CREATING YOUR OWN 2D GAME ENGINE

Examples using SDL2 and C++

Milan Babuškov
milan@bigosaur.com

IndieCade Europe 2016

Motivation

Why?

- For fun
- To learn stuff
- To squeeze the last bit of hardware performance
- It could be faster than learning some full engine
- 2D typically takes about 3 weeks to build, test, debug
- Fix bugs yourself - no need to wait for 3rd party

Who am I?

- 39 years old, programming since I was 9, built a first full game when I was 17
- BASIC on C64
- Assembly on Amiga 500
- C (direct to VGA on PC)
- C++ (DirectX, SDL1.x)
- Java applets, later libGDX
- PHP, Javascript, Node.js, HTML5 on Web
- Currently LibGDX and SDL2/C++

19 game projects started.
18 shipped,
1 in Early Access (45k LOC already)

Game Engine Components

- Managing resources
- Handling input
- Drawing graphics on screen
- Playing music and sound effects
- Menu system
- Storing user settings and game data

Asset Management

- Graphics, Music, Sound effects, Fonts
- Access by name – avoid duplicates
- Handling resource life cycle (load, use, free up)
  - `SDL_CreateTextureFromSurface , SDL_DestroyTexture`
  - `Mix_LoadWAV, Mix_FreeChunk, etc.`
- Multiple AssetManagers
  - grouping resources for levels, cut-scenes, transition screens and releasing all in one batch
- Unified interface to filesystem
  - SDL_GetBasePath()

.

Textures, Images, Animations

- Texture
  - (a dumb wrapper around SDL_Texture)
- Image
  - Refers to a texture
  - Expiry flag to remove temporary one-off textures generated on-the-fly
  - Sprites (game's actors and objects) have 1-n images
  - Holds animation info
- Animations

Animations

- Position
- Rotation
- Scale
- Crop
- Colors/tint
- Transparency
- Internal storage: frame → frame data
- Interpolation algorithm. Linear interpolation example:
  - q = (frame - previous) / (next-prev)
  - value = q * next + (1-q) * prev

Handling audio

- SDL_Mixer for .wav, .ogg, .mp3 files
- Transitions and problem with latency
  - SDL_Music blocks the game when playing starts
- Solutions
  - Use a different library like FMOD, OpenAL
  - Keep songs in RAM, use Mix_PlayChannel
  - Implement your own fade in/out logic

Fade in / Fade out

- Problem:
  - Separate sound effects / music volume
  - Track available channels or ...
  - use Mix_PlayChannel(-1...)
- ChannelVolume array (from silent to full)
- Update channelVolume[] array from channelFade[] array values every tick
- Track which channel plays music
- Set Mix_Volume() to
  - channelVolume[] * global music/sfx volume

Outlined text

Drop Shadow


Play local game
Play network game
Achievements
Settings
Credits
Quit

Fonts and text

- SDL_ttf renders text to a surface (expensive)
- Outline support
  - TTF_SetFontOutline
- Drop shadow (draw dark + regular with offset)
- Covert final surface to texture
- Cache texture for: string + shadow + outline

Text in the game

- Permanent, never changes
- Dynamic, always different
  - Numbers with large spans, chat, etc.
- Dynamic, but repeated often
  - Numbers, phrases
- CachedText
- CachedTextCollection
  - Store a text template (font, shadow, outline, color)
  - CachedText *get(“desired string”)

Camera

- Translate world vs screen coordinates
  - Simple additon/subtraction
- Zoom support
  - Adds multiplication
- Dynamic camera
  - Update() once per tick
  - MoveTo(x,y), CenterOn(x,y)
  - Smooth movement (linear interpolation)
  - Screenshake

Screenshake

- Shake radius (intensity) + angle (random)
- Algorithm does the following on every tick:
  - Reduce shake radius (ex. Radius *= 0.9)
  - Set new random angle
  - Use sin/cos to get x,y offset
  - When drawing code asks for screen position offset all the coordinates

Input handling

- Mouse
- Keyboard
- Controllers
- Network events (remote player actions)
- System events (ex. controller added/removed)
- Use SDL2 event loop and process

Configurable controls

- Instead of mapping key → action
- Use key → internal mapping → action
- Make internal mapping available through menus
- Use the same interface for keyboard and controllers and only give “action codes” to your play code. Actions like iaJump, iaLeft, iaRight, etc.
- Cover common controls
  - ESC to exit on keyboard, B on Xinput controllers
  - ESC to pause on keyboard, Start button on controller, etc.

