RocketFight Class Graph Details

Die Farbe zeigt an, ob die Klasse auf dem Server, den Client oder beiden verwendet wird. Die Legende befindet sich dazu im Graphen.

# NetworkManager

## Client

* Stellt die Verbindung zum Server her.

## Server

* Initialisiert den Server selbst.
* Verwaltet verbundene Clients.
* Spawn *Player Prefabs* für jeden Client und weist diese zu.

# Player (Prefab)

## Client

### InputManager

* Liest Spieler Input aus. (Maus, Tastertur, Gamepad,…)
* Übersetzt Input in verwendbare Werte (Vectoren, Winkel,…)
* Sendet diese als RPC an den *PlayerLogic (Server)*
* Sendet Input-Auswertung an localen *Predictor*

### Predictor

* Bekommt Information über Input, Explosionen, *PlayerState*
* Nutzt diese Daten um Bewegung des vom Spieler gesteuerten Player vorher zu sagen
* Gib vorhersage an *Moveable* und *Physic* weiter zur Umsetztung
* Benutzt gebufferte *PlayerStates* von remote Players um deren Bewegung zu interpolieren und zu glätten.

## Server

### PlayerLogic

* Überprüft Input-RPCs und gibt diese an *Moveable* zur Umsetztung
* Erzeugt Rakten when shoot=true und cooldown abgelaufen ist
* Empfängt Explosions-Daten, prüft und lässt diese Effekte von *Physics* umsetzten
* Reagiert ggf. auf Collision mit anderen Objekten

## Shared

### Moveable

* Variablen und Methoden um Player Bewegung umzusetzten (auf Server und für Prediction)

### Physics

* Variablen und Methoden um Player Physik verhalten zu realisieren (auf Server und für Prediction)

### PlayerState

* Aktueller (buffered?) Status des Player Prefabs
  + Position
  + Rotation
  + ExplosionForce
  + Speed
  + …

### NetwerkView

* Unity3D Klasse
* Synchronisiert PlayerState zwischen Server und Client

### Collider

* Unity3D Klasse
* Macht Kollisionsabfragen

# Rocket (Prefab)

## Shared

### RocketState

* Aktueller Status der Rakete
  + Position
  + Speed
  + Richtung
  + …

### NetworkView

* Unity3D Klasse
* Synchronisiert RocketState zwischen Server und Client

### RocketLogic

* Initialisiert Richtung, Speed, usw.
* Kümmert sich um die Explosionsauswirkung bei Kollision (*Predictor* & *PlayerLogic*)
* (Auf Server) Zerstört die Rocket Instanz, nach der Kollision

## Collider

* Unity3D Klasse
* Macht Kollisionsabfragen