NetQuarry, Inc.

Training

100 – Basics and Infrastructure

Resources

FogBugz	http://support.netquarry.com
Help	http://help.netquarry.com
Wiki	http://wiki.netquarry.com
Latest Install	https://dev.netquarry.com/install/latest (password protected)

Using FogBugz

Creating Issues

If you need to create an issue to report to us, or even to assign internally we always like to see the following information gathered and displayed in the issue.

- One or more screenshots of the problem if appropriate. Download and install the FogBugz screenshot taker from here: http://support.netquarry.com/default.asp?pg=pgScreenshotDownload to have a convenient way of adding screenshots to create a new issue, or to add to an existing issue.
- An optional devlog of the reproduction steps. (truncate first)
- A set of reproduction steps including where you logged on (production or QA), who you logged on as (any user, client user, specific logon), what you clicked, what you entered.
- If necessary a description of how the feature should work with the logic clearly stated.
- The correct release target for the issue

Closing Issues

If you have an issue assigned to you and you have fixed it and checked it in, we always like to see.

- A check in comment referencing the bug number (more than simply BugzID: 222 – Fixed. A better example is BugzID: 3757 - Added Workflow and Order Templates to default settings when creating company
- A comment in the issue regarding what you did to fix it and any instructions regarding change of functionality or how to confirm the fix has been made. When an issue is marked to Fixed, the issue is automatically assigned to the person who opened the issue to verify. Mostly that’s ok, but sometimes you may want to remember to assign the fixed issue to Alex for verification.
- The build number when the fix is included and ready to test. (a build occurs at 4am, 8am, 10pm Pacific Time every day and it takes approximately 25 minutes to complete). Check the build numbers from the build notification emails and choose the next incremental build number.

Schema Management

Schema Scripts

1)	Customers\<CustomerName>\ InitialConversionScript.sql	Run only once
2)	NetQuarry\Scripts\nq_op_schema.sql	idempotent
3)	Customers\<CustomerName>\dbname_data-alter-schema.sql	idempotent
4)	Customers \<CustomerName>\ dbname_data-reindexing.sql	idempotent
5)	Customers \<CustomerName>\ dbname_data-views.sql	idempotent
6)	Customers \<CustomerName>\ dbname_data-procs.sql	idempotent
7)	Customers \<CustomerName>\ dbname_data-migration.sql	idempotent
8)	Customers \<CustomerName>\ dbname_meta-fixup.sql	idempotent

The first script is optional for any implementation and is a script that's used for example to drop a number of obsolete schema objects, and/or perform any preparatory migration requirements. This script will only ever be run once against an operational database.

The other scripts can be (and are) run any number of times against the database. The scripts are constructed in a way that they verify the schema is in a certain condition before executing any necessary changes. If the schema is identical, then no changes occur. This is called being idempotent.

nq_op_schema.sql

This file is installed from the build and contains scripts to add NetQuarry operational schema tables such as preferences, auditing.

dbname_data-alter-schema.sql

Contains all the schema changes for tables and columns in the database. Any tables and columns that need to dropped, modified or created should be scripted in this file. If you are adding columns that require indexing and constraints, you would script those changes here.

dbname_data-reindexing.sql

Contains all schema changes for indexes that are not associated with adding or dropping a column or table.

dbname_data-views.sql

Contains all schema changes for views. Every view in the file has two scripts associated with it. A drop and a create.

dbname_data-procs.sql

Contains all the schema changes for stored procedures , functions or triggers.

dbname_data-migration.sql

Contains all the data manipulation scripts for operational data and metadata.

dbname_meta-fixup-migration.sql

Contains all the manipulation scripts for manipulating metadata specific to development or production environment. It is sometimes necessary to manipulate the metadata after it has been installed, so that the application can function correctly in every deployment location. For example, the location of shared files in a production web farm is different to the location of shared files in a QA environment and a development environment. In this script you would write statements to modify the appropriate metadata settings for when running in a specific environment.

Schema Scripts Do's and Dont's

- First and foremost, the schema script has to be idempotent.
    - There is a Wiki page that contains many examples of how to make idempotent schema changes for various types of schema object. We probably have examples of 99% of the types of schema changes you'll need to make either on the wiki, or in the script files themselves.
- The schema change must go in the right file. If in doubt, ask
- Use the SQL Management Studio to generate your scripts at all times. Hand rolled scripting is buggy.
    - In Tools, Options, Scripting. Do the following
        - Include IF NOT EXISTS clause – **true**
        - Script Extended Properties – **false**
        - Include Collation - **false**
        - Feel free to set any other options that make sense
- Adding string/text columns to table they MUST NOT INCLUDE collation
    - cross collation joins are a nightmare to manage
    - collation fixes are difficult to make
- Don't add sort clauses in views. They can be defined in your metadata.
- Always comment out lines of script with --, rather than blocks of /* */. The oSQL parser that runs the scripts during the install cannot handle blocked comments.

Metadata Management

How to Modify Metadata Safely

Before starting on making a metadata change, get all the latest database scripts and metadata.

- SOS – Customers\<Customer>\Database, Customers\<Customer>\Database\Metadata
- Run the database scripts to ensure you have the latest schema
- Load all the latest metadata
- Open NQ Studio
  - or close all studio windows
  - or press F5 to refresh each window individually

Make your changes in the studio

After you complete your metadata changes, save and check in

- Check in any view changes
- In NQ Studio, go to Modules/Change Modules page.
  - review the list of changed modules
  - are they consistent with what you have changed?
  - sometimes the list contains unexpected changed entries
  - Make sure all modules that are changed are checked out
  - Go to Applications, click on the row that your changed modules belong to, right click and save all changed modules
- If some modules are modified unexpectedly, verify that they are valid changes.
- If some modules are modified and are not correct, individually save correct modules from changed module page.

After you finish saving your changes, DIFF the files in SOS.

What not to do with Metadata

- Check in without first DIFFING changes.
- Manually edit the metadata file

- Work on/check in meta at the same time as another person has the file checked out (if meta is checked out contact the other person). Meta files are much harder to merge than code files. (of course can be exceptions)

How to Update Schema and Meta

When you start development of your application you will want a convenient way to keep your schema and metadata up to date. The most convenient way to achieve this is to have a batch file containing the appropriate scripting commands in the right order.

You would save the batch file in Customers\<Customer>\Database folder. For example, update_database.bat

The batch file would contain the following commands…

```
ECHO *************************************************************
ECHO **** UPDATING DBNAME_DATA SYSOP SCHEMA                   ****
ECHO *************************************************************
          pushd C:\Inetpub\wwwroot\NetQuarry\Database\Scripts
                   call osql -S . -E -d {{dbname_data}} -i nq_op_schema.sql -n
          popd

ECHO.
ECHO *************************************************************
ECHO **** UPDATING DBNAME_META SYSOP SCHEMA                   ****
ECHO *************************************************************
          pushd C:\Inetpub\wwwroot\NetQuarry\Database\Scripts
                   call osql -S . -E -d {{dbname_meta}} -i nq_op_schema.sql -n
          popd
ECHO.
ECHO *************************************************************
ECHO *** UPDATING DBNAME_META SYSTEM SCHEMA                   ****
ECHO *************************************************************
          pushd C:\Inetpub\wwwroot\NetQuarry\Database\Scripts
                   call osql -S . -E -d {{dbname_meta}} -i meta-schema.sql -n
                   call osql -S . -E -d {{dbname_meta}} -i system-data.sql -n
          popd
ECHO.
ECHO *************************************************************
ECHO ***** UPDATING {{DBNAME_DATA}} OP SCHEMA                   ******
ECHO *************************************************************
          pushd C:\NetQuarry\Customers\<CustomerName>\Database
                   call osql -S . -E -d {{dbname_data}} -i {{dbname_data}}-alter-schema.sql -n
                   call osql -S . -E -d {{dbname_data}} -i {{dbname_data}}-reindexing.sql -n
                   call osql -S . -E -d {{dbname_data}} -i {{dbname_data}}-views.sql -n
                   call osql -S . -E -d {{dbname_data}} -i {{dbname_data}}-procs.sql -n
                   call osql -S . -E -d {{dbname_data}} -i {{dbname_data}}-migration.sql -n
          popd
ECHO.
ECHO *************************************************************
ECHO **** UPDATING METADATA IN {{DBNAME_META}}                   ****
ECHO *************************************************************
          pushd C:\NetQuarry\Customers\<CustomerName>\Database\Metadata
                   call load-meta.bat
          popd
ECHO.
ECHO *************************************************************
ECHO ***** {{DBNAME_META}} ENVIRONMENT SPECIFIC SETTINGS        ******
ECHO *************************************************************
          pushd C:\NetQuarry\Customers\<CustomerName>\Database
                   call osql -S . -E -d {{dbname_meta}} -i {{dbname_meta}}-fixup.sql -n
          popd
pause
```

Development Process

1. New features that are other than trivial will be controlled by versioning of meta-data and or code such that the feature will not be manifest in production until that feature is explicitly versioned in. The NetQuarry platform provides integrated versioning functionality, called "named versions", to support this for both code and meta-data. We have used this in the past for significant features, but will now do so on a much more comprehensive basis.

2. Bug fixes amenable to versioning will use the same versioning mechanisms and release process.

3. Versioned changes will be versioned-out in both test and production environments until the developer believes that the feature/fix has been developed to the point where feedback and/or verification by QA is required. At that point the feature/fix will be enabled ONLY ON QA using a SQL script statement to version-in the feature/fix for the test environment only. The SQL script for this will be typically located in the dbname_meta-fixup.sql script file. Typically at this point the associated issue(s) will be marked as Resolved and handed over to QA.

4. Until the feature/fix issue(s) are CLOSED by QA, the feature/fix will not be promoted to production. This is a break with our process in the past where we allowed some features/fixes to be promoted to production once Resolved if the developer felt it was safe.

5. Once a feature/fix is CLOSED by QA, the developer will update the meta-data to version-in the feature/fix and check in that change. No other changes are to be performed to the code-base with regard to that feature/fix.

6. Test builds will continue on a four-times-a-day schedule. The 10:30 pm Pacific build is considered the preferred build for testing unless otherwise specified.

Build and Install

Getting New Features Into Build and Installer

New MetaData Modules

Send us an email so we can add to the install process.

New Code Modules

Send us an email so we can add to the install process.

New Scheduled Tasks

Should be set to disabled and turned on for QA site when ready to test (dbname_meta-fixup.sql)

New Features (version controlled)

Should be set to old version and set to latest version for QA site when ready to test (dbname_meta-fixup.sql)

Production Install

The production install process is intended to be a very simple task. Fundamentally it's as simple as backup up the old installer, downloading the new installer and then running a batch file.

How To Install

To perform an install, you logon to the database server.

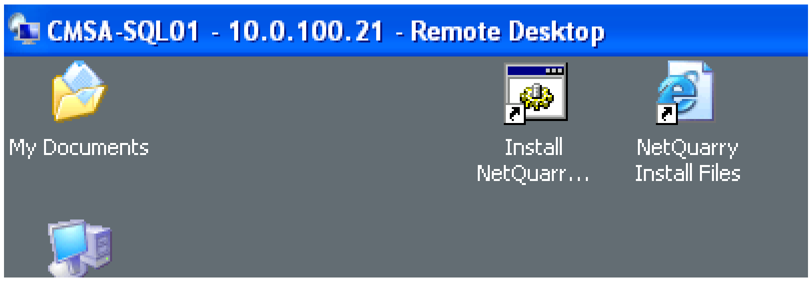
Backup the previous installation files

First step is to open Windows Explorer and go to the C:\NetQuarry folder.

Copy the two installer packages, NetQuarry.msi and cnet.msi into the C:\NetQuarry\ROLLBACK folder.

Download new installation files

There are two shortcuts on the desktop, for all users, to assist with the installation process

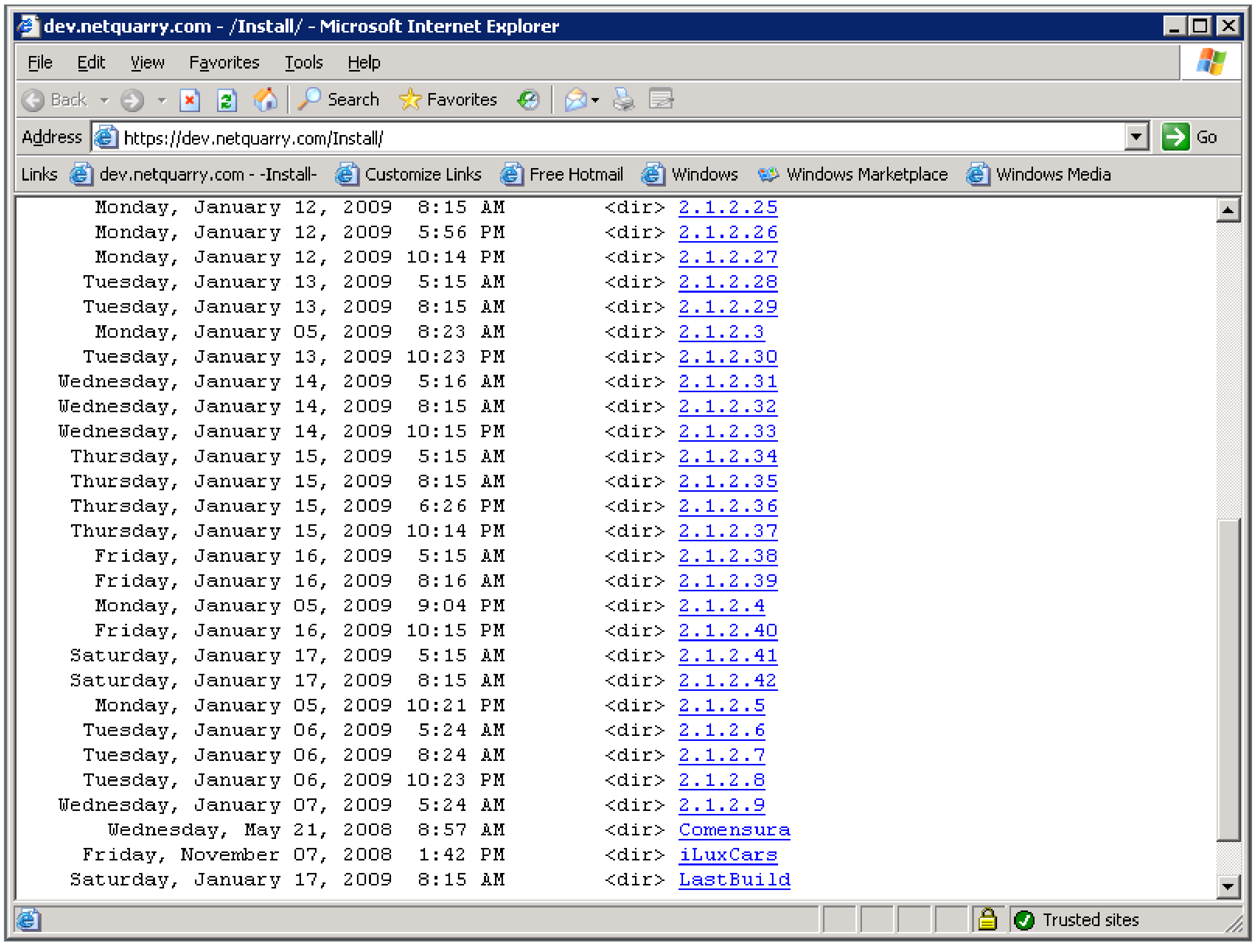


First you have to download the appropriate installation files, so use the IE shortcut to get to the install file location.

