Architektur von Videospielen

Nico Hauff, Vincent Langenfeld, Frank Schüssele

November 9, 2023

Organisation

Beginnen Sie jetzt mit dem Programmieren

- z.B.: `/<spielname>/<spielname>.sln`
- Jenkins und Sonar bald
- Arbeiten Sie an einem Projekt, integrieren Sie kontinuierlich
- MS01: Nächste Woche (Spielobjekt in der Welt bewegbar, bewegliche Ansichten, Level laden/speichern, Soundausgabe)
- Nehmen Sie die wöchentlichen Aufgaben als Chance sich zu verbessern

Ideenpräsentation (Do. Nächste Woche)

- Ab 14 Uhr ct. hier
  - Ab 14 Uhr st. Technik testen
  - Gesamte Gruppe
- max 10min (+ 5min Fragen)
- Bereiten Sie sich vor:
  - Worum geht es?
  - Zentrale Spielmechanik?
  - Spielablauf? (gewinnen, verlieren)
  - Warum macht es Spaß?

Architektur

Architektur

Engine

- Eingabe
- Menüs und Popups
- Persistente Einstellungen
- Speichern/Laden
- Rendern und Kamera
- Pathfinding
- Kollisionserkennung
- Netzwerk
- Objektverwaltung

Spielmechanik

- Spielobjekte und Eigenschaften
- Interaktion
- KI

Tools

- Modelle integrieren
- Texturen integrieren
- Sounds integrieren
- Level erstellen
- Debugging output

Content

- Sound und Musik
- Modelle
- Sprites
- Levels

Architektur

Engine

- Eingabe
- Menüs und Popups
- Persistente Einstellungen
- Speichern/Laden
- Rendern und Kamera
- Pathfinding
- Kollisionserkennung
- Netzwerk
- Objektverwaltung

Spielmechanik

- Spielobjekte und Eigenschaften
- Interaktion
- KI

Tools

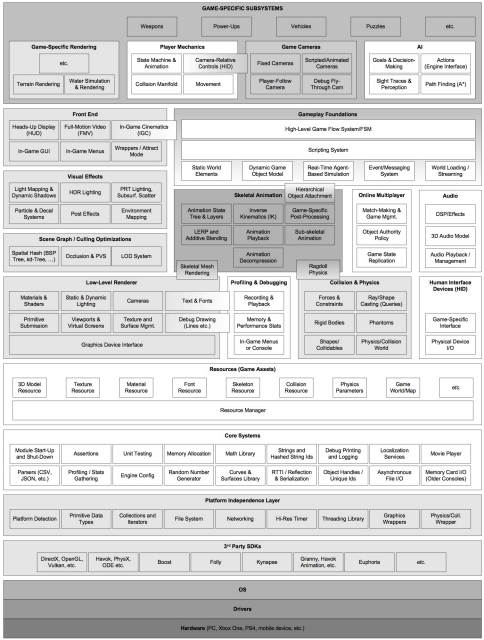
- Modelle integrieren
- Texturen integrieren
- Sounds integrieren
- Level erstellen
- Debugging output

Content

- Sound und Musik
- Modelle
- Sprites
- Levels

A Man Ran (WS17/18)

Architektur

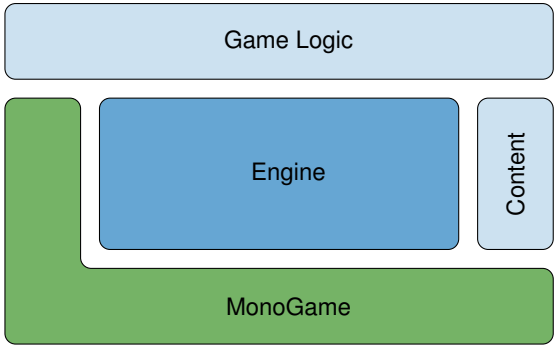


Gregory, *Game Engine Architecture*, 2019

Architektur

Am Besten Top-Down anfangen:

- Zuerst Struktur aufbauen
- Funktionalität aufteilen
- Separation of Concerns



MonoGame und .NET

MonoGame und .NET

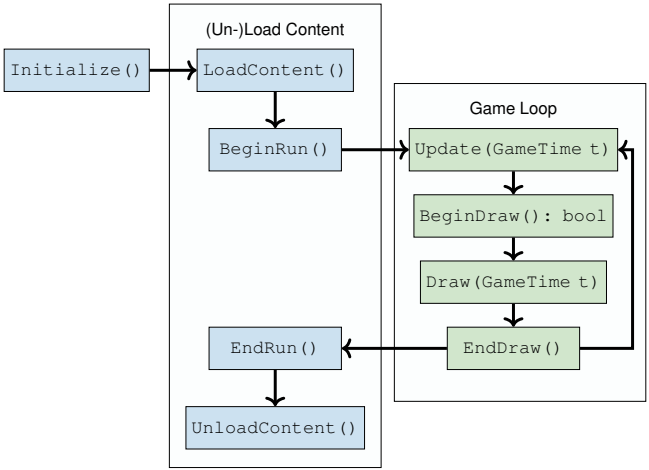
.net

- Mathematik
- Zufallszahlen
- (De-)Serialisierung in Json/XML/Binär
- Datenstrukturen (Liste, Menge, ...)
- Debugging
- Profiling

MonoGame

- Abstraktion
  - Grafik
  - Sound
  - Input
- Content Pipeline
- Einfache Anzeigemethoden (BasicEffect, SpriteBatch)
- Datentypen
  - Vector2, Vector3
  - Matrix
  - ...
- Game Loop

MonoGame Lifecycle



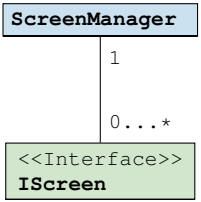
Bausteine

Screen Management

Ziel

Entkoppeln von Verwaltung der Spielzustände und deren konkreter Implementierung

- Screens implementieren ihre jeweilige Funktionalität
  - Hauptmenü
  - Optionsmenü
  - Spielansicht
  - HUD
  - Ladebildschirm
  - ...
- ScreenManager verwaltet alle Screens
  - Kontrollfluss
  - Hinzufügen
  - Entfernen

