



# Scrivere documenti per Firebird

Paul Vinkenoog

7 Maggio 2007 – Versione 1.3.1-it - versione italiana

Traduzione in italiano: Umberto Masotti

---

---

## Sommario

Introduzione .....	3
Scopo di questa guida .....	3
Conoscenze necessarie .....	3
Argomenti esposti in questa guida .....	3
Dove incontrarsi con gli altri autori .....	4
La pagina web del sottoprogetto documentazione .....	4
La lista di posta degli autori Firebird .....	4
Il newgroup su Atkin .....	4
Scegliere un argomento .....	5
Prepararsi a scrivere: farsi uno schema! .....	5
DocBook XML – una introduzione .....	6
Un'introduzione generale a XML .....	6
Un'introduzione a DocBook XML .....	9
Strumenti per scrivere in DocBook XML .....	11
Editor di testi .....	11
Editor XML .....	12
Impostare il vostro documento DocBook .....	13
Creare il documento .....	13
Scrivere il testo .....	15
Gli elementi che si usano più di frequente .....	16
Elementi gerarchici .....	16
Liste .....	20
Link .....	21
Listati, schermate, vista letterale ed esempi .....	23
Tabelle .....	25
Immagini .....	28
Avvertimenti .....	29
Testate di paragrafo .....	30
Vari elementi in linea .....	31
Considerazioni finali sugli elementi .....	34
Lingua e stile .....	34
Lingua .....	35
Stile .....	35
Problemi di diritti d'autore .....	36
Usare il materiale scritto da altri .....	37
Proprietà dei diritti d'autore e PDL .....	38
Aggiungere il vostro documento al modulo del manuale .....	42
Richiedere i diritti di deposito .....	42
Cosa fare e cosa no, avendo i diritti di deposito .....	42
Depositare il proprio lavoro .....	43
Pubblicare i documenti sul sito web di Firebird .....	44
Dare un nome al file PDF .....	44
HTML unica pagina .....	44
Depositare il PDF .....	45
Depositare i file HTML multipagina .....	45
Aggiornare l'indice della documentazione di Firebird .....	45
Appendice A: Cronologia .....	48
Appendice B: Note di licenza .....	51

---

# Introduzione

## *Scopo di questa guida*

Questa guida mostra i vari aspetti della scrittura di documenti per il progetto Firebird. È stata scritta per coloro che intendono aiutare a scrivere documenti per il progetto Firebird, o che considera di farlo. Dopo aver letto questa guida si hanno le conoscenze necessarie per scrivere i documenti nel formato da noi scelto, cioè DocBook XML.

## *Conoscenze necessarie*

Prima di leggere questa guida, bisogna avere una certa padronanza su:

- che cos'è il modulo del manuale di Firebird.
- che cos'è CVS, e come si usa un CVS client per scaricare il modulo del manuale ultima revisione.
- come costruire la documentazione a partire da modulo scaricato.

Questa conoscenza è fondamentale per poter contribuire al nostro progetto di documentazione. Se non si è sicuri riguardo ad uno o più di questi punti, sarebbe meglio leggere prima [Come scaricare e ricostruire il modulo del manuale di Firebird](#), e poi proseguire la lettura.

## *Argomenti esposti in questa guida*

Si inizia con argomenti rapidi quali:

- La mailing list firebird-docs.
- Selezionare un argomento.
- Fare una traccia dell'organizzazione del documento.

Dopodichè ci sarà una lunga digressione sui fondamenti del DocBook XML, perchè in quel formato andranno preparati i documenti. Gli argomenti trattati saranno:

- Che cos'è DocBook XML?
- Ragioni per preferire DocBook così fortemente rispetto ad altri formati.
- Strumenti utili per produrre testi in DocBook.

Anche se la parola "DocBook" non significa ancora molto ancora, la necessaria abilità potrà essere acquisita in meno di un'ora, e ci saranno possibilità per cui si potrà beneficiare di queste capacità per altri progetti, dovunque ci sia da scrivere documentazione tecnica.

La parte successiva riguarda il vero e proprio modo di scrivere per DocBook:

- Impostare il documento.
- Usare gli elementi di DocBook.
- Una parola o due sul linguaggio e lo stile di scrittura.

- I diritti d'autore (Copyright) and la Licenza di Pubblica Documentazione (Public Documentation License).

Infine, si mostra come aggiungere il documento finito al progetto Firebird. Gli argomenti principali in questa sezione sono:

- Inviare il documento finito al modulo del manuale.
- Dove chiedere per i diritti di invio se non ce li hai.
- Cosa fare e cosa non fare una volta ottenuti i questi diritti.
- Pubblicare le versioni HTML e PDF dei documenti sul sito web di Firebird.

## Dove incontrarsi con gli altri autori

### *La pagina web del sottoprogetto documentazione*

La pagina web del sottoprogetto documentazione è in:

<http://www.firebirdsql.org/index.php?op=devel&sub=doc>

Contiene notizie riguardo alle attività, link ai documenti già pubblicati, pianificazioni per le future versioni, ed altro ancora.

### *La lista di posta degli autori Firebird*

Se si crede in un serio impegno nello scrivere per Firebird, la prima cosa da fare sarebbe sottoscrivere l'iscrizione alla lista di posta (mailing list) dove si discutono pianificazioni e lavoro. È aperta a tutti, e iscriversi non implica nessun obbligo, tranne scrivere solo in lingua inglese. È sufficiente inviare una mail a:

```
<firebird-docs-request@lists.sourceforge.net>
```

con la parola «subscribe» o come soggetto o come prima parola nel corpo del messaggio. In alternativa, si può riempire il modulo a questa pagina web:

<http://lists.sourceforge.net/lists/listinfo/firebird-docs>

Qualsiasi sia il metodo scelto, si riceverà una posta elettronica automaticamente entro pochi minuti. Basta seguire le istruzioni nel messaggio (al solito in inglese) e si sarà nella lista.

### *Il newgroup su Atkin*

C'è anche una interfaccia al gruppo di posta per questa ed altre mailing list relative ai sottoprogetti di Firebird. Talvolta non funziona bene, qualche volta ha problemi, (io non lo userei per inviare i messaggi *N.d.A.* - invece io lo uso *N.d.T.*), ma va benissimo per scorrere tutte le mail archiviate. Le mail vengono tenute per molto tempo, per cui scaricando tutti i messaggi che sono sul server si ha un bel pezzo della storia recente fino alla vostra sottoscrizione al newsgroup.

Per accedere al gruppo, utilizzate un lettore di news e configurategli il server:

news.atkin.com

e scaricate la lista dei gruppi. Sottoscrivete i gruppi che preferite. Notate che la lista firebird-docs corrisponde al gruppo sourceforge.firebird-doc (*senza la "s" finale*) sul server Atkin.

Può essere che il lettore di posta che avete riesca a configurarvi automaticamente il tutto adoperando questo link:

<news://news.atkin.com/sourceforge.firebird-doc>

Si può inviare posta al gruppo anche se non si è iscritti alla lista di posta (mailing list), ed in tal caso il messaggio verrà trattenuto fino all'approvazione da parte di una persona che funge da moderatore. Ciò significa che il messaggio può ritardare la sua pubblicazione di parecchie ore (o raramente anche di qualche giorno).

## Scegliere un argomento

I seguenti suggerimenti possono essere d'aiuto nel trovare un argomento su cui scrivere:

- Prima di tutto assicurarsi di sapere cosa c'è già - non c'è bisogno di scrivere tre guide per la conversione da MS-SQL a Firebird.
- Poi chiedersi cosa manca, e che potrebbe essere utile all'utente Firebird generico ma anche per un particolare gruppo di utenti.
- Deve essere anche una cosa di cui *piace* scrivere. La scelta più logica dovrebbe essere qualcosa di cui sei pratico, ma anche qualcosa di cui prima si da studiare ancora qualcosa (ovviamente questo comporta maggior lavoro, ma anche una grande esperienza acquisita se si ha il tempo da dedicarsi).
- Non si deve necessariamente scrivere un intero libro, una guida o un articolo. Ci sono già magari altri che stanno lavorando su un testo più grande a cui si può contribuire; si possono scrivere uno o più capitoli di un libro, oppure anche si possono fornire documenti o materiali riguardo un soggetto di cui si è esperti.
- Un suggerimento è quello di parlare delle proprie idee o di cercare idee sulla lista firebird-docs. La frequenza di nuovi messaggi può essere molto bassa in certi momenti, ma tutti i messaggi *saranno* sicuramente letti e qualcuno risponde sempre.

