1 Package and Deployment

1.1 K2 Package and Deployment Overview

K2 Package and Deployment

K2 Package and Deployment provides the ability to package K2 artifacts, such as SmartObjects, Forms, Views and Workflows, from one environment (the source) into a package file stored locally or on a network drive. This package is then opened in another K2 environment (the target) where some or all of the artifacts in the package are deployed.

The following concepts are used to define the process of creating and deploying packages:

1. Source Environment: Typically a development environment where a K2 solution was designed and tested.
2. K2 Package: A file containing all the solution artifacts necessary for the solution to function in the target environment.
3. Target Environment: Typically a test/QA or production environment to which a K2 solution must be deployed.
4. Conflict Resolution: Steps required to successfully deploy a package to the target environment when there are artifact conflicts between source and target.

There are two types of references that can be packaged:

- A reference-only object. This refers to a dependent object required to successfully deploy a package where the object is 'packaged' only by a reference to it, as it is a non-deployable item. For example, a custom service type or an out-of-the box default item that will always be on the server. These objects must be present in the target environment in order for the package to deploy successfully.
- A reference and its associated object definition. This refers to a dependent object required to successfully deploy a package, where the dependent object need not exist on the target environment. It will be included in the package, and can be deployed.

Note: Packages created using Beta versions of K2 Package and Deployment are not compatible with this release. They will need to be re-created using the K2 Package and Deployment release version.

Supported Artifacts

Artifacts designed using K2 Studio, K2 for Visual Studio and the K2 Designer can be packaged and deployed. These include:

- SmartObjects.
- SmartObject Data.
- Workflows.
- Forms.
- Views.
- Supporting artifacts such as
  - Categories,
  - Roles,
  - Service instances,
  - Notification events and
  - Environment library fields.

Unsupported Artifacts

The following artifacts are not supported at this time:

- Artifacts utilizing Workflow Integration (such as InfoPath and SharePoint).
- Artifacts designed using the K2 Designer for SharePoint.
- Non-K2 artifacts (such as SharePoint lists and libraries).
- Custom environment configurations (for example, changes to configuration files like 'K2HostServer.exe.config', cannot be packaged. These configurations must be manually applied to the target environment).

Service Instances

When packaging and deploying Oracle, SQL Server, CRM or SharePoint Service Instances, be sure that the associated Oracle databases/SQL Databases/Entities/Lists and Libraries a) exist on the target environment and b) have the same schema before attempting to deploy the package.

When using K2 Package and Deployment with SalesForce Service Instances, an associated SalesForce Service Instance must exist on the target environment. You will have to rebind your packaged Service Instance to the existing Service Instance on the target, once the package has been deployed.

Custom Service Brokers

When using a custom Service Broker, be sure to register the broker at the target environment using a tool such as BrokerManagement.exe, or the SmartObject Service Tester utility.

Security

- All packaging and deployment actions take place using the K2 Server Service account.
- However, all calls to the target server are validated using the users credentials. The user running the deployment tool requires 'Export' rights on the target server. If SmartObject security has been set, then the deployment user requires 'Publish SmartObject' permissions as well.
- After being granted deployment rights, the user will be able to deploy any packaged artifact to the target server, as this will happen within the K2 Server context.
- The deploying user subsequently becomes the owner of that process, and will need to enable sharing prior to other users being able to see/edit the process.

Localization and Globalization

- Date and time values are always generated on the server time zone.
- Any dates required on the database are supplied by the server.
- No time-zone information is stored in the database as all date values are local-time for the servers, and all servers in the current farm should be in the same time zone.
- All comparisons are case-insensitive.

SmartObjectData

SmartObjectData allows the packaging of SmartBox data for the given SmartObject within the source K2 environment. Note that this functionality is only available to SmartBox SmartObject data. SmartObject data derived from other sources cannot be packaged at this time.

SmartObjectData is deselected by default so that SmartObject data from one environment is not automatically deployed to a new environment. This is the default behavior in order to prevent deployment packages from becoming oversized (due to the potentially large amount of data stored within a SmartObject). If the data contained within a SmartObject must be packaged, the relevant SmartObject node must be opened and the required SmartObject Data must be manually selected.

SmartObjectData can only be deployed as part of the parent SmartObject artifact. If a SmartObject definition has already been deployed to the target environment, you will need to configure the deployment action to either overwrite the existing SmartObject definition, or use the existing definition.

1.2 K2 Package and Deployment Installation

K2 Package and Deployment Installation

The K2 Package and Deployment tool uses a Microsoft Management Console plugin, and optionally a PowerShell snapin for deployment scripting. The Server Components must be installed on the K2 Server in the source and target environments. The Client Components may be installed on any server in the environment that is able to connect to the K2 Server and the K2 Database and has a K2 Client (designer) installed.

We recommend, however, that packaging and deployment only be performed on the K2 Server in order to reduce network traffic.

Prerequisites

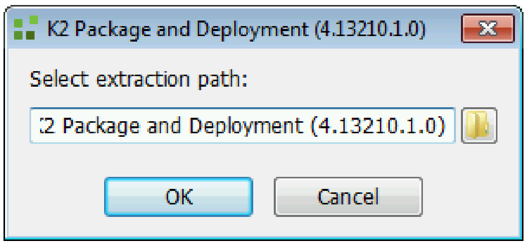
The following applications are required for installation of K2 Package and Deployment:

- K2 blackpearl version 4.6.5 or later.
- Microsoft Management Console (MMC) 3.0.
- K2 smartforms is an optional requirement for Package and Deployment. However, if K2 smartforms is installed, version 1.0.3 or higher must be used.

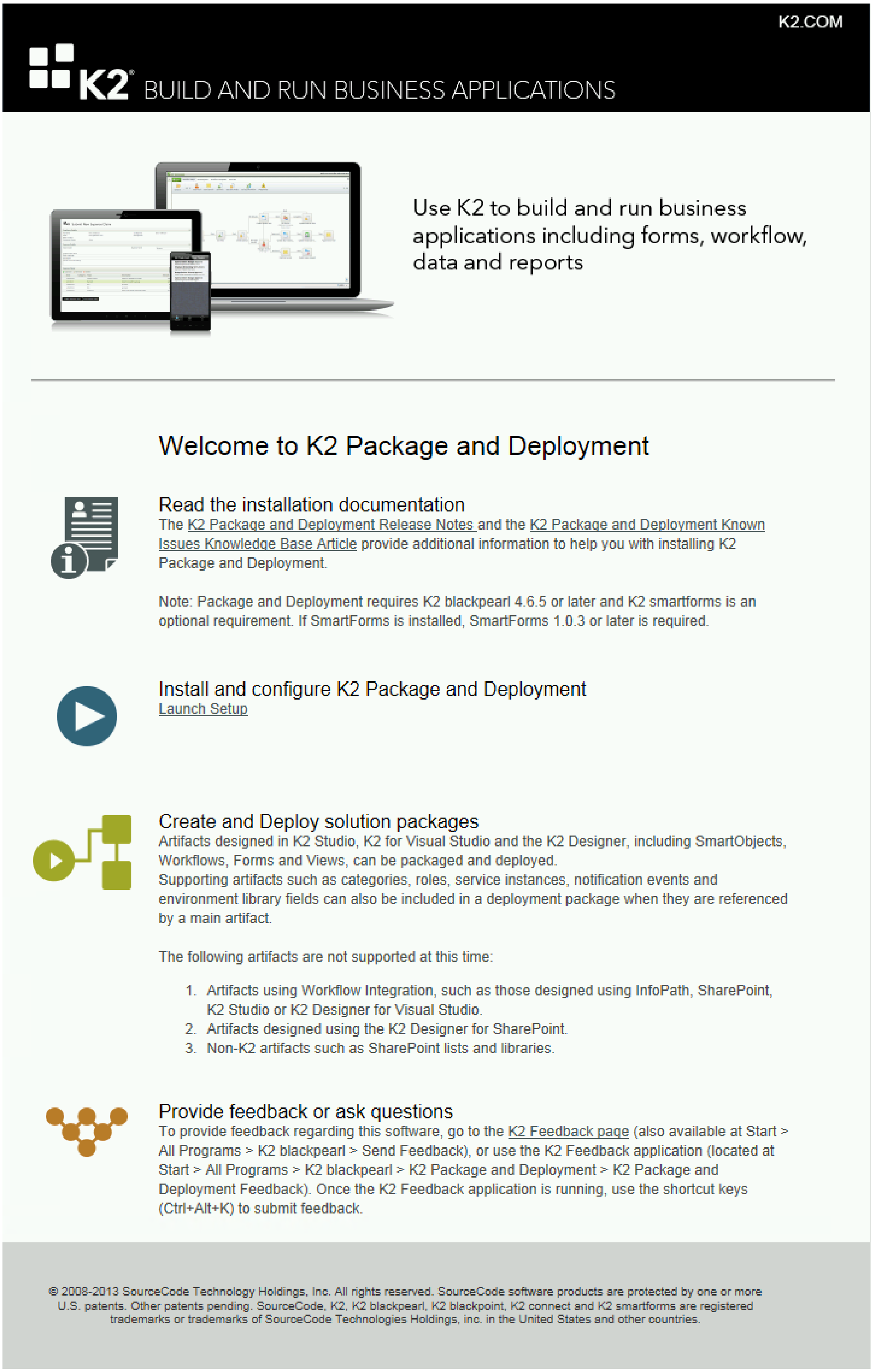
The source and target servers must both be running the same versions of K2 blackpearl, and K2 smartforms if applicable.

Installation steps for a new installation

1. After downloading the K2 Package and Deployment installation package, extract the installation package.


K2 Package and Deployment (4.13210.1.0)
Select extraction path:
:2 Package and Deployment (4.13210.1.0)
OK
Cancel

2. The Installer Splash Screen will load. Click the Launch Setup link to start the installer.



Use K2 to build and run business applications including forms, workflow, data and reports

Welcome to K2 Package and Deployment

Read the installation documentation

The K2 Package and Deployment Release Notes and the K2 Package and Deployment Known Issues Knowledge Base Article provide additional information to help you with installing K2 Package and Deployment.

Note: Package and Deployment requires K2 blackpearl 4.6.5 or later and K2 smartforms is an optional requirement. If SmartForms is installed, SmartForms 1.0.3 or later is required.

Install and configure K2 Package and Deployment

Launch Setup

Create and Deploy solution packages

Artifacts designed in K2 Studio, K2 for Visual Studio and the K2 Designer, including SmartObjects, Workflows, Forms and Views, can be packaged and deployed.
Supporting artifacts such as categories, roles, service instances, notification events and environment library fields can also be included in a deployment package when they are referenced by a main artifact.

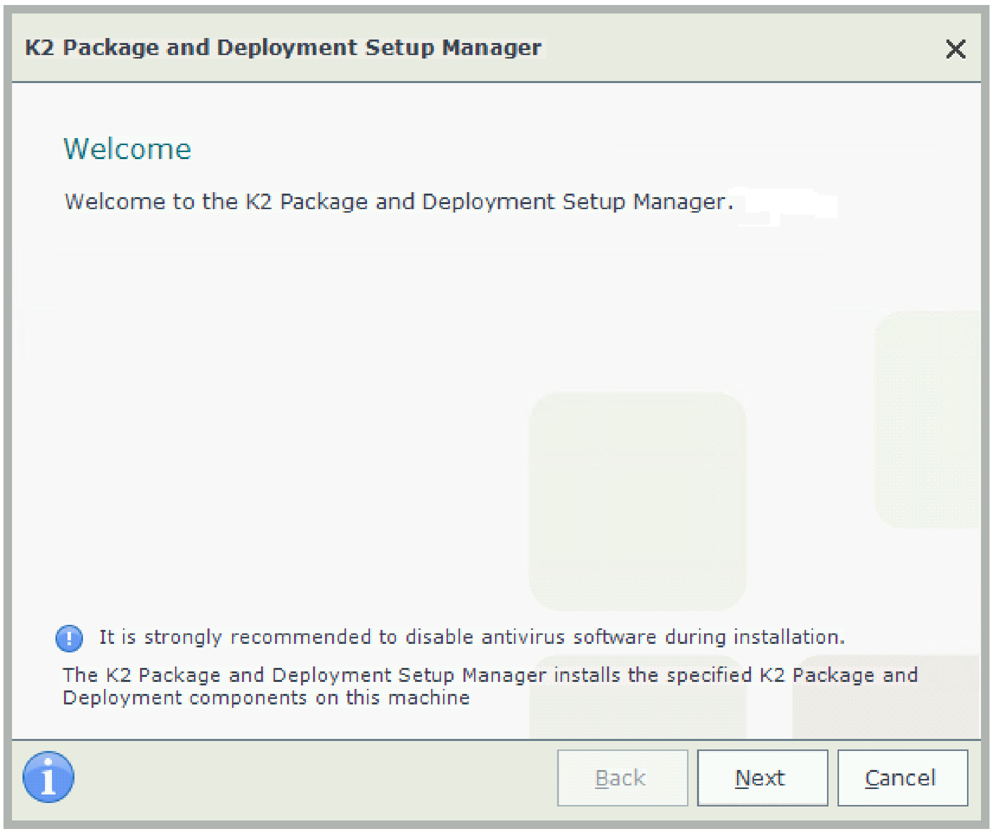
The following artifacts are not supported at this time:

1. Artifacts using Workflow Integration, such as those designed using InfoPath, SharePoint, K2 Studio or K2 Designer for Visual Studio.
2. Artifacts designed using the K2 Designer for SharePoint.
3. Non-K2 artifacts such as SharePoint lists and libraries.

Provide feedback or ask questions

To provide feedback regarding this software, go to the K2 Feedback page (also available at Start > All Programs > K2 blackpearl > Send Feedback), or use the K2 Feedback application (located at Start > All Programs > K2 blackpearl > K2 Package and Deployment > K2 Package and Deployment Feedback). Once the K2 Feedback application is running, use the shortcut keys (Ctrl+Alt+K) to submit feedback.

3. The K2 Package and Deployment Setup Manager Welcome screen will appear. Click **Next** to continue with the installation.



K2 Package and Deployment Setup Manager

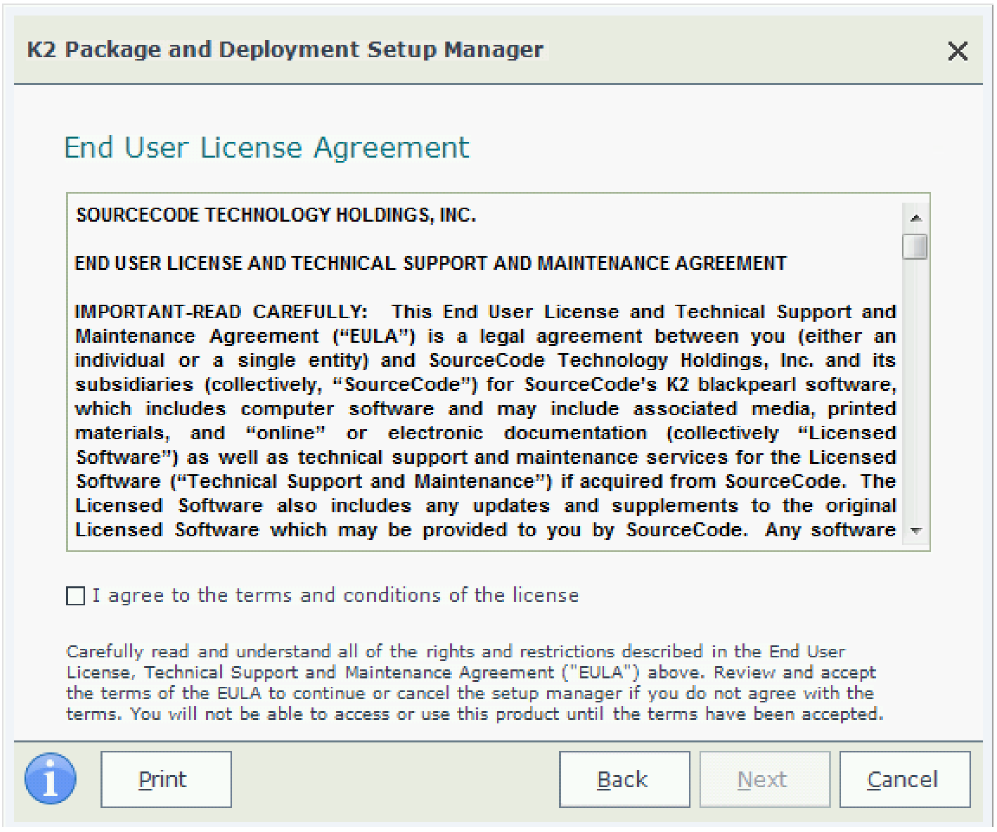
Welcome

Welcome to the K2 Package and Deployment Setup Manager.

It is strongly recommended to disable antivirus software during installation.

The K2 Package and Deployment Setup Manager installs the specified K2 Package and Deployment components on this machine

4. Check the End User License Agreement **I agree to the terms and conditions of the license** check box, then click **Next**.



K2 Package and Deployment Setup Manager

End User License Agreement

SOURCECODE TECHNOLOGY HOLDINGS, INC.

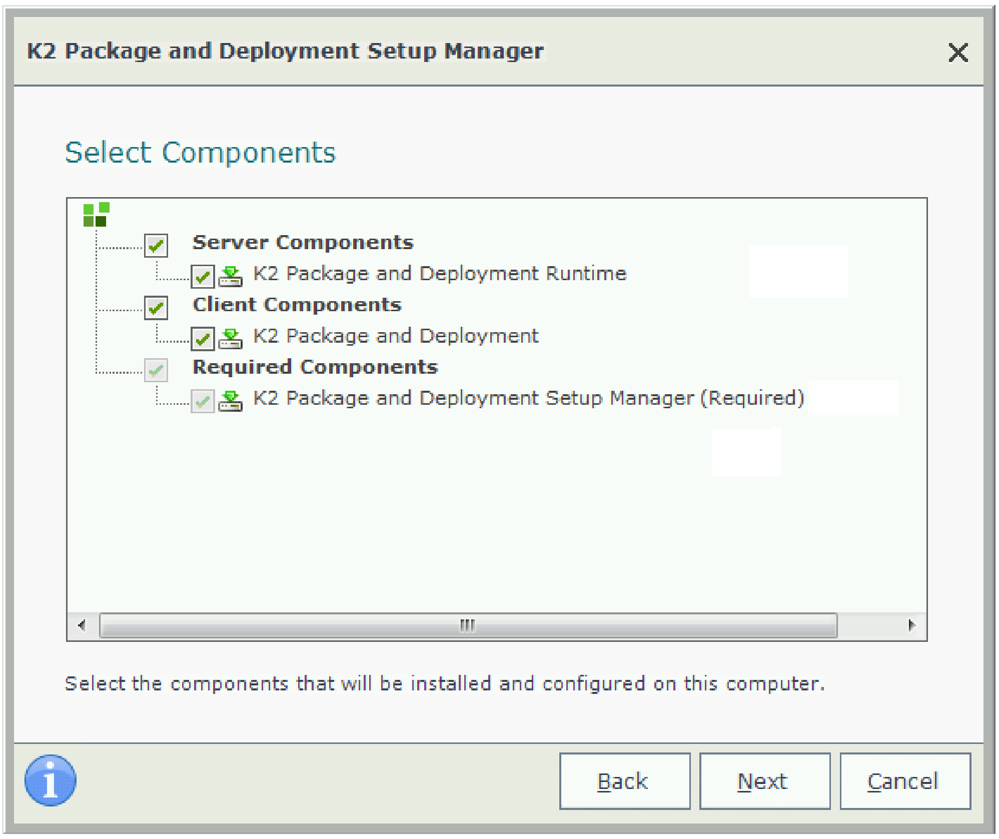
END USER LICENSE AND TECHNICAL SUPPORT AND MAINTENANCE AGREEMENT

IMPORTANT-READ CAREFULLY: This End User License and Technical Support and Maintenance Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and SourceCode Technology Holdings, Inc. and its subsidiaries (collectively, "SourceCode") for SourceCode's K2 blackpearl software, which includes computer software and may include associated media, printed materials, and "online" or electronic documentation (collectively "Licensed Software") as well as technical support and maintenance services for the Licensed Software ("Technical Support and Maintenance") if acquired from SourceCode. The Licensed Software also includes any updates and supplements to the original Licensed Software which may be provided to you by SourceCode. Any software

☐ I agree to the terms and conditions of the license

Carefully read and understand all of the rights and restrictions described in the End User License, Technical Support and Maintenance Agreement ("EULA") above. Review and accept the terms of the EULA to continue or cancel the setup manager if you do not agree with the terms. You will not be able to access or use this product until the terms have been accepted.

5. Select or de-select the specific components to be installed, then click **Next**.

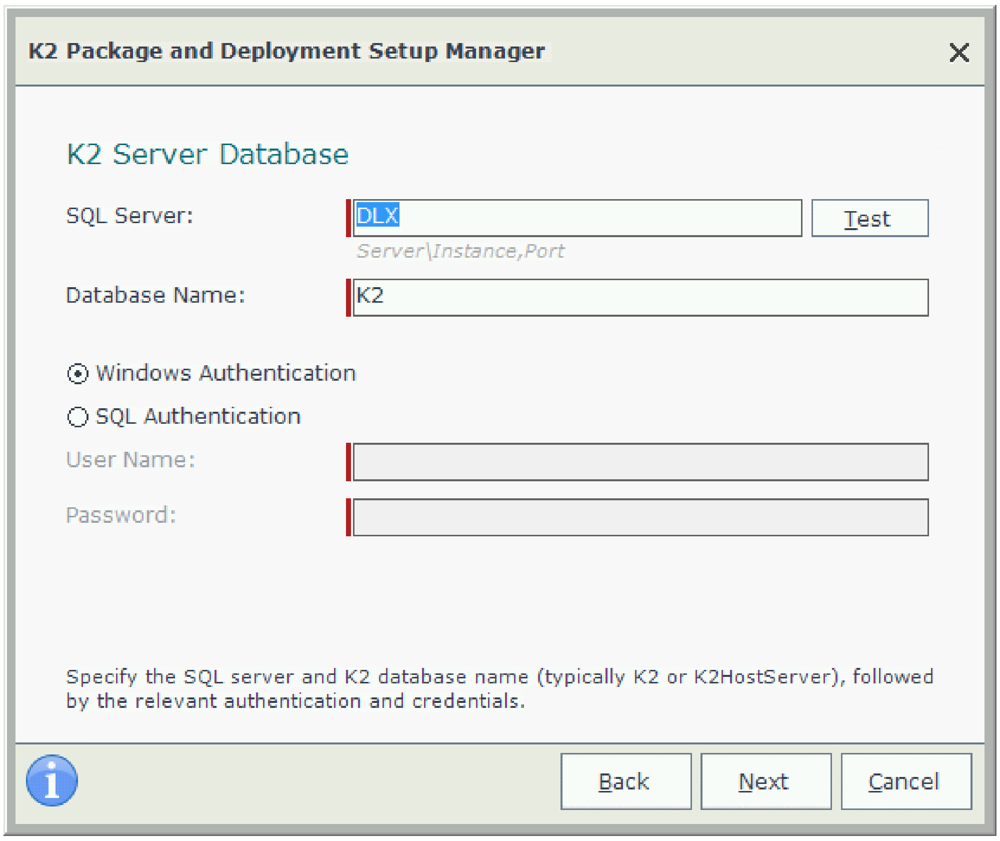

K2 Package and Deployment Setup Manager

Select Components

- [✓] Server Components
  - [✓] K2 Package and Deployment Runtime
- [✓] Client Components
  - [✓] K2 Package and Deployment
- [✓] Required Components
  - [✓] K2 Package and Deployment Setup Manager (Required)

Select the components that will be installed and configured on this computer.

6. Enter the connection to the Microsoft SQL Server which hosts the K2 database, and the K2 database name. If necessary, use the **Test** button to test the connection to the Database Server.
7. Enter the required credentials for Windows or SQL authentication.
8. Click **Next** to continue.


K2 Package and Deployment Setup Manager

K2 Server Database

SQL Server: DLX
Server\Instance,Port

Database Name: K2

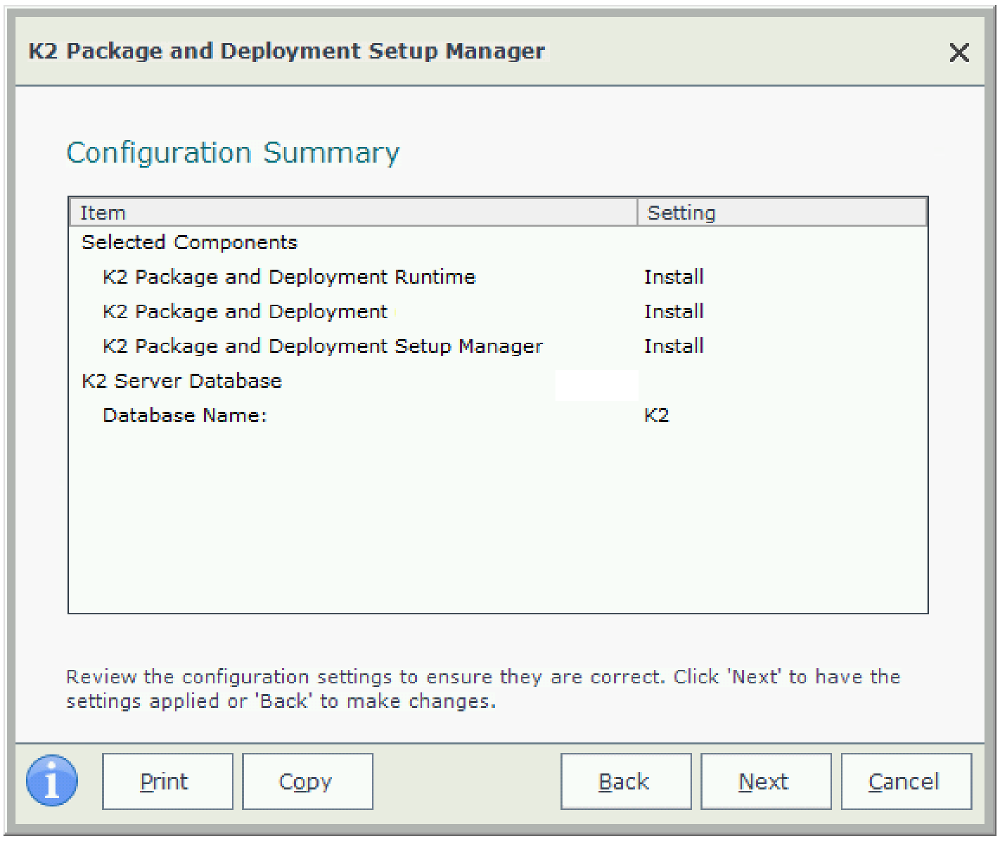
[✓ Windows Authentication] SQL Authentication

User Name:

Password:

Specify the SQL server and K2 database name (typically K2 or K2HostServer), followed by the relevant authentication and credentials.

9. The Configuration Summary screen will now list all items to be installed. Click **Print** if you need a report on what will be installed. Use the **Copy** button to copy the summary to the clipboard (if, for example, you will need to use the summary in another document). Click **Next** to start the installation.



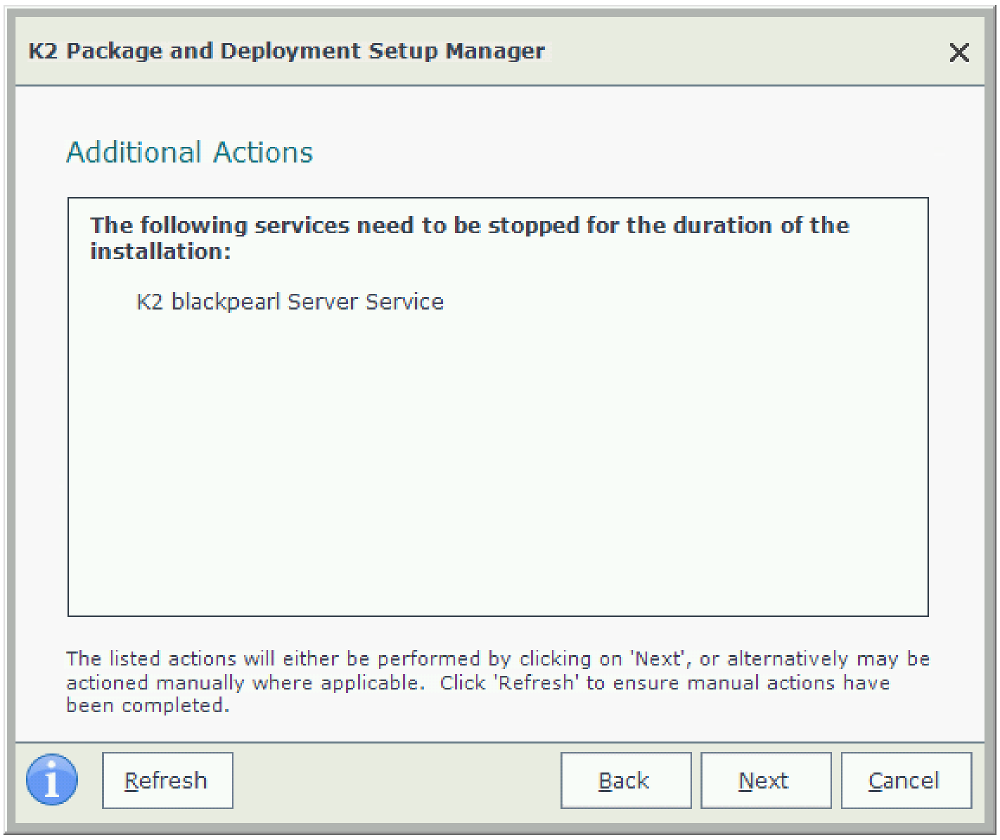
K2 Package and Deployment Setup Manager

Configuration Summary

Item	Setting
Selected Components	
K2 Package and Deployment Runtime	Install
K2 Package and Deployment	Install
K2 Package and Deployment Setup Manager	Install
K2 Server Database	
Database Name:	K2

Review the configuration settings to ensure they are correct. Click 'Next' to have the settings applied or 'Back' to make changes.

10. If there are any additional actions to be performed during the installation process (i.e. restarting the K2 blackpearl Server), the Additional Actions screen will present this information. If any manual actions must be performed, click **Refresh** once they have been completed. Click the **Next** button to proceed with the installation.



