# Firebase

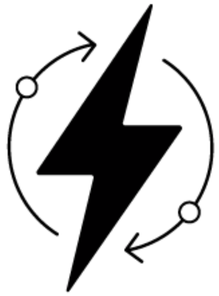
MVP TO PRODUCTION IN ONE PLATFORM

# What is it?

# Features

## AUTHENTICATION

Client SDKs; Custom tokens



## REALTIME DATABASE

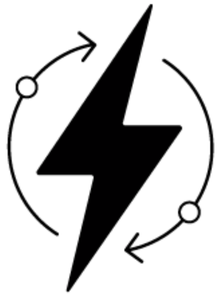
NoSQL Store & Sync

## HOSTING

CLI deploy; SSL default

icons: Firebase, Google Inc., firebase.com

AUTHENTICATION



## REALTIME DATABASE

NoSQL Store & Sync

HOSTING

REALTIME

Creative Commons: update by Christopher Holm-Hansen from the Noun Project

Features > Database > Realtime

## INSTANT UPDATES TO DATA

build a responsive system

## IMMEDIATE TRANSACTIONS

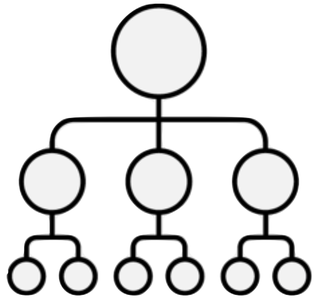
build a fluid experience

SECURE

Public Domain: Secure by Wayne Thayer, CA from the Noun Project

## ENCRYPTED DATA TRANSFER

guarantee SSL with 2048-bit cert



## GRANULAR ACCESS CONTROL

write security rules with json

AVAILABLE

Creative Commons: update by Christopher Holm-Hansen from the Noun Project

UNINTERRUPTED
OFFLINE WORKFLOW

syncs when connected

BUSINESS
GRADE SLA

guarantee 99.95% uptime

“Our servers manage millions of concurrent connections and billions of operations per month.”

– firebase.com/features

AUTHENTICATION

Client SDKs; Custom tokens

REALTIME DATABASE

HOSTING

- Easily authenticate users from Android, iOS, and JavaScript SDKs in just a few lines of code.
- Built-in email & password, Facebook, Twitter, GitHub, Google, and anonymous auth.
- Apps w/ built-in auth services can handle user login entirely with client-side code, saving you time and the headache of operating your own backend.
- Integrate authentication with your existing backend servers using our custom auth tokens.

```
// Authenticate users with a custom authentication token
ref.authWithCustomToken("<token>", authHandler);

// Alternatively, authenticate users anonymously
ref.authAnonymously(authHandler);

// Or with an email/password combination
ref.authWithPassword({
  email    : 'bobtony@firebase.com',
  password : 'correcthorsebatterystaple'
}, authHandler);

// Or via popular OAuth("facebook", "github", "google", or "twitter")
ref.authWithOAuthPopup("<provider>", authHandler);
ref.authWithOAuthRedirect("<provider>", authHandler);
```

AUTHENTICATION

REALTIME DATABASE

## HOSTING

CLI deploy; SSL default

- Production-grade static asset hosting
- All content delivered over SSL from global CDN
- Deploy & rollback using the Firebase CLI
- Every app gets its own firebaseapp.com domain
- Paid apps can deploy to a custom domain.
- Every app is served over a secure connection
- Firebase takes care of provisioning the SSL cert

```
$ firebase deploy
```

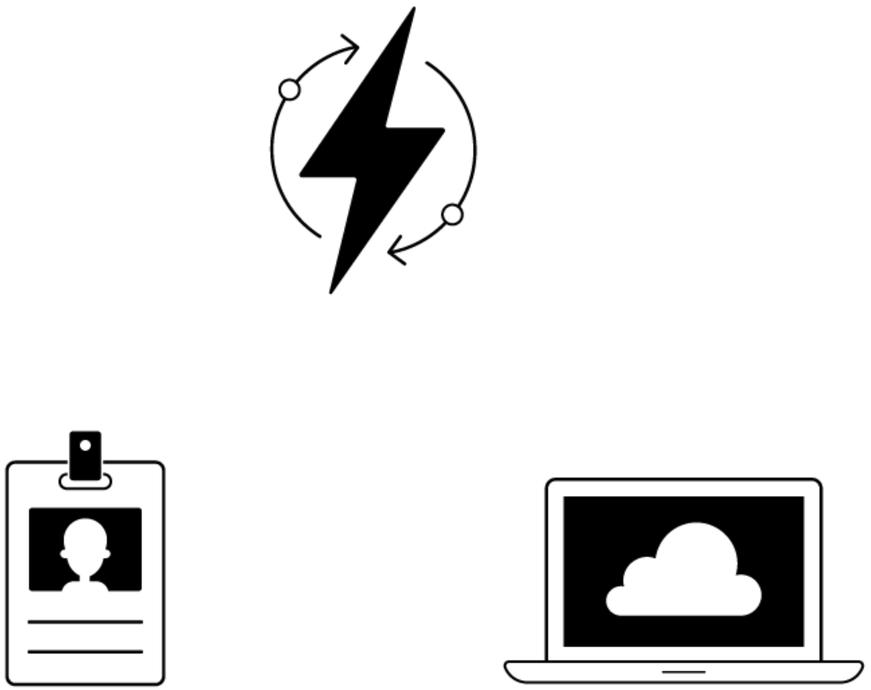
https://www.firebase.com/docs/hosting/command-line-tool.html

```
4 10:41:02 ~/Dev/rha/rha-angular (master)
$ firebase deploy
Public Directory: /Users/sinanbolel/Dev/rha/rha-angular/app
Preparing to deploy Public Directory...
Uploading 37 files with total compressed size = 609.61 kB
progress: 100%
Successfully deployed
Site URL: https://dev-nurha.firebaseapp.com, or use firebase open
Hosting Dashboard: https://firebase.com/account then view the hosting section of your app
```

