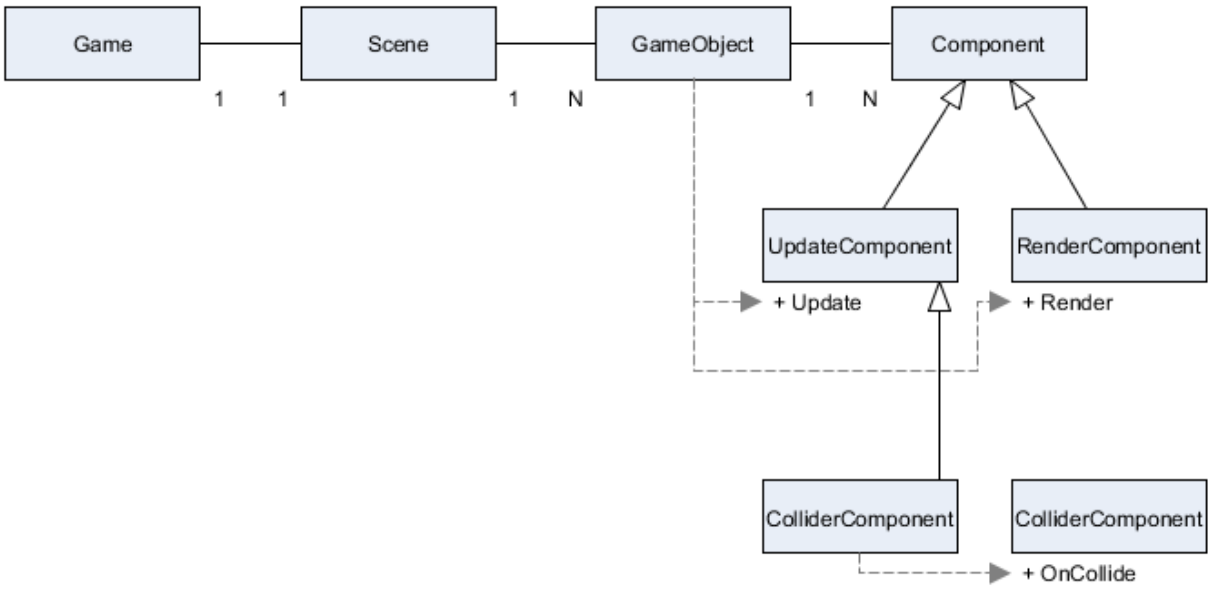
Strukturanalyse SpaceWars

Die ausgewählte Struktur für SpaceWars bzw. das Framework ist eine Delegation der Update- und Render-Zyklen vom Spielfenster an die aktive Szene und von dort über Spielobjekte an Komponenten.



