

Tedm



Theodore Ahlfeld
Matthew Haigh
David Watkins

The background is white and features several abstract geometric elements. In the top left, there is a large orange ring with a dashed red inner circle, overlapping a solid yellow circle. Below these is a small pink circle. In the bottom left, there is a large lime green circle, a small cyan circle, and a small green circle with a dashed green outline. In the top right, there is a large green circle with a white center, a small orange circle, and a lime green circle with a dashed yellow outline. In the bottom right, there is a large cyan ring and a cyan circle with a dashed blue outline. In the center, there is a large, empty dashed light blue circle.

Background
Opposite of foreground



UNREAL
ENGINE



Game Library Market

- Owned by big players
- Many different language solutions
- Have many different skillsets
- Not Portable
- Most of the time: overkill

A decorative graphic on the left side of the slide featuring several overlapping circles and rings in various colors: a large cyan ring at the top left, an orange circle below it, a yellow ring, a small pink circle, a large lime green circle at the bottom left, and a small green circle. There are also dashed circles in cyan, yellow, and red. On the right side, there are more circles: a cyan ring at the top right, a yellow circle, a small cyan circle, and a small lime green circle, along with a dashed red circle.

Distilling the Market

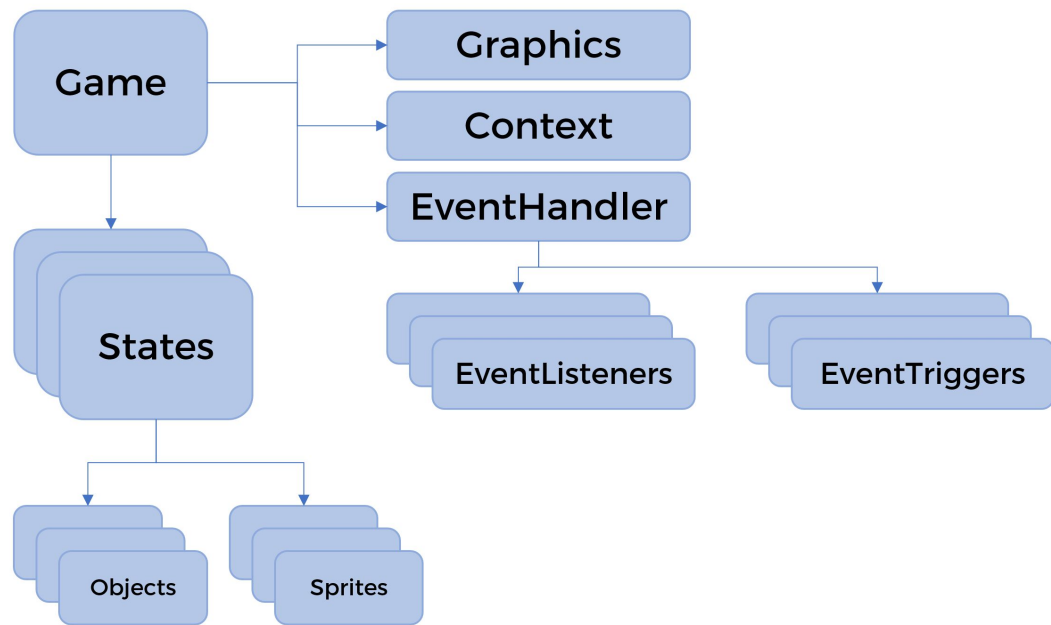
- 2D game genre
- Basic event listeners
- Sprites
- States

The background is white and decorated with various geometric shapes. In the top left, there is a large orange circle with a dashed red outline, overlapping a solid yellow circle. Below them is a small pink circle. In the top center, there is a large, faint dashed blue circle. In the top right, there is a green circle with a white center, a small orange circle, and a lime green circle with a dashed yellow outline. In the bottom left, there is a green circle with a dashed green outline, a large lime green circle, and a small cyan circle. In the bottom right, there is a large cyan circle with a white center, and a cyan circle with a dashed blue outline.

