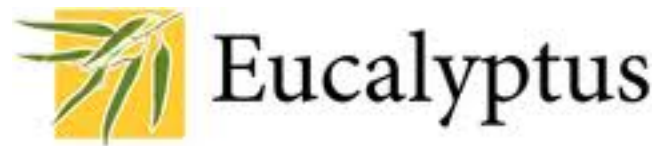


# **Eucalyptus Manual of Style**





# Contents

<b>Chapter 1: Welcome.....</b>	<b>5</b>
<b>Chapter 2: Writing the Welcome Page.....</b>	<b>7</b>
<b>Chapter 3: Writing Introductions.....</b>	<b>9</b>
Writing Prerequisites.....	10
Writing a Product or Feature Introduction.....	10
Introduction Writing Process.....	10
<b>Chapter 4: Describing and Explaining Concepts.....</b>	<b>13</b>
Glossaries, Concept Sections, and JIT.....	14
Writing Concept Descriptions.....	14
<b>Chapter 5: Writing Lists.....</b>	<b>15</b>
Simple Lists.....	16
List Tables.....	16
Verbose Lists.....	17
Topic/Comment Lists.....	18
Definition Lists.....	19
List Writing Rules.....	21
Basic Rules.....	21
List Punctuation and Case.....	21
Examples.....	21
<b>Chapter 6: Writing Processes and Procedures.....</b>	<b>23</b>
Is It a Process or a Procedure?.....	24
Writing Procedures.....	24
Define the Task.....	24
Lay Out the Steps.....	24
Procedure Standards.....	25
Examples.....	26
Conditional Step Examples.....	28
Describing Window and Menu Navigation.....	30
Writing Processes.....	30
<b>Chapter 7: Providing Navigation, Links, and References.....</b>	<b>33</b>
What is Navigation?.....	34
How is Navigation Handled?.....	34
References within the Same Product Documentation.....	34
References to Supplementary Information.....	35
References to Structured Information.....	35
Topic and Related Topic Lists.....	36
Topic Lists.....	36

Related Topics.....	36
Topic List Standards.....	36
Windows and menus.....	36
<b>Chapter 8: Formatting Content.....</b>	<b>39</b>
Examples.....	40
Formal Examples.....	40
Informal Example.....	40
Example Standards.....	40
Graphics.....	41
Tables.....	42
<b>Chapter 9: Recognizing, Avoiding, and Resolving Ambiguities.....</b>	<b>45</b>
Semantic Ambiguity.....	46
Lexical Ambiguity.....	47
Syntactic Ambiguity.....	48
Summary.....	49
<b>Chapter 10: Punctuation.....</b>	<b>51</b>
<b>Chapter 11: Editing.....</b>	<b>53</b>
<b>Chapter 12: Terms.....</b>	<b>55</b>

---

# Chapter 1

---

## Welcome

---

This is the Eucalyptus style guide for technical publications.

Welcome to the Eucalyptus Manual of Style. This guide covers Eucalyptus-specific styles and describes how to build certain types of content. For stylistic issues not covered in this guide, use the Yahoo Style Guide or, if not covered in the YSG, the Chicago Manual of Style.

Eucalyptus technical publications encompass all administration guides, user guides, and white papers.



---

# Chapter 2

---

## Writing the Welcome Page

---

| Concept definition.





---

# Chapter

## 3

---

### Writing Introductions

---

#### Topics:

- [Writing Prerequisites](#)
- [Writing a Product or Feature Introduction](#)
- [Introduction Writing Process](#)

This section of the Eucalyptus Style Guide details information about writing introductions for a publication.

This guideline describes how to write introductions to documents and topics. This is a critical piece that quickly gives the reader basic understanding of the product.



**Note:** The first sentence of every new document and every new topic **must** summarize the content that follows. Browser searches return this sentence. So, if it isn't descriptive, it's useless to someone searching for that information.

The introduction is not the same as the Welcome page, which only provides the briefest summary of the doc content. The introduction follows the Welcome and provides a more detailed product overview. For more information about Welcome pages, see [Writing the Welcome Page](#) on page 7.

## Writing Prerequisites

---

A prerequisites section sets the tone and expectations. It is extremely useful to help the readers decide whether it would waste their time to continue.

Too frequently technical reviewers assume the reader knows more about Eucalyptus or the product than we should. If there are no prereqs, you either have to explain a lot more about the product and the features as you go along, or run the risk of frustrating and losing your readers.

Before you write the introduction, ask yourself the following questions:

- What are we assuming the reader already understands about this topic? Are these assumptions obvious, or should I make them clear?
- Do we want the user to read something else, like a Getting Started, first?

### Example Prerequisite as a Tip

In many cases, you can state a prerequisite as a tip:



**Tip:** We highly recommend you read through the yet-to-be-written Eucalyptus Getting Started Guide before using this section.

### Example Prerequisites as a List

At other times, a simple list of prerequisites is sufficient:

This service enables you to store and manipulate databases in the cloud. Explaining the basics of constructing a relational database are therefore beyond the scope of this documentation. We assume you are familiar with the following:

- Databases
- SQL
- CIDR Ranges

## Writing a Product or Feature Introduction

---

It's always important to remember that we are writing technical documentation, not marketing collateral. Our job is to get the customer excited by showing them how easy the product is to use—not telling by them how cool it is.

If you aren't assuming that the readers already know about and understand the product, then you have to describe it.

## Introduction Writing Process

---

This section provides a list of guidelines for writing an introduction narrative.

1. State the formal name of the product or the feature you are describing clearly. Get the names right and use variables to assure consistent use.
2. Describe succinctly what it does or how it works:
  - Be crisp and to the point. The key is to keep it short and simple. You just want the reader to get the big picture at this point.
  - Depending on how complex it is, you might need a high-level flow diagram, a process description, or a couple simple and short declarative statements.

3. Give an example showing how someone could apply the product/feature. If you do it right, you can start a base example here and build on it throughout the doc.
4. Tell the readers where to get more info (e.g., greater details, how-tos, and references).
5. Stop. Don't overdo it.



---

# Chapter

# 4

---

## Describing and Explaining Concepts

---

### Topics:

- [Glossaries, Concept Sections, and JIT](#)
- [Writing Concept Descriptions](#)

This guideline covers how to describe and explain new concepts without losing or boring the reader. This is a critical balance that can make the difference in the users' effectiveness with the product.

This guideline covers how to describe and explain new concepts without losing or boring the reader. This is a critical balance that can make the difference in the users' effectiveness with the product.

After you have written the introduction and given the reader an overview of the product and how it works, then you develop the individual concepts that they need to understand.

Concept descriptions differ from the product introduction because they are more topic focused and less holistic. The introduction follows the Welcome and provides a detailed product overview. Concept descriptions can appear throughout the documentation and give more exacting information on specific components or ideas. For more information about introductions, see [Writing Introductions](#) on page 9.



**Important:** Before you start, the first step is to check all egos at the door. Readers don't want to be impressed with our knowledge; they want to understand the product.

You can use a variety of methods to describe a concept. You might use diagrams, provide an example, or even write a short tutorial. The amount of material you provide is basically only bounded by three considerations:

- How little do they need to know to be productive?
- How much do they need to know to avoid problems?
- How much more productive will they be if I teach them to fish?

Remember that your readers want to spend as little time with the documentation as possible, and have even less patience if you dive them into the background details. On the other hand, the effort is worthwhile if you can show them how they can avoid returning to the documentation again.

## Glossaries, Concept Sections, and JIT

---

This section defines and presents usage information for glossaries, concepts, and just-in-time information.

Glossaries provide centralized definitions and are important as a reference and for keeping the documentation concise. With a glossary, you can define a simple term once and refer to the glossary whenever you use a term for the first time in a major section.

Just in time (JIT) concepts and definitions allow you to present simple concepts when you introduce processes, procedures or examples. You use JIT concepts when the concept is only needed in one context, or you can easily write, reuse and maintain a short concept in multiple locations.

Develop and use a new concepts section when:

- The product has many components that are central to its function.

This section should be a part of the product overview.

- The product is complex and has many special features.

Major features often have their own sections. The concepts will likely follow the feature introduction.

## Writing Concept Descriptions

---

Provides guidelines for providing conceptual information.

All concepts whether they are in a special section or JIT should provide the following information:

- Name each concept and use the same naming consistently.

Inconsistent naming and terminology slows comprehension, and can lead to misunderstandings and errors.

- What makes it important?

If you can't explain why a concept is important, either you don't understand it or the user doesn't need to be bothered with it.

- When do you use it?

Whenever possible, our explanations should be functional. Always steer the focus to making the customer productive.

- How does it work?

If it is a function or process that is outside the user's control, use a diagram or examples to show how it functions or flows.

- Point (link) to any information on how to use the concept.

Don't put instructions into the concept descriptions. You can enhance a procedure with concepts (i.e. the JIT model). However, users shouldn't have to hunt through a conceptual section to find information on how to use the product.



**Important:** Always make it easy to jump directly to the hands-on instructions.

---

# Chapter

# 5

---

## Writing Lists

---

### Topics:

- [Simple Lists](#)
- [List Tables](#)
- [Verbose Lists](#)
- [Topic/Comment Lists](#)
- [Definition Lists](#)
- [List Writing Rules](#)

The purpose of a list is to itemize data relating to a topic or to present conceptually similar information in a format that is easy for the reader to scan. Lists shouldn't replace normal text or the structured presentation of tables.

