

**Special Issue**

**1**

UNDERSTANDING SHAREPOINT JOURNAL

---

Bjørn Christoffer Thorsmæhlum Furuknap

# Using Nintex Workflow 2007

---

UNDERSTANDING SHAREPOINT JOURNAL

# Using Nintex Workflow 2007

---

*This book is dedicated to my wife.*

© Understanding SharePoint  
Rubina Ranasgt. 10  
N-0190 Oslo, Norway  
Phone +47 91 39 85 86 • Web <http://www.understandingsharepoint.com/>

---

# Credits

## About the Author



Bjørn Christoffer Thorsmæhlum Furuknap is a senior solutions architect, published author of *Building the SharePoint User Experience*, speaker, and passionate SharePointaholic. He has been doing software development professionally for 16 years for small companies as well as multinational corporations. He has also been a teacher at a college-level school, teaching programming and development to aspiring students, a job that inspired him to begin teaching what he has learned and learns every day.

## About Understanding SharePoint Journal

*Understanding SharePoint Journal* is a periodical published by UnderstandingSharePoint.com. The journal covers few topics in each issue, focusing to teach a deeper understanding of each topic while showing how to use SharePoint in real-life scenarios.

You can read more about *USP Journal*, as well as get other issues and sign up for regular updates, discounts, and previews of upcoming issues, at <http://www.understandingsharepoint.com/journal>.

## Other Credits

A great big thanks to Kim Wimpsett for doing the copyedit. The quality of work in this issue is greatly attributed to her skill.

---





# Table of Contents

Credits.....	i
About the Author .....	i
About Understanding SharePoint Journal .....	i
Other Credits .....	i
Table of Contents.....	i
Exercises .....	2
Legal Disclaimer .....	4
Introduction .....	5
A Message from Nintex.....	6
What Is Nintex Workflow 2007? .....	1
Logic.....	4
Loops .....	4
Settings and Configuration Scope .....	6
State Machines and Sequential Workflows .....	7
What's Up Next? .....	9
What You Will Need .....	9
Setup and Administration .....	12
Installing Nintex Workflow 2007.....	12
Prerequisites .....	13
Installation and Deployment .....	13
Central Administration Setup.....	15
Features and Activation .....	17
Optional Workflow Settings .....	22
Allowed Actions .....	22
LazyApproval .....	23
Holidays .....	26
Workflow Constants .....	27
Message Templates .....	30
Global Settings .....	31
Enforce Safe Looping .....	32
Enforce Allowed Actions at Run Time .....	32
Custom Workflow Start Page .....	32
Getting to Know the Workflow Designer.....	35
Your First Nintex Workflow.....	35
Configuring Actions .....	40



















---

Workflow Lookups in Nintex Workflow.....	42
Playing Around with Values .....	44
Saving and Publishing Your Workflow .....	48
Snippets and Templates .....	49
Workflow Variables.....	52
Variables vs. Constants .....	52
Defining New Variables .....	52
Sample Tasks .....	56
Approval .....	56
Using Start Data Variables.....	61
Automatically Delegating a Task .....	65
Following the Execution Path of a Workflow .....	70
Reporting .....	74
Working with External Data.....	77
Using Web Services from a Workflow .....	77
Using Stored Credentials for Web Service Authentication .....	78
Regular Expression Boogie .....	81
Querying XML in a Workflow.....	83
Get Well Soon!.....	87
Task Overview .....	87
Creating the “Get Well” Card Workflow .....	88
Collecting Data from Multiple Users.....	92
Building a Dynamic String of Greetings .....	97
Approving and Sending the Greetings .....	98
State Machines .....	104
State Machine Workflows .....	104
Building a State Machine.....	105
Advancing the State Machine.....	109
Final Thoughts and Additional Resources.....	116
<i>USP Journal</i> Affiliate Program .....	117

## Exercises

 Installing Nintex Workflow .....	13
 Configuring Nintex Workflow .....	16
 Activating and Testing Nintex Workflow on a Site.....	17
 Configuring Allowed Actions .....	22

---

	Setting Up LazyApproval .....	24
	Creating Your First Nintex Workflow .....	35
	Playing with Values .....	44
	Creating a Workflow Snippet .....	49
	Creating a Simple Approval Workflow .....	57
	Testing the Workflow .....	58
	Assigning a Dynamic Approver .....	61
	Working with Branches .....	64
	Calling a Web Service from a Workflow .....	77
	Removing Default Namespace from XML .....	82
	Querying XML in a Workflow .....	83
	Creating a List of Sick Employees .....	88
	Creating a Collection of Users .....	88
	Assigning Tasks to Multiple Users .....	92
	Building a Dynamic String .....	97
	Requesting Approval and Nagging About It .....	99
	Adding a State Machine Action .....	106
	Restarting a Collect Data Task .....	109

---

## Legal Disclaimer

Applicable copyright law protects this work. You may not redistribute this work in any form without the prior written approval of *Understanding SharePoint Journal*. You should never use the code in *Understanding SharePoint Journal* in a production environment. If you do not abide by this disclaimer, your peanut-butter sandwich will fall face down on the floor.

---

# Introduction

*Asking the right question, at the right time, to the right people, can trigger a landslide.*

Welcome to this very special issue of *Understanding SharePoint Journal*. This issue is special in several ways, and I will explain why in a moment. First, however, I would like to introduce new readers to what this journal is all about.

*Understanding SharePoint Journal* started after a reader of my blog sent me a simple question that required a complex answer. The question was simply, “How can I know how many users are currently on my site?” The answer was so complex that I practically had to write a book to explain not just why the answer was complex but also how to solve the problem.

In the end, I developed a complete solution and posted it online, while I put the details of how to develop the solution into the first issue of *USP Journal*. And that’s how this journal was born.

As I said, the issue you are reading now is special. For starters, this is the first issue that is completely free of charge.

The reason for this is also the second reason why this issue is special. For the first time, I have a sponsor to help pay for the issue and ensure that you get to learn something useful, free of charge. Nintex has been kind enough to provide both financial and technical support to make sure that this issue is now in your hand—or, as it may be, on your computer.

Now, you may think that having a sponsor would interfere with the editorial integrity of the journal. Rest assured that the complete and sole editorial responsibility resides safely in my hands, and Nintex had no editorial control. What I write here, except for the message from Nintex on the next page, are my own words and my own opinions.

I asked the Nintex folks to tell me what issues their users most often face and which tasks users most often perform; I also asked them several technical questions to resolve issues I have faced. Other than that, the regular *USP Journal* staffers are the only people to touch this issue.

That said, in this issue I will introduce you to Nintex Workflow, which offers workflow tools that enable an improved workflow design and management experience in SharePoint. Nintex Workflow has several nice features that make it a popular tool among nonprogrammers, especially those who find the limits of the built-in workflows or SharePoint Designer workflows to be an issue.

---



# A Message from Nintex

*Dear reader,*

Nintex welcomed the opportunity to sponsor this issue of *Understanding SharePoint Journal* to provide a training journal to teach people how to get the most out of Nintex Workflow 2007.

To begin, we'll give you some product background. Released in April 2007, Nintex Workflow 2007 is Nintex's second-generation SharePoint workflow product. Nintex Workflow 2007 extends Microsoft SharePoint 2007 technologies including Microsoft Office SharePoint Server (MOSS) 2007 and Microsoft Windows SharePoint Services (WSS) 3.0.

Nintex Workflow 2007 provides advanced workflow capabilities via a graphical web-based interface embedded within SharePoint. Nintex Workflow 2007 empowers users across the organization to automate business processes, review workflow activities, and automate common SharePoint administrative tasks.

Nintex Workflow 2007 is available in a number of languages including English, German, Spanish, Hungarian, Czech, Italian, and Latvian, with more language translations in progress.

This issue of the independently produced training journal *Understanding SharePoint Journal* will be a valuable resource for those wanting to familiarize themselves with the product functionality of Nintex Workflow 2007. We hope this journal's issue inspires you to investigate Nintex Workflow 2007 further. Other publicly available Nintex Workflow 2007 resources include a 30-day free product trial (via download or hosted site), demonstration and scenario videos, workflow tutorials, and the Nintex Workflow 2007 product solution showcase, all of which are available at [www.nintex.com/workflow](http://www.nintex.com/workflow).

We encourage readers of this journal to join Nintex Workflow customers, partners, and users on Nintex Connect (<http://connect.nintex.com>), where you can ask product questions, share answers, discuss technical issues, and gain access to the product knowledge base, learning materials, and white papers.

We hope you enjoy the journal.

The Nintex team

---

## What Is Nintex Workflow 2007?

*Introducing Nintex Workflow...*

---

### ICON KEY

---

⌘ Valuable information

---

✎ Test your knowledge

---

📖 Exercise

---

☠ Caution

---

Nintex Workflow 2007 is a product made by a company called Nintex. Now, if you have worked with business process management at all, you know the power that workflows bring to the table. However, as you likely also know, authoring these workflows can be tedious.

### Note

If you have no idea what a workflow is, you may be in the wrong issue of Understanding SharePoint Journal. In issue 4 on SharePoint Designer workflows, you will learn more about what a workflow is. Luckily, the free preview chapter of that issue contains the definition of a workflow; you can get the preview by going to <http://www.understandingsharepoint.com/journal/volume-1/issue-4>.

True, you have tools, such as SharePoint Designer 2007, that do a decent job of authoring simple workflows. Not just that, but by using tricks and techniques, you can even create fairly advanced SharePoint Designer workflows, even to the point where you are performing tasks that Microsoft never intended for SharePoint Designer workflows at all.

But here is the catch: if you want anything but the simplest of static workflows, you need to perform tricks to make SharePoint Designer do tasks for which it is not very well suited. This means fighting with the tool more than focusing on the logic, and at best you will have a maintenance nightmare. The long-term survivability of such complex SharePoint Designer solutions leaves much to be desired.

If you want to go the best-practice route, you need to fire up Visual Studio and, at the same time, open a can of worms that can eat you out of your sanity within minutes.

Of course, you could hire someone who has had their proper mental training and is capable of whipping up Visual Studio workflows as easily as other people take deep breaths, but these developers tend to be expensive, hard to find, and never available or interested when all you need is to make a small change or ensure that an item is following a very simple approval workflow.

This is where Nintex Workflow 2007 comes in and, if not saves the day, at least makes the day much more comfortable. Nintex Workflow bridges the gap between the feature-limited SharePoint Designer–type workflows and the complex and sanity-breaking Visual Studio workflows.

Throughout this issue, I will introduce you to Nintex Workflow 2007 and show you some example scenarios where Nintex Workflow 2007 will save you tons of time and effort.

**Note**

Oh, and since I'm already tired of writing Nintex Workflow 2007, I'm just going to abbreviate it from now on to NW.

NW addresses a lot of the problems with SharePoint Designer such as lack of scheduling, loops and iterations, nested conditional branches, and flow control. Although third-party add-ons can extend SharePoint Designer to support these scenarios, other situations, such as scheduling and nested conditional branching, are just not possible without extensive modeling and complex logic.

In addition, NW has a very powerful feature that both Visual Studio and SharePoint Designer lack. In NW, you can define snippets consisting of several tasks and apply those snippets repeatedly in several workflows. This allows organizations to create complex tasks and then reuse those complex tasks as though they were a single activity.

You can even create workflow *templates*, which are complete workflows that are exportable and reusable not only across sites and site collections but also across entire farms. Although not as powerful and flexible as full Visual Studio workflows, the NW workflow templates provide nonprogrammers with the ability to create complete workflows in an authoring environment and then deploy those workflows into the production environment.

Of course, this also means you can test and debug workflows outside the production environment, something that SharePoint Designer is never able to do. And, since the templates are also exportable, you can keep them in a separate document library to give you the power of document libraries, such as versioning, approval, and backup.

NW also brings a new workflow editor to the table. However, rather than relying on a separate program, NW uses SharePoint as the host for the workflow designer. This means that you do not need to distribute any client programs to authors; as long as they have the proper permissions to the site, they can author and edit workflows without any special tools.

Figure 1 shows the workflow designer.

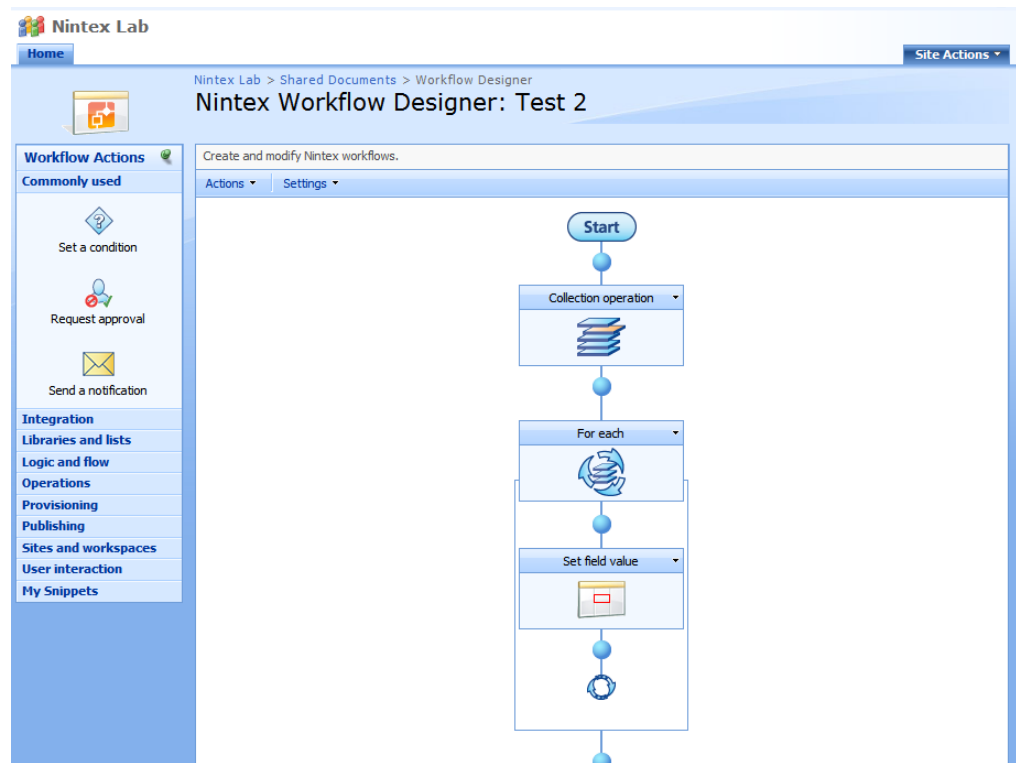


FIGURE 1. NINTEX WORKFLOW DESIGNER

Another feature of NW that is very useful is the ability to schedule workflows. This feature is not available in either SharePoint Designer or Visual Studio workflows. In fact, if you want to schedule workflows using SharePoint Designer or Visual Studio, you need to create a custom timer job to trigger workflows.

Nintex has done that for you and created a customizable and powerful schedule tool for you, allowing you to set up one-time scheduling as well as repeated scheduling, either per hour, day, or month. Sadly, you do not get any higher resolution than one hour, but for most uses, this may not be a problem.

We will explore the features of NW further throughout this issue, and I will show you how to apply them in real-life scenarios.

## **Logic**

Being able to control the flow of your workflow is incredibly important. Sadly, the methods for doing so in SharePoint Designer are quite limited or nonexistent, and going the Visual Studio path is complicated and does not allow for easy modifications after you have deployed a workflow.

## **Loops**

One of the most annoying shortcomings of SharePoint Designer is the lack of loop control. For example, if you want to iterate through a list of items, you need to apply some pretty complex workflow logic. If you want to remind someone every now and then to check their uncompleted tasks, well, the logic is even more complex.

And I haven't even started to talk about how you can escalate a task that hasn't been completed with a given time frame. At this point, we are talking several recursive and secondary workflows, at least three or four columns in an item, just to track workflow progress.

NW supports several methods for looping. You have the option of performing a task or set of tasks on all items in a list. Combine this with some conditional branching or a filter, and you can run workflows only on items that meet certain criteria.

Another option for looping is to loop on a collection. A *collection* in NW terms is a variable type that contains, well, a collection of something. For example, you may have a collection of strings of text you want to add to a message, or you may have a list of users who need to approve an item. These strings or users can be stored in a collection, and you can use a "For each" action to iterate this list and perform an action or set of actions for each item in the collection.

You can also create escalating workflows very easily. An escalating workflow will basically perform some action if another action has not happened for a given period of time.

For example, you may assign a task to an employee, and then that employee gets sick or leaves for vacation. Or, who knows, in these times, employees have been known to simply vanish in the great unknown called *downsizing*.

In this case, an escalating workflow is able to take control after a given period of inactivity and delegate the task to either another employee, a group of employees, or a manager. You can set up this delegation to happen after any time period, even excluding weekends and holidays, as shown in Figure 2.

The screenshot shows a dialog box titled "Configure Action -- Webpage Dialog" with a sub-header "Delegate workflow task". The configuration is as follows:

- Action ID\***: A dropdown menu set to "Collect feedback action".
- Delegate after**: Time fields for Days (10), Hours (0), and Mins (0). A checkbox for "Workdays only" is checked.
- Delegate to\***: A text field containing "Nintex Lab Members".
- Comments**: A large empty text area with an "Insert Reference" button to its right.
- Apply to**: Two radio button options: "All pending tasks" (selected) and "First pending task (other pending tasks will be not required)".

At the bottom left is the number "10903" and at the bottom right are "Save" and "Cancel" buttons.

FIGURE 2. "DELEGATE WORKFLOW" TASK

### Note

The tasks I mention here are possible to solve in SharePoint Designer, but that requires you to perform tasks for which SharePoint Designer workflows are not well suited. Although I do not condone using such methods in a production environment, I'll show you how in the companion issue 4 of Understanding SharePoint Journal on SharePoint Designer workflows.

## Settings and Configuration Scope

NW allows you to configure settings to control your workflow experience. For example, you can do the following:

- Control workflow permissions
- Set up messaging and email templates
- Approve and disapprove individual actions
- Define holidays to prevent workflows when people are not working
- Create and manage workflow constants such as strings, numbers, or even credentials

The cool thing here is that your settings can be *scoped*, meaning you can configure settings for individual sites, the site collection, or even the entire farm.

The not-so-cool thing here is that you need to set up these settings manually through the web interface if you want to override the parent settings. Also, when editing settings, you need to pay attention to which scope you are editing, because the page used to configure site collection settings and site settings is the same and differs only in the text displayed on it, as shown in Figure 3.

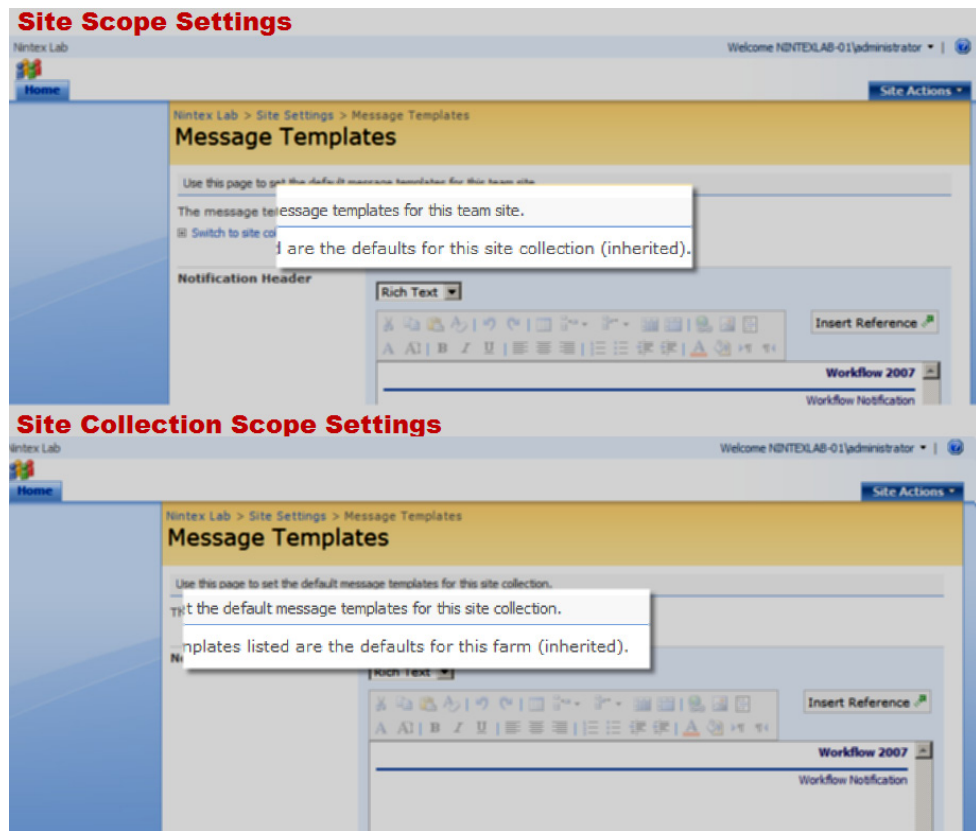


FIGURE 3. SETTINGS SCOPE DIFFERENCES

Still, scope gives you the power, with a bit of manual work, to tailor exactly how you want workflows to function, who gets to author workflows, how the message templates should appear, and all the other settings you can configure.

What are these settings, you ask? I'll tell you in Chapter 2.

## State Machines and Sequential Workflows

Although I will not go too deep into workflow basics in this issue, one concept is especially important to understanding NW.

Workflows come in two varieties, as either a sequential workflow or a state machine workflow. Although they're two terms, the two forms are closely connected.

The simplest form is the *sequential workflow* in which several actions occur in a predefined sequence. You can use conditions to branch the workflow, but in the end it is just a matter of going from one end to the other in order. SharePoint Designer workflows are sequential workflows.



A *state machine workflow* is a bit more complex, but if you think of state machine workflows as a collection of sequential workflows, you may find them easier to understand. Each state in a state machine is really just a separate sequential workflow. Either during or at the end of each such subworkflow you can enter another state and thus trigger a second sequential workflow.

Another difference is that while a sequential workflow always ends, a state machine workflow may keep going indefinitely, at least in theory. Think of a workflow to have someone exchange the calendars in an office; unless time itself stops or we stop using calendars, that workflow will keep going forever.

SharePoint Designer does not really support state machines. This is a problem, because state machines are really the best model for workflows that interact with humans. The reason is that state machines can adapt to the unpredictable nature of people, while sequential workflows go from start to finish, regardless of what the person does or wants.

If you want to create state machine workflows in SharePoint, you need to use another method of workflow development than SharePoint Designer. Visual Studio is one option, and, as might have guessed by now, NW is another. Figure 4 shows a typical state machine. In fact, I borrowed this from the Nintex folks, and I hope they don't mind.

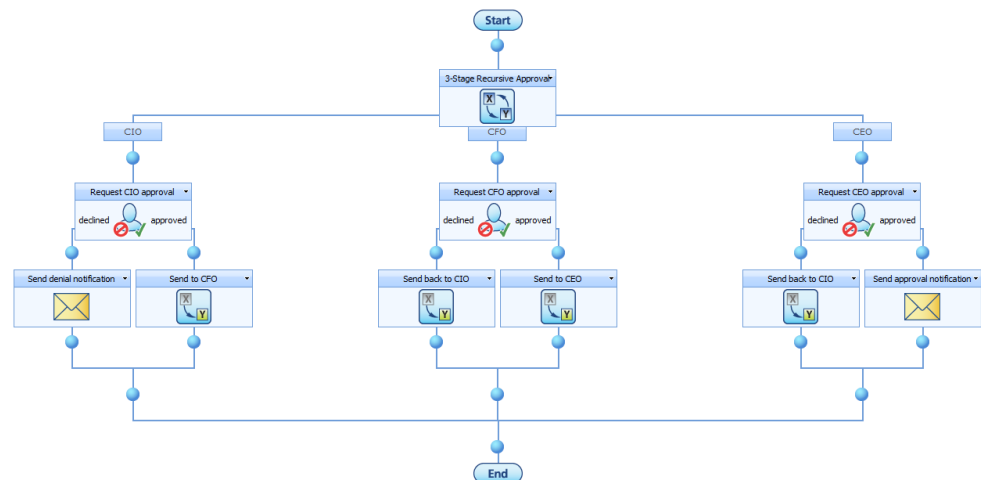


FIGURE 4. STATE MACHINE WORKFLOW

I will teach you the basics of state machines in Chapter 7.

## What's Up Next?

Throughout this special issue, I will introduce you to many of the features of NW. Sadly, space and time limits prevent me from explaining the full range of features; however, I hope that the features I do show you will inspire you to explore more on your own.

So, in the next chapter, I will show you how to get your NW environment up and running, from installing to configuring to setting up your workflow environment. You'll learn what the various settings do, how to test and verify your environment, and how to activate NW for your sites.

In Chapter 3, I will introduce you to the Nintex Workflow Designer and the principles behind designing workflows in NW. I will also introduce some of the activities, show you variable management, and explain the most common dialog boxes in NW.

Chapter 4 will further introduce you to the activities in NW and show you some simple tasks you may recognize. I'll show you how to set up an approval workflow, which may be familiar to you if you have worked with Microsoft Office SharePoint Server. I'll also demonstrate some other tasks that should make you comfortable with the workflow authoring experience.

One particularly powerful option in NW is to work with external data. In Chapter 5, I will show you how you can gather data from an external data source using a web service and then use that data in your workflow. We will do this by retrieving the list of users in the SharePoint site and iterating through those users in a loop.

In Chapters 6 and 7, we will create a complete sample application. The application will be a lighter exercise, with "light" meaning not-so-business-oriented. More to the point, I will show you how to create a "get well" card workflow where a sick employee will receive good wishes from their fellow co-workers.

### What You Will Need


NW does not require MOSS to be installed, so your regular WSS installation should suffice. That said, having MOSS installed does not adversely affect your options in any way, but some of the NW actions can interact with MOSS if installed.

You also need NW 2007 installed, and you can download a trial version from the Nintex website listed in the left margin. The trial will be for the enterprise version of Nintex, so you will get to test all the features, including the enterprise-only exercises in this issue.

Note that you will need trial key that comes as part of your download. The key will be active for 30 days from the day you request it, not from the day you install NW.

**Nintex download:**  
<http://www.nintex.com/en-US/Products/Pages/TrialDownload.aspx?v=NWF.2007>

In the next chapter, I will show you how to install and set up your NW environment. Before any of this, however, let's review some concepts from this chapter. The answers to these questions are on the next page.

 REVIEW QUESTIONS

- ❓ 1. Why is creating complex workflow logic such as looping and nested branching a bad idea in SharePoint Designer?
- ❓ 2. What tool do you use to author Nintex workflows?
- ❓ 3. Which workflow model, sequential or state machine, is best suited for human interaction, and why?

 REVIEW QUESTIONS ANSWERED

- ! 1. Creating complex workflow logic is a bad idea because the tool was never designed to perform such tasks, and it therefore forces you to work around the tool rather than focus on the logic of your workflow.
- ! 2. You would use the Workflow Designer tool in SharePoint to author Nintex workflows.
- ! 3. State machine workflows are best suited for human interaction, because humans are unpredictable and because state machines can accommodate that unpredictability.

## Setup and Administration

*Set up NW, and teach it to behave.*

Nintex Workflow 2007 does not require any client-side software, but you do need to install NW on the server. You may want to perform the installation and set up in a lab environment first, if you have never done so. There are pitfalls, so don't risk your production environment.

In this chapter, you will learn how to install and configure NW on a server with only WSS installed. I will assume that you know how to set up SharePoint or have a SharePoint site set up for you.

Next, I will guide you through the setup, including installing your license and getting ready to author your first workflow.

Let's get down to business.

### Installing Nintex Workflow 2007

Before you do anything, pay close attention to the following caution:

#### Caution

Any version of NW prior to build 10904, released May 7, 2009, is not compatible with Service Pack 2 for SharePoint. You need to get or upgrade your Nintex version to at least build 10904. If you are downloading Nintex for the first time now, this should not be a problem, but check your existing installation before you install Service Pack 2 for SharePoint. For more information, see <http://connect.nintex.com/forums/thread/3640.aspx>.

With that in mind, feel free to make sure you upgrade to Service Pack 2 for SharePoint. If you want to compare SharePoint Designer 2007 as well, you should know that there is also a Service Pack 2 for SharePoint Designer.

## Prerequisites

You need administrative permissions on the SharePoint server to install NW. This includes farm administrator permissions. You also need to have permissions to create new databases on your database server. NW uses a separate database to store data, and I will show you how to set this up later in this chapter.

You should also configure your SharePoint environment with incoming and outgoing email settings. Workflow features often rely on email communications to relay task assignments or to allow workflow interaction through incoming email. For example, in NW, the LazyApproval feature depends on incoming email to function at all.

So, at this point, you should have downloaded Nintex Workflow 2007 and installed SharePoint on your server.

## Installation and Deployment



Double-click the installer .msi file to start the installation wizard, which will take you through the normal steps of asking you to accept the end user license agreement and specify the installation path.

After this, the wizard asks you whether you want the solution added to SharePoint automatically. If you select Yes, the default, the wizard will add the solution and reset IIS to ensure that all XML files are properly added to SharePoint.

If you do not choose to let the wizard add the solution for you, you need to manually add the solution after the wizard completes. This allows you to perform the IIS reset during off-peak hours to avoid interrupting regular work on the site.

## Note

If you select not to add the solution now, you will need to do so manually. Make a note of the installation path from step 2 of the installation wizard if you want to go with this option. The default is C:\Program Files\Nintex\Nintex Workflow 2007. You should find NintexWorkflow2007.wsp in that folder.

Then, use the following command to add the solution to SharePoint:

```
stsadm -o addsolution -filename [path to NintexWorkflow2007.wsp]
```

Repeat these steps for the NintexWorkflow2007EnterpriseFeatures.wsp solution if you are deploying or evaluating the enterprise features of Nintex as well.

After the install wizard completes its magic, which is what I believe wizards do, and you have made sure that you have added the solution if you need to do so, you can now go to the Central Administration site to deploy your solution.

You will find the Solution Management link on the Operations page of Central Administration. You should see the Nintex solution or solutions there, as shown in Figure 5. In that figure, I have added only the standard NW feature solution.

