Application Development for Mobile and Ubiquitous Computing

TimeTracker - Final Presentation

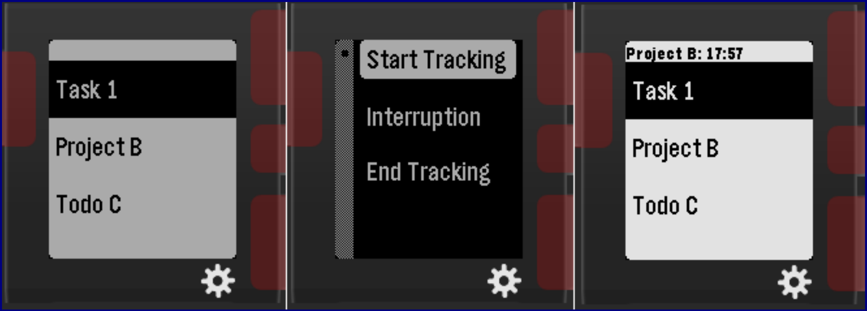
Robin Gander - Riccardo Steffan

Technische Universität Dresden - Faculty of Computer Science

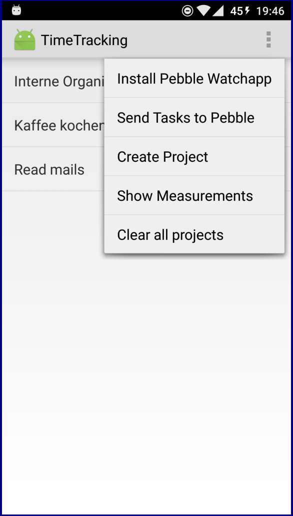
26.01.2018

Summary

TimeTracker



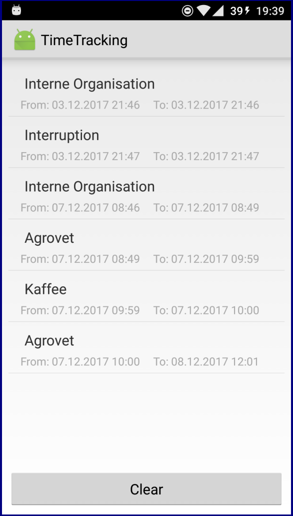
TimeTracker



45 19:46

TimeTracking

Interne Organi
Kaffee kochen
Read mails



39 19:39

TimeTracking

Interne Organisation
From: 03.12.2017 21:46 To: 03.12.2017 21:46

Interruption
From: 03.12.2017 21:47 To: 03.12.2017 21:47

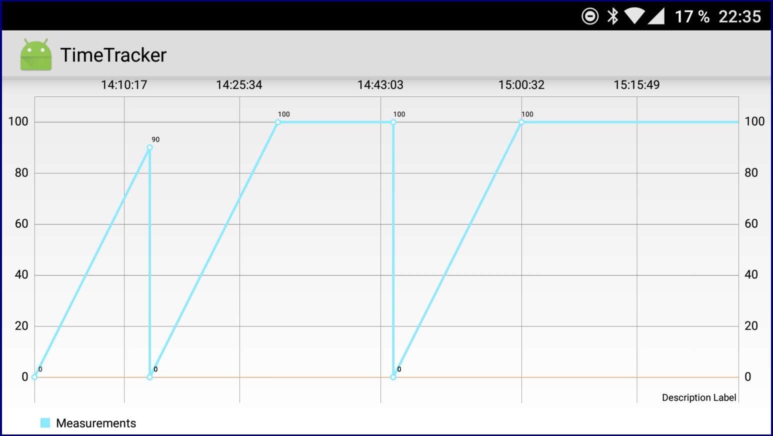
Interne Organisation
From: 07.12.2017 08:46 To: 07.12.2017 08:49

