LoopBack

A Node.js API Framework

SWK Magdeburg - Björn Strobach - Dezember 2017

What is LoopBack?

"LoopBack is a highly-extensible, open-source Node.js framework"

Quickly create dynamic end-to-end REST APIs.

Connect devices and browsers to data and services.

Use Android, iOS, and AngularJS SDKs to easily create client apps.

Add-on components for file management, 3rd-party login, and OAuth2.

Runs on-premises or in the cloud

Node.js

Asynchronous event driven JavaScript runtime

Built on Chrome's V8 JavaScript engine

Designed to build scalable network applications

Simple Node.js server

```javascript
const http = require('http');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`)
});
```

How?

Install Node.js (Better: use a Node version manager)

Save above content to a file (e.g. `app.js`) and run:

```
$ node app.js
```

Visit 127.0.0.1:3000

Demo

Express.js

Minimal and flexible Node.js framework

With a myriad of HTTP utility methods and middleware

Provides a thin layer of fundamental web application features, without obscuring Node.js features

Simple Express.js app

```javascript
const express = require('express')
const app = express()

app.get('/', (req, res) => res.send('Hello World!'))

app.listen(3000, () => console.log('Example app listening on p
```

How?

Install Node.js (Better: use a Node version manager)

Save above content to a file (e.g. `app.js`) and run:

```
$ npm i express && node app.js
```

Visit 127.0.0.1:3000

Express Generator

Quickly create an application skeleton

```
$ npm i [-g] express-generator
```

```
$ ./node_modules/.bin/express myApp
```

Folder structure

```
.
├── app.js
├── bin
│   └── www
├── package.json
├── public
│   ├── images
│   ├── javascripts
│   └── stylesheets
│       └── style.css
├── routes
│   ├── index.js
│   └── users.js
└── views
    ├── error.jade
    ├── index.jade
    └── layout.jade

7 directories, 9 files
```

Next steps

Install dependencies

```
$ cd myApp && npm i
```

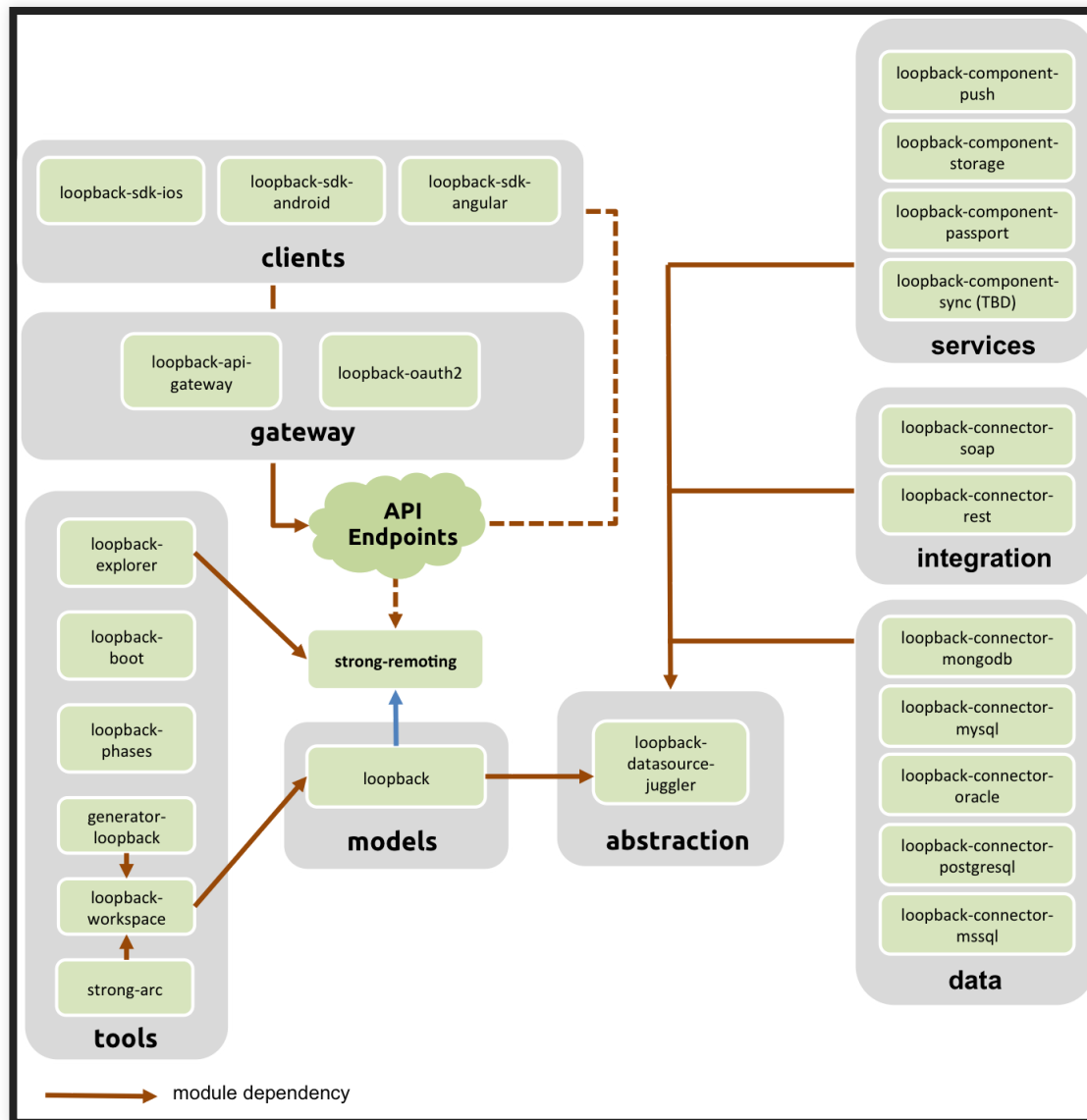## Run the app

```
$ DEBUG=myapp:* npm start
```

