# Architektur

Im Zuge der Designphase haben wir einen ersten Entwurf der Systemarchitektur erstellt. Nun, in der Implementationsphase, hat sich dieser Entwurf konkretisiert und wurde entsprechend neuer Erkenntnisse angepasst. Im Folgenden sollen nun die wesentlichen Designentscheide und Änderungen zur letzten Version vorgestellt und begründet werden.

## Wesentliche Änderungen und Entscheide

### Entfernung des Renderer-Pakets

In der ersten Version der Architektur war geplant, die visuelle Darstellung der Spielobjekte in eigene Renderklassen in einem Renderer-Paket auszulagern. Diese Idee wurde aber bereits in der ersten Entwurfsphase wieder verworfen. Grund dafür war in erster Linie, dass sich dieses Konzept nicht gut mit dem Scene2D-System von libGDX vertrug, welches wir nutzen wollten. Dort wird davon ausgegangen, dass alle Actors (also alle Spielobjekte) selber für ihre Darstellung verantwortlich sind. Natürlich könnten unsere GameObjects in ihren Rendermethoden die Renderobjekte aufrufen, aber das würde zu einer Verletzung des Schichtenprinzips führen, weil Abhängigkeiten nur zu unteren Schichten bestehen dürfen.

Weiterhin war unsere Begründung, das Rendering auszulagern, in erster Linie diese, dass wir vorläufige Renderfunktionen später einfach mit den finalen austauschen können. In der Realität kamen wir aber mit der Grafik sehr zügig vorwärts, so dass sich uns in diesem Bezug gar kein Vorteil mehr bot.

Aus diesen Gründen entschieden wir uns, auf das Renderer-Paket komplett zu verzichten und die Renderfunktionen direkt in den GameObjects zu implementieren.

### Nutzung von Scene2D

Dieser Punkt ist nicht im Paketdiagramm ersichtlich, aber dennoch ein essentieller Entscheid. Das Java-Spieleentwicklungsframework libGDX bietet den Scene 2D Scene Graph um UI- und Spielelemente einfach verwalten zu können. Das Konzept beruht auf der Analogie mit einem Bühnenspiel: Ein Stage-Objekt dient als Container für Actor-Objekte und handelt sowohl Input-Events als auch das Rendering. Actor-Objekte können auch gruppiert und so hierarchisch strukturiert werden. Die weiteren Features von Scene2D, insbesondere auch in Bezug auf das Rendering, können unter <https://github.com/libgdx/libgdx/wiki/Scene2d> eingesehen werden.

Das System eignet sich sehr gut für unseren relativ simplen Spielaufbau und nimmt uns viel Arbeit ab. So erben alle Klassen in GameObjects (Ship, Train, Container, Crane) von der Actor-Klasse und werden vom Game in eine Stage eingefügt.

## Game-Paket

Die Levelklasse wurde zumindest vorläufig in das Game-Paket verschoben. Fachlich macht das Sinn, weil ausschliesslich das CareerGame die Levelklasse nutzt und der Level schliesslich ein Spiel definiert. Sollte es sich ergeben, dass wir noch weitere Levelklassen benötigen, wäre es dann sinnvoll, ein :Domain::Level oder :Domain::Game::Level Paket zu erstellen.

Ausserdem wurde noch die neue Klasse LoadRating ins Game-Paket aufgenommen. Sie enthält die Bewertungsalgorithmen, die von den Games genutzt werden.

## User-Paket

Im User-Paket wurde um die Handicap-Klasse erweitert. Diese enthält und verwaltet die Handicap-Einstellungen des Benutzers (vgl. Domänenmodell).

