Gaming-Bets

Version 1.1

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 15.11.2015 | 1.0 | Document created | Nicals Petersohn |
| 18.11.2015 | 1.1 | Rearanged Stuff | Felix Morsbach |
|  |  |  |  |
|  |  |  |  |

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 4

4. Use-Case View 4

4.1 Use-Case Realizations 5

5. Logical View 5

5.1 Overview 5

5.2 Architecturally Significant Design Packages 5

6. Process View 5

7. Deployment View 5

8. Implementation View 5

8.1 Overview 5

8.2 Layers 5

9. Data View (optional) 6

10. Size and Performance 6

11. Quality 6

# Introduction

## 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 document shows the architecture of our GamingBets Application.

## Definitions, Acronyms, and Abbreviation

MVC – Model View Controller

tbd – to be determined

## References

blogsiteloremipsum.de

## Overview

tbd

# Architectural Representation

This Project will use MVC Principles for development.

# Architectural Goals and Constraints

Since our App will mainly have to deal with data management (e.g. user data, game data) using an MVC structure in combination with a client-server technology will be the approach. The inherited MVC-Pattern by Android will be our way to go here. Dividing View and Controllers as a focus.

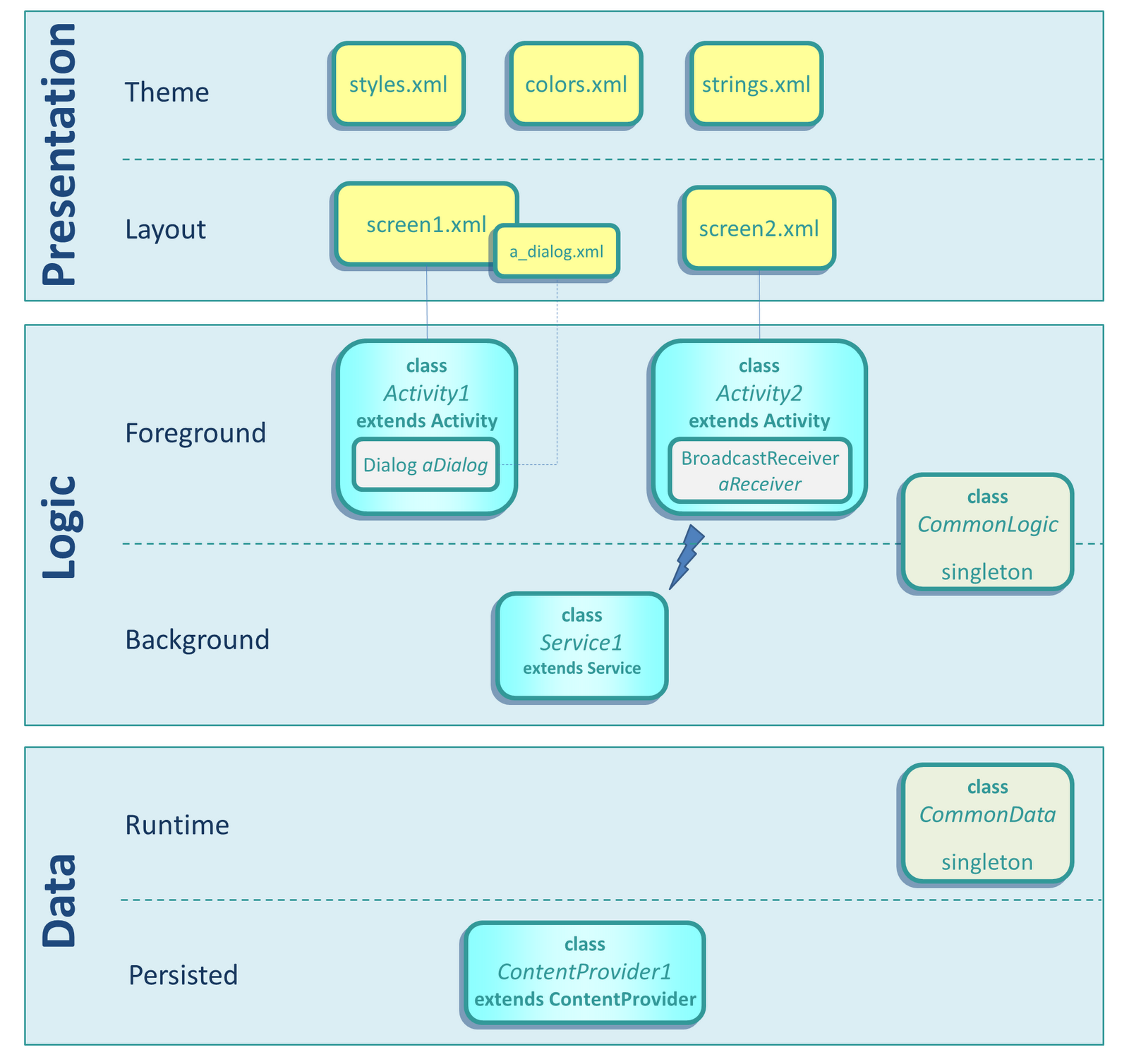
# Use-Case View

tbd

## Use-Case Realizations

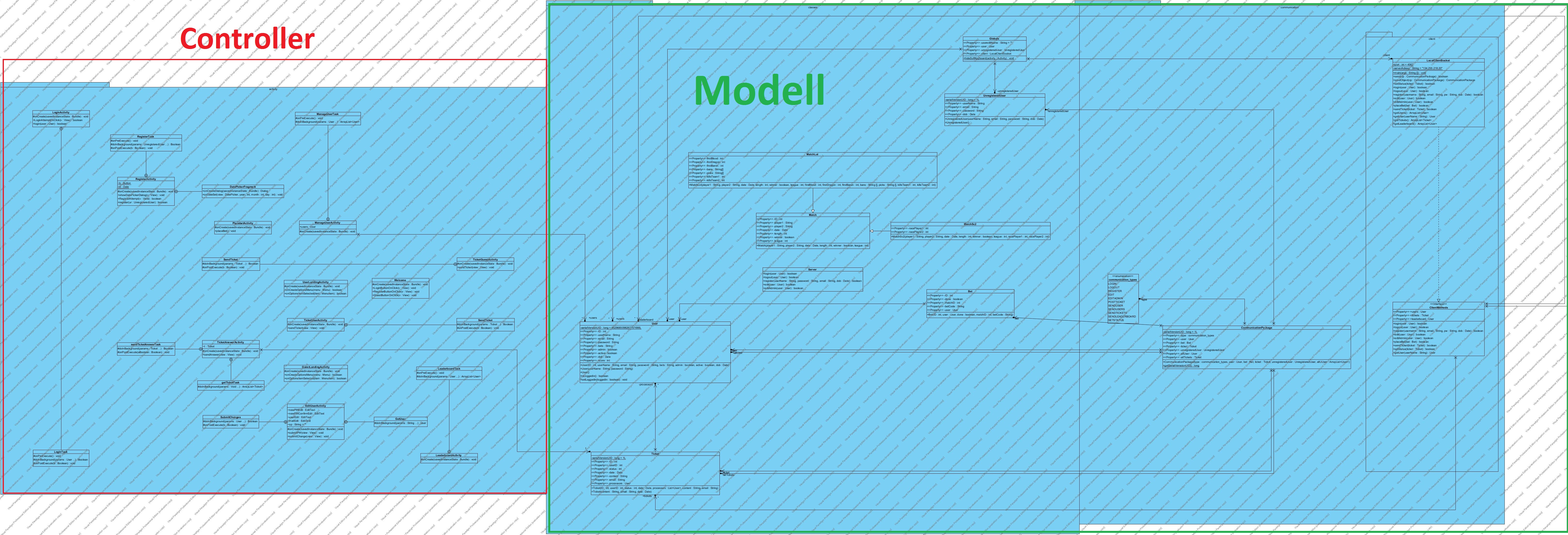
tbd

# Logical View



## Overview

## Architecturally Significant Design Packages

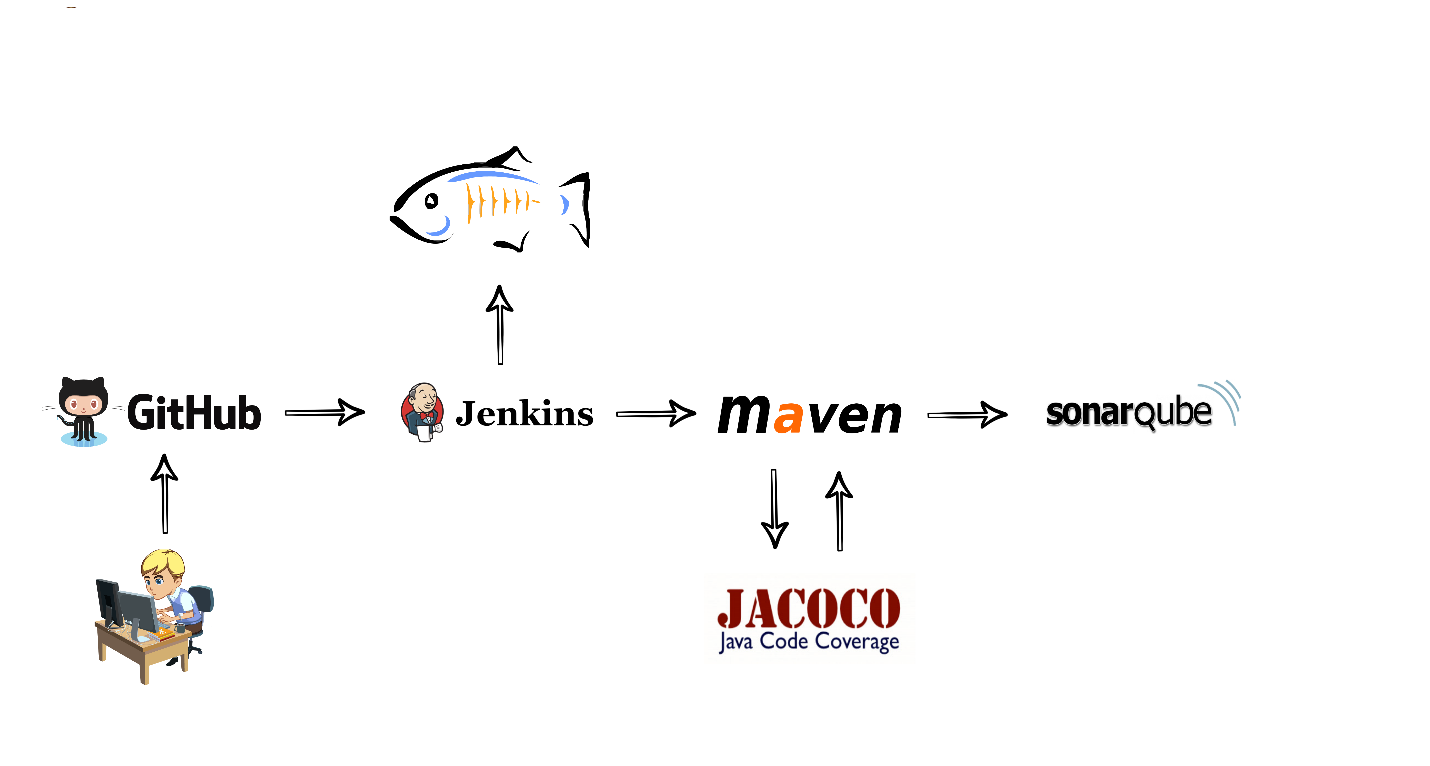


