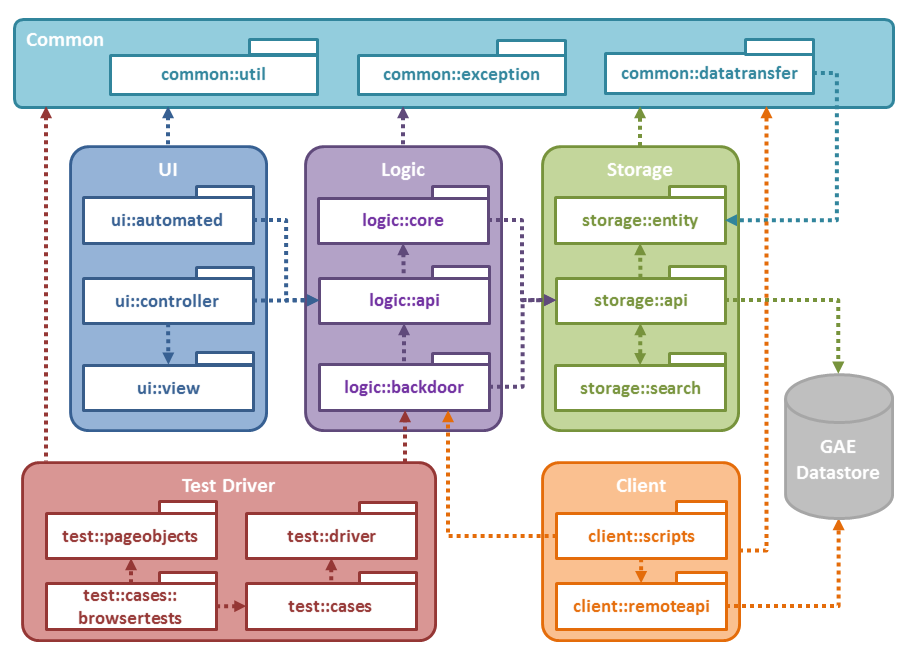
TEAMMATES Modular Software Architecture

The aim of this document is to describe a complete and correct lists of the modules, their assigned (implementing) software units, and the architectural rules. The participants can use this document to register the software architecture (SA) in different SA-tools and we can compare the different results and approaches of the tools.

The underlying specifications are based on the Design doc and source code of TEAMMATES release 5.110 downloaded at 2017-08-25.

# Modules

## Overview



## Modules and their (java) Implementation Units

Source included: ..\teammates-5.110\src

|  |  |  |
| --- | --- | --- |
| **Id** | **Module** | **Assigned Software Units** |
| 1 | UI (layer) | teammates.ui (package) |
| 2 | automated (subsystem) | teammates.ui.automated (package) |
| 3 | controller (subsystem) | teammates.ui.controller (package) |
| 4 | view (subsystem) | teammates.ui.datatransfer (package) |
|  |  | teammates.ui.pagedata (package) |
|  |  | teammates.ui.template (package) |
| 5 | Logic (layer) | teammates.logic (package) |
| 6 | core (component) | teammates.logic.core (package) |
| 7 | coreFacade (interface) | teammates.logic.api (package) |
| 8 | backdoor (subsystem) | teammates.logic.backdoor (package) |
| 9 | Storage (layer) | teammates.storage (package) |
| 10 | entity (subsystem) | teammates.storage.entity (package) |
| 11 | api (subsystem) | teammates.storage.api (package) |
| 12 | search (subsystem) | teammates.storage.search (package) |
| 13 | Common (subsystem) | teammates.common (package) |
| 14 | util (subsystem) | teammates.common.util (package) |
| 15 | exception (subsystem) | teammates.common.exception (package) |
| 16 | datatransfer (subsystem) | teammates.common.datatransfer (package) |
| 17 | Client (subsystem) | teammates.client (package) |
| 18 | remoteapi (subsystem) | teammates.client.remoteapi (package) |
| 19 | scripts (subsystem) | teammates.client.scripts (package) |
| 20 | Test Driver (subsystem) | teammates.test (package) |
| 21 | Datastore (external library) | xLibraries.com.googlecode.objectify (library) |

## Notes (including answers to questions from Wilson Kurniawan)

1. The completeness of the list of modules
   1. No modules are missing.
   2. Subsystem Client is not used in production, so similar to the test driver component, it may be preferable to skip this. The usage of this component is for system administrators to connect to the back-end logic to perform administrative tasks such as data schema migration, calculating statistics, doing data analytics, etc.
   3. Datastore is the only external library that has significant function to the application because it essentially is the application’s database layer.
      1. There exists other external libraries, but they can be omitted for this purpose. Examples include Task Queue APIs or third-party email APIs (refer to the component diagram for Logic package).
2. The types of modules  
   In this terminology:
   1. A subsystem is a module without additional semantics.
   2. A layers is a module that is part of a layered architecture (Layers pattern).
      1. UI, Logic, and are Storage layers.
      2. Layers are not applied elsewhere.
   3. A component is a module shielded by an interface (Façade pattern). Usage is allowed only via its Interface (a module that represents software units that may be used from outside the component).
      1. Logic.core is a component shielded by logic.api package.
      2. The facade pattern is not applied elsewhere.
3. Correctness of the assignment of software units to modules
   1. In the table above, module UI.view has also as assigned package: ui::datatransfer (in contrast to the Design documentation), with the intention: All source code should be assigned to a module.
      1. Feedback Wilson: It is more accurate to say that UI.view is assigned to ui::pagedata and ui::template packages. ui::datatransfer serves as a small bridge (i.e. short-lived data transfer objects) between the controller subsystem and the view subsystem.
   2. The frontend UI.view is implemented by the following sources:
      1. ..\teammates-5.110\src\main\resources
      2. ..\teammates-5.110\src\main\webapp
   3. Usage of com.googlecode.objectify represents usage of Datastore.
      1. No other external libraries indicate usage of a module in the architecture.