The full path is https://dev.netquarry.com/Install

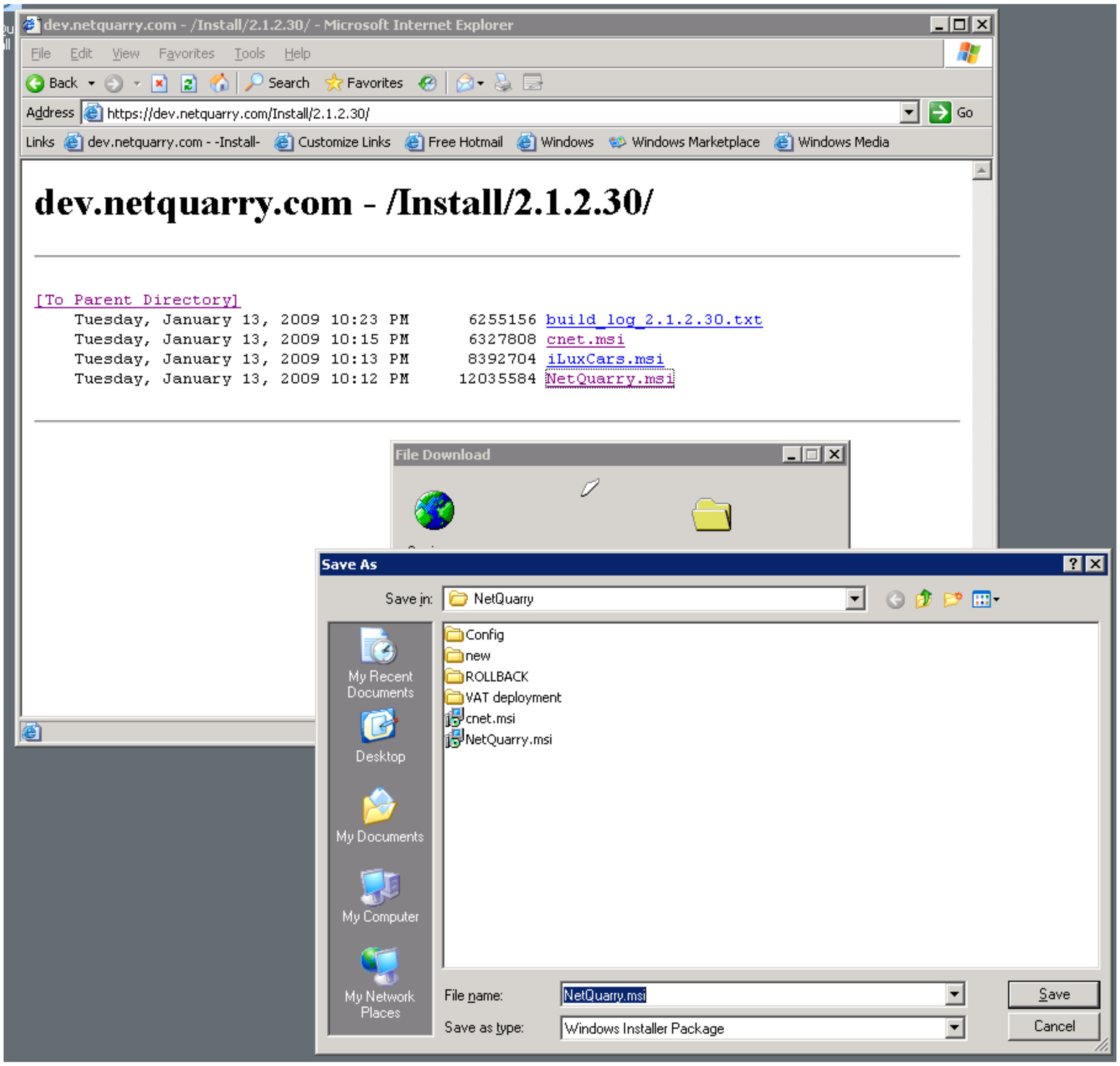
The authentication settings are;- userid: **nqinstaller**, pwd: **netquarry1470**

Notice that the date/times for these files are in Pacific time. Ideally you need to identify the version of the build and navigate to the folder by version and not timestamp.


dev.netquarry.com - /Install/ - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://dev.netquarry.com/Install/
Go
Links dev.netquarry.com - -Install- Customize Links Free Hotmail Windows Windows Marketplace Windows Media
Trusted sites

```
     Monday, January 12, 2009  8:15 AM        <dir> 2.1.2.25
     Monday, January 12, 2009  5:56 PM        <dir> 2.1.2.26
     Monday, January 12, 2009 10:14 PM        <dir> 2.1.2.27
    Tuesday, January 13, 2009  5:15 AM        <dir> 2.1.2.28
    Tuesday, January 13, 2009  8:15 AM        <dir> 2.1.2.29
     Monday, January 05, 2009  8:23 AM        <dir> 2.1.2.3
    Tuesday, January 13, 2009 10:23 PM        <dir> 2.1.2.30
  Wednesday, January 14, 2009  5:16 AM        <dir> 2.1.2.31
  Wednesday, January 14, 2009  8:15 AM        <dir> 2.1.2.32
  Wednesday, January 14, 2009 10:15 PM        <dir> 2.1.2.33
   Thursday, January 15, 2009  5:15 AM        <dir> 2.1.2.34
   Thursday, January 15, 2009  8:15 AM        <dir> 2.1.2.35
   Thursday, January 15, 2009  6:26 PM        <dir> 2.1.2.36
   Thursday, January 15, 2009 10:14 PM        <dir> 2.1.2.37
     Friday, January 16, 2009  5:15 AM        <dir> 2.1.2.38
     Friday, January 16, 2009  8:16 AM        <dir> 2.1.2.39
     Monday, January 05, 2009  9:04 PM        <dir> 2.1.2.4
     Friday, January 16, 2009 10:15 PM        <dir> 2.1.2.40
   Saturday, January 17, 2009  5:15 AM        <dir> 2.1.2.41
   Saturday, January 17, 2009  8:15 AM        <dir> 2.1.2.42
     Monday, January 05, 2009 10:21 PM        <dir> 2.1.2.5
    Tuesday, January 06, 2009  5:24 AM        <dir> 2.1.2.6
    Tuesday, January 06, 2009  8:24 AM        <dir> 2.1.2.7
    Tuesday, January 06, 2009 10:23 PM        <dir> 2.1.2.8
  Wednesday, January 07, 2009  5:24 AM        <dir> 2.1.2.9
       Wednesday, May 21, 2008  8:57 AM        <dir> Comensura
    Friday, November 07, 2008  1:42 PM        <dir> iLuxCars
   Saturday, January 17, 2009  8:15 AM        <dir> LastBuild
```

For both the NetQuarry.msi file and cnet.msi file, copy them to the C:\NetQuarry folder


dev.netquarry.com - /Install/2.1.2.30/ - Microsoft Internet Explorer

Address https://dev.netquarry.com/Install/2.1.2.30/

dev.netquarry.com - /Install/2.1.2.30/

```
[To Parent Directory]
     Tuesday, January 13, 2009 10:23 PM      6255156 build_log_2.1.2.30.txt
     Tuesday, January 13, 2009 10:15 PM      6327808 cnet.msi
     Tuesday, January 13, 2009 10:13 PM      8392704 iLuxCars.msi
     Tuesday, January 13, 2009 10:12 PM     12035584 NetQuarry.msi
```

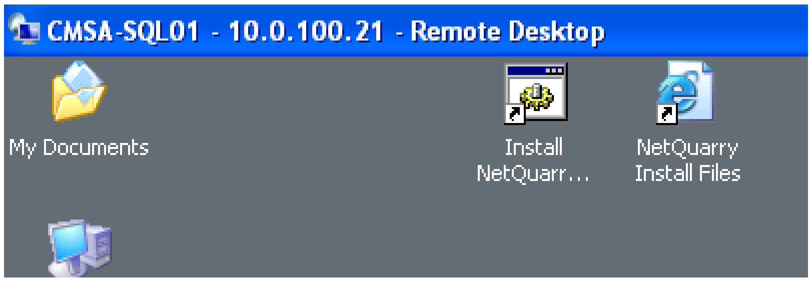
Save in: NetQuarry

Config
new
ROLLBACK
VAT deployment
cnet.msi
NetQuarry.msi

File name: NetQuarry.msi
Save as type: Windows Installer Package

Perform the installation

The installation of NetQuarry platform and the CNET application is all automated from a single batch file "Install NetQuarry + CNET"



The script automatically stops the web servers and scheduler processes on each of the three application servers. Uninstalls the previous version and installs the new version, updates the database schema and meta data and restarts the web servers and scheduler processes.

There is no need for any intervention during the install process. The whole install process takes less than 5 minutes.

Once finished, you should perform some sanity checking on the installation to make sure people can login and that any new features you expect to see, are available.

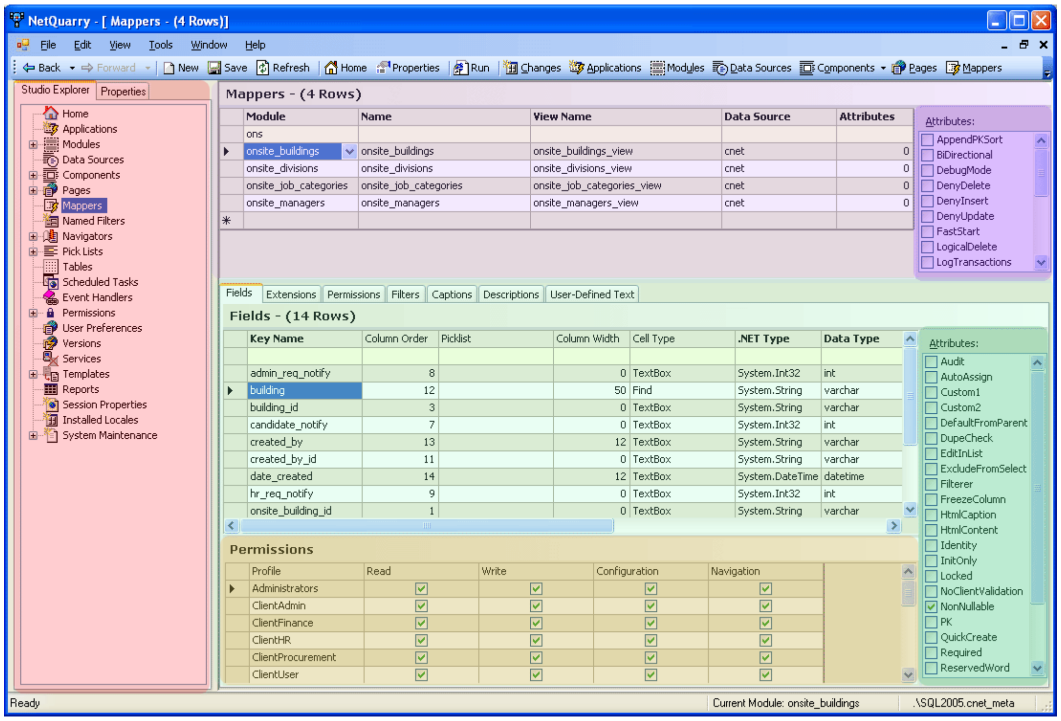
Be sure to check each app server individually, from the internal location so you can confirm the correct operation of each web server's installation. Checking operation from the outside, would only test a single app server due to the effect of the load balancer for external requests.

Production Rollback

In the event that you discover a show stopping issue with the latest install, to roll back to the last good version, you simply logon to the database server and copy the two install files from the C:\NetQuarry\ROLLBACK folder to C:\NetQuarry folder.

Then run the batch file.

Navigating the Studio


NetQuarry - [Mappers - (4 Rows)]
File Edit View Tools Window Help
Back Forward New Save Refresh Home Properties Run Changes Applications Modules Data Sources Components Pages Mappers
Studio Explorer Properties
Home
Applications
Modules
Data Sources
Components
Pages
Mappers
Named Filters
Navigators
Pick Lists
Tables
Scheduled Tasks
Event Handlers
Permissions
User Preferences
Versions
Services
Templates
Reports
Session Properties
Installed Locales
System Maintenance
Fields Extensions Permissions Filters Captions Descriptions User-Defined Text

Mappers - (4 Rows)

Module	Name	View Name	Data Source	Attributes
ons				
onsite_buildings	onsite_buildings	onsite_buildings_view	cnet	0
onsite_divisions	onsite_divisions	onsite_divisions_view	cnet	0
onsite_job_categories	onsite_job_categories	onsite_job_categories_view	cnet	0
onsite_managers	onsite_managers	onsite_managers_view	cnet	0

Attributes:
☐ AppendPKSort
☐ BiDirectional
☐ DebugMode
☐ DenyDelete
☐ DenyInsert
☐ DenyUpdate
☐ FastStart
☐ LogicalDelete
☐ LogTransactions

Fields - (14 Rows)

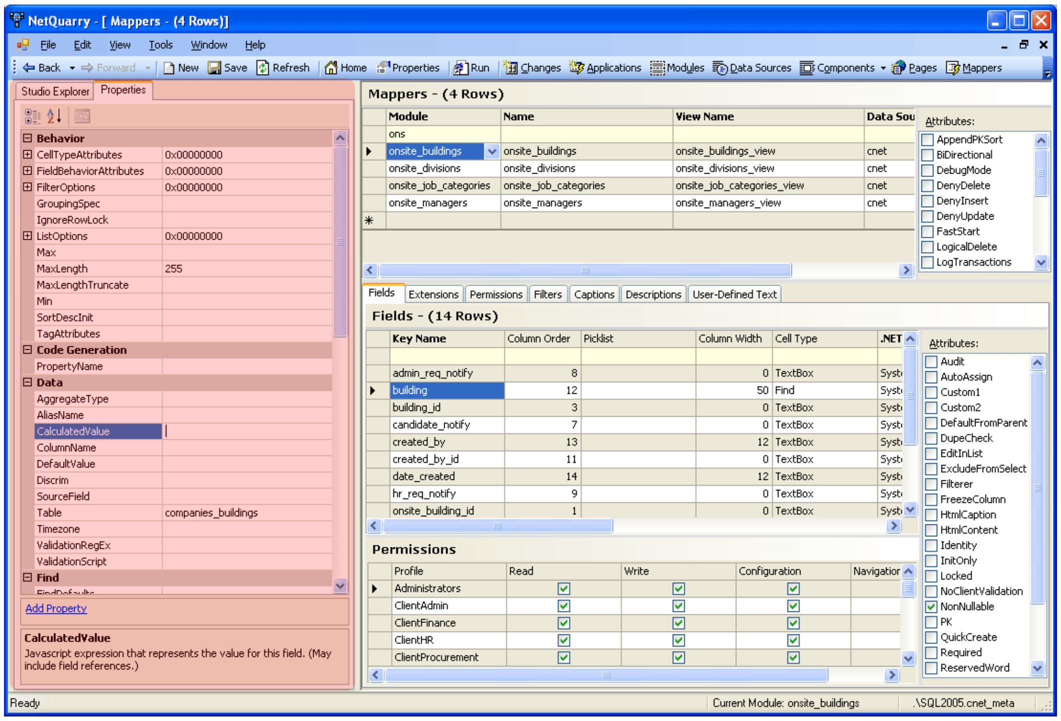
Key Name	Column Order	Picklist	Column Width	Cell Type	.NET Type	Data Type
admin_req_notify	8		0	TextBox	System.Int32	int
building	12		50	Find	System.String	varchar
building_id	3		0	TextBox	System.String	varchar
candidate_notify	7		0	TextBox	System.Int32	int
created_by	13		12	TextBox	System.String	varchar
created_by_id	11		0	TextBox	System.String	varchar
date_created	14		12	TextBox	System.DateTime	datetime
hr_req_notify	9		0	TextBox	System.Int32	int
onsite_building_id	1		0	TextBox	System.String	varchar

Attributes:
☐ Audit
☐ AutoAssign
☐ Custom1
☐ Custom2
☐ DefaultFromParent
☐ DupeCheck
☐ EditInList
☐ ExcludeFromSelect
☐ Filterer
☐ FreezeColumn
☐ HtmlCaption
☐ HtmlContent
☐ Identity
☐ InitOnly
☐ Locked
☐ NoClientValidation
✓ NonNullable
☐ PK
☐ QuickCreate
☐ Required
☐ ReservedWord

Permissions

Profile	Read	Write	Configuration	Navigation
Administrators	✓	✓	✓	✓
ClientAdmin	✓	✓	✓	✓
ClientFinance	✓	✓	✓	✓
ClientHR	✓	✓	✓	✓
ClientProcurement	✓	✓	✓	✓
ClientUser	✓	✓	✓	✓

Ready | Current Module: onsite_buildings | .\SQL2005.cnet_meta


NetQuarry - [Mappers - (4 Rows)]
File Edit View Tools Window Help
Back Forward New Save Refresh Home Properties Run Changes Applications Modules Data Sources Components Pages Mappers
Studio Explorer Properties
Fields Extensions Permissions Filters Captions Descriptions User-Defined Text

Behavior	
CellTypeAttributes	0x00000000
FieldBehaviorAttributes	0x00000000
FilterOptions	0x00000000
GroupingSpec	
IgnoreRowLock	
ListOptions	0x00000000
Max	
MaxLength	255
MaxLengthTruncate	
Min	
SortDescInit	
TagAttributes	
Code Generation	
PropertyName	
Data	
AggregateType	
AliasName	
CalculatedValue	
ColumnName	
DefaultValue	
Discrim	
SourceField	
Table	companies_buildings
Timezone	
ValidationRegEx	
ValidationScript	
Find	

Add Property

CalculatedValue
Javascript expression that represents the value for this field. (May include field references.)

