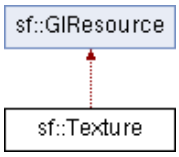
Textures

In this tutorial we will see how to load and mapped the texture into a graphical entities and much more.

A texture is just an image(collection of pixels) that's mapped into a 2D or 3D objects.



Loading a texture

Before creating any sprite we need a valid texture. The class that encapsulates texture in SFML is `sf::Texture`. Since the only role of a texture is to be loaded and mapped to graphical entities, almost all its functions are about loading and updating it.

The most common way of loading a texture is from an image file on disk, which is done with the `loadFromFile` function.

```
sf::Texture texture;
if (!texture.loadFromFile("image.png"))
{
    // error...
}
```

Note

The `loadFromFile` function can sometimes fail with no obvious reason. First, check the error message that SFML prints to the standard output (check the console). If the message is **unable to open file**, make sure that the **working directory** (which is the directory that any file path will be interpreted relative to) is what you think it is: When you run the application from your desktop environment, the working directory is the executable folder. However, when you launch your program from your IDE (Visual Studio, Code::Blocks, ...) the working directory might sometimes be set to the **project** directory instead. This can usually be changed quite easily in the project settings.

Loading small section of the Image into a texture

There is an optional arrgument in loadFromFile( ) function which will allow you to load a part of image.

For Example-

```
// load a 32x32 rectangle that starts at (10, 10)
if (!texture.loadFromFile("image.png", sf::IntRect(10, 10, 32, 32)))
{
    // error...
}
```

The `sf::IntRect` class is a simple utility type that represents a rectangle. Its constructor takes the coordinates of the top-left corner, and the size of the rectangle.

Create and Update Texture

If you don't want to load a texture from an image, but instead want to update it directly from an array of pixels, you can create it empty and update it later :

Create texture

```
// create an empty 200x200 texture
if (!texture.create(200, 200))
{
    // error...
}
```

Note that the contents of the texture are undefined at this point.

Update texture

To update the pixels of an existing texture, you have to use the `update` function. It has overloads for many kinds of data sources :

```
// update a texture from an array of pixels
sf::Uint8* pixels = new sf::Uint8[width * height * 4]; // * 4 because pixels have 4 components (RGBA)
...
texture.update(pixels);

// update a texture from a sf::Image
sf::Image image;
...
texture.update(image);

// update the texture from the current contents of the window
sf::RenderWindow window;
...
texture.update(window);
```

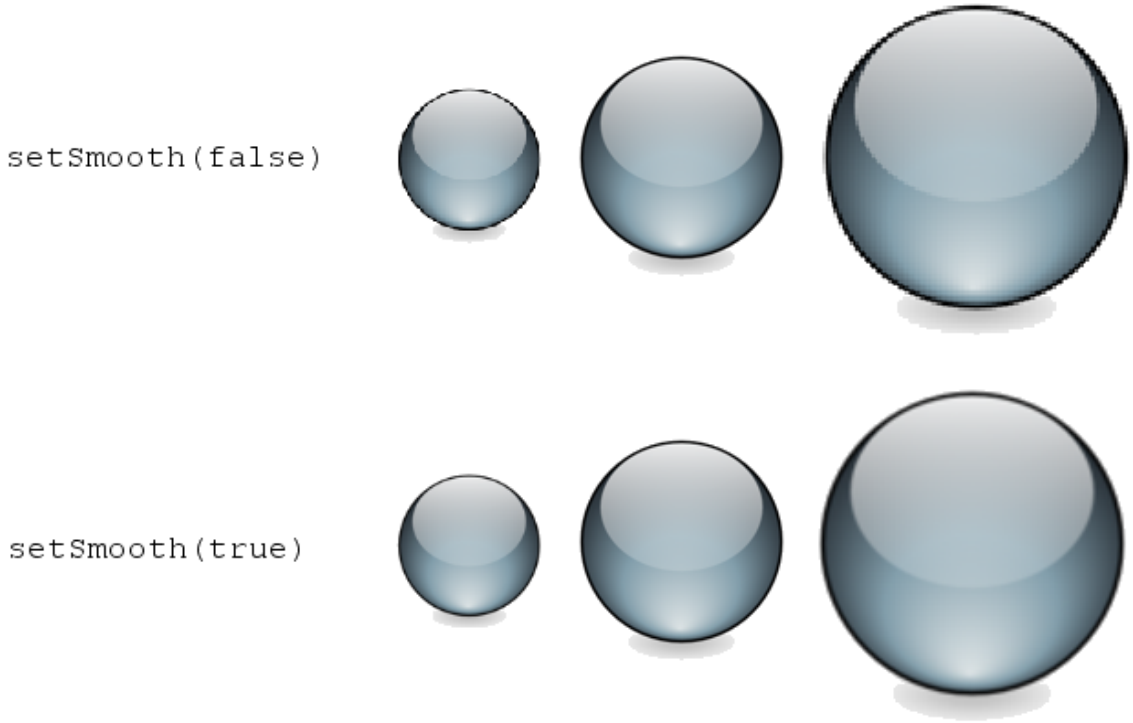
These examples all assume that the source is of the same size as the texture.

