Use the Octopus Deploy step templates

In this tutorial, you'll deploy a database from a NuGet package, using the DLM Automation step templates for Octopus Deploy.

On this page

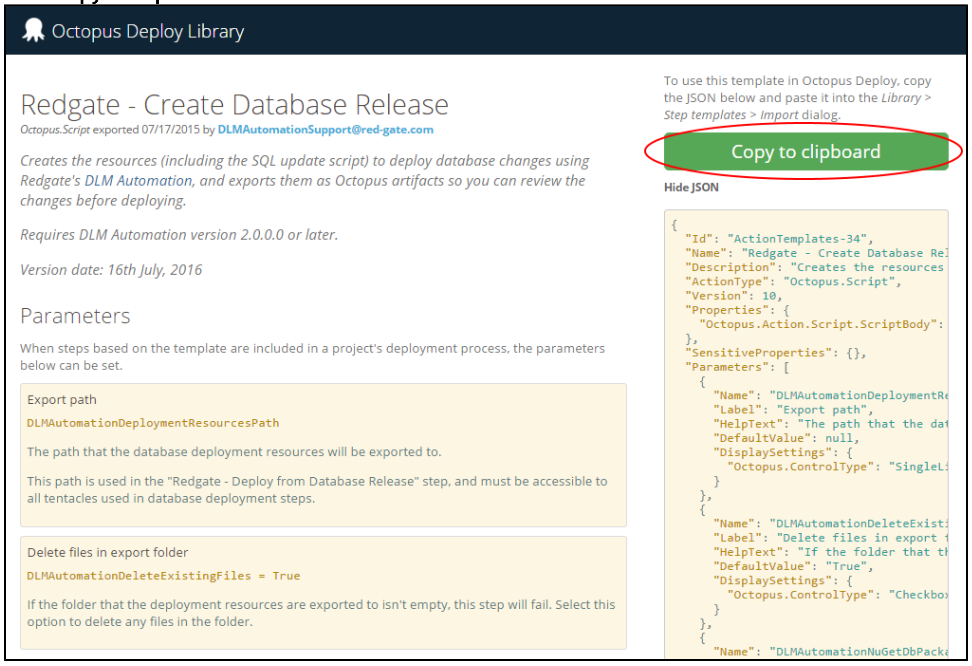
Before you start

1. Create a NuGet package containing a Redgate database scripts folder.
   If you use DLM Automation for continuous integration, your build server can automatically build a package on every commit. See examples of how to do this using TeamCity or Visual Studio Team Services.
2. Install DLM Automation. See Installing.
3. On the same machine, install Octopus Deploy. Click here for more details.
4. Install an Octopus Tentacle and assign the **db-server** role. Click here for more details.
5. Create an environment called **Production**, and add the machine running the Tentacle agent to it.

1. Copy the DLM Automation step templates to your Octopus Deploy library

After you've copied the DLM Automation step templates to your Octopus Deploy library, they're available whenever you add a process step in an Octopus Deploy project:

1. Go to the DLM Automation step templates on the Octopus Deploy library website.
2. In this example, we'll use the "Redgate - Create Database Release" and "Redgate - Deploy from Database Release" step templates. Click on the "Redgate - Create Database Release" template.
3. Click **Copy to clipboard**:



Redgate - Create Database Release

Octopus.Script exported 07/17/2015 by DLMAutomationSupport@red-gate.com

Creates the resources (including the SQL update script) to deploy database changes using Redgate's DLM Automation, and exports them as Octopus artifacts so you can review the changes before deploying.

Requires DLM Automation version 2.0.0.0 or later.

Version date: 16th July, 2016

Parameters

When steps based on the template are included in a project's deployment process, the parameters below can be set.

Export path
`DLMAutomationDeploymentResourcesPath`
The path that the database deployment resources will be exported to.
This path is used in the "Redgate - Deploy from Database Release" step, and must be accessible to all tentacles used in database deployment steps.

Delete files in export folder
`DLMAutomationDeleteExistingFiles = True`
If the folder that the deployment resources are exported to isn't empty, this step will fail. Select this option to delete any files in the folder.

To use this template in Octopus Deploy, copy the JSON below and paste it into the *Library > Step templates > Import* dialog.

```
{
  "Id": "ActionTemplates-34",
  "Name": "Redgate - Create Database Rel
  "Description": "Creates the resources
  "ActionType": "Octopus.Script",
  "Version": 10,
  "Properties": {
    "Octopus.Action.Script.ScriptBody":
  },
  "SensitiveProperties": {},
  "Parameters": [
    {
      "Name": "DLMAutomationDeploymentRe
      "Label": "Export path",
      "HelpText": "The path that the dat
      "DefaultValue": null,
      "DisplaySettings": {
        "Octopus.ControlType": "SingleLi
      }
    },
    {
      "Name": "DLMAutomationDeleteExisti
      "Label": "Delete files in export f
      "HelpText": "If the folder that th
      "DefaultValue": "True",
      "DisplaySettings": {
        "Octopus.ControlType": "Checkbox
      }
    },
    {
      "Name": "DLMAutomationNuGetDbPacka
```

You're now ready to paste the script from your clipboard into your Octopus Deploy library:

1. In Octopus Deploy, at the top of the page, click **Library**.
2. On the **Step templates** tab, click **Import**.
3. In the Import window, paste the copied template into the empty field.
4. Click **Import**.
5. Click **Save**.
6. Repeat steps 1 to 5 to copy the "Redgate - Deploy from Database Release" step template in the same way.