At Eucalyptus, we use three basic types of lists: simple, verbose, and topic/comment:

- The simple list is usually a short series of items. This is the most common kind of list.
- A verbose list displays a series of short closely-related sentences together. Don't use this style to replace normal paragraphs on related subjects.
- The topic/comment list is balanced between the simple list, the verbose list, tables, glossaries, and the formal paragraph. Use the topic/comment list when you have a short simple list that needs a brief explanation for each item.

When the content becomes more complex or lengthy, the scanning effect of a list is lost. Although, many writers use a list anyway for the special formatting, at Eucalyptus you should use a series of formal paragraphs.

## Simple Lists

The simple list is usually a short series of items. This is what you use most of the time.

Quick guidelines:

- Each list item is a single word or concept phrase.
- Only punctuate a simple list if the list items are complete sentences.
- You can include a parenthetical thought or phrase.



**Important:** If a simple list is over half a page long, consider putting it into a multicolumn table. If the list exceeds an entire page in length, put it in a table.

### Example Very Simple List

Note that the introductory sentence ends in a colon.

```
<p>Following is a list of common barnyard poultry:</p>
<ul>
<li>Chickens</li>
<li>Ducks (domestic)</li>
<li>Geese</li>
</ul>
```

The preceding markup gives the following output.

Following is a list of common barnyard poultry:

- Chickens
- Ducks (domestic)
- Geese

Following is another good example of a simple list.

### Example Short Phrase Simple List

Each error is comprised of the following elements:

- A **code** that identifies the type of error
- A **detail** that might give additional details about the error
- A **message** that describes the error condition
- A **type** that identifies whether the error was from a receiver or sender

## List Tables

In order to avoid a lengthy bullet list that takes up vast amounts of white space, wrap the items into a table.

- |              |            |                |
|--------------|------------|----------------|
| • Benedetto  | • Eastman  | • Gibson       |
| • Carvin     | • Epiphone | • Gretsch      |
| • Danelectro | • Fender   | • Guild        |
| • D'Angelico | • Ibanez   | • Rickenbacker |
| • D'Aquisto  | • Heritage | • Vox          |

Use the following sample markup:

```
<table id="listtable" frame="none">
  <tgroup cols="3">
    <colspec colnum="1" colname="col1" colsep="0" colwidth="2*" /><colspec
colnum="2" colname="col2" colsep="0"
```



```

colwidth="2*"/><colspec colnum="3" colname="col3" colsep="0"
colwidth="2*"/>
<tbody>
<row>
<entry>
<ul>
<li>Benedetto</li>
<li>Carvin</li>
<li>Danelectro</li>
<li>D'Angelico</li>
<li>D'Aquisto</li>

</ul>
</entry>
<entry>
<ul>
<li>Eastman</li>
<li>Epiphone</li>
<li>Fender</li>
<li>Ibanez</li>
<li>Heritage</li>
</ul>
</entry>
<entry>
<ul>
<li>Gibson</li>
<li>Gretsch</li>
<li>Guild</li>
<li>Rickenbacker</li>
<li>Vox</li>
</ul>
</entry>
</row>
</tbody>
</tgroup>
</table>

```

## Verbose Lists

The verbose list displays a series of short, closely-related sentences together.

Quick guidelines:

- Always fully punctuate a verbose list.
- Don't use a list if the paragraphs are more than two or three sentences long.
- Don't use a list if the topics are only loosely connected.
- Use a table or formal paragraphs if a verbose list has more than seven items.



**Note:** Don't use this style to replace normal paragraphs on related subjects. You should change from a list to paragraphs either when there are many sentences for each topic or when the parallelism between the topics isn't strong. For more discussion, see the section on formal paragraphs.

### Example Verbose List

Here are some suggestions for making the best use of Amazon EC2 instances:

- Do not rely on an instance's local storage for valuable, long-term data. When instances fail, the data on the local disk is lost. Use a replication strategy across multiple instances to keep your data safe, or store your persistent data in Amazon S3, or use Amazon EBS.

- Define images based on the type of work they perform. For "Internet applications," you might define one image for database instances and another for web servers. Image creation and storage are cheap and easy operations, so you can individualize and customize as necessary. Specialized images can result in smaller AMI sizes, which boot considerably faster.
- Keep your Amazon EC2 firewall permissions as restrictive as possible. Only open up permissions that you require. Use separate groups to deal with instances that have different security requirements. Consider using additional security measures inside your instance (such as using your own firewall).

## Topic/Comment Lists

The topic/comment list is balanced between the simple list, the verbose list, tables, glossaries, and the formal paragraph. It is comprised of a list of topic ideas (a simple list) each followed by a description (a simple or verbose list).

When to use topic/comment:

- Use the topic/comment list when you have a short, simple list that needs a brief explanation for each item.
- If you need a more than two sentences to clarify the topics, use formal paragraphs instead.
- If a topic/comment list has more than seven items, use a table or formal paragraphs instead.

Formatting guidelines:

- Only use title with a topic/comment list when you need the list to stand out significantly from the flow.
- The topic is a brief phrase to orient the reader.
- The comment is a short explanation of the topic.
- Capitalize the first word following the em-dash in a topic/comment list.
- If you need a slightly longer discussion of the topic, put it on a separate line.

Punctuation guidelines:

- Use an em-dash to separate the topic from the comment. This is a part of the coding (see the following example).
- Don't put spaces before or after the dash.
- Only use sentence punctuation if the list items are complete sentences.

### Example Topic/Comment Coding

The example menu shows how to apply the `<b>` an em-dash. The description follows immediately after the dash. This example also shows how to add a second line to a list item using `<p>`.

```
<ul>
  <li><b>Chicken</b>Grown in Washington
    <p>We serve only free range, Washington chicken.</p>
  </li>
  <li><b>Duck</b>a l'orange
    <p>Our oranges are freshly imported from Spain every
week.</p>
  </li>
  <li><b>Goose</b>Stuffed with oysters
    <p>We select the finest oysters from our own Hood Canal
oyster beds.</p>
  </li>
</ul>
```

The preceding markup gives the following output.

- **Chicken**—Grown in Washington We serve only free range, Washington chicken.
- **Duck**—a l'orange Our oranges are freshly imported from Spain every week.
- **Goose**—Stuffed with oysters We select the finest oysters from our own Hood Canal oyster beds.

Don't overuse topic/comment lists to build constructions that a table would handle more effectively. In particular, if the list is longer than about seven items or the content is essentially a number of key product terms followed by descriptions (as in the following example), use a table.

**Example List That Should Be a Table**

*This list should be a table because it is actually a set of parameter descriptions.*

The following response elements are typically used in the display of promotion information:

- **BenefitDescription**—Describes the benefit, which is the item(s) that the customer receives as a result of the promotion.

This element will not be present if the item is not part of the promotional benefits.

- **EligibilityDescription**—Describes the items the customer must purchase to qualify for the promotion.

This element will not be present if the item does not qualify the customer to receive the promotional benefit.

- **TermsAndConditions**—Specifies the terms and conditions of the promotion.
- **ComponentType**—Specifies what the promotion applies to, for example, Shipping, ItemPrice, Subtotal.

## Definition Lists

---

Whenever the content becomes more complex, it is often better to use a definition list. Even though the content is similarly structured and parallel, the effect of a list is lost because of the length and complexity. In the following example, the definition list term titles replace the title component and the definition of the term is the content. Note that it would be cumbersome to construct a table to achieve the same effect.

**Example Definition List Coding**

```
<dl>
  <dlentry>
    <dt>Topic title goes here</dt>
    <dd>Descriptive text goes here.</dd>
  </dlentry>
</dl>
```

**Example Definition List in Practice**

You can get a list of parts that you have uploaded for a specific multipart upload or a list of all your in-progress multipart uploads. Each of these operations is explained in more detail in this section.

<b>Multipart Upload Initiation</b>	<p>When you send an initiate multipart upload request, Amazon S3 returns a response with an upload ID that is a unique identifier for your multipart upload. You must include this upload ID whenever you upload parts, complete an upload, or abort an upload. If you want to provide any metadata describing the object being</p>
--	---

## Parts Upload

uploaded, you must provide it in the multipart upload initiation request.

When uploading a part, in addition to the upload ID, you must specify a part number. You can choose any part number from 1 to 10,000. A part number uniquely identifies a part and its position within the object you are uploading. If you upload a new part using the same part number as a previously uploaded part, that previously uploaded part is overwritten.

Whenever you upload a part, Amazon S3 returns an ETag header in its response. For each part upload, you must record the part number and the ETag value. You need to include these values in the subsequent complete multipart upload request.

## Multipart Upload Listings

You can list the parts of a specific multipart upload or all in-progress multipart uploads. The list parts operation returns the parts information that you have uploaded for a specific multipart upload. For each list parts request, Amazon S3 returns the parts information for the

specified multipart upload, up to a maximum of 1,000 parts. If more than 1,000 parts exist in the multipart upload, you must send a series of list part requests to retrieve all the parts. Note that this parts list does not include parts that haven't completed uploading.

## List Writing Rules

---

### Basic Rules

These are the basic guidelines for all lists.

