Worms-like Scampi di Mare

Projektdokumentation

Autoren:

Timo Nadolny

Lennart Sparbier

Alexander Voigt

Inhalt

[Programmablaufplan 3](#_Toc148214204)

[Szenenbeschreibungen 4](#_Toc148214205)

[Titel 4](#_Toc148214206)

[Login 4](#_Toc148214207)

[Lobby 6](#_Toc148214208)

[Abbildung 1: Vereinfachter PAP der Szenenlogik 3](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996403)

[Abbildung 2: Spieler-Objekt v1 5](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996404)

[Abbildung 3: ServerAntwort-Objekt 5](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996405)

[Abbildung 4: Beispielhafte Lobbyszene 6](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996406)

[Abbildung 5: Konfigurations-Objekt 7](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996407)

[Abbildung 6: Waffen-Objekt 7](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996408)

[Abbildung 7: TrefferVerhalten-Objekt 7](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996409)

[Abbildung 8: FlugVerhalten-Objekt 7](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996410)

[Abbildung 9: SplitterVerhalten-Objekt 7](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996411)

[Abbildung 10: Nachricht-Objekt 7](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996412)

[Abbildung 11: Partie-Objekt 7](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996413)

[Abbildung 12: Level-Objekt 7](file:///C:\Users\alexa\IdeaProjects\wise23-24_grpl_shrimps\dokumentation\ScampiDiMare_Projektdokumentation.docx#_Toc148996414)

[Tabelle 1: Login Kommunikation 4](#_Toc148996395)

[Tabelle 2: ServerAntwort Attribute 5](#_Toc148996396)

[Tabelle 3: Lobby Kommunikation 6](#_Toc148996397)

[Tabelle 4: Partie Attribute 8](#_Toc148996398)

[Tabelle 5: Konfiguration Attribute 8](#_Toc148996399)

[Tabelle 6: Level Attribute 9](#_Toc148996400)

[Tabelle 7: Nachricht Attribute 9](#_Toc148996401)