2. Create an Octopus Deploy project

1. In Octopus Deploy, click **Projects** and **All**.

2. Click **Add project**.
3. In the **Name** field, enter *Widget Deployment*.
4. Click **Save**.

You'll now add a series of deployment process steps to your Octopus Deploy project.

3. Add the "Download and extract database package" step

This step picks up the NuGet package of the database schema you're going to deploy.

1. Set up your NuGet package feed by doing one of the following:
    - register your existing external NuGet package feed with Octopus. For more details, see Adding external package feeds.
    - configure your build server to push packages to the Octopus built-in repository. For more details, see Using the built-in repository.
2. In the **Widget Deployment** project, on the **Process** tab, click **Add step** and select **Deploy a NuGet package**.
3. In the **Step name** field, enter *Download and extract database package*.
4. In the **Machine roles** field, enter *db-server* and press **Enter**.
   This must match the role you assigned to the Tentacle.
5. In the **NuGet feed** field, select either the name of the external feed you registered when you set up your NuGet feed, or the *Octopus Server (built-in)* repository.
6. In the **NuGet package ID** field, enter the name of the package without the version number. For example, if the package was called *WidgetShopLatest.0.1.nupkg*, you'd only enter *Widget.*
   When the package is generated, NuGet package manager automatically adds a number. If we included it here, Octopus would only deploy the package that matched that name and version number. By removing the number, we're telling Octopus to always look for the latest package with that name.
7. In the **Environments** field, select *Production*.
   If you leave this blank, the step will be accessible to all environments.
8. Click **Save**.

4. Add the "Create database release" step

This step creates the database deployment resources, including the Update.sql script.

1. On the project **Process** tab, click **Add step** and **Redgate - Create Database Release**.
2. In the **Machine roles** field, enter *db-server* and press **Enter**.
   This must match the role you assigned to the Tentacle.
3. In the **Export path** field, enter the path the database deployment resources will be exported to.
   This path will later be used in the "Deploy from Database Release" step. It must be accessible to all tentacles used in database deployment steps.
4. In the **Database package step** field, select *Download and extract database package*.
5. In the **Target SQL Server instance** field, enter the fully qualified SQL Server instance for the database you're deploying to.
6. In the **Target database name** field, enter the name of the database you're deploying to.
7. In the **Username (optional)** and **Password (optional)** fields, enter the SQL Server username and password used to connect to the target database.
   If you leave these blank, Windows authentication will be used to connect to the target database.
8. In the **Environments** field, select *Production*.
   If you leave this blank, the step will be accessible to all environments.
9. Click **Save**.

5. Add the "Review database deployment resources" step

This step pauses deployment to let you review the database deployment resources, including the Changes.html report, before allowing deployment to go ahead.

1. On the project **Process** tab, click **Add step** and select **Manual intervention required**.
2. In the **Step name** field, enter *Review database deployment resources.*
3. In the **Instructions** field, copy and paste this text:

```
Please review the schema and static data changes, warnings and SQL change script in 'Changes.html'.
```

4. In the **Environments** field, select *Production*.
   If you leave this blank, the step will be accessible to all environments.
5. Click **Save**.