Agrovet
From: 07.12.2017 08:49 To: 07.12.2017 09:59

Kaffee
From: 07.12.2017 09:59 To: 07.12.2017 10:00

Agrovet
From: 07.12.2017 10:00 To: 08.12.2017 12:01

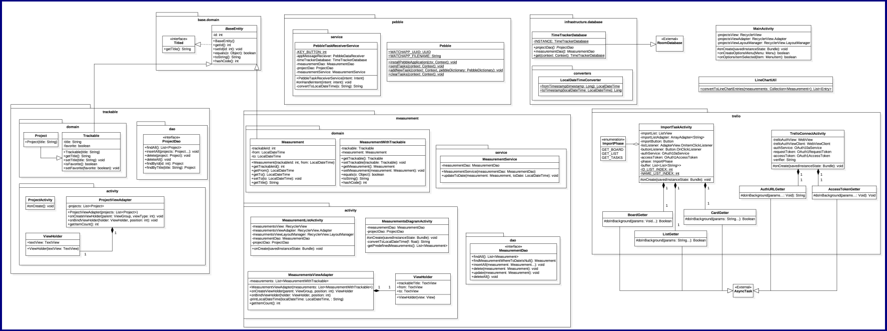
TimeTracker



Architecture: Subsystems

- **base.domain**: definition of base entity
- **trackable**: definition of trackable items (i.e. projects) and related Android activities and DAO
- **measurement**: definition of measurement items and related Android activities and DAO
- **infrastructure.database**: app persistent storage
- **pebble**: pebble integration, i.e. sending and receiving tasks and measurements to/from the app
- **trello**: Android activities for connecting to a Trello account and import tasks from it