Demo

LoopBack

A set of Node.js modules

Can be used independently or together

Overview of modules

clients
- loopback-sdk-ios
- loopback-sdk-android
- loopback-sdk-angular

gateway
- loopback-api-gateway
- loopback-oauth2

services
- loopback-component-push
- loopback-component-storage
- loopback-component-passport
- loopback-component-sync (TBD)

integration
- loopback-connector-soap
- loopback-connector-rest

tools
- loopback-explorer
- loopback-boot
- loopback-phases
- generator-loopback
- loopback-workspace
- strong-arc

models
- loopback

abstraction
- loopback-datasource-juggler

data
- loopback-connector-mongodb
- loopback-connector-mysql
- loopback-connector-oracle
- loopback-connector-postgresql
- loopback-connector-mssql

API Endpoints

strong-remoting

→ module dependency

Most important concepts

Models

Application logic

Datasource and connectors

LoopBack components

Models

Represent backend data sources (such as databases or services like REST, SOAP)

JavaScript objects with Node and Rest-APIs

One base model object, with hooks and data validation methods

Connected to a data source Every model comes with a predefined REST API (full set of CRUD)

Built-in models

Every LoopBack application comes with predefined built-in models

User, Role, and Application

So you don't have to create these common models from scratch

Custom Models

Define own models specific to the application

One can extend the built-in models

Stored in model definition JSON file

Model relations

One can define relations between models

E.g. BelongsTo, HasMany, HasAndBelongsToMany, and more

Model CRUD operations

Operation	REST	LoopBack model method	Corresponding SQL Operation
create	PUT|POST /modelname	create()	INSERT
read (retrieve)	GET /modelName? filter=...	find()	SELECT
update (modify)	PUT /modelName	updateAll()	UPDATE
delete (destroy)	DELETE /modelName/modelID	destroyByID()	DELETE

Application logic

Remote methods (custom REST endpoints)

Remote hooks triggered by remote methods

Operation hooks triggered by CRUD

Boot scripts

Middleware (Express)

LoopBack components

Push notifications

Storage components

Third party login

Synchronization

OAuth2

Getting started

Install Node.js (Better: use a Node version manager)

Install LoopBack CLI tool

```
$ npm i -g loopback-cli
```

Run the LoopBack application generator

```
$ lb
```

Demo

Q & A

Thank you for your attention.

GitHub: @bstrobach

Twitter: @bstrobach

Xing: http://xing.to/bstrobach

Slides: https://bstrobach.github.io/loopback-swk-md/