# Rules

The dependency arrows in the UML package diagram indicate that usage is allowed.  
Consequently, if there is no dependency arrow usages are not allowed.  
The table below describes these constraints. Wilson has extended an initial set of rules and added the Column Rationale.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Exception** | **From module** | **Rule type** | **To module** | **Rationale/rules applied** | |
| 1 |  | Client.remoteapi | Is not allowed to use | Client.scripts | Client is an independent subsystem. The only way to access the business logic is via *logic.backdoor*. |  |
| 2 |  | Client | Is not allowed to use | UI |  |
| 3 |  | Client | Is not allowed to use | Logic |  |
|  | Exception | Client | Is allowed to use | Logic.backdoor | Delegation pattern |
| 4 |  | Client | Is not allowed to use | Storage |  |
| 5 |  | Client | Is not allowed to use | Test Driver |  |
| 6 |  | Common | Is not allowed to use | UI | Common is a set of reusable utilities and helpers that should not handle any business logic. |  |
| 7 |  | Common | Is not allowed to use | Logic |  |
| 8 |  | Common | Is not allowed to use | Storage |  |
|  | Exception | Common.datatransfer | Is allowed to use | Storage.entity | Wrapper pattern |
| 9 |  | Common | Is not allowed to use | Client |  |
| 10 |  | Common | Is not allowed to use | Test Driver |  |
| 11 |  | Logic | Is not allowed to use (back call) | UI | The core business logic which is the only component that can connect to the Storage component. |  |
| 12 |  | Logic | Is not allowed to use | Storage.entity | Adapter pattern (via storage.api) |
| 13 |  | Logic | Is not allowed to use | Storage.search |
| 14 |  | Logic | Is not allowed to use | Client |  |
| 15 |  | Logic | Is not allowed to use | Test Driver |  |
| 16 |  | Logic.core | May only be used via its interface (Facade convention) |  | *logic.core* is the package that provides the bridge to the Storage component. | Façade pattern |
| 17 |  | *Logic.core* | *Is not allowed to use* | *Logic.core.coreFacade* |
| 18 |  | *Logic.core* | *Is not allowed to use* | *Logic.backdoor* |  |
| 19 |  | Logic.core.coreFacade | Is not allowed to use | Storage |  |
| 20 |  | *Logic.core.coreFacade* | *Is not allowed to use* | *Logic.backdoor* |  |  |
| 21 |  | Storage | Is not allowed to use (back call) | UI | Storage is the component that connects to the application database. It should not be aware of others. |  |
| 22 |  | Storage | Is not allowed to use (back call) | Logic |  |
| 23 |  | Storage | Is not allowed to use | Client |  |
| 24 |  | Storage | Is not allowed to use | Test Driver |  |
| 25 |  | Storage.api | Is the only module allowed to use | Datastore | Gateway pattern |
|  | Exception | Client.remoteapi | Is allowed to use | Datastore |
| 26 |  | Storage.entity | Is not allowed to use | Storage.api | *storage.entity* is a collection of persistable classes relevant only to database processes. |  |
| 27 |  | Storage.entity | Is not allowed to use | Storage.search |  |
| 28 |  | Storage.search | Is not allowed to use | Storage.entity |  |
| 29 |  | Test Driver | Is not allowed to use | UI | Test driver is an independent subsystem. The only way to access the business logic is via *logic.backdoor*. |  |
| 30 |  | *Test Driver* | *Is not allowed to use* | *Logic* |  |
|  | *Exception* | *Test Driver* | *Is allowed to use* | *Logic.backdoor* | Delegation pattern |
| 31 |  | Test Driver | Is not allowed to use | Storage |  |
| 32 |  | Test Driver | Is not allowed to use | Client |  |
| 33 |  | UI | Is not allowed to use (skip call) | Storage | The entry point of the application, UI, can only connect to business logic via the facade *logic.api*. |  |
| 34 |  | UI | Is not allowed to use | Client |  |
| 35 |  | UI | Is not allowed to use | Test Driver |  |
| 36 |  | *UI* | *Is not allowed to use* | *Logic.backdoor* | Delegation pattern (via logic.api) |
| 37 |  | UI.view | Is not allowed to use | UI.automated | Graphical user interface; only relevant for user-invoked actions (i.e. called via the controller subsystem) |  |
| 38 |  | UI.view | Is not allowed to use | UI.controller |  |
| 39 |  | UI.view | Is not allowed to use | Logic |  |
| 40 |  | *UI.automated* | *Is not allowed to use* | *UI.view* |  |