Christmas Comes To Babylon.js

Basic setup and scene

Here we set up Babylon, the ground, lighting and materials to be used.

```
<!DOCTYPE html>
<meta content="text/html;charset=utf-8" http-equiv="Content-Type">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Babylon.js Merry Christmas</title>
</head>

<body>
  <div id="rootDiv">
    <canvas id="renderCanvas"></canvas>
  </div>
  <script src="babylon.1.14.js"></script>
  <script src="mergemeshes.js"></script>
  <script>
    // Get the Canvas element from our HTML below
    var canvas = document.getElementById("renderCanvas");
    canvas.width = document.documentElement.clientWidth * 0.8;
    canvas.height = document.documentElement.clientHeight * 0.8;

    // Load BABYLON 3D engine
    var engine = new BABYLON.Engine(canvas, true);
    var scene = new BABYLON.Scene(engine);

    var mtlSnow = new BABYLON.StandardMaterial("mtlSnow", scene);
    mtlSnow.specularColor = new BABYLON.Color3(0,0,0);
    mtlSnow.ambientColor = new BABYLON.Color3(0.9, 0.9, 1);
    mtlSnow.emissiveColor = new BABYLON.Color3(0.3, 0.3, 1);

    var mtlCarrot = new BABYLON.StandardMaterial("mtlCarrot", scene);
    mtlCarrot.specularColor = new BABYLON.Color3(1,0,0);
    mtlCarrot.ambientColor = new BABYLON.Color3(1, 0, 0);
    mtlCarrot.diffuseColor = new BABYLON.Color3(1, 0, 0);

    var mtlCoal = new BABYLON.StandardMaterial("mtlCoal", scene);
    mtlCoal.specularColor = new BABYLON.Color3(0,0,0);
    mtlCoal.ambientColor = new BABYLON.Color3(0,0,0);
    mtlCoal.emissiveColor = new BABYLON.Color3(0,0,0);
    mtlCoal.diffuseColor = new BABYLON.Color3(0, 0, 0);

    var mtlTrunk = new BABYLON.StandardMaterial("mtlTrunk", scene);
    mtlTrunk.diffuseColor = new BABYLON.Color3(165/255,100/255,6/255);

    var mtlFir = new BABYLON.StandardMaterial("mtlFir", scene);
    mtlFir.diffuseColor = new BABYLON.Color3(0,1,0);

    // Create a camera looking at the origin (0,0,0)
    var camera = new BABYLON.ArcRotateCamera("Camera", 1, 1.3, 200, new BABYLON.Vector3(0, 0, 0), scene);

    // Create a light
    var light0 = new BABYLON.HemisphericLight("Hemi0", new BABYLON.Vector3(0, 1, 0), scene);
    light0.diffuse = new BABYLON.Color3(1, 1, 1);
    light0.specular = new BABYLON.Color3(1, 1, 1);
    light0.groundColor = new BABYLON.Color3(0, 0, 0);

    // Attach the camera to the scene
    scene.activeCamera.attachControl(canvas);

    var ground = BABYLON.Mesh.CreateGround("ground", 500, 500, 2, scene);
    ground.material = mtlSnow;

    function makeSnowman(posx, posz){
    }
```

```
    function makeTree(posx, posz, k){
    }

    makeSnowman(0,0);
    makeTree(10, 10);

    // Once the scene is loaded, just register a render loop to render it
    engine.runRenderLoop(function () {
      scene.render();
    });
  </script>
</body>
</html>
```