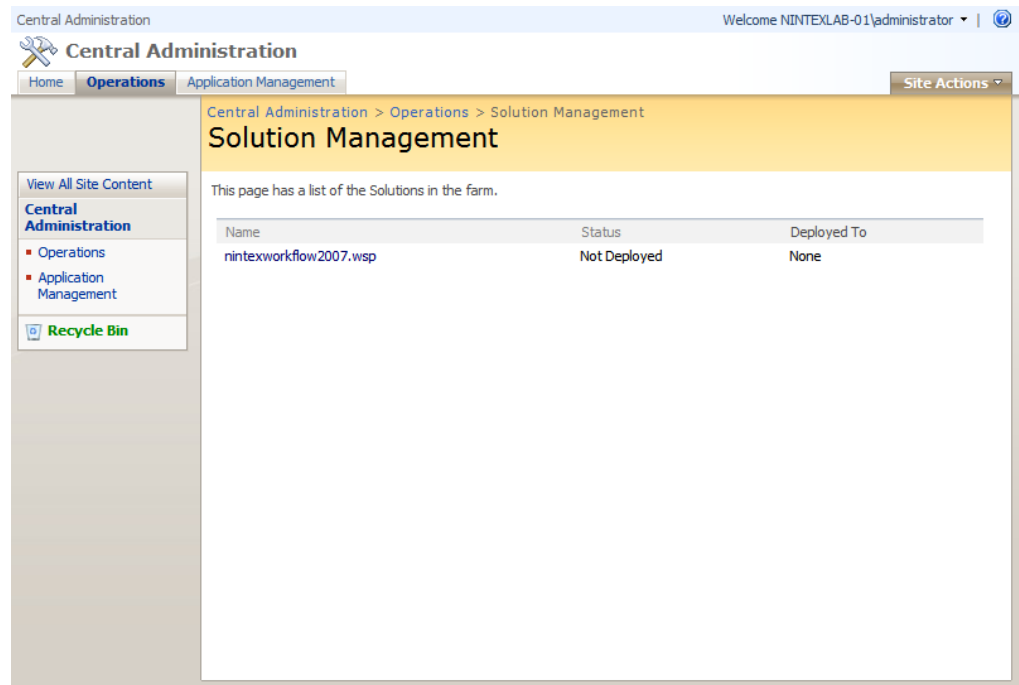


FIGURE 5. SOLUTION MANAGEMENT

For each of your Nintex solutions, click the name of the solution, and then click Deploy Solution. Select the web application to which you want to deploy NW, and choose the time for deployment. In a lab environment, “All content web applications” and “Now” are very valid choices.

Hit OK, and your deployment should be done.

**Note**

If you are installing in a new lab environment, make sure you have at least one web application created before you attempt to deploy the solution. If you do not, deployment will fail, with an error message stating “This solution contains resources scoped for a Web application and must be deployed to one or more Web applications.”

Yeah, I know I could you have told you that before.

## Central Administration Setup

Once you have deployed the solution, hop over to the Application Management page of Central Administration. You will see a new section called Nintex Workflow Management, as shown in Figure 6.

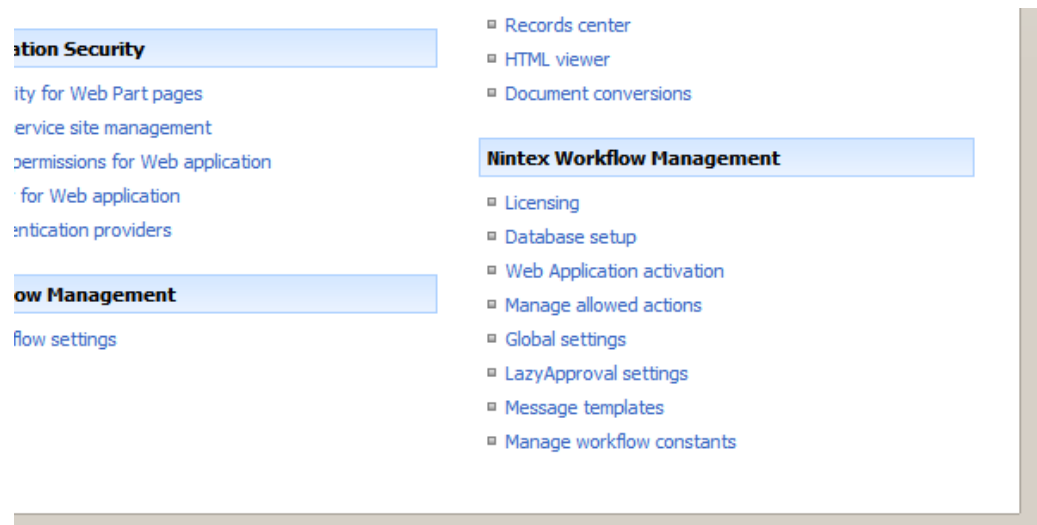


FIGURE 6. NINTEX WORKFLOW MANAGEMENT



 **Configuring  
Nintex  
Workflow**

Your first step to getting everything up and running is to install the license. Simply click the Licensing link in the Nintex Workflow Management section, then click Import, browse to the license file you have received from Nintex, and finally hit Import to add your license. Your license page should now display a nice plaque with your license details.

Your second step will be to set up the Nintex database. NW uses a separate database to store configuration and workflow content. Go to the database configuration by clicking “Database setup.”

The first time you enter this page, you will only have the option to create a new configuration database. Click the Create button to do so. You will be asked for the database server, which may be the same server you use for SharePoint, as well as the database name.

**Note**

If you are restoring your environment from a backup or have an existing database available, you can connect to an existing configuration database by selecting the checkbox under the “Database name” input box.

If you are using a separate SQL account for database access, you can set up that in the last section of the page or just leave the recommended Windows Authentication enabled.

NW will create the configuration database when you click OK and then take you back to the Database Setup page where you will now see your settings for the configuration database, as shown in Figure 6.

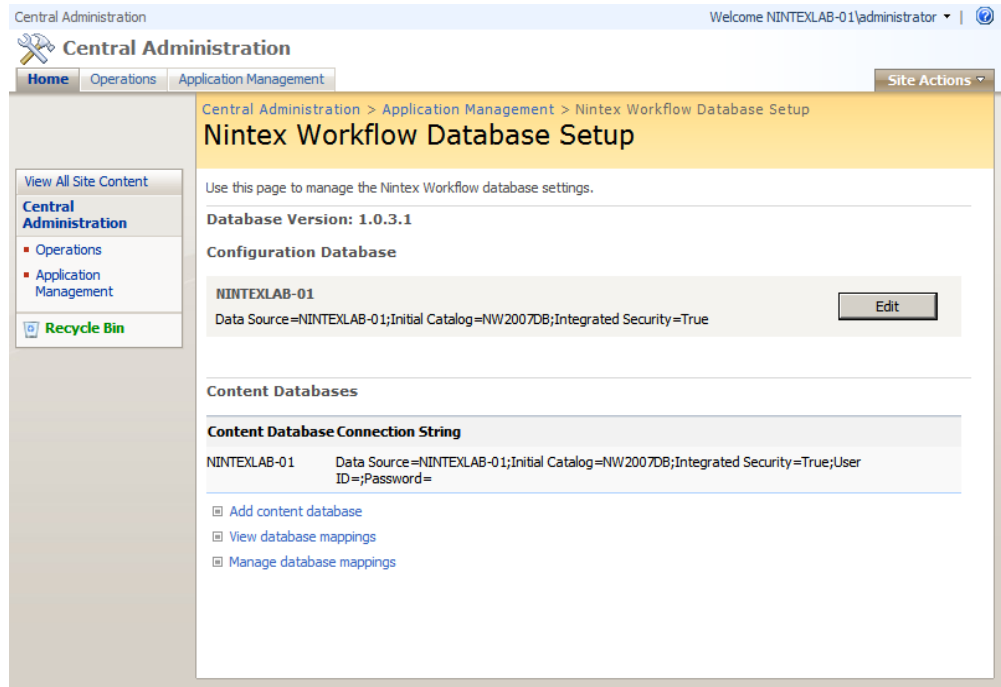


FIGURE 7. DATABASE SETUP PAGE

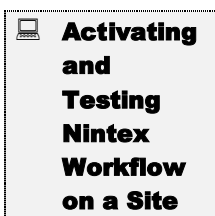
The third step is to go back to the Application Management page and click “Web Application activation.” This page will allow you to activate NW for site collections in specific web applications or in all web applications for the site. Select the web application for which you want to activate NW, optionally select “All content web applications,” and click Activate.

At this point, you have done all the mandatory configuration steps for testing NW in your site. I will take a short break here and show you what you need in order to test it, but note that we still have some tasks that we really should do and some options that are just fun to do.

## Features and Activation

So, it is time to see NW in action and ensure that our setup is correct. This will be a dry run, so to speak, because we still need to perform some additional configuration to ensure that we can author and modify workflows.

First, go to the site where you want to use NW. On that site, go to the Site Settings page, available from the Site Actions menu. Then, go to Site Collection Features. You should see two features related to Nintex Workflow, as shown in Figure 8. Activate the feature called Nintex Workflow 2007. You can leave the web parts feature deactivated for now.



**Activating  
and  
Testing  
Nintex  
Workflow  
on a Site**

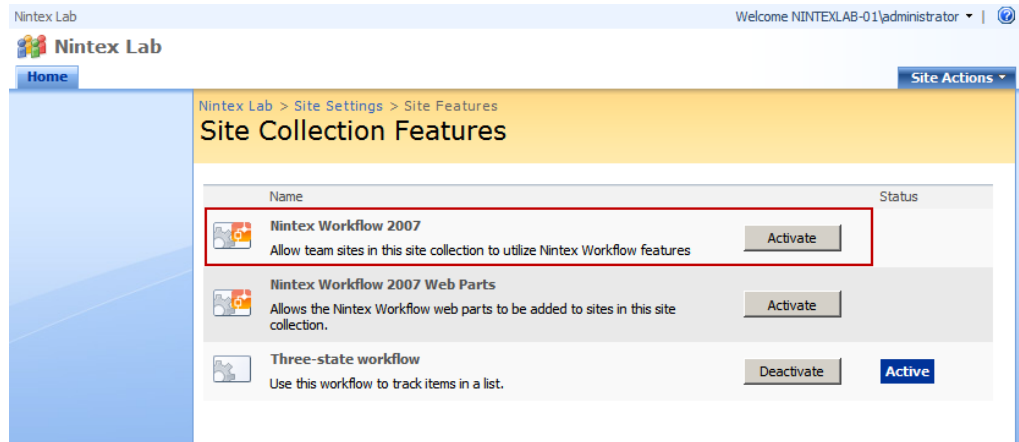


FIGURE 8. SITE COLLECTION FEATURES

**Note**

You should activate the feature from the Site Collection Features page first. Confusingly, there is also a Nintex Workflow 2007 feature on the Site Features page, but you must activate the site collection–scoped feature first.

Activating the Nintex Workflow 2007 site collection–scoped feature will prepare the site collection for using NW. Technically, the feature will add a hidden list template, several content types, and some site columns, all used to support the authoring and running of workflows.

**Tip**

You can verify the activation of the site collection–scoped feature by clicking the “Site content types” link and the “Site columns” link on the Site Settings page.

After activating the site collection–scoped Nintex Workflow 2007 feature, you should return to the Site Settings page and then to the Site Features page. You will find another feature named Nintex Workflow 2007. Activate the site feature now.

This feature will have a more apparent impact on your site. First, return to the Site Settings page, and note that at the bottom you will now find a new section called Nintex Workflow, shown in Figure 9. This is where you will configure NW settings for this site or, in the case of the root web of a site collection, the settings for the site collection.

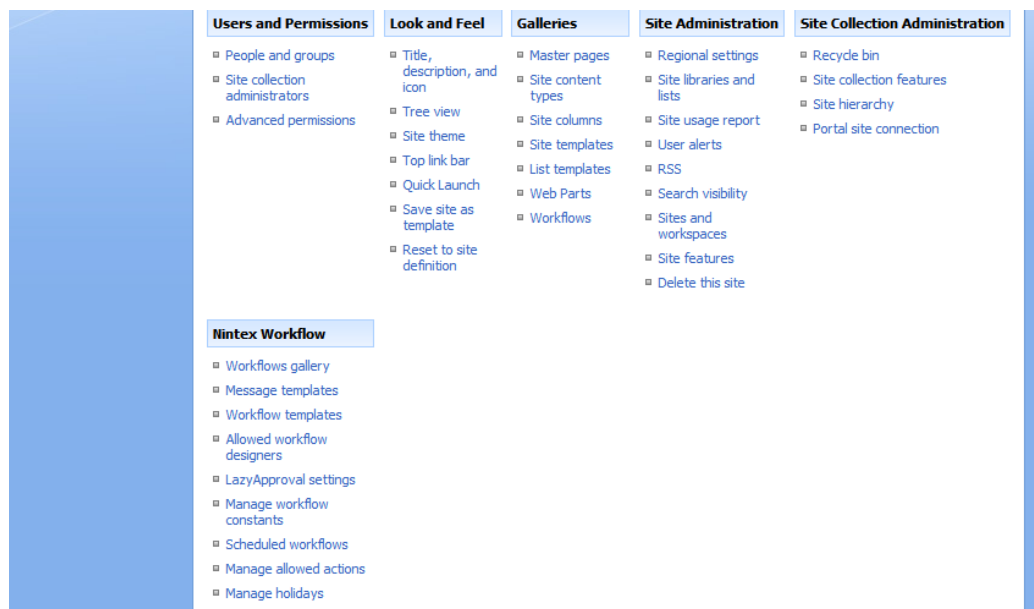


FIGURE 9. NINTEX WORKFLOW SETTINGS

Next, go to any list or library in your site. If you have used a standard Team Site template for your site, you can, for example, go to the Shared Documents library. If you have created a blank site or another template without lists, just create any list or library, because you can delete it after testing.

Once in the list, click the Settings menu, located in the toolbar for the list. Notice that you have two new items in the Settings menu, called Manage Workflows and Create Workflow, as shown in Figure 10.

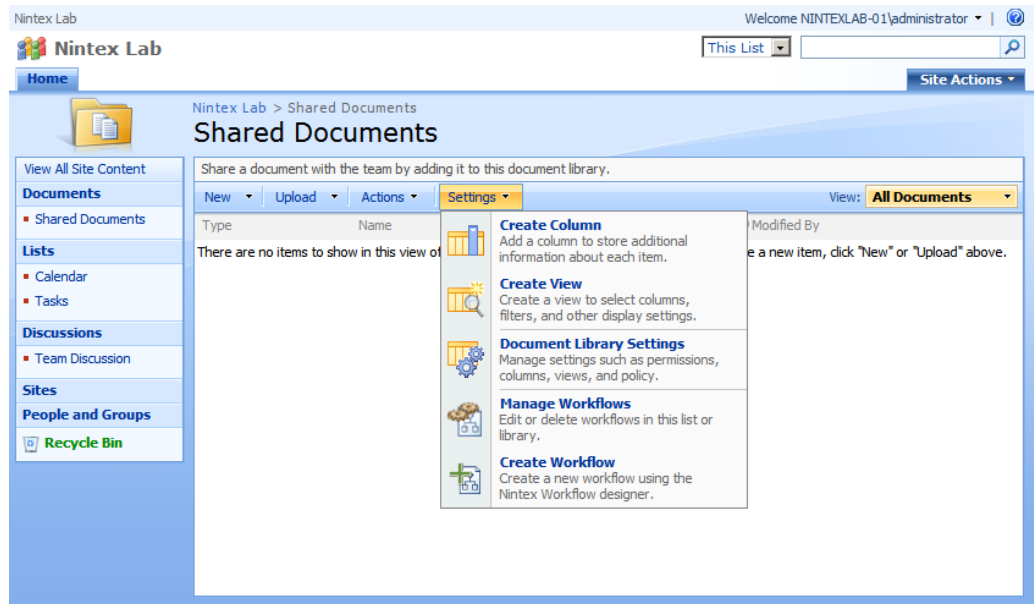


FIGURE 10. SETTINGS MENU

The Manage Workflows page is where you will manage your workflows. Since you have yet to create any workflows, this page serves little purpose at this time, but we will get back to it in a moment.

For now, click Create Workflow to start a new workflow.

**Note**

At this point, we still haven't set up everything we need to author any workflows. However, included in the installation of NW are several premade templates for common tasks.

When you go to create a new workflow in NW, the first thing that pops up is a dialog box, prompting you to select a template for your new workflow. NW ships with several premade templates from which you can choose, and you can even add your own templates to this collection.

The blank template is your choice if you want to create your own workflow from scratch. This is very useful when you want to completely design the entire business process, but consider learning at least which premade templates are available. I'll tell you why in a moment.

Figure 11 shows the template selection dialog box.

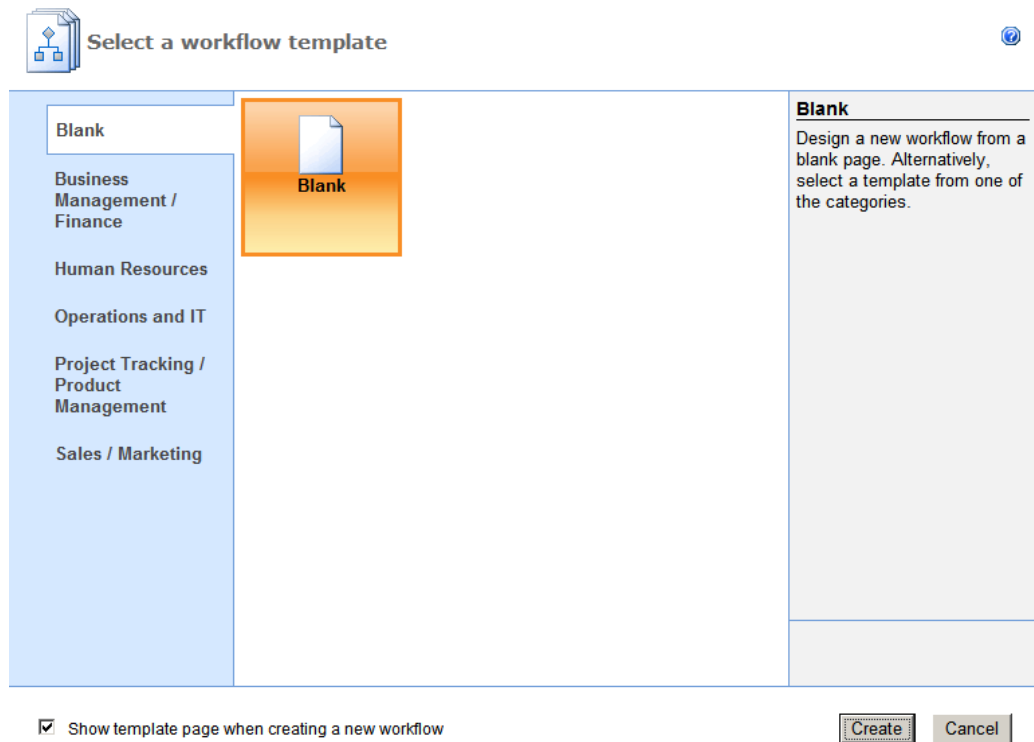


FIGURE 11. TEMPLATE SELECTION DIALOG BOX

The premade templates are somewhat useful out of the box, but the true benefit comes from modifying the templates. Start with one existing template, and then modify the workflow based on that template. Not only will you save time, but you can take inspiration from the templates and use NW for scenarios you may not have considered already. For example, did you know that you have a premade template for employee absence management?

Go to the Human Resources category, and select Absence Manager Approval from the templates. Click the Create button to fill the Workflow Designer window with the premade template, as shown in Figure 12.

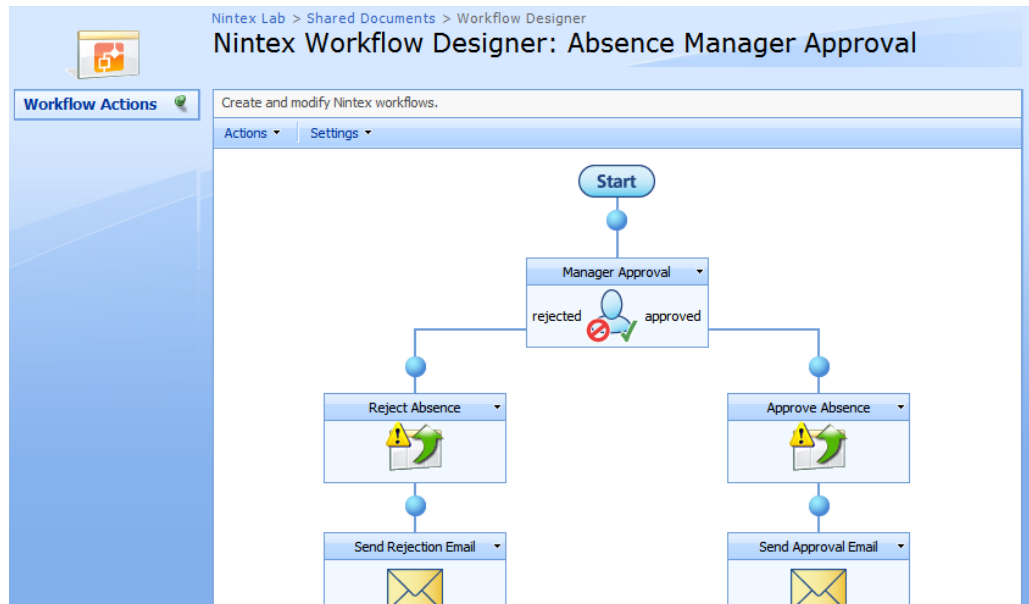


FIGURE 12. ABSENCE MANAGER APPROVAL TEMPLATE

We will explore the default templates in the next chapter when we learn more about the workflow editor, but notice that the Workflow Actions pane is empty.

For now, however, let's continue with the remaining configuration options.

## Optional Workflow Settings

If all you wanted to do was utilize the premade workflow templates, then you could skip the rest of this chapter. And, since the rest of this issue deals with authoring workflows, you could pretty much skip the rest of the issue as well, but I sincerely hope you would rather learn more about the options you have available for customizing your workflow experience.

### Allowed Actions

The first thing you will likely want to configure is which actions and activities are permitted in workflow authoring. By default, NW does not allow any actions in the workflow editor, as you can see from the lack of any workflow actions in Figure 12.

You can allow workflow actions per site or site collection, and you can also allow actions for the entire farm. To manage the allowed actions, go to the Site Settings page, and click the "Manage allowed actions" link in the Nintex Workflow section.



When you first enter the Manage Allowed Actions page, you will not see any actions at all, because no actions are allowed by default. To show the allowable actions, click the "Switch to admin view – Show all actions" link. Your page now shows all available actions, as shown in Figure 13.

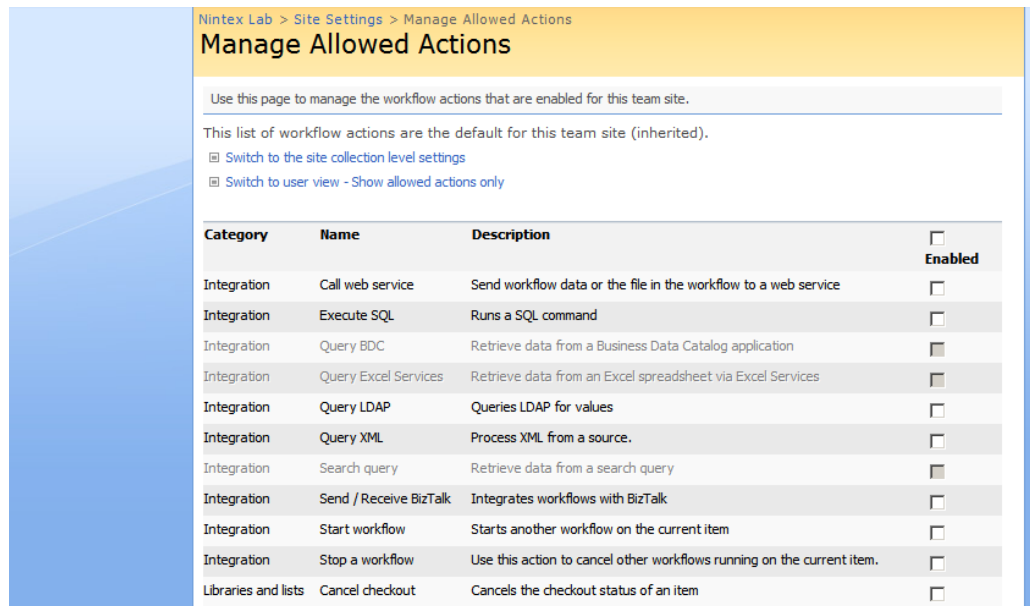


FIGURE 13. ALL ALLOWABLE ACTIONS

Notice that some actions are disabled. You can hover your mouse pointer over a checkbox to see why the action is not allowable. For example, the Query BDC and Query Excel Services actions require that you have MOSS installed.

Notice also that you have a link at the top of the page named “Switch to the site collection level settings.” This link will, as promised, change the scope of your selection to the site collection scope. However, if you click that link, you need to go back to the Site Settings page in order to go back to the site-scoped settings, because the site collection–scoped settings have no link back to the site scope.

For now, select the checkbox above Enabled in the rightmost column to enable all actions, and then scroll to the bottom of the page to click the Ok button.

You can also set the allowed actions on a farm level. To do so, you would go to Central Administration and follow the link called “Manage allowed actions” in the Nintex Workflow Management section of the Application Management page.

### LazyApproval

I absolutely love any software feature with the word *lazy* in it. NW has a feature called LazyApproval, and I think this is a fantastic feature.



In short, LazyApproval allows task assignees to respond by email to interact with a workflow. For example, a manager may approve or disapprove a vacation request simply by responding to an email and include certain configurable words in that email. Users can also fill in data for Collect Data tasks and send them as email, allowing users to complete their tasks without going to the SharePoint site. This feature is very useful if people are on the road or outside the office.

 **Setting Up  
LazyApproval**

To set up LazyApproval, you need to ensure that you have set up your server to support incoming email. You do this via the Operations page on the Central Administration site. Contact your server administrator to set this up if you require assistance.

**Hey, I Am the Server Administrator!**

I hate it when someone tells me to contact my server administrator when, in fact, I am the server administrator.

A deep dive into how to configure incoming email is beyond the scope of this issue. However, if you are testing NW in a lab environment, you can use the IIS SMTP service.

You can add the SMTP service used in IIS by going to Add/Remove Programs, clicking Add/Remove Windows Components, going into the Application Server details, and then going to the Internet Information Services details.

Next, go into the Operations page of Central Administration, click the Incoming Email Settings link, and enable incoming email using the Automatic option.

The next step to set up LazyApproval is to click the “Global settings” link on the Application Management page of Central Administration. I will go deeper into the detail of the global settings later in this chapter, but you need to ensure that at least the settings for outgoing email are correct. Fill in your outgoing email server as well as the email address and reply to address.

**Tip**

Here’s a little tip. Set the reply to address to the same address you will use for LazyApproval, for example, lazyapproval@yourdomain. That way, when people get notifications about tasks to complete, they can simply reply to the email to interact with the workflow.

When you are done, click Ok at the bottom of the page, and then click the “LazyApproval settings” link.

First, click the Enable/Disable LazyApproval link. Then, select Enabled, provide the email alias to which you will receive LazyApproval interactions (for example, lazyapproval@yourdomain), and then hit Ok. Figure 14 shows my setup.

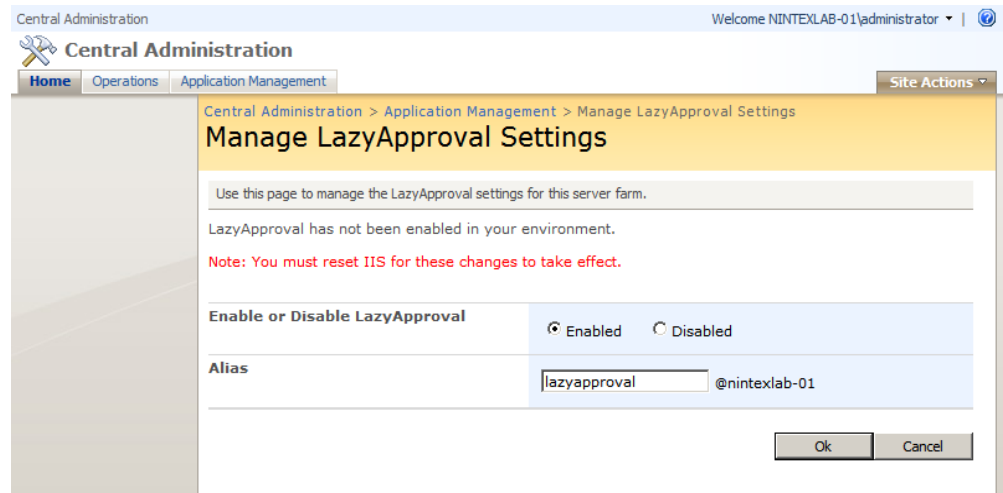


FIGURE 14. LAZYAPPROVAL ENABLED

When you have hit Ok, the LazyApproval page changes and shows you the default terms for approval and denial of tasks. I'll show you how to add, delete, and modify the terms used in a moment.

You can change the notification footer text to provide users with additional instructions for how the LazyApproval feature works, but personally I think the default text serves its purpose.

If you want to modify the approval or denial terms globally, you can do so from Central Administration. However, you can also do this on the site or site collection level. To do so, go back to the Site Settings page of your site, and click the "LazyApproval settings" link.

Next, click the link named "Create a new LazyApproval term" for the current Team Site, and fill in the form with an appropriate new term for approval or rejection, as shown in Figure 15.

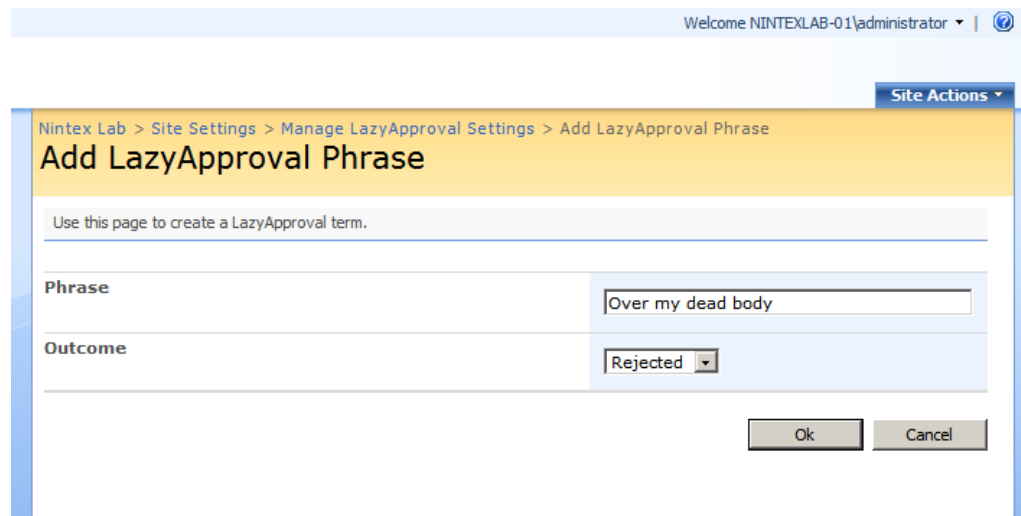


FIGURE 15. LAZYAPPROVAL REJECTION TERM

Users are now able to reply to the email they receive using the term you define to reject or approve the request tasks assigned to them.