- Make lists parallel in content and structure. Don't mix single words with phrases, don't start some phrases with a noun and others with a verb, and don't mix verb forms.
- Place any mundane commentary about the items in a simple list outside the list (see the following example).
- Use lists to call out important ideas, concepts, or information.
- Whenever the order of items in a list is arbitrary, present the items in alphabetical order.

### List Punctuation and Case

Follow these guidelines on case and punctuation.

- Capitalize the first letter of the first word of each list item.
- Capitalize the first word following the em-dash in a topic/comment list.
- Don't circumvent the topic/comment list style by dividing list item phrases with colons.
- Don't close lists with a final period if the rest of the list isn't punctuated.
- End the phrase that introduces a list with a colon.
- If a list item contains more than one sentence, use normal paragraph punctuation.
- Whenever punctuation is necessary, then use it consistently within that list. See the MSTP and the CMS for further information.

### Examples

#### Example List Parallelism and Ordering

Wrong:

- Accounts
- What you do on first of the month
- Weekly actions
- Start up
- Getting your tools

Right:

- Getting started
- Getting your tools

- Managing accounts
- Organizing each week
- Starting the month

### Example Simple List With an Admonition

#### Wrong:

For each message returned, the components of the response are presented in the following order:

- Message ID
- Receipt handle



**Note:** The receipt handle is the identifier you must provide when deleting the message (for more information, see Queue and Message Identifiers). You received the message ID when you sent the message to the queue. For information about MD5, go to <http://faqs.org/rfcs/rfc1321.html>).

- MD5 digest of the message body
- Message body
- Request ID

#### Wrong:

For each message returned, the components of the response are presented in the following order:

- Message ID
- Receipt handle

The receipt handle is the identifier you must provide when deleting the message (for more information, see Queue and Message Identifiers). You received the message ID when you sent the message to the queue. For information about MD5, go to <http://faqs.org/rfcs/rfc1321.html>).

- MD5 digest of the message body
- Message body
- Request ID

#### Right:

For each message returned, the components of the response are presented in the following order:

- Message ID
- MD5 digest of the message body
- Message body
- Receipt handle
- Request ID



**Note:** The receipt handle is the identifier you must provide when deleting the message (for more information, see Queue and Message Identifiers). You received the message ID when you sent the message to the queue. For information about MD5, go to <http://faqs.org/rfcs/rfc1321.html>).

---

# Chapter 6

---

## Writing Processes and Procedures

---

### Topics:

- [\*Is It a Process or a Procedure?\*](#)
- [\*Writing Procedures\*](#)
- [\*Writing Processes\*](#)

This topic describes how to know the difference between a process and a procedure, and provides step-by-step instructions for writing them.



**Tip:** The Five Cs: All technical writing should follow the rubric: Clear, Concise, Complete, Correct, Consistent

## Is It a Process or a Procedure?

---

The major difference between a process and a procedure is that a process amplifies a concept, and a procedure is a set of elemental instructions. A process either clarifies how something works or describes a series of procedures. By contrast, a procedure instructs how to do something—describing each necessary step completely. You can keep that distinction clear for the reader by using different styles.

A process can answer the questions: How does it work?, What happens when?, and When do I need to do it?

A procedure primarily answers the question: How do I do it?. But it can also answer: Where do I do it?, and What does the result look like?

At Eucalyptus we use the following process and procedure terminology:

- **Task**—A discrete, complete activity. A procedure completes a task; a process can be composed of several tasks.
- **Step**—An elemental action in a procedure.
- **Substep**—An incremental pieces of a step. Substeps are often necessary to break down a step when it has multiple components.

Art, skill, and house style are involved in defining the gray line between a complex procedure with many substeps and a process containing multiple procedures. Deciding which is better comes as a part of the developmental edit. As you gain experience with our house style, your writing becomes more consistent, requiring fewer edits.

## Writing Procedures

---

A procedure can stand alone, or be a part of a greater process or life cycle. These distinctions are important in deciding where and how to present the procedure. If it's part of a process or life cycle, you might want to group its elements together and in sequential order with other procedures of the same kind. If it stands alone, you might need to decide how much prominence it needs and how to make it easy for the reader to find.

### Define the Task

Before you begin to write a procedure, be clear in your own mind what the task is. If you don't know, it's a sure bet that your readers won't.

It's a common error to try to pack more than one task in to a single procedure because the writer understands the bigger picture and wants to tell the whole story quickly. The reader, however, might only want to do one of the tasks—for reasons or circumstances you don't foresee.

For example, you might write a procedure for logging in to a new product. But before readers can actually log in, they have to sign up for the product and get their security credentials. So, you might combine the two and call the procedure "To log in to ProductX." If however, at a later time, the user wants to remember how to look up the credentials, it's harder to find the right procedure, because it's buried in the middle of other instructions.



**Tip:** Remember to be concise: One task, one procedure.

After you have defined the task, you need to lay out the steps to accomplish that task in the simplest terms.

### Lay Out the Steps

Procedures are instructions; they step the reader through a task from beginning to end. Each step might require some detail, such as navigation, substeps, or examples; but before you write the details, you need to sketch out the complete list of steps and verify that you have it all.



**Important:** Every procedure is based on some prerequisite knowledge. That knowledge might be rudimentary, like knowing what the ALT key is. In technical writing, it's more likely that you will set a



knowledge level more like: "Familiarity with SQL commands."

When writing a procedure that is a part of a process or life cycle, you might write some simple procedures (such as accessing a UI), which you can assume the reader has mastered for the remaining procedures in that group.

Procedures are comprised of a list of one or more steps. Each step is written as a simple command, beginning with either a verb or a navigation point. If navigation in the UI is needed before an action, give the navigation instruction first.

## Procedure Standards

### Headings Cascade for Procedural Sections

We diverge from classic style in our headings to be more direct in identifying sections that contain a procedure:

- Using the Widget Service [major section level]
- Working with Whatsit Fields [section level]
- How to Modify a Whatsit Field [topic level]
- To modify a whatsit field [procedure heading]

The procedure heading must always precede a formal procedure. The "How To" format is a terminal heading that contains a discrete procedure. The gerund heading form indicates a container that can have multiple processes and procedures, or subsections with processes and procedures.

### General

- Don't present alternate methods for completing an instruction. Choose the best one for the audience and context, and use it consistently.
- Each instruction must be complete and not assume prior knowledge without adequate reference. For example, an instruction to go to a web site should be accompanied by the URL or a reference to where the URL is defined.
- Each step contains only one instruction.
- Number optional steps with the rest, but introduce them with an If-clause, such as "If you can't see the downloaded file, use the Search button."
- Use the approved terminology for procedures from our style guide or the Yahoo Style Guide.

### Length

- An ideal procedure has no more than 5 steps. Under no circumstances should a procedure exceed 10 steps.
- Divide a lengthy procedure into 2 or more procedures or reorganize it into a shorter procedure with substeps.
- Keep steps short and simple. If a step requires an explanation, put it in a Note or Tip.

### Formatting

- Precede every procedure with a heading in the infinitive, "To XYZ..." format. The heading is in sentence case and is not punctuated.
- The first step immediately follows the heading. No comment, admonition, or description can precede the first step—place such text before the procedure heading.
- Precede each step in the list with an Arabic number.
- A single-step procedure is not numbered, but uses a bullet to mark the instruction.
- Precede each substep with a lowercase letter.
- Follow standard formatting rules for text within the procedure (e.g. labels in <uicontrol> and action names in <apiname> etc.).
- Admonitions that apply to a single step follow the instruction on a new line.
- If there is a visible result that is a direct consequence of completing an instruction, it should follow the instruction, and any admonitions, on a new line.
- Examples or screen shots related to a step follow all the descriptive text. That is, no text related to a step should follow the example.

## Conditional Steps

- Conditional steps are not optional, but can differ depending on the user's needs to provide multiple or alternate options. You might have multiple options for a substep, or you might have different subprocedures depending upon a certain condition. Both can be handled with procedure boxes. See the following examples.
- Conditional steps allow you to present a series of optional actions.
- Conditional steps allow you to have different sub-procedures depending upon a certain condition.

## Examples

The following examples and sample markup demonstrate this standard applied to Eucalyptus documentation.

### Example Simple Procedure with <uicontrol> Tag

This example shows a typical procedure with references to objects in a UI.

1. Go to the [AWS web site](#).
2. Point to the **Your Web Services Account** button on the top-right corner of the page.
3. Click **AWS Access Identifiers**.
4. If you haven't signed in, follow the prompts to log in.
5. Your access key information is displayed in the **Access Key ID and Secret Access Key** area.

```
<steps>
  <step>
    <cmd>Go to the <xref href="aws.amazon.com">AWS web site</xref>.</cmd>
  </step>

  <step>
    <cmd>Point to the <uicontrol>Your Web Services Account</uicontrol> button on the top-right corner of the page.</cmd>
  </step>

  <step>
    <cmd>Click <uicontrol>AWS Access Identifiers</uicontrol>.</cmd>
  </step>

  <step>
    <cmd>If you haven't signed in, follow the prompts to log in.</cmd>
  </step>

  <step>
    <cmd>Your access key information is displayed in the <uicontrol>Access Key ID and Secret Access Key</uicontrol> area.</cmd>
  </step>
</steps>
```

### Example Procedure with Admonition

The following example shows proper placement of an admonition.