Design Overview

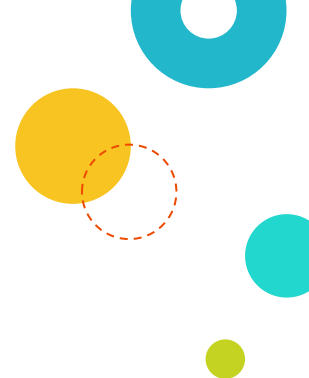
This is where the fun begins

System Overview





Graphics

- Encapsulate SDL renderer
 - All drawable objects use Graphics
 - Hide SDL pointers
- 

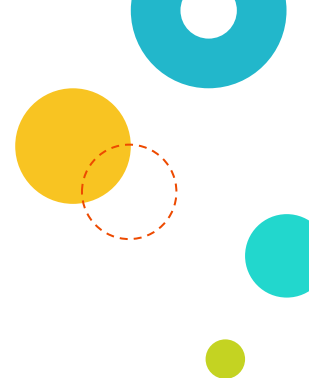
The background of the slide is decorated with various colorful geometric shapes. On the left side, there is a large cyan ring, a solid orange circle, a yellow ring, a small pink circle, a large lime green circle, and a small green circle with a white dot. On the right side, there is a yellow circle, a dashed orange circle, a small cyan circle, and a small lime green circle. In the top right corner, there is a cyan ring. The word "Objects" is written in a light gray font to the right of the large cyan ring.

Objects

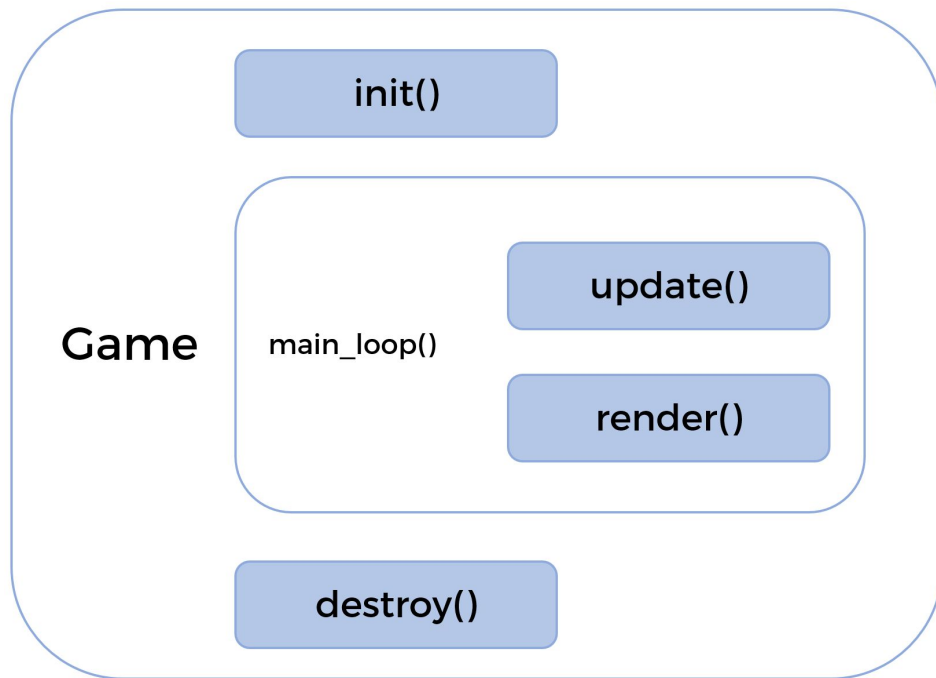
- Manage position on x,y grid
- Able to draw itself
- All actors extend Object

A decorative graphic on the left side of the slide featuring several overlapping circles and rings in various colors: a large cyan ring, an orange circle, a yellow ring, a pink circle, a green circle, and a large lime green circle. Some circles have dashed outlines in matching colors.

