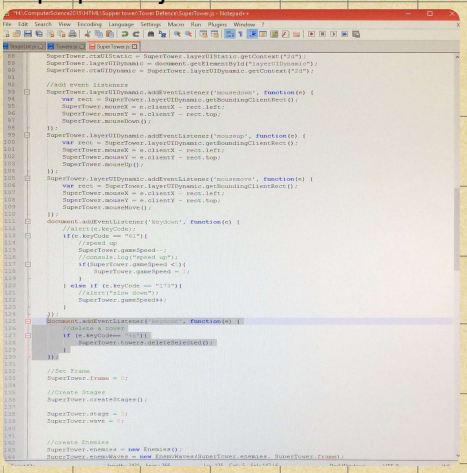
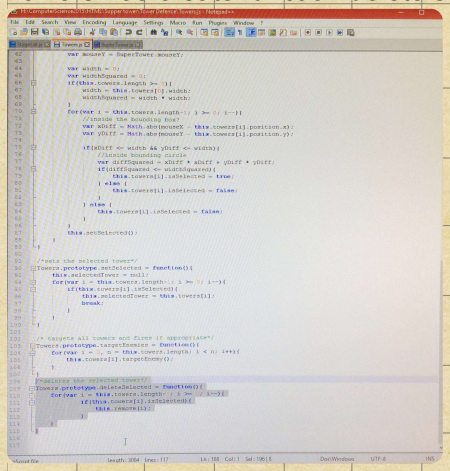
Refund towers

Superpower.js 125 check if delete button is pressed --> call tower delete selected



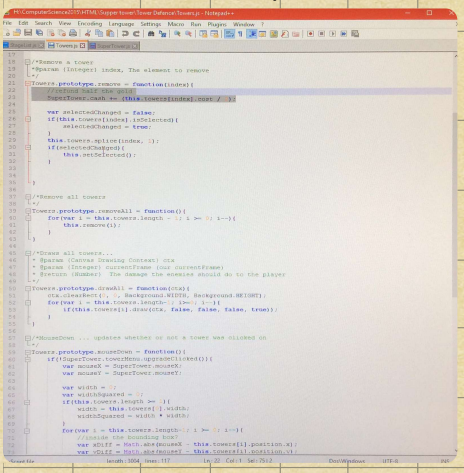
```
    SuperTower.ctxUIStatic = SuperTower.layerUIStatic.getContext("2d");
    SuperTower.layerUIDynamic = document.getElementById("layerUIDynamic");
    SuperTower.ctxUIDynamic = SuperTower.layerUIDynamic.getContext("2d");

    //add event listeners
    SuperTower.layerUIDynamic.addEventListener('mousedown', function(e) {
        var rect = SuperTower.layerUIDynamic.getBoundingClientRect();
        SuperTower.mouseX = e.clientX - rect.left;
        SuperTower.mouseY = e.clientY - rect.top;
        SuperTower.mouseDown();
    });
    SuperTower.layerUIDynamic.addEventListener('mouseup', function(e) {
        var rect = SuperTower.layerUIDynamic.getBoundingClientRect();
        SuperTower.mouseX = e.clientX - rect.left;
        SuperTower.mouseY = e.clientY - rect.top;
        SuperTower.mouseUp();
    });
    SuperTower.layerUIDynamic.addEventListener('mousemove', function(e) {
        var rect = SuperTower.layerUIDynamic.getBoundingClientRect();
        SuperTower.mouseX = e.clientX - rect.left;
        SuperTower.mouseY = e.clientY - rect.top;
        SuperTower.mouseMove();
    });
    document.addEventListener('keydown', function(e) {
        //alert(e.keyCode);
        if(e.keyCode == "61"){
            //speed up
            SuperTower.gameSpeed--;
            //console.log("speed up");
            if(SuperTower.gameSpeed <1){
                SuperTower.gameSpeed = 1;
            }
        } else if (e.keyCode == "173"){
            //alert("slow down");
            SuperTower.gameSpeed++;
        }
    });
    document.addEventListener('keydown', function(e) {
        //delete a tower
        if (e.keyCode== "32"){
            SuperTower.towers.deleteSelected();
        }
    });

    //Set Frame
    SuperTower.frame = 0;

    //Create Stages
    SuperTower.createStages();

    SuperTower.stage = 0;
    SuperTower.wave = 0;


    //create Enemies
    SuperTower.enemies = new Enemies();
    SuperTower.enemyWaves = new EnemyWaves(SuperTower.enemies, SuperTower.frame);
```