## Prepararsi a scrivere: farsi uno schema!

Sarebbe bene farsi sempre uno schema di quello che si intende scrivere, prima di iniziare. Farsi uno schema aiuta ad «organizzarsi»; riduce la possibilità di dimenticarsi qualcosa d'importante, e rende un po' più semplice l'effettiva scrittura del documento.

Si può seguire questa procedura nel farsi uno schema:

- Definire esattamente cosa si desidera che i lettori imparino dal proprio lavoro.
- Dividere il lavoro in unità logiche - tipo capitoli, sezioni, sottosezioni ecc.
- Assicurarsi che l'ordine delle unità logiche abbia un senso, specialmente per i manuali, le esercitazioni, le guide per l'utente. Sarebbe a dire: strutturare le varie unità in modo tale che qualunque cosa il lettore debba fare o capire prima, sia anche prima nella documentazione scritta.

- Una volta pronto, è bene presentare lo schema alla lista firebird-docs su sourceforge.net per chiedere commenti.

Una volta che lo schema è pronto e revisionato, lo si ricontrolla per determinare se si hanno tutte le informazioni necessarie per scrivere. Sarebbe bene avere tutte le informazioni pronte prima di iniziare la scrittura, perchè talvolta un'informazione non ben conosciuta all'inizio potrebbe comportare una struttura diversa del documento. Sarebbe stato meglio allora avere quell'informazione nella fase di preparazione.

## DocBook XML – una introduzione

Il formato scelto per la documentazione che è nel modulo del manuale di Firebird è *DocBook XML*. Per coloro che non hanno molto sentito parlare di XML e/o di DocBook, seguirà una breve introduzione a XML in generale e a DocBook XML in particolare. Si avverte che queste introduzioni sono giusto per dare un'immagine molto semplificata. Ovviamente sarà idonea a sufficienza: non è necessario essere un esperto in DocBook XML per poter scrivere documentazione per Firebird. È necessario avere solo un po' di conoscenze basilari - che si possono apprendere in mezz'oretta dai paragrafi successivi - ed un po' di esperienza nell'applicare gli elementi di DocBook XML al vostro testo (cosa che si otterrà molto presto appena si inizia a scrivere).

[Salta l'introduzione a XML](#) Se conosci già tutto sugli elementi, le etichette e gli attributi di XML, sul rendering (cioè restituzione grafica) ed il multichannel publishing (editoria e pubblicazione multicanale).

[Salta entrambe le introduzioni](#) Se si è anche un esperto autore con DocBook.

### Nota

Nonostante si richieda che almeno si *provi* a sviluppare il proprio lavoro con il formato DocBook, si è altresì coscienti che non tutti hanno il tempo per apprenderlo (o per convertire la propria documentazione esistente in DocBook). Nel caso, se ne parli sulla lista firebird-docs (possibilmente in inglese). Certamente non si rifiuta della documentazione utile solo perchè non è nel giusto formato.

## Un'introduzione generale a XML

XML sta' per *Extensible Markup Language*, che è semplice testo piano con delle etichette di demarcazione. Un tipico frammento di testo XML potrebbe comparire come il seguente:

```
<paragraph>
<loud>'No!'<</loud> she screamed. <scary>But the bloody hand
<italics>kept on creeping</italics> towards her.</scary>
<picture file="bloody_hand.png" />
</paragraph>
```

### Marcatori ed attributi

Nell'esempio precedente, le parole e le frasi racchiuse fra parentesi angolari sono i marcatori. `<italics>` è un *marcatore iniziale (start tag)*, mentre `</italics>` è un *marcatore finale (end tag)*, e `<picture file="bloody_hand.png" />` è un marcatore autonomo, detto comunemente *marcatore di elemento vuoto (empty-element tag)*. I marcatori XML hanno sempre uno di questi tre formati:

**Tabella 1. Formato dei marcatori XML**

Tipo del marcatore	Comincia con	Finisce con
Marcatore iniziale	<	>
Marcatore finale	</	>
Marcatore di elemento vuoto	<	/>

Ancora riferendosi al nostro esempio, le parole `paragraph`, `loud`, `scary`, `italics` e `picture` sono i *nomi dei marcatori* (*tag names*). Nel marcatore `<picture... />`, la parte `file="bloody_hand.png"` viene detta un *attributo* (*attribute*), dove `file` è il *nome dell'attributo* (*attribute name*) ed invece `bloody_hand.png` è il *valore dell'attributo* (*attribute value*). I valori degli attributi devono essere sempre tra apici, e sono permessi sia i singoli che i doppi apici.

XML permette di definire i marcatori a piacere, purchè siano costruiti correttamente. Così `<questotag>`, `<altrotag>`, and `<io_non_sono_un_tag />` sono tutti marcatori XML corretti. (Un testo XML che segue lo standard viene detto *well-formed* (ben costruito o *corretto*); Il termine *valido* (*valid*) è utilizzato solo per implementazioni specifiche, ad esempio DocBook XML)

Ovviamente i marcatori in quanto tali non sono stati pensati per apparire nel documento finale (cioè il documento come verrà presentato ai lettori). Contengono, invece, istruzioni che influenzeranno il suo aspetto. XML, quando usato per scrivere documenti, è un tipico *formato sorgente* (*source format*), con lo scopo di essere processato da altro software per produrre documenti ben formattati. Questo procedimento viene usualmente chiamato *restituzione* (*rendering*).

Alcuni marcatori sono senza tema d'errore indicazioni di formattazione:

```
<italics>kept on creeping</italics>
```

significa ovviamente che le parole *words kept on creeping* devono essere visualizzate o stampate in corsivo (*italics in inglese*). Tuttavia,

```
<loud>'No! '</loud>
```

è un po' meno ovvio. La parola *No!* deve apparire in grassetto? Oppure sottolineata? O ancora in corsivo? O vuol dire che questo testo debba essere letto ad alta voce da un sintetizzatore vocale, e il marcatore `<loud>` lo informa di alzare la voce? Tutte questi casi sono possibili, e c'è di più: spesso un sorgente XML può essere convertito in vari formati - ad esempio, un documento PDF, una pagina web in HTML, un file audio. Questa è chiamata *editoria multicanale* (*multichannel publishing*). Con l'editoria multicanale, `<loud>` potrebbe essere trasformato in grassetto nel documento PDF; in un carattere grassetto rosso nella pagina web; e un aumento del 50% del volume per il sintetizzatore vocale.

Osservando gli altri marcatori, `<picture... />` (cioè immagine) è ovviamente l'istruzione per inserire l'immagine `bloody_hand.png` (`mano_insanguinata`) nel documento, e `<scary>`, be'... questo è ancora meno chiaro di `<loud>`. Potrebbe essere che la frase tra i marcatori `<scary>` debba essere insanguinata. Può essere che ci sia da mettere in sottofondo musica dell'orrore. Tutto dipende da chi definisce i marcatori, e dal programma che viene utilizzato per la restituzione.

Il marcatore `<paragraph>`, infine, è di tipo strutturale. Informa sulla posizione che queste linee devono assumere nella struttura gerarchica interna al documento. Nel risultato finale, i paragrafi potrebbero o meno essere separati da linee vuote. Ancora, ciò dipende dal software di restituzione e pure da opzioni configurabili dall'utente. Altri marcatori di struttura che si potrebbero considerare sono ad esempio `<chapter>`, `<section>`, e `<subdocument>` che significano, nell'ordine, capitolo, sezione e sottodocumento.

## Caratteri speciali ed Entità

Poichè il carattere «<» ha un significato particolare in quanto inizio di marcatore, non si può includere direttamente come valore, invece affinché i lettori possano vedere una parentesi angolare aperta, bisogna scrivere questo:

```
&lt;
```

Che è una «e commerciale» (in inglese ampersand), seguita dalle lettere «l» e «t» (che stanno per «minore di» cioè *less than*), seguite da un punto e virgola. Si può usare anche `&gt;` (*greater than* cioè «maggiore di») per il carattere di parentesi angolare chiusa «>», ma non è necessario.

