

CHAPTER 3

A Practical Approach to FOSI

The author contends the mechanisms described in the previous chapter for specifying and resolving FOSI formatting are a little too complicated for comfort, and instead advocates a more practical, streamlined approach. This approach utilizes a subset of FOSI capabilities with no loss of formatting functionality. The result is that FOSI development is faster and easier. This proven methodology has been used successfully by many FOSI developers for almost twenty years.

This is not to say that other methods do not work. There is every reason to believe they do. However, such FOSIs may very well be more difficult to develop and maintain. For example, see **Figure 32 Theoretical FOSI** on page 58, which is designed to show the complexity that can result when the “practical” approach is not followed

NOTE: With the exception of **Figure 32**, the code examples in this book and the FOSI tutorials adhere to the author's “practical FOSI” methodology.

TIP 

Specific information on **Coding a FOSI** can be found on page 694.

Practical FOSI design principles

Code as little as possible

It is not desirable to code every characteristic in every `e-i-c` — that makes a FOSI difficult to maintain. For ease of FOSI development and maintenance, utilize inheritance/defaulting capabilities and FOSI parameterization as much as possible.

Do not use environments

Don't use environments because they just add complexity. Use:

- `docdesc` for the primary formatting used in the document
- `charlist` for element-specific coding
- `charsubset` for `charlist` coding that is shared by multiple `e-i-cs`
- `text entities` for `attspec` coding that is shared by multiple attribute rules
- `file entities` for `e-i-cs` and other FOSI components that are the same in multiple FOSIs

Always enable `charlist` and category inheritance


As **Figure 19 Charlist and category inherit attribute matrix** on page 43 shows, the possible combinations for `charlist` and category inheritance create a lot of complexity. Fortunately, it doesn't have to be that complicated.


The Practical FOSI approach is to always code `inherit="1"` for all `charlists` and all inheriting categories unless there is a specific reason not to. When `charlist` and category inheritance are both enabled, the shaded areas in **Figure 29** no longer apply. There is only one possibility instead of nine, which makes it a great deal easier to evaluate `e-i-cs` in a FOSI.

Figure 29 Simplified charlist and category inherit attribute matrix

		Value of category's inherit attribute		
		1	0	unspecified [†]
Value of charlist's inherit attribute	1	1 category inherits	0 category defaults	1 category inherits
	0	1 category inherits	0 category defaults	0 category defaults
	unspecified [‡]	1 category inherits	0 category defaults	0 category defaults

[†]# IMPLIED in OutSpec DTD
[‡]0 (zero) in OutSpec DTD

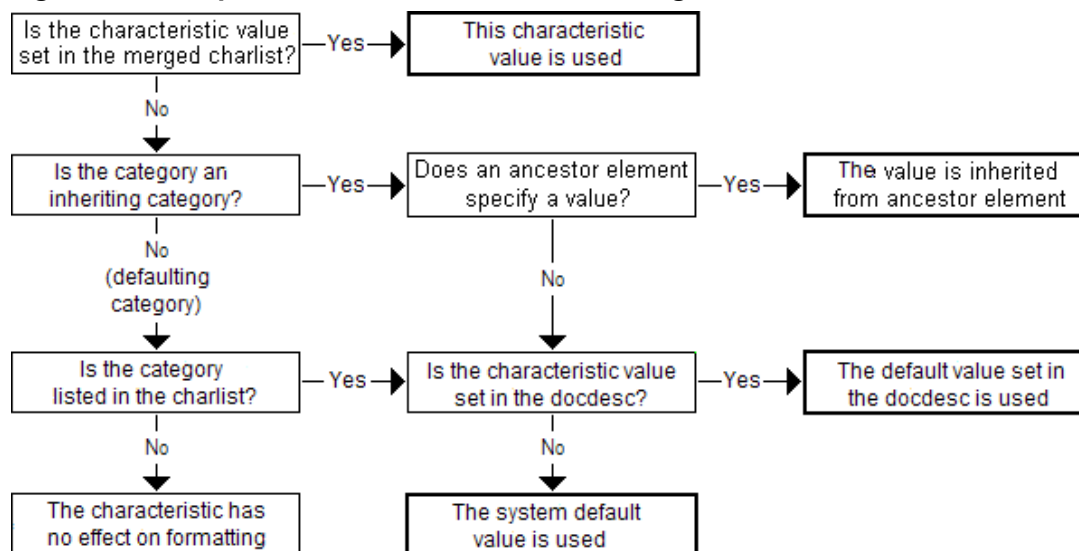
FOSI TIP 
 When inheritance is disabled, add a comment to the e-i-c describing the reason.

