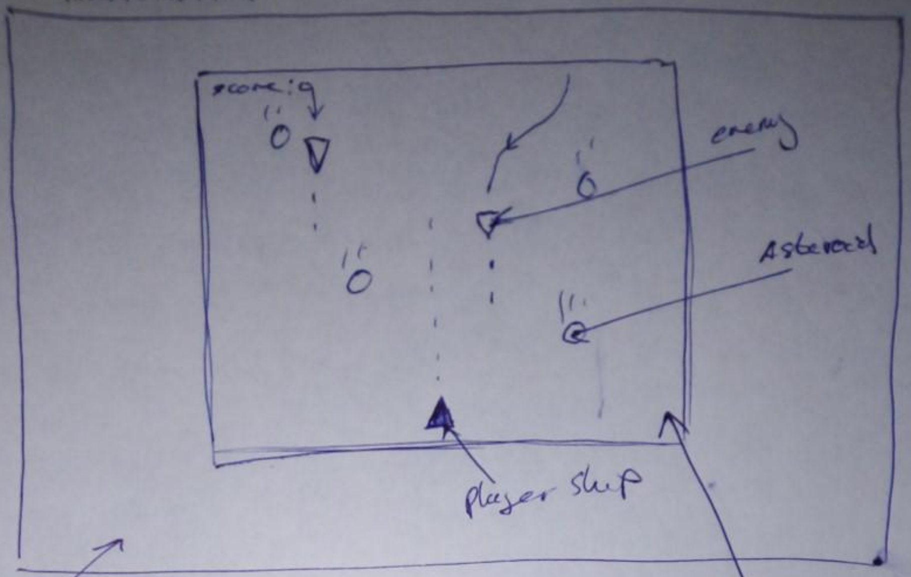
index.html



Muted blue background (easier if user stares at screen for long periods of time)

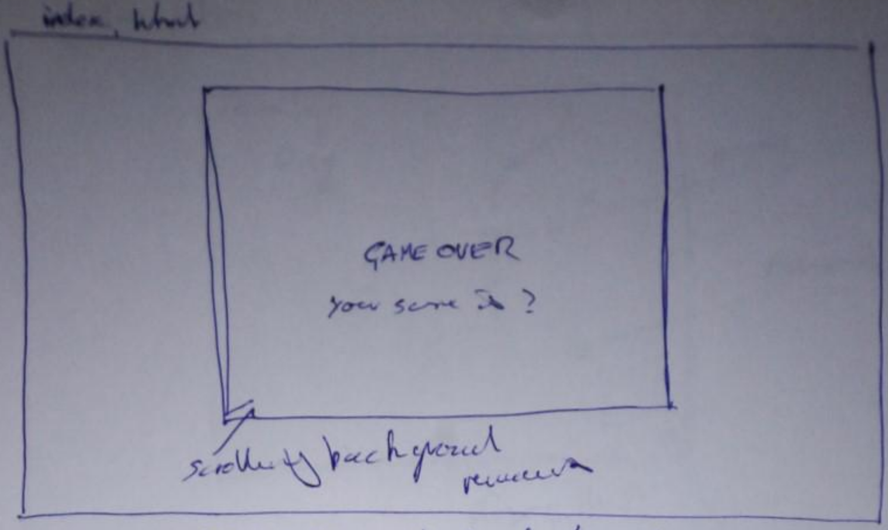
Scrolling background removed.

