**Getting Started - Chapter 3 - Wheel Animation**

**@@启程-第三章-轮子的动画@@**

**Foundation of Animation**

**@@动画的基础@@**

We will start with a wheel and rotate it about its axle. Remember that in order to have the car the upright we rotated it about the x axis and so the axle is along the y axis of the cylinder.

@@我们将从一个轮子开始，绕着它的转轴旋转它。不要忘了为了让小车立正，我们将小车网格绕x轴旋转了一下，因此轮子的转轴其实是沿着圆柱体的y轴的。@@

We need to create a new animation object

@@我们需要建立一个新的动画对象。@@

const animWheel = new BABYLON.Animation("wheelAnimation", "rotation.y", 30, BABYLON.Animation.ANIMATIONTYPE\_FLOAT, BABYLON.Animation.ANIMATIONLOOPMODE\_CYCLE);

which has the parameters - name, property to animate, animation frames per second, property to animate type, loop mode, in this case repeat animation.

@@这个对象的构造方法具有以下参数——名字、需要通过动画改变的属性、每秒钟的动画帧数、动画改变的属性的数据类型、循环模式——这个例子里是重复动画。@@

We follow this with the key frame array where we set values for the property to animate by frame number

@@接下来我们处理关键帧数组，在关键帧数组中我们按照帧数设置动画属性的值。@@

const wheelKeys = [];

//At the animation key 0, the value of rotation.y is 0 在动画帧0，rotation.y的值为0

wheelKeys.push({

frame: 0,

value: 0

});

//At the animation key 30, (after 1 sec since animation fps = 30) the value of rotation.y is 2PI for a complete rotation 在动画帧30，（因为动画帧率是30，所以是1秒之后）rotation.y的值为2PI，也就是转了一整圈

wheelKeys.push({

frame: 30,

value: 2 \* Math.PI

});

Finally we link the key frame array to the animation, the animation to the mesh and begin it.

@@最后我们把关键帧数组与动画对象关联在一起，再把动画对象和网格关联在一起，然后启动动画。@@

//set the keys 设置关键帧

animWheel.setKeys(wheelKeys);

//Link this animation to the right back wheel 把动画对象和右后轮关联起来

wheelRB.animations = []; 先清空网格的动画数组

wheelRB.animations.push(animWheel); 推入一个动画，当然也可以推入多个动画

//Begin animation - object to animate, first frame, last frame and loop if true 启动动画——参数是要启动动画的网格（事实上不只是网格，相机、光照、精灵、自定义对象都可以），第一帧，结束帧，是否循环。

scene.beginAnimation(wheelRB, 0, 30, true);

Simple Wheel Animation

简单轮子动画

https://playground.babylonjs.com/#KDPAQ9#14

Since all the wheels rotate the same we can use the same animation for all.

@@因为所有的轮子使用相同的旋转方法，我们可以对所有轮子使用同一个动画对象。@@

scene.beginAnimation(wheelRF, 0, 30, true);

scene.beginAnimation(wheelLB, 0, 30, true);

scene.beginAnimation(wheelLF, 0, 30, true);

To ensure that in later playgrounds new code is not overwhelmed by a large amount of previous coding we will save the car as a model and import and animate it as an item.

@@为了确保在之后的训练场中，新的代码不被大量的旧代码淹没，我们将把小车保存为一个模型，然后把它导入，再然后把它作为一个对象启动动画。@@

BABYLON.SceneLoader.ImportMeshAsync("", "url to model car", "car.babylon").then(() => {

const wheelRB = scene.getMeshByName("wheelRB");

const wheelRF = scene.getMeshByName("wheelRF");

const wheelLB = scene.getMeshByName("wheelLB");

const wheelLF = scene.getMeshByName("wheelLF");

scene.beginAnimation(wheelRB, 0, 30, true);

scene.beginAnimation(wheelRF, 0, 30, true);

scene.beginAnimation(wheelLB, 0, 30, true);

scene.beginAnimation(wheelLF, 0, 30, true);

});

Animating All 4 Wheels

启动四个轮子的动画

https://playground.babylonjs.com/#KDPAQ9#15

We can now animate the car itself and add it into the village

@@现在我们可以为整个小车添加动画，然后把它添加到村庄场景中。@@

（译者注）

导出小车模型的训练场代码：

const createScene =  () => {

    const scene = new BABYLON.Scene(engine);

    const camera = new BABYLON.ArcRotateCamera("camera", -Math.PI / 2, Math.PI / 2.5, 3, new BABYLON.Vector3(0, 0, 0));

    camera.attachControl(canvas, true);

    const light = new BABYLON.HemisphericLight("light", new BABYLON.Vector3(0, 1, 0));

    const car = buildCar();

    car.rotation.x = -Math.PI / 2;

    const wheelRB = scene.getMeshByName("wheelRB");

      const wheelRF = scene.getMeshByName("wheelRF");

      const wheelLB = scene.getMeshByName("wheelLB");

      const wheelLF = scene.getMeshByName("wheelLF");

      console.log(wheelRB.animations)

      scene.beginAnimation(wheelRB, 0, 30, true);

      scene.beginAnimation(wheelRF, 0, 30, true);

      scene.beginAnimation(wheelLB, 0, 30, true);

      scene.beginAnimation(wheelLF, 0, 30, true);

    return scene;

