Writing a Compass Plugin

Anna Herlihy
Software Engineer on MongoDB Compass

#MDBlocal

1. Compass Feature Tour
2. *Where* can my plugin fit into Compass?
3. *How* can my plugin fit into Compass?
4. Example

Feature Tour

Why write a plugin?

- Limitless potential: Customized charts, specific DB commands, show cluster health, and more...
- Can solve your specific problems. Plugins can be as personalized as you like.
- Can use plugins to share views, tools, or debug
- It is fun and easy!

Where does my plugin fit visually into Compass?

Plugin Roles

1. Header.Item
2. Instance.Tab
3. Database.Tab
4. Collection.Tab
5. CollectionHUD.item (Heads Up Display)

MongoDB Compass Dev - localhost:27017/Venues.Restaurants

My Cluster

4 DBS 3 COLLECTIONS

filter

Venues

Restaurants

admin

local

test

localhost:27017 STANDALONE MongoDB 3.6.0 Enterprise

Venues.Restaurants

DOCUMENTS	TOTAL SIZE	AVG. SIZE	INDEXES	TOTAL SIZE	AVG. SIZE
25.4k	10.1MB	419B	1	240.0KB	240.0KB

Documents Schema Explain Plan Indexes Validation

FILTER { field: 'value' } OPTIONS FIND

INSERT DOCUMENT VIEW LIST TABLE

Displaying documents 1 - 20 of 25356

```
  _id: ObjectId("59f7306e037d92ff4dc2a272")
  borough: "Brooklyn"
  cuisine: "American"
> grades: Array
  name: "C & C Catering Service"

  _id: ObjectId("59f7306e037d92ff4dc2a273")
> address: Object
  borough: "Brooklyn"
  cuisine: "Chinese"
> grades: Array
  name: "May May Kitchen"
  restaurant_id: "40358429"

  _id: ObjectId("59f7306e037d92ff4dc2a274")
> address: Object
  borough: "Manhattan"
  cuisine: "American"
  name: "1 East 66Th Street Kitchen"
  restaurant_id: "40359480"

  _id: ObjectId("59f7306e037d92ff4dc2a275")
> address: Object
  cuisine: "Jewish/Kosher"
> grades: Array
  name: "Seuda Foods"
  restaurant_id: "40360045"
```

Header.Item

- **Purpose:**
  - Top-level: display global information (e.g. MongoDB version, topology)

- **UI Considerations:**
  - Header space is very small
  - Lengthy information should be hidden until asked for

- **Potential Plugins:**
  - Current user information

MongoDB Compass Dev - localhost:27017

My Cluster

4 DBS 3 COLLECTIONS

filter

- Venues
- admin
- local
- test

localhost:27017 STANDALONE

MongoDB 3.6.0 Enterprise

Databases Performance

name option renders the tab name

CREATE DATABASE

component option renders the component in the frame

Database Name	Storage Size	Collections	Indexes
Venues	4.5MB	1	1
admin	16.0KB	0	2
local	36.0KB	1	1
test	32.0KB	1	1

Instance.Tab

- **Purpose:**
  - Large instance-level features (e.g. real time server stats). Currently have “Databases” and “Performance”.

- **Potential Plugins:**
  - Cluster Health
  - List of All Users

MongoDB Compass Dev - localhost:27017/Venues

My Cluster

4 DBS 6 COLLECTIONS

filter

- Venues
  - Bars
  - Clubs
  - Hotels
  - Restaurants
- admin
- local
- test

localhost:27017 STANDALONE

MongoDB 3.6.0 Enterprise

Collections ← name option renders the tab name

CREATE COLLECTION

component option renders the component in the frame

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size
Bars	1,255	33.0 B	40.4 KB	1	24.0 KB
Clubs	688	33.0 B	22.2 KB	1	4.0 KB
Hotels	1,800	33.0 B	58.0 KB	1	28.0 KB
Restaurants	25,356	418.8 B	10.1 MB	1	240.0 KB

Database.Tab

- **Purpose:**
  - Large database-level feature. Right now we just have a list of collections.

