AWS Toolkit for Eclipse

Getting Started Guide

Version 2.0

AWS Toolkit for Eclipse: Getting Started Guide

What is the AWS Toolkit for Eclipse?

The AWS Toolkit for Eclipse is an open source plug-in for the Eclipse integrated development environment (IDE) that makes it easier for developers to develop, debug, and deploy Java applications that use Amazon Web Services. It enhances the Eclipse IDE with additional features:

- The AWS SDK for Java is included and installed as part of the Toolkit for Eclipse
- AWS Explorer, an interface to Amazon Web Services that allows you to manage your AWS resources from within the Eclipse environment.
- AWS Elastic Beanstalk Deployment and Debugging
- An AWS CloudFormation Template Editor
- Support for multiple AWS Accounts

This, the *AWS Toolkit for Eclipse Getting Started Guide*, describes how to set up and use the AWS Toolkit for Eclipse

Topics

What's in this Guide?

This documentation is divided into the following chapters:

About Amazon Web Services

Amazon Web Services (AWS) is a collection of digital infrastructure services that developers can leverage when developing their applications. The services include computing, storage, database, and application synchronization (messaging and queuing). AWS uses a pay-as-you-go service model. You are charged only for the services that you—or your applications—use. Also, to make AWS more approachable as a platform for prototyping and experimentation, AWS offers a free usage tier. On this tier, services are free below a certain level of usage. For more information about AWS costs and the Free Tier, go to Test-Driving AWS in the Free Usage Tier. To obtain an AWS account, go to the AWS home page and click Sign Up.

Getting Started with the AWS Toolkit for Eclipse

This section describes how to get started with the AWS Toolkit for Eclipse, including information about how to install and configure the Toolkit for Eclipse.

Topics

Prerequisites for Installing and Using the AWS Toolkit for Eclipse

The Toolkit for Eclipse has the following prerequisites:

- **An Amazon Web Services account** – To obtain an AWS account, go to the AWS home page and click **Sign Up Now**. Signing up will enable you to use all of the services offered by AWS.
- **A supported operating system** – The Toolkit for Eclipse is supported on Windows, Linux, OS X, or Unix.
- **Java 1.6 or later**
- **Eclipse IDE for Java Developers 3.6 or later** – We recommend using the Eclipse IDE for Java EE Developers 4.3 ("Kepler").

  The Enterprise Edition (EE) includes the Eclipse Web Tools Platform (WTP) and the Eclipse Data Tools Platform (DTP). The WTP is required to use Amazon Elastic Block Store features, and the DTP is required for Amazon SimpleDB features.

  You can also install the WTP and DTP separately.
- **(Optional) Google Android Development Tools (ADT)** – if you want Toolkit for Eclipse support for the AWS SDK for Android, you must have the ADT installed *before* installing the Toolkit for Eclipse.

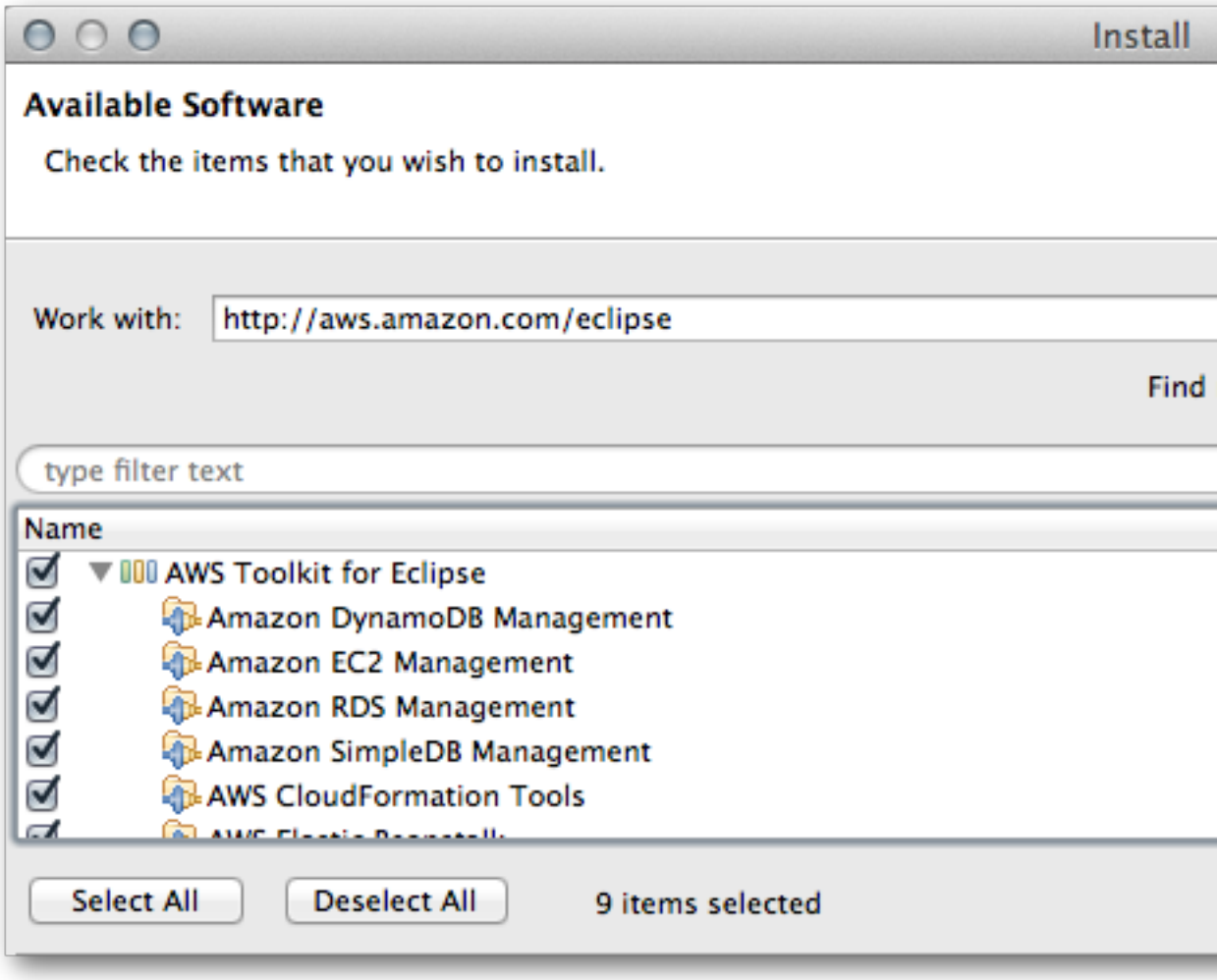
Once you have met these prerequisites, you can install the AWS Toolkit for Eclipse (p. 4).

Installing the AWS Toolkit for Eclipse

You can install the Toolkit for Eclipse using the Eclipse user interface.

To install the Toolkit for Eclipse from the AWS website using the Eclipse user interface

1. Start Eclipse.
2. Click **Help** and then click **Install New Software**.
3. In the **Work with** box, type **`http://aws.amazon.com/eclipse`** and then press **Enter**.
4. In the list that appears, expand **AWS Toolkit for Eclipse**.
5. Add a check mark next to **AWS Toolkit for Eclipse**.



Important
Support for the SDK for Android requires that you have the Google Android Development Tools (ADT) installed before installing the Toolkit for Eclipse. *If you do not have the ADT installed, you must uncheck* **AWS SDK for Android**, or an installation error will occur.

6. Click **Next**, and the Eclipse wizard will guide you through the remaining installation steps.

Once you have set up the AWS Toolkit for Eclipse you should configure your AWS Credentials (p. 5).

Working with Amazon Web Services Access Credentials

To access Amazon Web Services with Toolkit for Eclipse, you must configure the Toolkit for Eclipse with your AWS account credentials. In addition to allowing the Toolkit for Eclipse to access your account, your access keys are used to sign web services requests to AWS. Signing web services requests ensures that only authorized programs can make such requests. Also, by associating access keys with each web services request, AWS is able to track service usage for billing purposes.

Topics

Getting your AWS Access Keys

To get your access key ID and secret access key

Access keys consist of an access key ID and secret access key, which are used to sign programmatic requests that you make to AWS. If you don't have access keys, you can create them by using the AWS Management Console.

> **Note**
> To create access keys, you must have permissions to perform the required IAM actions. For more information, see Granting IAM User Permission to Manage Password Policy and Credentials in *Using IAM*.

1. Go to the IAM console.
2. From the navigation menu, click **Users**.
3. Select your IAM user name.
4. Click **User Actions**, and then click **Manage Access Keys**.
5. Click **Create Access Key**.

   Your keys will look something like this:

   - Access key ID example: AKIAIOSFODNN7EXAMPLE
   - Secret access key example: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

6. Click **Download Credentials**, and store the keys in a secure location.

   Your secret key will no longer be available through the AWS Management Console; you will have the only copy. Keep it confidential in order to protect your account, and never email it. Do not share it outside your organization, even if an inquiry appears to come from AWS or Amazon.com. No one who legitimately represents Amazon will ever ask you for your secret key.

Related topics

- What Is IAM? in *Using IAM*
- AWS Security Credentials in *AWS General Reference*

Adding your AWS Access Keys to the Toolkit for Eclipse

The Toolkit for Eclipse now uses the same system for locating and using AWS access keys as that used by the AWS CLI and AWS Java SDK. Access keys entered in the Eclipse IDE are saved to a *shared AWS credentials file* (called `credentials`) in the `.aws` sub-directory within your home directory.

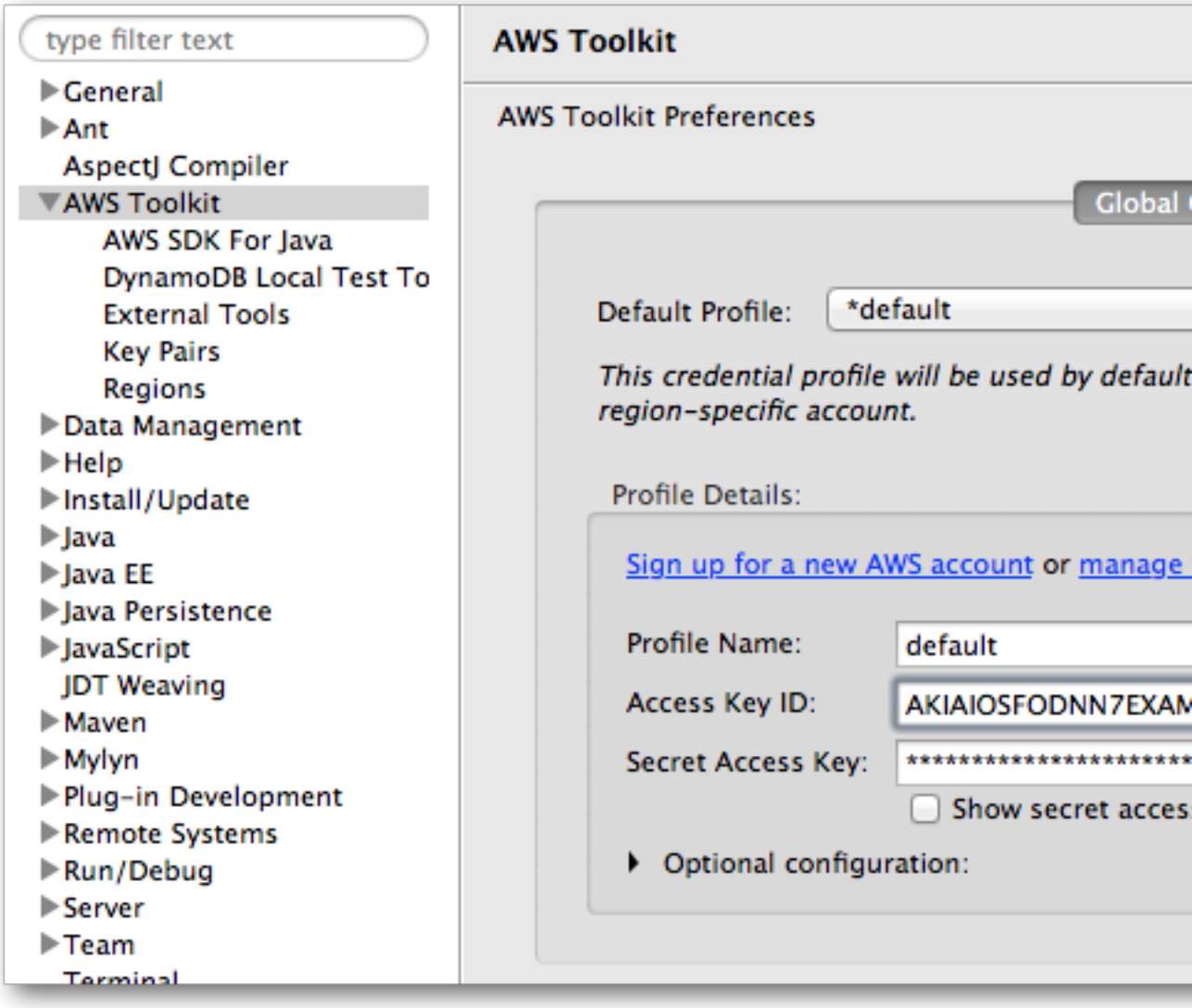
> **Note**
> The location of the credential file can be modified. For information about setting the location of this file, see Changing the AWS Credentials File Location (p. 8).

If you have already set your AWS credentials using the AWS CLI, then the Toolkit for Eclipse will automatically detect and use those credentials. For more information about using the AWS CLI, see the AWS Command Line Interface User Guide.

To add your access keys to the Toolkit for Eclipse

1. Open Eclipse's **Preferences** dialog box and click **AWS Toolkit** in the sidebar.
2. Type or paste your access key ID in the **Access Key ID** box.
3. Type or paste your secret access key in the **Secret Access Key** box.
4. Click **Apply** or **OK** to store your access key information.

Here's an example of a configured set of default credentials:



Using Multiple AWS Accounts with the Toolkit for Eclipse

The **Preferences** dialog box allows you to add information for more than one AWS account. Multiple accounts can be useful, for example, to provide developers and administrators with separate resources for development and for release/publication.

Separate sets of AWS credentials are stored as *profiles* within the shared AWS credentials file described in Adding your AWS Access Keys to the Toolkit for Eclipse (p. 6). All of the configured profiles can be seen in the drop-down box at the top of the AWS Toolkit Preferences Global Configuration screen, labeled **Default Profile**.

To add a new set of access keys

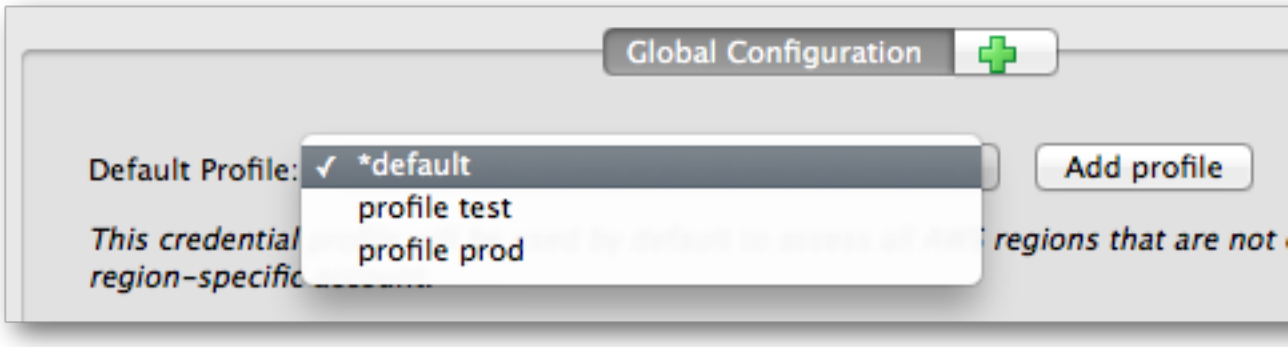
1. On the **AWS Toolkit Preferences** screen in Eclipse's **Preferences** dialog box, click **Add profile**.
2. Add your new account information to the **Profile Details** section.

Choose a descriptive name for the **Profile Name**, and enter your access key information in the **Access Key ID** and **Secret Access Key** boxes.

3. Click **Apply** or **OK** to store your access key information.

You can repeat this procedure for as many sets of AWS account information that you need.

When you have entered all of your AWS account information, select the default account by choosing one of the accounts from the **Default Profile** drop-down. AWS Explorer displays resources associated with the default account, and when you create a new application through the Toolkit for Eclipse, the application uses the credentials for the configured default account.



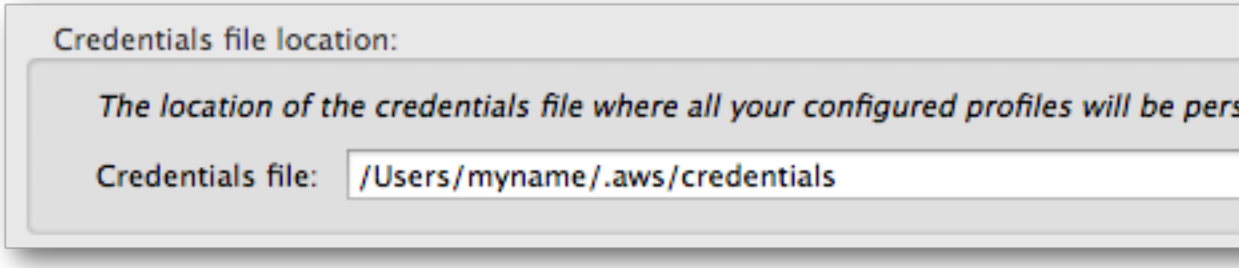
Note
For an alternative approach to separate your AWS resources, see Differentiating AWS Resources with Naming (p. 38).

Changing the AWS Credentials File Location

Using the Toolkit for Eclipse Preferences screen, you can change the location used by the Toolkit to store and load credentials.

To set the AWS credentials file location

- In the AWS Toolkit Preferences dialog, locate the **Credentials file location** section, and enter the pathname of the file where you would like your AWS credentials stored.



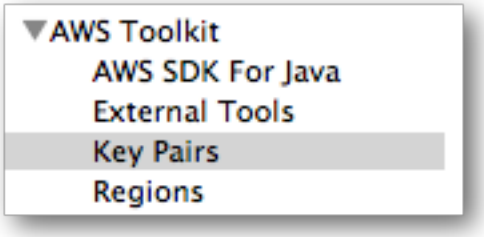
Important
It is *strongly recommended* that you don't store your AWS credential information within any network-shared directory or within any source-control-managed projects. You should always retain strict control of your AWS access keys!

Associate Private Keys with Your Amazon EC2 Key Pairs

The Toolkit for Eclipse can obtain your Amazon EC2 key pairs from AWS. However, you will need to associate private keys with them to use them in the Toolkit for Eclipse.

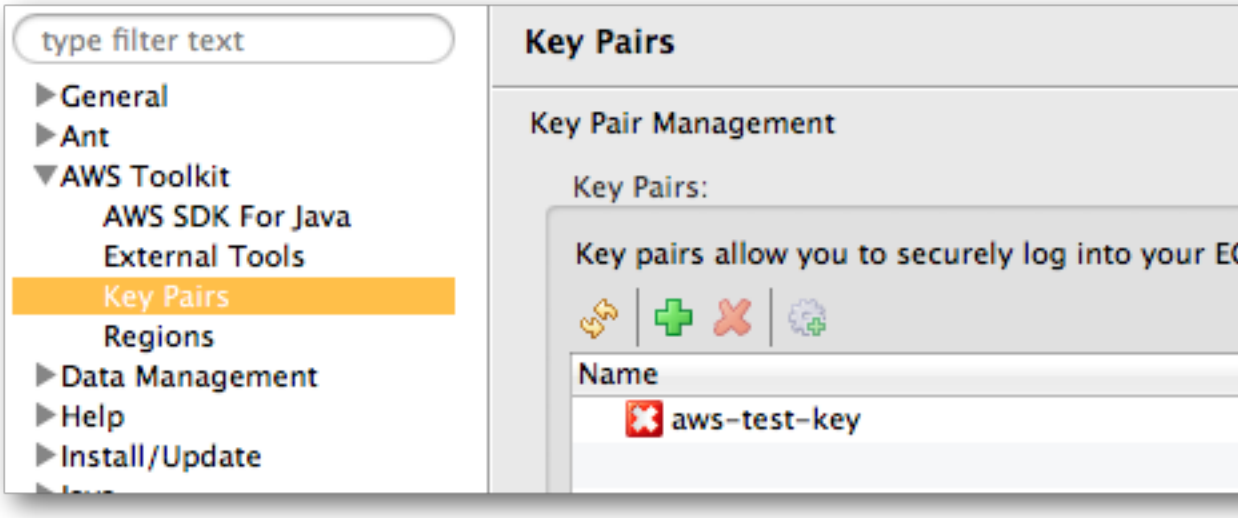
To view your Amazon EC2 key pairs in the Toolkit for Eclipse and associate private keys with them

1. Open Eclipse's **Preferences** dialog box and click the triangle next to **AWS Toolkit** in the sidebar to show additional categories of Toolkit for Eclipse settings.

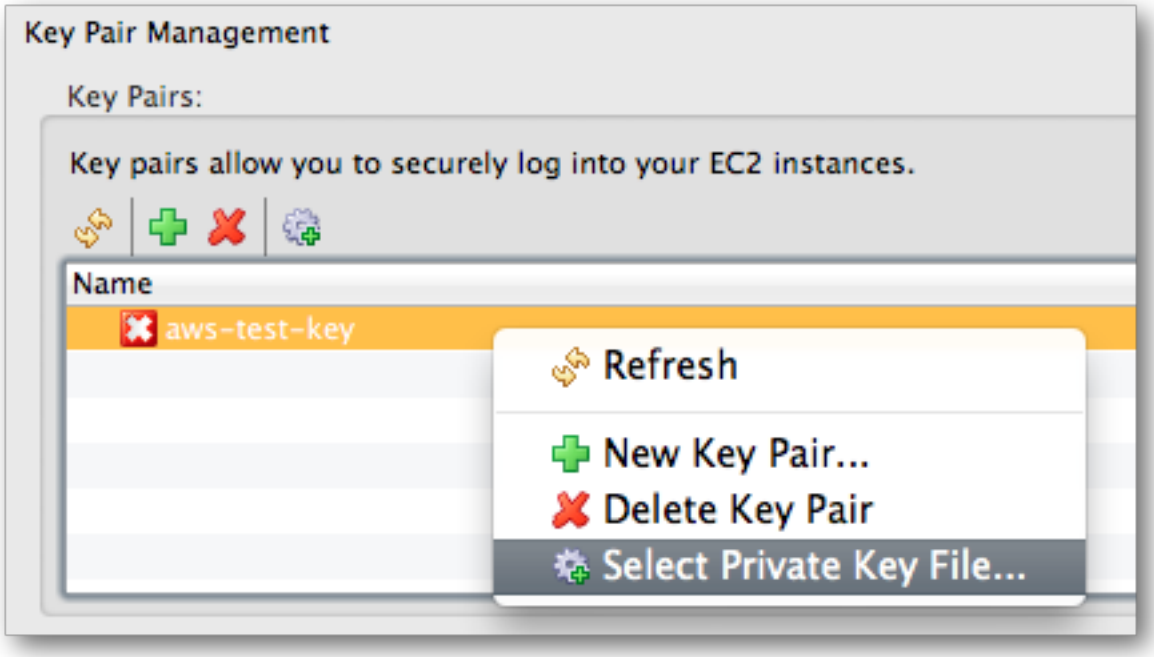


2. Select **Key Pairs**.

   Eclipse displays a scrollable list of your key pairs. If a key pair has a red **X** next to it, you will need to associate a private key with the key pair to use it.



3. Right-click the key pair and, from the context menu, select **Select Private Key File...**



4. Navigate to the private key file and select it to associate it with your key pair.

AWS Toolkit for Eclipse Basics

This chapter provides information about how to accomplish common development tasks with the AWS Toolkit for Eclipse.