1. On the **Configuration** tab in the Eucalyptus Dashboard, change the number of buckets for each user, and then click **Save Walrus configuration**.
2. Click **Add more files** to select the files to upload from your computer.
3. Select one or more files and click **Open**.



**Note:** You cannot select a folder to upload its entire contents. You must select each file you want to upload.

#### 4. Confirm the list of files and click **Start Upload**.

```
<steps>
  <step>
    <cmd>On the <uicontrol>Amazon S3</uicontrol> tab in the AWS
    Management Console, click the bucket
      you want to upload files into, and then click
    <uicontrol>Upload</uicontrol>.</cmd>
  </step>
  <step>
    <cmd>Click <uicontrol>Add more files</uicontrol> to select the
    files to upload from your
      computer.</cmd>
  </step>
  <step>
    <cmd>Select one or more files and click <uicontrol>Open</
    uicontrol>.</cmd>
    <info><note>You cannot select a folder to upload its entire
      contents. You must select each file
        you want to upload.</note></info>
  </step>
  <step>
    <cmd>Confirm the list of files and click <uicontrol>Start
    Upload</uicontrol>.</cmd>
  </step>
</steps>
```

#### Example Procedure with Substeps

This procedure shows how to break up a longer procedure with a set of substeps.

1. From the Windows Start menu, open the WSE Configuration tool.
2. Open the `wse3policyCache.config` file for the Enqueue project (`..\C#\SimpleQueueService\Enqueue\`).
3. On the **General** tab, select **Enable this project for Web Services Enhancements**.
4. If you haven't signed in, follow the prompts to log in.
5. Click the **Policy** tab to create a new policy:
  - a) Select **Enable Policy**.
  - b) Under **Edit Application Policy**, click **Add** to specify the policy to enable.  
The **Add or Modify Policy Friendly Name** dialog box opens.
  - c) In the **Add or Modify Policy Friendly Name** dialog box, type a name for the name of the policy, and then click **OK**.  
The **WSE Security Settings Wizard** is displayed.
  - d) Follow the instructions in the wizard to create your new policy.

```
<steps>
  <step>
    <cmd>From the Windows Start menu, open the WSE Configuration
    tool.</cmd>
  </step>
  <step>
    <cmd>Open the <filepath>wse3policyCache.config</filepath> file
    for the Enqueue project
      (<filepath>..\C#\SimpleQueueService\Enqueue\</filepath>).</cmd>
  </step>
  <step>
    <cmd>On the <uicontrol>General</uicontrol> tab, select
    <uicontrol>Enable this project for Web
      Services Enhancements</uicontrol>.</cmd>
  </step>
```

```

<step>
  <cmd>If you haven't signed in, follow the prompts to log in.</cmd>
</step>
<step>
  <cmd>Click the <uicontrol>Policy</uicontrol> tab to create a
  new policy:</cmd>
  <substeps>
    <substep>
      <cmd>Select <uicontrol>Enable Policy</uicontrol>.</cmd>
    </substep>
    <substep>
      <cmd>Under <uicontrol>Edit Application Policy</uicontrol>,
      click <uicontrol>Add</uicontrol>
      to specify the policy to enable.</cmd>
      <stepresult>The <uicontrol>Add or Modify Policy Friendly
      Name</uicontrol> dialog box
      opens.</stepresult>
    </substep>
    <substep>
      <cmd>In the <uicontrol>Add or Modify Policy Friendly Name</uicontrol> dialog box, type a
      name for the name of the policy, and then click
      <uicontrol>OK</uicontrol>.</cmd>
      <stepresult>The <uicontrol>WSE Security Settings Wizard</uicontrol> is
      displayed.</stepresult>
    </substep>
    <substep>
      <cmd>Follow the instructions in the wizard to create your new
      policy.</cmd>
    </substep>
  </substeps>
</step>
</steps>

```

## Conditional Step Examples

The following examples show procedures with multiple options and different subprocedures.

### Example Choice Table

The choice table allows you to present a series of optional actions. The following example details the steps to perform a simple search.

1. From the Select drop-down list box, select the server that has your report result.
2. If the filter does not show, click **Simple Search**.
3. Choose one of the following actions:

#### To...

**List all available reports results.**

**Search for a report name.**

#### Do This...

Click the Search button.

In the Report Name text box, type part of your report's name, and then click the Search button.

```

<steps>
<step>
  <cmd>From the Select drop-down list box, select the server that
  has your report result.</cmd>

```

```

</step>
<step>
  <cmd>If the filter does not show, click <uicontrol>Simple
  Search</uicontrol>.</cmd>
</step>
<step>
  <cmd>Choose one of the following actions:</cmd>
  <choicetable>
    <chhead>
      <choptionhd>To...</choptionhd>
      <chdeschd>Do This...</chdeschd>
    </chhead>
    <chrow>
      <choption>List all available reports results.</choption>
      <chdesc>Click the Search button.</chdesc>
    </chrow>
    <chrow>
      <choption>Search for a report name.</choption>
      <chdesc>In the Report Name text box, type part of your
      report's name, and then click the
      Search button.</chdesc>
    </chrow>
  </choicetable>
</step>
</steps>

```

### Example Conditional Split Procedure

This example shows a procedure that splits into different paths following a condition. The example that follows is for steps to handle retry failures.

1. Retrieve the row corresponding to the caller reference.
2. Retry or fail the transaction.

#### To Retry a Transaction

**Change transaction status to Pending.**

**Poll results using GetResults.**

**Change transaction status to Failure.**

**E-mail the customer a description of the failure.**

#### To Fail a Transaction

Call RetryTransaction.

Discard results using DiscardResults.

Change download status to Download Canceled.

Discard results using DiscardResults.

```

<steps>
  <step>
    <cmd>Retrieve the row corresponding to the caller reference.</
    cmd>
  </step>
  <step>
    <cmd>Retry or fail the transaction.</cmd>
    <choicetable>
      <chhead>
        <choptionhd>To Retry a Transaction</choptionhd>
        <chdeschd>To Fail a Transaction</chdeschd>
      </chhead>
      <chrow>
        <choption>Change transaction status to Pending.</choption>
        <chdesc>Call RetryTransaction.</chdesc>
      </chrow>
      <chrow>

```

```

    <choption>Poll results using GetResults.</choption>
    <chdesc>Discard results using DiscardResults.</chdesc>
  </chrow>
  <chrow>
    <choption>Change transaction status to Failure.</choption>
    <chdesc>Change download status to Download Canceled.</chdesc>
  </chrow>
  <chrow>
    <choption>E-mail the customer a description of the failure.</choption>
  </chrow>
  <chdesc>Discard results using DiscardResults.</chdesc>
</choicetable>
</step>
</steps>

```

## Describing Window and Menu Navigation

When describing a navigation through a series of windows or menus, follow these guidelines.

:

- For navigation through a series of menus, use one of the following methods to describe it:

- From the **File** menu, go to **History**, click **Last 7 Days**, and then make your selection.
- On the **File** menu, point to **History**, **Last 7 Days**, and then make your selection.



**Note:** Don't use arrows or slashes for menu navigation.

- For navigation through a series of windows, use one of the following methods to describe it:

- From the **Windows Control Panel**, open the **Administrative Tools and Local Security Settings**, and then from the **Action** menu, select the **Export List**.
- From the **Windows Control Panel**, go to **Administrative Tools > Local Security Settings**, and then from the **Action** menu, select the **Export List**.



**Note:** Don't take readers through the Windows Start menu unless it is absolutely necessary. Different versions of Windows have different navigations through Start and users know how to find basics such as the programs list and the control panel.

Don't use other types of arrows or slashes for window navigation. Only use the simple right arrow bracket (>) as the shortcut notation for navigating through multiple menu layers.

## Writing Processes

A process can answer the questions: How does it work?, What happens when?, and When do I need to do it? Because they can describe general sequences, processes are less rigorously defined than procedures are. For example, whereas procedures are comprised of a discrete collection of elemental instructions, processes have a variety of components—of unequal complexity. The one basic rule is that a process must convey a cohesive whole that progresses over time. If the concept can't be expressed as a temporal sequence, then it isn't a process.

### Process Standards

- The content of processes can vary according to the situation and needs. Basically, however, a process should illustrate how a single concept works or flows. Anything extraneous to that one concept should be removed, or only briefly referred to.

- Most processes are greatly enhanced with a diagram, which helps the reader see the entire concept at a glance. However, a diagram should supplement, not replace, written descriptions.
- Mark up a process as an ordered list with special attributes. The XSL and CSS will handle the rest of the formatting.
- Refer to a collection of different components as tasks. That is, a process that is a collection of tasks, procedures, and steps that should be collectively called tasks in the documentation.

### **Example Processes for Different Uses**

This example is a simple enumeration of tasks that must be completed. It is not supported with any kind of graphic. The following sample code shows how the list is constructed.

1. Create a Walrus bucket.
2. Place your content in the Walrus bucket.
3. Launch an instance of your web server machine image.
4. Specify the Walrus bucket containing the web content.





