Mondes Virtuels

Eric Maisel

ENIB

Automne 2019

Objectifs

- Savoir créer un monde virtuel
- Prendre en main une bibliothèque 3d

Mondes virtuels

Pour la réalité virtuelle

Réalité Virtuelle (Wikipedia)

Technologie informatique qui simule la présence physique d'un utilisateur dans un monde artificiellement généré par des logiciels.

Mondes (Larousse)

Ensemble de choses ou d'êtres formant un tout à part, organisé, un microcosme.

Virtuel

- En devenir, potentiel
- Par opposition à actuel.

Mondes virtuels

Mondes virtuels (Wikipedia)

Monde créé artificiellement par logiciel, pouvant héberger une communauté d'utilisateurs sous forme d'avatars ayant la capacité de s'y **déplacer** et d'y **interagir**.

La **représentation** de ce monde est 2D ou **3D**

Un monde virtuel peut **simuler** le monde réel avec ses **lois physiques** ou tout au contraire être régie par d'autres. Les **lois humaines** peuvent également être reproduites.

3D et Web

Pourquoi ?

- Facilité d'installation
- Base pour RV distribuée

Avec quoi ?

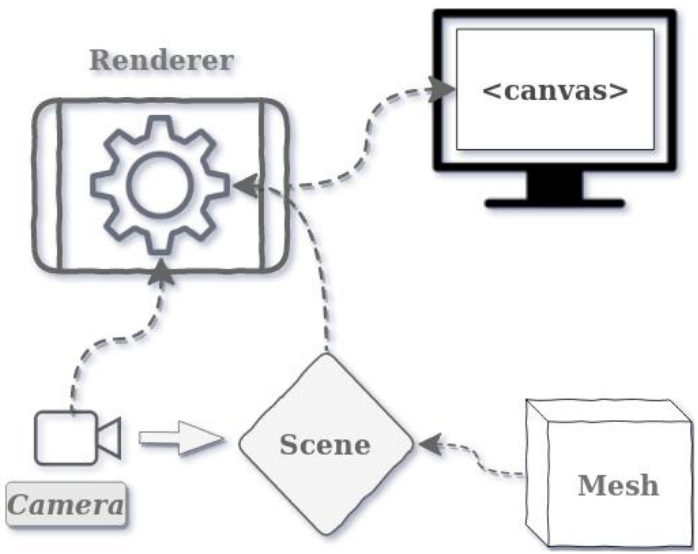
- WebGL
- Three.js, Babylonjs
- AFrame

Vers la RV

- WebVR

Un peu de 3D

Une base : le rendu visuel

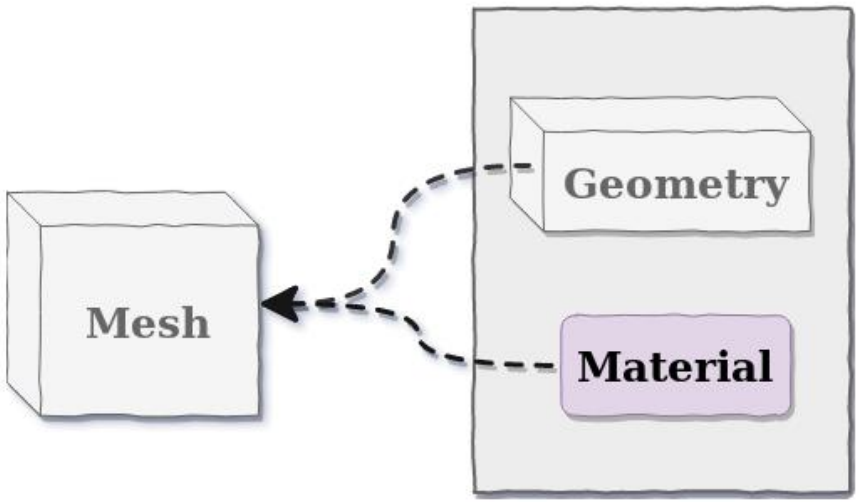


- Canvas : zône où placer l'image dans le navigateur
- Renderer : procédure de calcul des images
- Scene : collection d'objets 3d
- Camera : point de vue d'où sont calculées les images