Topics

Building a Java Application with Access to Amazon Web Services Using the AWS Toolkit for Eclipse

In this section, we'll use the AWS Toolkit for Eclipse to build and run a local Java application that accesses AWS resources.

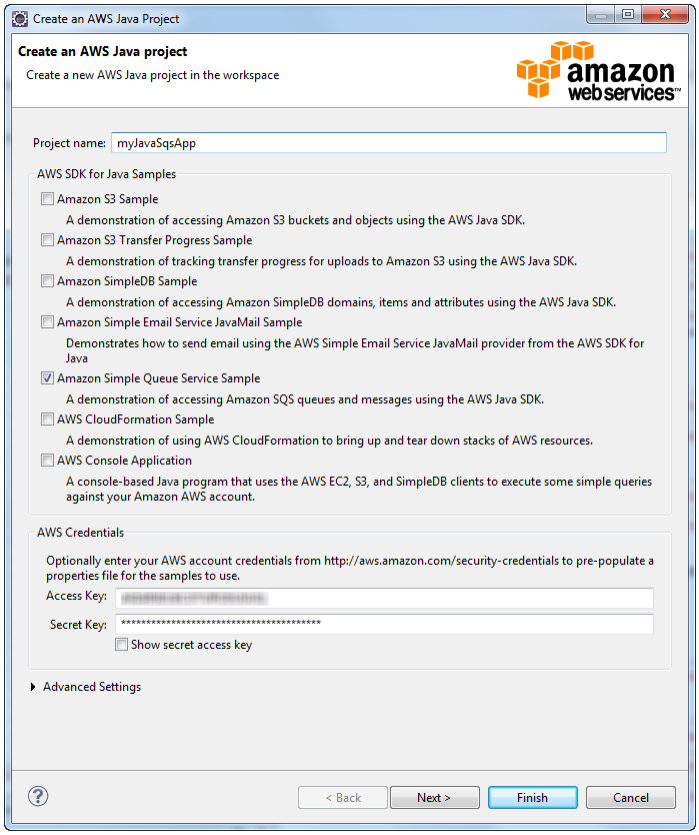
The AWS Toolkit for Eclipse includes the AWS SDK for Java and a number of Java sample programs. The Toolkit for Eclipse makes it easy to build and run any of these samples. To demonstrate how the Toolkit for Eclipse can help you build and run AWS applications in Java, we'll use the *AmazonSimpleQueueService* sample as an example. The AWS Explorer that is provided with the Toolkit for Eclipse can be used to view the running Amazon SQS queue.

> **Note**
> For a full list of the samples and more information about the SDK for Java, see AWS SDK for Java Code Samples in the *AWS SDK for Java Developer Guide*

Build and Run the Amazon Simple Queue Service Sample

To build and run the Amazon Simple Queue Service sample

1. Click the AWS icon on the Eclipse toolbar, and then click **New AWS Java Project**. In the dialog box that appears, type a name for the project in the **Project name** box and select **Amazon Simple Queue Service Sample**. Click **Finish**.



2. The sample application appears in **Project Explorer**. Expand the tree view for the project. Beneath the **src** node, double-click the SimpleQueueService.java source file to open it in the Eclipse editor pane. Locate the following line:

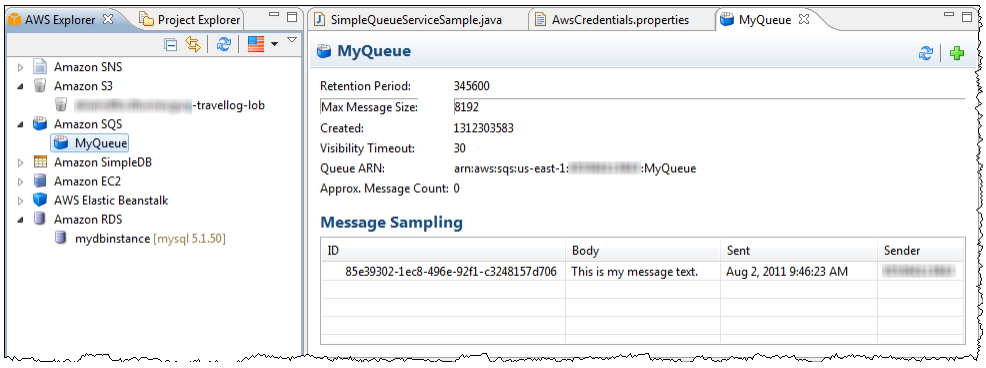
```
System.out.println("Receiving messages from MyQueue.\n");
```

   Right-click in the left margin of the editor pane, and select **Toggle Breakpoint**.

3. Right-click the project node in **Project Explorer**—in our example, this would be the node named **myJavaSqsApp**. Then click **Debug As | Debug Configurations**. In the **Debug Configurations** dialog box, select the SQS application, located beneath the **Java Application** node, and then click **Debug**.

4. When the application stops at the breakpoint, Eclipse will ask if it should switch to the Debug perspective. Click **No**; the Debug perspective does not include **AWS Explorer**.

   Instead, go to **AWS Explorer** and expand the **Amazon SQS** node. Double-click **MyQueue** and view the contents of the queue that was created by the Java client application.



5. Press **F8** and the Java client application will continue running and terminate normally. If you refresh the view in **AWS Explorer**, you will see that the **MyQueue** queue is no longer present; the application deletes the queue before the application exits.

   Note
   If you run this sample application repeatedly, you should wait at least 60 seconds between subsequent runs. Amazon SQS requires that at least 60 seconds elapse after deleting a queue before creating a queue with the same name.

Deploying an Application Using AWS Elastic Beanstalk

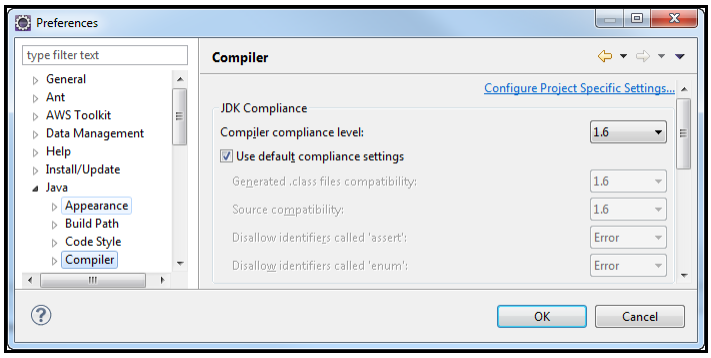
In this section, we'll use the AWS Toolkit for Eclipse to deploy a web application to AWS using the AWS Elastic Beanstalk service. The AWS Toolkit for Eclipse includes a sample application, Travel Log, that you can use to explore several of the Toolkit features. We'll use Travel Log as the web application to deploy. For more information about AWS Elastic Beanstalk, go to the AWS Elastic Beanstalk Developer Guide.

In addition to deploying AWS Elastic Beanstalk applications, the AWS Toolkit for Eclipse also enables you to *debug* such applications. For more information, go to the remote debugging section in the *AWS Elastic Beanstalk Developer Guide*.

When deploying to AWS Elastic Beanstalk from the AWS Toolkit for Eclipse, you must set your Java compiler target version to 1.6. To configure this setting in the Eclipse Toolkit, go to:

Window | **Preferences** | **Java** | **Compiler**

and set the **Compiler compliance level** to 1.6.

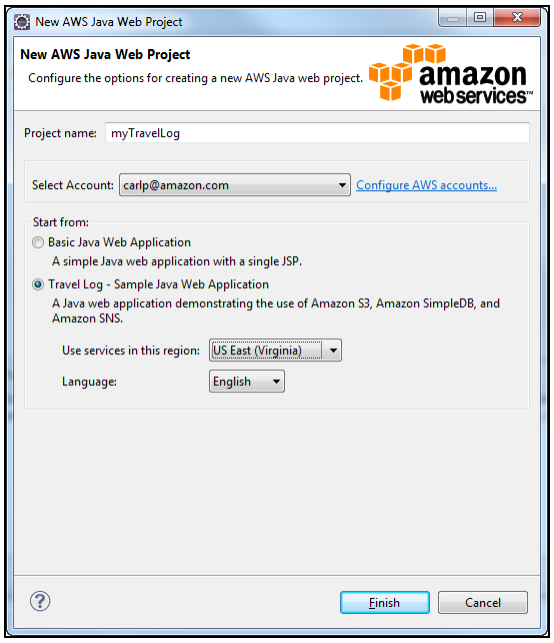


Deploy the Travel Log Application

To deploy the Travel Log application

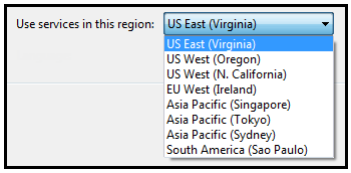
1. On the Eclipse toolbar, click the AWS icon, and then click **New AWS Java Web Project**. In the **New AWS Java Web Project** dialog box, in the **Start from** area of the dialog, select **Travel Log -- Sample Java Web Application** and enter a name, such as **`myTravelLog`** into the **Project name** box. Click **Finish**. The Toolkit creates the project, and the project appears in **Project Explorer**.

   If Project Explorer isn't visible in Eclipse, on the **Window** menu, click **Show View** and select **Project Explorer**.

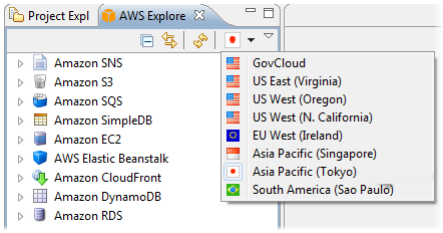


Configure AWS Region and Sample Language

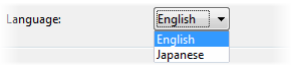
The AWS Java Web Project dialog box enables you to select either the US East or Asia Pacific region for the region in which your application runs.



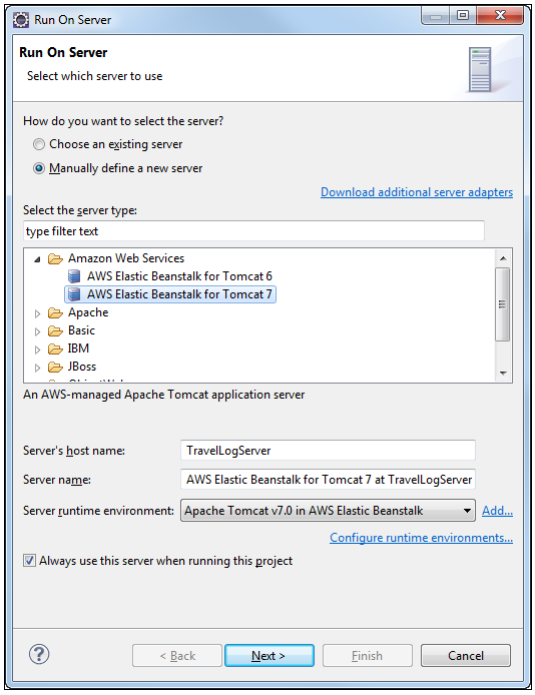
You should configure your deployment region to be the same as the region in which the application runs (configured above).



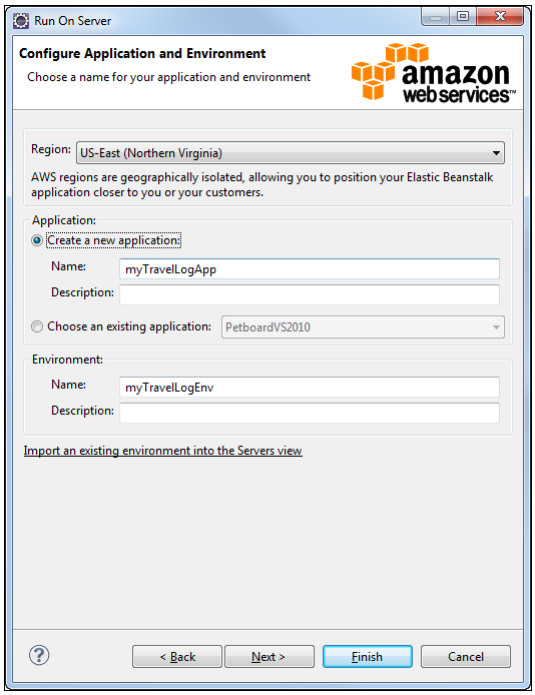
The Travel Log sample is available in both English and Japanese. Select your preferred language from the dialog box.



2. In **Project Explorer**, right-click the myTravelLog application, and then click **Run As | Run on Server**. In the **Run on Server** dialog box, click **Manually define a new server**, and then select **AWS Elastic Beanstalk for Tomcat 7** from the list of servers. Enter a name, such as **`TravelLogServer`** into the **Server's host name** box. Finally, select **Always use this server when running this project**. Click **Next**.



3. In the **Run On Server** dialog box, shown below, enter an application name, such as **`myTravelLogApp`**, and an environment name, such as **`myTravelLogEnv`**. Click **Next**.



4. The **Run On Server | Advanced configuration** dialog box enables you to specify additional parameters for your application deployment.

   Deploy with a key pair
   Select this option to use SSH to connect to the Amazon EC2 instance that hosts your application. By connecting to the Amazon EC2 instance, you can inspect the deployed application in the context of the server. AWS Elastic Beanstalk deploys your application to the **ROOT** subdirectory of the Tomcat **webapps** directory. For example:

   ```
   /opt/tomcat7/webapps/ROOT
   ```

   The dialog box lists your existing key pairs. If you see an X next to a key pair, you need to specify the private key file for that key pair. Right-click the key pair and from the context menu, choose **Select Private Key File**, and navigate to the private key file.

   The context menu also enables you to create a new key pair to use—or to delete one of the displayed key pairs.

   SSL certificate Id
   If your application requires an SSL certificate, use this box to specify the Amazon Resource Name (ARN) for the certificate stored in the AWS Identity and Access Management (IAM) service. For information about how to upload your certificate and obtain the certificate's ARN, see Creating and Upload Server Certificates in the IAM documentation.

   Assign CNAME to new server
   Use this box to specify a prefix for the domain name that AWS Elastic Beanstalk returns for the application server. If you do not specify a CNAME prefix, AWS Elastic Beanstalk will use your environment name as the prefix for the server's domain name.

   Application health check URL
   AWS Elastic Beanstalk will periodically check this URL to assess whether your application is active.

Email address for notifications

If you specify an email address in this text area, AWS Elastic Beanstalk will send notifications to the email address with information about the progress of the application deployment.

Use incremental deployment

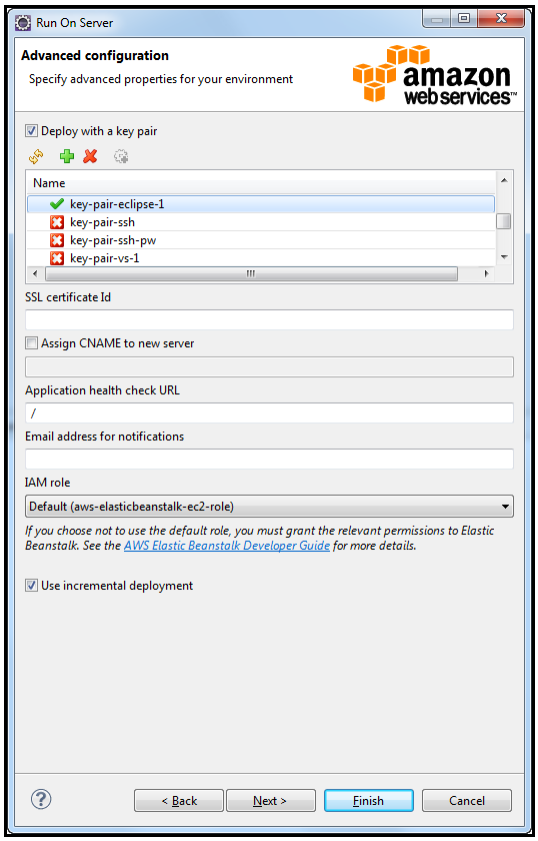
With incremental deployment, the first time that you deploy your application, all application files are copied to the server. If you later update some of your application files and redeploy, only the changed files are copied, which potentially reduces the amount of time required for redeployment. Without incremental deployment, all of your application files are copied to the server with each redeployment whether the files were changed or not. Select this check box to use incremental deployment.

IAM role

This dialog provides the option to select an IAM role. An IAM role provides applications and services access to AWS resources using temporary security credentials. For example, if your application requires access to DynamoDB, it must use AWS security credentials to make an API request. The application can use these temporary security credentials so you do not have to store long-term credentials on an Amazon EC2 instance or update the Amazon EC2 instance every time the credentials are rotated. In addition, AWS Elastic Beanstalk requires an IAM role to rotate logs to Amazon S3.

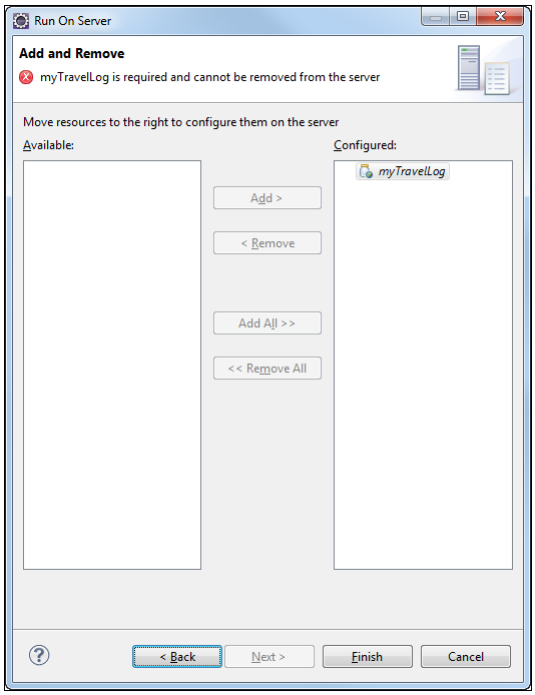
The IAM role list displays the roles available for your AWS Elastic Beanstalk environment. If you do not have an IAM role, you can select **Use the default role**. In this case, AWS Elastic Beanstalk creates a default IAM role and updates the Amazon S3 bucket policy to allow log rotation. If you choose not to use the IAM role, you need to grant permissions for AWS Elastic Beanstalk to rotate logs. For instructions, see Using a Custom Instance Profile. For more information about log rotation, see Configuring Containers with AWS Elastic Beanstalk. For more information about using IAM roles with AWS Elastic Beanstalk, see Using IAM Roles with AWS Elastic Beanstalk.

The credentials that you use for deployment must have permission to create the default IAM role.

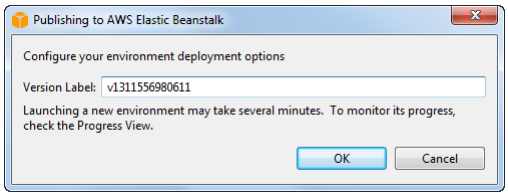


After you have specified any parameters appropriate to your application, click **Next**.

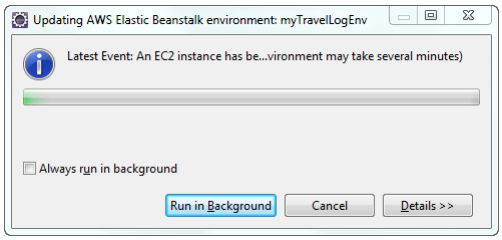
5. The **Run On Server | Add and Remove** dialog box, enables you to include additional resources with the deployment of your application. The left side of the dialog box shows resources that are present in your workspace. Use the **Add**, **Remove**, **Add All**, and **Remove All** buttons to transfer resources to and from the deployment.



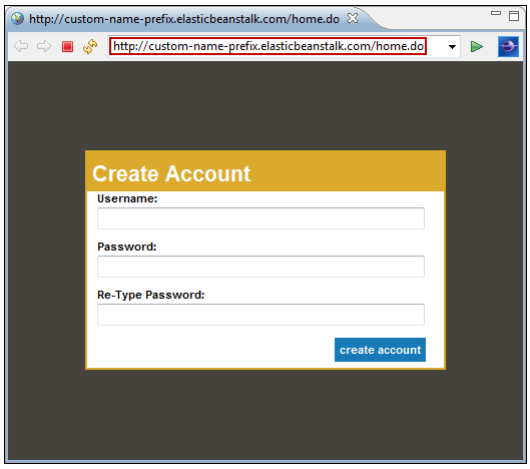
6. Before deploying your application to AWS, the Toolkit displays a dialog box in which you can set a **Version Label**. The AWS Toolkit generates a unique version label based on the current time. Simply leave this default version label and click **OK**.



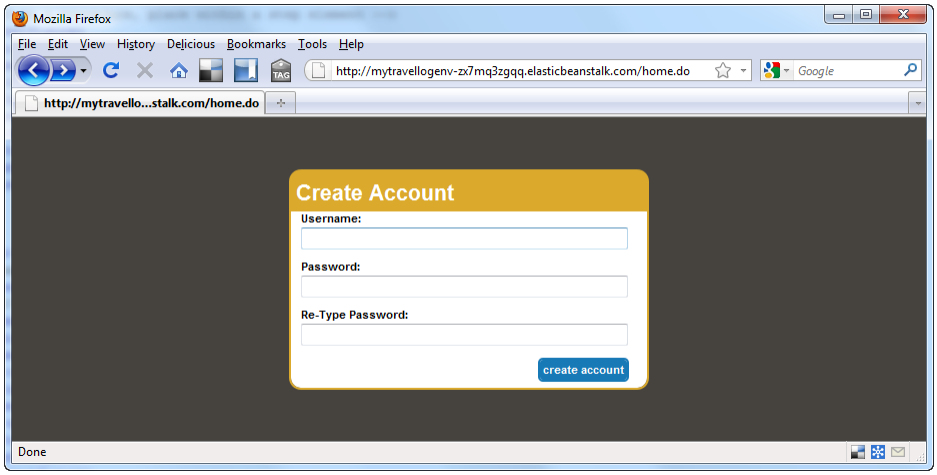
7. While your application is deploying, the Toolkit displays a progress message.



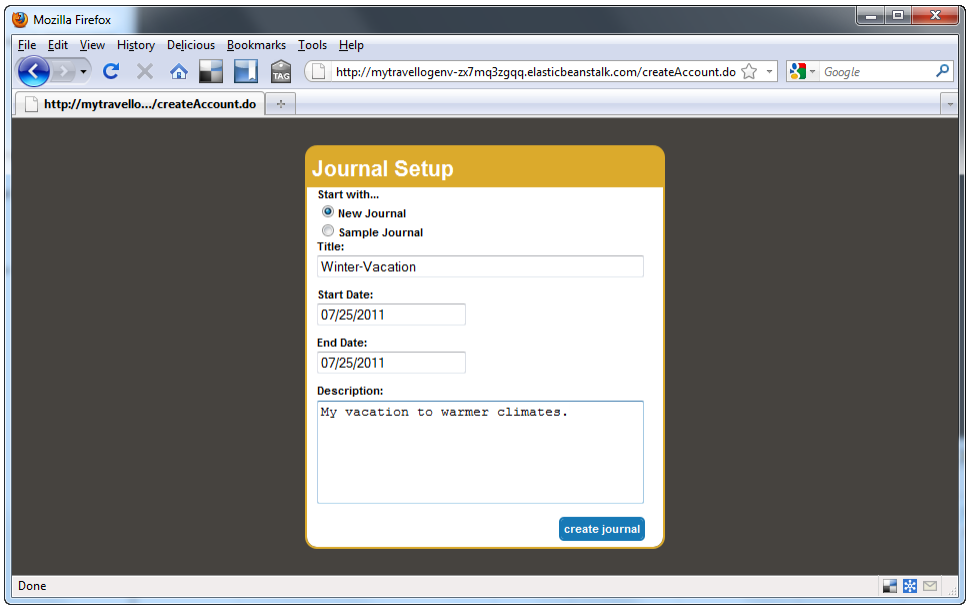
8. When the deployment completes, you will see the following image displayed in the Eclipse editor pane. This is the user interface for the Travel Log application, which is now running on an Amazon EC2 instance.