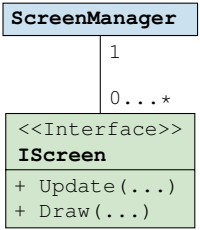


Screen Management

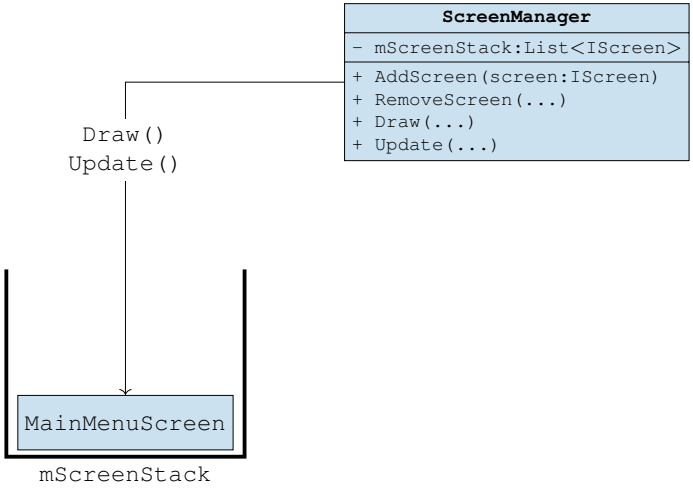
Ziel

Entkoppeln von Verwaltung der Spielzustände und deren konkreter Implementierung

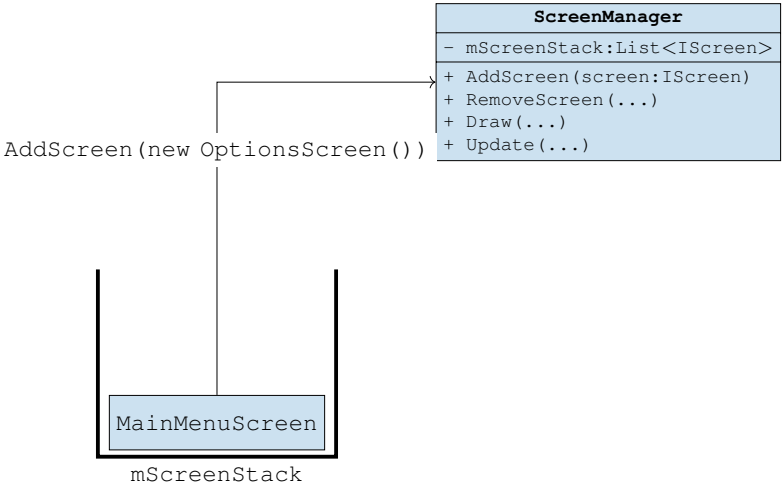
- Screens implementieren ihre jeweilige Funktionalität
  - Hauptmenü
  - Optionsmenü
  - Spielansicht
  - HUD
  - Ladebildschirm
  - ...
- ScreenManager verwaltet alle Screens
  - Kontrollfluss
  - Hinzufügen
  - Entfernen