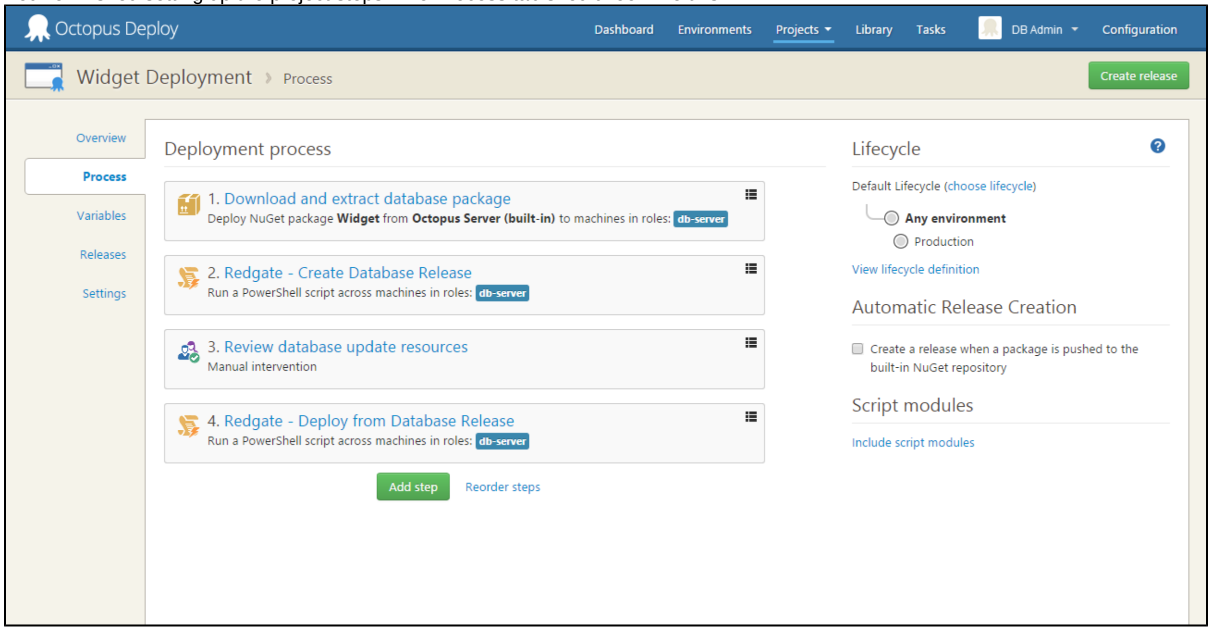
6. Add the "Deploy from database release" step

This step uses the database deployment resources to deploy the database changes.

1. On the project **Process** tab, click **Add step** and select **Redgate - Deploy from Database Release**.
2. In the **Machine roles** field, enter *db-server* and press **Enter**.
   This must match the role you assigned to the Tentacle.
3. In the **Export path** field, enter the path the database deployment resources will be exported to.
   This must match the export path you entered in 4. Add the 'Create database release' step.

4. In the **Database package step** field, select *Download and extract database package*.
5. In the **Target SQL Server instance** field, enter the fully qualified SQL Server instance for the database you're deploying to.
6. In the **Target database name** field, enter the name of the database you're deploying to.
7. In the **Username (optional)** and **Password (optional)** fields, enter the SQL Server username and password used to connect to the target database.
   If you leave these blank, Windows authentication will be used to connect to the target database.
8. In the **Environments** field, select *Production*.
   If you leave this blank, the step will be accessible to all environments.
9. Click **Save**.

You've finished setting up the project steps. The Process tab should look like this:



Deployment process

1. Download and extract database package
Deploy NuGet package **Widget** from **Octopus Server (built-in)** to machines in roles: db-server

2. Redgate - Create Database Release
Run a PowerShell script across machines in roles: db-server

3. Review database update resources
Manual intervention

4. Redgate - Deploy from Database Release
Run a PowerShell script across machines in roles: db-server

Lifecycle

Default Lifecycle (choose lifecycle)

Any environment
Production

View lifecycle definition

Automatic Release Creation

☐ Create a release when a package is pushed to the built-in NuGet repository

Script modules

Include script modules

7. Create a release

Now all the steps are set up, you can run your deployment process to create a release:

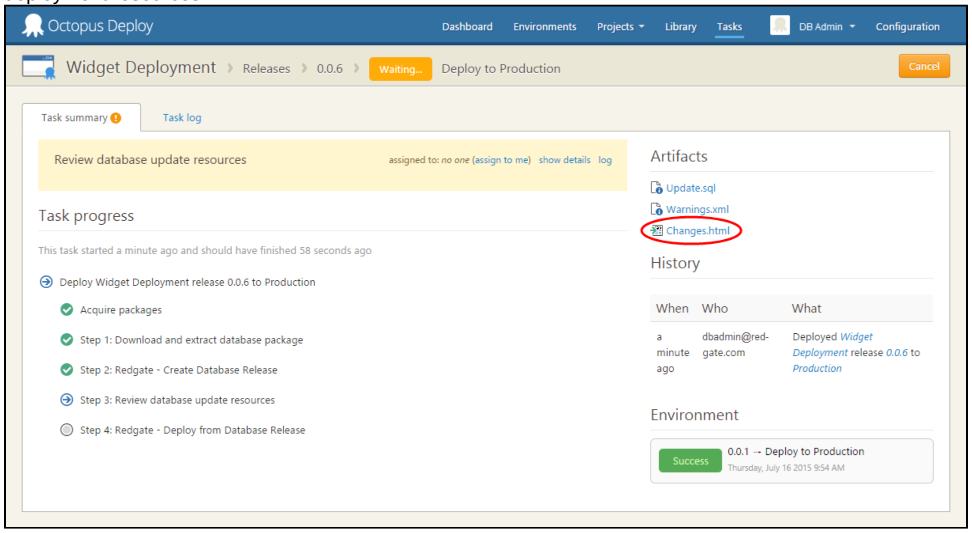
1. Create a blank database called *WidgetProduction*:

   a. Open SQL Server Management Studio (SSMS).
   b. Click **New Query**.
   c. Execute the following SQL query to create the database:

```
CREATE DATABASE WidgetProduction
GO
USE WidgetProduction
GO
```

2. In the **Widget Deployment** project, on the **Process** tab, click **Create release**.
   This page lets you add an optional release note.
3. Click **Save**.
4. Click **Deploy to Production** (or if there's more than one environment, click **Deploy** and select *Production*).

