

Hey there,

This is a quick tutorial how to present my application.
It will help you adding the Watson User Modeling Service Live.

0. Prepare

1. Upload an alone standing application to Bluemix
2. Show how to use the alone standing application
3. Add the User Modeling Service to it
4. Activate the User Modeling Service
5. Upload the application
6. Show how to use the new application

If you do not know what my application does, check out my other slideshare presentation

0. Prepare

Instructions

There are some steps you have to do before you can present the application.

- Go to github.com/JDihlmann/moodlocator and follow the steps in the section presentation
- If you are presenting moodlocator you can start your presentation at any point you want, but you have to do all steps behind the scene if you wouldn't show it.

Read the GitHub instructions until point 5

At this point I'm assuming you read the GitHub instructions and set up everything

- Before we upload our alone standing application we have to make some changes to it. The default of the moodlocator application is set to show the Watson User Modeling Service.
- To change this default value we have to open the code

0. Prepare

Open style.css

Open the style.css file with a text editor you will find it in

/ node / public / stylesheets / style.css

Scroll to the bottom of the page

You will find something that looks like that, delete the text in line 449 and 452. I marked it red.

Before

```
448 /* For Presentation activate oder deactivate the Section below */  
449 /*  
450 #footer{ visibility: hidden; }  
451 #map_canvas{ height: 100%; }  
452 */
```

After

```
448 /* For Presentation activate oder deactivate the Section below */  
449  
450 #footer{ visibility: hidden; }  
451 #map_canvas{ height: 100%; }  
452 
```

Save your changes. Perfect to understand what it does, it enables or disables the white Watson User Modeling bar at the bottom of the page.

0. Prepare

Open app.js

Open the app.js file with a text editor you will find it in

/ node / app.js

Scroll to line 189

You will find something that looks like that, add this bracket „ /* “ in line 190 and this bracket „ */ “ in line 211.
I marked the position red where to add them.

0. Prepare

Before

```
189 //For presentation start commenting here
190
191 //Create a profile request with the text and the https options and call it
192 create_profile_request(profile_options,textToWatson)(function(error,profile_string) {
193     if (error) console.log(error.message);
194     else {
195         //Parse the Profile and format it
196         var profile_json = JSON.parse(profile_string);
197         var flat_traits = flatten.flat(profile_json.tree);
198
199         //Sent Watson Data to Client
200         var watsonObj = {
201             id: msg.id,
202             watsonJSON: flat_traits
203         }
204
205
206         //Send Watson Data to Client
207         socket.emit('watson message', watsonObj);
208         //console.log(flat_traits);
209     }
210 });
211 //For presentation stop commenting here
212 }
```

After

```
189 //For presentation start commenting here
190 /*
191 //Create a profile request with the text and the https options and call it
192 create_profile_request(profile_options,textToWatson)(function(error,profile_string) {
193     if (error) console.log(error.message);
194     else {
195         //Parse the Profile and format it
196         var profile_json = JSON.parse(profile_string);
197         var flat_traits = flatten.flat(profile_json.tree);
198
199         //Sent Watson Data to Client
200         var watsonObj = {
201             id: msg.id,
202             watsonJSON: flat_traits
203         }
204
205
206         //Send Watson Data to Client
207         socket.emit('watson message', watsonObj);
208         //console.log(flat_traits);
209     }
210 });
211 */
212 }
```

0. Prepare

Stay in app.js

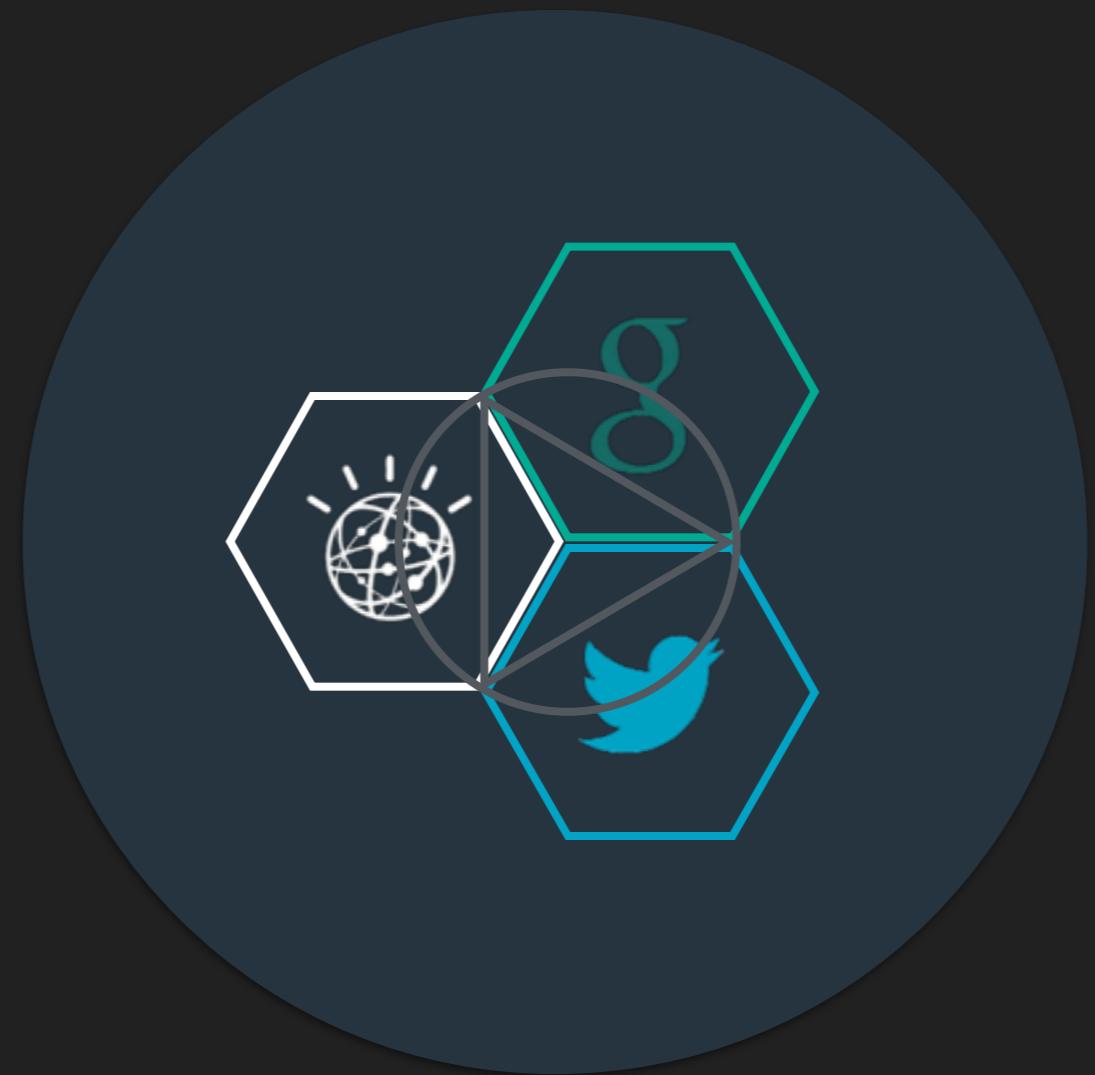
Scroll to line 238

You will find something that looks like that, start in line 238 and delete everything what comes after it, till the end of the document.

Save your changes

Okay we prepared our application now lets start presenting

```
238 //For presentation start delete this section
239
240 //Function you will find in the User Modeling documentation
241 var create_profile_request = function(options,content) {
242     return function /*function*/ callback) {
243         // create the post data to send to the User Modeling service
244         var post_data = {
245             'contentItems' : [
246                 'userid' : 'dummy',
247                 'id' : 'dummyUuid',
248                 'sourceid' : 'freetext',
249                 'contenttype' : 'text/plain',
250                 'language' : 'en',
251                 'content': content
252             ]
253         };
254         // Create a request to POST to the User Modeling service
255         var profile_req = https.request(options, function(result) {
256             result.setEncoding('utf-8');
257             var response_string = '';
258
259             result.on('data', function(chunk) {
260                 response_string += chunk;
261             });
262
263             result.on('end', function() {
264
265                 if (result.statusCode != 200) {
266                     var error = JSON.parse(response_string);
267                     callback({message: error.user_message}, null);
268                 } else {
269                     callback(null, response_string);
270                 }
271             });
272
273             profile_req.on('error', function(e) {
274                 callback(e,null);
275             });
276
277             profile_req.write(JSON.stringify(post_data));
278             profile_req.end();
279         }
280     };
281
282 //For presentation stop deleting this section
```