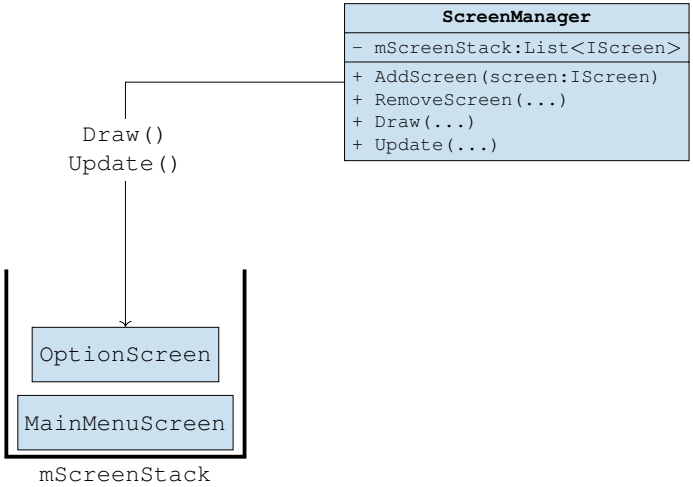
Screen Management



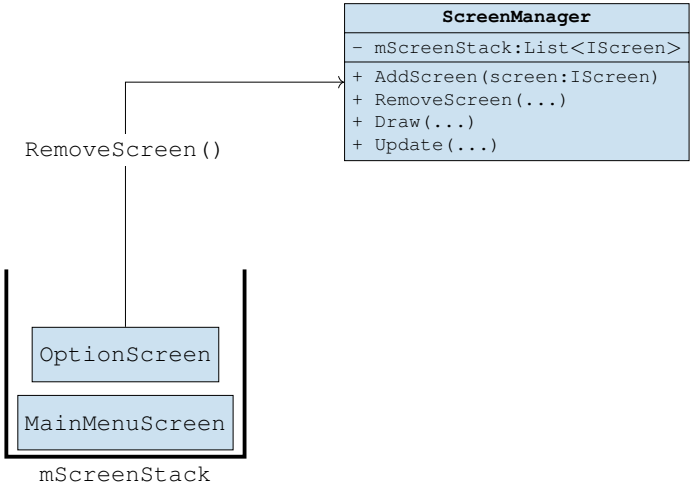
Screen Management



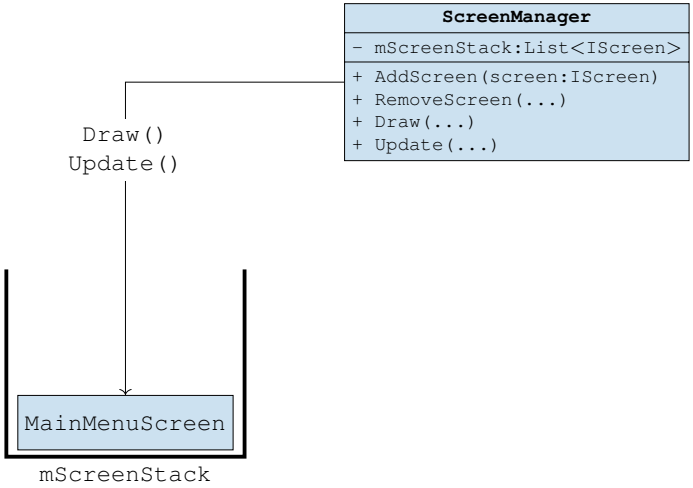
Screen Management



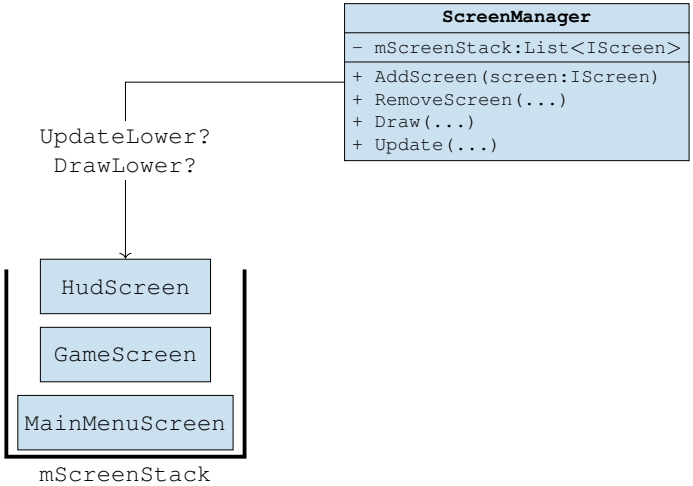
Screen Management



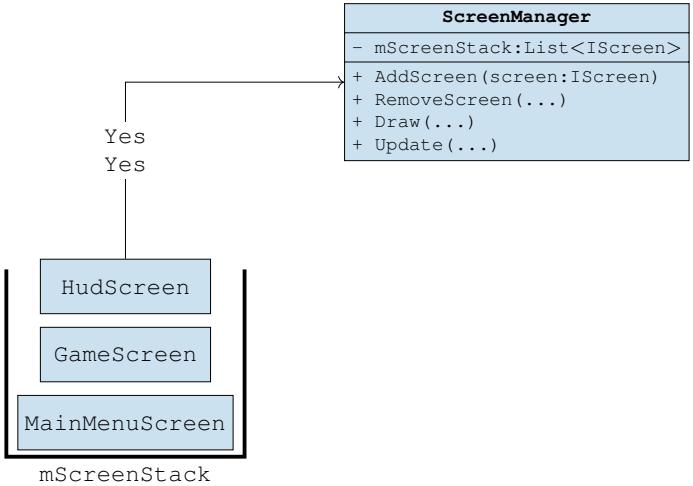
Screen Management



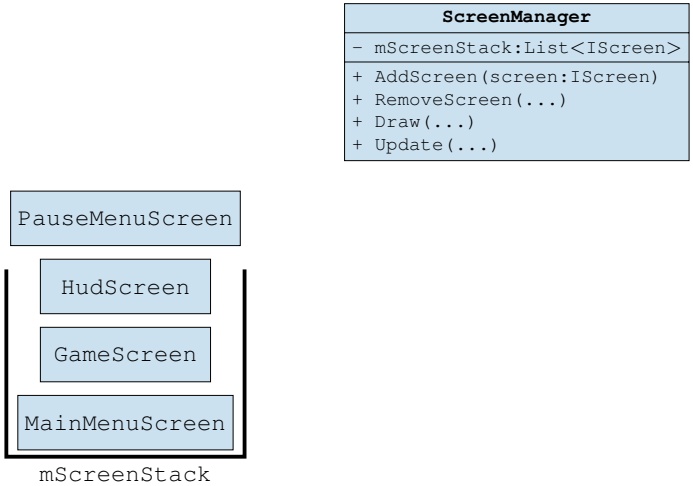
Screen Management



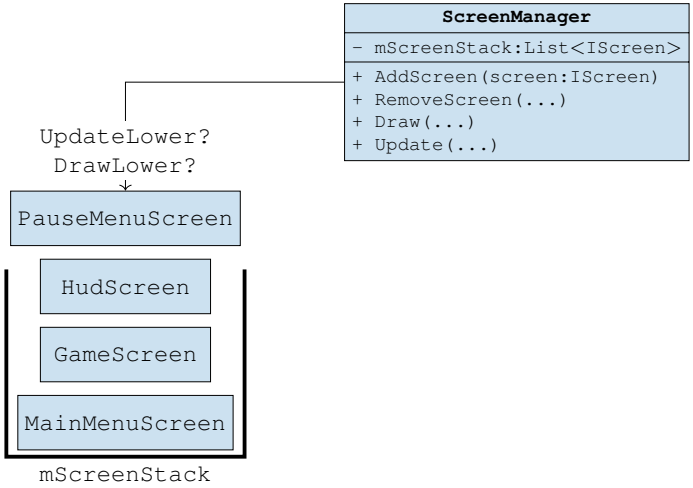
Screen Management



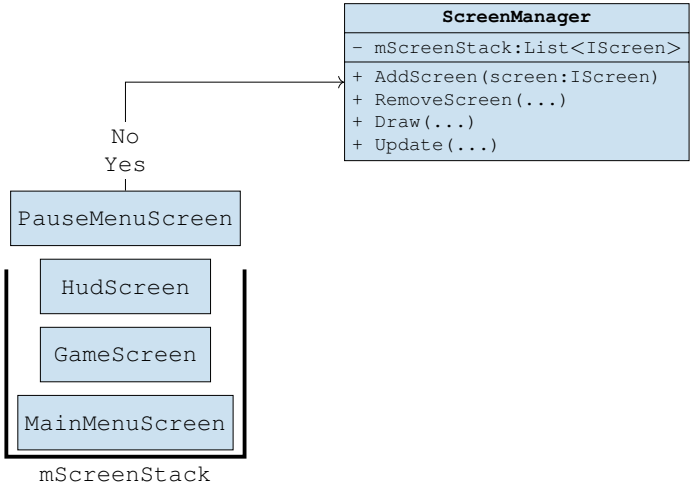
Screen Management



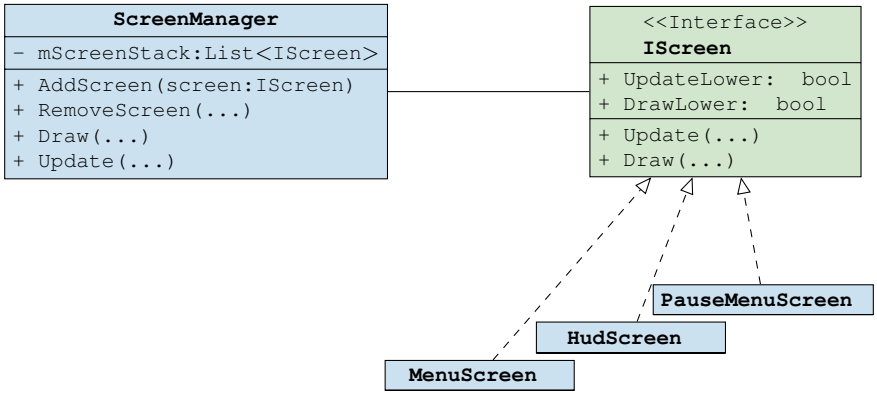
Screen Management



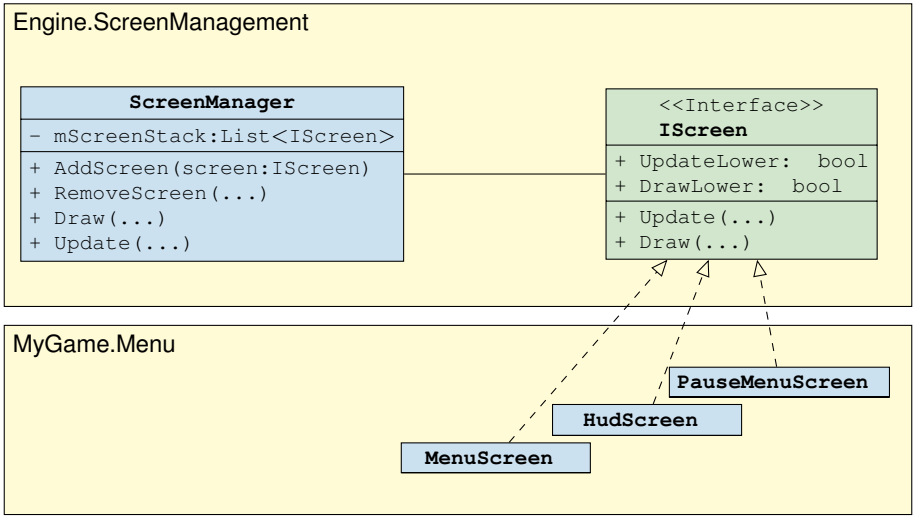
Screen Management



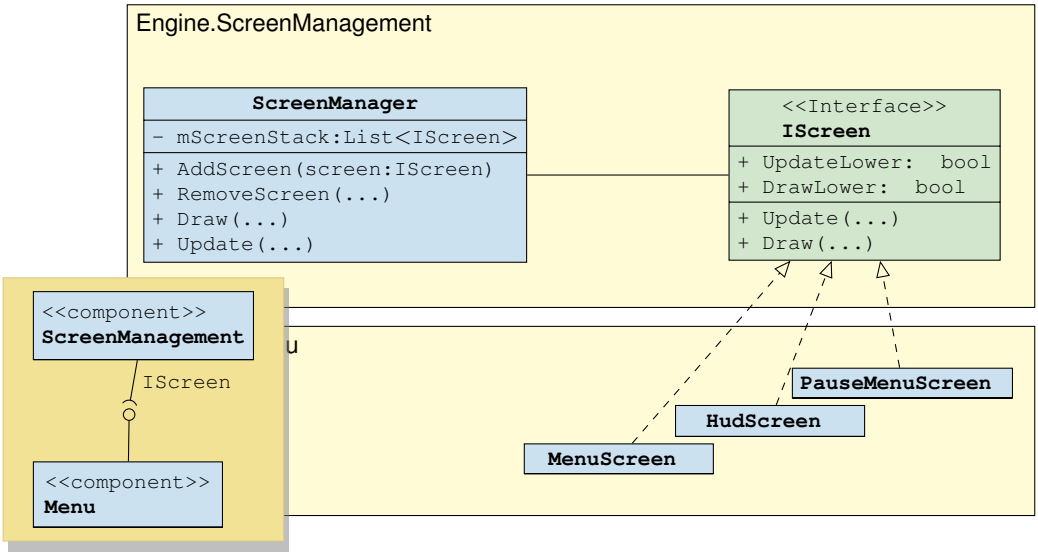
Screen Management



Screen Management



Screen Management



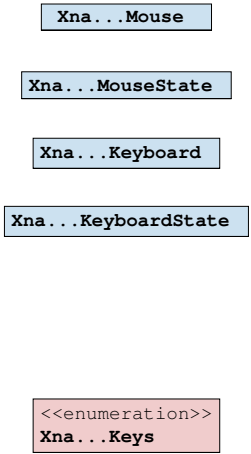
Input Management

Ziel

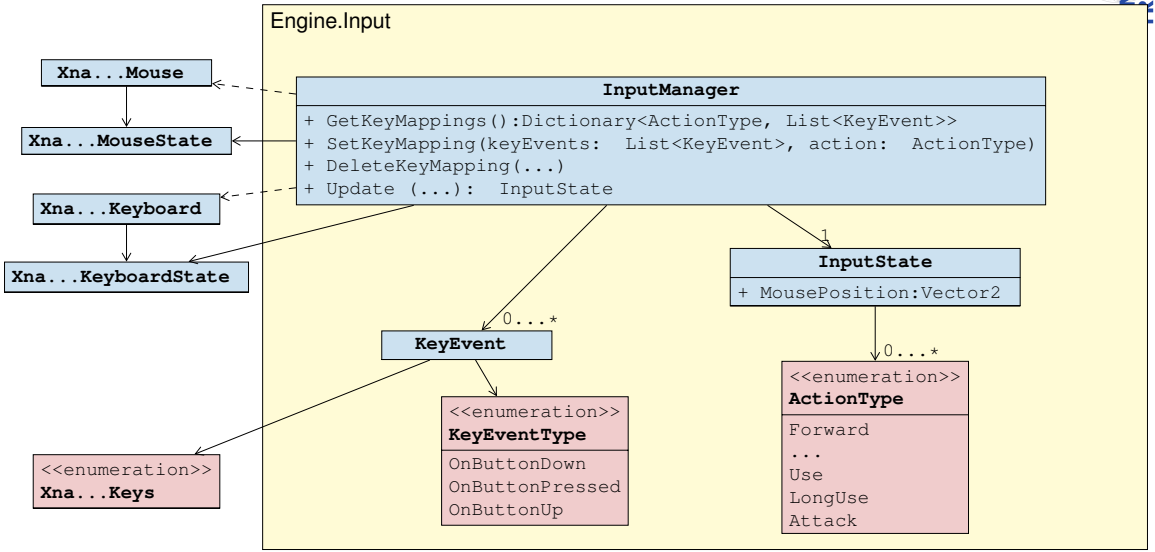
Abstraktion der konkreten physischen Eingaben (und Eingabegeräte)

- Tastatur, Maus, Gamepad, ...
- MonoGame Namespace `Microsoft.Xna.Framework.Input`
