C++ Programming for Engineers

Petar Jervic
Francisco Luro
Andrew Moss (course-resp)

May 2, 2012

Writing interactive applications

- SDL overview
- Event based systems
- Polling
- Callbacks
- Application main loop
- Game logic vs display
- SDL project setup
- Example - Game of Life

Simple Direct Media Layer (SDL)

- A library to make interactive applications
  - games, simulations, audio/video players, etc
- Supports the critical parts needed to build games
  - graphics, sounds, events, network sockets, timers
- Cross-platform
  - Same code for different targets (Win, Linux, Mac,...)
  - Windows handling, keyboard handling, mouse, sounds, images
- Very useful to tackle the final project for the course

Event based systems

- What is an event?
  - Something that can happen at any time...
    - Pressing a key on the keyboard
    - Receiving a new email
    - Clicking with the mouse
    - 10 seconds passed since...
- How our application knows about events?
  - Check every X time, did this event happen ?
    - This technique is often called "polling"
  - Register our interest in a specific event, and get called when it happens
    - This is referred to as "callbacks"
    - We can have different pieces of code trying to access the same data

Polling the system (SDL)

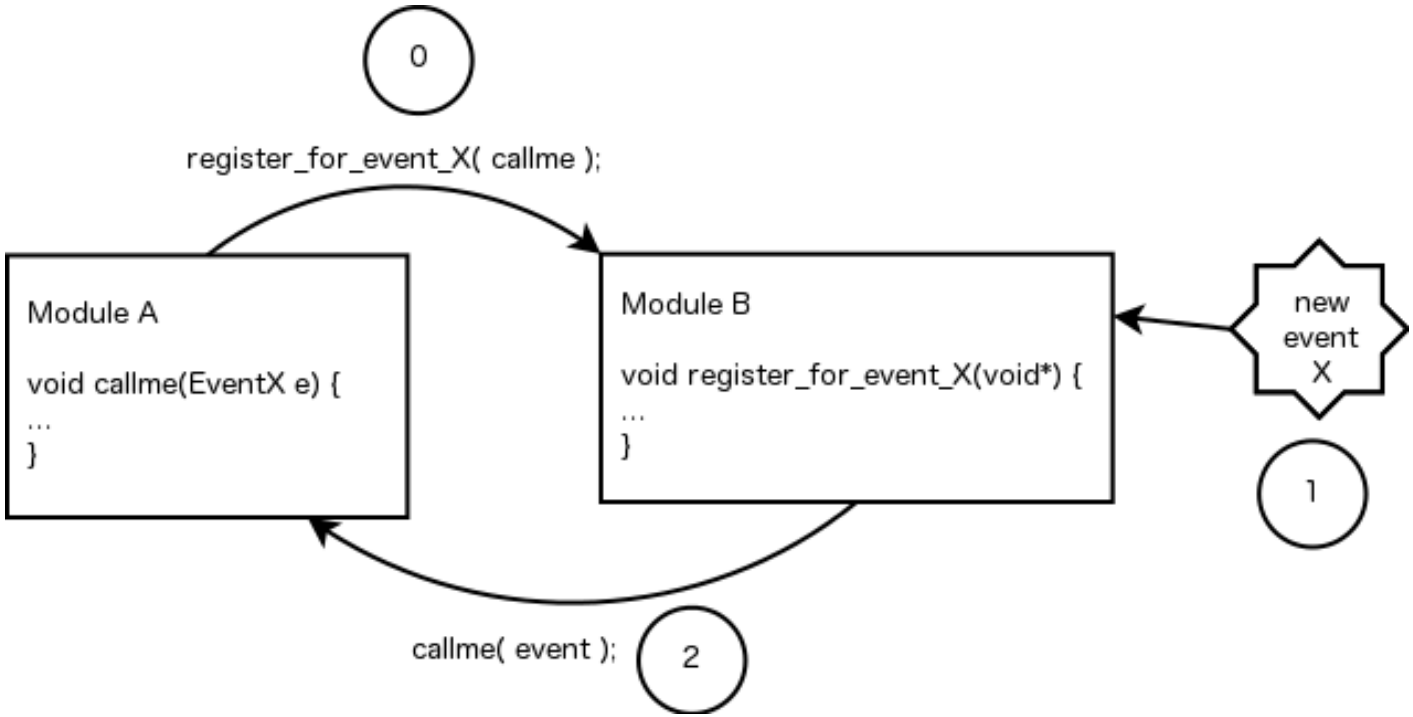
- Every X amount of time, check for events
- If there are no events, the condition of the inner while will be *false*

Polling

```
SDL_Event e;
// main loop
while (!quit) {
    //process all events
    while( SDL_PollEvent(&e) ) {
        switch(e.type) {
            case SDL_KEYDOWN:     // do stuff
            case SDL_KEYUP:       // do stuff
            ...
        }
    }
    // rest of main loop code
}
```

Callbacks

- Register to get called when an event occurs
- We need to provide the *address*(a pointer!) of the function that will be called when the event happens
- More efficient, no need to check every time