The View area is not realized as Java-Classes, in Android these are XML-Documents, which automatically creates a separation between View and Controller.

# Process View

n/a

# Deployment View



The Deployment process is based on Jenkins. Every push into our master branch on GitHub will initiate a Jenkins build. This build will be executed with maven and build a *.war* file. It will run unit test with every build and report the test coverage with JaCoCo. Test coverage and static code analysis is then send to SonarQube, with each build. After a successful maven build, the *.war* file is then remotely deployed to our GlassFish 3 application server.

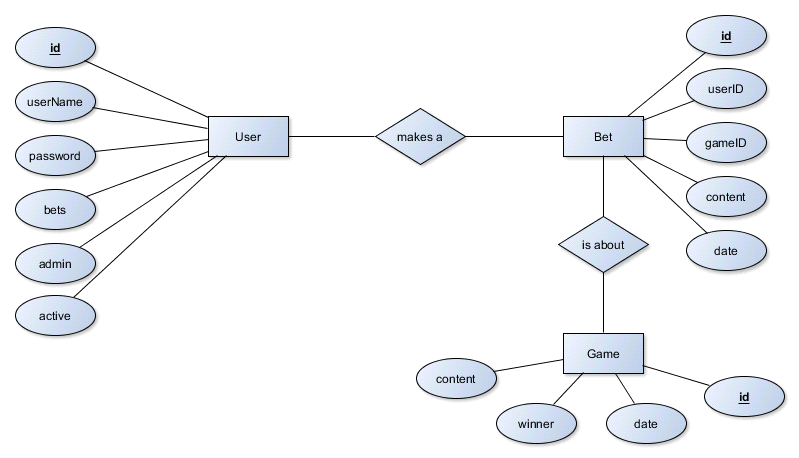
# Implementation View

n/a

## Overview

## Layers

# Data View (optional)



# Size and Performance

# Quality