  - `KeyboardState mKeyboardState = Keyboard.GetState();`
  - Statusinformation per Frame: Zustand der Tasten, Mausposition
  - keine Historie
- Wir brauchen
  - Statusinformation in Abhängigkeit von Zeit
    (z.B.: A wurde dieses Frame losgelassen)
  - Abstraktion von verschiedenen
    Eingabegeräten/Tasten zu Aktionen
  - Eventuell abstraktion von verschiedenen
    Eingabegeräten zu Aktionen

Input Management



Input Management



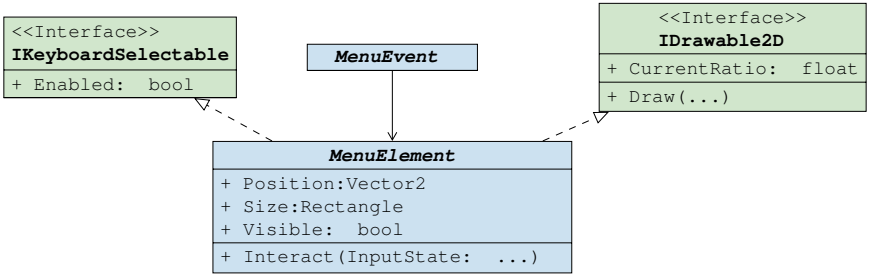
Menüs

Ziel

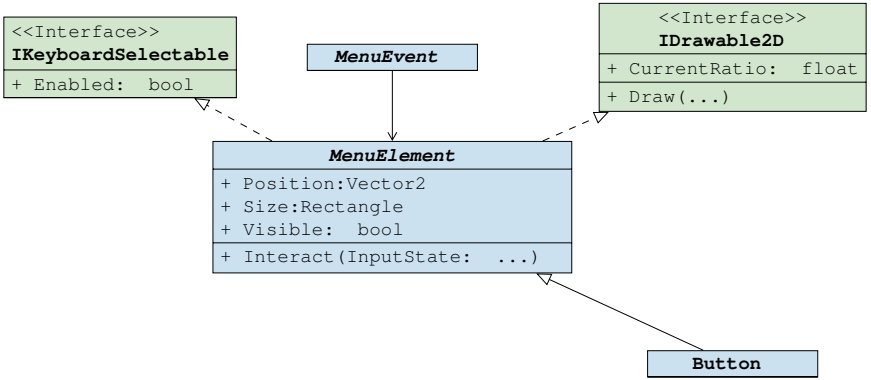
Alle Eingabeelemente einheitlich verwalten und einfach erweiterbar machen.

- Viele Elemente die sich sehr ähnlich verhalten
  - Panels
  - Buttons
  - Labels
  - Textboxen
  - Checkboxen
- Auch im Hud verwendbar

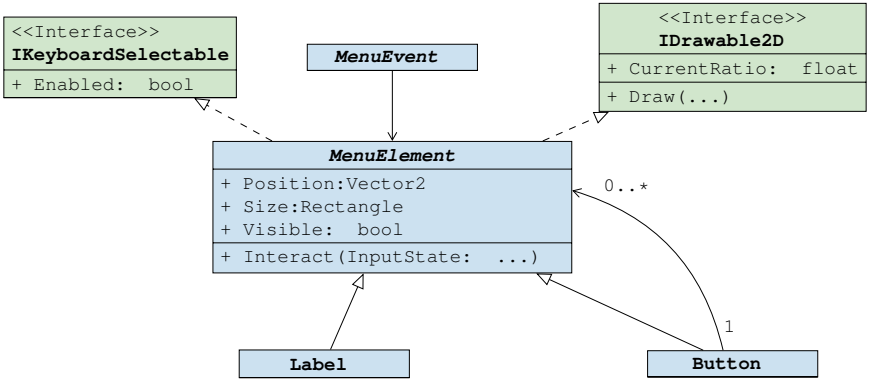
Menüs


<<Interface>>
IKeyboardSelectable
+ Enabled: bool
MenuEvent
<<Interface>>
IDrawable2D
+ CurrentRatio: float
+ Draw(...)
MenuElement
+ Position:Vector2
+ Size:Rectangle
+ Visible: bool
+ Interact(InputState: ...)