# FOCUS ON WHAT MATTERS

build what matters


Build Once
RUN EVERYWHERE



Universal

build for everything

# Universal

one stack

TO RULE THEM ALL

Creative Commons: Ladder by Mike Ashley from the Noun Project

# Build Once

run everywhere

# Universal > JavaScript

AngularJS

Web & Mobile

Firebase

Backend


Node.js

Server


JS


build for growth


# Native SDKs

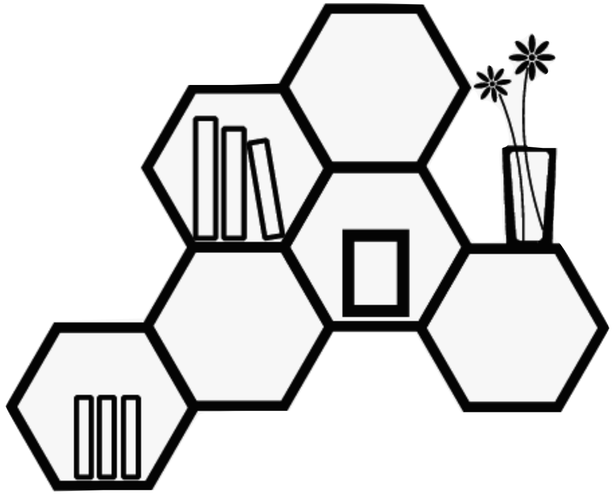
Firebase plays well with others

go mobile with

# Ionic Framework

part of

Google Cloud Platform



# MODULAR DESIGN

# Results

FUN TIME

Creative Commons: event by Clea Doltz from the Noun Project


questions?

```
{
    // rooms contains only meta info about each room
    // stored under the room's unique ID
    "rooms": {
      "one": {
        "name": "room alpha",
        "type": "private"
      },
      "two": { ... },
      "three": { ... }
    },

    // room members are easily accessible (or restricted)
    // we also store these by room ID
    "members": {
      // we'll talk about indices like this below
      "one": {
        "mchen": true,
        "hmadi": true
      },
      "two": { ... },
      "three": { ... }
    },

    // messages are separate from data we may want to iterate quickly
    // but still easily paginated and queried, and organized by room ID
    "messages": {
      "one": {
        "m1": { "sender": "mchen", "message": "foo" },
        "m2": { ... },
        "m3": { ... }
      },
      "two": { ... },
      "three": { ... }
    }
}
```

- so:29164753
  - series
    - -JkqSvwyDw5F9DvGUu6b
      - createdAt: 1426842959884
      - creators: "Michael Dante DiMartino,Bryan Konietzko"
      - description: "The series follows the adventures of protagonis..."
      - episode_reset: true
      - genres: "Action,Adventure,Fantasy,Comedy-drama"
      - keywords: "avatar,the last airbender"
      - name: "Avatar: The Last Airbender"
      - official_website: "http://nicktoons.nick.com/shows/avatar"
      - publish_date: "2005-02-21"
      - updatedAt: 1426843055127
  - users
    - anonymous:-Jkybn2Jl7pM2XrmlRMb
      - auth
        - provider: "anonymous"
        - uid: "anonymous:-Jkybn2Jl7pM2XrmlRMb"
      - expires: 1427066161
      - provider: "anonymous"
      - token: "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ2IjowLC..."
      - uid: "anonymous:-Jkybn2Jl7pM2XrmlRMb"
      - updatedAt: 1426980325952
    - anonymous:-Jw5TQedqUcdYQu6iZYJ
- so:29179389
  - companies
    - company1

```
{
  "rules": {
    "room_names": {
      // any logged in user can get a list of room names
      ".read": "auth !== null",

      "$room_id": {
        // this is just for documenting the structure of rooms, since
        // they are read-only and no write rule allows this to be set
        ".validate": "newData.isString()"
      }
    },

    "members": {
       // I can join or leave any room (otherwise it would be a boring demo)
       // I can have a different name in each room just for fun
       "$room_id": {
          // any member can read the list of member names
          ".read": "data.child(auth.uid).exists()",

          // room must already exist to add a member
          ".validate": "root.child('room_names/'+$room_id).exists()",

          "$user_id": {
             ".write": "auth.uid === $user_id",
             ".validate": "newData.isString() && newData.val().length > 0 && newData.val().length < 20"
          }
       }
    },