Neat, eh?

### **Holidays**

You can set up holidays from sites or site collections. Many workflow actions allow you to avoid user interaction during defined holidays, as you will see in the action configuration dialog boxes later.

However, holiday management in NW leaves a bit to be desired. First, you have no global administration of holidays. If you have many site collections, this means a lot of work.

Second, you can define only single days at a time. If you want to add a week of holidays, you need to add five separate days.

Despite these minor issues, holiday management is very useful for preventing task advancement when no one is working.

### Note

Users can also set up their own holidays by going to the personal actions menu and selecting the Nintex Workflow 2007→Task Delegation menu option.

To set up holidays for the entire site or site collection, go to the Site Settings page, and click “Manage holidays.” As for other management pages, you can switch to the site collection scope using the link near the top of the Manage Holidays page.

Adding a new holiday is as easy as clicking the “Add a new holiday” link and specifying a name and date for the holiday. You can also set up annual holidays by selecting the checkbox, as shown in Figure 16.

The screenshot shows a web browser window with the URL 'Welcome NINTEXLAB-01\administrator'. The page title is 'Create Holidays' and the breadcrumb trail is 'Nintex Lab > Site Settings > Manage Holidays > Create Holidays'. The form contains the following fields:

Use this page to add a holiday date.	
Name*	Björn's Birthday
Date*	7/21/2009
Annual holiday (repeat yearly)	<input checked="" type="checkbox"/>
Ok Cancel	

FIGURE 16. ADD NEW HOLIDAY

### Workflow Constants

On the other end of the useful scale lie workflow constants. Workflow constants are values that are available to workflows and actions, managed on the farm, site collection, or site level.

**Note**

Workflow constants should not be confused with workflow variables, which are values that exist only for a single workflow and that lose their value after workflow completion.

The workflow constants have no equivalent in SharePoint at all, but they are extremely useful. For example, you can define a global constant with the contact information for the server administrators so that you can report any problems with a workflow without knowing who is currently in charge of the server. Or, you can set a seasonal greeting in a string value and use the greeting as part of emails to customers, employees, or others.

The management scope of constants means you will be able to manage the values centrally without ever touching the actual workflows.

Now, to top this off, NW has a special constant data type for storing credentials. This allows users to provide credentials for authentication to external systems without knowing the username or password for that external system. NW encrypts the credentials in the database so they are stored safely from prying eyes.

**Note**

You can also ensure database encryption of other data types by selecting the Sensitive checkbox when you create or edit a constant. You will still be able to see the clear text or value when you edit a constant even if you choose to encrypt the value in the database.

You define workflow constants for the farm inside Central Administration, using the “Manage workflow constants” link. Click the “Add a new workflow constant to this server farm” link, and select the correct type of constant before filling in the rest of the form, as shown in Figure 16.

Welcome NINTEXLAB-01\administrator |

ministration

Application Management | Site Actions

Central Administration > Application Management > Manage Workflow Constants > Add Workflow Constant

### Create Workflow Constant

Title*	HR System authentication
Type	Credential
Username*	HR User
Password*	••••••
Description	Credentials used for authentication to the HR System application.
Constant is only visible to server farm administrators	<input type="checkbox"/>

Ok Cancel

FIGURE 17. CREATE WORKFLOW CONSTANT

When you go to edit a workflow task, you can now look up the constant and use it for configuring the action, as shown in Figure 18.

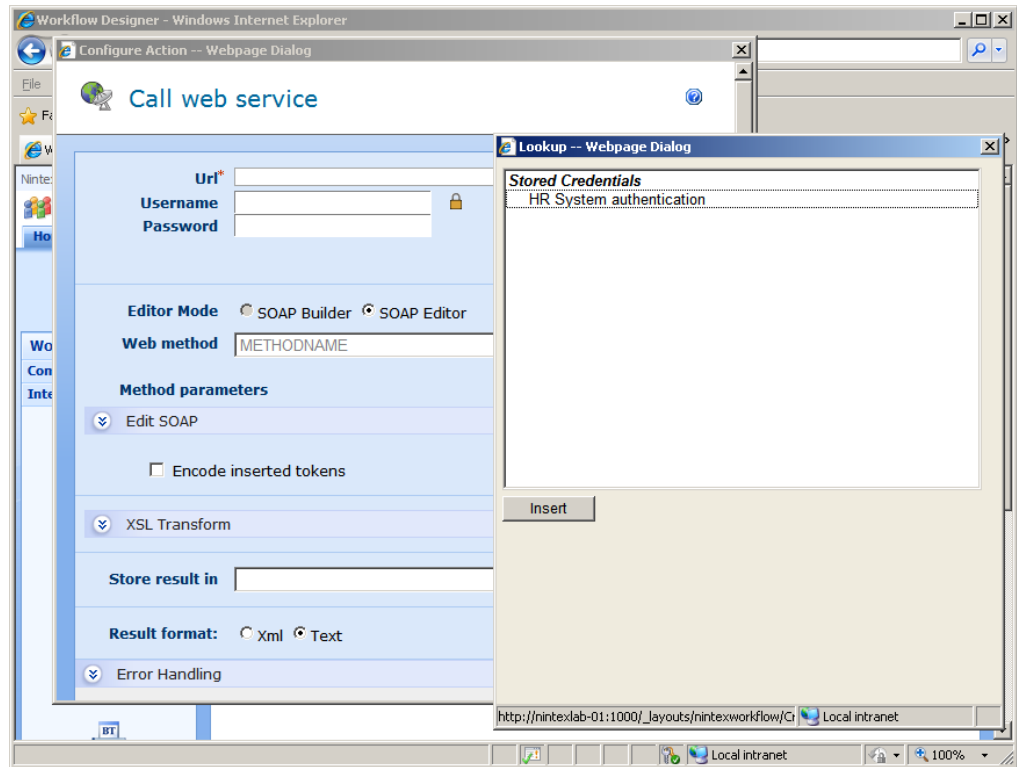


FIGURE 18. USING WORKFLOW CONSTANTS

**Note**

We will explore the action editors more in the next chapter.

**Message Templates**

NW also allows you to define the templates used for various notification messages, a common request from SharePoint Designer users. Customizing these templates allows you to provide detailed information to the assignee or change the language or content to your business needs.

As for most settings in NW, you can define messages on a farm, site collection, or site level. The link, called “Message templates,” is located both on the Application Management page of Central Administration as well as on the Site Settings page. I won’t bore you by repeating how you switch from site to site collection scope.

Note the Insert Reference button in Figure 19. You can use this button to, well, insert a reference. What kind of reference? The most common use is to provide a reference to the tasks assigned or to the item on which the workflow is running.

However, you can also include workflow constants defined in any parent scope. This might be a good place to add that seasonal greeting to ensure your manager is in a good mood before you request your next leave of absence.

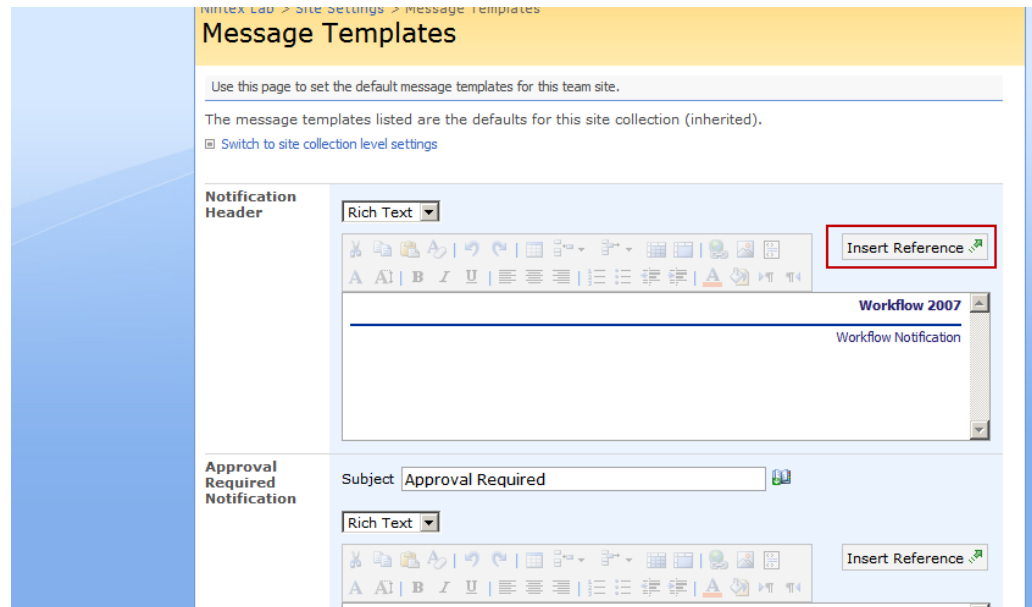


FIGURE 19. MESSAGE TEMPLATES PAGE

You can also define the subject of messages. This option allows you either a static text or a lookup value, but you can also combine the two. You can, for example, combine the text “Approval required for task by” and the name of the initiator for the workflow.

In a regular email, sent as part of an action, you can also use a Build Dynamic String action to accomplish the same thing. In SharePoint Designer, this would be your only option for creating a dynamic subject in an email.

## Global Settings

Before we finish this chapter and review the key concepts, I want to take you through some of the global settings you can configure for your workflow environment.

You will find the link to the global settings on the Application Management page of Central Administration. You will have been to this page before, if you set up LazyApproval earlier in the chapter.

The global settings are very well explained on the page itself, but I wanted to give you a few pointers on some of the settings.



## **Enforce Safe Looping**

A very common problem for any kind of developer is the endless loop. For example, consider a workflow that subtracts one from the number of days before Christmas this year as long as the number of days before Christmas is not zero.

This works well, until you start that workflow between Christmas and New Year 's Eve, when the number of days to Christmas is negative. Your workflow will continue to subtract forever, since no amount of subtracting will bring the remaining days to zero.

This is what is called an *endless loop*.

No amount of technical solutions can prevent you from making such mistakes, but in NW you can set an option to prevent an endless loop from spinning out of control. The option, the "Enforce safe looping" setting will add a slight time delay to your loop workflows, and although it will not get Christmas to come any sooner, it will at least make sure the server does not crash from trying to count down to an event that has already happened.

## **Enforce Allowed Actions at Run Time**

As you have already seen, NW allows you to selectively allow or disallow workflow actions on several scopes. However, what happens if you disallow an action that has already been added to a workflow?

Using the Enforce Allowed Actions at Run Time setting, you can define whether workflows with such actions are allowed to run, even if they are using currently disallowed actions.

The default setting is to allow existing workflows to run.

The only thing I think is missing from this feature is a per-action setting. You can only specify all or nothing; either all disallowed actions are permitted to run or no disallowed actions are permitted to run. It would be very useful to allow some disallowed actions to run, especially when debugging workflow actions before releasing them to your users. Ideally Nintex will add this in a future version.

## **Custom Workflow Start Page**

When starting workflows, you may often want to provide instructions for the user, for example, explaining how the workflow should behave and what to expect or perhaps even adding a nice drawing or video to guide users on how to use the workflow.

The default workflow start page in NW is very good and provides the initiator with an overview of the workflow as it is designed. This is so much better than in SharePoint Designer, where the default start page has only a button and optional comments and input controls for initiation data.

Despite this, I do have a couple of concerns with the way NW handles workflow pages.

In NW, you can allow the workflow designer to specify a custom page used for workflow initiation by setting the “Allow workflow designers to specify a start page” setting to Yes. These pages can be designed in HTML editors like SharePoint Designer.

However, unlike SharePoint Designer that creates a unique workflow start page for each workflow, NW uses a common default start page for all workflows. This means that to create a custom workflow start page you will need to copy the default page that NW uses and then modify the copy. This process could have been handled more smoothly in my opinion.

That being said, I have rarely had the need to customize these pages at all. With the improved start page provided by NW and with the option to create a custom start page, albeit a bit awkward, I believe that most scenarios can indeed be solved.

And, who knows, perhaps the next version will include better page management.

Right, with that said, I think we have completed the setup as far as we need, so it is time to get our hands dirty and start creating our own workflows.

Oh, I nearly forgot, here are the review questions. Don’t cheat; the answers are on the next page.

 REVIEW QUESTIONS

- ❓ 1. Why should you make sure your incoming and outgoing email settings are configured when working with workflow?
- ❓ 2. After installing, what should be your first step to configure Nintex Workflow?
- ❓ 3. Why is storing credentials in Nintex Workflow safe?

 REVIEW QUESTIONS ANSWERED

- ! 1. It is important to configure email because workflows often rely on email to communicate with users.
- ! 2. Your first step after installing should be to configure the license.
- ! 3. Storing credentials in Nintex Workflow is safe because the credentials are encrypted in the database.

## Getting to Know the Workflow Designer

*Reader, meet the workflow designer. Workflow designer, meet the reader.*

You now have your Nintex environment set up and ready to use. Great, but you still need to learn how to use it. That's up next.

In this chapter, I will introduce you to the basics of authoring and editing workflows in NW. I will show you how the Nintex Workflow Designer works, the most common dialog boxes, how to look up values for use in your workflows, and, of course, how to publish and test your workflows.

### Note

I assume that you have performed the exercises in the previous chapter. If not, well, all sorts of weird things may happen.

## Your First Nintex Workflow

No reason to hang around, we'll just get started.

In my examples, I will be using a standard Team Site, but any definition will do. You will need at least one list for this first exercise, and I will use the Shared Documents library. After that, you will create the list you need for the remaining exercises.

**Creating Your First Nintex Workflow**

Go into the list of your choice, and open the Settings menu. Click Create Workflow. The Select a Template dialog box should open. Feel free to look through the available workflow templates because, as I said in the previous chapter, knowing what templates are available can save you tons of time and inspire you to use SharePoint even better.

Go ahead, I'll wait right here.

When you are done looking through the various categories, go to the Blank category, and select the Blank template before you hit the Create button on the bottom right. After doing so, the dialog box will close and turn your page into something resembling Figure 20.

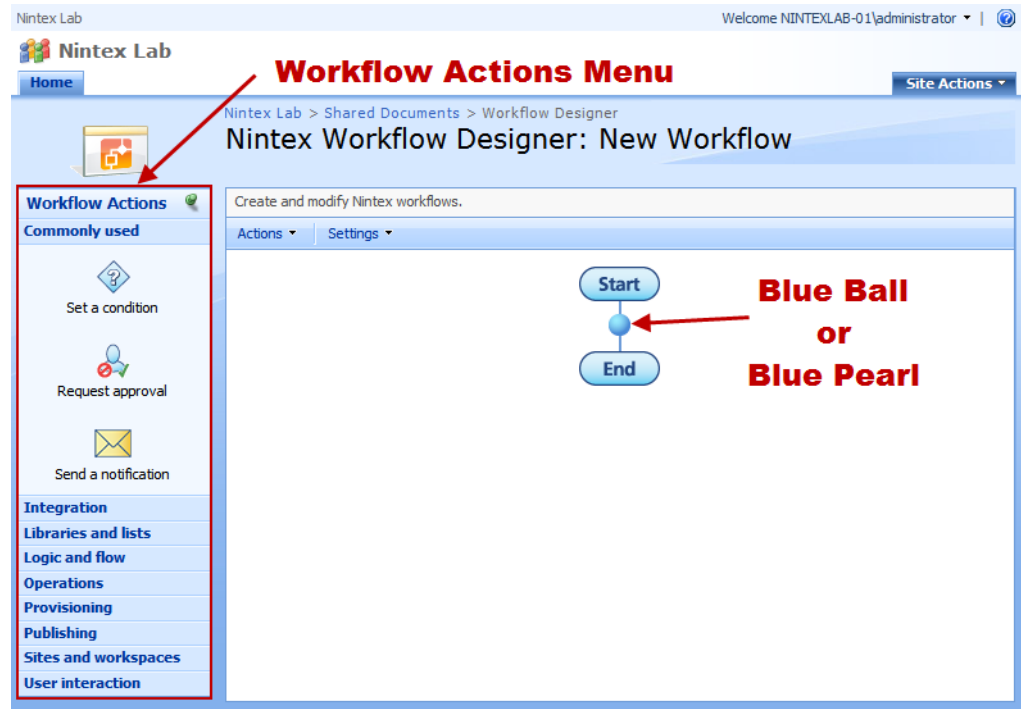


FIGURE 20. BLANK WORKFLOW TEMPLATE

No, you won't get my cool annotations. I have just added those to the figure to show you two important things.

First, notice the Workflow Actions menu on the left. This is where you will find all the allowed actions for the current site. You can see more actions by clicking the headlines, such as Integration, "Libraries and lists," "Logic and flow," and so on.

Second, notice the small blue ball or pearl between the Start and End in the designer surface. This is where you will add the actions in your workflow. Multiple pearls will appear as you add multiple actions, and some actions contain subactions and thus add new pearls inside the actions. I'll show you some of these actions in a moment.

You have two options for adding actions. First, you can drag and drop actions from the Workflow Actions menu on any blue pearl on the designer surface. Second, you can click any blue pearl and select the action you want to add from the Insert Action menu that appears, as shown in Figure 21.

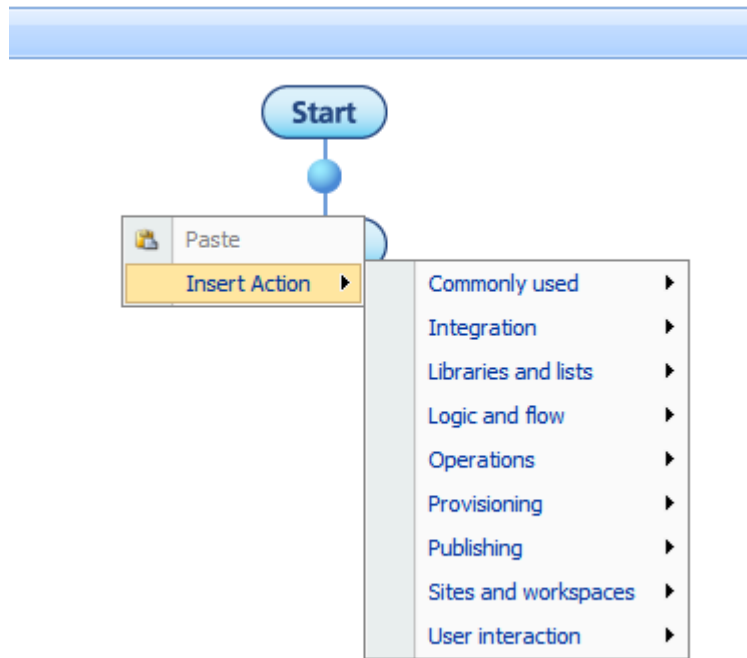


FIGURE 21. INSERT ACTION MENU

Add a “Send a notification” action to the single blue pearl now. Drag and drop it or use the menu as you see fit.

When you select or drag and drop an action, your selected action will appear on the designer surface, and you will see new blue pearls both before and after your newly added action, as shown in Figure 22.

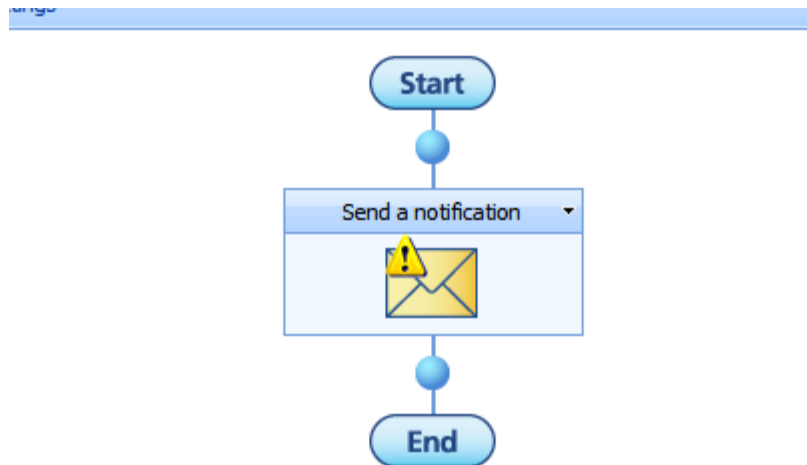


FIGURE 22. ADDED ACTION

To add multiple actions, simply repeat this process where you want your new action to appear.

When you have multiple actions, as you probably will in most workflows, you can rearrange actions by dragging the action icon, in this case the envelope, to a new blue pearl.

When you add an action, you need to configure that action. Until you do, the action will have a yellow exclamation mark to warn you that you have some configuration to do. If you hover your mouse pointer over the action, you will get more information about what you need to configure, as well as any errors in the configuration, as shown in Figure 23.

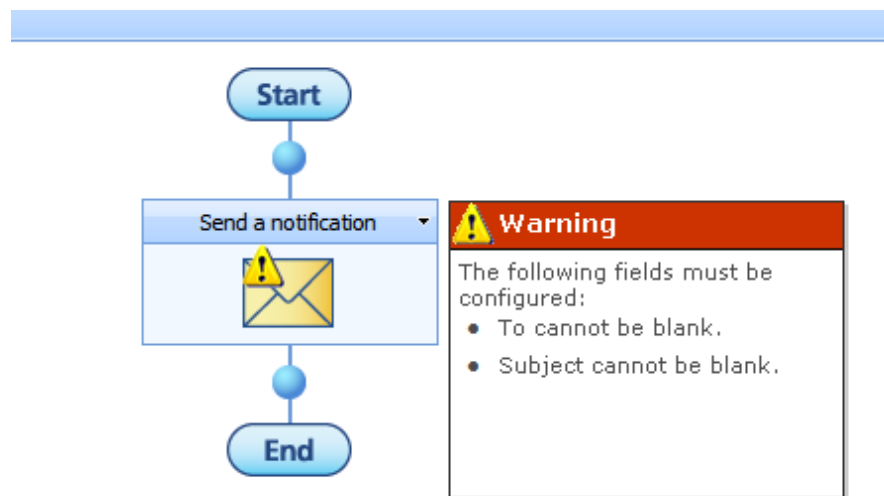


FIGURE 23. CONFIGURATION WARNING

To configure the action and correct any errors, click the title bar of your action, and notice the Configure item on the menu, as shown in Figure 24. You can also double-click the action icon to start the configuration.

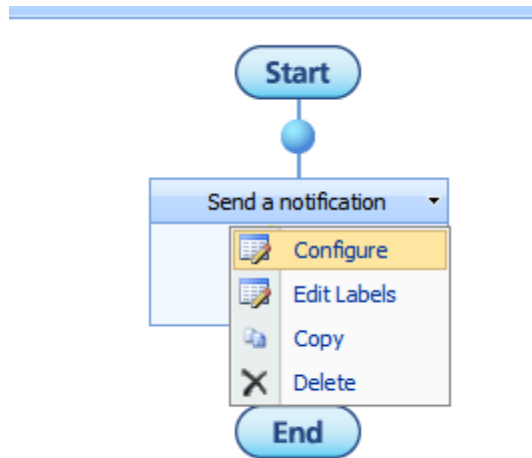


FIGURE 24. ACTION MENU

We will configure the action in just a moment, but first I want to take you through the other options here.

The Action menu allows you to perform cosmetic changes to the appearance of the action by using the edit labels. For example, you can change the title of the action, as well as add text to the left, right, and bottom of the action, as shown in Figure 25.

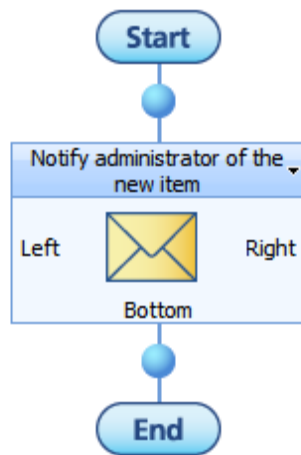


FIGURE 25. EDIT LABELS RESULT

This is most useful when you are using branching activities, where you can annotate the branches to better explain the branching logic.



Another mostly cosmetic change you can perform with the edit labels is the Expected duration. This option will tell the workflow how long you expect a certain activity to last. The sum of these expectations will yield the total expected run time for the entire workflow and is useful to give the administrator and others an idea of how long a workflow should take, especially in workflow reporting and management scenarios. This is only cosmetic and does not affect workflow execution.

You can also copy an action from the Action menu. This is very useful if you want to create similar actions, for example, in a branching activity. If you copy an action, you can paste the copy when clicking a blue pearl.

Finally, you can delete an activity from the Action menu. Doing so will also delete any child actions, for example, in a branching or looping activity.

#### **Note**

Nintex Workflow's UI makes heavy use of AJAX and scripting, and if your browser settings don't recognize the SharePoint server as belonging to the Local Intranet zone, you may not be able to see Action warnings or the Action menu. If this happens, one of the following should address the issue:

1. Add this site to your browser's Local Intranet zone.
2. Relax your browser's settings for the Internet zone to allow scripting.
3. Upgrade from Internet Explorer 7 to Internet Explorer 8, which seems to get less confused about such things.
4. Save your workflow and reopen it, which seems to fix the issues for existing actions. When you add new actions, though, the problem will come back until you again save and reopen the workflow.

Options 1 or 3 are best. Option 2 actually introduces a security hole, so I don't recommend it unless you're desperate.

### **Configuring Actions**

Clicking Configure from the Action menu will spawn a Configure Action dialog box. Every action has slightly different Configure Action dialog boxes, but once you get to know some of them, you will quickly recognize common dialog box options. The Configure Action dialog box for the "Send a notification" action is shown in Figure 26.

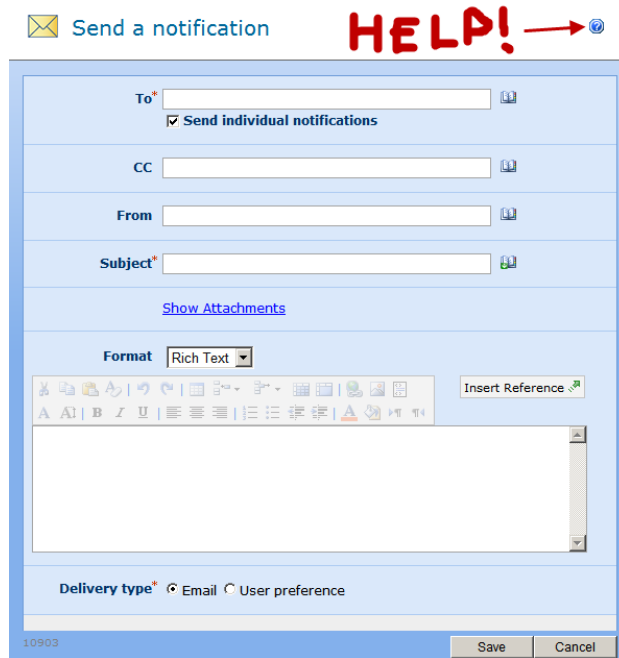



FIGURE 26. SEND A NOTIFICATION ACTION CONFIGURATION

If you get stuck in one of these dialog boxes, and chances are you will, you can click the Help button on the top right. Contrary to a lot of other software I have used, the help in NW is very good and gives you a good overview of how to use each configuration dialog box. Just remember to scroll down in the Help window, because most dialog boxes need a lengthy explanation for all the details.

### Note

I just saw *Cast Away* with Tom Hanks, as you can see from the annotation in Figure 26.

However, some elements are common to most dialog boxes, so I'll guide you through a few of these now.

The address book icon to the right of the To field () launches the lookup for that particular field type. In the case of an email address, you get a Select People and Groups lookup, as shown in Figure 26.

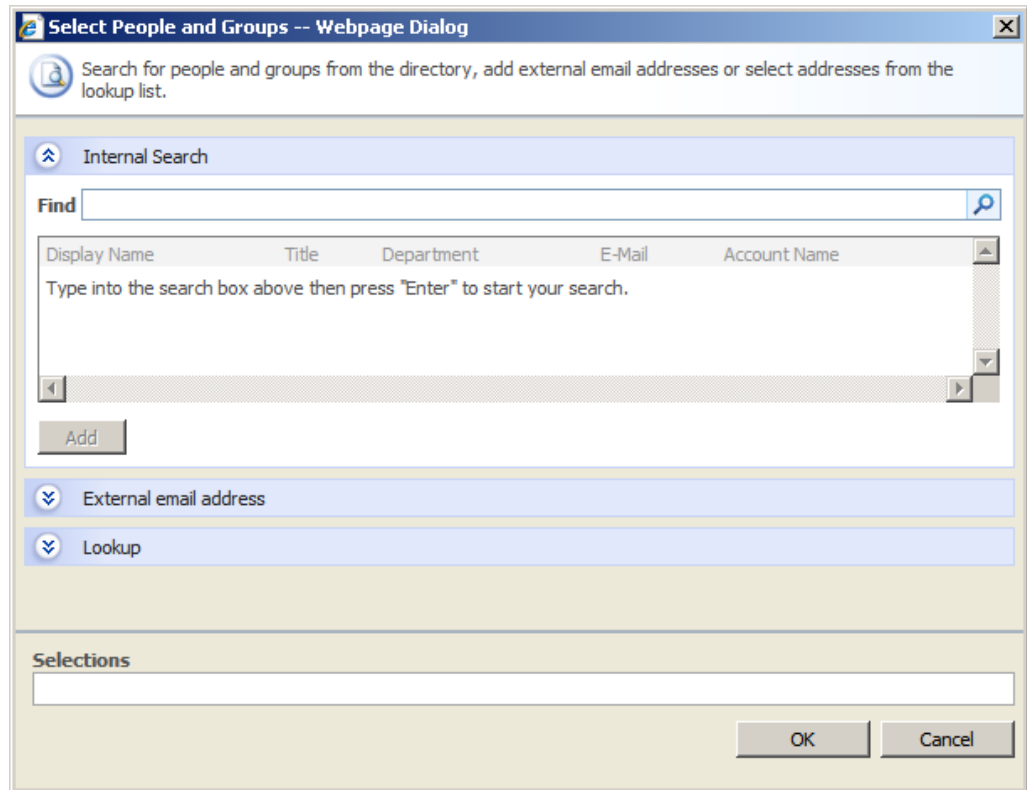


FIGURE 27. SELECT PEOPLE AND GROUPS LOOKUP

The Select People and Groups lookup dialog box has three sections. The first, Internal Search, searches through SharePoint users and groups and adds the users you select. The second, “External email address,” allows you to type in any email address and add it to the list.

