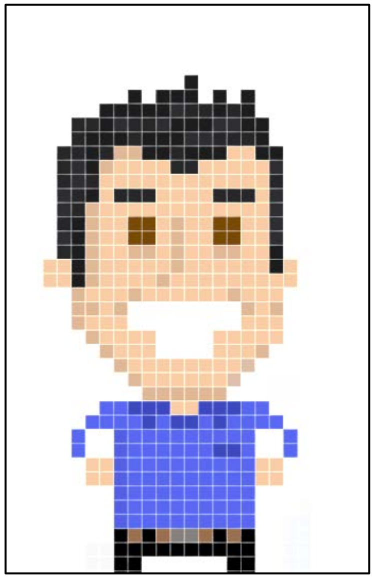


# Framework for Computer Games Industry Application Portfolio

Neil Suttie, Sandy Louchart, Ruth Aylett, Stefano Padilla

# Score?

£1B

Largest Europe

Large franchises

Employs 1K

Tax Relief

## Problem?

Graduates were **struggling to break into the industry**. Computer Science degrees are often cited as a key requirement and often preferred to more targeted Games Programming courses.

Students increasingly required to display knowledge beyond programming through **developing complete games**. Course didn't give the time to develop a portfolio before course completion

"Portfolios of your past work and projects are an essential requirement, otherwise you will find it incredibly difficult to get a job within the games industry." **Game Recruiter**

## Goals

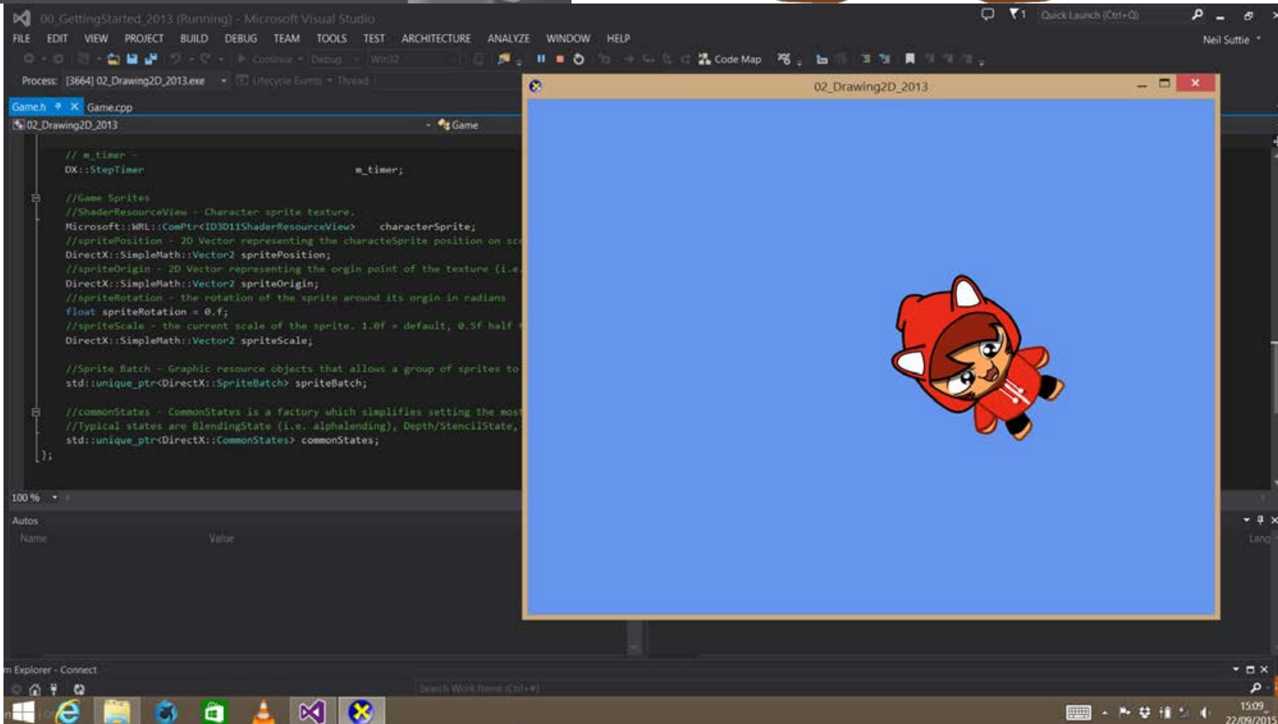
**Modernise course to target industry standard** technologies. Teach industry practice and tools. Expanded materials to cover a wider range of development techniques.

Showcase full development process. Give students the **tools to develop a portfolio** quality app. Create creative common art assets for program sessions

# Progress

- ✓ Consulted industry contacts
- ✓ Developed learning outcomes
- ✓ Developed game framework based on LOs.
- ✓ Help student Video Game Society (game jams, industry sponsorship, knowledge base).
- ✓ Produce materials (graphics, animations, code, …)

A
Exit


00_GettingStarted_2013 (Running) - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Neil Suttie
Code Map
Process: [3664] 02_Drawing2D_2013.exe
Game.h Game.cpp
02_Drawing2D_2013
Game
02_Drawing2D_2013
100 %
Autos
Name Value