```

Data
Security & Rules
Simulator
Analytics
Login & Auth
Hosting
Secrets

Clear Changes

Security & Rules Documentation

## FIREBASE RULES

Save Rules

```
{
  "rules": {
    ".read": false,
    ".write": false,
    "app": {
      ".read": false,
      ".write": false,
      "content": {
        ".read": true,
        ".write": false
      },
      "admin": {
        ".read": false,
        ".write": false
      }
    },
    "sessions":{
      ".read": false,
      ".write": "auth != null",
      ".indexOn": ["uid","startedAt"],
      ".validate": "newData.hasChildren(['uid', 'provider', 'startedAt'])"
    },
    "users":{
      ".read": false,
      ".write": false,
      "$providerKey":{
        "$userKey": {
          ".read": "auth.provider === $providerKey && auth.uid === $userKey",
          ".write": "auth.uid === $userKey",
          ".validate": "newData.hasChildren(['uid', 'provider'])"
        }
      }
    },
    "users:sessions":{
      "$userKey": {
        ".read": "auth.uid === $userKey",
        ".write": "auth.uid === $userKey",
```

# Code

## FIREBASE & ANGULAR


https://www.firebase.com/docs/web/guide/

```
var ref = new Firebase("https://docs-examples.firebaseio.com/web/saving-data/fireblog");

var usersRef = ref.child("users");
usersRef.set({
  alanisawesome: {
    date_of_birth: "June 23, 1912",
    full_name: "Alan Turing"
  },
  gracehop: {
    date_of_birth: "December 9, 1906",
    full_name: "Grace Hopper"
  }
});

var hopperRef = usersRef.child("gracehop");
hopperRef.update({
  "nickname": "Amazing Grace"
});

dataRef.set("I'm writing data", function(error) {
  if (error) {
    alert("Data could not be saved." + error);
  } else {
    alert("Data saved successfully.");
  }
});

```

```
// Get a reference to our posts
var ref = new Firebase("https://docs-examples.firebaseio.com/web/saving-data/fireblog/posts");

// Retrieve new posts as they are added to our database
ref.on("child_added", function(snapshot, prevChildKey) {
  var newPost = snapshot.val();
  console.log("Author: " + newPost.author);
  console.log("Title: " + newPost.title);
  console.log("Previous Post ID: " + prevChildKey);
});
```


https://www.firebase.com/docs/web/libraries/angular/guide/

```
1  // define our app and dependencies (remember to include firebase!)
2  var app = angular.module("sampleApp", ["firebase"]);
3
4  // inject $firebaseObject into our controller
5  app.controller("ProfileCtrl", ["$firebaseObject",
6    function($firebaseObject) {
7      var ref = new Firebase("https://<YOUR-FIREBASE-APP>.firebaseio.com");
8
9      // download physicsmarie's profile data into a local object
10     // all server changes are applied in realtime
11     $scope.profile = $firebaseObject(ref.child('profiles').child('phsyicsmarie'));
12   }
13 ]);
14
```

```
1  // define our app and dependencies (remember to include firebase!)
2  var app = angular.module("sampleApp", ["firebase"]);
3  // inject $firebaseArray into our controller
4  app.controller("ProfileCtrl", ["$firebaseArray",
5    function($firebaseArray) {
6      var messagesRef = new Firebase("https://<YOUR-FIREBASE-APP>.firebaseio.com/messages");
7      // download the data from a Firebase reference into a (pseudo read-only) array
8      // all server changes are applied in realtime
9      $scope.messages = $firebaseArray(messagesRef);
10     // create a query for the most recent 25 messages on the server
11     var query = messagesRef.orderByChild("timestamp").limitToLast(25);
12     // the $firebaseArray service properly handles database queries as well
13     $scope.filteredMessages = $firebaseArray(query);
14   }
15 ]);
```


