Petit Prince

SAPIENZA
UNIVERSITÀ DI ROMA

Interactive Graphics 2017/18

Chiara Mele - Alessia Palleschi

CONTENTS

PROJECT'S IDEA

The "**Petit Prince**" is inspired by the book of the same name written by Antoine de Saint-Exupéry.

The game is set on the asteroid B612, that is the planet of the petit prince, constituted by baobab trees and little volcanos. The **fox** runs to collect the ***gold coins*** (+1), without taking the ***red coins*** (-1) and without taking the cone that represents the ***little volcanos***.

CONTENTS

- ☆ Project’s idea
- ★ **Environment**
- ☆ Technical aspects
- ☆ Components
- ☆ User manual

ENVIRONMENT

The environment used for the development of the game is three.js

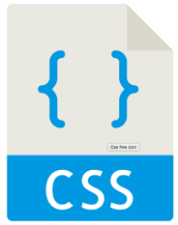
with the addition of the following libraries:

- **OrbitControls.js**: for the position of the camera
- **OBJLoader.js**: for the loading of the 3d model
- **MTLLoader.js**: for loading the texture of the 3d model
- **LoaderSupport.js**: for the loading of the resources
- **CanvasRenderer.js**: for displaying the scene and creating the render

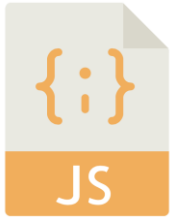
CONTENTS

TECHNICAL ASPECTS

The HTML file is the container of all the code and describes the structure of the page. In addition, it contains the connection with the js file and libraries and the css part.



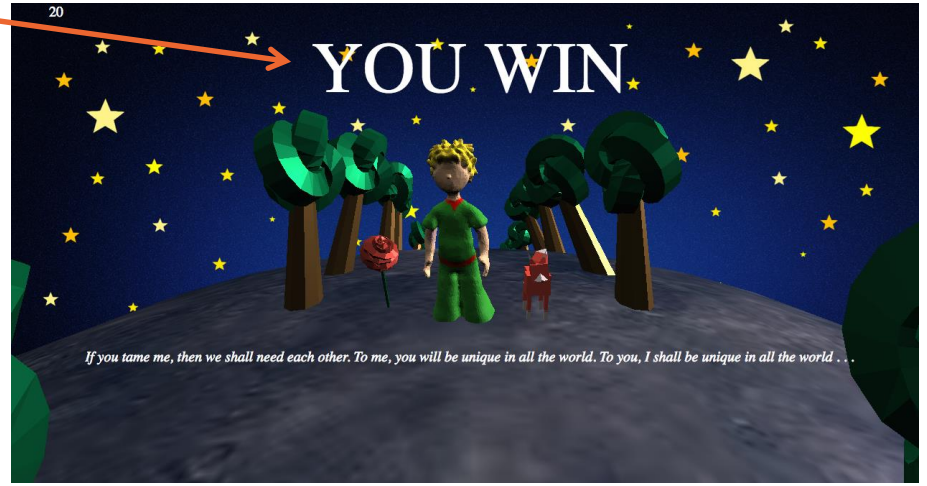
The CSS part defines the style of the components into the HTML page.



The JS file is the main file that describes the entire project, managing the objects into the scene and the animations.

TECHNICAL ASPECTS

```
<div id="world">
    <div id="score">
        <button onclick="setup()">START</button>
    </div>
</div>
<div id="gameover">
</div>
```



- **world** is where the canvas for the scene is attached.
- **score** displays the score during the game
- **gameover** is the div for the result of the game.

