



## MCTS: .NET Framework 3.5, Windows Forms Applications, Exam 70-505

### About this Exam

This exam is intended to test a candidate on their ability to create Windows Forms applications on .NET 3.5



Questions that contain code will be presented in either VB or C#. Candidates can select one of these languages when they start the exam.

### Audience Profile

Candidates for this exam work on a team in a development environment that uses Microsoft Visual Studio .NET 2008 and the Microsoft .NET Framework 3.5 to create Windows-based applications. Candidates should have at least one year of experience developing Windows-based applications by using the .NET Framework 2.0 and should be able to demonstrate the following:

- a solid understanding of Windows Forms applications in the context of the .NET Framework 3.5 solution stack
- experience programming against the System.Windows.Forms object model
- experience creating graphical user interface applications
- experience creating data-driven user interfaces (UI)
- experience deploying Windows applications

### Credit Toward Certification

Exam 70-505: TS: Microsoft .NET Framework 3.5, Windows Forms Application Development: counts as credit toward the following certification(s):

MCTS: .NET Framework 3.5, Windows Forms Applications

### How long is this course?

2KO International presents this course as a [five day full time course](#).

We occasionally run this course as a part time course over 5 Saturdays (based on demand)



### **Where does the course take place?**

At our training centre in Greenpoint, Cape Town, South Africa.

Or we can run it in-house at your business, anywhere in the world (requires individualised quotation).



We aim to offer training at the best prices – if you get a cheaper quote than us, send it to us (official quotes only) and we will beat it. Simple!

### **Skills Being Measured**

This exam measures your ability to accomplish the technical tasks listed below. The percentages indicate the relative weight of each major topic area on the exam.

#### **Creating a UI for a Windows Forms Application by Using Standard Controls (13%)**

- Add and configure a Windows Form.  
This objective may include but is not limited to: Add a Windows Form to a project at design time. Configure a Windows Form to control accessibility, appearance, behavior, configuration, data, design, focus, layout, style, and other functionality
- Manage control layout on a Windows Form.  
This objective may include but is not limited to: Group and arrange controls by using the Panel control, GroupBox control, TabControl control, FlowLayoutPanel control, and TableLayoutPanel control
- Add and configure a Windows Forms control.  
This objective may include but is not limited to: Use the integrated development environment (IDE) to add a control to a Windows Form or other container control of a project at design time, add controls to a Windows Form at run time, configure controls on a Windows Form at design time to optimize the UI, modify control properties
- Create and configure menus.  
This objective may include but is not limited to: Create and configure a MenuStrip component on a Windows Form, change the displayed menu structure programmatically, create and configure the ContextMenuStrip component on a Windows Form
- Create event handlers for Windows Forms and controls.  
This objective may include but is not limited to: Manage mouse and keyboard events within Windows Forms applications, create event handlers at run time to respond to system or user events dynamically, connect multiple events to a single event handler



### **Integrating Data in a Windows Forms Application (22%)**

- Implement data-bound controls.  
This objective may include but is not limited to: Use the DataGridView control to display and update the tabular data contained in a data source, use a simple data-bound control to display a single data element on a Windows Form, implement complex data binding to integrate data from multiple sources, navigate forward and backward through records in a DataSet in Windows Forms, define a data source by using a DataConnector component, create data forms by using the Data Form Wizard
- Manage connections and transactions.  
This objective may include but is not limited to: Configure a connection to a database by using the Connection Wizard, configure a connection to a database by using Server Explorer, configure a connection to a database by using the Connection class, connect to a database by using specific database Connection objects, handle exceptions when connecting to a database, perform transactions by using the Transaction object
- Create, add, delete, and edit data in a connected environment.  
This objective may include but is not limited to: Retrieve data by using a DataReader object, build SQL commands in Server Explorer, build SQL commands in code, create parameters for a Command object, perform database operations by using a Command object, retrieve data from a database by using a Command object, perform asynchronous operations by using a Command object
- Query data from data sources by using LINQ  
This objective may include but is not limited to: LINQ to SQL, LINQ to Objects, LINQ to Microsoft ADO.NET, LINQ to XML
- Create, add, delete, and edit data in a disconnected environment.
- This objective may include but is not limited to: Create a DataSet graphically, create a DataSet programmatically, add a DataTable to a DataSet, add a relationship between tables within a DataSet, navigate a relationship between tables, merge DataSet contents, copy DataSet contents, create a typed DataSet, create DataTables, manage data within a DataTable, create and use DataViews, represent data in a DataSet by using XML, use the OleDbDataAdapter object to access an ADO Recordset or Record, generate DataAdapter commands automatically by using the CommandBuilder object, generate DataAdapter commands programmatically, populate a DataSet by using a DataAdapter, update a database by using a DataAdapter, resolve conflicts between a DataSet and a database by using a DataAdapter, respond to changes made to data at the data source by using DataAdapter events, perform batch operations by using DataAdapters
- Manage XML by using the XML Document Object Model (DOM).  
This objective may include but is not limited to: Read XML data into the DOM, modify an XML document by adding and removing nodes, modify nodes, write data in an XML format by using the DOM, handle DOM events



- Read, write, and validate XML by using the XmlReader class and the XmlWriter class.

