|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | |  | |  | |  | |  | |
| TOPAZ MasterRindApp  Technische Dokumentation |
|  |

Inhaltsverzeichnis

TOPAZ MasterRindApp 1

Einleitung 1

Benutzte Entwurfsmuster 2

Datenmodel 1

Datenbankstruktur 1

Fehlerbehandlung 1

Testen 1

# TOPAZ MasterRindApp

### Einleitung

Ziel dieser Dokumentation ist, einen Einblick in Entwicklung der MasterRindApp zugeben und die Einarbeitung in das Projekt zu erleichtern.

Auf den nachfolgenden Seiten werden die Entwurfsmuster, die bei der Erstellung benutzt wurden, beschrieben. Des Weiteren werden die wichtigsten Klassen des Datenmodels besprochen und es wird die Datenbankstruktur erklärt. Es werden auch die Methoden zur Fehlerbehandlung und zum Testen vorgestellt.

Diese Dokumentation ist nicht vollständig und wird im Laufe des Projektes weiter wachsen.

### Benutzte Entwurfsmuster

Entwurfsmuster sind bewährte Lösungsschablonen, die für die wiederkehrenden Probleme in der Softwarearchitektur- und Entwicklung benutzt werden. In diesem Abschnitt werden die Entwurfsmuster, die bei der Erstellung benutzt wurden, einzeln erklärt.

#### Model View Controller (MVC)

Model View Controller ist ein Muster zur Strukturierung von Softwareentwicklung. Ziel dieses Musters ist es, einen flexiblen Programmentwurf zu ermöglichen, bei dem die Änderungen einfach durchgeführt werden können. Hierbei wird der Code in drei Einheiten aufgeteilt:

* Datenmodel (model)
* Präsentation (view)
* Programmsteuerung (controller)

Jede dieser Einheiten wird unabhängig von einander entwickelt und kann einfach ersetzt werden, ohne dabei die Projektstruktur zu zerstören.

Das Titaniumframework bietet eine vordefinierte MVC-Struktur an. Diese Struktur wird für die Erstellung der MasterRindApp benutzt. Die Abbildung 1 zeigt die Projektstruktur der MasterRindApp. Wie man aus der Abbildung gut erkennen kann, sind in dem Projekt unter anderen die Ordner controllers, models und views enthalten. In dem Ordner „views“ werden die XML-Dateien abgelegt, die die Struktur der GUI bilden und in dem Ordner „styles“ befinden sich die TSS-Dateien, die das Aussehen der GUI bestimmen. Zusammen bilden diese beiden Ordner die Präsentationseinheit der MasterRindApp. Der Ordner „models“ enthält die JavaScript-Dateien, in denen die Daten abgebildet werden und in dem Ordner „controllers“ wird die Programmsteuerung implementiert. Zur welchen Einheit der einzelne Ordner gehört, lässt sich aus dem Namen leicht ableiten.

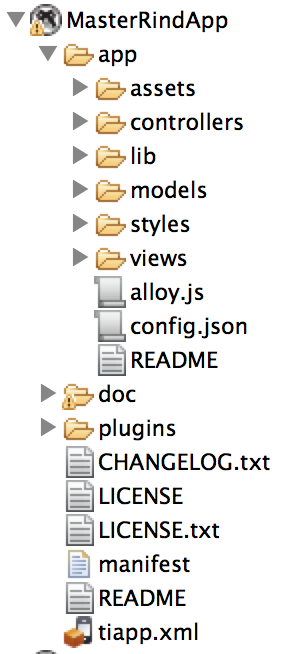


Abbildung : Projektstruktur im Titanium

#### Abstrakte Fabrikmethode

Abstrakte Fabrikmethode ist ein Objekterzeugungsmuster, dabei werden gleichartige Objekte in einer extra Klasse erzeugt. Diese Klasse bildet die Schnittstelle für die anderen Klassen des Systems. Dadurch werde die Klassen voneinander entkoppelt, was die Erweiterung der Architektur erleichtert. Abbildung 2 zeigt ein verallgemeinertes UML-Diagramm einer Abstrakten Fabrik.

