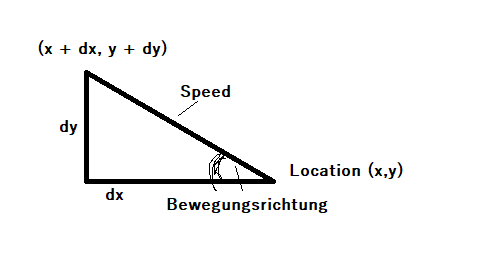


Hier ist der Link: <https://app.creately.com/diagram/SZas1R0PIFW/edit>

### **Klassenbeschreibung**

Raumschiff

* Handling: wie stark wird gelenkt bei User Input
* Groesse: Hitbox in Form eines Kreises, Float ist Radius
* Bewegungsrichtung: Winkel von 0-2pi
* move(): bewegt Location mit 1 \* Speed in Richtung Bewegungsrichtung (Bild unten ist spiegelverkehrt)



Spieler

* warum current\_barricade und barricade benötigt werden können wir am Mittwoch erklären

Barrikade

* delete\_by\_vicinity: es soll einen Bomben-buff geben welcher Barrikaden zerstört. Dafür wird das benötigt

Kiste:

* expire : Int sind die Nummer an Frames bis die Kiste verschwindet

Buff:

* apply und revert sind zwei funktionen welche das Spieler Objekt verändern. Wenn die Kiste von einem Spieler eingesammelt wird, verschwindet Sie und der Spieler bekommt das Buff Objekt in sein Buffarray. Desweiteren wird apply auf den Spieler angewandt. Soblad die duration in Frames im Buffarray 0 erreicht wird revert auf den Spieler angewandt und der Buff wird aus dem Buffarray glöscht.

Spiel:

* crash() : checkt ob Spieler mit Barrikaden oder den Wänden kollidieren
* step() : updated das Spiel um einen Frame

Server

* current\_actions : Der Server erhält Input Eingaben des Spielers als Strings und speichert diese in dieser Variable. Beim nächsten Frame werden alle Actions in der variable dann mit realize\_actions() realisiert.

Bin mir nicht ganz sicher ob das so schlau ist

* broadcast(): sendet an alle Clients ein Status Objekts des Spiel Objekts des Servers

Client

* inform() : Wird vom EventListener gerufen und sendet eine String Nachricht (je nach Input) an den Server sendet

Status:

* enthält Informationen welche benötigt werden um ein Spielobjekt zu aktualisieren

Zu den globalen Parametern:

Ich denke dass ich den Frame Count in die Spiel Klasse stecken werde, welche diesen dann in der step() Funktion an die anderen Objekte weiterleitet.