



Installing ET Using a Zip File

Introduction

There is now a Binary release of Enterprise Tester available. This release allows you to do an installation by using a "zip" file rather than using the Windows Installer (.MSI file).

Installing Enterprise Tester from the binary release files is recommended for use by purchased users or those trial users who wish to deploy Enterprise Tester into an environment where they cannot configure the IIS Default Website to use the values required by the installer temporarily.

If you are trialing Enterprise Tester, we strongly advise using our MSI based installer which allows you to easily install Enterprise Tester and its pre-requisite components.

Installing Enterprise Tester from the binary zip release involves:

- Ensuring any prerequisite components are installed.
- Unpacking the binary zip file into a directory.
- Configuring the web.config file with the correct database settings.
- Configuring where attachments will be stored.
- Creating an application pool in IIS for Enterprise Tester.
- Creating a new application for Enterprise Tester.
- Accessing Enterprise Tester for the first time, so that ET can automatically run the required database migrations to prepare the database for first use.

Pre Requisites

The Enterprise Tester binary zip release requires the following prerequisite components to be installed:

- Installing all the pre-requisite, users must have the following installed:
 - .Net Framework 3.5 SP1
 - IIS
 - A support database (Sql Server, Oracle, PostgreSQL, MySql)

Advantages of Performing A Zip File Installation

- During upgrades it's easier to revert to the previous version just by backing up/restoring the application directory.
- You can have more than one version of Enterprise Tester installed concurrently (every Enterprise Tester license allows for 1 production installation and 1 development/staging instance of Enterprise Tester to be installed).
- The upgrade or downgrade process does not require any IIS settings to be changed.

Where can I get the Zip File from?

When you download Enterprise Tester there will be an additional file available called: EnterpriseTester-X.X.XXXXX-Bin.zip

What's in the Zip File?

When you unpack the zip file you will find this structure:

 Data	7/06/2011 10:32 p.m.	File folder	
 Web	7/06/2011 10:33 p.m.	File folder	
 changes.txt	7/06/2011 6:32 p.m.	Text Document	60 KB
 install.txt	7/06/2011 4:06 a.m.	Text Document	16 KB
 upgrade.txt	7/06/2011 4:06 a.m.	Text Document	3 KB

The Data and Web folders are the same as those installed into the "Enterprise Tester" directory, when using the existing Windows Installer (.MSI file).

Additionally there are 3 text files available:

- changes.txt
This is the release notes file.
- install.txt
This contains instructions on how to install Enterprise Tester using the zip release.
- upgrade.txt
This contains instructions on how to upgrade an existing Enterprise Tester installation (installed from a previous binary build) with the latest version one

Note: Each time an upgrade is performed users must check the Upgrade text file and follow specific instructions relating to the release.

Install pre-requisites

Prior to installing please ensure you have the following installed:

- .Net Framework 3.5 SP1 installed (<http://msdn.microsoft.com/en-us/netframework/cc378097>)
- IIS Installed (via Windows Add/remove components) including ASP.Net support (<http://msdn.microsoft.com/en-us/library/ms178477.aspx>)
- A supported database (Sql Server, Oracle, PostgreSQL, MySql)

If you do not have a database, we recommend installing Sql Server Express 2008 r2 (<http://www.microsoft.com/express/Database/>)

Unzip the install files

Prior to installing Enterprise Tester you must unpack the zip file into a suitable directory.

If you do not have a suitable tool installed for unzipping files we suggest downloading 7-zip, which is a powerful (and free) zip/unzip tool for windows operating systems. <http://www.7-zip.org/>

Once you have a suitable unzip tool, unpack the binary release file to a suitable directory - if you are unsure, popular locations include:

```
c:\inetpub\wwwroot\Enterprise Tester\  
c:\EnterpriseTester\App\  
c:\Program Files (x86)\Catch Limited\Enterprise Tester\  

```

When the files have been unpacked you will find that you have a structure like this:

 Data	7/06/2011 10:32 p.m.	File folder	
 Web	7/06/2011 10:33 p.m.	File folder	
 changes.txt	7/06/2011 6:32 p.m.	Text Document	60 KB
 install.txt	7/06/2011 4:06 a.m.	Text Document	16 KB
 upgrade.txt	7/06/2011 4:06 a.m.	Text Document	3 KB

Data Folder - MySql Express database is found here

Web Folder - Application files are in this folder

changes.txt - readme file summarizing changes in each version of Enterprise Tester

install.txt - the document you are reading now

Configure the web.config file

Within the web folder you will find the file "web.config" - this is a XML file containing a number of settings which configure settings for the Enterprise Tester application, including what type of database to connect to and it's connection string details.