Context

- Persistent object
 - Easily extended
 - Set window width or fps
- 
- A decorative graphic on the right side of the slide featuring several overlapping circles and rings in various colors: a yellow circle, a cyan circle, a small lime green circle, and a blue ring. Some circles have dashed outlines in matching colors.

Game Object Life Cycle





Game Object Init

```
//TODO add check for startstate existing
std::shared_ptr<State> currentState = state_id_dict[startStateId];

if(!init()) {
    log.log_error("Initialization failure; aborting execution");
    exit(-1);
} else if(!currentState->init()) {
    log.log_error("User initialization failure; aborting execution");
    destroy();
    exit(-1);
}
```

Game Render/Update

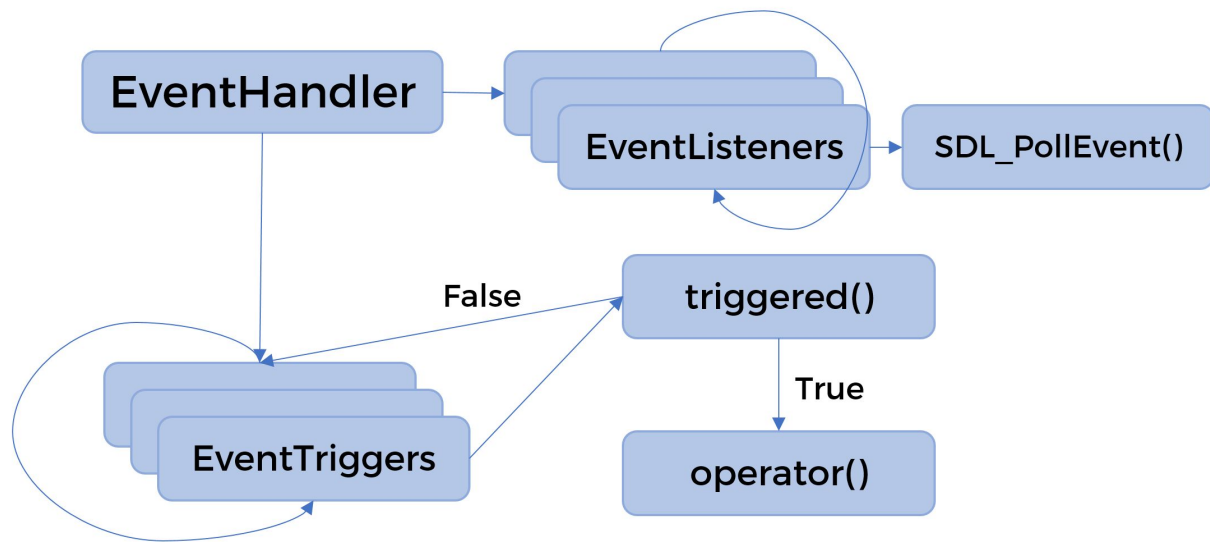
```
// check events
eventHandler.checkListeners();

if(!context.isPaused) {
    // update the scene
    currentState->update();

    // render the scene
    currentState->render();
    //Flush graphics buffer to screen
    graphics.present();

    if( fps.get_ticks() < 1000 / context.targetFramerate ) {
        SDL_Delay( ( 1000 / context.targetFramerate ) - fps.get_ticks() );
    }
}
```

Event Handling





EventHandler Loop

```
void Tedm::EventHandler::checkListeners() {  
    while(event.poll())  
        process();  
  
    std::for_each(_eventTriggers.begin(), _eventTriggers.end(),  
        [&](const std::shared_ptr<EventTrigger> &eventTrigger) {  
            if(eventTrigger->triggered())  
                (*eventTrigger)();  
        })  
    );  
}
```