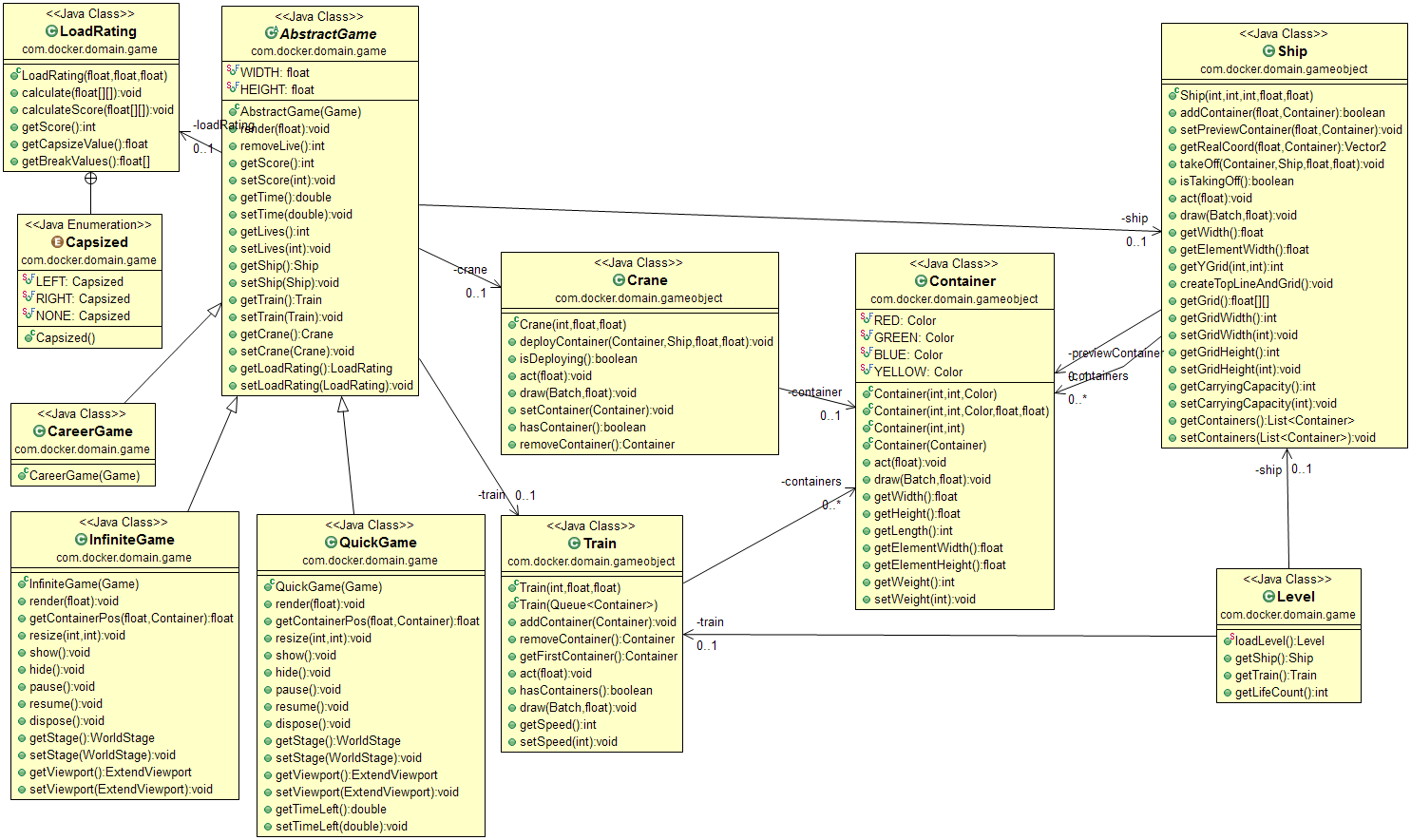
## Menus-Paket

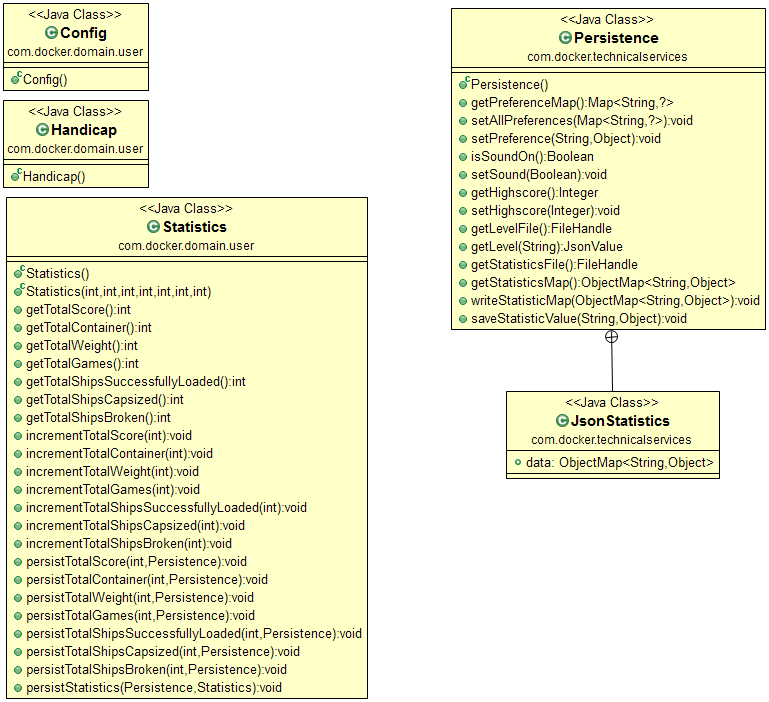
Hier kam neu die Klasse StatisticsMenu hinzu. Dem Namen entsprechend handelt es sich um den Statstikbildschirm im Menü.