K2 Package and Deployment Setup Manager

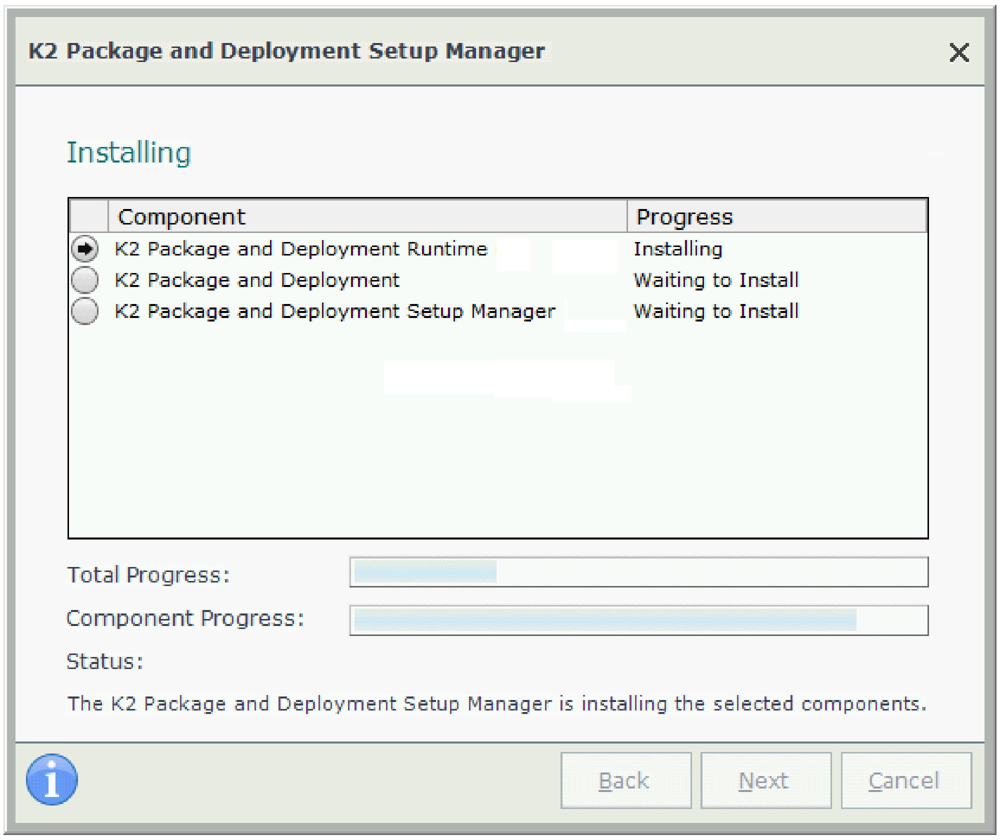
Additional Actions

The following services need to be stopped for the duration of the installation:

K2 blackpearl Server Service

The listed actions will either be performed by clicking on 'Next', or alternatively may be actioned manually where applicable. Click 'Refresh' to ensure manual actions have been completed.

11. The K2 Package and Deployment Setup Manager will now show the progress of the components being installed.


K2 Package and Deployment Setup Manager

Installing

Component	Progress
K2 Package and Deployment Runtime	Installing
K2 Package and Deployment	Waiting to Install
K2 Package and Deployment Setup Manager	Waiting to Install

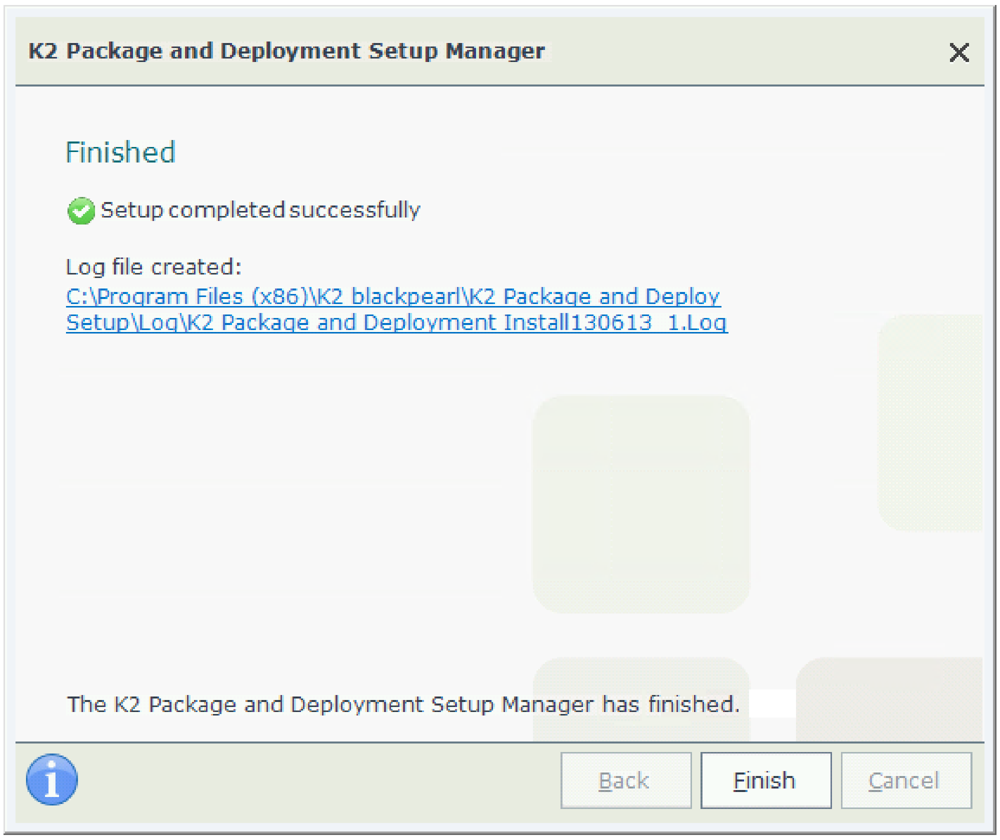
Total Progress:

Component Progress:

Status:

The K2 Package and Deployment Setup Manager is installing the selected components.

12. Finally, once all components are configured, the **Finished** screen will present a summary of the installation and show whether the components were successfully installed and configured. The installation log file can be read by clicking on the link presented.
13. Click **Finish** to end the K2 Package and Deployment installation.


K2 Package and Deployment Setup Manager

Finished

Setup completed successfully

Log file created:
C:\Program Files (x86)\K2 blackpearl\K2 Package and Deploy Setup\Log\K2 Package and Deployment Install130613_1.Log

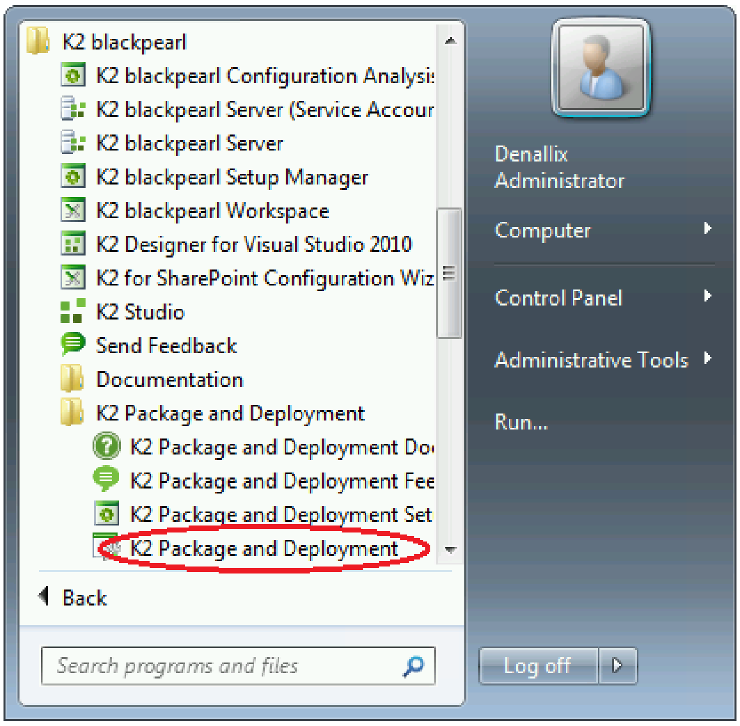
The K2 Package and Deployment Setup Manager has finished.

14. You may be prompted to manually restart the K2 blackpearl service.

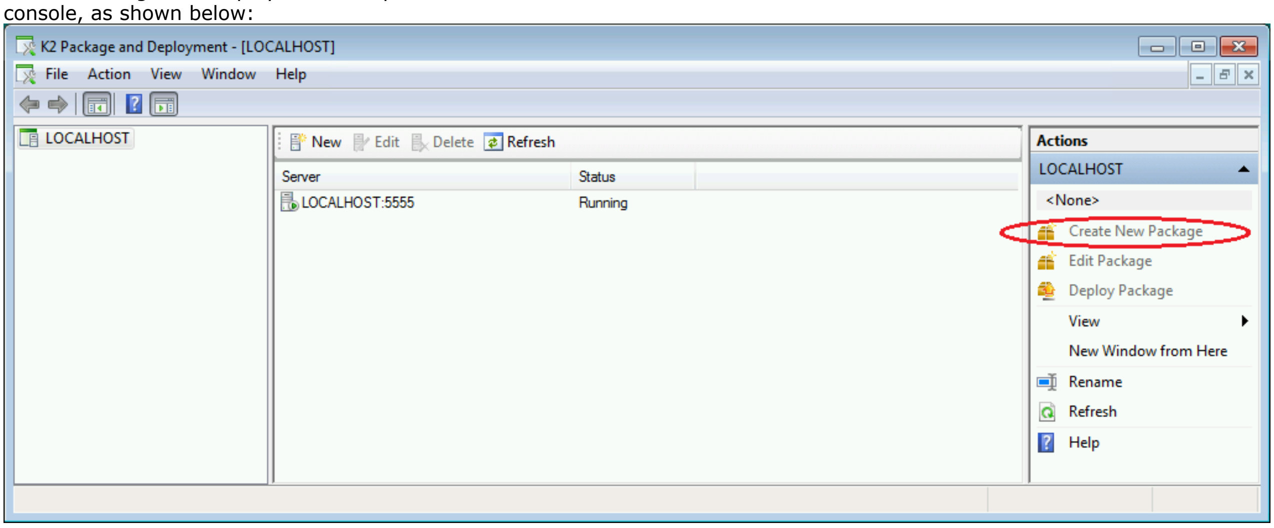
1.3 Using K2 Package and Deployment

Using the K2 Package and Deployment Tool

The K2 Package and Deployment tool can be launched by using the **K2 blackpearl** -> **K2 Package and Deployment** shortcut in the Windows Start Menu, as shown below:

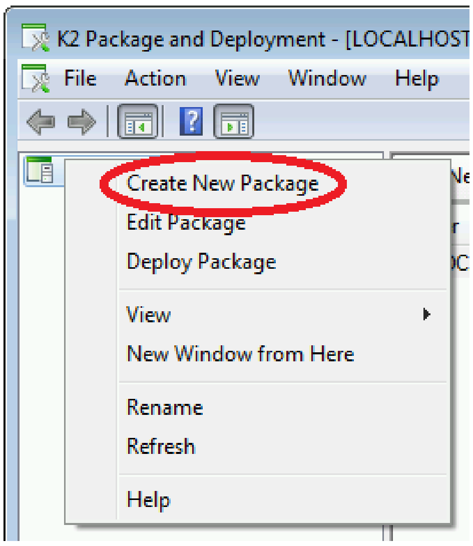

K2 blackpearl
K2 blackpearl Configuration Analysis
K2 blackpearl Server (Service Accour
K2 blackpearl Server
K2 blackpearl Setup Manager
K2 blackpearl Workspace
K2 Designer for Visual Studio 2010
K2 for SharePoint Configuration Wiz
K2 Studio
Send Feedback
Documentation
K2 Package and Deployment
K2 Package and Deployment Do
K2 Package and Deployment Fee
K2 Package and Deployment Set
K2 Package and Deployment
Back
Search programs and files
Denallix Administrator
Computer
Control Panel
Administrative Tools
Run...
Log off

1. The K2 Package and Deployment Tool presents two action menus. The first is located in the **Actions** column of the MMC console, as shown below:



Server	Status
LOCALHOST:5555	Running

2. The second is shown by right-clicking on the K2 Package and Deployment Snap-in node under the MMC Console Root:



Possible Actions

Action	What it does
Create New Package	Starts the Package Creation wizard used to select the K2 artifacts that will be included in a deployment package. See Creating a Package.
Edit Package	Starts the Package Editing wizard used to add or delete artifacts and dependencies within deployment packages. See Editing an Existing Package.
Deploy Package	Starts the Package Deployment wizard used to extract and deploy K2 artifacts from a K2 Package, to a target K2 environment. See Deploying a Package.
View	Allows the administrator to customize the MMC toolbars and layout.
New Window from Here	Opens a new MMC Window.
Rename	Allows the administrator to rename the selected Snap-in.
Refresh	Refreshes the console screen.
Help	Opens Microsoft Management Console Help.

1.3.1 Creating a Package

Creating A Package using K2 Package and Deployment

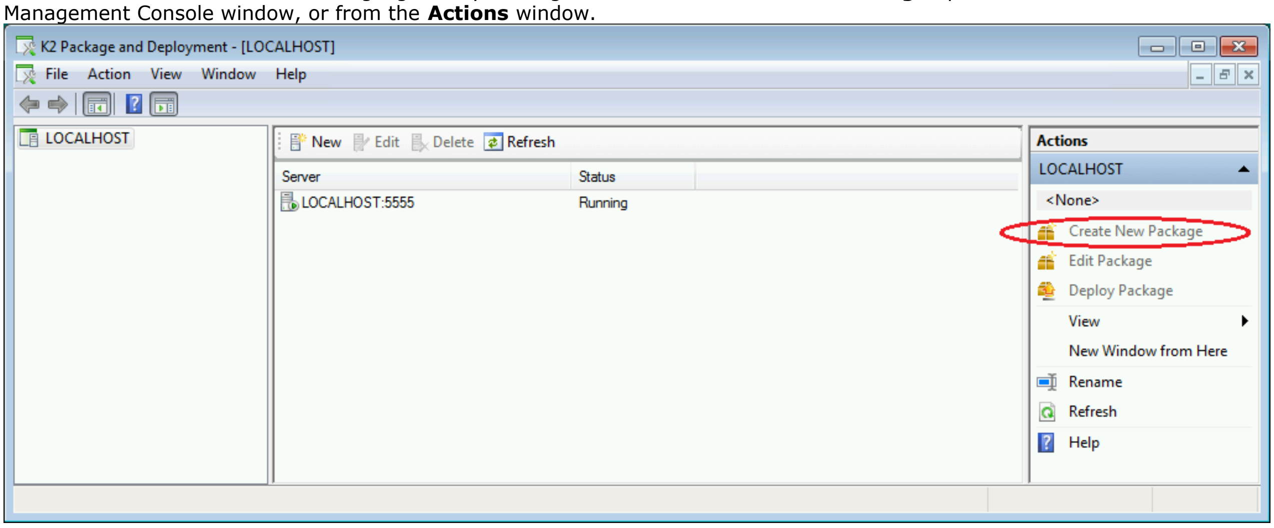
K2 Package and Deployment can be used to create a package file containing all the information needed to re-create those artifacts in a different environment. For example, the tool can be used to create a package of a SmartObject definition, and then deploy that SmartObject definition to another K2 environment.

For more information regarding which types of artifacts are and are not supported see the K2 Package and Deployment Overview topic.

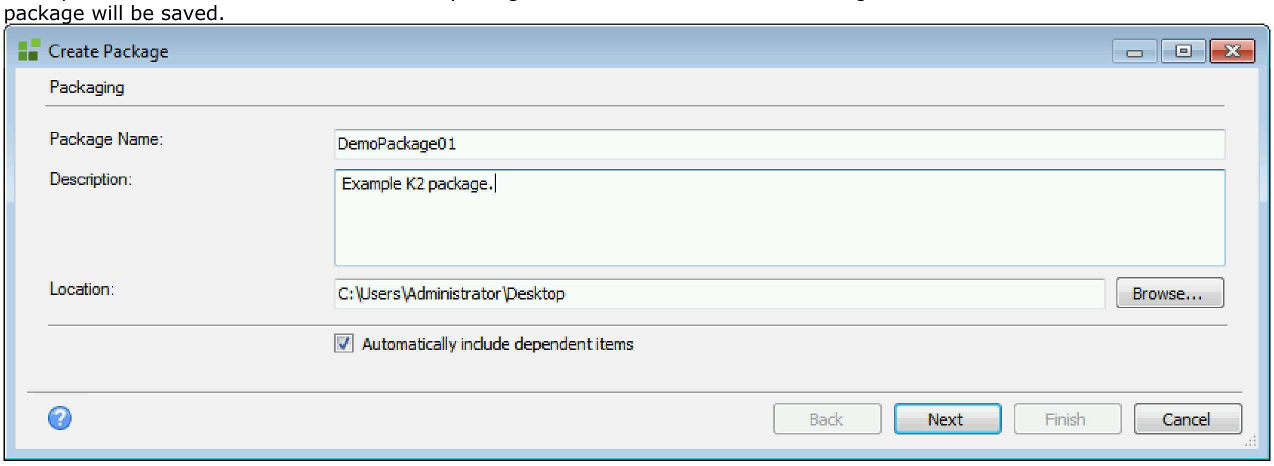
When packaging and deploying Service Instance artifacts, ensure that the associated backend entities exist on the target environment before attempting to deploy the package. See K2 Package and Deployment Overview for more information.

Creating a new K2 Deployment Package

1. Ensure that the relevant Server is highlighted by clicking it. Select the **Create New Package** option from the Microsoft Management Console window, or from the **Actions** window.



2. Once the **Create Package** window has loaded, enter the name of the package in the Package Name text box, plus a description of what will be defined within the package. Use the **Browse** button to navigate to the location to which the package will be saved.



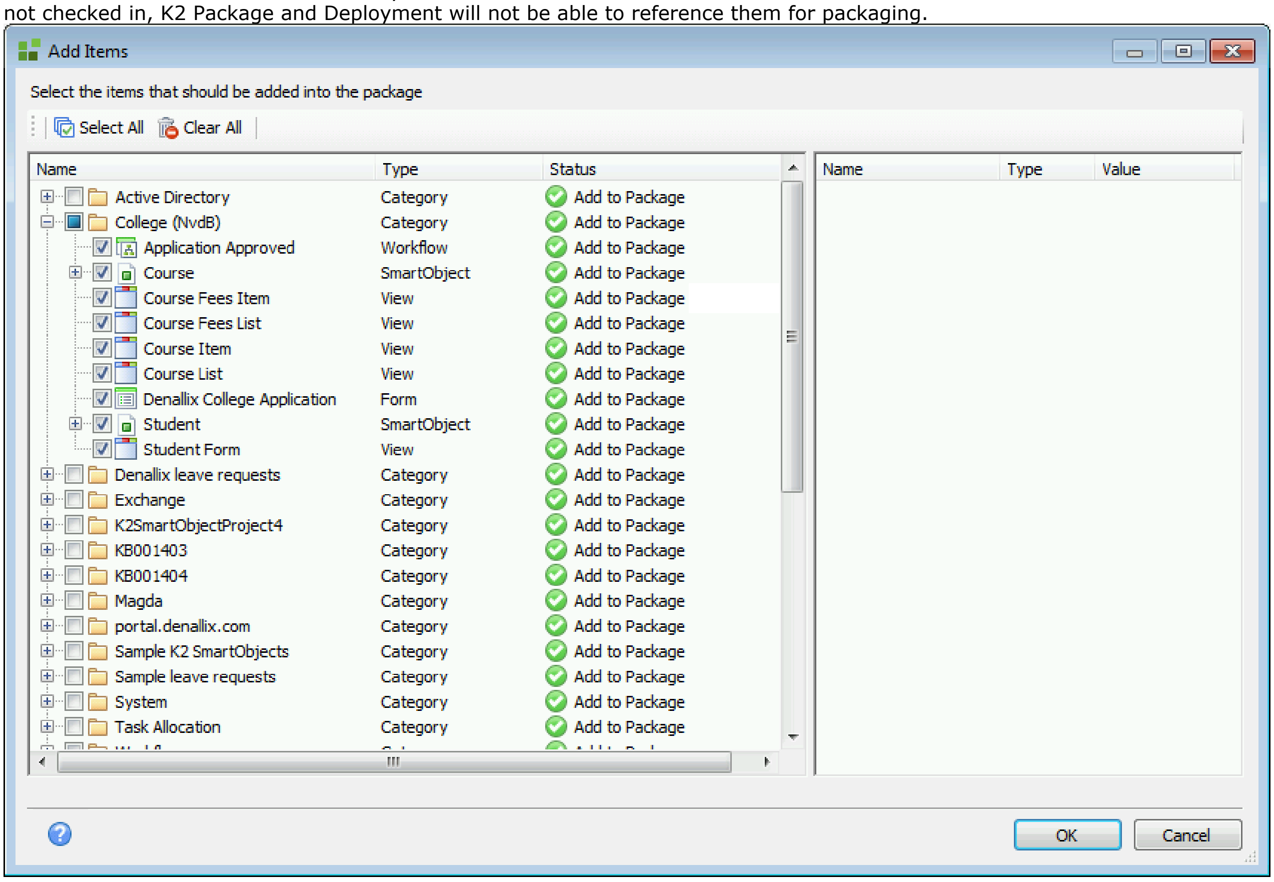
Package Name: DemoPackage01
Description: Example K2 package.
Location: C:\Users\Administrator\Desktop
☑ Automatically include dependent items

The **Automatically include dependent items** check box should only be deselected by administrator-level developers who have advanced knowledge of K2 systems. If this check box is deselected, the K2 Package and Deployment tool will no longer check for artifact dependencies, and the resulting package may be created with dependency issues. Deploying a package with dependency issues may cause system errors. Workflows and SmartObjects have many levels of dependencies, and not including an association or service instance could have serious consequences.

If you deselect the **Automatically include dependent items** check box, you must manually expand and select each of the nodes and sub-nodes of the artifacts you wish to include in the package.

3. Click on **Next** to navigate to the **Add Items** window.

4. The Package and Deployment tool will query the K2 Server and compile a list of K2 artifacts which can be packaged. The status of all artifacts which are currently checked out will be marked with a red exclamation mark. If K2 artifacts are not checked in, K2 Package and Deployment will not be able to reference them for packaging.



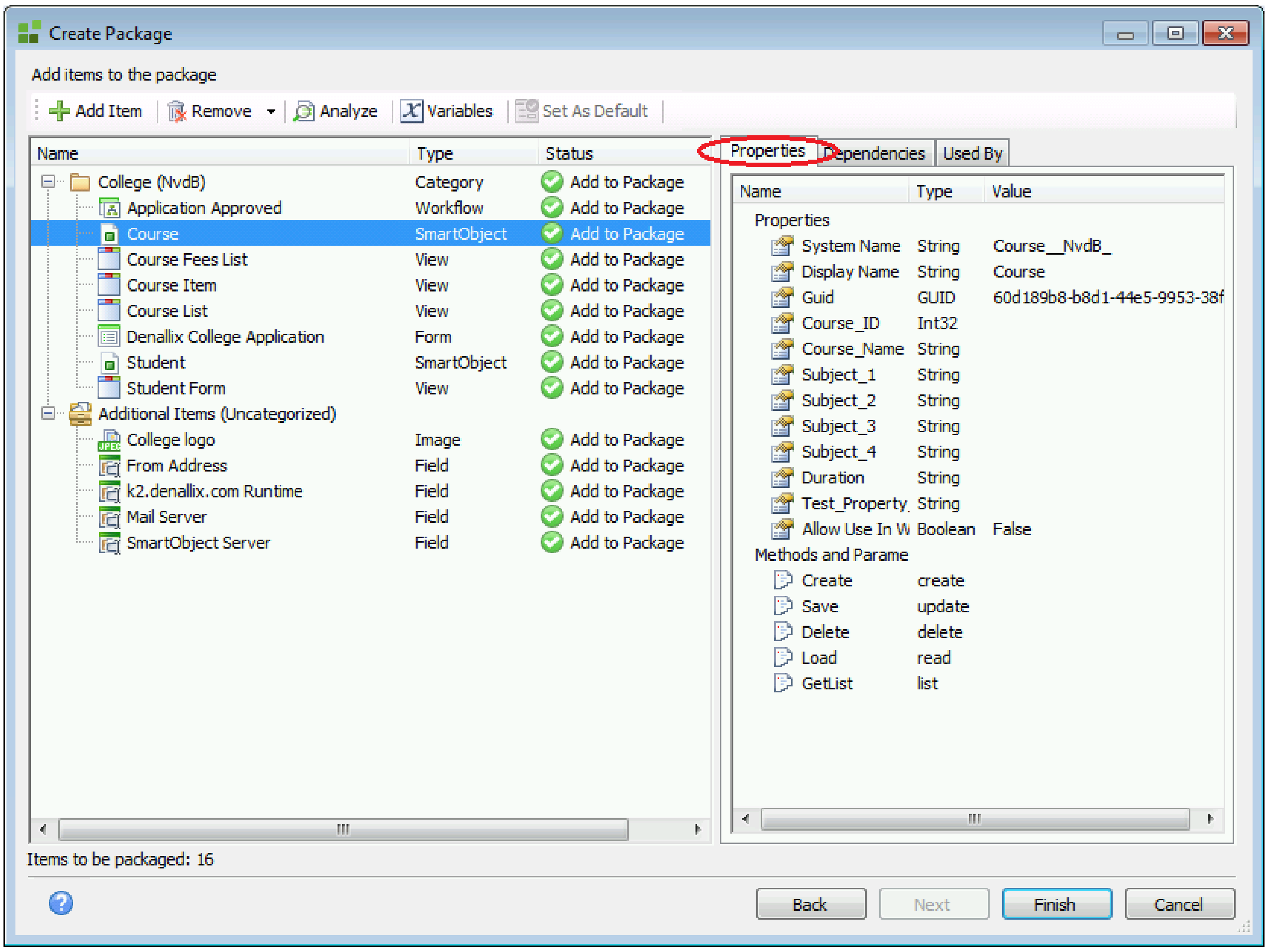
Name	Type	Status
Active Directory	Category	Add to Package
College (NvdB)	Category	Add to Package
Application Approved	Workflow	Add to Package
Course	SmartObject	Add to Package
Course Fees Item	View	Add to Package
Course Fees List	View	Add to Package
Course Item	View	Add to Package
Course List	View	Add to Package
Denallix College Application	Form	Add to Package
Student	SmartObject	Add to Package
Student Form	View	Add to Package
Denallix leave requests	Category	Add to Package
Exchange	Category	Add to Package
K2SmartObjectProject4	Category	Add to Package
KB001403	Category	Add to Package
KB001404	Category	Add to Package
Magda	Category	Add to Package
portal.denallix.com	Category	Add to Package
Sample K2 SmartObjects	Category	Add to Package
Sample leave requests	Category	Add to Package
System	Category	Add to Package
Task Allocation	Category	Add to Package

Name	Type	Value

5. Select the K2 artifact or artifacts you want to package.
   - The **Type** column shows what the K2 artifact is defined as.
   - Clicking on the K2 artifact in the left column will show the properties of the object in the right-hand column.

By default, SmartObjectData is **deselected** so that SmartObject SmartBox data from one environment is not automatically deployed to a new environment. This is done in order to prevent deployment packages from becoming oversized (due to the potentially large amount of data stored within a SmartObject). If you need to package the SmartBox data associated with a particular SmartObject, open the SmartObject node and manually select the required SmartObjectData. For more information, see K2 Package and Deployment Overview.

6. Once you have selected one or more of the K2 artifacts listed, click **OK** to add them to the deployment package list. The **Create Package** screen will now display the selected artifacts. The right-hand window displays the artifact properties, dependencies, and what the artifact is used by. The number of items to be included in the package is displayed in the bottom left-hand corner of the window. In the screenshot below, the object Properties tab has been selected, allowing the user to view the properties of the selected object. The top menu provides options for adding, removing, re-analyzing, and creating or modifying the variables for a package.