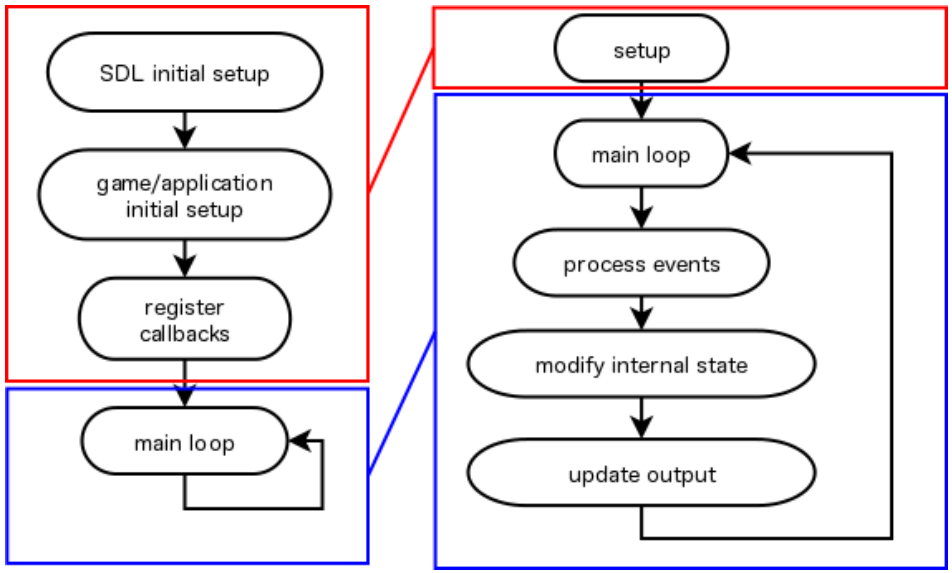
Callbacks (example in SDL)

- Create a timer, and register a callback
- The timer will call us back after the time specified

```
// specific function prototype for SDL
Uint32 callback(Uint32 interval, void* param) {
  // do stuff here
  // return 0 to stop getting called!
  return interval;
}
void main(int argc, char **argv) {
  // call this function every 1000 milliseconds
  SDL_TimerID timer = SDL_AddTimer(1000, callback);
  // continue execution normally
  ...
}
```

Event based system high-level view

- Initialize the system or library (SDL in our case)
- Initialize the game logic
- Register some functions for different events (callbacks)
- Enter in an endless loop
  - 1. (Optionally) Poll the system for additional events
  - 2. Process events
  - 3. Update internal state based on events
  - 4. Update user display



Main loop in a SDL application

```
// includes...
// global variables and function prototypes
int main(int argc, char **argv) {
  int quit = 0;
  SDL_Event event;
  Initialize_SDL(W,H);
  Initialize_Game();
  while(quit!=0){
     PolllEvents();
     ProcessEvents();
     UpdateGameLogic();
     UpdateDisplay();
  }
  FreeGameResources();
  SDL_Quit();
  return 0;
};
```

Game logic vs display

- Clear separation between logic and visual representation
- This approach will simplify the design and development tasks
- Design the logic of the game
  - Identify the key elements in the application/game
- Design the interaction model
  - Keyboard and mouse events.
  - Different states during the game, and transitions between them.
- Visualization of the current state of the game
  - Animations
  - How to display transitions between states
  - This can always be improved, but should not affect the logic.

SDL project setup

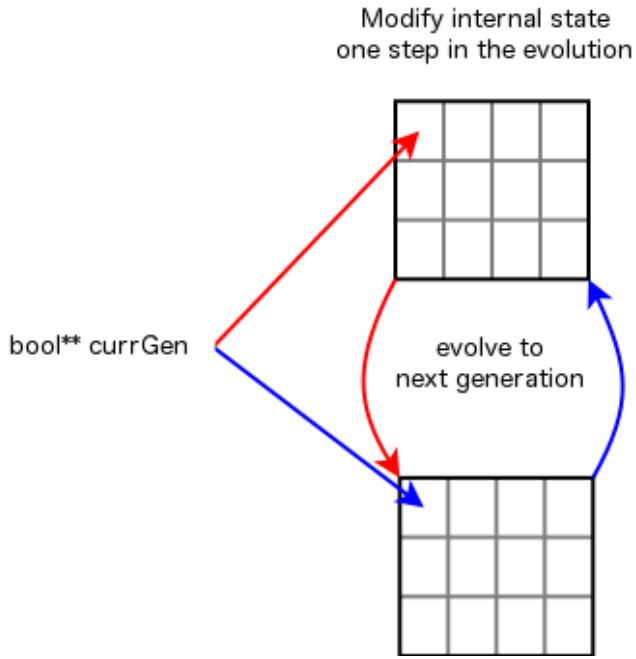
- Download SDL library
  - http://www.libsdl.org/release/SDL-devel-1.2.15-VC.zip
- Unzip the file anywhere within your personal files
- Setup Visual Studio as explained in last week exercises
  - include folder contains the header files
  - lib/x86 folder contains the binary library and dll
  - SDL requires you to declare main as
    - *int main(int argc,char** argv)*
  - Place the SDL.dll file in the same location as your executable
- Demo...

Break

After the break, example project using SDL

Example - Game of Life

- Simulation logic
  - Rectangular group of cells
  - Each cell can be dead or alive.
  - Set of rules to determine the future of each cell based on its neighbors
  - All cells have to change at the same time, from the current state.
  - One step in the evolution will produce a new group of cells based on the previous state.



Example - Game of Life (cont)

```
class Life {
    bool **currGen;
    bool **nextGen;
    void swapGen();
    int neightbors(int x, int y);
public:
    Life(int w,int h);
    ~Life();
    const bool** getCells();
    void evolve();
    ... more functions here
}
```

Example - Game of Life (cont)

- The class Life only provides "read" access to the cells
  - We can use this to read the values and draw on the screen
- The class has to provide a way to modify its cells
  - Add a public function to set an individual cell
  - Add functions to build the basic game patterns
    - Search for patterns in Wikipedia's page for the game
- The game logic, will create one instance of this class Life
  - Set the initial configuration for the cells
  - Start the simulation
  - One iteration on the loop, one step in the evolution
- Update screen after each evolution step

Demo...