The third section, Lookup, allows you to use values from the current item or any other lookup value to use as the value. This lookup is important, so I’ll take a moment to elaborate a bit on this particular feature.

### **Workflow Lookups in Nintex Workflow**

Workflow lookup is a powerful feature that allows you to use data available to the workflow as values in actions or messages. These lookups evaluate when the workflow runs, so their values are dynamic.

For example, you may want to use the current item’s Created value to send someone an email about when an item was created. In the lookup value selector, you will see the values available to the current action, including any workflow variables and constants defined, as shown in Figure 28.

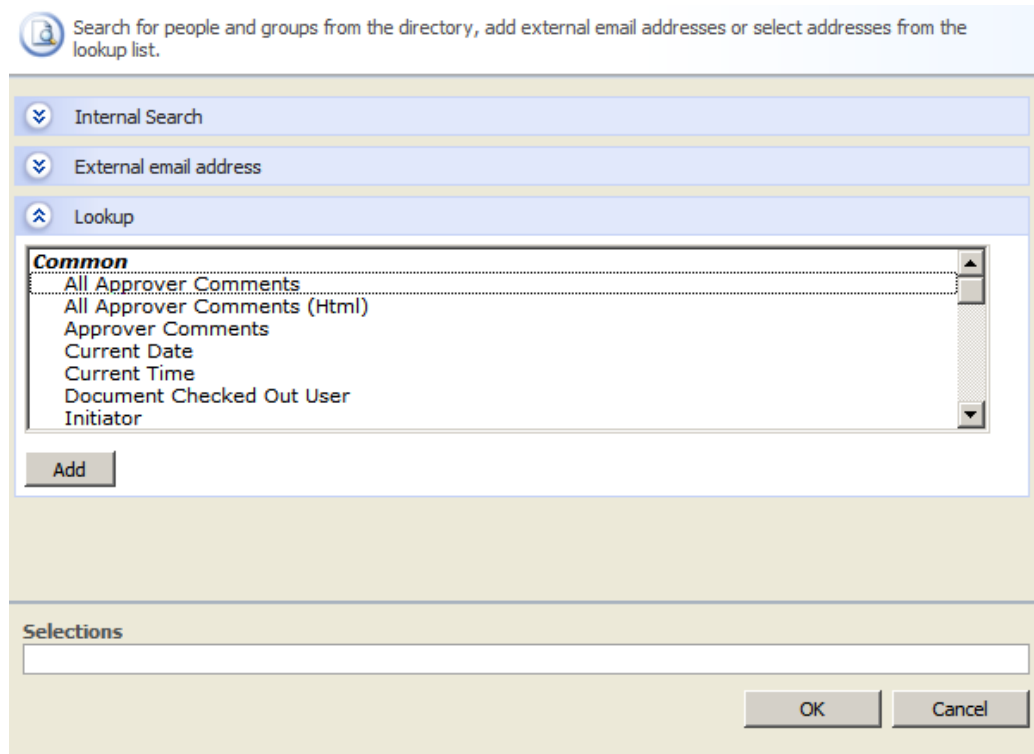


FIGURE 28. LOOKUP VALUE SELECTOR

Workflow lookups are incredibly powerful and allow you to find any value anywhere in the site and use that value as part of your workflow. You are not limited to information in the current item or the current context; any information in a site or even outside a site can be looked up.

For example, you may want to look up the To value of an approval task notification from a separate list containing managers or approvers for a particular department.

You accomplish this using a technique called *recursive lookup*. Recursive lookups use the result of one lookup to find the next lookup value. These recursive lookups can in theory be infinitely deep, allowing you to traverse multiple levels of references.

If you have worked with SharePoint Designer workflows, you are probably expecting me to say that recursive lookups work just like in SharePoint Designer. Sorry to disappoint you, but lookups do *not* work the same way in NW as in SharePoint Designer.

The workflow lookup feature in NW does not support inline recursive lookups. Rather, you need to look up the value outside the current action and assign the value to a workflow variable.

This is either good or bad, depending on your point of view. The proponents of “this is bad” will say that this is an added level of complexity, since you need an extra action and need a separate variable.

The proponents of “this is good” will say that storing a lookup value in a variable is always a good practice anyway because it allows for the reuse of the lookup value and also that you can have multiple actions access and build the lookup value. The result is greater flexibility and less work for complex workflows.

Personally, I don’t vote in either direction. I certainly appreciate the easy access for inline recursive lookups that SharePoint Designer provides, especially in simple and one-off workflows. However, the time I save in slightly more complex workflows and the greater flexibility and power I have with external recursive lookups is well worth the somewhat more complex authoring experience.

#### Note

If you want to learn more about recursive lookups in SharePoint Designer, you can pick up issue 4 of USP Journal from <http://www.understandingsharepoint.com/journal/volume-1/issue-4>.

For now, however, you can just select any recipient from any of the sections. Note that you can also add multiple recipients to the To field. Simply separate the multiple addresses with a semicolon.

The Cc and From fields work the same as the To field.

### Playing Around with Values

For the Subject field, however, I want to show you another cool feature of NW, which I mentioned this in the previous chapter. In the Subject field, type **Item created on** and follow it with a space. Then, leaving the text cursor after the last space, click the address book icon to start the lookup value selector. Scroll down to the Item Properties heading, and find the Created property.

Click Insert, and note that a reference to the Created property for the current item is added to the end of the subject.

Don’t want the Created reference at the end? Fine! Just click and drag the reference to wherever you want it in the subject.

You can use this form of inline string building in many other fields as well.

To do the same thing in SharePoint Designer, you would need to first build a dynamic string and then use that string as the single value in the Subject column. You still have that option in NW, but the inline string building is easily accessible.

 **Playing  
with  
Values**

Oh, but you haven't seen anything yet. Here's another cool trick, bordering on very advanced now, but I'll keep this very short.

Let's say you wanted to include the time elapsed since the document was created, as part of the subject. Now, without resorting to third-party extensions in SharePoint Designer, this would be incredibly difficult.

NW, however, supports what is known as *inline functions*. Inline functions are operations that NW performs on a value when the workflow runs. For example, you can perform calculation operations, modify text, format dates, and do a range of other functions. Not just that, but you, or someone who knows programming, can create separate functions that you can import into NW to extend the inline functions further.

For this example, add the following text after the red underlined Created reference in your subject:

```
(fn-DateDiffDays( , ) days ago)
```

Next, click the address book icon again to add a reference to the Current Date property. Repeat the process to add a reference to the Item Property value called Created, just like you initially did for the subject.

Finally, click and drag the Created reference just before the comma, and click and drag the Current Date reference to just after the comma. Your final result should resemble Figure 29.

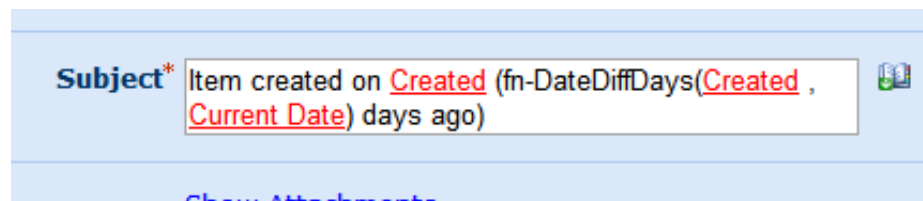


FIGURE 29. ADVANCED INLINE STRING BUILDING

Sadly, this makes for some very strange numbers. The fn-DateDiffDays function returns the number of days in a decimal format, as shown in Figure 30.

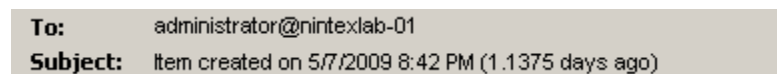


FIGURE 30. WHAT NUMBER OF DAYS?

The fix, however, is equally simple, and it shows off another nice feature of inline functions. You can nest the inline functions and use the result of one function as the parameter for another function.

Update your Subject line as such:

Item created on Created (fn-Round(fn-DateDiffDays(Created , Current Date)) days ago)


Now your workflow will produce a better result, as shown in Figure 31.

**To:** administrator@nintexlab-01  
**Subject:** Item created on 5/7/2009 8:42 PM (1 days ago)

FIGURE 31. AH, BETTER

This kind of inline string building can be done in any textbox that supports adding references.

**Note**

I won't go too deep into inline functions here and rather refer you to the help section called "Inline functions." You can access that help section by going to any help topic, clicking the Home icon at the top () , and then clicking Using the Workflow Designer and then "Inline functions."

The last thing I want you to notice in the Send a Notification dialog box is the message editor. There's nothing extremely special about it, really, but note the Insert Reference button. This button is available in many dialog boxes.

Although very similar to regular lookup buttons, one difference is important here. Click the Insert Reference button now, and notice that below the available lookup values is a checkbox named "Create a hyperlink" built from this data. Click that box now. Your dialog box should change into something resembling Figure 32.

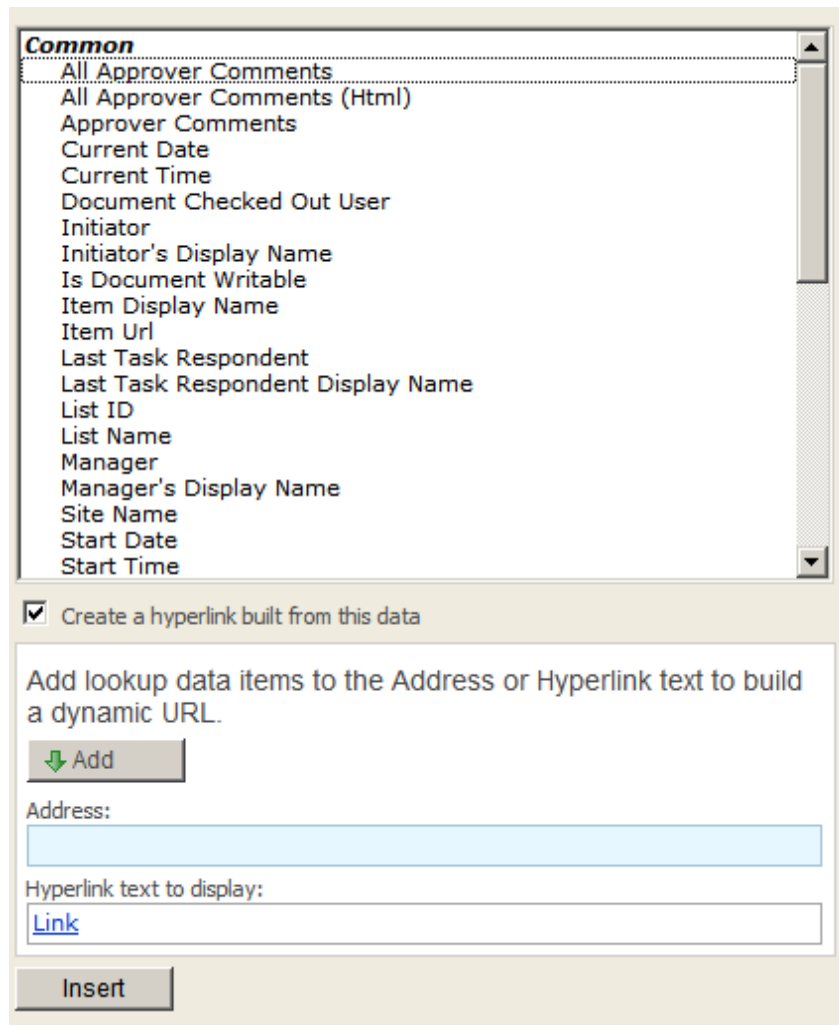


FIGURE 32. HYPERLINK EDITOR

In the hyperlink editor, you can use the lookup values inside the Address field to build a link to any text. For example, select the Item Url value from the lookup values, and click the Add button.

Next, change the “Hyperlink text to display” field to something like “Click here to access item.” Your hyperlink editor should now resemble Figure 33.



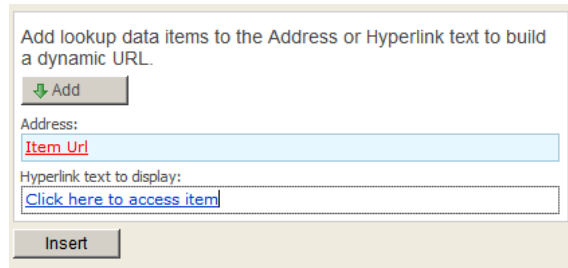


FIGURE 33. LINK TO ITEM

Click Insert, and your new link is added to the message area.

Finally, notice that you have the option of sending rich-text emails, not just plain text. The hyperlink editor for building links is available only when you are sending rich-text emails.

Phew, I think that was quite a bit of material, just to cover one simple dialog box for one simple action. However, there is a very good reason why I wanted to be this thorough.

The object of explaining this dialog box so thoroughly is that you have now learned about many of the common features of other Configure Action dialog boxes as well. I will spare you the excruciating detail when discussing other actions.

Still, we do have some ground to cover in this chapter, and important ground as well, so keep reading.

## **Saving and Publishing Your Workflow**

Now that you have configured your action, it is time to deploy and test the workflow. Yes, I realize this is an exceptionally simple workflow at this point, hardly worth a workflow at all, but bear with me, because there is a good reason for this. Also, we'll get more advanced as we move along.

In the Actions menu in the toolbar of your workflow designer, not to be confused with the Action menu of an action, you will find an option called Publish. Clicking this item will link the workflow to the list or library and make the workflow available for use.

Figure 34 shows the Actions menu.

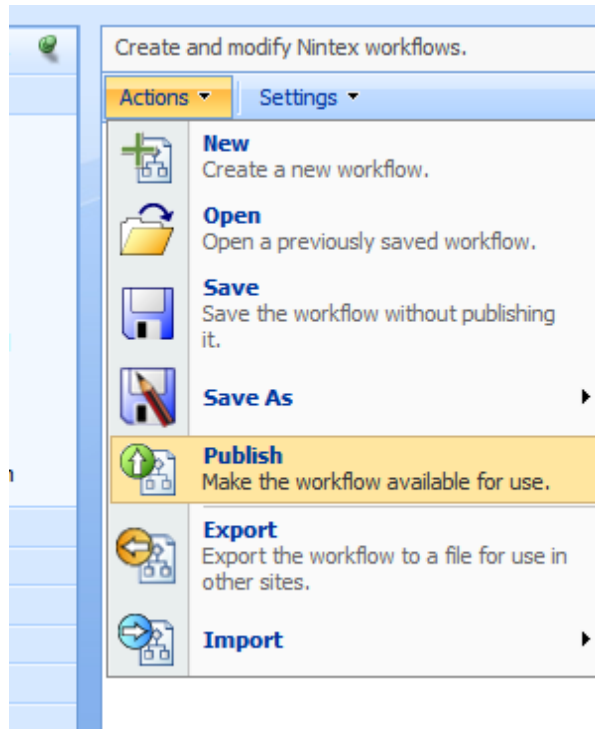


FIGURE 34. ACTIONS MENU

If you are not yet ready to make the workflow available, you can also save the workflow without publishing it. To do so, use the Save option.

The first time you use any of these two options, NW will ask you for a name and description for your workflow.

## Snippets and Templates

The Save As option, however, does not save your workflow. I know, this naming convention may be a bit confusing, but once you realize how cool this option really is, you'll forgive Nintex in a flash.

The Save As option gives you two child options, Save as Snippet and Save as Template.

 **Creating a Workflow Snippet**

The Save as Snippet option will save your current workflow into a single action item so that you can reuse your workflow as part of another workflow. At this point, we have created a very simple workflow that only notifies the administrator about a new item in a library. This may be useful in other situations as well, so select the Save as → Snippet option from the Actions menu. You will be asked for a name for your snippet. Call it something like Notify Administrator of New Item.

After the snippets save, click OK to close the message stating that saving was successful. Check the Workflow Actions menu, and you will now see a new section called My Snippets, as shown in Figure 34.

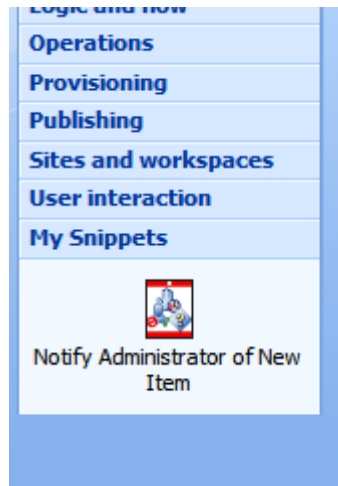


FIGURE 35. MY SNIPPETS

Now, whenever you want to notify the administrator of a new item, you can simply drag that snippet into the workflow designer surface and save yourself a lot of time.

Cool, eh?

You can do the same for more complex workflows, not only single action workflows as you just did. This allows you to create a toolbox of useful snippets and apply these rapidly.

NW has a special action that is particularly suited for creating such snippets. The “Action set” action, located in the “Logic and flow” category, even has a special item on the Action menu to allow you to save only the child actions of that action set, as shown in Figure 35.

**Tip**

The “Action set” action is incredibly useful if you are authoring workflows that might be used as snippets later. Of course, if you have not used the action set, you can always just add a new “Action set” action and then click and drag the actions you want to save as a snippet into the action set.

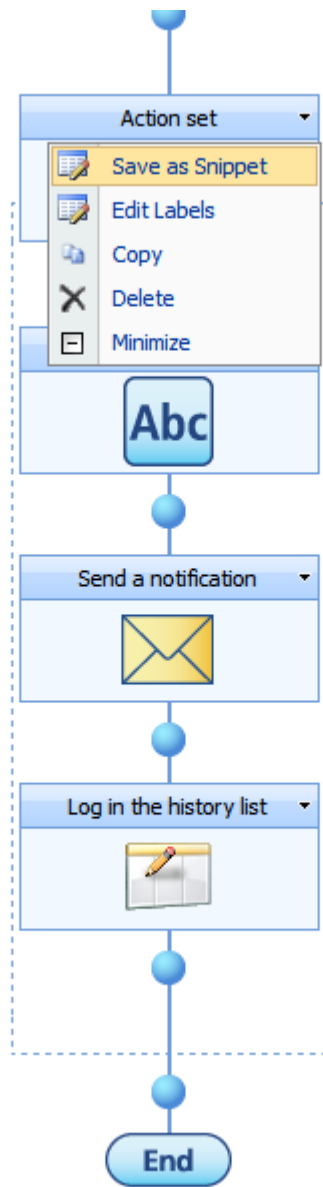


FIGURE 36. ACTION SET

You can also save a complete workflow into a template. Recall the template selector you see when you create a new workflow. You can select Save as → Template to create a new template for the template selector. You can even create new categories for your templates.

Using the template option, you can allow users to create similar workflows, specific to your organization, with a minimum of effort.

## Workflow Variables

During the course of a normal workflow, you will want to access and store values between different actions. You have two tools for this: workflow variables and workflow constants.

### Variables vs. Constants

In the previous chapter, I explained *workflow constants*; these are values that are set outside the workflow and can be used in any workflow within the scope of the constant.

Inside a workflow, however, you have access to *workflow variables*. As for constants, these are values that you can use inside your workflows.

The vital difference here is that variables live inside only one workflow. Also, variables are dynamic, meaning you can both set and change them during the course of the workflow.

Constants exist across all workflows inside the scope of a constant. Also, constants do not change during a workflow run—at least they will not change because of what a workflow does.

### Defining New Variables

You define workflow variables by going to the Settings menu of your workflow and selecting the Workflow Variables option. When you start with a blank workflow, no variables exist when you first begin to author the workflow.

The only complex thing about workflow variables is the data type of the variable. Most of the data types should be familiar, but the Action ID and Collection data types require a bit of explanation.

The Action ID data type is used to store a reference to a human interaction action. For example, when you assign a task of approving something to a user, you can store a reference to that task in a variable of type Action ID.

Storing such a reference is very useful for interacting with that task. For example, you may want to remind someone of a task they have been assigned, or you may want to escalate a task if it has not been performed within a certain time frame.

We will look into working with tasks in later chapters of this issue.

The other variable type that requires explanation is Collection. A *collection* is just that, a group of values. If you have worked with programming before, think of a collection as an array.

If you haven't work with programming, however, the concept needs a bit more explanation.

Some actions in NW can loop through several items. A common action for looping is the "For each" action that performs an action or set of actions for each value in a collection.

In the "For each" action configuration, you need to supply a collection that the action will iterate. Each of the items in the collection is then stored in a separate variable you can access inside the "For each" loop. Figure 37 shows a typical "For each" action configuration.

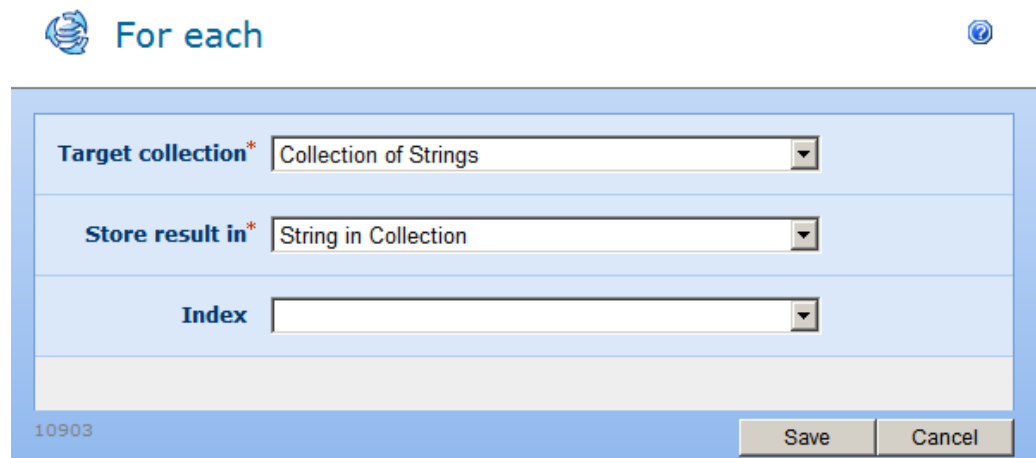


FIGURE 37. "FOR EACH" ACTION CONFIGURATION

Inside the "For each" look action, you can then access each of the strings in the collection named Collection of Strings through the variable String in Collection. For example, you can create a list item for each of the strings in the collection, as shown in Figure 38.

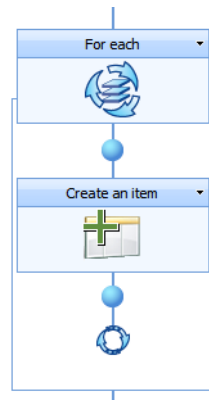


FIGURE 38. "FOR EACH" LOOP

The “Create an item” action in this loop uses a workflow lookup to set the Title property of each new item to the String in Collection workflow variable, thus creating a new item for each string in the collection.

**Note**

Values stored in collections can, in theory, be any data type. You must ensure that the collection actually contains the expected data type.

Collections have a special action designed to perform collection operations. Using the “Collection operation” action, you can manipulate a collection in many different ways. I strongly suggest looking at the help section for the complete list, although we will use the “Collection operation” action later in this issue.

One example of a “Collection operation” action is to add responses to a “Collect data” action to a collection. You can ask the people in the office to provide a greeting for a “get well” card workflow and then use a collection operation to append those greetings to a single message sent to whomever needs to get better.

Who said workflows had to be serious all the time? Before that, though, we need a few more tools in our belt, so in the next chapter, I will show you some more basic tasks that you should know.

First, here are some review questions to make sure you understand the core concepts of this chapter.

 REVIEW QUESTIONS

- ❓ 1. In addition to drag and drop, what are your options for adding actions to the Workflow Designer tool?
- ❓ 2. How should you configure a recursive lookup in Nintex Workflow?
- ❓ 3. What is the difference between a workflow variable and a workflow constant?

 REVIEW QUESTIONS ANSWERED

- ! 1. In addition to drag and drop, you can also click the blue pearls and select the action from the menu.
- ! 2. In Nintex Workflow, you need to look up a value outside the current action and store that value in a workflow variable.
- ! 3. A workflow variable lives only within a given workflow and can be changed inside that workflow, while a workflow constant is available to all workflows within its scope and cannot be changed from within a workflow.



## Sample Tasks

*Examples of how to use Nintex for common, everyday tasks.*

We have now covered the basics of using the Nintex Workflow Designer, so I think we should take this to the next level and start building slightly more complex workflows.

In this chapter, I will show you some smaller workflows that introduce you to various aspects and actions of NW. These will cover several of the tools and features you need to compose larger and more complex workflows.

### Approval

A very common workflow task is to request approval for something. For example, an employee may ask a manager for approval of vacation time, expenses, travel plans, or purchases. The task is so common that MOSS even includes a precreated Approval workflow that you can use for various approval scenarios.

However, what happens if the MOSS Approval workflow does not match your needs? You may want to set a deadline for approvals and perform some action if the approval is not granted within that deadline. Or, you may want to remind the assignee of the approval request after a certain period. Or, you may want to perform some other action as part of the approval request, for example, to gather responses from other people before submitting a request on behalf of a group.

Good as it may be, the default Approval workflow in MOSS is not very flexible, and you would need to author your own workflow from scratch if you wanted to change anything.

### Note

You will not win any prizes for guessing what I am about to show you.

In this exercise, I will show you how to work with such approval requests in NW. The task should be familiar, so we can focus on learning the benefits that NW brings to the table.

We will then expand this simple approval workflow to include some logic that is often related to such requests.

### Creating a Simple Approval Workflow

Start by creating a new workflow for any list, for example, the Shared Documents library. Recall that you create a new workflow from the Settings menu of the list or library in which you want the workflow created.

From the “Commonly used” category of the Workflow Actions menu, drag the “Request approval” action to the first blue pearl, as shown in Figure 39.

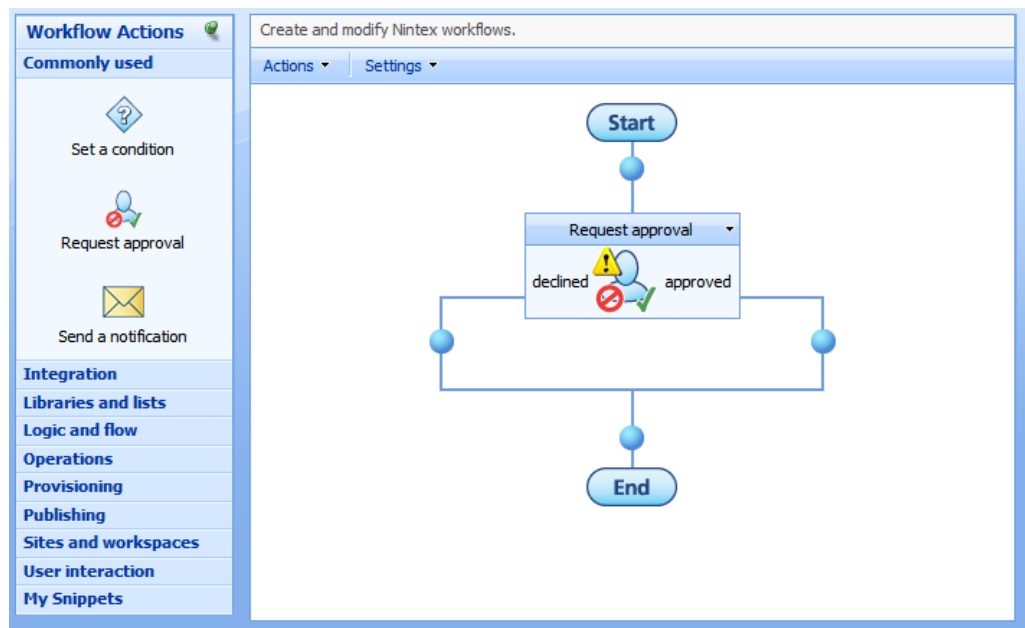


FIGURE 39. “REQUEST APPROVAL” ACTION ADDED

Next, click the title bar of the action, and select Configure from the menu. You can also double-click the action icon to configure the action.

In the Approvers field, either type the username of the person who should approve the request or click the address book button to search for the username. I have used Administrator as my approver.

We will keep the extremely simple for now, so just scroll down and hit Save to complete configuring the action.

Now, publish your workflow, and name it something like **Simple approval**. That's it. You're done. You have created a simple approval request workflow, albeit too simple to be of any practical use but still an approval workflow.

If you have worked with other approval workflows before, the next exercise is entirely optional, so feel free to skip ahead a couple of pages. We are going to go through how to test the workflow and understand how the approval workflow works.

 **Testing the Workflow**

Go to the list or library where you created the workflow, and either create or upload an item or document. Next, click the item drop-down menu and then Workflows, as shown in Figure 40.

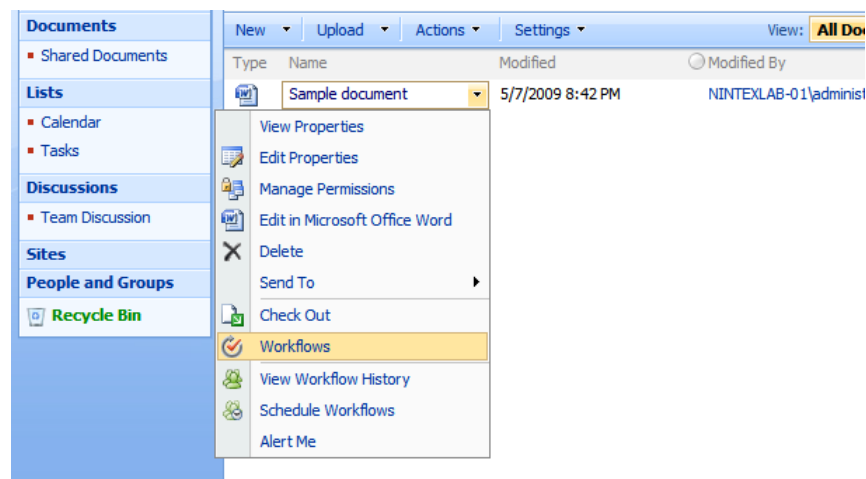


FIGURE 40. WORKFLOWS IN DOCUMENT DROP-DOWN MENU

Click the “Simple approval” workflow to launch the workflow initiation form, and then hit Start to launch the workflow.

Your list or library view will now include a column for the new workflow, and the column will include the workflow status or result, as shown in Figure 40.