TECHNICAL ASPECTS

The javascript file can be divided into two different parts:

1) Deals with the positioning and the creation of the objects

```
Init
   createScene
      insertElement
         createVolcano
         createBad
         createCoin
      insertPrinceAndRose
         loadPrince
         createRose
      addB612
      addLight
      addExplosionCoin
      addExplosionBad
      onWindowResize
   createFox
```

2) Deals with the objects' animation within the scene

```
   loop
      animFox
      updateFox
      update
         addInPath
            addObject
               createBaobab
         checkCollision
            explode
         explosionCoin
         explosionBad
      requestAnimationFramesetup
   insertBaobab
      addObject
```

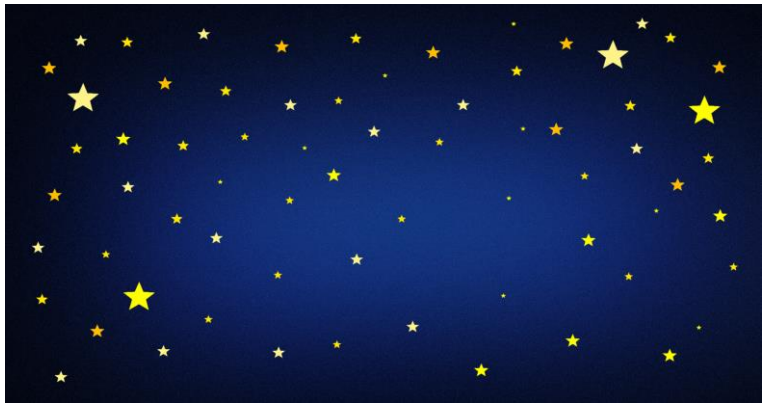
TECHNICAL ASPECTS

- `createScene()` function creates the scene to be displayed: starts the clock, initializes the renderer and assigns the positions to the camera.

The scene is mainly composed of the **planet**, a sphere on top of which **baobab trees**, **coins** and **volcanoes** are placed through the use of the sphericalhelper function.

The baobabs are positioned in such a way as to delimit a path in which the fox can be moved to the right and to the left.

Moreover, coins and volcanoes are placed in specific locations by utilizing an appropriate function



Background of the sky

The planet seen from the user position

Default screen

TECHNICAL ASPECTS

Using the right and the left arrow the fox can be moved from one lane into another one

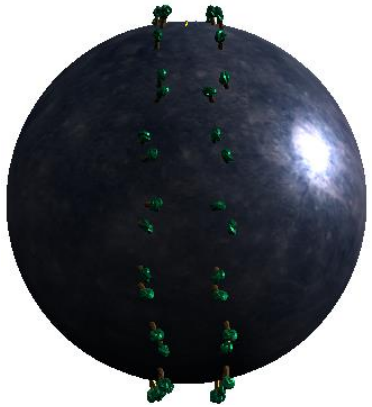
Left lane

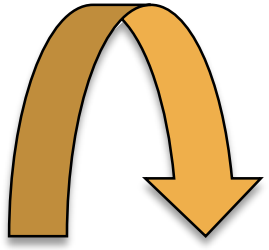
Middle lane

Right lane

TECHNICAL ASPECTS

- **Rotation of the planet**





```
b612.rotation.x += planetSpeed;
```

The **rotation of B612** is the instrument through which the game platform is advanced: as long as the game continues, the rotation is incremented in the *update* method via *planetSpeed*, a numeric variable declared a priori.