XML ha molti codici come questi, e sono chiamati *entità* (*entities*). Alcune rappresentano caratteri, come `&lt;` e `&auml;` (a minuscola con umlaut) ed altri con scopi totalmente diversi. Ad ogni modo, tutti cominciano con una e commerciale e finiscono con un punto e virgola.

Ma, un momento... se una e commerciale serve è l'inizio di una entità, come includere quel carattere nel proprio testo? Be', c'è un'entità anche per questo:

```
&amp;
```

Così questa linea di XML:

```
Kernigan & Ritchie hanno scelto '&lt;' come operatore minore-di per il linguaggio C.
```

si trasformerà nel documento finale così:

```
Kernigan & Ritchie hanno scelto '<' come operatore minore-di per il linguaggio C.
```

E ci sono pure le buone notizie: usando un editor specifico per l'XML per scrivere il vostro documento, probabilmente si potrà digitare direttamente «<» e «&» dovunque si desidera. L'editor si farà carico della trasformazione direttamente in `&lt;` e `&amp;` nel XML quando lo salva su disco. Più oltre in questa guida ci saranno riferimenti ad alcuni editor XML/DocBook.

## Elementi

C'è un altro importante concetto di XML da sapere: gli *elementi*. Un elemento è la combinazione di un marcatore di inizio, un corrispondente marcatore di fine, e tutto ciò che c'è in mezzo. «Tutto ciò che c'è in mezzo» viene definito il *contenuto* dell'elemento, e può includere altri elementi. Gli elementi prendono il nome dal loro marcatore, pertanto si parla di elementi di paragrafo, elementi in corsivo, ecc.

### Nota

In effetti, gli elementi sono un concetto più basilare dei marcatori: i marcatori sono di fatto gli identificatori degli elementi. Pertanto sarebbe stato meglio dire che i marcatori prendono il nome dai loro elementi. Ma poichè i marcatori sono più facili da riconoscere di un intero elemento, si è pensato di spiegarli prima.

Questo è un elemento:

```
<loud>'No! '</loud>
```

Anche questo è un elemento:

```
<paragraph>Questo è un elemento che contiene <bold>un altro</bold>
elemento!</paragraph>
```

I marcatori di un elemento vuoto sono un caso un po' particolare: l'elemento che marcano non ha contenuto, e l'elemento non ha una *coppia* di marcatori, iniziale e finale.

```
<picture file="bloody_hand.png"/>
```

### Importante

Non si confonda il contenuto con gli attributi. Il contenuto sta' *tra* due marcatori, gli attributi sono *nei* marcatori. Il marcatore di elemento vuoto dell'esempio precedente ha un attributo, ma non ha contenuto.

Si sta' insistendo sul concetto di elemento perchè molta documentazione tende a parlare di «chapter elements», «title elements» (elementi capitolo, elementi titolo) ecc. invece che di «chapter tags» e «title tags» (marcatori di capitolo, marcatori di titolo). I termini sono usati spesso in modo intercambiabile, ma vi sono casi in cui è importante sapere la differenza.

## XML: Conclusione

Bene, questo è tutto quello che c'è da sapere su XML. Per il momento si dovrebbe avere un'idea generale su come deve apparire un testo XML, che cosa sono i marcatori e gli elementi e a cosa servono. Come già detto, si è data giusto un'idea molto semplificata, ma è già abbastanza per i nostri scopi.

Dovrebbe essere anche chiaro che scrivere con un XML inventato e sui generis è inutile, a meno di non avere un software che riesce a comprendere i *propri* marcatori. Come, se no, riuscire a trasformare il proprio sorgente XML in un documento ben composto e presentabile?

Fortunamente non si ha la necessità di sviluppare le proprie definizioni degli elementi ed i software di conversione. Ci sono un buon numero di tipi formalizzati di XML disponibili, ognuno che definisce un insieme di marcatori e, cosa altrettanto importante, un insieme di regole sul come usarli. DocBook XML è uno di questi tipi.

## Un'introduzione a DocBook XML

DocBook è stato progettato per semplificare la scrittura di documenti strutturati usando SGML o XML (non ci si deve preoccupare di SGML – noi usiamo solo XML). È particolarmente adatto a scrivere libri ed articoli tecnici, specialmente su argomenti relativi ai computer. DocBook XML è definito nel suo *Document Type Definition* o *DTD*: un insieme di definizioni e di regole che descrivono esattamente come deve essere strutturato un documento DocBook valido. DocBook sta' diventando rapidamente uno standard di fatto per documentazione tecnica, e viene supportato da un crescente numero di strumenti ed applicazioni.

## Caratteristiche di DocBook XML

Le maggiori caratteristiche di DocBook – al contrario del «generico» XML – sono:

- Il DTD di DocBook definisce un numero limitato e preciso di marcatori, e da' regole esatte su come vanno usati: quali attributi sono possibili per un dato marcatore, quali elementi possono stare all'interno un dato elemento, e così via. Il documento che non usa marcatori definiti, o che non segue le regole, non sarà più un documento DocBook (e gli strumenti di supporto potrebbero non funzionare più).

- I marcatori di DocBook determinano sempre e solo struttura logica e semantica (significato), ma *mai* l'estetica. In DocBook, si troveranno marcatori strutturali come <book>, <part>, <chapter>, <section>, <para>, <table>; inoltre vi sono marcatori semantici come <filename>, <warning>, <emphasis>, <postcode>; ma mai nulla come <font>, <bold>, <center>, <indent>, <backgroundcolor> – nulla cioè che abbia a che fare con l'aspetto estetico finale del documento.
- A causa di ciò, era necessario prendere una decisione su come i marcatori in qualche modo dovessero essere tradotti nell'aspetto estetico finale. Questa decisione (o meglio, le regole di restituzione) avrebbero potuto essere fissate negli strumenti software, ma questo avrebbe reso le cose poco flessibili. Questo è il motivo per cui le regole sono in gran parte definite in *fogli di stile* o *stylesheets*. Un foglio di stile è un documento che informa gli strumenti su cose come:

«Stampa i titoli di capitolo in una font a 24 punti nera; inizia ogni capitolo a pagina nuova; usa il corsivo per l'enfasi; visualizza gli avvisi in una font rossa 12 punti in grassetto; usa il maiuscolo per gli acronimi; ecc. ecc.»

Questo metodo permette all'utente di modificare gli stylesheets se non gli piace il risultato del documento finale. Sarebbe molto più difficile, sebbene non impossibile, modificare gli strumenti software stessi.

#### Nota

I fogli di stile che sono usati per convertire i testi DocBook XML in altri formati, sono chiamati *transformation stylesheets*. Questi sono scritti in un altro tipo di XML, chiamato *XSLT* (eXtensible Stylesheet Language for Transformations).

## Benefici del DocBook XML

DocBook ha molti vantaggi per tutti coloro che scrivono documentazione tecnica. Queste sono secondo noi le più importanti:

- Un documento DocBook XML consiste di puro, pulitissimo, *contenuto*. Non bisogna per nulla preoccuparsi dell'aspetto presentazionale mentre si scrive il documento; ci si può concentrare sulla struttura e sul contenuto. Questo potrebbe sembrare un po' strano a chi è abituato a scrivere in Word, ad esempio, ma ve lo promettiamo: presto comincerete ad apprezzarlo.
- Poichè DocBook è solo struttura e significato, verrà facilissimo trasformare lo schema preparato in uno scheletro DocBook.
- Molti preparano documenti per il modulo del manuale. Se ognuno di essi usasse formati diversi, ma anche un solo formato come Word o HTML, ognuno di loro renderà con aspetto molto diverso il proprio lavoro perchè prenderà le proprie decisioni in modo autonomo sull'estetica. Ovviamente si potrebbero sviluppare tutta una serie di convenzioni, ma ugualmente ogni scrittore dovrebbe esserne a conoscenza e preoccuparsi di applicarle sempre e comunque. No... meglio mettere le regole in un punto centralizzato: i fogli di stile, e fare in modo che gli scrittori si preoccupino del contenuto e non dell'aspetto estetico. I fogli di stile assicureranno che tutta la nostra documentazione avrà esattamente lo stesso aspetto.
- Se non piace l'aspetto estetico della trasformazione, si può facilmente modificare, fintantochè le regole sono in un foglio di stile. Non deve essere modificato nulla nel documento sorgente DocBook; tutto quello che c'è da fare, dopo aver modificato i fogli di stile, è rigenerare i documenti finali. I documenti appena rifatti, automaticamente avranno il nuovo aspetto. Provate a farlo se le istruzioni di formattazione sono sparpagliate fra tutti i documenti in posti diversi in modi diversi...!