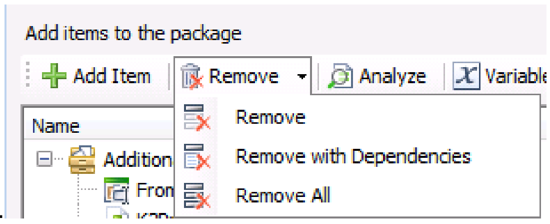


Name	Type	Status
College (NvdB)	Category	Add to Package
Application Approved	Workflow	Add to Package
Course	SmartObject	Add to Package
Course Fees List	View	Add to Package
Course Item	View	Add to Package
Course List	View	Add to Package
Denallix College Application	Form	Add to Package
Student	SmartObject	Add to Package
Student Form	View	Add to Package
Additional Items (Uncategorized)		
College logo	Image	Add to Package
From Address	Field	Add to Package
k2.denallix.com Runtime	Field	Add to Package
Mail Server	Field	Add to Package
SmartObject Server	Field	Add to Package

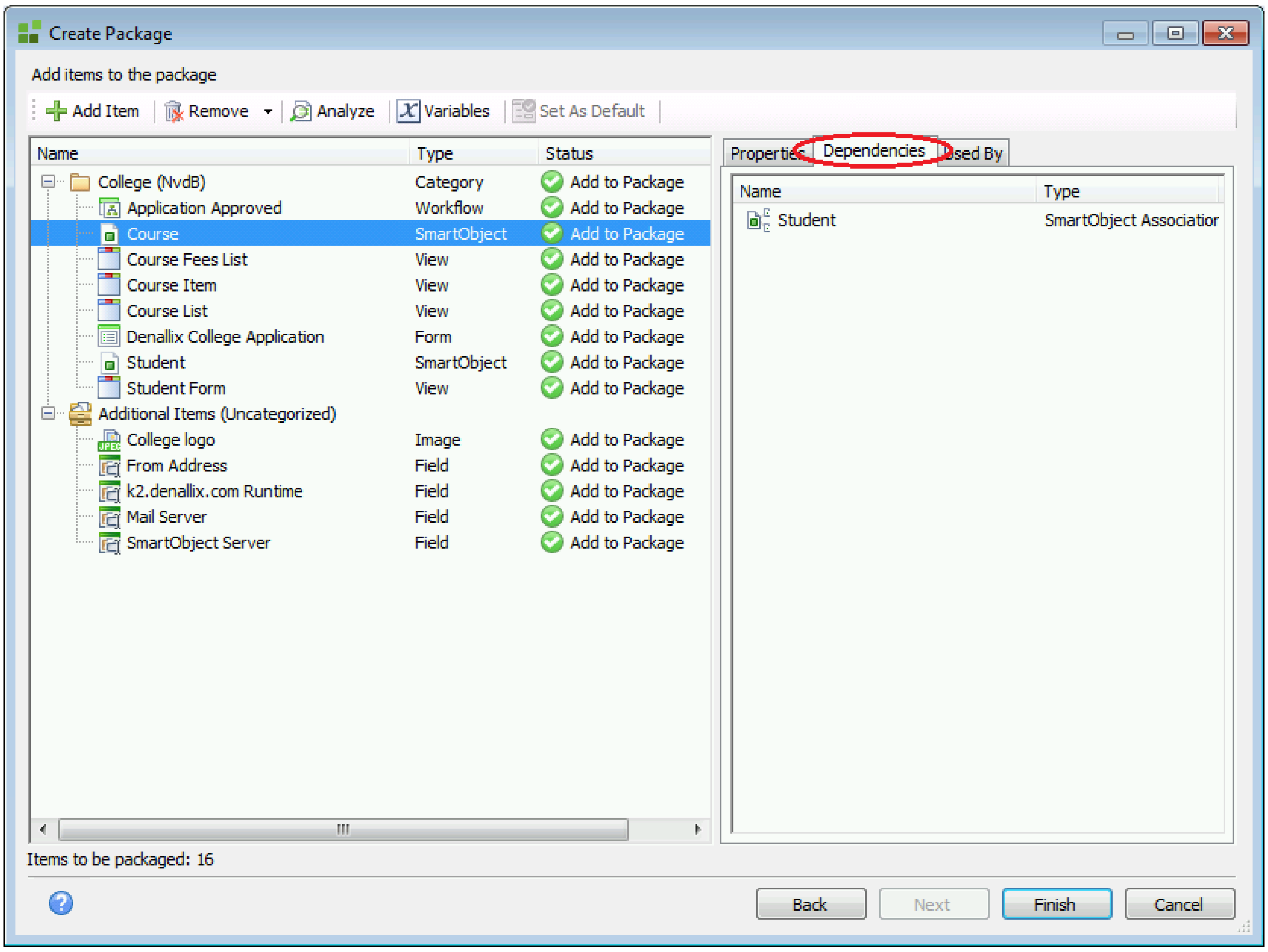
Name	Type	Value
Properties		
System Name	String	Course__NvdB_
Display Name	String	Course
Guid	GUID	60d189b8-b8d1-44e5-9953-38f
Course_ID	Int32	
Course_Name	String	
Subject_1	String	
Subject_2	String	
Subject_3	String	
Subject_4	String	
Duration	String	
Test_Property	String	
Allow Use In W	Boolean	False
Methods and Parame		
Create	create	
Save	update	
Delete	delete	
Load	read	
GetList	list	

Items to be packaged: 16

7. The **Remove** option provides three sub-options; **Remove**, **Remove with Dependencies**, and **Remove All**. These provide the package creator with specialized artifact removal options. Artifacts that have been added to the package must be selected before using these sub-options.



8. In the screenshot below, the object Dependencies tab has been selected, allowing the user to view the dependencies for the selected object.

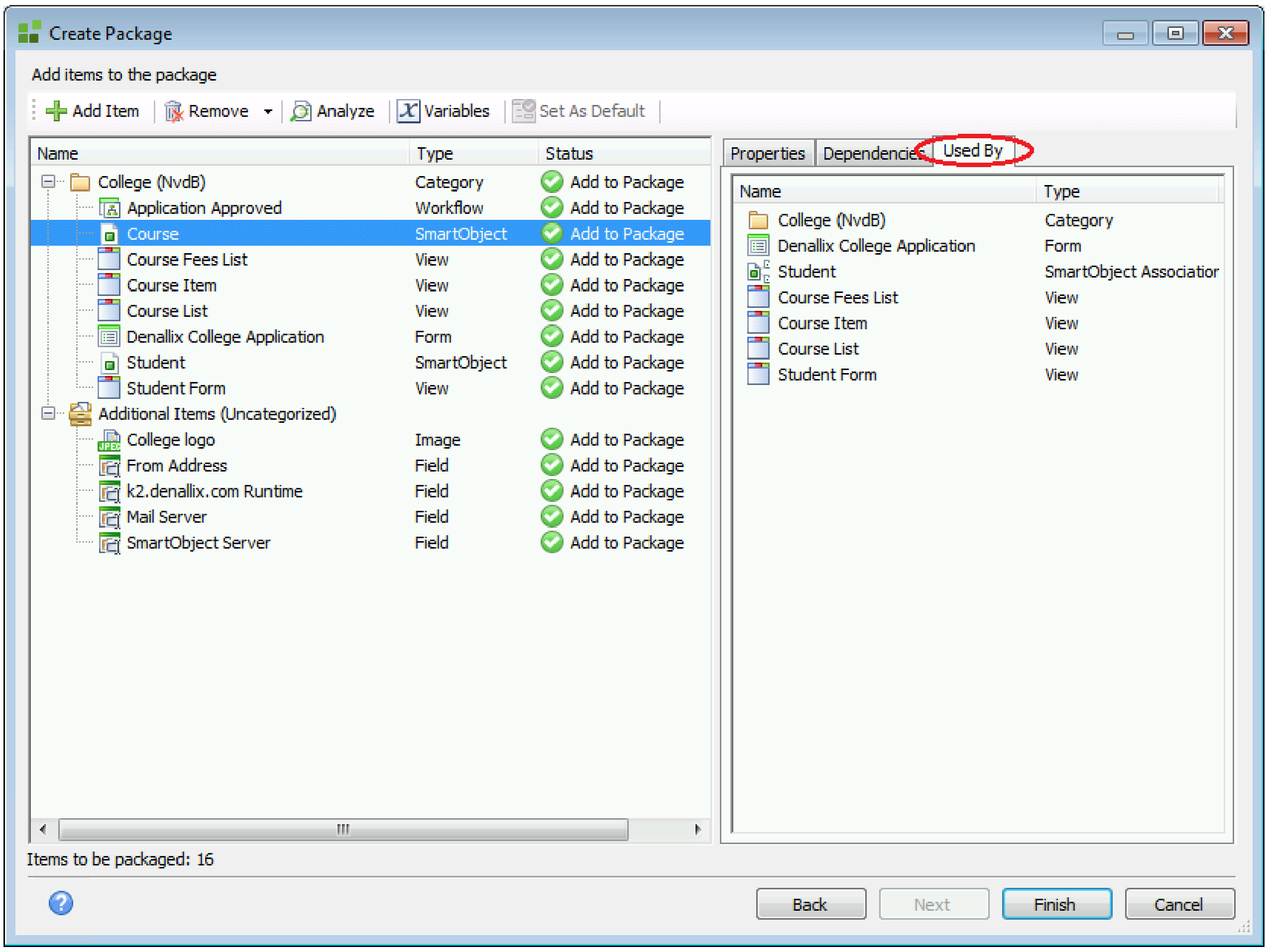


Name	Type	Status
College (NvdB)	Category	Add to Package
Application Approved	Workflow	Add to Package
Course	SmartObject	Add to Package
Course Fees List	View	Add to Package
Course Item	View	Add to Package
Course List	View	Add to Package
Denallix College Application	Form	Add to Package
Student	SmartObject	Add to Package
Student Form	View	Add to Package
Additional Items (Uncategorized)		
College logo	Image	Add to Package
From Address	Field	Add to Package
k2.denallix.com Runtime	Field	Add to Package
Mail Server	Field	Add to Package
SmartObject Server	Field	Add to Package

Name	Type
Student	SmartObject Associatior

Items to be packaged: 16

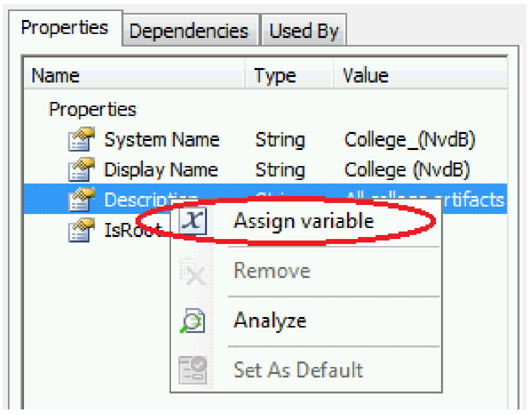
9. In the screenshot below, the Used By tab has been selected, allowing the user to view other objects the selected object is used by.



Name	Type	Status
College (NvdB)	Category	Add to Package
Application Approved	Workflow	Add to Package
Course	SmartObject	Add to Package
Course Fees List	View	Add to Package
Course Item	View	Add to Package
Course List	View	Add to Package
Denallix College Application	Form	Add to Package
Student	SmartObject	Add to Package
Student Form	View	Add to Package
Additional Items (Uncategorized)		
College logo	Image	Add to Package
From Address	Field	Add to Package
k2.denallix.com Runtime	Field	Add to Package
Mail Server	Field	Add to Package
SmartObject Server	Field	Add to Package

Name	Type
College (NvdB)	Category
Denallix College Application	Form
Student	SmartObject Associatior
Course Fees List	View
Course Item	View
Course List	View
Student Form	View

10. At this point, you can create assignable variables for specific properties of any deployment package objects which require different values within a new environment (for example, service instance credentials). See the Assigning Variables topic for more information.



Name	Type	Value
Properties		
System Name	String	College_(NvdB)
Display Name	String	College (NvdB)
Description	String	All college artifacts
IsRoot		

11. Once any needed variables have been assigned, click **Finish** to create the package and close the Create Package dialogue.

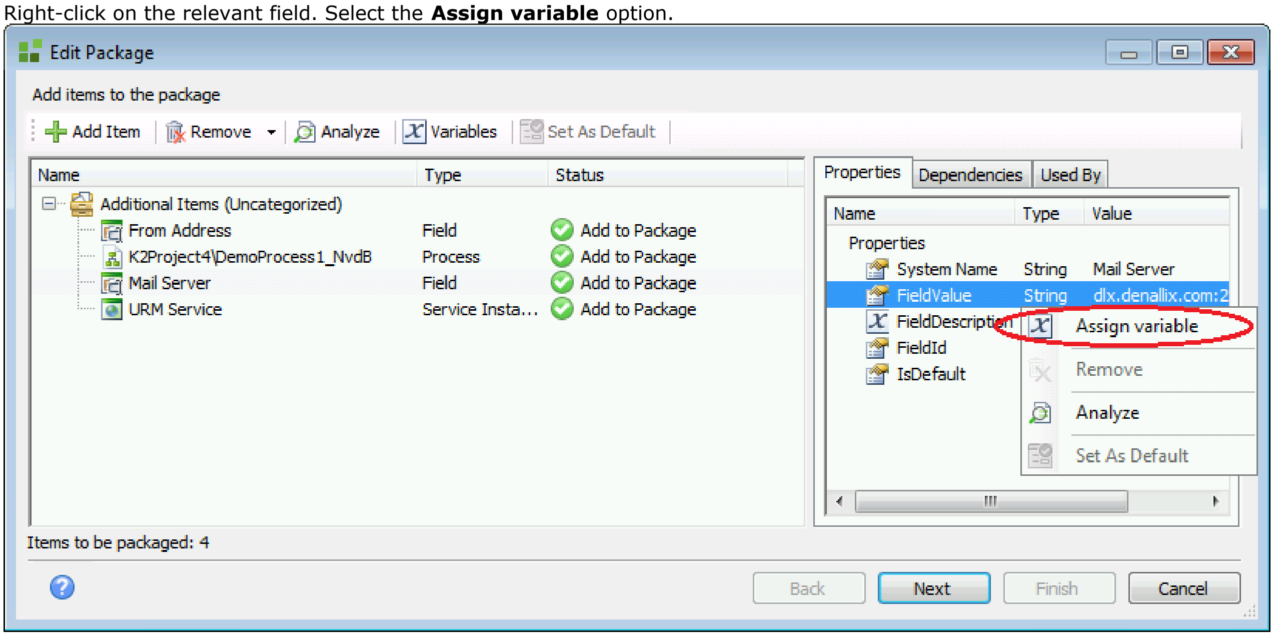
1.3.1.1 Assigning Variables

If the properties of packaged artifacts will be different in a target environment (for example, service instance credentials), create variables for those properties.

If the environment information is not known at the time of creating the package, use the conflict resolution dialogue during deployment (for more information, see Conflict Resolution.)

Assigning Deployment Package Variables

1. Right-click on the relevant field. Select the **Assign variable** option.



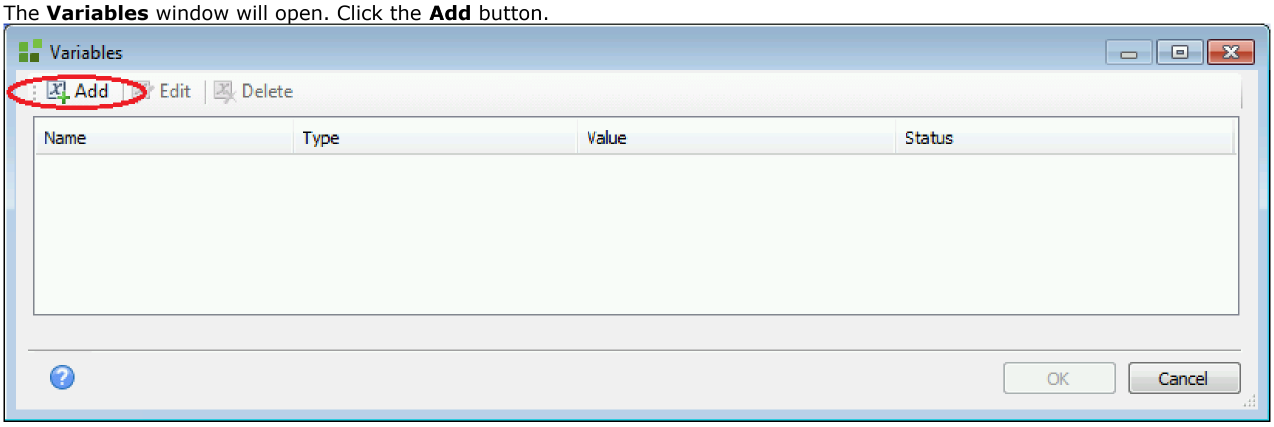
Name	Type	Status
Additional Items (Uncategorized)		
From Address	Field	Add to Package
K2Project4\DemoProcess1_NvdB	Process	Add to Package
Mail Server	Field	Add to Package
URM Service	Service Insta...	Add to Package

Name	Type	Value
Properties		
System Name	String	Mail Server
FieldValue	String	dlx.denallix.com:2
FieldDescription		
FieldId		
IsDefault		

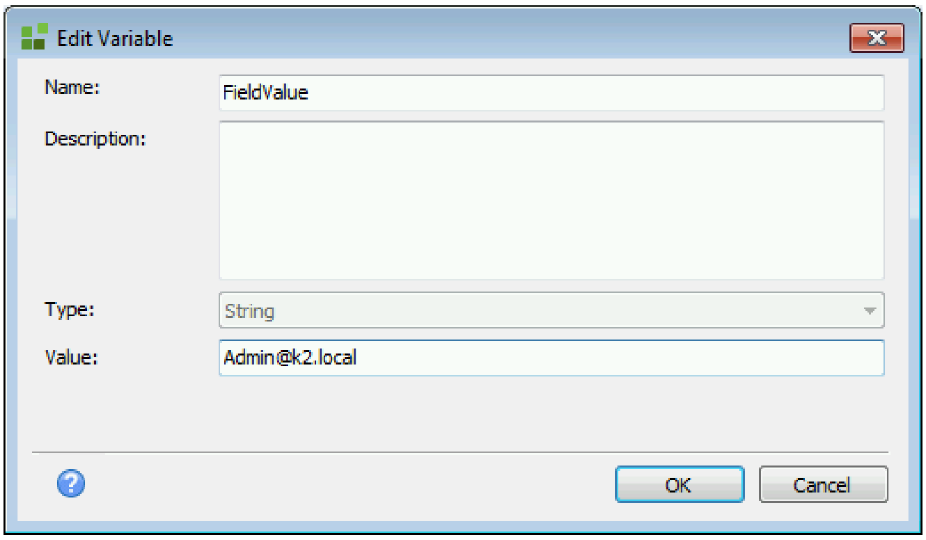
Items to be packaged: 4

If the **Assign variable** option does not appear, the field to which you are attempting to assign a variable is not a property to which a variable can be assigned.

2. The **Variables** window will open. Click the **Add** button.

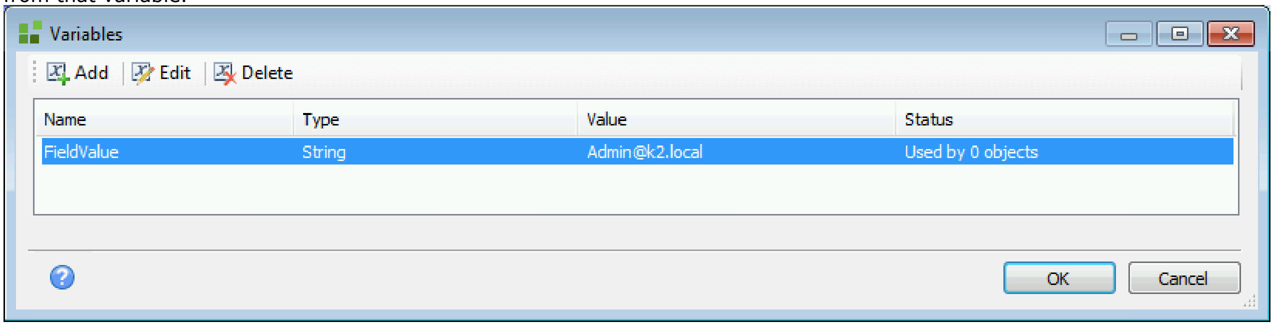


3. If you have already created a variable for this field during this MMC session, it will be listed here. You may choose to select it from the list instead of adding a new one.
4. In the **Edit Variable** window, specify a **Name**, **Description**, and new **Value** for the variable.


Edit Variable

Name:	FieldValue
Description:	
Type:	String
Value:	Admin@k2.local

5. Click **OK** to assign this variable to the selected field.
6. Once a variable has been assigned, the **Status** column will show how many deployable fields have been assigned a new value from that variable.


Variables

Name	Type	Value	Status
FieldValue	String	Admin@k2.local	Used by 0 objects

1.3.2 Editing an Existing Package

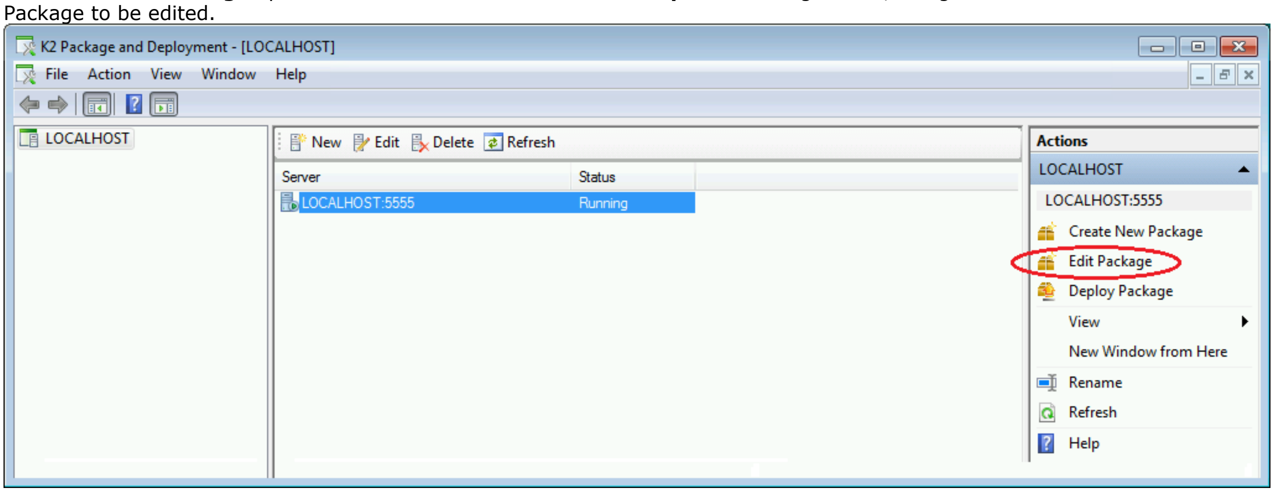
Editing A Package using K2 Package and Deployment

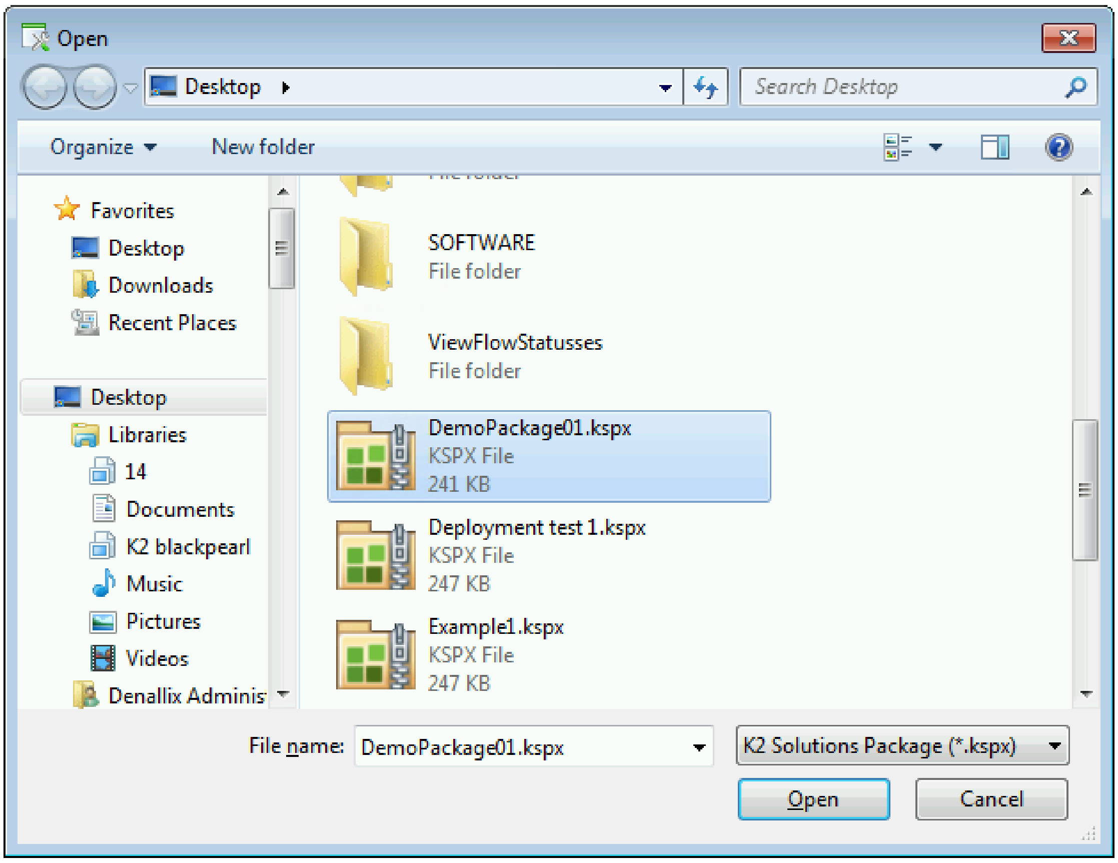
K2 Package and Deployment can be used to edit the contents of an existing package file by adding and removing artifacts. The tool can then be used to deploy the edited package definition to another K2 Environment. This is useful for altering packaged variables and for adding new artifacts to a package.

When packaging and deploying Service Instance artifacts, ensure that the associated backend entities exist on the target environment before attempting to deploy the package. See K2 Package and Deployment Overview for more information.

Editing an existing K2 Deployment Package

1. Select the **Edit Package** option from the MMC window. From the **Open** file dialog screen, navigate to and select the K2 Package to be edited.





2. Click **Open** to open the K2 Package.
3. Refer to the Creating a Package topic for further information on adding or changing artifacts in the package.

1.3.3 Deploying a Package

Deploying A Package

Once the package (.kspx file) has been created, you can use the Deploy Package function to deploy the package to the target K2 environment.

It is strongly recommended that the K2 Package and Deployment Tool be used only by K2 Administrators. It is also recommended that a backup of the K2 database be performed before proceeding with deployment.

K2 Package and Deployment uses the Secure Hash Algorithm 1 (SHA-1) protocol to determine whether artifacts on the target server are an exact match for artifacts in the package. When matches are found, the packaged artifact is not deployed. If an exact match is not found, the artifact from the package is deployed as a new version, or overwrites the artifact on the target server (depending on whether the artifact is versioned).

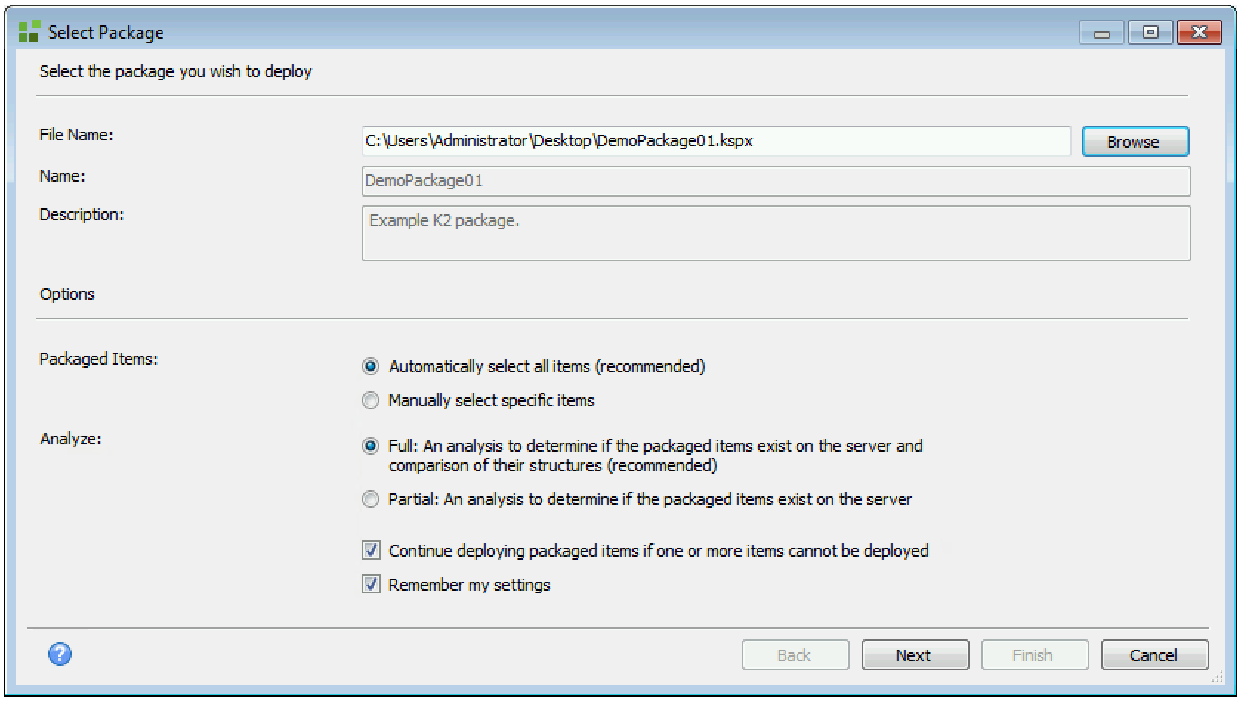
Note that packages which have been created using Beta versions of K2 Package and Deployment are not compatible with the release version of Package and Deployment. Such packages will have to be re-created.

When packaging and deploying Service Instance artifacts, ensure that the associated back-end entities exist on the target environment before attempting to deploy the package. See K2 Package and Deployment Overview for more information. For example, the following artifacts must be present on the target server before the package is deployed:

- K2 smartforms Custom Controls
- Custom Service Types
- Custom Inline Functions

To Deploy a K2 Deployment Package

1. Select **Deploy Package** from the Microsoft Management Console (MMC) window.
2. Use the **Browse** button to locate and load the K2 Deployment Package into the **Select Package** dialog screen. (Note that the file path in the screen shot is for demonstration purposes only.)


Select Package

Select the package you wish to deploy

File Name: C:\Users\Administrator\Desktop\DemoPackage01.kspx

Name: DemoPackage01

Description: Example K2 package.