EventTrigger

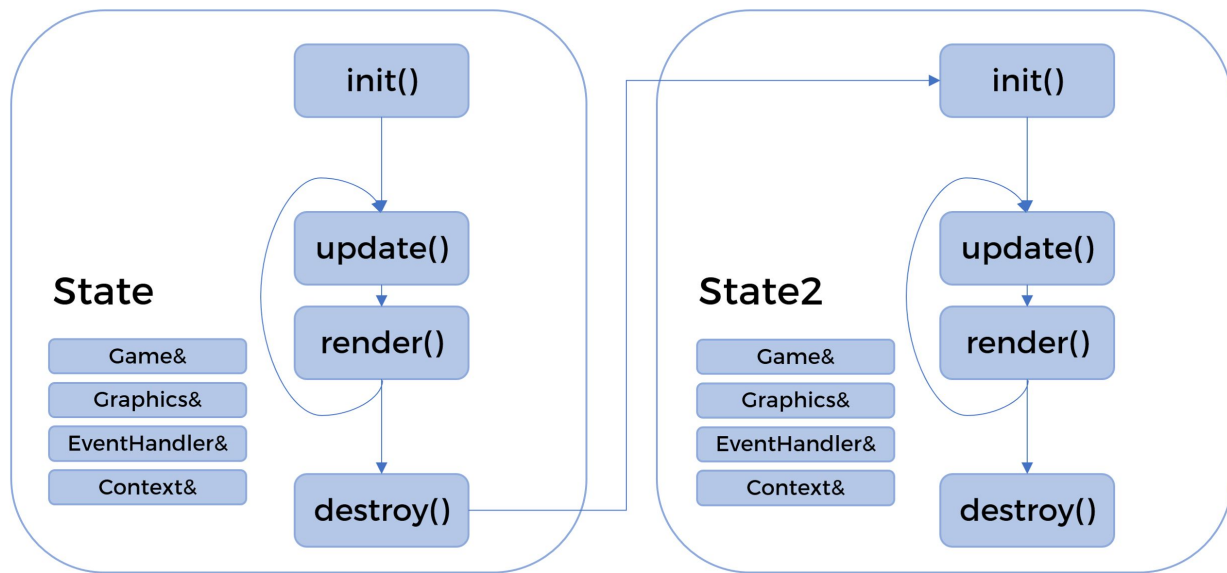
```
class EventTrigger {  
public:  
    virtual bool triggered() = 0;  
    virtual void operator>()() = 0;  
};
```



EventListener

```
class KeyEventListener {  
public:  
    virtual void operator()(SDL_Keycode sym) = 0;  
};
```


State Life Cycle



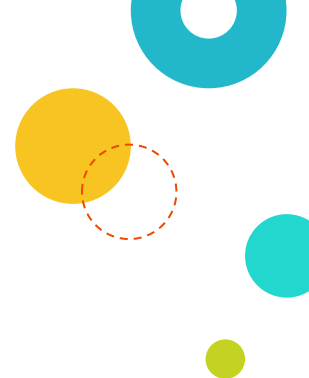
The background is white and decorated with various geometric shapes. In the top left, there is a large orange ring with a dashed red inner circle, overlapping a solid yellow circle. Below these is a small pink circle. In the top center, there is a large, faint dashed blue circle. In the top right, there is a green circle with a white center, a small orange circle, and a lime green circle with a dashed yellow border. In the bottom left, there is a green circle with a dashed green border, a large lime green circle, and a small cyan circle. In the bottom right, there is a large cyan ring and a cyan circle with a dashed blue border.

