DOCUMENTATION : Moteur Physique

ARCHITECTURE DE NOTRE BIBLIOTHEQUE LOGICIEL MOTEUR PHYSIQUE :

Notre architecture est décomposée en trois grands modules qui sont les suivant :

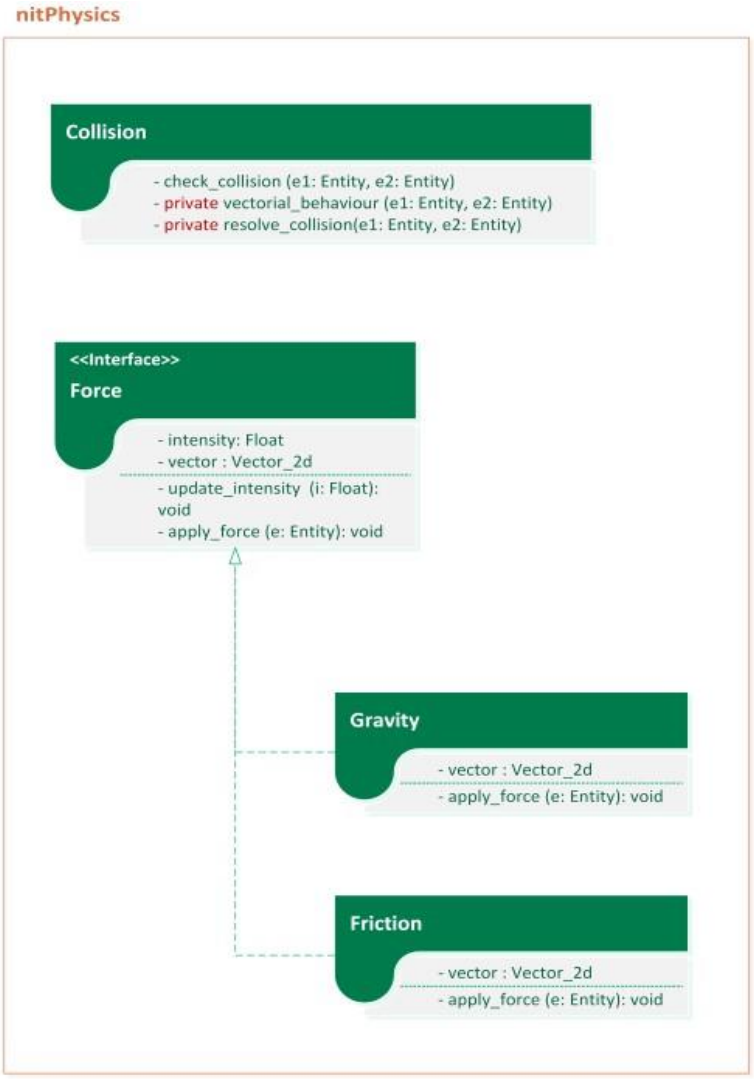
Module nitSimulation : Ce module contient les classes qui sont en charge de la représentation de notre environnement virtuel et des différentes entités qui le composes.

Module nitPhysics : Ce module est structuré par les différentes classe qui régissent les principes physiques de notre monde (détection des collisions, gestion des collisions, principe de gravité…).

Module nitConstraints : Ce module est composé de différents objets qui seront utilisées comme attributs lors des différentes calcule réalisé par notre moteur physique.

MODULE NITPHYSICS :

Le module nitPhysics est composé des classes et interfaces suivantes :



La Classe Collision

Elle contient les fonctions suivantes :

check_collision **:** elle détecte la collision entre deux entités. Elle prend en paramètres deux objets de type Entity.

vectorial_behaviour : elle calcule les vecteurs directionnels des entités après une collision. Elle prend en paramètre deux objets de type Entity.

resolve_collision : elle résoud le comportement des entités après une collision. Elle prend en paramètre deux objets de type Entity.

L'interface Force

Elle contient les attributs suivant :

intensity : variable de type Float qui correspond l'intensité d'une force. vector : objet

de type Vector_2d qui correspond au vecteurs directionnel d'une force.

Elle contient les fonctions suivantes :

update_intensity : elle met à jour l'intensité d'une force et prend en paramètre une intensité de type Float.

apply_force : elle applique une force sur une entité. Elle prend en paramètre un objet de type Entity.

La Classe Gravity

Elle contient l'attribut suivant :

vector : objet de type Vector_2d qui correspond au vecteurs directionnel de la force de gravité.

Elle contient la fonction suivante :

apply_force : elle applique une force de gravité sur une entité. Elle prend en paramètre un objet de type Entity.