Options

Packaged Items:
[✓ Automatically select all items (recommended)]
Manually select specific items

Analyze:
[✓ Full: An analysis to determine if the packaged items exist on the server and comparison of their structures (recommended)]
Partial: An analysis to determine if the packaged items exist on the server

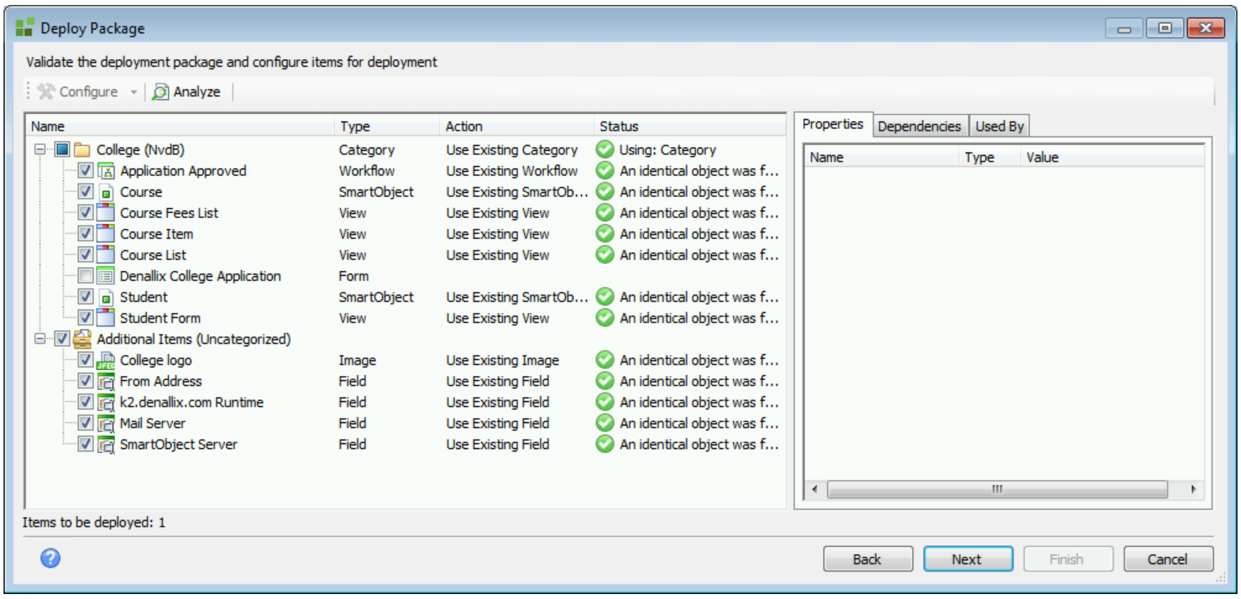
[✓ Continue deploying packaged items if one or more items cannot be deployed]
[✓ Remember my settings]

3. **Packaged Items** radio buttons:
   - Select the **Automatically select all items** option if all items in the current deployment package should be deployed. When **Next** is clicked in the **Select Package** window, all items within the deployment package will be selected by default.
   - Select the **Manually select specific items** option if only some items within the current package should be deployed. After clicking **Next,** individually select the artifacts to be packaged.
4. **Analyze** radio buttons:
   The analysis function compares the artifacts on the server (if they exist) with the artifacts in the K2 package. If any items to be deployed contain content which differs from existing items, the analysis function offers the option to replace the existing artifacts and dependencies with current versions.
   - Select the **Full** option to analyze all artifacts, as well as all dependencies comprising those artifacts.
   - Select the **Partial** option to analyze all artifacts without analyzing the dependencies comprising the artifacts.

5. The **Continue deploying packaged items if one or more items cannot be deployed** check box is checked by default.

- Deselecting the check box will halt the deployment entirely if any artifacts cannot be found, or are incompatible with the current deployment.
- Selecting the check box will halt the deployment of individual artifacts if the relevant artifacts cannot be found or are incompatible with the current deployment. Package and Deployment will continue to deploy all artifacts which are not in conflict with the target environment.

1. The **Remember my settings** check box is checked by default. This allows K2 Package and Deployment to remember which package-selection options have been specified. Should different package-selection options be needed for future deployments, deselect the check box.
2. Click **Next**. The **Deploy Package** window will open. If variables must be assigned at this point, see Assigning Variables for more details.



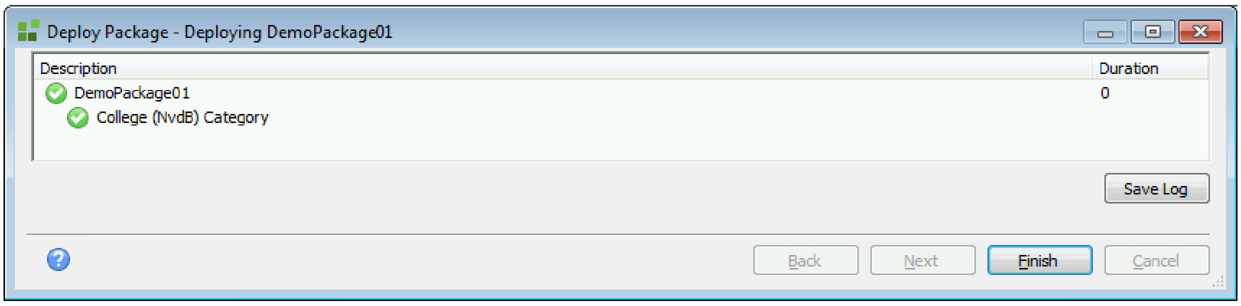
Validate the deployment package and configure items for deployment

Name	Type	Action	Status
College (NvdB)	Category	Use Existing Category	Using: Category
Application Approved	Workflow	Use Existing Workflow	An identical object was f...
Course	SmartObject	Use Existing SmartOb...	An identical object was f...
Course Fees List	View	Use Existing View	An identical object was f...
Course Item	View	Use Existing View	An identical object was f...
Course List	View	Use Existing View	An identical object was f...
Denallix College Application	Form		
Student	SmartObject	Use Existing SmartOb...	An identical object was f...
Student Form	View	Use Existing View	An identical object was f...
Additional Items (Uncategorized)			
College logo	Image	Use Existing Image	An identical object was f...
From Address	Field	Use Existing Field	An identical object was f...
k2.denallix.com Runtime	Field	Use Existing Field	An identical object was f...
Mail Server	Field	Use Existing Field	An identical object was f...
SmartObject Server	Field	Use Existing Field	An identical object was f...

Name	Type	Value

Items to be deployed: 1

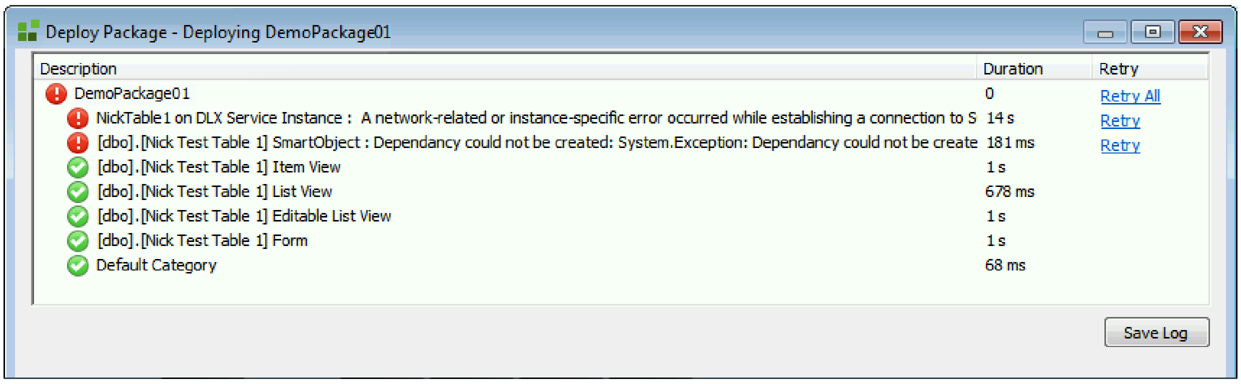
3. Check that all needed artifacts and dependencies have been checked for inclusion in the deployment package. The total number of items to be deployed appears in the lower left corner of the window.
4. All conflicts must be resolved before the package can be deployed to the new environment. If a conflict cannot be resolved, the artifact or dependency causing the conflict must be de-selected. See Conflict Resolution for more details. Once all conflicts have been resolved, run the **Analyze All** option to update the tool's internal status check.
5. Click **Next**. K2 Package and Deployment will interrogate both the K2 artifact definitions and the K2 environment to be deployed to.
6. The **Deploy Package – Deploying** window will appear. This is a list of the packaged items and their readiness status. The time taken to deploy the package's artifacts and dependencies will be shown in the right-hand column once deployment is complete. Where a status icon to the left side of a deployed item shows a green check mark, the item has been successfully deployed.



Description	Duration
DemoPackage01	0
College (NvdB) Category	

As deployment times may vary depending on the size of the package, we recommend that large packages only be deplyed during after hours.

7. If an artifact fails to deploy due to circumstances beyond K2 Package and Deployment's control, a red exclamation mark will be displayed next the artifact that fails to deploy.

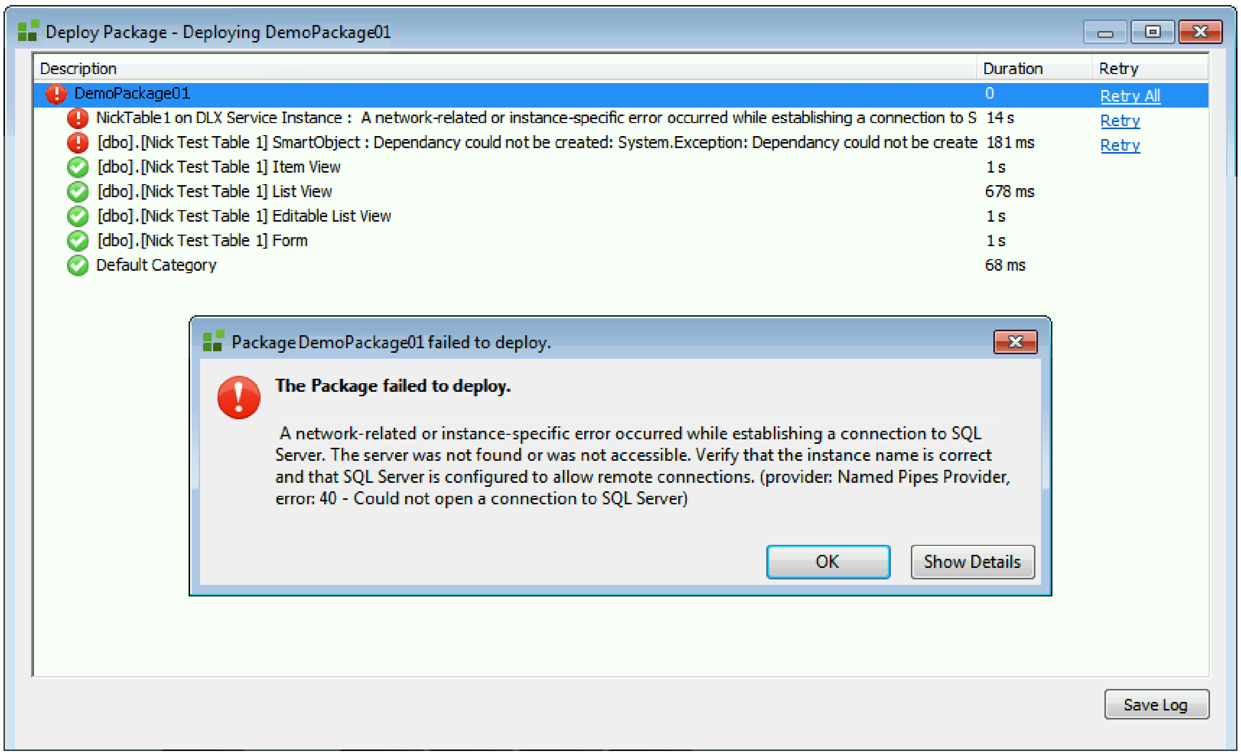


Deploy Package - Deploying DemoPackage01

Description	Duration	Retry
DemoPackage01	0	Retry All
NickTable1 on DLX Service Instance : A network-related or instance-specific error occurred while establishing a connection to S	14 s	Retry
[dbo].[Nick Test Table 1] SmartObject : Dependancy could not be created: System.Exception: Dependancy could not be create	181 ms	Retry
[dbo].[Nick Test Table 1] Item View	1 s	
[dbo].[Nick Test Table 1] List View	678 ms	
[dbo].[Nick Test Table 1] Editable List View	1 s	
[dbo].[Nick Test Table 1] Form	1 s	
Default Category	68 ms	

Save Log

8. If such a deployment failure occurs, double-click on the artifact responsible to view an explanation for the failure.



Deploy Package - Deploying DemoPackage01

Description	Duration	Retry
DemoPackage01	0	Retry All
NickTable1 on DLX Service Instance : A network-related or instance-specific error occurred while establishing a connection to S	14 s	Retry
[dbo].[Nick Test Table 1] SmartObject : Dependancy could not be created: System.Exception: Dependancy could not be create	181 ms	Retry
[dbo].[Nick Test Table 1] Item View	1 s	
[dbo].[Nick Test Table 1] List View	678 ms	
[dbo].[Nick Test Table 1] Editable List View	1 s	
[dbo].[Nick Test Table 1] Form	1 s	
Default Category	68 ms	

Package DemoPackage01 failed to deploy.

The Package failed to deploy.

A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct and that SQL Server is configured to allow remote connections. (provider: Named Pipes Provider, error: 40 - Could not open a connection to SQL Server)

9. When the reason for the deployment failure has been rectified, click on the **Retry** or **Retry All** link next to the relevant artifact or dependency to re-attempt deployment.

   If the K2 Server has been rebooted since the deployment of the package, using the Retry option will not work.

10. If a deployment log needs to be created and saved, click the **Save Log** button. The **Save As** window will open, allowing you to re-name and save the log file.
11. Close the **Deploy Package – Deploying** window. This will conclude the deployment process.

1.3.3.1 Conflict Resolution

Deployment Package Conflict Resolution

K2 Package and Deployment enables the K2 Administrator to edit each property in the deployment package that is in conflict with the target environment.

Typically, there are three types of conflicts:

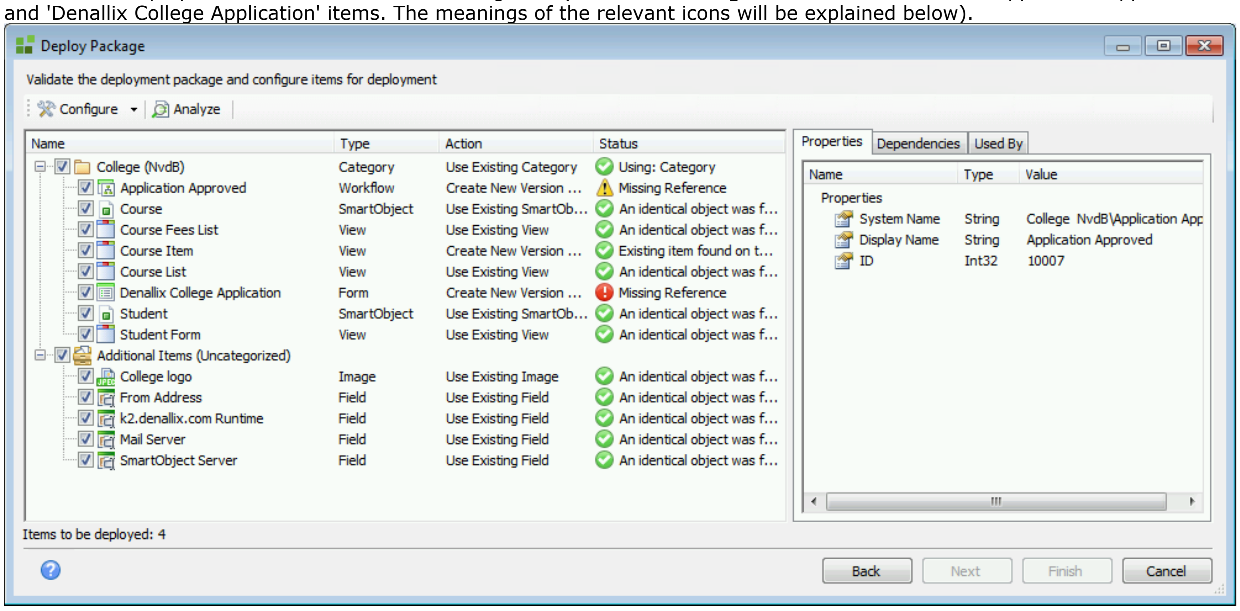
1. A **name** conflict exists when a source item (an item to be deployed) has an identifier which is different to the identifier used by a target item (an item to be replaced). This would normally occur when the SmartObject system name is the same, but the Unique Identifier is different.
2. A **version** conflict exists when an item is present at both the source and target, but will break compatibility with existing items.
3. A **missing reference** conflict exists when artifacts in the package are dependent on objects that cannot be located in the target environment.

Typically, there are three actions that can be taken if an item is in conflict:

1. Overwrite, or deploy a new version of the artifact (if the artifact supports versioning).
2. Change the system name or display name of the artifact. Note that this option is not available for some artifacts.
3. Exclude the artifact from the deployment package.

To Resolve a Package Conflict

1. Select the package to be deployed, and deploy the package.
2. All items for deployment will be validated and configured. (Note the **Missing Reference** status of the 'Application Approved' and 'Denallix College Application' items. The meanings of the relevant icons will be explained below).



Deploy Package

Validate the deployment package and configure items for deployment

Configure ▾ | Analyze

Name	Type	Action	Status
College (NvdB)	Category	Use Existing Category	✓ Using: Category
Application Approved	Workflow	Create New Version ...	⚠ Missing Reference
Course	SmartObject	Use Existing SmartOb...	✓ An identical object was f...
Course Fees List	View	Use Existing View	✓ An identical object was f...
Course Item	View	Create New Version ...	✓ Existing item found on t...
Course List	View	Use Existing View	✓ An identical object was f...
Denallix College Application	Form	Create New Version ...	❗ Missing Reference
Student	SmartObject	Use Existing SmartOb...	✓ An identical object was f...
Student Form	View	Use Existing View	✓ An identical object was f...
Additional Items (Uncategorized)			
College logo	Image	Use Existing Image	✓ An identical object was f...
From Address	Field	Use Existing Field	✓ An identical object was f...
k2.denallix.com Runtime	Field	Use Existing Field	✓ An identical object was f...
Mail Server	Field	Use Existing Field	✓ An identical object was f...
SmartObject Server	Field	Use Existing Field	✓ An identical object was f...

Properties | Dependencies | Used By

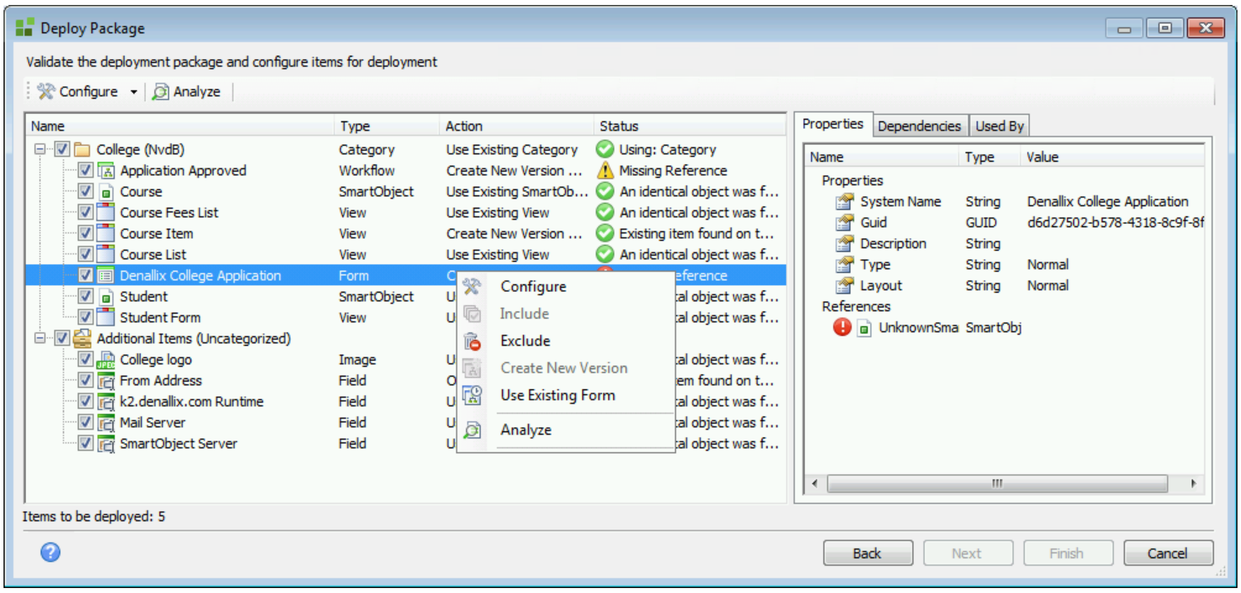
Name	Type	Value
Properties		
System Name	String	College NvdB\Application App
Display Name	String	Application Approved
ID	Int32	10007

Items to be deployed: 4

To ensure that the deployment is successful, make sure that all conflicts have been resolved, then check for warning icons.

Icon	Description
✓	This indicates that the action selected for the deployment of the artifact has no system warnings or conflicts, and the artifact is ready to be deployed.
⚠	This icon has two potential meanings: a. The action selected for the deployment of the artifact has raised a system warning. The action should be reviewed to make sure it will perform the desired end result in the target K2 environment. This warning is typically given if the selected action will overwrite an existing Service Instance or Environment Field. An artifact with this status can be deployed. b. A property of a dependent item is in conflict.
❗	This indicates that the action selected is in system conflict with the K2 environment. An artifact with this status cannot be deployed.

3. To resolve a package conflict by re-configuring a conflicting object, right-click on the item which is in conflict. Select an action from the dropdown menu, or right-click the affected item on the main Deployment window, and select an option from the menu to resolve the issue.



Deploy Package

Validate the deployment package and configure items for deployment

Name	Type	Action	Status
College (NvdB)	Category	Use Existing Category	Using: Category
Application Approved	Workflow	Create New Version ...	Missing Reference
Course	SmartObject	Use Existing SmartOb...	An identical object was f...
Course Fees List	View	Use Existing View	An identical object was f...
Course Item	View	Create New Version ...	Existing item found on t...
Course List	View	Use Existing View	An identical object was f...
Denallix College Application	Form	C	eference
Student	SmartObject	U	al object was f...
Student Form	View	U	al object was f...
Additional Items (Uncategorized)			
College logo	Image	U	al object was f...
From Address	Field	O	em found on t...
k2.denallix.com Runtime	Field	U	al object was f...
Mail Server	Field	U	al object was f...
SmartObject Server	Field	U	al object was f...

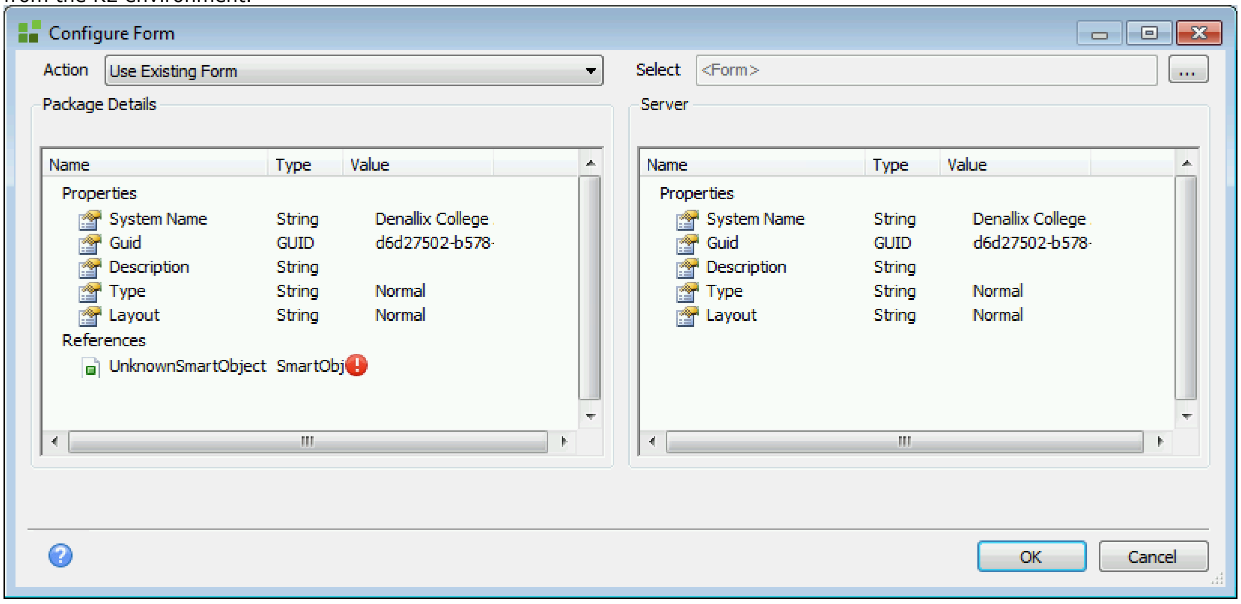
Name	Type	Value
Properties		
System Name	String	Denallix College Application
Guid	GUID	d6d27502-b578-4318-8c9f-8f
Description	String	
Type	String	Normal
Layout	String	Normal
References		
UnknownSma	SmartObj	

Items to be deployed: 5

4. Depending on whether the item is categorized or uncategorized, the menu options will be as follows:

Menu Option	Description
Configure	This option opens the conflict resolution window.
Include	Click this option to include an object in the deployment package. Note that if the object is already included, this action will be grayed out.
Exclude	This action will remove the selected artifact from the deployment package. Note that if the object is already excluded, this action will be grayed out.
Create New Version	Selecting this action configures the artifact to be deployed as a new object, or new version of the existing object, within the K2 environment.
Use Existing	Selecting this action configures the artifact deployment not to use the artifact data from the deployment package. The existing local K2 environment artifact will be used instead, and links will be created from the existing artifact to any other new or updated artifacts from the deployment package that will be deployed to the K2 environment.
Overwrite	Selecting this action configures deployment of the artifact to overwrite the artifact's existing definition/data within the K2 environment.
Analyze	Re-analyzes the selected object for changes in configuration. If the object's status has changed, the new status will be displayed. To see detailed analysis results for the selected object, click the **Properties**, **Dependencies** and **Used By** tabs in the **Deploy Package** window.

5. If the **Configure** option is selected, the conflict resolution window for the artifact type will load. (Note that in this instance, a form has been selected, and the conflict resolution window has opened as **Configure Form**.)
6. To select a different SmartObject, click on the ellipsis button next the 'Select' field box and select the required SmartObject from the K2 environment.



Configure Form

Action: Use Existing Form

Select: <Form>

Package Details

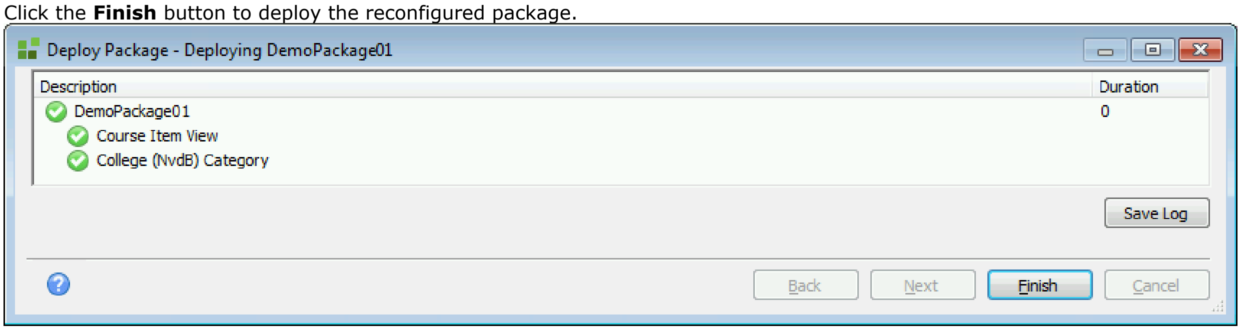
Name	Type	Value
Properties		
System Name	String	Denallix College
Guid	GUID	d6d27502-b578-
Description	String	
Type	String	Normal
Layout	String	Normal
References		
UnknownSmartObject	SmartObj	

Server

Name	Type	Value
Properties		
System Name	String	Denallix College
Guid	GUID	d6d27502-b578-
Description	String	
Type	String	Normal
Layout	String	Normal

7. Once the conflicting properties have been resolved, click **OK**.
8. When all conflicts have been resolved, click the **Next** button in the **Deploy Package** window.

9. Click the **Finish** button to deploy the reconfigured package.



Description	Duration
✓ DemoPackage01	0
✓ Course Item View	
✓ College (NvdB) Category	

10. If you need to refresh the analysis of an artifact, right-click on the artifact, and select Analyze from the pop-up menu.

1.3.3.2 PowerShell Deployments

Deploying K2 Packages Using PowerShell

The PowerShell Snap-in helps to automate the deployment of single or multiple artifact packages from source environments to target environments without the need for user intervention. This may be especially useful where it is necessary to make regular or frequent package deployments, or where script-based installations and updates might be preferred.

The PowerShell Snap-in functionality cannot be used to create deployment packages, but only to deploy them. Package creation can only be done through the K2 Package and Deployment interface.

Note: Powershell package deployments utilize Namespaces. For a detailed explanation, please refer to the Important Considerations section below.

Prerequisites

Microsoft Management Console (MMC) 3.0 or later (part of the Windows Management Framework) is a prerequisite for the K2 Package and Deployment Snap-in for Microsoft Powershell.

The relevant Windows Management Framework package can be downloaded from http://www.microsoft.com/en-za/download/details.aspx?id=34595.

Quick steps

Automated deployments using Powershell are carried out as follows:

1. Create a Deployment Package using the K2 Package and Deployment console
2. Open Powershell. Add the K2 Package and Deployment Snap-in.
3. Run the **Write-DeploymentConfig** command to create the deployment configuration *.XML file file for the .kspx file generated in step one.
4. Edit the deployment configuration file to customize the default settings for the specific deployment.
5. Run the **Deploy-Package** command to deploy the package.

Automating a package deployment using PowerShell

1. Open Microsoft PowerShell with Administrator privileges.

   Note: When copying code from this help file to the PowerShell interface, make sure there are no unwanted line breaks in the code text.

2. Enter the PowerShell environment setup command. This command must be entered every time PowerShell is launched when performing a PowerShell deployment.

   Copy Code

   ```
   #Adds the PowerShell Deployment snapin.
   add-pssnapin SourceCode.Deployment.PowerShell
   ```

3. The **Write-DeploymentConfig** command examines the K2 Package and creates a default configuration *.XML file. This file is used by the Deploy-Package command to create or update K2 artifacts within the target environment. A sample Write-DeploymentConfig command might be written as follows:

   Copy Code

   ```
   #Scripts the sample XML configuration file to be used for
   deployment.
   Write-DeploymentConfig 'C:\Users\Dewald\Desktop\Perf
   Packages\RTM\Package3.kspx' 'C:\Users\Dewald\Desktop\Perf
   Packages\RTM\Package3.xml'
   ```

4. The following options may be used to customize the **Write-DeploymentConfig** command:
   - **-InputFile** (where 'InputFile' is the *.KSPX (K2 Package) file to be deployed.)
   - **-OutputFile** (where 'OutputFile' is the corresponding *.XML configuration file which will be generated.)
   - An optional command is **-ConnectionString** (Connection string to K2 server, for example:

"Host=LOCALHOST;Port=5555;Integrated=True;IsPrimaryLogin=True". If no value is specified, known defaults will be used.)