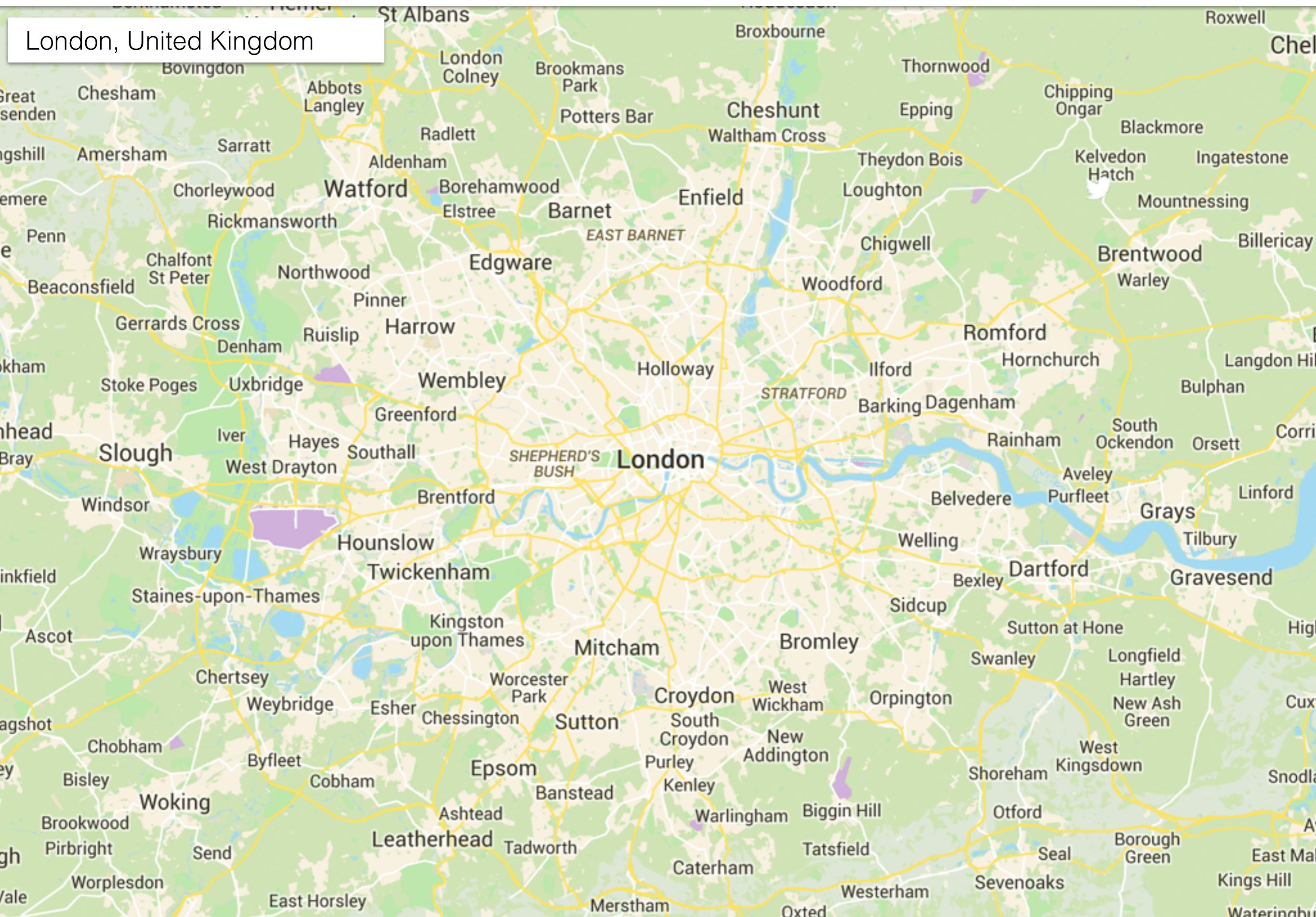


Bluemix Demo

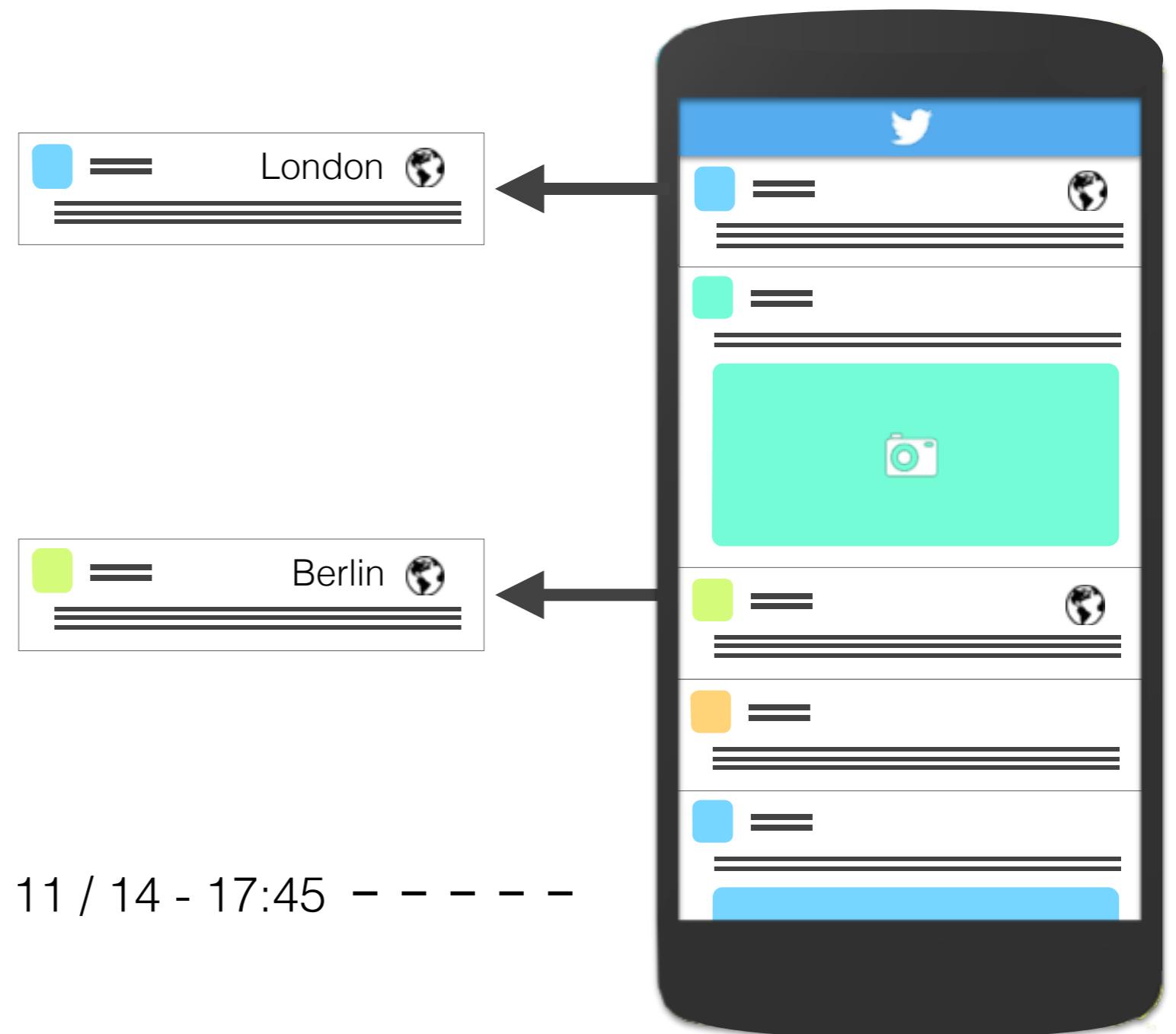
MoodLocator

Google Maps API

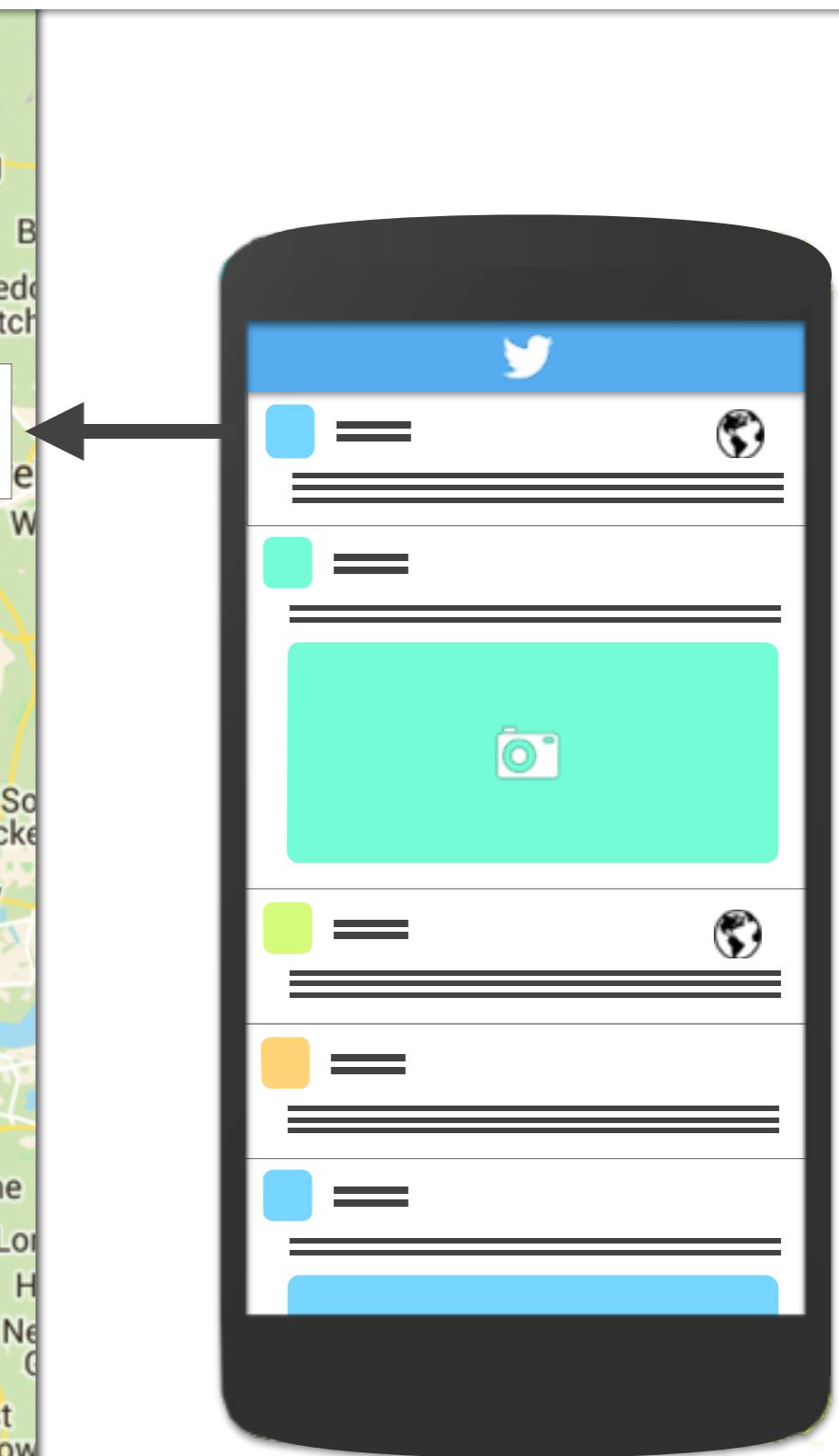
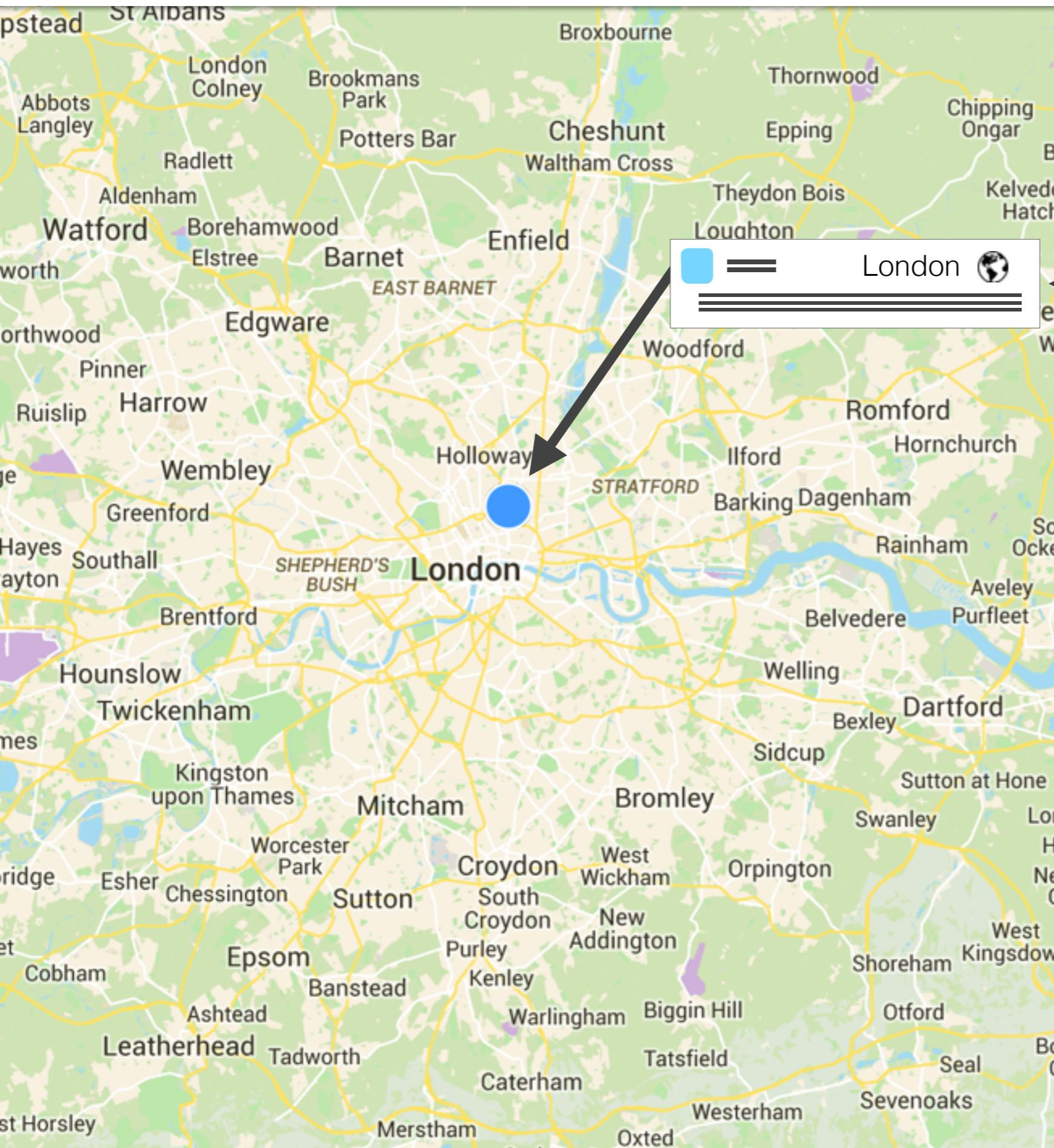
London, United Kingdom