Mappers - (4 Rows)

Module	Name	View Name	Data Sou
ons			
onsite_buildings	onsite_buildings	onsite_buildings_view	cnet
onsite_divisions	onsite_divisions	onsite_divisions_view	cnet
onsite_job_categories	onsite_job_categories	onsite_job_categories_view	cnet
onsite_managers	onsite_managers	onsite_managers_view	cnet

Attributes:
☐ AppendPKSort
☐ BiDirectional
☐ DebugMode
☐ DenyDelete
☐ DenyInsert
☐ DenyUpdate
☐ FastStart
☐ LogicalDelete
☐ LogTransactions

Fields - (14 Rows)

Key Name	Column Order	Picklist	Column Width	Cell Type	.NET
admin_req_notify	8		0	TextBox	Syst
building	12		50	Find	Syst
building_id	3		0	TextBox	Syst
candidate_notify	7		0	TextBox	Syst
created_by	13		12	TextBox	Syst
created_by_id	11		0	TextBox	Syst
date_created	14		12	TextBox	Syst
hr_req_notify	9		0	TextBox	Syst
onsite_building_id	1		0	TextBox	Syst

Attributes:
☐ Audit
☐ AutoAssign
☐ Custom1
☐ Custom2
☐ DefaultFromParent
☐ DupeCheck
☐ EditInList
☐ ExcludeFromSelect
☐ Filterer
☐ FreezeColumn
☐ HtmlCaption
☐ HtmlContent
☐ Identity
☐ InitOnly
☐ Locked
☐ NoClientValidation
✓ NonNullable
☐ PK
☐ QuickCreate
☐ Required
☐ ReservedWord

Permissions

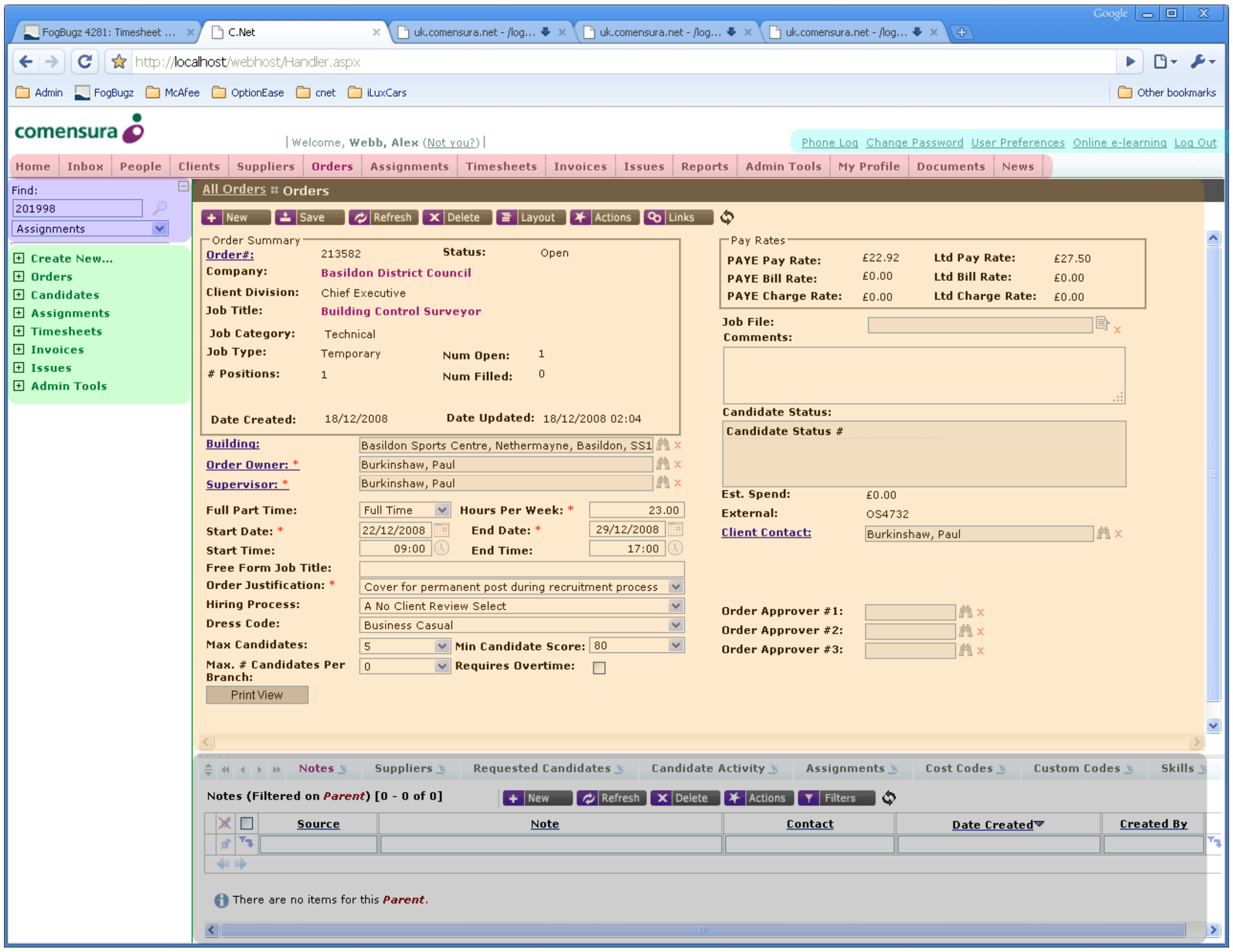
Profile	Read	Write	Configuration	Navigatior
Administrators	✓	✓	✓	
ClientAdmin	✓	✓	✓	
ClientFinance	✓	✓	✓	
ClientHR	✓	✓	✓	
ClientProcurement	✓	✓	✓	

Ready | Current Module: onsite_buildings | .\SQL2005.cnet_meta

The screenshot above displays the main regions of the web application

Region	**Description**
Red	This is called the Studio Explorer. It provides a way to quickly navigate to different sets of Metadata.
Purple	The region where the meta data is managed for the top level object. In this screenshot, the top level object is a Mapper. Other top level objects include Pages, and Navigators.
Dark Purple	Is typically where the attribute manager is located for any top level object. The attributes are a set of flgas that control behavior of the object.
Green	The region where meta data is managed for data related to a top level object. In this screenshot, the related data is for defining a set of fields, but you can see also that related to mappers are extensions, permissions, filters, etc, …
Dark Green	Is typically where the attribute manager is located for any related object. Not all related objects will have attributes.
Orange	The region where meta data is managed for data related to data, related to a top level object. Not all top level meta data objects have a third layer of associated meta data. Typically the third layer is related to setting permissions on the second layer.
Red (second screenshot)	In the second screenshot the red region is the Property Sheet. The property sheet is where a substantial amount of time is spent to set up the required behavior for the application. The property sheet is context sensitive, so that the available properties will change depending on which item currently has the focus.

Navigating the Layout



| Welcome, **Webb, Alex** (Not you?) |

Find: 201998

Assignments

All Orders :: Orders

Order Summary

Order#:	213582	**Status:**	Open
Company:	Basildon District Council		
Client Division:	Chief Executive		
Job Title:	Building Control Surveyor		
Job Category:	Technical		
Job Type:	Temporary	**Num Open:**	1
# Positions:	1	**Num Filled:**	0
Date Created:	18/12/2008	**Date Updated:**	18/12/2008 02:04

Pay Rates

PAYE Pay Rate:	£22.92	**Ltd Pay Rate:**	£27.50
PAYE Bill Rate:	£0.00	**Ltd Bill Rate:**	£0.00
PAYE Charge Rate:	£0.00	**Ltd Charge Rate:**	£0.00

Job File:

Comments:

Candidate Status:

Candidate Status #

Building:	Basildon Sports Centre, Nethermayne, Basildon, SS1		
Order Owner: *	Burkinshaw, Paul		
Supervisor: *	Burkinshaw, Paul		
Full Part Time:	Full Time	**Hours Per Week:** *	23.00
Start Date: *	22/12/2008	**End Date:** *	29/12/2008
Start Time:	09:00	**End Time:**	17:00
Free Form Job Title:			
Order Justification: *	Cover for permanent post during recruitment process		
Hiring Process:	A No Client Review Select		
Dress Code:	Business Casual		
Max Candidates:	5	**Min Candidate Score:**	80
Max. # Candidates Per Branch:	0	**Requires Overtime:**	☐

Est. Spend:	£0.00
External:	OS4732
Client Contact:	Burkinshaw, Paul
Order Approver #1:	
Order Approver #2:	
Order Approver #3:	

Notes (Filtered on *Parent*) [0 - 0 of 0]

Source	Note	Contact	Date Created	Created By

There are no items for this *Parent*.

The screenshot above displays the main regions of the web application

Region	Description
Blue	This is called the app bar. Items in this region of the screen are managed by a NavBar navigator called “appbar”. The “appbar” navigator is a built in navigator to which custom pages are connected.
Pink	This is called the main bar. Items in this region of the screen are managed by a NavBar navigator called “main”. The “main” navigator is a built in navigator to which custom pages are connected.
Purple	This is called the quick find bar, or search bar. The search bar is only visible when one or more pages have the SupportsSearch page attribute specified.
Green	This region is called the Tree, or Navigator Tree.
Purple+Green	The purple and green region together form the “SideSlot”. The SideSlot is just a place to render some data using a control. The control renders the data, not the Slot itself. In this application, when a SideSlot is visible, it is rendered as a tree control using the NavBar control. The SideSlot is not always visible. The Page meta data tells the application how the data should be presented in the SideSlot.
Orange	This region is where the detail and list views of data are displayed. Detail and List views are displayed by Pages.
Grey	This region is called the Subform. If a page has no subforms, there is no divider and the main page fills the whole region.
Orange+Grey	The orange and grey region together form the “MainSlot”. The MainSlot is a place to render some data using a control. The control renders the data, not the Slot itself. There is always a MainSlot. The Page meta data tells the application how the data should be presented in the MainSlot

Debugging Techniques

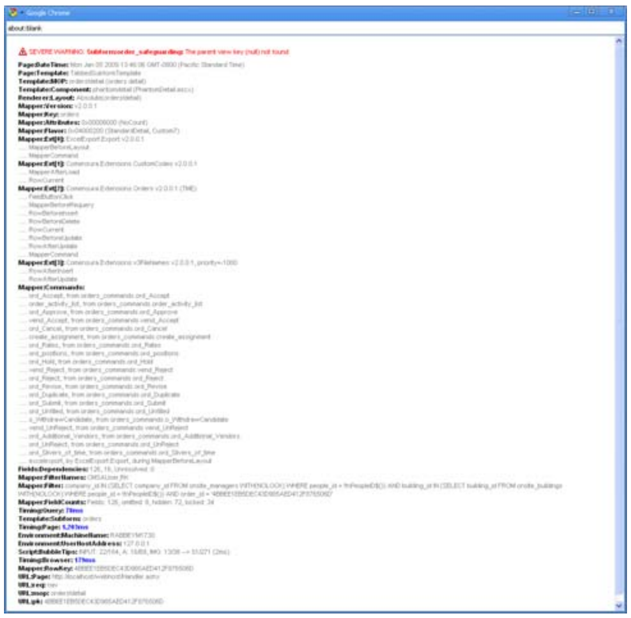
As you begin working with the NetQuarry environment, you are going to make some mistakes along the way. The quickest way to get past them and get it working, is to use the debugging features provided in the environment

F8 Debugging

On nearly every page of the web interface, we have provided F8 debugging at two different levels. At the page level, the debug information tells you a massive amount of information about the page that's displayed as well as the page that was navigated from, flavors being applied, filters, extensions and mapper. At the Field level the debug information can tell you the name of the underlying field but also why the field could be locked (there are many reasons for a field to be locked and this is the quickest way to find out the reason).