- Un altro vantaggio è che DocBook è uno standard «aperto», cioè non proprietario di una qualsiasi applicazione commerciale o di un particolare sistema operativo. Scaricando il modulo del manuale di Firebird, si possono ricostruire i documenti HTML e PDF a partire dal sorgente DocBook sia sotto Linux che sotto Windows, e si potrebbero supportare altri sistemi operativi se necessario.
- Un documento DocBook è puro testo, che è ideale per essere utilizzato con CVS. Naturalmente, una struttura CVS può contenere anche file binari, ma molte delle utilità offerte da CVS (mostrare la differenza fra due versioni di un file, ad esempio) funzionano solo con i file di testo.

Certamente, nessuno di questi benefici è unico solo per DocBook. Ma DocBook li ha tutti, ed è supportato ampiamente. Questo lo rende la scelta ideale per la documentazione di Firebird.

## La documentazione su DocBook in Internet

Seguono alcuni link (in inglese) nel caso si desiderasse saperne di più su DocBook:

- <http://opensource.bureau-cornavin.com/crash-course/>

*Writing Documentation Using DocBook – A Crash Course* di David Rugge, Mark Galassi e Eric Bischoff. Una introduzione ben fatta, anche se molti degli strumenti trattati non sono gli stessi nostri.

- <http://docbook.org/tdg/en/>

*DocBook – The Definitive Guide*, di Norman Walsh e Leonard Mueller. Una guida che non è per inesperti. Difatti la prima parte è praticamente incomprensibile se si è principianti di DocBook. La ragione per menzionarla qui, è perchè è un riferimento importantissimo online, che viene spesso consultato dagli esperti mentre scrivono.

- <http://www.tldp.org/HOWTO/DocBook-Demystification-HOWTO/>

Questo *DocBook Demystification Howto* è interessante se si desidera sapere un po' di più di quanto si è detto qui riguardo a XML e DocBook. C'è anche molto materiale su SGML e su strumenti che non vengono usati per il sottoprogetto documentazione di Firebird.

- <http://sourceforge.net/projects/docbook>

È il sito del progetto open source di DocBook a SourceForge.

Nel caso si conoscessero altre buone risorse online riguardo a DocBook, siete pregati di mettere un messaggio sulla lista di firebird-docs.

## Strumenti per scrivere in DocBook XML

### *Editor di testi*

Poichè DocBook è un formato non binario, si può usare qualsiasi editor di testo semplice, come emacs, pico, il blocco note di Windows, oppure vi di Linux/Unix per scrivere il documento. E alcuni scrittori preferiscono proprio questo metodo, rispetto ad altri editor più sofisticati, perchè permette il completo controllo sul loro testo ed i marcatori scritti per esteso sono sempre visibili. Il rovescio della medaglia è che gli editor di testo semplice non possono *validare* il documento e gli errori si potranno notare solo quando la ricostruzione non funziona.

Inoltre la struttura di un documento, specialmente di quelli grandi, è difficile da seguire in modo testo, anche se un consistente uso dell'indentazione può aiutare.

Volendo usare questo metodo, o almeno per provarlo, sarebbe bene almeno utilizzare un editor di testo in grado di evidenziare la sintassi XML. Uno adatto, ed al momento di uso gratuito, è ConText, disponibile a <http://www.fixedsys.com/context/>. Purtroppo ConText non è in grado di salvare i caratteri in formato UTF-8. Questo non è un problema per i documenti in US-ASCII (basta salvarli come DOS o Unix), ma appena si usano caratteri accentati o qualsiasi carattere oltre ASCII 127, ConText diventa inutile. Un'alternativa, sempre gratuita, è SciTE a <http://scintilla.sourceforge.net/SciTEDownload.html>. È meno intuitivo, ma molto potente.

#### **Avvertimento**

Non salvare documenti contenenti caratteri non US-ASCII in formato 8-bit, sia in ConText che in qualsiasi altro editor! Qualsiasi cosa che non sia US-ASCII deve essere salvato in codifica Unicode, ad esempio UTF-8 (per molti linguaggi) o UTF-16 (per altri linguaggi, se la lunghezza del file in UTF-16 è minore o non molto più grande di UTF-8). Di fatto, questi motivi legati alle codifiche, sono un altro buon motivo per usare gli editor XML, perchè di solito salvano nel formato corretto automaticamente.

## **Editor XML**

Gli editor dedicati a XML hanno spesso un'interfaccia adatta a gestire i marcatori in modo più semplice (anche se in qualche caso fastidioso); molti permettono di collassare ed espandere gli elementi strutturali in modo da verificare la struttura del documento ed allo stesso tempo di zoomare nell'elemento su cui si sta lavorando; permettono inoltre di passare tra diverse modalità di vista. Molte possono validare il documento rispetto al DTD di DocBook, ed alcune anche hanno perfino una modalità di scrittura che permette di scrivere quasi come in un word processor.

L'autore di questa guida ne ha provati un gran numero (gratuiti, economici, versioni di valutazione) e ha trovato che XMLMind XML Editor è il più utile. Questa è un'opinione personale, naturalmente (N.d.T. coincide, se è permesso, con quella del traduttore); naturalmente l'esperienza di ciascuno può essere diversa.

Alcuni XML editors da prendere in considerazione sono:

- XMLMind XML Editor, detto brevemente XXE. L'edizione standard è gratuita.

<http://www.xmlmind.com/xmleditor/>

Esiste per Linux, Windows e Mac OS X. Richiede Java, ma comunque Java sarebbe necessario altrimenti non si riuscirebbe a ricostruire la documentazione dai sorgenti, vedi [Come scaricare e ricostruire il modulo del manuale di Firebird](#).

Caratteristiche: vista ad albero (tutti gli elementi collassabili) e vista testo (capitoli e sezioni collassabili). Questa è la modalità in cui di solito si lavora: mostra il documento in modo che sembri il semplice e chiaro word processor d'una volta, definito con dei fogli di stile che vengono con il programma. Entrambe le viste possono essere attive contemporaneamente. La modalità DocBook non permette di immettere nulla che non sia aderente alle specifiche DocBook. Selezionatore di elementi, editor degli attributi. Funzioni di ricerca e sostituzione. Dizionario (purtroppo non ancora in italiano). Selezione di caratteri speciali. Pulsanti per creare gli elementi usati più di frequente come sezioni, liste, tabelle, ecc. Sembra mancare una vista del testo XML vero e proprio.

- Oxygen XML Editor. \$ 48 per usi non commerciali. Prova gratuita per 30 giorni.

<http://www.oxygenxml.com>

Esiste per Windows, Mac OS X, Linux, Eclipse. Richiede Java.

Caratteristiche: editor del sorgente XML. Editor dell'albero. Editor di attributi. Pannello della struttura. Suggerimenti dei marcatori DocBook. XSLT debugger (uno strumento molto potente, non importante per scrivere ma per lavorare sui fogli di stile delle trasformazioni). Validazione, ristrutturazione, dizionario, ecc. ecc. Un XML editor proprio buono.

- epcEdit. € 89 per usi non commerciali. Prova gratuita per 60 giorni.

<http://www.epcedit.com>

Esiste per: Linux, Windows, Solaris. Richiede Tcl/Tk 8.1 o superiore (incluso nel pacchetto).

Caratteristiche: Pannello della struttura ad albero. Selezione degli elementi. Editor di attributi. Il pannello del documento può passare dalla vista testo puro a quella grafica XML. Nessuna particolare modalità per DocBook, ma può validare ogni documento XML basandosi sul suo DTD.

- Altova XMLSpy. La versione Home Edition è adesso gratuita.

[http://www.altova.com/products\\_ide.html](http://www.altova.com/products_ide.html)

Esiste per: Windows, Eclipse. (Si dice che funzioni su Linux usando Wine, e su Mac OS X usando Virtual PC 6.)

Caratteristiche: Vista testo e navigatore. Tutti gli elementi sono collassabili nella vista navigatore, che però è in sola lettura. Selezione degli elementi. Selezione degli attributi. Funzioni modifica e cerca. Selezione dei caratteri speciali.