Build Environment

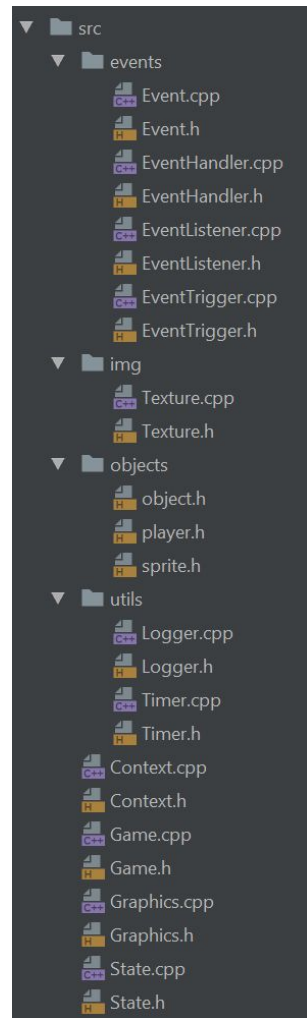
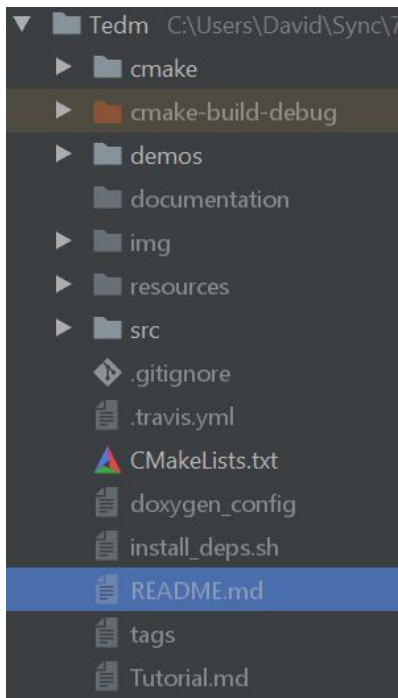
Yakkety Yak



Commonly Used Tools

- Cmake 3.7
 - g++ 6.2
 - Ubuntu 16.10
 - SDL2 and SDL2_image
 - Travis-CI
 - CLion
 - vi
- 

File Structure

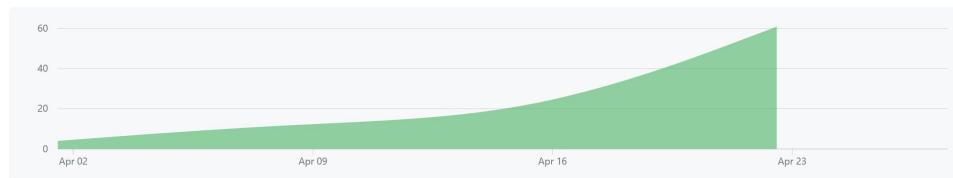


Git Commit History

Apr 2, 2017 – Apr 28, 2017

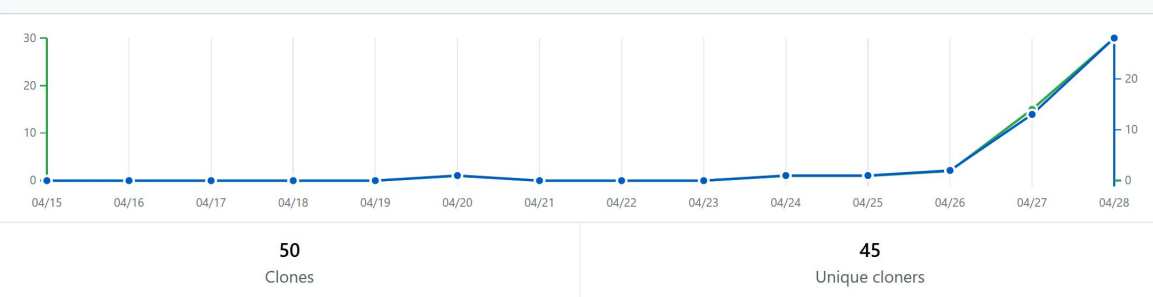
Contributions to master, excluding merge commits

Contributions: **Commits** ▾



Clone Fiasco

Git clones



The background is white and decorated with various geometric shapes. In the top left, there is a large orange circle with a dashed red outline, overlapping a solid yellow circle. Below them is a small pink circle. In the top center, there is a large dashed light blue circle. In the top right, there is a green circle with a white center, a small orange circle, and a lime green circle with a dashed yellow outline. In the bottom left, there is a green circle with a dashed green outline, a large lime green circle, and a small cyan circle. In the bottom right, there is a large cyan circle with a white center, and a cyan circle with a dashed blue outline.