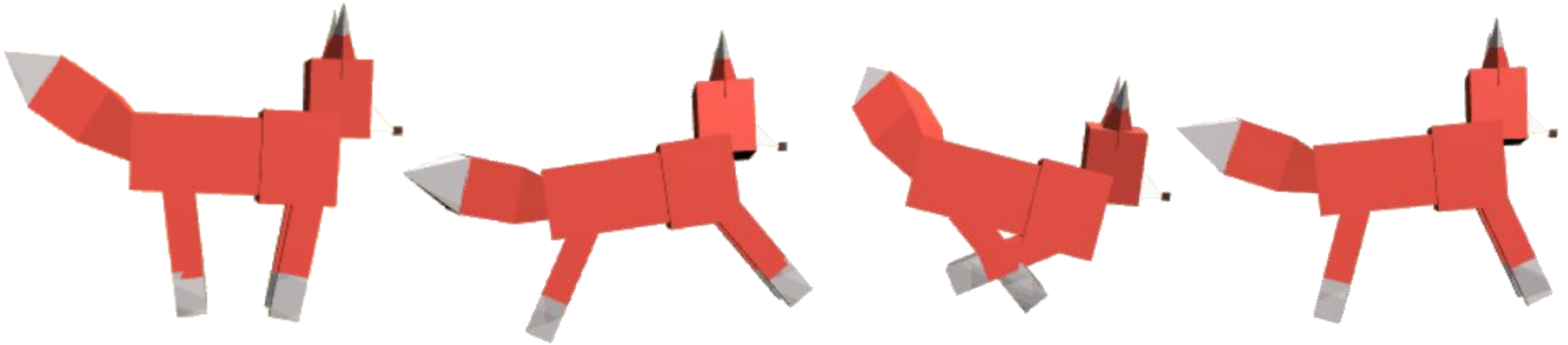
In addition of the planet's rotation, at every time new elements are added into the path. These elements disappear when they are out of the camera's cone vision.

TECHNICAL ASPECTS

Movement of the fox

```
function animFox(object, duration, offset){
    object.mesh.rotation.z
    object.tail.rotation.z .x .y
    object.head.rotation.z .x
    object.legFR.rotation.z
    object.legBR.rotation.z
    object.legFL.rotation.z
    object.legBL.rotation.z
}
```

The **`animFox`** method manages the animation of the fox: for each component of the mesh representing the animal, a rotation is computed with the product between the sinusoidal computed on the date, a fixed duration and offset, the PI for a fixed amount.



TECHNICAL ASPECTS

Collision with objects

The **`checkCollision`** function manages the collisions: while the game is running and is not finished, it checks the distance between the object and the fox; if it happens between the fox and a gold coin, the score is incremented, if happens with a red coin, the score is decremented. The object is then removed from the scene, even if no collision has occurred.

Explosion of a gold coin

Explosion of a gold coin and a red one

The volcanoe in front of the fox

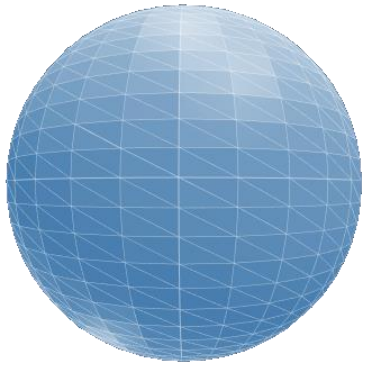
CONTENTS

- ☆ Project’s idea
- ☆ Environment
- ☆ Technical aspects
- ★ **Components**
- ☆ Interactions