La Classe Friction :

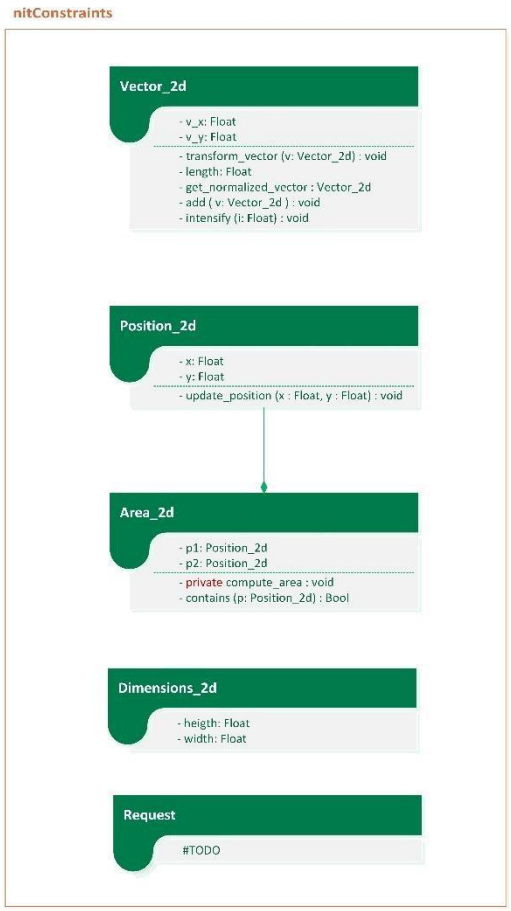
Elle contient l'attribut suivant :

vector : objet de type Vector_2d qui correspond au vecteurs directionnel de la force de frottement.

Elle contient la fonction suivante :

apply_force : elle applique une force de frottement sur une entité. Elle prend en paramètre un objet de type Entity.

Module nitConstraints :



Le module nitPhysics est composé des classes suivantes :

La Classe Vector_2d

Elle contient les attributs suivants :

v_x : vecteur directionnel de l'axe x de type Float . v_y

:vecteur directionnel de l'axe y de type Float .

Elle contient les fonctions suivantes :

transform_vector : elle applique une transformation de vecteur en fonction d'un autre. Elle prend en paramètre un objet de type Vector_2d.

length : calcule la taille d'un vecteur. Elle retourne un Float

get_normalized_vector : elle transforme un vecteur en vecteur normalisé. Elle prend en paramètre un objet de type Vector_2d. Elle retourne un objet de type Vector_2d. add : elle ajoute un nouveau vecteur. Elle prend en paramètre un objet de type Vector_2d. intensify : elle intensifie le vecteur en le multipliant par un facteur d'intensification de type Float.

La Classe Position_2d

Elle contient les attributs suivants :

x : la position x de type Float. y : la

position y de type Float. Elle

contient la fonction suivante :

update_position : elle met à jour la position. Elle prend en paramètre les coordonnées x et y de type Float.

La Classe Area_2d

Elle contient les attributs suivants :

p1 : une position 1 de type Position_2d. p2

: une positon 2 de type Position_2d.

Elle contient les fonctions suivantes :

compute_area : elle calcule l'air entre les deux positions.

contains : elle vérifie si un point est contenu dans l'air. Elle prend en paramètre un objet de type Position_2d. Elle retourne un booléen.

La Classe Dimension_2d :

Elle contient les attributs suivants :

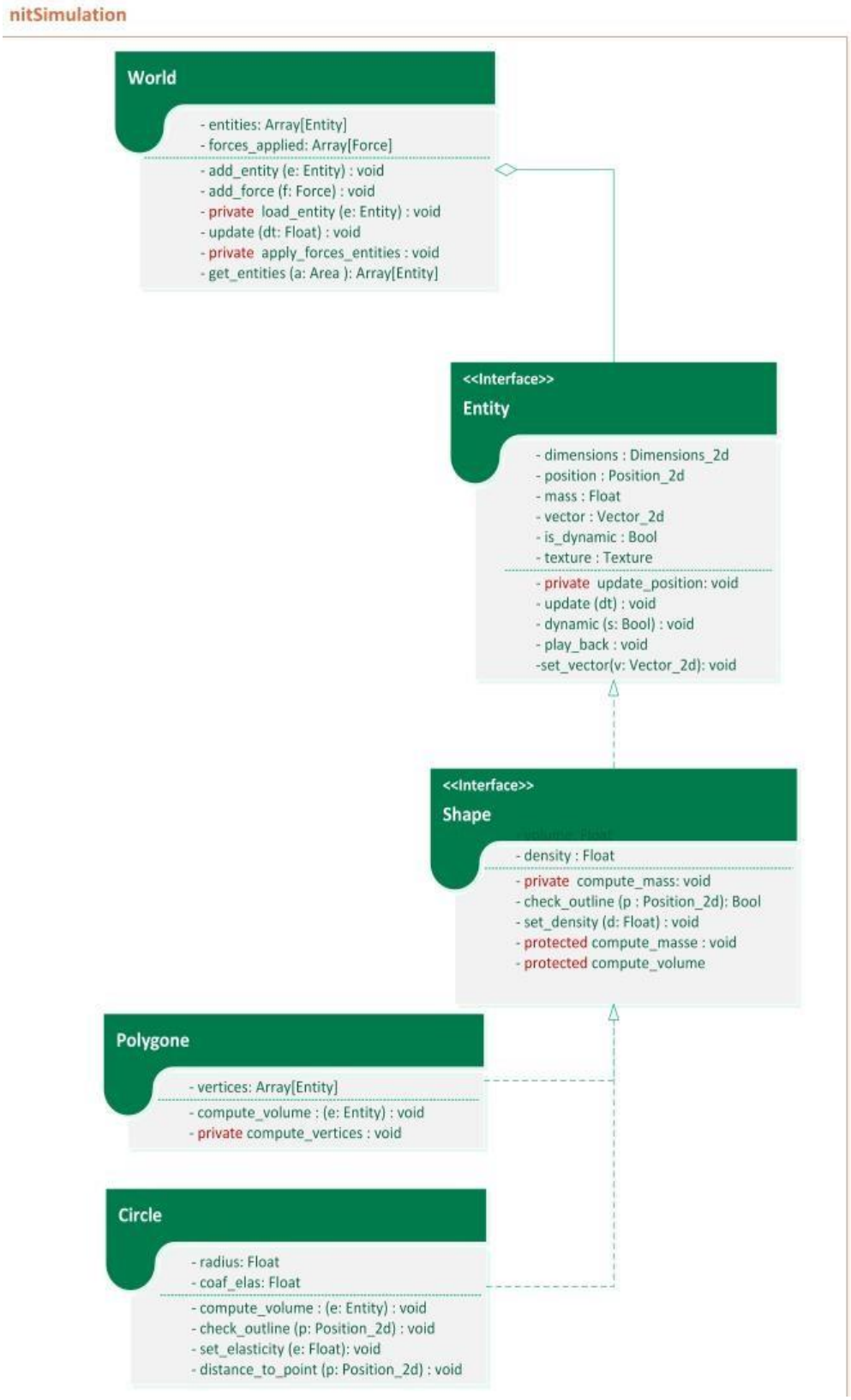
height : la hauteur de l'entité de type Float. width