---

# Chapter

# 7

---

## Providing Navigation, Links, and References

---

### Topics:

- [\*What is Navigation?\*](#)
- [\*References within the Same Product Documentation\*](#)
- [\*References to Supplementary Information\*](#)
- [\*References to Structured Information\*](#)
- [\*Topic and Related Topic Lists\*](#)
- [\*Windows and menus\*](#)

We can help readers navigate more quickly and reduce translation costs if we use consistent methods and phrases to introduce referents and links. Navigation eases readers' experience with documentation. It enables them to see what information is available, understand how it might be inter-related, and sometimes to move quickly from point to point within and across information resources.

## What is Navigation?

---

For documentation, navigation is the ability to move from any point in a document to any point in another reference.

For ease of discussion here, we will use the following terms:

- **Referent**—the navigation's point of departure
- **Link**—Any kind of hyperlinking mechanism
- **Target**—The destination that the referent points to
- **Jump**—Moving the readers' view from referent to target

Not all referents are links. For example, a referent might target a table immediately following it, or it might suggest an offline resource.

Targets include:

- Appendices
- Document or site hierarchies
- Document structure elements (top/bottom of page; next/previous pages; top of doc)
- External documents and resources (e.g. IETF reference docs, W3C)
- External resources (e.g. W3C)
- Glossaries
- Indexes
- Objects in a document (i.e. examples, tables, or images)
- Related documents (e.g. another product's Dev Guide)
- Related resources (e.g. code samples)
- Related topics in the same document
- Related topics in a companion document (e.g. from Dev Guide to GSG)
- Subtopics of a main topic

Some of these are generated automatically by our output production process; others are deliberate constructions for the writer to specify.

## How is Navigation Handled?

There are several methods available for navigation.

- **Button**—A graphic you click to jump
- **Hypertext**—Text you can click to jump to the target
- **Textual reference**—A general reference that doesn't automatically jump the reader to the target

In addition, there are GUI mechanisms, such as menus, tabs and list boxes. These are mentioned for completeness, but won't be discussed further here, because they are uncommon features for technical documentation and behave the same as other links.

## References within the Same Product Documentation

---

Within a document you have a couple options to orient the reader to additional information. Basically, you can point to content on the same page or to information elsewhere in the doc. Use the following guidelines for same-page and off-page references.

### Same Page References

When the target is another paragraph or object on the same page as the referent:

- Don't use a link. Instead, refer to the target as "following" or "preceding."

Exception: If the page is more than three screens long, and the target is also in another major section on the page, use a link.

- Don't use "above" or "below." These are visual-based terms that don't align with usability recommendations.
- Search the document for "above" and "below" and replace them.

### Off-Page References

Off-page references can target sections, graphics, tables etc. Almost all off-page references are to supplementary information. The notable exceptions are references to glossary terms, actions, data types, and similar structured content. For more information, see the following section on links to structured content. For all other off-page references, use the following guidelines.

## References to Supplementary Information

---

Scan the document for all links and apply the following standards.

- Don't use "here" or "click here" to indicate a link.
- Search the document for "here" and replace with text that describes the target of the link.
- To refer to a specific topic such as widgets, use the literal phrase, "For more information about widgets, see xyz," where xyz is the link to the target that is directly related to the discussion.
  - When the subject of the target is unambiguous, use the phrase "For more information see xyz." This reduces translation costs still further.
  - To increase conformity and decrease translation costs, don't invert to "See xyz for more information about widgets"
  - Always use the word "see" for internal links.
  - Don't replace it with "refer to," "consult," "viz." or anything else.
  - When casually referring to a target within a sentence, use parentheses. Otherwise, make it an independent sentence.
- Use the exact text of headings for internal cross-links.

Don't paraphrase. If you use the method `<xref href="target.dita"/>`, the correct title is automatically drawn from the target.

- If further information is related to the entire section rather than a specific point, put it in a Related Topics list.
- Don't use admonitions just to add a link that doesn't fit in a paragraph; put it in the Related Topics list.

## References to Structured Information

---

If the information is directly integral to your subject matter, such as a glossary term, action, or data type, you can simply use linked text for your reference.

### Glossary Terms

Directly link to the other page without explanation if the link connects to a definition, or is a part of a list or table of references.

Link to terms defined in the glossary the first time the term appears on each HTML page. For instructions on working with glossaries, go to ZonBook Tips.

### API Actions and Data Types

Link to an action/function in the API reference only the first time the term appears on each HTML page. (Caveat: extremely long pages can have more than one link.) Like a glossary reference, no explanatory text is required.

### SDK and Command Line Tool Links

Link to a command or method only the first time the term appears on each HTML page. Like a glossary reference, no explanatory text is required.

## Topic and Related Topic Lists

---

The goal of this standard is to improve the reader's experience through better navigation and greater consistency across the documentation for different services. In principle, any section that has subsections should identify and link to its subsections.

### Topic Lists

Documents built according to the Microsoft Manual of Style for Technical Publications (MSTP)—even though we're not following it anymore— all follow the principle of transparent navigation. Note that they all present simple topic lists without commentary or explanation.

We follow the defining principles of consistency and simplicity. That is, always doing the same thing in the simplest way. This section begins with a topics list, which is automatically generated from the structure of the ditamap.

The list automatically adds a bulleted list of topic links with no description. If description is truly needed, add it in the following paragraphs or the intro of the relevant subsection.

### Related Topics

Use the following format and markup for related topic lists. These are placed at the very end of a section to direct the reader to parallel information that hasn't been specifically reference in the preceding text.

```
<related-links>
  <link href="related_link01.dita" />
  <link href="related_link02.dita" />
</related-links>
```

### Topic List Standards

- The topics list must be the very first element in a section; it precedes even the introductory text
- The related topics list must be the very last element in a section
- Use the correct XML markup, as shown in the previous examples
- This will assure correct content, formatting, and punctuation
- Provide a topics list whenever a section has two or more subsections on separate HTML pages
- Provide a topics list when a section has two or more subsections over several screenfuls on a single HTML page
- Add a related topics list whenever there are other sections in the same guide that could be helpful, but that aren't directly linked in the body.

## Windows and menus

---

When describing a navigation through a series of windows or menus, follow these guidelines.

- For navigation through a series of menus, use one of the following methods to describe it—depending on the UI behavior:
  - From the File menu, click History, click Last 7 Days, and then make your selection.
  - On the File menu, point to History, Last 7 Days, and then make your selection.



**Note:** Don't use arrows or slashes for menu navigation.

- For navigation through a series of windows, use one of the following methods to describe it:
  - From the Windows Control Panel, open the Administrative Tools and Local Security Settings, and then from the Action menu, select the Export List.

- From the Windows Control Panel, go to Administrative Tools > Local Security Settings, and then from the Action menu, select the Export List.



**Note:** Don't take readers through the Windows Start menu unless it is absolutely necessary. Different versions of Windows have different navigations through Start and users know how to find basics such as the programs list and the control panel.

Don't use other types of arrows or slashes for window navigation. Only use the simple right arrow bracket (>) as the shortcut notation for navigating through multiple menu layers.



---

# Chapter

# 8

---

## Formatting Content

---

### Topics:

- [Examples](#)
- [Graphics](#)
- [Tables](#)

This section covers standards, methods, and guidelines for writing common document elements.

## Examples

---

The difference between a formal and an informal example is a matter of markup and the presence of a title for the example.

### Formal Examples

The formal example is set off from the rest of the text with a title.

Following is the markup for a formal example.

```
<example>
  <title>ANT</title>
  <p>You can add brief, optional text here to clarify the following example.</p>
  <codeblock>$ant html</codeblock>
</example>
```

The text to the title gives the following result.

#### ANT

You can add brief, optional text here to clarify the following example.

```
$ant html
```

### Informal Example

Often you don't want to break up the flow of the reading with a new heading. In these cases, you use an informal example.

Following is an informal example.

```
<example>
  <codeblock>%finger batman</codeblock>
</example>
```

This markup produces the following output.

Following is an informal example.

```
%finger batman
```

## Example Standards

Follow these standards whenever you construct an example.

### Example Guidelines

- In general use with a heading, a formal example stands on its own and needs no lead-in sentence.
- When there is no heading, you must have a lead-in sentence.
- There are two styles you can follow for the lead-in:
  - Use a complete sentence ending in a period as you would to introduce a figure or a table.
  - End a sentence with the phrase for example: immediately before the example.



**Note:** When using the phrase for example, always close the phrase with a colon.

- Don't use the phrase for example: by itself.



- When giving a brief example within a sentence, use the phrase such as and use either the element `<i>` for a textual example or the element `<tt>` for a programming or data entry example.
- Don't interrupt a sentence with a block example and then continue it after the example.
- Examples within procedures don't need lead-in sentences.

## Right and Wrong Formatting

### Wrong

The phrase "for example" is used alone.

You can calculate a request signature to sign authenticated requests to AWS.

For example:

```
package amazon.webservices.common;
import java.security.SignatureException;
etc
```

.

This sentence doesn't really explain what the following example is or does.

You need a request signature to sign authenticated requests to AWS.

```
package amazon.webservices.common;
import java.security.SignatureException;
etc.
```

A block example is inserted into the middle of a sentence.

Opening and closing tags must be nested correctly, for example:

```
<p><note>Sentence</note></p>
```

is well formed whereas

```
<note><p>Sentence</note></p>
```

is not.

