CIS367 - Computer Graphics

Other WebGL Libraries

Erik Fredericks - frederer@gvsu.edu


100
50
0

three.js ➜ https://threejs.org/
babylon.js ➜ https://www.babylonjs.com/

phaser(.js) ➜ https://phaser.io/
pixi.js ➜ https://pixijs.io/

What are all of these?

Abstraction layers over WebGL

- You know how much fun it is to draw vertex by vertex...


3

https://davidlyons.dev/threejs-intro

https://medium.com/javascript-in-plain-english/javascript-in-3d-an-introduction-to-three-js-780f1e4a2e6d

Vectors and boxes

Vector3 ➜ (x, y, z)

- Similar to vec3 from book's library

```
const vect = new THREE.Vector3(1, 1, 1);
```

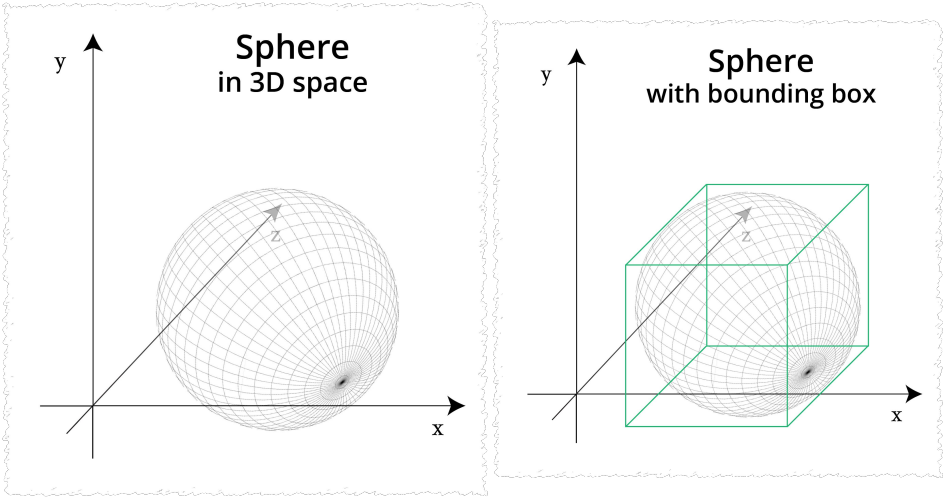
Box3 ➜ 3D box

- Bounding box for other objects
- Aligned with world x, y, z axes

```
const box = new THREE.box3(vect);
```

Box from object

```
const box = new THREE.Box3();
box.setFromObject(object);
```



Visible things!

Mesh is the generic visual element in three.js

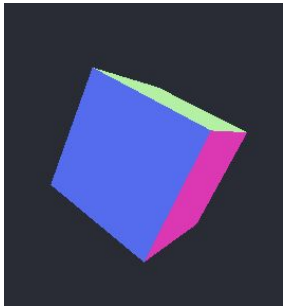
- Geometry ➜ defines shape
- Material ➜ defines appearance

(vertex/fragment shader analogue)

Cube:

```
const geometry = new THREE.BoxGeometry(20, 20, 20);
```

```
const geometry = new
THREE.BoxGeometry(20, 20, 20);
```



```
const geometry = new
THREE.SphereGeometry(20, 64, 64);
```

Ok, basics now:

```
<!DOCTYPE html>
<html>
  <body>
    <script src="js/three.js"></script>
    <script>
        // Our Javascript will go here.
    </script>
  </body>
</html>
```

David Lyons Slides

https://davidlyons.dev/threejs-intro

Orbit controls: https://faculty.computing.gvsu.edu/frederer/three/orbit.html

Materials: https://faculty.computing.gvsu.edu/frederer/three/mat.html

Resources:

- https://threejs.org/docs/#manual/en/introduction/Installation
- https://threejs.org/docs/#examples/en/controls/OrbitControls
- https://threejs.org/manual/#en/materials
- https://threejs.org/manual/#en/lights

http://jaredmmoore.com/EvoEnv/evo_main.html


BABYLON
Rosa-Luxemburg-Straße
30

https://doc.babylonjs.com/examples/

https://www.babylonjs.com/community/

https://developer.mozilla.org/en-US/docs/Games/Techniques/3D_on_the_web/Building_up_a_basic_demo_with_Babylon.js?utm_campaign
➜ https://jsfiddle.net/end3r/8r66fdvp/

–

https://doc.babylonjs.com/setup/starterHTML
https://playground.babylonjs.com/

https://www.reddit.com/r/javascript/comments/a7zbfp/threejs_or_babylonjs_and_why/
https://www.slant.co/versus/11077/11348/~babylon-js_vs_three-js

Differences between 3 and Babylon?

Good question!

Both are equally viable!!

Babylon *tends* to be recommended more often than not

- More "complete" (less plugins)
- Better documented
- Quick bugfixing
- Industry support

But,

- Three.js has been around longer and has more market penetration
  - (lots of tutorials)

phaser.io

One thing you'll notice ... tutorials/links/etc. for multiple versions. Current is Phaser 3. Many games out there run Phaser 2.

➜ Not all tutorials match up!

https://www.phaser.io/tutorials/getting-started-phaser3

Web server required!

The old XSS issue again, so setup a local webserver or use ~~GCP~~ EOS.

You'll also need the library somewhere accessible, per usual

- https://www.phaser.io/download/stable

phaser necessities

The library, obviously (get via CDN, GitHub, download, etc.)

config

```
var config = {
    type: Phaser.AUTO,  // will try to pick WebGL canvas
    width: 800,
    height: 600,
    scene: {
        preload: preload,
        create: create,
        update: update
    }
};
```

phaser necessities

The library, obviously (get via CDN, GitHub, download, etc.)

functions:

- `preload`
- `create`
- `update` (not in hello-world, but in the 10-part demo game)

preload

```
function preload ()
{
    this.load.image('sky',    'assets/sky.png');
    this.load.image('ground', 'assets/platform.png');
    this.load.image('star',   'assets/star.png');
    this.load.image('bomb',   'assets/bomb.png');

    this.load.spritesheet('dude',
        'assets/dude.png',
        { frameWidth: 32, frameHeight: 48 }
    );
}
```

Spritesheet?

```
function create ()
{
    this.add.image(400, 300, 'sky');
    this.add.image(400, 300, 'star');
}
```

https://faculty.computing.gvsu.edu/frederer/phaser/part3.html

https://faculty.computing.gvsu.edu/frederer/phaser/part4.html

```
var platforms;

function create ()
{
    this.add.image(400, 300, 'sky');

    platforms = this.physics.add.staticGroup();

    platforms.create(400, 568,
'ground').setScale(2).refreshBody();

    platforms.create(600, 400, 'ground');
    platforms.create(50, 250, 'ground');
    platforms.create(750, 220, 'ground');
}
```

```
var config = {
    type: Phaser.AUTO,
    width: 800,
    height: 600,
    physics: {
        default: 'arcade',
        arcade: {
            gravity: { y: 300 },
            debug: false
        }
    },
    scene: {
        preload: preload,
        create: create,
        update: update
    }
};
```

Step through some CODE!!!11!

https://faculty.computing.gvsu.edu/frederer/phaser/part10.html

pixi.js

https://pixijs.io/examples/

https://github.com/kittykatattack/learningPixi

p5 and JS!

flow field: https://editor.p5js.org/frederer/sketches/DbWjEErKy
logo seeker: https://editor.p5js.org/frederer/sketches/I6MDAxIca

https://efredericks.github.io/fishyRL (simple roguelike prototype in vanilla JS)
https://efredericks.github.io/CozyRL (overworld roguelike prototype in vanilla JS)

In-Class Things

Pick **one** 2D framework and describe a 'simple' demo project you can do

- Any project, just try something out!
- Might be part of your homework...

Jot down 3 things you'll need to do to accomplish this task