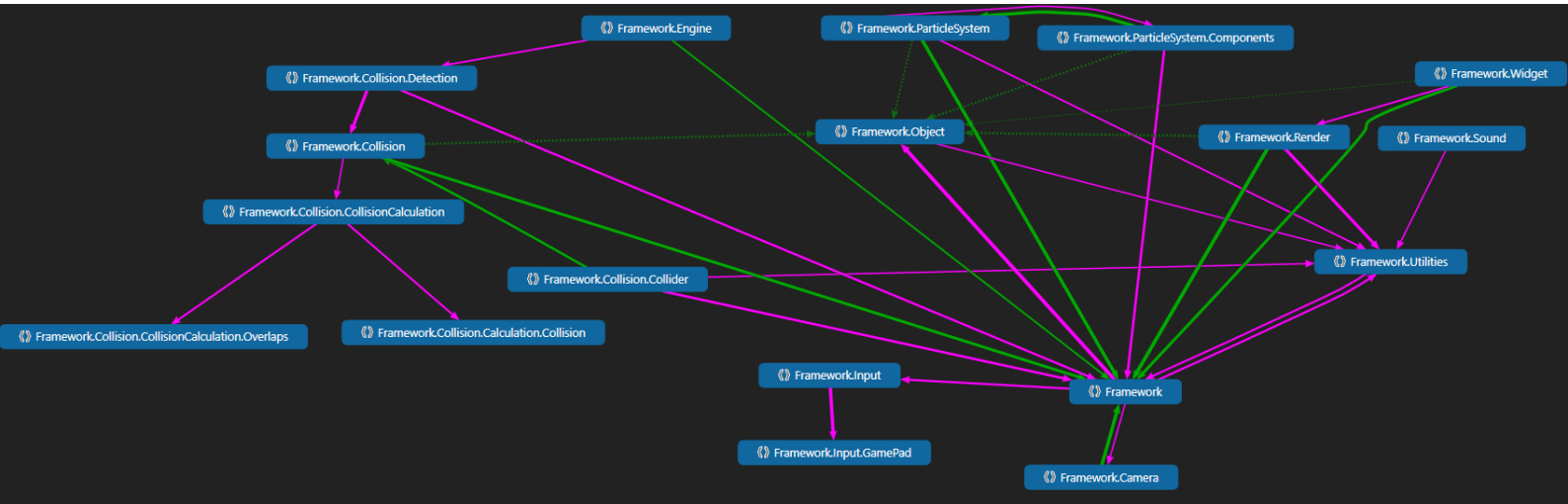
Das Spiel enthält immer eine aktive Szene, welche die Methoden Update und Render vom Spiele Fenster an die Spielobjekte in einer Liste weiter gibt.
Die Spielobjekte geben diesen Aufruf an die Liste der einzelnen Komponenten weiter.

Um die Objekte und Komponenten zu manipulieren hat jede Klasse Steuerungsmethoden:

- Game kann die aktuelle Szene setzen
- Scene kann Spielobjekte hinzufügen und entfernen und auflisten
- GameObject kann Komponenten hinzufügen und entfernen und auflisten
- Component kann zum zugehörigen Spielobjekt navigieren

Alle vom Framework bereitgestellten Möglichkeiten leiten von GameObject, Component, UpdateComponent oder RenderComponent ab. Bspw. RenderTextureComponent ist eine RenderComponent, welche eine Textur an die vom Spielobjekt vorgegebene Position, Rotation und Skalierung zeichnet.

Codemap Framework



Erweiterung

Mit den bereitgestellten Möglichkeiten durch UpdateComponent und RenderComponent lass sich auch weitere Komponenten-Typen definieren. Bspw. läuft im Hintergrund (sofern man das FrameworkEngineGameObject der Szene hinzugefügt hat) eine Kollisionserkennung ab und ruft falls Kollisionen stattfinden bei den beteiligten Spielobjekten und deren ColliderComponent die OnCollide-Methode auf.

Realisierung des Spiels

Das spiel ist unter Verwendung dieses Spiels in folgende (grobe) Struktur aufgebaut:

- Game - Main Methode
- Menu
  - MenuScene – Scene mit UI Spielobjekten
- Play
  - PlayScene – Scene mit Spielobjekten fürs Spiel
  - Enemy
    - ...
    - EnemySpawnBehaviour - Spielobjekt
  - Field
    - Background – Spielobjekt
    - Border – Spielobjekt
    - BorderShader - RenderShaderComponent
  - Player
    - FollowingCameraBehaviour – Spielobjekt
    - GameOverObservingBehaviour – Spielobjekt
    - Player – Spielobjekt
    - PlayerAttributes – Component
    - PlayerCollisionController – CollisionComponent
    - PlayerHelper – statische Klasse
    - PlayerMovementController – UpdateComponent
    - PlayerParticleEmitter – ParticleEmitter
    - PlayerShotController – UpdateComponent
  - Shot
    - Shot – Spielobjekt
    - ShotCollisionController – CollisionComponent
    - ShotMovementController – UpdateComponent
  - UI
    - …

Die Struktur des Frameworks bietet durch die nicht strikte Trennung zwischen Controller und Model die Möglichkeit flexibel den Inhalt des Spiels zu definieren. Sofern das Naming der Klassen in einem vorhanden Projekt gut ist, hat man auch als neuer Entwickler die Möglichkeit schnell in das Projekt einzusteigen.