C'è una matrice di comparazione delle caratteristiche delle versioni Home, Professional e Enterprise a [http://www.altova.com/matrix\\_x.html](http://www.altova.com/matrix_x.html).

Questo elenco non intende essere completo; tuttavia, conoscendo un *buon* editor XML (cioè adatto a scrivere documento per Firebird) e ritenendo debba essere in questa lista, si può farcelo come al solito sapere attraverso la mailing list.

## Impostare il vostro documento DocBook

Bene, ci siete? So che abbiamo passato un po' di tempo spiegando XML e DocBook, ma sono realmente convinto che fosse necessario perchè sono concetti nuovi per molta gente. Dando solo dei link e dicendo di trovarsi da soli come fare, ci avrebbe fatto perdere degli scrittori di documentazione altrimenti validi.

Comunque, siamo arrivati e siamo pronti a scrivere il documento. Questa sezione spiega come impostare il documento DocBook; la successiva spiega come applicare i marcatori giusti al posto giusto.

### Creare il documento

Ogni documento nel nostro manuale è parte di un `<set>`. Questo è l'elemento in cima nella gerarchia di DocBook. Un set (lett. insieme) contiene un certo numero di `<book>s` (cioè libro/i), che a loro volta contengono `<chapter>s` (cioè capitolo/i), e così via.

Un vantaggio di mettere i libri in un insieme è che si possono fare riferimenti l'un l'altro, cioè si possono porre link in un documento che puntano esattamente ad una precisa posizione in un altro libro. Questo vantaggio è limitato dal fatto che non funzionano tra file PDF diversi (un limite che non esiste nei file HTML). Un altro vantaggio è la creazione automatica dell'indice, detta anche tabella dei contenuti o ToC (acronimo di Table of Contents).

Fortunatamente, mettere i libri in un unico set non implica necessariamente che debbano generare un solo grande file. DocBook permette di impostare un documento principale, come mostrato sotto. Al momento non si consideri la sezione che inizia con "<!DOCTYPE" fino a tutto il segno ">": non si avrà mai il bisogno di scrivere da soli una cosa così orribile. Alla peggio si copia una riga e la si edita, oppure, se si deve tradurre un set esistente in un nuovo linguaggio, si copia con altro nome e si modifica l'intero documento principale da una versione esistente; ad esempio per la versione italiana, ho copiato e modificato l'originale versione inglese.

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE set PUBLIC "-//OASIS//DTD DocBook XML V4.1.2//EN"
  "docbookx/docbookx.dtd" [
  <!ENTITY preface SYSTEM "firebirddocs/preface.xml">
  <!ENTITY fb-intro SYSTEM "firebirddocs/firebirdintro.xml">
  <!ENTITY ...>
  <!ENTITY ...>
]>

<set id="firebird-books">
  &preface;
  &fb-intro;
  ...
  ...
</set>
```

Con il documento principale impostato come sopra, i vari libri possono essere in file distinti: `preface.xml`, `firebirdintro.xml`, ecc., il che permette di editarli indipendentemente. Il libro, ad esempio quello che si sta scrivendo, è pressapoco strutturato così:

```
<?xml version="1.0" encoding="UTF-8"?>

<book id="fbintro">
  <chapter id="fbintro-preface">
    ...
    ...
  </chapter>
  <chapter id="fbintro-installing-firebird">
    ...
    ...
  </chapter>
  ...
  ...
</book>
```

Naturalmente, scrivendo un nuovo documento, questo deve essere fatto riconoscere all'insieme principale, ma di questo se ne parlerà quando si sarà pronti a scrivere. Al momento non si può dare una regola generale, in quanto dipende da quello che si intende scrivere: un libro, un articolo, un capitolo, un paio di capitoli... e da come il lavoro si incastra con il resto già esistente.

Ogni documento DocBook deve iniziare con questa linea:

```
<?xml version="1.0" encoding="UTF-8"?>
```

(Notare che per alcune lingue, UTF-16 dovrebbe essere la scelta migliore.)

Se si scrive il documento in un editor di testi, cioè testo XML «nudo e crudo», bisogna scriversi quella linea così com'è. Usando un editor specifico per XML, viene inserita automaticamente creando un nuovo documento.

## Posizioni degli insiemi in funzione del linguaggio

I files del set di documentazione per l'inglese deve essere posto nell'albero di directory `manual/src/docs/firebirddocs`. I documenti in altre lingue vanno nell'albero della directory della lingua: ad esempio, per il francese `manual/src/docs/firebirddocs-fr`, per lo spagnolo `manual/src/docs/firebirddocs-es`, per l'italiano in `manual/src/docs/firebirddocs-it`, ecc.

A partire dal gennaio 2006 c'è la possibilità di creare set di base aggiuntivi, il primo dei quali è stato `rlsnotes`, l'insieme delle Release Notes (note di versione). Anche in questo caso si utilizza la stessa logica: le Release Notes in inglese vanno in `manual/src/docs/rlsnotes`, in francese in `manual/src/docs/rlsnotes-fr`, e così via.

Ognuno di questi alberi di directory (`firebirddocs`, `firebirddocs-es`, `firebirddocs-nl`, `rlsnotes`, `rlsnotes-it`, ecc.) contiene un diverso `<set>`, con un documento principale (`master document`) ed un certo numero di file da includere (`include files`).

## Scrivere il testo

Scrivendo DocBook XML con un text editor come il Blocco Note di Windows, emacs oppure ConText, si può andare a capo, usare l'indentazione, e spaziare il testo a piacere. Ogni «*spazio bianco*» (*whitespace*), cioè una sequenza di uno o più caratteri spazio, tabulazione, a capo, salto pagina (`space`, `tab`, `linefeed`, `formfeed`), viene convertito in un singolo carattere di spazio in uscita. Così:

```
<section><title>Firebird Architectures</title><para>Now let's have a
look at Firebird's different architectures.</para><itemizedlist>
<listitem><para>First, there's the so-called <firsttterm>Classic Server
</firsttterm>.</para></listitem><listitem><para>Then there is <firsttterm>
Superserver</firsttterm> architecture.</para></listitem><listitem><para>
And finally, with the release of Firebird 1.5 we also have the
<firsttterm>embedded server</firsttterm>.</para></listitem></itemizedlist>
</section>
```

avrà lo stesso aspetto di questo:

```
<section>
  <title>Firebird Architectures</title>
  <para>Now let's have a look at Firebird's different
    architectures.</para>
  <itemizedlist>
    <listitem>
      <para>First, there's the so-called
        <firsttterm>Classic Server</firsttterm>.</para>
    </listitem>
    <listitem>
      <para>Then there is <firsttterm>Superserver</firsttterm>
        architecture.</para>
    </listitem>
```

