# Building APIs with LoopBack

Raymond Camden

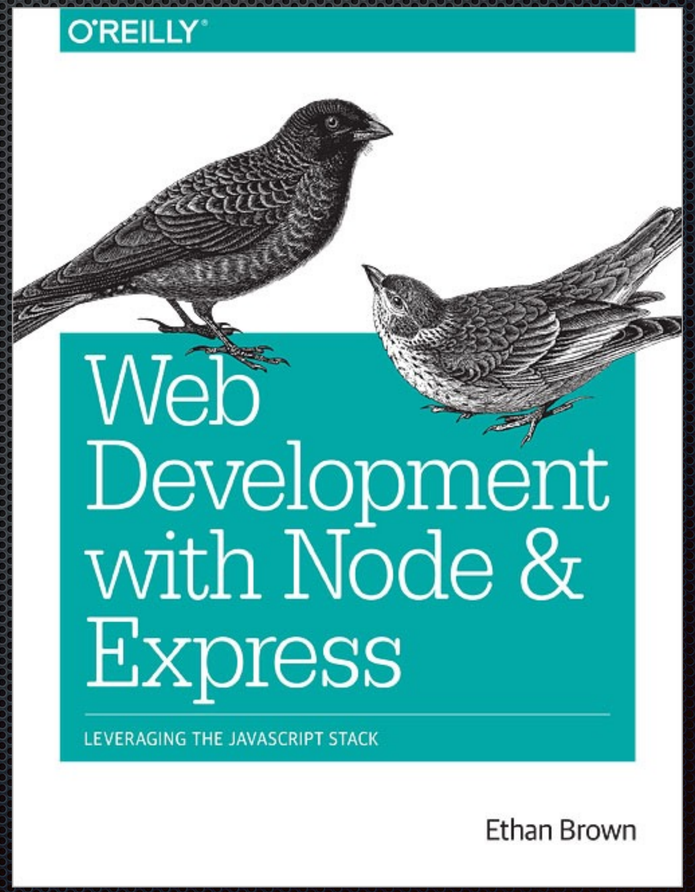
# Before we get started

# Things you should know (part 1)

- Assets: https://github.com/cfjedimaster/LoopBackWorkshop
- Node.js
  - Kinda
  - If you "Stack Overflow"-know Node, you're golden
- Express
  - Kinda

# "Web Development with Node & Express"

- Ethan Brown
- Covers everything (ok, not *everything*)
- http://shop.oreilly.com/product/0636920032977.do



# Things you should know (part 2)

- Captain Crunch's full name is...
  - Captain Horatio Magellan Crunch
- His ship is the...
  - S.S. Guppy
- There have been 26 varieties
  - https://en.wikipedia.org/wiki/Cap%27n_Crunch#Variations

# Things you should install

- Node (but if you don't have this already...)
- A good editor (hint - Visual Studio Code)
- Mongo (totally optional)

# Gameplan

- 30-40 minutes of me talking about LoopBack and showing some examples
- Quick Break (5 minutes)
- Exercises

# Raymond Camden

- Developer Advocate for IBM
- StrongLoop, Bluemix, Cordova/PhoneGap, Node, and web stuff in general
- Blogging at raymondcamden.com
- Tweeting at @raymondcamden

# Why build APIs?

# APIs give you...

- A way to share your data or processes with others
- A way to provide data for your client-side/mobile/hybrid/TV/etc app