FOSI TIP 
 It may be desirable to disable category inheritance, for example, to avoid having the contents of the table cell inherit any prevailing indent. See **Figure 194 Indent in table cells** on page 354.

In addition, the practical FOSI inheritance and defaulting flowchart shown in **Figure 30** is much simpler than the flowchart on **Figure 20** on page 45.

FOSI TIP 

Enabling `charlist` and category inheritance when an `e-i-c` is created is faster and easier than adding the code at a later time.

Figure 30 Simplified inheritance and defaulting flowchart

Practical FOSI examples

Enabling `charlist` and category inheritance works well in practice. For one thing, it is very useful to have inheritance in place in case it is needed. For example, if you are asked to change a FOSI to make everything in `<intro>` italic, the change is as simple as adding the `italic` charsubset to the `e-i-c` for `<intro>` (as shown in **Figure 31**) — *if* `charlist` and category inheritance are already in place. If not, then every `e-i-c` in context of `<intro>` will need to be edited.

Figure 31 Context-specific inherited font posture

```

<docdesc>
<font posture="upright" ...>
...
<e-i-c gi="body">
<charlist inherit="1 charsubsetref="section">
...
<e-i-c gi="intro">
<charlist inherit="1 charsubsetref="section italic">
...
  
```

To demonstrate the complexity that can result with “theoretical FOSI” coding, the FOSI fragment shown in **Figure 32** utilizes all the following mechanisms.

- `charlist` coding
- referencing `charsubsets`
- `charlist` inheritance
- category inheritance
- defaulting to an `envdesc`
- defaulting to the `docdesc`

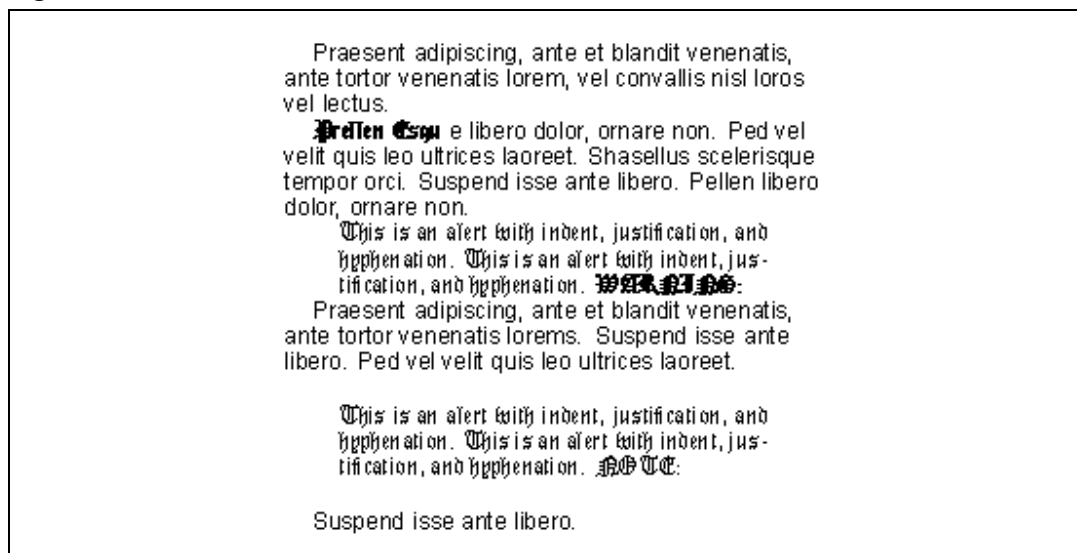
This example demonstrates several things:

- Why inline elements should inherit formatting. Notice what happens when the `<bold>` inline element does not.
- The difference between an `envdesc` and a `charsubset`: a `charsubset` specifies an explicit value; an `envdesc` provides a default value in case one is needed.
- The difference between inheriting and non-inheriting categories: a defaulting category must be present in the merged `charlist` to have any effect; an inheriting category has an effect even when it is not specified in the merged `charlist`.
- How different combinations of `charlist` and category inheritances require each `e-i-c` to be carefully evaluated in order to determine the resolved formatting environment. This is a major issue when it is necessary to make changes to a FOSI. Without a complete understanding of the resolved value of formatting characteristics, a simple change can have serious unintended consequences.

NOTE: This FOSI is constructed to show the complexity of this approach, and may not reflect actual usage. For example, the Old English font in this example is not appropriate for a default setting but is used so it stands out.

NOTE: This example uses the `charsubsets` defined in **Figure 7 Useful charsubsets** on page 30.