Example of F8 information on detail page

Click on a piece of unused screen and press F8 to get page level information

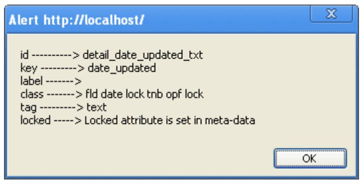


```
SEVERE WARNING: Subform:order_safeguarding: The parent view key (null) not found
PageDateTime: Mon Jan 26 2009 13:46:06 GMT-0800 (Pacific Standard Time)
PageTemplate: TabbedCustomTemplate
TemplateMOP: ordersdetail (orders detail)
TemplateComponent: phantomdetail (PhantomDetail.ascx)
RendererLayout: Absolute(ordersdetail)
MapperVersion: v2.0.0.1
MapperKey: orders
MapperAttributes: 0x00000000 (NoCount)
MapperFlavor: 0x04000200 (StandardDetail, Custom7)
MapperExt[0]: ExcelExport.Export v2.0.0.1
.... MapperBeforeLayout
.... MapperCommand
MapperExt[1]: Comensura.Extensions.CustomCodes v2.0.0.1
.... MapperAfterLoad
.... RowCurrent
MapperExt[2]: Comensura.Extensions.Orders v2.0.0.1 (TME)
.... FieldButtonClick
.... MapperBeforeRequery
.... RowBeforeInsert
.... RowBeforeDelete
.... RowCurrent
.... RowBeforeUpdate
.... RowAfterUpdate
.... MapperCommand
MapperExt[3]: Comensura.Extensions.v3Filenames v2.0.0.1, priority=1000
.... RowAfterInsert
.... RowAfterUpdate
MapperCommands:
.... ord_Accept, from orders_commands.ord_Accept
.... order_activity_list, from orders_commands.order_activity_list
.... ord_Approve, from orders_commands.ord_Approve
.... vend_Accept, from orders_commands.vend_Accept
.... ord_Cancel, from orders_commands.ord_Cancel
.... create_assignment, from orders_commands.create_assignment
.... ord_Rates, from orders_commands.ord_Rates
.... ord_positions, from orders_commands.ord_positions
.... ord_Hold, from orders_commands.ord_Hold
.... vend_Reject, from orders_commands.vend_Reject
.... ord_Reject, from orders_commands.ord_Reject
.... ord_Revise, from orders_commands.ord_Revise
.... ord_Duplicate, from orders_commands.ord_Duplicate
.... ord_Submit, from orders_commands.ord_Submit
.... ord_Unfilled, from orders_commands.ord_Unfilled
.... s_WithdrawCandidate, from orders_commands.s_WithdrawCandidate
.... vend_UnReject, from orders_commands.vend_UnReject
.... ord_Additional_Vendors, from orders_commands.ord_Additional_Vendors
.... ord_UnReject, from orders_commands.ord_UnReject
.... ord_Slivers_of_time, from orders_commands.ord_Slivers_of_time
.... excelexport, by ExcelExport.Export, during MapperBeforeLayout
FieldsDependencies: 126, 18, Unresolved: 0
MapperFilterNames: CMSAUser_RK
MapperFilter: company_id IN (SELECT company_id FROM onsite_managers WITH(NOLOCK) WHERE people_id = fnPeopleID$() AND building_id IN (SELECT building_id FROM onsite_buildings WITH(NOLOCK) WHERE people_id = fnPeopleID$()) AND order_id = '4B8EE1EB5DEC43D905AED412F8765060'
MapperFieldCounts: Fields: 126, omitted: 8, hidden: 72, locked: 34
TimingQuery: 78ms
TemplateSubforms: orders
TimingPage: 5,243ms
EnvironmentMachineName: RABBEYM1730
EnvironmentUserHostAddress: 127.0.0.1
ScriptBubbleTips: INPUT: 22/164, A: 18/88, IMG: 13/38 --> 55/271 (2ms)
TimingBrowser: 17ms
MapperRowKey: 4B8EE1EB5DEC43D905AED412F8765060
URLPage: http://localhost/webhost/Handler.ashx
URLreq: nav
URLmop: ordersdetail
URLpk: 4B8EE1EB5DEC43D905AED412F8765060
```

Typically you look here to identify what page metadata is being used to display the page. Which mapper is used as the basis of the page. What business rules are set up on the page. What navigation parameters were used to open the page. Timing information about the page.

Use the escape key to close the popup window.

Then you can click in a field on the detail. It pops up a simpler dialog providing some simple information about a field



```
id ----------> detail_date_updated_txt
key ---------> date_updated
label -------> 
class -------> fld date lock tnb opf lock
tag ---------> text
locked -----> Locked attribute is set in meta-data
```

Typically you would use this identify the keyname of a field on the page so you can identify which field to find on the mapper, or in your code. The other piece of information that is provided which can be extremely useful is to identify why the field is locked. In this case it's telling you the field is locked because of the fields locked attribute.

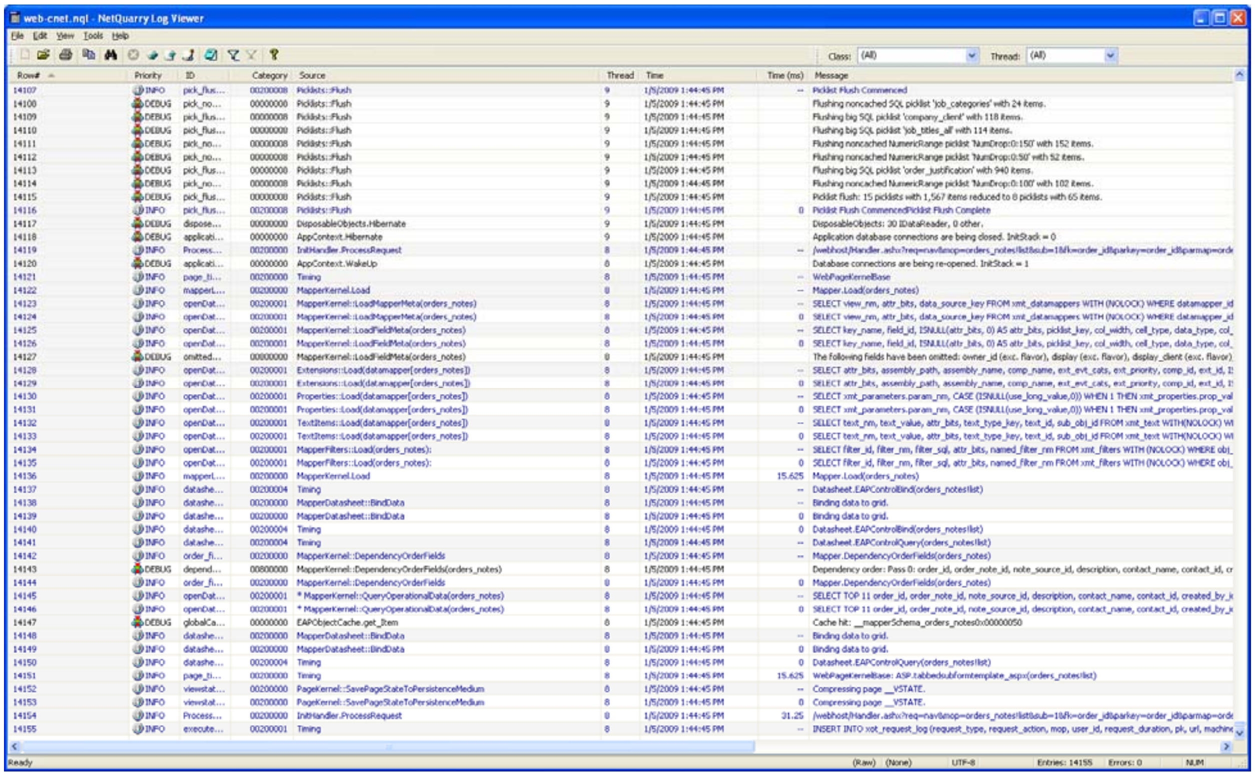
Devlog

The devlog is an xml based file that is generated by the NetQuarry platform. Every NetQuarry application generates a log file. There is one for the App Server, Studio, Scheduler, Code Generator. Typically the log files will be found in the Logs folder of the installed location. E.g. in production, C:\Inetpub\wwwroot\<app name>\logs. In development environment it's probably C:\Inetpub\wwwroot\NetQuarry\logs

The devlog is being written to all the time and you can modify the settings of the devlog to restrict the size of each log file and to keep a certain number of log files as backup. The default log settings specify 20MB sized log files and to keep the last 10. Ideally you would change the devlog settings for an application in order to keep about a day's worth of logging history before the log files start rolling over. The size of the log file should be kept below 50MB just for the convenience of being able to read them. So the main parameter you would want to change is the number of files to keep. Of course the overall limitation is the amount of disk space on the server. The log files do zip exceptionally well, however. About 97% compression is typical.

You would use the devlog to identify what is happening on the system at run time. It records what users are doing, what events are being fired and what database requests are being made. When errors occur, the call stack of the error is also recorded.

The log files are read by the Log Viewer application.


web-cnet.nql - NetQuarry Log Viewer
File Edit View Tools Help
Class: (All)
Thread: (All)

Row#	Priority	ID	Category	Source	Thread	Time	Time (ms)	Message
14107	INFO	pick_flus...	00200008	Picklists::Flush	9	1/5/2009 1:44:45 PM	--	Picklist Flush Commenced
14108	DEBUG	pick_no...	00000008	Picklists::Flush	9	1/5/2009 1:44:45 PM		Flushing noncached SQL picklist 'job_categories' with 24 items.
14109	DEBUG	pick_flus...	00000008	Picklists::Flush	9	1/5/2009 1:44:45 PM		Flushing big SQL picklist 'company_client' with 118 items.
14110	DEBUG	pick_flus...	00000008	Picklists::Flush	9	1/5/2009 1:44:45 PM		Flushing big SQL picklist 'job_titles_all' with 114 items.
14111	DEBUG	pick_no...	00000008	Picklists::Flush	9	1/5/2009 1:44:45 PM		Flushing noncached NumericRange picklist 'NumDrop:0:150' with 152 items.
14112	DEBUG	pick_no...	00000008	Picklists::Flush	9	1/5/2009 1:44:45 PM		Flushing noncached NumericRange picklist 'NumDrop:0:50' with 52 items.
14113	DEBUG	pick_flus...	00000008	Picklists::Flush	9	1/5/2009 1:44:45 PM		Flushing big SQL picklist 'order_justification' with 940 items.
14114	DEBUG	pick_no...	00000008	Picklists::Flush	9	1/5/2009 1:44:45 PM		Flushing noncached NumericRange picklist 'NumDrop:0:100' with 102 items.
14115	DEBUG	pick_flus...	00000008	Picklists::Flush	9	1/5/2009 1:44:45 PM		Picklist flush: 15 picklists with 1,567 items reduced to 8 picklists with 65 items.
14116	INFO	pick_flus...	00200008	Picklists::Flush	9	1/5/2009 1:44:45 PM	0	Picklist Flush CommencedPicklist Flush Complete
14117	DEBUG	dispose...	00000000	DisposableObjects.Hibernate	9	1/5/2009 1:44:45 PM		DisposableObjects: 30 IDataReader, 0 other.
14118	DEBUG	applicati...	00000000	AppContext.Hibernate	9	1/5/2009 1:44:45 PM		Application database connections are being closed. InitStack = 0
14119	INFO	Process...	00200000	InitHandler.ProcessRequest	8	1/5/2009 1:44:45 PM	--	/webhost/Handler.ashx?req=nav&mop=orders_notes!list&sub=1&fk=order_id&parkey=order_id&parmap=orde
14120	DEBUG	applicati...	00000000	AppContext.WakeUp	8	1/5/2009 1:44:45 PM		Database connections are being re-opened. InitStack = 1
14121	INFO	page_ti...	00200000	Timing	8	1/5/2009 1:44:45 PM	--	WebPageKernelBase
14122	INFO	mapperL...	00200000	MapperKernel.Load	8	1/5/2009 1:44:45 PM	--	Mapper.Load(orders_notes)
14123	INFO	openDat...	00200001	MapperKernel::LoadMapperMeta(orders_notes)	8	1/5/2009 1:44:45 PM	--	SELECT view_nm, attr_bits, data_source_key FROM xmt_datamappers WITH (NOLOCK) WHERE datamapper_id
14124	INFO	openDat...	00200001	MapperKernel::LoadMapperMeta(orders_notes)	8	1/5/2009 1:44:45 PM	0	SELECT view_nm, attr_bits, data_source_key FROM xmt_datamappers WITH (NOLOCK) WHERE datamapper_id
14125	INFO	openDat...	00200001	MapperKernel::LoadFieldMeta(orders_notes)	8	1/5/2009 1:44:45 PM	--	SELECT key_name, field_id, ISNULL(attr_bits, 0) AS attr_bits, picklist_key, col_width, cell_type, data_type, col_
14126	INFO	openDat...	00200001	MapperKernel::LoadFieldMeta(orders_notes)	8	1/5/2009 1:44:45 PM	0	SELECT key_name, field_id, ISNULL(attr_bits, 0) AS attr_bits, picklist_key, col_width, cell_type, data_type, col_
14127	DEBUG	omitted...	00000000	MapperKernel::LoadFieldMeta(orders_notes)	8	1/5/2009 1:44:45 PM		The following fields have been omitted: owner_id (exc. flavor), display (exc. flavor), display_client (exc. flavor)
14128	INFO	openDat...	00200001	Extensions::Load(datamapper[orders_notes])	8	1/5/2009 1:44:45 PM	--	SELECT attr_bits, assembly_path, assembly_name, comp_name, ext_evt_cats, ext_priority, comp_id, ext_id, I:
14129	INFO	openDat...	00200001	Extensions::Load(datamapper[orders_notes])	8	1/5/2009 1:44:45 PM	0	SELECT attr_bits, assembly_path, assembly_name, comp_name, ext_evt_cats, ext_priority, comp_id, ext_id, I:
14130	INFO	openDat...	00200001	Properties::Load(datamapper[orders_notes])	8	1/5/2009 1:44:45 PM	--	SELECT xmt_parameters.param_nm, CASE (ISNULL(use_long_value,0)) WHEN 1 THEN xmt_properties.prop_val
14131	INFO	openDat...	00200001	Properties::Load(datamapper[orders_notes])	8	1/5/2009 1:44:45 PM	0	SELECT xmt_parameters.param_nm, CASE (ISNULL(use_long_value,0)) WHEN 1 THEN xmt_properties.prop_val
14132	INFO	openDat...	00200001	TextItems::Load(datamapper[orders_notes])	8	1/5/2009 1:44:45 PM	--	SELECT text_nm, text_value, attr_bits, text_type_key, text_id, sub_obj_id FROM xmt_text WITH(NOLOCK) WI
14133	INFO	openDat...	00200001	TextItems::Load(datamapper[orders_notes])	8	1/5/2009 1:44:45 PM	0	SELECT text_nm, text_value, attr_bits, text_type_key, text_id, sub_obj_id FROM xmt_text WITH(NOLOCK) WI
14134	INFO	openDat...	00200001	MapperFilters::Load(orders_notes):	8	1/5/2009 1:44:45 PM	--	SELECT filter_id, filter_nm, filter_sql, attr_bits, named_filter_nm FROM xmt_filters WITH (NOLOCK) WHERE obj_
14135	INFO	openDat...	00200001	MapperFilters::Load(orders_notes):	8	1/5/2009 1:44:45 PM	0	SELECT filter_id, filter_nm, filter_sql, attr_bits, named_filter_nm FROM xmt_filters WITH (NOLOCK) WHERE obj_
14136	INFO	mapperL...	00200000	MapperKernel.Load	8	1/5/2009 1:44:45 PM	15.625	Mapper.Load(orders_notes)
14137	INFO	datashe...	00200004	Timing	8	1/5/2009 1:44:45 PM	--	Datasheet.EAPControlBind(orders_notes!list)
14138	INFO	datashe...	00200000	MapperDatasheet::BindData	8	1/5/2009 1:44:45 PM	--	Binding data to grid.
14139	INFO	datashe...	00200000	MapperDatasheet::BindData	8	1/5/2009 1:44:45 PM	0	Binding data to grid.
14140	INFO	datashe...	00200004	Timing	8	1/5/2009 1:44:45 PM	0	Datasheet.EAPControlBind(orders_notes!list)
14141	INFO	datashe...	00200004	Timing	8	1/5/2009 1:44:45 PM	--	Datasheet.EAPControlQuery(orders_notes!list)
14142	INFO	order_fi...	00200000	MapperKernel::DependencyOrderFields	8	1/5/2009 1:44:45 PM	--	Mapper.DependencyOrderFields(orders_notes)
14143	DEBUG	depend...	00800000	MapperKernel::DependencyOrderFields(orders_notes)	8	1/5/2009 1:44:45 PM		Dependency order: Pass 0: order_id, order_note_id, note_source_id, description, contact_name, contact_id, cr
14144	INFO	order_fi...	00200000	MapperKernel::DependencyOrderFields	8	1/5/2009 1:44:45 PM	0	Mapper.DependencyOrderFields(orders_notes)
14145	INFO	openDat...	00200001	* MapperKernel::QueryOperationalData(orders_notes)	8	1/5/2009 1:44:45 PM	--	SELECT TOP 11 order_id, order_note_id, note_source_id, description, contact_name, contact_id, created_by_i
14146	INFO	openDat...	00200001	* MapperKernel::QueryOperationalData(orders_notes)	8	1/5/2009 1:44:45 PM	0	SELECT TOP 11 order_id, order_note_id, note_source_id, description, contact_name, contact_id, created_by_i
14147	DEBUG	globalCa...	00000000	EAPObjectCache.get_Item	8	1/5/2009 1:44:45 PM		Cache hit: __mapperSchema_orders_notes0x00000050
14148	INFO	datashe...	00200000	MapperDatasheet::BindData	8	1/5/2009 1:44:45 PM	--	Binding data to grid.
14149	INFO	datashe...	00200000	MapperDatasheet::BindData	8	1/5/2009 1:44:45 PM	0	Binding data to grid.
14150	INFO	datashe...	00200004	Timing	8	1/5/2009 1:44:45 PM	0	Datasheet.EAPControlQuery(orders_notes!list)
14151	INFO	page_ti...	00200000	Timing	8	1/5/2009 1:44:45 PM	15.625	WebPageKernelBase: ASP.tabbedsubformtemplate_aspx(orders_notes!list)
14152	INFO	viewstat...	00200000	PageKernel::SavePageStateToPersistenceMedium	8	1/5/2009 1:44:45 PM	--	Compressing page __VSTATE.
14153	INFO	viewstat...	00200000	PageKernel::SavePageStateToPersistenceMedium	8	1/5/2009 1:44:45 PM	0	Compressing page __VSTATE.
14154	INFO	Process...	00200000	InitHandler.ProcessRequest	8	1/5/2009 1:44:45 PM	31.25	/webhost/Handler.ashx?req=nav&mop=orders_notes!list&sub=1&fk=order_id&parkey=order_id&parmap=orde
14155	INFO	execute...	00200001	Timing	8	1/5/2009 1:44:45 PM	--	INSERT INTO xot_request_log (request_type, request_action, mop, user_id, request_duration, pk, url, machine

