Microsoft

LESSON 3.2

10754 Microsoft .NET Fundamentals

Understand the Use of Strong Naming

LESSON 3.2

10754 Microsoft .NET Fundamentals

Lesson Overview

Understand the use of strong names.

In this lesson, you will:

- Learn about the components of a strong name.
- Identify the benefits and limitations of strongly named assemblies.
- Sign an assembly with a strong name.
- Learn about the Global Assembly Cache (GAC).

Guiding Questions

1. What are strongly named assemblies?
2. What data comprises a strong name?
3. What is the GAC?

Anticipatory Set

Imagine that each student in the class has his or her own combination padlock. Also imagine that those padlocks are labeled with the owner's name and kept unlocked on a table in the corner of the room, along with several metal boxes that could be locked by a padlock.

- How could you use these objects to send a secure, private message to a classmate?
- What are the limitations and flaws of this type of system?

Strong Names

- The Microsoft® .NET Framework provides the ability to create “strong names” for an assembly to provide a layer of security; they also provide a unique identity for an assembly, which helps prevent name collisions.
- A strong name is a .NET assembly name plus a version number (and other information) that uniquely identifies the assembly.

Strong Name Security

- Strong names ensure that an assembly is authentic—that it has not been modified or replaced by an unauthorized party.
  - If assemblies are not secured, third parties (including people with malicious intentions) could replace an assembly with one of their own creation and compromise the integrity and security of the application.
- Strong names work by using public key cryptography.
  - A public key can be used to verify that the assembly was created by someone who had the corresponding private key.
  - If the private key has not been compromised, you can safely assume that the assembly is legitimate.
- The security of a strongly named assembly is compromised if the assembly references an unsecured assembly; therefore, strongly named assemblies can reference only other strongly named assemblies.

Components of a Strong Name

In the .NET Framework, a strong name consists of five parts:

1. A simple name—typically just the file name (without extension)
2. The version number
3. A public key
4. (Optional) Culture information, such as “en-us” (English – United States) or “fr” (France), for example
5. (Optional) The target processor architecture

Limitations of Strongly Named Security

- The private key must be secure.
  - If the private key is compromised, assemblies can be altered and re-signed; .NET Framework does not have a mechanism for revoking assemblies if the private key is compromised.
- The public key must be authentic.
  - If you receive an altered public key but believe it to be genuine, the security system is compromised.
- Because a strong name includes version information, applications will always try to load the exact version that it was built against.
  - If an application uses version 1.2.0.0 of an assembly, and that assembly is updated to 1.2.0.1, the application must be rebuilt.
  - We will discuss how using the GAC addresses this limitation later in this presentation.

Signing an Assembly

- In Microsoft Visual Studio®, create a strong name from the Signing pane in the Project Properties page.
- After checking Sign The Assembly, select a strongly named key file (or choose New and enter a file name and, optionally, a password).
- Visual Studio will prompt for a password to protect the key file.

By using this process, the key file is created in the same folder as your source code, which is not an ideal situation. If someone gains access to the source code, they will also have access to the public key file.

Delay Signing

- Delay signing offers an alternative to storing the private key with your source code.
- Delay signing allows you to generate a partial signature during development with access only to the public key.
  - The private key can be stored securely away from the code (and away from other developers on the project) and used to apply the final strong name signature just before shipping the finished assembly.

GAC

- Each computer installed with the common language runtime has a machine-wide code cache called the Global Assembly Cache (GAC).
  - The GAC is a system folder, typically C:\Windows\assembly.
- The GAC stores assemblies specifically designated to be shared by several applications on the computer.
  - Share assemblies by installing them into the GAC only when needed.

Advantages of Using the GAC

- Security
  - The system folder used for the GAC typically is protected by administrator rights. When combined with strong naming, using the GAC makes sharing with it secure.
- Version management
  - Because multiple versions of an assembly can all be stored in the GAC, applications built against older versions are still able to execute.
- Central location
  - The GAC provides a centralized location for all shared assemblies.

Ticket Out the Door

- Explain the purpose and components of a strongly named assembly.
- What questions do you have about strongly named assemblies?