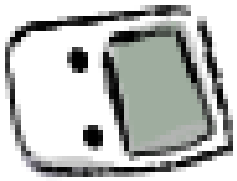
Creating a sprite object in ThePlanet game step by step

1. Create your sprite
   Use a .png if you need it to be transparent (normally what you would want to do)



2. Head to Textures.cpp
3. Create a new object texture for your sprite

```
//Object Textures.
SDL_Texture* Textures::objectTexture;
SDL_Texture* Textures::objectTexture2;
SDL_Texture* Textures::objectTexture3;
SDL_Texture* Textures::objectTexture4;
SDL_Texture* Textures::objectTexture5;
SDL_Texture* Textures::objectTexture6;
SDL_Texture* Textures::objectTexture7;
SDL_Texture* Textures::objectTextureAirBox;
SDL_Texture* Textures::objectTexturePipe;
SDL_Texture* Textures::objectTexturePipeAction;
SDL_Texture* Textures::objectTextureLantern;
SDL_Texture* Textures::objectTextureSparkle;

SDL_Texture* Textures::myPDATexture
```

4. Head to Textures.h
5. Add a static entry for your new sprite, just with ‘static’ in front of the text:

```
static SDL_Texture* objectTexture;
static SDL_Texture* objectTexture2;
static SDL_Texture* objectTexture3;
static SDL_Texture* objectTexture4;
static SDL_Texture* objectTexture5;
static SDL_Texture* objectTexture6;
static SDL_Texture* objectTexture7;
static SDL_Texture* objectTextureAirBox;
static SDL_Texture* objectTexturePipe;
static SDL_Texture* objectTexturePipeAction;
static SDL_Texture* objectTextureLantern;
static SDL_Texture* objectTextureSparkle;

static SDL_Texture* myPDATexture;
```

6. Head back to Textures.cpp

7. Find the method for the scene textures you would like to add your sprite to. For example for Scene1Textures it is here:

```
void Textures::Scene1Textures() {

    //Background dimensions (x,y,width,height).
    //Main background Rect
    menuBackground = { 0, 0, 1200, 768 };
```

8. Now add your texture to the method you will be using.

```
    //Objects
    imageSurface = IMG_Load("Objects/pda.png");
    invTexture1 = SDL_CreateTextureFromSurface(Scene1::renderer, imageSurface);
    myPDATexture = SDL_CreateTextureFromSurface(Scene1::renderer, imageSurface);
    SDL_FreeSurface(imageSurface);
```

Note: See the part where it says invTexture1.. Well that is for objects that you plan to pick up. There are 7 slots you can use, but remember that if one is already being used, then that object would need to have been used before assigning one of those invTextures. The object will be rendered in one of the slots when you pick it up.

```
//Inventory Textures.
SDL_Texture* Textures::invTexture1;
SDL_Texture* Textures::invTexture2;
SDL_Texture* Textures::invTexture3;
SDL_Texture* Textures::invTexture4;
SDL_Texture* Textures::invTexture5;
SDL_Texture* Textures::invTexture6;
SDL_Texture* Textures::invTexture7;
```

9. Now the difficult bit.
10. Head over to PlayerObjects.cpp
11. At the top you will need to add a couple of SDL_Rect entries. This is for animating the object, but even if you plan not to animate the object, then you will still need to add them.
12. Add a new incremental srcrect and dstrect as shown below:

```
SDL_Rect PlayerObjects::srcrect;
SDL_Rect PlayerObjects::dstrect;
SDL_Rect PlayerObjects::srcrect2;
SDL_Rect PlayerObjects::dstrect2;
SDL_Rect PlayerObjects::srcrect3;
SDL_Rect PlayerObjects::dstrect3;
SDL_Rect PlayerObjects::srcrect4;
SDL_Rect PlayerObjects::dstrect4;
SDL_Rect PlayerObjects::srcrect5;
SDL_Rect PlayerObjects::dstrect5;
SDL_Rect PlayerObjects::srcrect6;
SDL_Rect PlayerObjects::dstrect6;
```

Finally, add the following to PlayerObjects.h but add static to the front of them.

```
class PlayerObjects
  {


   public:
       static int boxOpened;
       static SDL_Rect srcrect;
       static SDL_Rect dstrect;
       static SDL_Rect srcrect2;
       static SDL_Rect dstrect2;
       static SDL_Rect srcrect3;
       static SDL_Rect dstrect3;
       static SDL_Rect srcrect4;
       static SDL_Rect dstrect4;
       static SDL_Rect srcrect5;
       static SDL_Rect dstrect5;
       static SDL_Rect srcrect6;
        static SDL_Rect dstrect6;
```