# Libraries

## PLETHORA OF AWESOME EXTENSIONS

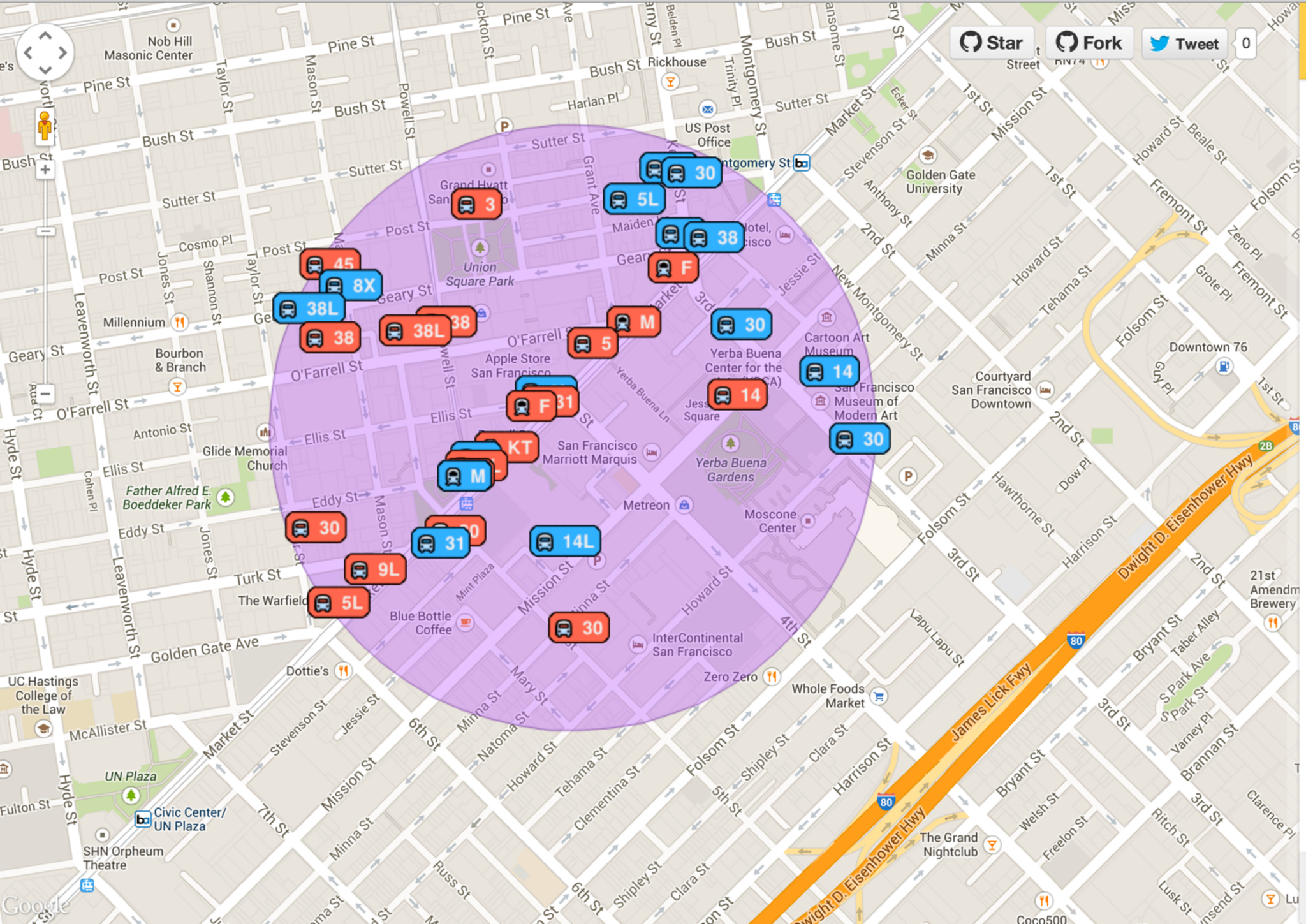
# From Firebase

| Name | Description |
|---|---|
| **AngularJS** | |
| angularfire | AngularJS bindings for Firebase |
| generator-angularfire | Yeoman generator for AngularFire: Angular + Firebase apps |
| **Tools** | |
| firebase-tools | Firebase CLI Tools to administer Firebase account, interact with Firebase hosting |
| firebase-queue | A fault-tolerant, multi-worker, multi-stage job pipeline built on the Firebase realtime database. |
| firebase-util | This is a collection of power toys (mostly experimental) and utilities for use in Firebase - [NormalizedCollection, Paginate] |
| **Authentication** | |
| firebase-token-generator-dotNet | Firebase Token Generator for .NET |
| firebase-token-generator-java | Firebase Token Generator for Java |
| firebase-token-generator-ruby | Firebase Token Generator for Ruby |
| firebase-token-generator-node | Firebase Token Generator for Node |
| **Security Rules** | |
| blaze_compiler | The blaze compiler simplifies building security rules for your Firebase database. It drastically reduces the amount of copy and pasting involved. Blaze compiler security rules are shorter, and the syntax is less fussy. |
| **Data Management** | |
| vulcan | Realtime data viewer and editor for Chrome Dev Tools. |
| firebase-import | Helper utility for importing large JSON files into your Firebase database. It breaks the JSON into smaller chunks and uploads them individually through the Firebase API. |
| firebase-streaming-import | Utilizes ijson python json streaming library along with requests to import a large (>250 MB) json piecemeal into Firebase. |