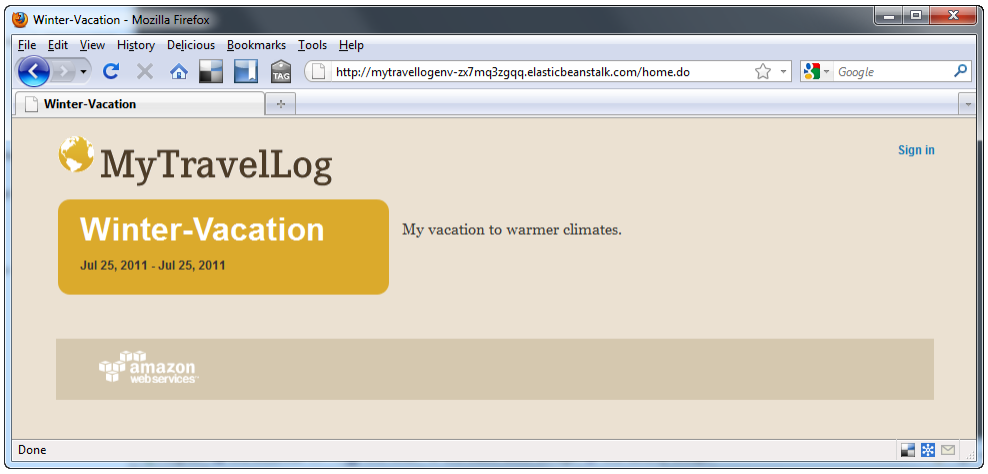


The Amazon EC2 instance that hosts the travel log application is running on the public Internet; if you copy the URL for the application, indicated in the preceding image, and paste it into a regular web browser, the Travel Log UI will appear.

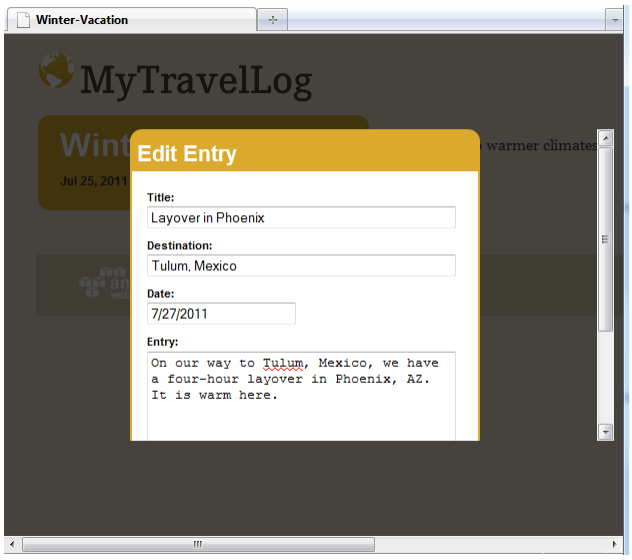


9. Create an account for yourself. Then start a new Travel Log journal, or select **Sample Journal** to load in a sample journal that comes with Travel Log.



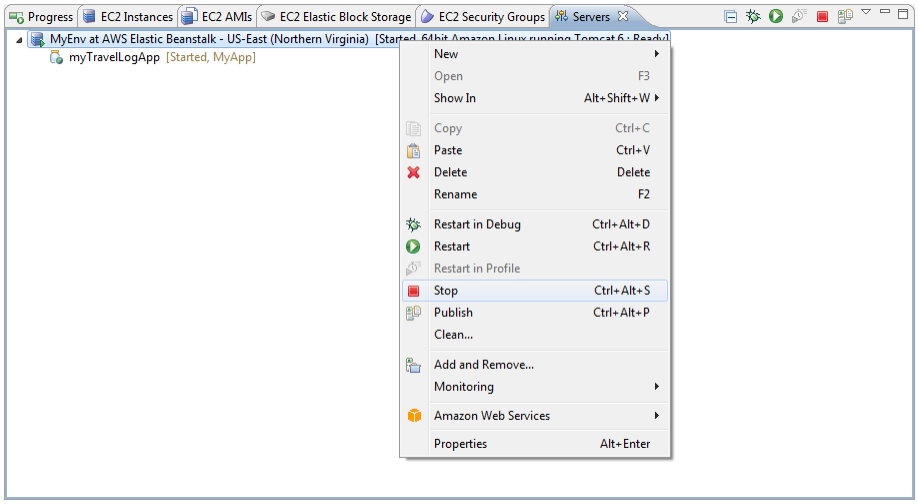


10. Sign in to the journal and create a new entry. You might need to scroll down to see the **Save** button for the new entry.



We will use your account and journal entry as we explore other features of the AWS Toolkit for Eclipse.

11. When you are done, simply close the browser window. Then shut down the Travel Log application in order to avoid incurring any further charges from AWS. To shut down the application, navigate to the Eclipse servers view, right-click the environment node (MyEnv in the example), and then click **Stop**.



If the Eclipse servers view isn't visible, click the **Window** menu, and then click **Show View | Other**.

Using the AWS CloudFormation Template Editor for Eclipse

The AWS Toolkit for Eclipse now includes a built-in AWS CloudFormation template editor. Among the features supported:

- The ability to create and update stacks directly from the Eclipse IDE from the currently-edited template.

- A JSON validator to help ensure that your template complies with JSON formatting and content rules.

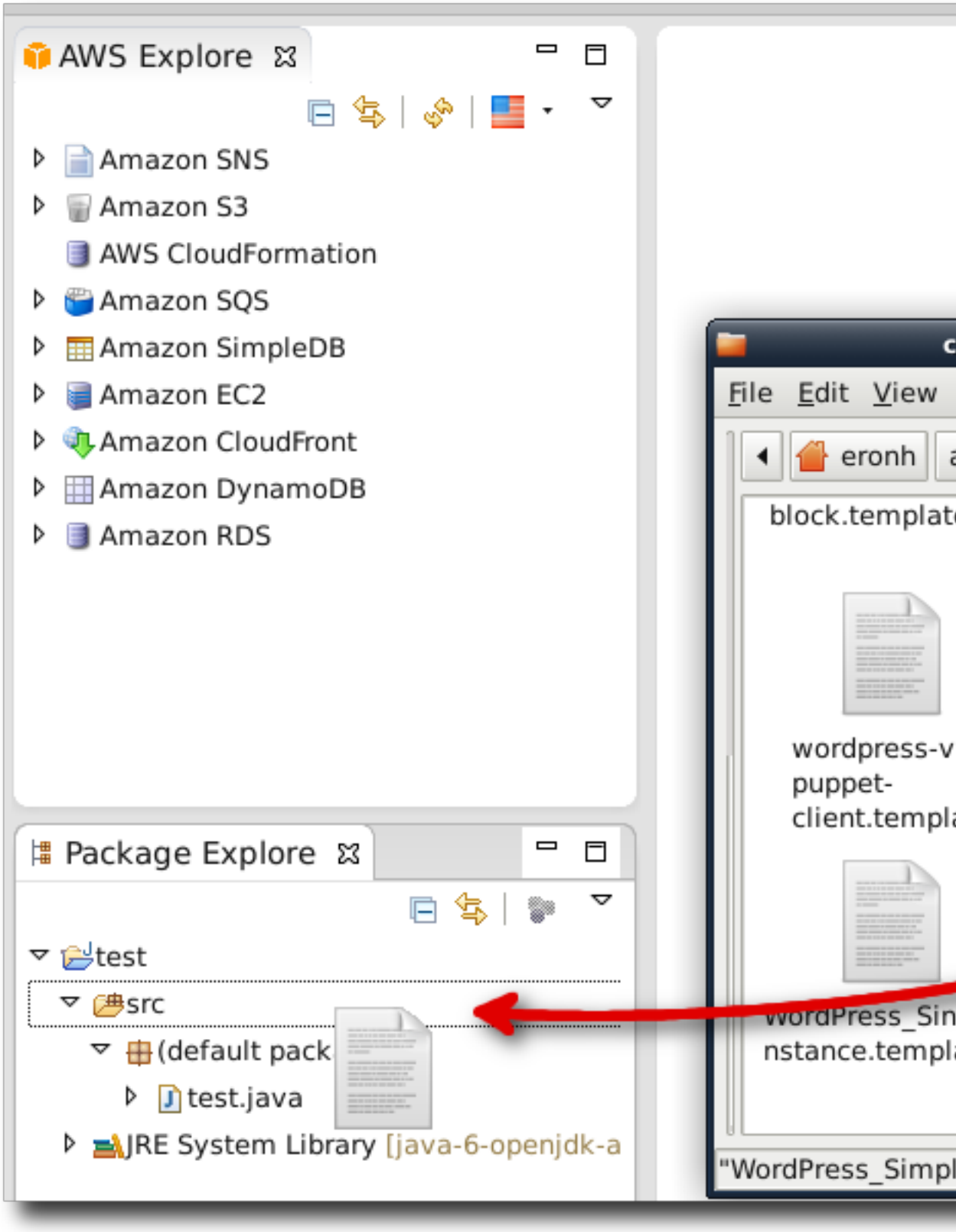
This section provides procedures for accomplishing common tasks with the AWS CloudFormation template editor.

In This Section

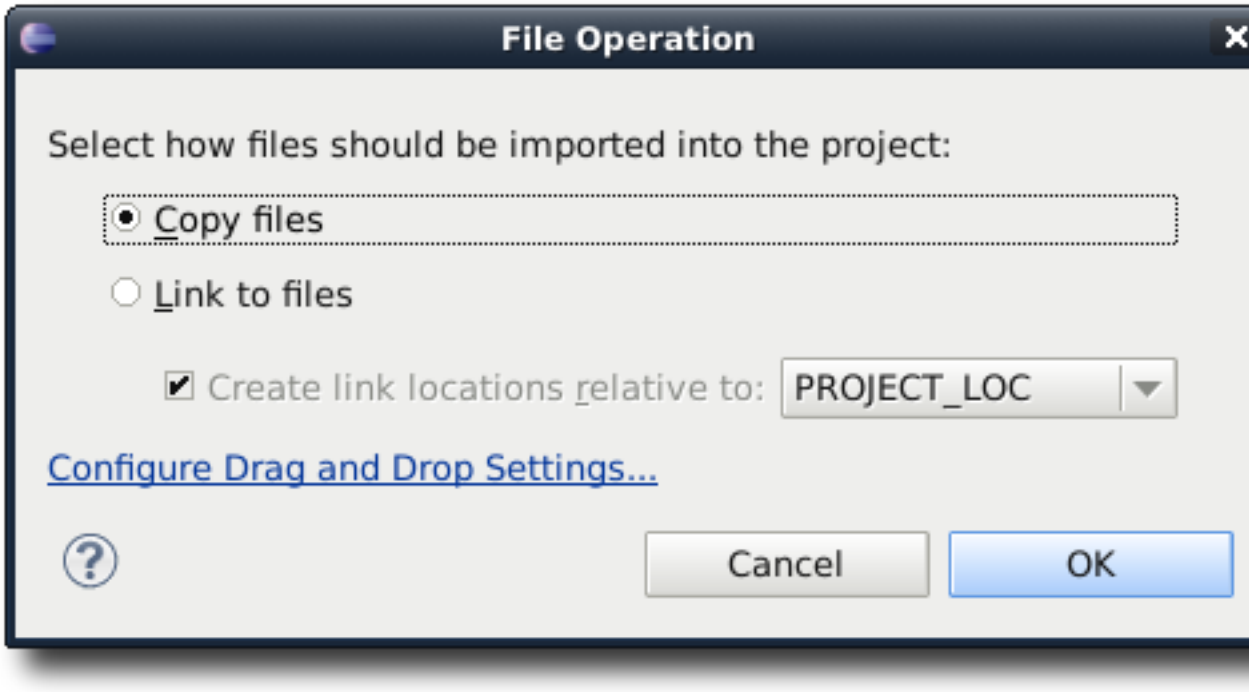
Adding and Accessing AWS CloudFormation Templates in Your Eclipse Project

To add a AWS CloudFormation template to your Eclipse project

1. Locate the template you'd like to add to your project in your system's file manager, and drag the file into your project's **Package Explorer** window.


AWS Explore
Amazon SNS
Amazon S3
AWS CloudFormation
Amazon SQS
Amazon SimpleDB
Amazon EC2
Amazon CloudFront
Amazon DynamoDB
Amazon RDS
Package Explore
test
src
(default pack
test.java
JRE System Library [java-6-openjdk-a
File Edit View
eronh
block.templat
wordpress-v
puppet-
client.templa
WordPress_Sin
nstance.templ
"WordPress_Simpl

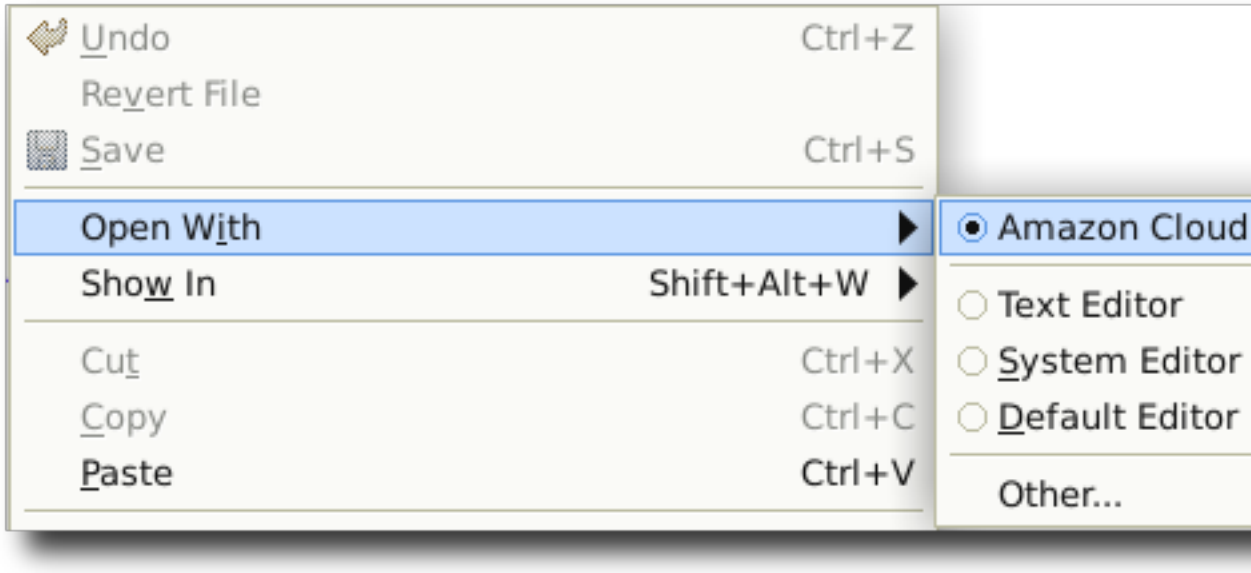
2. Choose how you would like to add the file to your project, and click **OK**.



To access a AWS CloudFormation template in your Eclipse project

- Double-click the template name in **Package Explorer** to begin editing the file.

Note
Files that end with `.template` or `.json` will automatically use the AWS CloudFormation template editor. If your file is not automatically recognized as an AWS CloudFormation template, you can select the editor by right-clicking the file name in **Package Explorer** or by right-clicking in the editor window with the file loaded, selecting **Open With**, then **AWS CloudFormation Template Editor**

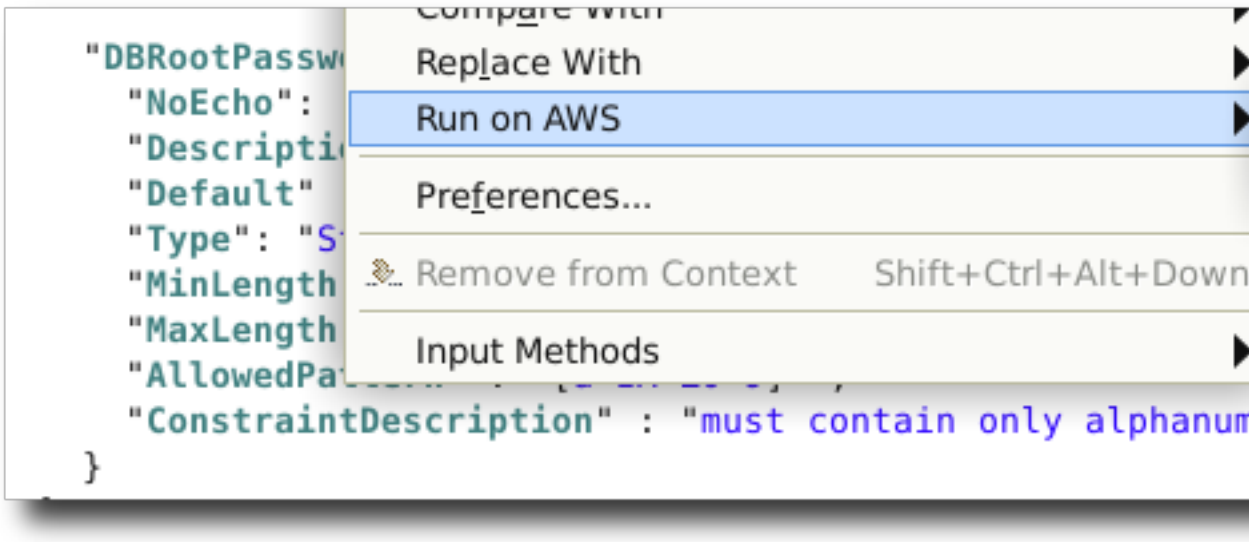


Deploying an AWS CloudFormation Template in Eclipse

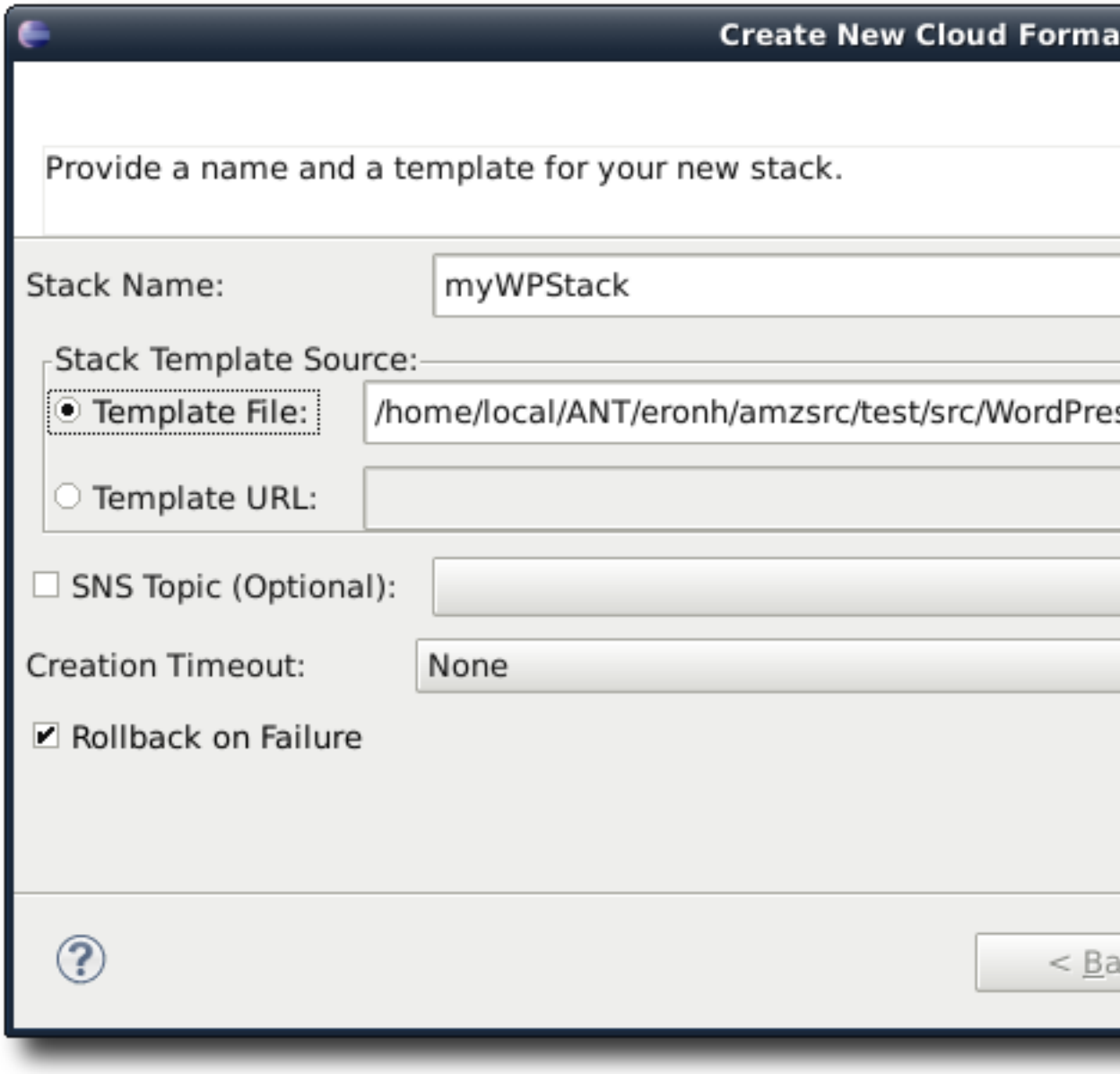
Note
At this time, only files that end in `.template` can be launched from the Eclipse IDE. If your file ends with another extension, such as `.json`, you will need to rename it first with a `.template` extension to use this feature.

To deploy an AWS CloudFormation template from Eclipse

1. With your AWS CloudFormation `.template` file open in the AWS CloudFormation template editor (see Adding and Accessing Templates (p. 23) for more information), right-click on the open template and select **Run on AWS**, then **Create Stack** on the context menu.



2. In the **Create New CloudFormation Stack** dialog, enter your stack name in the **Stack Name** field. Your template file should be automatically chosen in the **Template File** field.

