DHBW Software Engineering - Team Deminder

Deminder
Software Architecture Document

Version <1.0>

Revision History

Date	Version	Description	Author
29/11/2018	1.0	Filled out SAD	Team Deminder
12/06/2019	1.2	Pattern	Team Deminder

Table of Contents

1. Introduction 4
    1.1 Purpose 4
    1.2 Scope 4
    1.3 Definitions, Acronyms, and Abbreviations 4
    1.4 References 4
    1.5 Overview 4

2. Architectural Representation 4

3. Architectural Goals and Constraints 5

4. Use-Case View 6
    4.1 Use-Case Realizations 6

5. Logical View 7
    5.1 Overview 7
    5.2 Architecturally Significant Design Packages 7

6. Process View 7

7. Deployment View 8

8. Implementation View 8
    8.1 Overview 8
    8.2 Layers 8

9. Data View (optional) 8

10. Size and Performance 8

11. Quality 8

Software Architecture Document

1. Introduction

1.1 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.

1.2 Scope

This document describes the architecture of the Deminder project.

1.3 Definitions, Acronyms, and Abbreviations

Abbrevation	Description
N.A.	Not Applicable
MVC	Model View Controller
UC	Use case

1.4 References

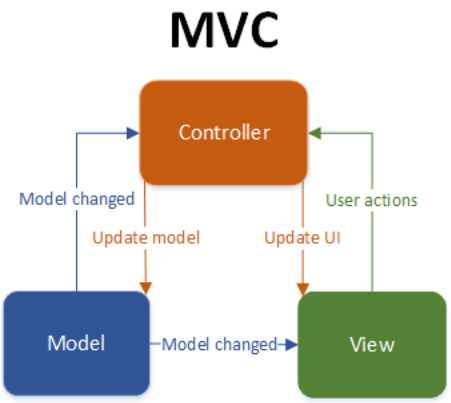
Title	Date	Publishing organization
UC Add deadline	22/10/2018	Deminder Team
UC Manage deadline	01/11/2018	Deminder Team
UC Add subtask	01/11/2018	Deminder Team
UC Manage subtask	01/11/2018	Deminder Team
UC Show deadline list	01/11/2018	Deminder Team
icalendar	29/11/2018	Wikipedia
UC_Export_Data	12/06/2019	Deminder Team
UC_Import_Data	12/06/2019	Deminder Team
UC_Widget	12/06/2019	Deminder Team
UC_Show_Subtasks	12/06/2019	Deminder Team
UC_Sort_Deadline_List	12/06/2019	Deminder Team

1.5 Overview

This document contains an overview of the architecture of the software Deminder.

2. Architectural Representation

We develop an Android App which uses MVC pattern.



3. Architectural Goals and Constraints

There is no server used in our app (yet), so there's only a client side which uses **MVC**. Our View displays deadlines which can be managed by the User. It is written in **Java** and uses the **icalendar format** for storing the deadlines.