## From Community

| Name | Description |
| --- | --- |
| firebase-admin | Programmatically instantiate and modify Firebase instances. |
| **my-firebase-toolkit** | |
| firebase-http-server | HTTP server that understands the firebase.json file and (approximately) emulates Firebase Hosting. Especially useful if you've set up any rewrites, since otherwise you can't really test your app without uploading it to Firebase. |
| firebase-on-error | Monkeypatches Firebase so that you can install a global error handler that will be triggered when any operation returns an error.<br>This is useful if you want to report all errors in a single spot, or crash-and-burn when anything goes wrong. |
| nodefire | A promise-centric Firebase library for NodeJS |
| **High-Level Tools** | |
| pyro | pyro provides a high level dsl for interacting with a Firebase datastore |
| firebase-login-custom | Wrapper to Firebase custom login including all dependencies with the exception of Firebase itself |
| firebase-cli | |
| **Support for Frameworks** | |
| rest-firebase | Ruby Firebase REST API client built on top of rest-core |


https://www.firebase.com/blog/2014-06-23-geofire-two-point-oh.html




# GeoFire

powered by Firebase

## What is GeoFire?

GeoFire is an open-source JavaScript library that allows you to store and query a set of items based on their geographic location. GeoFire uses Firebase for data storage, allowing query results to be updated in realtime as they change. GeoFire does more than just measure the distance between locations; it selectively loads only the data near certain locations, keeping your applications light and responsive, even with extremely large datasets.