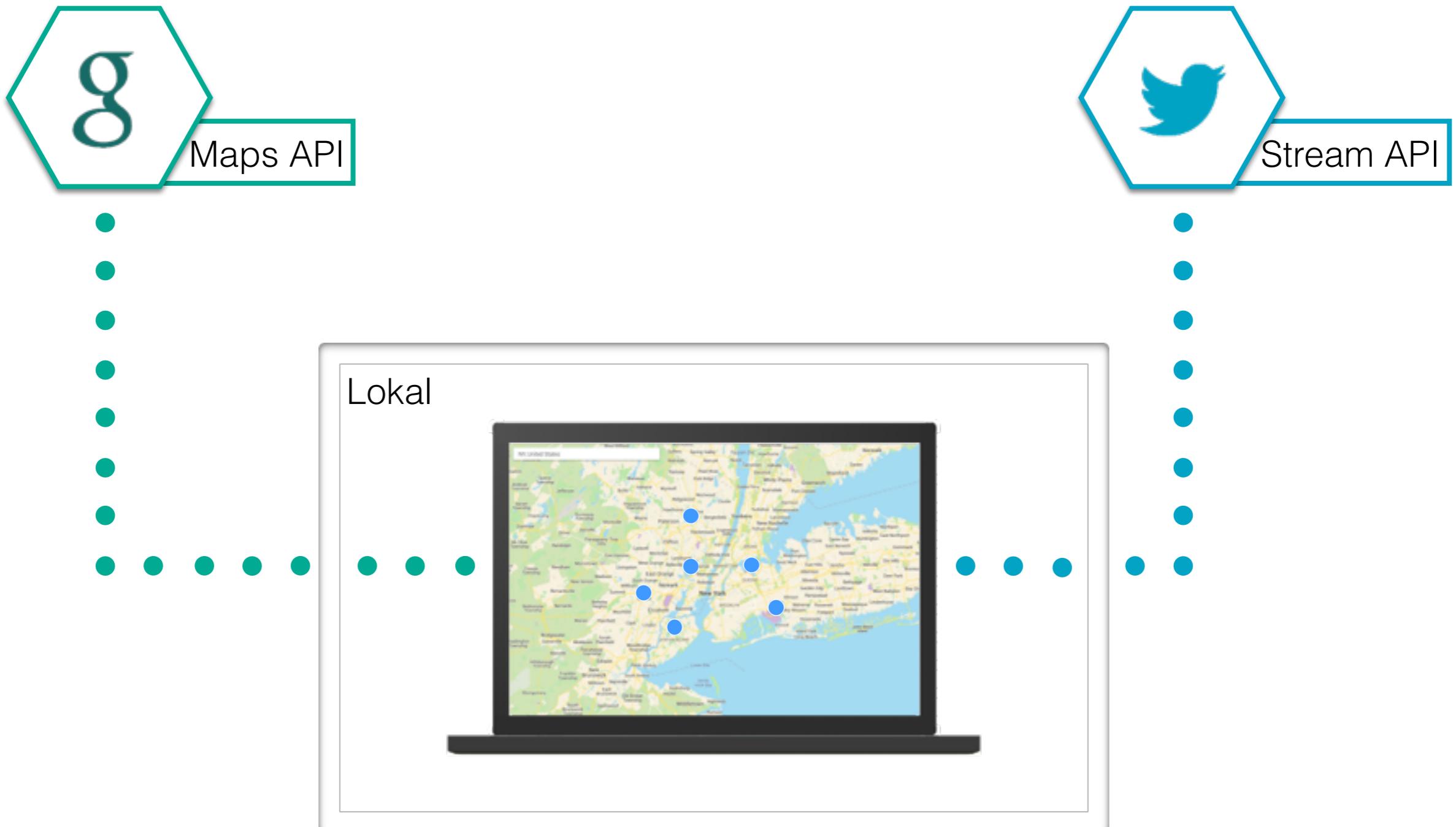
Twitter Stream API



Google & Twitter



Structure





Live Demo

Bluemix Homepage

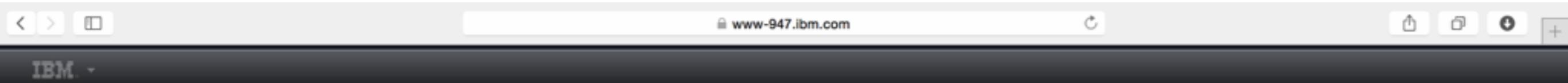
The screenshot shows the IBM Bluemix homepage. At the top, there's a navigation bar with icons for back, forward, and search, followed by the URL 'ace.ng.bluemix.net'. The main header reads 'IBM® Watson™ Services' with a subtitle 'Rapidly prototype and build cognitive applications in the cloud'. Below this are two buttons: 'Sign up for free' and 'Learn more'. A large blue globe graphic is visible in the background. At the bottom, there's a section titled 'Connect with Bluemix:' with icons for Twitter, YouTube, GitHub, Facebook, and LinkedIn. A small screenshot of the Bluemix dashboard is also shown at the very bottom.

Some of this steps you have already done, you can show them to your listeners, but if you don't want to you want have to do them again.

The future is yours.

If you're a developer, you now have a hand in creating the future. Bluemix offers you all the instant services, runtimes, and infrastructure you need to push your ideas into the present.

Login



One key, many possibilities.

Your IBM id provides access to services, communities, support, online purchasing, and much more.

[Create IBM id](#)

Sign in

[Help and FAQ](#)

[Forgot password?](#)

Dashboard

The screenshot shows the IBM Bluemix dashboard at ace.ng.bluemix.net. The top navigation bar includes links for DASHBOARD, CATALOG, PRICING, DOCS, and COMMUNITY, along with a REGION: US South dropdown set to 215. The left sidebar shows the organization 'jdihlman@de.ibm.com' and a space named 'dev'. Under 'dev', there are 0 applications and 0 services. The main dashboard features three circular metrics: 0B out of 1GB of Memory, 0 APP HEALTH issues, and 0/4 SERVICES. Below these are sections for 'Applications' and 'Services', each with a large '+' button and a 'CREATE AN APP' or 'ADD A SERVICE' link.

ace.ng.bluemix.net

IBM Bluemix

DASHBOARD CATALOG PRICING DOCS COMMUNITY REGION: US South > 215

ORG: jdihlman@de.ibm.com

+ Create a Space

dev

APPS (0)

SERVICES (0)

0B out of 1GB of Memory

APP HEALTH

0/4 SERVICES

CREATE AN APP

ADD A SERVICE

Create an application

Catalog

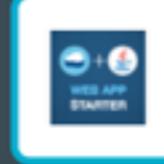
ace.ng.bluemix.net

IBM Bluemix DASHBOARD CATALOG PRICING DOCS COMMUNITY REGION: US South > 215

jdihlman@de.l... Type here to begin

Starters // Choose a package of sample code and services, or start from scratch

Boilerplates
Get started with a new app, now

| | | | | | |
|---|--|---|---|--|---|
|  Internet of Things Foun... IBM |  Java Cache Web Starter IBM |  Java Cloudant Web Sta... IBM |  Java DB Web Starter IBM |  Mobile Cloud IBM |  Node.js Cache Web St... IBM |
|  Node.js Cloudant DB ... IBM |  User Modeling Java We... IBM |  User Modeling Node.js ... IBM |  User Modeling Ruby W... IBM |  Node-RED Starter Community |  Vaadin Rich Web Starter Community |

Runtimes
Run an app in the language of your choice

| | | | | |
|--|--|---|--|--|
|  Liberty for Java™ IBM |  SDK for Node.js™ IBM |  Ruby on Rails Community |  Ruby Sinatra Community |  Bring Your Buildpack Community |
|--|--|---|--|--|

Node.js

ace.ng.bluemix.net

DASHBOARD CATALOG PRICING DOCS COMMUNITY REGION: US South > 215

jdihlman@de.l... Type here to begin

Category

- Watson
- Mobile
- DevOps
- Web and Application
- Integration
- Data Management
- Big Data
- Security
- Business Analytics
- Internet of Things

Starters // Choose a package of sample code and services, or start from scratch

Boilerplates

Get started with a new app, now

Internet of Things Foun... Java Cache Web Starter Java Cloudant Web Sta... Mobile Cloud Node.js Cache Web St...

IBM IBM IBM IBM

Node.js Cloudant DB ... User Modeling Java We... User Modeling Node.js ... User Modeling Ruby W... Node-RED Starter Vaadin Rich Web Starter

IBM IBM IBM IBM Community Community

Runtimes

Run an app in the language of your choice

.java liberty .js .rb on rails .rb Sinatra Bring Your Buildpack

Liberty for Java™ SDK for Node.js™ Ruby on Rails Ruby Sinatra Bring Your Buildpack

IBM Community Community Community

View More

Choose Node.js



Create App

The screenshot shows the IBM Bluemix Catalog interface. The top navigation bar includes links for DASHBOARD, CATALOG (which is underlined in green), PRICING, DOCS, COMMUNITY, REGION: US South, and a user icon. A search bar at the top right contains the text "ace.ng.bluemix.net". On the left, there's a sidebar for the "SDK for Node.js™" package, version 1.1, type Application, with a "VIEW DOCS" button. The main catalog area displays a card for the "SDK for Node.js™" package, which includes a description: "Develop, deploy, and scale server-side JavaScript® apps with ease. The IBM SDK for Node.js™ provides enhanced performance, security, and serviceability." Below this is a "Pick a plan" section with a table comparing "Default" and "Standard" plans based on CPU, RAM, and VGB-Hour. A large callout bubble points to the "Default" plan with the text "Enter a unique name". To the right, configuration fields are shown: "Start with a runtime:" (Name: moodlocator, Host: moodlocator.mybluemix.net), "Selected Plan:" (Default), and a "CREATE" button.

ace.ng.bluemix.net

IBM Bluemix

DASHBOARD CATALOG PRICING DOCS COMMUNITY REGION: US South 215

Back to Starters

.js

SDK for Node.js™

IBM

VERSION 1.1

TYPE Application

VIEW DOCS

Develop, deploy, and scale server-side JavaScript® apps with ease. The IBM SDK for Node.js™ provides enhanced performance, security, and serviceability.

Pick a plan

Monthly prices shown are for country or region: United States

| Plan | Features | CPU | RAM | VGB-Hour |
|-----------|----------|-----|------|----------|
| ✓ Default | Thick | 1 | 1 GB | 100 |
| Standard | Thin | 2 | 2 GB | 200 |

Enter a unique name

TERMS

Start with a runtime:

Name: moodlocator

Host: moodlocator.mybluemix.net

Selected Plan:

Default

CREATE

App Info

The screenshot shows the IBM Bluemix App Info interface for the 'moodlocator' application. The top navigation bar includes links for DASHBOARD, CATALOG, PRICING, DOCS, COMMUNITY, REGION: US South, and a notifications count of 215. The left sidebar provides navigation options like Back to Dashboard, Overview, SDK for Node.js™, Files and Logs, and SERVICES. The main content area displays the app's name 'moodlocator' with a robot icon, a 'VIEW QUICK START' section with a Node.js logo, memory settings (MEMORY QUOTA: 128 MB per Instance), available memory (1,000GB), and an 'APP HEALTH' status (Stopped). The 'ACTIVITY LOG' section shows two entries from 26.11.14 at 06:18: one for updating routes and another for creating the app. A button at the bottom right allows estimating the cost of the app.

ace.ng.bluemix.net

DASHBOARD CATALOG PRICING DOCS COMMUNITY REGION: US South > 215

Back to Dashboard

moodlocator

Overview >

SDK for Node.js™

Files and Logs

SERVICES

.js

Check out the SDK for Node.js™ quick start to get started with your new app.