5. If necessary, edit the contents of the configuration *.XML file to customize the default settings, using the **ResolveOptions** options below.
6. The **ResolveOptions** section of the *.XML configuration file contains two sub-sections:
   - **defaultOptions** (This section sets the default action for the Namespaces contained in the K2 package.)
   - **specificOptions** (This section sets actions for each of the Artifacts contained in the K2 package.)

   The following configuration options are supported by the K2 PowerShell snap-in. If needed, these options must be manually inserted into the *.XML configuration file:

   - **Default** (Applies the default action previously set within the GUI.)
   - **Deploy** (Deploys the given item, providing the named item is contained within the relevant *.KSPX file).
   - **Exclude** (Excludes the given item, providing the named item is contained within the relevant *.KSPX file).
   - **UseExisting** (Directs use of the existing item, providing the named item is contained within the relevant *.KSPX file).
     In order for the **UseExisting** option to function, an item must already exist on the target environment to which the new item is being deployed. If the item being deployed has a different name to the existing item, the **targetName** and **targetNamespace** attributes of the existing item must be specified within the *.XML configuration file.

   For example, where "GIVEN ITEM" is the given item, the following configuration option will deploy the item:

   <resolve name="GIVEN ITEM" action="Deploy" namespace="urn:SourceCode/Files"/>

   For more information on variables, refer to the 'Assigning Variables' sub-section of 'Creating a Package'.

7. Once the package configuration *.XML file has been created and edited, use the **Deploy-Package** command to deploy the package. This command is essential and must be entered. Options for the Deploy-Package command are as follows:
   - **-FileName** (Required) The full name and file extension of the K2 package *.KSPX file are required for the file to be deployed.
   - **-ConfigFile** (Optional) Should an *.XML configuration file with the same name as the package file already exist in the same target location, it will be used. If the *.XML configuration file is not located in the same location as the package file, specify the location using this command option.
   - **-ConnectionString** (Optional) Denotes the connection string to K2 server, for example: "Host=LOCALHOST;Port=5555;Integrated=True;IsPrimaryLogin=True". If no value is specified, known defaults will be used.
   - **-NoAnalyze** (Optional) This is a switch parameter, indicating whether the package should be loaded in 'No Analyze' mode. If not specified, the **$false** switch value will be used, i.e. the package will be analyzed by default.

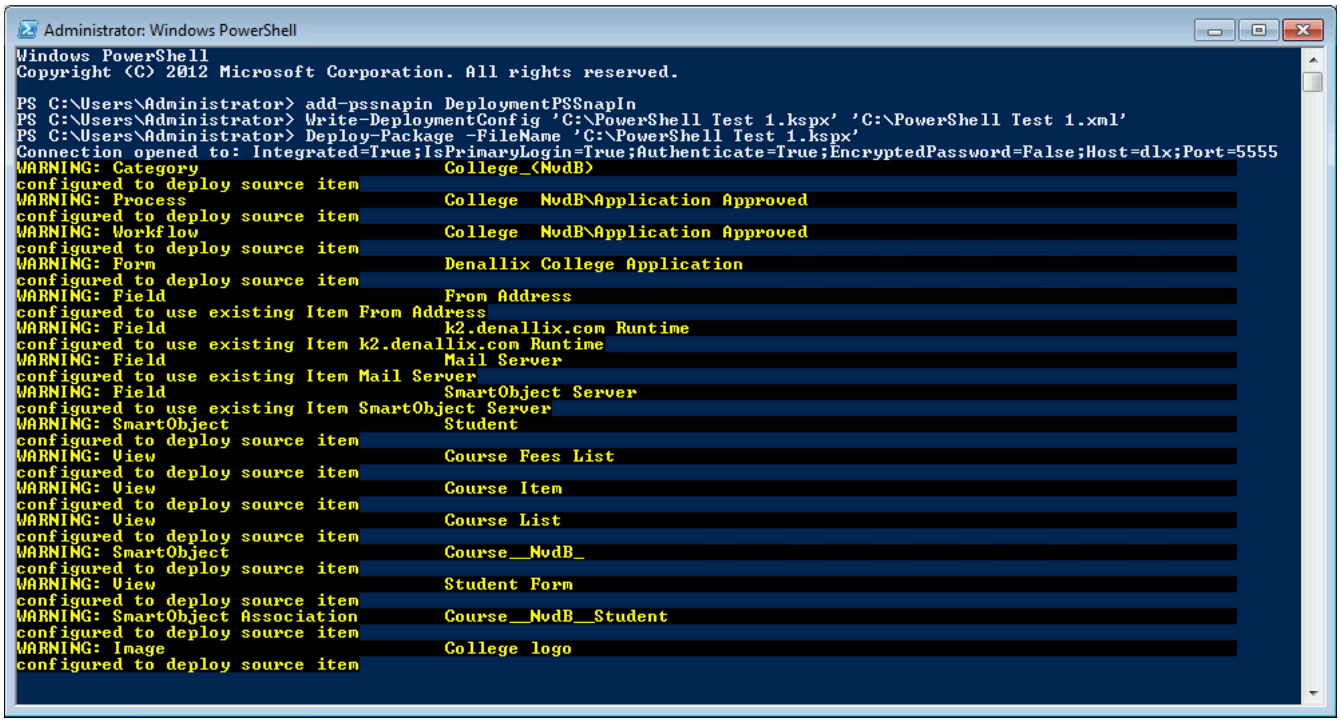
8. If needed, the **-ConnectionString** option may be broken down to the following advanced values:
   - **-K2Host** An option denoting the K2 Server host name. if no value is specified, a "LOCALHOST" default will be used.
   - **-Port** An option denoting the port through which Package and Deployment will connect to the K2 Server. Should no value be specified, the "5555" default will be used.
   - **-Integrated** (Optional) This is used to define if the connection is integrated (e.g. $true) or not. If no value is specified, the "$true" default will be used.
   - **-IsPrimaryLogin** (Optional) If no value is specified, the "$true" default will be used.
   - **-UserName** (Required if the 'Integrated' value above is set as 'False') It denotes the user name to be used for connection to the K2 Server.
   - **-Password** (Required if the 'Integrated' value above is set as 'False') The relevant password for the user name must be entered.
   - **-WindowsDomain** (Required if the 'Integrated' value above is set as 'False') The domain relevant to the **UserName** option
   - **-SecurityLabel** (Required if the 'Integrated' value above is set as 'False') For example, 'K2'.

   A sample Deploy-Package command might be written as follows:

   Copy Code

```
#Scripts the deployment of a package
Deploy-Package -FileName 'C:\TestPackage.kspx' -ConfigFile
'C\TestPackage.XML'
```

9. If the automated deployment is successful, the deployed items will be shown on the PowerShell interface. (Note that the screen shot below does not show all the available commands and is for demonstration only.)



```
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> add-pssnapin DeploymentPSSnapIn
PS C:\Users\Administrator> Write-DeploymentConfig 'C:\PowerShell Test 1.kspx' 'C:\PowerShell Test 1.xml'
PS C:\Users\Administrator> Deploy-Package -FileName 'C:\PowerShell Test 1.kspx'
Connection opened to: Integrated=True;IsPrimaryLogin=True;Authenticate=True;EncryptedPassword=False;Host=dlx;Port=5555
WARNING: Category                       College_(NvdB)
configured to deploy source item
WARNING: Process                        College  NvdB\Application Approved
configured to deploy source item
WARNING: Workflow                       College  NvdB\Application Approved
configured to deploy source item
WARNING: Form                           Denallix College Application
configured to deploy source item
WARNING: Field                          From Address
configured to use existing Item From Address
WARNING: Field                          k2.denallix.com Runtime
configured to use existing Item k2.denallix.com Runtime
WARNING: Field                          Mail Server
configured to use existing Item Mail Server
WARNING: Field                          SmartObject Server
configured to use existing Item SmartObject Server
WARNING: SmartObject                    Student
configured to deploy source item
WARNING: View                           Course Fees List
configured to deploy source item
WARNING: View                           Course Item
configured to deploy source item
WARNING: View                           Course List
configured to deploy source item
WARNING: SmartObject                    Course__NvdB_
configured to deploy source item
WARNING: View                           Student Form
configured to deploy source item
WARNING: SmartObject Association        Course__NvdB__Student
configured to deploy source item
WARNING: Image                          College logo
configured to deploy source item
```

Important Considerations

The K2 Deployment PowerShell Snap-in uses a combination of Namespace/Class structures and Name/Namespace structures in order to deploy the package to the target environment.

It must be noted that Service Instances use a Namespace/Class structure. All other packaged items use a Name/Namespace structure. Note that all Namespace/Class and Name/Namespace structures must be unique to the specific item being described.

As an example, Namespace/Class conditions for Service Instances must be taken into account as follows within the configuration *.XML file. (The %2E in this case represents a full stop [i.e. '.']):

Copy Code

```
#urn:SourceCode/SmartObjects/ServiceInstance

SourceCode%2ESmartObjects%2EServices%2ESQL%2EsqlServerService
```

The Package and Deployment namespace structure relates to namespace and class as defined within the code itself. For example, the code in the block above is not a reference to the Assembly Name, but rather to the Namespace and Class within the Assembly, which inherits properties from ServiceAssemblyBase. For the example above, the name of the Assembly is 'SourceCode.SmartObjects.Services.SqlServer.dll'. However, the namespace and class structure is as per the code block below:

Copy Code

```
namespace

SourceCode.SmartObjects.Services.SQL

 {

public class SqlServerService : ServiceAssemblyBase

   {
```

Some examples are as follows:

urn:SourceCode/Categories?AdventureWorks#Path.%2F represents a 'root' category called 'AdventureWorks'.

Thus, the output would be:

1. AdventureWorks

The representation above is explained as follows:

urn:SourceCode/Categories defines the namespace of the object.

?AdventureWorks defines the 'Instance name' of the object (In this case, a category called AdventureWorks (i.e. '/AdventureWorks/')).

#Path defines the object path. In this instance, the object path is 'blank', as AdventureWorks is the 'root' category.

%2F is the path-separator symbol, being URI code for the 'forward slash' used as a separator.

urn:SourceCode/Categories?Production#Path.%2FAdventureWorks %2F represents a sub-category under 'AdventureWorks' called 'Production'.

Thus, the output would be:

1. AdventureWorks

a. Production

The definitions are the same as for the first example. However, in this instance, **Path.%2FadventureWorks%2F** defines the object path (In this case, a category below 'AdventureWorks' called 'Production' (i.e. /AdventureWorks/Production/)).

urn:SourceCode/Categories?Forms#Path.%2FAdventureWorks%2FProduction%2F represents a sub-category under 'AdventureWorks' and 'Production' called 'Forms'.

Thus, the output would be:

1. AdventureWorks

a. Production

i. Forms

The definitions are the same as for the first example. However, in this instance, **Forms#Path.%2FAdventureWorks%2FProduction%2F** defines the object path (in this case, a sub-category below 'AdventureWorks' and 'Production', called 'Forms', i.e. /AdventureWorks/Production/Forms/).

PowerShell deployments to distributed environments

When PowerShell is used to make a deployment to a distributed environment, you will need to specify named parameters, including a connection string, in the following format: -FileName, -ConfigFile, -ConnectionString.

A sample command for deployments to distributed environments might be presented as follows:

Copy Code

```
Deploy-Package -FileName 'C:\GenericPackage.kspx' -ConfigFile
'C:\GenericPackage.XML' -ConnectionString
'Host=main.hostserver.local;Port=5555;Integrated=True;
IsPrimaryLogin=True;SecurityLabelName=K2;UserID=User1;
Password=mypassword;WindowsDomain=HOSTSERVER'
```

Deployment log file

By default, the deployment log file is saved to the same location as the package (*.KSPX) file used for the given deployment. Should any deployment errors occur and require troubleshooting, these errors will be recorded within the log file.

1.3.4 Troubleshooting K2 Package and Deployment

Overview

The articles within this topic describe potential issues which may appear during installation and/or usage of K2 Package and Deployment, with suggested solutions for each issue.

The K2 Package and Deployment installation fails

Symptom: K2 Package and Deployment encounters installation issues, or does not install, due to missing software pre-requisites.

Explanation/Solution: In order for K2 Package and Deployment to be successfully installed, the following software packages are required:

- K2 blackpearl 4.6.5 or later.
- Microsoft Management Console (MMC) 3.0.
- K2 smartforms is an optional requirement. Should K2 smartforms be installed, version 1.0.3 or later is required.

Packaging and/or deployment speed is low

Symptom: When K2 Package and Deployment is used to deploy packages containing large numbers of artifacts (800 or more), packaging and/or deployment speed is lower than normal.

Explanation: Solutions containing very high numbers of artifacts require more system hardware resources for processing. This may result in lower-than-normal processing speeds and greater RAM usage.

Solution: Should you encounter lower-than-normal processing speeds and/or RAM usage is too high when large numbers of artifacts are processed, it is recommended that the CPU and RAM used to run the source and target environments be upgraded.

Some or all SmartObjects are missing from a deployed package

Symptom 1: When a package containing SmartObjects is deployed, it is discovered that some or all SmartObject data is missing from the deployed package.

Explanation: When SmartObjectData is packaged, the relevant SmartObject definitions are included for deployment. However, the relevant SmartObject data is de-selected. This is the default behavior, allowing the user to control whether potentially oversized SmartObject data artifacts will or will not be deployed.

Solution: If data comprising a given SmartObject must be included in a deployment package, the user must open the relevant SmartObject node, and manually select the required SmartObject Data artifact.

Symptom 2: SmartObject data derived from a non-SmartBox source cannot be included in a deployment package.

Explanation: At this time, SmartObject packaging functionality is only available for SmartObject data derived from SmartBox sources. SmartObject data derived from other sources is not currently compatible with K2 Package and Deployment.

Solution: If SmartObject data must be contained within a deployment package, the relevant data must be obtained from SmartBox sources only.

Note: SmartObjectData may only be deployed as part of the parent SmartObject artifact. Should a relevant SmartObject definition already be deployed within the target environment, the user will be required to configure the deployment action to use the existing SmartObject definition, or to overwrite the existing definition.

Symptom 3: For manual deployments made using the Package and Deployment console, and automated deployments made using the PowerShell Snap-in, an exception stating that the destination table cannot be accessed is given.

Explanation: If a SmartObject has been renamed and the new name includes spaces, the SmartObject Service will automatically change the space to an underscore (i.e. '_') when the back-end system name for the SmartObject is created. However, the SmartObject Service does not redefine data node names this way. Therefore, the space will remain in the data node name, and if a SmartObject is renamed to include spaces in the object name before being included in a K2 package, deployment of that package will fail.

Solution: Where SmartObjects are to be renamed before being deployed, all words which make up the name should be separated by underscores (i.e. '_'). Spaces must not be used.

Dependency issues during package creation

Symptom: A package does not deploy in the expected manner, or does not respond as expected when deployed.

Explanation/Solution: When creating a package, the **Automatically include dependent items** check box must be selected. It is strongly recommended that this check box be de-selected only by a K2 Administrator. If this check box is de-selected, each node and sub-node of all artifacts to be packaged must be manually expanded and selected.

- When packaging and deploying Service Instance artifacts, ensure that the associated back-end entities exist on the target environment before attempting to deploy the package. See the K2 Package and Deployment Overview for more information.
- The **Continue deploying packaged items if one or more items cannot be deployed** check box is checked by default. Deselect this check box where it is necessary that the deployment be stopped if any artifacts or SmartObjects cannot be found, or are incompatible with the current deployment.
- All conflicts must be resolved before the package can be deployed to the new environment. If a conflict cannot be resolved, the artifact or dependency causing the conflict must be deselected. See the Conflict Resolution section for more details. Once all conflicts have been resolved, run the **Analyze All** option to update the internal status check.

Oracle, SQL Server, CRM or SharePoint Service Instances do not deploy

Symptom: When packaging and deploying Service Instances created using Oracle, SQL Server, CRM or SharePoint, the relevant Service Instances cannot be deployed to the target environment.

Explanation/Solution: Before packaging and deploying Oracle, SQL Server, CRM or SharePoint Service Instances, ensure that the associated Oracle databases/SQL Databases/Entities/Lists and Libraries exist on the target environment, and have the same schema as the Service Instances to be deployed.

If you are deploying a SalesForce Service Instance using K2 Package and Deployment, an existing SalesForce Service Instance must already be present on the target environment. The packaged Service Instance must be re-bound to the existing Service Instance once the package has been deployed.

User barred from executing deployments (Deployment permissions denied)

Symptom: K2 Package and Deployment does not allow the user to deploy certain artifacts and/or dependencies due to lack of appropriate permissions.

Explanation: All calls to the deployment server are validated. In addition, if SmartObject security has been set, then the deploying user must also be granted 'Publish SmartObject' permissions. Users possessing deployment rights are able to deploy any packaged artifact to the target server, and such users subsequently become the 'owner' of the deployed process.

Solution: Users who are required to deploy packages, artifacts or process rights must be granted K2 deployment (i.e. export) rights. Should any user other than the process owner (i.e. deploying user) require the right to read and/or edit the deployed process, the process owner must enable sharing of the relevant process.

Issues caused by unsupported artifact types

Symptom: K2 Package and Deployment does not allow certain artifact and/or dependency types to be included in a deployment package.

Explanation: The following artifacts and dependencies (including SmartObjects, Workflows, Forms and Views) can be packaged and deployed using K2 Package and Deployment:

- K2 Studio artifacts and dependencies.
- K2 for Visual Studio artifacts and dependencies.
- Supporting artifacts such as categories, roles, service instances, notification events and environment library fields may also be included in a package wherever a main artifact references them.

The following artifacts and dependencies are not supported at this time:

- Microsoft InfoPath artifacts and dependencies.
- Microsoft SharePoint artifacts and dependencies.
- Workflow-integrated artifacts designed using K2 Studio or K2 Designer for Visual Studio.
- Artifacts and dependencies designed using K2 Designer for SharePoint.
- Non-K2 artifacts, such as Microsoft SharePoint lists and libraries.
- Custom environment configurations (for example, any changes to configuration files, such as **K2HostServer.exe.config**) cannot be packaged. These configurations must be manually applied to the target environment.

Solution: Deployment packages should only include artifacts and dependencies which are supported by K2 Package and Deployment.

User not able to apply SmartActions commands

Symptom: If SmartActions is enabled at the source environment, but disabled at the target environment, users will be unable to apply SmartActions-related commands within the target environment as the needed commands will be unavailable.

Explanation: To be able to apply SmartActions-related commands within the target environment, SmartActions must be enabled within the target environment.

It is strongly recommended that any anti-virus software be disabled before proceeding with K2 blackpearl configuration or modification.

Solution: Within the target environment:

1. Click the Windows **Start** button. Navigate to **All Programs**.
2. Hover the mouse cursor over **K2 blackpearl**. Click **K2 blackpearl Setup Manager**.
3. Within the **K2 blackpearl Maintenance** setup screen, select the **Configure K2 blackpearl** radio button. Click **Next**.
4. The K2 blackpearl Setup Manager will launch.
5. Navigate to the **SmartActions Configuration** screen. Ensure that the **Enable SmartActions for Exchange** check box is checked.
6. Click **Next** to navigate through the remaining screens until the **Configuration Analysis** screen shows that all components have been correctly configured (i.e. all items show green check marks next to them).
7. Click **Next**. The **Finished** screen will appear.
8. Click **Finish**. SmartActions will now be available within the target environment.

1.3.5 Example 1 - Packaging and Deploying a set of SmartObjects

K2 Package and Deployment can be used to transfer SmartObjects and their Associations between one K2 environment and another (for example, between a development environment and a production environment).

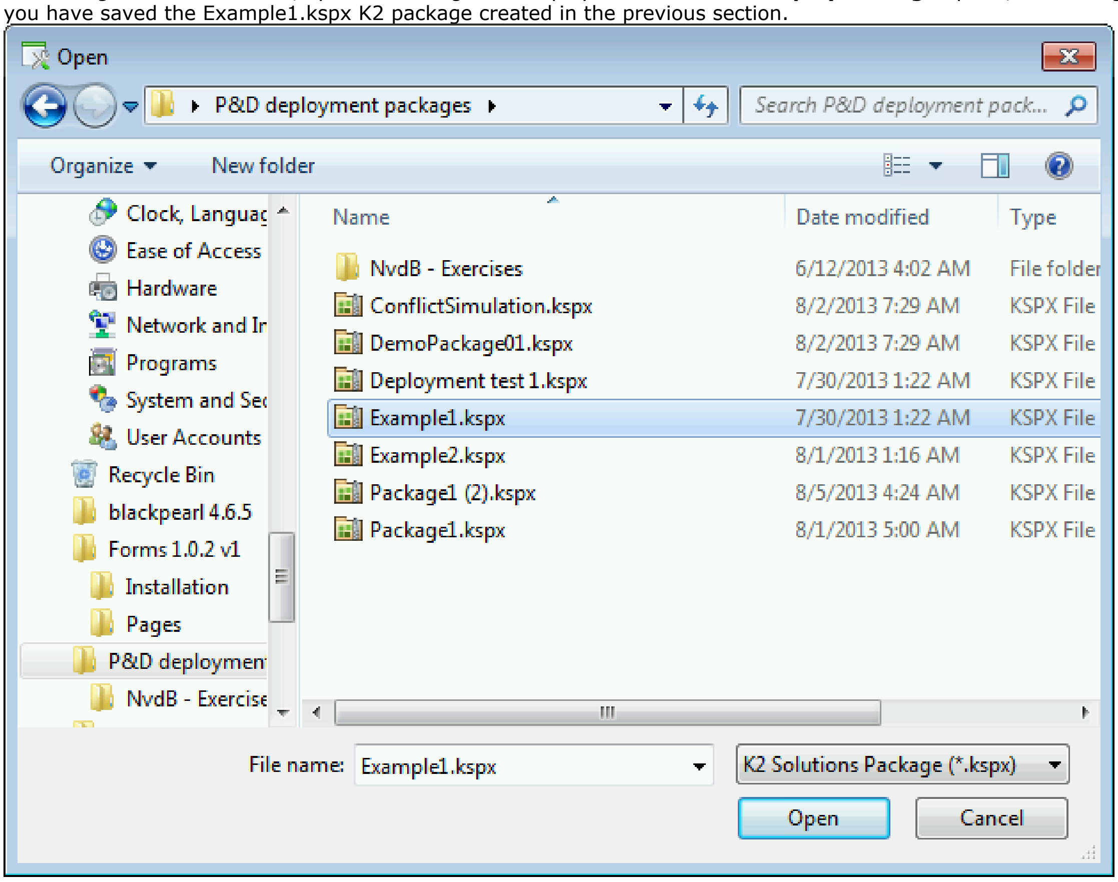
This example uses the College Application example project from the K2 smartforms documentation. The SmartObjects and their defined association sets will be selected during the package creation step. They will then be deployed to a new K2 environment.

Note that in order for the SmartObjects from the K2 smartforms College Application demonstration to be discovered by the K2 Package and Deployment tool, the associated Views must be checked in.

Creating the Package

Deploying the Package in a New Environment

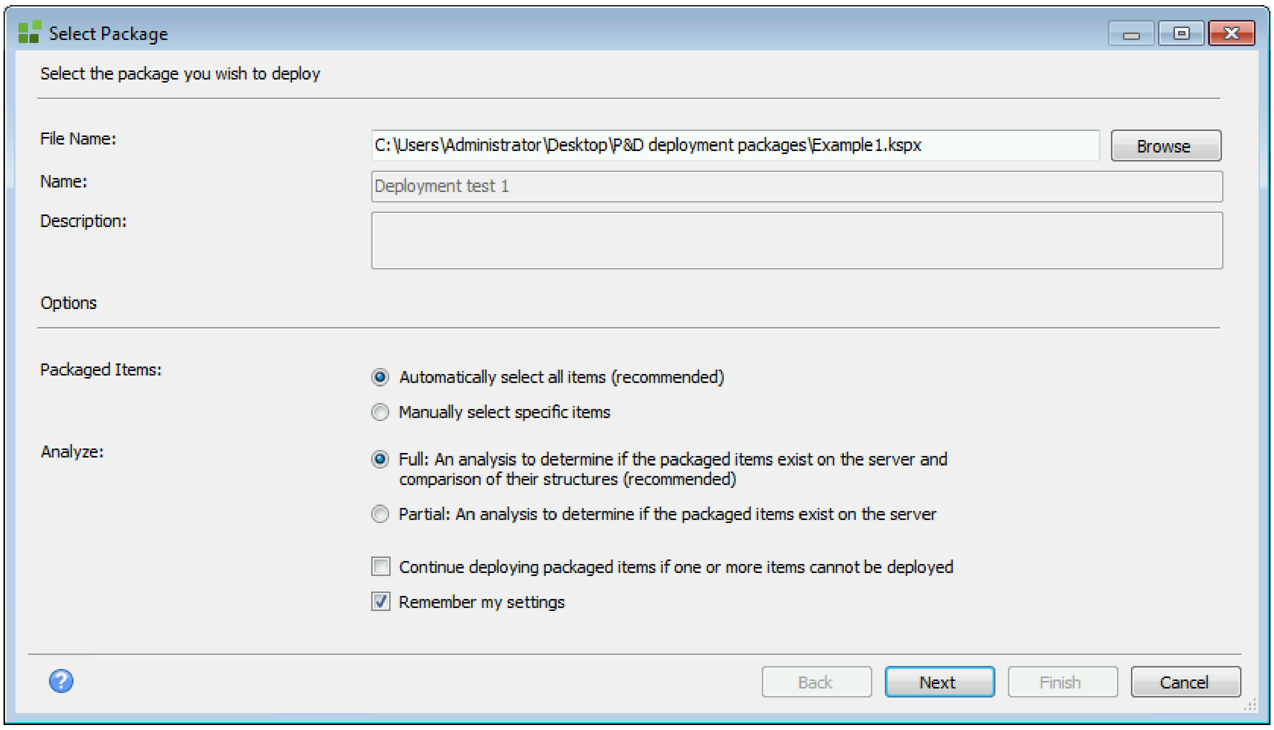
1. In the target K2 environment, open K2 Package and Deployment. Select the **Deploy Package** option, then navigate to where you have saved the Example1.kspx K2 package created in the previous section.



Name	Date modified	Type
NvdB - Exercises	6/12/2013 4:02 AM	File folder
ConflictSimulation.kspx	8/2/2013 7:29 AM	KSPX File
DemoPackage01.kspx	8/2/2013 7:29 AM	KSPX File
Deployment test 1.kspx	7/30/2013 1:22 AM	KSPX File
Example1.kspx	7/30/2013 1:22 AM	KSPX File
Example2.kspx	8/1/2013 1:16 AM	KSPX File
Package1 (2).kspx	8/5/2013 4:24 AM	KSPX File
Package1.kspx	8/1/2013 5:00 AM	KSPX File

File name: Example1.kspx
K2 Solutions Package (*.kspx)

2. Click the **Open** button to load the package into the Deploy Package wizard.



Select Package

Select the package you wish to deploy

File Name:	C:\Users\Administrator\Desktop\P&D deployment packages\Example1.kspx
Name:	Deployment test 1
Description:	

Options

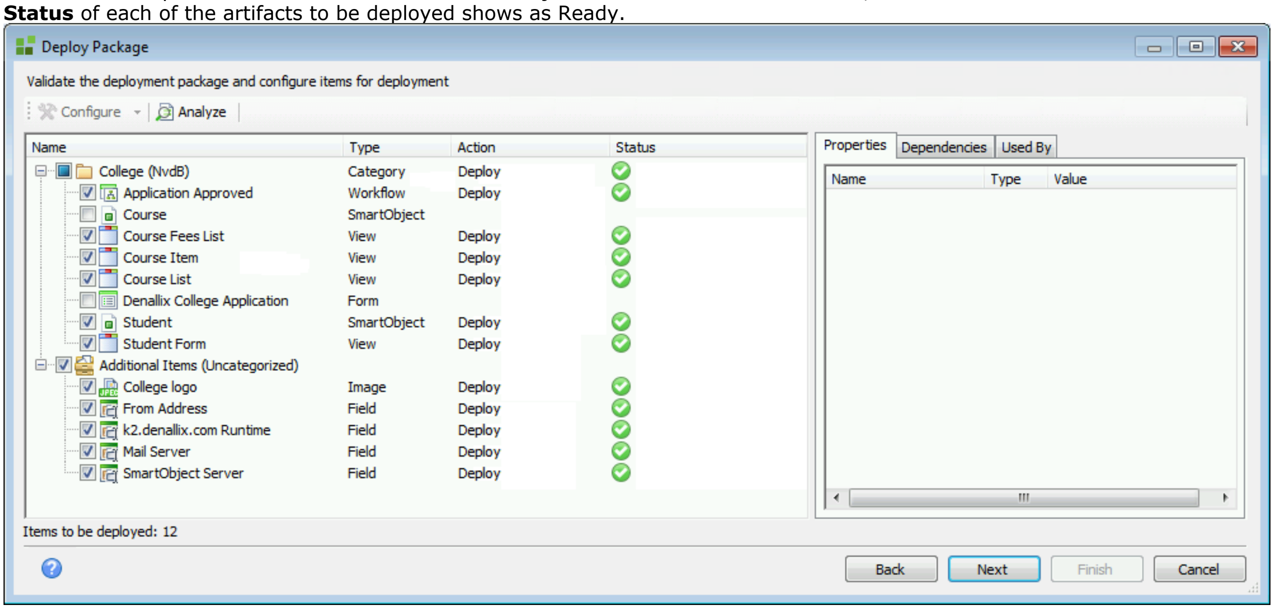
Packaged Items: [✓ Automatically select all items (recommended)] Manually select specific items

Analyze: [✓ Full: An analysis to determine if the packaged items exist on the server and comparison of their structures (recommended)] Partial: An analysis to determine if the packaged items exist on the server

☐ Continue deploying packaged items if one or more items cannot be deployed

☑ Remember my settings

3. Click **Next** to open the Deploy Package Wizard.
4. As there are no previous definitions for these SmartObjects in the new K2 environment, there are no conflicts to resolve. The **Status** of each of the artifacts to be deployed shows as Ready.



