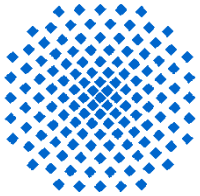


Universität Stuttgart

Institute of Parallel and
Distributed Systems (IPVS)

Universitätsstraße 38
D-70569 Stuttgart

Mobile Computing Lab

Assignment 5

Realtime Databases

Frank Dürr, Saravana Murthy, Ahmad Slo, Zohaib Riaz

Outline

- Realtime Databases
- Firebase Database
- Task 1: Read/Write
- Task 2: Subscribe to Changes
- Organizational issues

Realtime Databases

Mobile applications require data shared between multiple devices

Examples:

- Route-Planning App receives traffic reports
- Social Networks: Users post messages, other users get notified

Challenges:

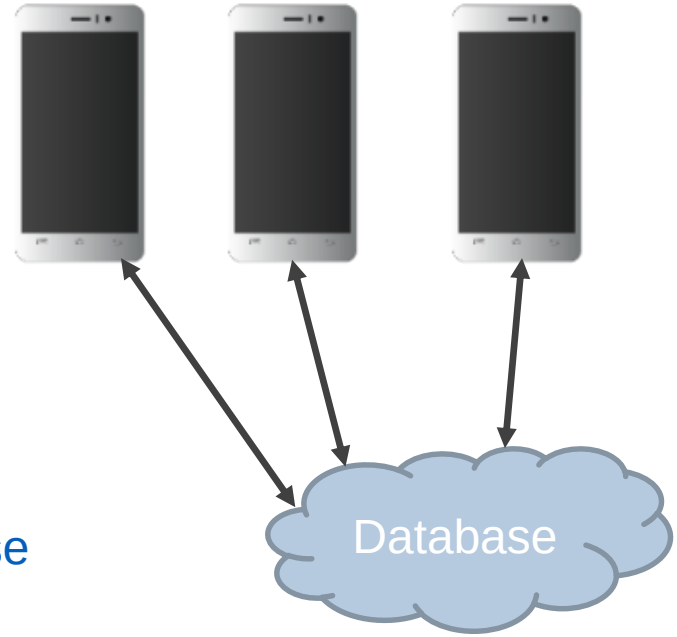
- Make changes to database available immediately
- Huge number of mobile devices
- Devices not always connected
- Energy efficiency
- …

Realtime Databases

Use Realtime Databases as an abstraction!

Operations on such a Database:

- Read
- Write
- Subscribe



Example: Publish sensor values to other devices

- Devices read sensor values and update database
- Other devices subscribe to updates
- Devices receive notification when value is changed

Firebase Database

Firebase provides Realtime Database

- Available for Android, iOS, in the Browser
- Background: Firebase got acquired by Google, included into Google Services

Infrastructure for Database is provided by Firebase

- Simply store data, no need to setup own server
- You need to pay (if you have more than 1.000 clients)
- Privacy

Event Driven Programming: Callbacks for reads/notifications

NoSQL Database: JSON **Tree**

Firebase Database: Tree Structure

Data in Firebase Database is Stored as **JSON Tree**

Get Root of Instance as **`DatabaseReference`**

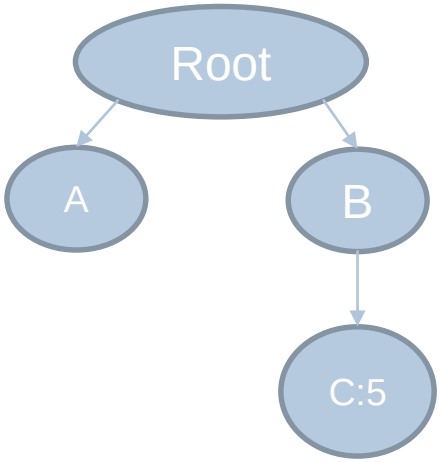
```
> DatabaseReference mRef = FirebaseDatabase.getInstance().getReference()
```

Access to Subtree using **`child()`**

```
> DatabaseReference NodeA = mRef.child("A")
```

Update Subtree using **`setValue()`**

```
> mRef.child("B").child("C").setValue(5)
```



Firebase Database: Concurrent Writes

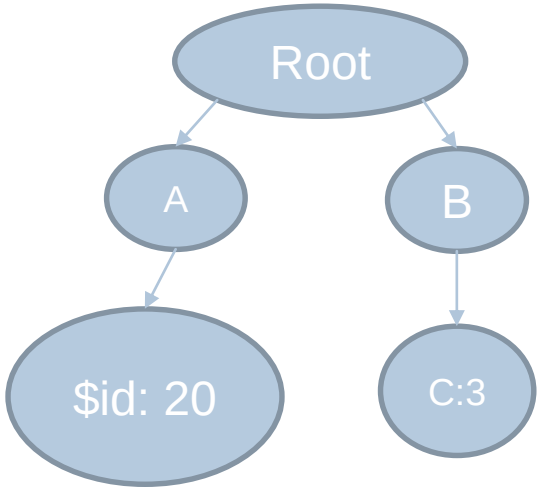
Problem:

