PlayerObjects.cpp

SetSpritePosition

This method will place the player at the positions that are passed into the method (x,y).

```
void PlayerObjects::SetSpritePosition(int x, int y) {

    Scene1::gdSprite.x = x;
    Scene1::gdSprite.y = y;
   // SDL_WarpMouseInWindow(Scene1::window, 400, 400);

}
```

HoverObjects for menu

This is where you display the message of which menu item that you have hovered over. For example, if I hover over the USE menu item, then the message will be USE.

The `&& Scene1::SceneBackground !=0` is simply to prevent the menu hover message appearing on the splash screen.

```
std::string PlayerObjects::HoverObjects(int x, int y, int scene,int gd, int gy) {

    std::string gameObject;
    Inventory inv;
    int i = 0;
    int items = 0;

    std::string message;

    //Menu Hover Messages

    if (x > 190 && x < 227 && y > 676 && y < 690 && Scene1::SceneBackground !=
"0") {
        message = "Use";
    }

    if (x > 61 && x < 146 && y > 643 && y < 690 && Scene1::SceneBackground !=
"0") {
        Scene1::hoverSound = 1;
        message = "Pick up";
    }

    if (x > 178 && x < 232 && y > 723 && y < 744 && Scene1::SceneBackground !=
"0") {
        message = "Open";
    }

    if (x > 57 && x < 145 && y > 621 && y < 647 && Scene1::SceneBackground !=
"0") {
```

```
        message = "Look";
    }
    if (x > 59 && x < 114 && y > 723 && y < 744 && Scene1::SceneBackground != "0") {
        message = "Pull";
    }
```

Inventory Hover Messages

This is where the hover message displays the name of the inventory item the cursor hovers over. As you can see it first checks if the player has the item in their inventory. There is a space before the name of the item, this is to allow the formation of an action like ‘USE PDA’

```
//Inventory Hover Messages

    if (x >= 696 && x <= 736 && y >= 653 && y <= 687) {
        if(inv.checkItem("PDA") != 0)
            message = Scene1::actionStatement + " PDA";
    }

    if (x >= 764 && x <= 817 && y >= 695 && y <= 755) {
        if(inv.checkItem("Tent") != 0)
            message = Scene1::actionStatement + " Self Inflating Tent";
    }

    if (x >= 771 && x <= 821 && y >= 796 && y <= 798) {
        if(inv.checkItem("Flag") != 0)
            message = "Flag";
    }

    if (x >= 888 && x <= 915 && y >= 653 && y <= 684) {
        if(inv.checkItem("Tape") != 0)
            message = Scene1::actionStatement + " Ape Tape";
    }
    if (x >= 764 && x <= 815 && y >= 695 && y <= 753) {
        if (inv.checkItem("Battery Lantern") != 0)
            message = Scene1::actionStatement + " Battery Lantern";
    }

    if (x >= 888 && x <= 915 && y >= 653 && y <= 684) {
        if(inv.checkItem("Pipe") != 0)
            message = Scene1::actionStatement + " Pipe";
    }

    if (x >= 753 && x <= 827 && y >= 699 && y <= 788) {
        if (inv.checkItem("Disc") != 0)
            message = Scene1::actionStatement + " Disc";
    }
```

Scene Hover Messages

This will show the message of an area on the scene that the players cursor hovers over.
Notice the part where it says:
Message = Scene1::actionStatement + " " This is because that particular area of the scene can have an interaction, like **Look at Mound.**

Scene1::SecretTrigger This is used to tell the game that a secret has been found. For example, if the player found the secret, you wouldn't want the hover message to appear again and again because it had already been discovered.

```
    if (Scene1::SceneBackground == "4a" && x <= 442 && x >= 319 && y < 198 && y > 127) {
        message = Scene1::actionStatement + " Mound";
    }


    if (Scene1::SceneBackground == "4a" && x <= 837 && x >= 699 && y < 504 && y > 483 && Scene1::secretTrigger < 5) {
        message = Scene1::actionStatement + " Markings";
    }


    if (Scene1::SceneBackground == "4a" && x <= 851 && x >= 729 && y < 470 && y > 406) {
        message = Scene1::actionStatement + " Entrance";
    }

    if (Scene1::SceneBackground == "3d" && Scene1::secretTrigger > 2 && x <= 1000 && x >= 959 && y < 402 && y > 292) {
            message = Scene1::actionStatement + " Drawing";
    }

    if (Scene1::SceneBackground == "3d" && x <= 1009 && x >= 948 && y < 457 && y > 409) {
        message = Scene1::actionStatement + " Strange peg";
    }

    if (Scene1::SceneBackground == "1da" && x <= 971 && x >= 843 && y < 231 && y > 142) {
        message = Scene1::actionStatement + " Spaceflix";
    }

    if (Scene1::SceneBackground == "1da" && x <= 411 && x >= 364 && y < 402 && y > 396) {
        message = Scene1::actionStatement + " Pot plant";
    }

    if (Scene1::SceneBackground == "3f" && x <= 510 && x >= 490 && y < 262 && y > 247 && Scene1::secretTrigger == 2 && inv.checkItem("Disc") != 1) {
        message = Scene1::actionStatement + " Sparkling object";
    }
    if (Scene1::SceneBackground == "3f" && x <= 560 && x >=483 && y < 262 && y > 202 && Scene1::secretTrigger <2) {
        message = Scene1::actionStatement + " Loose rocks";
    }
```

```
    if (Scene1::SceneBackground == "3f" && x <= 20 && y < 414 && y > 250 ) {
        message = Scene1::actionStatement + " Leave crator";
    }
```

DestroyObjects

The **DestroyObjects** method is used to specify the ID for each object that is picked up and then the object render class will not render them if they have been picked up by the player.

```
std::string PlayerObjects::DestroyObjects(std::string gameObject) {

    std::string objectToDestroy;

    if (gameObject == "PDA") {
        SDL_DestroyTexture(Textures::objectTexture);
        objectToDestroy = "1";
    }

    if (gameObject == "Tape") {
        SDL_DestroyTexture(Textures::objectTexture4);
        objectToDestroy = "3";
    }

    if (gameObject == "Flag") {
        SDL_DestroyTexture(Textures::objectTexture2);
        objectToDestroy = "2";
    }
```