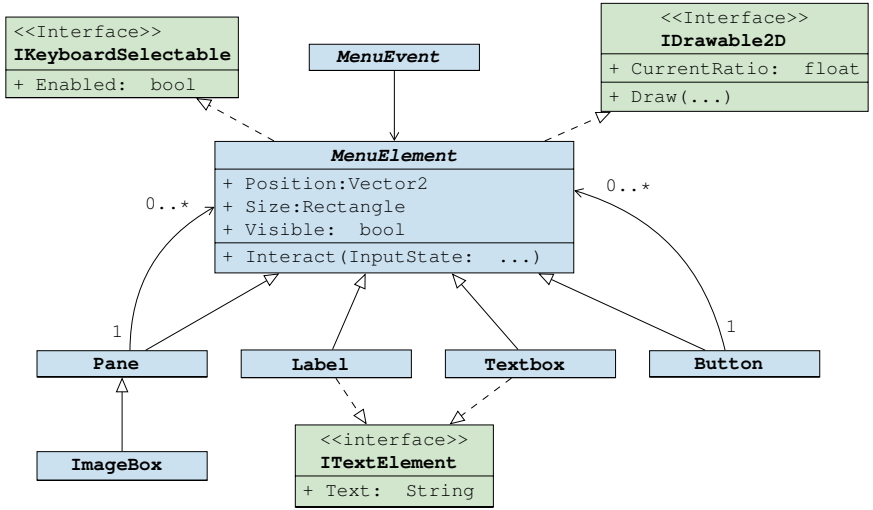
Menüs


<<Interface>>
IKeyboardSelectable
+ Enabled: bool
MenuEvent
<<Interface>>
IDrawable2D
+ CurrentRatio: float
+ Draw(...)
MenuElement
+ Position:Vector2
+ Size:Rectangle
+ Visible: bool
+ Interact(InputState: ...)
Button

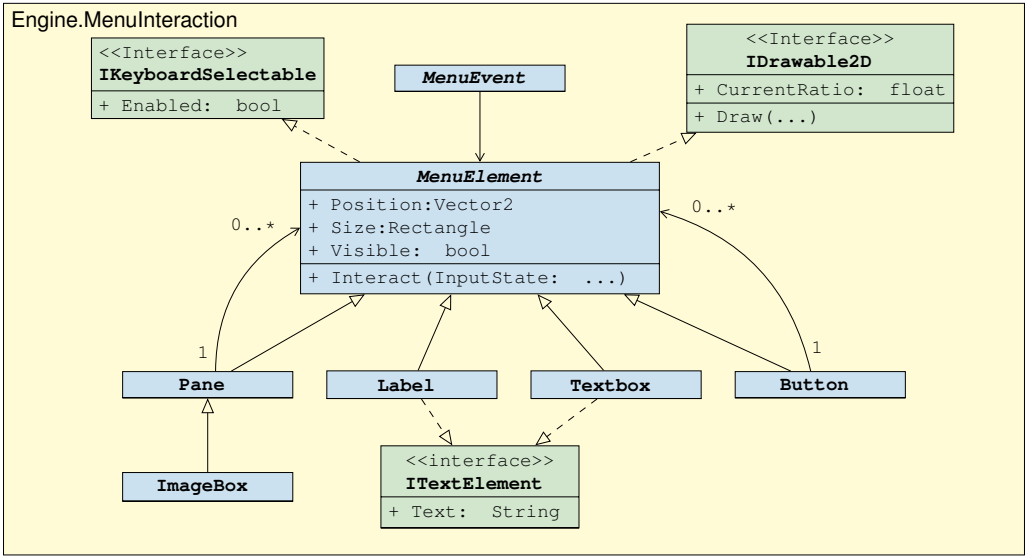
Menüs


<<Interface>>
IKeyboardSelectable
+ Enabled: bool
MenuEvent
<<Interface>>
IDrawable2D
+ CurrentRatio: float
+ Draw(...)
MenuElement
+ Position:Vector2
+ Size:Rectangle
+ Visible: bool
+ Interact(InputState: ...)
0..*
1
Label
Button

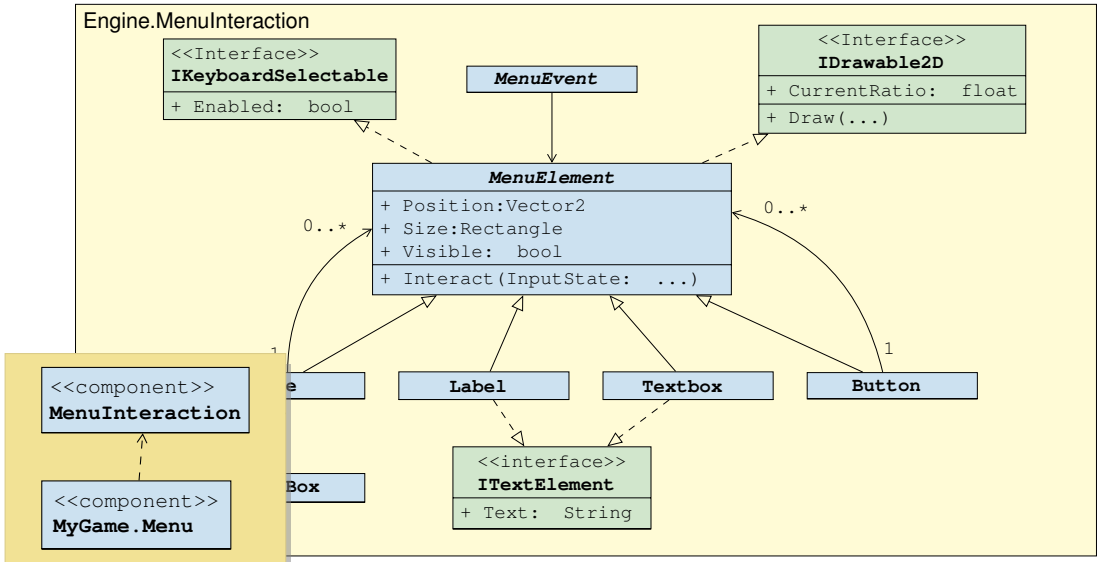
Menüs


<<Interface>>
IKeyboardSelectable
+ Enabled: bool
MenuEvent
<<Interface>>
IDrawable2D
+ CurrentRatio: float
+ Draw(...)
MenuElement
+ Position:Vector2
+ Size:Rectangle
+ Visible: bool
+ Interact(InputState: ...)
0..*
0..*
1
1
Pane
Label
Textbox
Button
ImageBox
<<interface>>
ITextElement
+ Text: String

Menüs


Engine.MenuInteraction
<<Interface>>
IKeyboardSelectable
+ Enabled: bool
MenuEvent
<<Interface>>
IDrawable2D
+ CurrentRatio: float
+ Draw(...)
MenuElement
+ Position:Vector2
+ Size:Rectangle
+ Visible: bool
+ Interact(InputState: ...)
0..*
0..*
1
1
Pane
Label
Textbox
Button
ImageBox
<<interface>>
ITextElement
+ Text: String

Menüs


Engine.MenuInteraction
<<Interface>>
IKeyboardSelectable
+ Enabled: bool
MenuEvent
<<Interface>>
IDrawable2D
+ CurrentRatio: float
+ Draw(...)
MenuElement
+ Position:Vector2
+ Size:Rectangle
+ Visible: bool
+ Interact(InputState: ...)
0..*
0..*
1
1
Label
Textbox
Button
<<interface>>
ITextElement
+ Text: String
<<component>>
MenuInteraction
<<component>>
MyGame.Menu

Pathfinding

Offline

Auf dem kürzesten Weg von A nach B

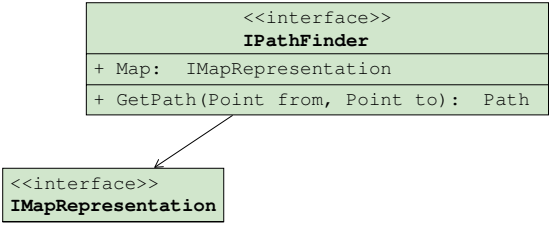
- Berücksichtigt Welt und unbewegliche Objekte
- Meistens A*
- Viele mögliche Weltrepräsentationen:
  Grid, Hierarchical Grid, Waypoint Graph,
  Navigation Mesh,...