13. In PlayerObjects.cpp, go to: std::string PlayerObjects::DestroyObjects(std::string gameObject)

14. Add an additional entry shown below:
    Make sure that the number is incremental to the last one in the list. This for me is 8.

```
if (gameObject == "PDA") {
        objectToDestroy = "8";
}
```

15. Now find (in PlayerObjects.cpp):
    std::string PlayerObjects::ObjectInteractionM1(int playerCurrentLocationX, int playerCurrentLocationY) {

16. Add your sprite object here and specify which scene it will appear on. Make sure you give it the same name as you did in the previous step. You can see here that it is for scenebackground 1 and the location of where the player will be able to pick it up on the screen.

```
if (Scene1::SceneBackground == "1" && playerCurrentLocationX >= 609 &&
playerCurrentLocationX <= 719) {
        message = "PDA";
    }
```

17. Now find (in PlayerObjects.cpp):

```
std::tuple<int, int, int, int, int> PlayerObjects::ObjectSettings(int
scene, int objectID, int b, int c, int d) {
```

18. Add your sprite object as an entry shown below:

```
if (scene == 1 && objectID == 1) {
        return  std::make_tuple(NULL, NULL, NULL, 40, 40);
}
```

Just specify here the size that you want the image to be, I've specified 40,40. And it is important to ensure the objectID is incremental to any other objects that there might already be. So, if there were already 8 objects, then objectID would be 9.

For your information:
Notice the NULL,NULL,NULL in our code above.. Well, here is where you might add frames if you wished to animate as shown in an example below:

```
   if (scene == 1 && objectID == 3) {
        //Stars
        return  std::make_tuple(NULL, 15, NULL, 183, 347); //You can see here
that I set the scroll speed to 15 because the star is animated. This will scroll
the RECT at a speed of 15 from left to right.
```

The last 2 parameters are the size of the window to scroll through (animate).

We are not trying to animate, so you can ignore this highlighted in green above.

19. Now find (in PlayerObjects.cpp):

```
std::tuple<int, int, int, int, int> PlayerObjects::placeObject(int scene,
int objectID, int b, int c, int d) {
```

20. Add your sprite object as shown below:

```
if (scene == 1 && objectID == 1) {
        //PDA
        return  std::make_tuple(1, 685, 523, 20, 14);
}
```

Here is where it gets confusing. The first value in the parameters is number of animations in your image or number of sprites. We only are dealing with a static image so we would specify 1.

Next is the x and y position of the location where you would like to place the object. Here I have specified 685,523.

Finally, specify the dimensions of the object, the exact size you would like the object to appear on the screen. You can play around with these values until you are satisfied. Here I have specified 20,14.

21. Now find (in PlayerObjects.cpp):

```
void PlayerObjects::ObjectController() {
```

22. Add your sprite object as shown below:

```
std::tie(numberSprites, objectP1, objectP2, objectHeight, objectWidth) =
pob.ObjectSettings(1, 1, NULL, NULL, NULL);
    std::tie(numberSprites, objectP3, objectP4, objectP5, objectP6) =
pob.placeObject(1, 1, NULL, NULL, NULL);

    int object1 = (ticks / 100) % numberSprites;
    PlayerObjects::srcrect6 = { object1 * objectP1, objectP2, objectHeight,
objectWidth };
        PlayerObjects::dstrect6 = { objectP3, objectP4, objectP5, objectP6 };
```

As you can see above, it looks quite complicated. The only 2 things you need to pay attention to are:

Pob.ObjectSettings(1,1, NULL,NULL, NULL)
The highlighted 1 is the object ID, so this needs to be the same as the ID you set earlier.

PlayerObjects::srcrect6 =
PlayerOjbects::dstrect6 =

The highlighted 6 means that we are dealing with the correct SDL_RECT that we assigned earlier, so make sure that matches.

23. Now head over to the **objectRender.cpp** class.
24. Now we will render our object to the screen AND to our inventory if we wish to pick it up. Add the following code below to the method:
    Void ObjectRender::objectRender()

```
//PDA Inventory item.
    if (Scene1::objectToDestroy.find("8") != std::string::npos) {
        if (Inventory::inv.find("8")) {
            SDL_RenderCopy(Scene1::renderer, Textures::invTexture1, NULL,
&Inventory::inv1);
        }
    }

else if (Scene1::SceneBackground == "1") { SDL_RenderCopy(Scene1::renderer,
Textures::myPDATexture, &PlayerObjects::srcrect6, &PlayerObjects::dstrect6); }
```

In the above code that we added, make sure the ObjectToDestory.find() matches the number you gave the object earlier. We gave it an 8 remember.

Make sure you specify the texture you created (myPDATexture) and also the correct srcrect and dstrect which we specified as 6.

Notice the Inventory::inv.find(“8”):

This means render if the item is in the player’s inventory.

ELSE render it on the scene.