Property of texture

Smooth the texture

The first property allows one to smooth the texture. Smoothing a texture makes pixel boundaries less visible (but the image a little more blurry).

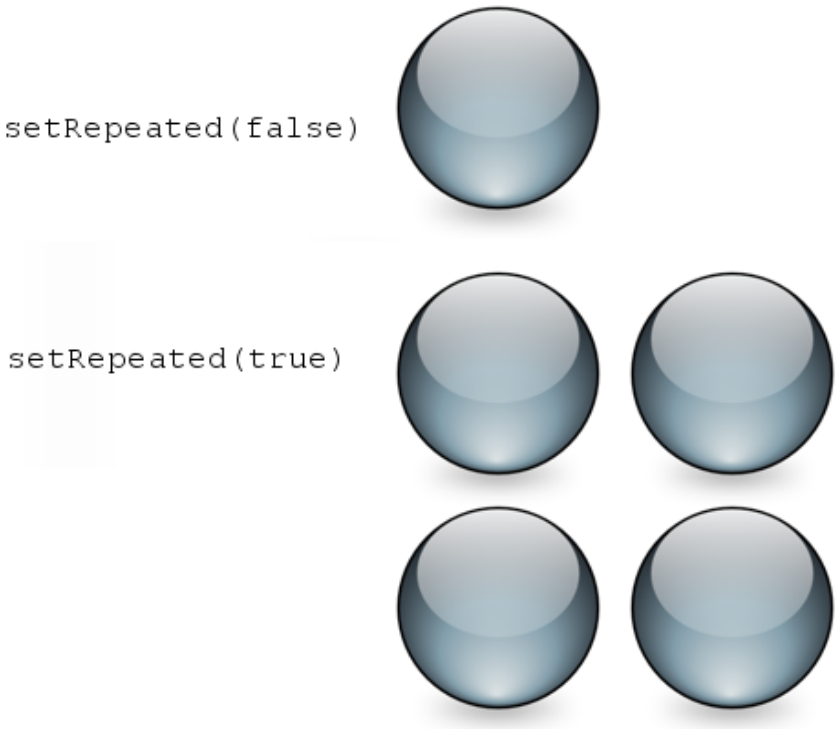
```
texture.setSmooth(true);
```



Repeat the texture

The second property allows a texture to be repeatedly tiled within a single sprite. This only works if your sprite is configured to show a rectangle which is larger than the texture, otherwise this property has no effect.

```
texture.setRepeated(true);
```



To know more about functions related to texture then follow the below link :)

sf::Texture Class Reference (SFML / Learn / 2.5.1 Documentation)

Image living on the graphics card that can be used for drawing. sf::Texture stores pixels that can be drawn, with a sprite for example. A texture lives in the graphics card memory, therefore it is very fast to draw a texture to a

https://www.sfml-dev.org/documentation/2.5.1/classsf_1_1Texture.php