Ready (Raw) (None) UTF-8 Entries: 14155 Errors: 0 NUM

By default, the devlog opens up in normal mode where it displays information, warnings, errors. However, not all problems raise errors and you want to see what the underlying system is really doing. This is where you would want to view the timing information. To toggle the viewing of timing information, use the keystroke Ctrl T, or from the menu, View, Timing Entries. The above screenshot shows timing entries.

Getting Mixed Install and Debug Environment Setup

Software required

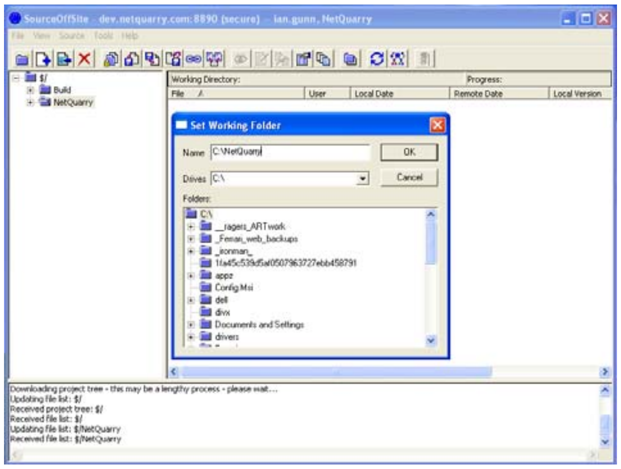
NetQuarry requires the following:

- Microsoft Windows XP (Service Pack 3) or Microsoft Windows Server 2003 (Service Pack 1)
- Microsoft SQL Server 2005 (or later)
- Internet Information Services (version 5.0 for Windows XP, version 6.0 for Windows 2003)
- Microsoft .NET Framework Version 2.0
- Visual Studio 2005
- Visual Studio Web Deployment Tool
- SourceOffSite 4.2 (or later)

Setup

First go to SourceOffSite and open the NetQuarry database.

Set your working folder for NetQuarry\Build project to C:\NetQuarry\Build. Then get the latest code from the NetQuarry\Build project, making sure the "Get" is recursive.



Name: C:\NetQuarry

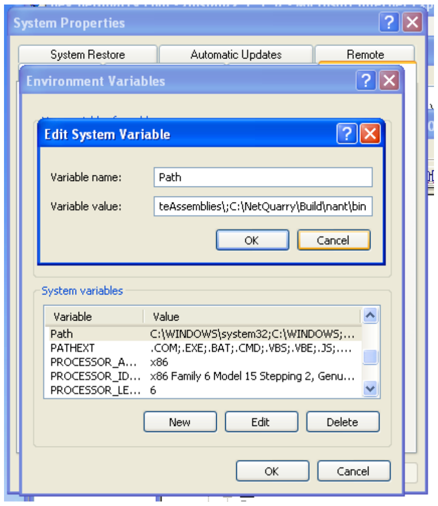
Drives: C:\

```
Downloading project tree - this may be a lengthy process - please wait...
Updating file list: $/
Received project tree: $/
Received file list: $/
Updating file list: $/NetQuarry
Received file list: $/NetQuarry
```

This process will download the files required for automating the build on your machine.

Go to the environment settings of your machine and add the following directory to your PATH environment variable

C:\NetQuarry\Build\nant\bin



Variable name: Path

Variable value: teAssemblies\;C:\NetQuarry\Build\nant\bin

Variable	Value
Path	C:\WINDOWS\system32;C:\WINDOWS;...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;....
PROCESSOR_A...	x86
PROCESSOR_ID...	x86 Family 6 Model 15 Stepping 2, Genu...
PROCESSOR_LE...	6

Now connect to the source control database that contains your source code. Set the working folder for your source tree to be C:\Netquarry\Customers\<Customer Name>. Get the latest making sure the "Get" is recursive.

Download the latest netquarry.msi and <customer_name>.msi files from the https://dev.netquarry.com/install and save the downloaded files to

C:\NetQuarry\Customers\<Customer Name>\Setup

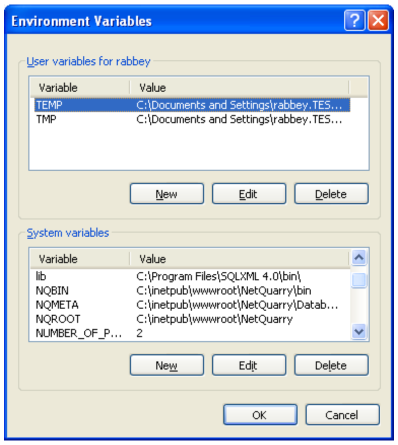
This folder would be created when you got the latest from your source tree.

Create a batch file called Install.bat in the C:\NetQuarry\Customers\<Customer Name>\Setup folder with the following contents…

```
@echo off

@echo Un-installing NetQuarry Platform...
call "%SystemRoot%\System32\MsiExec.exe" /x {642ADDFC-6BBC-4C4B-A8C9-377727C18024} /passive

@echo Installing NetQuarry Platform...
call "%SystemRoot%\System32\msiexec.exe" /i "netquarry.msi" TARGETPORT=80 TARGETVDIR=NetQuarry /passive

IF NOT EXIST %NQBIN%\eap.core.dll GOTO ERROREXIT

@echo Installing core databases...
cd %NQROOT%\Database\Scripts

@echo Install studio repository...
call %NQROOT%\Database\Scripts\db-install.bat

goto END

:ERROREXIT
@echo One or more errors occurred installing or the installation was cancelled.
GOTO FINAL

:END
@echo Install complete...
pause

:FINAL
```

Run the batch file you just created.

It will install the NetQuarry platform, but not complete successfully. The first install has to create the NetQuarry environment variables.



User variables for rabbey

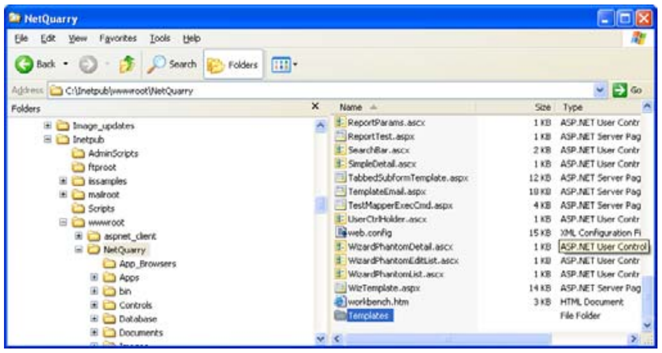
Variable	Value
TEMP	C:\Documents and Settings\rabbey.TES...
TMP	C:\Documents and Settings\rabbey.TES...

System variables

Variable	Value
lib	C:\Program Files\SQLXML 4.0\bin\
NQBIN	C:\inetpub\wwwroot\NetQuarry\bin
NQMETA	C:\inetpub\wwwroot\NetQuarry\Datab...
NQROOT	C:\inetpub\wwwroot\NetQuarry
NUMBER_OF_P...	2

Run the batch file again to install the NetQuarry platform. That should complete successfully and also result in the nq_studio database being created. If the nq_studio database has not been created in your database, then do not continue until you have resolved the problem

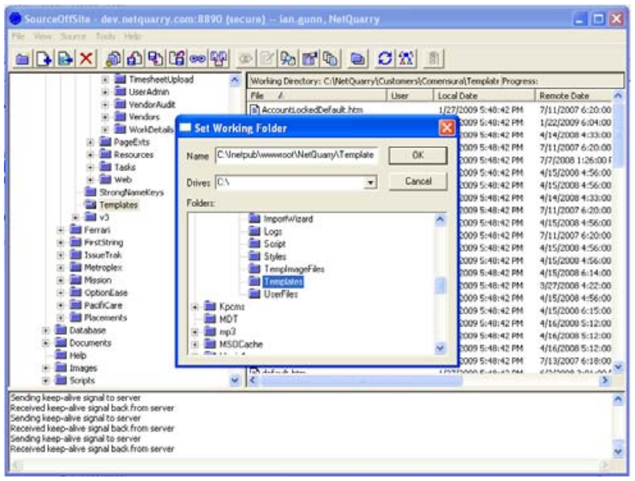
Create Templates folder in C:\Inetpub\wwwroot\NetQuarry



Address: C:\Inetpub\wwwroot\NetQuarry

Name	Size	Type
ReportParams.ascx	1 KB	ASP.NET User Contr
ReportTest.aspx	1 KB	ASP.NET Server Pag
SearchBar.ascx	2 KB	ASP.NET User Contr
SimpleDetail.ascx	1 KB	ASP.NET User Contr
TabbedSubformTemplate.aspx	12 KB	ASP.NET Server Pag
TemplateEmail.aspx	10 KB	ASP.NET Server Pag
TestMapperExecCmd.aspx	4 KB	ASP.NET Server Pag
UserCtrlHolder.ascx	1 KB	ASP.NET User Contr
web.config	15 KB	XML Configuration Fi
WizardPhantomDetail.ascx	1 KB	ASP.NET User Control
WizardPhantomEditList.ascx	1 KB	ASP.NET User Contr
WizardPhantomList.ascx	1 KB	ASP.NET User Contr
WizTemplate.aspx	14 KB	ASP.NET Server Pag
workbench.htm	3 KB	HTML Document
Templates		File Folder

In SourceOffSite, set the working folder of /Customers/<Customer Name>/Templates to C:\Inetpub\wwwroot\NetQuarry\Templates



SourceOffSite - dev.netquarry.com:8890 (secure) -- ian.gunn, NetQuarry

Working Directory: C:\NetQuarry\Customers\Comensural\Template Progress:

Set Working Folder

Name: C:\Inetpub\wwwroot\NetQuarry\Template

Drives: C:\

```
Sending keep-alive signal to server
Received keep-alive signal back from server
Sending keep-alive signal to server
Received keep-alive signal back from server
Sending keep-alive signal to server
Received keep-alive signal back from server
```

Go to the folder, C:\NetQuarry\Customers\<Customer Name>\Setup, run the batch file, install-<customer>-application.bat.

This will reinstall the NetQuarry environment and also install your customer specific environment.

Now run the batch file install-<customer>-database.bat

This will create the metadata database and update the metadata and update the schema on the operational database

Go to folder C:\NetQuarry\Customers\<Customer Name>\Database, run the batch file, reload-dbname_meta.bat

This will reload the meta and update the operation schema again, but this confirms the day to day meta update process will work OK.

Go back to SourceOffSite.   Go to folder $/NetQuarry/Customers/<Customer Name>/Images and set the working folder to C:\Inetpub\wwwroot\NetQuarry\Apps\<Application>\images.

Get the latest for this project folder.

Go to folder $/NetQuarry/Customers/<Customer Name>/Source/Web and set the working folder to C:\Inetpub\wwwroot\NetQuarry\Apps\<Application>.

Get the latest for this project folder.

Go to C:\NetQuarry\Customers\<Customer Name>\Source folder and run the batch file debug_build.bat.

This will build all of the Customer source code in a debug environment.

Go to C:\InetPub\wwwroot\NetQuarry\Logs folder. Add a read only text file to the folder. Call it "do not delete.txt". It can be an empty file. Make the file read only. This prevents the log folder from being deleted when the platform re-installs

On the same FOLDER, set full access permissions for Everyone. This allows the web server to write log entries to this folder. If you don't add the read only file as mentioned, you have to reset this permission after every NetQuarry platform install.

Go to SQL Management Tool. Add Logon roles for ASPNET, IUSR, IWAM users and for each user set the dbo role for both dbname_meta and dbname_data databases. Ensure any SQL Server roles are added to the database at this time if you are not using integrated security to connect to the database.

Included with this document is a zip file containing a template set of config files for any environment. Put this zip file in the folder C:\InetPub\wwwroot\NetQuarry. Extract to the same folder. It will put a config.app file in the same folder and four similar config.app files in the bin folder. To make the config files applicable to your environment, perform a simple substitution of strings in the files as follows…

{{appname}} -> your application name

{{dbname_data}} -> your operational data database name

{{dbname_meta}} -> your metadata database name

{{Customer_name}} -> your customer name

You may need to make further modifications other than the basic replacements. If you do need to make changes, make them in the .config.app files

These config.app files will be copied over the installed config files from a batch file that installs the NetQuarry platform

Go to the C:\NetQuarry\Customers\<Customer Name>\Setup folder

Edit the batch file you created earlier, install.bat

Replace the content of that file with these instructions

```
@echo off

@echo Un-installing NetQuarry Platform...
call "%SystemRoot%\System32\MsiExec.exe" /x {642ADDFC-6BBC-4C4B-A8C9-377727C18024} /passive

@echo Installing NetQuarry Platform...
call "%SystemRoot%\System32\msiexec.exe" /i "netquarry.msi" TARGETPORT=80 TARGETVDIR=NetQuarry /passive

IF NOT EXIST %NQBIN%\eap.core.dll GOTO ERROREXIT

@echo Installing core databases...
cd %NQROOT%\Database\Scripts

@echo Install studio repository...
call %NQROOT%\Database\Scripts\db-install.bat

@echo -------------------------------------------------------------------------------
@echo Updating local .config files
@echo -------------------------------------------------------------------------------
IF EXIST %NQROOT%\web.config.app copy %NQROOT%\web.config.app %NQROOT%\web.config
IF EXIST %NQBIN%\EAP.Tools.exe.config.app copy %NQBIN%\EAP.Tools.exe.config.app %NQBIN%\EAP.Tools.exe.config
IF EXIST %NQBIN%\EAP.Scheduler.exe.config.app copy %NQBIN%\EAP.Scheduler.exe.config.app %NQBIN%\EAP.Scheduler.exe.config
IF EXIST %NQBIN%\EAP.VSAddIn.dll.config.app copy %NQBIN%\EAP.VSAddIn.dll.config.app %NQBIN%\EAP.VSAddIn.dll.config
IF EXIST %NQBIN%\EAP.CodeGen.dll.config.app copy %NQBIN%\EAP.CodeGen.dll.config.app %NQBIN%\EAP.CodeGen.dll.config


goto END

:ERROREXIT
@echo One or more errors occurred installing or the installation was cancelled.
GOTO FINAL

:END
@echo Install complete...
pause

:FINAL
```

It’s effectively the same as before, but we’ve added instructions to copy correctly configured config files over the generic installed set.