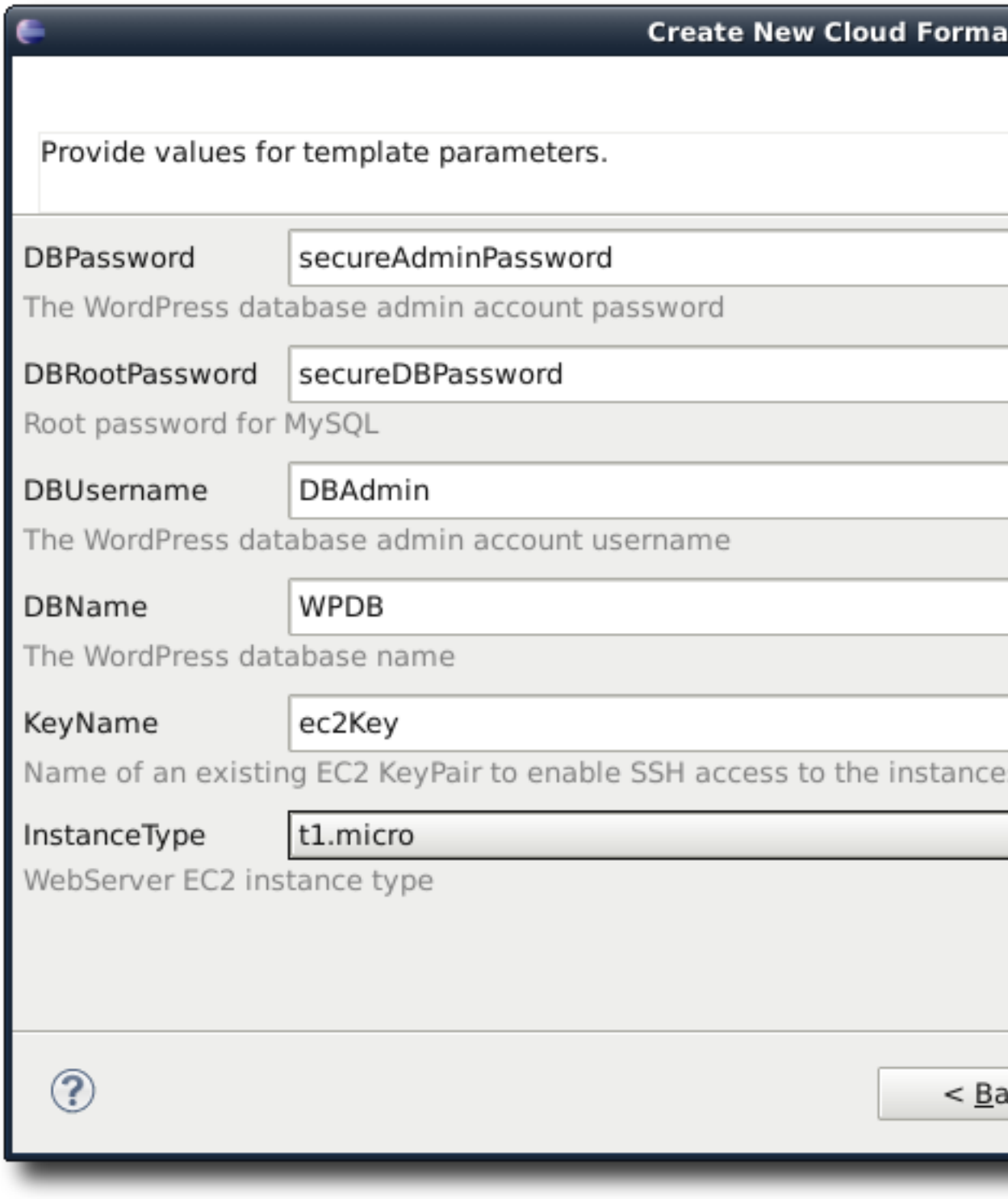


3. Choose any (or none) of the following options:

   - **SNS Topic** – choose an existing SNS topic from the list to receive notifications about the stack's progress, or create a new one by typing an email address in the box and clicking **Create New Topic**.
   - **Creation Timeout** – choose how long AWS CloudFormation should allow for the stack to be created before it is declared failed (and rolled back, unless the **Rollback on failure** option is unchecked.
   - **Rollback on failure** – if you want the stack to rollback (delete itself) on failure, check this option. Leave it unchecked if you would like the stack to remain active, for debugging purposes, even if it has failed to complete launching.

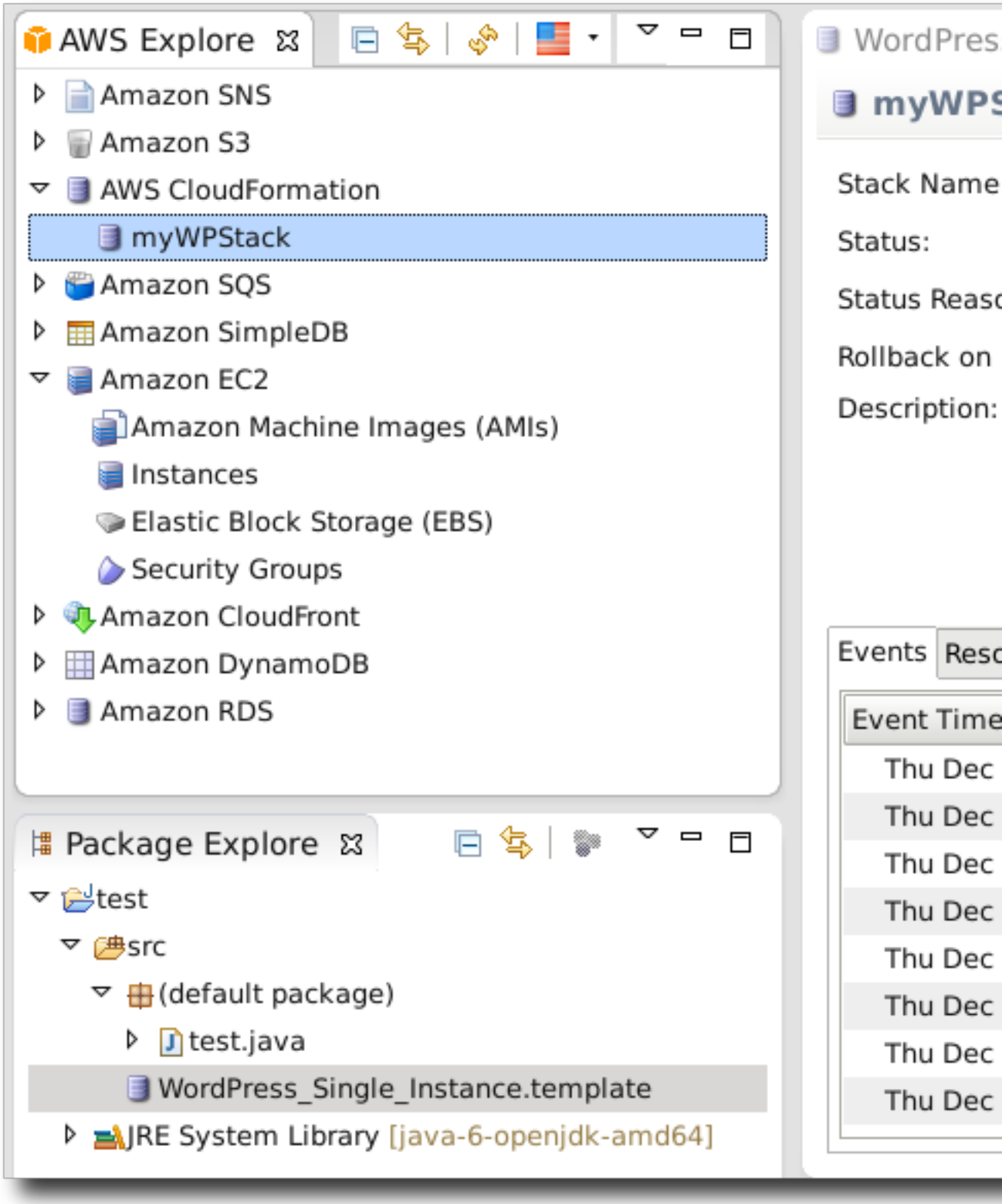
4. Click **Next** to continue with entering parameter values.

5. If your stack has parameters, you will enter values for them next. For parameters with a predefined list of possible responses, you can choose a value from the list provided.

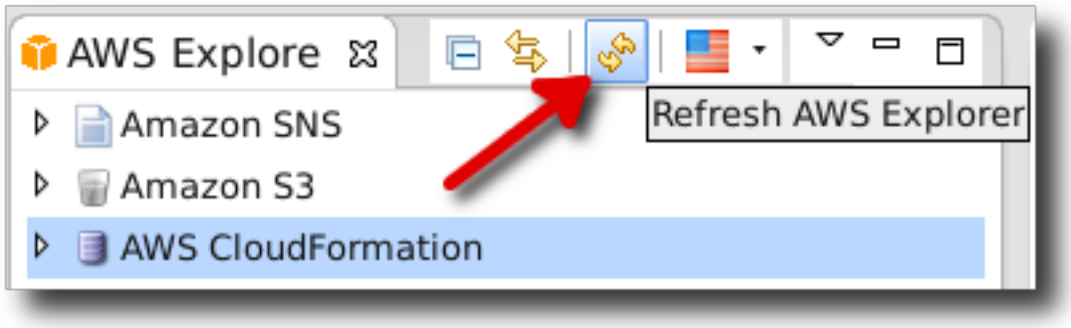


6. Click **Finish** to begin launching your stack.

While your stack is being launched, you can view its status by double-clicking the stack name beneath the **AWS CloudFormation** node in the **AWS Explorer** view, or by right-clicking the stack name and selecting **Open in Stack Editor** on the context menu.


AWS Explore
Amazon SNS
Amazon S3
AWS CloudFormation
myWPStack
Amazon SQS
Amazon SimpleDB
Amazon EC2
Amazon Machine Images (AMIs)
Instances
Elastic Block Storage (EBS)
Security Groups
Amazon CloudFront
Amazon DynamoDB
Amazon RDS
Package Explore
test
src
(default package)
test.java
WordPress_Single_Instance.template
JRE System Library [java-6-openjdk-amd64]
WordPres
myWPS
Stack Name
Status:
Status Reaso
Rollback on
Description:
Events
Resc
Event Time
Thu Dec
Thu Dec
Thu Dec
Thu Dec
Thu Dec
Thu Dec
Thu Dec
Thu Dec

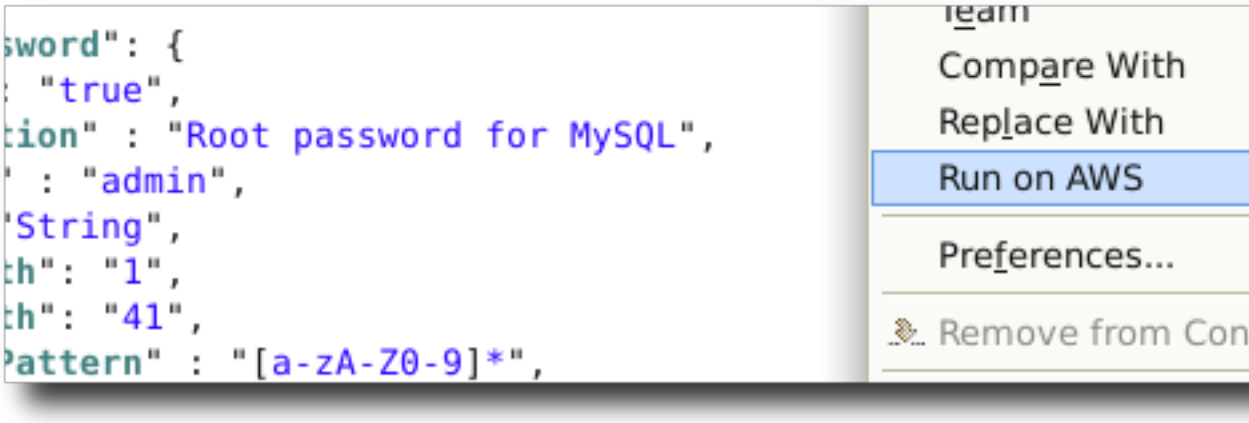
Note
If you cannot see the stack you launched in **AWS Explorer**, you may need to manually refresh the view by clicking the **Refresh AWS Explorer** icon at the top of the **AWS Explorer** view.



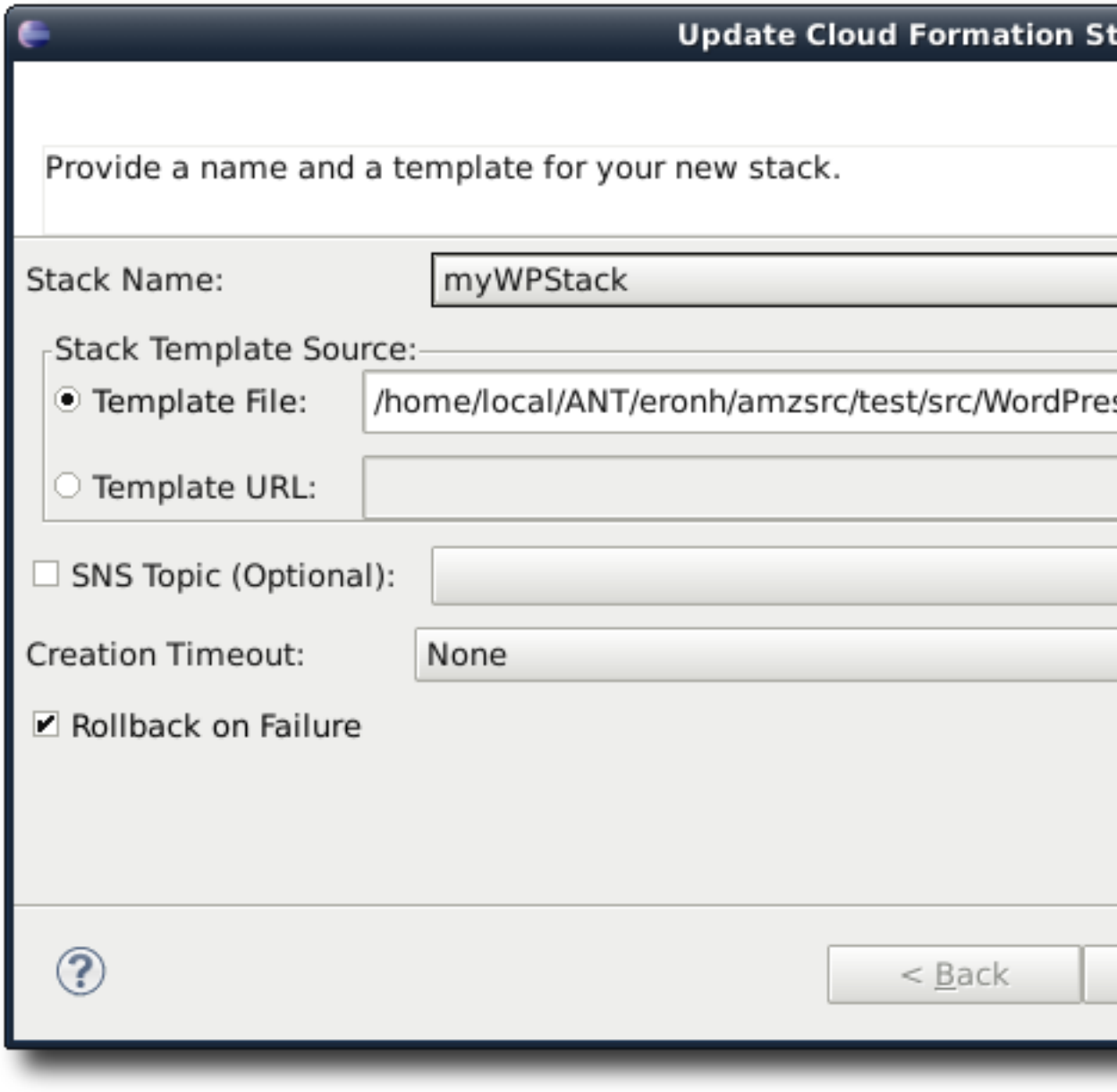
Updating an AWS CloudFormation Template in Eclipse

To update an AWS CloudFormation template from Eclipse

1. With your AWS CloudFormation `.template` file open in the AWS CloudFormation template editor (see Adding and Accessing Templates (p. 23) for more information), right-click on the open template and select **Run on AWS**, then **Update Stack** on the context menu.



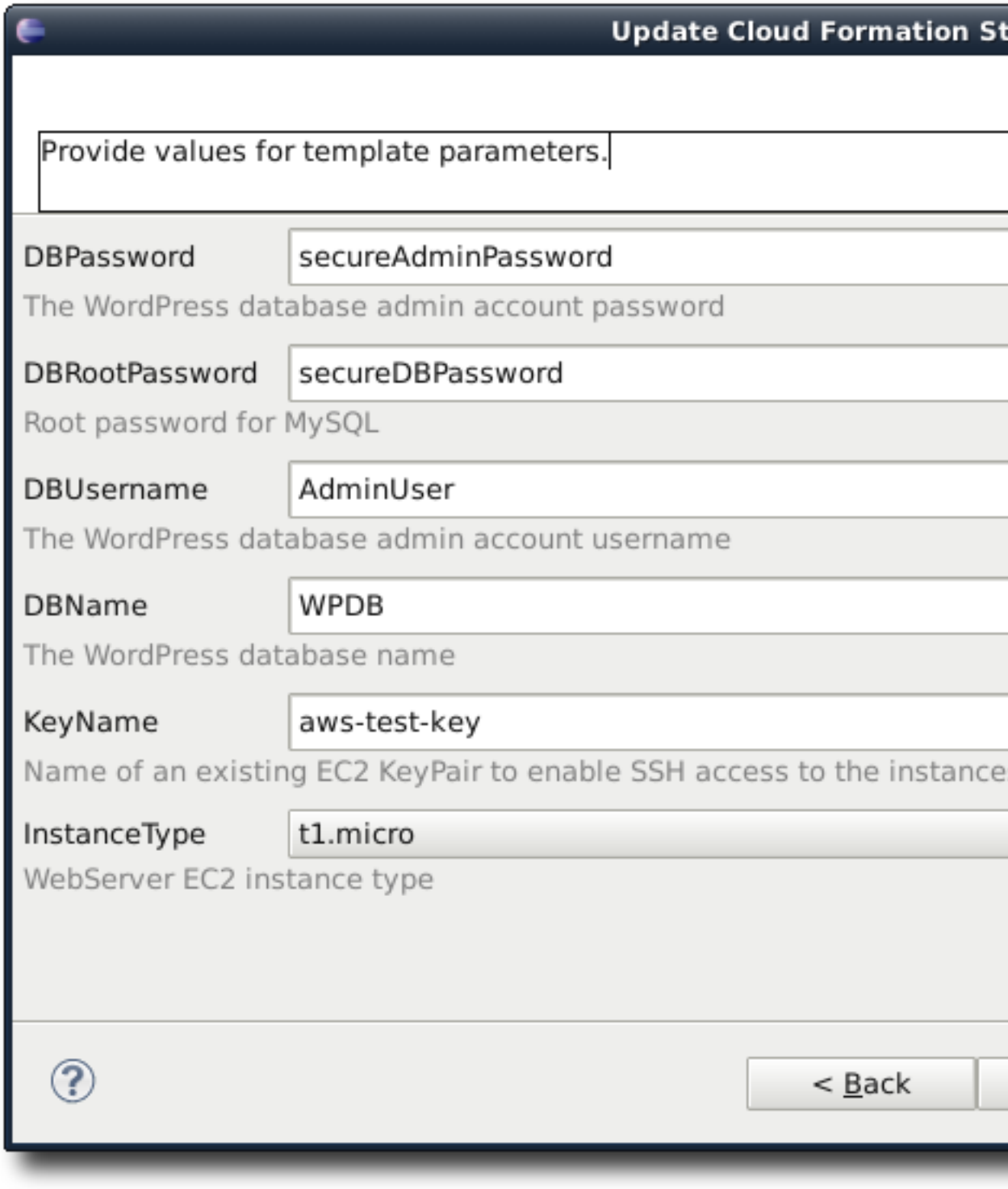
2. In the **Update CloudFormation Stack** dialog, select your stack name in the **Stack Name** field if it has not been automatically selected for you. Your template file should also be automatically chosen in the **Template File** field.



3. Choose any (or none) of the following options:

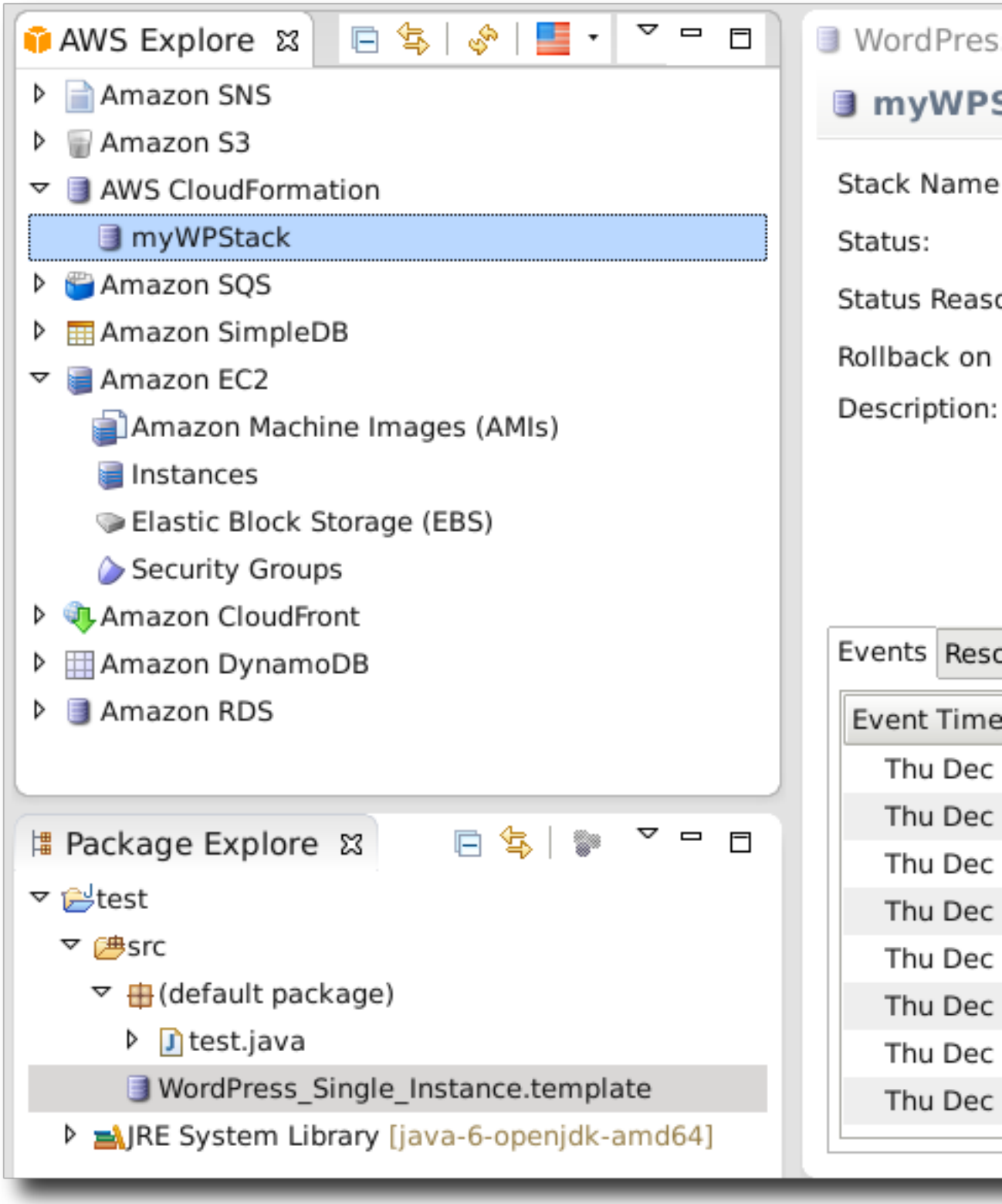
   - **SNS Topic** – choose an existing SNS topic from the list to receive notifications about the stack's progress, or create a new one by typing an email address in the box and clicking **Create New Topic**.
   - **Creation Timeout** – choose how long AWS CloudFormation should allow for the stack to be created before it is declared failed (and rolled back, unless the **Rollback on failure** option is unchecked.
   - **Rollback on failure** – if you want the stack to rollback (delete itself) on failure, check this option. Leave it unchecked if you would like the stack to remain active, for debugging purposes, even if it has failed to complete launching.

4. Click **Next** to continue with entering parameter values.
5. If your stack has parameters, you will enter values for them next. For parameters with a predefined list of possible responses, you can choose a value from the list provided.