5. Click **Deploy Now**.
   As the deployment process runs, Octopus Deploy shows the task progress list. The deployment pauses so you can review the database deployment resources:



Review database update resources   assigned to: *no one* (assign to me)   show details   log

Task progress

This task started a minute ago and should have finished 58 seconds ago

- Deploy Widget Deployment release 0.0.6 to Production
  - Acquire packages
  - Step 1: Download and extract database package
  - Step 2: Redgate - Create Database Release
  - Step 3: Review database update resources
  - Step 4: Redgate - Deploy from Database Release

Artifacts

- Update.sql
- Warnings.xml
- Changes.html

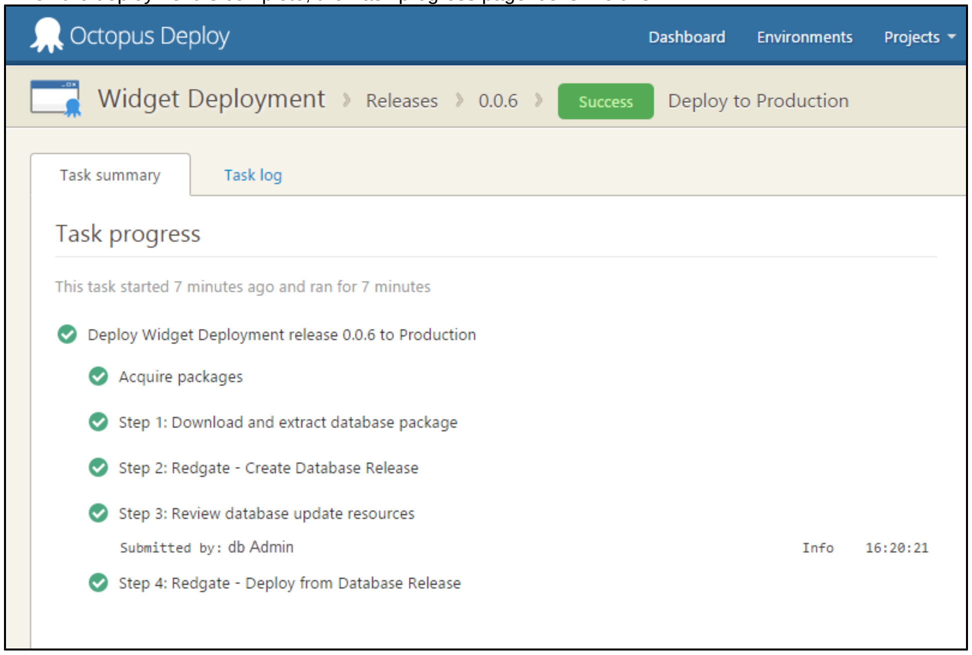
History

When	Who	What
a minute ago	dbadmin@red-gate.com	Deployed *Widget Deployment* release *0.0.6* to *Production*

Environment

Success   0.0.1 → Deploy to Production
Thursday, July 16 2015 9:54 AM

6. Click **Changes.html** to download the Change report.
   Use the report to review the update script, warnings, and details of what'll be added, removed or modified if you go ahead with deployment.
7. In Octopus Deploy, click **assign to me** and, in **Notes**, enter a comment to say you've reviewed the database deployment resources.
8. If you're happy with the report, click **Proceed**.
   When the deployment is complete, the Task progress page looks like this:



Task progress

This task started 7 minutes ago and ran for 7 minutes

- Deploy Widget Deployment release 0.0.6 to Production
  - Acquire packages
  - Step 1: Download and extract database package
  - Step 2: Redgate - Create Database Release
  - Step 3: Review database update resources

```
Submitted by: db Admin                                  Info     16:20:21
```

  - Step 4: Redgate - Deploy from Database Release

You've now completed the deployment of the database package.

What now?

Other DLM Automation step templates are available on the Octopus Deploy Library website:

- "Redgate - Deploy from Database" deploys the schema of a source database to a target database without a review step.
- "Redgate - Deploy from Package" deploys a NuGet package containing a database schema to a target database without a review step.

For more information, see Octopus Deploy step templates reference.