}

const buildCar = () => {

    //base

    const outline = [

        new BABYLON.Vector3(-0.3, 0, -0.1),

        new BABYLON.Vector3(0.2, 0, -0.1),

    ]

    //curved front

    for (let i = 0; i < 20; i++) {

        outline.push(new BABYLON.Vector3(0.2 \* Math.cos(i \* Math.PI / 40), 0, 0.2 \* Math.sin(i \* Math.PI / 40) - 0.1));

    }

    //top

    outline.push(new BABYLON.Vector3(0, 0, 0.1));

    outline.push(new BABYLON.Vector3(-0.3, 0, 0.1));

    //back formed automatically

    //car face UVs

    const faceUV = [];

    faceUV[0] = new BABYLON.Vector4(0, 0.5, 0.38, 1);

    faceUV[1] = new BABYLON.Vector4(0, 0, 1, 0.5);

    faceUV[2] = new BABYLON.Vector4(0.38, 1, 0, 0.5);

    //car material

    const carMat = new BABYLON.StandardMaterial("carMat");

    carMat.diffuseTexture = new BABYLON.Texture("https://assets.babylonjs.com/environments/car.png");

    const car = BABYLON.MeshBuilder.ExtrudePolygon("car", {shape: outline, depth: 0.2, faceUV: faceUV, wrap: true});

    car.material = carMat;

    //wheel face UVs

    const wheelUV = [];

    wheelUV[0] = new BABYLON.Vector4(0, 0, 1, 1);

    wheelUV[1] = new BABYLON.Vector4(0, 0.5, 0, 0.5);

    wheelUV[2] = new BABYLON.Vector4(0, 0, 1, 1);

    //car material

    const wheelMat = new BABYLON.StandardMaterial("wheelMat");

    wheelMat.diffuseTexture = new BABYLON.Texture("https://assets.babylonjs.com/environments/wheel.png");

    const wheelRB = BABYLON.MeshBuilder.CreateCylinder("wheelRB", {diameter: 0.125, height: 0.05, faceUV: wheelUV})

    wheelRB.material = wheelMat;

    wheelRB.parent = car;

    wheelRB.position.z = -0.1;

    wheelRB.position.x = -0.2;

    wheelRB.position.y = 0.035;

    //Animate the Wheels

    const animWheel = new BABYLON.Animation("wheelAnimation", "rotation.y", 30, BABYLON.Animation.ANIMATIONTYPE\_FLOAT, BABYLON.Animation.ANIMATIONLOOPMODE\_CYCLE);

    const wheelKeys = [];

    //At the animation key 0, the value of rotation.y is 0

    wheelKeys.push({

        frame: 0,

        value: 0

    });

    //At the animation key 30, (after 1 sec since animation fps = 30) the value of rotation.y is 2PI for a complete rotation

    wheelKeys.push({

        frame: 30,

        value: 2 \* Math.PI

    });

    //set the keys

    animWheel.setKeys(wheelKeys);

    //Link this animation to a wheel

    wheelRB.animations = [];

    wheelRB.animations.push(animWheel);

    wheelRF = wheelRB.clone("wheelRF");

    wheelRF.position.x = 0.1;

    wheelLB = wheelRB.clone("wheelLB");

    wheelLB.position.y = -0.2 - 0.035;

    wheelLF = wheelRF.clone("wheelLF");

    wheelLF.position.y = -0.2 - 0.035;

    return car;

}

克隆导入的小车时，Babylon.js会自动修改克隆网格的name，以下是为两辆小车启动动画的例子：

var car2=car.clone("car2");

    car2.position.z=1;

    const wheelRB = scene.getMeshByName("wheelRB");

      const wheelRF = scene.getMeshByName("wheelRF");

      const wheelLB = scene.getMeshByName("wheelLB");

      const wheelLF = scene.getMeshByName("wheelLF");

      console.log(wheelRB.animations)

      scene.beginAnimation(wheelRB, 0, 30, true);

      scene.beginAnimation(wheelRF, 0, 30, true);

      scene.beginAnimation(wheelLB, 0, 30, true);

      scene.beginAnimation(wheelLF, 0, 30, true);

      const wheelRB2 = scene.getMeshByName("car2.wheelRB");

      const wheelRF2 = scene.getMeshByName("car2.wheelRF");

      const wheelLB2 = scene.getMeshByName("car2.wheelLB");

      const wheelLF2 = scene.getMeshByName("car2.wheelLF");

      console.log(wheelRB.animations)

      scene.beginAnimation(wheelRB2, 0, 30, true);

      scene.beginAnimation(wheelRF2, 0, 30, true);

      scene.beginAnimation(wheelLB2, 0, 30, true);

      scene.beginAnimation(wheelLF2, 0, 30, true);