# Programmablaufplan

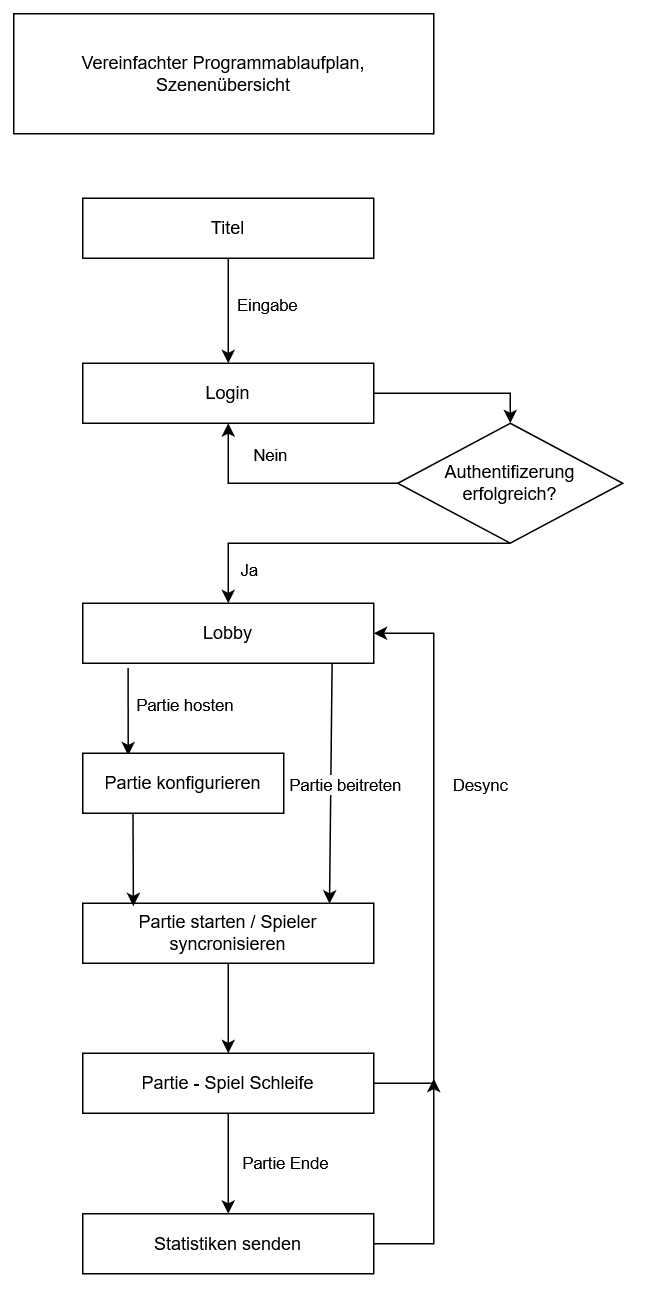


Abbildung 1: Vereinfachter PAP der Szenenlogik

# Szenenbeschreibungen

## Titel

Einfache Darstellung des Titels und gegebenenfalls Nennung der für den Client zuständige Person. Durch beliebige Eingabe geht es weiter zum Login.

*Mögliche Datenkommunikation:*

* Keine

## Login

Hier soll es möglich sein sich zum Beispiel über einen Nutzernamen bzw. E-Mail plus Passwort gegenüber dem Server zu authentifizieren. Es sollte auch ein extra Formular gegen um sich zu registrieren.

*Mögliche Datenkommunikation:*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Methode | Parameter |  | Antwort | Pull | Push | Sync | Async |
| RegistriereNutzer | Mail, Name, MD5(Passwort) | -> | Ok/Error | X |  | X |  |
| AuthentifiziereNutzer | Mail, Name, MD5(Passwort) | -> | Ok/Error | X |  | X |  |

Tabelle 1: Login Kommunikation

*Mögliche Strukturen:*

ServerAntwort

Spieler

antwort : Boolean

antwort\_bezug : String

nachricht : String

referenz\_objekt : Object

id : GUID

email : String

name : String

passwort : String

Abbildung 2: Spieler-Objekt v1

Abbildung 3: ServerAntwort-Objekt

Das „Spieler“-Objekt wird weiter unten noch erweitert, darum erfolgt hier keine genaue Beschreibung.

Das „ServerAntwort“-Objekt soll vom Server als Antwort auf die meisten Pull-Aufrufe verwendet werden. Dabei haben die Attribute folgenden Zweck:

|  |  |  |
| --- | --- | --- |
| Attribut | Typ | Zweck |
| Antwort | Boolean | True => Ok; False => Fehlerfall |
| Antwort\_Bezug | String | aufgerufene Methode |
| Nachricht | String | (Optional) Erläuternder Text |
| Referenz\_Objekt | Object | (Optional) Angefordertes Objekt, holeKarte(KarteID) |

Tabelle 2: ServerAntwort Attribute

## Lobby

Diese Szene bietet einen Chat, wo sich verschiedene potentielle Spieler Nachrichten schreiben können. Hier gibt es dann zum Beispiel die Möglichkeit eine Nachricht in den Lobbychat selber zu schicken, oder an einen anderen Spieler in Form einer „DM[[1]](#footnote-1)“. Des Weiteren soll es natürlich eine Möglichkeit geben eine Partie selbst zu erstellen sowie einer auf Spieler wartenden Partie beizutreten.

Als Feature könnte hier noch stehen, dass man Spieler zu einer privaten Partie per DM einlädt, zum Anfang sollten jedoch alle „wartenden“ Partien öffentlich sichtbar sein.

Abbildung 4: Beispielhafte Lobbyszene

Spieler123: Wer will ne Runde rumspielen?