6. Click **Finish** to begin updating your stack.

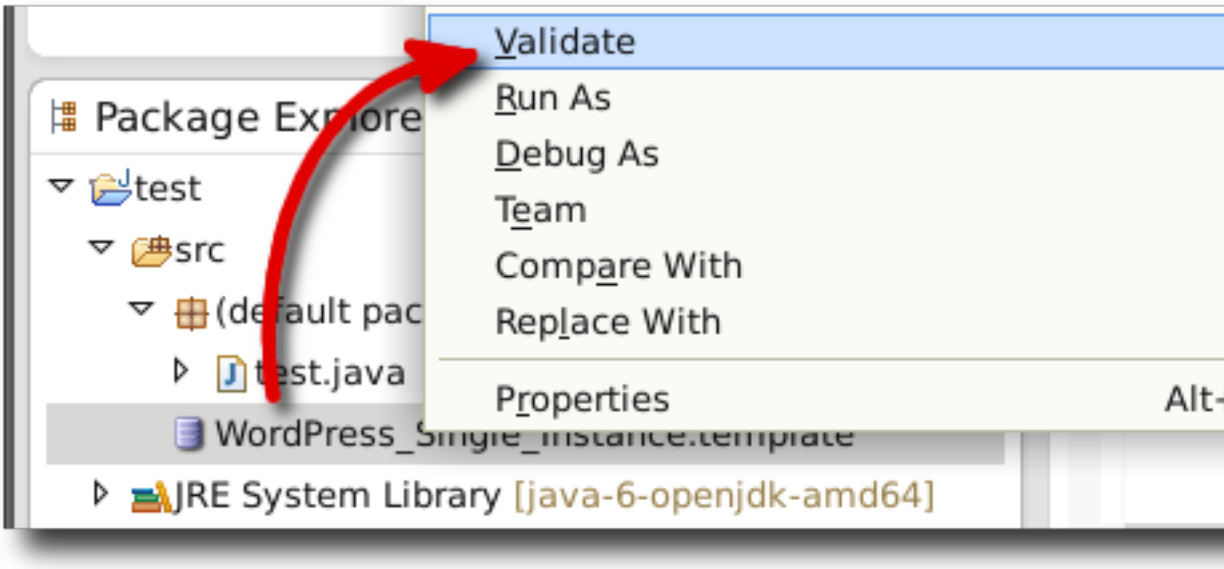
While your stack is being updated, you can view its status by double-clicking the stack name beneath the **AWS CloudFormation** node in the **AWS Explorer** view, or by right-clicking the stack name and selecting **Open in Stack Editor** on the context menu.


AWS Explore
Amazon SNS
Amazon S3
AWS CloudFormation
myWPStack
Amazon SQS
Amazon SimpleDB
Amazon EC2
Amazon Machine Images (AMIs)
Instances
Elastic Block Storage (EBS)
Security Groups
Amazon CloudFront
Amazon DynamoDB
Amazon RDS
Package Explore
test
src
(default package)
test.java
WordPress_Single_Instance.template
JRE System Library [java-6-openjdk-amd64]
WordPres
myWPS
Stack Name
Status:
Status Reaso
Rollback on
Description:
Events
Resc
Event Time
Thu Dec
Thu Dec
Thu Dec
Thu Dec
Thu Dec
Thu Dec
Thu Dec
Thu Dec

Validating an AWS CloudFormation Template in Eclipse

To validate an AWS CloudFormation template in Eclipse

- Perform either one of the following actions:
  - Right-click the template name in the **Package Explorer** view and click **Validate** on the context menu.



  - Right-click the template that you are editing in the editor pane and click **Validate** on the context menu.

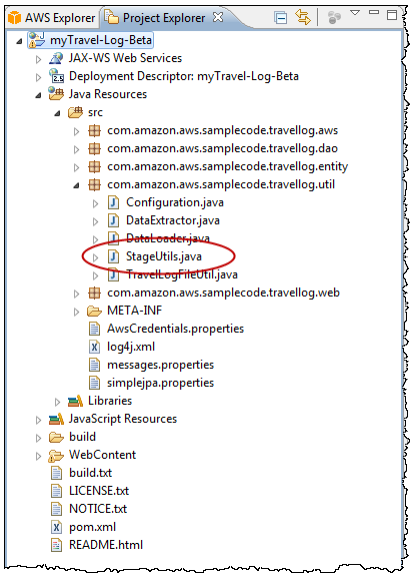
Note
Your template will be validated for *JSON correctness* only; it will not be validated for *AWS CloudFormation correctness*. A stack template validated in this way can still fail to launch or update.

Differentiating AWS Resources with Naming

During development of new products or features, it is useful to keep AWS resources that are used for development separate from resources that are used for production. One approach to maintaining this separation was discussed in the Working with AWS Access Credentials (p. 5), that is, to use different accounts for development and production resources. That approach works especially well when using AWS Explorer, because AWS Explorer displays resources based on account credentials. This section will discuss an alternative approach in which a naming convention is used to differentiate between development and production resources—and in which support for the naming convention is implemented in code.

The basic idea is to distinguish your AWS resources, such as Amazon Simple Storage Service (Amazon S3) buckets or Amazon SimpleDB domains, by adding a designated string value to the resource name. For example, instead of naming your Amazon SimpleDB domain "customers", you would name it "customers-dev" for development use or "customer-prod" for production use. However, an issue arises if you need to move development code into production. At that point, you would need to change all these strings, perhaps with a number of global search and replace operations; that could be tedious or error prone. A more efficient method would be to add support for the naming convention in the code. The Travel Log sample application demonstrates this support.

Travel Log includes a Java file called `StageUtils.java` that implements the StageUtils class.



The `StageUtils` class exposes the following method.

```
public static String getResourceSuffixForCurrentStage()
```

The `getResourceSuffixForCurrentStage` method returns a string that corresponds to the "stage" in the software life cycle for which the resource is used, such as "dev" or "beta" or "prod". This string can then be appended to resource identifiers used in code. The Travel Log application uses `getResourceSuffixForCurrentStage` to construct resource names. For example, the following

method, `getTopicName`, from Travel Log returns a unique name for an Amazon SNS topic. Notice how it embeds the return value from `getResourceSuffixForCurrentStage` in this name.

```
private String getTopicName (Entry entry) {
    return "entry" + StageUtils.getResourceSuffixForCurrentStage() + "-" +
entry.getId();
}
```

The value returned by `getResourceSuffixForCurrentStage` is retrieved from the Java system property, "application.stage". You can specify this value by setting the system property in the container configuration for AWS Elastic Beanstalk.

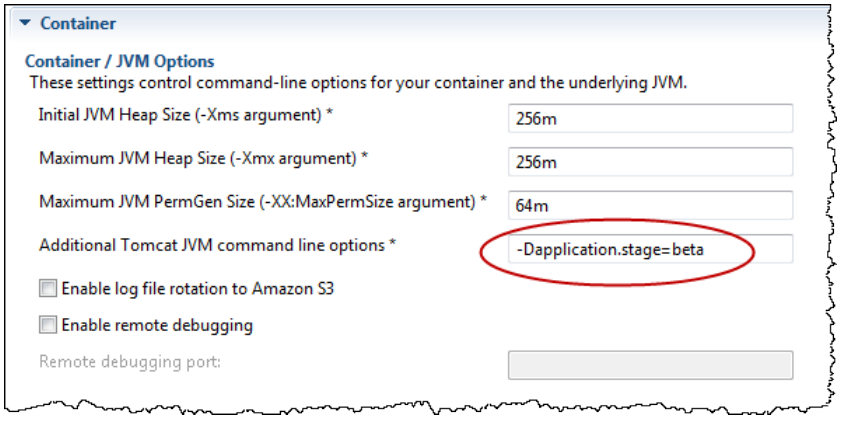
Note
In the AWS Toolkit for Eclipse, your AWS Elastic Beanstalk application needs to be up and running in order for you to access the container configuration. Changing and saving the configuration causes the application to automatically restart with the new configuration.

To access the Container/JVM Options panel for your AWS Elastic Beanstalk application

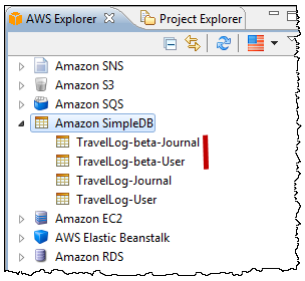
1. In **AWS Explorer**, expand the **AWS Elastic Beanstalk** node and your application node.
2. Beneath the application node, double-click your AWS Elastic Beanstalk environment.
3. At the bottom of the **Overview** pane, click the **Configuration** tab.
4. In the **Container** area, configure the container options.
5. In the **Additional Tomcat JVM command line options** box, specify the value for the application.stage system property by adding a **-D** command line option. For example, you could use the following syntax to specify that the string value should be "-beta".

   ```
   -Dapplication.stage=beta
   ```

   Note that `getResourceSuffixForCurrentStage` automatically prepends a hyphen character to whatever string value you specify.



6. After you have added the system property value, click the **File** menu, and then click **Save**. Eclipse will save the new configuration. The application should restart automatically. You can check the **Events** tab—at the bottom of the Eclipse editor—for the event that indicates that the new configuration was successfully deployed to the environment.
7. After the application restarts, expand the **Amazon SimpleDB** node in **AWS Explorer**. You should now see a new set of domains that use the string value that you specified.



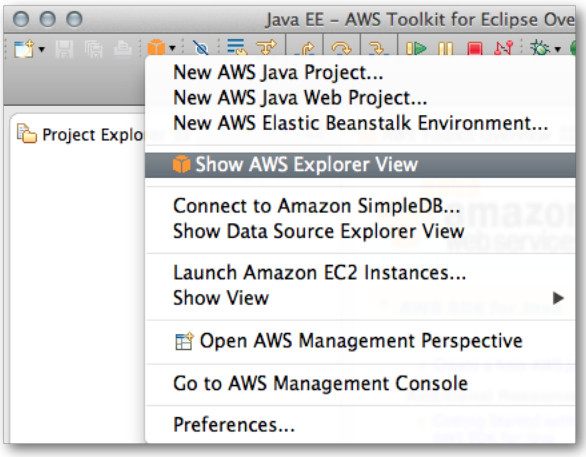
For more information about configuring the container, go to the *AWS Elastic Beanstalk Developer Guide*.

Using AWS Explorer

The AWS Toolkit for Eclipse augments the Eclipse user interface with AWS Explorer, which provides a view across multiple Amazon Web Services simultaneously. AWS Explorer supports the following services.

- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Storage Service (Amazon S3)
- Amazon Simple Queue Service (Amazon SQS)
- Amazon SimpleDB
- Amazon Elastic Compute Cloud (Amazon EC2)
- AWS Elastic Beanstalk
- Amazon Relational Database Service (Amazon RDS)

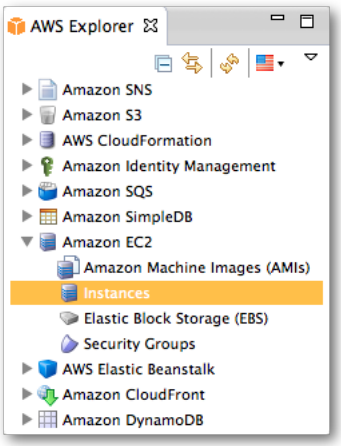
To display **AWS Explorer**, click the AWS icon on the toolbar, and select **Show AWS Explorer View**.



Note
If the AWS icon is not visible on the toolbar, click the **Window** menu, and then click **Open Perspective | Other**. Click **AWS Management** from the list of Eclipse perspectives.

You can expand each node in AWS Explorer to view resources on AWS that are associated with your account. For example, if you click the white triangle to the left of the **Amazon EC2** node, it will expand

and display Amazon EC2 resources associated with your AWS account. The AWS Toolkit for Eclipse uses the AWS account that you configured in the Working with AWS Access Credentials (p. 5) to determine which resources to display.



If you double-click any of the subnodes to Amazon EC2, Eclipse will open a view with detailed information about those resources. For example, double-clicking **Instances** opens a view that lists information about each of your Amazon EC2 instances such as its public DNS name, availability zone, and launch time.

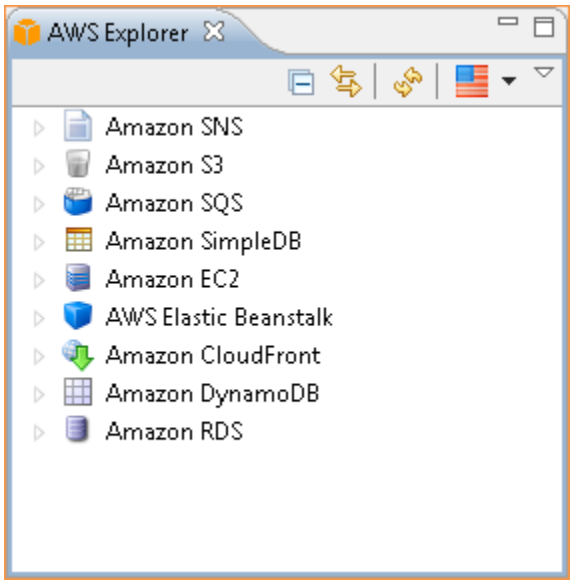
In the following sections, we'll use AWS Explorer to interact with the various services.

Topics

Using Amazon DynamoDB from AWS Explorer

Amazon DynamoDB is a fast, highly scalable, highly available, cost-effective, non-relational database service. Amazon DynamoDB removes traditional scalability limitations on data storage while maintaining low latency and predictable performance. The AWS Toolkit for Eclipse provides functionality for working with Amazon DynamoDB in a development context. For more information about Amazon DynamoDB, see the detail page on the AWS website.

In the AWS Toolkit for Eclipse, AWS Explorer displays all the Amazon DynamoDB tables associated with the active AWS account.

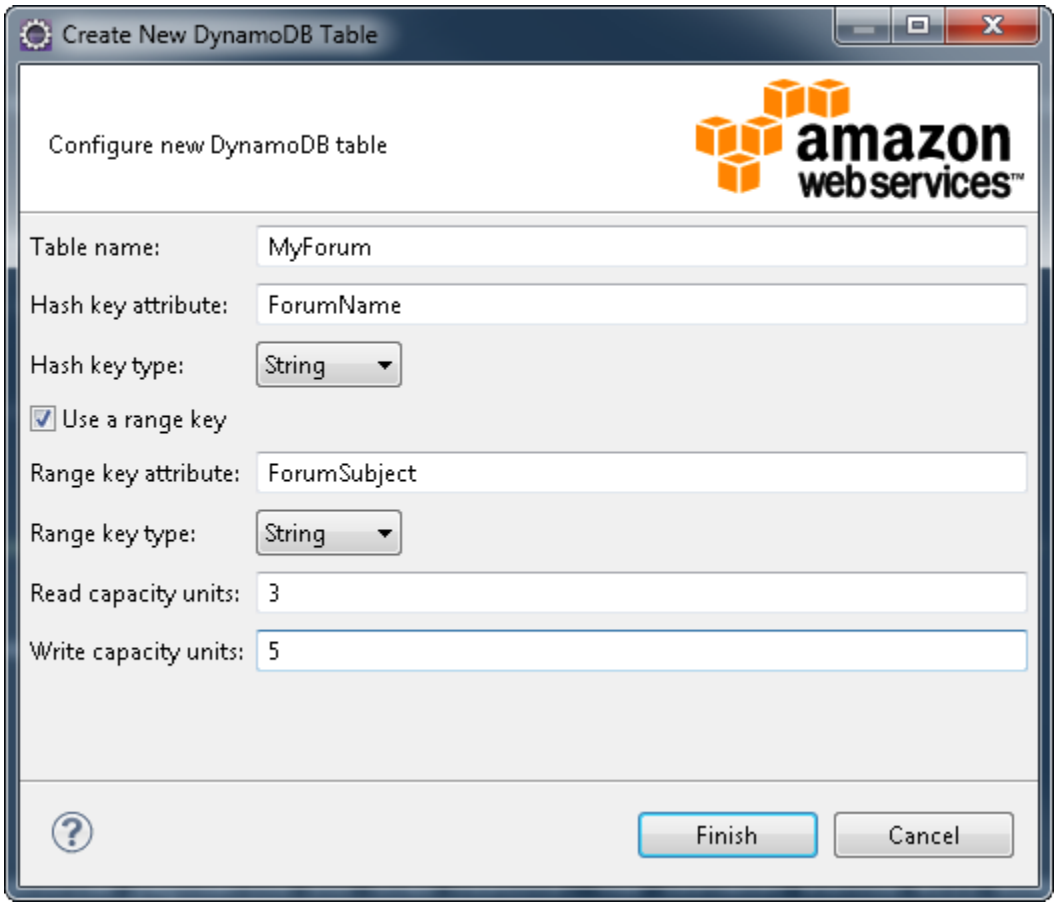


Creating an Amazon DynamoDB Table

Using the Toolkit for Eclipse, you can create a new Amazon DynamoDB table.

To create a new table in AWS Explorer

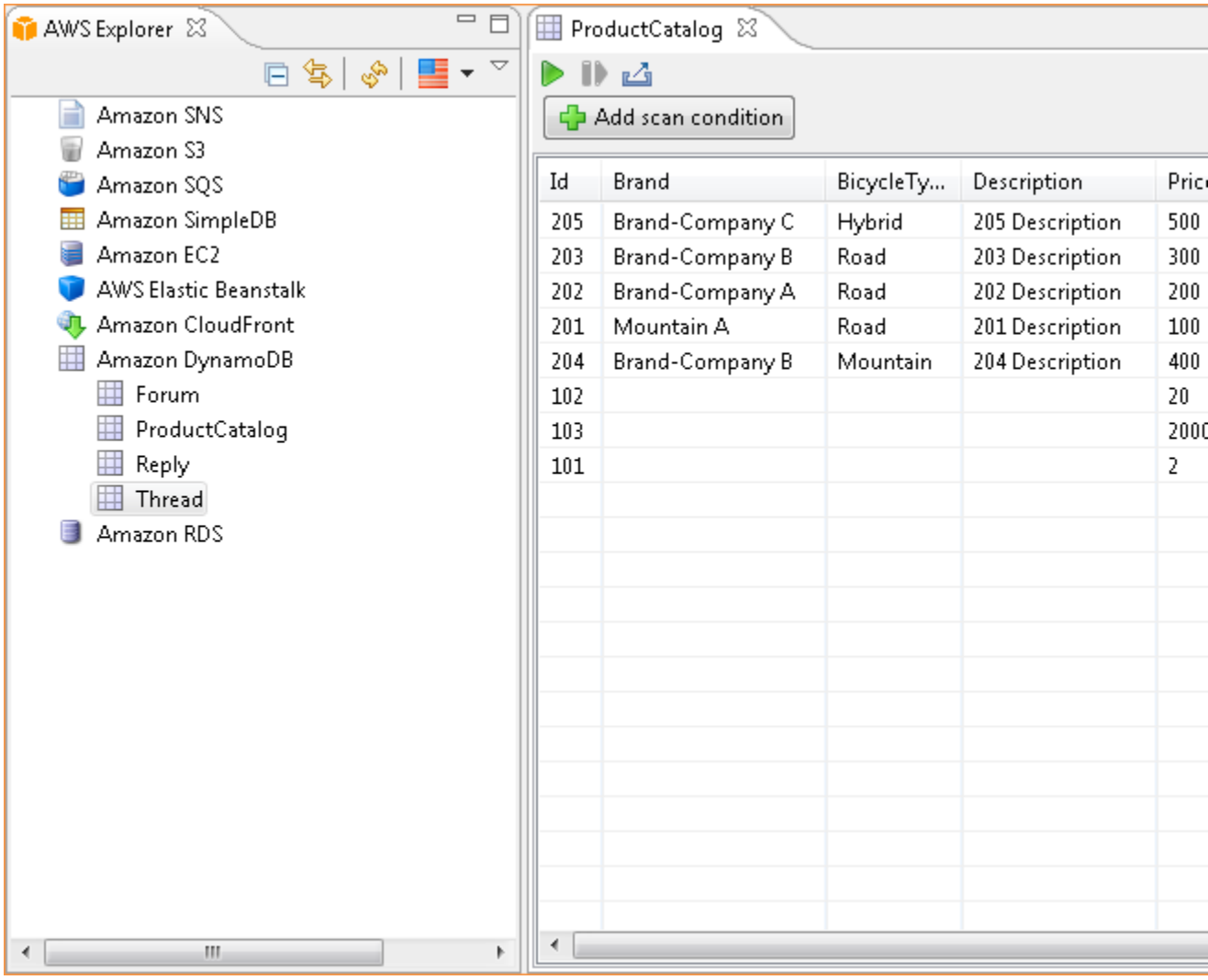
1. In **AWS Explorer**, right-click **Amazon DynamoDB**, and then click **Create Table**. The **Create New DynamoDB Table** wizard appears.
2. Enter a table name in the **Table name** box.
3. Enter a primary hash key attribute in the **Hash key attribute** box, and select the hash key type from the **Hash key type** drop-down list. Amazon DynamoDB builds an unordered hash index using the primary key attribute and an optional sorted range index using the range primary key attribute. For more information about the primary hash key attribute, go to the Primary Key section in the *Amazon DynamoDB Developer Guide*.
4. Optionally, specify a range primary key by selecting **Use a range key**. Enter a range key attribute in the **Range key attribute** box, and select a range key type from the **Range key type** drop-down list.
5. Specify the number of read capacity units in the **Read capacity units** box, and specify the number of write capacity units in the **Write capacity units** box. You must specify a minimum of 3 read capacity units and 5 write capacity units. For more information about read and write capacity units, go to the Provisioned Throughput in Amazon DynamoDB section in the *Amazon DynamoDB Developer Guide*.
6. Click **Finish** to create the table. Click the refresh button in **AWS Explorer** to view your new table in the table list.



Viewing an Amazon DynamoDB Table as a Grid

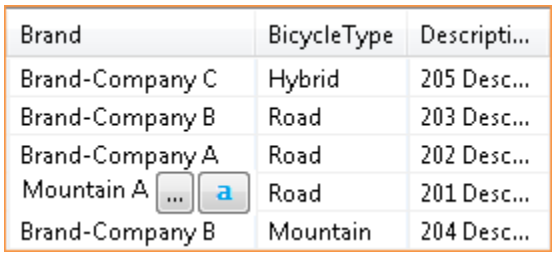
To open a grid view of one of your Amazon DynamoDB tables, double-click the subnode in **AWS Explorer** that corresponds to the table. From the grid view, you can view the items, attributes, and values stored in the table. Each row corresponds to an item in the table. The table columns correspond to attributes. Each cell of the table holds the values associated with that attribute for that item.

An attribute can have a value that is a string or a number. Some attributes have a value that consists of a *set* of strings or numbers. Set values are displayed as a comma-separated list enclosed by square brackets.



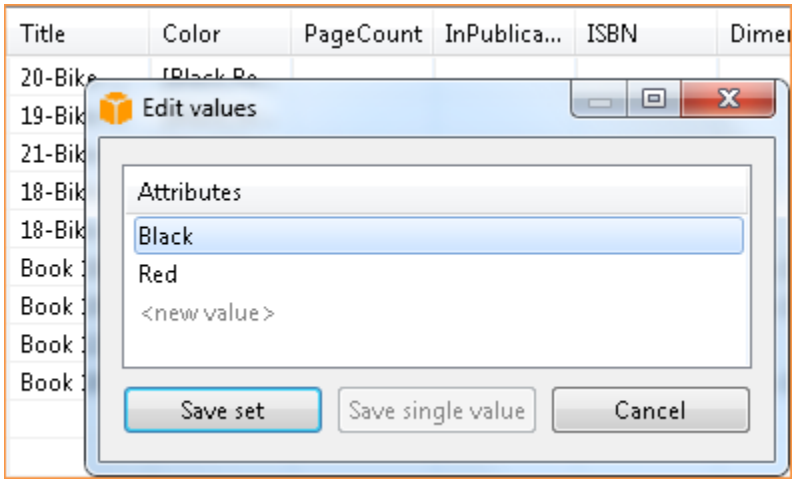
Editing Attributes and Values

The table grid view is *editable*; by double-clicking a cell, you can edit the values for the item's corresponding attribute. For set-value attributes, you can also add or delete individual values from the set.



The editing UI enables you not only to change the value of an attribute, but also to change the format of the value for the attribute—with some limitations. For example, any number value can be converted into a string value. If you have a string value, the content of which is a number, such as "125", the editing UI enables you to convert the format of the value from string to number. Also, the editing UI enables you to

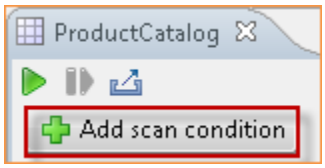
convert a single-value to a set-value. However, you cannot generally convert from a set-value to a single-value; an exception is when the set-value has, in fact, only one element in the set.



