Firebase

Google Firebase (NoSQL on Android (, iOS & Web))

A short introduction

Speaker

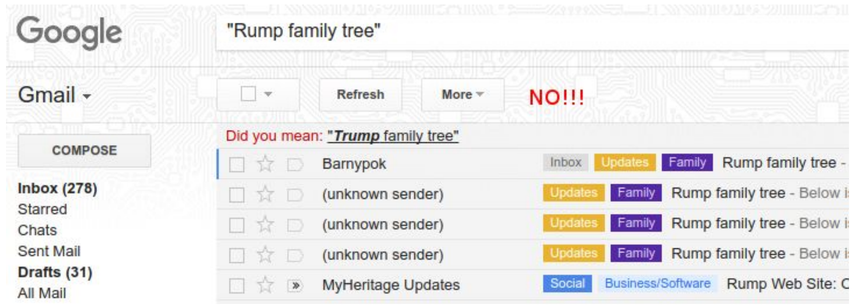
Handle: Andrew Rump (andrew+meetup@rump.dk)

Email: and_comp+meetup@rump.dk (for easy filtering)
LinkedIn: https://www.linkedin.com/in/andrewrump/ (not looking for a job)
Twitter: andrewrump - mostly about (T)rump & Android right now

Work for BusinessNow implementing ServiceNow

BusinessNow Make sense servicenow

Feel free to ask questions at any time



Did you mean: "*Trump* family tree"

Barnypok	Inbox Updates Family	Rump family tree -
(unknown sender)	Updates Family	Rump family tree - Below i
(unknown sender)	Updates Family	Rump family tree - Below i
(unknown sender)	Updates Family	Rump family tree - Below i
MyHeritage Updates	Social Business/Software	Rump Web Site: C

Why Google Firebase?

Needed a (RESTful) (database) API for my WhereIs… app

php

{ REST }

MySQL

We all know what the leads to...

What does Google Firebase provide?

- Authentication
  - Username, Google, Facebook, ...
- Real time database
  - NoSQL, offline-”capability”
- Storage
- Hosting
- Test lab
- Crash reporting
- Cloud messaging

Analytics, ...

- Notification
- Remote config
- App Indexing
- Dynamic links
- Invites
- Adwords
- AdMob

More to come…

- Fabric (Twitter), Crashlytics, ...

Firebase crash reporting & database rules

Demo

Firebase real time database field types

- Standard types
  - String
  - Long
  - Double
  - Boolean
  - Map<String, Object>
  - List<Object>
- Array when key are numbers in sequence
- Java class (only requirement: a default constructor)
  - Member variables become key values
  - **NOTE**: Use update() when writing or all other values/child nodes are wiped away!

Firebase real time JSON database (Don’t)

```
{ // Not the way to do it!!!
  "chats": {
    "one": {
      "title": "Historical Tech Pioneers",
      "messages": {
        "m1": { "sender": "ghopper", "message": "Relay malfunction found. Cause: moth." },
        "m2": { ... },
        // a very long list of messages
      }
    },
    "two": { ... }
  }
}
```

Firebase real time JSON database (part 1)

```
{
  // Chats contains only meta info about each conversation
  // stored under the chats's unique ID
  "chats": {
    "one": {
      "title": "Historical Tech Pioneers",
      "lastMessage": "ghopper: Relay malfunction found. Cause: moth.",
      "timestamp": 1459361875666
    },
    "two": { ... },
    "three": { ... }
  },
```

Firebase real time JSON database (part 2)

```
// Conversation members are easily accessible
// and stored by chat conversation ID
"members": {
  // we'll talk about indices like this below
  "one": {
    "ghopper": true,
    "alovelace": true,
    "eclarke": true
  },
  "two": { ... },
  "three": { ... }
},
```

Firebase real time JSON database (part 3)

```
  "messages": {
    "one": {
      "m1": {
        "name": "eclarke",
        "message": "The relay seems to be malfunctioning.",
        "timestamp": 1459361875337
      },
      "m2": { ... },
      "m3": { ... }
    },
    "two": { ... },
    "three": { ... }
  }
}
```

Firebase NoSQL real time database

- Online & offline-support
  - FirebaseDatabase.getInstance().setPersistenceEnabled(true);
- All users share the same online Firebase database
  - Security rules may be used to separate users and/or restrict content
- Authentication may (read: should) be applied
- No schema
- Mobile devices (Android & iOS) and web
- Notifies all devices on changes - if listening
- 10MB database stored on the device

// Write to your database

```
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference myRef = database.getReference("message");
```

Async, i.e. you can call the methods in your main (UI) thread - beware of size

```
myRef.setValue("Hello, World!");

myRef.push().setValue(“Unique”);

    -K2WLjgH0es40OGWp6Ln: “Unique”
    -K2YyDkM4lUotI12OnOs: “Unique”

myRef.setKey(“key”).setValue(“value”);

myRef.setKey(“key”).update(“update”);
```

Data may be collected and written in one go as Map or List

// Read from the database

```
myRef.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        // This method is called once with the initial value and again
        // whenever data at this location is updated.
        String value = dataSnapshot.getValue(String.class);
        Log.d(TAG, "Value is: " + value);
    }

    @Override
    public void onCancelled(DatabaseError error) {
        Log.w(TAG, "Failed to read value.", error.toException());
    }
});
```

// Read from the database (once)

```
myRef.addListenerForSingleValueEvent(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        // This method is called once with the initial value and again
        // whenever data at this location is updated.
        String value = dataSnapshot.getValue(String.class);
        Log.d(TAG, "Value is: " + value);
    }

    @Override
    public void onCancelled(DatabaseError error) {
        Log.w(TAG, "Failed to read value.", error.toException());
    }
});
```

More Firebase real time database functionality

- ValueEventlistener vs. ChildEventListener
  - onChildAdded()
  - onChildChanged()
  - onChildRemoved()
  - onChildMoved()
- removeEventListener

May be bound directly to RecyclerView/UITableView/Angular/.../.../...

Firebase database demo

Demo

Any questions (about Firebase (database))?