```
<listitem>
  <para>And finally, with the release of Firebird 1.5 we also
    have the <firstterm>embedded server</firstterm>.</para>
</listitem>
</itemizedlist>
</section>
```

Inutile a dirsi, questa seconda forma è molto più facile da leggere e capire da parte di una persona. Così, scrivendo il testo XML in chiaro, è meglio formattare il testo in modo che la struttura sia la più chiara possibile.

Usando un editor XML dedicato, bisogna rendersi conto che il tasto **Invio** chiude automaticamente il <para>grafo corrente e ne apre uno nuovo. Quindi bisogna essere coscienti di come il proprio editor si comporta sotto questo aspetto e comportarsi di conseguenza. Inoltre bisogna verificare cosa succede quando si premono spazi successivi, poiché alcuni editor XML possono usare trucchi speciali per conservarli.

## Gli elementi che si usano più di frequente

Questa sezione spiega l'uso degli elementi usati più frequentemente nella documentazione Firebird. Include inoltre molti esempi in formato DocBook XML. Usando uno strumento di scrittura specifico per XML, viene visualizzata una cosa completamente diversa rispetto agli esempi, ma aprendo il file con un editor di testi, o potendo visualizzare nel proprio editor la vista sorgente XML, si dovrebbe vedere appunto il testo XML vero e proprio come negli esempi. Si possono anche studiare i sorgenti XML del modulo del manuale, per vedere come altri autori hanno utilizzato ed applicato i vari marcatori.

Preghiamo di leggere comunque la sezione sugli elementi gerarchici anche sapendo scrivere testi DocBook fluentemente, in quanto contiene alcune linee guida relative al nostro progetto. Dopodiché potete, stante l'ipotesi precedente, saltare il resto delle sezioni su DocBook.

I novizi non siano scoraggiati dalla lunghezza apparente di questa sezione. La raccomandazione è quella di leggere *attentamente* la sezione sugli elementi gerarchici, e farsi un'idea sulle altre. Non preoccuparsi se qualcosa risulta incomprensibile a prima vista, e soprattutto non cercate di imparare tutto a in blocco! Basta aver a portata di mano questa guida scrivendo, e rivedere le sezione degli elementi utili man mano che servono (quando si hanno incertezze).

### Elementi gerarchici

La gerarchia più comune è, partendo dalla cima: <set> – <book> – <chapter> – <section> – <para>. Un libro (book) può anche contenere uno o più <article> (articolo) invece di <chapter>.

Adesso tratteremo alcuni aspetti legati alla struttura del documento.

### L'attributo id

Insieme, libri, capitoli o articoli e le sezioni principali dovrebbero avere un attributo id. Anche altri elementi potrebbero averlo. L'attributo id permette all'elemento che lo contiene di essere referenziato partendo da un'altra parte del documento, e perfino da un altro documento del set. Gli id non sono visibili nel documento reale (tranne che nel sorgente HTML), ma sono utilizzati per generare i nomi di file HTML.

Tutti gli attributi `id` di un set devono essere univoci e quindi fra loro distinti. Notare che le versioni nelle diverse lingue sono in set diversi, quindi si possono pure lasciare gli `id` originali in una traduzione.

Per convenzione interna al progetto Firebird, all'interno di un libro o di un articolo, ogni `id` deve avere un valore che inizia con la stessa parola in minuscolo, esempio `usersguide`, seguita da un meno, seguiti da una o più parole in minuscolo sempre separate da un meno. Esempi sono `usersguide-intro` e `usersguide-download-install`. Attenzione: questa non è una prerogativa di DocBook, ma una convenzione interna nostra.

## L'attributo `lang` per i set non inglesi

Creando un nuovo set, o traducendone uno, bisogna impostare l'attributo `lang` nell'elemento radice, ad esempio:

```
<set id="firebird-books-it" lang="it">
```

Questo assicura che venga generato il testo giusto per note, attenzione, suggerimenti, ecc., e che vengano generati i corretti simboli in base alla lingua. Sarebbe bene definire l'attributo anche all'inizio di ogni documento, nel caso che siano rigenerati fuori del contesto di un set.

Per i set in inglese, l'attributo `lang` è opzionale.

## Titoli

I vari set, libri, articoli, capitoli e sezioni devono avere sempre un `title`, o come elemento figlio diretto o all'interno di un elemento `xxxinfo` (vedi oltre). È legale includerlo in entrambi, ma in questo caso i due titoli *devono* essere identici. Diversamente dall'`id`, che è un attributo, `title` è un elemento. E, parimenti, il titolo apparirà nel documento finale.

Se il titolo è lungo, si dovrebbe aggiungere subito dopo un elemento `titleabbrev` (titolo abbreviato), contenente, appunto, una forma abbreviata del titolo. La ragione è che ogni pagina HTML generata contiene una cosiddetta barra della gerarchia, come dire «voi-siete-qui», sia in cima che in fondo. Questa linea mostra il percorso a partire dalla cima (il `set`) fino alla pagina in cui siete. I singoli elementi sono cliccabili così oltre a informare dove si è, permette di navigare risalendo agli elementi superiori facilmente. Siccome ha un aspetto migliore quando sta' tutto in una sola linea, viene mostrato pertanto il titolo abbreviato quando l'elemento ne ha uno; altrimenti si usa il titolo. La stessa tecnica viene utilizzata nel pannello della struttura dei documenti PDF (cioè la finestra per navigare rapidamente che è alla sinistra).

## Elementi informativi

Scrivendo un libro o un articolo, bisogna includere un elemento `bookinfo` o un `articleinfo` all'inizio. All'interno si possono mettere informazioni sull'autore ed altro. Esistono altri elementi `xxxinfo`, ma raramente se ne ha bisogno.

```
<book id='usersguide' lang='en'>
  <bookinfo>
    <title>Firebird Users Guide</title>
    <author>
      <firstname>William</firstname>
      <surname>Shakespeare</surname>
    </author>
    <edition>25 January 2006 - Document version 1.2</edition>
```

```
</bookinfo>
...
...
</book>
```

Se l'autore è una società, o un'altra organizzazione, o un gruppo a cui ci si può riferire collettivamente, si usa `corpauthor` invece di `author`:

```
<corpauthor>IBPhoenix Editors</corpauthor>
```

Se ci sono più autori, desiderando nominarli separatamente, si crea un elemento `author` (o `corpauthor`) per ciascuno e li si riunisce in un elemento `authorgroup` tutti internamente all'elemento `xxxinfo`.

## Tipi di sezioni

Gli elementi di sezione sono leggermente diversi dagli altri, in quanto ne esistono due tipi:

- C'è l'elemento `<section>` come già anticipato. Può essere usato recursivamente, cioè si può avere una `<section>` dentro una `<section>` dentro una `<section>`... Questo ha il vantaggio di poter muovere intere sottostrutture su e giù per la gerarchia senza dover cambiare i marcatori.
- Poi ci sono i `<sect1>`, `<sect2>` ... `<sect5>`. Questi elementi devono essere propriamente posizionati, con una `<sect1>` più esterna, più `<sect2>` entro la `<sect1>` ecc. Non si può mettere una `<sect3>` direttamente in una `<sect1>`. È meno flessibile di `<section>`, ma raramente crea problemi. D'altra parte la stessa rigidità è in `<set>`, `<book>` e `<chapter>` ed ugualmente non ci si fa caso.

### Nota

Nelle prime versioni di queste guide, era raccomandata per ragioni di presentazione la forma `<sectN>`. Per miglioramenti successivi nei fogli di stile, questo non è più necessario e si può scegliere come si vuole.

## Appendici

Si possono aggiungere uno o più elementi `appendix` dopo l'ultimo capitolo di un libro, o dopo l'ultima sezione di un articolo. Le appendici possono contenere più o meno tutto quello che può essere contenuto in una `section`, incluse altre sezioni.

## Esempio di struttura

L'esempio seguente da' un'idea di come strutturare il documento:

```
<?xml version="1.0" encoding="UTF-8"?>
<book id="usersguide" lang="it">
  <bookinfo>
    <title>Firebird - Guida utente</title>
    <author>
      <firstname>Giovanni</firstname>
      <surname>Pascoli</surname>
```

```

</author>
<edition>25 gennaio 2006 - Documento versione 1.2</edition>
</bookinfo>

<chapter id="usersguide-intro">
  <title>Introduzione</title>
  <para>Buongiorno! Questa è l'introduzione alla guida utente
    del RDBMS Firebird.</para>
</chapter>

<chapter id="usersguide-download-install">
  <title>Per scaricare ed installare Firebird</title>
  <para>In questo capitolo si mostra come scaricare ed
    installare Firebird.</para>
  <section id="usersguide-download">
    <title>Scaricare Firebird</title>
    <para>Per scaricare Firebird da Internet, per prima cosa andate
      alla seguente URL: ecc. ecc. ecc.</para>
    ...altri paragrafi, e sono possibili eventuali sottosezioni...
  </section>
  <section id="usersguide-install">
    <title>Installare Firebird</title>
    <para>Installare Firebird sul vostro sistema è una cosa così:
      ecc. ecc.</para>
    ...altri paragrafi, e sono possibili eventuali sottosezioni...
  </section>
</chapter>

...altri capitoli ancora...

<appendix id="usersguide-dochist">
  <title>Elenco (o storia) delle revisioni</title>
  ...ne parliamo dopo!