VIEW QUICK START

MEMORY QUOTA: 128 (MB per Instance)

AVAILABLE MEMORY: 1,000GB

RESET SAVE

APP HEALTH

Started

ACTIVITY LOG

26.11.14 06:18 jdihlman@de.ibm.com updated moodlocator app
• changed routes

26.11.14 06:18 jdihlman@de.ibm.com created moodlocator app

Estimate the cost of this app

1. Upload an alone standing application to Bluemix

Go back to GitHub and take a look at section 6

Quick Start

The screenshot shows the IBM Bluemix dashboard for the 'moodlocator' application. The application icon is a blue robot head. The routes are listed as 'moodlocator.mybluemix.net'. The configuration shows 1 instance, a memory quota of 128 MB, and available memory of 896,0 MB. A large callout box contains the text: 'We will follow these steps'. Below it, a message says: 'You don't have any services yet. Add a service from the catalog or bind an existing service from your dashboard to start creating!' On the right, a 'QUICK START AND NOTIFICATION' panel provides step-by-step instructions for getting started with Node.js:

1. Install the cf command-line tool.
2. Download the starter application package.
3. Extract the package and cd to it.
4. Connect to Bluemix:

```
cf api https://api.ng.bluemix.net
```
5. Log into Bluemix:

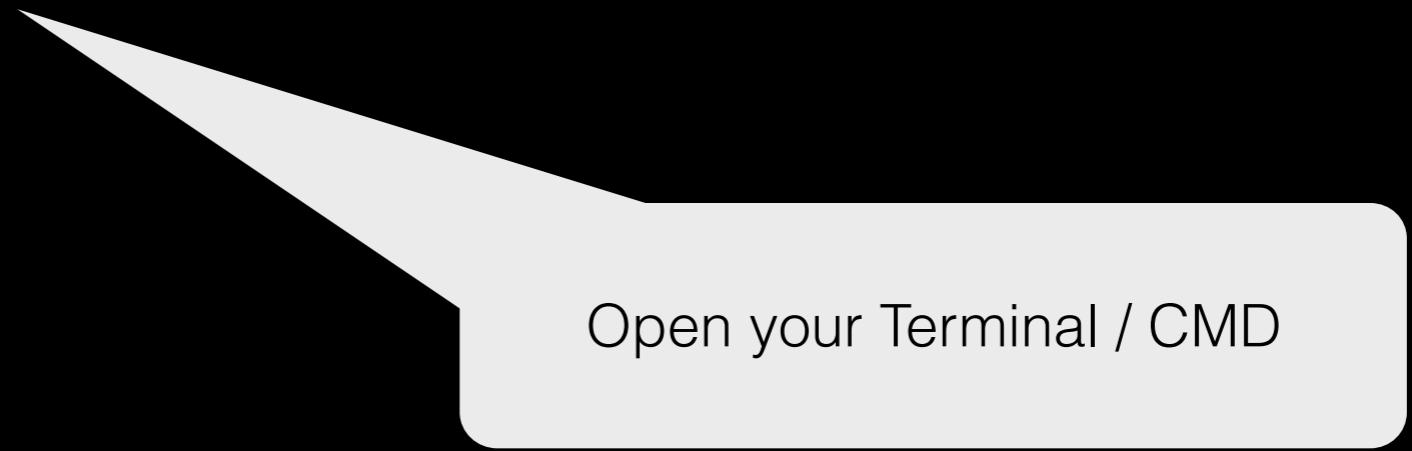
```
cf login -u jdihlman@de.ibm.com  
cf target -o jdihlman@de.ibm.com -s
```
6. Deploy your app:

```
cf push moodlocator
```
7. Access your app:moodlocator.mybluemix.net

At the bottom right, there are 'View Docs' and '06:18' buttons.

Terminal

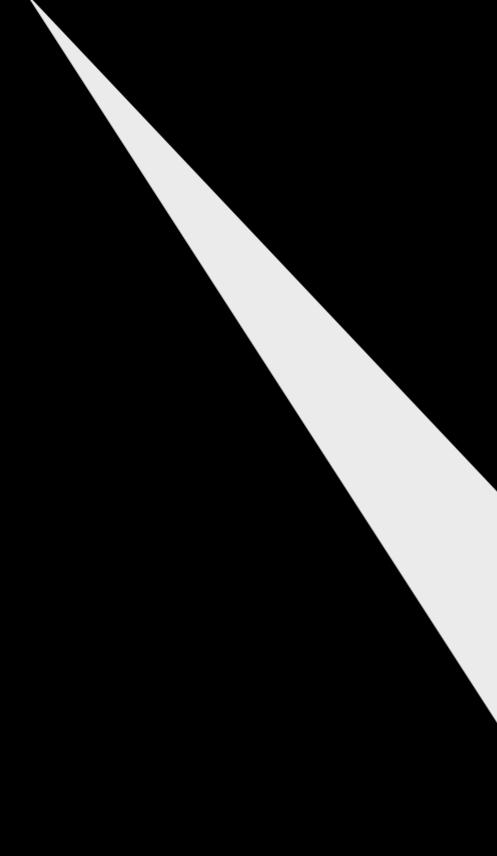
Dihlmann:MoodlocaterPr Dihlmann\$ █



Open your Terminal / CMD

cd MoodLocator

```
Dihlmann:~ Dihlmann$ cd Desktop/  
Dihlmann:Desktop Dihlmann$ cd MoodlocaterPr/  
Dihlmann:MoodlocaterPr Dihlmann$ █
```



Instead of „MoodlocaterPr“ write
„node“

Connect to Bluemix

```
Dihlmann:~ Dihlmann$ cd Desktop/  
Dihlmann:Desktop Dihlmann$ cd MoodlocatorPr/  
Dihlmann:MoodlocatorPr Dihlmann$ cf api https://api.ng.bluemix.net  
Setting api endpoint to https://api.ng.bluemix.net...  
OK
```

```
API endpoint: https://api.ng.bluemix.net (API version: 2.14.0)  
User: jdihlman@de.ibm.com  
Org: jdihlman@de.ibm.com  
Space: dev  
Dihlmann:MoodlocatorPr Dihlmann$
```

Copy & Paste

Login

```
Dihlmann:~ Dihlmann$ cd Desktop/  
Dihlmann:Desktop Dihlmann$ cd MoodlocaterPr/  
Dihlmann:MoodlocaterPr Dihlmann$ cf api https://api.ng.bluemix.net  
Setting api endpoint to https://api.ng.bluemix.net...  
OK
```

```
API endpoint: https://api.ng.bluemix.net (API version: 2.14.0)  
User: jdihlman@de.ibm.com  
Org: jdihlman@de.ibm.com  
Space: dev  
Dihlmann:MoodlocaterPr Dihlmann$ cf login -u jdihlman@de.ibm.com  
API endpoint: https://api.ng.bluemix.net
```

```
Password>  
Authenticating...  
OK
```

```
Targeted org jdihlman@de.ibm.com
```

```
Targeted space dev
```

```
API endpoint: https://api.ng.bluemix.net (API version: 2.14.0)  
User: jdihlman@de.ibm.com  
Org: jdihlman@de.ibm.com  
Space: dev  
Dihlmann:MoodlocaterPr Dihlmann$
```

Enter the password

cf push

```
Dihlmann:~ Dihlmann$ cd Desktop/  
Dihlmann:Desktop Dihlmann$ cd MoodlocatorPr/  
Dihlmann:MoodlocatorPr Dihlmann$ cf api https://api.ng.bluemix.net  
Setting api endpoint to https://api.ng.bluemix.net...  
OK
```

```
API endpoint: https://api.ng.bluemix.net (API version: 2.14.0)
```

```
User: jdihlman@de.ibm.com
```

```
Org: jdihlman@de.ibm.com
```

```
Space: dev
```

```
Dihlmann:MoodlocatorPr Dihlmann$ cf login -u jdihlman@de.ibm.com
```

```
API endpoint: https://api.ng.bluemix.net
```

```
Password>
```

```
Authenticating...
```

```
OK
```

```
Targeted org jdihlman@de.ibm.com
```

```
Targeted space dev
```

Instead of „moodlocator“ enter
your application name

```
API endpoint: https://api.ng.bluemix.net (API version: 2.14.0)
```

```
User: jdihlman@de.ibm.com
```

```
Org: jdihlman@de.ibm.com
```

```
Space: dev
```

```
Dihlmann:MoodlocatorPr Dihlmann$ cf push moodlocator
```

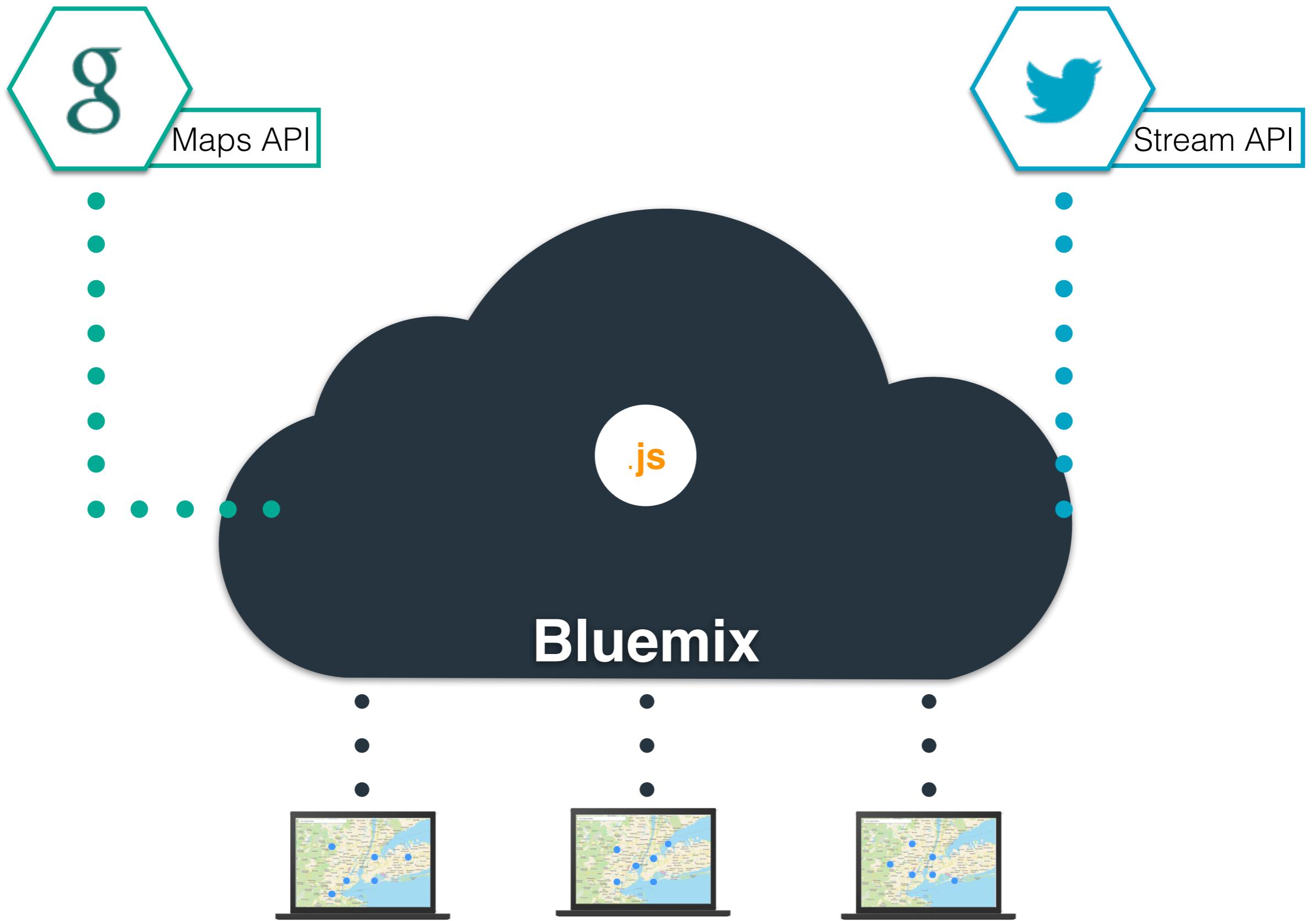
```
Using manifest file /Users/Dihlmann/Desktop/MoodlocatorPr/manifest.yml
```

```
Updating app moodlocator in org jdihlman@de.ibm.com / space dev as jdihlman@de.ibm.com...
```

```
OK
```