Online

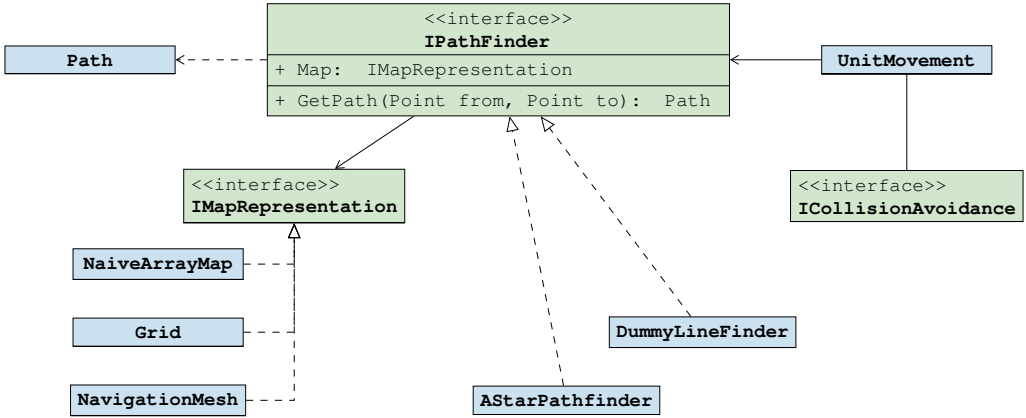
Unterwegs mit niemandem zusammenstoßen

- Interaktion mit anderen beweglichen Objekten
  (z.B.: ausweichen)
- Verschiedene Verfahren:
  Steering, Flocking, Flow Fields, ...

Pathfinding



Pathfinding



Spielobjekte

Spielobjekte sind alle Dinge, die eine Repräsentation in der Spielwelt haben.

- Charaktere, Fahrzeuge, Bäume, Raketen, Gras, Steine, Trigger, Lichter, Sounds, ...
- Spielobjekte müssen manchmal
  - gezeichnet werden
  - sich bewegen
  - zerstörbar sein
  - miteinander kollidieren
  - etc.
- Wie verwaltet man so viele (verschiedene) Objekte effizient?

Spielobjekte

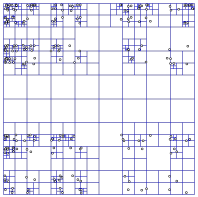
Spielobjekte sind alle Dinge, die eine Repräsentation in der Spielwelt haben.

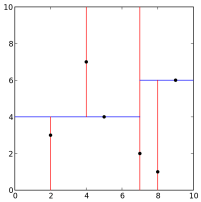
- Charaktere, Fahrzeuge, Bäume, Raketen, Gras, Steine, Trigger, Lichter, Sounds, ...
- Spielobjekte müssen manchmal
  - gezeichnet werden
  - sich bewegen
  - zerstörbar sein
  - miteinander kollidieren
  - etc.
- Wie verwaltet man so viele (verschiedene) Objekte effizient?
  - Zur Laufzeit?
  - Im Softwaredesign?

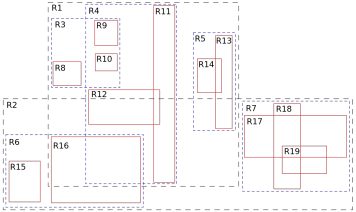
Szenengraph

Ein Szenengraph ist eine zentrale Datenstruktur, die der logischen und räumlichen Verwaltung der Spielobjekte dient.

- Antwort auf räumliche Fragen
- Update- und Drawaufrufe an (relevante) Spielobjekte weitergeben
- Beispiele: Liste, Heap, Quad-/ Octree, KD-Tree, R-Tree, ...







Spielobjekte

Objektzentriert

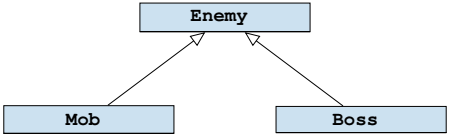
- Spielobjekte sind Klassen mit
  - Eigenschaften des Spielobjekts
  - Verhalten des Spielobjekts
- Spielwelt ist eine Menge von Instanzen der Spielobjekte

Eigenschaftszentriert

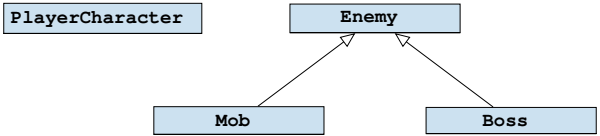
- Jedes Spielobjekt ist nur eine ID
- Tabellen für Eigenschaften aller Objekte
- Verhalten durch Operationen auf Tabellen
- Ähnlich zu Datenbanken