Keyboard controls

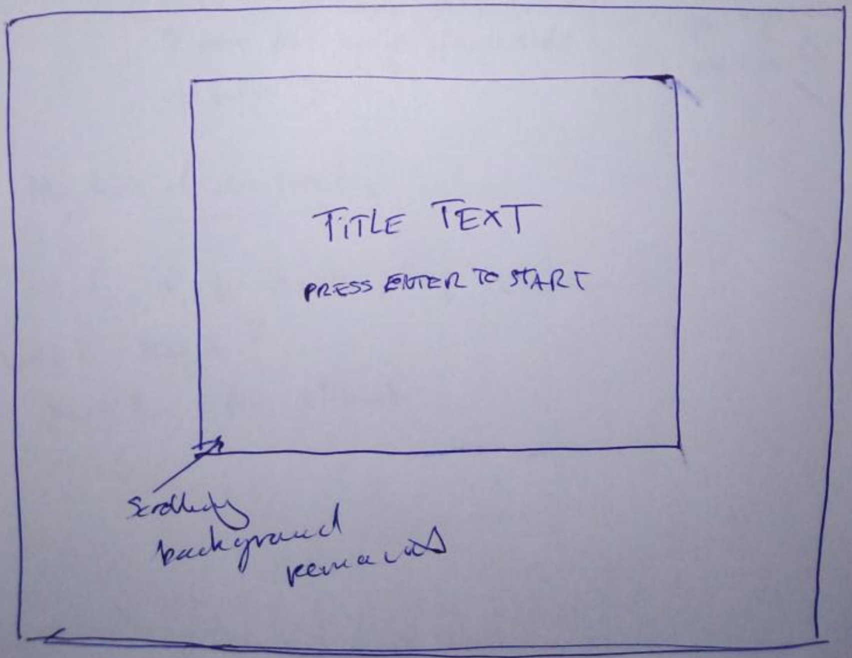
← ↑ → ↓ arrow keys

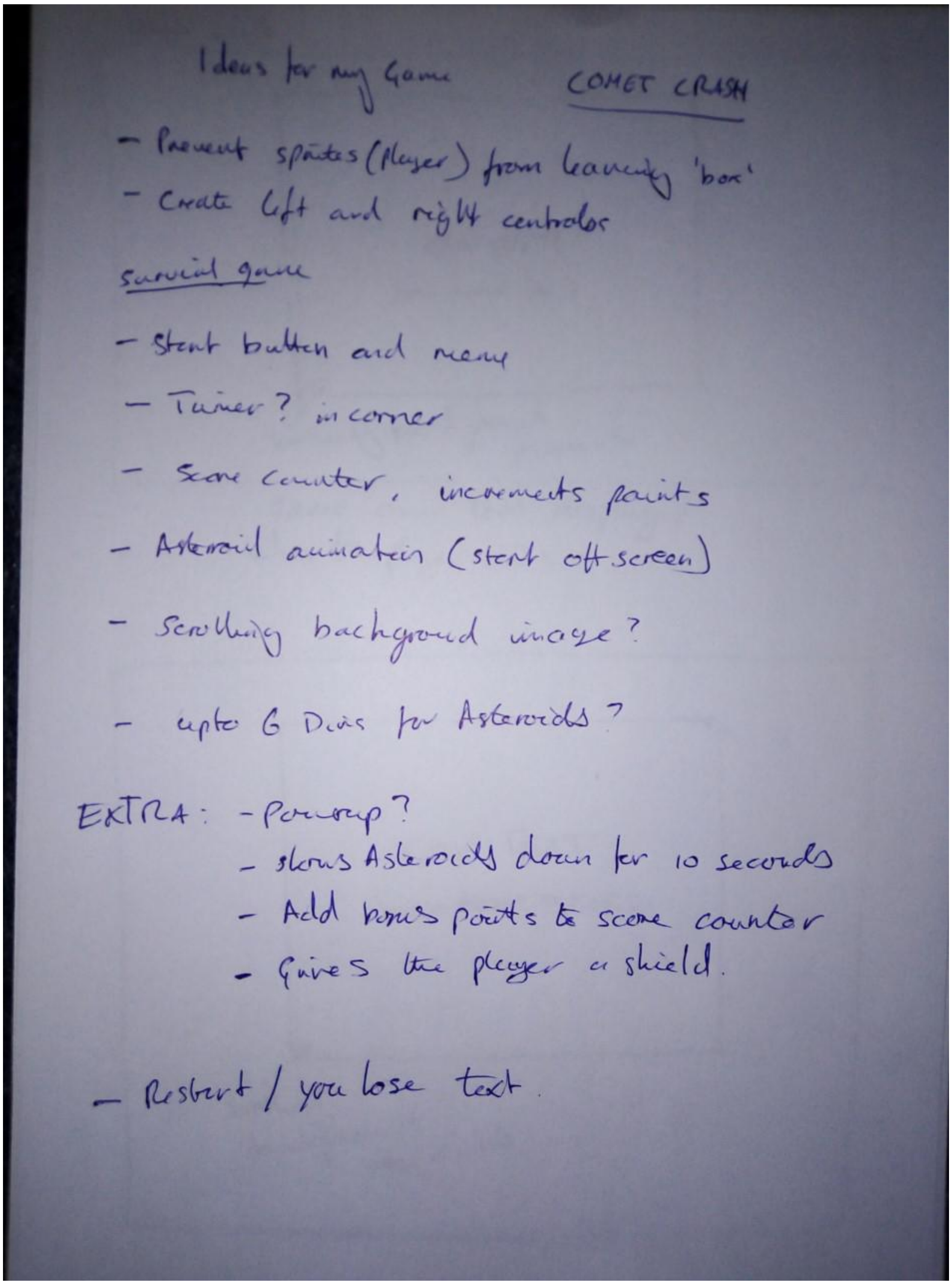
WASD keys?