Schrimpfan: An dir? Keiner :P

ILuvTurtlz: Noice, 2vs2? Ich kann Kmplz noch rdyn

Spieler123 has disconnected

Spieler123

ILuvTurtlz

Schrimpfan

Hier tippen um Nachricht in die Lobby zu senden …

Spiel erstellen

Spiel beitreten

*Mögliche Datenkommunikation:*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Methode | Parameter |  | Antwort | Pull | Push | Sync | Async |
| ErstellePartie | Spieler, ConfigId | -> | Ok/Error, PartieId | X |  | X | X |
| ErstelleKonfiguration | Spieler, Config | -> | Ok/Error, ConfigId | X |  | X | X |
| SendeNachrichtLobby | Spieler, Nachricht | -> | Ok/Error | X |  | X | X |
| SendeNachrichtSpieler | Spieler, Spieler, Nachricht | -> | Ok/Error | X |  | X | X |
| EmpfangeNachrichten | -Ohne- | <- | Ok/Error, Chat+DMs | X | X | X | X |

Tabelle 3: Lobby Kommunikation

Die Liste möglicher Funktionen ist stark abhängig davon wie viele Features insgesamt umgesetzt werden sollen. Die hier aufgeführten stellen die grundlegenden dar.

*Mögliche Strukturen:*

Abbildung 5: Konfigurations-Objekt

Konfiguration

id : GUID

levelId : GUID

rundenBisSuddenDeath : Int

erlaubteWaffen[] : Waffe[]

rundenZeitlimitInS : Int

Abbildung 6: Waffen-Objekt

Waffe

id : GUID

name : String

image : Image

flugVerhalten : FlugVerhalten

trefferVerhalten : TrefferVerhalten

SplitterVerhalten : SplitterVerhalten

basisSchaden : Int

Abbildung 7: TrefferVerhalten-Objekt

TrefferVerhalten

id : GUID

explosionsZonen[] : [Float, Float]

explosionsMaxRadius : Float

FlugVerhalten

id : GUID

masse : Float

luftwiderstand : Float

muendungsgeschwindigkeit : Float

Abbildung 8: FlugVerhalten-Objekt

Nachricht

id : GUID

zeitstempel : DateTime

sender : Spieler

empfaenger : Spieler

inhalt : String

einladungInPartie : Partie

Abbildung 9: SplitterVerhalten-Objekt

SplitterVerhalten

id : GUID

waffe : Waffe

anzahl : Int

verteilungsWinkel : Float

Abbildung 10: Nachricht-Objekt

Level

id : GUID

hintergrund : Image

vordergrund : Image

maske : Image

hoehe : Int

breite : Int

features : FeatureFlagEnum

Abbildung 11: Partie-Objekt

Partie

id : GUID

zeitstempel : DateTime

konfiguration : Konfiguration

teilnehmer : [Spieler, Status]

status : PartieStatusEnum

Abbildung 12: Level-Objekt

In diesem Abschnitt werden die Waffen und dazugehörige Verhalten nicht erläutert, die Erläuterung findet sich dann im Kapitel „Partie – Spiel Schleife“ im Unterabschnitt Waffeneinsatz.

Das Objekt Partie enthält alle nötigen Informationen einer Partie, wie die durch den Partie-Host eingestellte Konfiguration, die Teilnehmer der Partie und deren jeweiliger Status, wie „bereit“, „waffenwahl“ oder „desync“ falls der Client unerwartet nicht antwortet, sowie den aktuellen Status der Partie und ein Zeitstempel wann die Partie erstellt wurde um zu bestimmen ob sie noch gültig (im Sinne von wahrscheinlich das diese noch weitergeführt wird) ist.