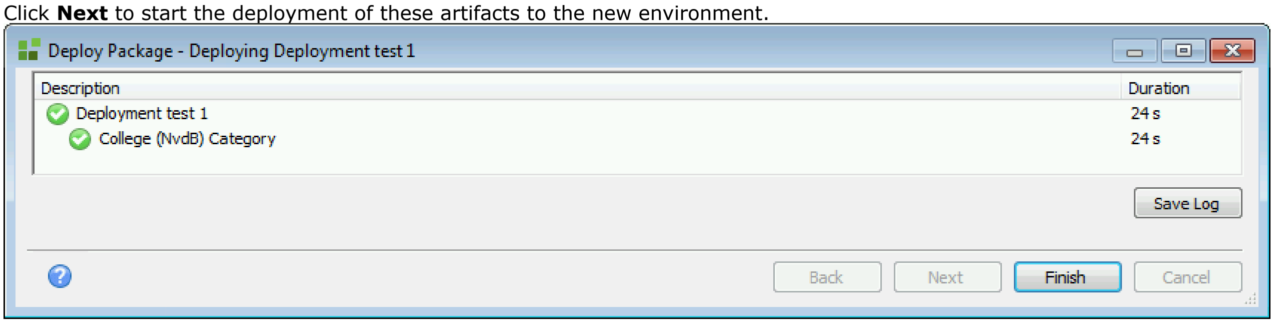
Deploy Package

Validate the deployment package and configure items for deployment

Name	Type	Action	Status
College (NvdB)	Category	Deploy	✓
☑ Application Approved	Workflow	Deploy	✓
☐ Course	SmartObject		
☑ Course Fees List	View	Deploy	✓
☑ Course Item	View	Deploy	✓
☑ Course List	View	Deploy	✓
☐ Denallix College Application	Form		
☑ Student	SmartObject	Deploy	✓
☑ Student Form	View	Deploy	✓
☑ Additional Items (Uncategorized)			
☑ College logo	Image	Deploy	✓
☑ From Address	Field	Deploy	✓
☑ k2.denallix.com Runtime	Field	Deploy	✓
☑ Mail Server	Field	Deploy	✓
☑ SmartObject Server	Field	Deploy	✓

Items to be deployed: 12

5. Click **Next** to start the deployment of these artifacts to the new environment.



Deploy Package - Deploying Deployment test 1

Description	Duration
Deployment test 1	24 s
College (NvdB) Category	24 s

6. K2 Package and Deployment will report on the results of the deployment to the K2 Server. In the view above, all artifacts have been successfully deployed to the new K2 environment.
7. If you need to save a log of this action then select the **Save Log** option. Click the **Finish** button to close the Deployment wizard.

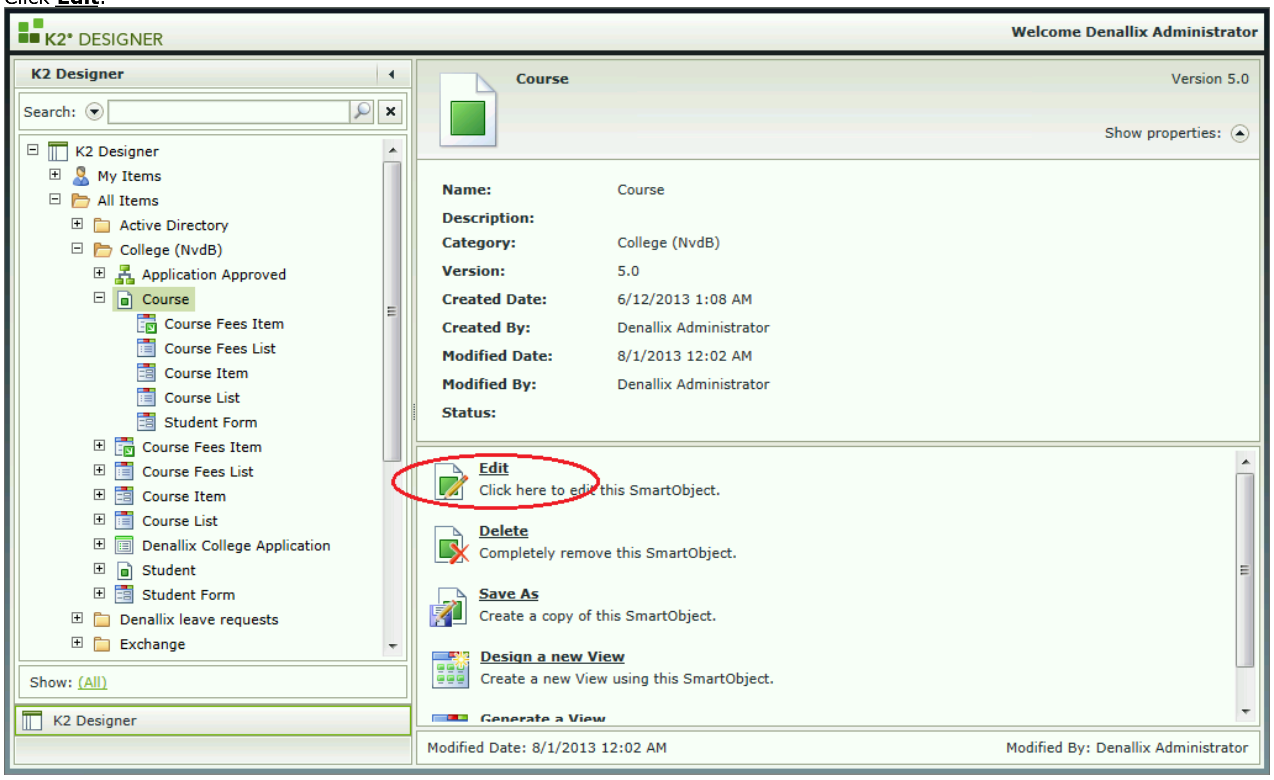
1.3.6 Example 2 - Packaging and Deploying an Updated Set of SmartObjects

K2 Package and Deployment can be used to update SmartObjects between one K2 environment and another. This example uses the College Application example project from the K2 smartforms documentation. The SmartObjects will be selected during the package creation step. They will then be deployed to a K2 environment where earlier versions of these SmartObjects and associations have already been deployed.

Creating the Package

This example uses the same SmartObjects as Example 1. Using K2 smartforms, you will edit and redeploy one of the SmartObjects created for the College Application.

1. In the source K2 environment, open the web browser. Launch K2 Designer.
2. Expand the **K2 Designer** node, the **All Items** node, and the **College** node as in the view below.
3. Click on a SmartObject to be changed. In the example below, the **Course** object is being used.
4. Click **Edit**.



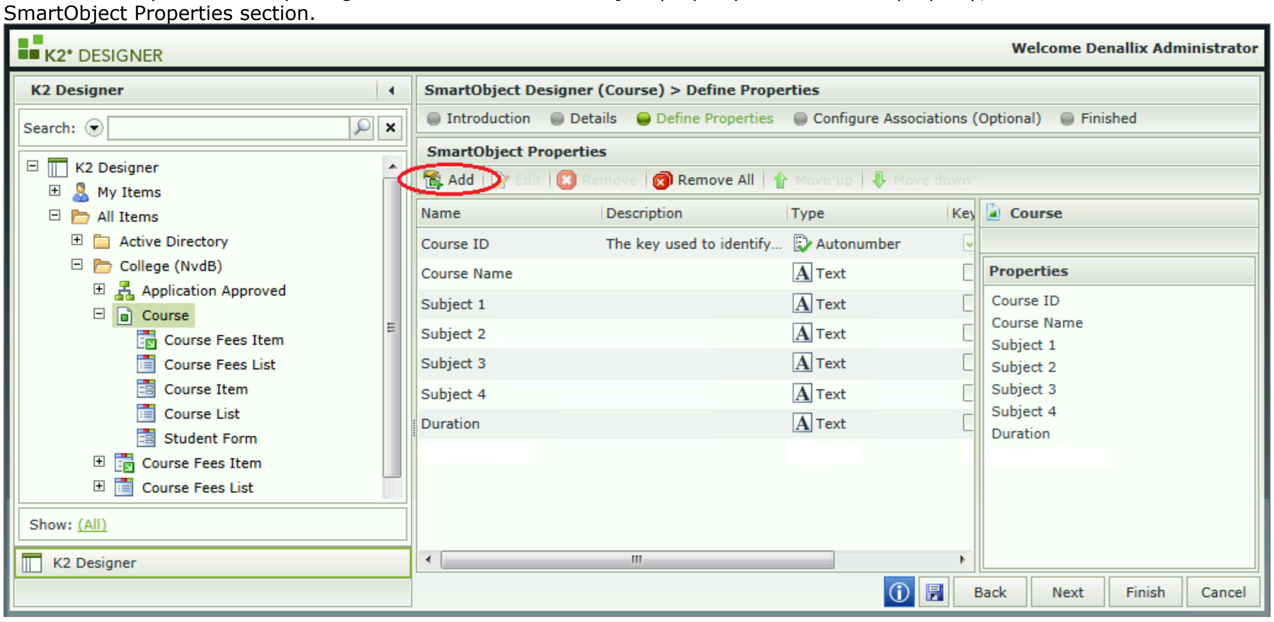
Course — Version 5.0

Name:	Course
Description:	
Category:	College (NvdB)
Version:	5.0
Created Date:	6/12/2013 1:08 AM
Created By:	Denallix Administrator
Modified Date:	8/1/2013 12:02 AM
Modified By:	Denallix Administrator
Status:	

Edit — Click here to edit this SmartObject.
Delete — Completely remove this SmartObject.
Save As — Create a copy of this SmartObject.
Design a new View — Create a new View using this SmartObject.

Modified Date: 8/1/2013 12:02 AM — Modified By: Denallix Administrator

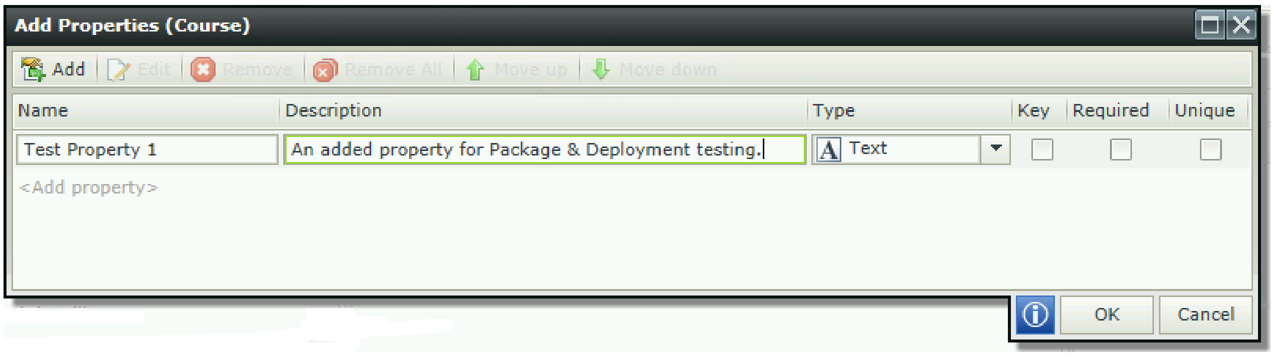
5. The easiest way to alter the package is to add a new SmartObject property. To add a new property, click **Add** in the SmartObject Properties section.



SmartObject Designer (Course) > Define Properties

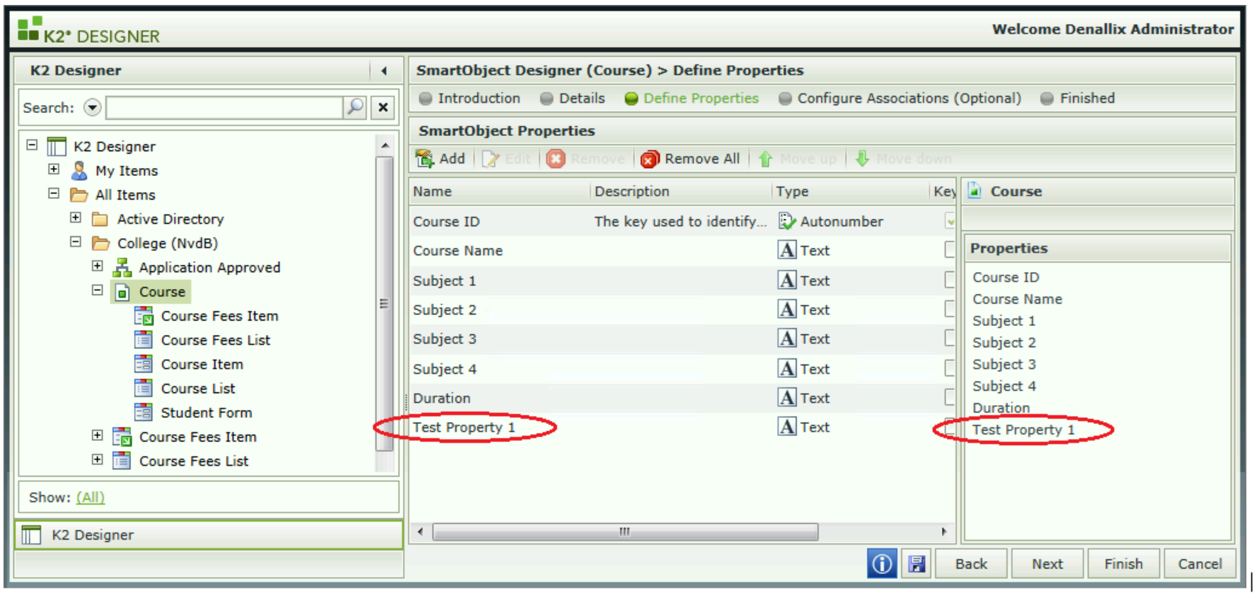
Name	Description	Type	Key
Course ID	The key used to identify...	Autonumber	✓
Course Name		Text	
Subject 1		Text	
Subject 2		Text	
Subject 3		Text	
Subject 4		Text	
Duration		Text	

6. The **Add Properties** dialog will open. Enter a Name and Description for the new property, and specify a property Type. Click **OK**.



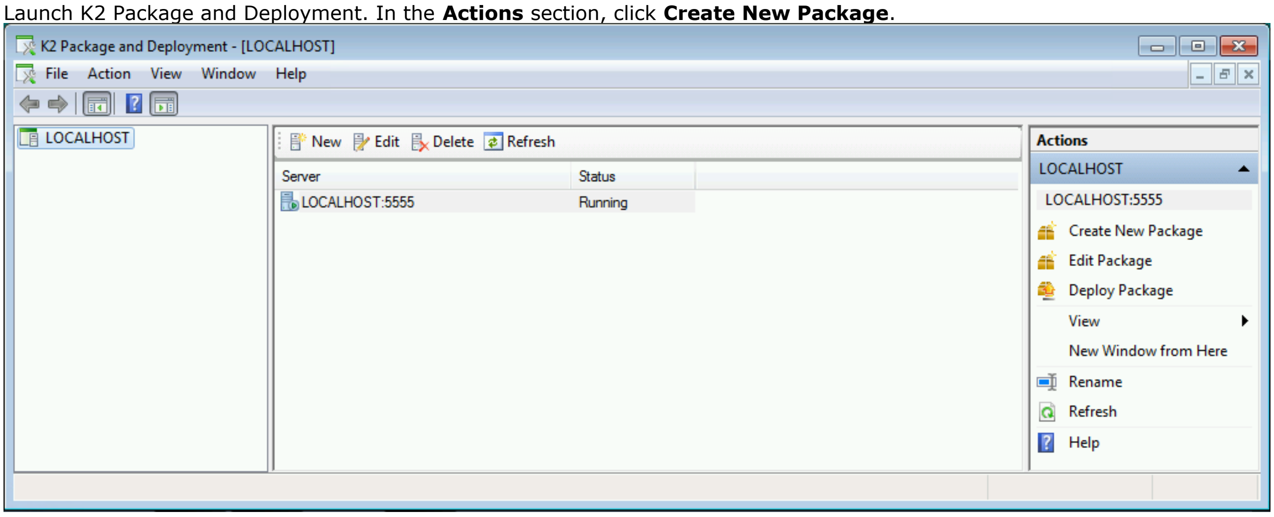
Name	Description	Type	Key	Required	Unique
Test Property 1	An added property for Package & Deployment testing.	Text	☐	☐	☐

7. The new property will now be listed in the **SmartObject Properties** and **Properties** sections of the designer. Click **Finish**.



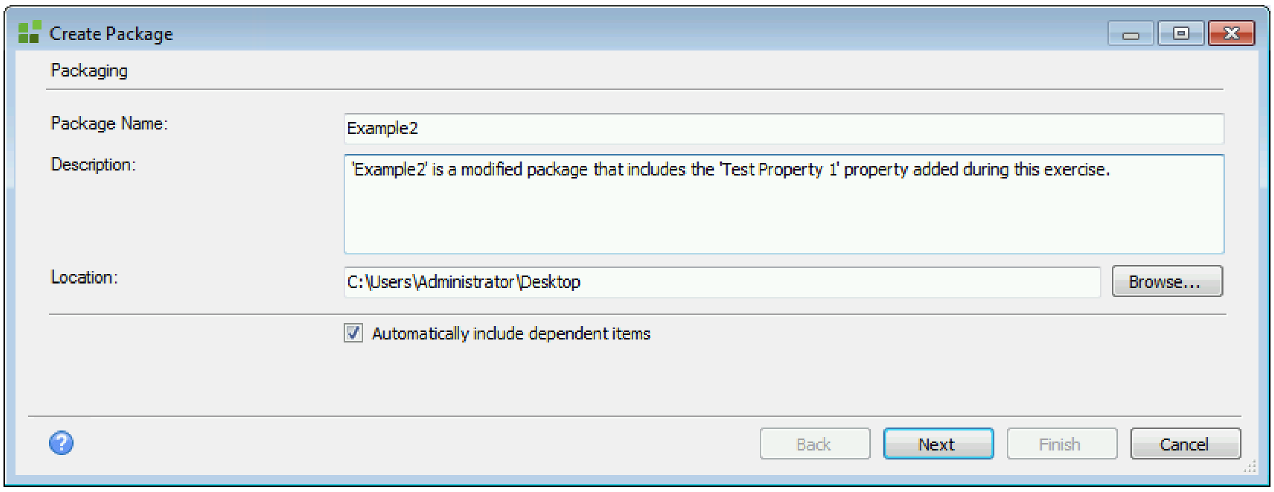
Name	Description	Type	Key
Course ID	The key used to identify...	Autonumber	✓
Course Name		Text	☐
Subject 1		Text	☐
Subject 2		Text	☐
Subject 3		Text	☐
Subject 4		Text	☐
Duration		Text	☐
Test Property 1		Text	

8. Close the browser.
9. Launch K2 Package and Deployment. In the **Actions** section, click **Create New Package**.



Server	Status
LOCALHOST:5555	Running

10. The **Create Package** window will open. Enter a Package Name and Description for the new package. Make sure that the **Automatically include dependent items** check box is checked.
11. Click **Next**.



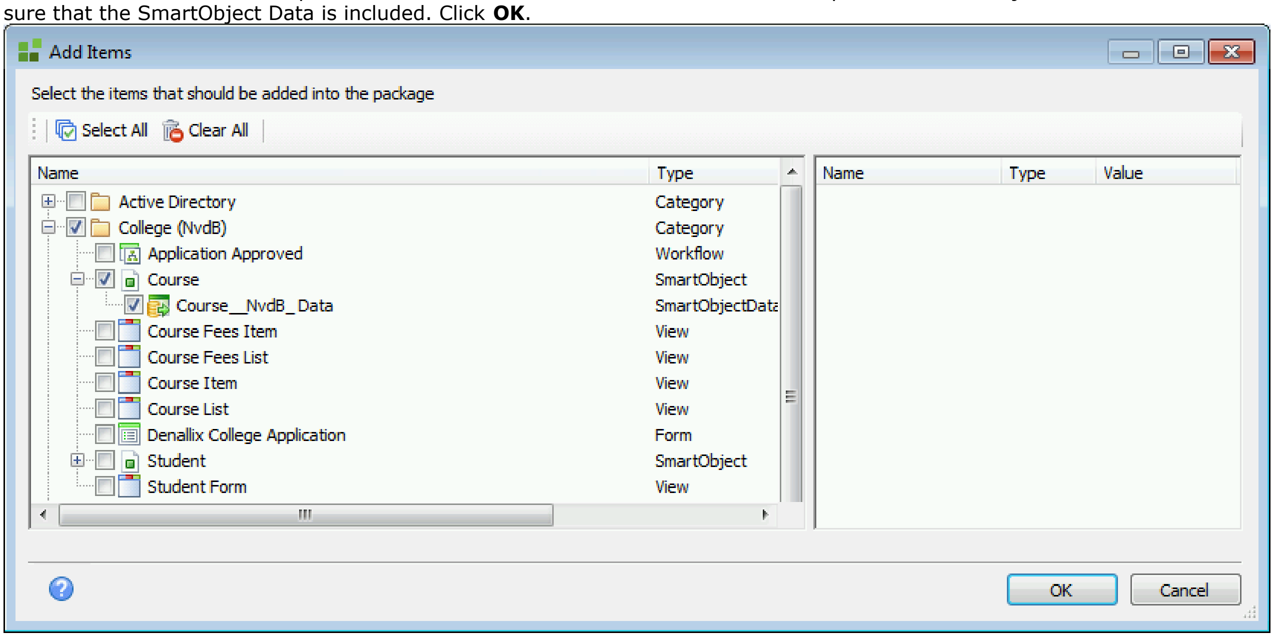
Package Name: Example2

Description: 'Example2' is a modified package that includes the 'Test Property 1' property added during this exercise.

Location: C:\Users\Administrator\Desktop

☑ Automatically include dependent items

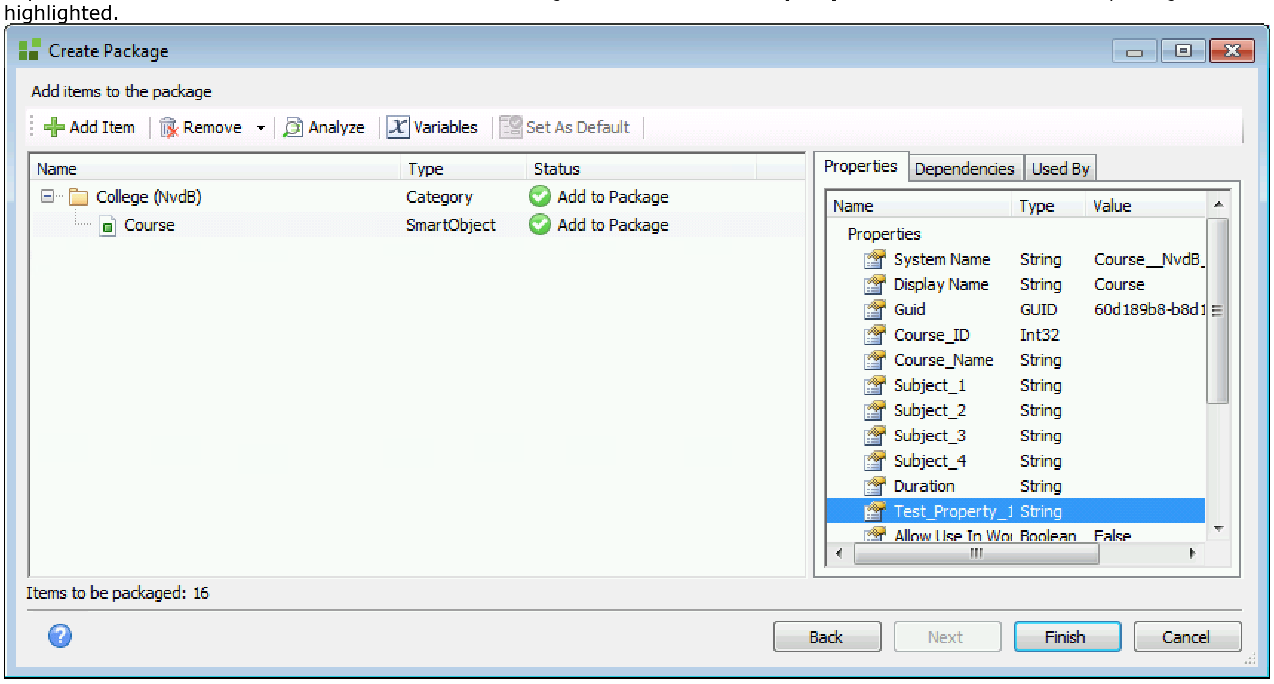
12. The **Add Items** window will open. Check the check box for the **Course** item and expand the SmartObject artifact to make sure that the SmartObject Data is included. Click **OK**.



Select the items that should be added into the package

Name	Type
☐ Active Directory	Category
☑ College (NvdB)	Category
☐ Application Approved	Workflow
☑ Course	SmartObject
☑ Course__NvdB_Data	SmartObjectData
☐ Course Fees Item	View
☐ Course Fees List	View
☐ Course Item	View
☐ Course List	View
☐ Denallix College Application	Form
☐ Student	SmartObject
☐ Student Form	View

13. The **Create Package** window will open. Click the **Add Item** button to add items to the package. Make sure that all dependent items are also selected. Note that in the image below, the **Test Property 1** which was added to the package is highlighted.



Add items to the package

Name	Type	Status
College (NvdB)	Category	Add to Package
Course	SmartObject	Add to Package

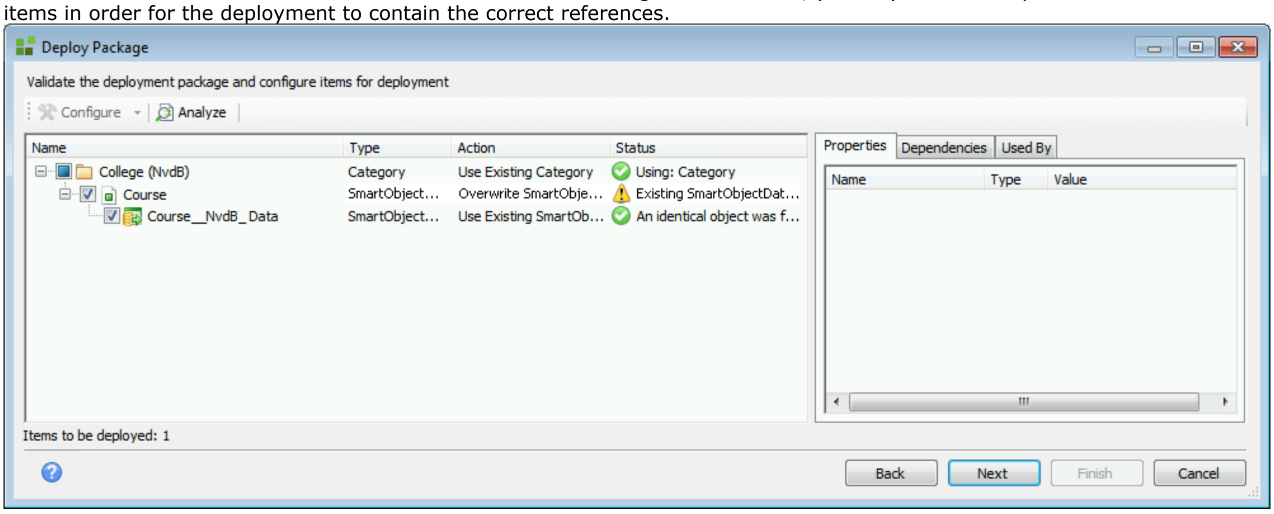
Name	Type	Value
Properties		
System Name	String	Course__NvdB_
Display Name	String	Course
Guid	GUID	60d189b8-b8d1
Course_ID	Int32	
Course_Name	String	
Subject_1	String	
Subject_2	String	
Subject_3	String	
Subject_4	String	
Duration	String	
Test_Property_1	String	
Allow Use In Wo	Boolean	False

Items to be packaged: 16

14. Click **Finish**. A new *.KSPX file will be generated for deployment.

Deploying the Package in a K2 Environment where the Artifacts Exist

1. In the Package and Deployment console, click on the relevant host server. In the **Actions** section, click **Deploy Package**. The Deploy Package window will open. As this deployment will update the K2 SmartObject definitions, the Status of each of the artifacts to be deployed will show which action must be taken for each artifact.
2. If there are differences between the source environment and target environment, you may edit the Properties of relevant items in order for the deployment to contain the correct references.



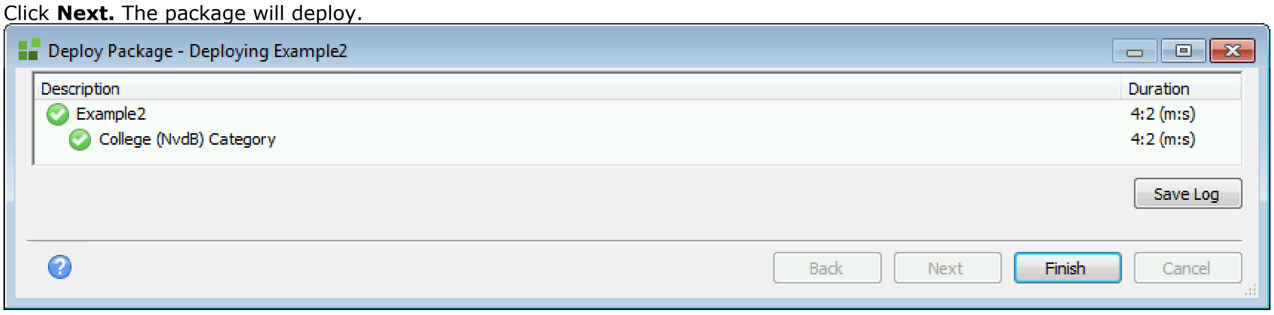
Deploy Package

Validate the deployment package and configure items for deployment

Name	Type	Action	Status
College (NvdB)	Category	Use Existing Category	Using: Category
Course	SmartObject...	Overwrite SmartObje...	Existing SmartObjectDat...
Course__NvdB_Data	SmartObject...	Use Existing SmartOb...	An identical object was f...

