Camera Application

Final documentation

Mobile Project TTOW0630

Marek Korenciak

2016/2017 JAMK University of Applied Sciences

Content

Introduction

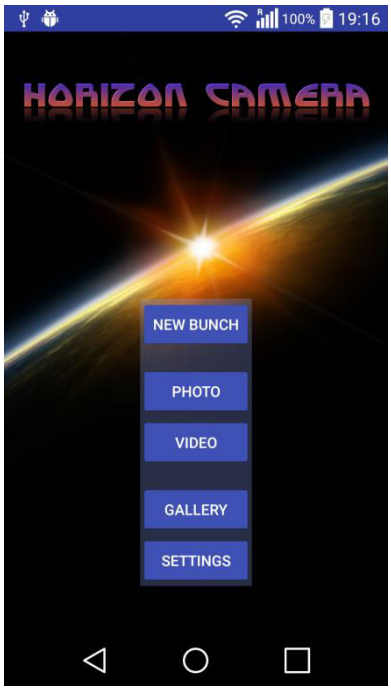
SolarGis is Slovak company providing long time solar forecast for almost whole of the world. They have lot of clients, who need the special information about solar irradiation on their concrete place. The company uses meteo satellites to gather basic data. They can make the map of the solar irradiation this way with at most 7% tolerance. But this data can be used mainly for the bigger areas. For example, we want to find place for build new solar power plant. We can choose, via the maps, some bigger area for this special purpose. However, how to find the best place in this bigger area?? We have to make the terrain research in this area. Problem is, that the solar irradiation of the chosen area can be influenced by the local terrain (forests, hills, buildings,...).

Planned application called Horizon Camera will be an unofficial Android application used for the special purposes of SolarGis. This application can make the terrain work easier in this company. Point is to take the pictures/video of the horizon. This pictures (frames of videos too) will have additional information- GPS position, orientation and angle from the gyroscope. All this data will be send to server side, where it will be processed and used for more exact expected solar irradiation for the place. Data can be stored temporarily in the mobile phone, until some internet connection appears.

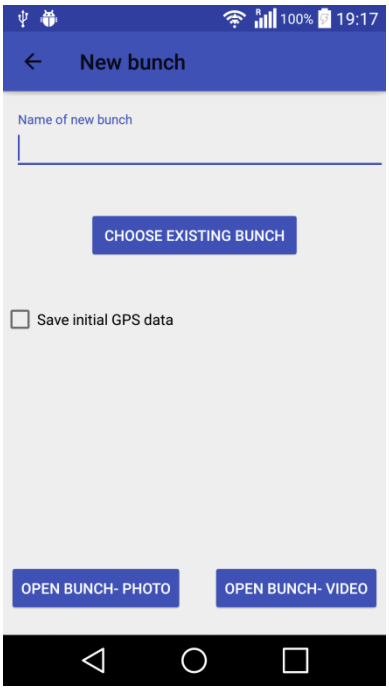
Processing and calculating of the gathered data on the server side are not part of this project.

Final application screens

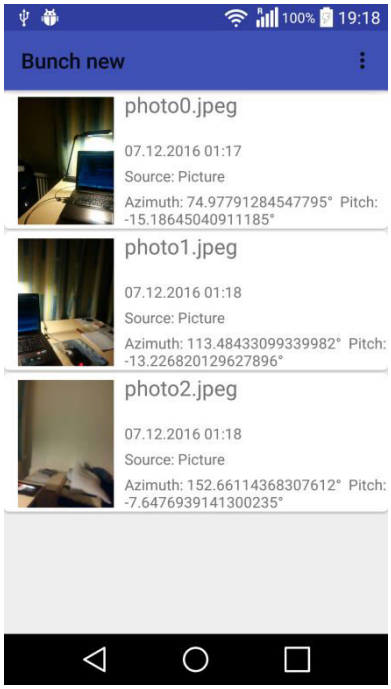
Final application has graphical and activity layout same like was planned in the planning documentation. Difference is just added new functionality to video mode- panorama.