<appendix id="usersguide-license">
  <title>Licenza d'uso</title>
  ...ne parliamo dopo!
</book>

```

## Alcuni punti da notare

- Per prima cosa, notare ancora che i valori degli attributi devono essere sempre fra apici. Ovviamente, usando un editor di attributi, non vanno messi, perchè ci pensa l'editor a farlo.
- Come si può vedere dall'esempio, i <chapter> e le <section> possono iniziare direttamente con uno o più elementi <para>. Ma attenzione, una volta inclusa una sezione in un capitolo o una sottosezione in una sezione, non si possono più aggiungere elementi <para> dopo di essi, solo internamente ad essi. Solo i buoni editor che riconoscono XML DocBook non permettono di fare un errore simile, ma scrivendo il testo XML DocBook a mano è una di quelle cose che è bene sapere.
- Usando un editor XML, è raramente necessario creare un elemento <para> direttamente. Ad esempio, inserendo un elemento <chapter> o <section> con XMLMind XML Editor, viene creato automaticamente un primo <para> vuoto. E quando si digita il tasto **INVIO** all'interno del testo di un paragrafo, in quel punto il paragrafo viene automaticamente chiuso da un </para> e ne viene aperto uno nuovo.

[Saltare il resto delle sezioni sugli elementi](#) conoscendo già tutto sull'argomento elementi di DocBook.

## Liste

DocBook ha disponibili diversi tipi di elementi a lista, dei quali, i più frequentemente usati sono i seguenti:

### *itemizedlist*

Una lista *itemizedlist* viene usata per elencare una serie di cose il cui ordine non è importantissimo:

```
<itemizedlist spacing="compact">
  <listitem><para>Le arance sono succose</para></listitem>
  <listitem><para>Le mele si dice facciano bene</para></listitem>
  <listitem><para>Molti ritengono i limoni troppo aspri</para>
  </listitem>
</itemizedlist>
```

Gli oggetti in lista sono generalmente evidenziati con un pallino nel documento finale:

- Le arance sono succose
- Le mele si dice facciano bene
- Molti ritengono i limoni troppo aspri

Senza mettere l'attributo `spacing`, viene ritenuto `normal`, che significa che viene inserito uno spazio verticale (usualmente di una linea) tra ogni oggetto in lista.

### *orderedlist*

Si usa una *orderedlist* quando si desidera evidenziare l'ordine degli elementi:

```
<orderedlist spacing="compact" numeration="loweralpha">
  <listitem><para>Sumeri 3300 aC - 1900 aC</para></listitem>
  <listitem><para>Impero Assiro 1350 aC - 612 aC</para></listitem>
  <listitem><para>Impero Persiano 600 aC - 330 aC</para>
  </listitem>
</orderedlist>
```

Per default, vengono messi numeri arabi (1, 2, 3, ...) per elencare gli oggetti, ma si può cambiare attraverso l'attributo `numeration`. Risultato:

- a. Sumeri 3300 aC - 1900 aC
- b. Impero Assiro 1350 aC - 612 aC
- c. Impero Persiano 600 aC - 330 aC

### *procedure*

Una *procedure* ha spesso lo stesso aspetto di una *orderedlist*, ma il significato è diverso: una procedura elenca una serie di *passi* da eseguire in un preciso ordine:

```
<procedure>
  <step><para>Apri la serratura</para></step>
  <step><para>Svaligia la casa</para></step>
  <step><para>Fatti arrestare</para></step>
</orderedlist>
```

Ecco come diventa l'esempio:

1. Apri la serratura

2. Svaligia la casa
3. Fatti arrestare

Dentro uno *step* si possono includere elementi *substeps*, che a loro volta possono avere più *step*.

#### *variablelist*

Una *variablelist* è fatta di *varlistentry*s, ognuno dei quali contiene un *term* seguito da un *listitem*:

```
<variablelist>
  <varlistentry>
    <term>Marcatore</term>
    <listitem>
      <para>Un testo incluso fra parentesi angolari</para>
    </listitem>
  </varlistentry>
  <varlistentry>
    <term>Elemento</term>
    <listitem>
      <para>Un marcatore iniziale, un corrispondente marcatore finale,
        e tutto quel che c'è dentro</para>
    </listitem>
  </varlistentry>
  <varlistentry>
    <term>Contenuto di un elemento</term>
    <listitem>
      <para>Tutto quel che c'è tra due marcatori corrispondenti</para>
    </listitem>
  </varlistentry>
</variablelist>
```

Risultato:

#### *Marcatore*

Un testo incluso fra parentesi angolari

#### *Elemento*

Un marcatore iniziale, un corrispondente marcatore finale e tutto quel che c'è dentro

#### *Contenuto di un elemento*

Tutto quel che c'è tra due marcatori, iniziale e finale, corrispondenti

Notare che questa lista, che elenca i vari tipi di liste, è anch'essa una *variablelist* con i nomi degli elementi (*itemizedlist*, *orderedlist*, etc.) come termini. La prossima sezione (i *Links*) consisterà di una frase iniziale seguita da una *variablelist*.

## Link

Si possono creare link a destinazioni nel proprio documento, in un altro documento del set, o su internet.

#### *link*

*link* è l'elemento generico per indirizzare ad un altro punto del documento o del set. L'attributo *linkend* deve essere sempre presente ed il suo valore deve essere l'id dell'elemento a cui ci si vuole collegare (il *link target*).

Cliccare `<link linkend="docwritehowto-introduction-it">qui</link>` per tornare all'introduzione.

Nel documento finale, la parola «qui» sarà il *punto caldo* (*hot text*), cioè: il link cliccabile che porta alla destinazione, in questo esempio all'introduzione:

Cliccare [qui](#) per tornare all'introduzione.

#### Attenzione

Per quanto `link` possa puntare dovunque nell'intero set, si dovrebbe farlo solo se la destinazione è all'interno dello stesso file PDF del link. La versione HTML è in grado di fare qualsiasi hyperlink, ma i link dei PDF non funzionano tra documenti diversi. I nostri PDF sono costituiti da un `book` o da un `article`; se la destinazione è fuori del documento è meglio usare un `ulink` (vedi oltre).

#### *ulink*

Si usa un `ulink` per puntare ad una risorsa Internet. L'attributo `url` è obbligatorio:

Cliccare `<ulink url="http://docbook.org/tdg/en/">questo link</ulink>` per leggere The Definitive Guide on DocBook (in inglese).

Le parole «questo link» genereranno un hyperlink a `http://docbook.org/tdg/en/`, come questo:

Cliccare [questo link](http://docbook.org/tdg/en/) per leggere The Definitive Guide on DocBook (in inglese).

#### *email*

Si può fare un link ad una posta elettronica con `ulink`, ma è più facile usare l'elemento `email` con cui l'indirizzo di email viene visualizzato come un link cliccabile. Questo frammento XML:

Inviare una mail a  
`<email>firebird-docs-request@lists.sourceforge.net</email>` per iscriversi.

genera il seguente risultato:

Inviare una mail a `<firebird-docs-request@lists.sourceforge.net>` per iscriversi.

Desiderando evidenziare un testo diverso da quello della email di destinazione stessa, si può usare `ulink` e l'attributo `url` col protocollo `mailto://`

#### Avvertimento

Includendo link ad indirizzi di mail, che siano essi con `email` o con `ulink`, o solo *menzionandoli* nel testo, se il testo viene pubblicato in internet, questi indirizzi di email sono esposti ai rognosi robot cacciatori di indirizzi usati dagli spammer. Questo può aumentare la quantità di porcheria che arriva a quell'indirizzo. È necessario assicurarsi sempre che il proprietario di quell'indirizzo sia d'accordo nel pubblicarlo!

#### *anchor*

Un `anchor` è un elemento vuoto che etichetta un punto esatto nel documento. Essendo completamente invisibile, non viene visto dai lettori, ma può essere utilizzato come destinazione di un link. È utile quando si vuol andare direttamente a metà o alla fine di lungo paragrafo:

`<para id="naufrago">`  
Bla bla bla bla...