Items to be deployed: 1

3. Click **Next.** The package will deploy.



Deploy Package - Deploying Example2

Description	Duration
Example2	4:2 (m:s)
College (NvdB) Category	4:2 (m:s)

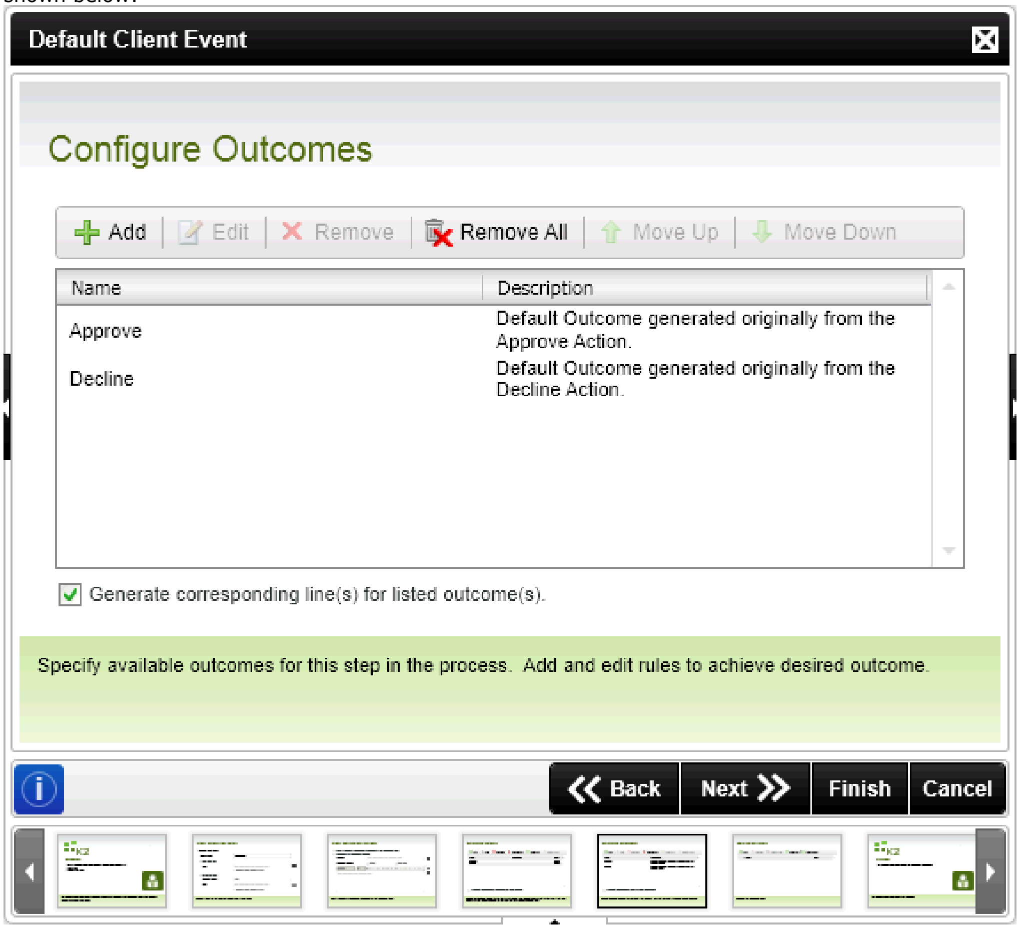
4. The K2 Package and Deployment tool will report back on the status of the deployment. As shown above, all artifacts have been successfully deployed to the new K2 environment. If you need to save a log of this action, select the **Save Log** option.
5. Click the **Finish** button to close the Deployment wizard.

1.3.7 Example 3 - Packaging a Workflow Created with K2 Studio

K2 Package and Deployment can be used to transfer a K2 Process created in K2 Studio from one environment to another. This example shows the transfer of a simple approval process created with E-mail events. This example does not show all the steps for creating the workflow process, and assumes that the user has previous experience in designing with K2 Studio.

Create a Workflow Process using K2 Studio

1. In K2 Studio, create a workflow process called **DemoProcess1**. Add a Default Client Event with two outcome actions, as shown below:

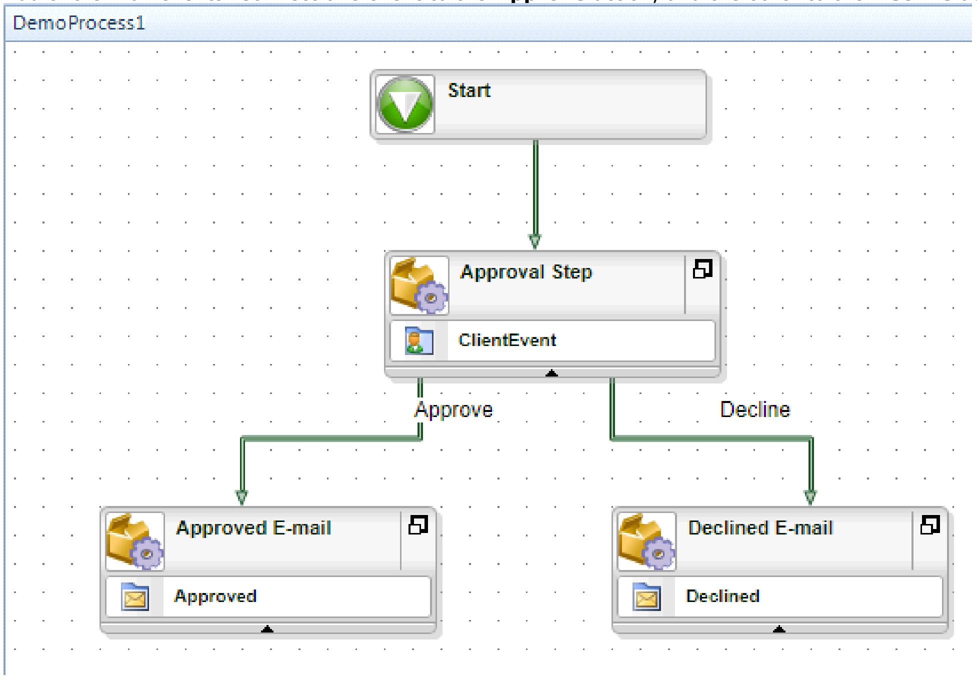


Name	Description
Approve	Default Outcome generated originally from the Approve Action.
Decline	Default Outcome generated originally from the Decline Action.

☑ Generate corresponding line(s) for listed outcome(s).

Specify available outcomes for this step in the process. Add and edit rules to achieve desired outcome.

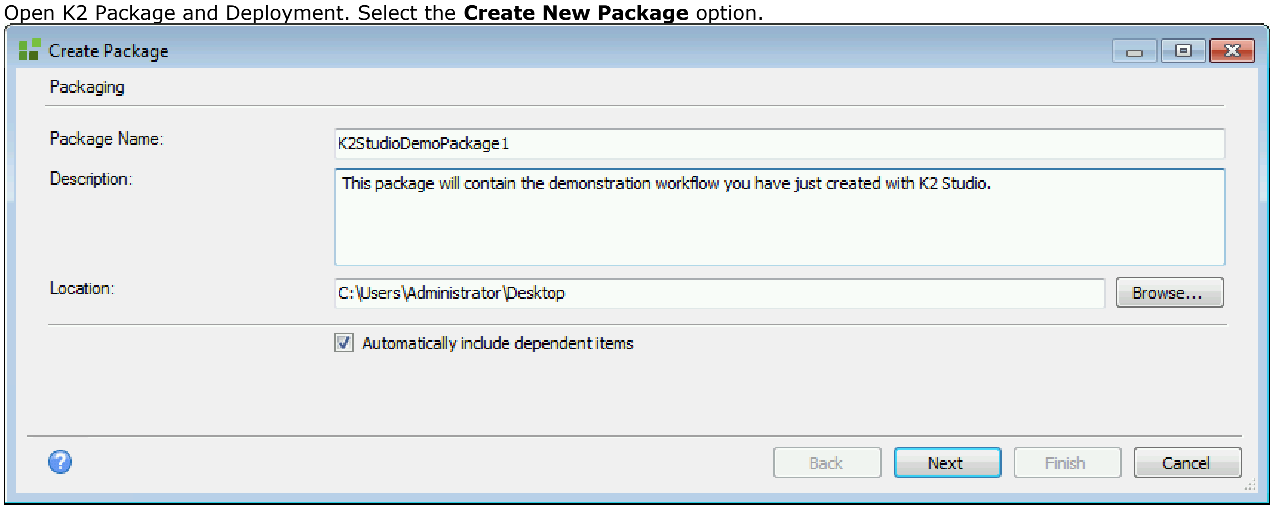
2. Add two e-mail events. Connect one event to the **Approve** action, and the other to the **Decline** action as shown below:



3. Deploy the workflow process.

Create a Package of the Workflow Process

1. Open K2 Package and Deployment. Select the **Create New Package** option.



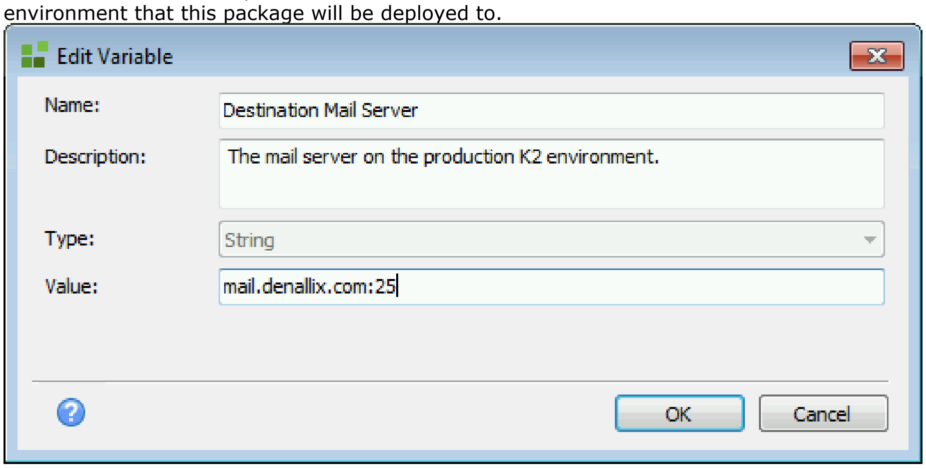
Package Name: K2StudioDemoPackage1

Description: This package will contain the demonstration workflow you have just created with K2 Studio.

Location: C:\Users\Administrator\Desktop

☑ Automatically include dependent items

2. Enter K2StudioDemoPackage1 in the **Package Name** text box.
3. Make sure that the **Automatically include dependent items** check box is checked.
4. Click **Next** to navigate to the **Create Package** window. Click the **Add Item** button to navigate to the **Add Items** screen. Check the check box for the **DemoProcess1** item.
5. Click **OK** to navigate to the **Edit Package** window.
6. In the **Edit Package** window, click on the **Mail Server** node to load the Properties.
7. In the **Properties** column, right-click on the FieldDescription with the Mail Server string value. Select the **Assign variable** option.
8. In the **Variables** screen, click on the **Add** button.
9. Enter a name and description for the variable. In the **Value** field, enter the Mail Server connection string for the K2 environment that this package will be deployed to.



Name: Destination Mail Server

Description: The mail server on the production K2 environment.

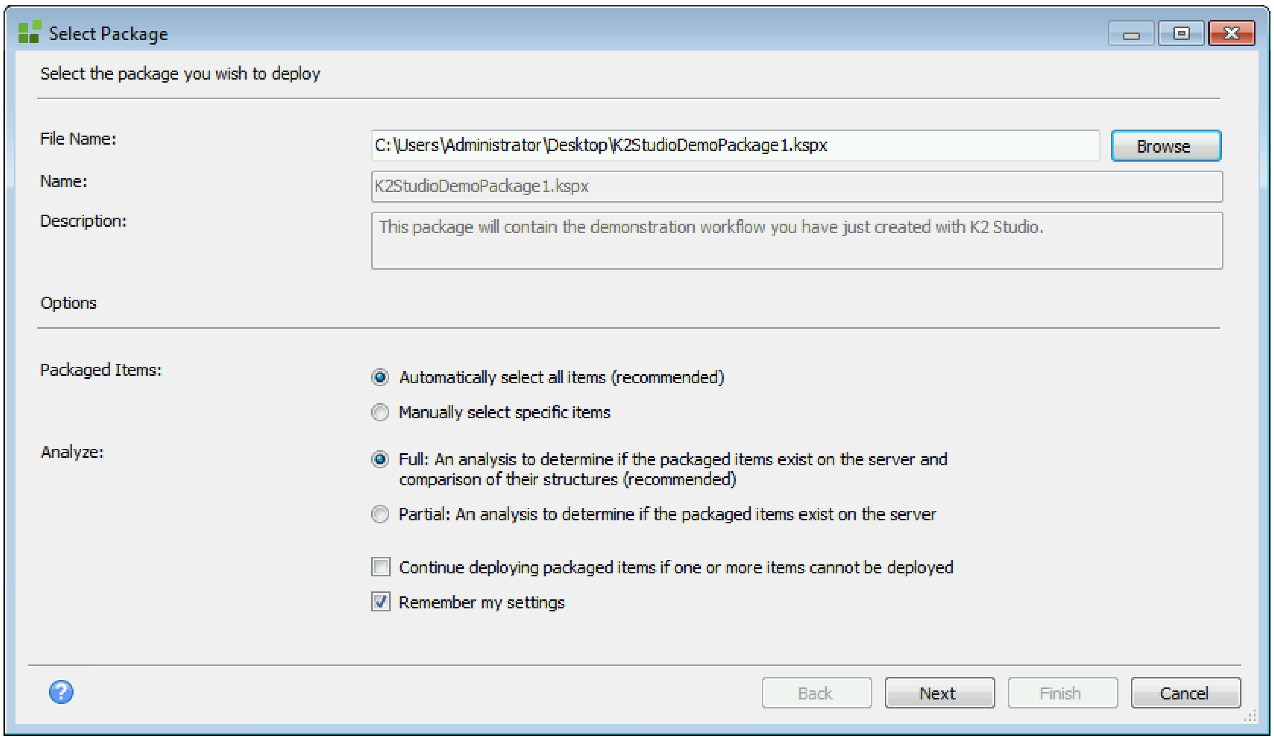
Type: String

Value: mail.denallix.com:25

10. Click **OK**.
11. Click **Finish** to create and save the package.

Deploying the K2 Process To A New Environment

1. In the target K2 environment, open K2 Package and Deployment. Select the **Deploy Package** option.
2. In the **Select Package** window, use the **Browse** button to navigate to where you saved the K2StudioDemoProcess1.kspx package created in the previous section.


Select Package

Select the package you wish to deploy

File Name:	C:\Users\Administrator\Desktop\K2StudioDemoPackage1.kspx
Name:	K2StudioDemoPackage1.kspx
Description:	This package will contain the demonstration workflow you have just created with K2 Studio.

Options

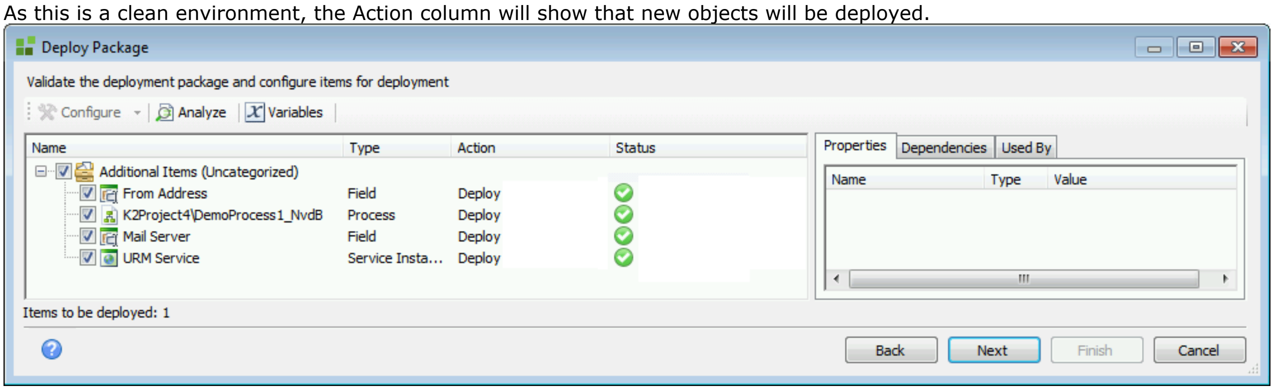
Packaged Items: [✓ Automatically select all items (recommended)] Manually select specific items

Analyze: [✓ Full: An analysis to determine if the packaged items exist on the server and comparison of their structures (recommended)] Partial: An analysis to determine if the packaged items exist on the server

☐ Continue deploying packaged items if one or more items cannot be deployed

☑ Remember my settings

3. As this is a clean environment, the Action column will show that new objects will be deployed.

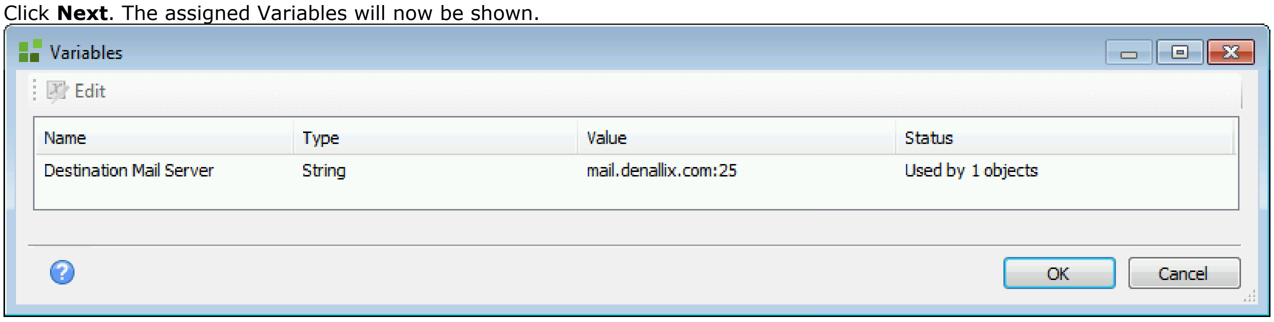

Deploy Package

Validate the deployment package and configure items for deployment

Name	Type	Action	Status
Additional Items (Uncategorized)			
From Address	Field	Deploy	✓
K2Project4\DemoProcess1_NvdB	Process	Deploy	✓
Mail Server	Field	Deploy	✓
URM Service	Service Insta...	Deploy	✓

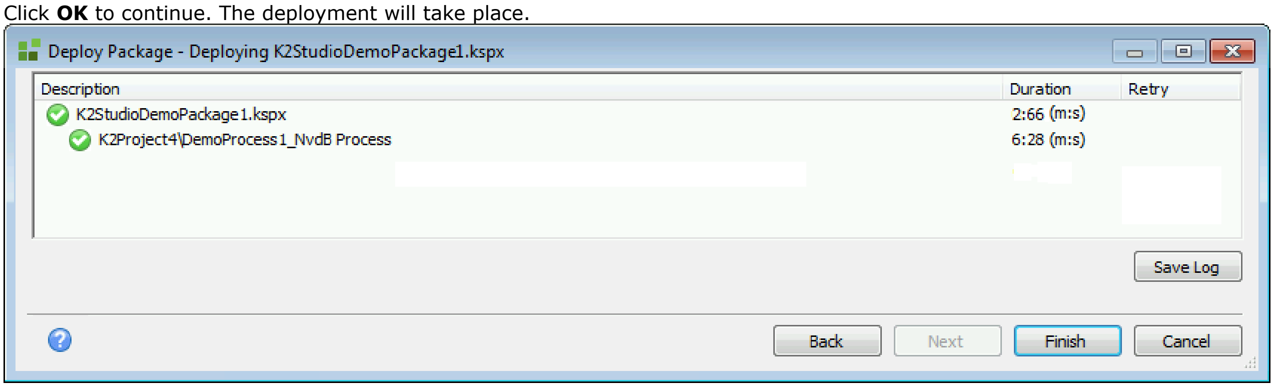
Items to be deployed: 1

4. As there are no previous definitions for this process in the new K2 environment, there are no conflicts to resolve. The **Status** of each of the artifacts to be deployed shows as Ready.
5. Click **Next**. The assigned Variables will now be shown.


Variables

Name	Type	Value	Status
Destination Mail Server	String	mail.denallix.com:25	Used by 1 objects

6. Click **OK** to continue. The deployment will take place.


Deploy Package - Deploying K2StudioDemoPackage1.kspx

Description	Duration	Retry
✓ K2StudioDemoPackage1.kspx	2:66 (m:s)	
✓ K2Project4\DemoProcess1_NvdB Process	6:28 (m:s)	

7. In this example, all the artifacts have successfully been deployed to the new K2 environment. If you need to save a log of this

action, click **Save Log**.

8. Click **Finish** to close the Deployment wizard.

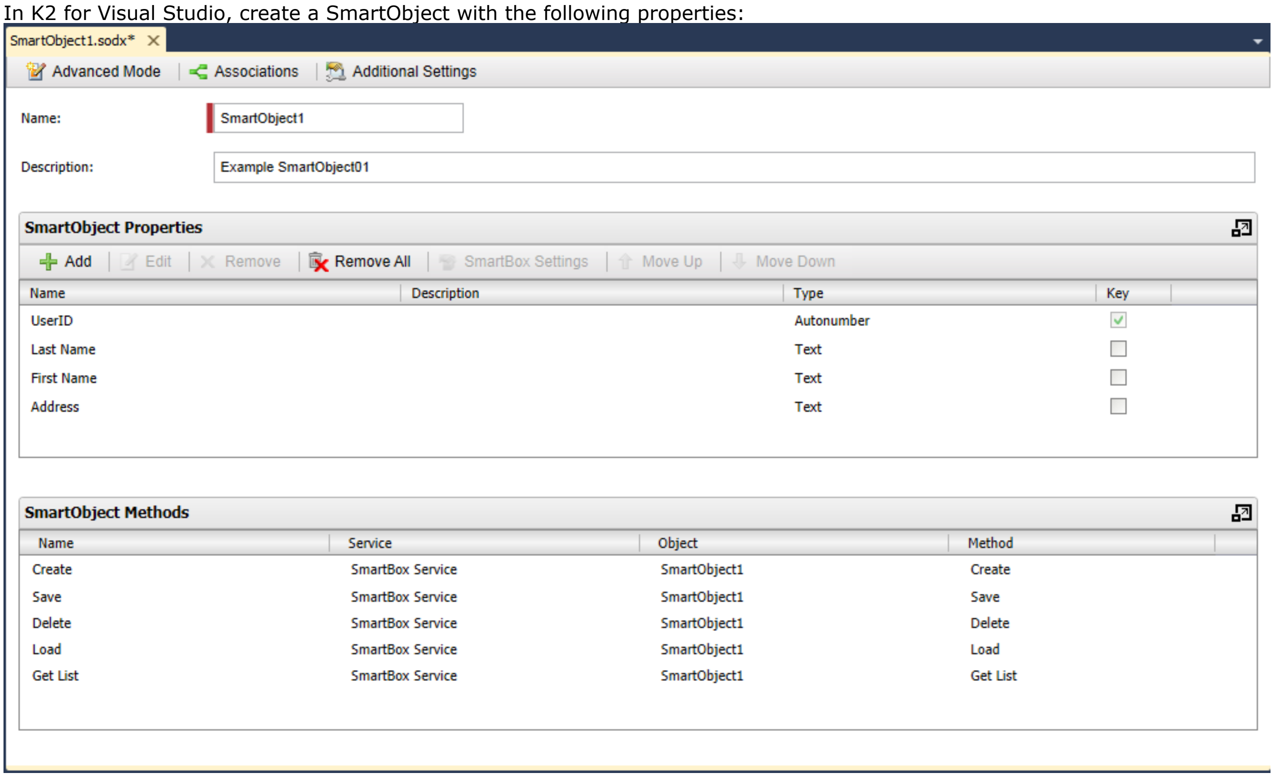
1.3.8 Example 4 - Packaging a SmartObject and Workflow created in K2 for Visual Studio

K2 Package and Deployment can be used to transfer SmartObjects and K2 Processes created using K2 for Visual Studio from one environment to another. This example shows the transfer of a simple SmartObject-integrated approval process which includes two E-mail events.

Note: This example does not show all the steps for creating the workflow process, and assumes that the user has previous experience in designing with K2 for Visual Studio.

Create a SmartObject using K2 for Visual Studio

1. In K2 for Visual Studio, create a SmartObject with the following properties:



Name: SmartObject1

Description: Example SmartObject01

SmartObject Properties

Name	Description	Type	Key
UserID		Autonumber	✓
Last Name		Text	☐
First Name		Text	☐
Address		Text	☐

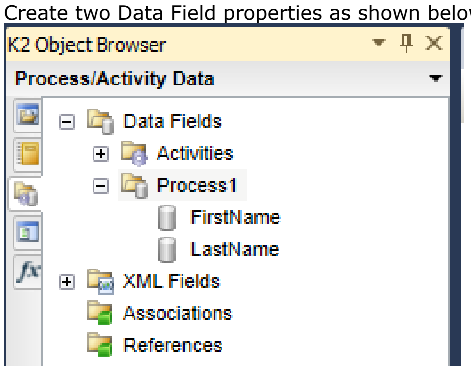
SmartObject Methods

Name	Service	Object	Method
Create	SmartBox Service	SmartObject1	Create
Save	SmartBox Service	SmartObject1	Save
Delete	SmartBox Service	SmartObject1	Delete
Load	SmartBox Service	SmartObject1	Load
Get List	SmartBox Service	SmartObject1	Get List

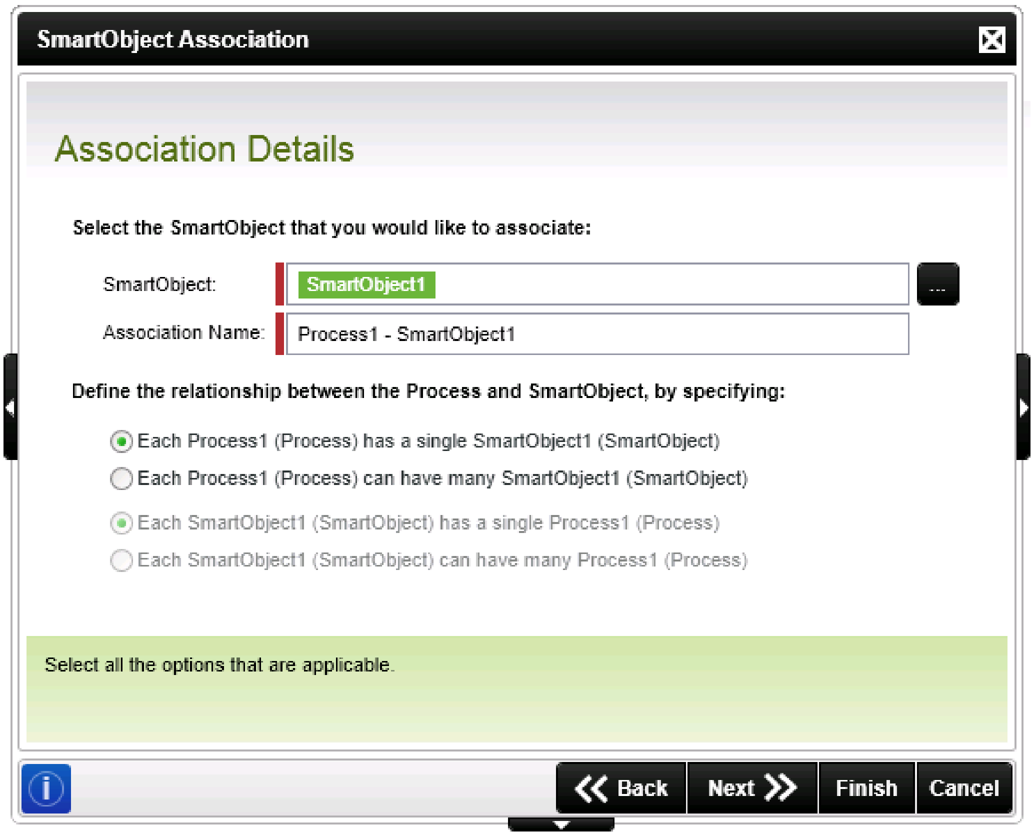
2. Deploy the SmartObject to the K2 Server.

Create a Workflow Process using K2 for Visual Studio

1. In K2 for Visual Studio, create a workflow process called **DemoProcess1**.
2. Create two Data Field properties as shown below:



3. Associate the workflow Data Fields with the SmartObject created in the previous section, with the associations configured as shown below:



Association Details

Select the SmartObject that you would like to associate:

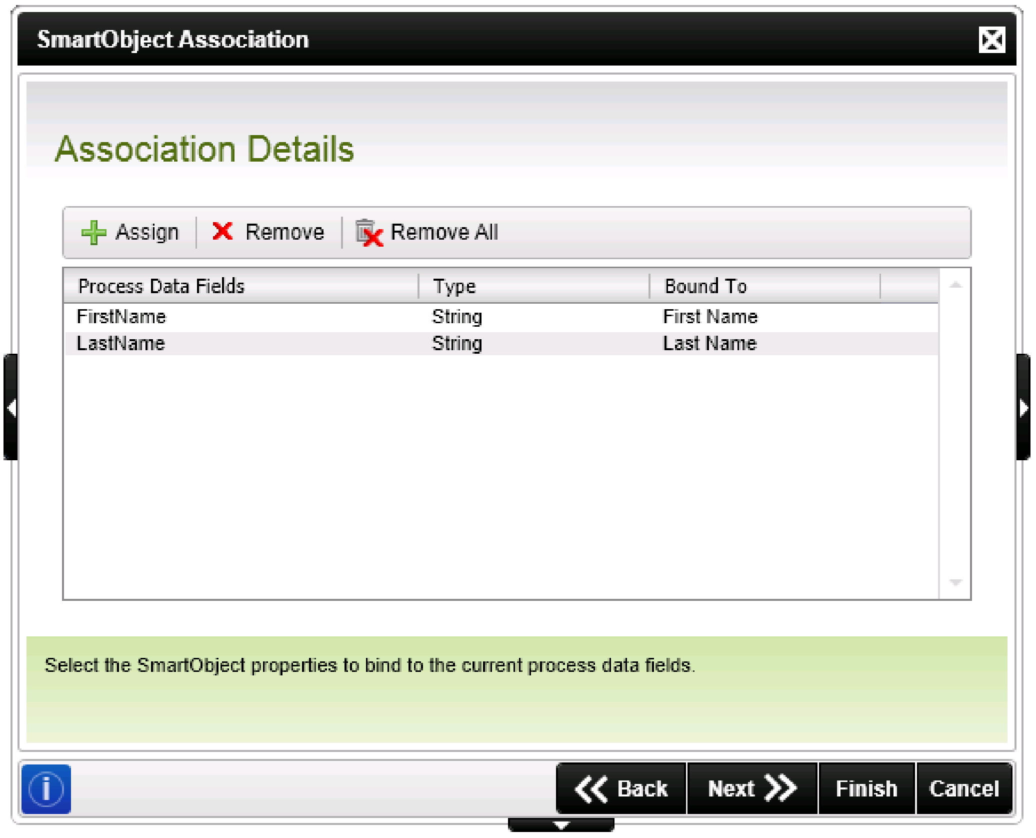
SmartObject: SmartObject1

Association Name: Process1 - SmartObject1

Define the relationship between the Process and SmartObject, by specifying:

- [✓ Each Process1 (Process) has a single SmartObject1 (SmartObject)]
- Each Process1 (Process) can have many SmartObject1 (SmartObject)
- [✓ Each SmartObject1 (SmartObject) has a single Process1 (Process)]
- Each SmartObject1 (SmartObject) can have many Process1 (Process)

Select all the options that are applicable.