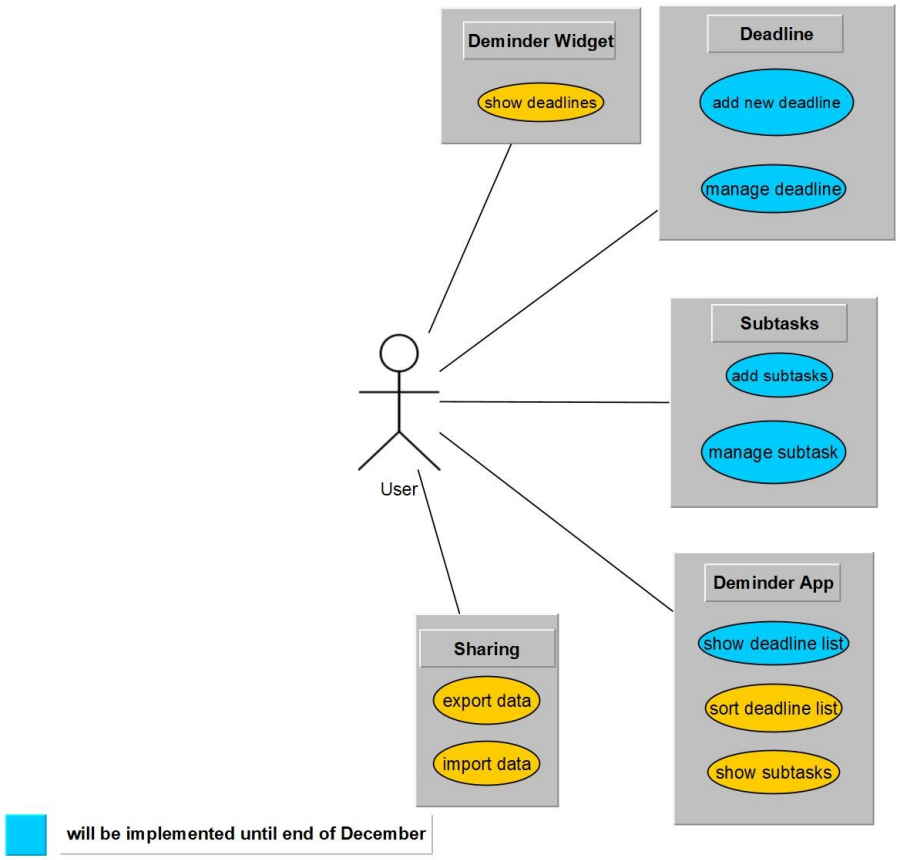
MVC

The main goal of the MVC architecture is to separate the view from the logic. The controller takes care of handling actions by the user and telling View or Model, that there were actions performed that concern them. Models contain the data that is displayed in the views.

Data Storage

There is no database use. Instead of that, the data is stored in the icalendar format on the phone drive. The icalendar format (.ics) is the current standard for saving appointments and tasks. The files may be reused in different calendar or todo apps.

