Unit 42: Programming in .NET

Unit code: H/601/1537

QCF Level 5: BTEC Higher National

Credit value: 15

Aim

To provide learners with an understanding of the principles of programming using a .NET framework as an underpinning technological concept in the fields of programming and systems development.

Unit abstract

The .NET framework defines a range of reusable class libraries that define the interactions used for Windows operating systems based development of utilities, applications, web based resources, games as well as data integration.

Whilst specific to Microsoft products, the .NET framework defines interactions with servers, workstations and mobile devices. The .NET framework also describes interactions and data exchange with other programming and development systems and is designed to enable cross-platform interaction.

This unit allows learners to become familiar with the underpinning concepts of .NET framework programming, without needing to develop particular skills in one chosen language. Each of the languages has the capacity to develop event driven solutions and it is not important which language is chosen as long as the skills being developed and evidenced relate to the key .NET focus.

The focus of the unit is on developing solutions to meet identified user needs while emphasising the importance of testing and reviewing.

Learning outcomes

On successful completion of this unit a learner will:

1 Understand the principles of programming using a .NET framework.

2 Be able to design .NET solutions

3 Be able to implement .NET solutions

4 Be able to test and document .NET solutions.

Unit content

1 Understand the principles of programming using a .NET framework

Version: current version; backwards compatibility; design considerations; alternative implementations

Design features: interoperability, common runtime engine, language independence, base class library, deployment, security, portability

.NET languages: eg C#, C++, F#, J#, PowerShell, JScript .NET, IronPython, IronRuby, Visual Basic, IronLISP, L#, P#

Architecture: Common Language Infrastructure (CLI), assemblies, metadata, security, class library, memory management; framework versions (architecture) eg 3.5, 3.0, 2.0; common language runtime and the .NET framework class libraries

2 Be able to design .NET solutions

Selection: identification of .NET compatible programming language, identification of .NET programming libraries, selection of development environment

Design methodology: reuse of existing system, adaptation of code, GUI template, graphical interface, design guides, state and interaction diagrams, screen layouts, data storage, event procedures and descriptions

Specification: input, output, processes, user need, purpose

Creation of application: use of development environment; debugging

Delivery environment: mobile, handheld, web based, desktop, dedicated device, server

Interaction: exchange of data, compliance, compatibility, recognition of standards employed, environment

3 Be able to implement .NET solutions

Tools and techniques: use of tool boxes and controls, selection, loops, event handlers, event driven triggers, listeners, objects and object properties, menus, debugging tools

Data: variables, data types, declaring variables, scope of variables, constants

Programming: use of methods, use of 'traditional coding'

Complexity: multiple .NET classes; multiple code elements

4 **Be able to test and document .NET solutions**

Mechanisms: valid declarations; debugging code; comment code; naming conventions; checking functionality against requirements; documentation

Error handling: management of extremes, use of system imposed statements, interaction between .NET classes

Impact testing: range testing, input testing, load testing, system compatibility

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

Documentation: user eg onscreen help to assist users of the programme, pop-ups, help menu, hot-spots; technical eg designs, delivery system, platform, environment, file structures, coding, constraints, documentation for maintenance of programme

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 the principles of programming using a .NET framework	1.1 discuss the principles, characteristics and features of programming using a .NET framework 1.2 critically compare different types of .NET framework architectures 1.3 critically evaluate the components that support the .NET framework
LO2 Be able to design .NET solutions	2.1 design a .NET programming solution to a given problem 2.2 explain the components and data and file structures required to implement a given design 2.3 evaluate potential delivery environments and interaction
LO3 Be able to implement .NET solutions	3.1 implement a .NET programming solution based on a prepared design 3.2 implement event handling using control structures to meet the design algorithms 3.3 identify and implement opportunities for error handling and reporting 3.4 make effective use of an Integrated Development Environment (IDE) including code and screen templates
LO4 Be able to test and document .NET solutions	4.1 critically review and test a .NET programming solution 4.2 analyse actual test results against expected results to identify discrepancies 4.3 evaluate independent feedback on a developed .NET program solution and make recommendations for improvements 4.4 create user documentation for the developed .NET program solution 4.5 create technical documentation for the support and maintenance of a .NET program solution.

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 6: Software Design and Development	Unit 18: Procedural Programming	Unit 39: Computer Games Design and Development
Unit 14: Event Driven Programming	Unit 19: Object Oriented Programming	Unit 40: Distributed Software Applications
Unit 15: Object Oriented Programming	Unit 20: Event Driven Programming Solutions	Unit 41: Programming in Java
Unit 16: Procedural Programming	Unit 21: Software Applications Testing	
	Unit 22: Office Solutions Development	
	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 event driven languages are commercially available, there are also free languages available incorporating an advanced set of .NET features deployed on many platforms. Centres must ensure that in the case of mobile platforms the applicable emulators are available.

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.

Learners must develop an application that may be event driven and work on a range of .NET platforms. It may be web based, GUI based, a games console or a deliverable for a mobile platform, amongst many other solutions.

Resources

Books

Esposito D – *Programming Microsoft ASP.NET MVC* (Microsoft, 2010) ISBN-10: 0735627142

Libert J, Horovitz A – *Programming .NET 3.5* (O'Reilly, 2008) ISBN-10: 059652756X

Lowy J – *Programming .NET Components: Design and Build .NET Applications Using Component-Oriented Programming* (O'Reilly, 2005) ISBN-10: 0596102070

Websites

http://msdn.microsoft.com/en-gb/library/zw4w595w.aspx

www.dotnet-guide.com/

www.programmingtutorials.com/vbnet.aspx

Employer engagement and vocational contexts

Working with a local programming-based organisation or using internet-based open source projects would enhance the learners' experience and offer a relevant vocational context.