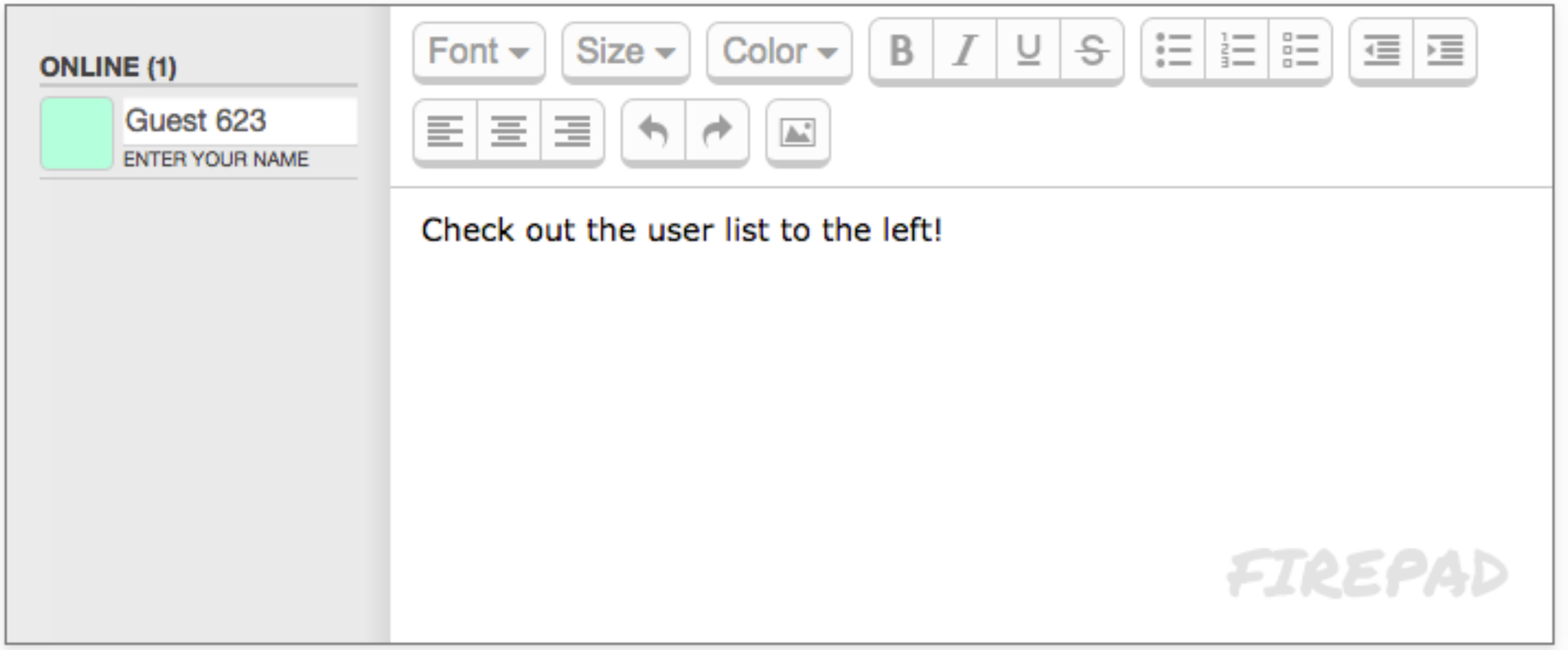
## About the demo

Firebase provides a Transit Open Data Set which contains the realtime locations of public transit vehicles in several major US cities, including San Francisco MUNI. The data set also contains GeoFire data for each vehicle which we used to create this demo. Drag around the purple circle to the left to see the vehicles which are currently within its radius. The results update in realtime as you move the circle and as vehicles travel around the city. GeoFire handles all of the hard work, telling you exactly when vehicles enter and exit the circle. It also selectively loads only the data geographically close

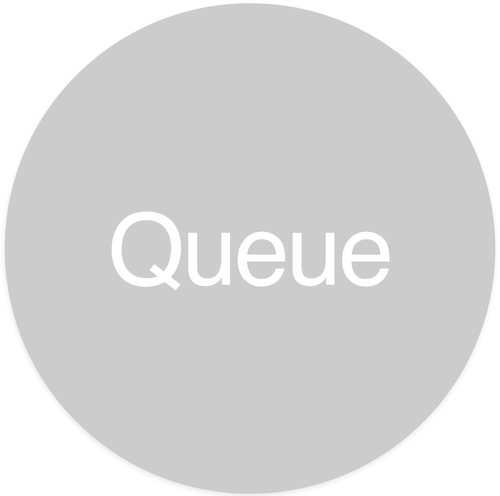
Download

```
// Create GeoFire instance
var firebaseRef = new Firebase("https://<your-app>.firebaseio.com/geofire/");
var geoFire = new GeoFire(firebaseRef);

// Set a key
geoFire.set("some_key", [37.785326, -122.405696]).then(function() {
  console.log("Provided key has been added to GeoFire");
}, function(error) {
  console.log("Error: " + error);
});

// Get a key
geoFire.get("some_key").then(function(location) {
  if (location === null) {
    console.log("Provided key is not in GeoFire");
  }
  else {
    console.log("Provided key has a location of " + location);
  }
}, function(error) {
  console.log("Error: " + error);
});

// Run a query
var geoQuery = geoFire.query({
  center: [37.4, -122.6],
  radius: 1.609 //kilometers
});

// Query result callback
geoQuery.on("key_entered", function(key, location, distance) {
  console.log("Bicycle shop " + key + " found at " + location + " (" + distance + " km away)");
});

```


http://www.firepad.io/


ONLINE (1)
Guest 623
ENTER YOUR NAME
Font
Size
Color
B
I
U
S
Check out the user list to the left!
FIREPAD

```
<div id="firepad"></div>
<script>
  var firepadRef = new Firebase('<FIREBASE URL>');
  var codeMirror = CodeMirror(document.getElementById('firepad'), { lineWrapping: true });
  var firepad = Firepad.fromCodeMirror(firepadRef, codeMirror,
      { richTextShortcuts: true, richTextToolbar: true, defaultText: 'Hello, World!' });
</script>

```




https://www.firebase.com/blog/2015-05-15-introducing-firebase-queue.html

- If you’re writing your server code in Node, Firebase Queue makes it easy to handle background jobs with a worker queue.
- Firebase using it in private backups feature, where they reliably handle hundreds of jobs per day across multiple machines.

queue_populator.js

Raw

```
var Firebase = require('firebase');

// Same ref as in queue_worker.js
var ref = new Firebase('https://<your-firebase>.firebaseio.com/queue');

// This doesn't need to be set every time, but helps us
// define the spec for the task in this example
ref.child('specs').set({
  task_1: {
    in_progress_state: 'task_1_in_progress',
    timeout: 10000
  }
});

// Add tasks onto the queue
var taskNumber = 0;
setInterval(function() {
  ref.child('tasks').push({
    taskNumber: ++taskNumber
  });
}, 1000);
```

queue_worker.js

Raw

```
var Queue = require('firebase-queue'),
    Firebase = require('firebase');

// The location of the Queue - can be any Firebase Location
var ref = new Firebase('https://<your-firebase>.firebaseio.com/queue');

// Creates the Queue
var options = {
  specId: 'task_1',
  numWorkers: 10
};

var queue = new Queue(ref, options, function(data, progress, resolve, reject) {
  // Read and process task data
  console.log(data);

  // Do some work
  var percentageComplete = 0;
  var interval = setInterval(function() {
    percentageComplete += 20;
    if (percentageComplete >= 100) {
      clearInterval(interval);
    } else {
      progress(percentageComplete);
    }
  }, 1000);

  // Finish the task
  setTimeout(function() {
    resolve();
  }, 5000);
});
```

# Showcase

firebase.com/customers


Firebase powers Koding's team collaboration app


Firepad (and Firebase) powers Nitrous.IO's collaborative text editing features


Firebase powers Stash, Atlassian's (enterprise) collaborative Git management feature


Firebase powers Instacart's collaborative "Show with Friends" feature


Twitch uses Firebase to notify users when a stream goes live


Summon is an iPhone app that uses Firebase to get you a ride instantly


# Case Studies

firebase.com/customers


# GoToMeeting & Talkboard

instant video conferencing & realtime collaborative whiteboarding

Using Firebase as the core of the application, the GoToMeeting team was able to

save one month of development time.