|  |  |  |
| --- | --- | --- |
| Attribut | Typ | Zweck |
| Id | GUID | Referenz auf dieses Objekt |
| Zeitstempel | DateTime | Zum bestimmen wann Partie abläuft |
| Konfiguration | Konfiguration | Einstellungen, wie Leveln, Rundenanzahl, usw. |
| Teilnehmer | [Spieler, Status] | Alle Teilnehmer dieser Partie und deren Status |
| Status | PartieStatusEnum | Status der Partie, wie „warte auf Spieler“, „Partie läuft“ oder „Partie beendet“ usw. |

Tabelle 4: Partie Attribute

Das Objekt Konfiguration könnte demnach alle Einstellungen, die unabhängig von der Karte / dem Level getroffen werden sollen, enthalten. Dies könnte zum Beispiel die Anzahl an Runden sein, die gespielt werden können, bis der SuddenDeath-Modus anfängt, und damit der Bolognese-Spiegel steigt, eine Einschränkung der Waffenwahl oder auch ein Zeitlimit die ein Spieler hat.

|  |  |  |
| --- | --- | --- |
| Attribut | Typ | Zweck |
| Id | GUID | Referenz auf dieses Objekt |
| levelId | GUID | Die GUID der zu spielenden Level |
| rundenBisSuddenDeath | Int | -x = Kein SD, 0 = Sofort SD, x = x Runden bis SD |
| erlaubteWaffen | [Waffe, Int] | Liste aller Waffen, mit Anzahl an „Munition“ < 0 => Kein Limit |
| rundenZeitlimitInS | Int | Sekunden, bis der Server den aktiven Spieler/Shrimp weiter schaltet. |

Tabelle 5: Konfiguration Attribute

Das Level-Objekt beschreibt hier die Daten, die ein Client braucht um das entsprechende Level darzustellen. Während des Spielens eines Levels ändert sich hier hauptsächlich nur die Maske, die dazu dient den Vordergrund zu beschneiden um die Zerstörung der Level abzubilden. Die Angaben für Höhe und Breite könnten hier auch für Skalierungszwecke dienen. Die FeatureFlagEnum soll dann besondere Eigenschaften der Level widerspiegeln. Diese Features könnten so etwas sein wie (Säure-)Regen, mögliche Booster die auftauchen, oder eben andere Besonderheiten, sind aber ein erweiterndes Feature für später.

|  |  |  |
| --- | --- | --- |
| Attribut | Typ | Zweck |
| Id | GUID | Referenz auf dieses Objekt |
| Hintergrund | Image/Blob | Das Hintergrundbild des Levels |
| Vordergrund | Image/Blob | Das Vordergrundbild des Levels |
| Maske | Image/Blob | Maske zum Beschneiden des Vordergrunds |
| Hoehe | Int | Höhe der Level (und damit der Bilder) |
| Breite | Int | Breite der Level (-“-) |
| Features | FeatureFlagEnum | Features die dieses Level beinhält |

Tabelle 6: Level Attribute

Nachrichten spezifizieren durch ihre Attribute ihren Adressaten. Ist der Empfänger „null“ so ist der Adressat die Lobby, bzw. der allgemeine Level-Chat, sofern der entsprechende Spieler an einer aktiven Partie teilnimmt. Ist der Empfänger dagegen nicht leer so soll es eine Nachricht an einen anderen Spieler sein, der diese in z.B. einem persönlichen Postkasten erhält.

|  |  |  |
| --- | --- | --- |
| Attribut | Typ | Zweck |
| Id | GUID | Referenz auf dieses Objekt |
| Zeitstempel | DateTime | Zeitpunkt an dem die Nachricht versendet wurde |
| Sender | Spieler | Der Sender der Nachricht |
| Empfaenger | Spieler | (Optional) Empfänger der Nachricht |
| Inhalt | String | Text der Nachricht |
| EinladungInPartie | Partie | Einladung der referenzierten Partie beizutreten |

Tabelle 7: Nachricht Attribute

ENDE.

1. Direct Message, Direkte Nachricht oder *auch* private Nachricht. [↑](#footnote-ref-1)