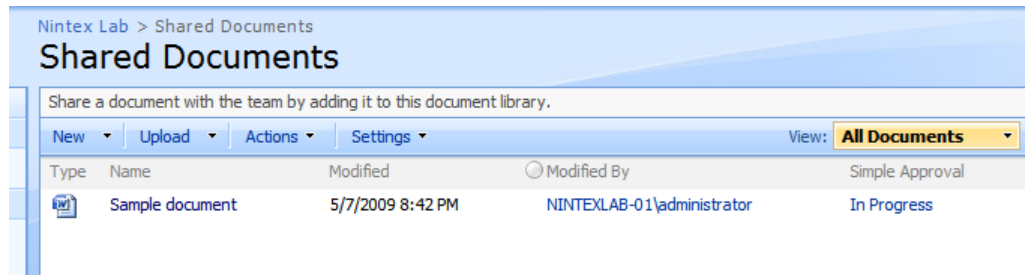


FIGURE 41. WORKFLOW COLUMN ADDED

You can click the In Progress text of the workflow column to get an overview of the workflow status. On that page, you will see a task assigned to the Administrator or to whomever you chose as the approver for the workflow.

To complete a task, you need to edit the task. You can do this directly from the workflow status page by opening the item menu, as shown in Figure 42.

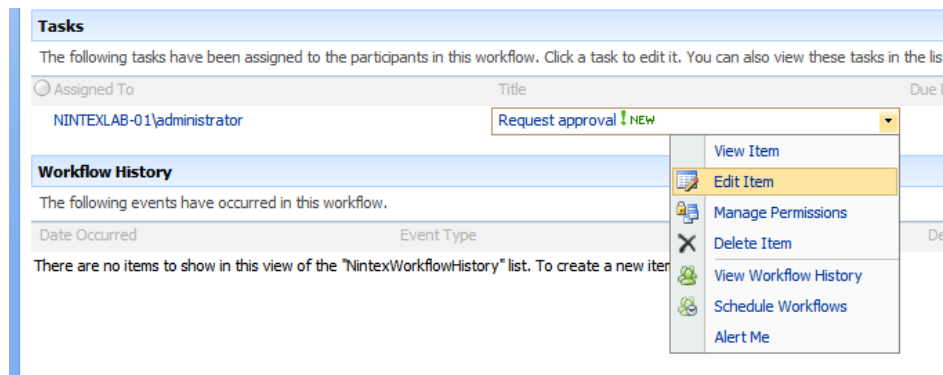


FIGURE 42. EDIT TASK ITEM

You can also open the task list used by the workflow. However, contrary to what is common in SharePoint Designer workflows, the default task list is not Tasks but a list called Workflow Tasks. You can, if you want, change the default tasks list for NW by going to Central Administration and then to the Global settings for Nintex Workflow Management on the Application Management page.

You are not bound to editing the task from the task list, however. In Nintex Workflow, you have several options for reaching the task page to complete the task. You don't even need to go to the task page; if you have set up LazyApproval, the user can simply reply to the email sent with one of the recognized words or phrases in order to respond to the task.

Another option is to add a web part that comes as part of Nintex Workflow, called My Workflow Tasks. The My Workflow Tasks (MWT) gives users a quick overview of the tasks assigned to them. However, contrary to the regular task lists, the MWT web part can be configured to show tasks from anywhere in the site collection or, if you have the enterprise license, even for the entire farm.

**Note**

Before adding the MWT web part, you must activate the site collection feature Nintex Workflow 2007 Web Parts.

Figure 43 shows the MWT web part.

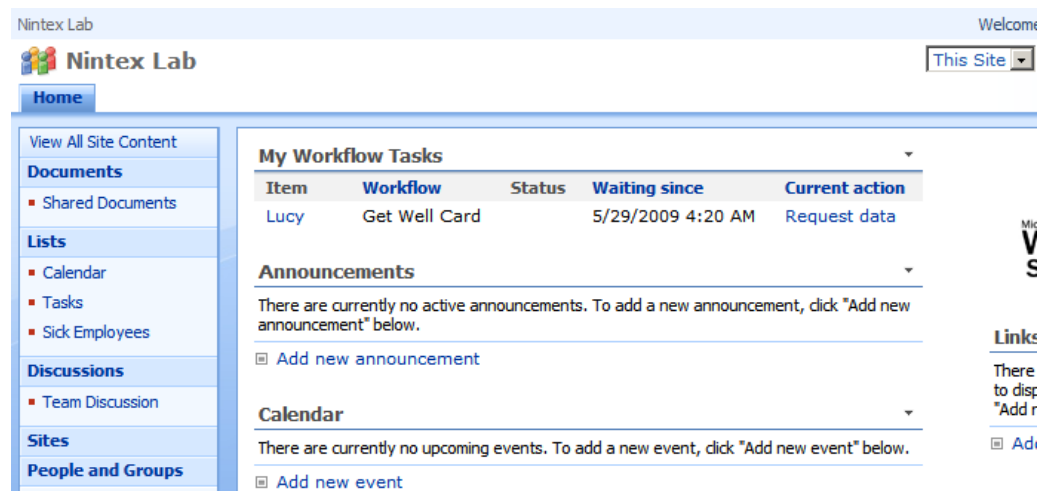


FIGURE 43. MY WORKFLOW TASKS

Yes, I know it says 4:20 a.m. I like worms, and according to some, that’s what early birds get.

You can configure the MWT web part like any other web part by clicking the Modify web part from the web part menu. This will allow you to set options such as the scope from which the web part will show tasks, how many tasks should appear per page of the web part, and several useful display options. For example, you may want to include the name of the user who initiated the workflow as well as for how long the workflow has run thus far.

Clicking the Current action link will also take you to the workflow information page. We will explore this page later in the chapter, but you should notice the link in the toolbar called “Respond to task.” This is a link to the task page where you can complete the task assigned.

No matter how you go about editing the task, you will next go to the task page, shown in Figure 44. This page will allow the assignee to approve or deny the request, as well as provide comments.

Use this page to approve or reject submissions. Note that rejecting an item does not delete it. [Learn about requiring approval.](#)

**Status**  
Approve / reject the item.

Approved  
 Rejected

Or you can [delegate](#) this task to another person.

**Comment**  
Use this field to enter any comments about why the item was approved or rejected.

**Item Properties**  
The following properties have been set for this item.

**Item:** [Sample document](#)  
**Workflow status:** [View](#)  
**Title:**

Ok Cancel

FIGURE 44. EDIT TASK PAGE

Notice that there is a link to delegate the task. In NW, the assignee has the option of assigning or delegating a task to someone else. Not just that, but you can also set a task to be automatically delegated if it is not completed within a certain deadline. I'll show you how in a moment, but first approve the task and note that the workflow column in the list or library now says Completed.

At this point, you have created a very simple approval workflow. The workflow does not do anything except ask for approval, regardless of whether the request is approved or denied.

Also, the assignee is set to a static user, and that is not very flexible. Let's see whether we can improve the workflow and add some flexibility.

### Using Start Data Variables

In this exercise, I will show you how to use initiation or startup variables to ask the initiator for information about the workflow. We will ask the user to select the person who should receive the approval request.

 **Assigning a Dynamic Approver**

First, go back into the workflow by going to the list or library, and choose Manage Workflows from the Settings menu. Click your "Simple approval" workflow to open the Workflow Designer with your workflow loaded.

In the Workflow Designer, click Settings and then Start Data. The dialog box that opens will allow you to specify data that the user should enter when the workflow starts.

Enter a name for the start data, such as Approver, and then click Person or Group as the type. Notice that the dialog box changes slightly to allow you to specify further data for the variable, such as the default user, from which groups the user should be able to select, as well as whether to allow selection of SharePoint groups, security groups, or just individual users.

Figure 45 shows my dialog box as I have configured it.

Hit Add, and your dialog box should now list the single variable you have created, as well as options for adding more variables. You can also click the red button in front of your variable name to delete a variable.

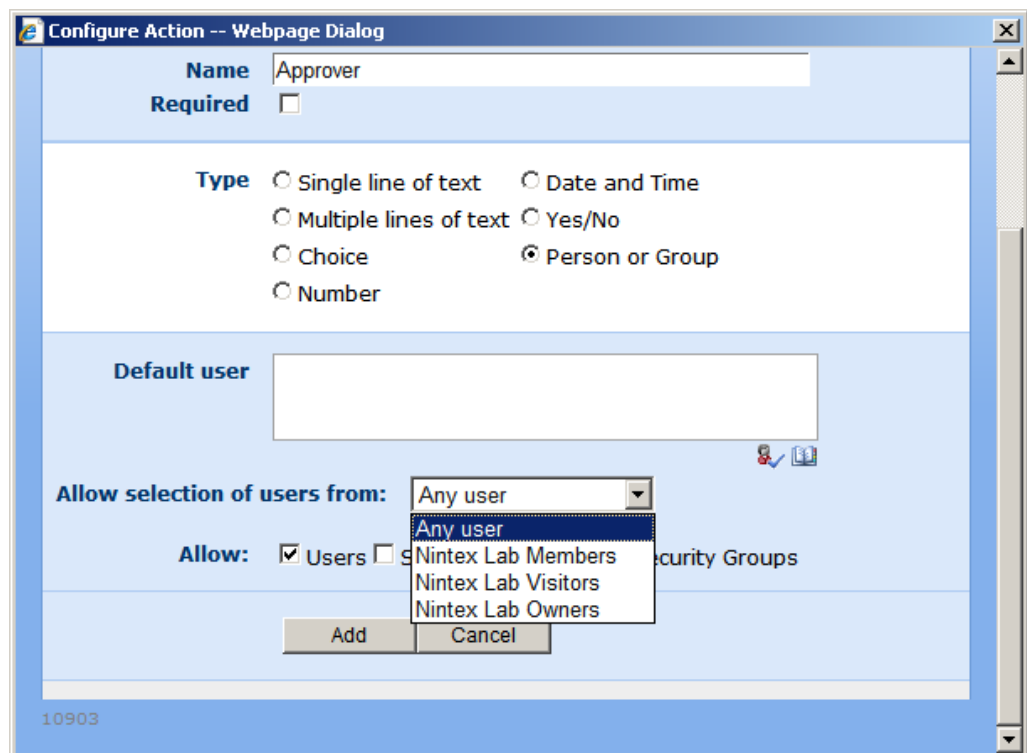


FIGURE 45. PERSON OR GROUP START VARIABLE

I want you to add one more start data variable that we will use in the “Request approval” action. Add a new start data variable called “Request detail” of type “Multiple lines of text.” Click Add to add the new variable, and finally click Save in the Start Data dialog box.

### Caution

You must hit the Save button in the Start Data dialog box whenever you change, add, or delete start data variables. If you simply close the window, your change will be lost!

### Caution

To use the selections the user has made, double-click Request Approval to configure the action.

First, click the address book icon next to the Approvers field to launch the lookup dialog box. Go to the Lookup section, and scroll to the very bottom where you will see your two start data variables listed in the Workflow Variables section. Click Approver and then Add and finally OK, as shown in Figure 46.

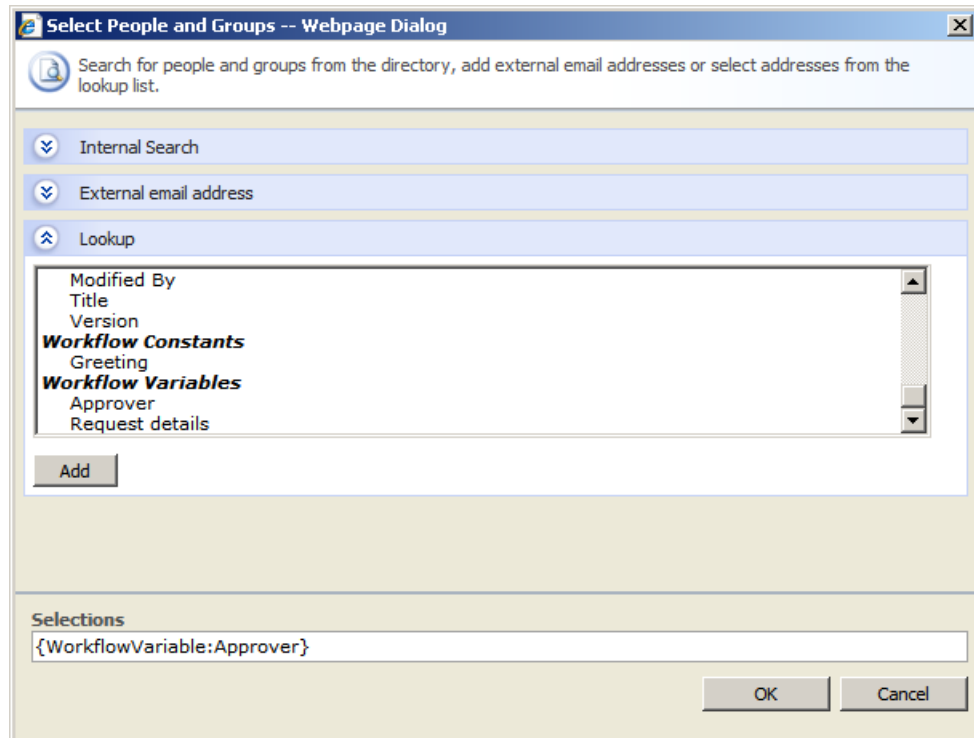


FIGURE 46. SELECTING THE APPROVER WORKFLOW VARIABLE

Next, open the Edit Task Description panel of the Configure Action dialog box. Repeat the previous step to add the request details using the Insert Reference button.

Finally, save the new action configuration, and feel free to publish and test your workflow. When you do, notice that the initiation form for starting the workflow now has two fields for users to configure, as shown in Figure 47.



FIGURE 47. START DATA INITIATION FORM

The task assigned will now include the request details, as specified by the user, in the description at the top of the page, as shown in Figure 48.

FIGURE 48. REQUEST DETAILS.

Great, one small step for your workflow, and, frankly, not that great a leap for mankind, but at least we have a more flexible workflow now.

We are still not acting on the result of the request, so let's at least log whether the request is approved or denied.

**Working with Branches**

In your workflow designer, open the “Lists and libraries” category in the Workflow Actions menu, and drag a “Log in the history list” action into each of the blue pearls that emanate from the “Request approval” action.

Double-click each of the newly added “Log in the history list” actions, and add some appropriate text to log, for example “Request approved” to the approved action and “Request denied” or “Not in this lifetime” for the declined action.

### Note

You can add multiple actions in each of the branches of the Request approval. For a much more useful example, you may want to send a notification back to the initiator with the comments and result of the request.

I will leave that as an exercise for you to perform on your own.

### Automatically Delegating a Task

However, I want to show you something even cooler with branching. As I mentioned earlier, you can have Nintex automatically delegate the task if the request has not been completed within a deadline.

To accomplish this, we need to use a “Run parallel actions” action in a Nintex workflow. The “Run parallel actions” action allows you to perform multiple actions at the same time in two or more branches. In this case, we want to have one branch request the approval and have another branch delegate the task after a certain period.

First, go to the “Logic and flow” category in the Workflow Actions menu, and drag a “Run parallel actions” action to the first pearl of your workflow. You can also click the first pearl and select the action from the menu. When you add the “Run parallel actions” action, the existing “Request approval” action will shift down.

The “Run parallel actions” action has two branches by default. If you want to add more actions, you can do so from the action menu, or you can click the title of any of the existing branches to add new branches to the left or right of that branch. You can also delete existing branches in this manner.

Now, one branch should run the “Request approval” action we created earlier. However, you don’t need to re-create the entire action. Instead, click and drag the main icon of the “Request approval” action into the left branch of the “Run parallel actions” action. When you do, the entire “Request approval” action, including the child “Log in the history list” actions, is moved into the correct branch, as shown in Figure 49.

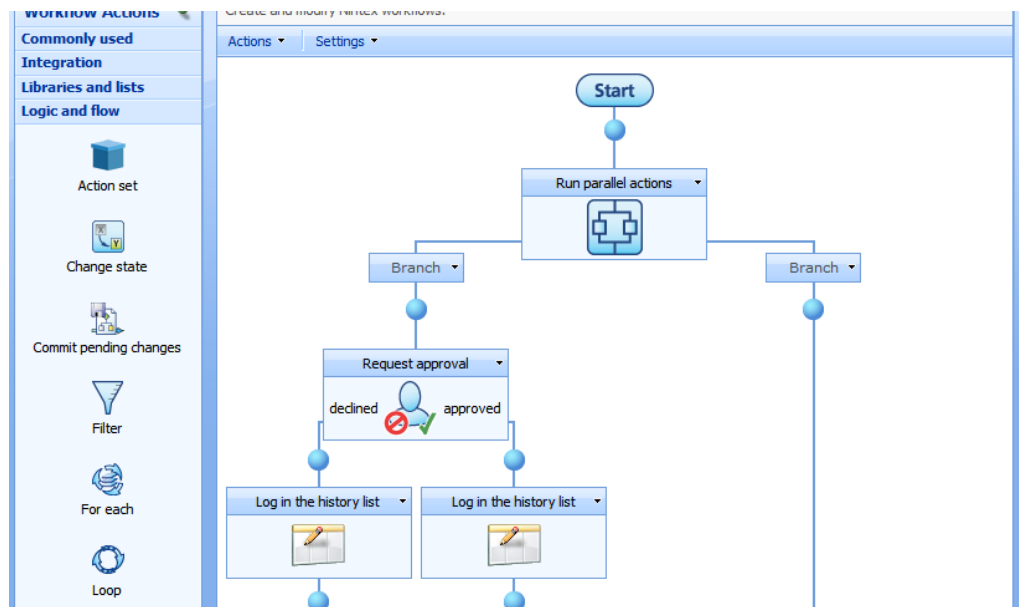


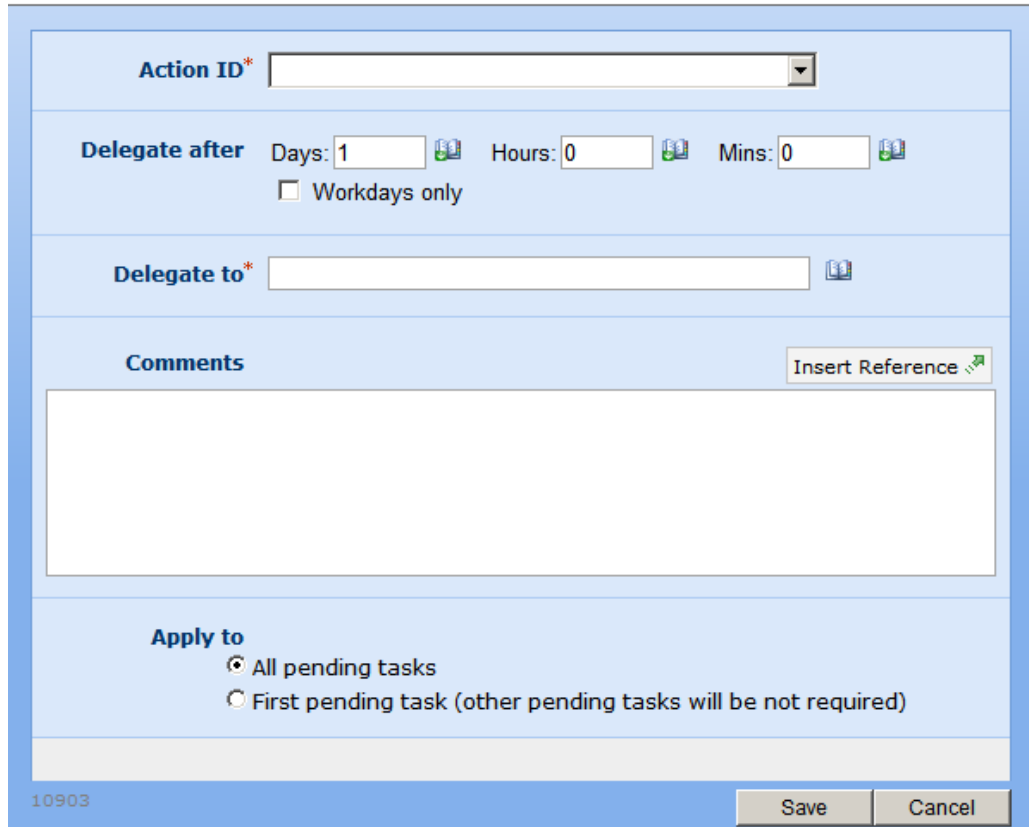
FIGURE 49. FIRST BRANCH DONE

The problem with actions that create tasks such as the “Request approval” workflow is that the workflow will stop completely until that task completes. However, by putting the action inside a separate branch, we can allow the parallel branch to operate normally.

The parallel branch is where we will add our delegate task action. Since this branch runs independently from the first branch, we can perform some task maintenance here without waiting for the first action to complete.


Open the “User interaction” category in the Workflow Actions menu. Then, drag a “Delegate workflow task” action into the second branch, and then double-click the action to start the configuration, as shown in Figure 50.

## Delegate workflow task



**Action ID\***

**Delegate after** Days:  Hours:  Mins:   
 Workdays only

**Delegate to\***  

**Comments**

**Apply to**

- All pending tasks
- First pending task (other pending tasks will be not required)

10903

FIGURE 50. DELEGATE WORKFLOW TASK CONFIGURATION

Now, you'll see that two fields are required, the Action ID and the "Delegate to" box. The "Delegate to" box should be self-explanatory and contains the user to which the task should be delegated. Set yourself or the administrator in the "Delegate to" box for this test.

The Action ID is a bit more complex, but we will get to that in a moment; I just want to explain the logic here first.

Notice the "Delegate after" section, containing fields for days, hours, and minutes. The period you define here will be the wait time before the task is delegated. For this exercise, feel free to set the minutes to 1 to test.

However, note that the time count is not exact. The reason is that time tracking in workflows is controlled by a SharePoint timer job. That timer job runs only every five minutes, meaning that any time will be one to four minutes off.

Now, since we create one branch that asks for approval and one branch for delegating, these actions happen at the same time. The approval request is sent, and the countdown to the delegation of the task begins.

If the task is not completed by the time the delegation happens, the task is assigned to someone else, and the delegate branch completes.

If the task is completed, however, then the task branch completes, and the delegation will not happen.

There's a catch with this approach, though, and I'll explain that shortly.

Let's get back to that Action ID. NW needs to know which task it should delegate. You may have multiple tasks waiting, and thus you need some method for tracking the various tasks.

This is done using an Action ID variable. I'll show you how to create one, but first click Save to save your changes.

Next, create a new workflow variable of type Action ID, and name it something like "Request approval" task. Remember, you create new workflow variables from the Settings menu.

Then, double-click your "Request approval" action to open the configuration screen again. At the very bottom, you will see a drop-down called "Store Action ID in." From that drop-down, select your newly created workflow variable.

Now you have identified the task in your workflow and can tell the "Delegate task" action about the correct task. Double-click the "Delegate workflow task" action again, and select the correct Action ID variable from the drop-down menu. Hit Save, and make sure your workflow now resembles Figure 51.

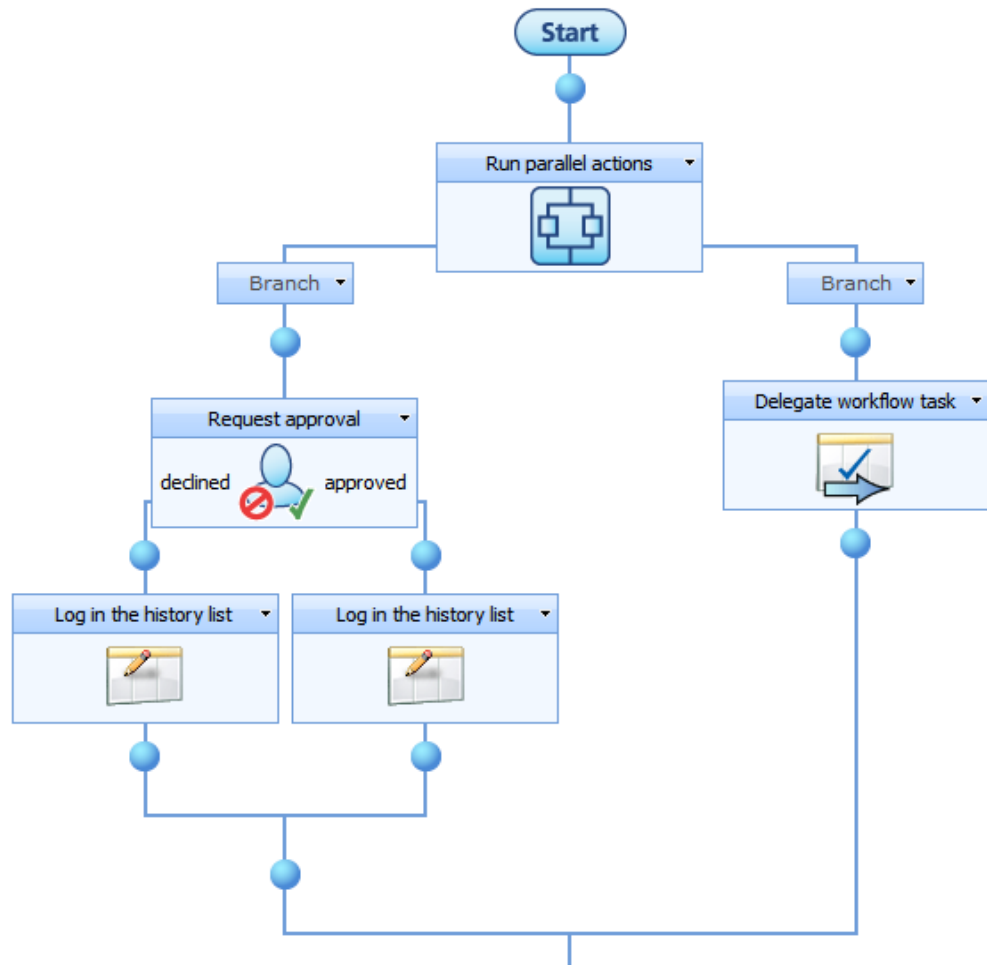


FIGURE 51. WORKFLOW COMPLETE

Great, you're done, and this works very well. However, I promised you a catch. Here it is: both branches must complete before the workflow moves on.

You may wonder why that is a catch, but consider this: if you set the timeout for the task delegation to one day and the request is approved or denied within five minutes, the delegate branch must still wait for the end of that day before the rest of the workflow completes.

Oh, and if the task is not completed by the time it is delegated, the new assignee must still complete the task before the workflow moves on.

In Figure 52, I have added a few "Log to history" actions to let you know at which time the various events happens. Examine the workflow history, and notice that even though the task is completed at 9:05 p.m., the workflow does not complete until five minutes later, after waiting for the delegate task action to happen.

Workflow History				
The following events have occurred in this workflow.				
Date Occurred	Event Type	User ID	Description	Outcome
5/11/2009 9:05 PM	Comment		Started at 9:05 PM	
5/11/2009 9:05 PM	Task Completed	NINTEXLAB-01\administrator	(NINTEXLAB-01\administrator)	Approved
5/11/2009 9:05 PM	Comment		Approved at 9:05 PM	
5/11/2009 9:10 PM	Comment		Task delegate at 9:10 PM	
5/11/2009 9:10 PM	Comment		Done at 9:10 PM	

FIGURE 52. DELEGATION TASK HISTORY

This is an inconvenience for the most part, because you can simply continue your workflow in the task branch while the delegate branch waits, but you need to be aware of the issue. If you add the rest of your workflow outside the branching action, you will be “stuck” until the timeout for the delegate branch is passed.

Probably a much worse effect of this is what happens if the task is indeed delegated automatically. When that happens, the task branch will keep waiting for the new assignee to complete the task indefinitely.

So, although the automatic delegation of tasks is a very powerful feature, and sorely missed in SharePoint Designer workflows, pay attention to your workflow architecture, or you may get more problems than you bargained for.

### Following the Execution Path of a Workflow

Let’s talk about one more thing to learn before we move on, which is very cool. In the previous workflow history example, I added “Log to history” actions to show you the path the execution of the workflow takes.

However, there is a hidden gem in NW that really makes the whole tracking business much better: the Nintex workflow status.

I’m saying *hidden* here, because, really, the Nintex workflow status is hidden in plain sight, cleverly disguised behind the menu option View Workflow History on the item context menu, shown in Figure 53.

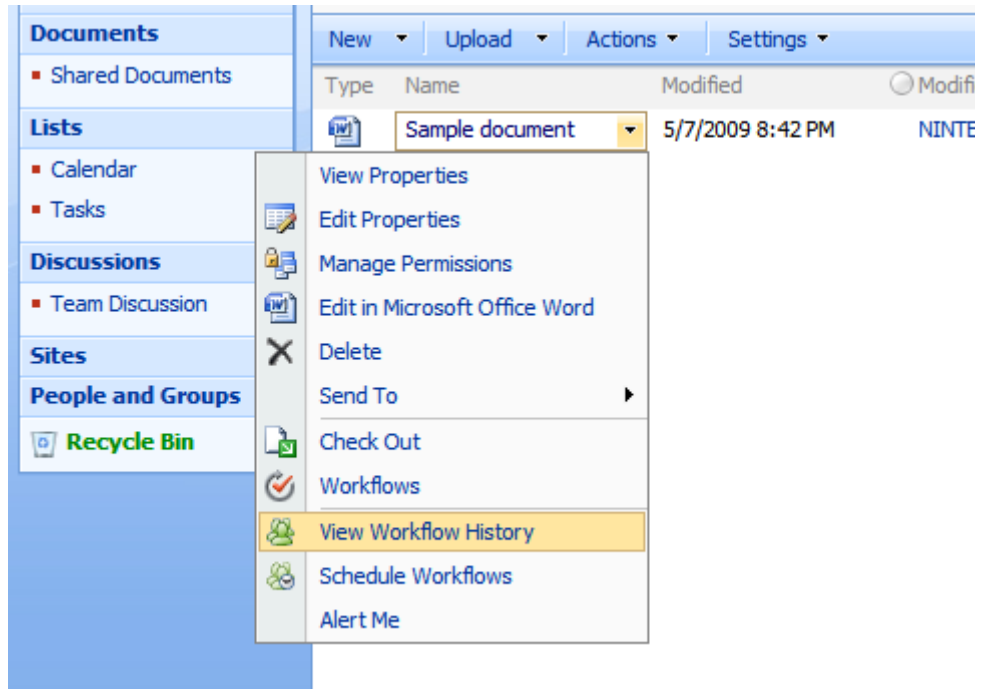


FIGURE 53. VIEW WORKFLOW HISTORY. OR IS IT?

When you click this menu option, you are sent to a list of all the workflow executions for your item, including any running, completed, or failed workflows. To uncover the true power here, click any of the workflows, and behold a visual representation of the workflow execution, as shown in Figure 54.