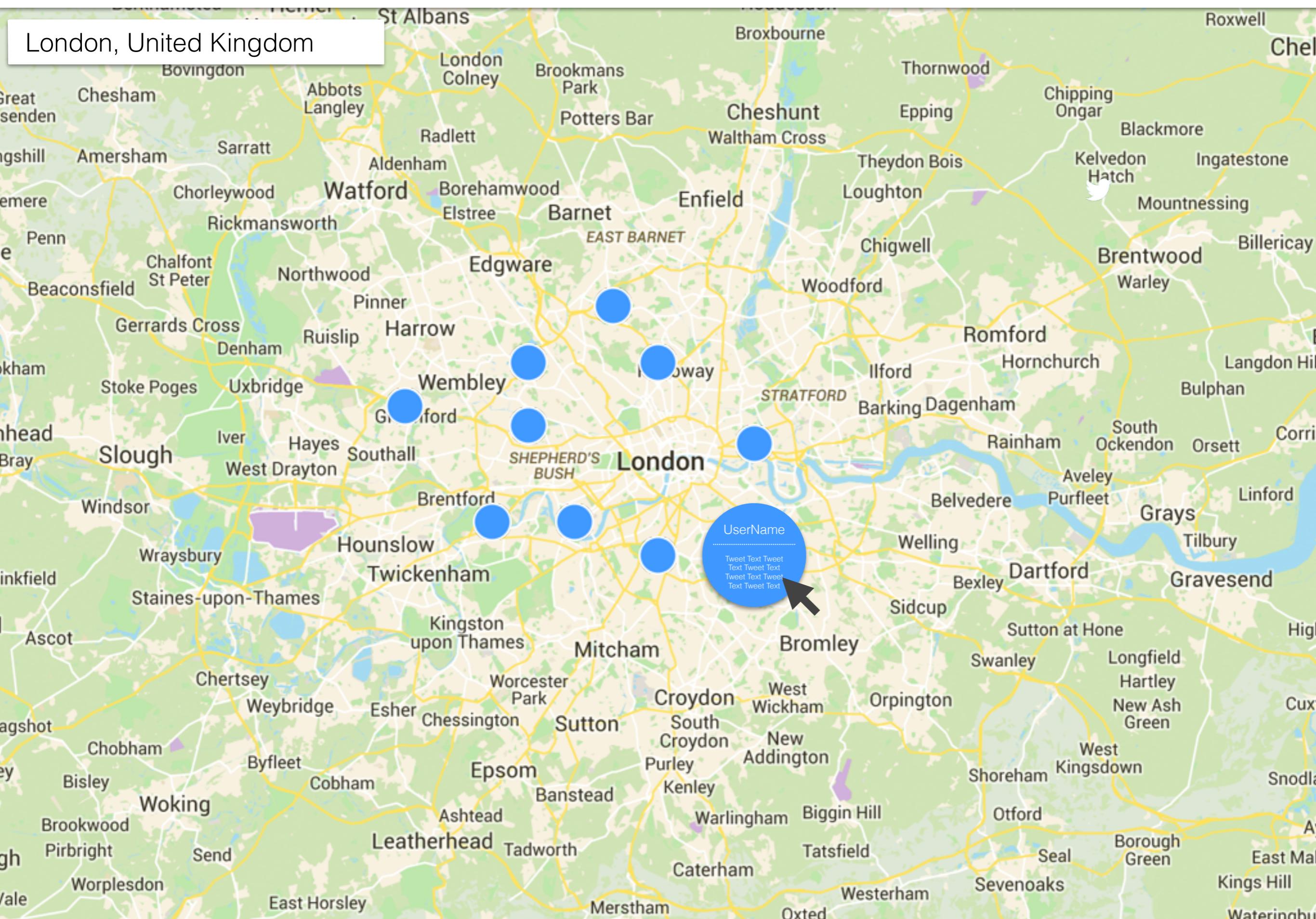
Your stand alone application is
uploading. It may take 1minute

Structure



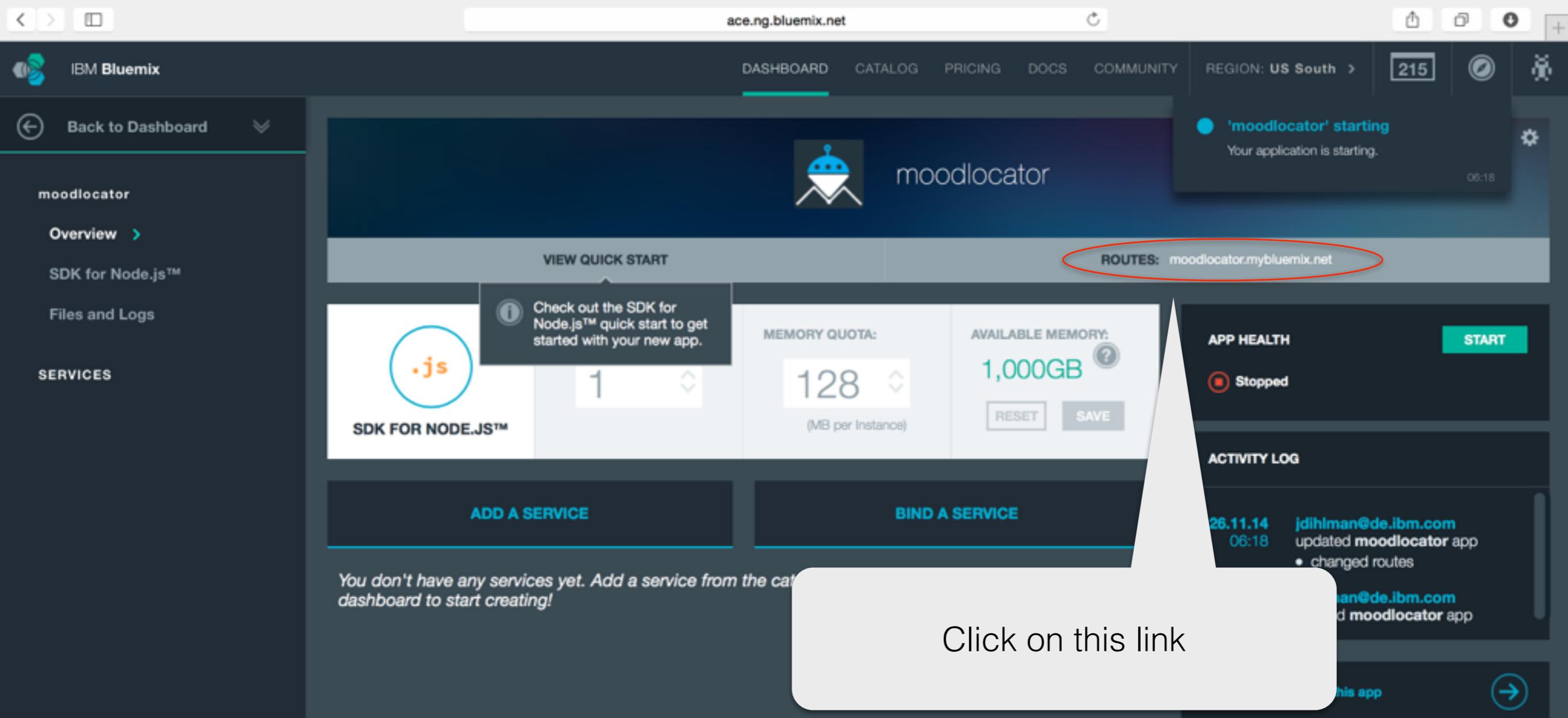
Twitter Locator

London, United Kingdom



2. Show how to use the alone standing application

App Info



The screenshot shows the IBM Bluemix App Info interface for an application named "moodlocator".

Header: The top navigation bar includes links for DASHBOARD, CATALOG, PRICING, DOCS, COMMUNITY, REGION: US South, and a notifications badge showing 215 messages.

Left Sidebar: A sidebar on the left lists the app's name ("moodlocator") and provides links for Overview, SDK for Node.js™, and Files and Logs.

Services Section: This section is titled "SERVICES" and features a "SDK FOR NODE.JS™" card with a ".js" icon. It includes a "VIEW QUICK START" button and a note about the quick start guide.

Resource Allocation: Shows MEMORY QUOTA: 128 MB per Instance and AVAILABLE MEMORY: 1,000GB.

Routes: A red circle highlights the ROUTES section, which displays the URL `moodlocator.mybluemix.net`.

App Health: Shows the app is currently Stopped.

Activity Log: Displays a log entry from 26.11.14 at 06:18 by jdhilman@de.ibm.com, indicating an update to the moodlocator app.

Callout: A large callout box with the text "Click on this link" points to the "moodlocator.mybluemix.net" URL in the Routes section.

3. Add the User Modeling Service to it

Go back to GitHub and take a look at section 5

Catalog

ace.ng.bluemix.net

IBM Bluemix DASHBOARD CATALOG PRICING DOCS COMMUNITY REGION: US South > 215

jdihlman@de.i...

Type here to begin

Starters // Choose a package of sample code and services, or start from scratch

Boilerplates

Get started with a new app, now

Internet of Things Foun... Java Cache Web Starter Java Cloud Foundry Sta... Java DB Web Starter Mobile Cloud Node.js Cache Web St...

IBM IBM IBM IBM IBM IBM

Node.js Cloud Foundry IBM

IBM Watson User Modeling Node-RED Starter Vaadin Rich Web Starter

Community Community

Runtimes

Run an app in the language of your choice

.java liberty .js .rb on rails .rb Sinatra Bring Your Buildpack

Liberty for Java™ IBM SDK for Node.js™ IBM Ruby on Rails Ruby Sinatra Community Community

Open Bluemix Catalog

Services

Screenshot of the IBM Bluemix Catalog interface (ace.ng.bluemix.net) showing the Watson service category.

The page title is "Catalog - Bluemix" and the URL is "ace.ng.bluemix.net". The top navigation bar includes links for DASHBOARD, CATALOG (which is underlined), PRICING, DOCS, COMMUNITY, REGION: US South, and a notifications count of 215.

The left sidebar shows the user "jdihlman@de.i..." and a search bar with placeholder text "Type here to begin".

The main content area displays the "Watson" category under "Services". A descriptive text states: "Build cognitive apps that help enhance, scale, and accelerate human expertise".

Watson services listed (all marked as IBM BETA):

- Concept Expansion
- Language Identification
- Machine Translation
- Message Resonance
- Question and Answer
- Relationship Extraction
- User Modeling

A large callout bubble with the text "Search for Watson" is overlaid on the page.

Category filters on the left:

- Watson
- Mobile
- DevOps
- Web and Application
- Integration
- Data Management
- Big Data
- Security
- Business Analytics
- Internet of Things

Support filters on the left:

- IBM
- Third Party
- Experimental
- Beta

DevOps section at the bottom:

- From development to deployment

User Modelling

Screenshot of the IBM Bluemix Catalog interface showing the Watson User Modeling service.

The catalog page displays various IBM Watson services under the "Watson" category, all marked as "IBM BETA". A callout bubble highlights the "User Modeling" service.

Watson Services:

- Concept Expansion
- Language Identification
- Machine Translation
- Message Resonance
- Question and Answer
- Relationship Extraction
- User Modeling

Mobile Services:

- Mobile Application Security
- Mobile Data

DevOps Services:

- App User Registry
- Auto Scaling
- Delivery Pipeline
- Monitoring and Analytics
- Track & Plan
- Bluemix Motor

Category Filter: jdihlman@de.ibm.com

Support Filter:

- IBM
- Third Party
- Experimental
- Beta

Callout Bubble Text: Choose the Watson User Modeling Service

User Modeling Info

ace.ng.bluemix.net

Catalog - Bluemix

MoodLocator

IBM Bluemix

DASHBOARD CATALOG PRICING DOCS COMMUNITY REGION: US South > 215

Back to All Categories

 User Modeling IBM

PUBLISH DATE 15.11.2014

TYPE Service

LOCATION US South

[VIEW DOCS](#)

BETA The User Modeling service uses linguistic analytics to extract a set of personality and social traits from the way a person communicates. The service can analyze any communication the user makes available such as their text messages, tweets, posts, email, and more. Users of the service can understand, connect, and communicate with people on a more personally tailored level by analyzing personality and social traits.

Pick a plan Monthly prices shown are for country or region: [Germany](#)

| Plan | Features | Price |
|--------|----------|-------|
| ✓ Beta | | Free |
| ⓘ Beta | | |

Add Service

App: Select an application

Service name: User Modeling-Ir

Selected Plan: Beta

CREATE

Click on View Docs

User Modeling Docs

The screenshot shows the User Modeling service page on the IBM Watson Developer Cloud. The top navigation bar includes links for Catalog - Bluemix, MoodLocator, and User Modeling | IBM Watson Developer Cloud. Below the navigation is a secondary header with links for Getting Started, Services, Docs, Content Marketplace, and Community. On the left, there's a sidebar with a 'User Modeling' icon, a 'BETA' label, and links for Intro, Live Demo, How It Works, Intended Use, Docs, and Try on Bluemix. The main content area features a large blue 'User Modeling' title, a green circular icon with a network graph, and a description of improved understanding of user preferences. A detailed text block explains the service's function: "The IBM Watson User Modeling service uses linguistic analysis, extract cognitive and social characteristics, including Big Five, and Needs, from communications that the user makes available, such as email, text messages, tweets, forum posts, and more. By deriving cognitive and social preferences, the service helps users to understand, connect to, and communicate with other people on a more personalized level." To the right, there's a diagram showing a central user profile icon with arrows pointing to 'artistic' (green arrow) and 'intellectual' (purple arrow) traits, each associated with a smiley face icon. Below this diagram is a callout box labeled "User Modeling Documentation". At the bottom, there's a graphic of a computer monitor displaying a globe with network connections, and a call-to-action button labeled "View a Live Demo".