## Paketdiagramm



# Klassendiagramm



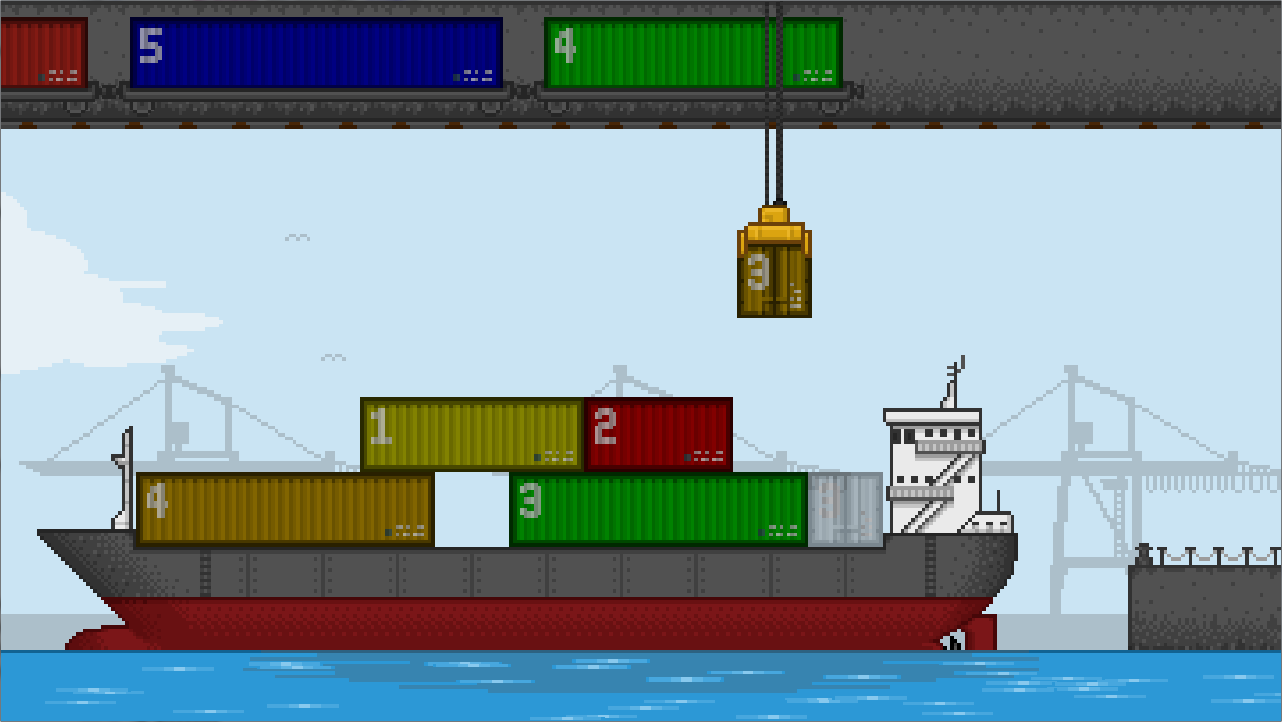


# GUI-Design

## Menüs

Unsere Menüstruktur ist sehr simpel und flach. Ausserdem wollen wir uns in dieser Phase noch auf den Spielablauf konzentrieren, weshalb das Menü optisch noch nicht gestaltet wurde. Generell werden wir im Umfang des SEPS-Projekts das Menü nicht priorisiert behandeln und werden es erst verschönern, wenn sich die zeitliche Gelegenheit dazu bietet.

## Spielbildschirm



Der Spielbildschirm hat sich sehr stark am ursprünglichen Mock-Up orientiert und ist bereits fast in einem auslieferungsfähigen Zustand. Einige Ecken und Enden müssen noch ausgebessert werden, etwa die Platzierung der Gewichtszahl muss nach rechts verschoben werden, da man sie so auf dem Zug zu spät sieht.