Open the file using notepad, within it you will need to change 4 settings:

- "migration.providerName"
- Default connection string
- connection.driver_class
- dialect

These are the default setting:

"migration.providerName"

```
<appSettings>
  <add key="migration.providerName"
value="Migrator.Providers.SqlServer.SqlServer2005Dialect" />
</appSettings>
```

Default connection string

```
<add name="Default" connectionString="Data Source=.;Initial
Catalog=EnterpriseTester;Integrated Security=True;MultipleActiveResultSets=true;" />
```

connection.driver_class

```
<add key="connection.driver_class" value="NHibernate.Driver.SqlClientDriver" />
```

dialect

```
<add key="dialect" value="NHibernate.Dialect.MsSql2005Dialect" />
```

Following are the web.config values you will need for each type of database.

Note: We provide example connection strings for each database, but each database supports a number of different connection options - we recommend consulting the <http://connectionstrings.com/> website for listings of the different connection strings you can try.

Sql Server

By default the web.config file comes with the settings necessary to connect to Sql Server or Sql Server Express - all you need to do is provide a connection string.

Your options are:

A. Use the provided database with user-attached connection string.

```
<add name="Default" connectionString="Data
Source=.\SQLEXPRESS;AttachDbFilename=c:\inetpub\wwwroot\EnterpriseTester\Data\Enterpris
eTester.mdf;Initial Catalog=EnterpriseTester;Trusted_Connection=Yes;" />
```

B. Connect to an existing database, using integrated security.

Note: when using integrated security, the database will be connected to using the username configured as the identify for the application pool in IIS.

```
<add name="Default" connectionString="Data Source=.\SQLExpress;Initial Catalog=EnterpriseTester;Integrated Security=True;" />
```

C. Connect to an existing database, using a sql username & password.

```
<add name="Default" connectionString="Server=Server123;Database=EnterpriseTester;User ID=bob;Password=pass123;Trusted_Connection=False;" />
```

Oracle

Configuring Enterprise Tester to use Oracle requires some additional manual steps.

Install the Oracle ODAC/ODP.Net Client libraries for your edition of Oracle, as you normally would.

Next you must determine if your operating system is 32bit or 64bit, and download the appropriate ODAC xcopy release from here (32bit: <http://www.oracle.com/technetwork/database/windows/downloads/index-101290.html> , 64bit: <http://www.oracle.com/technetwork/database/windows/downloads/index-090165.html>)

Download the "XCopy" zip file.

Once downloaded, unzip the file to a directory, and then copy the following files from the zip distribution in the \web\bin\ folder of your Enterprise Tester directory.

```
\ODAC112021Xcopy\instantclient_11_2\oci.dll  
\ODAC112021Xcopy\instantclient_11_2\oranzsbb11.dll  
\ODAC112021Xcopy\instantclient_11_2\oraocci11.dll  
\ODAC112021Xcopy\instantclient_11_2\oraociei11.dll  
\ODAC112021Xcopy\odp.net20\bin\OraOps11w.dll  
\ODAC112021Xcopy\odp.net20\odp.net\bin\2.x\Oracle.DataAccess.dll
```

Once copied, edit the web.config file, locate the end of the config sections element:

```
</sectionGroup>  
</configSections>
```

And insert the following block of xml on the next line:

And insert the following block of xml on the next line after </configSections>

```
<oracle.dataaccess.client>  
  <settings>  
    <add name="DllPath" value="C:\Program Files (x86)\Catch Limited\Enterprise Tester\Web\bin"></add>  
    <add name="FetchSize" value="65536"></add>  
    <add name="PromotableTransaction" value="promotable"></add>  
    <add name="StatementCacheSize" value="10"></add>  
    <add name="TraceFileName" value="c:\temp\odpnet2.log"></add>  
    <add name="TraceLevel" value="0"></add> <!-- 63 -->  
    <add name="TraceOption" value="0"></add> <!-- 1 -->  
  </settings>  
</oracle.dataaccess.client>
```

Updating the DllPath value to point to the \web\bin\ directory of your Enterprise Tester install.