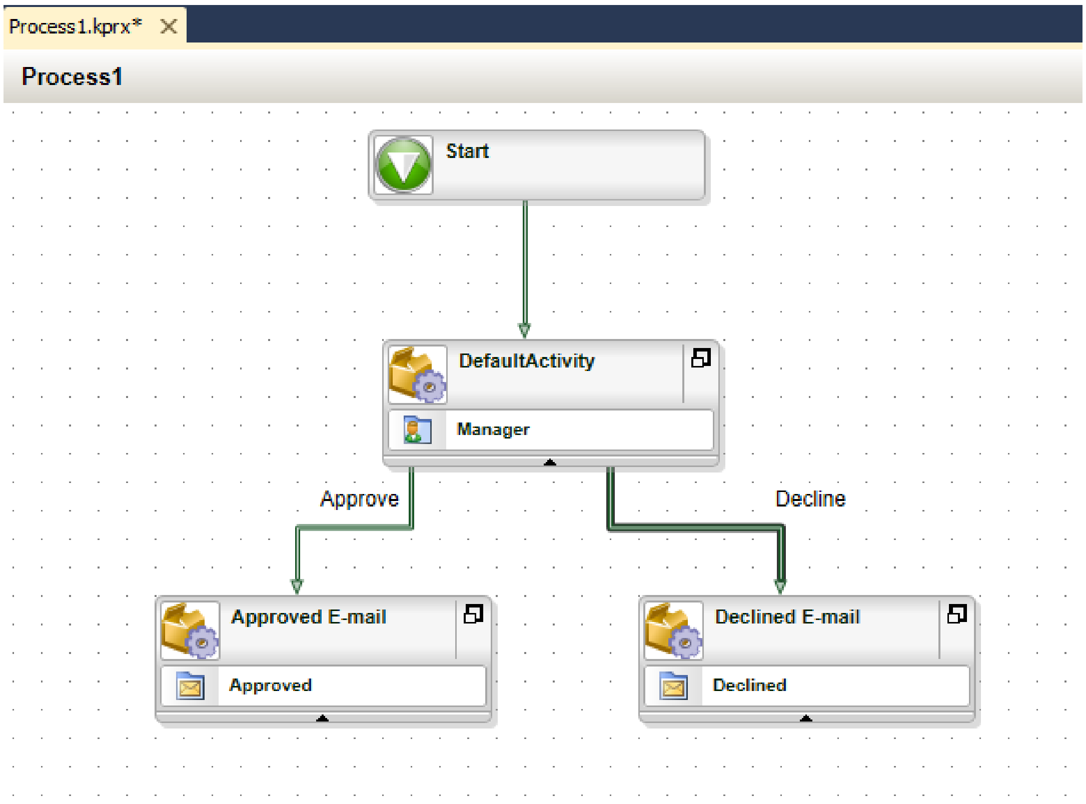


Association Details

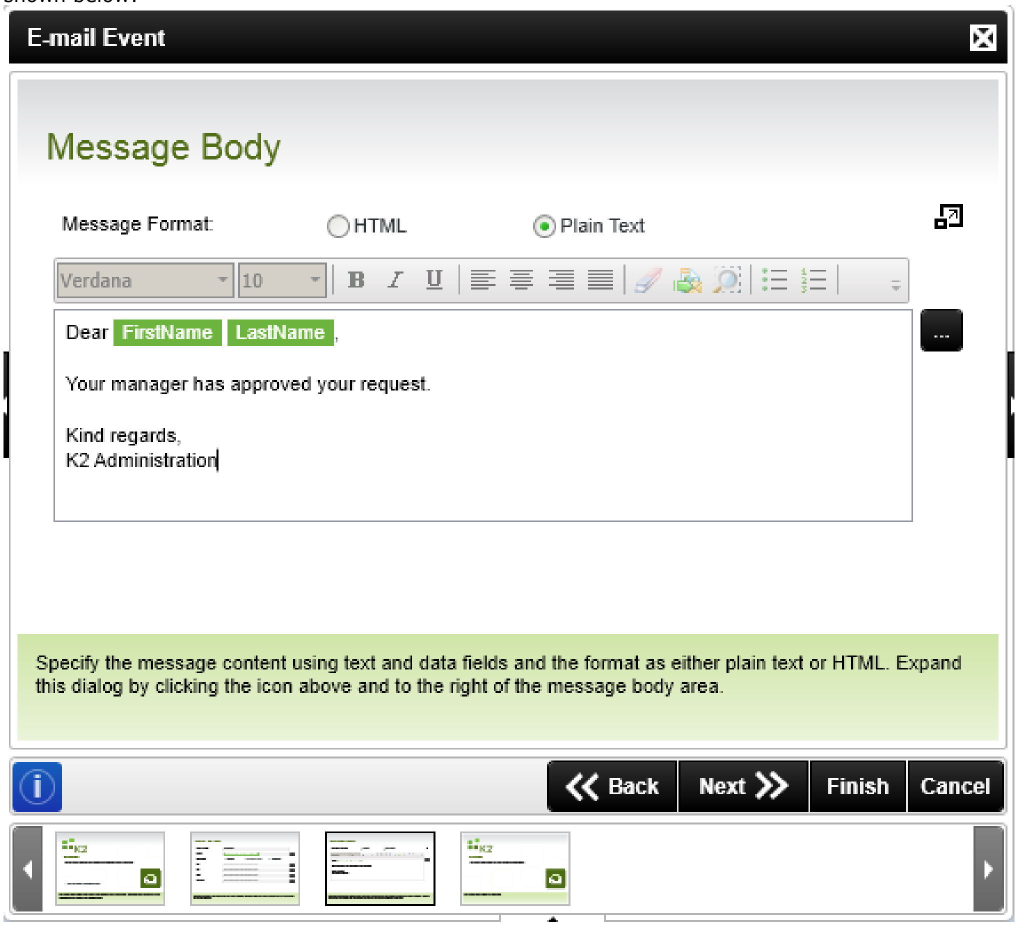
Process Data Fields	Type	Bound To
FirstName	String	First Name
LastName	String	Last Name

Select the SmartObject properties to bind to the current process data fields.

4. Now add a Default Client Event to the process, with two outcome actions:



5. Create E-mail events as the outcomes for the actions, using the integrated SmartObject data within the E-mail messages as shown below:



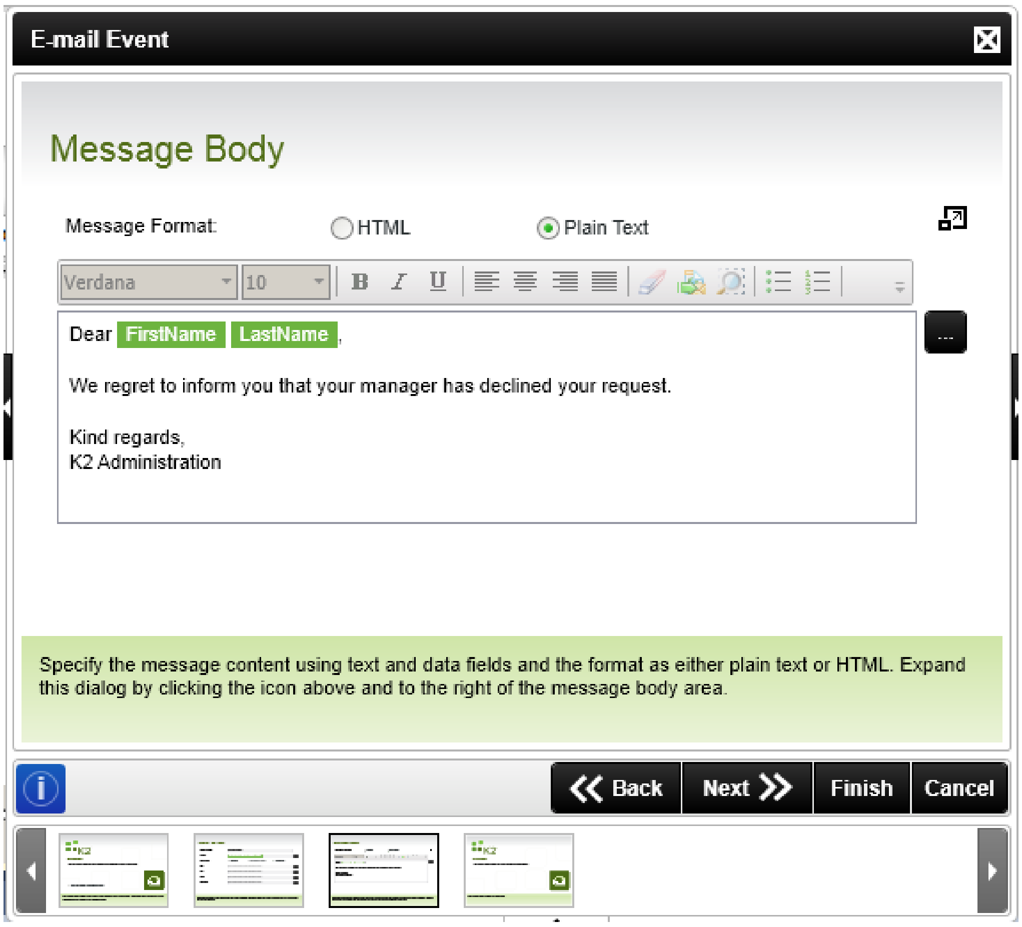
Message Format: HTML [✓ Plain Text]

Dear FirstName LastName,

Your manager has approved your request.

Kind regards,
K2 Administration

Specify the message content using text and data fields and the format as either plain text or HTML. Expand this dialog by clicking the icon above and to the right of the message body area.



Message Format: ( ) HTML [✓ Plain Text]

Dear FirstName LastName,

We regret to inform you that your manager has declined your request.

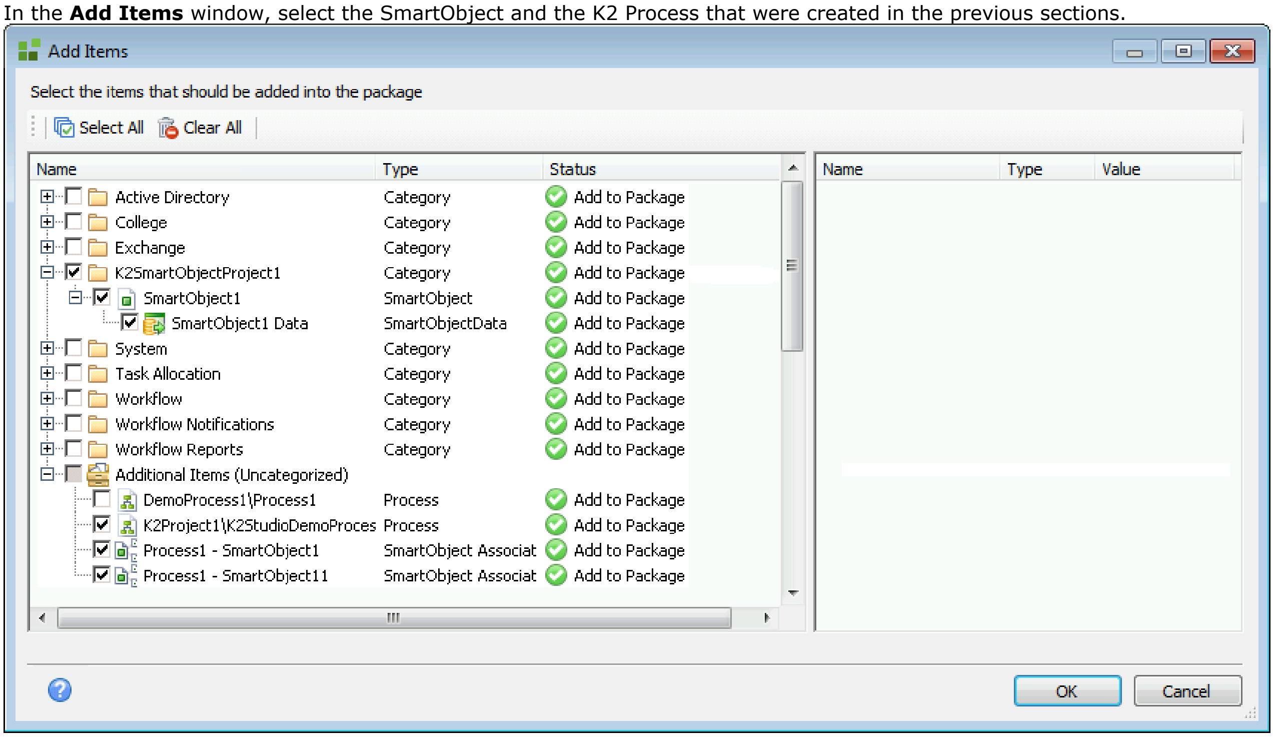
Kind regards,
K2 Administration

Specify the message content using text and data fields and the format as either plain text or HTML. Expand this dialog by clicking the icon above and to the right of the message body area.

6. Deploy the workflow process to the K2 Server.

Create a Package of the SmartObject and Workflow Process

1. Open K2 Package and Deployment.
2. Select the **Create New Package** option.
3. Name the new package **K24VSDemoProcessPackage**. Click **Next**.
4. Click the **Add Item** button to navigate to the **Add Items** window.
5. In the **Add Items** window, select the SmartObject and the K2 Process that were created in the previous sections.



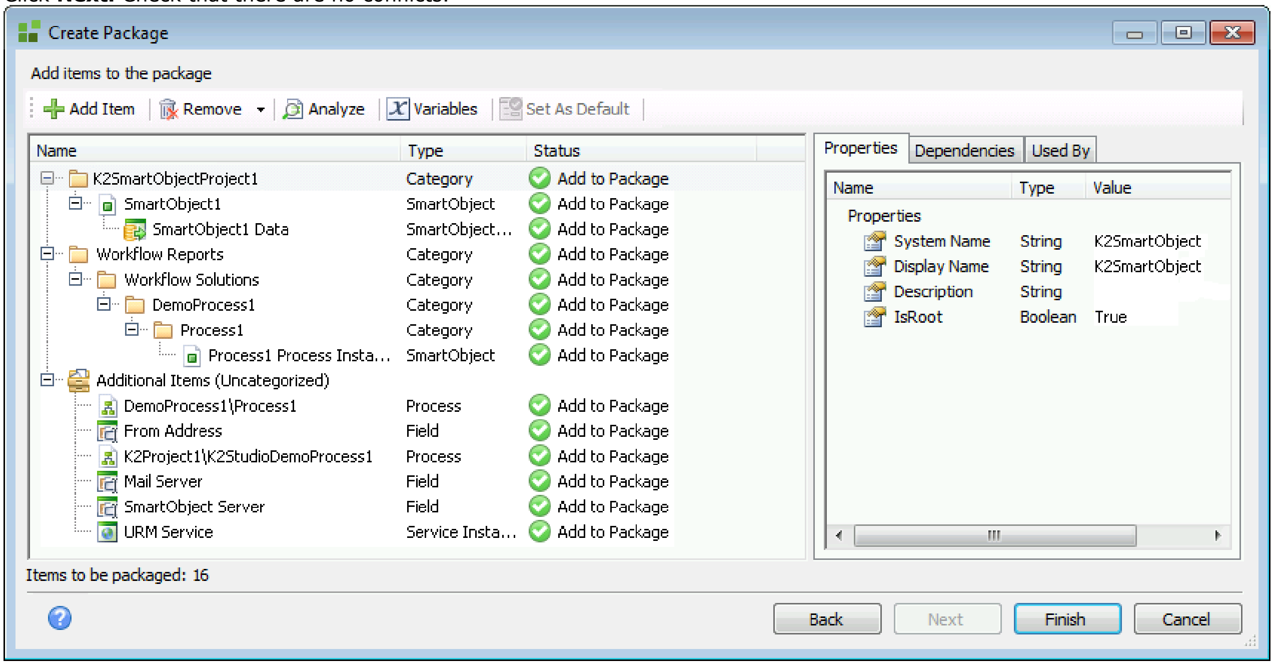
Select the items that should be added into the package

Name	Type	Status
[ ] Active Directory	Category	Add to Package
[ ] College	Category	Add to Package
[ ] Exchange	Category	Add to Package
[✓] K2SmartObjectProject1	Category	Add to Package
[✓] SmartObject1	SmartObject	Add to Package
[✓] SmartObject1 Data	SmartObjectData	Add to Package
[ ] System	Category	Add to Package
[ ] Task Allocation	Category	Add to Package
[ ] Workflow	Category	Add to Package
[ ] Workflow Notifications	Category	Add to Package
[ ] Workflow Reports	Category	Add to Package
Additional Items (Uncategorized)		
[ ] DemoProcess1\Process1	Process	Add to Package
[✓] K2Project1\K2StudioDemoProces	Process	Add to Package
[✓] Process1 - SmartObject1	SmartObject Associat	Add to Package
[✓] Process1 - SmartObject11	SmartObject Associat	Add to Package

Name	Type	Value

By default, SmartObjectData artifacts are deselected so that SmartObject data from one environment is not automatically deployed to a new environment. This is the default behavior in order to prevent deployment packages from becoming oversized, due to the potentially large amount of data stored within a SmartObject. If you must package the data contained within a SmartObject, you will need to open the SmartObject node and manually select the relevant SmartObject Data. See K2 Package and Deployment Overview.

6. Click **OK.**
7. Click **Next.** Check that there are no conflicts.



Name	Type	Status
K2SmartObjectProject1	Category	Add to Package
SmartObject1	SmartObject	Add to Package
SmartObject1 Data	SmartObject...	Add to Package
Workflow Reports	Category	Add to Package
Workflow Solutions	Category	Add to Package
DemoProcess1	Category	Add to Package
Process1	Category	Add to Package
Process1 Process Insta...	SmartObject	Add to Package
Additional Items (Uncategorized)		
DemoProcess1\Process1	Process	Add to Package
From Address	Field	Add to Package
K2Project1\K2StudioDemoProcess1	Process	Add to Package
Mail Server	Field	Add to Package
SmartObject Server	Field	Add to Package
URM Service	Service Insta...	Add to Package

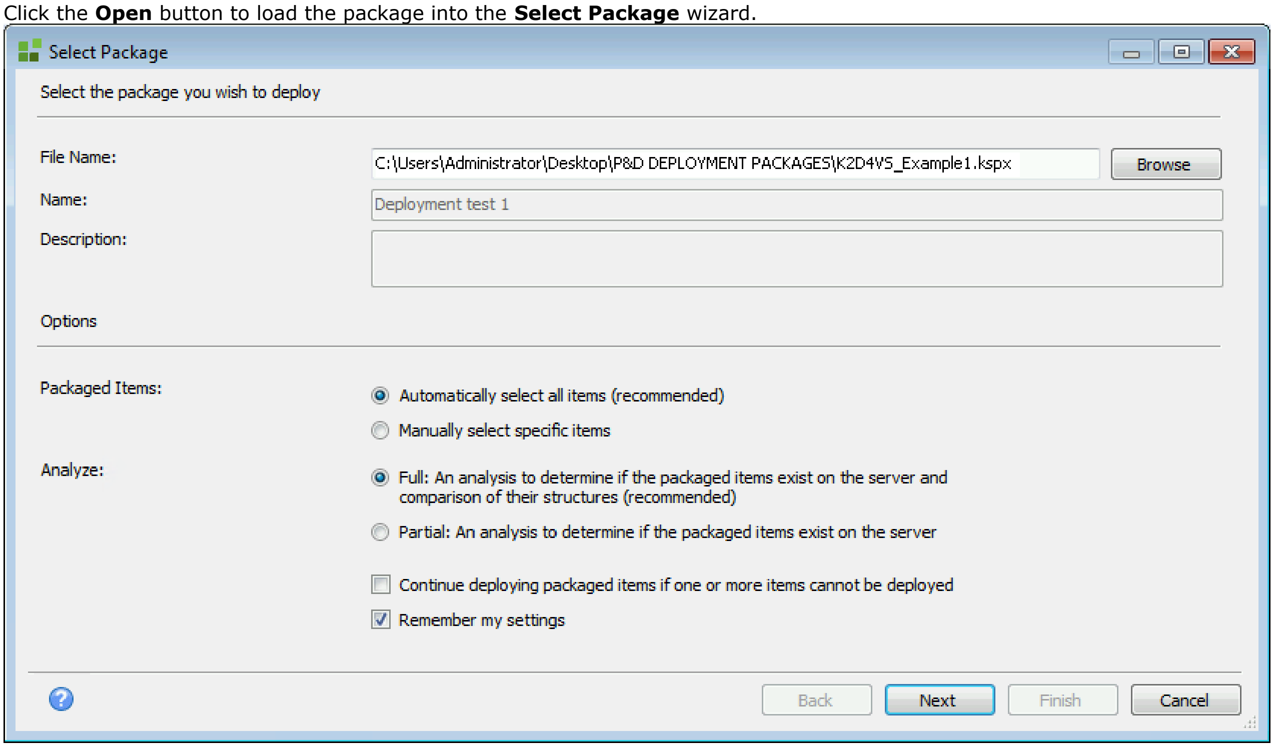
Name	Type	Value
Properties		
System Name	String	K2SmartObject
Display Name	String	K2SmartObject
Description	String	
IsRoot	Boolean	True

Items to be packaged: 16

8. Click **Finish** to create and save the package.

Deploying the Package to a New Environment

1. In the target K2 environment, open the K2 Package and Deployment console.
2. Select the **Deploy Package** option. Navigate to where you saved the **K24VSDemoProcessPackage.kspx** package created in the previous section.
3. Click the **Open** button to load the package into the **Select Package** wizard.



File Name: C:\Users\Administrator\Desktop\P&D DEPLOYMENT PACKAGES\K2D4VS_Example1.kspx

Name: Deployment test 1

Description:

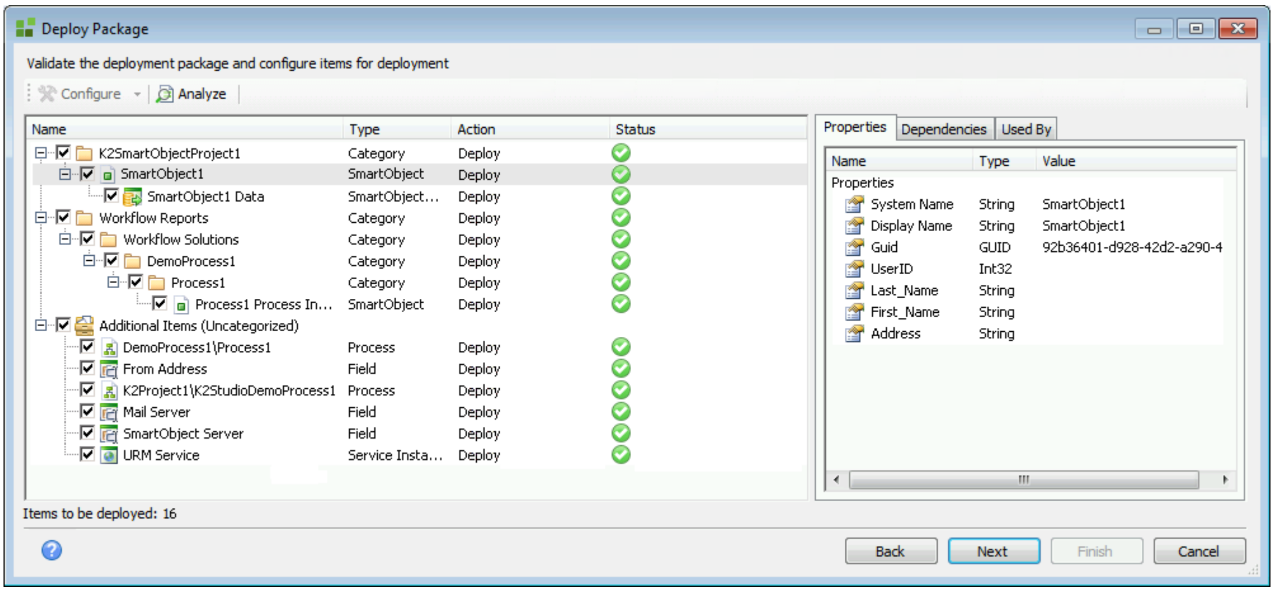
Packaged Items: [✓ Automatically select all items (recommended)] Manually select specific items

Analyze: [✓ Full: An analysis to determine if the packaged items exist on the server and comparison of their structures (recommended)] Partial: An analysis to determine if the packaged items exist on the server

☐ Continue deploying packaged items if one or more items cannot be deployed

☑ Remember my settings

4. As this is a clean environment, the **Actions** will default to creating new artifacts.



Validate the deployment package and configure items for deployment

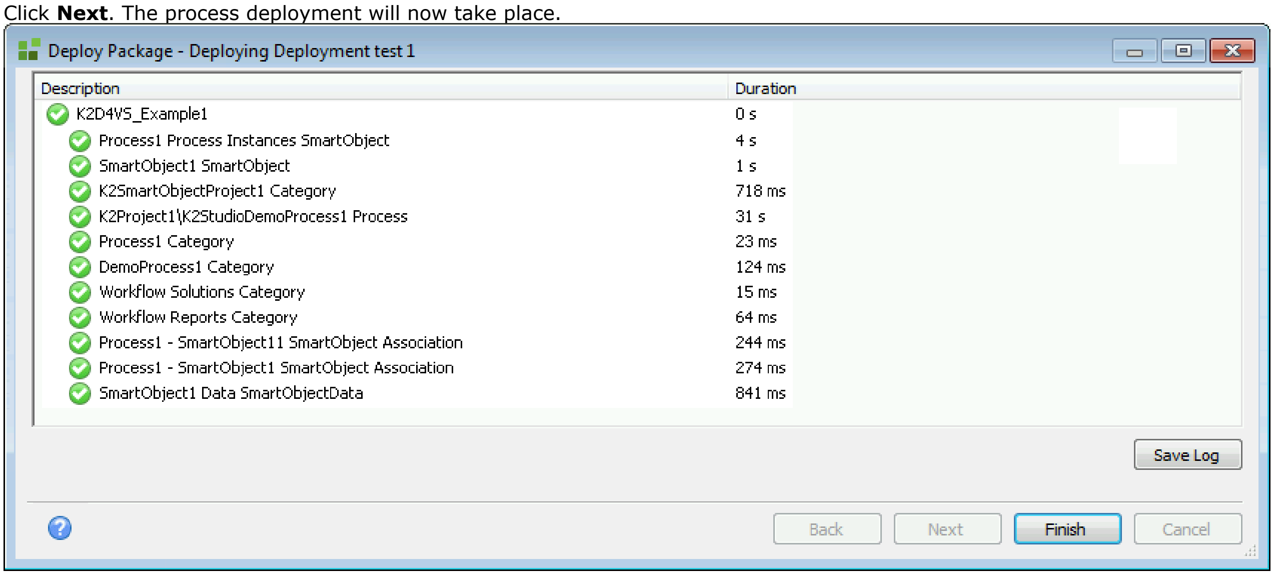
Name	Type	Action	Status
K2SmartObjectProject1	Category	Deploy	✓
SmartObject1	SmartObject	Deploy	✓
SmartObject1 Data	SmartObject...	Deploy	✓
Workflow Reports	Category	Deploy	✓
Workflow Solutions	Category	Deploy	✓
DemoProcess1	Category	Deploy	✓
Process1	Category	Deploy	✓
Process1 Process In...	SmartObject	Deploy	✓
Additional Items (Uncategorized)			
DemoProcess1\Process1	Process	Deploy	✓
From Address	Field	Deploy	✓
K2Project1\K2StudioDemoProcess1	Process	Deploy	✓
Mail Server	Field	Deploy	✓
SmartObject Server	Field	Deploy	✓
URM Service	Service Insta...	Deploy	✓

Properties | Dependencies | Used By

Name	Type	Value
Properties		
System Name	String	SmartObject1
Display Name	String	SmartObject1
Guid	GUID	92b36401-d928-42d2-a290-4
UserID	Int32	
Last_Name	String	
First_Name	String	
Address	String	

Items to be deployed: 16

5. As there are no previous definitions for this process in the new K2 environment, there are no conflicts to resolve, and the **Status** of each of the artifacts to be deployed shows as Ready.
6. Click **Next**. The process deployment will now take place.



Description	Duration
K2D4VS_Example1	0 s
Process1 Process Instances SmartObject	4 s
SmartObject1 SmartObject	1 s
K2SmartObjectProject1 Category	718 ms
K2Project1\K2StudioDemoProcess1 Process	31 s
Process1 Category	23 ms
DemoProcess1 Category	124 ms
Workflow Solutions Category	15 ms
Workflow Reports Category	64 ms
Process1 - SmartObject11 SmartObject Association	244 ms
Process1 - SmartObject1 SmartObject Association	274 ms
SmartObject1 Data SmartObjectData	841 ms

7. All the artifacts have been successfully deployed to the new K2 environment. If you need to save a log of this action, click **Save Log**.
8. Click the **Finish** button to close the Deployment wizard

1.4 Copyright