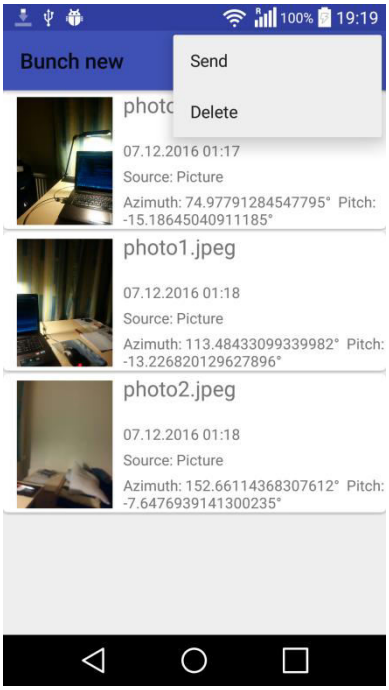
Main page



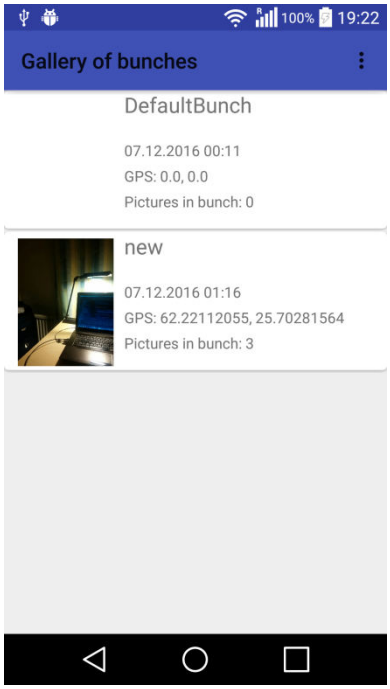
New bunch activity



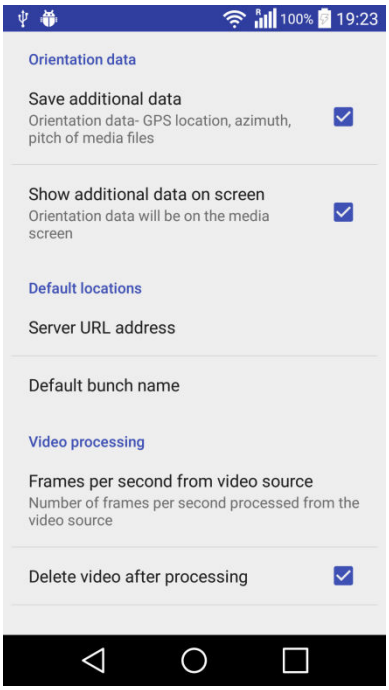
Gallery activity



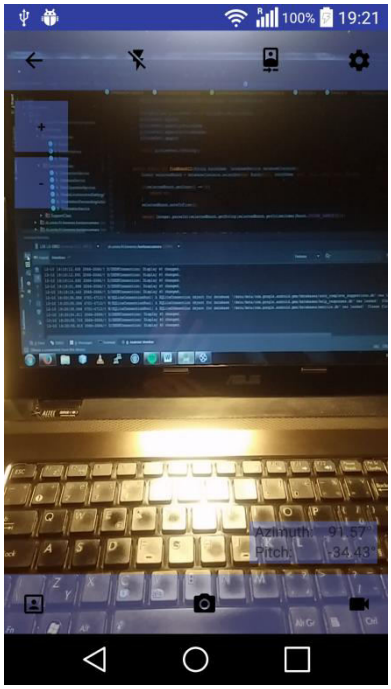
Gallery activity options



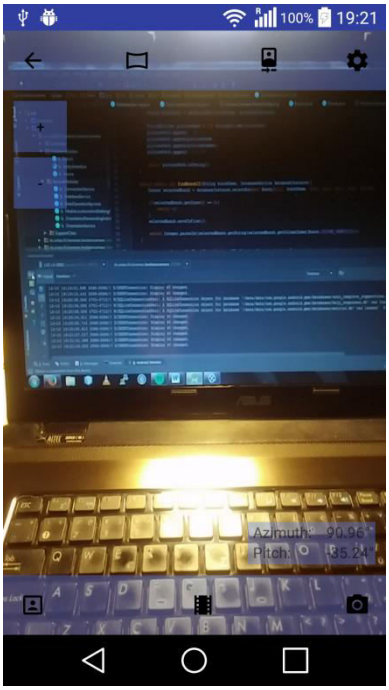
Bunch gallery



Settings activity



Picture activity



Video activity

OOP

OOP in the final version of project is almost same. There are just small differences in screen fragments, Media screen activity was added and gallery activity is not implemented with fragments. These changes were necessary for better structure of project.

Main activity: Main activity contains background picture with name of the application and five simple buttons in the middle of the screen. This buttons can navigate to next activities.

Media screen activity: Fragment activity which contains FrameLayout. To this layout is added screen fragments for view that. In this activity is also management of adding screen fragments to this layout.

Picture and Video screen fragment: Fragments are viewed within MediaScreenActivity, FragmentActivity class, and these fragments extend CameraDisplayFragment class, which provides screen services- create screen, connect with camera and common functions of screen fragments. These fragments hold FrameLayout. It is used to create real time camera screen in the application. These activities have also more buttons- take picture/record video, go to Gallery activity, go to Settings activity, go back to Main activity, turn on front/back camera, turn on flash, turn on picture/video mode. These settings are placed on the toolbars on the top and down of the screen. In video mode is also possible to turn on panorama mode.

New bunch activity: Activity has form with basic properties of bunch to choose- name, save the additional information, media format (picture or video).

Gallery activity: Gallery activity consists of RecyclerView with complex records. Records have information about created bunches- datum, GPS, number of frames. It is able to open this bunches and show the frames inside. Activity has menu on the top of the screen. It is able to choose actions delete or send. Then you can choose bunches and click on the button on the screen (delete or send) to execute the action.