- Append to list is common operation
- **Write Conflicts** if multiple users want to append data

Solution: Use **`push()`**

- **`push()`** generates subtree with unique identifier
- Returns **`DatabaseReference`**

```
> nodeA.push().setValue(20)
```



Firebase Database: Read Values

Read is asynchronous using callbacks

`ValueEventListener`: all changes in subtree

`ChildEventListener`: only changes of childs

Read once:

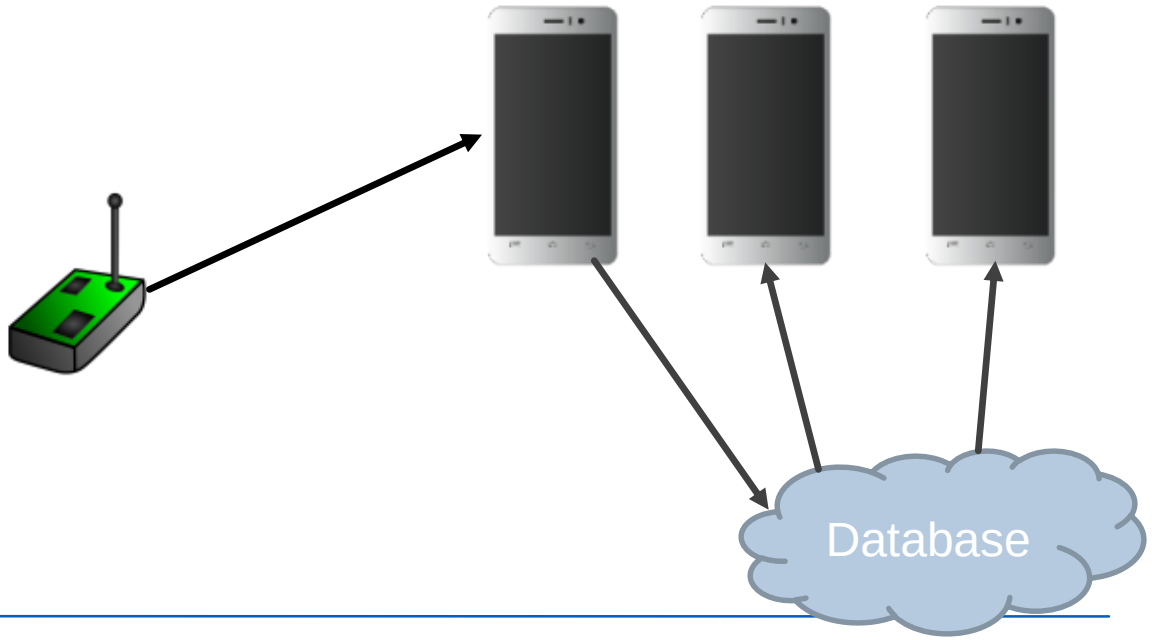
```
> childRef.addListenerForSingleValueEvent(new ValueEventListener() {
>     public void onDataCahge(DataSnapshot dataSnapshot) {
>         Integer i = dataSnapshot.getValue(Integer.class);
>     }
>     // Override other methods
> });
```