# LoopBack.io

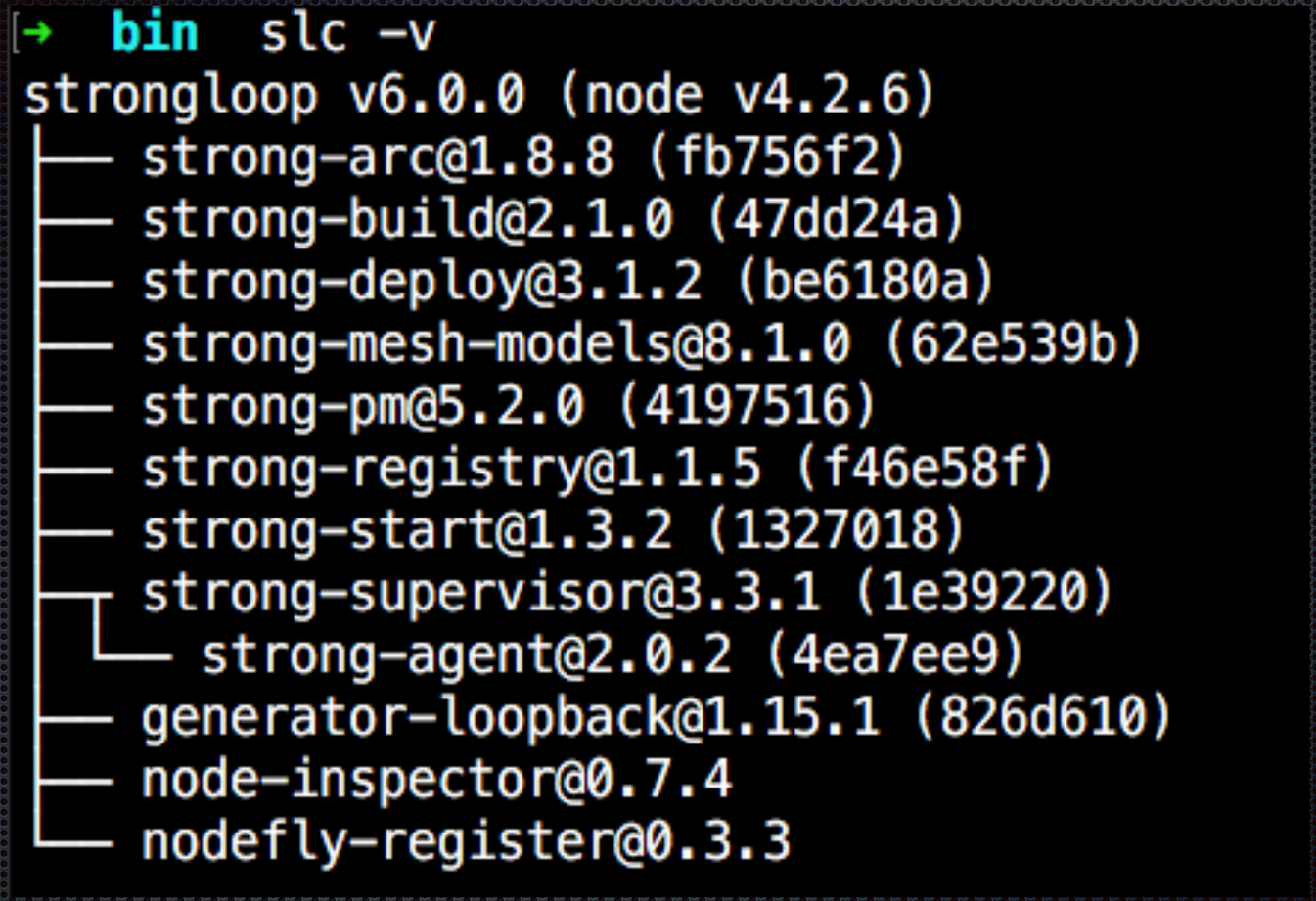
# LoopBack

- Open source framework designed for APIs
- Rapid creation of Models and REST APIs
- Simple ORM system for CRUD
- Built on Express
- Very detailed/configurable security policies
- loopback.io

# Getting Started

# Installation

- Requires npm (Node Package Manager)
- Install Node.js (nodejs.org)
- npm install –g strongloop
- CLI: slc


➜ bin slc -v
strongloop v6.0.0 (node v4.2.6)
├── strong-arc@1.8.8 (fb756f2)
├── strong-build@2.1.0 (47dd24a)
├── strong-deploy@3.1.2 (be6180a)
├── strong-mesh-models@8.1.0 (62e539b)
├── strong-pm@5.2.0 (4197516)
├── strong-registry@1.1.5 (f46e58f)
├── strong-start@1.3.2 (1327018)
├── strong-supervisor@3.3.1 (1e39220)
│ └── strong-agent@2.0.2 (4ea7ee9)
├── generator-loopback@1.15.1 (826d610)
├── node-inspector@0.7.4
└── nodefly-register@0.3.3

# Creating an Application

- `slc loopback`
- Follow the prompts
- Run the application


ENOUGH TALK
TIME TO DEMO
memegenerator.net

# LoopBack and APIs

- You aren't defining an API
  - I want to support GET /cats
  - I want to handle PUT /cats
- You are defining data
  - I work with Cats (they have names and weights)
- "Ok, let me handle that for you!"

# Models

- Models define your API
- You describe properties - LoopBack does the rest (get it - "REST"... heh)
- Inheritance + Relationships
- Out of the box - full CRUD + search


ENOUGH TALK
TIME TO DEMO
memegenerator.net

# Datasources

- Where LoopBack stores crap
- Referred to as connectors as well
- Cloudant, DB2, Memory, MongoDB, MySQL, Oracle, PostgreSQL, Redis, SQL Server
- npm install loopback-connector-mongodb --save
- Roll your own


ENOUGH TALK
TIME TO DEMO
memegenerator.net

# Security

# Security

- Security system models built in
- Access Control List
- What users can do what
- Users can be grouped into Roles
- Methods can be grouped into read/write


Meh security system.
Let me showz u him!
ICANHASCHEEZBURGER.COM

# Customizations

- Remote Hooks
- Operation Hooks
- Remote Methods
- Crap ton of stuff for Express too


ENOUGH TALK
TIME TO DEMO
memegenerator.net

# Additional Random Stuff

- File Attachments
- Models based on APIs (an API for an API)
- And more...

# Questions?

# Exercises - Let's Build an API!

# General Plan

- A Guestbook
- Users can write a note for the site saying hello, I love you, etc.
- Ajax based application
- Server-side validation
- Prevent delete

# Exercise 1

- Create a LoopBack application (name it whatever)
- Run it
- Confirm you see the home page and explorer

# Exercise 2

- Create a Guestbook Model
- Properties: Name, Email, Message, Posted

# Exercise 3

- Add persistance
- Either file based or Mongo
  - npm install loopback-connector-mongodb --save

# Exercise 4

- Build the front end
- GET all posts and render
- Form to POST a new post
- Ray: Show them middleware.json

# Exercise 5

- Sort by newest first
- Show a max of 10 items
- Reference: https://docs.strongloop.com/display/public/LB/Querying+data

# Exercise 6

- Default created to now()
- Two ways:
  - set the default to a function (defaultFn:"now")
  - use a remote hook to set it

# Exercise 7

- Sanitize time!
- Intercept name and body and strip out HTML
- Block 'created'

# Remote Hooks

- beforeRemote
- afterRemote
- afterRemoteError

# Example

```
modelName.beforeRemote( methodName, function( ctx, next) {
  //stuff here
  next();
});
```

# Exercise 8

- Prevent deletes
- A small part of a greater security model

# Example

- `slc loopback:model`
- Methods: updateAttribures, upsert, destroyById
- Reference: https://docs.strongloop.com/display/public/LB/Controlling+data+access

# Another Example

- `Product.disableRemoteMethod('create', true);`
- https://docs.strongloop.com/display/public/LB/Exposing+models+over+REST#ExposingmodelsoverREST-Read-Onlyendpointsexample