Run the batch file, install.bat to verify that the config files are copied from the config.app equivalent files

Day To Day Updates

Once you have set up the environment for the first time, you can keep the NetQuarry platform up to date as frequently as you need.

Simply download the latest NetQuarry.msi file to C:\NetQuarry\Customers\<Customer Name>\Setup and then run the install.bat file from the same folder.

That will update your platform

Get the latest code and meta from SourceOffSite from your Source database

Go to C:\NetQuarry\Customers\<Customer Name>\Source folder and run the batch file debug_build.bat.

This will build all of the customer source code in a debug environment.

Go to folder C:\NetQuarry\Customers\<Customer Name>\Database, run the batch file, reload-dbname_meta.bat

This will reload the metadata and update the operation schema. IMPORTANT – This will remove any metadata changes you have as a work in progress. If you run this batch file you will lose any unsaved work. If you want to save what you have worked on, check out the modified modules, save from studio and then run the batch file.

Setting Up Production Environment

The setup of a production environment is a relatively simple task. This document will describe how those tasks would be performed if the production environment consisted of a SQL server cluster and a web farm environment.

SQL Cluster: Two database servers SQL_1 and SQL_2 set up as a cluster, called SQL_Cluster.

Web Farm: Three application servers Web_1, Web_2, Web_3

The initial decision to make is to choose the machine from which to perform the installation. There are certain requirements for a NetQuarry installation to take place.

For a database to be modified, the batch files that modify the database schema and metadata require the **osql** utility that comes with a SQL Server installation. Therefore the machine running the database scripts must have a SQL Server instance. It doesn't have to be the same instance as the production database. In order for the environment that runs the database scripts to be configured correctly, it must have IIS installed on the same machine.

This gives you two choices in terms of setup requirements. You run the setup from a database server with IIS installed, or you run the setup from a web server with a SQL Server installation. In this document, we will assume that we are going to set up from a database server with an instance of IIS installed.

Prerequisites

Database Cluster

- Microsoft SQL Server 2005 (or later)
- Internet Information Services (version 5.0 for Windows XP, version 6.0 for Windows 2003)
- PsTools suite (http://technet.microsoft.com/en-us/sysinternals/bb896649.aspx)

Application Servers

- Internet Information Services (version 5.0 for Windows XP, version 6.0 for Windows 2003)

Domain User/Access

- The person logging onto the Database server must be in a role that has administrative rights to the database server and application servers
- The application servers must be accessible from the database server (e.g. must be able to reference \\Web_1\c$ to access the files\folders on the three app servers).

Setup

Folder Structure

The first step is to create a folder on the database server where you will perform the installation

Create the following folder structure. C:\NetQuarry, C:\NetQuarry\Rollback, C:\NetQuarry\Config, C:\NetQuarry\Config\Bin

Then, create a folder called C:\PsTools and unzip the PsTools executables into that folder.

Batch Files

Now you will create a set of batch files on the C:\NetQuarry folder that will perform the installation. The names of the batch files have generic names, as does the content of the batch files. To make the batch files work in your environment, make the following replacements where appropriate

{{appname}} -> the name of your application

{{Customer_name}} -> your company name

{{dbname_meta}} -> the name of your metadata database

{{dbname_data}} -> the name of your operational data database

install-all.bat

Create a batch file called install-all.bat with the following content. This is the main batch file that will perform the production server install to the database cluster and web farm

```
@echo off

@echo ******************************************************************
@echo Installing application locally
@echo ******************************************************************

call install-{{appname}}-application.bat

@echo ******************************************************************
@echo Installing databases locally
@echo ******************************************************************

call install-{{appname}}-databases.bat

cd\NetQuarry

call c:\NetQuarry\install-remote.bat \\WEB_1
call c:\NetQuarry\install-remote.bat \\WEB_2
call c:\NetQuarry\install-remote.bat \\WEB_3

@echo **********************************************************************
@echo Disabling Scheduler Service on SQL_CLUSTER
@echo **********************************************************************

call c:\pstools\psservice \\SQL_CLUSTER setconfig "NetQuarry Scheduler" disabled

@echo **********************************************************************
@echo Starting service on App servers
@echo **********************************************************************

call c:\pstools\psservice \\WEB_1 start "NetQuarry Scheduler"
call c:\pstools\psservice \\WEB_2 start "NetQuarry Scheduler"
call c:\pstools\psservice \\WEB_3 start "NetQuarry Scheduler"

@echo **********************************************************************
@echo Install complete
@echo **********************************************************************

pause
```

install-{{appname}}-application.bat

Create a batch file called install-{{appname}}-application.bat with the following content. This batch file will perform an uninstall and install of the web application on the target server.

```
@echo off

@echo ---------------------------------------------------------------------------------
@echo Starting production install
@echo ---------------------------------------------------------------------------------

@echo ---------------------------------------------------------------------------------
@echo Stopping NetQuarry Scheduler
@echo ---------------------------------------------------------------------------------
call net stop "NetQuarry Scheduler"

@echo ---------------------------------------------------------------------------------
@echo Killing potential running tasks...
@echo ---------------------------------------------------------------------------------
call taskkill /IM EAP.Tools.exe /F
call taskkill /IM EAP.LogView.exe /F
call taskkill /IM EAP.Scheduler.exe /F

@echo ---------------------------------------------------------------------------------
@echo Recycling application pools...
@echo ---------------------------------------------------------------------------------
call recycle.bat

@echo ---------------------------------------------------------------------------------
@echo Un-installing NetQuarry Platform...
@echo ---------------------------------------------------------------------------------
call "%SystemRoot%\System32\MsiExec.exe" /x {642ADDFC-6BBC-4C4B-A8C9-377727C18024} /passive

@echo ---------------------------------------------------------------------------------
@echo Un-installing {{Customer_name}} components...
@echo ---------------------------------------------------------------------------------
call "%SystemRoot%\System32\MsiExec.exe" /x {19CA63A9-DE00-43F6-AA92-5B2F5DC39E03} /passive

@echo ---------------------------------------------------------------------------------
@echo Installing NetQuarry Platform...
@echo ---------------------------------------------------------------------------------
call "%SystemRoot%\System32\msiexec.exe" /i "NetQuarry.msi" /log install-netquarry.log TARGETVDIR=V4
/passive

IF NOT EXIST "%NQBIN%\EAP.Core.DLL" GOTO ERROREXIT

@echo ---------------------------------------------------------------------------------
@echo Installing {{Customer_name}} Components...
@echo ---------------------------------------------------------------------------------
call "%SystemRoot%\System32\msiexec.exe" /i "{{appname}}.msi" /log install-{{appname}}.log
TARGETDIR=%NQROOT% /passive

IF NOT EXIST "%NQROOT%\Apps\{{appname}}\bin\{{Customer_name}}.Data.dll" GOTO ERROREXIT

goto END

:ERROREXIT
@echo ---------------------------------------------------------------------------------
@echo One or more errors occurred installing or the installation was cancelled.
@echo ---------------------------------------------------------------------------------
GOTO FINAL

:END
@echo Install complete...

:FINAL
@echo ---------------------------------------------------------------------------------
@echo Installation complete
@echo ---------------------------------------------------------------------------------
```

install-{{appname}}-databases.bat

Create a batch file called install-{{appname}}-databases.bat with the following content. This batch file will run the schema and metadata scripts on the database cluster.

```
@echo ----------------------------------------------------------------------------------
@echo Installing {{dbname_meta}}
@echo ----------------------------------------------------------------------------------

cd %NQROOT%\Database\Scripts
call {{dbname_meta}}-install-prod.bat


@echo ----------------------------------------------------------------------------------
@echo Updating operational schema ({{dbname_data}})
@echo ----------------------------------------------------------------------------------

cd %NQROOT%\Database\Scripts
call osql –S SQL_CLUSTER -E -d {{dbname_data}} -i {{dbname_data}}-alter-schema.sql -n
call osql –S SQL_CLUSTER -E -d {{dbname_data}} -i {{dbname_data}}-reindexing.sql -n
call osql –S SQL_CLUSTER -E -d {{dbname_data}} -i {{dbname_data}}-views.sql -n
call osql –S SQL_CLUSTER -E -d {{dbname_data}} -i {{dbname_data}}-procs.sql -n
call osql –S SQL_CLUSTER -E -d {{dbname_data}} -i {{dbname_data}}-migration.sql -n
call osql –S SQL_CLUSTER -E -d {{dbname_meta}} -i {{dbname_meta}}-fixup.sql -n

@echo ----------------------------------------------------------------------------------
@echo Setting production values in metadata ({{dbname_meta}})
@echo ----------------------------------------------------------------------------------
call osql –S SQL_CLUSTER -E -d {{dbname_meta}} -i
C:\NetQuarry\{{dbname_meta}}-production-properties.sql -n


@echo ----------------------------------------------------------------------------------
@echo Installing studio repository...
@echo ----------------------------------------------------------------------------------
cd %NQROOT%\Database\Scripts
call %NQROOT%\Database\Scripts\db-install.bat SQL_CLUSTER
```

install-remote.bat

Create a batch file called install-remote.bat with the following content. This batch file will perform the installation of the app servers remotely from the database server.

```
@echo *********************************************************************
@echo Copying installation files to %1
@echo *********************************************************************

cd\NetQuarry

call copy NetQuarry.msi %1\c$\NetQuarry
call copy {{appname}}.msi %1\c$\NetQuarry
call copy install-{{appname}}-application.bat %1\c$\NetQuarry
call copy uninstall-{{appname}}-application.bat %1\c$\NetQuarry
call copy recycle.* %1\c$\NetQuarry


@echo *********************************************************************
@echo Installing on %1
@echo *********************************************************************
call c:\pstools\psexec.exe %1 -w c:\NetQuarry c:\netquarry\install-{{appname}}-application.bat


@echo *********************************************************************
@echo Copying config files to %1
@echo *********************************************************************
cd\NetQuarry
call copy c:\NetQuarry\Config\*.config %1\c$\Inetpub\wwwroot\v4
call copy c:\NetQuarry\Config\bin\*.config %1\c$\Inetpub\wwwroot\V4\bin

del %1\c$\Inetpub\wwwroot\V4\bin\EAP.Tools.exe.manifest /q /f

del %1\C$\NetQuarry\*.* /q /f

iisreset %1

@echo *********************************************************************
@echo Install to %1 complete
@echo *********************************************************************
```

recycle.bat

Create a batch file called recycle.bat with the following content. This batch file will recycle the application pool on the app server.

```
@echo off
call cscript recycle.vbs
```

recycle.vbs

Create a script file called recycle.vbs with the following content. This script file will recycle all of the application pools on the app server.

```
strComputer = "."
Set objWMIService = GetObject _
   ("winmgmts:{authenticationLevel=pktPrivacy}\\" _
      & strComputer & "\root\microsoftiisv2")

Set colItems = objWMIService.ExecQuery("Select * From IIsApplicationPool")

For Each objItem in colItems
   WScript.Echo "Recycling " + objItem.Name
   objItem.Recycle
Next
```

{{dbname_meta}}-production-properties.sql

Create a sql file called {{dbname_meta}}-production-properties.sql with the following content. This script file will set up the application to be in production mode, rather than debug mode.

```
UPDATE xmt_properties set prop_value = 'True' WHERE param_id IN (SELECT param_id FROM xmt_parameters
WHERE param_nm = 'IsProduction')
GO
UPDATE xmt_properties set prop_value = '1' WHERE param_id IN (SELECT param_id FROM xmt_parameters WHERE
param_nm = 'MetadataCache')
GO
```

Config files

The next step in setting up the production environment is to create a set of template config files that will be copied over to each app server by the install batch files. That ensures that each server uses the environment specific configuration and not the default installed configuration.

The config files are copied over to each remote app server in the batch file install-remote.bat

The following section provides example and descriptions of the config files.

To make the batch files work in your environment, make the following replacements where appropriate