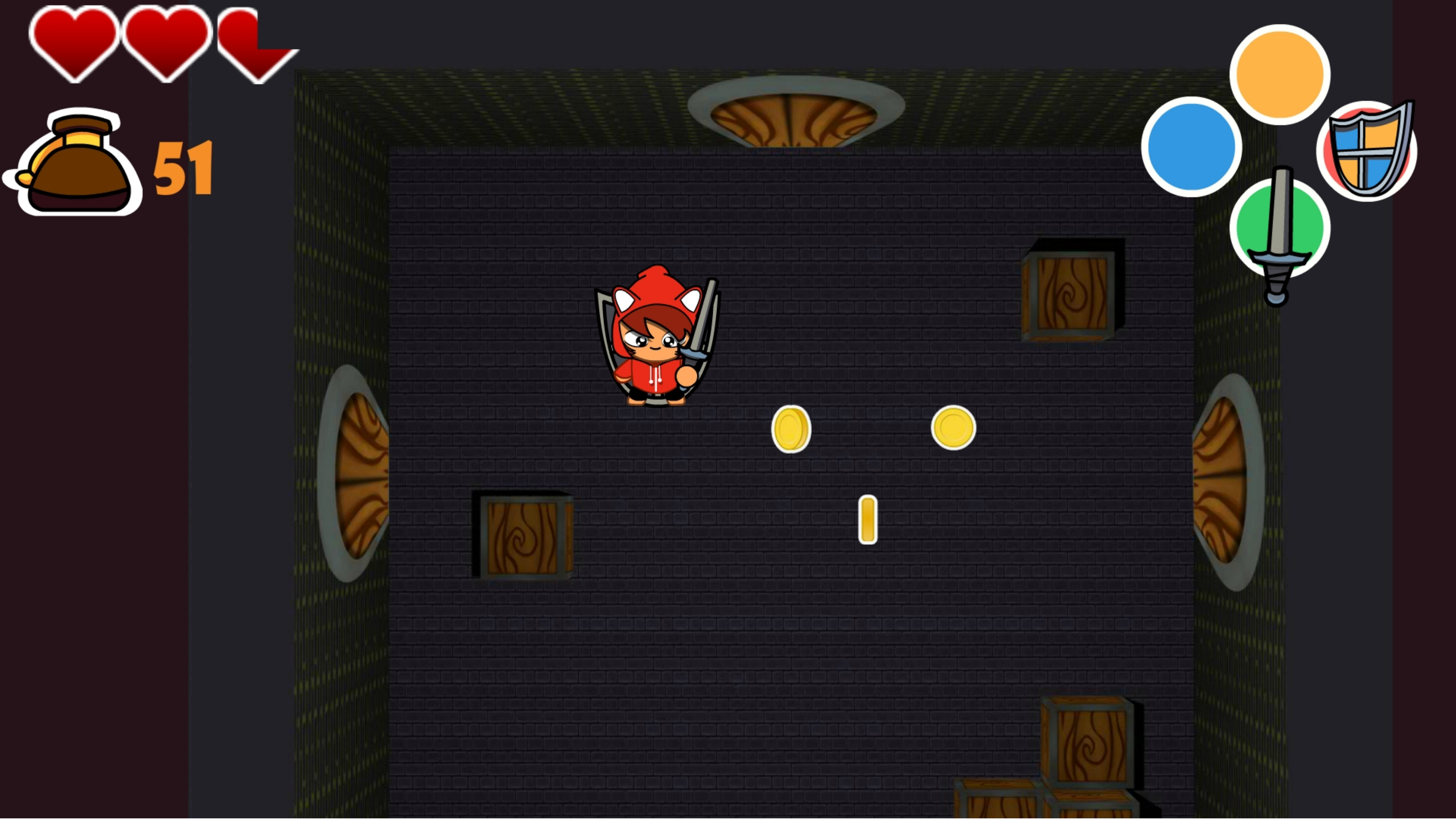
```
    // m_timer -
    DX::StepTimer                                   m_timer;

    //Game Sprites
    //ShaderResourceView - Character sprite texture.
    Microsoft::WRL::ComPtr<ID3D11ShaderResourceView>    characterSprite;
    //spritePosition - 2D Vector representing the characteSprite position on sc
    DirectX::SimpleMath::Vector2 spritePosition;
    //spriteOrigin - 2D Vector representing the orgin point of the texture (i.e.
    DirectX::SimpleMath::Vector2 spriteOrigin;
    //spriteRotation - the rotation of the sprite around its orgin in radians
    float spriteRotation = 0.f;
    //spriteScale - the current scale of the sprite. 1.0f = default, 0.5f half
    DirectX::SimpleMath::Vector2 spriteScale;

    //Sprite Batch - Graphic resource objects that allows a group of sprites to
    std::unique_ptr<DirectX::SpriteBatch> spriteBatch;

    //commonStates - CommonStates is a factory which simplifies setting the most
    //Typical states are BlendingState (i.e. alphalending), Depth/StencilState,
    std::unique_ptr<DirectX::CommonStates> commonStates;
};
```

## LO

- Implement a computer game using industry-standard techniques.
- Apply a knowledge of C++ syntax in the construction of a games application
- Formulate and use 3-D matrices for standard transformations and projections.
- Perform collision detection calculations with circles and squares.
- Fundamental game engine architecture and game loops
- Introduction to Game AI (Pathfinding and decision making)
- Particle effects and procedural content generation
- Memory and memory management in C++

## Materials

- Companion document deployed as PDFs linked to web resources
- Content available through Game Development Society
- Fully commented code samples
- Video examples



# Framework for Computer Games Industry Application Portfolio

Neil Suttie, Sandy Louchart, Ruth Aylett, Stefano Padilla