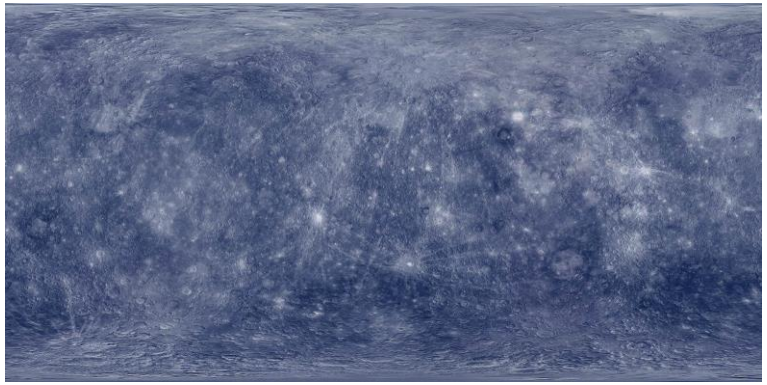
COMPONENTS

1. PLANET

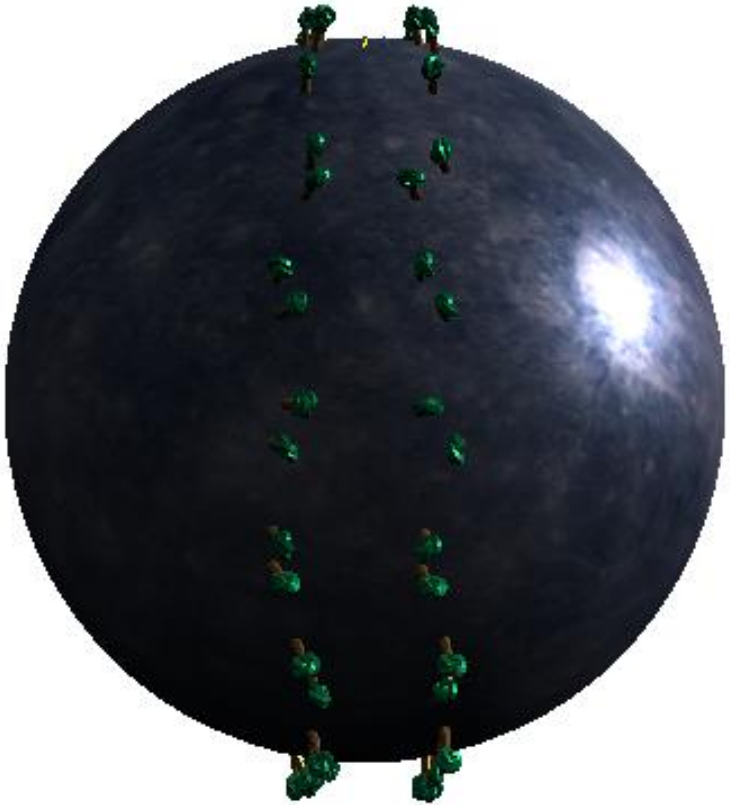
The planet is composed of a sphere, to which a texture similar to B612's color is applied.



+



=

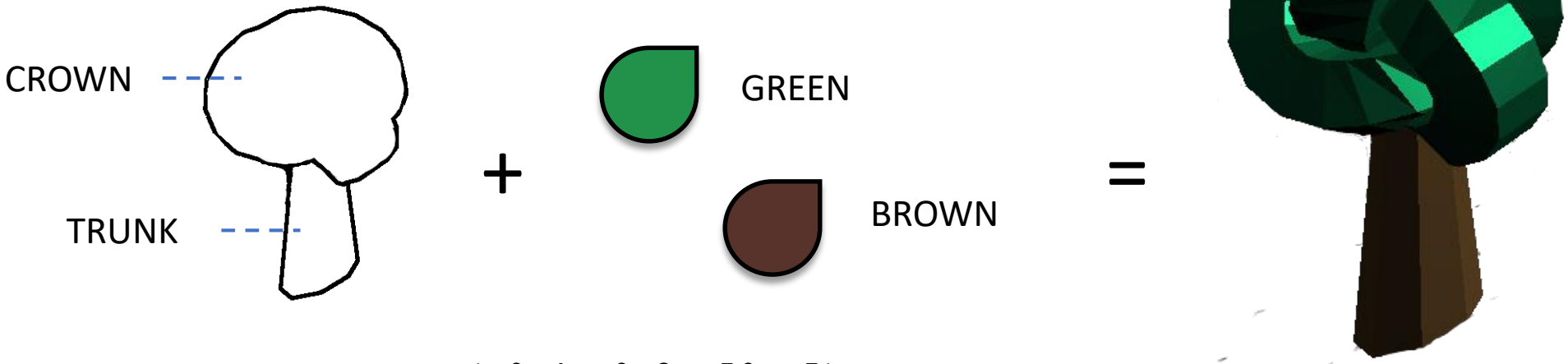


```
THREE.SphereGeometry(26, 40, 40);
THREE.MeshStandardMaterial();
THREE.TextureLoader().load('https://i.imgur.com/A9dmI0U.jpg');
```

COMPONENTS

2. BAOBAB

The baobab, that is the tree on B-612, is formed by two different pieces: the trunk, that is a cylinder, and the top, that is a torus. In this case we haven't applied any texture, but we have only colored the material of the two parts.