Un peu de 3D

Une base : le rendu visuel

- Scene = collection d'objets 3D
- Objet 3D =
  - Géométrie
  - Matériau
  - Pose (position, orientation)



Un exemple simple

Un monde virtuel

```
var scene, camera, renderer, cube ;

function initThree(){
  scene = new THREE.Scene() ;

  var l, h = window.innerWidth, window.innerHeight ;
  camera = new PerspectiveCamera(0.50,l/h,0.1,1000.0) ;
  camera.position.set(2,1.7,5) ;
  camera.lookAt(new THREE.Vector3(0,1,0)) ;

  renderer = new THREE.WebGLRenderer() ;
  renderer.setSize(l,h) ;
  document.body.appendChild(renderer.domElement) ;
}
```

Un exemple simple

Un monde virtuel

```
function initScene(){

  // Ajout d'une source lumineuse
  var light = new THREE.HemisphereLight(0xffffff,0xff00ff,1);
  light.position.set(-2,5,5);
  scene.add(light) ;

  // Ajout d'un cube
  var geo = new THREE.BoxGeometry(1,1,1) ;
  var mat = new THREE.MeshStandardMaterial({color:0xee3333});
  cube = new THREE.Mesh(geo, mat) ;
  scene.add(cube) ;
}
```

Un exemple simple

Un monde virtuel

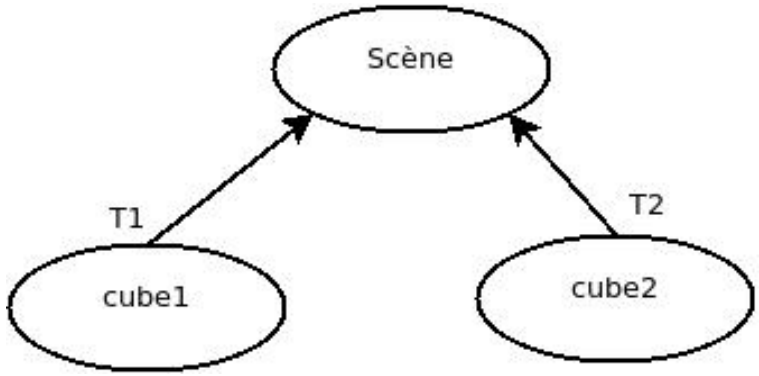
```
function animer(){
  requestAnimationFrame(animer) ;

  cube.rotation.y += 0.0001 ;
  cube.rotation.x += 0.0005 ;

  renderer.render(scene, camera) ;

}
```

Un exemple simple

Un monde virtuel

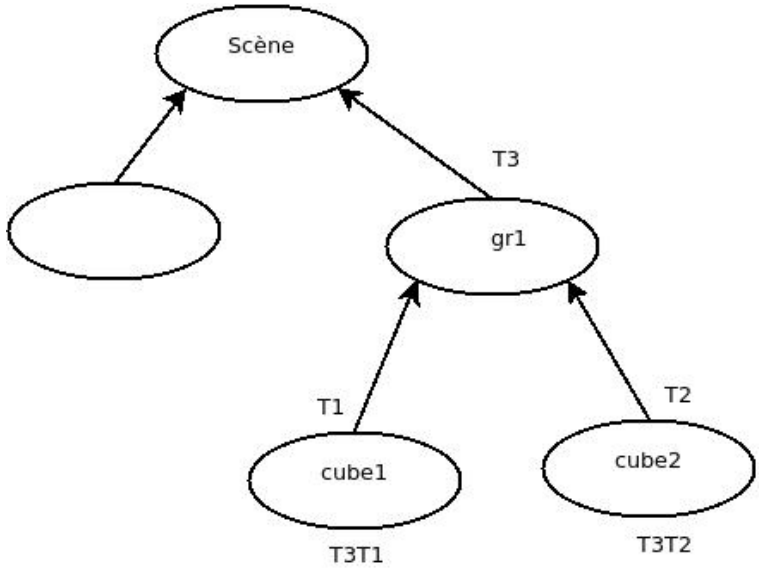
```
function main(){
    initThree() ;
    initScene() ;
    animer() ;
}

main() ;
```

