Z",
    "updatedAt": "2014-08-07T13:38:21.517Z",
    "id": 1,
    "have": 10
},
```

TRY IT OUT

http://localhost:1337/beer


```
[
    {
        "name": "All Day IPA",
        "brewery": "Founders",
        "createdAt": "2014-08-07T13:11:10.536Z",
        "updatedAt": "2014-08-07T13:38:21.517Z",
        "id": 1,
        "have": 10
    }
]
```

WHAT JUST HAPPENED

- Sails created a Model (`Beer.js`) and Controller (`BeerController.js`)
- Sails *blueprints* automatically configure new routes for the model (eg `POST /beer` and `/beer/create`).
- Sails uses whatever storage layer you want for persistence

ANATOMY OF A SAILS.JS APP

- `api/controller/` - controllers
- `api/models/` - models
- `api/policies/` - authentication / authorization
- `api/responses/` - `res.xyz()` handlers
- `api/services/` - services
- `assets/` - static assets
- `config/` - app config
- `tasks/` - grunt / cli tasks
- `views/` - views

MODELS

- http://sailsjs.org/#!documentation/models
- `api/models/*`
- Uses Waterline ORM (best parts of Active Record, Hibernate and Mongoose)

```
module.exports = {
  attributes: {
    name: {
      type: 'string',
      required: true
    },
    brewery: {
      type: 'string',
      required: true
    },
    have: {
      type: 'integer',
      defaultTo: 1
    }
  }
};
```

CONTROLLERS

- http://links.sailsjs.org/docs/controllers
- `api/controllers/*`
- With Blueprints (`config/blueprints.js`), you automatically get CRUD REST and "shortcut" routes

BLUEPRINT ROUTES

By default, Sails creates three types of *blueprint* routes:

- **RESTful routes** for `/:model` (HTTP `GET`, `DELETE`, etc)
- **Shortcut routes** to easily test your model via HTTP `GET` requests, such as `/:model/delete/:id` (should be turned off in production)
- **Action routes** for any additional actions in your controllers

Authentication and access control are handled via policies

BLUEPRINT ROUTES

	REST	**Shortcut**
Query	GET /api/:model	
Fetch	GET /api/:model/:id	
Create	POST /api/:model	GET /api/:model/create
Update	PUT /api/:model/:id	GET /api/:model/update/:id
Delete	DELETE /api/:model/:id	GET /api/:model/destroy/:id

BLUEPRINT ACTIONS

By default, Sails creates actions on your controllers for ORM functionality:

- find
- findOne
- create
- update
- destroy
- populate
- add
- remove

CUSTOM ACTIONS AND ROUTES

```
// config/routes.js
module.exports.routes = {
  '/': {
    view: 'homepage'
  },
  'post /beer/:id/drink': 'BeerController.drink'
}
```

```
// BeerController.js
module.exports = {
  drink: function (req, res) {
    if (!req.params.id) { return res.badRequest('ID not supplied'); }

    Beer.findOne({ id: req.params.id }, function(err, model) {
      if (err || !model) {
        return res.badRequest('Beer not found');
      }

      model.have--;

      model.save(function(err) {
        return res.ok(model);
      });
    });
  }
};
```

VIEWS

- http://sailsjs.org/#!documentation/views
- `api/views/*`
- Uses EJS by default but can switch to Jade, etc

```
// BeerController.js
module.exports = {
  show: function(req, res) {
    Beer.find({}, function (err, beers) {
      res.view('show-beers', { title: 'Beers', beers: beers });
    });
  }
};
```

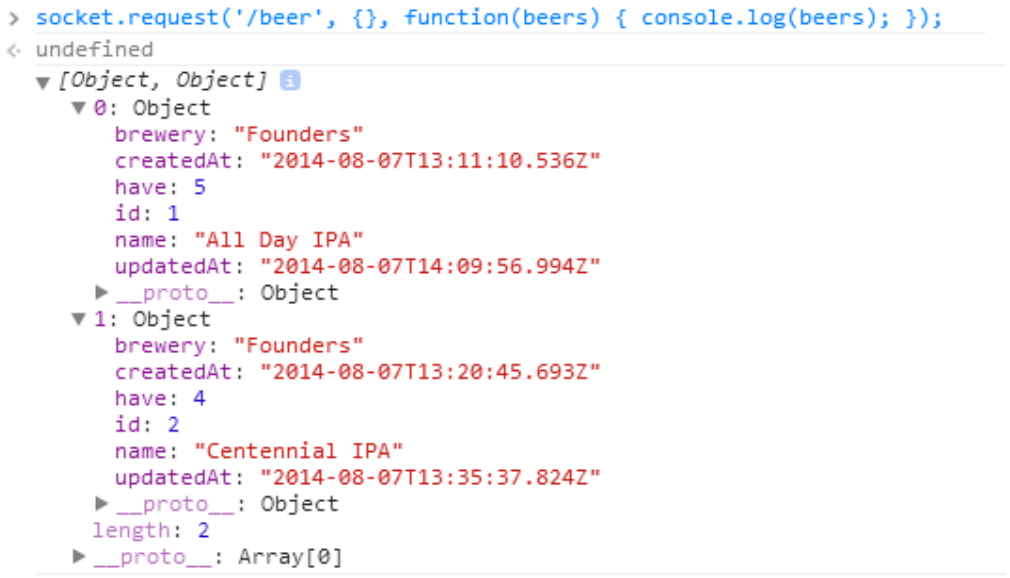
```
// show-beers.ejs
<ul>
<% for(var i=0; i<beers.length; i++) {%>
   <li><%= beers[i].name %></li>
<% } %>
</ul>
```

SOCKET.IO INTEGRATION

Sails automatically translates incoming socket.io messages into Express requests

Also gives you Pub/Sub via `res.broadcast()` and `req.join()`

```
var socket = io.connect('http://localhost:1337');
socket.request('/beer', {}, function(beers) { console.log(neers); });
```



LINKS

- Sails.js: sailsjs.org
- SailsCasts: irlnathan.github.io/sailscasts/
- Presentation: slideshare.net/nicjansma
- Code: github.com/nicjansma/talks/

Thanks - Nic Jansma - nicj.net - @NicJ