: la largeur de l'entité de type Float.

La Classe Request :

Cette n'est pas encore implémentée, elle contiendra des requêtes.

Module nitSimulation :



Le module nitSimulation est composé des classes et interfaces suivantes :

La Classe World :

Elle contient les attributs suivant :

entities : tableau qui contient la liste des entités du world. Il est composé d'objets de type Entity.

forces_applied : tableau qui contient la liste des forces présentes dans le world. Il est composé d'objets de type Force.

Elle contient les fonctions suivantes :

add_entity : elle ajoute une entité au world. Elle prend en paramètre un objet de type Entity.

add_force : elle ajoute une force au world. Elle prend en paramètre un objet de type Force.

load_force : elle applique une force sur une entité. Elle prend en paramètre un objet de type Entity.

update : elle actualise le comportement du world et de ses différentes entités. Elle prend en paramètre dt (un temps) de type Float. apply_forces_entities : Elle applique les différentes force du world sur ses entités.

get_entities : elle récupère les différentes entités présentes dans un espace. Elle prend en paramètre un objet de type Area et retourne un tableau d'objet de type Entity.

L'interface Entity :

Elle contient les attributs suivants :

dimensions : dimensions de l'entité de type Dimensions_2d.

position : position de l'entité de type Position_2d. mass :

masse de l'entité de type Float.

vector : vecteur directionnel de l'entité de type Vector_2d.

is_dynamic : booléen qui indique si l'entité est mobile. texture

: texture (image) de l'entité de type Texture.

Elle contient les fonctions suivantes : update_position

: elle met à jour la positon de l'entité

update :met à jour les force appliquées sur l'entité. Elle prend en paramètre un temps dt de type Float.

dynamic : met à jours l'état de l'entité. Elle prend en paramètre un booléen.

play_back : remet l'entité au position à l'instant t-1.

set_vector : met à jour le vector directionnel de l'entité. Elle prend en paramètre un objet de type Vector_2d.

L'interface Shape :

Elle contient les attribut suivants :

volume : volume de la shape de type Float. density

: densité de la shape de type Float.

Elle contient les fonctions suivantes :

compute_mass : elle calcule la mass d'une shape compute_volume : elle calcule le

volume d'une shape. check_outline : elle vérifie si un point est en contact avec les

bordures d'une shape.

set_density : elle met à jour la densité d'une shape. Elle prend en paramètre une densité de type Float.

La Classe Polygone :

Elle contient l'attribut suivant :

vertices : un tableau qui contient les côtés du polygone de type Position_2d.

Elle contient les fonctions suivantes :

compute_volume : elle calcule le volume d'une shape. compute_vetices

: elle calcule le nombre de côtés d'un polygone.

La Classe Circle :

Elle contient les attributs suivants :

radius : le rayon du cercle de type Float.

coaf_elas : le coefficient d' élasticité du type Float.

Elle contient les fonctions suivantes :

compute_volume : elle calcule le volume de l'entité circle. check_outline : elle

vérifie si un point est en contact avec les bordures du cercle.

set_elasticity : elle met à jour l'élasticité du cercle. Elle prend en paramètre un coéfient d'élasticité de type Float.

distance_to_point : elle calcule la distance entre le cercle et un point. Elle prend en paramètre un point de type Position_2d.