- **Potential Plugins:**
  - System.profile Charts
  - List all users

MongoDB Compass Dev - localhost:27017/Venues.Restaurants

My Cluster

4 DBS 6 COLLECTIONS

filter

- Venues
  - Bars
  - Clubs
  - Hotels
  - Restaurants
- admin
- local
- test

localhost:27017 STANDALONE

MongoDB 3.6.0 Enterprise

Venues.Restaurants

DOCUMENTS	TOTAL SIZE	AVG. SIZE	INDEXES	TOTAL SIZE	AVG. SIZE
25.4k	10.1MB	419B	1	240.0KB	240.0KB

Documents Schema Explain Plan Indexes Validation

name option renders the tab name

FILTER { field: 'value' } OPTIONS FIND

INSERT DOCUMENT VIEW LIST TABLE

Displaying documents 1 - 20 of 25356

component option renders the component in the frame

```
_id: ObjectId("59f7306e037d92ff4dc2a272")
borough: "Brooklyn"
cuisine: "American"
> grades: Array
name: "C & C Catering Service"

_id: ObjectId("59f7306e037d92ff4dc2a273")
> address: Object
borough: "Brooklyn"
cuisine: "Chinese"
> grades: Array
name: "May May Kitchen"
restaurant_id: "40358429"

_id: ObjectId("59f7306e037d92ff4dc2a274")
> address: Object
borough: "Manhattan"
cuisine: "American"
name: "1 East 66Th Street Kitchen"
restaurant_id: "40359480"

_id: ObjectId("59f7306e037d92ff4dc2a275")
> address: Object
cuisine: "Jewish/Kosher"
> grades: Array
name: "Seuda Foods"
```

Collection.Tab

- **Purpose:**
  - Large collection-level features (e.g. CRUD, Explain Plan, or schema analysis)

- **Potential Plugin:**
  - Object size histogram of schema sample

MongoDB Compass Dev - localhost:27017/Venues.Restaurants

My Cluster

4 DBS 6 COLLECTIONS

filter

- Venues
  - Bars
  - Clubs
  - Hotels
  - Restaurants
- admin
- local
- test

localhost:27017 STANDALONE

MongoDB 3.6.0 Enterprise

Venues.Restaurants

DOCUMENTS	TOTAL SIZE	AVG. SIZE	INDEXES	TOTAL SIZE	AVG. SIZE
25.4k	10.1MB	419B	1	240.0KB	240.0KB

Documents Schema Explain Plan Indexes Validation

FILTER { field: 'value' } OPTIONS FIND

INSERT DOCUMENT VIEW LIST TABLE

Displaying documents 1 - 20 of 25356

Restaurants

	_id ObjectId	borough String	cuisine String	grades Array	name String
1	59f7306e037d92ff4dc2a272	"Brooklyn"	"American"	[] 4 elements	"C & C Catering Service"
2	59f7306e037d92ff4dc2a273	"Brooklyn"	"Chinese"	[] 5 elements	"May May Kitchen"
3	59f7306e037d92ff4dc2a274	"Manhattan"	"American"	No field	"1 East 66Th Street Kit
4	59f7306e037d92ff4dc2a275	No field	"Jewish/Kosher"	[] 5 elements	"Seuda Foods"
5	59f7306e037d92ff4dc2a276	"Brooklyn"	"Ice Cream, Gelato, Yogurt,	[] 5 elements	"Carvel Ice Cream"
6	59f7306e037d92ff4dc2a277	"Queens"	"Ice Cream, Gelato, Yogurt,	[] 3 elements	"Carvel Ice Cream"
7	59f7306e037d92ff4dc2a278	"Brooklyn"	"Delicatessen"	[] 3 elements	"Nordic Delicacies"
8	59f7306e037d92ff4dc2a279	"Manhattan"	"American"	[] 5 elements	"Glorious Food"
9	59f7306e037d92ff4dc2a27a	"Brooklyn"	"American"	[] 4 elements	"The Movable Feast"
10	59f7306e037d92ff4dc2a27b	"Queens"	"Delicatessen"	[] 4 elements	"Sal'S Deli"
11	59f7306e037d92ff4dc2a27c	"Manhattan"	"Delicatessen"	[] 4 elements	"Bully'S Deli"
12	59f7306e037d92ff4dc2a27d	"Queens"	"Delicatessen"	[] 4 elements	"Steve Chu'S Deli & Gro
13	59f7306e037d92ff4dc2a27e	"Manhattan"	"Chicken"	[] 6 elements	"Harriet'S Kitchen"
14	59f7306e037d92ff4dc2a27f	"Manhattan"	"American"	[] 4 elements	"P & S Deli Grocery"
15	59f7306e037d92ff4dc2a280	"Manhattan"	"American"	[] 4 elements	"Angelika Film Center"