Catalog - Bluemix MoodLocator User Modeling | IBM Watson Developer Cloud

IBM Watson Developer Cloud

Getting Started Services Docs Content Marketplace Community

BETA

User Modeling

Intro

Live Demo

How It Works

Intended Use

Docs

Try on Bluemix

User Modeling

Improved understanding of people's preferences to help engage users on their own terms

The IBM Watson User Modeling service uses linguistic analysis, extract cognitive and social characteristics, including Big Five, and Needs, from communications that the user makes available, such as email, text messages, tweets, forum posts, and more. By deriving cognitive and social preferences, the service helps users to understand, connect to, and communicate with other people on a more personalized level.

Documentation Use in Bluemix

artistic

intellectual

User Modeling Documentation

Check out the User Modeling Service in Action

View a Live Demo

View a Live Demo

The screenshot shows the User Modeling service page on the IBM Watson Developer Cloud. The top navigation bar includes links for Catalog - Bluemix, MoodLocator, and User Modeling | IBM Watson Developer Cloud. Below the navigation is a main header with the IBM logo, the title "IBM Watson Developer Cloud", and a "User Modeling" section. The "User Modeling" section is labeled "BETA" and features a green circular icon with a network graph. To the right of the icon is the title "User Modeling" in large blue text, followed by a subtitle: "Improved understanding of people's preferences to help engage users on their own terms". A detailed description follows, explaining how the service uses linguistic analytics to extract cognitive and social characteristics. On the right side, there are two buttons: "Documentation" and "Use in Bluemix". Below these buttons is a callout box containing the text "Click on view a live demo" and a link "View a Live Demo". At the bottom of the page, there is a large blue banner with the text "Check out the User Modeling Service in Action" and a "View a Live Demo" button.

Catalog - Bluemix MoodLocator User Modeling | IBM Watson Developer Cloud

IBM Watson Developer Cloud

User Modeling

BETA

User Modeling

Intro

Live Demo

How It Works

Intended Use

Docs

Try on Bluemix

User Modeling

Documentation

Use in Bluemix

artistic

outgoing

Click on view a live demo

View a Live Demo

Check out the User Modeling Service in Action

Live Demo

watson-um-demo.mybluemix.net

Catalog - Bluemix MoodLocater User Modeling | IBM Watson Developer Cloud User Modeling Demonstration



User Modeling

The Watson User Modeling service uses linguistic analytics to extract a spectrum of cognitive and social characteristics from the text data that a person generates through text messages, tweets, posts, and more.

Try the service

Output

Call me Ishmael. Some years ago-never mind how long precisely-having little or no money in my purse, and nothing particular to interest me on shore, I thought I would sail about a little and see the watery part of the world. It is a way I have of driving off the spleen and regulating the circulation. Whenever I find myself growing grim about the mouth; whenever it is a damp, drizzly November in my soul; whenever I find myself involuntarily pausing before coffin warehouses, and bringing up the rear of every funeral I meet; and especially whenever my hypos get such an upper hand of me, that it requires a strong moral principle to prevent me from deliberately stepping into the street, and methodically knocking people's hats off-then, I account it high time to get to sea as soon as I can. This is my substitute for pistol and ball. With a philosophical flourish Cato throws himself upon his sword; I quietly take to the ship. There is nothing surprising in this. If they but knew it, almost all men in their degree, some time or other, cherish very

Clear Analyze

Click on analyze

Analyze



User Modeling Info

ace.ng.bluemix.net

Catalog - Bluemix MoodLocator User Modeling | IBM Watson Developer Cloud User Modeling Demonstration

IBM Bluemix DASHBOARD CATALOG PRICING DOCS COMMUNITY REGION: US South 215 +

Back to All Categories

 User Modeling
IBM

PUBLISH DATE
15.11.2014

TYPE
Service

LOCATION
US South

[VIEW DOCS](#)

BETA The User Modeling service uses linguistic analytics to extract a set of personality and social traits from the way a person communicates. The service can analyze any communication the user makes available such as their text messages, tweets, posts, email, and more. Users of the service can understand, connect, and communicate with people on a more personally tailored level by analyzing personality and social traits.

Pick a plan Monthly prices shown are for country or region: [Germany](#)

| Plan | Features | Price |
|--------|----------|-------|
| ✓ Beta | | Free |
| ⓘ Beta | | |

Add Service

App:

Service name:

Select Plan:

CREATE

Add it to your application

Add Service Process

The screenshot shows the IBM Bluemix dashboard for the 'moodlocator' application. The application icon is a blue robot head with a white base. The routes listed are `moodlocator.mybluemix.net`. The app health status is green, indicating the app is running. The activity log shows several recent events:

- 26.11.14 06:23 jdihlman@de.ibm.com started moodlocator app
- 26.11.14 06:21 jdihlman@de.ibm.com updated moodlocator app
 - instances to 1
 - memory to 128 MB
 - disk quota to 1024 MB
 - modified environment
- 26.11.14 06:18 jdihlman@de.ibm.com started moodlocator app

A modal dialog box is centered on the screen with the title "Restage Application". It contains the message: "Your 'moodlocator' app needs to be restaged to use the new 'User Modeling-lr' service. Do you want to restage it now?". There are two buttons at the bottom: "OK" and "CANCEL". A callout bubble with the text "Hit ok" points to the "OK" button.

User Modeling Doc

The screenshot shows the User Modeling page on the IBM Watson Developer Cloud. The top navigation bar includes links for Dashboard - Bluemix, MoodLocator, User Modeling | IBM Watson Developer Cloud, and User Modeling Demonstration. Below the navigation is a secondary header with links for Getting Started, Services, Docs, Content Marketplace, and Community.

The main content area features a large blue title "User Modeling" with a green circular icon containing a network graph to its left. A "BETA" label is positioned above the title. To the right of the title are two buttons: "Documentation" and "Use in Bluemix".

A central callout box contains the text "Scroll down" and a small graphic showing a user profile icon with a green arrow pointing to it, labeled "outgoing".

The left sidebar contains a navigation menu with the following items: Intro (underlined), Live Demo, How It Works, Intended Use, Docs, and Try on Bluemix. Above the sidebar, there are navigation arrows and a "User Modeling" section with a green circular icon.

At the bottom, there is a blue banner with the text "Check out the User Modeling Service in Action" and a "View a Live Demo" button.

View full Docs

A screenshot of a web browser showing the IBM Watson Developer Cloud User Modeling page. The URL in the address bar is `ibm.com`. The page title is "User Modeling | IBM Watson Developer Cloud". The navigation bar includes links for "Getting Started", "Services", "Docs", "Content Marketplace", and "Community". On the left, there's a sidebar with a "User Modeling" icon and links for "Intro", "Live Demo", "How It Works", Intended Use, "Docs", and "Try on Bluemix". The main content area has a heading "Intended Use" and text about the service being contextually specific and knowledgeable. A callout bubble points from the "View full docs" button in the documentation section below to this text.

Intended Use

Certain Services in Watson Developer Cloud are contextually specific and knowledgeable depending on the domain model and content set they are connected to.

This BETA Service uses any content.

The data set:

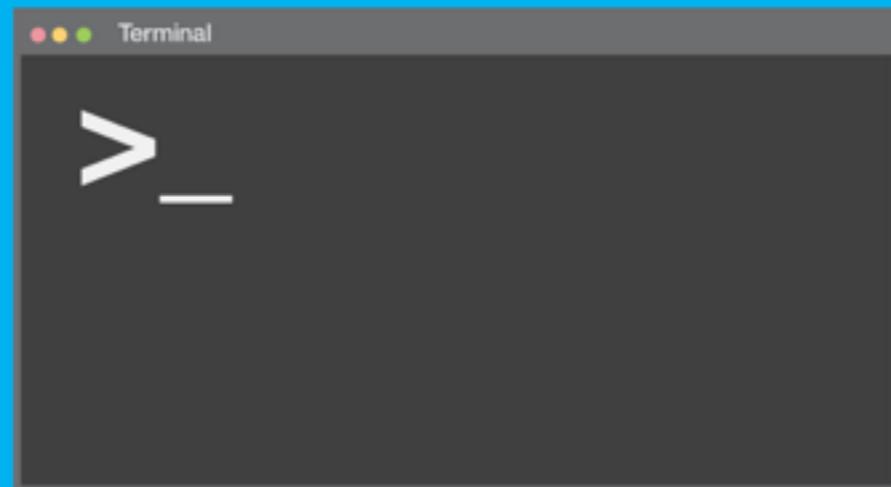
At least 1000 words of text written by one individual.

Click on view full docs

Documentation

Ready to get down to the details? Full documentation detailing how to get started using this Service in Bluemix is available for each Watson service.

[View full docs](#)



Watson Doc

The screenshot shows a browser window with the URL [ibm.com](#) in the address bar. The page title is "MoodLocator" under "Service Documentation | IBM Watson Developer Cloud". The main navigation menu includes "Getting Started", "Services", "Docs", "Content Marketplace", and "Community". On the left, there's a sidebar with a "Getting started" section and links for "Connecting to Bluemix", "A sample User Modeling application in Bluemix", "The sample application in Java", "The sample application in Node.js", "The sample application in Ruby", "Deploying and running the application in Bluemix", "Building and running the application locally", "Using the User Modeling service in Java", "API reference", and "User Modeling service". The main content area is titled "Getting started with the User Modeling service". It describes the service's purpose of extracting insights from text to help businesses understand client preferences. It includes links to a quick demo, sample code on GitHub, and the Watson developer forum.

Getting started with the User Modeling service

The IBM Watson™ User Modeling service provides an Application Programming Interface (API) that enables applications to derive insights from social media, enterprise data, or other digital communications. The service uses linguistic analytics to extract cognitive and social characteristics, including Big Five personality, values, and needs, from text. These insights help businesses to understand their clients' preferences and improve customer satisfaction by anticipating customer needs and recommending the next best actions. This allows businesses to improve client acquisition, retention, and engagement, and to strengthen relations with their clients.

To see a quick demo of the User Modeling service in action, click [here](#).

To download an article and sample code that shows how to integrate the User Modeling service with another Watson product, Watson Explorer, see [GitHub](#).

We'd love to hear from you! Please provide comments or ask questions about **User Modeling** in the [Watson dev forum](#).

We are always looking for ways to improve and learn from your experience with our services. You can help Watson learn by providing personalized feedback. The User Modeling service includes a feedback API that can be used to submit comments, observations, and suggestions that Watson and its developers can use to educate and improve this service. The User Modeling API, which includes the feedback API, is documented [here](#).

Connecting to Bluemix

Bluemix is the IBM Platform as a Service (PaaS) environment. For more information about Bluemix, see the [Bluemix documentation](#). To connect your application to Bluemix, you need to install command-line or graphical tools. This is explained in [Getting started with Bluemix](#).

Select sample application in
Node.js

Once you have installed the tools that you want to use, the first step in developing an application in Bluemix that uses the User Modeling service is to [connect to the API and log in to Bluemix](#). At this time, you can also create an instance of the

Sample Application

The screenshot shows a web browser window with the URL ibm.com in the address bar. The page title is "MoodLocator" under "Service Documentation | IBM Watson Developer Cloud". The main content area is titled "The sample application in Node.js". It explains how to create a sample Node.js application using the User Modeling service in Bluemix, mentioning Express and Jade frameworks. A sidebar on the left lists various sample application links for Java, Node.js, Ruby, and Bluemix environments. A callout bubble highlights the "Example code you can scroll in here" section, which contains the following Node.js code:

```
/*jshint node:true*/
// app.js
// This file contains the server side JavaScript code for your application.
// This sample application uses express as web application framework (http://expressjs.com/),
// and jade as template engine (http://jade-lang.com/).

var express = require('express');
var https = require('https');
var url = require('url');
var querystring = require('querystring');
var extend = require('util')._extend;

var dummy_text = require('./dummy-text').text
var flatten = require('./flatten');
```

Dashboard - Bluemix MoodLocator Service Documentation | IBM Watson Developer Cloud User Modeling Demonstration

IBM Watson Developer Cloud Getting Started Services Docs Content Marketplace Community

IMPORTANT

service in Bluemix. The sample uses Express as the web application framework and Jade as a template engine.

Complete the following steps to prepare the sample application:

1. Download the .zip file that contains the sample application code. Extract the file on your local machine, and change the working directory for the sample application.