Now change the following values within the web.config:

```
migration.providerName: "Migrator.Providers.Oracle.OracleDialect"  
connection.driver_class: "NHibernate.Driver.OracleDataClientDriver"  
dialect: "NHibernate.Dialect.Oracle10gDialect"
```

And update the connection string to be something like this (this will rely on TNSNames)

```
<add name="Default" connectionString="User ID=SYSTEM;Password=password;Data  
Source=192.168.1.10:1521/orcl"/>
```

Alternatively, you can specify the connection details inline, like this:

```
<add name="Default" connectionString="Data  
Source=(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=192.168.1.10)  
(PORT=1521)))(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=ETPROD)));User  
Id=SYSTEM;Password=password;" />
```

For more examples of Oracle ODP.Net connection strings, please see this site:

<http://connectionstrings.com/oracle#p12>

PostgreSql

Change Enterprise Tester to use these values:

```
migration.providerName: "Migrator.Providers.PostgreSQL.PostgreSQL82Dialect"  
connection.driver_class: "NHibernate.Driver.NpgsqlDriver"  
dialect: "NHibernate.Dialect.PostgreSQL82Dialect"
```

And update the connection string to be like this:

```
<add name="Default" connectionString="Server=192.168.1.10;Port=5432;Database=et-  
prod;User Id=postgres;Password=Password123;Timeout=60;CommandTimeout=120;" />
```

MySql

Change Enterprise Tester to use these values:

```
migration.providerName: "Migrator.Providers.MySql.MySqlDialect"  
connection.driver_class: "NHibernate.Driver.MySqlDataDriver"  
dialect: "EnterpriseTester.Core.Dialects.MySqlDialectEx, EnterpriseTester.Core"
```

And update the connection string to be like this:

```
<add name="Default" connectionString="Server=192.168.1.10;Database=et-  
prod;Uid=etuser;Pwd=password;" />
```

More examples of connection strings can be found here:

<http://connectionstrings.com/mysql#p28>

Attachment configuration (optional)

By default Enterprise Tester will store attachments in the database. This behaviour can be configured, such that attachments are stored on the file system. To do so, open the web.config file and locate the appSettings section:

```
<appSettings>
  <add key="migration.providerName"
value="Migrator.Providers.SqlServer.SqlServer2005Dialect" />
  ...
</appSettings>
```

Now after the last "add" entry, append these settings:

```
<add key="attachment.storage.garbagecollection.dwell" value="5" />
<add key="attachment.storage.garbagecollection.disable" value="false" />
<add key="attachment.storage.method" value="FileSystemCas" />
```

This will then begin storing attachments in the default location of ../Data/Attachments/

Please consult the Enterprise Tester Installation guide for more details (section 6.5.3 Storing Attachments in a File System).

Configuring an application pool

Before configuring an application pool for Enterprise Tester to use, you should give some thought to the application pools identity.

By default in IIS6 application pools are configured with the identity "Network Service" or "Local System", under IIS7/7.5 they are created with "ApplicationPoolIdentity" by default, but can also default to "Local System", "Local Service" or "Network Service".

If this service is a member of an Active Directory domain we recommend creating a specific domain service account for use by Enterprise Tester, this will be useful when configured network filesystem access for the Enterprise Tester and makes it easier to identify Enterprise Tester and give it permissions to a database when using integrated security.

IIS7/7.5:

- * Go to Control Panel -> Administrative Tools
- * Launch IIS Administrator
- * Expand the server node.
- * Click on "Application Pools"
- * Click on "Add Application Pool..." under Actions.
- * Give the new application pool a descriptive name such as "Enterprise Tester".
- * Select ".Net Framework v2.0.50727" from the .Net Framework version list.
- * Select "Integrated" for the managed pipeline mode.
- * Click OK to create the new application pool.
- * Right click on the new application pool and select "Advanced Settings.."
- * Under the "Process Model" section, change the Identity to your service account (if you have decided to use one).
- * (Optional) - If you are using a 64bit operating system, and plan on using Enterprise Architect integration with .EAP files, you will need to change the option "Enable 32-Bit Applications" to true. Note: This will not be possible if you are using the Oracle 64bit ODP.Net drivers.