Subscribe: use **`addValueEventListener`** or **`addChildEventListener`**

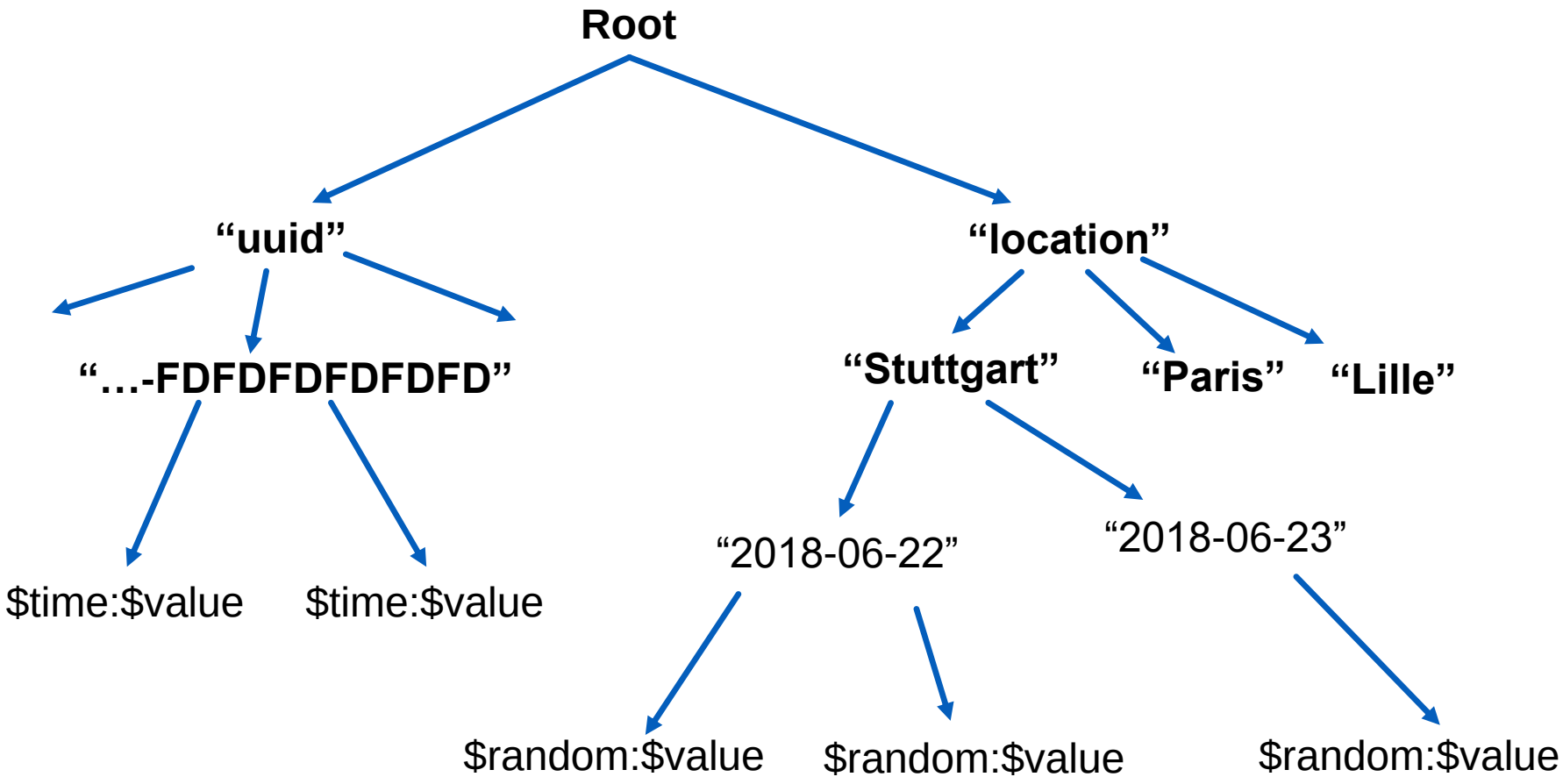
Application: Update Temperature Values

IoT Application:

- BLE Assignment: read temperature values from sensor
- Now: Subscribe to sensor -- get notified when sensor value updated
- Mobile device updates database when sensor is read
- Database notifies other mobile devices



Database Structure for Temperature Values



Task 1: Write/Read from Database

1.1: Write to Database:

- BLE Assignment: Scan for Sensors and read values
- Add new value to datastructure of Database
  - .. to uuid subtree
  - .. to location subtree (assume you are always in “Stuttgart”)
- Use System.currentTimeMillis().toString() as timestamp for uuid subtree

1.2: Read from Database

- Read Previous value from one sensor, identified by uuid
- Present last update to the user (time and value)

