# Introduction

## Goals of project

Learn programming

Deliver a real application

## Story-based development

Based on agile development

Each story adds some new value to user

Each story kept to the minimum complexity that adds some value

## Learning approach

Programming patterns, tools & techniques introduced only as they are used

Where possible, each is introduced in a context outside the app

The student is then invited to apply this learning, with some hints and guidelines

Model code available at the end of each module

## Technical approach

Web-based

Microsoft tooling platform

TypeScript rather than JavaScript

## Professional perspective

What’s realistic and what’s not

Stories have been nicely ordered

While there is some change to code, this is reduced because we know where we are going.

Watch out for ‘Professional Perspective’

# Create a web page with a welcome screen

## Learnings: patterns & techniques

Use Visual Studio

Create a web-based project

The index page

The run-edit cycle

Some simple Html

## Exercises

## Code written

Index.html

# Create a canvas within which we can draw

Large green square with an outline. Text to the side

## Learnings: patterns & techniques

Canvas

Script files

TypeScript (& JavaScript)

Renderer

Rectangle

Fill & stroke

Arc

Procedural programming - executed in order

Stepping through your code

Global variables (which we want to avoid)

## Code written

Add canvas and script file to Index.html

In script file, add OnLoad function

Get hold of canvas and renderer

Use strokeStyle, fillStyle, rectangle & arc methods

# Create a square in a particular position

## Learnings: patterns & techniques

Separation of concerns - domain logic from presentation

Beginnings of object modelling

Classes and instances

Writing a generalised function

Passing params

## Code written

Separate files for model and drawing

drawSquare function

# Create a whole board

## Learnings: patterns & techniques

Constructor

Loops

Debugging

## Code written

Board object

drawBoard function

# Set up the starting position

## Learnings: patterns & techniques

Enum

Importance of naming & renaming

## Code written

Side enum

Modify square to know if occupied

drawPiece function

# Move a cursor around the board

## Learnings: patterns & techniques

Reacting to events (we already did this with on-load but less consciously)

Using an external library (LoDash)

Using NuGet

Writing automated test

Switch Case statement

Functional programming

## Code written

Separate drawing, eventManagement and model files

Draw a cursor

Function to locate square within board

Unit test for that function

Add instructions into Index page

# Place a piece on the board

# Manage turns

## Learnings: patterns & techniques

Refactoring?

Placing text into a web-page

## Code written

GameManager class

Drawing functions

Status section on Index

# Can only move onto an empty square

## Learnings: patterns & techniques

Rules and constraints

## Code written

Function for is valid move

Changing the colour of the cursor

# New piece must be adjacent to a piece of opposite colour

# Flip any captured pieces

# Move only valid if pieces are captured

# Keep track of piece counts

# Allow move to be skipped

# Identify the end of the game