Graphe de scène



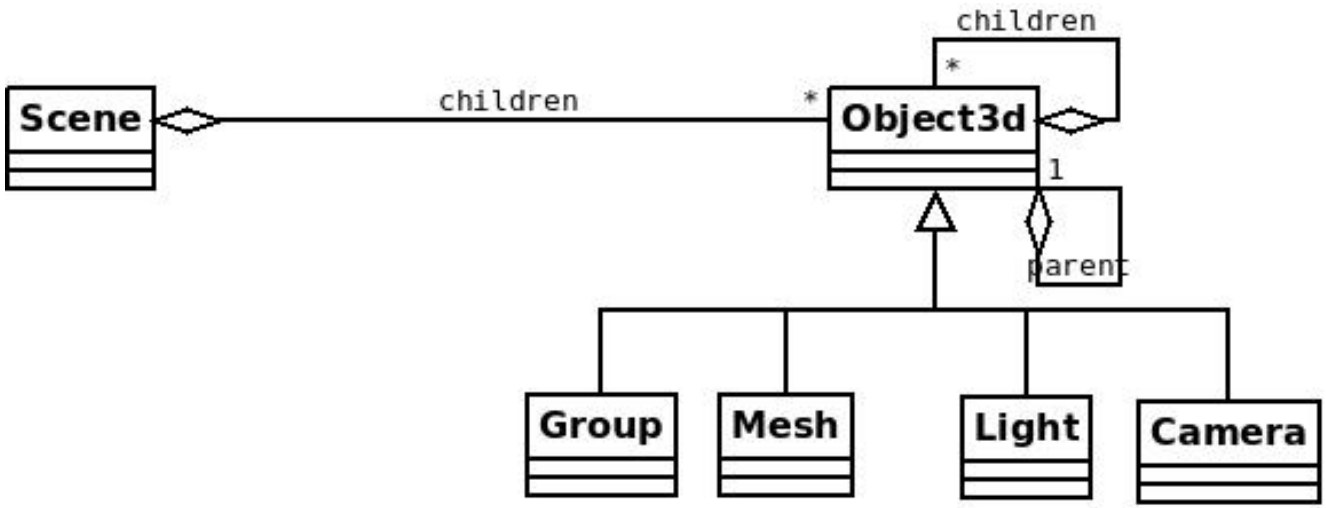

```
var cube1 = creerCube({taille:1,couleur:0xff0000}) ;
cube1.position.set(0,0.5,0) ;
cube1.rotation.set(0,Math.PI/6.0,0.0) ;
scene.add(cube1) ;
// ====================================
var cube2 = creerCube({taille:1,couleur:0x0000ff}) ;
cube2.position.set(2,0.5,0) ;
scene.add(cube2) ;
```

Graphe de scène




```
var cube1 = creerCube({taille:1,couleur:0xff0000}) ;
var cube2 = creerCube({taille:1,couleur:0x0000ff}) ;
var gr = new THREE.Group() ;
scene.add(gr) ;
gr.add(cube1) ; // transfo : T1T3
gr.add(cube2) ; // Transfo : T2T3
```

Graphe de scène



Méthodes de Object3d

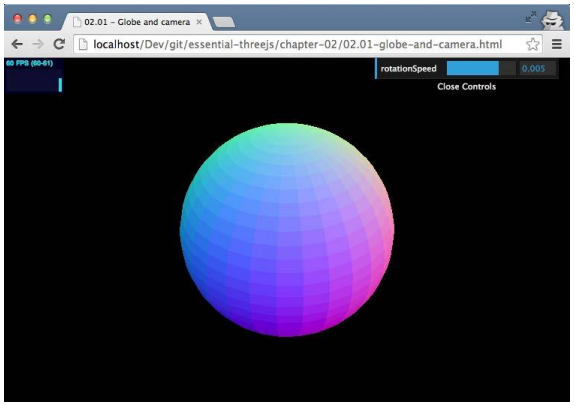
- .add(fils)

