Evendo

Software Architecture Document

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 02.12.2019 | 1.0 | See new information on github | Niclas Schmuck |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 5

1.1 Purpose 5

1.2 Scope 5

1.3 Definitions, Acronyms, and Abbreviations 5

1.4 References 5

1.5 Overview 5

2. Architectural Representation 5

3. Architectural Goals and Constraints 5

4. Use-Case View 6

4.1 Use-Case Realizations 9

5. Logical View 9

5.1 Overview 10

5.2 Architecturally Significant Design Packages 11

5.2.1 Implementation View 11

5.2.2 Overview 12

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung 12

6. Process View 14

7. Deployment View 14

8. Size and Performance 14

9. Quality 15

Software Architecture Document

# Introduction

Evendo is a tool to stay organized. We will provide an android app, which is connected to a backend service to sync your calendar entries and todo’s.

## Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

## Scope

This software architecture document gives you an overview about our used technologies to keep evendo running.

## Definitions, Acronyms, and Abbreviations

Backend : NodeJS driven HTTP Rest Service

Frontend: Kotlin driven Android App

## References

You can find all relevant images which are not added in our SAD in our github: <https://github.com/gnaatz/evendo>

## Overview

In the following steps you will get known with our software architecture and we will show you how our app is communicating with our system.

# Architectural Representation

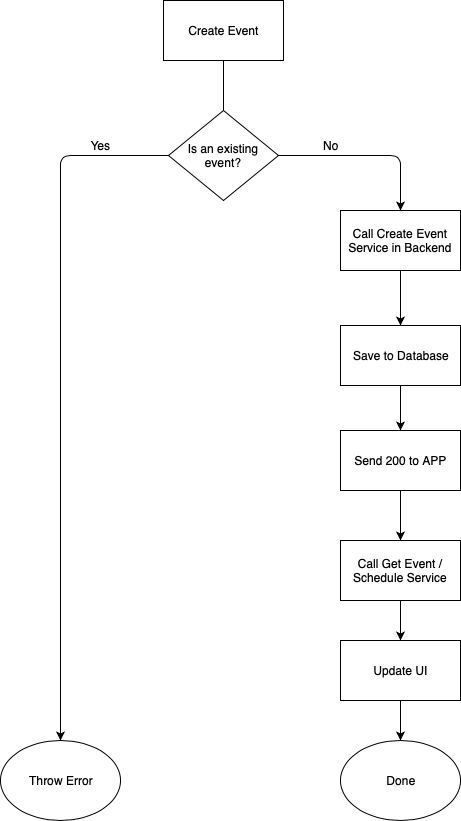
You’ll find our architectural representation on our github.

# Architectural Goals and Constraints

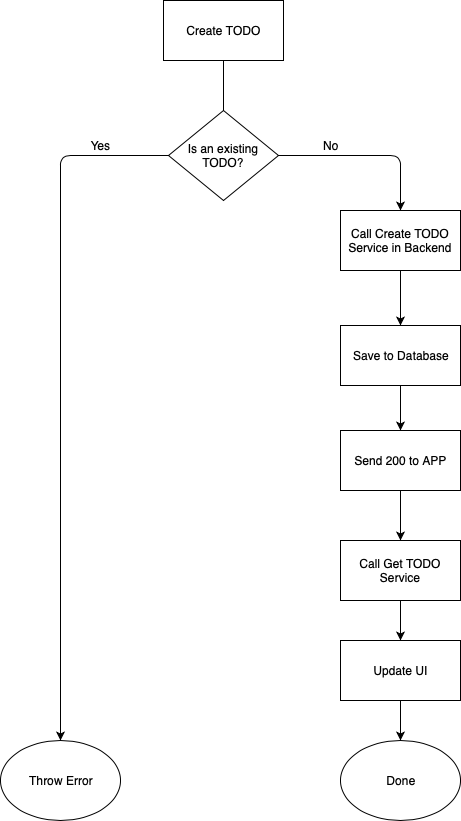
* Our main goal is to keep our architecture fast working and safe. We will implement different changes every month to keep our app secured.
* We want to have a service level agreement to our customers with around 99,7% Uptime.
* If we can’t keep this goal, we have to change hard or software.

# Use-Case View

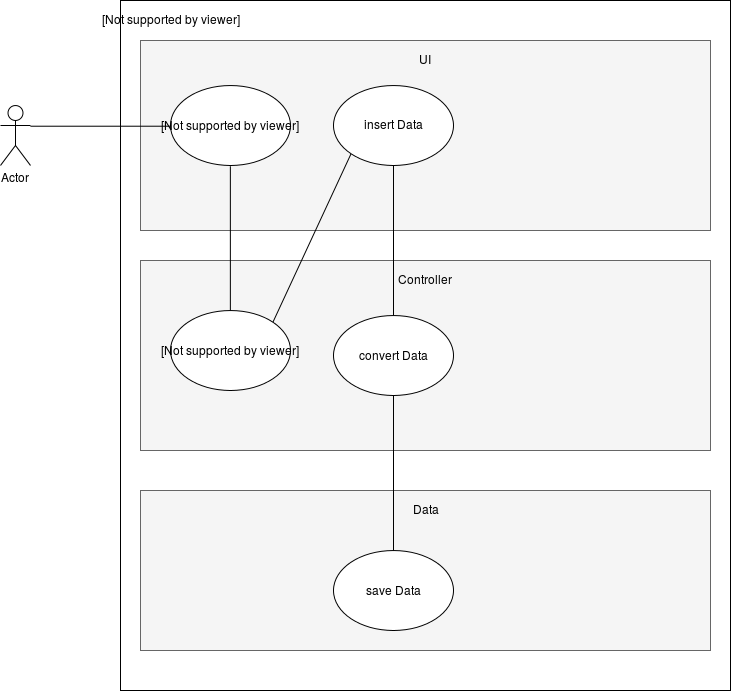
#### This is our use case to create an event



#### This one is our usecase to create an todo



This one is about to create an todo for an event:



This one is to an event:

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

The last one is about to change the date:

Ein Bild, das Text enthält.