{{appname}} -> the name of your application

{{Customer_name}} -> your company name

{{dbname_meta}} -> the name of your metadata database

{{dbname_data}} -> the name of your operational data database

{{external_root_url}} -> the external url of the application

{{uid}} -> the database user

{{pwd}} -> the password for the database user

{{app_user_id}} -> the id of an application user

{{app_user_password}} -> the password of an application user

web.config

This is the config file for the application. The default web.config file will be located in the C:\NetQuarry\Config folder.

```
<?xml version="1.0"?>
<configuration>
  <configSections>
    <section name="hostMappings"
         type="System.Configuration.NameValueSectionHandler" />
  </configSections>

  <appSettings>
    <!—{{appname}}-->
    <add key="{{appname}}_repository_connect"
value="Provider=SQLOLEDB;Data Source=SQL_CLUSTER;Initial Catalog={{dbname_meta}};User ID={{uid}};Password={pwd}}" />
    <add key="{app_name}}_connect"
value="Provider=SQLOLEDB;Data Source= SQL_CLUSTER;Initial Catalog={{dbname_data}};User ID={{uid}};Password={{pwd}};Max Pool Size=200" />

    <!-- web log path, note that you should NEVER store the log in the application or bin folder  -->
    <add key="devlogPath" value="%NQROOT%\logs\web.nql" />

    <!-- Options for the log from the DevLogOptions enum.
         This value should be a decimal number that represents a mask of one or more
         of the DevLogOptions values.
         For example, to turn off Debug level logs, set the value to 1.
```

```
        NoDebug|NoSQL == 5
        NoDebug|NoInfo == 3 (note that this turns off SQL and Timing entries as well)
        NoDebug|AutoFlush = 33 (0x00000001 | 0x00000020)
        AutoFlush == 32 (by default the file is buffered)
        Synchronize == 64 (by default the file allows overlapping writes)
    -->
    <add key="devLogOptions" value="1" />

    <!--
      You may also set the max size of a file (before truncation) and the max number of backup files
      to create when a file is truncated.
      The MaxFileSize is in bytes, in decimal format: (1048576 == 1 MB). The default is 10 MB (0x00A00000 or 10,485,760)
    -->
    <add key="devLogMaxFileSize" value="10485760" />
    <add key="devLogNumBackupFiles" value="10" />

    <!-- the default application to use when calculating the default page and repository -->
    <add key="defaultAppKey" value="{{appname}}" />
    <add key="{{appname}}_DefaultPage" value="Apps/{{appname}}/default.htm" />

    <!--
      Indicates whic servers are in this farm.  Used when clearing cache to make sure all servers in farm
      have their cache's cleared at the same time
      Provide a semi-colon separate list of root url properties on which to communicate cache update over http
    -->
        <add key="farmServers" value="web_1/v4;web_2/v4;web_3/v4" />
  </appSettings>

  <!--
    This section is used to map a host name to a default application.
    For example:
  -->
  <hostMappings>
    <add key="localhost" value="{{appname}}" />
  </hostMappings>

  <runtime>
    <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
      <probing privatePath="bin;Apps/{{appname}}/bin" />
      <dependentAssembly>
        <assemblyIdentity name="EAP.IO.CSV" version="1.1.0.0" />
      </dependentAssembly>
    </assemblyBinding>
  </runtime>
  <system.web>

    <urlMappings enabled="true">
      <!-- {{appname}} password reset from the home page -->
      <add url="~/apps/{{appname}}/lostpassword.aspx" mappedUrl="~/Handler.ashx?appkey={{appname}}pr&amp;uid={{appname}}pr&amp;req=login" />
      <add url="~/resetpassword.aspx" mappedUrl="~/Handler.ashx?req=nav&amp;mop={{app_name}}_password!lostpassword" />
    </urlMappings>

    <!-- because we restart on demand,
      this key is to ensure that any validation done on the viewstate isn't domain generated -->
    <machineKey
       validationKey="30C49272199C10B16A14E93F7BC08B7AB306BB33C55B89778CDCDEB99DEAF86D534610166A790D02CB45B25E900D57E4866BB175FE7126FACD82921973AFD99B"
       decryptionKey="12759A5473061515B703D0323489B6D188922165E4C84FB4"
       validation="SHA1" />

    <!--
                     enableEventValidation allows us to modify picklists dynamically in client-side
                     javascript and NOT have .Net throw an error.
          -->
    <pages enableEventValidation="false" viewStateEncryptionMode="Never" enableViewStateMac="false">
    </pages>

    <!--  DYNAMIC DEBUG COMPILATION
          Set compilation debug="true" to enable ASPX debugging.  Otherwise, setting this value to
          false will improve runtime performance of this application.
          Set compilation debug="true" to insert debugging symbols (.pdb information)
          into the compiled page. Because this creates a larger file that executes
          more slowly, you should set this value to true only when debugging and to
          false at all other times. For more information, refer to the documentation about
          debugging ASP.NET files.
    -->
    <compilation defaultLanguage="c#" debug="true">
    </compilation>
    <!--  CUSTOM ERROR MESSAGES
          Set customErrors mode="On" or "RemoteOnly" to enable custom error messages, "Off" to disable.
          Add <error> tags for each of the errors you want to handle.

          "On" Always display custom (friendly) messages.
          "Off" Always display detailed ASP.NET error information.
          "RemoteOnly" Display custom (friendly) messages only to users not running
           on the local Web server. This setting is recommended for security purposes, so
           that you do not display application detail information to remote clients.
    -->
    <customErrors mode="Off"/>
    <!--  AUTHENTICATION
          This section sets the authentication policies of the application. Possible modes are "Windows",
          "Forms", "Passport" and "None"

          "None" No authentication is performed.
          "Windows" IIS performs authentication (Basic, Digest, or Integrated Windows) according to
           its settings for the application. Anonymous access must be disabled in IIS.
        "Forms" You provide a custom form (Web page) for users to enter their credentials, and then
         you authenticate them in your application. A user credential token is stored in a cookie.
        "Passport" Authentication is performed via a centralized authentication service provided
         by Microsoft that offers a single logon and core profile services for member sites.
    -->
    <authentication mode="None"/>
    <!--  AUTHORIZATION
          This section sets the authorization policies of the application. You can allow or deny access
          to application resources by user or role. Wildcards: "*" mean everyone, "?" means anonymous
          (unauthenticated) users.
    -->
    <authorization>
      <allow users="*"/>
      <!-- Allow all users -->
      <!--  <allow     users="[comma separated list of users]"
                             roles="[comma separated list of roles]"/>
                <deny      users="[comma separated list of users]"
                             roles="[comma separated list of roles]"/>
            -->
    </authorization>
    <!--  APPLICATION-LEVEL TRACE LOGGING
          Application-level tracing enables trace log output for every page within an application.
          Set trace enabled="true" to enable application trace logging.  If pageOutput="true", the
          trace information will be displayed at the bottom of each page.  Otherwise, you can view the
          application trace log by browsing the "trace.axd" page from your web application
          root.
    -->
    <trace enabled="false" requestLimit="10" pageOutput="true" traceMode="SortByTime" localOnly="true"/>
    <!--  SESSION STATE SETTINGS
          By default ASP.NET uses cookies to identify which requests belong to a particular session.
          If cookies are not available, a session can be tracked by adding a session identifier to the URL.
          To disable cookies, set sessionState cookieless="true".
    -->
    <sessionState mode="InProc" stateConnectionString="tcpip=127.0.0.1:42424" sqlConnectionString="data source=127.0.0.1;Trusted_Connection=yes"
cookieless="false" timeout="10"/>
    <!--  GLOBALIZATION
          This section sets the globalization settings of the application.
    -->
    <globalization requestEncoding="utf-8" responseEncoding="utf-8"/>

    <!--
      Global handlers. Note that you MUST include both of these, in order.
    -->
    <httpHandlers>
      <add verb="GET" path="WebResource.axd" type="System.Web.Handlers.AssemblyResourceLoader" />
      <add verb="*" path="*" type="NetQuarry.InitHandler" validate="false" />
    </httpHandlers>

    <xhtmlConformance mode="Legacy"/>
  </system.web>
</configuration>
```

The text in bold black, are options you may want to modify based on your production environment.   Just to summarize those options:- Max Pool Size for database connection, Devlog settings for size and number of files to keep, Specify the app servers in the web farm.

The farmServers setting allows the app servers to communicate cache flushing requests to each other behind any load balancing system as the initial cache flush request from outside is only to one server.

EAP.Reporting.config

This is the config file for the reporting services. The default EAP.Reporting.config file will be located in the C:\NetQuarry\Config\Bin folder. You only need this config file if you are using SSRS reporting and the reports in your application are configured to have picklist resolution of key values to display text.

The authentication information is used when the reporting services needs to call back to the application to get the required picklist information.

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <!--
    Reporting config file.
    The configSections below are NetQuarry specific.
    If you add an application you need to add a section handler as well.
    AppIDs are CASE-SENSITIVE!
  -->
  <configSections>
    <section name="{{appname}}"
         type="System.Configuration.NameValueSectionHandler" />
  </configSections>

  <{{appname}}>
    <!-- connection strings -->
    <add key="repository_connect" value="Provider=SQLOLEDB;Data Source=SQL_CLUSTER;Initial Catalog={{dbname_meta}};Integrated Security=SSPI" />
    <add key="{{appname}}_connect" value="Provider=SQLOLEDB;Data Source= SQL_CLUSTER;Initial Catalog={{dname_data}};Integrated Security=SSPI" />

    <!-- optional, normally set by the web application context -->
    <add key="RootPath" value="%NQROOT%" />
    <add key="RootURL" value="{{external_root_url}}" />

    <!-- namespace to use for the generated TypedMapper classes -->
    <add key="Namespace" value="{{Customer_name}}.Data" />

    <add key="AuthenticationProvider1" value="NetQuarry.Security.GenericAuthenticationProvider" />
    <add key="UserID" value="{{app_user_id}}" />
    <add key="Password" value="{{app_user_password}}" />
  </{{appname}}>
</configuration>
```

EAP.Scheduler.exe.config

This is the config file for the scheduler application. The default EAP.Scheduler.exe.config file will be located in the C:\NetQuarry\Bin folder

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <!--
    The configSections below are NetQuarry specific.
    If you add an application you need to add a section handler as well.
  -->
  <configSections>
    <section name="serviceSettings"
       type="System.Configuration.NameValueSectionHandler" />

      <section name="activeTasks"
         type="System.Configuration.NameValueSectionHandler" />

      <section name="{{appname}}"
         type="System.Configuration.NameValueSectionHandler" />
  </configSections>

  <serviceSettings>
      <!-- the installer puts the log here, you can change it, but NEVER put it in the bin or root folder
                                  or the asp.net worker process will be restarted every time this log is written to -->
      <add key="devLogPath" value="%NQROOT%\logs\EAP.Scheduler.exe.nql" />
      <add key="devLogOptions" value="1" />
      <add key="devLogMaxFileSize" value="10485760" />
      <add key="devLogNumBackupFiles" value="10" />
  </serviceSettings>

  <activeTasks>
    <add key="{{appname}}" value="{{appname}}" />
  </activeTasks>

  <<{{appname}}>
    <add key="AppID" value="{{appname}}" />
    <add key="repository_connect" value="Provider=SQLOLEDB;Data Source=SQL_CLUSTER;Initial Catalog={{dbname_meta}};User ID={{uid}};Password={{pwd}}" />
    <add key="{{appname}}_connect" value="Provider=SQLOLEDB;Data Source=SQL_CLUSTER;Initial Catalog={{dbname_data}};User ID={{uid}};Password={{pwd}}" />
    <add key="RootPath" value="%NQROOT%" />
    <add key="RootURL" value="{{external_root_url}}" />
  </<{{appname}}>
</configuration>
```

EAP.Tools.exe.config

This is the config file for the studio application. The default EAP.Tools.exe.config file will be located in the C:\NetQuarry\Bin folder

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <!-- the installer puts the log here, you can change it, but NEVER put it in the bin or root folder
                                        or the asp.net worker process will be restarted every time this log is written to -->
    <add key="devLogPath" value="%NQROOT%\logs\EAP.Tools.exe.nql" />

    <!-- Options for the log from the DevLogOptions enum.
         This value should be a decimal number that represents a mask of one or more
         of the DevLogOptions values.
         For example, to turn off Debug level logs, set the value to 1.
         NoDebug|NoSQL == 5
         NoDebug|NoInfo == 3 (note that this turns off SQL and Timing entries as well)
         NoDebug|AutoFlush == 33 (0x00000001 | 0x00000020)
         AutoFlush == 32 (by default the file is buffered)
    -->
    <add key="devLogOptions" value="32" />
    <!-- always the studio repository -->
    <add key="repository_connect" value="Provider=SQLOLEDB;Data Source=SQL_CLUSTER;Initial Catalog=nq_studio;USER ID={{uid}];PASSWORD={{pwd}}" />
    <add key="{{appname}}_connect" value="Provider=SQLOLEDB;Data Source= SQL_CLUSTER;Initial Catalog={{dbname_data}};USER ID={{uid}};PASSWORD={{pwd}}" />
    <add key="MetadataPath" value="%NQMETA%" />
  </appSettings>
  <system.windows.forms jitDebugging="true" />
</configuration>
```

Localization and Translation

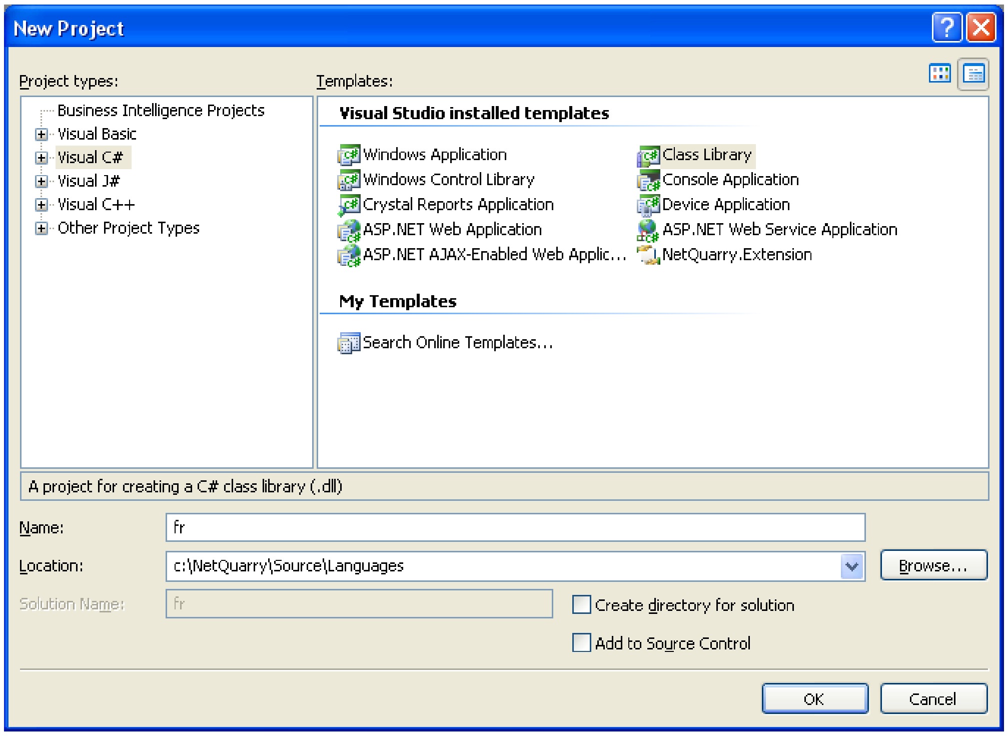
Applications developed on the NetQuarry Platform are multi-lingual and support globalization of dates, numbers, calendars, and currencies. User interface strings displayed in an application developed on the Platform are fully localized and can be translated into any language.

Strings are stored in two formats. Platform specific strings such as error messages and built-in tooltips are localized via a .NET resource file. User interface strings such as field and page captions, mapper and application strings are stored in application specific metadata.

Resource Files

Resource files are .NET assemblies that contain strings used by the Platform. The platform ships with a single resource file for the base English language: EAP.Resources.en.dll. If you are translating into another language you translate the resource strings to the target language. Once the strings are translated, you will need to compile the strings into the correct assembly format and name. For example, to create a resource file for French, follow these steps:

1. Create a new class library project named EAP.Resources.**fr**.vsproj.



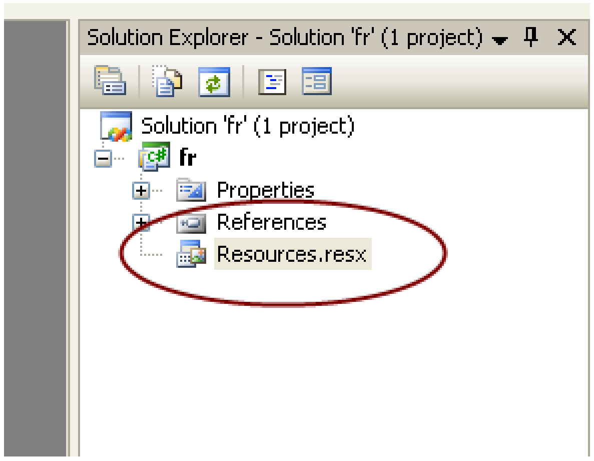
A project for creating a C# class library (.dll)

Name: fr

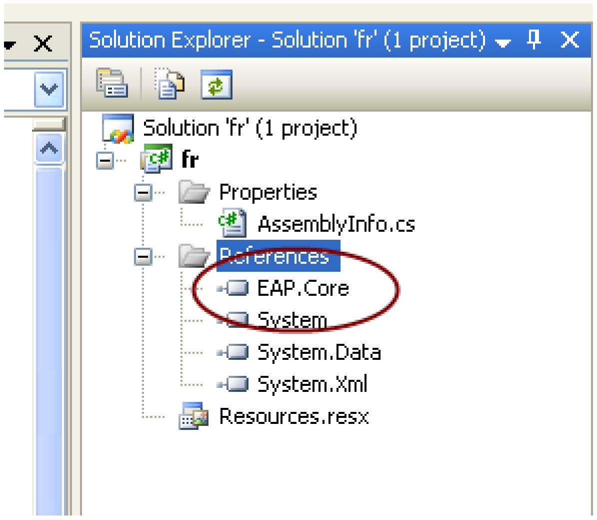
Location: c:\NetQuarry\Source\Languages

Solution Name: fr

2. Add the resources.resx file (installed to %NQROOT%/Resources) to the project.


Solution Explorer - Solution 'fr' (1 project)
Solution 'fr' (1 project)
fr
Properties
References
Resources.resx

3. Add a reference to EAP.Core.



4. In the Properties\AssemblyInfo.cs add the following:

```
using System.Reflection;
using System.Runtime.CompilerServices;
using System.Runtime.InteropServices;

[assembly: AssemblyTitle("NetQuarry Core Resources - French")]
[assembly: AssemblyDescription("")]
[assembly: AssemblyCompany(NetQuarry.Build.Info.Company)]
[assembly: AssemblyProduct(NetQuarry.Build.Info.Product)]
[assembly: AssemblyCopyright(NetQuarry.Build.Info.Copyright)]
[assembly: AssemblyTrademark(NetQuarry.Build.Info.Trademark)]
[assembly: AssemblyCulture("")]
[assembly: ObfuscateAssembly(false)]
[assembly: AssemblyVersion(NetQuarry.Build.Info.AssemblyVersion)]
[assembly: AssemblyFileVersion(NetQuarry.Build.Info.FileVersion)]
[assembly: AssemblyDelaySign(false)]
[assembly: System.Security.AllowPartiallyTrustedCallers]
```

5. Set the assembly properties so that the default namespace is NetQuarry.Globalization and the name of the assembly is EAP.Resources.**fr**.

6. Build the assembly and deploy it to the %NQBIN% folder. The typical way to do this is to add the assembly to your custom installer.

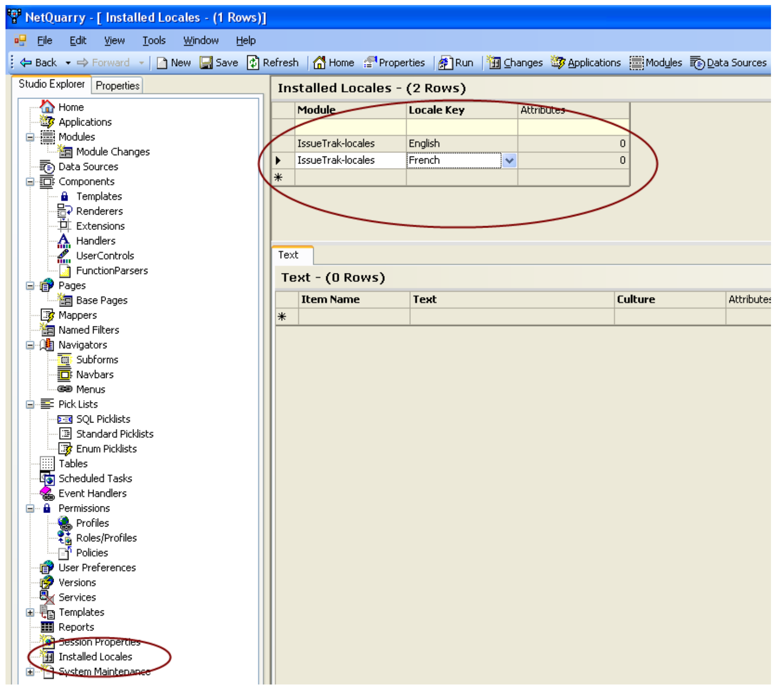
Translating Metadata Strings

Strings in the NetQuarry Studio are always stored and written in English. When you export modules, the metadata is saved ONLY with English strings.

When you add a language, you add an installed locale (culture), copy the English strings, and export the culture. The export produces a .META file (a NetQuarry metadata file) that can be loaded during installation. The export also produces an Excel 2003 spreadsheet that is suitable for use by a translation service.

To add a new supported locale, follow these steps:

1. From the tree, click on Installed Locales and add a new locale (e.g. French):



Installed Locales - (2 Rows)

Module	Locale Key	Attributes
IssueTrak-locales	English	0
IssueTrak-locales	French	0

Text - (0 Rows)

Item Name	Text	Culture	Attributes

2. Right Click on the new locale and choose **Create Strings**. Strings will be copied from the base language (English) and copied to the new language (French).
3. Right Click on the new locale and choose **Export Locale**. This will create the module (named fr-culture.meta) and the translation Excel file (named fr-culture.xml). Note that the Excel file is in Excel 2003 XML format (ExcelML) and can be read by both Excel 2003 and Excel 2007. ***The translation file should be kept in Excel 2003 XML format for import purposes.***
4. Using the Excel file, translate or have translated the strings. When strings are translated, you should mark the column 'Translated' with a Yes or No. This will be imported so that you can tell when new strings are added.
5. After translation, right click on the new locale and choose **Import Locale.** You should choose the Excel 2003 XML file and import the strings.

6. After the strings are imported, you should choose **Export Locale** again to produce a translated module for loading into your test or production database. (This file should generally be checked in.)

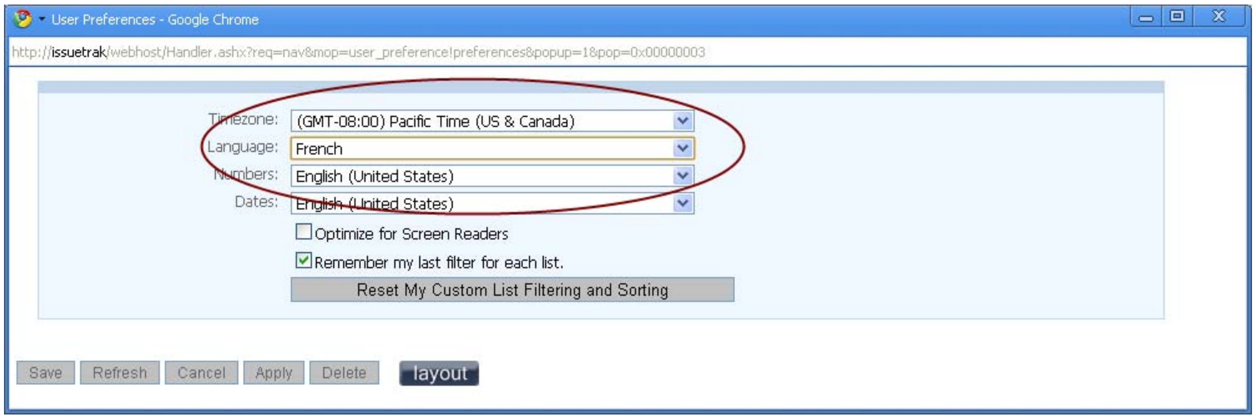
Deploying a new language

To deploy a new language you must add the compiled resources file to your custom installer and install the resource assembly (named correctly) into the %NQBIN% folder. You should install the <language>-culture.meta files into your /Apps/<appkey>/Metadata folder.

During install, add the following line to your install script:

```
SET METAUTILEXE=%NQBIN%"\EAP.Tools.MetaUtil.EXE"

call %METAUTILEXE% -file fr-culture.meta "Provider=SQLOLEDB;Data
     Source=.\sql2005;Initial Catalog=issues_meta;Integrated Security=SSPI"
```

This loads the strings into your metadata database and makes the language available in the list of available languages via the user preferences dialog:


User Preferences - Google Chrome
http://issuetrak/webhost/Handler.ashx?req=nav&mop=user_preference!preferences&popup=1&pop=0x00000003
Save
Refresh
Cancel
Apply
Delete
layout

Timezone: (GMT-08:00) Pacific Time (US & Canada)
Language: French
Numbers: English (United States)
Dates: English (United States)
☐ Optimize for Screen Readers
☑ Remember my last filter for each list.
Reset My Custom List Filtering and Sorting

Profiles and Permissions

The NetQuarry platform grants permissions to objects based on membership in a set of one or more Profiles. Profiles are metadata and runtime objects that are assigned to a user during authentication.

By default, permission to an object (such as a Page, Report, Navigation Target, Mapper, Field) is granted unless otherwise specified – e.g. a permissive model. For example, if you create a new page all profiles currently defined are granted Read (View) permission to the page.

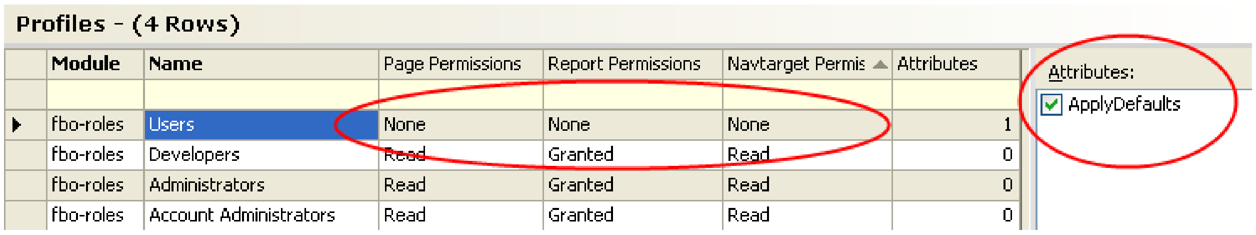
The platform also supports a restrictive model that can be defined by profile. For example, when a new Page is created for profiles with the ProfileAttrs.ApplyDefaults attribute set and the default permission for pages set to 'None' the profile will NOT be granted access to the page. In this case, you must explicitly grant the profile permission.

This idea is useful if you have a profile or profiles that you want to limit application access and you don't want to walk through hundreds of pages, reports, and navigators to make sure that the correct permissions are applied.

The restrictive model applies to Pages, Reports, and NavTargets. In general, these are the objects that present a danger to the security of an application. If you want to be 100% certain about your application's security, you should set the permission on all three objects to 'None.' It is nearly as safe to allow normal 'Read' access to NavTargets as NavTargets will not be rendered if the page they point to is restricted from the current user. The inverse is NOT true – e.g. a restricted NavTarget does NOT restrict access to the page it points to.

How to enable restrictive permissions on a profile

To enable a profile to support a restrictive permission model you simply need to set the default permissions for Pages, Reports, and NavTargets for the profile to 'None' and set the ApplyDefault attribute on the profile:

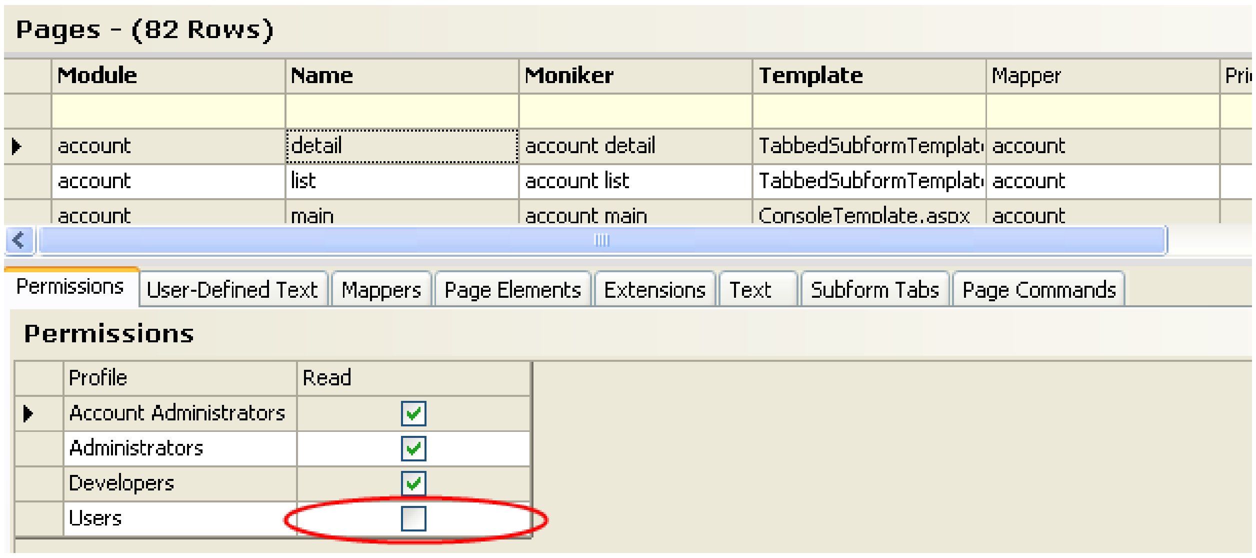


Profiles - (4 Rows)

Module	Name	Page Permissions	Report Permissions	Navtarget Permis	Attributes
fbo-roles	Users	None	None	None	1
fbo-roles	Developers	Read	Granted	Read	0
fbo-roles	Administrators	Read	Granted	Read	0
fbo-roles	Account Administrators	Read	Granted	Read	0

Attributes:
[✓] ApplyDefaults

Once the profile has been defined this way, all objects that do NOT have an explicit set of permissions defined will by default be off for the profile (in the above example). You will see an immediate effect in the studio, as seen below:



Pages - (82 Rows)

	Module	Name	Moniker	Template	Mapper	Pri
▶	account	detail	account detail	TabbedSubformTemplat	account	
	account	list	account list	TabbedSubformTemplat	account	
	account	main	account main	ConsoleTemplate.aspx	account	

Permissions

	Profile	Read
▶	Account Administrators	✓
	Administrators	✓
	Developers	✓
	Users	

In the above example, the account!detail page has NO explicit permissions defined but the Users profile is NOT enabled. This is true for all pages unless explicit permissions are defined (the box is checked in the studio).

How does this affect permissions of existing objects?

If you realize that you need to set up a profile with a restrictive permission halfway through development, what happens?   Well, for objects that already have specific permissions applied for that profile it cannot apply this default permission of deny read even if it looks like that permission is allow read on the object permissions.   This is because once a permission is set, then permissions records for all roles are saved for that object.

For object with no permissions specified (on any profile), then the default restrictive permission will be detected at runtime and the object will not be accessible to the user.

Here’s an example.

There are three profiles A, B, C. We have two pages, Page1, Page2

Page	Existing Permissions	Add Restrictive Default Permission	End Result
Page1	profileA – Read profileB – Read profileC – Read	profileC – DenyRead	profileA – Read profileB – Read profileC – DenyRead
Page2	profileA – DenyRead profileB – Read profileC – Read	profileC – DenyRead	profileA – DenyRead profileB – Read profileC – Read

Because there is one permission change on Page2, we actually write out three permission records, one for each profile. Therefore there is a physical record that says Page2 is visible to ProfileC.

Clearly, setting up of default restrictive permissions is something that should be specified at the beginning of development.

Creating New Profiles

There are two ways to create a new profile.

Duplicate

In the Studio, go to the Profiles and select a profile you want to copy. Right click and select Duplicate. This will create a new profile. During the duplicate, the studio creates an identical permission record for each object in the application where a permission record occurred for the original profile.

For example.

Page 1 has a DenyRead permission for ProfileA. You went and copied ProfileA to create (after renaming) ProfileB. When you go to the permissions of Page1, you will see that ProfileB also has a DenyRead permission for Page1.

Create a new Profile

In the Studio, go to the Profiles and simply enter a new profile into the insert row. When the new profile is created, the studio adds a new permission record to every object that has one or more existing permission records. When it adds a permission record, regardless of other permissions for that object, the new permission is to deny access on that object. If the

For example

Page 1 has a DenyRead permission for ProfileC and Read permission for ProfileD. Page2 has Read permission for ProfileC and ProfileD (there are no permissions records for Page2). You create a new profile ProfileE (not copied). When you go to permissions of Page1, you see the new permission for ProfileE with DenyRead access. On Page2, the permission for ProfileE is Read.

Warning!

NOTE: when you add a new profile to the application, you will find that virtually all of your application's metadata has been modified. If you do find the need to add new profiles at any point in the development process, it is important to coordinate this event with the development team. Ensure you have exclusive access to all the metadata module files so you can check in the permissions changes without risk of merge issues.