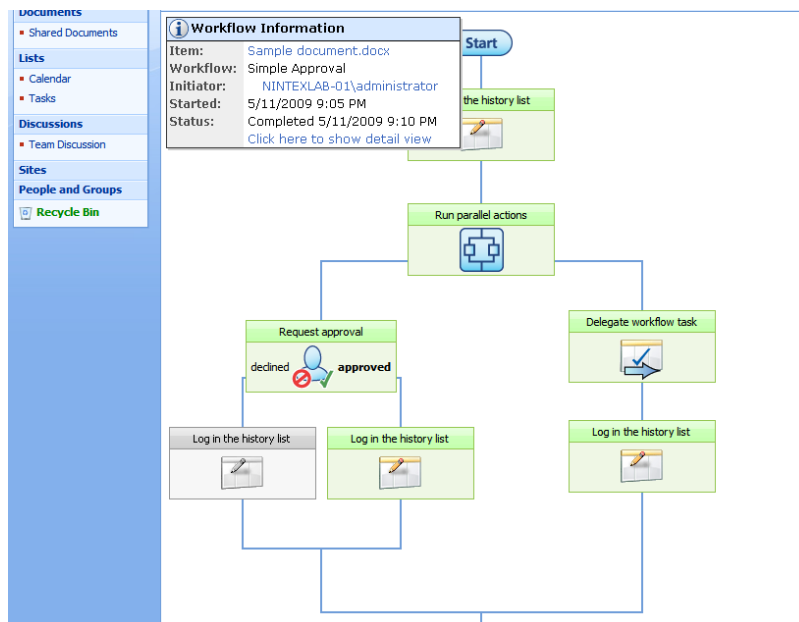


FIGURE 54. WORKFLOW EXECUTION DETAILS



In the workflow execution path, green actions have completed, while gray have not executed, either because of conditions not met or from the workflow not progressing to that action yet.

The really cool thing is what happens during workflow execution. In Figure 55, I have started the simple approval workflow but haven't approved or denied the request yet. As you can see, the actions that are currently executing, including the branch action, are now shown in yellow.

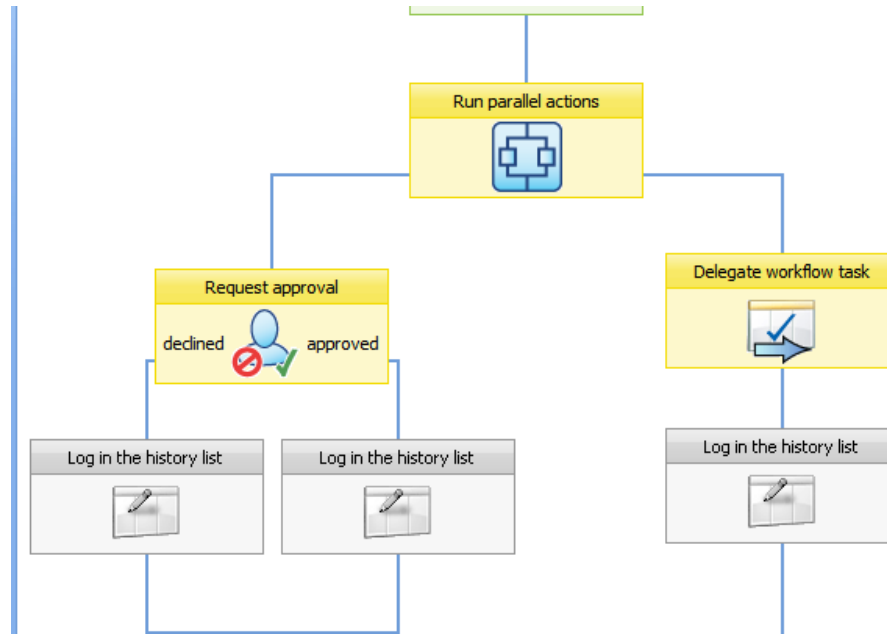


FIGURE 55. EXECUTING WORKFLOW

Tracking how a workflow executes just got a whole lot easier. There's no more need for copious amounts of logging and guesswork as to whether a condition or branch is configured correctly.

And, since we are already in the monitoring mood, let's check out the workflow statistics. The workflow statistics are really well hidden, but once you know where to find them, you'll be glad you know.

Go to the Site Settings page for your site, and click the "Workflow gallery" link in the Nintex Workflow section. At the top of the page, you will find a "Workflow statistics" link. Click that link, and notice that the page slightly only changes. Click the "Simple approval" workflow, and take a look at the statistics available for your workflow.

Figure 56 shows my result. I have cut the result slightly, because the statistics page can get quite long.

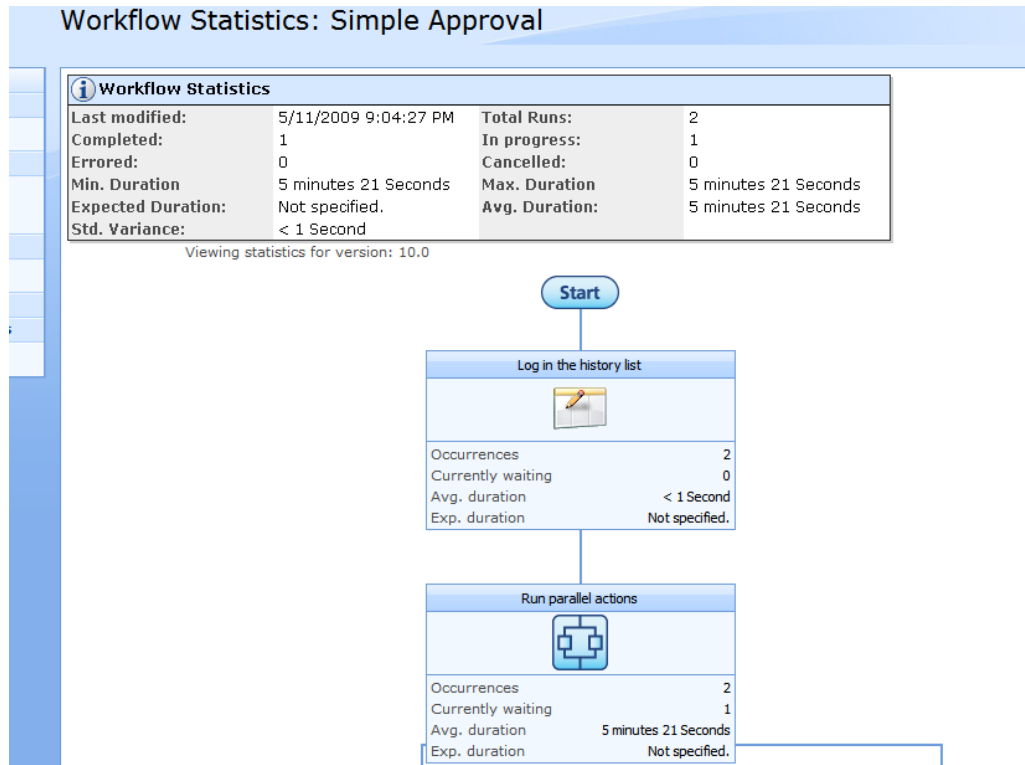


FIGURE 56. WORKFLOW STATISTICS

The Workflow Statistics page gives you a very good overview of how your workflow behaves, how long various tasks take, how many times each workflow and action has run, and other information.

Some information is also action-specific. For example, the “Request approval” action shows you how many times each user has approved or declined the request as well as how fast each user normally responds, as shown in Figure 57.

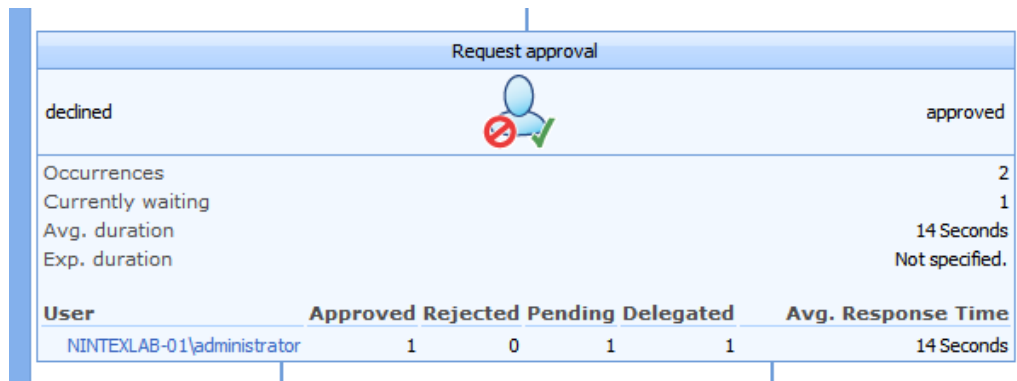


FIGURE 57. REQUEST APPROVAL ACTION STATISTICS

Note that the workflow statistics show only the most recent version of the workflow. There is no way of viewing statistics for previous versions of a workflow, which is a real shame, because it would have allowed you to compare improvements from version to version.

## Reporting

The workflow statistics is not the only option you have for tracking workflows, though. If you have the enterprise license of NW, you also have access to some very powerful reporting options.

### Note

The reporting web parts and pages are available only with the enterprise license.

To access the advanced workflow reporting option, first ensure that the enterprise solution has been deployed. Refer to Chapter 2 if you need a refresher on how to deploy the solution.

The second thing you need to do is to activate the site collection feature called Nintex Workflow 2007 Reporting Web Parts. Again, refer to Chapter 2 for instructions if you need them.

Finally, activate the site feature Nintex Workflow 2007 Enterprise Reporting. Once you do, you will find that the Nintex Workflow section of the Site Settings page now include a link to View Reports. Clicking this link will take you to the Workflow Report Center, shown in Figure 57.

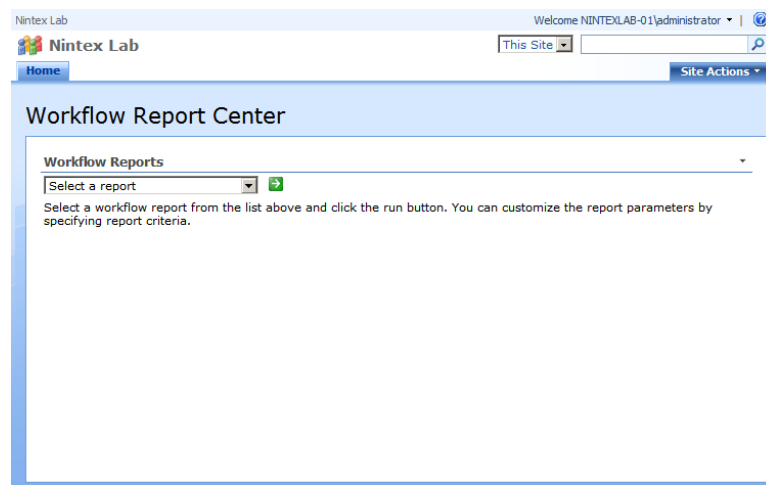


FIGURE 58. WORKFLOW REPORT CENTER

The various workflow reports available give you a range of reporting options. For example, the reports can show how often workflows are used, any overdue workflows, how long workflows take to execute, and even the performance of individual actions within a workflow.

One favorite of mine is the Approver Performance Statistics that shows how often individual users complete tasks on time, how many times they have delegated tasks, and how long they take to respond to a task.

You are not limited to using the Workflow Report Center, however. You can also add reporting web parts to other pages. For example, you can put the Approver Performance Statistics report on the front page of your employee portal to encourage friendly and performance-enhancing competition, as shown in Figure 58.

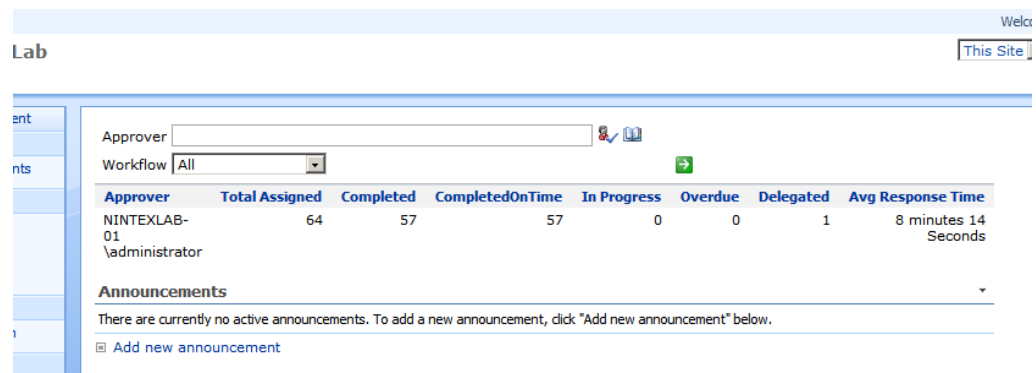


FIGURE 59. LET THE COMPETITION COMMENCE!

Now, if I could only find someone with which to compete, I am sure this would be very inspiring.

I think we have explored this topic far enough. Let's review and look at some other features, before we tackle the "get well" card workflow.

#### REVIEW QUESTIONS

- ❓ 1. Before adding the My Workflow Tasks web part, what must you remember to do to ensure this web part is available?
- ❓ 2. If your time control actions are not accurate, what might be the cause of this?
- ❓ 3. How could you track workflow execution during the course of a workflow?
- ❓ 4. Where would you find the link to the Reporting Center by default?

- ! 1. Before adding the My Workflow Tasks web part, you must remember to activate the Nintex Workflow 2007 Web Parts site collection feature.
- ! 2. One common cause for time control inaccuracies is that SharePoint runs the timer control in a timer job that runs only every five minutes.
- ! 3. You can track workflow execution from the workflow information page, such as from the context menu of items in lists, for example.
- ! 4. By default, you can find the Reporting Center link on the Site Settings page; it's named View Reports.

## Working with External Data

Although I would love to go on talking about features of Nintex Workflow for volumes still, time and space will simply not allow for infinite exploration—at least not until time travel becomes a reality.

However, I want to show you a couple of incredibly powerful features you can use to work with more or less any data source, anywhere in the world. I am talking about web service integration.

In this chapter, I will show you how to interact with a web service and use the results from that web service as part of a workflow. I will extract the current site's users and store them in a workflow collection variable.

However, there are some caveats and some pitfalls, so we need to be a bit creative and work around a particular issue.

### Note

This will not be a deep dive into web services. I will use XPath and regular expressions in this exercise, but I will not explain these concepts beyond showing you what to do.

## Using Web Services from a Workflow

First, start a new workflow for any list or library.

Second, drag a “Call web service” action from the Integration category of workflow actions onto your workflow designer surface. Double-click that action to start the action configuration, shown in Figure 60.

 **Calling a Web Service from a Workflow**

## Call web service



Uri\*

Username

Password

Editor Mode  SOAP Builder  SOAP Editor

Web method

Method parameters

Encode inserted tokens

Store result in

Result format:  Xml  Text

10903

FIGURE 60. “CALL WEB SERVICE” CONFIGURATION

The URL of the web service is the only mandatory field here, but we want to configure other settings as well.

For this exercise, we will retrieve the current site collection’s users, and for that, we can use the SharePoint built-in web service `usergroup.asmx`. Add the URL `http://[your server name]/_vti_bin/usergroup.asmx` to the Url field.

Second, since SharePoint usually requires authentication, you need to enter the credentials of a user with permissions to access the web service.

### Using Stored Credentials for Web Service Authentication

This is a perfect opportunity to demonstrate the use of workflow constants, discussed in Chapter 2. Save your action configuration, and exit the configuration dialog box.

### Note

Using a workflow constant is optional, so feel free to enter the username and password directly in the Username and Password fields if you do not want to set up the constant.

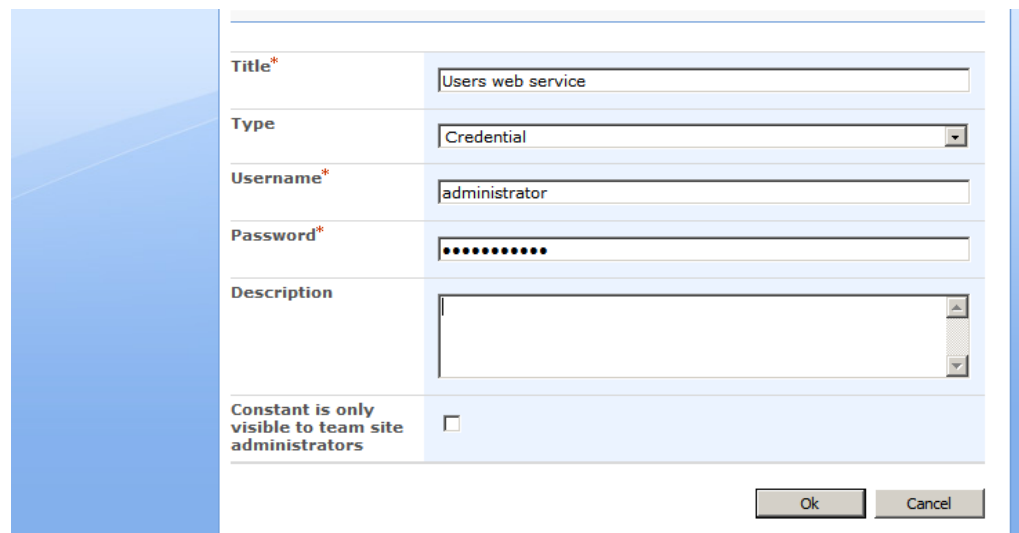
First, go to Site Settings, and then click “Manage workflow constants” in the Nintex Workflow section.

Next, click “Add a new workflow constant to this team site.” Select Credential from the Type drop-down, and fill in a name for the constant, as well as a username and password for a user who has access to the Users and Groups web service.

### Tip

Figure 61 shows my configuration, but note that although using the administrator account is certainly possible and the username and password are stored securely in the database, it is not recommended at all, so consider this a worst practice. The administrator has far-reaching privileges and can cause undesired side effects to your workflow...such as the whole server blowing up.

Instead, use a minimum privilege user, and grant access to just the resources needed.



Title*	Users web service
Type	Credential
Username*	administrator
Password*	.....
Description	
Constant is only visible to team site administrators	<input type="checkbox"/>

Ok Cancel

FIGURE 61. CREDENTIAL CONSTANT

Click Ok to save your constant, and then reopen your workflow and the action configuration for the “Call web service” action.



Once you are back in the action configuration dialog box, you can now click the small padlock next to the username, and the available credential constants are listed. Select your newly created constant, and click Insert to use those credentials.

Now that you have added the URL and the credentials, you can check the web service to see which methods are available. Click the Refresh button next to the “Web method” drop-down.

You may get a warning about current settings being lost, but this applies only if you have already set up a web method call, so we don’t need to worry about that. Simply respond OK to the warning, and after a short refresh, provided that your credentials and URL are correct, the dialog box populates the drop-down menu with the available web methods, as shown in Figure 62.

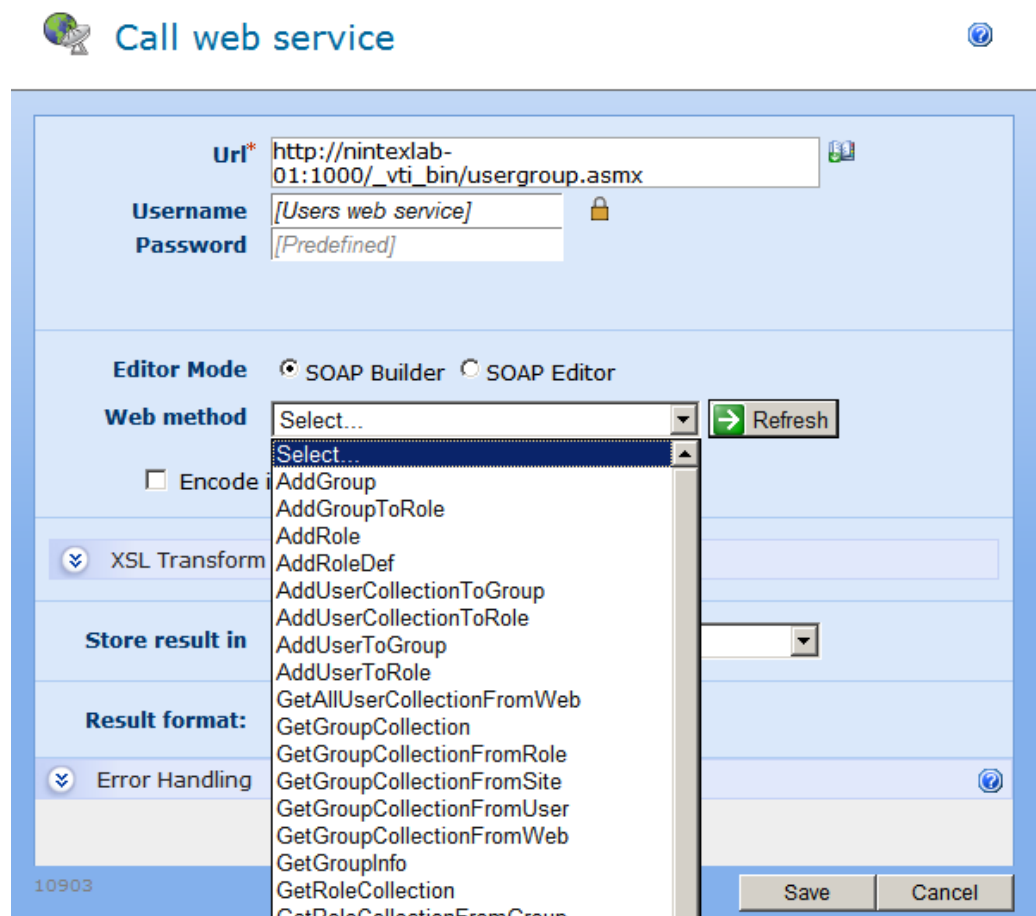


FIGURE 62. AVAILABLE WEB METHODS

As you can see, the single User and Groups web service has many methods from which you can choose. Other web services will have different methods available, so you will need to check the documentation of those web services in order to learn what methods are available as well as what data the methods return.

In our case, we want to use the `GetUserCollectionFromSite` method, so select that now. The `GetUserCollectionFromSite` method returns all current users in the site collection and requires no parameters. However, other methods, such as `GetUserCollectionFromGroup`, allow you to specify input parameters to the method.

Now, we want to store the output from the web service call for later processing. The data returned from web services is usually XML, so we will need to create a text variable to hold the data.

Save the current configuration, and create a new text type variable, for example, called `Users from Web Service`. Remember that you create new workflow variables from the Settings menu.

Reopen the configuration, and scroll to the very bottom to select `Store results in the Users from Web Service` variable. Also, ensure that the result format is set to `Xml` before you click `Save`, and pat yourself on the back for completing the first step of working with external data.

#### **Note**

I don't want to go too far into teaching web services here but rather show you how easy it is to work with external data in NW.

## **Regular Expression Boogie**

We now have some data stored in a text variable, and it really isn't helping us much at this point. After all, we are simply looking at XML data, which isn't very useful in its raw state.

Luckily, NW includes a specialized action for working with XML data, called the `Query XML` action. Drag one of them to the designer surface, and drop it on the blue pearl below the "Call web service" action.

The `Query XML` action allows you to query information from an XML source, either in the work of a URL where the XML document is located or from a variable, which is just what we have, using either `XPath` or `XSLT`.

However, and this is very important, there is a problem with `XPath` and the XML returned by many web services. Without getting too technical, `XPath` has a problem with what is known as a default namespace. The XML returned by SharePoint web services uses a default namespace, so we need to handle that in our workflow by removing the entire default namespace. The first XML tag of the result looks like this:

```
<GetUserCollectionFromSite  
xmlns="http://schemas.microsoft.com/sharepoint/soap/directory/">
```

The xmlns attribute needs to go.

To do so, add a “Regular expression” action from the Operations category between the “Call web service” action and the Query XML action. Double-click the “Regular expression” action, and enter **xmlns="http://schemas.microsoft.com/sharepoint/soap/directory/"** in the Pattern textbox.

In the Input textbox, click Insert Reference, and select the Users from Web Service variable. Leave the “Replace text” radio button selected, and leave the Replacement textbox empty. Finally, set “Store result in” to Users from Web Service to overwrite the result from the web service with the text without the default namespace.

Your completed dialog box should resemble Figure 63.

**Removing Default Namespace from XML**

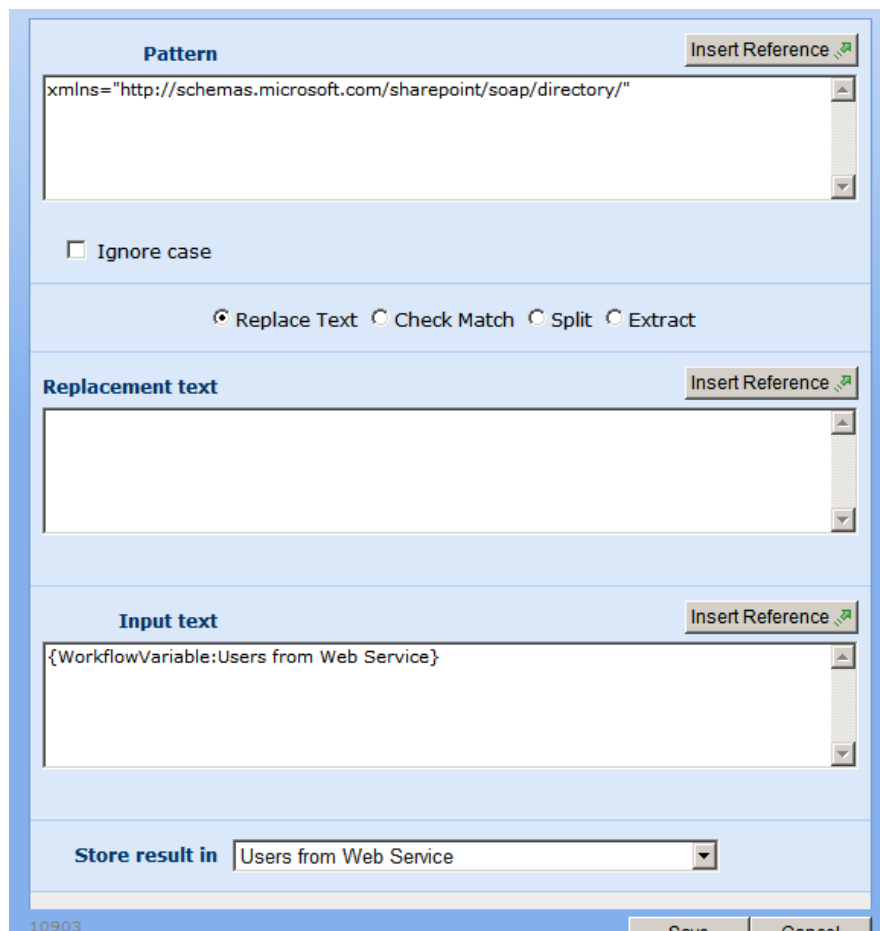


FIGURE 63. REGULAR EXPRESSION ACTION

The “Regular expression” action allows you to perform text manipulation as part of a workflow. Regular expressions use a very powerful but somewhat complex syntax to locate and optionally replace text in a string.

#### Note

The default namespace issue is not related to Nintex Workflow in any way but is simply a lack in functionality in the XPath language.

## Querying XML in a Workflow

Now that you have removed the default namespace, you can configure the Query XML action.

The Query XML action allows you to, well, query XML, using either XSLT or XPath. The XSLT language is mostly used for converting XML into other languages and is a tad too bulky for our purposes. So, we’ll go with the XPath option.



Also, since we are going to store a set of users, it makes sense to create a new “Collection workflow” variable. You may want to call it something like Users. Do so now.

With that done, double-click the Query XML action.

For the XML source, select Xml, and note that you get a new Xml textbox in the dialog box. Click Insert Reference, and find the Users from Web Service workflow variable, probably near the bottom of the list.

Change the “Process using” drop-down to XPath. You will get a new textbox where you can enter the XPath query.

We want to extract the LoginName attribute from any User elements in the XML code we get from the web service, and to get that, we need to enter the following XPath query:

**//User/@LoginName**

Finally, make sure you store the result in the “Users collection” variable you created. The final configured dialog box should look something like Figure 64.

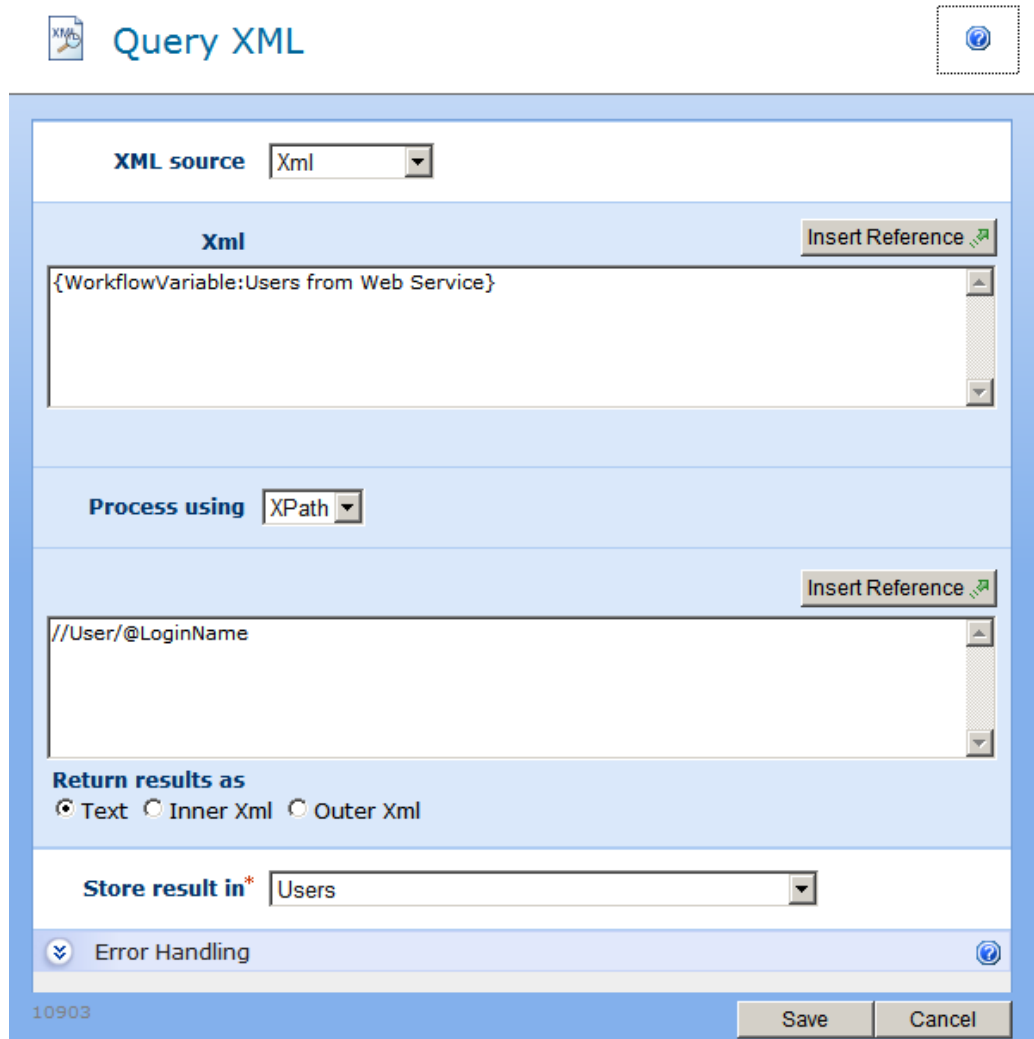


FIGURE 64. QUERY XML ACTION CONFIGURATION

Hit Save, and you are done.