Allow configuration in-game

Event Handlers

- EventHandler class with abstract methods for events:

```
class EventHandler
{
    virtual void setFocus(bool isFocused) {};
    virtual int draw(Camera &camera) = 0;
    virtual void onMouseWheel(SDL_Event &e) {};
    virtual void onMouseMove(SDL_Event &e) {};
    virtual bool onMouseDownLeft(SDL_Event &e) { return true; };
    virtual bool onMouseDownRight(SDL_Event &e) { return true; };
    virtual void onMouseUpLeft(SDL_Event &e) {};
    virtual void onTextInput(SDL_Event &e) {};
    virtual bool onKeyDown(SDL_Event &e) { return true; };
    virtual void onKeyUp(SDL_Event &e) {};
    virtual bool onJoyButtonDown(Joystick *j, int button) { return true; };
    virtual bool onJoyButtonUp(Joystick *j, int button) { return true; };
    virtual bool onJoyAxis(Joystick *j, int axis, int oldState, int newState) { return true; };
    virtual bool onJoyHat(Joystick *j, int hat, int oldState, int newState) { return true; };
```

Autoconfigure controllers

- Detect with SDL_JoystickGetGUIDString
- Different IDs on Windows, Linux, Mac!
- XBox360 controller is a common denominator
- Game Controller Database:
  - https://raw.githubusercontent.com/gabomdq/SDL_GameControllerDB/master/gamecontrollerdb.txt

Full controller support

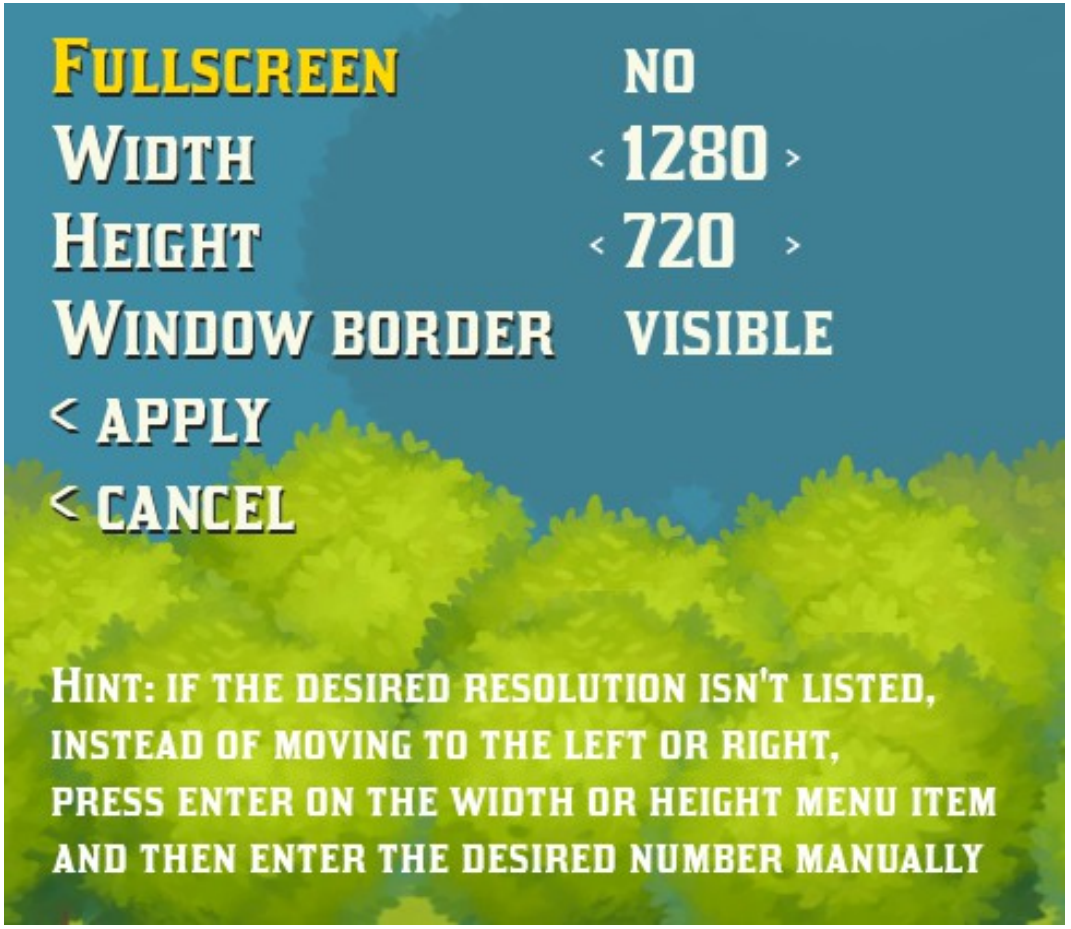
Persistent storage

- Save user settings
- Save game data (stats, saves, etc.)
- Formats: JSON, XML, binary...
- Windows, Linux, Mac are different
- Most systems cannot write to game folder
- Use SDL_GetPrefPath()
  - AppData/Roaming/YourGame on Windows
  - Library/Application Support/YourGame on Mac, etc.

Menu System

- Dedicated EventHandler
- Able to capture keyboard, controllers for configuration
- Mouse support even in non-mouse game (PC users want it)
- Store menus/items in a tree like structure
- MenuItem: text, value, actions (change, increase, etc.)

Menus


FULLSCREEN NO
WIDTH < 1280 >
HEIGHT < 720 >
WINDOW BORDER VISIBLE
< APPLY
< CANCEL
HINT: IF THE DESIRED RESOLUTION ISN'T LISTED,
INSTEAD OF MOVING TO THE LEFT OR RIGHT,
PRESS ENTER ON THE WIDTH OR HEIGHT MENU ITEM
AND THEN ENTER THE DESIRED NUMBER MANUALLY

Questions?