Solar System

Project using Babylon JavaScript



By Sunil Dangal

## Introduction

In this project, I am going to use Babylon.js to create visually appealing 3D objects and visuals. I am going to use Babylon.js for real time 3D engine that uses JavaScript library. It is done by using HTML canvas and WebGL. There are images in JavaScript library, and it is used by Babylon.js and we can implement in our Project and create wonderful 3D visuals in our project. We are going to use Babylon smash builder to create 3D objects. We will also create different types of lights, apply materials to our objects. At first, we are going to learn and use basic things on Babylon and after that,

I am going to work on project to build 3D Solar System that we can interact with. I am going to create Sun, Planets, Start and add scenes to the materials that will brings scenes to life. And at last, I am also going to loading external files from our machines to the project.

## Project Setting

Before, we start working on our project, we need to implement few things in our project. Such as

* Setting up Project Structure
* Installation of live server plugin
* And starting development server.

## Learning Basic First

Now, we are going to make a basic scene using Babylon.js. We are going to render basic scene with light, camera and understand the coordinate system in Babylon.js.

After making the development server, first thing we need to do to create a 3D scene in our index.html file, we need to create a canvas element.

WebGL will render all the content to the HTML canvas. Therefore, we need to create objects inside the canvas. After creating canvas, we need to load it in main.js file so that we can use Babylon webGL engine to render canvas. So inside, main.js file, we need to get the canvas and create Babylon engine.

So just like a movie, we need to create scene, camera, and light. So, we have created a scene. In Babylon.js, we can use different camera, and, in this project, we are going to use a free camera. So, this free camera would be like a view of first-person view in a shooting game. Inside the scene, we would be able to move the camera.

Babylon consists of x, y and z coordinate system and we need to give Free Camera and coordinates as a parameter in our camera variable.

Now, to light up our scene, we are going to use Hemispheric Light which is more like our room light which is at the top of the ceiling emitting light to the downwards.

Now, after that, we have introduced **engine.runRenderLoop** function which will allow to render the scene and also add cool animations.

## Learning Basic Second

In Babylon JavaScript, there are Basic elements and Parametric elements that we can create. So basic element is used to create basic shapes, like box, sphere, etc. whereas parametric allows us to create lines, designs using custom mathematical data.

Mesh Builder helps to create shapes that we want using methods. So, we are going to create a box, sphere, plane using Basic elements.

Now using Parametric elements, we can also make lines or custom designs using mathematical coordinates. Similarly, we can also update position, rotate, and scale objects.

Another thing, we can do is, we can modify materials of objects. Materials allows us to cover the meshes with color and textures. It needs a light source to be seen or we can use emissive texture. We can call using **BABYLON.StandardMaterial** and then apply color, textures.

## Learning Basic Third

Now, when it comes to animations and 3D effects, there are options to choose camera and lights in Babylon JavaScript. For example, there are point light, directional light, hemispheric light, and there are Free camera, Arc rotate camera, Universal camera, Device orientation camera and follow camera. So, choosing universal camera with free camera is better option at the end it does the same thing. We have follow-camera, which provides a target to follow in the scene and provide cinematic motions.

## Implementation

Let’s create a solar system project now. We are going to use **ArcRotateCamera** with more settings of camera.

**Sun:** We have added a sphere to make it a sun. We modified and limited the camera movement for the sphere.

**Planets**: Now, we are going to add planets in the scene by creating it first. After creating planets, we are going to apply texture to the planets body and at last we are going to add a background to give it a space look. When the light wave strikes, then it creates the shiny specular effect which we don’t want it to make a planet. So, we use specularColor and choose black color to remove it. Likewise, we have also made background color black by making the scene intensity half from the default and by updating in the createScene function, we have a space kinda look.

Now, we also need to think about ground color light. For example, we have light source Hemispheric light that comes from top and strikes down to the ground and reflects the little intensity of light to the top direction in our objects from the ground. We need to configure that.

**Sun Effect:** We are going to create materials for the sun and add a emissive texture look. To have an effect of sun emitting light from it towards other planets, we need to minimise the glare effect. After that, we need to add another light source which is point light for the sun. Now after this, we will have a sun emitting light to another planets.

**Adding Stars:** After having sun and planets, we need stars to make it space vibe. When we move our camera, we are going to have 3D effect like as if the stars are also moving. To add stars, we are going to use skyBox which is just a box, that we are going to add in the scene. We are going to get this box by applying reflective textures in a group of 6 pictures to have the 3D effect.

So, the box won’t be rendered in the screen because of Back Face Culling in Babylon JavaScript. Therefore, we need to set the Back Face Culling to False to render out, by default its set to true.

Now, to move the skyBox and give a feel of 3D effect, we first do skybox.infiniteDistance = true; and also we have to remove the reflection from the box so we have to update specular and diffuse color. Now to apply the star texture in the cube sides, there are pics inside assets folder, inside skybox folder which we have used.

**Adding Planets**: After adding the skybox to create space and stars visual, it is time to create planets and add rotation and revolving effect around the sun. Now we are going to add animation for orbiting effect. So, we are going to use Math.sin and Math.cos functions to add orbiting animation. Now, we are going to us for loop to add more planets.

**Adding external 3D object**: Now, we have added our planets, it is time to load 3D files inside the project. Babylon has plugins which is loaders plugin that helps to load external 3D files in the scenes. By default, Babylon main library doesn’t incudes loader plugin.

So firstly, we need to load the loaders library files. So, we are using stylised\_spaceship model made by Nuriddin.Xamidov. I am using for my personal project and this 3D object solely belongs to Nuriddin.Xamidov. Shoutout to Nuriddin.Xamidov for that. So that 3D object is loaded in the project.

So, at the end, all the 8 planets were added along with their respective texture. Added a slight camera motion to make it more space look. And the project was successful.