IIS6:

- * Go to Start -> Administrative Tools -> Internet Information Services (IIS) Manager
- * Expand the "Web Service Extension" node.
- * Click on the "ASP.NET v2.0.50727" Node.
- * Click the "Allow" button, if it's not already enabled.
- * Expand the "Application Pools" node.
- * Right click on "Application Pools" and select New -> Application Pool.
- * Give the Application Pool a descriptive ID such as "Enterprise Tester"
- * Right click on the new Application pool and select "Properties"
- * Move the to "Identity" tab and change the application pool identity to a configurable account, specifying the username and password of your domain service account.
- * Click OK to save the changes.

At this point you may find it useful to consult the timeout's mini-guide (which can be downloaded from the catch website) and configure the timeout/recycling settings for the application pool, a common complaint from customers is that the timeouts are too short by default in IIS, so changing these to longer values is advisable.

Creating a new application for IIS

Next you will need to create a new application in IIS for Enterprise Tester.

IIS7:

- * Within IIS Manager, expand the "Sites" node.
- * Expand the "Default Web Site" node.
- * Right click and select "Add Application".
- * Enter an alias for the site we suggest "EnterpriseTester"
- * Click the "Select..." button next to the application pool, and select the application pool you created in the previous step.
- * Click the elipsis "..." next to the Physical Path, and browser to the "Web" folder of your Enterprise Tester install.
- * Click "OK" to create the new application.

IIS6:

- * Within IIS Manager, expand the "Web Sites" node.
- * Expand the "Default Web Site" node.
- * Right click and select "New -> Virtual Directory"
- * Click "Next"
- * Enter an alias for the virtual directory, we suggest "EnterpriseTester".
- * Click "Next"
- * Click "Browse.." and select the path to the "Web" folder of your Enterprise Tester install.
- * Click "Next"
- * Select "Read" and "Run scripts" check boxes.
- * Click "Next"
- * Right click on the new site and select "Properties..."
- * Select the "Virtual Directory" tab and click "Create"
- * Select the application pool you created in the previous step from the "Application pool:" drop down list.
- * Click the "Configuration" button.
- * Under "wildcard application maps" click "Insert..."
- * Click "Browse..."
- * Browse to "C:\Windows\Microsoft.Net\Framework\v2.0.50727"
- * Select "aspnet_isapi.dll" and click "Open"
- * Uncheck "Verify that file exists"
- * Click "OK" to save the mapping.

- * Under "Application extensions" click the "Add..." button.
- * Click "Browse..."
- * Browse to "C:\Windows\Microsoft.Net\Framework\v2.0.50727"
- * Select "aspnet_isapi.dll" and click "Open"
- * Enter the extension ".rails"
- * Check "Script engine".
- * Uncheck "Verify that file exists".
- * Click "OK"
- * On the properties dialog for the new site, now switch to the "ASP.Net" tab.
- * From the "ASP.Net" version drop down ensure "2.0.50727" is selected.
- * Click OK to close the dialog and save changes.

You are now ready to launch Enterprise Tester for the first time.

Launching Enterprise Tester for the first time

From a browser on the same server as Enterprise Tester, access the website - by default this will probably be:

<http://localhost/EnterpriseTester/>

Or use whichever Url is appropriate for the Default Web site settings that are in place.

You will experience a long delay (normally between 1 and 2 minutes) while Enterprise Tester creates the necessary tables and populates them in the target database, after which you will see a "Step 1 of 3" wizard page displayed, prompting you to create a new organisation.

At this stage you can continue with the installation and setup of Enterprise Tester as per the installation guide available for download from the Enterprise Tester website:

<http://www.catchlimited.com/index.php?page=enterprise-tester>

Refer to Section 3.3 "Initial Configuration" for more details.