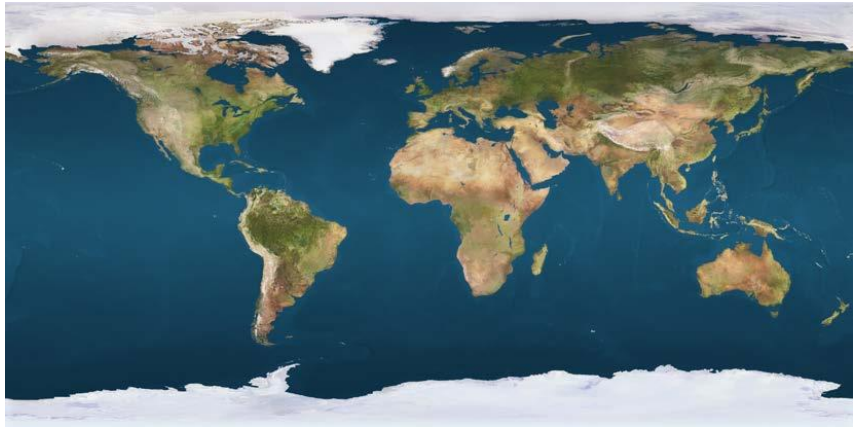
Propriétés de Object3d

- isVisible
- position
- rotation

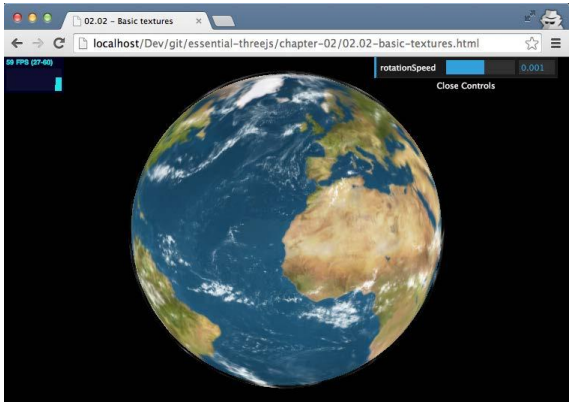
Représenter les détails

Textures





Placage de texture



Représenter les détails

Textures

```
var textureLoader = new THREE.TextureLoader() ;
var texture = textureLoader.load("terre.jpg") ;
var mat = new THREE.MeshStandardMaterial({
                                    color:0xffffff,
                                    map:texture
                                }) ;
var geo = new THREE.BoxGeometry(1,1,1) ;
var cube = new THREE.Mesh(geo,mat) ;
scene.add(cube) ;
```