Spielobjekte und Vererbung

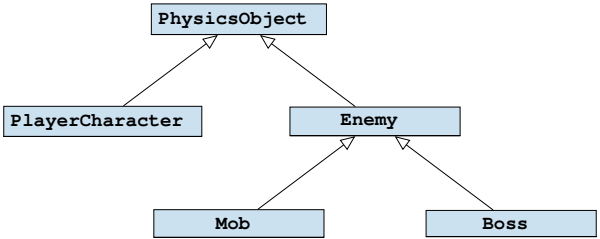
Spielobjekte und Vererbung



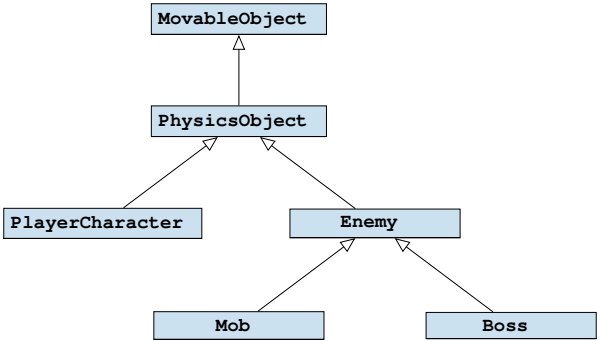
Spielobjekte und Vererbung



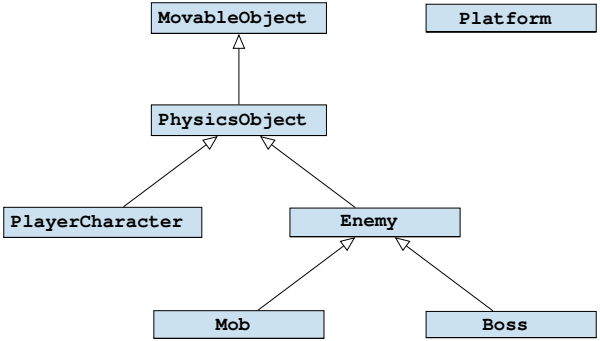
Spielobjekte und Vererbung



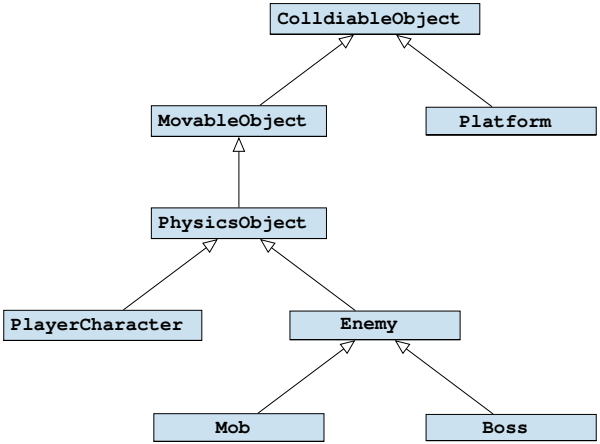
Spielobjekte und Vererbung



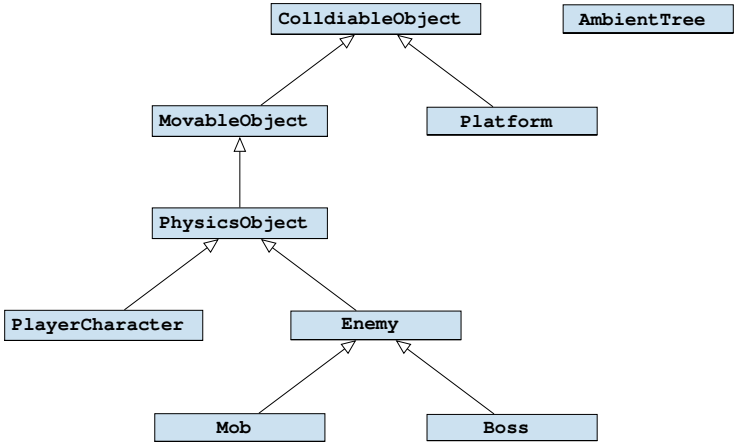
Spielobjekte und Vererbung



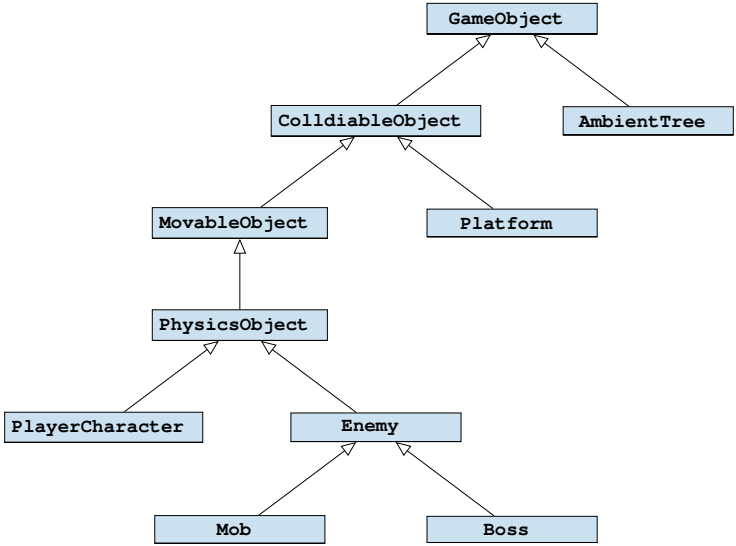
Spielobjekte und Vererbung



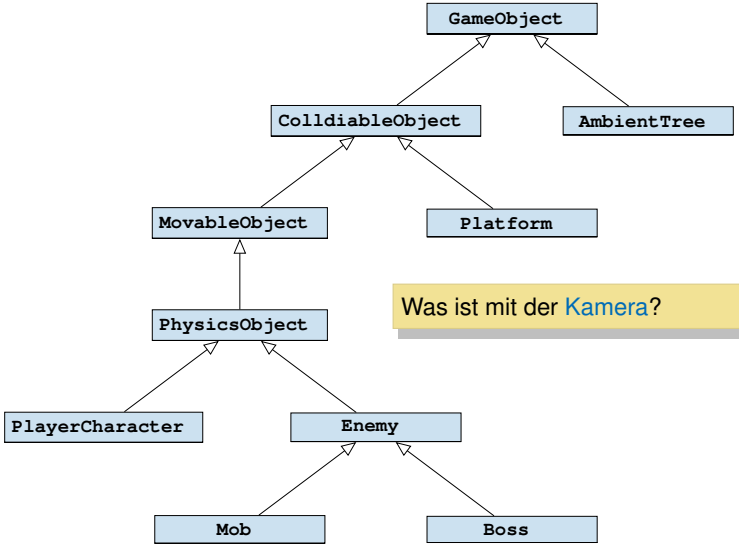
Spielobjekte und Vererbung



Spielobjekte und Vererbung



Spielobjekte und Vererbung

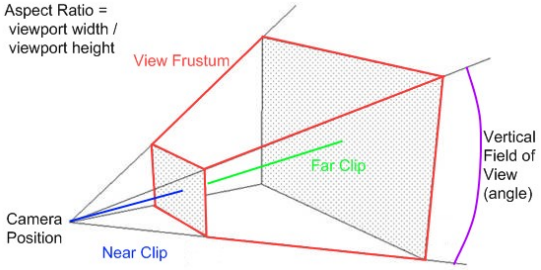


Was ist mit der Kamera?

Spielobjekte

Eine einfache Kamera

- Speichert z.B.:
  - View Matrix
  - Projection Matrix
- Grundlage für Picking
- Definiert View Frustrum

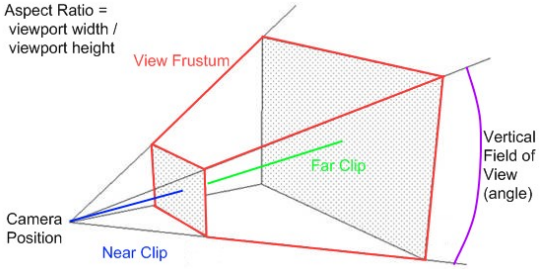


Spielobjekte

Eine einfache Kamera

- Speichert z.B.:
  - View Matrix
  - Projection Matrix
- Grundlage für Picking
- Definiert View Frustrum

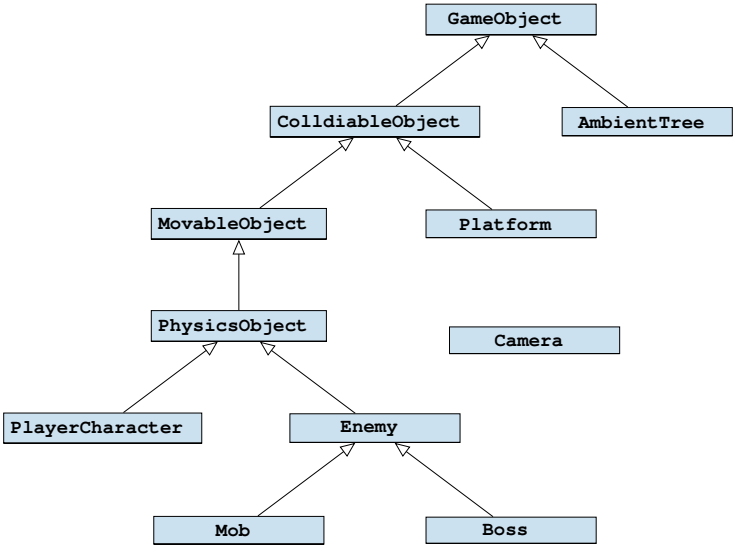
- Teil der Spielwelt
- Nicht kollidierend
- Beweglich