“GoToMeeting is built with WebRTC and uses the Firebase JavaScript client. With Firebase's instant notifications of model changes, it was easy to implement the signaling required to establish WebRTC connections.”

MATHIEU HOFMAN @ CITRIX

“We also have a Node server that communicates with Firebase to provision and create sessions. When a user opens GoToMeeting, the app makes a call to our Node server asking to join a specific meeting and the server returns an authentication token for that session.”

MATHIEU HOFMAN @ CITRIX

“Using Firebase lets us keep things simple on a stateless Node server.”

MATHIEU HOFMAN @ CITRIX

Using Firebase, the Citrix Talkboard team was able to

cut their development time by six months.

“The API was so simple that we implemented all the communications and data synchronization in a bit more than a week.”

FREDERIC MAYOT @ CITRIX

“Thanks to disk persistence, we allow our users to access and edit their projects and whiteboards anywhere, anytime, without worrying about being connected to the Internet.”

FREDERIC MAYOT @ CITRIX

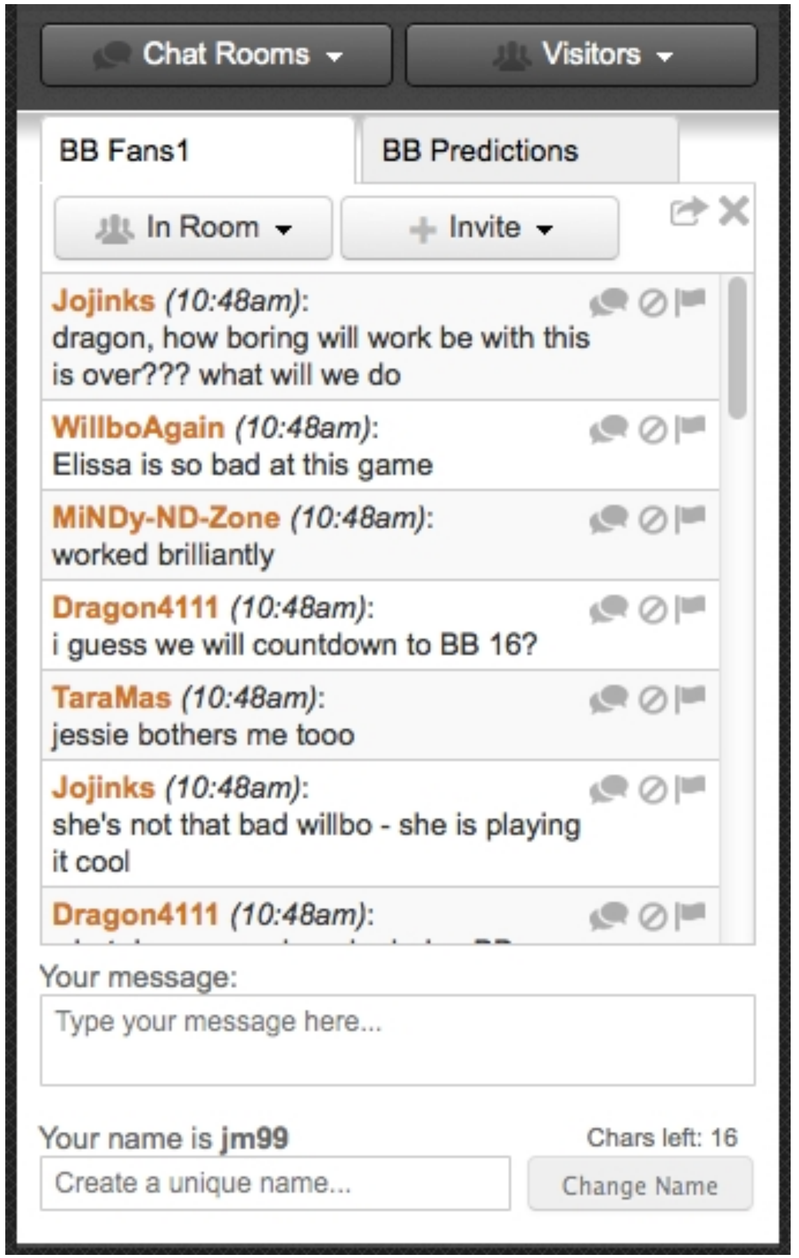
“We were very impressed to see such low latency, which really enabled the magic of showing the strokes being updated in real-time across devices. As questions came up in the development process, the Firebase team was very receptive and reactive.”