Représenter les détails

Images avec transparences

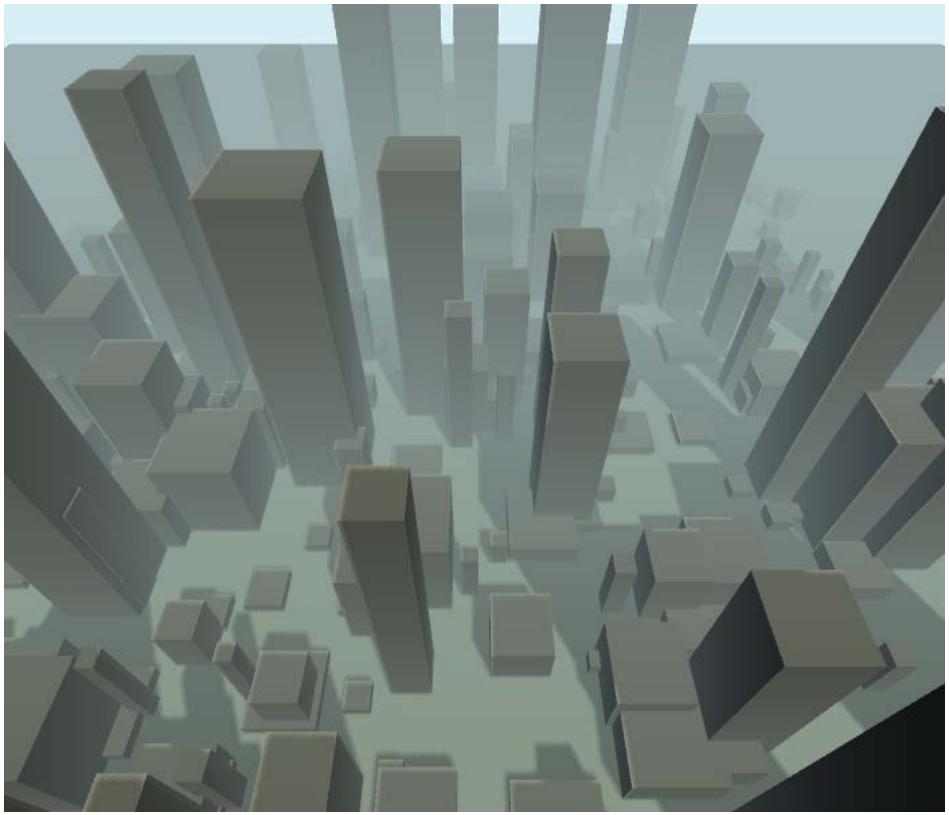
Placage de textures avec transparence

Représenter les détails

Images avec transparences

```
const material = {
  color,
  map : loader.load(url),
  opacity : 0.5,
  transparent : true,
  side : THREE.DoubleSide
}
```

Monde complexe



- Placements de plusieurs objets
- Effets lumineux : brouillard, ombres portées

Monde complexe

Couches rectangulaires

- Espace <-> Structure tabulaire (tableau, image, chaînes)
- Nature et placement des objets spécifiés par les éléments de la structure

Génération procédurale

Utilisation d'algorithmes pour placer des objets de façon semi-aléatoire

Editeur

- Utilisation d'outils externes
- Fichier exporté (ex/ en JSON)

Monde complexe

Brouillard

Brouillard exponentiel

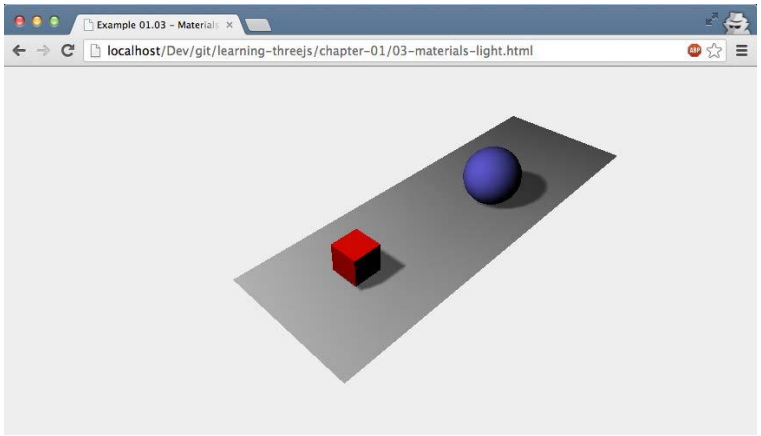
```
scene.fog = new THREE.Fog2(0x9db3b5,0.002) ;
```