CollectionHUD.Item

- **Purpose:**
  - Display statistical information relevant to a collection

- **UI Considerations:**
  - Should not distract users from the main content of the collection body
  - Should be text-based

- **Potential Plugin:**
  - Mongotop / server statistics

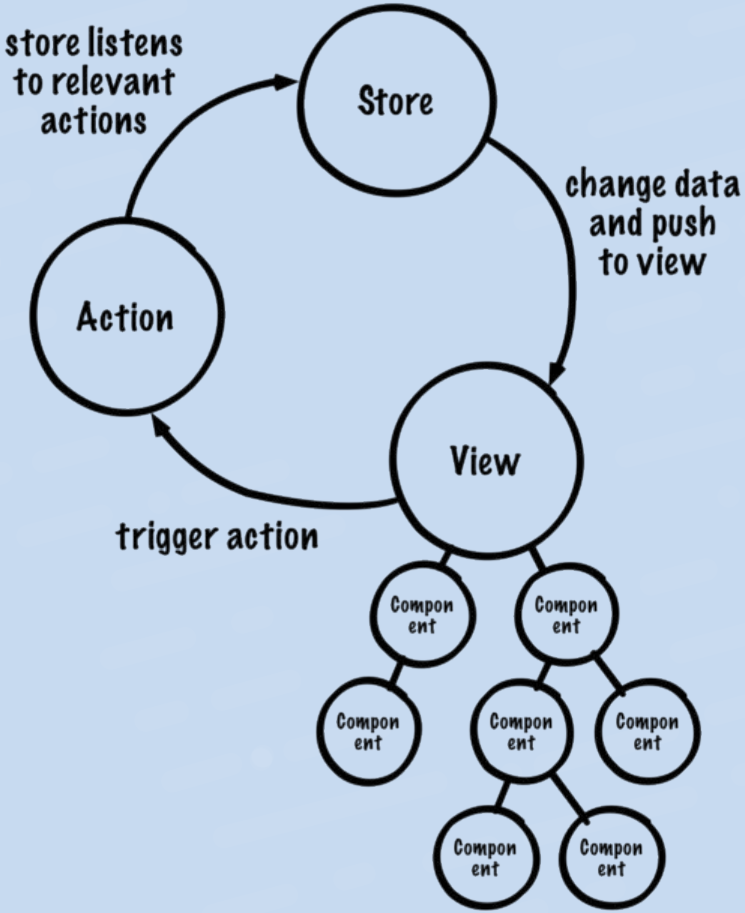
What makes up a plugin?

Compass and any plugins are React applications.

React

React is a framework made up of Stores, Components, and Actions.

- **Stores** are “smart”, they keep track of state information and pass it to the Components.
- **Components** are “dumb”, they just receive data from the store and render it.
- **Actions** are how components communicate with the store to trigger changes.



Actions

- **Events** triggered when a user performs an action.
- Can be button clicks, link clicks, text entry, etc.
- Actions are sent from Components and listened to by Stores.

Components

- Components make up the **User Interface**.
- This is where you use `JSX` to define what the plugin should look like.
- Styles are imported from a local `.less` file with the same name as the component.

Stores

- **Reflux/Redux Stores** that hold the application state.
- Each plugin can listen to other stores/have its state changes listened to by others.
- Flow of data:
  - Actions should be listened to by stores
  - Stores should be subscribed to by components

The AppRegistry

- **Holds all Compass + plugin actions, components, and stores**
  - All plugins must register with the AppRegistry.
- Information flows from Compass to plugins and back through the AppRegistry.
- It is available to all Plugins globally by calling **`global.hadronApp.appRegistry`**.