space bar for shoot.



game over text displayed when the player dies.





Ideas for my Game

<u>COMET CRASH</u>

- Prevent sprites (player) from leaving 'box'
- Create left and right controls

<u>survival game</u>

- Start button and menu
- Timer? in corner
- Score counter, increments points
- Asteroid animation (start off screen)
- Scrolling background image?
- upto 6 Divs for Asteroids?

EXTRA: - Powerup?
- slows Asteroids down for 10 seconds
- Add bonus points to score counter
- Gives the player a shield.

- Restart / you lose text.

bullets loop

- set inside a setTimeout()
- Maybe 1/4 a ~~sec~~ seconds, 250 milliseconds
  goal = slow down rate of fire.

- css. style game window div
- give border white/transparent.
- same size as JS defined viewport.

Enemies

drawEnemies()
Red ∇.png

- build collision to stop player leaving the screen.

```
If ( ship.x == app.view / 1 ) {
    ship.x = -= 0 ! ?
                            ? gameloop
  }
```

Death Conditions

```
If (ship.x &&
    ship.y == enemy.x && enemy.y) {
    ~~player = dead.~~
    player.dead = true!    ?
  }
```

update Bg ():
    place in gameloop (delta);

- Research Game loops in JS.

Asteroids
Enemies
Powerups

Powerup: bullets can destroy Asteroids for a short time only.

Level Ideas

1. Wave of enemies, then 'Asteroid belt' where you can collect powerups for your next battle.

2.
Enemies and Asteroids at same time with infrequent powerups which can be used against both.

1. Pros: Gives more depth to the game, more creative code design.
Cons: Longer time to create, potentially more difficult.

2. Pros: Keeps player permanently focused. (could also be a con?)
Cons: Screen potentially too cluttered, ensure enemies and Asteroid don't interact with each other.

NEED

Intuitive player movements.
Ability to stop animation and change direction.

—

place < script > tags at the Footer of HTML

—

Git commit -m " .... "

