Phaser.io Class Exercise (2: Platformer Extended)

Tasks:

1. Put assets together
    a. Example sound in the JSON:
    b. Using: **fx_game.mp3**

```
{
  "resources": [
    "./assets/audio/fx_game.ogg",
    "./assets/audio/fx_game.mp3"
  ],
  "spritemap": {
    "coin": {
      "start": 0,
      "end": 0.3,
      "loop": false
    },
    "jump": {
      "start": 0.5,
      "end": 0.9,
      "loop": false
    },
    "death": {
      "start": 1,
      "end": 2.8,
      "loop": false
    }
  }
}
```

    Packing timestamp in audio sprite (a demo will be shown in **Adobe Audition**)
    c. There's a way to hard-code (without JSON), **Audio Markers - Play**
    d. Free alternatives for audio editing if you don't want to go to the computer lab:
        i. **https://www.audacityteam.org/**
        ii. **https://www.ocenaudio.com/**
2. Time Delta and Time Scale:
    a. Browse through the Examples (If you forget how to setup the server, please go back to **Phaser.io Class Exercise (1: Platformer)** , for temporary testing, you can also use labs.phaser.io)
        i. Timestep Worldtime
        https://photonstorm.github.io/phaser3-docs/Phaser.Physics.Arcade.World.html
    b. Time Scale:
        i. Time Time Scale
    c. Time Event:
        i. **Time Timer Event**
    d. Global timeScale:
        i. **Notes of Phaser 3 - Timeline**
    e. Hint:

```
<scene>.physics.world.setFPS( that.physics.world.fps / 4 );
<scene>.tweens.timeScale *= 4;
<scene>.physics.world.timeScale = 1.0;
```

    f. Deliverables:
        i. Register a key to change between 2 global speeds.
        ii. Have the enemy spawn every X seconds (instead of having all the stars)
            1. Optional: Make the spawn faster each level
        iii. Have the enemy destroy itself after a certain bounce.
3. Extension of Scene:
    a. Refactoring (Optional), browse through the code in **Scenes->changing large scene**
        i. Put all the functions into the scene class
            1. Hint: you only have one scene, pay attention to the use of " **this** ", replace the config's " **scene** " field
            2. Notes: as time is limited, submission is not required, however it will be extremely useful later when you are implementing levels, for students who has finished the mandatory tasks fast, please at least take a look.
    b. Browse through the code in **Camera Follow Zoom Tilemap**
        i. Identify the session with **tilemap and ignore them** (Will be covered in Isometric tilemap)
        ii. In the example itself, find out why the plane clip through the bottom when you press "Down". (no need to submit, but will be needed in the extension of your platformer)
            1. Hint: Camera's bound is different than the world's bound.

iii. Find the code that extends the worlds size and add to your project
    1. Example uses following code to fill up the background. **Is it efficient ? How could you improve it ?**

```
for( let i = 0; i < 4000; i += 800 ){
        this.sky = this.add.image(400 + i, 300, 'sky').setScale(0.5);
}
```

iv. Extend the size of your base platforms
v. Add more in-air platforms
vi. Find the camera code in the example and add to the platformer.
    1. Hint: your text will be moving, why? Check example: **Camera Pan To**

c. Deliverable:
    i. The same game with a side scrolling gameplay.
        1. Optional: Add more height to the game
        2. Optional2: Add a second level if score higher than certain points (Using the scene switching code from : **Scenes->changing large scene**)