Architecture: UML class diagram

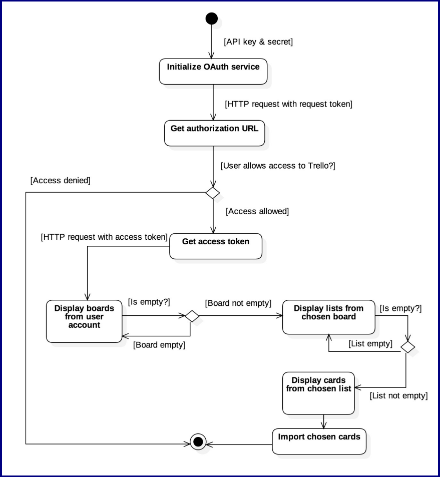

base.domain
«interface» Titled
+getTitle(): String
BaseEntity
-id: int
+BaseEntity()
+getId(): int
+setId(id: int): void
+equals(o: Object): boolean
+toString(): String
+hashCode(): int
trackable
domain
Project
+Project(title: String)
Trackable
-title: String
-favorite: boolean
+Trackable(title: String)
+getTitle(): String
+setTitle(title: String): void
+isFavorite(): boolean
+setFavorite(favorite: boolean): void
dao
«interface» ProjectDao
+findAll(): List<Project>
+insertAll(projects: Project...): void
+delete(project: Project): void
+deleteAll(): void
+findById(id: int): Project
+findByTitle(title: String): Project
activity
ProjectActivity
#onCreate(): void
ProjectViewAdapter
-projects: List<Project>
+ProjectViewAdapter(projects: List<Project>)
+onCreateViewHolder(parent: ViewGroup, viewType: int): void
+onBindViewHolder(holder: ViewHolder, position: int): void
+getItemCount(): int
ViewHolder
+textView: TextView
+ViewHolder(textView: TextView)
1
1
pebble
service
PebbleTaskReceiverService
-KEY_BUTTON: int
-appMessageReciever: PebbleDataReceiver
-timeTrackerDatabase: TimeTrackerDatabase
-measurementDao: MeasurementDao
-projectDao: ProjectDao
-measurementService: MeasurementService
+PebbleTaskReceiverService(intent: Intent)
#onHandleIntent(intent: intent): void
-convertToLocalDateTime(s: String): String
Pebble
+WATCHAPP_UUID: UUID
+WATCHAPP_FILENAME: String
+installPebbleApplication(ctx: Context): void
+sendTasks(context: Context): void
+addNewTask(context: Context, pebbleDictionary: PebbleDictionary): void
+clearTasks(context: Context): void
infrastructure.database
TimeTrackerDatabase
-INSTANCE: TimeTrackerDatabase
+projectDao(): ProjectDao
+measurementDao(): MeasurementDao
+get(context: Context): TimeTrackerDatabase
«External» RoomDatabase
converters
LocalDateTimeConverter
+fromTimestamp(timestamp: Long): LocalDateTime
+toTimestamp(localDateTime: LocalDateTime): Long
MainActivity
-projectsView: RecyclerView
-projectsViewAdapter: RecyclerView.Adapter
-projectsViewLayoutManager: RecyclerView.LayoutManager
#onCreate(savedInstanceState: Bundle): void
+onCreateOptionsMenu(Menu: Menu): boolean
+onOptionsItemSelected(item: MenuItem): boolean
LineChartUtil
+convertToLineChartEntries(measurements: Collection<Measurement>): List<Entry>
measurement
domain
Measurement
-trackableId: int
-from: LocalDateTime
-to: LocalDateTime
+Measurement(trackableId: int, from: LocalDateTime)
+getTrackableId(): int
+getFrom(): LocalDateTime
+getTo(): LocalDateTime
+setTo(to: LocalDateTime): void
+getTitle(): String
MeasurementWithTrackable
-trackable: Trackable
-measurement: Measurement
+getTrackable(): Trackable
+setTrackable(trackable: Trackable): void
+getMeasurement(): Measurement
+setMeasurement(measurement: Measurement): void
+equals(o: Object): boolean
+toString(): String
+hashCode(): int
service
MeasurementService
-measurementDao: MeasurementDao
+MeasurementService(measurementDao: MeasurementDao)
+updateToDate(measurement: Measurement, toDate: LocalDateTime): void
activity
MeasurementListActivity
-measurementsView: RecyclerView
-measurementsViewAdapter: RecyclerView.Adapter
-measurmentsViewLayoutManager: RecyclerView.LayoutManager
-measurementDao: MeasurementDao
-projectDao: ProjectDao
+onCreate(savedInstanceState: Bundle): void
MeasurementsDiagramActivity
-measurementDao: MeasurementDao
-projectDao: ProjectDao
#onCreate(savedInstanceState: Bundle): void
-convertToLocalDateTime(f: float): String
-getPredefinedMeasurements(): List<Measurement>
MeasurementsViewAdapter
-measurements: List<MeasurementWithTrackable>
+MeasurementViewAdapter(measurements: List<MeasurementWithTrackable>)
+onCreateViewHolder(parent: ViewGroup, position: int): ViewHolder
+onBindViewHolder(holder: ViewHolder, position: int)
-printLocalDateTime(localDateTime: LocalDateTime, : String)
+getItemCount(): int
1
1
ViewHolder
+trackableTitle: TextView
+from: TextView
+to: TextView
+ViewHolder(view: View)
dao
«interface» MeasurementDao
+findAll(): List<Measurement>
+findMeasurementWhereToDateIsNull(): Measurement
+insertAll(measurement: Measurement...): void
+delete(measurement: Measurement): void
+update(measurement: Measurement): void
+deleteAll(): void
trello
«enumeration» ImportPhase
GET_BOARD
GET_LIST
GET_TASKS
ImportTaskActivity
-importList: ListView
-importListAdapter: ArrayAdapter<String>
-importButton: Button
-listListener: AdapterView.OnItemClickListener
-buttonListener: Button.OnClickListener
-authService: OAuth10aService
-accessToken: OAuth1AccessToken
-phase: ImportPhase
-buffer: List<List<String>>
-ID_LIST_INDEX: int
-NAME_LIST_INDEX: int
#onCreate(savedInstanceState: Bundle): void
TrelloConnectActivity
-trelloAuthView: WebView
-trelloAuthViewClient: WebViewClient
-authService: OAuth10aService
-requestToken: OAuth1RequestToken
-accessToken: OAuth1AccessToken
-verifier: String
#onCreate(savedInstanceState: Bundle): void
AuthURLGetter
#doInBackground(params...: Void): String
AccessTokenGetter
#doInBackground(params...: Void): Void
BoardGetter
#doInBackground(params: Void...): Boolean
CardGetter
#doInBackground(params: String...): Boolean
ListGetter
#doInBackground(params: String...): Boolean
«External» AsyncTask

Trello Integration: Challenges

Development Challenges:

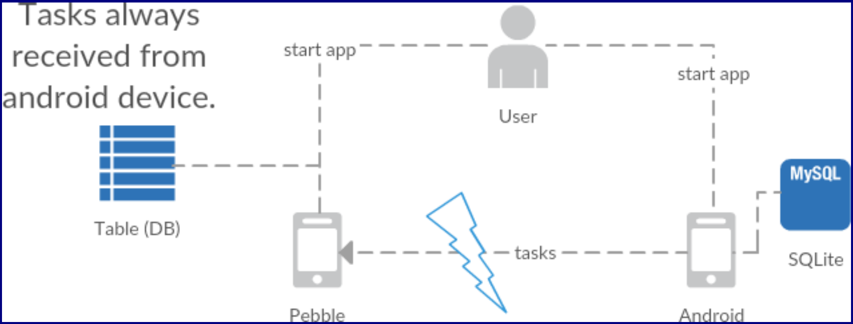
- **Authorize / Authenticate**: implemented using ScribeJava OAuth service, user must authorize access to its account so that an access token can be generated and used to sign subsequent HTTP requests to the Trello server.
- **Obtain project data**: three consecutive HTTP requests are made to the server via the ScribeJava OAuth service, which responds with the requested Trello object (board, list or cards) in JSON format.
- **Data selection**: server response is parsed from JSON to human readable and displayed through a single- (for boards and lists) or multi-choice (for cards) list layout so that the user can choose which items to import.