The Javascript source file for the application is named `app.js`. Examine the code and its comments to better understand the code structure of the application. The following example shows the `app.js` file:

```
// creates a request function using the https option
// the function that return receives a callback
var create_profile_request = function(options,content,callback) {
    // create the post data to send to the User Modeling service
    var post_data = {
        'contentItems' : [
            'userid' : 'dummy',
            'id' : 'dummyUuid',
            'sourceid' : 'freetext',
            'contenttype' : 'text/plain',
            'language' : 'en',
            'content': content
        ]
    };
    // Create a request to POST to the User Modeling service
    var options = {
        host: "https://mymodeling.mybluemix.net",
        port: 443,
        path: "/v1/profiles",
        method: "POST",
        headers: {
            "Content-Type": "application/json"
        }
    };
    var req = https.request(options, function(res) {
        var str = '';
        res.setEncoding('utf8');
        res.on('data',function(chunk){str+=chunk});
        res.on('end',function(){
            var obj = JSON.parse(str);
            if(obj.error)
                callback("Error creating profile: "+obj.error);
            else
                callback(null,obj);
        });
    });
    req.write(JSON.stringify(post_data));
    req.end();
}
```

You have to copy the exact same function you deleted before!

4. Activate the User Modeling Service

COPY THIS PART

You have to copy the exact same thing —>
out of the Watson Documentation into your
application

Select it

Copy it

cmd c / strg c

We will undo everything what we
did for preparation

```
//For presentation start delete this section

//Function you will find in the User Modeling documentation
var create_profile_request = function(options,content) {
    return function /*function*/ callback) {
        // create the post data to send to the User Modeling service
        var post_data = {
            'contentItems' : [
                'userid' : 'dummy',
                'id' : 'dummyUuid',
                'sourceid' : 'freetext',
                'contenttype' : 'text/plain',
                'language' : 'en',
                'content': content
            ]
        };
        // Create a request to POST to the User Modeling service
        var profile_req = https.request(options, function(result) {
            result.setEncoding('utf-8');
            var response_string = '';
            result.on('data', function(chunk) {
                response_string += chunk;
            });
            result.on('end', function() {
                if (result.statusCode != 200) {
                    var error = JSON.parse(response_string);
                    callback({message: error.user_message}, null);
                } else
                    callback(null,response_string);
            });
        });
        profile_req.on('error', function(e) {
            callback(e,null);
        });
        profile_req.write(JSON.stringify(post_data));
        profile_req.end();
    }
};

//For presentation stop deleting this section
```

Open app.js

Open the app.js file with a text editor you will find it in

/ node / app.js

Scroll to line 238

Paste the whole code at the end of your document, where you deleted it before.

Stay in app.js

Scroll to line 189

You will find something that looks like that, delete this bracket „ /* “ in line 190 and this bracket „ */ “ in line 211. I marked the position red where to add them.

Before

```
189 //For presentation start commenting here
190 /*  
191 //Create a profile request with the text and the https options and call it  
192 create_profile_request(profile_options,textToWatson)(function(error,profile_string) {  
193     if (error) console.log(error.message);  
194     else {  
195         //Parse the Profile and format it  
196         var profile_json = JSON.parse(profile_string);  
197         var flat_traits = flatten.flat(profile_json.tree);  
198  
199         //Sent Watson Data to Client  
200         var watsonObj = {  
201             id: msg.id,  
202             watsonJSON: flat_traits  
203         }  
204  
205  
206         //Send Watson Data to Client  
207         socket.emit('watson message', watsonObj);  
208         //console.log(flat_traits);  
209     }  
210 }); //For presentation stop commenting here
211 */
212 }
```

After

```
189 //For presentation start commenting here
190
191 //Create a profile request with the text and the https options and call it
192 create_profile_request(profile_options,textToWatson)(function(error,profile_string) {
193     if (error) console.log(error.message);
194     else {
195         //Parse the Profile and format it
196         var profile_json = JSON.parse(profile_string);
197         var flat_traits = flatten.flat(profile_json.tree);
198
199         //Sent Watson Data to Client
200         var watsonObj = {
201             id: msg.id,
202             watsonJSON: flat_traits
203         }
204
205
206         //Send Watson Data to Client
207         socket.emit('watson message', watsonObj);
208         //console.log(flat_traits);
209     }
210 }); //For presentation stop commenting here
211
212 }
```

Save your changes

Open style.css

Open the style.css file with a text editor you will find it in

/ node / public / stylesheets / style.css

Scroll to the bottom of the page

You will find something that looks like that, add this bracket „ /* “ in line 449 and this bracket „ */ “ in line 452. I marked the position red where to add them.

Before

```
448 /* For Presentation activate oder deactivate the Section below */
449 
450 #footer{ visibility: hidden; }
451 #map_canvas{ height: 100%; }
452 
```

After

```
448 /* For Presentation activate oder deactivate the Section below */
449 /*
450 #footer{ visibility: hidden; }
451 #map_canvas{ height: 100%; }
452 */
```

Save your changes. Perfect to understand what it does, it enables or disables the white Watson User Modeling bar at the bottom of the page.

Open style.css

Open the style.css file with a text editor you will find it in

/ node / public / stylesheets / style.css

Scroll to the bottom of the page

You will find something that looks like that, add this bracket „ /* “ in line 449 and this bracket „ */ “ in line 452. I marked the position red where to add them.

Before

```
448 /* For Presentation activate oder deactivate the Section below */
449 
450 #footer{ visibility: hidden; }
451 #map_canvas{ height: 100%; }
452 
```

After

```
448 /* For Presentation activate oder deactivate the Section below */
449 /*
450 #footer{ visibility: hidden; }
451 #map_canvas{ height: 100%; }
452 */
```

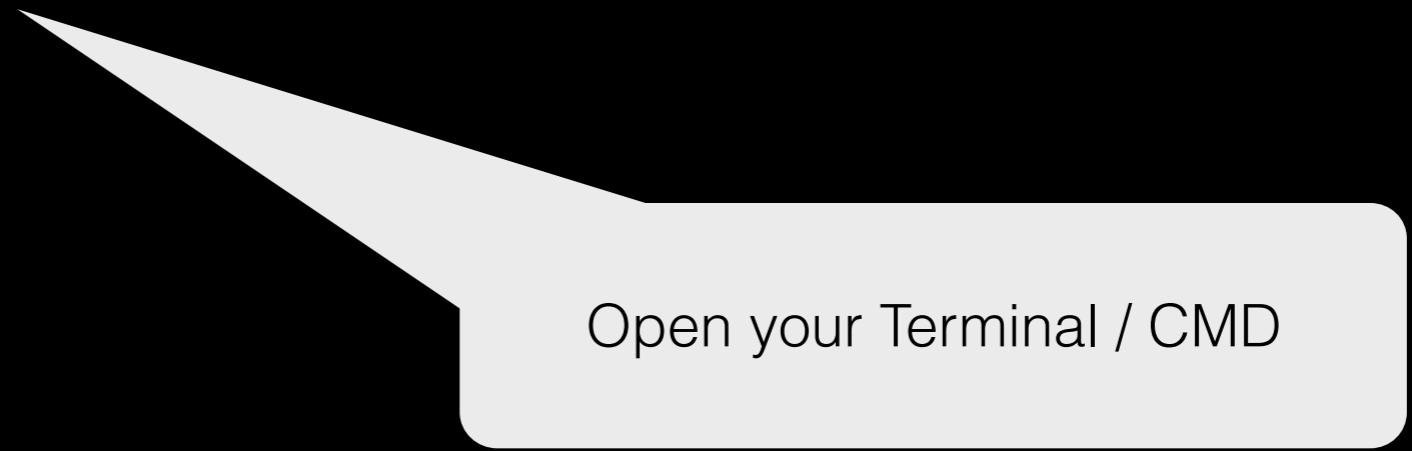
Save your changes.

5. Upload the application

Go back to GitHub and take a look at section 6

Terminal

Dihlmann:MoodlocaterPr Dihlmann\$ █



Open your Terminal / CMD

cd MoodLocator

```
Dihlmann:~ Dihlmann$ cd Desktop/  
Dihlmann:Desktop Dihlmann$ cd MoodlocaterPr/  
Dihlmann:MoodlocaterPr Dihlmann$ █
```



Instead of „MoodlocaterPr“ write
„node“

Connect to Bluemix

```
Dihlmann:~ Dihlmann$ cd Desktop/  
Dihlmann:Desktop Dihlmann$ cd MoodlocatorPr/  
Dihlmann:MoodlocatorPr Dihlmann$ cf api https://api.ng.bluemix.net  
Setting api endpoint to https://api.ng.bluemix.net...  
OK
```

```
API endpoint: https://api.ng.bluemix.net (API version: 2.14.0)  
User: jdihlman@de.ibm.com  
Org: jdihlman@de.ibm.com  
Space: dev  
Dihlmann:MoodlocatorPr Dihlmann$
```

Copy & Paste

Login

```
Dihlmann:~ Dihlmann$ cd Desktop/  
Dihlmann:Desktop Dihlmann$ cd MoodlocaterPr/  
Dihlmann:MoodlocaterPr Dihlmann$ cf api https://api.ng.bluemix.net  
Setting api endpoint to https://api.ng.bluemix.net...  
OK
```

```
API endpoint: https://api.ng.bluemix.net (API version: 2.14.0)  
User: jdihlman@de.ibm.com  
Org: jdihlman@de.ibm.com  
Space: dev  
Dihlmann:MoodlocaterPr Dihlmann$ cf login -u jdihlman@de.ibm.com  
API endpoint: https://api.ng.bluemix.net
```

```
Password>  
Authenticating...  
OK
```

```
Targeted org jdihlman@de.ibm.com
```

```
Targeted space dev
```

```
API endpoint: https://api.ng.bluemix.net (API version: 2.14.0)  
User: jdihlman@de.ibm.com  
Org: jdihlman@de.ibm.com  
Space: dev  
Dihlmann:MoodlocaterPr Dihlmann$
```

Enter the password

cf push

```
Dihlmann:~ Dihlmann$ cd Desktop/  
Dihlmann:Desktop Dihlmann$ cd MoodlocatorPr/  
Dihlmann:MoodlocatorPr Dihlmann$ cf api https://api.ng.bluemix.net  
Setting api endpoint to https://api.ng.bluemix.net...  
OK
```

```
API endpoint: https://api.ng.bluemix.net (API version: 2.14.0)
```

```
User: jdihlman@de.ibm.com
```

```
Org: jdihlman@de.ibm.com
```

```
Space: dev
```

```
Dihlmann:MoodlocatorPr Dihlmann$ cf login -u jdihlman@de.ibm.com
```

```
API endpoint: https://api.ng.bluemix.net
```

```
Password>
```

```
Authenticating...
```

```
OK
```

```
Targeted org jdihlman@de.ibm.com
```

```
Targeted space dev
```

Instead of „moodlocator“ enter
your application name

```
API endpoint: https://api.ng.bluemix.net (API version: 2.14.0)
```

```
User: jdihlman@de.ibm.com
```

```
Org: jdihlman@de.ibm.com
```

```
Space: dev
```

```
Dihlmann:MoodlocatorPr Dihlmann$ cf push moodlocator
```