Pong Walkthrough

Hope you like paddling

Creating the Game object

Game constructors

```
class Game {  
    friend class State;  
public:  
    /**  
     * @brief Constructor creates a game, with context and state, at default  
     * screen size  
     */  
    Game();  
  
    /**  
     * @brief Constructor creates a game, with context and state  
     */  
    Game(Context ctx);  
};
```


The State Object

```
class State {  
public:  
    State(Game &game, std::string id);  
    virtual ~State() {};  
    std::string getID() { return id; }  
    virtual bool init() = 0;  
    virtual void update() = 0;  
    virtual void render() = 0;  
    virtual void destroy() = 0;  
    virtual void paused() = 0;  
    virtual void resumed() = 0;
```

The Pong_State Object

```
class Pong_State : public State {
public:
    /* Creating Game objects within the state */
    Player p1, p2;
    Ball ball;

    /**
     * @brief The constructor initializes the paddles and ball.
     * @param game the main Game object
     */
    Pong_State(Game &game) :
        State(game, "pong"),
        p1{Player(graphics, "../resources/blue1.png", 15, 250)},
        p2{Player(graphics, "../resources/blue1.png", 750, 250)},
        ball{Ball(graphics, "../resources/blaster.png", 375, 295, 0, 0)} {}
};
```

The Event listener

```
class Player_KeyBoard_Listener : public KeyEventListener {
    Player *p1, *p2;
public:
    Player_KeyBoard_Listener(Player &p1, Player &p2) {
        this->p1 = &p1;
        this->p2 = &p2;
    }
    void operator()(SDL_Keycode sym) override {
        switch(sym) {
            case SDLK_w:
                p1->move_up();
                break;
            case SDLK_s:
                p1->move_down();
                break;
            case SDLK_UP:
                p2->move_up();
                break;
            case SDLK_DOWN:
                p2->move_down();
                break;
        }
    }
};
```

```
class Quit_Listener : public EventListener {
    bool &isRunning;
public:
    Quit_Listener(bool &b) : isRunning(b) {
    }

    void operator()() override {
        isRunning = false;
    }
};
```

The Pong_State Init

```
bool init() override {  
    game.setWindowTitle("Dat Pong");  
    background = graphics.add_background("../resources/dat_anakin.jpg");  
    eventHandler.addKeyListener(make_shared<Player_KeyBoard_Listener>(  
        Player_KeyBoard_Listener(p1, p2)));  
    eventHandler.addExitListener(make_shared<Quit_Listener>(Quit_Listener(context.isRunning)));  
    context.targetFramerate = 1000;  
    new_round();  
    return true;  
}
```

The Pong_State Update

```
void update() override {  
    ball.update_pos();  
  
    if(ball.collition(p1)) {  
        ball.update_trajectory(p1);  
    } else if (ball.collition(p2)) {  
        ball.update_trajectory(p2);  
    }  
  
    /* Hits Ceil or Floor of the game */  
    if(ball.get_y() <= 0 || ball.get_y() >= context.height-ball.height) {  
        ball.update_trajectory();  
    }  
  
    /* Hits the goal */  
    if(ball.get_x() <= 0 || ball.get_x() >= context.width-ball.width) {  
        new_round();  
    }  
}
```

The Pong_State Render

```
/**  
 * @brief the game will call render after update each frame. This function  
 * draws everything relevant for the current state  
 */  
void render() override {  
    graphics.draw(background);  
    p1.draw();  
    p2.draw();  
    ball.draw();  
}
```



Pong Putting it all together

```
int main(int argc, char*argv[]) {  
    Game pong = Game();  
    Pong_State pong_state(pong);  
    pong.registerState("", make_shared<Pong_State>(pong_state));  
    pong.mainLoop();  
}
```