### Right

You can calculate a request signature to sign authenticated requests to AWS, for example:

```
package amazon.webservices.common;
import java.security.SignatureException;
etc.
```

The following code sample demonstrates how to calculate a request signature to sign authenticated requests to AWS.

```
package amazon.webservices.common;
import java.security.SignatureException;
etc.
```

Opening and closing tags must be nested correctly. The following code is well-formed.

```
<p><note>Sentence</note></p>
```

The next example shows incorrect nesting.

```
<note><p>Sentence</note></p>
```

## Graphics

This section provides a guiding standard for development and use of graphics. The result is a more professional look to our materials and increased branding of our products. This document defines graphics and provides general documentation guidelines.



**Note:** This standard is a living document. Specific guidelines for individual graphic types may be developed over time.

## Definition

A graphic is any non-textual representation used in the documentation. Examples are listed here:

- Charts
- Diagrams
- Drawings
- Graphs
- Icons
- Photos
- Screenshots

## Graphic Standards

This section describes the basic guidelines for producing Eucalyptus graphics.

In terms of *tools*, keep the following in mind:

- Don't use canned clip art (Visio/Powerpoint/etc.)
- Use the Eucalyptus core or UI color palette...if we have one.
- SnagIt is the default graphics tool; other tools are permitted

In terms of *format*, keep the following in mind:

- Don't use borders or frames to enclose graphics.
- Don't use shadows or shading unless approved by design team
- Place graphics in the <fig> element; this sets the white spacing, and suppresses captions and numbering

```
<fig scale="120">
  <image href="euca2ools.png">
    <alt>A screenshot of the Euca2ools menu.</alt>
  </image>
</fig>
```

In terms of *style*, keep the following in mind:

- Don't use background images or textures
- Don't use graphics to deliver core text
- Graphics should be sharp; text must be readable
- If the graphic includes callouts, follow the graphic with a legend
- Keep the graphics simple; they should be instructive at a glance
- No captions or titles

## Tables

---

This section describes the standards and guidelines for all in-text tables.

- Avoid putting admonitions in tables
- Any new use of graphics in tables must be submitted for developmental edit
- Don't add a table without explanation or integration with the text
- Don't end a sentence preceding a table with a colon
- Don't precede a table with a sentence fragment
- Don't use table headings or captions
- Don't use unusual formatting and shading
- Don't use table headings or captions.

- Don't use unusual formatting and shading. Use the existing DITA table elements in <table> and allow the style sheets to take care of formatting.
- Verify PDF output to ensure that columns have enough width to handle longer headings.
- In general, admonitions don't belong inside tables. This is part of using writing skill to integrate the table with the body text. There are good cases for exceptions, but they are exceptions that require editorial approval.
- Tables shouldn't be dropped into the body without explanation or integration with the text. Any table or graphic worthy of presenting is also worth discussing in the text. Lengthy reference tables might only need something like:

"The following table provides a list of the allowed tags and their associated attributes."

Tag	Attributes
a	accesskey   charset   coords   href   hreflang   name   rel   rev   shape   tabindex   target   type

Other tables should be treated as an exhibit that you need to explain to the reader. That is, the table is accompanied with a verbose description of how to read it. For example:

"The following table provides a list of tasks together with the command line tools and SOAP APIs you can use to accomplish them. A task might have multiple options. In these cases, the first tool or API listed is the most common and is recommended."

Task	Command Line Tool	SOAP API
List the rules belonging to specified groups	ec2-describe-groups	DescribeSecurityGroups



**Note:** Don't precede a table with a sentence fragment.  
Don't end a sentence preceding a table with a colon.



---

# Chapter 9

---

## Recognizing, Avoiding, and Resolving Ambiguities

---

### Topics:

- [\*Semantic Ambiguity\*](#)
- [\*Lexical Ambiguity\*](#)
- [\*Syntactic Ambiguity\*](#)
- [\*Summary\*](#)

This section discusses the three kinds of ambiguity found in written communication.

## Semantic Ambiguity

---

Semantic ambiguity happens when a pivotal word has multiple meanings that work in the sentence. You avoid semantic ambiguity by replacing the polysemic word with a more specific one.

### Example #1

*All instances based on the same image start out identical and any information on them is lost when the instances are terminated or fail.*

Here, "on" has multiple meanings. Several prepositions in English carry heavy semantic loads that can easily lead to misreadings. The best resolution here is, first to know that certain jargon phrases are loaded and to watch for them; secondly, rewrite the content using a different and unambiguous term.

### Example 2

*Origin and CallerReference may not be updated.*

Might not, or are not allowed to? There's no real way to know. The word "may" has multiple meanings, which only occasionally are clear in context. Rather than double-check your sentences for a clear context, just use a more specific word in the first place.

### Example 3

*DateTime indicates the date and time by which the image item is expected to be available for creating.*

By which usually means "by means of." Even though the context makes it clear, because the preferred reading fits the first half of the sentence the reader has to stop, adjust the semantic set and then finish the sentence. This sentence is a typical contortion intended to obey the false rule that forbids Latin speakers from ending a sentence in a "preposition." We aren't 1st millennium Romans, *ergo* we don't have to worry about the rule.

### Example 4

*While it is possible to delete an image, you can create a new image.*

You create the new image during the time that it is possible to delete an old image? "While" is primarily a temporal word. Only in its 3rd and 4th meanings does it become a synonym for although and because respectively. However, because the context often doesn't help, again, just use a more precise term

### Example 5

*Some VPN devices have the ability to override the DF flag and fragment packets unconditionally as required. We recommend the use of this option as appropriate.*

Flags are commonly considered options. So, do you use the DF flag only as appropriate—depending on your VPN device, or do you use the device's ability to override the flag as appropriate? Assuming that DF flag is a defined term, the solution is to clarify which option is the referent in the second sentence.

## Lexical Ambiguity

---

Lexical ambiguity happens when a single, unspecified word can have multiple referents. It is most commonly seen with pronouns.

### Example 1

*The key itself is the private key in a key pair.*

Which key is itself? This is a fairly simple case that can be resolved by describing the key more clearly, such as: “The User key in the previous example is the private key in a key pair.”

### Example 2

*The changes to your distribution's logging configuration take effect within 12 hours, but it may take less time.*

"It" should refer either to the configuration or the distribution, but not to "changes"--which is plural. The quick way out of this example is to drop the word it; the ambiguity then disappears.

### Example 3

*Those parameters include all of the response parameters, including status parameters, which tell you whether or not the sender successfully changed the payment instrument.*

What is the antecedent to "which," all the parameters, or just those for status? Assuming the antecedent is “status,” you can put the parenthetical subclause into parentheses, and remove the confusing comma. “...parameter, including the status parameters (which tell you whether ...).

This brings up an important side point. The more commas you have in a sentence, the more likely someone is going to misread it.

### Example 4

*In that case, you need to use the verification process described in the Appendix.*

"In that case" is a common phrase from certain non-native writers. It is used to mean "When the last thing I stated happens." Frequently it works, but just as frequently there are two or three "cases" and no indication which one is "that case." Unlike other languages, English has no different word for “this.” So we have to use a couple more words. For example, “When authentication fails, you need to use the verification process...”

### Example 5

*By default, network access is turned off to a new RDS DBSecurityGroup; you must specifically authorize access to an IP range for a new security group after it is created.*

What is "it," the IP range or the security group? It might be obvious if you stop and think about it...but the point is to make the writing clear so that the reader doesn't have to stop and think.

### Example 6

*Accept the default values for this example, then click Continue.*

Does "for this example" refer to default values that have been put into the example or merely the act of accepting the values? Rewritten for clarity you can see the two meanings: “For this

example, accept any default values, and then click Continue.” Or: “Accept the default values given in this example, and then click Continue.”

### Example 7

*Eucalyptus runs a web server and configures it on the front end machine when it installs the software.*

Does the second "it" refer to Eucalyptus, the web server, or the front end machine? The first "it" was clearly the web server. Again, you can stop and reason it out, but you shouldn't have to. Remember that no matter how fast we read, it's still left-to-right; so when you stack up a row of nouns, and then put in an unqualified pronoun, the reader naturally wants to make the nearest noun into the antecedent.

## Syntactic Ambiguity

---

Syntactical ambiguity happens when the structure of the sentence is contorted, or the sentence has too many parts to make correct parsing easy. Sentence reconstruction or splitting is the most common solution.

### Example 1

*Auto Scaling in this example is configured to scale out by 1 when the application's average CPU Utilization exceeds a threshold of 80% and scale in by 1 when it drops below 40% for 10 minutes.*

Which of the two conditions must happen for 10 minutes—or is it both of them? To resolve the ambiguity, you rewrite to put the ambiguous phrase where it cannot be confused. Here, we clarify both meanings: “In this example, AS is configured to scale whenever a threshold is breached for 10 consecutive minutes; it scales out when..., and it scales in when...” Or: “In this example, AS is configured to scale out...whenever the...exceeds the 80% threshold; it scales in when utilization drops below a 40% threshold for 10 consecutive minutes.”

### Example 2

*Typically, you want your application to check whether or not a request generated an error before spending any time processing results.*

Who or what spends the time: you or the application? This is easily resolved by changing the key phrase “...before spending any time processing results....” “Before your application spends any time processing results....”