CROWN

```
THREE.TorusKnotGeometry( 0.4, 0.3, 50, 5);
THREE.MeshStandardMaterial({color:Colors.green,flatShading:THREE.FlatShading });
THREE.Mesh( baobabGeometry, baobabMaterial);
```

TRUNK

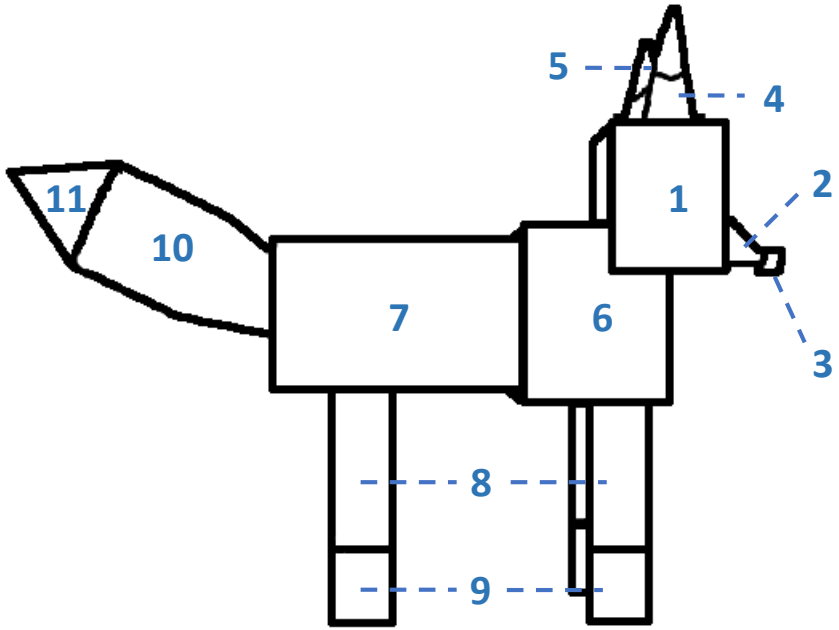
```
THREE.CylinderGeometry( 0.18, 0.45,4);
THREE.MeshStandardMaterial({ color: 0x886633,flatShading:THREE.FlatShading });
THREE.Mesh( baobabTrunkGeometry, trunkMaterial);
```

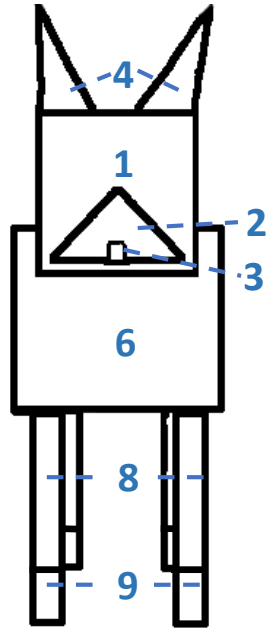
COMPONENTS

3. FOX

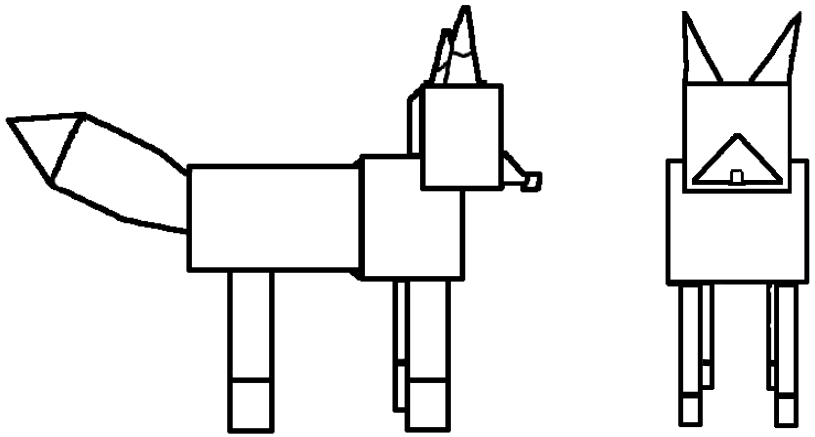
The fox is a hierarchical model composed of many elements:

1. HEAD
    2. SNOUT
    3. NOSE
    4. EARS
        5. EAR TIPS
6. CHEST
7. BODY
8. LEGS
    9. FEET
10. TAIL
    11. TAIL TIP





COMPONENTS



SKELETON

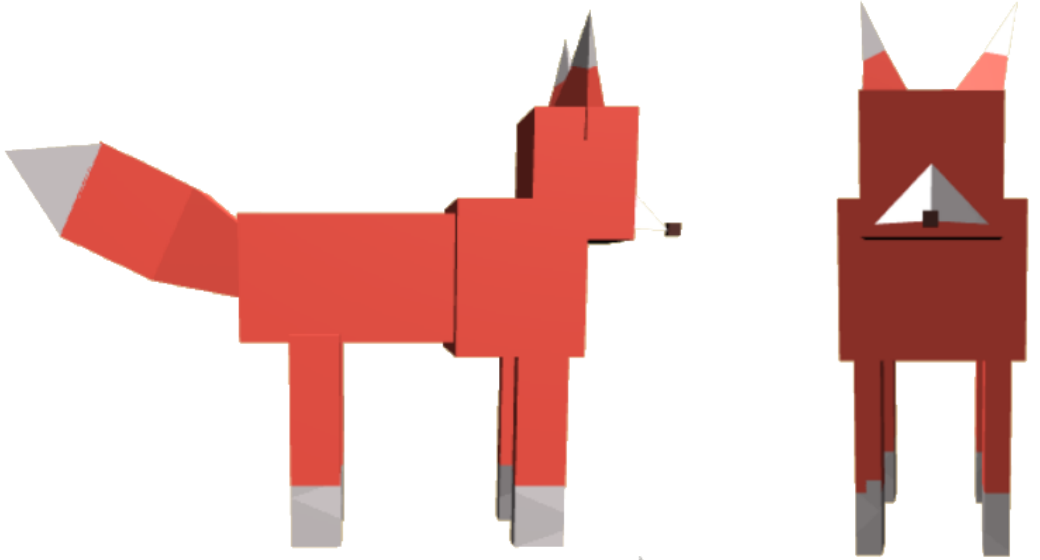
+

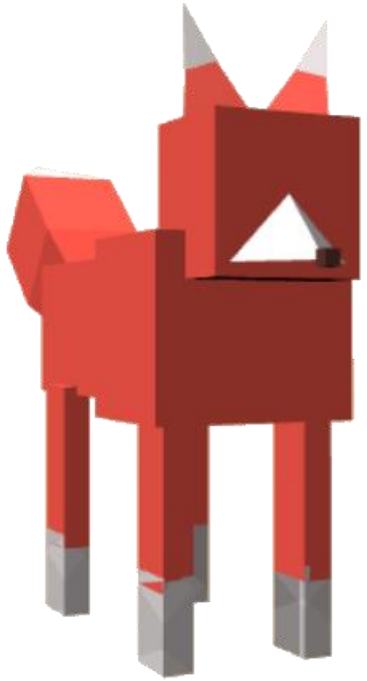
RED

GREY

BROWN

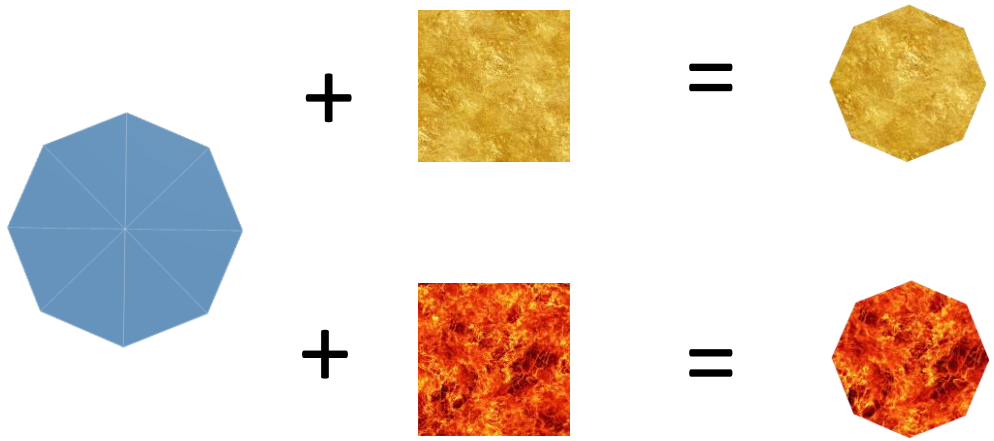
=





4. COINS AND OBSTACLES

The coins are cylinders with a small height. To differ them, we have applied two different textures.

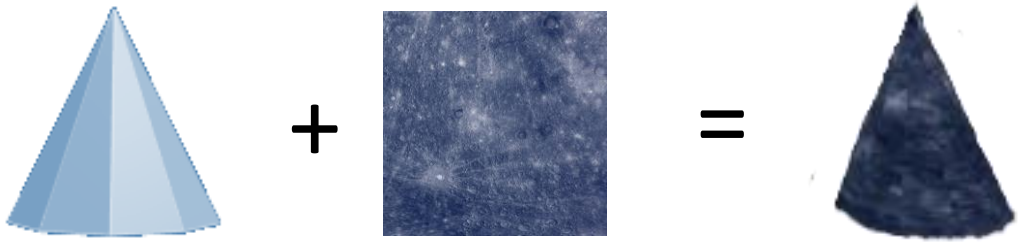


Gold coin +1

```
THREE.CylinderGeometry( 0.3, 0.3, 0.05, 8);
THREE.TextureLoader().load('goldcoin.png');
```

Red coin -1

```
THREE.CylinderGeometry( 0.3, 0.3, 0.05, 8);
THREE.TextureLoader().load('redcoin.png');
```