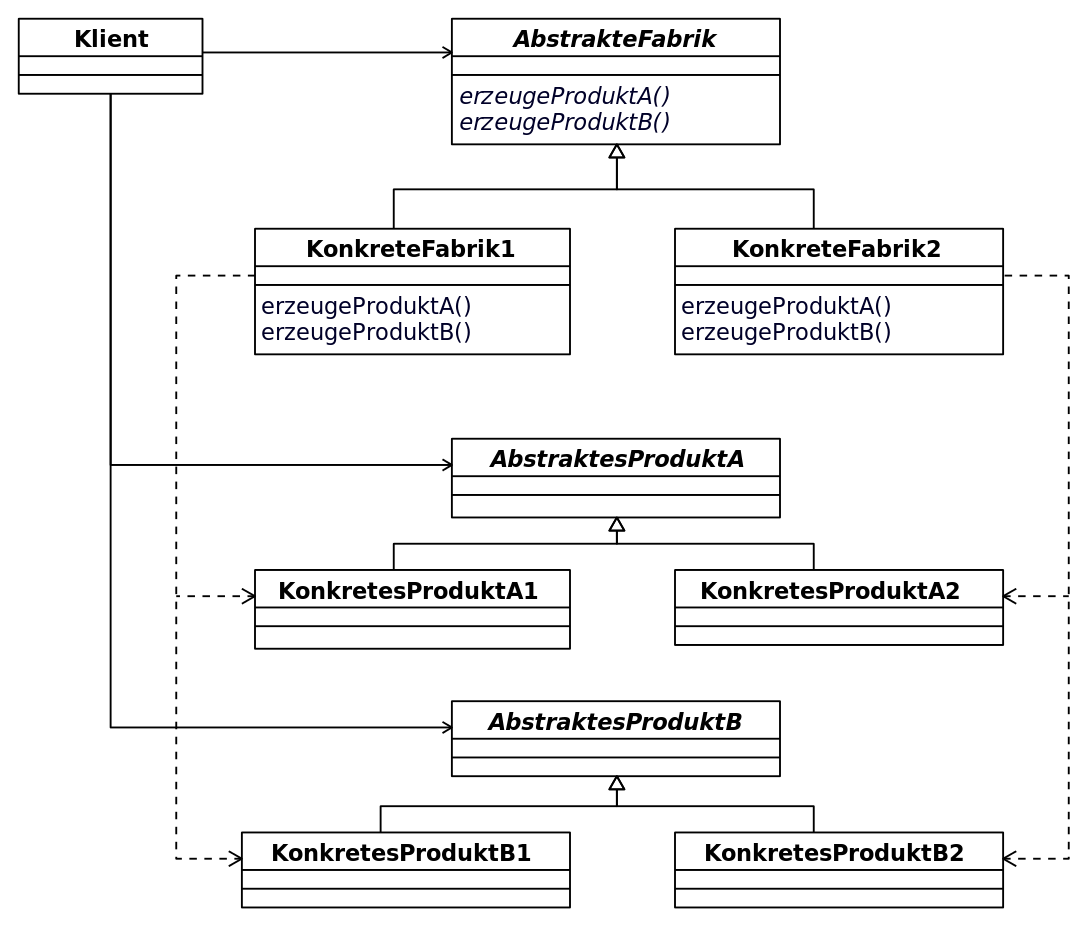


Abbildung : Abstrakte Fabrik

Abbildung 3 zeigt ein UML-Diagramm, wo die Abstrakte Fabrik auf die Klassen der MasterRindApp angewendet wurde. Da die Klassen im Abschnitt „Datenmodel“ beschrieben werden, werden hier die Klassen nicht weiter erläutert.

#### Fassade

Fassade ist ein Entwurfsmuster, der die Struktur der Klassen definiert. Er bietet eine Schnittstelle mit ausgewählten Methoden für die Klassen des Subsystems. In Abbildung 2 wurde Fassade für die Klassen der MasterRindApp angewendet.

### Datenmodel

#### Klassenbeschreibung

Cow

Favorite

Appointment

MasterRindFactory

MasterRindFacade

#### Namespacebildung

### Datenbankstruktur

### Fehlerbehandlung

### Testen

Literaturverzeichnis

Nachname, F. (Datum). Dolor Sit Amet. *Lorem Ipsum*, 1 - 10.

Nachname, F. (Datum). *Lorem Ipsum Dolor Sit Amet.* Ort: Verlag.

Nachname, F. (Datum). Lorem Ipsum Dolor Sit Amet. *Duis sed elit ante*, S. 10-20.