Trello Integration: Process View

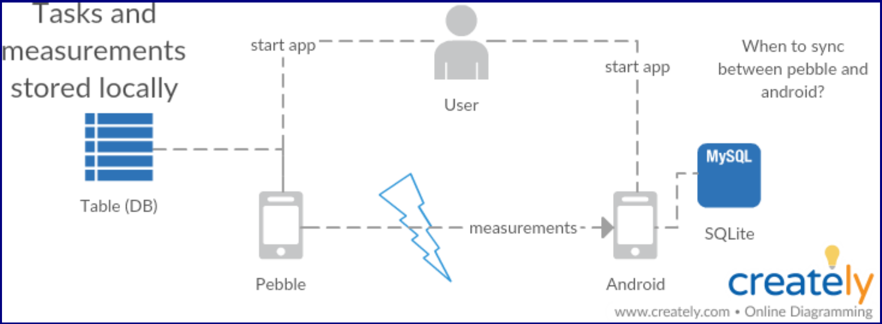


1. Initialize OAuth service with API key and secret to generate the request token.
2. Perform HTTP request with attached request token to obtain the authorization URL, i.e. where the user will authorize access to its Trello account.
3. After the user allows access, parse the resulting URI to obtain the OAuth verifier and use it with the request token to obtain an access token.
4. Obtain board/list/cards (i.e. tasks) data through HTTP requests signed with the access token.
5. Import the chosen tasks into the app internal database and return to main activity.

Implemented challenge - offline challenge 1



Implemented challenge - offline challenge 1


Tasks and measurements stored locally
Table (DB)
start app
User
start app
When to sync between pebble and android?
MySQL
SQLite
Pebble
measurements
Android
creately
www.creately.com • Online Diagramming

```
//at try to start measurement on pebble
DictionaryIterator *iter;
AppMessageResult result = app_message_outbox_begin(&iter);

if (result != APP_MSG_OK) {
  storeSelectedTaskOnPebble();
} else {
  if(sendStoredMeasurementsToPhone() != APP_MSG_OK) {
    storeSelectedTaskOnPebble();
  } else if(sendSelectedTaskToPhone() != APP_MSG_OK {
    storeSelectedTaskOnPebble();
  }
}
//is implemented as a single if statement in production
```

```
//the android side to receive data
this.appMessageReciever = new
    PebbleDataReceiver(Pebble.WATCHAPP_UUID) {

  @Override
  public void receiveData(Context context, int transactionId,
      PebbleDictionary data) {
    PebbleKit.sendAckToPebble(context, transactionId);
    /* parse the pebble data here */
  }
}
PebbleKit.registerReceivedDataHandler(getApplicationContext(),
    this.appMessageReciever);
```

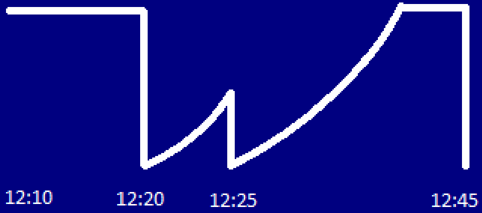
```
// Movement detection
health_service_events_subscribe(health_handler, NULL);

static void health_handler(HealthEventType event, void *context)
    {
  if (event == HealthEventMovementUpdate) {
    //if to inaccurate, can check step count with
    //health_service_sum(HealthMetricStepCount, now - seconds,
        now);
    showMovementUpdateMenu(); //first item is 'get back'
    vibes_short_pulse();
  }
}
```

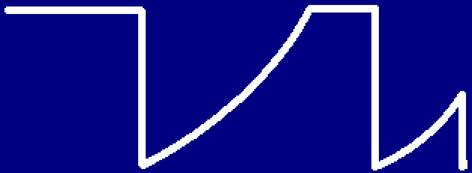
```
//simple usability implementations
PebbleKit.startAppOnPebble(getApplicationContext(),
    Pebble.WATCHAPP_UUID);
```

```
uint16_t clickDuration = 500;
window_long_click_subscribe(BUTTON_ID_SELECT, clickDuration,
    add_interruption_on_long_down_click, null);
//long up is null
```

Unimplemented challenge - form factor

- Idea: Store interruption instead of measurement





Technologies

1. Android API 19, 4.4 KitKat (Java 7)
2. Room Persistence Library
3. Pebble (C SDK)
4. PebbleKit for communication (send / receiver messages)
5. ScribeJava library for OAuth and HTTP communication