Bunch details activity: An Activity Bunch detail consists of RecyclerView with complex records. This activity will be shown after the click to some bunch in the Gallery activity. All frames/pictures will be open and it is able to delete or send some of them. Frame/picture will be open in phone gallery after the click.

Settings activity: Activity is implemented with PreferenceActivity and consists of more settings groups. Settings groups have the main title and declarative subtitles. It is able to tick checkboxes or type text in some settings. Back button on the top of the screen will return to previous activity.

Class UML diagram

The main structure of UML class diagram from planning documentation is the same in the final project. But there was added a lot of new UML classes to the structure, which was necessary for logic construction of project and for possibility to extend project easily.

There is attached image with UML diagram of structure. Name of this image is: "Simplified_MobileProject_ClassDiagram_FinalDocumentation.jsg". This UML diagram is simplified. There are viewed just activity classes and service modules important for project. Full UML diagram of project would be very complicated and disarranged.

Structure of the project is still, like in the planning documentation, module orientated. There are some modules, which provide functionalities in the project. Not all of these modules are static, like it was planned. It was necessary to not be static for some of these modules. Reason is android implementation- some functions needs activity context.

Media locations and Setting time service: Static class, which provides basic settings of project and store project configuration. This class can save and load settings of application. Also can find appropriate name for save media files and contain method for get actual timestamp in long value.

Connection service: This class can serialise data to JSON file- serialise all information of frame and also image to string format. After serialisation process can this class open connection and send this data to server, defined by URL address.

Database service: Class with basic functionalities to communicate with database- insert, select, delete data. It works just with EntityInterface objects. Instance of this class is created as singleton.

