```
// Sprite.h
#pragma once

#include "SDL.h"
#include <vector>
#include <map>

class Sprite
{
public:
   Sprite(int width, int height, SDL_Renderer* ren);
   ~Sprite(void);

   void setPos(int x, int y);
   void movex(int delta);
   void movey(int delta);
   int getX();
   int getY();
```

```
// makeFrame returns the unique index of the frame
// x & y define the upper left hand corner of the
// sprite frame, inside the texture
int makeFrame(SDL_Texture* texture, int x, int y);

// addFrameToSequence returns the number of frames
// in the sequence after the add
int addFrameToSequence(std::string seqName,
                       int frameIndex);

// show(int) renders the frame with the specified
// frameIndex
void show(int frameIndex);

// show(string) cycles through all frames in the
// specified sequence, one per call
void show(std::string sequence);
```

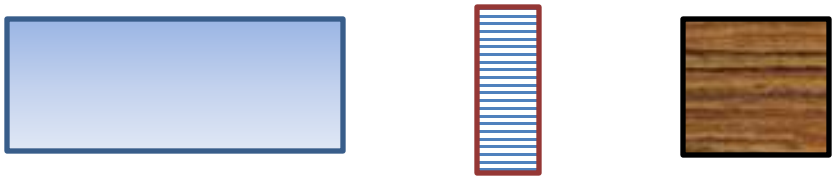
```
// The private part of the class is given as a hint
// or suggestion.
// In homework 3 you can make any modifications you
// want to the class's innards.
private:
   int width, height;
   int currX, currY;    // sprite coordinates

   SDL_Renderer* renderer;

   struct frame
   {
      SDL_Texture* texture;
      int x;
      int y;
   };
   std::vector<frame> frames;
```

```
    std::map<std::string, std::vector<int>>
                          sequenceList;

    // shared by all sequences;
    // it would be better to have
    // one for each sequence
    int sequenceIndex;

};
```

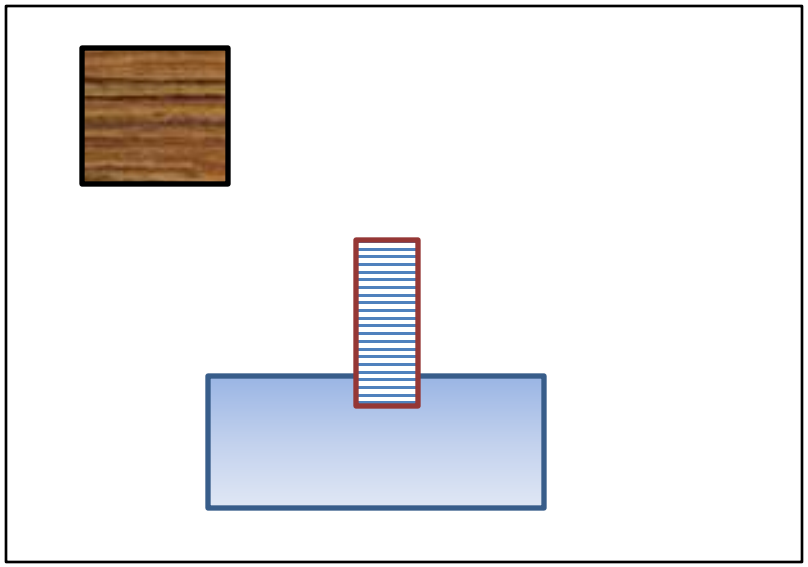
```
void logSDLError(std::ostream &os,
                 const std::string &msg)
{
   os << msg << " error: " << SDL_GetError()
      << std::endl;
   // And now for something completely different
   std::ostringstream errMsg;
   errMsg << " error: " << SDL_GetError()
          << std::endl;
   OutputDebugString(errMsg.str().c_str());
}
```

```
// from SpriteDemo.cpp

SDL_Renderer *renderer =
       SDL_CreateRenderer(window, ……);

const std::string resPath =
              getResourcePath("SpriteDemo");
SDL_Texture *background =
         loadTexture(resPath + "Background.png",
         renderer);

Sprite* spriteBG = new Sprite(SCREEN_WIDTH,
                        SCREEN_HEIGHT, renderer);
spriteBG->setPos(0,0);
int bgFrame = spriteBG->makeFrame(background,
                                  0, 0);

SDL_RenderClear(renderer);
spriteBG->show(bgFrame);
SDL_RenderPresent(renderer);
```

1. Three SDL_Textures:

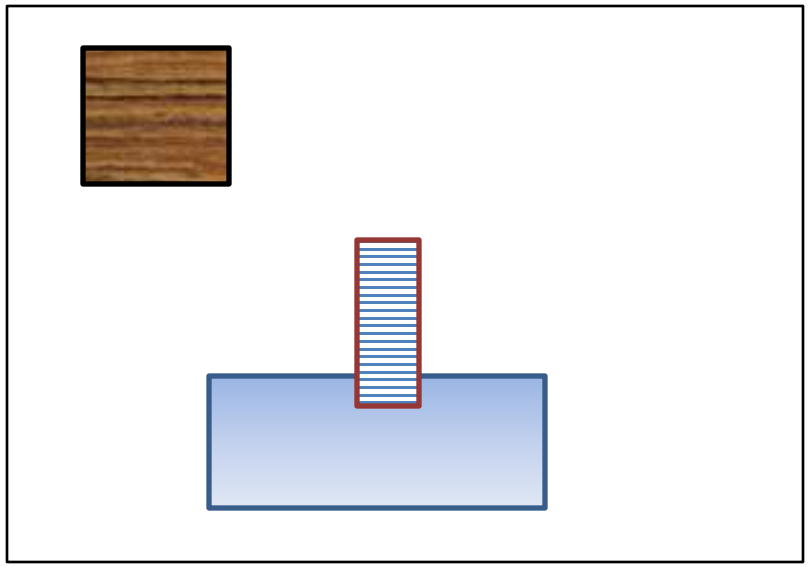


2. Copied onto an SDL_Renderer with SDL_RenderCopy:



Offscreen.

3. Made visible in an SDL_Window with SDL_RenderPresent:



Onscreen.