Figure 32 Theoretical FOSI

**XML fragment**

```
<section>
<para>Praesent adipiscing...</para>
<para><bold>Prellen Esqu</bold> e libero...</para>
<warning>This is an alert with indent, justification, and hyphenation.
This is an alert with indent, justification, and hyphenation.
</warning>
<para>Praesent adipiscing...</para>
<note>This is an alert with indent, justification, and hyphenation. This
is an alert with indent, justification, and hyphenation.
</note>
<para>Suspend isse ante libero.</para>
...
```

FOSI fragment

```
<charsubset charsubsetid="block">
<textbrk startln="1" endln="1">
</charsubset>
<charsubset charsubsetid="bold">
<font weight="bold">
</charsubset>
<docdesc>
<charlist>
<font famname="Old English Text MT" ...>
...
<hyphen hyph="0">
<indent leftind="0pt" rightind="0pt" firstln="*">
```

```

<quadding quad="left" lastquad="relative">
<highlt scoring="0" scorewt="0.5pt" allcap="0" scorespc="1">
<presp minimum="0pt" nominal="0pt" maximum="0pt" condit="discard"
priority="med">
<postsp minimum="0pt" nominal="0pt" maximum="0pt" condit="discard"
priority="high">
...
<envdesc envid="alert.env">
<charlist>
<hyphen hyph="1">
<indent leftind="@+2pi" rightind="@+2pi" firstln="*">
<quadding quad="justify">
<presp minimum="1pi" nominal="1pi" maximum="1pi" priority="med">
<postsp minimum="1pi" nominal="1pi" maximum="1pi" priority="high">
<usetext placemnt="after"></usetext>
</charlist>
</envdesc>
<e-i-c gi="bold">
<charlist inherit="0" charsubsetref="bold"></charlist>
</e-i-c>
<e-i-c gi="note">
<charlist envname="alert.env" inherit="0" charsubsetref="block">
<presp>
<postsp>
<usetext source="\NOTE: \" placemnt="before"></usetext>
...
<e-i-c gi="para" context="section">
<charlist inherit="1" charsubsetref="block">
<indent inherit="1" firstln="*+1pi">
...
<e-i-c gi="warning">
<charlist envname="alert.env" inherit="0" charsubsetref="block
keep-together">
<usetext source="\WARNING: \" placemnt="before">
<subchars charsubsetref="bold"></subchars>
</usetext>
...
<e-i-c gi="section">
<charlist inherit="1">
<font famname="Arial">
...

```

At first glance, the FOSI code appears to be correct, but the incorrect formatted output raises a lot of debugging questions.

Q: Why does the Old English font appear in some places?

A: Text at the beginning of the second paragraph uses the wrong font because the font category in the bold charsubset does not specify category

inheritance. Consequently, the `font famname` defaults to the `docdesc`, which specifies the Old English font.

`<note>` and `<warning>` appear in Old English font because their `e-i-cs` turn off `charlist` inheritance. As a result, the document description default value is used.

The `e-i-cs` for `<para>`, on the other hand, enables `charlist` inheritance, so the `font family` is inherited from the `e-i-c` for `<section>`, which specifies a different `font family` than the `docdesc`.

Q: Why do “Note: “ and “Warning: “ appear at the end of their respective elements instead of at the beginning of them.

A: The `placemnt` characteristic is not specified for the `usetexts` that output “Note: ” and “Warning: ”. Since `usetext` is a defaulting category, the default value specified in the `alert.env` environment (“after”) is used.

Q: Why is the vertical space missing before and after `<warning>`?

A: It seems like the vertical space should be there because `<warning>` references the `alert.env` environment, which contains `presp` and `postsp` categories with all characteristics specified. `<note>` also references this `envdesc`, and it is formatted correctly. The difference is that `<note>` includes empty `presp` and `postsp` categories and `<warning>` does not. `Presp` and `postsp` are defaulting categories that have no effect unless they are present in the merged `charlist`, in which case a default value is sought. If an `envdesc` is referenced that contains values for the missing characteristics, they are used. If not, document or system default values are used.

Q: The `hyphen`, `indent`, and `quadding` categories in the `alert.env` environment are applied to `<note>` and `<warning>` even though those categories are not in their `charlists` — why?