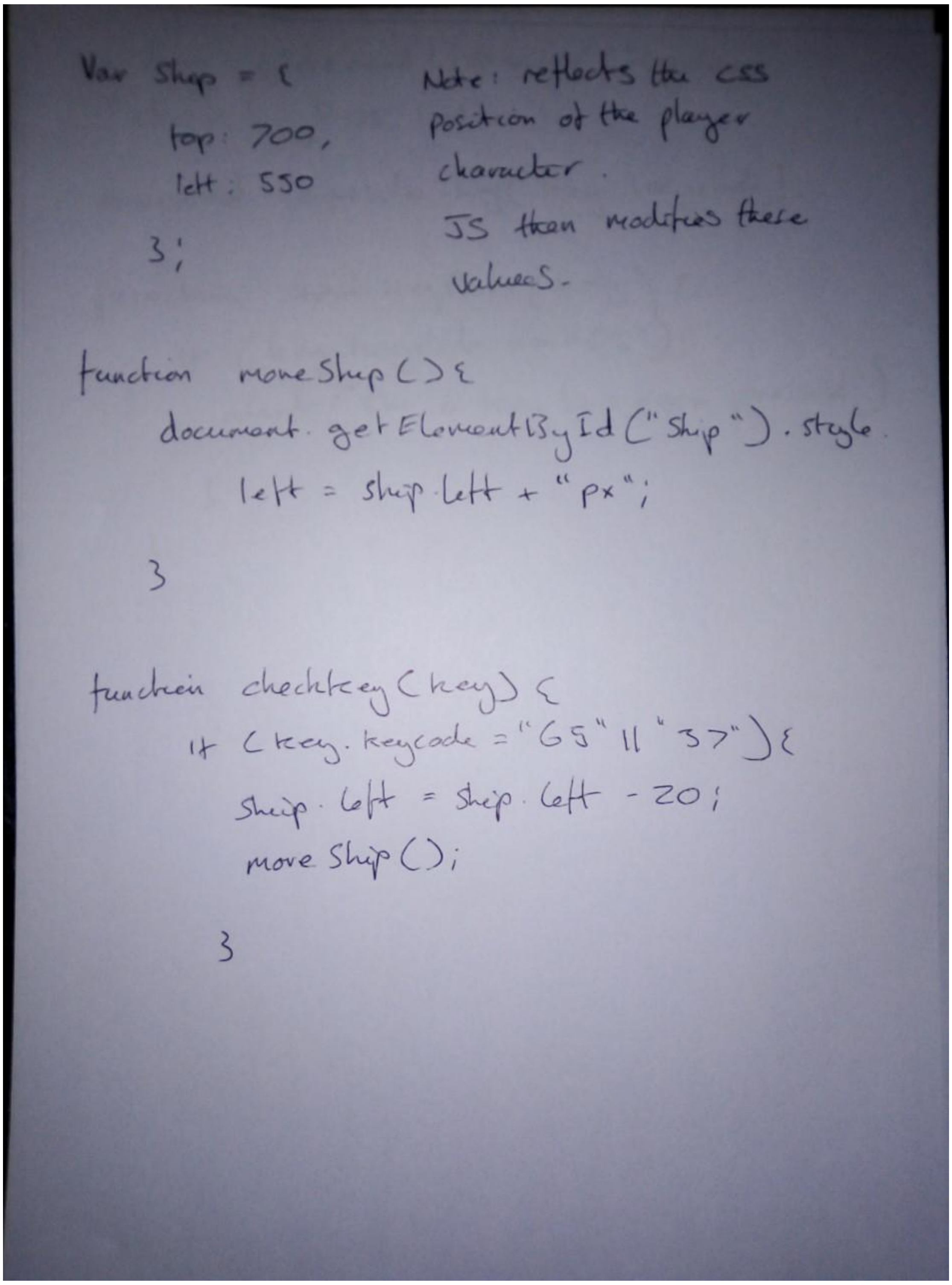
—

search for list of keycodes for keydown function. Not Ashley.

---

use "W A S D" for directional ship controls
<- -> (arrow keys) both ?

"37" = Left
"39" = Right.



```
Var Ship = {
    top: 700,
    left: 550
  };
```

Note: reflects the CSS position of the player character. JS then modifies these values.

```
function moveShip (){
    document.getElementById("ship").style.
        left = ship.left + "px";

  }

function checkkey(key) {
     if (key.keycode = "65" || "37"){
        ship.left = ship.left - 20;
        move Ship();

     }
```