Brouillard linéaire

```
scene.fog = new THREE.Fog(0x9db3b5,0,200) ;
```

Monde complexe

Ombres portées



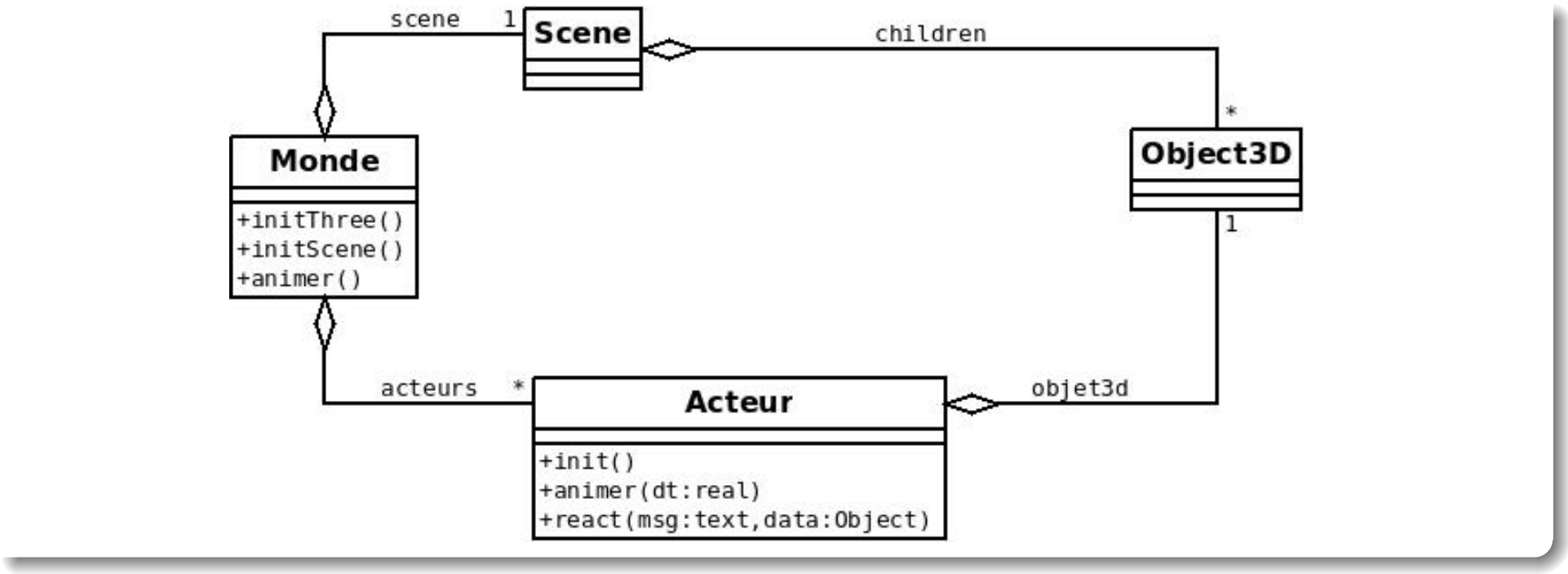

```
renderer.shadowMapEnabled = true ;
```

```
light.castShadow    = true ;
cube.castShadow     = true ;
sphere.castShadow   = true ;
plane.receiveShadow = true ;
```

Animation

```
var chrono = new THREE.Clock() ;
chrono.start() ;

function animer(){
  var dt = chrono.getDelta() ;
  temps += dt ;

  cube.rotation.y += dt ;
  cube.rotation.x += 2*dt ;

  renderer.render(scene, camera) ;

  requestAnimationFrame(animer) ;
}
```