The background is white and decorated with various geometric shapes. In the top left, there is a large orange circle with a dashed red outline, partially overlapping a yellow circle. Below the yellow circle is a small pink circle. In the top center, there is a large, faint dashed blue circle. In the top right, there is a green circle with a white dot in the center, a small orange circle, and a lime green circle with a dashed yellow outline. In the bottom left, there is a green circle with a dashed green outline, a large lime green circle, and a small cyan circle. In the bottom right, there is a large cyan circle with a white dot in the center, and a cyan circle with a dashed blue outline.

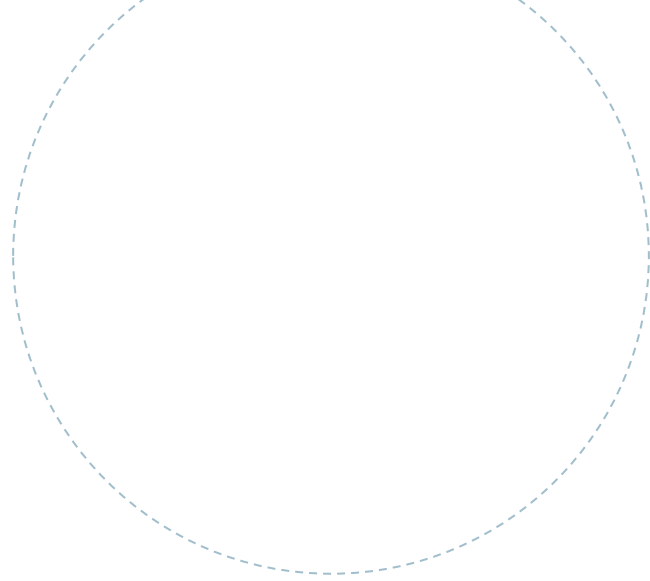
Testing Strategy

Getting testy up in here

A decorative graphic consisting of various colored circles and rings. In the top left, there is a large cyan ring and a smaller solid cyan circle. Below them is an orange circle and a yellow ring. Further down are a pink circle, a yellow dashed circle, and a green circle with a white dot. In the bottom left is a large lime green circle. On the right side, there is a yellow circle, a cyan circle, and a small lime green circle. Some circles have dashed outlines in matching colors.