Towers 108 create function delete selected tower



```
        var mouseY = SuperTower.mouseY;

        var width = 0;
        var widthSquared = 0;
        if(this.towers.length >= 1){
            width = this.towers[0].width;
            widthSquared = width * width;
        }
        for(var i = this.towers.length-1; i >= 0; i--){
            //inside the bounding box?
            var xDiff = Math.abs(mouseX - this.towers[i].position.x);
            var yDiff = Math.abs(mouseY - this.towers[i].position.y);

            if(xDiff <= width && yDiff <= width){
                //inside bounding circle
                var diffSquared = xDiff * xDiff + yDiff * yDiff;
                if(diffSquared <= widthSquared){
                    this.towers[i].isSelected = true;
                } else {
                    this.towers[i].isSelected = false;
                }
            } else {
                this.towers[i].isSelected = false;
            }
        }
        this.setSelected();
    }
}

/*sets the selected tower*/
Towers.prototype.setSelected = function(){
    this.selectedTower = null;
    for(var i = this.towers.length-1; i >= 0; i--){
        if(this.towers[i].isSelected){
            this.selectedTower = this.towers[i];
            break;
        }
    }
}

/* targets all towers and fires if appropriate*/
Towers.prototype.targetEnemies = function(){
    for(var i = 0, n = this.towers.length; i < n; i++){
        this.towers[i].targetEnemy();
    }
}
/*deletes the selected tower*/
Towers.prototype.deleteSelected = function(){
    for(var i = this.towers.length-1; i >= 0; i--){
        if(this.towers[i].isSelected){
            this.remove(i);
        }
    }
}
```

Towers 22 refund the prices



```
/*Remove a tower
*@param {Integer} index, The element to remove
*/
Towers.prototype.remove = function(index){
    //refund half the cost
    SuperTower.Cash += (this.towers[index].cost / 2);

    var selectedChanged = false;
    if(this.towers[index].isSelected){
        selectedChanged = true;
    }
    this.towers.splice(index, 1);
    if(selectedChanged){
        this.setSelected();
    }


}

/*Remove all towers
*/
Towers.prototype.removeAll = function(){
    for(var i = this.towers.length-1; i >= 0; i--){
        this.remove(i);
    }
}

/*Draws all towers...
 * @param {Canvas Drawing Context} ctx
 * @param {Integer} currentFrame (our currentFrame)
 * @return {Number}  The damage the enemies should do to the player
 */
Towers.prototype.drawAll = function(ctx){
    ctx.clearRect(0, 0, Background.WIDTH, Background.HEIGHT);
    for(var i = this.towers.length-1; i>=0; i--){
        if(this.towers[i].disp(ctx, false, false, false, true)){
    }
}

/*MouseDown ... updates whether or not a tower was clicked on
*/
Towers.prototype.mouseDown = function(){
    if(!SuperTower.towerMenu.upgradeClicked()){
        var mouseX = SuperTower.mouseX;
        var mouseY = SuperTower.mouseY;

        var width = 0;
        var widthSquared = 0;
        if(this.towers.length >= 1){
            width = this.towers[0].width;
            widthSquared = width * width;
        }

        for(var i = this.towers.length-1; i >= 0; i--){
            //inside the bounding box?
            var xDiff = Math.abs(mouseX - this.towers[i].position.x);
            var yDiff = Math.abs(mouseY - this.towers[i].position.y);
```