The **Edit Values** dialog box opens when you are editing a set of values. After editing the attribute value, click **Save set** to confirm your changes. If you want to discard your changes, click **Cancel**.

After confirming your changes, the attribute value is displayed in red. This indicates that the attribute has been updated, but that the new value has not been written back to the Amazon DynamoDB database. To write your changes back to Amazon DynamoDB, click **File**, and then click **Save**, or press **Ctrl+S** from the keyboard. To discard your changes, click **Scan Table**, and when the Toolkit asks if you would like to commit your changes before the Scan, click **No**.

Scanning an Amazon DynamoDB Table

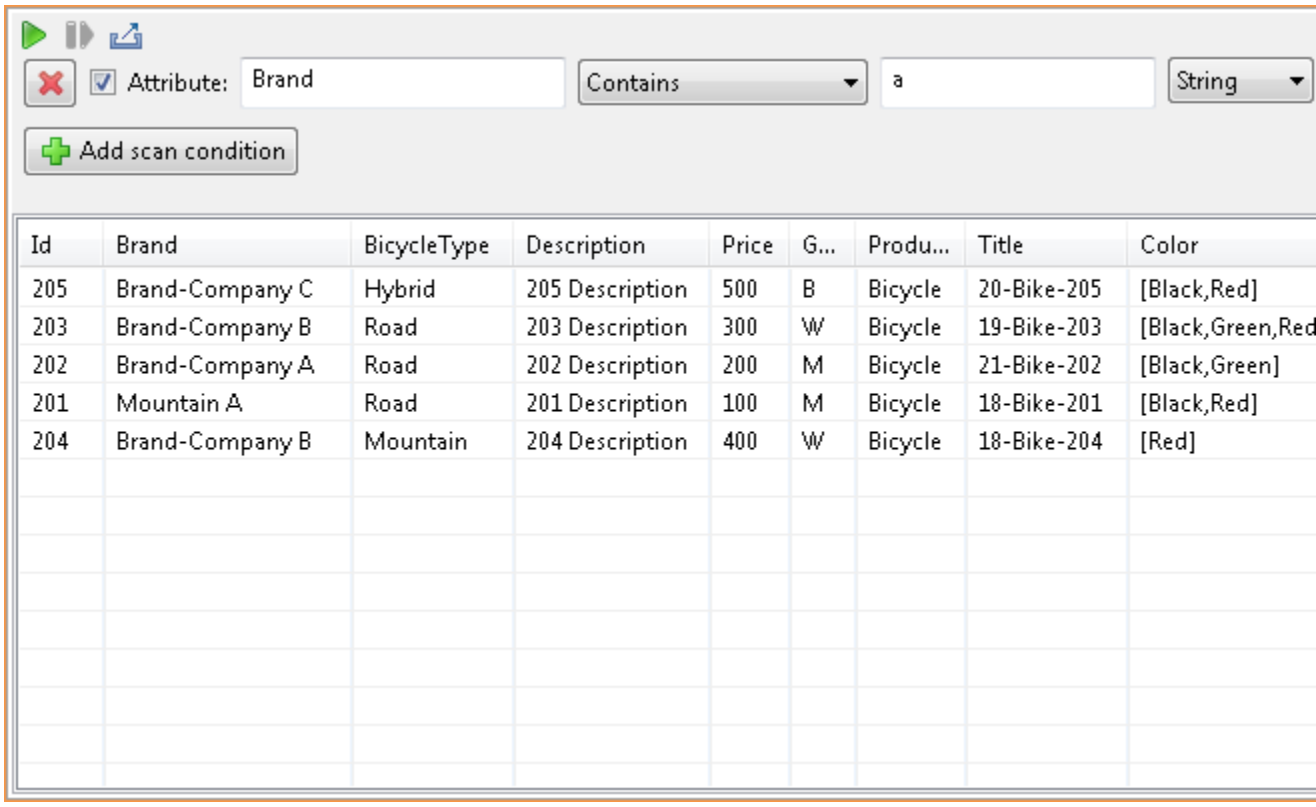


From the Toolkit, you can perform Scans on your Amazon DynamoDB tables. In a Scan, you define a set of criteria and the Scan returns all items from the table that match your criteria. Scans are expensive operations and should be used with care to avoid disrupting higher-priority production traffic on the table. For more recommendations on safely using the Scan operation, go to the *Amazon DynamoDB Developer Guide*.

To perform a Scan on an Amazon DynamoDB table from AWS Explorer

1. In the grid view, click **Add scan condition**. A UI appears that enables you to edit a new Scan clause.
2. In the Scan clause editor, specify the attribute to match against, how it should be matched (Begins With, Contains, etc.), what literal value it should match, and if the value is a string or a number.
3. Add more Scan clauses as needed for your search. The Scan will return only those items that match the criteria from all of your Scan clauses. Note that the Scan will perform a case-sensitive comparison when matching against string values.
4. On the button bar at the top of the grid view, click the green play button to run the scan.

To remove a Scan clause, click the red X to the left of each clause.



To return to the view of the table that includes all items, double-click **Amazon DynamoDB** in **AWS Explorer**.

Paginating Scan Results

At the top of the view are three buttons.

The second button provides pagination for Scan results.

Export Scan Result to CSV

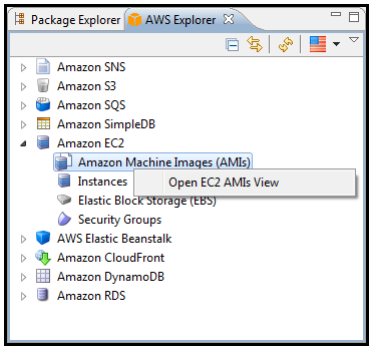
Clicking the rightmost button exports the results from the current Scan to a CSV file.

Launch an Amazon EC2 Instance from an Amazon Machine Image (AMI)

This section describes how to launch an Amazon EC2 instance from an Amazon Machine Image (AMI) in the Toolkit for Eclipse. Before launching an EC2 instance, you should create a security group that will permit network traffic that is appropriate to your application to connect to the instance. At a minimum, the security group should enable access on port 22, so that you can SSH into the EC2 instance. You may also want to create a keypair, although you can also create the keypair while going through the launch wizard. Finally, you should think about which instance type is appropriate to your application; the price for an EC2 instance is typically higher for the more powerful types of instances. You can find a list of instance types and pricing information on the AWS website.

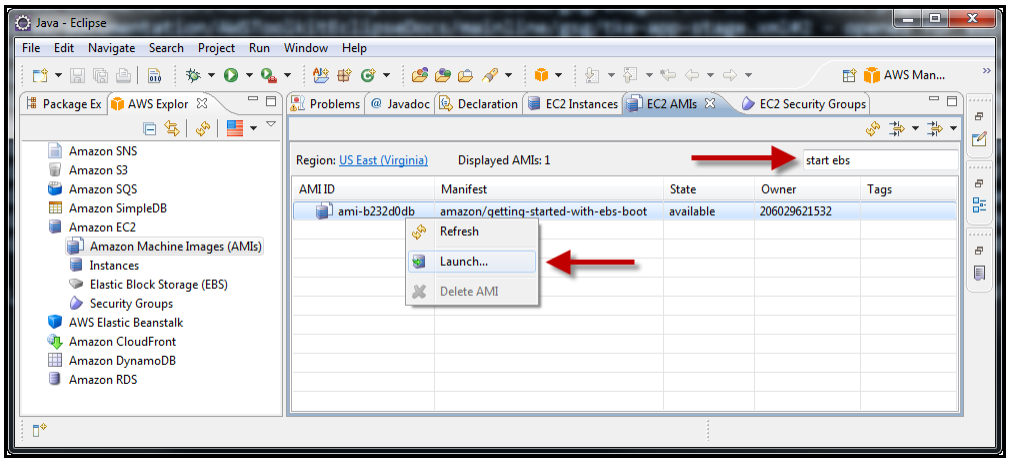
To launch an Amazon EC2 instance

1. In **AWS Explorer**, expand the **Amazon EC2** node. Right-click the **Amazon Machine Images (AMIs)** subnode and select **Open EC2 AMIs View**.



2. Configure the AMIs view to show the AMI that we'll use in this example. In the filter box, type **`start ebs`**. This filters the list of AMIs to show only those AMIs with names that contains both "start" and "ebs".

   Right-click the **amazon/getting-started-with-ebs** AMI and select **Launch** from the context menu.



3. In the **Launch EC2 Instance** dialog box, configure the AMI for your application.

   Number of Hosts
   Set this value to the number of EC2 instances to launch.

   Instance Type
   Select the type of the EC2 instance to launch. You can find a list of instance types and pricing information on the AWS website.

   Availability Zone
   Select an availability zone (AZ) in which to launch the instance. Note that not all AZs are available in all regions. If the AZ that you select is not available, the Toolkit will generate a message saying that you need to select a different AZ. For more information about AZs, go to the Region and Availability Zone FAQ in the *Amazon Elastic Compute Cloud User Guide*.

   Key Pair
   A key pair is a set of public/private encryption keys that are used to authenticate you when you connect to the EC2 instance using SSH. Select a keypair for which you have access to the private key. You can also create a new key pair by clicking 

Security Group

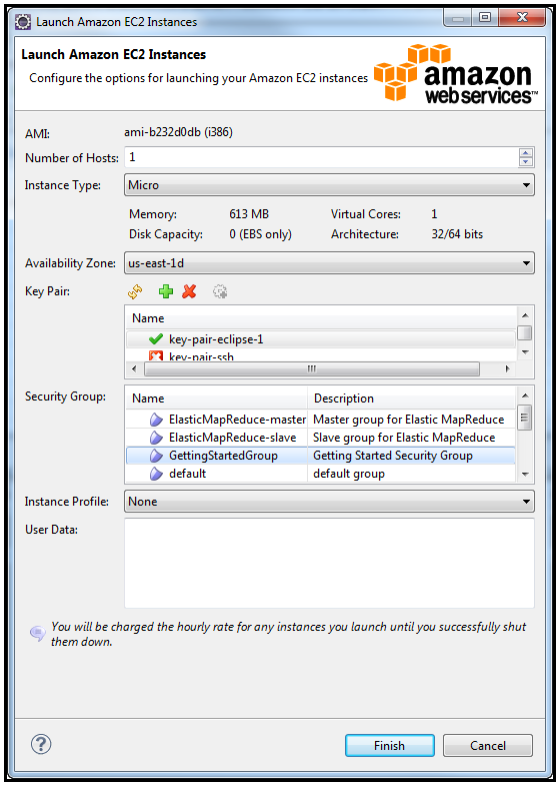
The security group controls what type of network traffic the EC2 instance will accept. You should select a security group that will allow incoming traffic on port 22, i.e. the port that is used by SSH, so that you can connect to the EC2 instance. For information about how to create security groups using the Toolkit, see Managing Security Groups from AWS Explorer (p. 50)

Instance Profile

The instance profile is a logical container for an IAM role. When you select an instance profile, you associate the corresponding IAM role with the EC2 instance. IAM roles are configured with policies that specify access to particular AWS services and account resources. When an EC2 instance is associated with an IAM role, application software that runs on the instance runs with the permissions specified by the IAM role. This enables the application software to run without having to specify any AWS credentials of its own, which makes the software more secure. For in-depth information about IAM roles, go to Working with Roles in *Using IAM*.

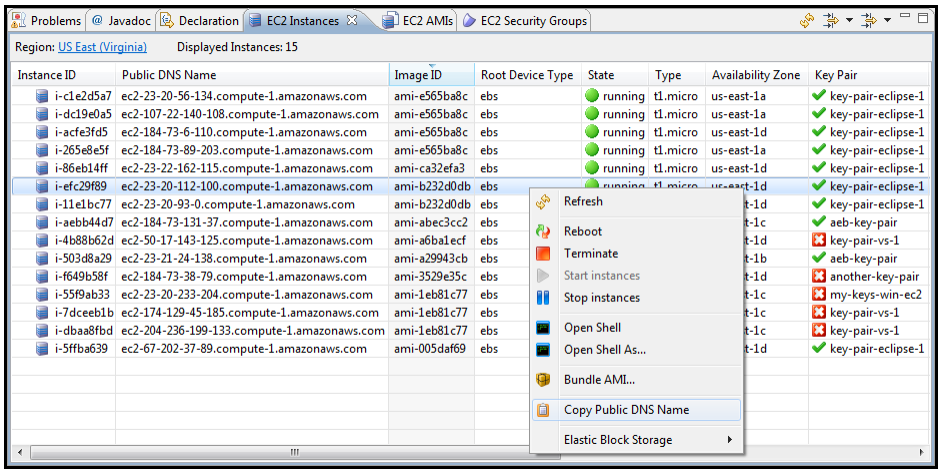
User Data

The user data is data that you provide to the application software that runs on your EC2 instance. The application software can access this data through the Instance Meta Data Service (IMDS)



4. Click **Finish**.
5. In AWS Explorer, under the **Amazon EC2** node, right-click the **Instances** subnode and select **Open EC2 Instances View**.

   Your EC2 instance should appear in the **EC2 Instances** view. It may take a few minutes for the instance to transition into the **running** state. Once the instance is running, you can right-click the instance to bring up a context menu of operations that you can perform on the instance. For example, you can terminate the instance from this menu. You can also copy the instance's public DNS address. You would use this address to connect to the instance using SSH.



Managing Security Groups from AWS Explorer

The AWS Toolkit for Eclipse enables you to create and configure security groups to use with Amazon Elastic Compute Cloud (Amazon EC2) instances. When you launch an Amazon EC2 instance, you need to specify an associated security group.

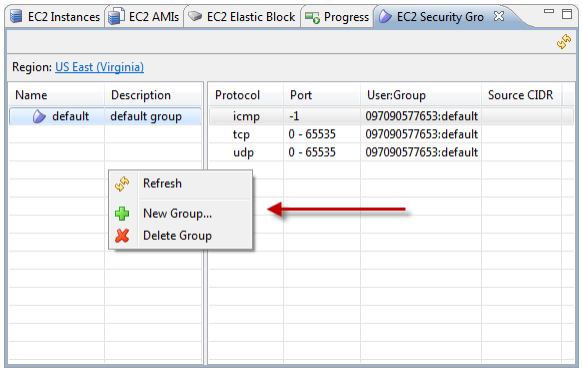
A security group acts like a firewall on incoming network traffic. The security group specifies what types of network traffic an Amazon EC2 instance will allow to be received. It can also specify that incoming traffic will be accepted only from certain IP addresses or only from other specified security groups.

Creating a New Security Group

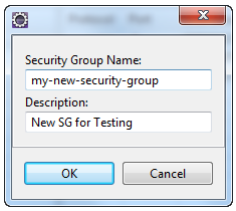
In this section, we'll create a new security group. Initially after creation, the security group will not have any permissions configured. Configuring permissions is handled through an additional operation.

To create a new security group

1. In **AWS Explorer**, beneath the **Amazon EC2** node, right-click **Security Groups**, and then click **Open EC2 Security Groups View**.
2. Right-click in the left pane of the **EC2 Security Groups** tab, and then click **New Group**.



3. In the dialog box, enter a name and description for the new security group. Click **OK**.

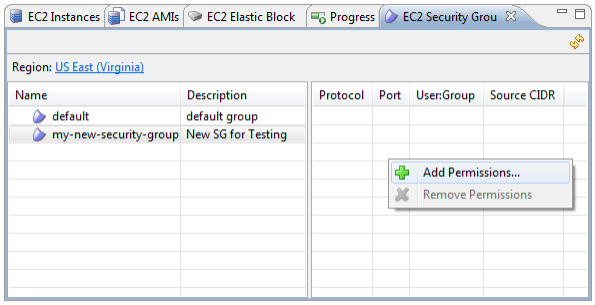


Adding Permissions to Security Groups

In this section, we'll add permissions to the new security group to allow other computers to connect to our Amazon EC2 instance using Secure Shell (SSH) protocol.

To add permissions to a security group

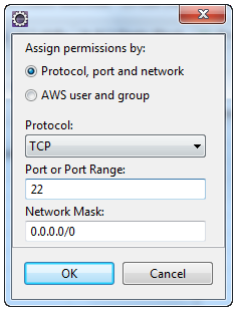
1. Right-click in the right pane of the **EC2 Security Groups** tab, and then click **Add Permissions**.



2. In the dialog box, select **Protocol, port and network**. Click **TCP** from the **Protocol** drop-down menu. Enter **22** for **Port or Port Range**. Port 22 is the standard port for SSH. The **Network Mask** box specifies the allowed source IP addresses in CIDR format; it defaults to 0.0.0.0/0, which specifies that the security group will allow a TCP connection to port 22 (SSH) from any external IP address.

   You could also, for example, specify that connections should be allowed only from computers in your local computer's subnet. In this case, you would specify your local computer's IP address followed by a "/10". For example, "xxx.xxx.xxx.xxx/10" where the "xxx" correspond to the distinct octet values that make up your local computer's IP address.

   Click **OK**.



You could also set permissions to the security group by specifying a UserID and security group name. In this case, Amazon EC2 instances in this security group would accept all incoming network traffic from

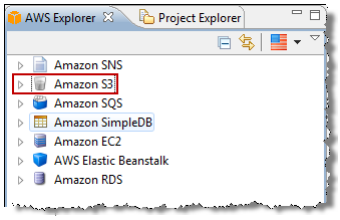
Amazon EC2 instances in the specified security group. It is necessary to also specify the UserID as a way to disambiguate the security group name; security group names are not required to be unique across all of AWS. For more information about security groups, go to the EC2 documentation.

Viewing and Editing Amazon S3 Buckets

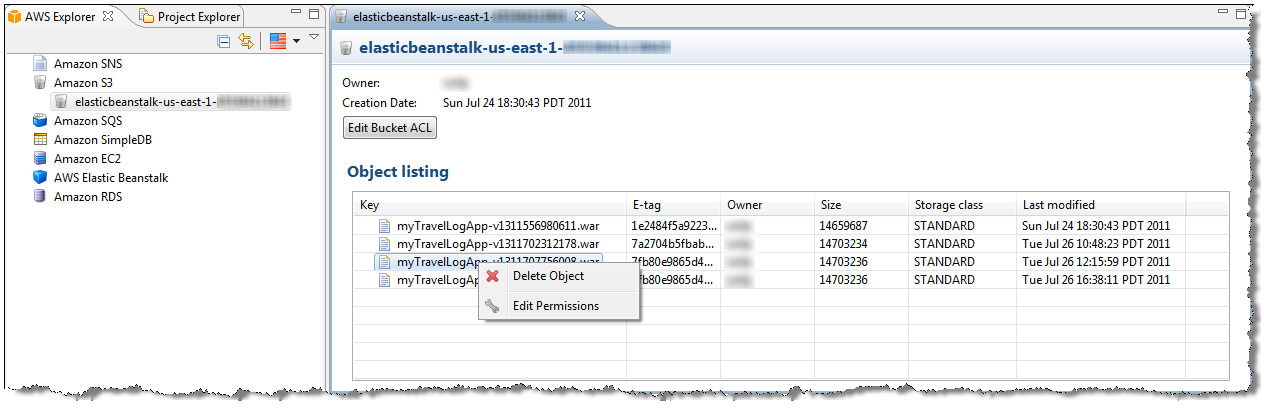
You can use the AWS Toolkit for Eclipse to view and edit permissions for buckets and objects in the Amazon Simple Storage Service (Amazon S3). Think of buckets as being roughly equivalent to folders in a computer file system, and objects within buckets as being similar to files. In this section, we'll use the AWS Toolkit for Eclipse to view the buckets associated with the Travel Log application that you deployed in Deploying an Application Using AWS Elastic Beanstalk (p. 13).

For AWS Elastic Beanstalk to deploy an application, the application must be in the form of a .war (web archive) file. During deployment, AWS Elastic Beanstalk stores a copy of the .war file in an Amazon S3 bucket. When you deployed the Travel Log application, the .war file for Travel Log was stored in Amazon S3. The .war file will persist in Amazon S3 even after you shut down the application. We'll view that Amazon S3 bucket using the Toolkit for Eclipse.

1. In the AWS Toolkit for Eclipse, click the **AWS Explorer** tab. If **AWS Explorer** is not visible, click the AWS icon from the toolbar, and then click **Open AWS Management Perspective**.



2. Click the arrow to the left of the **Amazon S3** node to display any associated Amazon S3 buckets. Double-click one of the Amazon S3 buckets to display a detailed view of the bucket in the Eclipse editor pane.



3. From the Eclipse editor, click **Edit Bucket ACL** to set permissions on the Amazon S3 bucket as a whole. You can also right-click any of the .war file objects to view a context menu from which you can set the permissions specifically on that object or delete the object.

   Note
   You might see more than one .war file stored in Amazon S3. If you deploy Travel Log multiple times, AWS Elastic Beanstalk will create a versioned instance of the .war file for each

deployment. The sequence of numbers used as the suffix for each .war file is the version label that you specify during the deployment process.

Viewing and Editing a Amazon SimpleDB Domain

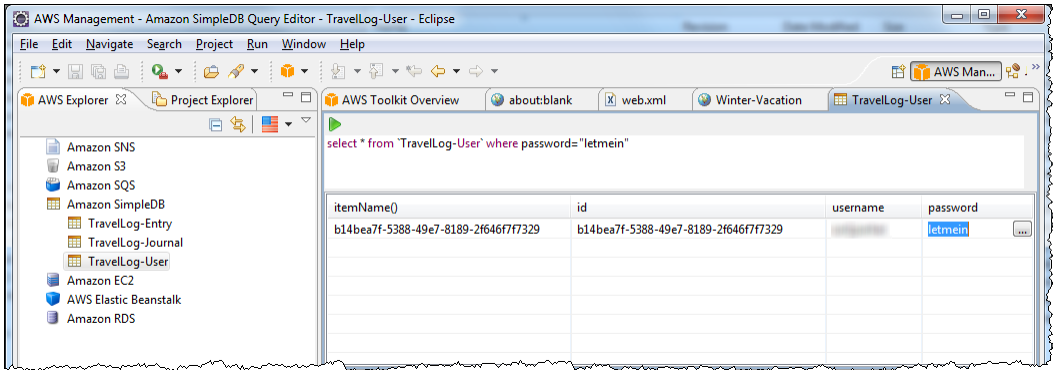
From AWS Explorer, you can view any of the Amazon SimpleDB domains that are associated with your AWS account. In addition to viewing these domains, you can also query and edit them.

The Travel Log application uses Amazon SimpleDB domains to track the data for its travel journals. For example, both users and journal entries are stored in domains. In this section, we'll use AWS Explorer to interact with these domains.

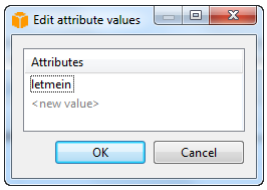
View a Amazon SimpleDB Domain

To view the Amazon SimpleDB domain that tracks the Travel Log application's users

1. In **AWS Explorer**, click the triangle to the left of the **Amazon SimpleDB** node. This expands the node and displays all of the domains used by the Travel Log application. Double-click the **TravelLog-User**domain. The AWS Toolkit for Eclipse displays a view of the domain in the Eclipse editor.
2. The domain view displays records that are retrieved by the query in the upper half of the Eclipse editor. Click the triangle above the query in order to run the query and retrieve records from the domain.
3. The displayed query is completely editable. You might try experimenting with different queries to see what they retrieve. For more information about the Amazon SimpleDB query language, go to the Amazon SimpleDB Developer Guide.



4. If you click any field in the list of records returned by the query, an ellipsis (**...**) button appears. Click the ellipsis button to edit or change the value of the field. For example, you could change the password for the account that you created earlier, and then go back to the instance of Travel Log in your browser and sign-in with the new password.