### Example 3

*Streaming differs from HTTP download in that the end user doesn't use the media until it's been fully downloaded to the local system over an HTTP connection.*

Which method doesn't use the media immediately: streaming or download? You have two clauses here, but as written, it isn't clear how to attach the dependent clause. “Streaming differs from HTTP download, which doesn't allow the user....” “Streaming differs from HTTP download, because streaming....”

### Example 4

(This one is from Scot's past. He cringes but offers it as a warning not to do as he does.)



*The CreateInboundShipmentPlan operation returns the information required to make a set of shipments for a given set of items and the ship from address.*

Does the ship from address relate to the set of items, the set of shipments, or is it returned with the information you need to create the set? Moving the highlighted text and reinstating deleted syntactic details clarifies the sentence: “The...operation returns the *ship from* address and the information needed...” “The...operation returns the information needed to create both the *ship from* address and a set...”

### Example 5

(Here's another from Scot's greatest hits.)

*A topic is a communication channel to send messages and subscribe to notifications, providing an access point for publishers and subscribers to communicate with each other.*

Does the subscription to the communication channel (topic) provide the access, do the notifications, or is it the topic? This is a classic dangling modifier. Break the sentence to correct. We have to assume here that the topic is the referent. So: “A topic is a communication channel to send messages and subscribe to notifications. It provides an access point for publishers and subscribers to communicate with each other.”

### Example 6

*Typically, however, you send them to the locale in which your customers reside for a variety of reasons, including reducing shipping costs.*

Customers generally do reside in different locales for a variety of reasons. Do they live where they do to reduce shipping costs? The modifier is too far away from its referent. Move it: “Typically however, you send them for a variety of reasons (such as reducing shipping costs) to the locale where your customers reside.”

### Example 7

*Item names are selected if any of the values match the predicate condition. To change this behavior, only use the every() operator to return results where every attribute matches the query expression.*

Does this mean only the every() operator should be used, or don't use the every() operator for anything else? The word “only” restricts both nouns and verbs, therefore you have to make its scope clear when you use it: “To change this behavior, only use the...operator when you want to return results that match every attribute to...” Or: “To change this behavior, use the...operator when you only want to return results that...”

## Summary

Anytime you break out from the simple subject-verb-object (SVO) order, you raise your chances of having an ambiguity.

You can avoid most ambiguities by:

- Keeping your sentences simple
- Clearly defining referents and antecedents
- Watching how you use polysemic words

When you must have more complex sentences remember that we read left-to-right and keep related words and phrases close together if you are referring between them.



---

# Chapter 10

---

## Punctuation

---

Punctuation is "a courtesy designed to help readers to understand a story without stumbling." (From a national British newspaper's style guide, quoted in *Eats, Shoots & Leaves*, by Lynne Truss.)



---

# Chapter 11

---

## Editing

---

Here are some thought processes and decisions for editing content<sup>1</sup>:

**Determine readability and suitability for audience.**

Is the audience technical? Do they speak English as their first language? What is their level of education? Most times, we can't know much about the audience. So, keep the language as plain and simple as possible, without 'talking down' to the readers or patronizing them. Clear, simple, unambiguous language is good for everyone.

**Match tenses.**

Watch for paragraphs that mix up past and future tenses ('be completed', 'to validate', 'be used'). Aim to make all the tenses the same, whether that was past, present or future. Keep consistent with the organization's style guide.

**Use parallel structure.**

It's not as apparent as in a bulleted list, but the structure of the paragraphs is made clear often by using parallelism. Especially with complicated ideas, make sure that subjects take actions on objects. This helps the reader gain clarity.

**Make sense.**

Critical to the sense of a sentence or paragraph is the 'who, what, when, where, why, and/or how'. Not every sentence or paragraph will have all these, but all sentences will have some of them. Ask 'Who is doing what?' (and perhaps 'to whom'), and 'Why are they doing it?' Also ask 'How is it to be done?'

**Pay attention to passive/active voice.**

Similar to the 'use parallel structure' above, make sure that subjects act upon objects. Reword all sentences in which objects are being acted upon by subjects.

---

<sup>1</sup> from <http://cybertext.wordpress.com/2011/05/20/editing-and-rewriting/>

**Remove redundancies.**

Take out all words in a paragraph that can be removed without altering meaning. For example, ‘for each of the units within this particular are’ can be shortened to ‘for each unit within this area’ or ‘for each unit in this area’, or perhaps ‘for this area’s units’. Another: ‘as the basis of forming a portfolio of evidence of competency for this area’ can be shortened to ‘the basis of an evidence portfolio of competency in this area’, or ‘the basis of a competency evidence portfolio’, though that may be getting away from the meaning a little and I’d have to check it with the writer.

---

# Chapter


# 12

---

## Terms

---

Concept definition.

<b>above</b>	<p>Use only for physical space or screen descriptions, e.g., "the outlet above the floor," or "the foo button above the bar pane."</p> <p>For orientation within a document use previous, preceding, or earlier.</p>
<b>acronyms</b>	<p>Spell uncommon terms out at first use. For example: Department of Homeland Security (DHS)</p> <p>Common acronyms (such as SOAP, RSA, WWW) don't need spelling out.</p> <p>Don't use an apostrophe to make an acronym plural. That is, use ASPs, not ASP's. action</p> <p>Capitalize as shown. Depending on the product, this can also be an operation or a function. Use the same term consistently within each product.</p>
<b>affect vs. effect</b>	<p>An affect is an influence; an effect is a result. If you affect the outcome, you modify it. If you effect the outcome, you cause it.</p>
<b>allow</b>	<p>Use exclusively to refer to permissions.</p>
<b>AM</b>	<p>Ante Meridian. Caps, no periods. Use only if absolutely necessary for times in local time zones between midnight and noon. For all other purposes, use UTC times.</p> <div><p><b>Note:</b> Note that midnight and noon are on the meridians and are neither AM nor PM.</p></div>

<b>Ampersand (&amp;)</b>	<p>In body text, use exclusively when quoting a reference or resource that requires the symbol. For example: AT&amp;T.</p> <p>Don't use to shorten any phrase that has the word "and" in it.</p>
<b>Amazon Web Services</b>	<p>All in caps, all the time. Use "AWS" only after first defining the acronym and when using "Amazon Web Services" seems like overuse. Do not use the capitalized form to refer to more than one AWS product. See also web service.</p>
<b>Amazon Web Services account</b>	<p>Used to describe what developers need in order to use the web services or sign up for other web services.</p>
<b>Amazon Web Services website</b>	<p>Used to describe all of the web properties under <a href="http://aws.amazon.com">http://aws.amazon.com</a>.</p> <p>Example: See the <a href="#">Amazon Web Services website</a> for the complete licensing restrictions.</p>
<b>American vs. British and other English dialects</b>	<p>There are a number of differences in spelling and grammar between American and other English dialects. Our formal, world-wide publications use American style. If a publication is written specifically for distribution to a local market, the stylistic rules of that market apply. That is, if an email is targeted at South Africa, use the spelling and grammar conventions of that region.</p> <p>When in doubt use the Merriam-Webster Dictionary for correct spellings. Always use the first spelling given; this is the preferred spelling in American English.</p>
<b>and/or</b>	<p>Although popular, the phrase is ambiguous and unnecessary in technical writing. Use either "both...and" or "either...or," as the situation requires.</p>
<b>app vs. application</b>	<p>Using "app" is acceptable when talking about a software application.</p>
<b>appear, display, and open</b>	<p>Messages and pop-up boxes appear. Windows, pages, and applications open. The verb display requires a definite object. For example: The system displays the error message.</p>



<b>appendix</b>	In technical documentation, use the plural appendices. Use appendixes when referring to the veriform appendage in the human anatomy.
<b>as</b>	<p>As is primarily an adverb. Use it in constructions like twice as long or Storage Controller as opposed to Walrus.</p> <p>Don't use it as a conjunction in place of because. This is frequently ambiguous and slows the reader down to parse your meaning.</p>
<b>as well as</b>	Ambiguous. Use in addition to or and as appropriate.
<b>backward compatible</b>	No S.
<b>base64</b>	<p>Standard usage for quadrosexagesimal (cool word, huh?) is the format base64. Note capitalization and spacing. There is no convention on hyphenation, so avoid it.</p> <p>Example: Use a base64 encoded URL. You must use base64 encoding.</p>
<b>below</b>	Use only for physical space or screen descriptions, e.g., "the outlet below the vent," or "the foo button below the bar pane." For orientation within a document use following or later.
<b>below the fold</b>	Jargon. Don't use in technical documentation.
<b>beta</b>	<p>Label products in beta as such. Use "(Beta)" in product titles, navigational categories, or other places where it's necessary to promote that a product is in a beta period. For example: Eucalyptus Enterprise Edition 4.0 (Beta). Placing the beta label in titles provides a first mention notation for beta products, so it is not necessary to always use the notation in content.</p>
<b>bits per second</b>	Use bps
<b>bold</b>	<p>Use only as an adjective, such as "The characters in bold indicate..."</p> <p>Don't use boldface.</p>
<b>Boolean</b>	<p>Always capitalized: a Boolean value</p> <p>Values for Booleans are true and false. These values appear in web</p>

