Exam 98-361
TA: Software Development Fundamentals

Exam Design

The Basics

This is a Microsoft Technology Associate (MTA) exam designed to assess candidates' knowledge of fundamental software development concepts and basic programming skills. MTA is a new certification under the Microsoft Certification Program that validates the foundational knowledge needed to begin building a career in Microsoft technologies. It can also serve as a stepping stone to the Microsoft Certified Technology Specialist exams. Successful candidates for this exam will earn an MTA certification as well as access to benefits of the Microsoft Certification Program.

We are specifying an item pool of 75 unique multiple-choice items, which will be used on 1 form. Approximately 70 percent of the items should be written to the knowledge/comprehension level and about 30 percent to the application level. For more information about cognitive levels, refer to the Cognitive Domain in Bloom's Taxonomy.

Categories in the cognitive domain of Bloom's Taxonomy (Anderson & Krathwohl, 2001)

The following anatomy is required of each knowledge-level item in this exam:

- Question Statement (ex: What should you do?)
- Answer Choices (Preferably Multiple Choice, choose 1, with 3 distracters)
    - Ex: Which of the following is a valid ASP.NET variable name? A. _foo; B. &foo; C. foo#; D. foo 1

The following anatomy is required of each application-level item in this exam:

- Concise scenario, including any constraints/requirements necessary to make distracter answers 100% incorrect
- Goal Statement (You need to…)

- Question Statement (ex: What should you do?)
- Answer Choices (Preferably Multiple Choice, choose 1, with 3 distracters)

Target Audience

Candidates for this exam are seeking to prove core software development skills. Before taking this exam, candidates should have a solid foundational knowledge of the topics outlined in this preparation guide. It is recommended that candidates be familiar with the concepts of and have hands-on experience with the technologies described here either by taking relevant training courses or by working with tutorials and samples available on MSDN® and in Microsoft® Visual Studio®.

Objective Domain

1. Understanding Core Programming

1.1. Understand computer storage and data types.
This objective may include but is not limited to: how a computer stores programs and the instructions in computer memory; memory stacks and heaps; memory size requirements for the various data storage types; numeric data and textual data

1.2. Understand computer decision structures.
This objective may include but is not limited to: various decision structures used in all computer programming languages; If decision structures; multiple decision structures such as If…Else and switch/Select Case; reading flowcharts; decision tables; evaluating expressions

1.3. Identify the appropriate method for handling repetition.
This objective may include but is not limited to: For loops, While loops, Do..While loops, and recursion

1.4. Understand error handling.
This objective may include but is not limited to: structured exception handling

2. Understanding Object-Oriented Programming

2.1. Understand the fundamentals of classes.
This objective may include but is not limited to: properties, methods, events, and constructors; how to create a class; how to use classes in code

2.2. Understand inheritance.
This objective may include but is not limited to: inheriting the functionality of a base class into a derived class

2.3. Understand polymorphism.

This objective may include but is not limited to: extending the functionality in a class after inheriting from a base class; overriding methods in the derived class

2.4. Understand encapsulation.
This objective may include but is not limited to: creating classes that hide their implementation details while still allowing access to the required functionality through the interface; access modifiers

3. Understanding General Software Development

3.1. Understand application life cycle management.
This objective may include but is not limited to: phases of application life cycle management; software testing

3.2. Interpret application specifications.
This objective may include but is not limited to: reading and translating application specifications into prototypes, code, and components

3.3. Understand algorithms and data structures.
This objective may include but is not limited to: arrays, stacks, queues, linked lists, and sorting algorithms; performance implications of various data structures; choosing the right data structure

NOT: algorithm analysis

4. Understanding Web Applications

4.1. Understand Web page development.
This objective may include but is not limited to: HTML, Cascading Style Sheets (CSS), JavaScript

4.2. Understand Microsoft ASP.NET Web application development.
This objective may include but is not limited to: page life cycle; event model; state management; client-side vs. server-side programming

4.3. Understand Web hosting.
This objective may include but is not limited to: creating virtual directories and Web sites, deploying Web applications; understanding the role of Internet Information Services

4.4. Understand Web services.
This objective may include but is not limited to: Web services that will be consumed by client applications; accessing Web services from a client application; SOAP and Web Service Definition Language (WSDL)

5. Understanding Desktop Applications

5.1. Understand Windows® Forms applications.

This objective may include but is not limited to: Windows Forms event model; visual inheritance; UI design; use of Multiple Document Interface(MDI) and Single Document Interface (SDI) applications

5.2. Understand console-based applications.
This objective may include but is not limited to: characteristics and capabilities of console-based applications

5.3. Understand Windows Services.
This objective may include but is not limited to: characteristics and capabilities of Windows Service

6. Understanding Databases

6.1. Understand relational database management systems.
This objective may include but is not limited to: characteristics and capabilities of database products; database design; Entity Relationship Diagrams (ERDs); normalization concepts

6.2. Understand database query methods.
This objective may include but is not limited to: structured query language (SQL), creating and accessing stored procedures, updating data, selecting data

6.3. Understand database connection methods.
This objective may include but is not limited to: connecting to various types of data stores such as flat file; XML file; in-memory object; resource optimization