Viewing and Adding Amazon SNS Notifications

You can use the AWS Toolkit for Eclipse to view Amazon Simple Notification Service (Amazon SNS) topics associated with your application. Amazon SNS is a service that enables your application to send notifications, using a protocol such as email, when specified events occur. To learn more about Amazon SNS, go to the Amazon SNS documentation.

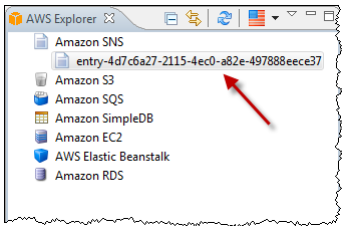
In the Travel Log sample application, if a user comments on a journal entry, they have the option of requesting notification of further comments on that entry. These notifications are implemented using Amazon SNS.

View an Amazon SNS Notification

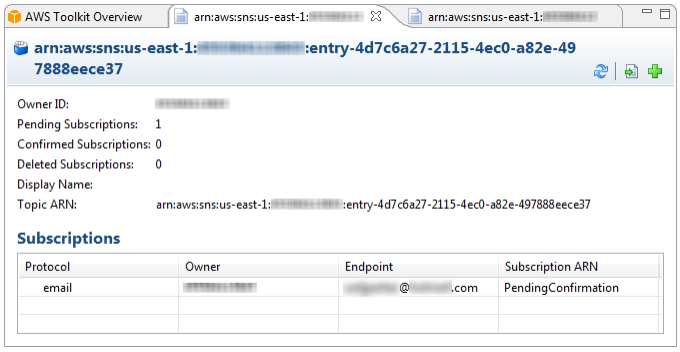
The following process illustrates how to view an Amazon SNS notification using notifications associated with the Travel Log application as an example.

To view a travel log notification

1. In **AWS Explorer**, click the triangle to the left of the **Amazon SNS** node to expand it and see the Amazon SNS topics it contains. If you have added a comment to a Travel Log journal entry and requested email notifications, you should see at least one topic beneath the **Amazon SNS** node.



2. Double-click this SNS topic to open a detail view in the Eclipse editor pane. In this example, the **Subscription ARN** column says that the topic is pending confirmation. Amazon SNS requires a confirmation from the individual specified by the email address before SNS will send email notifications to that individual.

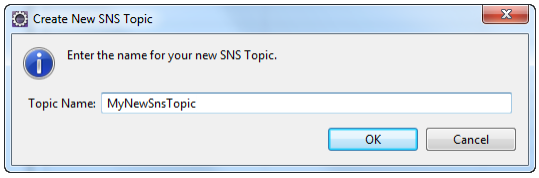


Add an Amazon SNS Notification

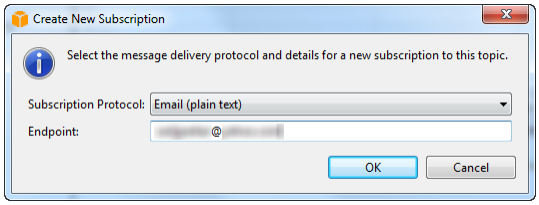
You can add new Amazon SNS notifications through AWS Explorer.

To add a new notification to the Travel Log application

1. In **AWS Explorer**, right-click **Amazon SNS**, and then click **Create New Topic**. Enter a name for the new topic and click **OK** .



2. Double-click the new topic to display the detail view for the topic. Right-click in the **Subscriptions** area, and then click **Create Subscription**. Leave the **Subscription Protocol** box as **Email (plain text)** and enter an email address for the endpoint. Click **OK** . The detail view for the notification will now include this subscription.



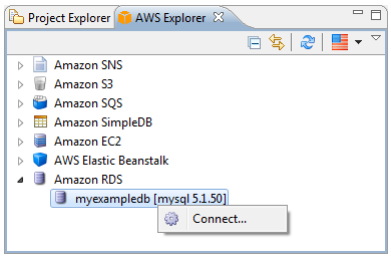
3. To delete the subscription, right-click the entry in the **Protocol** column for the subscription and click **Delete Subscription** .

Note
The creation of the subscription will cause a verification email to be sent to the individual specified by the subscription "endpoint" email address. This email address will be used by AWS only to send notifications. It will not be used for any other purpose by AWS or Amazon.com.

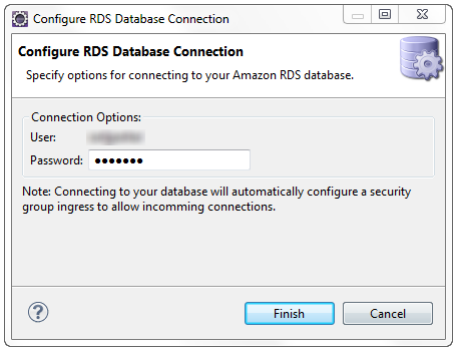
Connecting to Amazon Relational Database Service (Amazon RDS)

In this section, we'll use the AWS Toolkit for Eclipse to connect to a database instance on the Amazon Relational Database Service (Amazon RDS). Before stepping through the process described below, you will need to have an RDS database instance associated with your AWS account. You can create a database instance on RDS using the AWS Management Console. When you create a database instance, set the TCP port that the database uses to receive connections to a value that is accessible from your location. For example, if you are behind a firewall, choose a TCP port to which your firewall allows connections. For more information, go to the RDS documentation. In particular, the Amazon Relational Database Service Getting Started Guide steps you through the creation of a database instance.

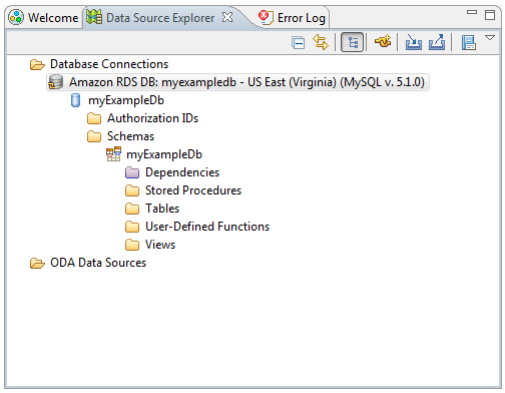
1. In **AWS Explorer**, expand the **Amazon RDS** node. You should see a list of the database instances that are associated with your AWS account. Right-click one of these instances, and then click **Connect** .



2. The AWS Toolkit for Eclipse displays an authentication dialog box. Enter the master password that you specified when you created the database instance. Click **Finish**.



3. The AWS Toolkit for Eclipse brings up the connection to the database instance in the Eclipse Data Source Explorer. From here, you can inspect the structure and data in the database.



Identity and Access Management

AWS Identity and Access Management (IAM) lets you control who can access your AWS resources and what they can do with them. Instead of sharing the password and security credentials for your account (the access key ID and secret access key), you can create *IAM users* that can each have their own password and security credentials. You can then attach *policies* to users; in the policies you specify permissions that determine what actions a user can take and what resources the user is allowed access to. For convenience, instead of adding policies to individual users, you can create *IAM groups* (for example, Admins and Developers) and attach policies to them, and then add users to those groups. You can also create *roles* that have policies with permissions. Roles can be assumed by users who are in other accounts, by services, and by users who do not have an IAM identity. For more information about IAM, see the Using IAM guide.

The AWS Toolkit lets you create and manage IAM users, groups, and roles. You can also set a password policy for users, which lets you specify password requirements like minimum length, and lets you specify whether users are allowed to change their own passwords.

> **Note**
> A best practice is for all users, even the account owner, to access AWS resources as IAM users. This ensures that if the credentials for one of the IAM users are compromised, just those credentials can be revoked without needing to change the root credentials for the account.

Topics

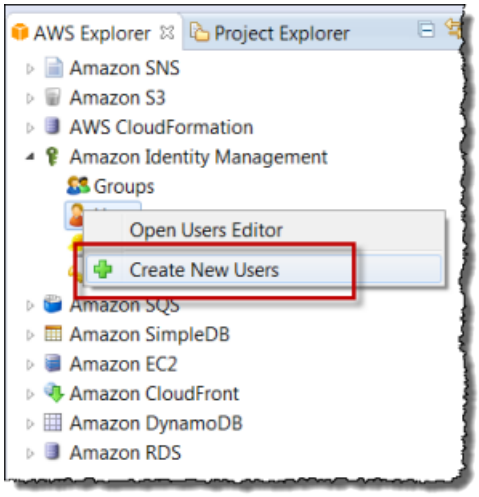
Create an IAM User

You create IAM users so that others in your organization can have their own AWS identity. You can assign permissions to an IAM user either by attaching an IAM policy to the user or by assigning the user to a group. IAM users that are assigned to a group derive their permissions from the policies that are attached to the group. For more information, see Create an IAM Group (p. 58) and Add an IAM User to an IAM Group (p. 60).

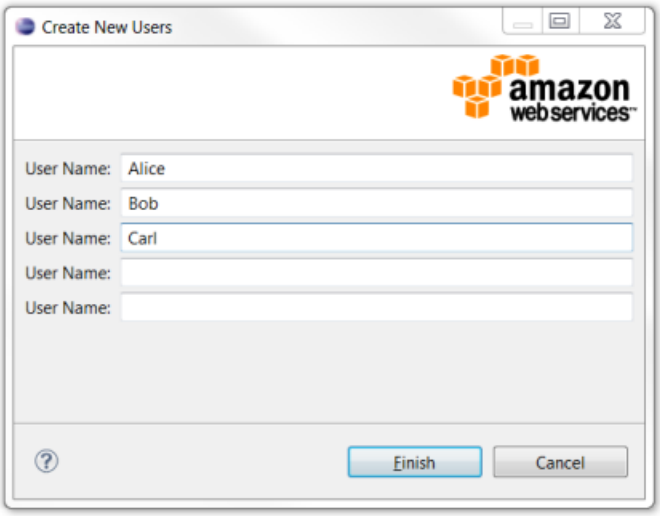
Using the Toolkit, you can also generate AWS credentials (access key ID and secret access key) for the IAM user. For more information, see Manage Credentials for an IAM User (p. 61).

To create an IAM User

1. In **AWS Explorer**, expand the **AWS Identity and Access Management** node, right-click the **Users** node, and then select **Create New Users**.



2. In the **Create New Users** dialog box, enter up to five names for new IAM users, and then click **Finish**. For information about constraints on names for IAM users, see Limitations on IAM Entities in the *Using IAM* guide.



For information about adding a user to a group, see Add an IAM User to an IAM Group (p. 60). For information about how to create a policy and attach it to the user, see Attach an IAM Policy to a User, Group, or Role (p. 67).

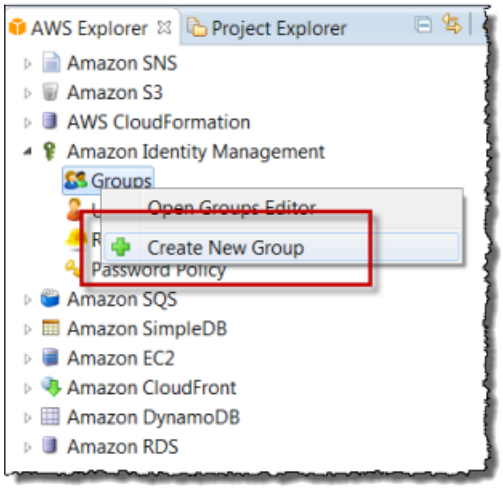
Create an IAM Group

You can add IAM users to groups in order to make it easier to manage permissions. Any permissions that are attached to the group apply to any users in that group. For more information about IAM groups, see Working with Users and Groups in the *Using IAM* guide.

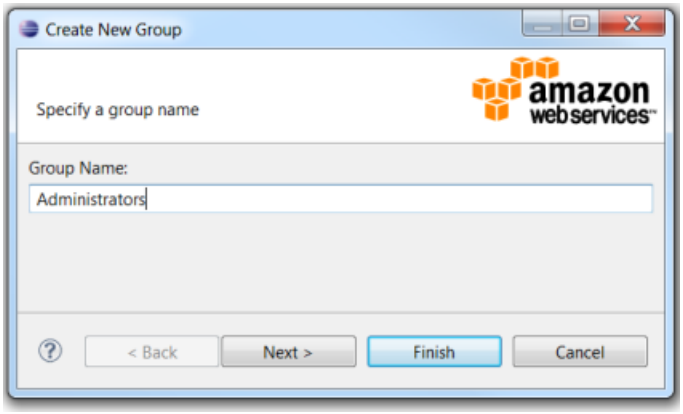
When you create a group, you can create a policy that includes the permissions that members of the group will have.

To create an IAM group

1. In **AWS Explorer**, expand the **AWS Identity and Access Management** node, right-click the **Groups** node, and then select **Create New Group**.



2. Enter a name for the new IAM group and then click **Next**.



3. Enter a name for the policy that establishes what members of the group are allowed to do. Enter the policy as a JSON document, and then click **OK**.

The policy name must be unique within your account. The JSON that you enter for the policy must validate, or you will not be able to save the policy. For information about how to create a policy, see Overview of Policies in the *Using IAM* guide.

4. Click **Finish**.

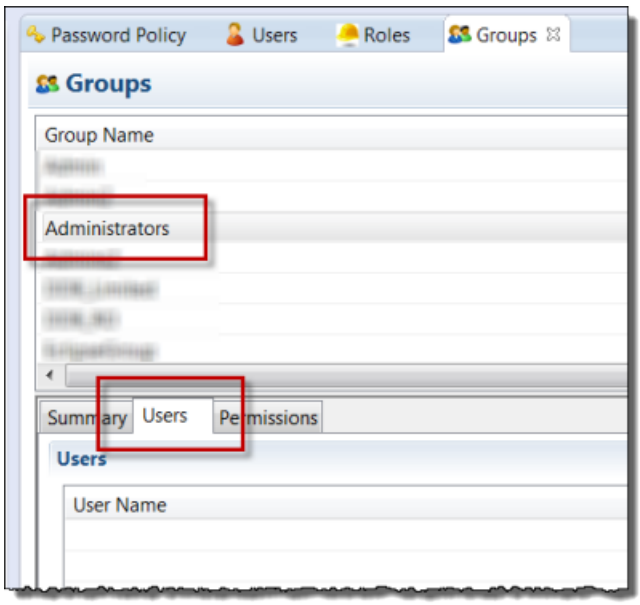
For information about attaching additional policies to the IAM group, see Attach an IAM Policy to a User, Group, or Role (p. 67).

Add an IAM User to an IAM Group

If an IAM user is added to a group, any policies that are attached to the group are also in effect for the user. For more information about IAM users, see Users and Groups in the *Using IAM* guide.

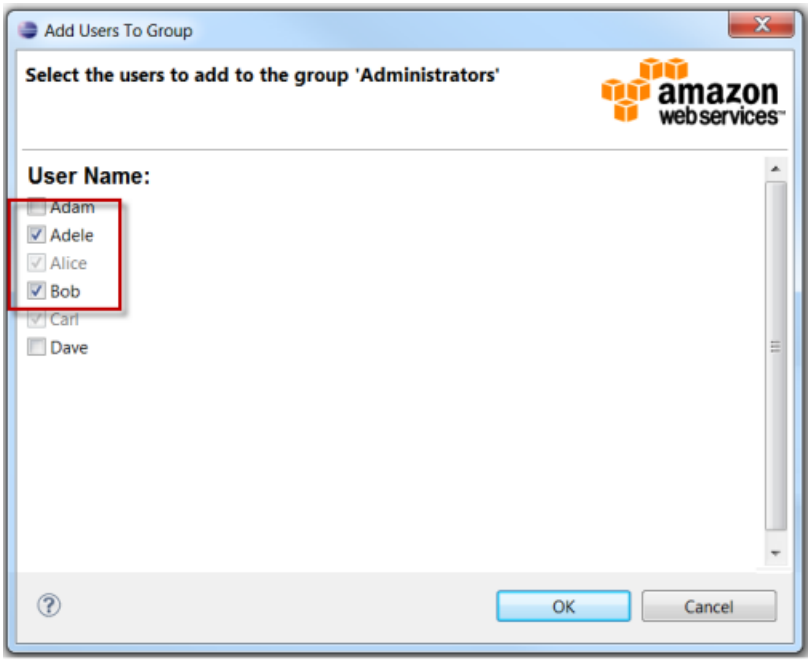
To add an IAM user to a IAM group

1. In **AWS Explorer**, expand the **AWS Identity and Access Management** node, right-click the **Groups** node, and then select **Open Groups Editor**. Note that you add IAM users to IAM groups from the **Groups** node in **AWS Explorer** rather than from the **Users** node.
2. In the **Groups** editor, select the group you want to add users to, and then click the **Users** tab.



3. On the right-hand side of the bottom pane, click the **Add Users** button.

4. In the **Add Users to Group** dialog box, select the users you want to add, and then click **OK**.

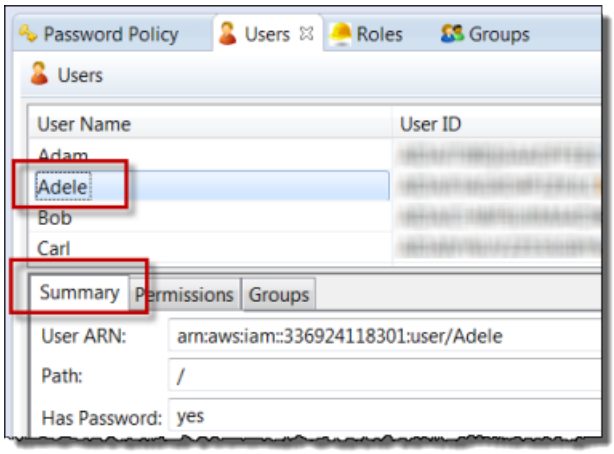


Manage Credentials for an IAM User

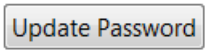
For each user, you can add a password. IAM users use a password to work with AWS resources in the AWS Management Console.

To create a password for an IAM user

1. In **AWS Explorer**, expand the **AWS Identity and Access Management** node, right-click the **Users** node, and then select **Open Users Editor**.
2. In the users listing, select the user you want to create a password for, and then click the **Summary** tab.

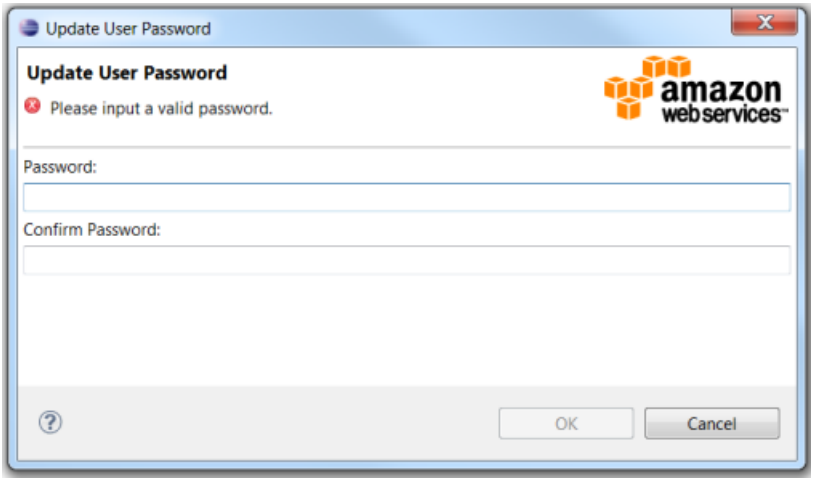


3. On the right-hand side of the bottom pane, click the **Update Password** button.



4. In the **Update User Password** dialog box, enter a password and then click **OK**.

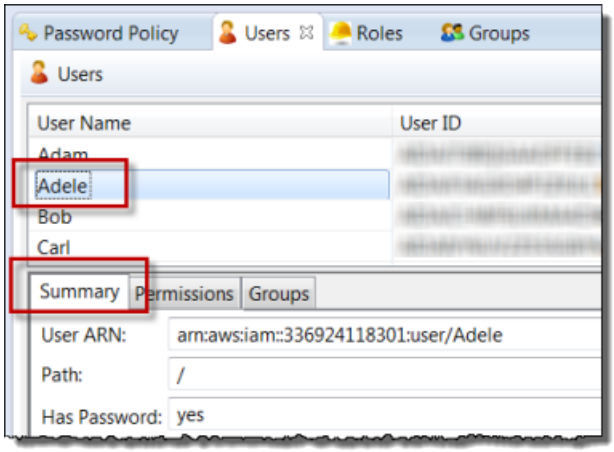
   Note
   The new password will overwrite any existing password.



For each user you can also generate a set of access keys (an access key ID and a secret access key). These keys can be used to represent the user for programmatic access to AWS—for example, to use the AWS command-line interface (CLI), to sign programmatic requests using the SDK, or to access AWS services through the Toolkit. (For information about how to specify credentials for use with the Toolkit, see Working with AWS Access Credentials (p. 5).)

To generate access keys for an IAM user

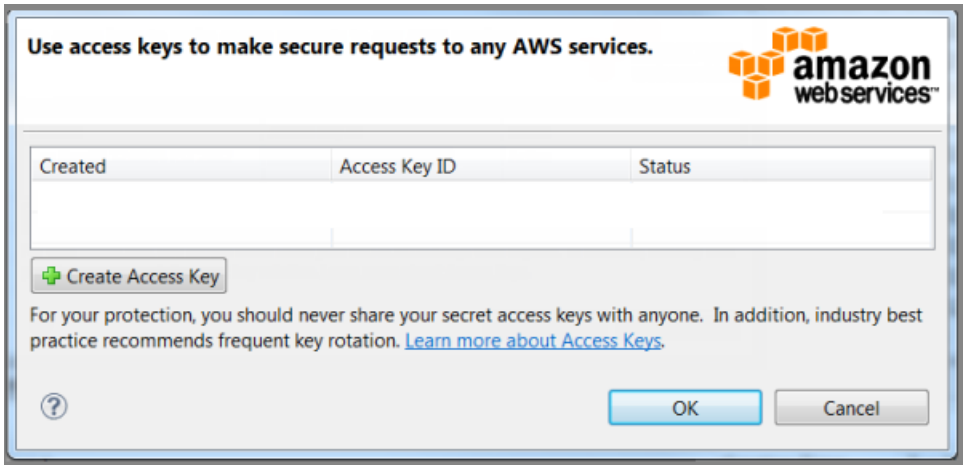
1. In **AWS Explorer**, expand the **AWS Identity and Access Management** node, right-click the **Users** node, and then select **Open Users Editor**.
2. In the users listing, select the user you want to generate keys for, and then click the **Summary** tab.



3. Click the **Manage Access Keys** button.

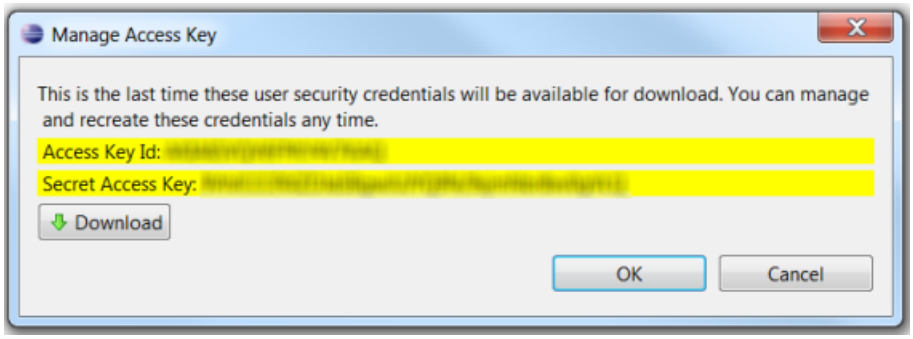
Manage Access Keys

A window is displayed where you can manage access keys for the user.



4. Click the **Create Access Key** button.

   The **Manage Access Key** dialog box is displayed.



5. Click the **Download** button to download a comma-separated value (CSV) file that contains the credentials that were generated.