Really, that's it. You have successfully consumed an external web service and then put the result of that request into a usable format for your workflow.

Want to see it in action? Well, you can now perform any operation that requires a collection, such as a "For each" loop action. Yes, we'll do that in the next chapter, after a short review and perhaps a break to catch your breath.

 REVIEW QUESTIONS

1. When querying web services in SharePoint, what is the most common output format?

- ❓ 2. Why will you have problems querying the output from a SharePoint web service using XPath?

 REVIEW QUESTIONS ANSWERED

- ! 1. The most common output format from web services in SharePoint is XML.
- ! 2. You will have problems querying the output from a SharePoint web service using XPath because XPath has a problem with default namespaces.

## Get Well Soon!

*A cozy view on workflow.*

Now that we have explored some of the features and actions of Nintex Workflow 2007, it is time to put some of that new knowledge into a practical workflow.

In this chapter, we will create a “get well” card workflow, in which an absent employee receives “get well” wishes from their fellow employees. We will utilize many of the topics discussed so far and also introduce a few new features, such as the “For each” loop.

First, though, let’s take a look at what we are going to do.

### Task Overview

The workflow we are going to create will trigger from a list of employees. This list could perhaps be a list of absent employees.

#### Note

Honestly, I just want to use a separate list for users to show you how to iterate a SharePoint list. Also, running workflows from users in the People and Groups list requires some modest hacking that is beyond the scope of this issue.

The workflow itself will iterate through the users in the list and assign each a task to provide a short greeting for their sick colleague. If a user has not responded within a certain period, the task is completed automatically with no greeting, and the next employee gets to submit their greeting.

When all the greetings have been collected, the workflow asks the initiator to approve the comments and, if approved, sends an email to the sick employee with all the greetings included.



Seems simple enough, doesn't it? I'm sure you have many ideas on how this can be accomplished, but here is my suggestion.

## Creating the “Get Well” Card Workflow

The first thing we need to do is set up the list of sick employees. We will use this list to launch the “get well” card workflows.



Start by creating a list based on the Custom List template. To do so, click the Site Actions menu, click Create, and select Custom List as the template. Name the list Sick Employees.

When the list appears, click Settings, and select Add Column. Call the new column Username, and select Person or Group as the type. You can leave the default settings for Additional Column Settings. Click OK to add the new column.

The list is done, so let's use it to improve our poor and unfortunate employees' leave of absence.

### Note

I'll agree this wasn't much of an exercise, but we're just warming up. After all, we're here to learn about workflows, not about creating lists.

Create a new workflow for your Sick Employees list. Start with the Blank workflow template.

The first task we need to complete is to create a list of users who should be asked to give a comment. Since we just used the web service method in the previous chapter, let's use that method here as well—with a twist, of course.



To create our list of users, we need a few workflow variables. Create these now:

Variable Name	Type
Users	Collection
Web Service Result	Text

TABLE 1. VARIABLES FOR USER LIST CREATION

Next, drag a “Call web service” action from the Integration category to the first blue pearl of your workflow. You should be well able to configure this action now, so I'll throw in a little twist.

Configure the action as you did in the previous chapter, but use the method `GetUserCollectionFromGroup`. This will return users from a specific group. Notice that you need to enter a parameter for the group name. You can enter the full name of any SharePoint group, for example `<site name> Members`, where `<site name>` needs to be replaced by, you guessed it, your site name.

Store the result in the Web Service Result variable, and hit Save to save the action configuration.

Next, perform the regular expression boogie:

1. Add a “Regular expression” action.
2. Configure it with Pattern set to `xmlns="http://schemas.microsoft.com/sharepoint/soap/directory/`.
3. Insert a reference to Web Service Result in the Input Text box.
4. Store the results in the Web Service Result variable.

Hit Save to save your changes.

The final task required for creating a collection of users is to add a Query XML action. Here are the steps for configuring that action:

1. Use Xml as the XML source.
2. Insert a reference to the Web Service Result variable in the Xml text area.
3. Set “Process using” to XPath.
4. Set the XPath query to be `//User/@LoginName`.
5. Store the results in the Users variable collection.

Hit Save, and you’re done with the user collection part of the workflow. It should resemble Figure 65.

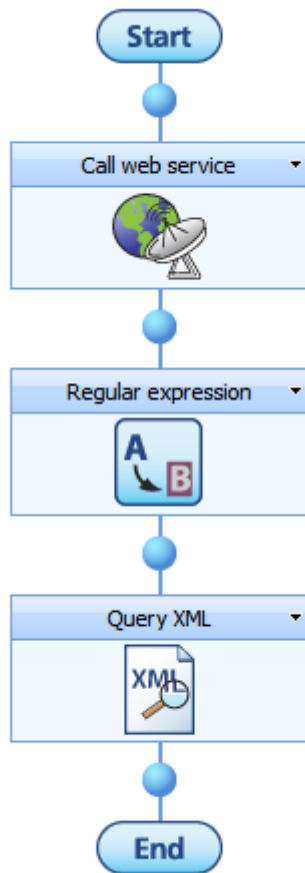


FIGURE 65. USER COLLECTION COMPLETE

You now have a list of users stored in your Users collection. We will iterate through this list in a moment, but first, here's a little tip.

**Tip**

Getting a list of users from a SharePoint group may be a common task in your future workflows. Rather than having to re-create all the steps every time, take the time to either save your current workflow as a snippet or, even better, assign the current actions to an “Action set” action.

To do so, drag a new “Action set” action from the “Logic and flow” category onto one of the blue pearls of your workflow. Next, drag, in order, the actions used in your workflow thus far into the “Action set” action. You can now save the contents of the action set as a snippet, and you can minimize the action set to save space on your designer surface. Both of these options are available from the Action menu, as shown in Figure 65.

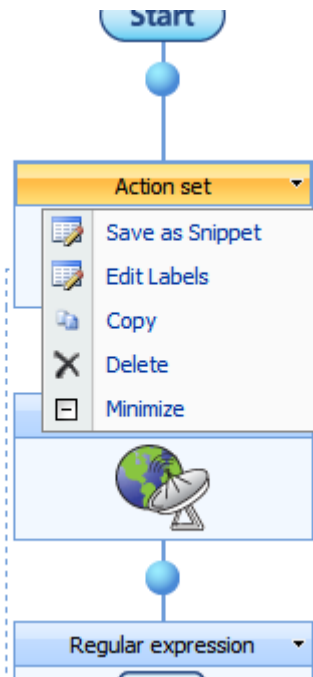


FIGURE 66. ACTION SET MENU

You may also want to rename the action to give a better description of what the set does. Use the Edit Labels option from the action menu to accomplish this. I have named mine Collect Users from Members Group. After minimizing, the workflow surface is much tidier, as shown in Figure 67.

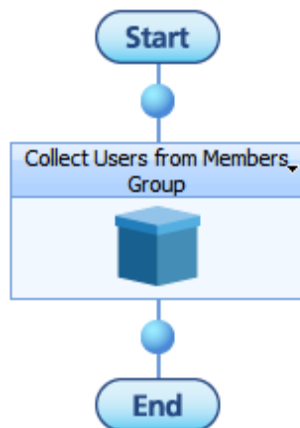


FIGURE 67. AH, SIMPLER, CLEANER, EASIER TO UNDERSTAND

Let's move on to the next step, assigning the tasks to users.

## Collecting Data from Multiple Users

For our next task, we want to iterate through the list of users and assign each a collect data task. However, as you learned in Chapter 4, tasks assigned will pause the entire workflow until the task completes. So, what happens if two users are sick on the same day? The workflow will grind to a halt, and the users will not receive any “get well” card until they come back to work and fill out the greetings themselves. That’s hardly a worthy solution.

We can use automatic delegation to assign someone to fill out the greeting on behalf of whoever is not able to respond in time, but that would still require the new assignee to be present and able to complete all the delegated tasks.

Instead, let’s try a new approach and just complete the workflow task after a certain time period. If the employee is not able to respond within a reasonable time, we’ll just move on to the next employee.

Let’s get started.

First, create a few new variables, shown in Table 2.

 **Assigning Tasks to Multiple Users**

Variable Name	Type
Current Username	Text
Current Greeting	Text
Greetings	Collection
Request Data Action	Action ID

TABLE 2. WORKFLOW VARIABLES FOR GREETINGS

Remember to hit Save after creating the new workflow variables.


Next, to your workflow, add a “For each” action, available from the “Logic and flow” category. Add the new action after the existing action set. Double-click the “For each” action to configure the action.

Select Users as the target collection, and set “Store results in” to Current Username before you click Save.

Now, it wouldn’t make much sense to ask sick employees to wish themselves a quick recovery, so let’s first add a check to ensure we’re not assigning a task to a sick employee.

Add a “Set a condition” action inside the “For each” action, available from the “Commonly used” or the “Logic and flow” category. The “Set a condition” action allows you to test a condition and perform one set of actions if the condition evaluates to yes or true while another set of actions if the condition evaluates to no or false.

We want to compare the username of the current sick employee with each of the usernames of the collection we created in the previous exercise. To do so, configure the “Set a condition” action as such:

1. Set Condition to the Compare “Sick Employees” field.
2. Set Where to Username.
3. Ensure that Equals is selected as the operator.
4. Click the Insert Reference button (  ), and select Workflow Data as the Source and Current Username as the Workflow variable. Hit OK.

Your action configuration should now look like Figure 68, and if it does, click Save to save your “Set a condition” action.

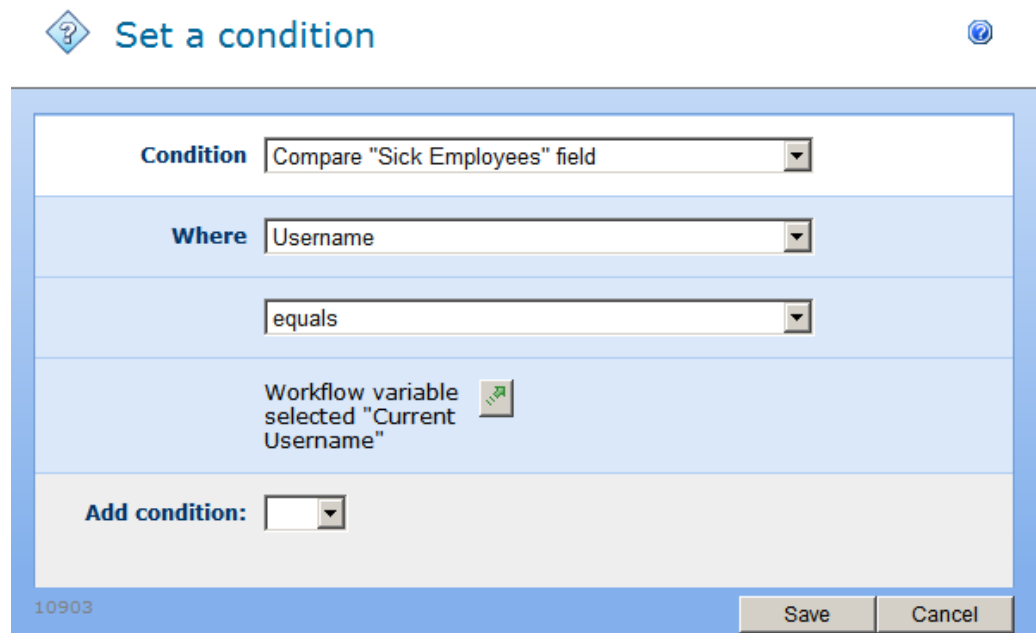


FIGURE 68. “SET A CONDITION” ACTION

We want to assign an action only for the No branch of the condition. Also, as you learned in Chapter 4, we need to use a “Run parallel actions” action to handle the potential automatic task completion.

Add a “Run parallel actions” action to the blue pearl under the No branch of the “Set a condition” action. To the left branch, drag a “Request data” action, available from the “User interaction” category.

Also, add a “Collection operation” action below the “Request data” action. This action will store the greeting in the Greetings collection you created earlier.

Finally, from the User interaction category, drag a “Complete workflow task” action to the right branch of the “Run parallel actions” action. Your result so far should resemble Figure 69.

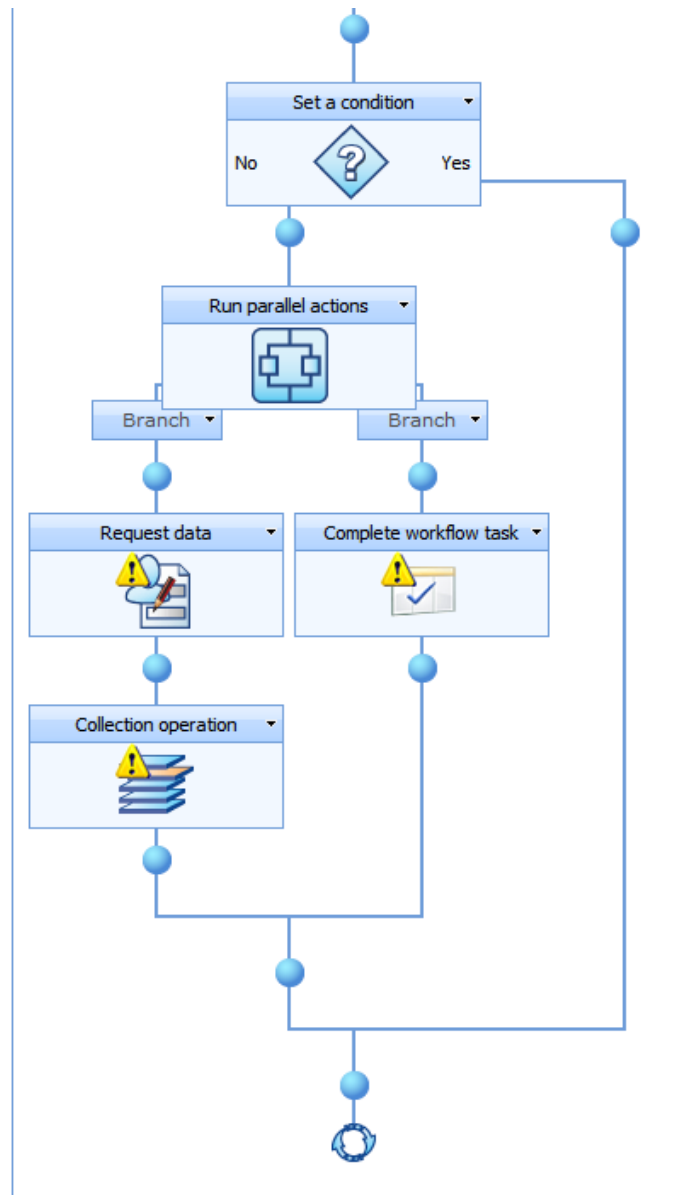


FIGURE 69. REQUEST DATA FROM USERS

Let's configure those actions to make sure we get what we need.

First, double-click the "Request data" action. For the "Collect data from" field, click the address book icon to look up the workflow variable Current Username from the Lookup section.

Next, for the content type, leave the "Create new" option enabled, and type in a descriptive name, such as **Retrieve Greeting**.

Click Add field to add a new column to the content type you just defined. Leave the type as "Single line of text," set the name to Greeting, and include a default value, such as **Get well soon!** Hit Save to add the column, and then set "Store results in" to Current Greeting.

You have created all the workflow variables, right?

Finally, close to the bottom, set "Store action ID in" to Request Data Action.

This concludes the "Request data action" configuration, so click Save to save the configuration.

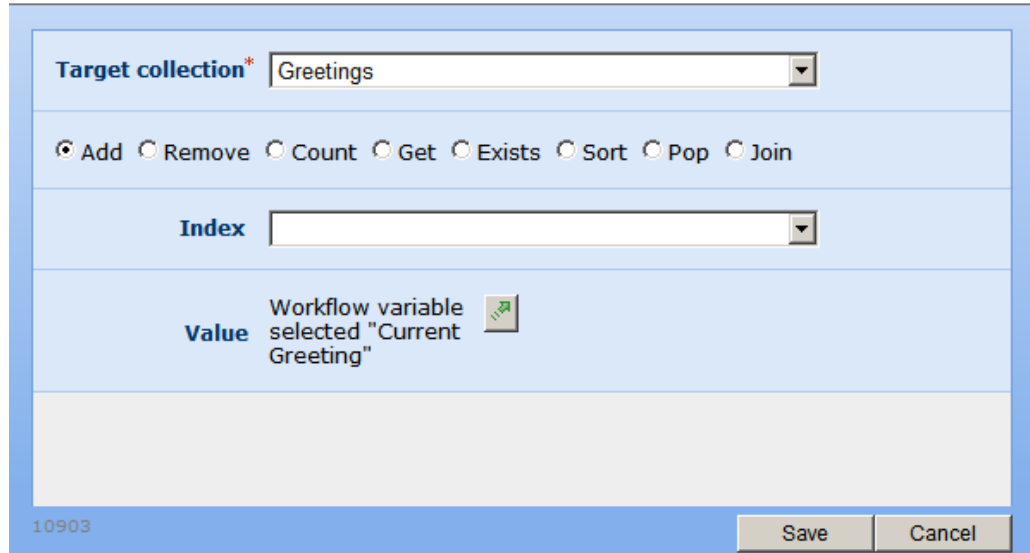
With the user now storing their greetings in the Current Greeting variable, we can move on to add that greeting to the Greetings collection. This is what the "Collection operation" action does best, so double-click that action now to configure it.

The "Collection operation" action allows you to perform operations on a collection of data, such as adding, removing, counting, and sorting, as well as a few other operations. We want to add each of the greetings to the Greetings collection, so let's configure the action as such.

In the "Collection operation" action configuration, set the Target collection to Greetings, and then click the Insert Reference button near the Value field to look up the workflow variable Current Greeting. Set Source to Workflow Data, and you should find the Current Greeting variable in the "Workflow variable" list.

Ensure that the Add radio box is selected, and hit Save to store your action, after ensuring that your dialog box resembles Figure 70.





Target collection\*

Add  Remove  Count  Get  Exists  Sort  Pop  Join

Index

Value Workflow variable selected "Current Greeting"

10903

FIGURE 70. COLLECTION OPERATION CONFIGURED

Great, that’s it for the first branch of the “Run parallel actions” action. The second branch is a lot easier.

To complete that branch, double-click the “Complete workflow task” action. In that action editor, set the Action ID to the Request Data Action variable, and then set a timeout for how long you want to wait for each user to provide a greeting. After this, the action is configured. You can optionally send a message to the user that their response is no longer required by leaving the Send ‘response not required’ message on.

**Note**

Remember that because of the nature of how workflows execute, only one task will be active at a time. Also, remember that both the branches must complete for the workflow to continue. The timeout you set here will thus dictate the minimum amount of time that each user task will take. You need to multiply that timeout by the number of users to find the total time required for the entire workflow to run.

**Tip**

You’re done with setting up the greeting requests from all users. Feel free to minimize the entire For each loop to save some space on your designer surface, and perhaps change the label of the action to provide a better description.

## Building a Dynamic String of Greetings

Just one more exercise, and you can test everything and lean back to the sound of roaring applause from your absent employees.

The final task is to send the message to the user. To accomplish this, we first need to build the completed greeting text and then submit that to the initiator for approval.

Let's just jump in; I think we've warmed up enough as it is.

Start by creating the workflow variable from Table 3:



Variable Name	Type
Completed Greeting	Text

TABLE 3. GREETINGS WORKFLOW VARIABLES

Our first task is to merge all the individual greetings, now stored in the Greetings collection, into a single piece of text. Since the Greetings variable is a collection, another For each loop will be perfect for this job.

Add a new “For each” action to your workflow, below the existing “For each” loop. Configure that loop with the Target collection as Greetings, and set “Store results in” as Current Greeting.

Notice that we are reusing the Current Greeting variable from the previous “For each” loop.

Next, drag a “Build dynamic string” action from the Operations category into the “For each” loop.

The “Build a dynamic string” action allows you to create texts from various data sources during the execution of your workflow. This is useful in a number of scenarios where you need to merge text-based data for use in other actions.

In our scenario, we want to merge the texts from the Greetings collection into a new string called Completed Greeting, and to do so, configure the “Build a dynamic string” action as such:

1. In the “Build string” box, insert a reference to the Completed Greeting variable, followed by a reference to the Current Greeting variable.
2. In the “Store results in” box, select the Completed Greeting variable.

By storing both the Completed Greeting variable and the Current Greeting variable, you are basically appending each of the greetings in the Greetings variable to the end of the Completed Greeting.

After this “For each” loop, your Completed Greeting variable will include all the greetings from the users who responded to the request.

Figure 71 shows the completed “Build a dynamic string” configuration.

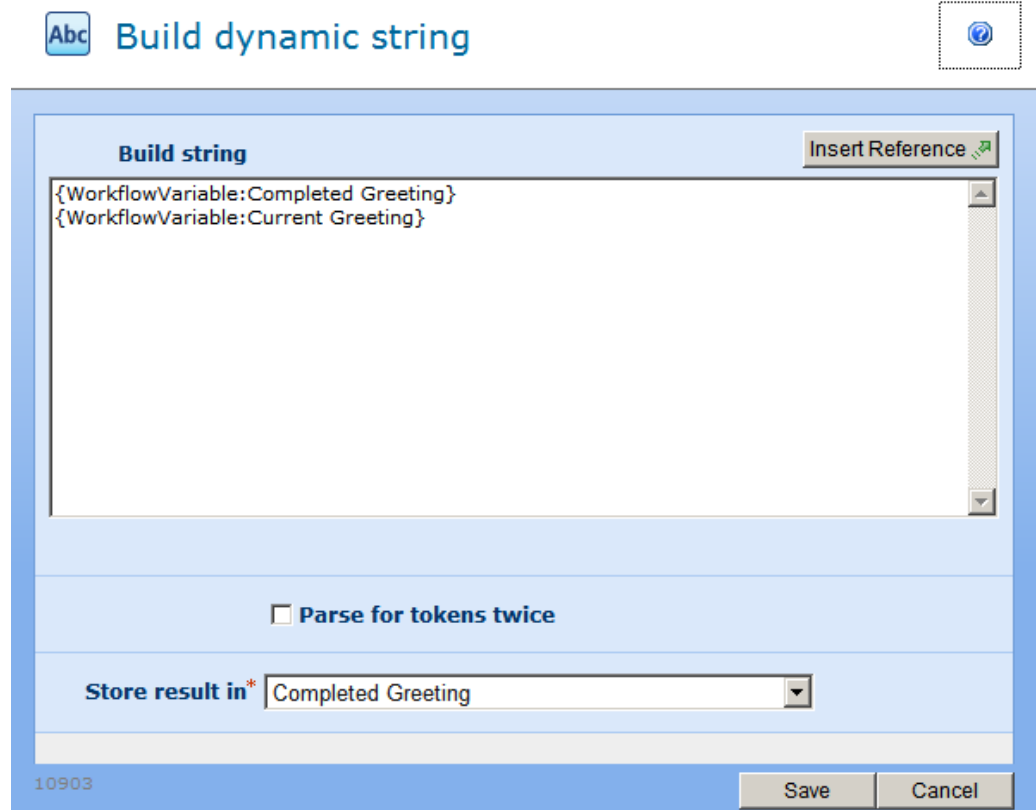


FIGURE 71. “BUILD A DYNAMIC STRING” STRING

Developers will recognize this construct as an elaborate way of saying this:

```
foreach (string CurrentGreeting in Greetings) {  
    CompletedGreeting += CurrentGreeting;  
}
```

## Approving and Sending the Greetings

All we need now is to ask the initiator to approve the messages to make sure someone doesn’t, well, abuse the system. And of course, the nicest part of this entire process, we need to send the greeting to the sick employee. Seems simple enough? Ah, but as in all good movies, the simplest of tasks, such as pointing a gun at the zombie and pulling the trigger, is never as simple as it seems.

For this final exercise, I think we can pull out all the heavy artillery and build a multilayered, multibranching workflow, just to show how you can create very complex logic to tailor your workflow exactly to your needs.

For example, what happens if the initiator forgets the greeting? I think we had better remind that person of the importance of taking care of sick employees. Well, to be honest, the real motivation is that I want to show you another method for handling uncompleted tasks.

Oh, and what happens if someone writes an inappropriate greeting or somehow the initiator decides that the greetings need to change?

We will solve these problems in this exercise.



Start by dragging a “Run parallel actions” action to the last blue pearl of your workflow, after the Build dynamic string “For each” action.

To the left branch, add a “Request approval” action, as well as a “Send a notification” action. To the Declined branch of the “Request approval” action, add a “Request data” action.

To the right branch of the “Run parallel actions” action, add a “Task reminder” action, as well as a “Complete workflow task” action.

Your “Run parallel actions” action should now look like Figure 72.

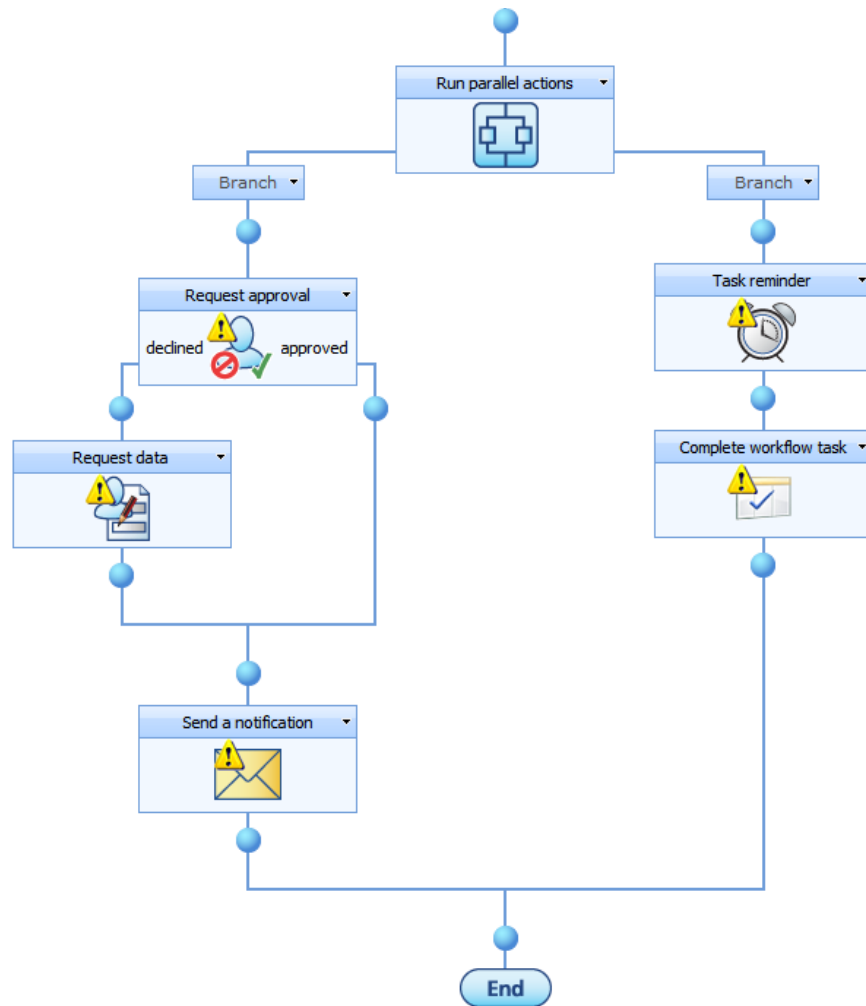


FIGURE 72. REQUEST APPROVAL LOOP

Configure the “Request approval” action as such:

1. For the Approvers, add a lookup to the common value Initiator.
2. Store the Action ID in “Request data” action. Yes, we are reusing the variable created in the first exercise of this chapter.
3. Hit Save.

Simple enough? Good, let’s move on.

Configure the “Request data” action as such:

1. In the “Collect data from” field, again add a lookup to the common value Initiator.

2. Create a new content type, named Collect Revised Greetings.
3. Add a field, named Revised Greeting, of type “Multiple lines of text.” Store the result of that column in the Completed Greeting variable.
4. Again, store the action ID in Request Data Action.
5. Hit Save.

A few things may pique your curiosity here. We are using the same action ID for both the request approval and the request data. This little trick allows us to use the same task reminder for two consecutive tasks.

We are storing the action ID in the first place in order to send a task reminder using the “Task reminder” action. Now, if the first task completes as declined, the second task will set the Request Data action and ensure that the “Task reminder” action will keep track of the new task.

We’re rapidly advancing here, because most of this more or less repeats information from previous chapters. See how easy this is once you have learned the basics first?

Configure the “Send a notification” action as such:

1. Set the To field to the Item property Username, found in the lookup section.
2. Set the subject to something nice, such as **Get well soon!**
3. Add an appropriate greeting to the body, such as **We all hope you get well soon. Here are some greetings from your fellow employees:**
4. Insert a reference to the Completed Greeting variable to the body of the message.
5. Hit Save.

Great, this concludes the left branch, so let’s quickly go over the right branch.

Double-click the Task reminder action and configure as such:

1. Set the action ID to the Request Data Action variable.
2. Set up a reasonable number of reminders and times, for example, four reminders spaced one hour apart.
3. Set the subject to something like **Please approve the greetings or provide new greetings.**

4. Optionally, add a message stating how important such a Get Well card is to the sick employee.
5. Hit Save.

The task reminder works very much like the “Delegate task” action, in that it stops execution of the branch until the specified time has elapsed. In this case, however, we are sending the email as soon as the approval or optional greeting correction tasks have completed, so the right branch will not pause the progress of what we want to achieve.