	<p>service messages as these words, in lowercase. All references to Boolean values in Eucalyptus technical documentation should be marked <code>true</code> and <code>false</code>, with that spelling and casing, including at the beginning of sentences. Avoid using a Boolean value name at the beginning of a sentence or sentence fragment.</p>
<b>bookmark</b>	<p>Industry standard term for a saved link. Don't use favorite.</p>
<b>bottom</b>	<p>Use only as a general screen reference, such as, "scroll to the bottom of the page." Don't use for window, page, or pane references to features or controls; use <i>lower</i> instead. For example: "Click the button on the lower left."</p>
<b>browse</b>	<p>Use when referring to scanning information. Don't use when describing how to navigate to a particular item on our site or a computer.</p> <p>Correct: "Navigate to the Eucalyptus Console."</p> <p>Incorrect: "Browse to the Eucalyptus Console."</p>
<b>build</b>	<p>Only use to mean compile and link code. It is also acceptable as a noun in programming documentation.</p>
<b>capitalization</b>	<p>This topic comes up often, which is why it's included here. Note the specific use of capitalization in the words on this list. In general, use title capitalization for heading titles.</p>
<b>cancel, canceled, canceling</b>	<p>The spelling with two <i>els</i> is British in origin and, although a recognized alternative, is non-standard for American English.</p> <p>Exception: <i>cancellation</i> is correct.</p>
<b>comma usage</b>	<p>Use the serial comma in sentences that contain lists of three or more items.</p> <p>Example: "...configure, install, and test."</p> <p>For more information about comma usage see the extended comma usage guidelines.</p>

<b>contractions</b>	Contractions such as you're, it's, can't, or don't are easier for the reader to parse and avoid confusion. These forms are preferred. For more discussion and examples, go to To Contract or not to Contract.
<b>copyrights</b>	Don't use the copyright symbol in Eucalyptus technical publications. We note copyrights for Eucalyptus on the copyright page.
<b>desire</b>	See wish.
<b>e.g.</b>	exempli gratis. Frequently confused with i.e. Use to mean for example.
<b>email</b>	One word, no hyphen, no plural.
<b>future tense</b>	Use present tense; avoid needless use of future tense. It only adds to the wordiness of a sentence and is more difficult to translate. Use will for clear cases of future effect.
<b>GMT</b>	Greenwich Mean Time (archaic). Use UTC instead.
<b>Hadoop</b>	Capitalize as shown.
<b>heading formats</b>	All headings use title case e.g. Error and Trace Reports (refer to the Chicago Manual of Style for detailed instructions on title case).  Exception: See Headings Cascade for Procedural Sections for guidelines on heading styles in procedure guides.
<b>HTTP</b>	Hypertext Transfer Protocol. Capitalize as shown.
<b>HTTPS</b>	Hypertext Transfer Protocol Secure. Capitalize as shown.
<b>IaaS</b>	Capitalization as shown. See Infrastructure as a Service.
<b>ID</b>	ID is the preferred capitalization as opposed to Id. To reduce a reader's confusion, the writer might avoid using the abbreviation ID and use instead a word equivalent, such as identity or token or identifier.
<b>i.e.</b>	id est. Frequently confused with e.g. Use to mean that is.
<b>Infrastructure as a Service</b>	Capitalization as shown. Abbreviated as IaaS.
<b>Internet</b>	Internet is always capitalized when referring to the world-wide TCP/IP inter-network. Describe other inter-

	networks, while technically internets, without using the word internet since it is confusing to readers.
<b>key pair</b>	Two words.
<b>latinisms</b>	Common terms that are widely known, like i.e., e.g., per, and via, are allowed. Use them correctly!
<b>may</b>	The word may is ambiguous in English. In some contexts it means can, in others allowed to or should, and in still others might. Use one of the unambiguous terms instead.
<b>metadata</b>	One word.
<b>must/shall/should</b>	Must and shall refer to requirements. If the reader doesn't follow the instruction, something won't work right.  Should is used with recommendations. If the reader doesn't follow the instruction, it might be harder or slower, but it'll work.
<b>namespace</b>	One word.
<b>PaaS</b>	Capitalization as shown. See Platform as a Service.
<b>parentheses</b>	Use parentheses (or parens for short) to add tangentially related information (such as an example or commentary) to a sentence. You can set off information that is more closely related to the topic of the sentence with commas or em-dashes.  Punctuation: If the parentheses are within a sentence, place all punctuation (commas, periods etc.) outside the parentheses.  If the parentheses stand alone outside any other sentence, all punctuation belongs inside the parens.
<b>passives</b>	See voice.
<b>per</b>	Use in the meaning of for each, such as miles per hour. Don't use to mean "according to, as in according to RFC 2142.
<b>Platform as a Service</b>	Capitalization as shown. Abbreviated as PaaS.
<b>PM</b>	Post Meridian. Caps, no periods. Use only if absolutely necessary for times

	<p>in local time zones between noon and midnight. For all other purposes, use UTC times.</p> <p>Note that midnight and noon are on the meridians and are neither AM nor PM.</p>
<b>pronouns</b>	<p>Always verify the antecedents. Is it clear who is doing what?</p>
<b>quotation marks</b>	<p>Usage: Use quotes for actually quoted words or phrases from other text sources. Don't use quotes to call out special, or unfamiliar, words or phrases; use italics instead.</p> <p>Punctuation: Follow American instead of Oxford style: Place periods and commas inside quotation marks, but place question and exclamation marks outside the quotes—unless they are a part of the quotation.</p>
<b>referer</b>	<p>Specifically for the HTTP referer. Originally misspelled in the specification, this is now the generally accepted spelling for this one case.</p> <p>Always spell as <b>referrer</b> for all other uses.</p>
<b>REST</b>	<p>Representational State Transfer. Always abbreviated, always capitalized. Generally used in discussions to contrast with SOAP.</p> <p>If the web service uses both query- and resource-based REST calls, keep the distinction clear using the terms REST and REST-Query.</p>
<b>SaaS</b>	<p>Capitalization as shown. See Software as a Service.</p>
<b>SSH</b>	<p>Secure SHell. When discussing the concept or technology, the initials are capitalized. Only when discussing or using the command ssh are they written in lowercase.</p>
<b>should</b>	<p>Indicates a recommendation.</p>
<b>since</b>	<p>Use only to describe time events. For example: "Getting datasets into Walrus is easy, ever since AWS Import/Export launched." Don't use in place of because.</p>

<b>SOAP</b>	Simple Object Access Protocol. Always abbreviated, always capitalized.
<b>Software as a Service</b>	Capitalization as shown. Abbreviated as SaaS.
<b>spam</b>	Mass noun, no plural. For example, "How much spam do you get?" Not, "How many..."
<b>start-up</b>	A fledgling business enterprise.  Start up is a slang verb form, as in "I want to start my own business up." The word up is deleted in formal writing.  Startup is a small town in Snohomish County, Washington.  (source: Merriam-Webster)
<b>subject/pronoun number agreement</b>	A pronoun should always agree in number with the antecedent noun it refers to. That is, a plural noun such as customers takes the plural pronoun they; a singular noun like Internet takes the singular pronoun it. Use of they or them to refer to a singular noun of indeterminate sex should be avoided. You can usually accomplish this by changing the noun to a plural. In other cases, you can rewrite a sentence to avoid the need for a pronoun altogether.
<b>they, them, themselves</b>	See subject-pronoun number agreement.
<b>user</b>	In most cases replace with the more direct form you. Reserve user for cases where you are referring to a third party (not the audience you are writing for).
<b>UTC</b>	Universal Coordinated Time, aka Zulu time (after the Z that is often added after UTC time descriptions). Use this instead of GMT when describing times and time zones.
<b>voice</b>	Whenever possible use the active voice instead of passive. Passives are wordier and often cause the writer to obscure the details of the action. For example, change the agentless passive it is recommended to the more direct we recommend.  The voice and tone of Eucalyptus technical publications is direct and

**Web vs. web**

friendly. We speak directly to the reader as you and refer to Eucalyptus as we. The overall tone is informal, informative, and friendly without getting chatty.

Always use web. We do this not only to avoid confusion around capitalizing things like web service, but also to match the progressive writing styles of other technology sites (such as Wired, and BBC).

**web service**

Note capitalization. Use web services to describe more than one. Otherwise, where it's important to note the distinction, use Eucalyptus offerings, Eucalyptus products, or web services offered by Eucalyptus.

When referring to AWS, don't use the redundant AWS web services.

**website**

One word, lower-case.

**while vs. although or whereas**

Restrict while to its meaning of "during an interval of time." Don't use it as a synonym for although because it is often ambiguous. Whereas is a better alternative to although in many cases, but it can sound overly formal.

**White list, whitelisting**

Avoid whenever possible. Use the alternate terms approved or verified. Although white list has many innocent uses and has been largely sanitized, it still carries its original, highly-discriminatory meaning for some readers.

**will**

Avoid needless use of future tense. It only adds to the wordiness of a sentence and is more difficult to translate. Use will for clear cases of future effect.

Example: If you delete your account, you will lose all your stored data.

**Wish/want/desire vs. need**

Wish and desire are weak versions of want; don't use them. Do not confuse wants with needs. Use the term that is appropriate to the situation. Need connotes a requirement or obligation; want indicates that you have an intent, but still a choice of valid actions.