FREDERIC MAYOT @ CITRIX


# Big Brother Live Feeds

**24/7 live-streaming subscription service**
**for fans of CBS Television's BIG BROTHER.**

- Firebase powered the Big Brother Live Feed's multi-room chat during the show's 2013 season.
- CBS.com's goal was to get chat services up and running quickly with custom features for the Live Feeds experience.
- CBS.com needed the flexibility on the backend to customize the service and tune chat features as the season went on.



The Firebase-powered chat service was tailor made to complement the experience of viewing the Big Brother Live Feeds and became **one of the most highly used features of the experience.**

Throughout the season, **over 27M chat entries were logged**. Big Brother Live Feeds subscribers made use of many social features available via the chat service, including:

- Chatting with other users in real-time
- Creating private chat rooms focused around a specific topic
- Participating in multiple chat rooms at one time
- Inviting a user to a one-on-one chat
- Live type-ahead search for all users logged into the Big Brother Live Feeds site
- Moderation tools: the ability to mute users from a chat or flag posts for deletion


# Interactive Physical Surfaces

**with the goal of helping emergency response teams better prepare for and manage emergency situations**

“Where we used to spend 50% of our development time building the backend, we can focus now 95% of our time on the actual application.”

STEPHEN GUERIN @ SIMTABLE


Firebase powered gaming backend

“How do I shard this?”
“Which servers do I spin-up and spin-down?”
“What are we going to do when it grows?”

RILEY DUTTON @ ROLL20

“I was dreading managing it myself.
I'm not in the business of real-time server management.
We are trying to be in the business of creating a gaming platform.”

RILEY DUTTON @ ROLL20

“I didn't have to spin up new servers, worry about additional capacity, or think about downtime.”

*“I just watched the number go up.”*

RILEY DUTTON @ ROLL20


# LiveShare

realtime presentation tool for designers

# Firebase Powered Features

- **Data storage:** When the meeting presenter opens a whiteboard, all of the sketches created are stored in Firebase. Participants who join a meeting after it starts are automatically shown all existing sketches. According to Shawn, "the flexible JSON data structure of Firebase allowed us to make painless changes to the data model during development without the cumbersome overhead of ALTER statements or a need to sync across development environments.""
- **AngularJS Integration:** InVision is built on AngularJS, and the team used Firebase's AngularFire bindings to build LiveShare. Shawn commented "the smooth and easy integration with AngularFire was like a miracle. The time savings and ease of use has been incredible."
- **Presence:** LiveShare uses Firebase’s presence system to keep track of when participants join and leave a LiveShare. Each participant’s avatar appears and disappears from the LiveShare bar in response to the presence callbacks in Firebase.

“Once I heard about Firebase and tried it, there was no turning back...”

SHAWN GRIGSON @ INVISION

“Gone was the necessity of animating transitions for cursors. We simply stream over a series of x and y coordinates, and watch the magic of Firebase display the participants' actions in realtime.”

SHAWN GRIGSON @ INVISION

“The speed of Firebase cut out a huge portion of bloated and buggy code and allowed us to focus on more meaningful issues.”

*“Firebase is the engine that drives LiveShare.”*

SHAWN GRIGSON @ INVISION

Pricing

| | Free | Spark | Candle | Bonfire | Blaze | Inferno |
|---|---|---|---|---|---|---|
| | $0 forever | $5 per month | $49 per month | $149 per month | $449 per month | $1,499 per month |
| **REALTIME DATABASE** | | | | | | |
| Connections | 100 | 100 | UNLIMITED* | UNLIMITED* | UNLIMITED* | UNLIMITED* |
| Storage | 1 GB | 1 GB | 10 GB | 30 GB | 100 GB | 300 GB |
| Transfer | 10 GB | 10 GB | 50 GB | 150 GB | 500 GB | 1.5 TB |
| Private Backups | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| **AUTHENTICATION** | | | | | | |
| Users | UNLIMITED | UNLIMITED | UNLIMITED | UNLIMITED | UNLIMITED | UNLIMITED |
| **HOSTING** | | | | | | |
| Storage | 1 GB | 1 GB | 10 GB | 10 GB | 10 GB | 10 GB |
| Transfer | 100 GB | 100 GB | 1 TB | 1 TB | 1 TB | 1 TB |
| Custom Domain | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | Sign Up | Purchase | Purchase | Purchase | Purchase | Purchase |

# Let’s code

BLOG + CMS

Firebase to store application data

AngularJS front-end for the blog and admin interface

Admin server for content management functionality

https://github.com/sbolel/launch-annapolis


# What do you think?

FEEDBACK & QUESTIONS