Lifecycle Hooks

How do Compass and my plugin become aware of each other?

`activate()`: Tell Compass about the plugin

- Compass will look in the plugins directory.
  - **`~/.mongodb/compass/plugins`** on OSX
- It will call the **`activate`** method in the root **`index.js`**.
- The plugin should register its actions, components, and stores with the AppRegistry in this method.
- **This is where you specify what role you want your plugin to occupy.**

whoami/src/index.js

```
5  const ROLE = {
6    name: 'Whoami',
7    component: WhoamiPlugin
8  };
9
10 /**
11  * Activate all the components in the Whoami package.
12  * @param {Object} appRegistry - The Hadron appRegistry
13  **/
14 function activate(appRegistry) {
15   // Register the WhoamiPlugin as a role in Compass
16   appRegistry.registerRole('Header.Item', ROLE);
17   appRegistry.registerAction('Whoami.Actions', WhoamiActions);
18   appRegistry.registerStore('Whoami.Store', WhoamiStore);
19 }
```

`onActivated(appReg)`: Tell plugin about Compass

- When all plugin registration is completed, the **`onActivated`** method is called on any **registered** store**.**
- In this method, the plugin can get and listen to any stores in Compass or another plugin.
- Guarantees everything has been registered before any plugin tries to get anything from the registry.

```
31
32  /**
33   * This method is called when all plugins are activated. You can register
34   * listeners to other plugins' stores here, e.g.
35   *
36   * appRegistry.getStore('OtherPlugin.Store').listen(this.otherStoreChanged.bind(this));
37   *
38   * If this plugin does not depend on other stores, you can delete the method.
39   *
40   * @param {Object} appRegistry - app registry containing all stores and components
41   */
42  onActivated(appRegistry) {
43    appRegistry.on('data-service-intialized', (dataService) => {
44        dataService.command(...)
45    });
46
47    appRegistry.on('collection-changed', (namespace) => {
48        this.setCollection(namespace.collection);
49    });
50
51    ...
52
```

The AppRegistry will emit events

- `data-service-connected`
- `collection-changed`
- `database-changed`
- `query-changed`

… or any other events you want to add!

My First Compass Plugin

Who is the current user?

- Since it’s global to the Compass instance, it can take the role **Header.Item**
- The **`connectionStatus`** DB command lists authenticated users and their roles.

Final Product

MongoDB Compass Beta - localhost:27017

localhost:27017 STANDALONE User: Role MongoDB 3.6.0 Enterprise

Databases Performance

CREATE DATABASE

Database Name	Storage Size	Collections	Indexes
Venues	4.5MB	1	1
admin	48.0KB	0	3
config	4.0KB	0	2
local	36.0KB	1	1

Final Product

MongoDB Compass Beta - localhost:27017

localhost:27017 STANDALONE

User: Role

MongoDB 3.6.0 Enterprise

Databases Performance

CREATE DATABASE

Database Name	Storage Size	Collections	Indexes
Venues	4.5MB	1	1
admin	48.0KB	0	3
config	4.0KB	0	2
local	36.0KB	1	1

Step 0: Get the Template

~/.mongodb/compass/plugins $

```
> npm install -g khaos
> khaos create mongodb-js/compass-plugin ./whoami
> git init && git add . && git commit -m “init”
> npm install
```

Compass **without** plugin

MongoDB Compass Dev - localhost:27017

localhost:27017 STANDALONE MongoDB 3.6.0 Enterprise

Databases Performance

CREATE DATABASE

Database Name	Storage Size	Collections	Indexes
Venues	4.5MB	1	1
admin	48.0KB	0	3
local	36.0KB	1	1

Compass with **unchanged** plugin

MongoDB Compass Beta - localhost:27017

localhost:27017 STANDALONE

The current status is: enabled

MongoDB 3.6.0 Enterprise

Databases Performance

CREATE DATABASE

Database Name	Storage Size	Collections	Indexes
Venues	4.5MB	1	1
admin	48.0KB	0	3
local	36.0KB	1	1

Plugin running **standalone**

Webpack App

Whoami Plugin

A plugin to indicate the current user.