Architecture pour l'animation



Animation

Architecture : 1ere forme

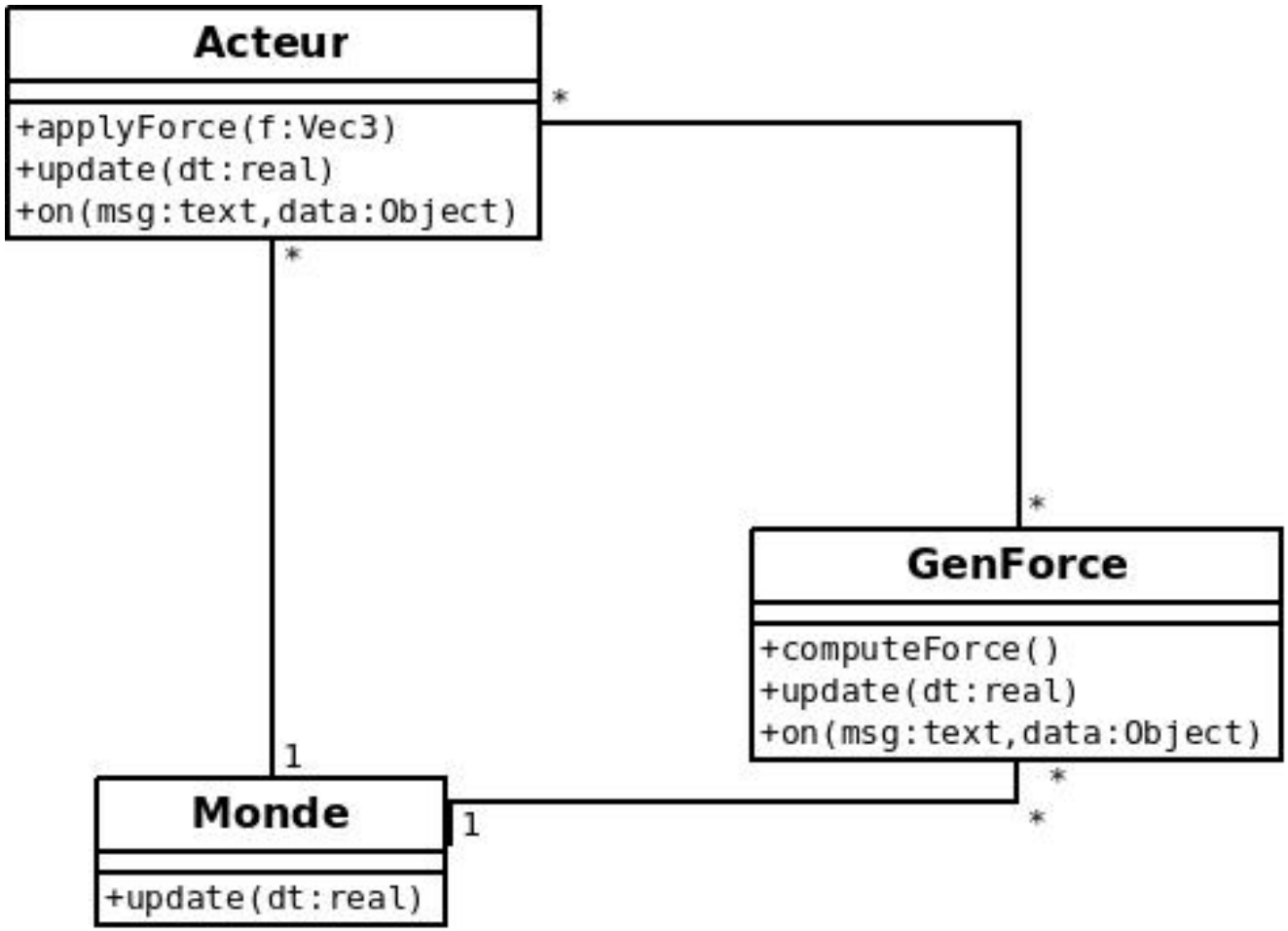
- Acteur : controleur d'objets 3d
- Interface : `update(dt)` appelé régulièrement

Lois de Newton

Schéma d'intégration

$$\begin{cases} P(t+dt) & = & P(t) + V(t)dt \\ V(t+dt) & = & V(t) + \gamma(t)dt \\ \gamma(t) & = & \frac{1}{m}\sum F_i \end{cases}$$

Dynamique pour l'animation



Dynamique pour l'animation

```
Monde.prototype.update = function(dt){
  var i ;
  for(i=0; i<this.genForces; i++)
    this.genforces[i].update(dt) ;
  for(i=0; i<this.acteurs; i++)
    this.acteurs[i].update(dt) ;
  this.acteurs[i].force.set(0,0,0) ;
}
```

```
GenForce.prototype.update = function(dt){
  var f = this.computeForce() ;
  for(var i=0; i<this.acteurs;length;i++)
    this.acteurs[i].force.accumuler(f) ;
}
```