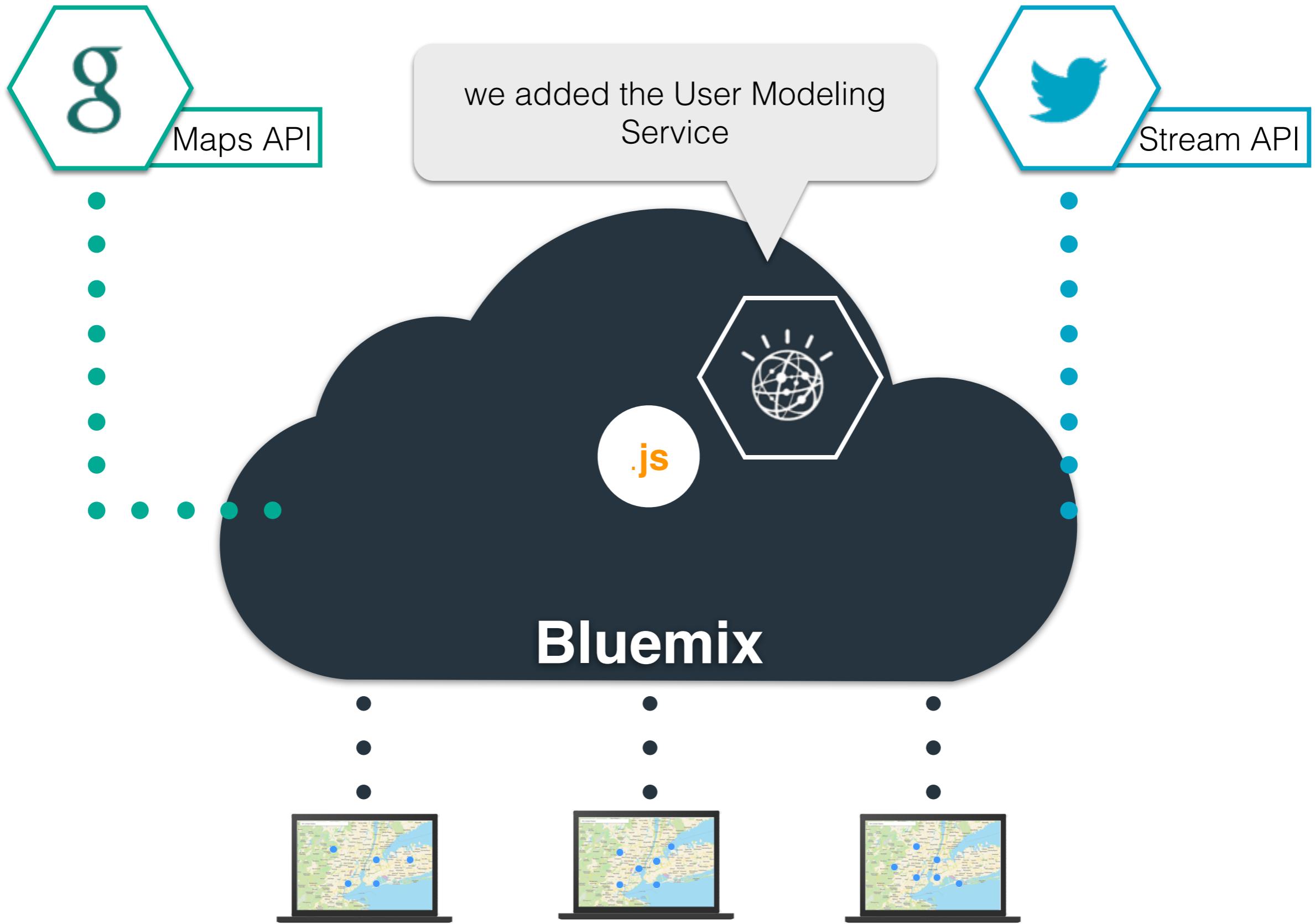
```
Using manifest file /Users/Dihlmann/Desktop/MoodlocatorPr/manifest.yml
```

```
Updating app moodlocator in org jdihlman@de.ibm.com / space dev as jdihlman@de.ibm.com...
```

```
OK
```

Your stand alone application is
uploading. It may take 1 minute

Structure



User Modeling Service

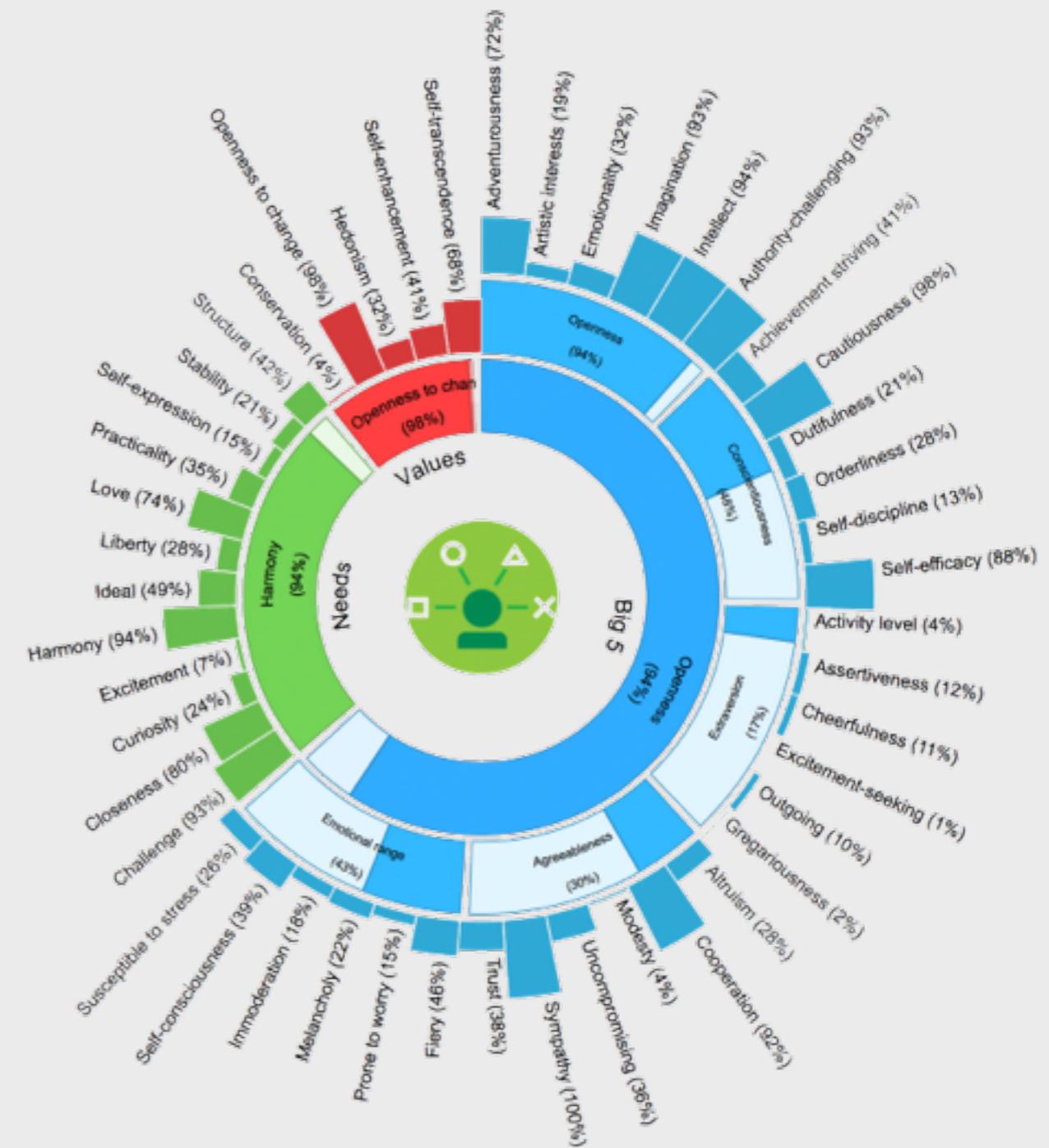
Text - at least 100 Words



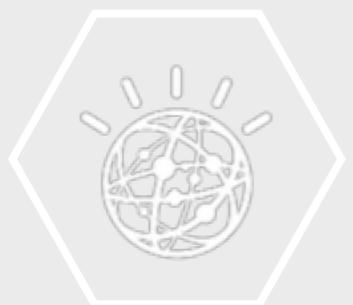
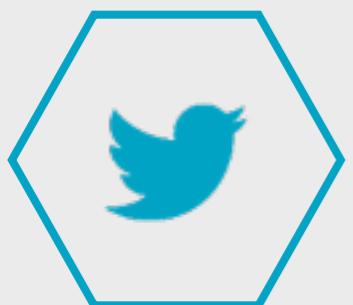
BETA:
Only english
Text

Get a fully character
analysis from the author
of the text

Possible character of the author



User Modeling Service



All Tweets from one city



BETA:
Only english
Tweets

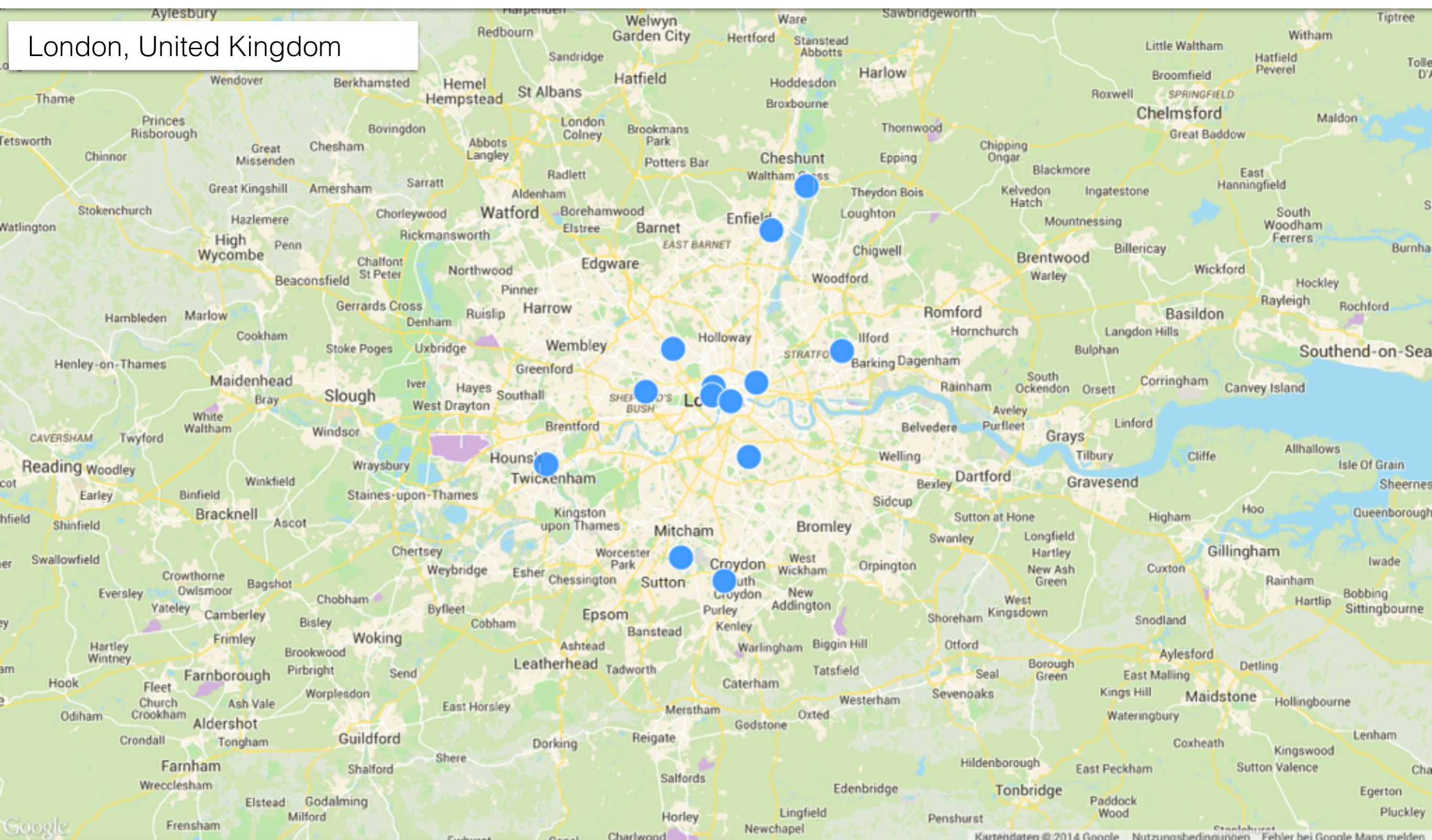


Get a fully character
analysis from all the
users combined who are
twittering in this city

6. Show how to use the new application

MoodLocator

London, United Kingdom



Openness

London 51%

Choose City

Jan-Niklas Dihlmann

< >

| Openness | Emotionality | Authority-challenging |
|----------|--------------|-----------------------|
|----------|--------------|-----------------------|

| | | |
|-----------------|-------------|-------------------|
| Adventurousness | Imagination | Conscientiousness |
|-----------------|-------------|-------------------|

| | | |
|--------------------|-----------|----------------------|
| Artistic interests | Intellect | Achievement striving |
|--------------------|-----------|----------------------|

November 2014

Tipp how to use

- User Modeling Service is a **BETA** version, is only analyzing english words. Therefore choose a city where people twitter mostly in english.
- Choose big cities like London, New York, Johannesburg.
- When you hover over one Tweet, only hover over the next Tweet after the first one closes.
- Test it yourself, enter your current location and send a Tweet with text and geolocation. Try to find it.

Have fun and enjoy it!

MoodLocator



Bluemix: moodlocator.mybluemix.com

GitHub: [/JDihlmann/moodlocator](https://github.com/JDihlmann/moodlocator)