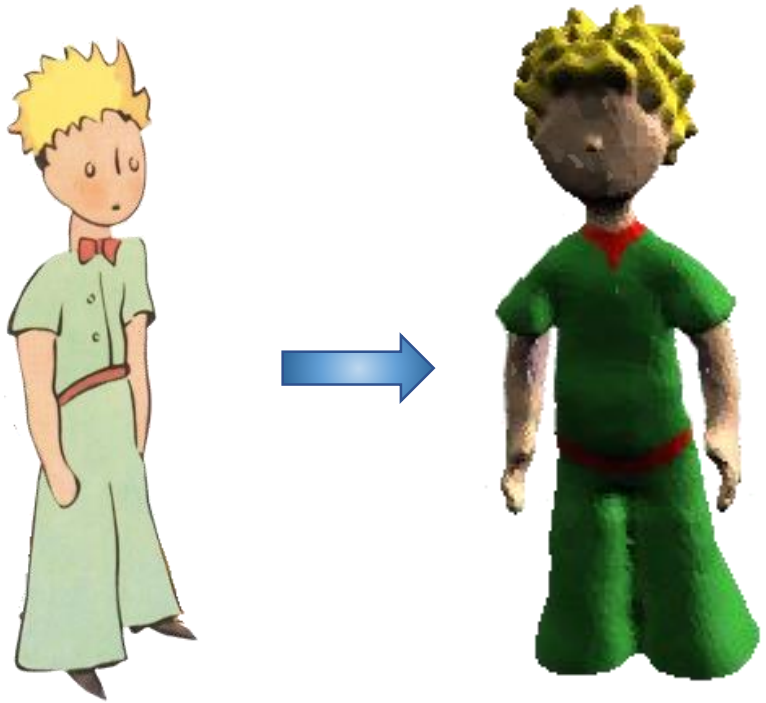
Volcano

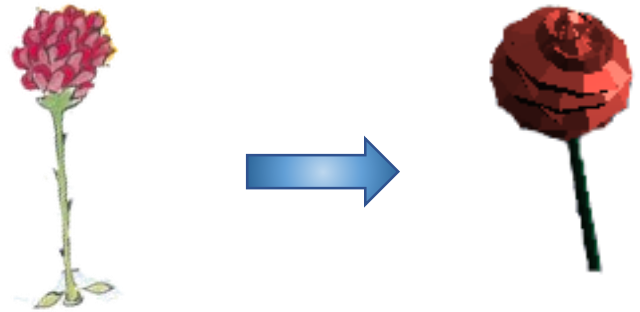
```
THREE.ConeGeometry( 0.5, 1.2, 10 );
THREE.TextureLoader().load('vulcano.png');
```

COMPONENTS

5. PRINCE AND ROSE

The **prince** has been downloaded from the web and modified for our needs.
The **rose** is a new object constituted of a torus as petals and a cylinder as stem.





FLOWER TOP

```
THREE.TorusKnotGeometry(0.05,0.5,50,4,5,3);
```

STEM

```
THREE.CylinderGeometry( 0.05, 0.05,2);
```

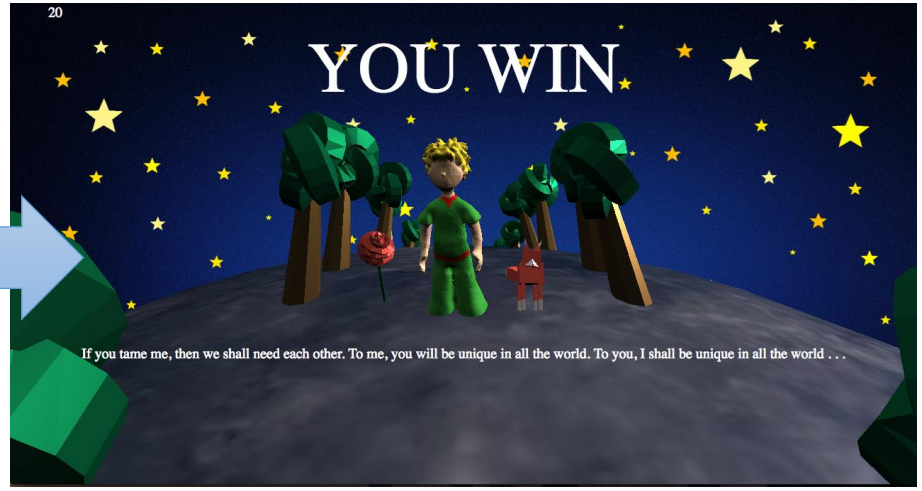
CONTENTS

- ☆ Project’s idea
- ☆ Environment
- ☆ Technical aspects
- ☆ Components
- ★ **User manual**

USER MANUAL

- In this sequence, he user hits the cone of the volcano and loses the game

- In this sequence, the user collects the coins and avoids the volcanoes. When he/she collects 20 coins, the petit prince and the rose appear.



"And now here is my secret, a very simple secret:

It is only with the heart that one can see rightly; what is essential is invisible to the eye."

"What is essential is invisible to the eye," the little prince repeated, so that he would be sure to remember.

"***It is the time you have wasted for your rose that makes your rose so important***."

Thank you
for your attention