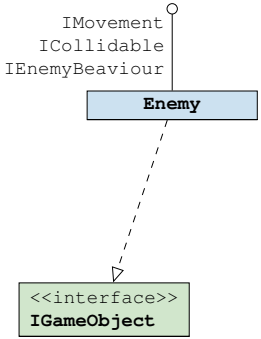
Spielobjekte

	Tree	Platform	PlayerCharacter	Mob	BossMob	Camera
GameObject	✓	✓	✓	✓	✓	✓
Collidable		✓	✓	✓	✓	
Movable			✓	✓	✓	✓
Physics			✓	✓	✓	
EnemyBehaviour				✓	✓	

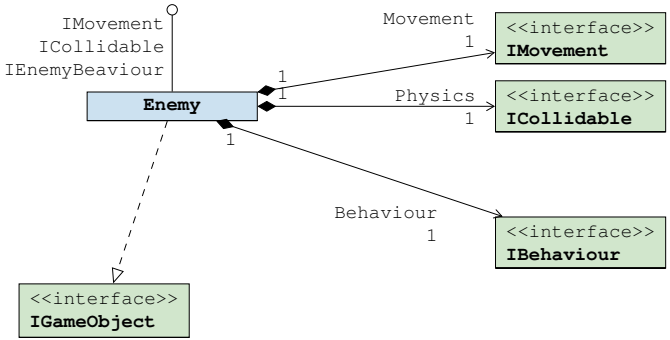
Spielobjekte und Vererbung



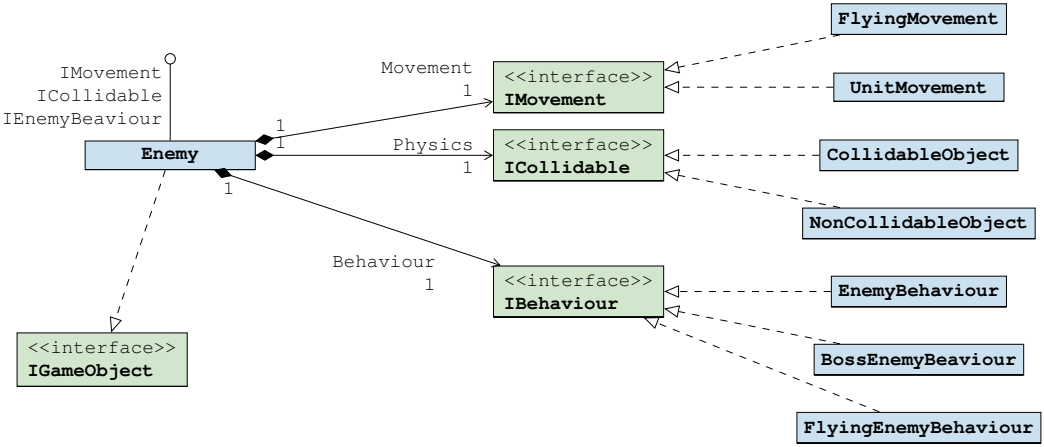
Spielobjekte und Komposition



Spielobjekte und Komposition



Spielobjekte und Komposition



Libraryfalle

Die Libraryfalle

Die Libraryfalle

- Es existieren viele Libraries
- Pro:
  - Rad nicht neu erfinden
  - Optimierte Lösungen für bekannte Probleme
- Aber:
  - Problem muss verstanden sein
  - Library muss verstanden werden
  - Versteckte Kosten wenn Library fehlerhaft/schlecht/unpassend/...

Die Libraryfalle

Die Libraryfalle

- Es existieren viele Libraries
- Pro:
  - Rad nicht neu erfinden
  - Optimierte Lösungen für bekannte Probleme
- Aber:
  - Problem muss verstanden sein
  - Library muss verstanden werden
  - Versteckte Kosten wenn Library fehlerhaft/schlecht/unpassend/...
- Vorher das Risiko abschätzen:
  - Wie zentral ist die Library?
  - Wie reif ist die Library?
  - Können wir das einfacher selbst machen?

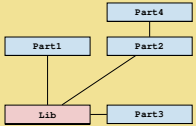
Die Libraryfalle

Die Libraryfalle

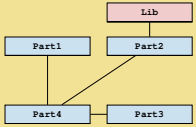
- Es existieren viele Libraries
- Pro:
  - Rad nicht neu erfinden
  - Optimierte Lösungen für bekannte Probleme
- Aber:
  - Problem muss verstanden sein
  - Library muss verstanden werden
  - Versteckte Kosten wenn Library fehlerhaft/schlecht/unpassend/...
- Vorher das Risiko abschätzen:
  - Wie zentral ist die Library?
  - Wie reif ist die Library?
  - Können wir das einfacher selbst machen?

Wie zentral ist die Library?

-Viel möglicher Schaden:



- Weniger möglicher Schaden:



Organisation

Architekturbesprechung

Ziel

Architektur anhand von Designprinzipien verbessern, Probleme frühzeitig aufdecken.

Ablauf:

- Kurze Zusammenfassung des Spiels
- Übersicht über die Architektur
  - Was machen die einzelnen Komponenten
- Vorstellen der Architektur anhand der Szenarien
- Fragen und zusätzliche Szenarien

Quelle: Dozent
Auslöser: Spieler
Umgebung: Ich befinde mich im Hauptmenü. Ich drücke auf Spiel Starten, es werden Spielobjekte erzeugt.

Architekturbesprechung

Ziel

Architektur anhand von Designprinzipien verbessern, Probleme frühzeitig aufdecken.

Ablauf:

- Kurze Zusammenfassung des Spiels
- Übersicht über die Architektur
  - Was machen die einzelnen Komponenten
- Vorstellen der Architektur anhand der Szenarien
- Fragen und zusätzliche Szenarien

Quelle: Dozent
Auslöser: Spieler
Umgebung: Ich befinde mich im Hauptmenü. Ich drücke auf Spiel Starten, es werden Spielobjekte erzeugt.

Von Anfang an Versuchen:
- Was ist Engine, was konkretes Spiel?
- Was gehört zusammen oder hat nichts miteinander zu tun? (Separation of Concerns)
- Wurde etwas noch nicht bedacht?

Was nun?

- Fangen Sie jetzt an zu programmieren
  - In einem Projekt
  - Arbeiten Sie gemeinsam
  - Versuchen Sie Ihr Spiel immer wieder zu spielen
- GDD-Reviews lesen und Probleme beheben
  - Inhaltliche Probleme früh lösen
  - Stellen Sie sicher, dass Sie entwickeln was der Kunde wünscht, damit Ihr Projekt Erfolg hat.
- Nicht vergessen:
  - Architekturbesprechung (Doodles per Ticket)
  - Ideenpräsentation

Was nun?

- Fangen Sie jetzt an zu programmieren
  - In einem Projekt
  - Arbeiten Sie gemeinsam
  - Versuchen Sie Ihr Spiel immer wieder zu spielen
- GDD-Reviews lesen und Probleme beheben
  - Inhaltliche Probleme früh lösen
  - Stellen Sie sicher, dass Sie entwickeln was der Kunde wünscht, damit Ihr Projekt Erfolg hat.
- Nicht vergessen:
  - Architekturbesprechung (Doodles per Ticket)
  - Ideenpräsentation

Sprechstunde:
Donnerstags 14-18 Uhr im Pool.