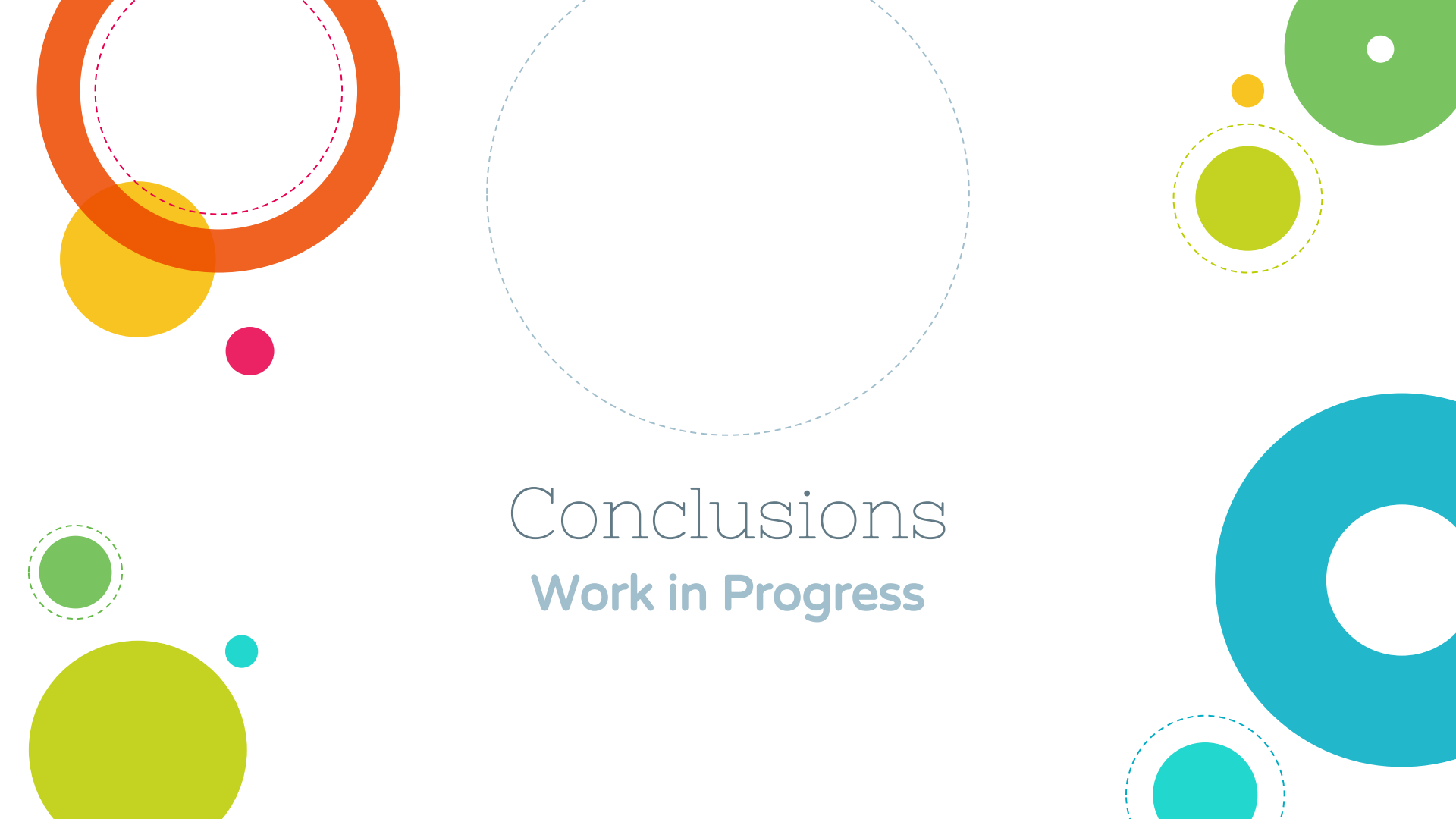
Units Tests in Game Libraries

- Game libraries != normal libraries
- Demos win
- Need lots of demos to affirm



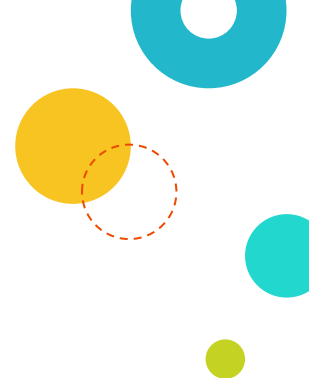
Conclusions

Work in Progress



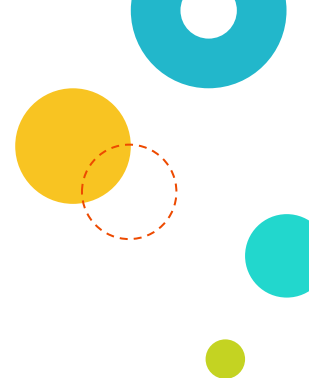


Future Work

- Need to add more interfaces
 - Hide more of SDL from user
 - Create more example games
 - Add multithreading
- 



Future Work

- Add sound
 - Add drawing shapes
 - Add internet plugins
 - Concepts support
- 

A decorative graphic consisting of various colored circles and rings in shades of teal, orange, yellow, pink, and green, scattered across the slide. Some are solid, some are dashed, and some are nested.

Lessons Learned

- Get started early! (not 8am today)
- Game libraries need to be well thought out
- C++ is very useful