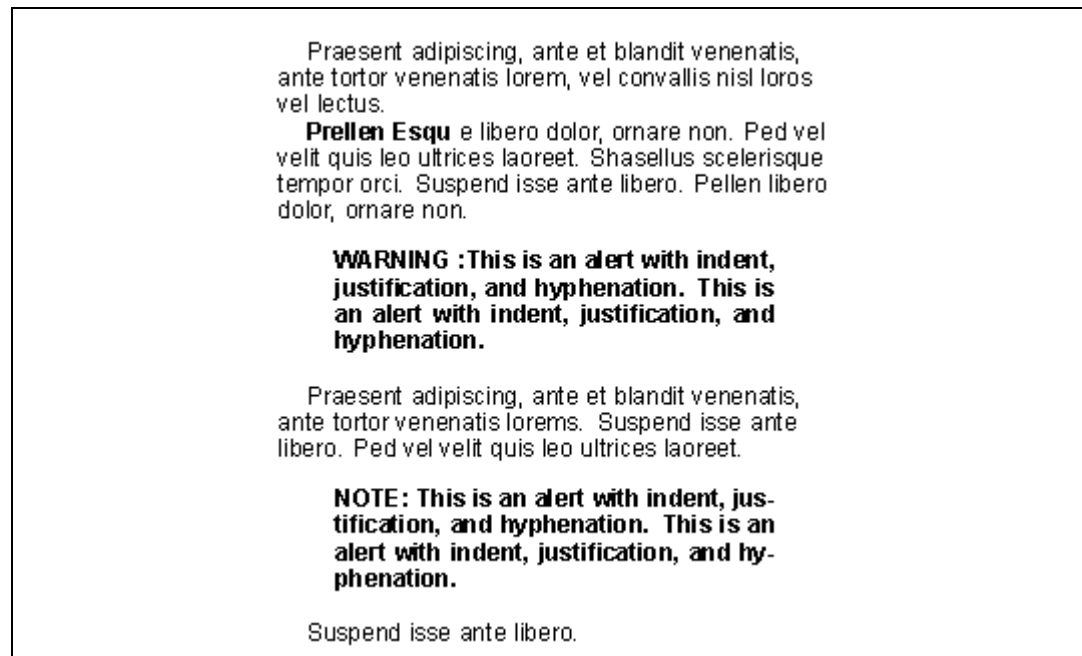
A: `Hyphen`, `indent`, and `quadding` are inheriting categories, which have an effect even when they are not present in the `charlist`. When the `inherit` attribute on these categories is set to 1, any missing values are inherited from an ancestor element, or, if no value is available, from the document description or ultimately the system default. When the `inherit` attribute on inheriting categories is set to 0, missing categories default. If the `inherit` attribute on inheriting categories is left unspecified, then the value of the `charlist inherit` attribute is used. If the `charlist inherit` attribute is missing, the DTD default value of 0 (zero) is assumed.

Fortunately, FOSI coding doesn't have to be this complicated.

In the following figure, a “practical FOSI” outputs the correct formatting from the XML fragment shown in **Figure 32 Theoretical FOSI** on page 58. This FOSI does not use any `envdescs`; instead it relies on standard `charsubsets` that are included in most FOSIs (shown in **Figure 7 Useful charsubsets** on page 30), plus one additional `charsubset`, `alert`. In addition, `charlist` and category inheritance are both enabled so a chain of inheritance is in place when it is needed, such as inheriting the `font famname` from the `e-i-c` for `<section>`.

With this approach, it is much easier to determine the value of an unspecified characteristic. In particular, the various possibilities for `charlist` and category inheritance are reduced to a single case: both inherit missing characteristics from an ancestor element or ultimately, the document or system default.

Figure 33 Practical FOSI



```
<charsubset charsubsetid="alert" charsubsetref="block
prespace postspace hyphen-on bold">
<indent inherit="1" leftind="@+2pi" rightind="@+2pi" firstln="*">
<quadding inherit="1" quad="justify">
</charsubset>
<docdesc>
```

```
<charlist>
<font famname="Old English Text MT" ...>
...
<hyphen hyph="0">
<indent leftind="0pt" rightind="0pt" firstln="*">
<quadding quad="left" lastquad="relative">
<highlt scoring="0" scorewt="0.5pt" allcap="0" scorespc="1">
<presp minimum="0pt" nominal="0pt" maximum="0pt" condit="discard"
priority="med">
<postsp minimum="0pt" nominal="0pt" maximum="0pt" condit="discard"
priority="high">
...
</docdesc>
<e-i-c gi="bold">
<charlist inherit="1" charsubsetref="bold"></charlist>
</e-i-c>
<e-i-c gi="para" context="section">
<charlist inherit="1" charsubsetref="block">
<indent inherit="1" firstln="*+1pi">
...
<e-i-c gi="note">
<charlist inherit="1" charsubsetref="alert">
<usetext source="\NOTE: \" placemnt="before"></usetext>
...
<e-i-c gi="section">
<charlist inherit="1">
<font inherit="1" famname="Arial">
...
<e-i-c gi="warning">
<charlist inherit="1" charsubsetref="alert">
<usetext source="\WARNING:\" placemnt="before"></usetext>
...
```