Unit 39: Computer Games Design and Development

Unit code: Y/601/1518

QCF Level 5: BTEC Higher National

Credit value: 15

Aim

To provide learners with an understanding of computer games development as an underpinning technological concept in the fields of computer gaming and systems development.

Unit abstract

It is often easy to forget that behind the polished high-definition graphics and increasingly cinematic content of modern computer games is a highly skilled team of designers and programmers. With more sophisticated environments and new ways of interacting with computers, computer game developers now have the choice to extend into many software development realms.

Linking to any of the programming units, this unit enables learners to use any suitable platform to explore design requirements and methods of user interaction as well as the coding demands required for the differing types of gaming environments. Whilst it is essential to offer learners an overview of the differing gaming environments, unit delivery for the development of a game should focus on one specific environment and the required user and technological interactions.

This unit is not suited to learners who do not have experience in programming and should ideally be delivered when the learner has completed procedural programming, object-oriented programming or event-driven programming.

Learning outcomes

On successful completion of this unit a learner will:

1 Understand computer games development

2 Be able to design computer games

3 Be able to develop computer games

4 Be able to test and document computer games.

Unit content

1 Understand computer games development

Types of computer game: genre eg action, role-play, adventure, strategy, simulation, sports, combat, educational, puzzle, personal development, skills based; development areas eg graphics, Artificial Intelligence (AI), audio, role, scripting; interaction design eg Graphical User Interface (GUI), online, social, integration with media

Platforms: devices eg personal computer, hand held console, stand-alone platform, mobile phone, internet, network, web page

Programming: requirements eg mathematical, simulated physics, GUI components, interface

User control: interaction eg voice, movement, mouse, keyboard, touch screen, floor based, headset, simulated artifact

Impact of gaming: concerns eg time spent, social isolation, cost, separation of reality from actuality, addiction; benefits eg development of thinking, skills development, social interaction, impact on device development, impact on device accessibility

Psychological factors: effects eg use of sound, high score listings, competitive element, peer pressure, fun, educational value, expectations, personal development, skills acquisition

2 Be able to design computer games

Design: tools eg storyboards, pseudo code, narratives, action lists, graphical tools, actor interaction dialogues

Development environment: language eg event driven, object oriented, procedural; considerations eg development facilities, gaming resource offered, library availability, interaction resources, platform compatibility, platform portability

Programming: use of eg data types, conditional statements, control structures, objects, listeners, syntax rules, parameter passing

Program design: considerations eg purpose, modularity, systematic approach, data dictionary, structure charts, flow charts, pseudo code, state diagrams

Units: elements eg functions, procedures, methods, widgets, GUI components, symbols, avatars, characters

Delivery: environments eg desktop, application, mobile app, web based, utility, web based, applet, handheld, console based

3 **Be able to develop computer games**

Implementation: language eg event driven, object oriented, procedural; working application

Programming: use of programming standards; relationship to program design

Coding: use of conventional language commands; material produced is unique; use of library classes

Pre-defined: types eg class library, downloaded, imported, reversion code

Complexity: implementation of user interaction: assurance of user benefit; assurance of use

Components: features eg multimedia, sound, audio, visual, data management, file management

Environment: tools eg games programming software

4 **Be able to test and document computer games**

Mechanisms: procedures eg checking valid declarations, debugging code, checking naming conventions, checking functionality against requirements, error detection, error messages, compiler errors, runtime errors, in code response, dry running

Supportive documentation: test plan; test results; technical documentation eg data dictionary, action charts, action tables, input-process-output tables, class and instance diagrams, data flow diagrams; user guidance; game playing instructions

Feedback: record feedback, eg surveys, questionnaire, interviews; analyze feedback; present results

Testing methods: test strategy eg black box, white box, interface; iterative approach (testing at various stages of development); test plans and test cases; test logs; test evidence; test reports; retests done

Learning outcomes and assessment criteria

Learning outcomes On successful completion of this unit a learner will:	Assessment criteria for pass The learner can:
LO1 Understand computer games development	1.1 critically compare different types of computer games and platforms 1.2 evaluate the characteristics of user interaction 1.3 evaluate the impact of computer-based gaming
LO2 Be able to design computer games	2.1 design a computer game for a given requirement 2.2 identify the components and data and file structures required to develop a computer game 2.3 evaluate alternative designs and solutions to meet a given requirement
LO3 Be able to develop computer games	3.1 implement a computer game to a given design using a suitable programming environment 3.2 implement components to meet design requirements 3.3 implement a game user interface to meet design requirements 3.4 identify and implement opportunities for error handling and reporting
LO4 Be able to test and document computer games	4.1 critically review and test a computer game 4.2 analyse actual test results against expected results to identify discrepancies 4.3 critically evaluate independent feedback on a developed computer game and make recommendations for improvements 4.4 create documentation for the installation, set-up and support for a developed computer game.

Guidance

Links to National Occupational Standards, other BTEC units, other BTEC qualifications and other relevant units and qualifications

The learning outcomes associated with this unit are closely linked with:

Level 3	Level 4	Level 5
Unit 22: Developing Computer Games	Unit 18: Procedural Programming	Unit 40: Distributed Software Applications
Unit 40: Computer Game Design	Unit 19: Object Oriented Programming	Unit 41: Programming in Java
	Unit 20: Event Driven Programming Solutions	Unit 42: Programming in .NET
	Unit 21: Software Applications Testing	
	Unit 23: Mathematics for Software Development	

This unit has links to the Level 4 and Level 5 National Occupational Standards for IT and Telecoms Professionals, particularly the areas of competence of:

- Software Development.

Essential requirements

Whilst some games development systems and programming languages are commercially available, there are also free resources available incorporating an advanced set of gaming oriented features deployed on many platforms. You must ensure that in the case of mobile platforms the applicable free emulators are available or where security policies dictate, local work stations are equipped with virtualised operating systems containing the programming environment.

Learners must have access to facilities which allow them the opportunity to fully evidence all of the criteria of the unit. If this cannot be guaranteed then centres should not attempt to deliver this unit.

The learner must develop a game that has a level of interaction or challenge and may be applied on a range of platforms. Therefore it may be web based, GUI based, games console, mobile deliverable or a range of other platforms.

Implementation must be based on a suitably complex problem that ensures use of multiple actions and user interaction.