



## **Module Specification**

### **More Games in C++**

Version: 2023-24, v3.0, 27 Feb 2023

#### **Contents**

<b>Module Specification .....</b>	<b>1</b>
<b>Part 1: Information .....</b>	<b>2</b>
<b>Part 2: Description .....</b>	<b>2</b>
<b>Part 3: Teaching and learning methods .....</b>	<b>4</b>
<b>Part 4: Assessment.....</b>	<b>5</b>
<b>Part 5: Contributes towards .....</b>	<b>6</b>

## Part 1: Information

**Module title:** More Games in C++

**Module code:** UFCFXG-30-2

**Level:** Level 5

**For implementation from:** 2023-24

**UWE credit rating:** 30

**ECTS credit rating:** 15

**Faculty:** Faculty of Environment & Technology

**Department:** FET Dept of Computer Sci & Creative Tech

**Partner institutions:** None

**Field:** Computer Science and Creative Technologies

**Module type:** Module

**Pre-requisites:** Games in C++ 2023-24

**Excluded combinations:** None

**Co-requisites:** None

**Continuing professional development:** No

**Professional, statutory or regulatory body requirements:** None

## Part 2: Description

**Overview:** More Games in C++ is the follow-on module to Games in C++ in which students are to create a selection of more advanced games using the industry standard C++ programming language.

**Features:** Not applicable

**Educational aims:** The aim of this module is to build upon the foundations set out in the 1st year. The games developed in this module will be more advanced and

encourage students to implement a wider range of technical solutions and gameplay mechanics, all whilst retaining creative control of their visions.

**Outline syllabus:** The following outlines a typical syllabus applicable to this module, however, it should be noted that the syllabus may change in response to industry shifts and practices.

The module is designed to expose students to the following topics:

C++ Standard Library:

Rationale for using C++ in Games Development

Modern approaches to managing memory within a C++ code-base

The production of C++ standards compliant code

Development of games that can run across multiple platforms

Defensive code design and their appropriate programming techniques

Templates

Function and Class Templates

Smart Pointers

The C++ template utilities library

Software development:

Cross platform build practices

Professional practices and their application within the games industry

Practical considerations – APIs, IDEs, libraries and SDKs

Compiler directives and representation of language features

Object Orientation: composition vs inheritance

The use of Design Patterns within C++, origins and implementation specifics of particular interest to games development.

Threading and networking:

Threading: Concepts, libraries and implementation approaches

Networking: Concepts, libraries and implementation approaches

STL:

Standard Template Library and its implementation of data structures

Be able to identify the appropriate STL container for a given task

Correctly make use of the various algorithms included within the STL

### **Part 3: Teaching and learning methods**

**Teaching and learning methods:** Students should approach their sessions as an opportunity to work with the module team and to ask questions. Whilst the sessions will be structured, the types of content delivered will be partially driven by the students. A typical session will involve discussion around the the topic of C++ game development accompanied with live coding and time assigned specifically for questions and answers. These sessions are primarily a chance for students to enhance their knowledge to aid them with work on their ongoing portfolios.

As programming is a practical subject matter that requires practice and understanding to obtain competency, students should endeavour to engage and attend these sessions as well as look to make the most of the expert knowledge the team possesses.

Starter code will be provided for their games and any additional API's or frameworks will be introduced where appropriate. At times in order to allow more expansive game design and wider scope, students will be expected to work within small groups. This allows them to develop important communication skills and engage in professional practices that reflect the often communal nature of games development.

**Module Learning outcomes:** On successful completion of this module students will achieve the following learning outcomes.

**MO1** Be able to develop games using recent C++ standards, best practices and appropriate APIs/Frameworks

**MO2** Implement simple threaded and networked applications.

**MO3** Utilise standard template libraries (STLs) and its related algorithms to produce stable and consistent functionality.

**MO4** Design and implement object orientated applications that make appropriate use of mechanisms such as polymorphism and composition

**Hours to be allocated:** 300

**Contact hours:**

Independent study/self-guided study = 228 hours

Face-to-face learning = 72 hours

Total = 300

**Reading list:** The reading list for this module can be accessed at [readinglists.uwe.ac.uk](https://uwe.rl.talis.com/modules/ufcfg-30-2.html) via the following link <https://uwe.rl.talis.com/modules/ufcfg-30-2.html>

## Part 4: Assessment

**Assessment strategy:** A number of individual tasks will be set across the teaching year with assessment culminating in a small group project. Tasks will be summative, though formative feedback is available through in-class discussions and demo sessions. Students will be expected to explore multiple programming techniques to complete their games. Students will be asked to support their submissions with a developer diary.

Resit: Students will be expected to revise and resubmit work undertaken in the term.

**Assessment tasks:**

**Portfolio** (First Sit)

Description: Portfolio of individual worksheets, culminating in a small scale group project, supported with a developer diary

Weighting: 100 %

Final assessment: Yes

Group work: No

Learning outcomes tested: MO1, MO2, MO3, MO4

**Portfolio (Resit)**

Description: Portfolio of individual worksheets, culminating in a small scale group project, supported with a developer diary

Weighting: 100 %

Final assessment: Yes

Group work: No

Learning outcomes tested: MO1, MO2, MO3, MO4

**Part 5: Contributes towards**

This module contributes towards the following programmes of study:

Games Technology [Frenchay] BSc (Hons) 2022-23

Games Technology {Foundation} [Sep][FT][Frenchay][4yrs] BSc (Hons) 2021-22

Games Technology {Foundation} [Sep][SW][Frenchay][5yrs] BSc (Hons) 2021-22