Animation

Animation contrôlée par steering

Principe

- Objectif $\rightarrow V_d$
- $F = V_d - V_c$

Passer par un point

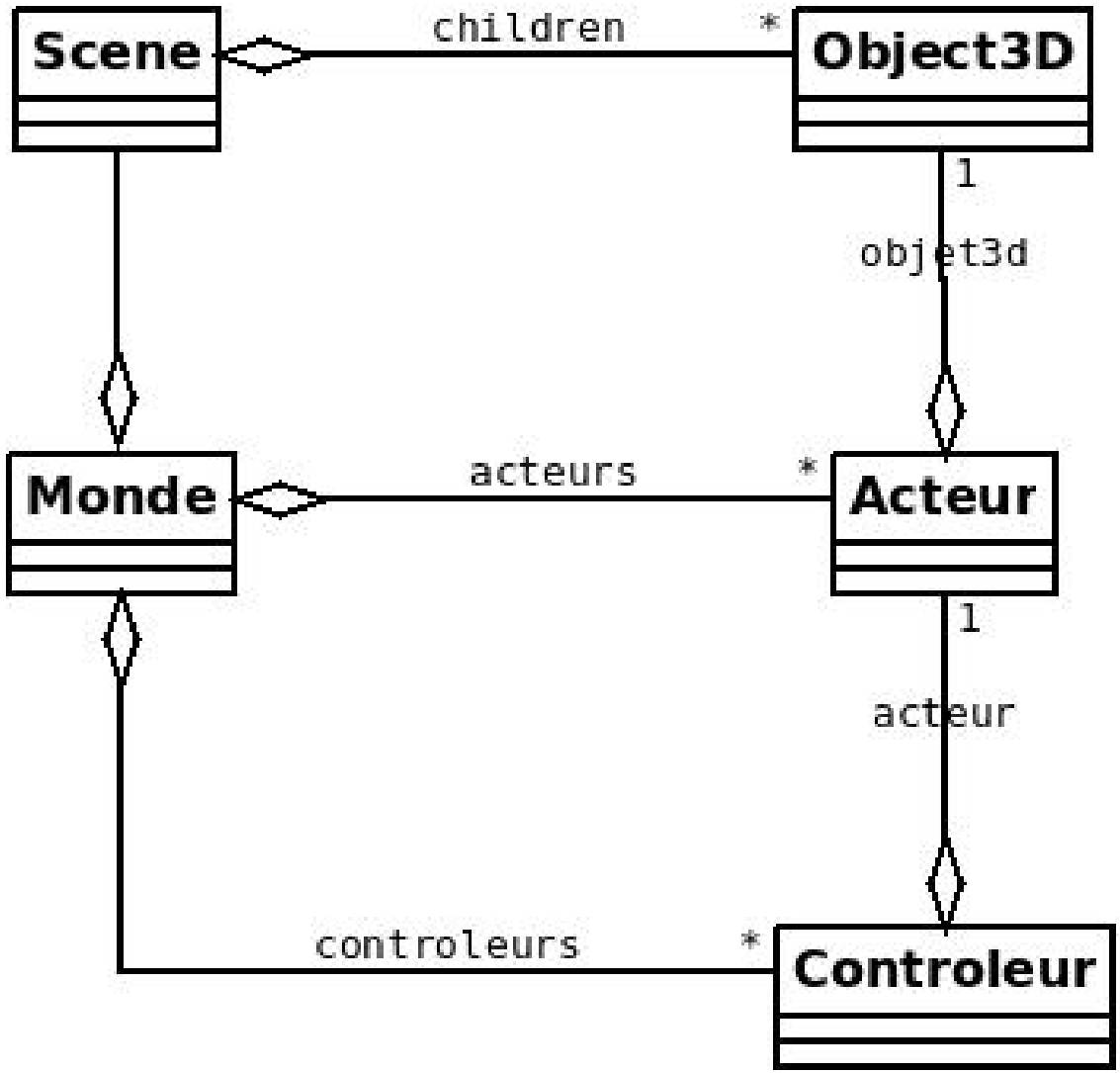
$$V_d = V_m \frac{PC}{||PC||}$$

Autres objectifs

s'arréter en un point, suivre une trajectoire, suivre un mur, éviter un objet, ...

Animation

Animation contrôlée par steering



Animation

Animation conditionnée par sélection

Principe

- Création d'un rayon
- Calcul d'intersection rayon/objets

```
document.addEventListener("click",onMouseClick,false) ;
```

Animation

Animation conditionnée par sélection

Implémentation

```
var raycaster           = new THREE.Raycaster() ;
var mouse               = new THREE.Vector2() ;
var selectedObject  = null ;

function onMouseClick(e){
  mouse.x =  (e.clientX/window.innerWidth)*2-1;
  mouse.y = -(e.clientY/window.innerHeight)*2+1;
  raycaster.setFromCamera(mouse,camera) ;
  var intersectedObjects =
            raycaster.intersectObjects(scene.children) ;
  if(intersectedObjects.length > 0){
    selectedObject = intersectedObjects[0] ;
  }
}
```

Animation

Animation conditionnée par trigger

- Mise en oeuvre 3d d'une structure conditionnelle
- Motif Observateur/Observé : notification d'evts à des observateurs abonnés
- Test sur la situation dans une région de l'espace
  - On objet est présent dans la région
  - Un objet entre dans la région
  - Un objet sort de la région

Animation

Animation conditionnée par trigger