This objective may include but is not limited to: read XML data, elements, and attributes, read specific elements or attributes, use XmlTextReader, XmlNodeReader, XmlValidatingReader, and XmlWriter classes

### **Implementing Printing and Reporting Functionality in a Windows Forms Application (11%)**

- Manage the print process by using print dialogs.  
This objective may include but is not limited to: Configure print options at run time, change printers attached to a user's computer, configure the PrintPreviewDialog control, set page details for printing by using the PageSetupDialog
- Construct print documents.  
This objective may include but is not limited to: Configure the PrintDocument component, print a text document in a Windows form, print graphics in a Windows form, print a document by using the PrintDialog component, alert users to the completion of a print job
- Enable security features for printing in a Windows Forms application
- Create a customized PrintPreview component.  
This objective may include but is not limited to: setting the Document property to establish the document to be previewed, set Columns and Row properties, set the UseAntiAlias property for smoother text, configure zoom settings, set StartPage property, add custom methods and events to a PrintPreview control

### **Enhancing Usability (13%)**

- Perform drag and drop operations.  
This objective may include but is not limited to: perform drag and drop within an application and across applications, perform drag and drop by using the Treeview control
- Implement globalization and localization for a Windows Forms application.  
This objective may include but is not limited to: work with resource files for localization, determine installed locales
- Implement accessibility features
- Create and configure multiple-document interface (MDI) forms.  
This objective may include but is not limited to: create parent and child forms, identify active child form, send data to an active child form, arrange child forms, create menus for an MDI application
- Create, configure, and customize user assistance controls and components.
- This objective may include but is not limited to: Configure the PropertyGrid component, configure the ProgressBar control, configure StatusStrip, configure ToolTip, configure ErrorProvider, configure HelpProvider controls, configure timer components
- Persist Windows Forms application settings between sessions



### **Implementing Asynchronous Programming Techniques to Improve the User Experience (15%)**

- Manage a background process by using the BackgroundWorker component.  
This objective may include but is not limited to: Run a background process, announce completion of a background process, cancel a background process, report on the progress of a background component, request status of a background component
- Change the appearance of a UI element by using triggers.  
This objective may include but is not limited to: using multiple triggers; using property triggers; using event triggers; using data triggers
- Implement an asynchronous method.  
This objective may include but is not limited to: Create an asynchronous method, create a new process thread, implement advanced asynchronous techniques

### **Deploying Windows Forms Controls (11%)**

- Create a composite Windows Forms control.
- This objective may include but is not limited to: create properties, methods and events, expose properties of constituent controls, create custom dialog boxes, customize a control's paint and render, set visibility at run time, provide a toolbox bitmap
- Create a custom Windows Forms control by inheriting from the control class
- Create an extended control by inheriting from an existing Windows Forms control

### **Configuring and Deploying Applications (15%)**

- Configure the installation of a Windows Forms application by using ClickOnce technology.  
This objective may include but is not limited to: install a Windows Forms application on a client computer, install a Windows Forms application from a server, configure the required permissions of an application
- Install a Windows Presentation Foundation (WPF) browser application by using ClickOnce
- Install a Visual Studio Tools for Office (VSTO) application by using ClickOnce
- Configure and work with Windows Vista User Account Control (UAC) by using ClickOnce deployments
- Create a Windows Forms setup application.  
This objective may include but is not limited to: configure setup project to add icons during setup, set deployment project properties, configure conditional installation based on operating system versions, set appropriate Launch Conditions based on the .NET Framework version, add custom actions to a setup project, add error-handling code to a setup project
- Set appropriate security permissions to deploy the application.  
This objective may include but is not limited to: elevated permissions



## Course Outline 2011/2012

---

- Configure Trusted Application deployments
- Configure security features in an application.  
This objective may include but is not limited to: Configure code access security, configure the application to work with UAC, configure Windows manipulation permissions, configure appropriate file access permissions for the application, control printing security for the application