4. Use-Case View

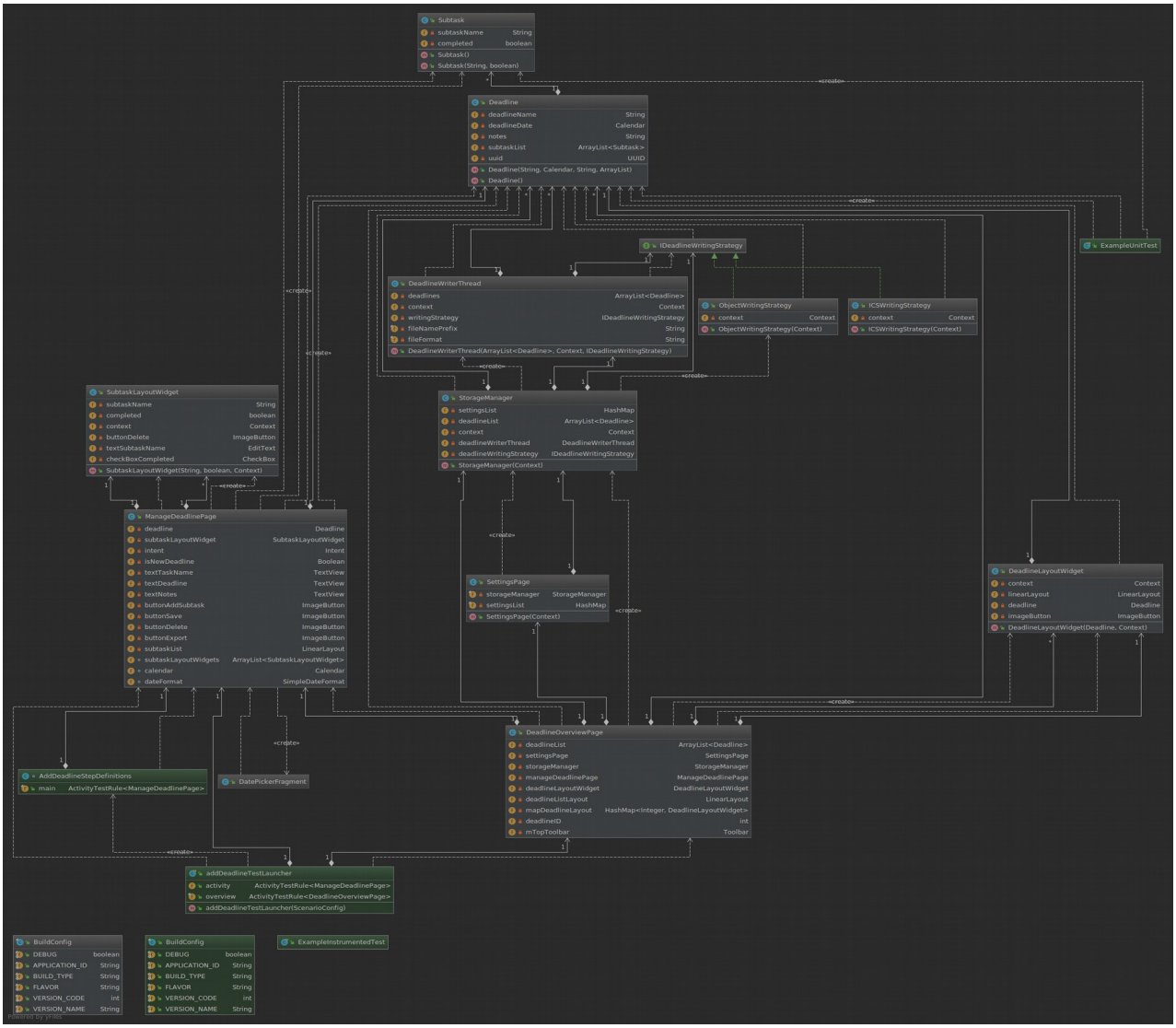


4.1 Use-Case Realizations

N.A.

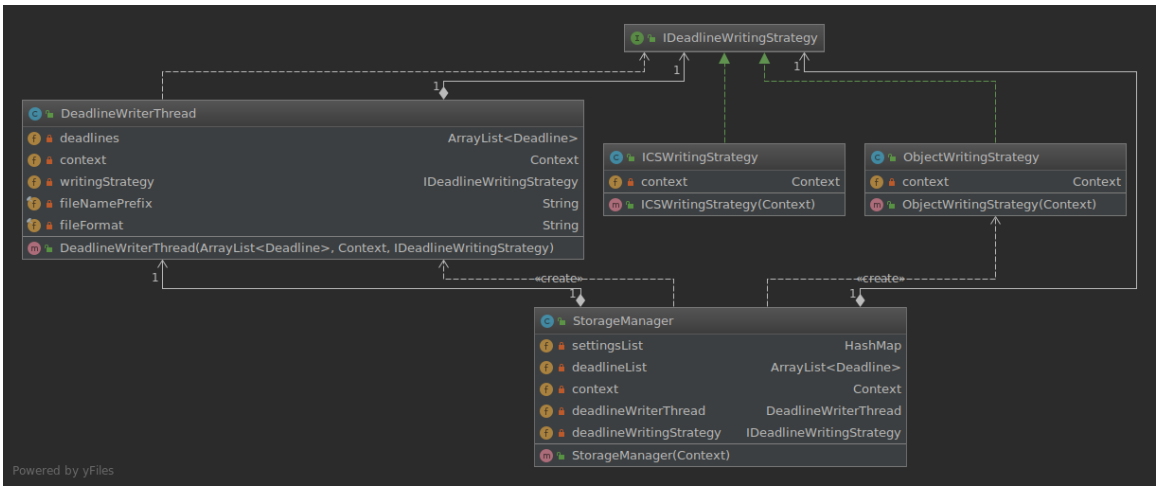
5. Logical View

5.1 Overview



5.2 Architecturally Significant Design Packages

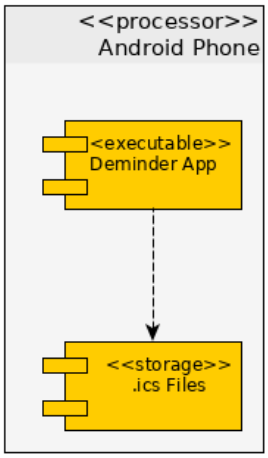
The art of saving the deadlines to the memory has to be changeable. Due to this, the thread writing deadlines to files uses a writing strategy.



6. Process View

N.A.

7. Deployment View



8. Implementation View

N.A.

8.1 Overview

N.A.

8.2 Layers

N.A.

9. Data View (optional)

Data is stored on the android phone in .ics format. These files are accessed by the app directly. There is no database used.

10. Size and Performance

The goal is to create a fast running app which is only using the libraries it needs. Further more, loading and saving files to the phone memory is handled within threads.

11. Quality

The app is intuitive regarding its usability.