Make snowman

Made of three slightly flattened spheres, one on top of the other, with coal eyes and a carrot nose.

```
function makeSnowman(posx, posz){
  // base
  var d = 20;
  var scale = 0.9;
  var base = BABYLON.Mesh.CreateSphere("base", 10.0, d, scene);
  base.material = mtlSnow;

  base.scaling.y = scale;
  var y = d / 2 * scale - 2;
  base.position.y = y;
  base.position.x = posx;
  base.position.z = posz;

  // body
  d = 16;
  y = d * scale - 3;
  var sphere = BABYLON.Mesh.CreateSphere("body", 10.0, d, scene);
  sphere.material = mtlSnow;
  sphere.scaling.y = scale;
  sphere.position.y = y;
  sphere.parent = base;

  // head
  d = 12;
  y += d * scale;
  sphere = BABYLON.Mesh.CreateSphere("head", 10.0, d, scene);
  sphere.material = mtlSnow;
  //sphere.scaling.y = scale;
  sphere.position.y = y;
  sphere.parent = base;

  // eyes
  d /= 2; d -= 0.5;
  var offInRadsHor = 15 * (2 * Math.PI / 360);
  var offInRadsVer = 15 * (2 * Math.PI / 360);

  var offX = Math.sin(offInRadsHor);
  var offY = Math.sin(offInRadsVer);// Math.sin(offInRadsHor);
  var offZ = Math.cos(offInRadsVer);
  sphere = BABYLON.Mesh.CreateSphere("eye-left", 10.0, 1, scene);
  sphere.material = mtlCoal;
  sphere.position.y = y + d * offY;
  sphere.position.x = d * offX;
  sphere.position.z = d * offZ;
  sphere.parent = base;

  sphere = BABYLON.Mesh.CreateSphere("eye-right", 10.0, 1, scene);
  sphere.material = mtlCoal;
  sphere.position.y = y;
  offZ = Math.cos(-offInRadsHor);
  offX = Math.sin(-offInRadsHor);
  sphere.position.x = d * offX;
  sphere.position.y = y + d * offY;
  sphere.position.z = d * offZ;
```

```
  sphere.parent = base;

  //(name, height, diamTop, diamBottom, tessellation, [optional height subdivs], scene, updatable)
  var carrot = new BABYLON.Mesh.CreateCylinder("nose", 5, 1, 2, 6, 1, scene, false);
  carrot.parent = base;
  carrot.material = mtlCarrot;
  carrot.position.z = d;
  carrot.position.y = y;
  carrot.rotation = new BABYLON.Vector3(90 * (2 * Math.PI / 360),0,0);
}
```

Make tree

The trunk is easy – a simple cone. The branches are four sets of boxes spiralling up the trunk, each offset by 45°.

```
function makeTree(posx, posz){
  //(name, height, diamTop, diamBottom, tessellation, [optional height subdivs], scene, updatable)
  var h = 50;
  var trunk = new BABYLON.Mesh.CreateCylinder("trunk", h, 1, 2, 6, 1, scene, false);
  trunk.position.x=posx;
  trunk.position.y=h/2;
  trunk.position.z=posz;
  trunk.material = mtlTrunk;

  var d0 = (90 / h) * (2 * Math.PI / 360);
  var dl = 40 / h;
  var branches = []
  for(var i = 0; i < h-2; i++) {
    var branch = new BABYLON.Mesh.CreateBox("branch" + i + "a", 1.0, scene);
    branch.scaling.x = 40-(dl * i);
    //branch.scaling.z = 2;
    branch.rotation = new BABYLON.Vector3(0, i * d0, 0);
    branch.position.y = i - h/2 + 10;
    branch.material = mtlFir;
    branch.parent = trunk;
    branches.push(branch);

    var branch = new BABYLON.Mesh.CreateBox("branch" + i + "b", 1.0, scene);
    branch.scaling.x = 40-(dl * i);
    //branch.scaling.z = 2;
    branch.rotation = new BABYLON.Vector3(0, i * d0 + 45 * (2 * Math.PI / 360), 0);
    branch.position.y = i - h/2 + 10;
    branch.material = mtlFir;
    branch.parent = trunk;
    branches.push(branch);

    var branch = new BABYLON.Mesh.CreateBox("branch" + i + "c", 1.0, scene);
    branch.scaling.x = 40-(dl * i);
    //branch.scaling.z = 2;
    branch.rotation = new BABYLON.Vector3(0, i * d0 + 90 * (2 * Math.PI / 360), 0);
    branch.position.y = i - h/2 + 10;
    branch.material = mtlFir;
    branch.parent = trunk;
    branches.push(branch);

    var branch = new BABYLON.Mesh.CreateBox("branch" + i + "d", 1.0, scene);
    branch.scaling.x = 40-(dl * i);
    //branch.scaling.z = 2;
    branch.rotation = new BABYLON.Vector3(0, i * d0 + 135 * (2 * Math.PI / 360), 0);
    branch.position.y = i - h/2 + 10;
    branch.material = mtlFir;
    branch.parent = trunk;
    branches.push(branch);
  }
  var tree = mergeMeshes("tree", branches, scene);
  tree.material = mtlFir;
};
```