The current status is: `enabled`

TOGGLE STATUS

Elements Console Sources Network Performance Memory Application Security Audits

top ▼ Filter Info ▼

> |

Step 1: The Store

`src/stores/store.js:`

- We set the initial state to empty.
- **`onActivated`:** when the DataService is connected, we want to run the **`connectionStatus`** command.
- When the command returns, we set the state to the results.

Initialize State

```
getInitialState() {
  return {
    user: null, role: null
  };
}
```

Call DB command on activation

```
appRegistry.on('data-service-connected', (error, dataService) => {
 if (!error) {
   dataService.command(
     'admin',
     {connectionStatus : 1},
     (err, res) => {
       if (!err && res.authInfo && res.authInfo.authenticatedUsers) {
         this.setState({
           user: res.authInfo.authenticatedUsers[0].user,
           role: res.authInfo.authenticatedUserRoles[0].role
         })
       } else {
         this.setState({user: null, role: null});
       }
     }
   );
 }
});
```

Step 2: The Component

`src/components/whoami/whoami.jsx`

- The store state will be passed to the component as **`this.props.user`** and **`this.props.role`**
- Component must render the username and role
- If not provided, render “no user”

Define and Set Default Props

```
static propTypes = {
 user: PropTypes.string.isRequired,
 role: PropTypes.string.isRequired
};

static defaultProps = {
 user: '', role: ''
};
```

Render Props

```
render() {
  if (!this.props.user) {
    return (
      <div className={classnames(styles.root)}>
        <FontAwesome name="user-times"/>
        No User
      </div>
    );
  }
  return (
    <div className={classnames(styles.root)}>
      <FontAwesome name="user"/>
      <i> {this.props.user}:{this.props.role}</i>
    </div>
  );
}
```

Step 3: The Actions

(not used in this plugin)

Step 4: Styles

src/components/whoami/whoami.less

- Every component has a .less file with the same name located in the same directory
- In this case just want to remove the colors.

```
.root {
  background: #0e83cd;
  color: @pw;
  padding: 2.4rem;
}
```

Run It Standalone!

Webpack App

No User

Elements Console Sources Network Performance Memory Application Security Audits

top Filter Info

>

Run Compass!

MongoDB Compass Beta - localhost:27017

localhost:27017 STANDALONE

anna: root

MongoDB 3.6.0 Enterprise

Databases Performance

CREATE DATABASE

Database Name	Storage Size	Collections	Indexes
Venues	4.5MB	1	1
admin	48.0KB	0	3
config	4.0KB	0	2
local	36.0KB	1	1

MongoDB Compass Beta - localhost:27017

localhost:27017 STANDALONE

anna: root

MongoDB 3.6.0 Enterprise

A PLUGIN!

Databases Performance

CREATE DATABASE

Database Name	Storage Size	Collections	Indexes
Venues	4.5MB	1	1
admin	48.0KB	0	3
config	4.0KB	0	2
local	36.0KB	1	1

Writing a Plugin is Easy!

- **We are here to help!**
  - Email *team-compass@mongodb.com*
- **All plugin documentation:**
  - *https://docs.mongodb.com/compass/master/plugins/creating-compass-plugins*
- **The “whoami” plugin + these slides are on GitHub**
  - *https://github.com/aherlihy/compass-plugins-talk*

Security Restrictions

- **Accessing network resources** over any protocol outside of the DB connection. This includes network access via NodeJS or DOM APIs such as XMLHttpRequest.
- **Accessing the filesystem** outside of DOM APIs such as IndexedDB.
- **Spawning child processes.**

Future Plans

- We encourage everyone to share their plugins!
- MongoDB can provide marketing and support
- When there are enough plugins there will be a plugin marketplace
- Until then, MongoDB can list the plugins the same way we list community drivers

MONGODB.local

TEL AVIV

THANK YOU!

#MDBlocal

MONGODB.local
TEL AVIV

THANKS FOR COMING!

Anna Herlihy

anna@mongodb.com

@annaisworking

#MDBlocal