   Note
   This will be your only opportunity to view and download these access keys. If you lose these keys, you must delete them and create a new set of access keys.

You can generate only two sets of credentials per IAM user. If you already have two sets of credentials and you need to create an additional set, you must delete one of the existing sets first.

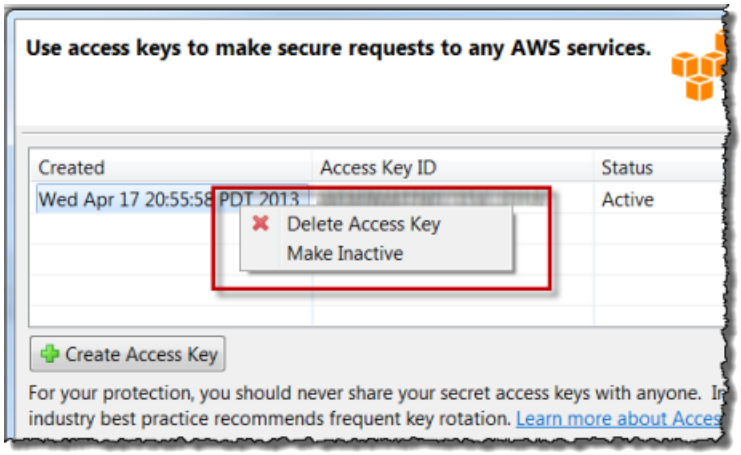
You can also deactivate credentials. In that case, the credentials still exist, but any requests to AWS that are made using those credentials will fail. This is useful if you want to temporarily disable access to AWS for that set of credentials. You can reactivate credentials that were previously deactivated.

To delete, deactivate, or reactivate access keys for an IAM user

1. In **AWS Explorer**, expand the **AWS Identity and Access Management** node, right-click the **Users** node, and then select **Open Users Editor**.

2. In the users listing, select the user you want to manage access keys for, click the **Summary** tab, and then click the **Manage Access Keys** button.
3. In the window that lists the access keys for that user, right-click the credentials you want to manage and then choose one of the following:

   - **Delete Access Key**
   - **Make Inactive**
   - **Make Active**



Create an IAM Role

Using the AWS Toolkit, you can create IAM *roles*. The role can then be *assumed* by entities that you want to allow access to your AWS resources. Policies that you attach to the role determine who can assume the role (the *trusted entity* or *principal*) and what those entities are allowed to do.

In the Toolkit, you can specify the following trusted entities:

- An AWS service. For example, you can specify that an Amazon EC2 can call other AWS services or that AWS Data Pipeline is allowed to manage Amazon EC2 instances. This is known as a *service role*.
- A different account that you own. If you have multiple AWS accounts, you might need to let users in one account use a role to get permissions to access resources that are in another account of yours.
- A third-party account. You might let a third-party vendor manage your AWS resources. In that case, you can create a role in which the trusted entity is the third party's AWS account.

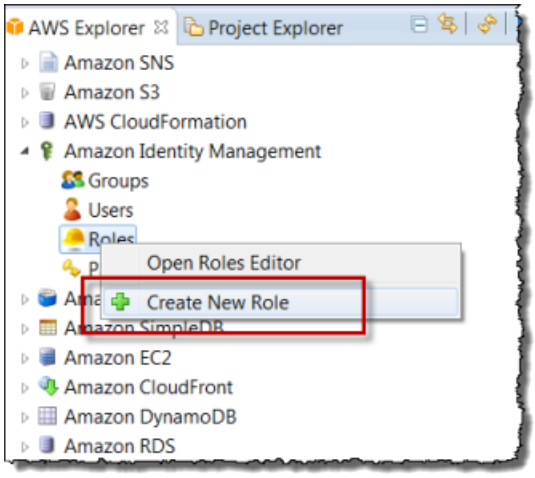
After you specify who the trusted entity is, you can specify a policy that determines what the role is allowed to do.

For example, you could create a role and attach a policy to that role that limits access to only one of your Amazon S3 buckets. You can then associate the role with an Amazon EC2 instance. When an application runs on the Amazon EC2 instance, the application can access only the Amazon S3 bucket that you allowed access to in the role's policy.

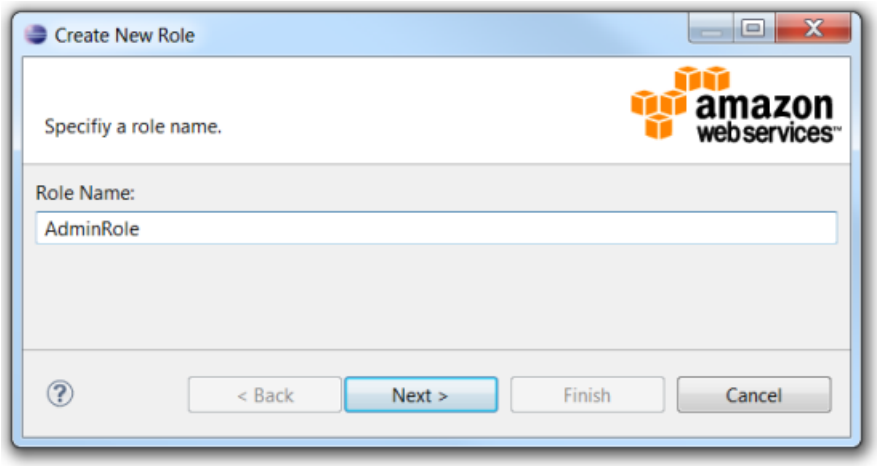
For more information about IAM roles, see Roles in the Using IAM guide.

To create an IAM role

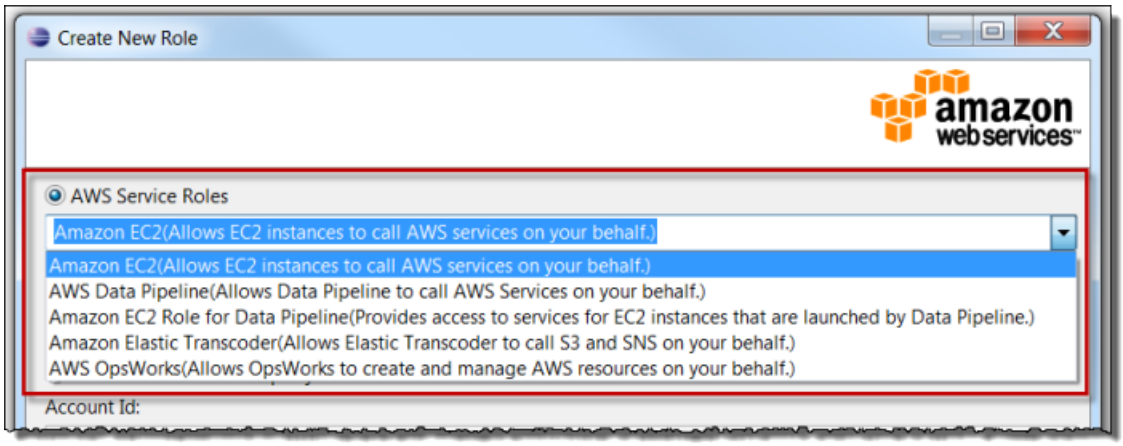
1. In **AWS Explorer**, expand the **AWS Identity and Access Management** node, right-click the **Roles** node, and then select **Create New Role**.



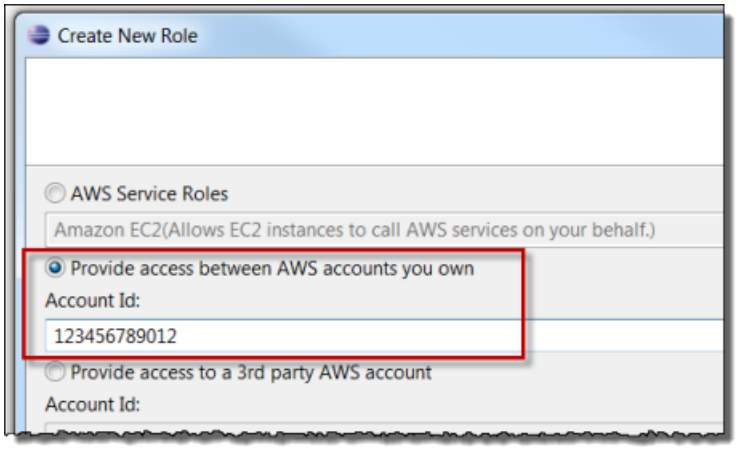
2. Enter a name for the IAM role and then click **Next**.



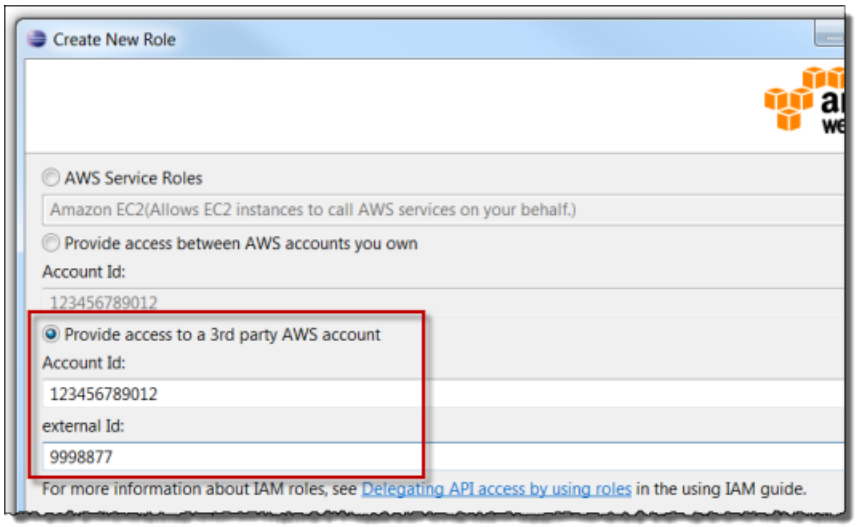
3. Select the trusted entity for the role. To create a service role, select **AWS Service Roles** and then select a service role from the drop-down list.



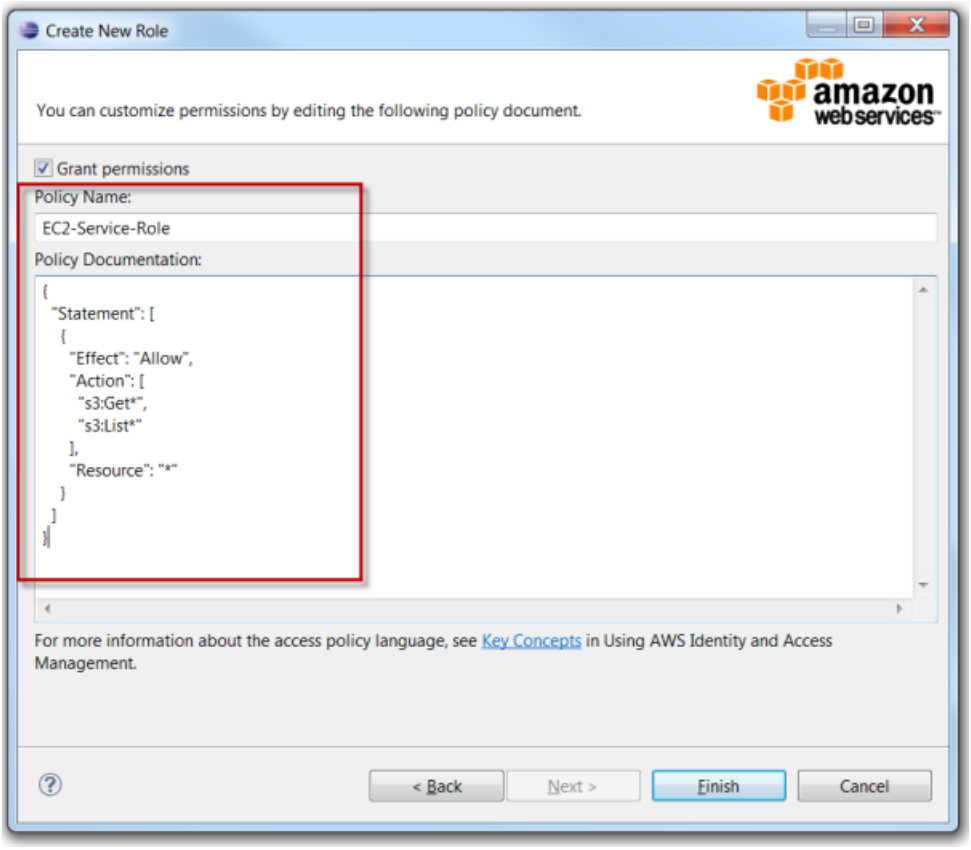
To provide access for a user that's defined in a different AWS account that you own, select **Account ID** and enter the AWS account number of the other account.



To provide access for a third-party account, select **Account ID** and enter the third party's AWS account number. If the third party has provided you with an external ID, enter that as well.



4. Click **Next**.
5. Enter a name for the policy that establishes what the role is allowed to do. Then enter the policy as a JSON document, and click **OK**.



The policy name must be unique within your account. The JSON that you enter for the policy must validate, or you will not be able to save the policy. For information about how to create a policy, see Overview of Policies in the *Using IAM* guide.

6. Click **Finish**.

The new IAM role appears in the **Roles** editor.

For examples of programs that show how to access AWS using the IAM role associated with an Amazon EC2 instance, see the following AWS developer guides:

- Java
- .NET
- PHP
- Ruby

Attach an IAM Policy to a User, Group, or Role

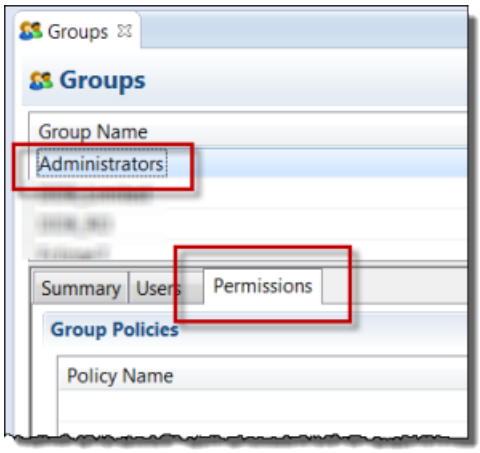
Policies are documents that define permissions. For example, a policy that's attached to a user can specify what AWS actions the user is allowed to call and what resources the user is allowed to perform the actions on. If the policy is attached to a group, the permissions apply to users in the group. If the policy is attached to a role, the permissions apply to whoever assumes the role.

The process for attaching a policy to a user or group is similar. For roles, you can attach a policy that specifies what the role is allowed to do. You use a separate process to attach or edit the policy that determines who is allowed to assume the role (that is, to manage the trust relationship.)

Note
If you attached a policy to a user, group, or role previously, you can use this procedure to attach an additional policy. To edit an existing policy on a user, group, or role, use the IAM console, command-line tools, or API calls.

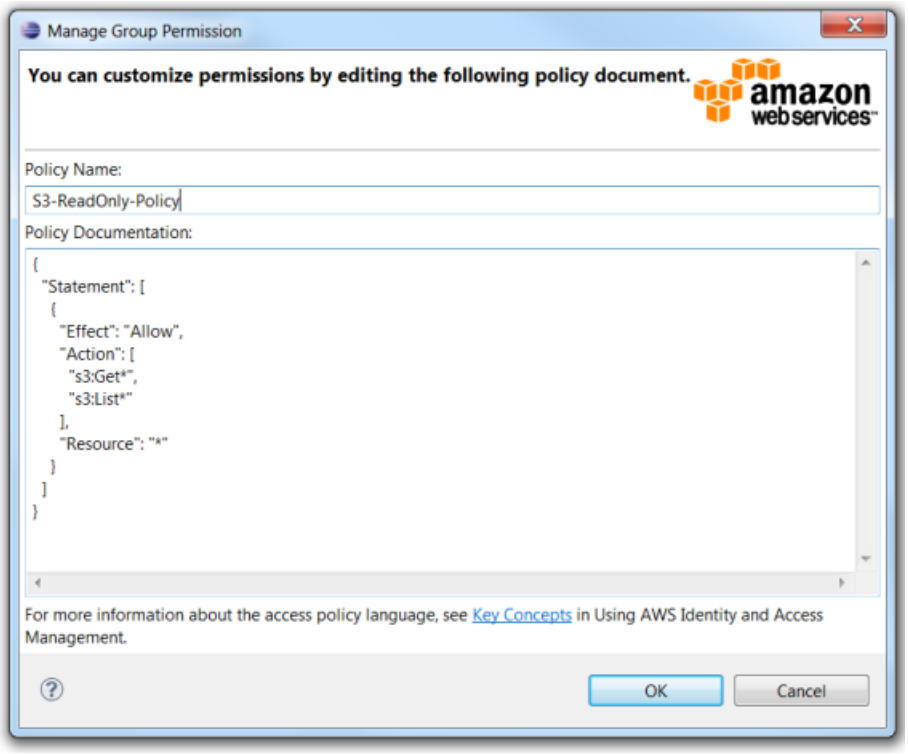
To create an IAM policy for a user, group, or role

1. In **AWS Explorer**, expand the **AWS Identity and Access Management** node and then double-click the **Groups** node, the **Users** node, or the **Roles** node.
2. Select the group, user, or role you want to attach the policy to, and then click the **Permissions** tab.



3. On the right-hand side of the bottom pane, click the **Attach Policy** button.

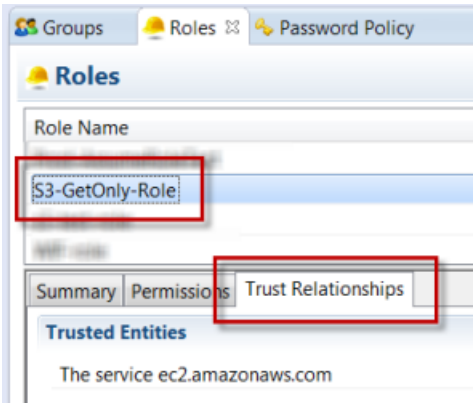
4. In the **Manage Group Policy**, **Manage User Policy**, or **Manage Role Permissions** dialog box, enter a name for the policy. Then enter the policy as a JSON document, and click **OK**.



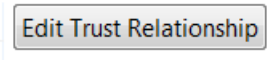
The policy name must be unique within your account. The JSON that you enter for the policy must validate, or you will not be able to save the policy. For information about how to create a policy, see Overview of Policies in the *Using IAM* guide.

To create or manage a trust relationship for a role

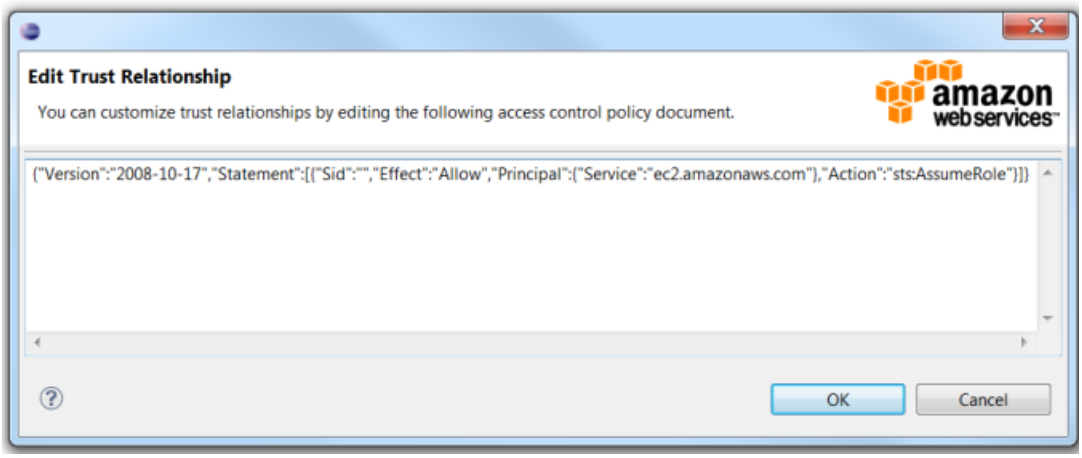
1. In **AWS Explorer**, expand the **AWS Identity and Access Management** node and then double-click the **Roles** node.
2. In the **Roles** editor, select the role you want to manage, and then click the **Trust Relationships** tab.



3. On the right-hand side of the bottom pane, click the **Edit Trust Relationship** button.



4. In the **Edit Trust Relationship** dialog box, edit the JSON policy document and then click **OK**.

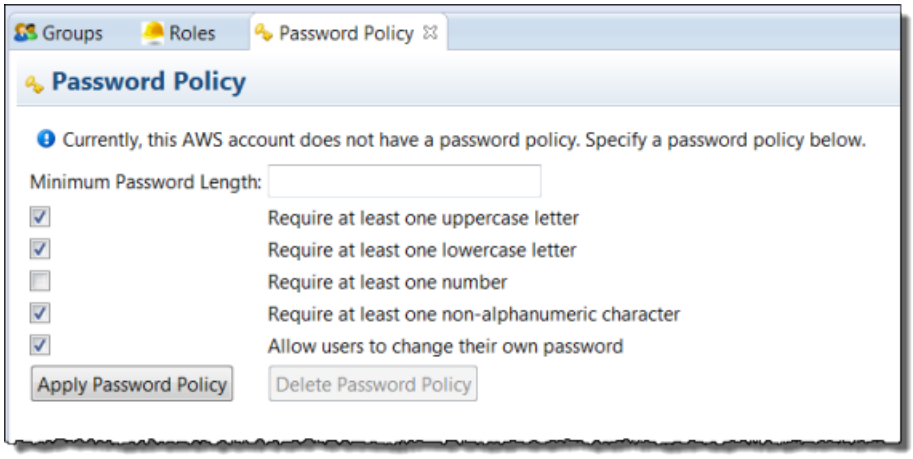


Set Password Policy

In the Toolkit for Eclipse you can set a password policy for your account. This lets you make sure that passwords that are created for IAM users follow certain guidelines for length and complexity. It also lets you specify whether users are allowed to change their own passwords. For more information, see Managing an IAM Password Policy in the *Using IAM* guide.

To create an IAM policy for a user or group

1. In **AWS Explorer**, under **Identity and Access Management**, double-click the **Password Policy** node.
2. In the **Password Policy** pane, specify the policy options that you want for your AWS account, and then click **Apply Password Policy**.



Additional Resources

For more information about the AWS Toolkit for Eclipse, please see some of these additional resources.

- Toolkit FAQ
- Getting Started with the AWS SDK for Java
- Installing the AWS Toolkit for Eclipse (Video)
- Using AWS Elastic Beanstalk with the AWS Toolkit for Eclipse (Video)
- Amazon SimpleDB Management in AWS Toolkit for Eclipse (Video)
- AWS Toolkit for Eclipse: Amazon EC2 Management (Video)

Document History

The following table describes the important changes since the last release of the *AWS Toolkit for Eclipse Getting Started Guide*.

API version: 2010-12-01

Last documentation update: May 30, 2014

Change	Description	Release Date
Updated Topic	The AWS Toolkit for Eclipse now uses the same system for storing and accessing AWS credentials as the AWS CLI and AWS SDKs, which includes the ability to use multiple profiles to store more than one set of credentials. For information, see the newly-updated topic: Working with AWS Access Credentials (p. 5).	May 30, 2014
Restructured Guide	The AWS Toolkit for Eclipse Getting Started Guide has been restructured in alignment with other AWS SDK Documentation (most notably, the AWS Java SDK upon which the Toolkit for Eclipse depends). Much of the restructuring should be logical and self-evident, but a description of each of the guide's major sections is provided in What is the AWS Toolkit for Eclipse? (p. 1).	May 30, 2014
Updated setup guide	Getting Started (p. 3) has been updated for Eclipse 4.3 ("Kepler").	September 27, 2013
New topic	This topic tracks recent changes to the *AWS Toolkit for Eclipse Getting Started Guide*. It is intended as a companion to the release notes history.	September 9, 2013