Make trees

Place a tree at regular interval, then randomly agitate its position. Skip it if it's too close to the snowman.

```
for(j = -200; j < 200; j+=90) {
  for(i = -200; i < 200; i+=90) {
    var ii = i + Math.random() * 75;
    var jj = j + Math.random() * 75;
    if(Math.abs(ii) > 30 || Math.abs(jj) > 30)
      makeTree(ii, jj);
  }
}
```

Make snow

Use Babylon's built-in particle system to set up the snow.

```
// snow
var src="..."; // for full string see addendum at end of document
var fountain = BABYLON.Mesh.CreateBox("fountain", 1.0, scene);
var particleSystem = new BABYLON.ParticleSystem("particles", 20000, scene);

//Texture of each particle
//particleSystem.particleTexture = new BABYLON.Texture("flare.png", scene);
particleSystem.particleTexture = new BABYLON.Texture('data:my_image_name', scene, true,
                         true, BABYLON.Texture.BILINEAR_SAMPLINGMODE,
                         null, null, src, true);

// Where the particles come from
particleSystem.emitter = fountain; // the starting object, the emitter
particleSystem.minEmitBox = new BABYLON.Vector3(-250, 250, 250); // Starting all from
particleSystem.maxEmitBox = new BABYLON.Vector3(250, 200, -250); // To...

// Colors of all particles
particleSystem.color1 = new BABYLON.Color4(0.7, 0.8, 1.0, 1.0);
particleSystem.color2 = new BABYLON.Color4(0.2, 0.5, 1.0, 1.0);
particleSystem.colorDead = new BABYLON.Color4(0, 0, 0.2, 0.0);

// Size of each particle (random between...
particleSystem.minSize = 1;
particleSystem.maxSize = 5;

// Life time of each particle (random between...
particleSystem.minLifeTime = 6;
particleSystem.maxLifeTime = 12;

// Emission rate
particleSystem.emitRate = 500;

// Blend mode : BLENDMODE_ONEONE, or BLENDMODE_STANDARD
particleSystem.blendMode = BABYLON.ParticleSystem.BLENDMODE_ONEONE;

// Set the gravity of all particles
particleSystem.gravity = new BABYLON.Vector3(0, 0, 0);

// Direction of each particle after it has been emitted
particleSystem.direction1 = new BABYLON.Vector3(0, -10, 0);
particleSystem.direction2 = new BABYLON.Vector3(0, -20, 0);

// Angular speed, in radians
particleSystem.minAngularSpeed = 0;
particleSystem.maxAngularSpeed = Math.PI;

// Speed
particleSystem.minEmitPower = 1;
particleSystem.maxEmitPower = 3;
particleSystem.updateSpeed = 0.005;

// Start the particle system
particleSystem.start();
```