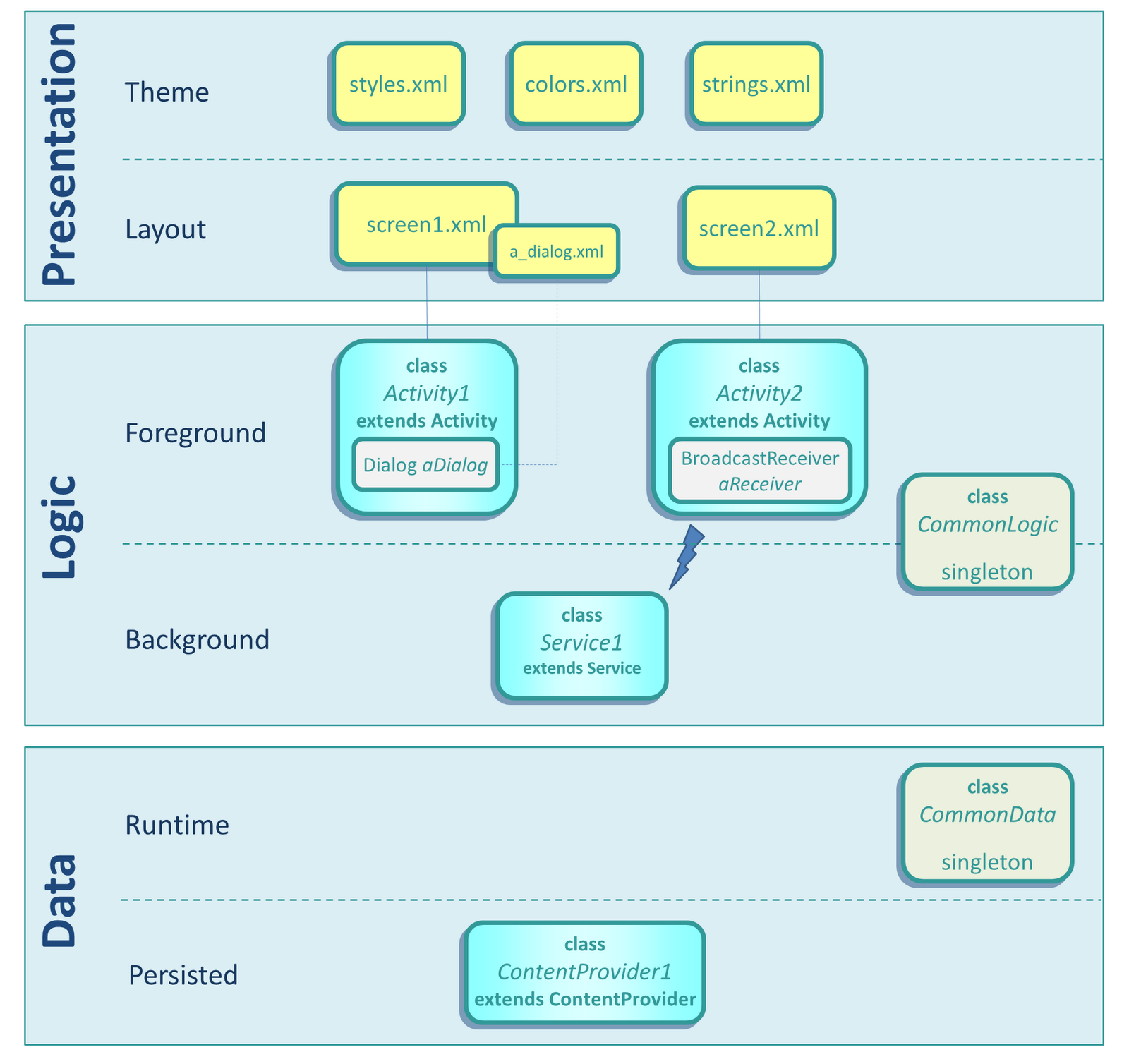
Automatisch generierte Beschreibung

## Use-Case Realizations

All usecases are currently getting implemented into our app. We’ll keep you updated if finished and will add Screenshots.

# Logical View

## Overview



## Architecturally Significant Design Packages

### Implementation View

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

### Overview

## Ein Bild, das Screenshot enthält. Automatisch generierte Beschreibung

Ein Bild, das sitzend, Metall, draußen enthält.

Automatisch generierte Beschreibung

# Process View

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

# Deployment View

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

# Size and Performance

The software should be scaleable up to 1000 parallel users watching, 100 users doing actions and up to half a million registered users.

Performance: Response Time should be lower than a second during 100% cpu usage.

# Quality

The Software should look great, the design should be intuitive, and the user interaction should be great. From our point, the code design is important for further development. From those parameters we will extract our resulting quality.