Use the `/teams/$teamnumber/` subtree for testing

Task 2: Subscribe to Temperature Changes

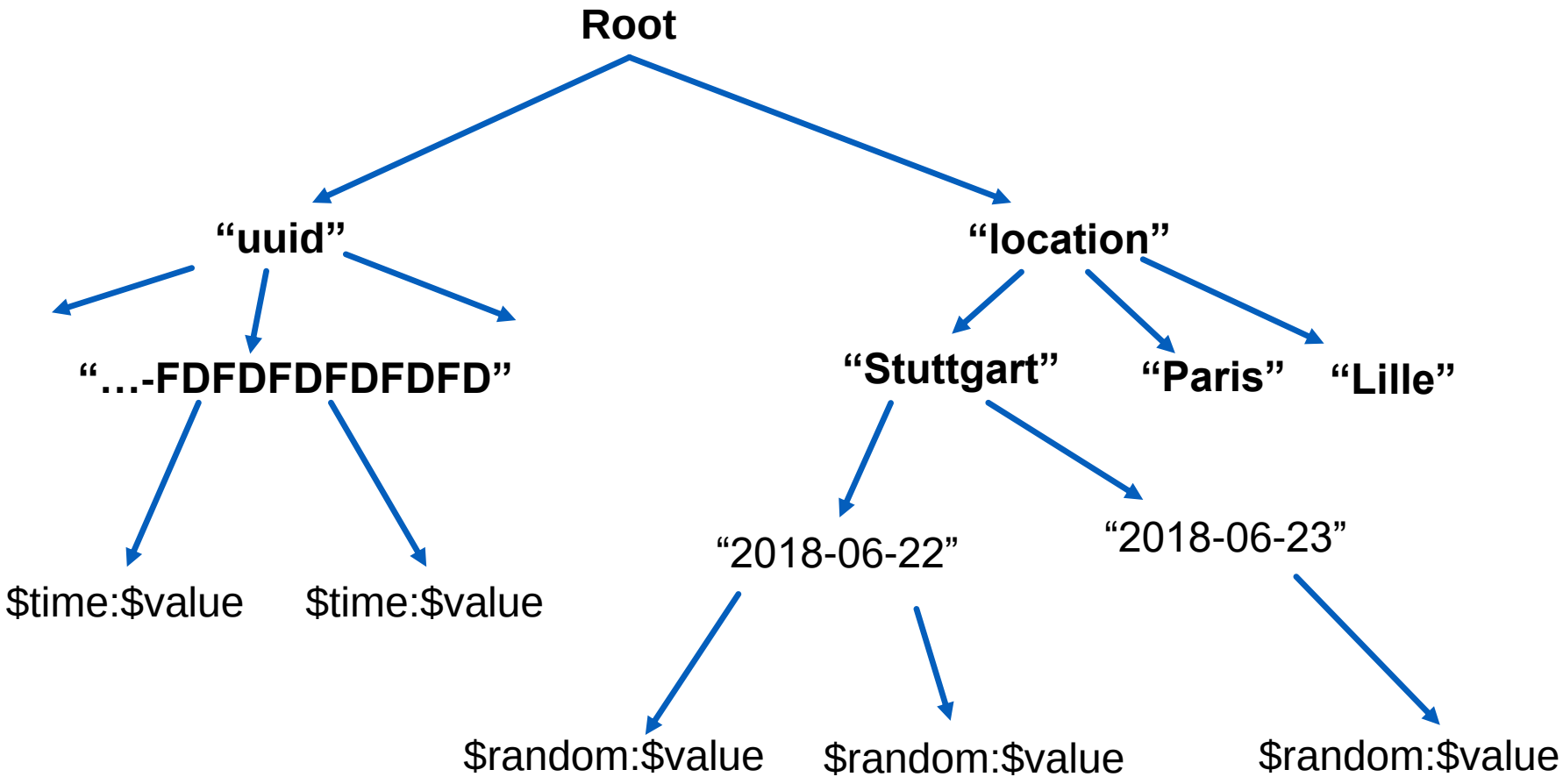
2.1: Continuously display latest value of sensor readings

- User can select sensor by uuid
- Any other device can update the sensor value
- User sees update immediately

2.2: Show average temperature of today at one location

- User selects location
- App shows average temperature of the day
- Average temperature updates when new readings become available

Database Structure for Temperature Values



Setting up Firebase Database

General basic steps to use Firebase Database

1. Add external libraries to Gradle build
2. Request JSON file including App-Key from the Firebase Console
3. Add JSON file to Android Studio

More information available at
https://firebase.google.com/docs/database/android/start/

Setup for this Assignment:

- JSON File with API-Key available via ILIAS
- Use package name: **de.uni_s.ipvs.mcl.assignment5**
- You can setup own Database for debugging (otherwise no access to Console)
- Setting up own Database requires Google Account

Setup: Modify Gradle Scripts

On Module Level

```
dependencies {
  // ...
  compile
'com.google.firebase:firebase-
core:9.0.2'
}

// ADD THIS AT THE BOTTOM
apply plugin: 'com.google.gms.google-
services'
```

On Project Level

```
buildscript {
    // ...
    dependencies {
        // ...
        classpath
'com.google.gms:google-
services:3.0.0'
    }
}
```

More Information:

- https://firebase.google.com/docs/android/setup
- https://firebase.google.com/docs/database/android/start/

Links

https://firebase.google.com/docs/database/

General information about the Firebase Database

https://firebase.google.com/docs/database/android/start/

Setup and basic functions of the Firebase Database

https://www.youtube.com/watch?v=tb2GZ3Bh4p8

Firebase overview talk at Google IO 2016

https://firebase.google.com/console/

Firebase Console, if you want to create your own database (requires Google Account)

Submission & Next Meeting

- Post questions on ILIAS forum

- You have 2 **weeks time** to work on this assignment
  - Demonstration of your results scheduled for **Wednesday July 18th 2018**

- **Submit via Ilias**
  - **Source code**
  - Group submission!

Questions?