Addendum

```
var src =
"
UAAAMAUExURQAAAAMDAQYFAQkIAgwKAw8MBBEOBBQRBRYTBhkVBhwXBx4ZCCAbCSMdCSUfCichCykjDCwlDC0mDTAoDjEqDzQsDzUtEDc
vETkwEjoyEj00Ez42FEA3FUE4FUM6FkU7F0Y9GEg+GEk/GUtBGkxCG01DHE9FHVBGHVFHHlNIH1RJIFVLIFZMIVdNIlhOI1pPJFtQJVxR
JV1SJl5TJ19UKGBVKWJXKWNYKmRZK2VaLGZbLWdcLmhdLmleL2pfMGtgMWxhMm1iM25jNG9kNXBlNXFmNnJnN3NoOHNoOXRpOnVqOnZrO
3dsPHhtPXluPnpvP3twQHxxQX1yQn1zQ350RIB2RYF3RoJ4R4N5SIR5SYV6SoZ7S4d8TIh9TYl+TYl/ToqAT4uBUIyCUY2DUo2DU46EVI
+FVZCGVpGHV5KIWJOJWZSKWpWLW5aMXJeNXZiOXpiPX5mQYJqRYZuSYpySY52TZJ6UZZ+VZp+WZ6CXaKGYaaKZaqKZa6OabKSbbaScbqW
db6aecKeecaifcqmgc6qhdKqhdauidqyjd6ykeK2lea6meq+me7CnfLGofbGpfrKqgLOrgbOrgrSsg7WthLauhbevhriwh7iwiLmxibmy
irqzi7u0jLy0jb21jr62kL63kb+4kr+4k8C5lMG6lcK7lsO8l8S9mMS9mcW+m8W/nMbAncfBnsjBn8nCoMnDocrEosvFo8vFpMzGpc3Hp
87IqM/Jqc/JqtDKq9DLrNHMrtLNr9LNsNPOsdTPstXQs9XQtNbRtdfSt9fTuNjUudnUutrVu9rWvNvXvtzYv9zYwN3Zwd7awt/bxN/bxe
DcxuDdx+HeyOLfyePfy+TgzOThzeXizuXi0Obj0efk0ujl0+jl1Onm1enn1+ro2Ovp2ezp2u3q3O3r3e7s3u7s3+/t4fDu4vHv4/Hv5PL
w5vLx5/Py6PPy6fTz6/X07Pb17fb17vf28Pf38fj48vn48/r59fr69vv79/v7+fz8+v39+/7+/P7+/v///wAAAAAAAFX2SBoAAAAJcEhZ
cwAADsMAAA7DAcdvqGQAAAAYdEVYdFNvZnR3YXJlAHBhaW50Lm5ldCA0LjAuNWWFMmUAAAksSURBVFhH7ZhpWFNXGscNZCEJWQghCRAW2
QKETQFZBBFRXCgKgqgIjFo3BHdcBm2p4zY6tk7dQGvVugzIdOrouNFSRQvZ9/3mXqBuiAsCBdl9njk3UuuHdhJlPvL/AIHk/PK+//Oec9
5zx4xqVKMa1R8KM6zhP0cuwLKzs7MHsrO3+7+AAdAei8XhHVAR8HgcFgUPv/lhAkgsjkAkUah0upOTE51KcSQR8NiRYDEYeyzBwZHGcGG
7cT08PT24bhyWM51CJODsPxQLoiSQqAyWu5cvLyg0NCw8PCSY5+fNZTtTyR+IBWHiSRSGq6c/Pzw6LnHylJSUKZMT46IjQvy93JhUMmrC
8EdtFsYORyAz2F4BoVEJKbPmZM/PXZSXOz97TtrUxJhwnhfHmULEvS8VMIlUJtcvNCZp5tzcpYXrS7aVlm7bvL7o40VZs5LjwgI8WDQis
GD44zYJpE5huPmFxU3LzCss2Xnw+OkLVdX/rLp4pvyLXVuLCrKmT4wI4DIphPexAGOHd3Tm8sYnfbRw1dZ95Rcu36qrF4rFwvq6miv/OH
GgtGhRenJUoAdKtRkKcgfMwKjkzMUbdpdX3bwnVeuNkNkMGfUaWX1N9cl9m5dlpUwI9mRS8DY7gME6OLnzoqZmL9926Pz1eqXBjLwVbFQ
LblUe2VGYMy0myMPZEW+jAaihbgGRKfNWfnrs2ztyA4zAZpNBD2QwATxsVN69fOIvRfOnRQdyGSTbDACGkpk+45KzVn5acaVBY0Zgk16r
kstkUplcqdGbwHfohNdO7SqalxLp70q1LVSQPN09JGHOsh0VV4V6BIH0arlEJBQAgcmSqXQQghgkN07tXJWVFD7WhYSzAQoCpbB8YmYWb
DlyWWhAEKNGIRY2AAkE6E+hWKY2IohJfK1i+9L0+GAulWBDqBgs0cmDPyVn7cFLP+nAaLUMIEGMEqlUIhYBsFCqAjYbBN99uSl3WqQvw5