Finally, configure the Complete workflow task as such:

1. Set the action ID to Request Data Action.
2. Set a short time span, such as the default one minute.
3. Ensure that the task outcome is Approved.
4. Hit Save.

The short time span here only means that we won’t wait for very long after the last reminder has been sent before we just accept the greetings as is.

Now, what happens if the task autocompletes here? Well, if the first “Request approval” task has not been completed, we autoaccept and bypass the Request Data action altogether, and the email is sent. If we are waiting for the Request Data action to complete, we are still just accepting that task, and the email is sent.

Neat, eh? I told you NW is a cool product.

Before we end, though, I’d like to draw your attention to the following review questions.

 REVIEW QUESTIONS

- ❓ 1. What happens to the flow of a workflow when a task is assigned to someone?
- ❓ 2. What is the purpose of the variable type Action ID?

 REVIEW QUESTIONS ANSWERED

- ! 1. When a task is assigned to someone, the flow of the workflow will stop pending task completion. Other branches of the workflow may continue.
- ! 2. The Action ID variable type stores a reference to another action in the workflow, such as a task assignment action.



## State Machines

*Harness the power!*

OK, we have seen a lot of nice stuff so far, but we are not even close to exploring all the possibilities that NW has to offer. I am not going to even attempt to explain everything, but I will introduce you to some of the more advanced features available.

In this chapter, we will continue to build on the “get well” card workflow to introduce the concept of state machines. As you read in Chapter 1, state machines require a slight change of how you think of workflows, but once you understand how to “think” state machine, you will likely realize that this is a far easier and more intuitive method of workflow development.

### State Machine Workflows

So far, our workflow is a sequential workflow. True, we do have some loops, and we do have the option of going a bit back and forth while asking for comments.

As you remember from Chapter 1, the concept of states means that a workflow can be in one state at a time. Each state is a sequential workflow in itself, and during the course of each such state, the workflow may transition into another state.

So, if we create a flowchart for the “get well” card workflow, it might be something like Figure 73.

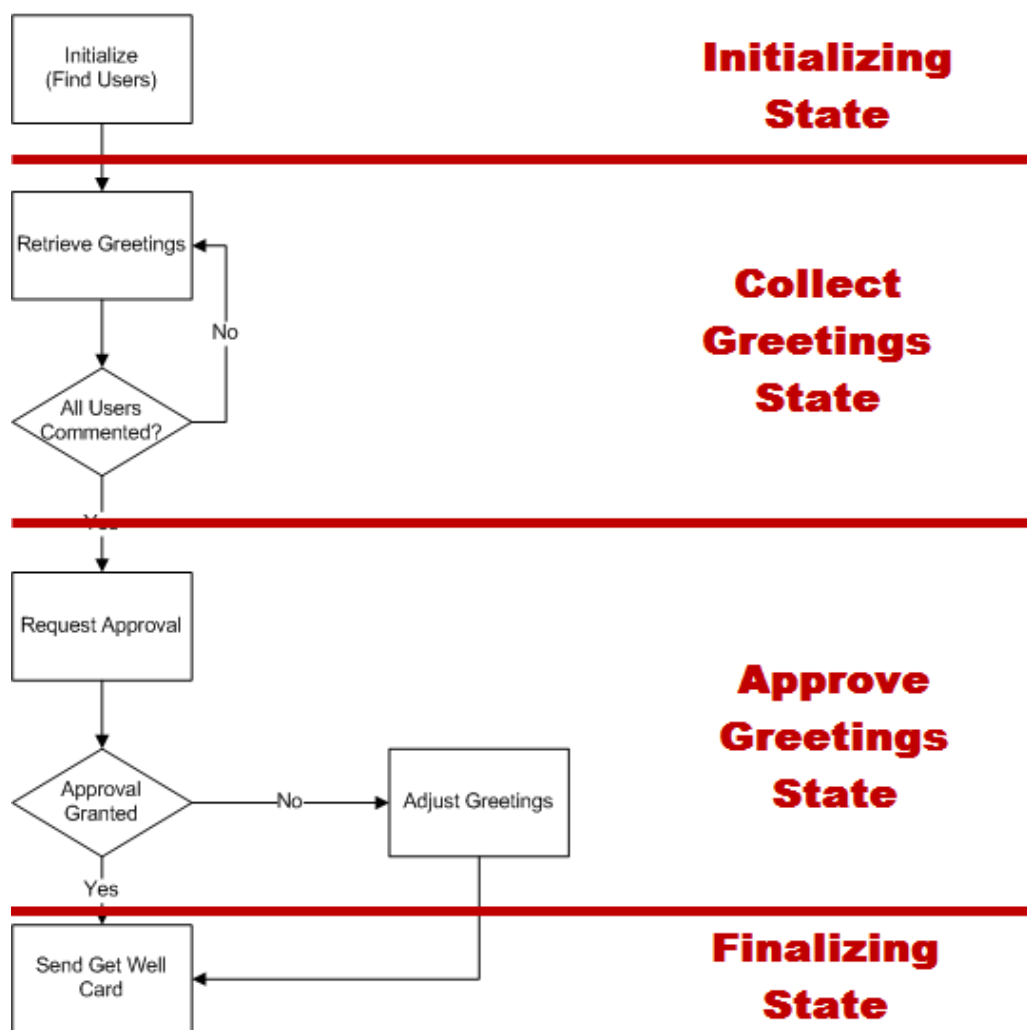


FIGURE 73. STATES FOR “GET WELL” CARD

One very unique aspect of workflows in Nintex Workflow is that you can combine state machines with regular sequential workflows inside the same workflow. Even if you go to Visual Studio to develop your workflows, you ultimately need to decide to do either a state machine or a sequential workflow, and if you change your mind later, you are stuck with your choice and need to re-create the entire workflow.

In Nintex Workflow, however, you can combine these two methods using an action called “State machine.” In the next exercise, I’ll show you how to augment the workflow to include a state machine.

### Building a State Machine

In this exercise, we are going to rebuild our workflow into a state machine workflow. To demonstrate how you can combine state machines with sequential workflows, we’ll use a state machine only for two of the states in our flowchart, the Collect Greetings state and the Approve Greetings state.

**Adding a State Machine Action**

First, click the pearl below the Collect Users from Members Group task “Action set,” and select to add a new state machine. The “State machine” action is available from the “Logic and flow” category, as shown in Figure 74.

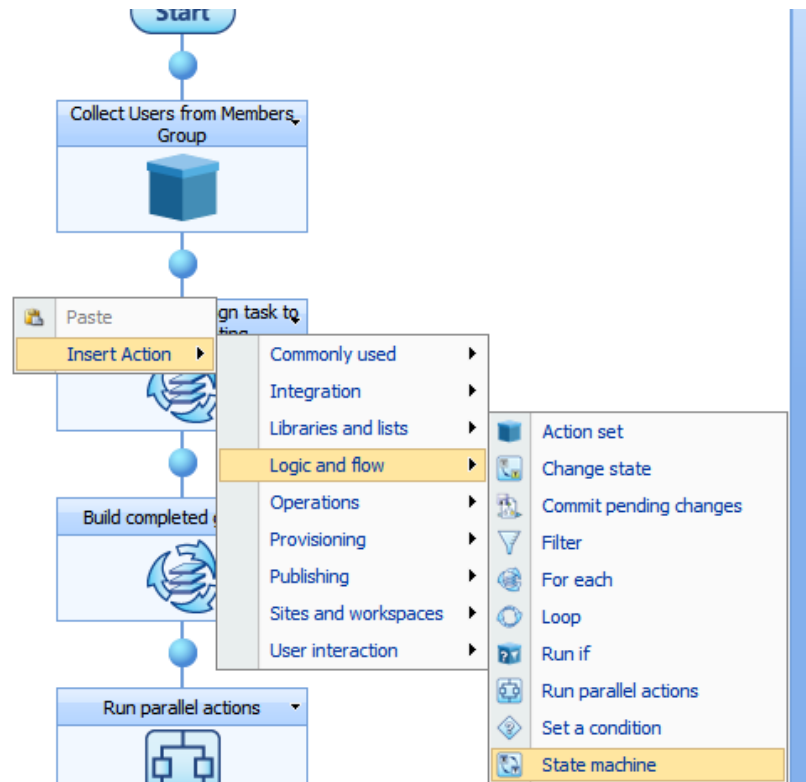


FIGURE 74. ADDING A STATE MACHINE

Once you add the “State machine” action, notice that you initially have two states in the state machine. You can add as many states as you need and move between them in any way you want. However, you must always have one initial state, which is why the state machine initially has a warning icon.

Configure the “State machine” action by double-clicking the icon or going to the action menu and selecting Configure.

First, you’ll see that you have the option of selecting which state should be the initial state. Select State 1 as the initial state.

Below that, you have the chance to set new names for the various states or add more states. The names State 1 and State 2 are not really good names, so name them Collect Greetings and Approve Greetings.

However, once you do, notice that the Initial State drop-down resets. State 1 no longer exists, so you need to reselect your newly named Collect Greetings state as the initial state. Your configuration dialog box should now resemble Figure 75.

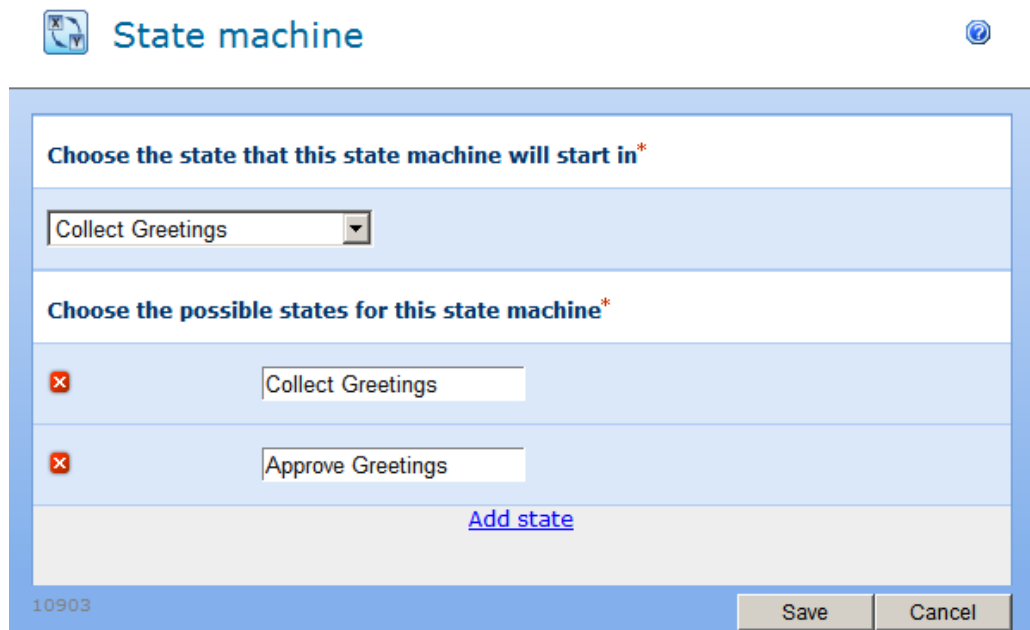


FIGURE 75. CONFIGURED STATE MACHINE DIALOG BOX

Hit Save, and your “State machine” action is configured for now.

Notice that the order of states is not important. By their very nature, state machines move back and forth between states more or less any way you like, so it really does not matter in which order you add or configure the individual states. In my example, I have the leftmost state be the Approve Greetings state and the rightmost state be the Collect Greetings state.

Next, it is time to rebuild our entire workflow into a state machine.

**Note**

If you had told a Visual Studio workflow developer you changed your mind midproject and wanted to do a state machine rather than a sequential workflow, you would probably see the back of that person leaving the room very fast. Chances are, you would need a visa and a lot of vaccines to go to the part of the world where that developer would be hiding.

Start by drag and dropping the action set named “For each user, assign task to provide greeting” onto the pearl in the Collect Greetings state branch. Next, drag the “Build completed greeting” action set on to the pearl below the last action set.

Finally, drag the “Request greeting approval” action to the Approve Greetings state branch.

You are done. Well, almost, but at least all the hard work is done, and when those three simple steps were the hardest, you should appreciate how easy Nintex Workflow makes complex tasks.

Of course, we still haven’t done anything to ensure that one state transitions into another state, and this is an important task. To mimic our previous functionality, add a new “Change state” action below the “Build completed greeting” action set, as shown in Figure 76.

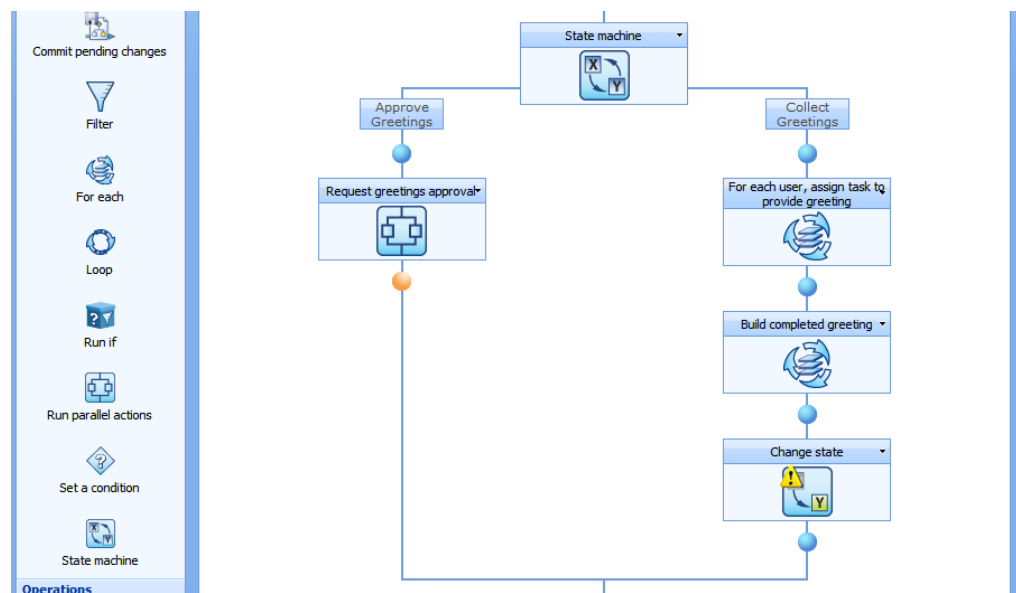


FIGURE 76. CHANGE STATE ACTION ADDED

The configuration of the “Change state” action is also very simple; just double-click the icon, and select the new state to which the workflow will transition. In this case, that would be the Approve Greetings state.

Finally, you need to add a “Change state” action to the Approve Greetings state to ensure that you exit the state machine. Add a “Change state” action after the “Request greetings approval” action set, and configure that “Change state” action to set the Next state to End State Machine, and hit Save to complete the state machine configuration.

Now you are done, and you should feel free to test your workflow to make sure that everything works as you’d expect, even with that massive restructuring you just did.

Once you've confirmed that everything still works as expected, you can sit down and ponder why you would go through so much trouble just to re-create something you have already created.

I will tell you why. Or rather, I'll show you why, which I think makes a bigger impact.

### **Advancing the State Machine**

In the step of the workflow where the initiator approves or declines the greetings, we currently ask the initiator to provide all the greetings if they decline the greetings submitted by the employees.

Now, instead of just having one person provide all the greetings, let's instead see whether we can jump back into the collect greetings state, to reask the employees for new greetings.

This may seem like a rather annoying feature, asking everyone to resubmit their greetings if even one greeting is not approved, but there are a few worthwhile points to this exercise.



Open the "Request greetings approval" action. Click the action menu of the "Request data" action, and delete the action.

Here is the first point I want to make with this exercise. If we want to restart the request greetings process, we need to clear out the current greetings first. If we simply restart the request greeting process, our new greetings will be appended only to the existing greetings.

The first step is fairly easy but has a slight pitfall. We need to set the "Completed greeting" variable to nothing. To do so, you can add a "Set a variable" action to the Declined branch of the "Request approval" action, and then double-click the "Set a variable" action to configure the action.

However, if you do, you will find that you cannot set the "Completed greeting" variable to nothing at all, because the action will require you to set some value, and blank is not a valid value.

A better approach is to use the "Build a dynamic string" action. The "Build a dynamic string" action allows you to input nothing in the "Build string" field, so add a "Build a dynamic string" action, double-click to configure it, and just set the "Store result in" field to Completed Greeting, as shown in Figure 77.

## Build dynamic string

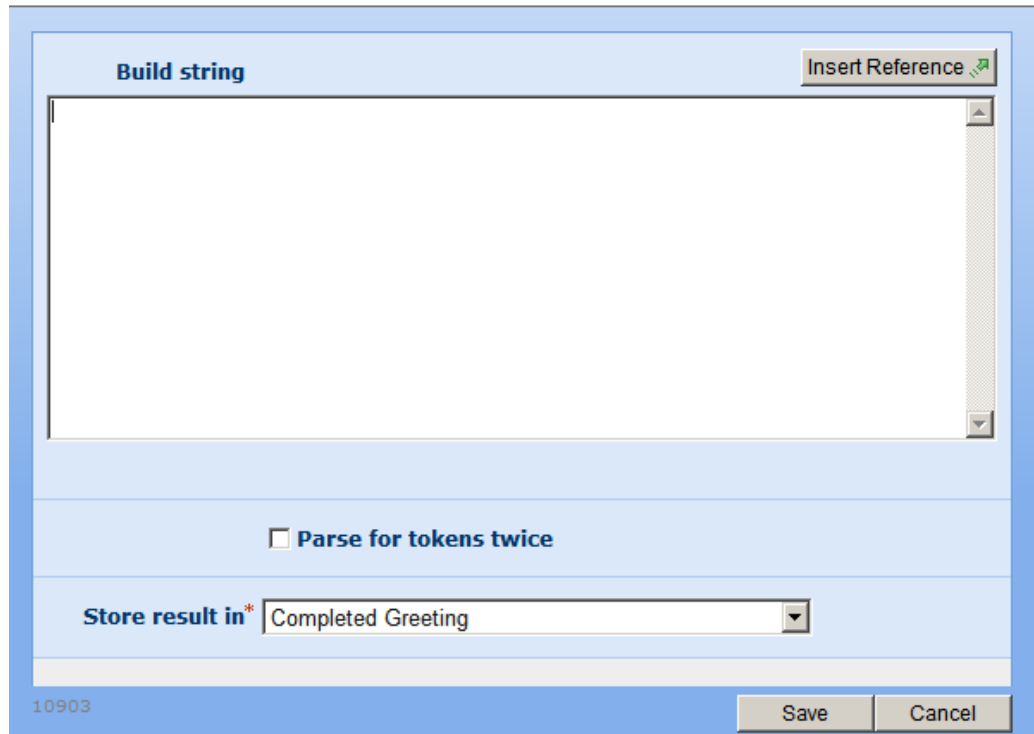


FIGURE 77. BUILD EMPTY DYNAMIC STRING

This action will now clear the Completed Greeting variable, but we still need to clear the “Greetings collection” variable, which this is a bit more complex.

To work with collections, we need to use the “Collection operation” action. However, we need to remove all the items from the collection, and the Collection operation does not have a “Clear collection” option. Instead, we need to use a “For each” action and iterate through the collection, removing one item from the collection at a time.

Add a new “For each” action from the “Logic and flow” category. Inside the “For each” action, add a “Collection operation” action from the Operations category, and double-click to configure. Now, you may think that you should use the Remove option as the operation.

However, doing so will make things more complex, because the Remove option requires you to have an index, which is a variable. This in turn will require you to keep track of that variable and increase it once per “For each” iteration. Not just that, but you would also need to ensure you reset the variable before the “For each” loop, in case the greetings are rejected a second time.

Instead, choose the Pop option. The Pop option will remove the last item in the collection and store that item in a variable. In our case, we can use the Current Greeting variable, because we won't use this variable at all.

Hit Save, and the variable clearing will be complete.

The last thing we really need to do now is to transition back to the Collect Greetings state to restart the process, right? Well, before you add a Change state after the “For each” loop, which won't work, you need to realize one very important thing regarding the transition of states.

The Change state action does not actually change the state at all; it merely sets which state should be the next state when the current state ends. Take a look at Figure 78.

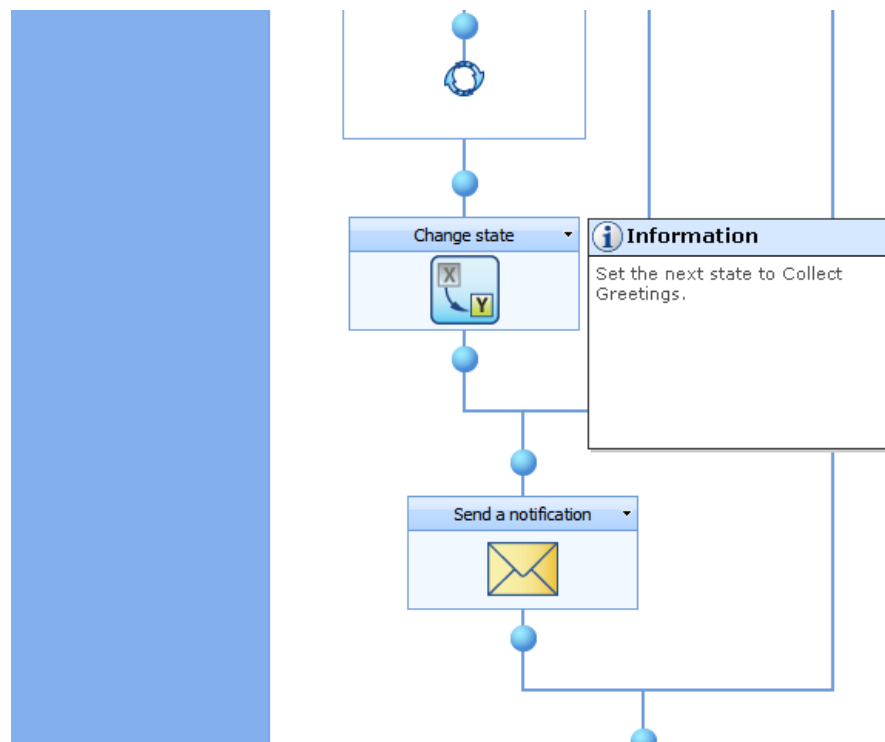


FIGURE 78. NOT CHANGING STATE AT ALL

In this example, the “Send a notification” action will happen, even if you have correctly set the Change state to transition into the Collect Greetings state.

The reason is that the state change happens at the end of the current state. So, rather than using the “Change state” action, we need to set a variable to store whether to end the state machine or to restart the Collect Greetings state after the current state ends.



To do so, first create a new workflow variable of type Yes/No, and name it “Request approved.” Then, add a “Set a variable” action after the “For each” action. Configure that action to set the “Request approved” variable to No.

Of course, we also need to ensure we set the “Request approved” variable to Yes if the request is indeed approved. As such, add a “Set a variable” action to the Approved branch of the “Request approval” action as well, and configure that action to set the “Request approved” variable to Yes.

Next, we need to send the email only if the Request approved variable is set to Yes. Otherwise, we should move into the Collect Greetings state again.

To perform this conditional branching, add a new “Set a condition” action just before the final “Send a notification” action. The “Set a condition” action can be a bit tricky, especially when working with Yes/No variables. In essence, you are checking whether the Yes/No variable is set to either Yes or No. If that is the case, the Yes branch of the action will execute, and if that is not the case (in other words, the variable is set to the opposite of the value you selected), then the No branch of the action will execute.

Configure the “Set a condition” action to “Compare any data source.” Click the lookup button next to the Where field, select “Workflow data” as the source, and select “Request approved” as the Workflow variable. Finally, select Yes as the value before hitting Save.

Now, as explained, if the “Request approved” variable is set to Yes, then the Yes branch will execute. So, drag the “Send a notification” action from outside the “Set a condition” action into the Yes branch of the action.

Then, drag the “Change state” action you created in the previous exercise to the blue pearl below the “Send a notification” action.

Finally, add a new “Change state” action to the No branch of the “Set a condition” action. Configure that “Change state” action to set the next state to Collect Greetings.

Your “Set a condition” action should now look like Figure 79.

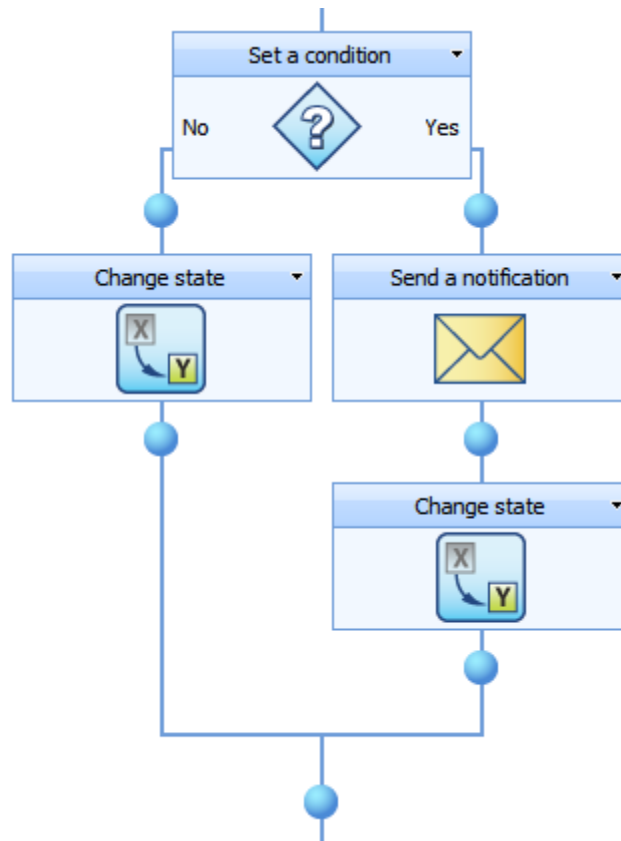


FIGURE 79. COMPLETED “SET A CONDITION”

OK, I know, we could have used the existing “Request approval” action and added the “Change state” actions to that branch, including the “Send a notification” in the Approved branch.

However, then I wouldn’t have the opportunity to explain the logic of state changes and when these happen. And I really wanted that, because the pitfall here is that you set a state change inside a set of complex logic, thinking that the state would change right away.

OK, I think that about does it for workflow creation. We have grown our workflow into a fairly complex application. However, using Nintex Workflow, we can focus on the logic of the workflow rather than wrestling with the tool.

We have come to the end of our workflow creation. Before we end, however, let’s review what you learned about state machines.

 REVIEW QUESTIONS

1. In what way does the order in which you define states affect the workflow?

- ❓ 2. Why would you need to clear out existing collections and variables before changing to a previous state?
- ❓ 3. What does the “Change state” action do?

 REVIEW QUESTIONS ANSWERED

- ! 1. The order in which you define states does not affect the workflow at all.
- ! 2. You would need to clear out existing collections and variables before changing to a previous state to avoid duplicate data in those variables.
- ! 3. The “Change state” action sets which state should come after the current state has completed.

# Final Thoughts and Additional Resources

*Never stop learning.*

Trying to cover an entire product like Nintex Workflow 2007 in a single issue is quite a challenge, and as you can see, I have ended up with the longest *USP Journal* issue yet. Still, there are lots of things to discover in NW, and I hope I have inspired you to seek out some of these things on your own.

I hope I have shown you that Nintex Workflow 2007 gives you a lot of power while maintaining a simple and understandable interface. None of the exercises we have performed here require any degree in rocket science, but you have still created a fairly sophisticated workflow using many of the available actions.

I would also like to thank Nintex for agreeing to sponsor this issue and thus provide you with this free issue of *Understanding SharePoint Journal*. If you like what you have read, feel free to check out the other issues as well.

To get information about the upcoming issues of *USP Journal*, make sure you sign up for the mailing list on the *USP Journal* website, <http://www.understandingsharepoint.com/journal>. Members of the list receive special offers, discounts, and previews of new issues.

You may also want to check out my book *Building the SharePoint User Experience*, which deals with every aspect of the user experience, including a thorough understanding of the SharePoint architecture. You can read more about the book on <http://www.understandingsharepoint.com/userexperience>.

Finally, check out my blog at <http://www.understandingsharepoint.com/url/1000> where you will find shorter articles, tips, questions and answers, and downloadable content.

Thank you for reading this issue, and I hope to see you next time,

.b

# Previous *USP Journal* Issues

*What you have missed so far, but can still get from our website.*

## **Issue 1: SPCurrentUsers Explained**

This issue deals with the solution SPCurrentUsers available from CodePlex and covers the following main topics:

- DelegateControls
- CustomAction, both CAML-based and Assembly-based
- Custom application pages in SharePoint, using code-behind files
- List deployment and column manipulation
- Solution setup using SharePoint event receivers

URL: <http://www.understandingsharepoint.com/journal/volume-1/issue-1>

## **Issue 2: Developing SharePoint Content Types**

If you want to learn everything about developing SharePoint content types, you should get your hands on issue 2, covering the following:

- Content type introduction
- Visual interface using FormTemplates
- Inheritance
- Behavior using event receivers
- Content types as folders
- Custom XmlDocument solutions

URL: <http://www.understandingsharepoint.com/journal/volume-1/issue-2>

## **Issue 3: SPTags Explained**

If you have ever tried to build custom field types in SharePoint, you definitely want to pick up issue 3, covering the following:

- Custom field type development
- SharePoint Web part development
- Custom page development

- Custom web part properties
- Field type custom property storage

URL: <http://www.understandingsharepoint.com/journal/volume-1/issue-3>

#### **Issue 4: SharePoint Designer Workflow**

In issue 4 of Understanding SharePoint Journal, you will learn how to create workflows in SharePoint Designer. The issue covers:

- Workflow in a NutShell
- SharePoint Designer workflow activities
- Workflow branching
- Working with variables
- Loops in SharePoint Designer workflows

URL: <http://www.understandingsharepoint.com/journal/volume-1/issue-4>

# ***USP Journal* Affiliate Program**

*Want to promote USP Journal— and make a buck?*

The *USP Journal* affiliate program is your chance to earn money by promoting *USP Journal* to your friends, readers, colleagues, or others who are interested in learning more about SharePoint.

Here are four great reasons to join the *USP Journal* affiliate program:

- You'll earn a generous share of sales just by recommending to others what you like yourself.
- You'll get free tools to help you refer more readers, including free advance previews of upcoming issues and excerpt content to use on your own site or blog.
- High-quality products mean high conversion rates and few returns. And, most importantly, they mean happy readers.
- You'll get the safety and easy of mind from regular payments, handled by E-junkie and PayPal.

To learn more about the program, visit  
<http://www.understandingsharepoint.com/affiliates>.