Data operations services: Static class which provides common data operations for activities and another classes- get image from storage, find bunch name/ID,… This class collaborates with Database service module.

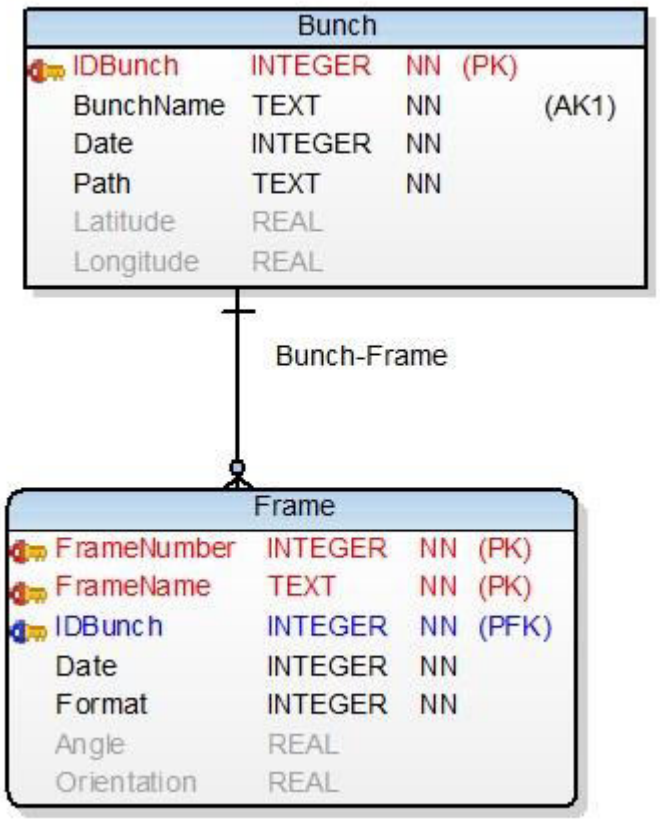
Orientation service: Class provides all orientation data- GPS, azimuth and pitch. In this class are two parts of sensors- GPS and Orientation sensors. It is possible to call data separately from these sensors. Data can be sent to activities in every update from sensor side, when these activities implement appropriate interfaces.

Database planning

Database is almost same like in the planning documentation. There was just added date column to the frame table.

Bunch table: Whole bunch of photos has the same date, GPS position, storage path and unique name.

Frame table: In the bunch can be more frames (pictures and video frames). FrameName in the Frame table is name of the subfolder- name of video, or global picture name. Frame number is identification of frame in this subfolder. There can be just one frame with certain number in subfolder (primary key). Every frame has date, orientation and angle in the 3D area and format- picture or video. Additional information- GPS position, orientation and angle are not mandatory.



Database model

Conclusion

Project was implemented with some small differences from planning documentation. There were some parts, which were not implemented- notifications and automatic data sending. Reason is lack of time in the end of project implementing.

I made working hours calendar while programming of this project. To this calendar is included implementation, documentations and final presentation creating. Calendar started on 2th of November with planning documentation.

Working calendar:

Week 44: Planning documentation- **7 hours.**

Week 45: Finishing of camera research- **5 hours.** Video fragment implementing- **12 hours.** Fragment graphic creating- **4 hours.** Fixing fragment bugs- **4 hours.** Database module implementing + testing- **5 hours.** Orientation module implementing + GPS module part testing- **5 hours.**

Week 46: Second part of orientation module implementing- **15 hours.**

Week 47: Gallery activities implementing- **20 hours.** Connection service implementing- **9 hours.**

Week 48: Saving and loading + settings activity functions implementing- **7 hours.** Orientation service accuracy improving and testing- **12 hours.** Panorama photo and video cutting implementing- **11 hours.** Refactoring and changing of structure of application- **4 hours.**

Week 49: Creating of presentation, demo and learning of presentation- **4 hours.** Finish documentation- **6 hours.**

Total work time- **130 hours.**