ZQQTmxfSPTl5edrVXDiEkpARyRVKHR6fU6jVIGsAKJElC1dRf3FGYm8F0p1l3F2DvQufyEBRsPX5VCIE7AFEoUWhPS/PP95kZIp5IKGwR
i4IBZcaNiW15qpAeDaLUAMFgyc2zE1GU7z9XpELNWBphSjanp4ZNnz188a33UDGmBHQKpBkKM9VX7CjPi/Vhkq/nbESicgNj0tX+/IjMj
BoUIANTQ/da2jq7u7q7OtqcPYJQqkuthWHnjWMmC5EA3mrWpwtgT6O78Sdmlp2q1iFkjaWgQq6EHzzq6e/v6+/t6X3U+f2hGLZGoIUR/9
1zZkhlhnk5EK2sVzD3DK2Lqot2VDUZLoCK58eennT39A0NAg/09vzy/DylFDUK5AYHE/zqwOiPSh2ltVWGwJBefyJlLP/+3DIZ1UkGDWI
M8aX81MDj0GmhoaKCno7VRKxUIJFozorp+ZH12jD/b0YqpGByZ5R+TvvLoTRXIXoxGdP95V98gikQ11P/qxUOTHLiqghBt7cmtC+MDORS
c3fDw3xeoUg4vPqP4BLDUhKaphB639wxY4kQ1NNDb0QKrwBsKI6KvO7M9PzHYzdqiAmvUNXDi3DVf39EjJgUaEPKks/dtoK9fD/b98rTp
TQqI8afzZX9KCnGn/e/yx4CKcg1KyFp3pg6MQaHqxtbO37JH8+961qyVWKCmhoufLU7iu9McbIKuPW0b9IItUOApBXiauearH3UWT0Uqu
KW9983cWzTY29naqAZQ4Knx3jefFCQGu1vbUzFgOwmInbO6/HsNYlYD6xSmh22v+t+ZqJ6Xj83otylNiP7O139elBDkCiZqePjvCyx9ll
902vIvrylgGK1Hqa75KZipYerQYF/Xs/t6UFJitRnR1JRvyonjcRytlBQofmfvcakF+78VQ4jBUo/mR21dfWBBASRgdr9sgUH2ApkOhuV
XDq2ZG+3LIltbUWDn8wxNXlB27i6oKZUYHY08ftnV2z8wODjQ39vd/qRJj+4oSmBpQ9We5R+N83a2ukzt8FS3oImZJcdvqmCzDh2u0De2
vOjs7unt7XnV2dbSbASOCqRglWprT5XmTeNzaQRrhz+6Tn2jZhburxaYLKECqg552PqivaOjve3poyaDUiywBApJLh9akz0pgGN960e3K
c+wyQWlJ2tUMKyXiwBVroWaHjxuaXn8oAnSKVAmcBTR3j5b9vGsCWOZ1ixF8ydQXXkxWcX7q+v1iNlCFcqUWgNkhiGjVoUeg0IZuvGLvj
u0Lic53N169pb8mV7hM/K3HvuP1NQIDhRAFYgkMoVSpZBLRYApAsxGs+LWV9uXpMXyWKDzGR76xwLzT3PjJWas2Hm6RmFuNOuVEvTYFwp
FIqEQPaHFCi3UCKtvX9i7OispzJvuYDV7IAyOxPAan5JT/NcLP6oA1aCWi1EaKnD+y1R60Alp6qoObliYGuXvSrapoUY3FXZgzIzcDQcu
1spNCAz6HoVMLAISS+VqHWimIFVd9aEtBWlxwR4MB9uaSbRH8eTHp+VvPHC+RqIHDRlk0GpUKqVKrTWg/ZlBVlv5xZYlsxNDvV2s7fq/C
u2jWWNDE9Lz1+89ffWe0giD/tEMAZnRV0Z1w/Vv/layOCMp3Jdtbdf7TaBBpbF9wibOyi3+7GjlzXoFmvKblhfSKwXfV5fvWleQPinCz5
VmvTt5K4w9kcYZGxo3fd7yzfsqKq/dFshUGq1Oo5KJ6m5cOnlg26oFMxPCfV3pJBuTR4XB4Ig0tjd/wpTZeau37j18uvLytVs//FBz48q
ls0f3lRYXZE6NDfHhoEybA0VtxTlQXTx44xJSM/NWbNix++Dh4xUnjh/5fM8nmwrzs6ZPigzyYoOL1PswLVQCxdnNhx+ZkDp7Xv6yonUb
S0o2rSteUTB/zvTECSG+7kyqwwdc+cBtl+7i4csfH5eUmpaRNS9nfk52Rtr0yfFRIX6ebCfH92cCASweXKLZXB8ePyIyOiY2LnZCZHgIz
8eDw6SR3+uy944s130SxcmFw/Xy8fMPCPD38/HmclycqKQRXPhRLA7vQKbSnZguLDabzWIy6FQyET8CJCr0EQqO4EAkkciOjo5kEpEAgh
zpQxQUC7josxkgLNbyWGakyDcCHJT95tfw/0Y1qlGNalTvasyY/wKvHdSvIHaxxwAAAABJRU5ErkJggg==";
```