```

ed ancora bla bla bla...
e poi bla bla bla, ma forse non vi ho detto bla bla bla...
Ed ora un punto interessante del paragrafo a cui voglio arrivare
direttamente:
<anchor id="salvataggio"/>Eccoci qui!
Ed altri paragrafi continuano interessanti
( come un convegno di tromboni... )
e dai, e dai, e dai...
e non finiscono più...
</para>

```

Avendo messa l'ancora (anchor), si può creare il link ad essa:

```

<link linkend="salvataggio">Vai al punto interessante</link> nel
mezzo di quel lunghissimo paragrafo.

```

Se la destinazione del link è un elemento breve, o l'inizio di un elemento, è più comodo dare all'elemento stesso un id e riferirlo nell'attributo linkend del link.

## Listati, schermate, vista letterale ed esempi

### *programlisting*

Per includere frammenti di codice sorgente nel documento, li si mette in un elemento `programlisting`. Quanto verrà scritto in un `programlisting` verrà reso identico, inclusi a capo, spazi, ecc.. Inoltre il documento sarà stampato con fonti a passo fisso. Il termine «listato sorgente», o «program listing», deve essere inteso in senso lasco: si può usare sia per frammenti di SQL come per esempi di DocBook XML. Questa guida, e specialmente la sezione sugli elementi che si sta' leggendo, è piena di `programlisting`, così è facile rendersi conto di come appaiono:

La resa di un `programlisting` è come in questa linea.

### **Importante**

Nel `programlisting` bisogna limitare la lunghezza della linea a circa 70 caratteri, altrimenti il testo esce dal lato destro del documento PDF. Lo stesso succede per tutti gli altri elementi che impongono la conservazione del formato, come `screen`, `literallayout`, ecc.

### *screen*

Si usa un elemento `screen` per mostrare ciò che un utente vede o potrebbe vedere sullo schermo di un computer in modalità testo, o in una finestra terminale. Anche in questo caso lo schema viene conservato e si usa una font a passo fisso, ma la semantica è diversa. La resa potrebbe essere del tutto diversa (oppure per niente) da un `programlisting`. Questo è un breve esempio, giusto per mostrare che succede cercando di ricostruire un modulo inesistente dell'albero del modulo del manuale (notare: il primo è un `programlisting`, il successivo è effettivamente uno `screen`):

```

<screen>
D:\Firebird>manual_incl_howto\src\build>build ugh
java version "1.4.2_01"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.4.2_01-b06)
Java HotSpot(TM) Client VM (build 1.4.2_01-b06, mixed mode)

Buildfile: build.xml

```

```
BUILD FAILED
Target `ugh' does not exist in this project.
</screen>
```

E questo è quel che ne risulta:

```
D:\Firebird>manual_incl_howto\src\build>build ugh
java version "1.4.2_01"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.4.2_01-b06)
Java HotSpot(TM) Client VM (build 1.4.2_01-b06, mixed mode)

Buildfile: build.xml

BUILD FAILED
Target `ugh' does not exist in this project.
```

### *literallayout*

*literallayout*, al pari di *screen* e *programlisting*, lascia intatto il formato ma normalmente non cambia la font, a meno di impostare l'attributo *class* a *monospaced*. È ancora più generico dei precedenti nel senso che al contenuto non viene attribuito nessun significato particolare: si può pertanto mettere qualsiasi testo di cui si voglia conservare lo schema.

Esempio:

```
<literallayout>
The Sick Rose

Oh Rose, thou art sick!
The invisible worm
That flies in the night,
In the howling storm,

Has found out thy bed
Of crimson joy,
And his dark secret love
Doth thy life destroy.

    - William Blake
</literallayout>
```

Risultato:

The Sick Rose

Oh Rose, thou art sick!  
The invisible worm  
That flies in the night,  
In the howling storm,

Has found out thy bed  
Of crimson joy,  
And his dark secret love  
Doth thy life destroy.

— William Blake

### *example*

Un *example* è un esempio formale con un titolo. Di solito gli si dà un *id* per potergli fare riferimento da altre parti del documento. L'indice degli esempi viene generato automaticamente nella stampa del documento. Spesso si possono trovare anche dei *programlisting* in un *example*, oppure *screen*, *para*, liste varie, ecc.

Ecco un esempio di *example*:

```
<example id="docwritehowto-sql-example">
  <title>Un esempio di SQL</title>
  <para>Con questo comando si possono listare tutti i record della
  tabella COUNTRY:</para>
  <programlisting>SELECT * FROM COUNTRY;</programlisting>
</example>
```

Il risultato sarà più o meno così:

#### **Esempio 1. Un esempio di SQL**

Con questo comando si possono listare tutti i record della tabella COUNTRY:

```
SELECT * FROM COUNTRY;
```

Volendo fare un *example* senza il titolo obbligatorio, si usa un *informalexample*. Gli esempi informali non sono compresi nell'indice degli esempi.

## **Table**

Non si avranno molte difficoltà a fare tabelle con DocBook, avendo già un po' di pratica con le tabelle in HTML fatte per un sito web. Sebbene differenze ci siano, le tabelle DocBook sono molto più ricche.

Una tabella consiste di un *title* e di uno o più *tgroup*, di solito uno solo basta e avanza. L'elemento *tgroup* ha un attributo obbligatorio: *cols*. Bisogna impostare il valore dell'attributo al numero di colonne desiderate nel *tgroup*. Nel *tgroup* si possono mettere elementi *thead*, *tfoot* e *tbody*. Ognuno di questi può avere una o più *row* (righe), che a loro volta possono avere tante *entry* (celle) quante specificate nell'attributo *cols*. (Si possono raggruppare le celle, ma non entriamo nel dettaglio ora.)

Questa è la struttura basilare. Segue un esempio, prima nel sorgente XML DocBook e poi la tabella risultante così come risulta nel documento. Si ignorino per il momento i vari *colspec*; sono elementi non obbligatori usati per dare un aspetto migliore.

```
<table id="docwritehowto-table-dboftheyear">
  <title>LinuxQuestions.org poll: Database of the year 2003</title>

  <tgroup cols="3">
    <colspec align="left" colname="col-dbname" colwidth="2*" />
    <colspec align="right" colname="col-votes" colwidth="1*" />
    <colspec align="right" colname="col-perc" colwidth="1*" />

    <thead>
      <row>
        <entry align="center">Database</entry>
        <entry align="center">Votes</entry>
        <entry align="center">Percentage</entry>
```

```

    </row>
</thead>

<tfoot>
  <row>
    <entry>Total</entry>
    <entry>1111</entry>
    <entry>99.99</entry>
  </row>
</tfoot>

<tbody>
  <row>
    <entry>MySQL</entry>
    <entry>405</entry>
    <entry>36.45</entry>
  </row>
  <row>
    <entry>Firebird</entry>
    <entry>403</entry>
    <entry>36.27</entry>
  </row>

  ... 5 more rows not shown here ....

</tbody>
</tgroup>
</table>

```

E questa è la tabella risultante:

**Tabella 2. LinuxQuestions.org poll: Database of the year 2003**

Database	Votes	Percentage
MySQL	405	36.45
Firebird	403	36.27
Postgres	269	24.21
Oracle	25	2.25
Berkeley DB	4	0.36
Sybase	3	0.27
DB2	2	0.18
<b>Total</b>	<b>1111</b>	<b>99.99</b>

Naturalmente, questi sono i risultati di una vera rilevazione fatta a LinuxQuestions.org. Come si può vedere, se solo tre persone avessero votato per Firebird si sarebbe vinto. Se qualcuno le conosce, per favore indicatele al nostro Inquisitore Capo. Desidera *tanto* fargli un certo «discorso»... #

Le tabelle vengono automaticamente messe in un loro indice. una `informaltable` ha la stessa struttura di una `table` ma non richiede titolo e non viene inclusa nell'indice. Per poter innestare delle tabelle (tabelle in tabelle), o si usa una `table/informaltable` all'interno di una `entry`, oppure una `entrytbl` invece di una `entry`.

Le tabelle hanno molte altre caratteristiche rispetto a quanto mostrato, ma al momento questo basta, e le altre cose si possono esplorare secondo necessità.

## Tabelle HTML

Dalla versione 4.3 di DocBook si possono riempire le tabelle come in HTML, con `tr` invece delle `row`, e `td/th` invece degli elementi `entry`. Il motivo? Ci sono due situazioni dove è utile usare le tabelle HTML:

- Avendo già la tabella HTML pronta, non si deve perdere tempo nel convertirla;
- Volendo usare differenti colori di sfondo nella tabella, è più semplice. Questo si può fare anche in DocBook, ma solo con *Istruzioni di Elaborazione*, una per ciascun oggetto di ogni elemento che necessita di un colore specifico. Con le tabelle in HTML si può usare l'attributo `bgcolor` dell'oggetto.

Una tabella HTML non può avere `tgroup`; si mettono i `tr` o direttamente nella tabella o negli elementi `thead` / `tfoot` / `tbody` che sono diretti figli della tabella. Inoltre ha un attributo `caption` invece di un `title`. (Ovviamente, una `informaltable` non nè l'uno nè l'altro.)

Ecco il sorgente di una tabella HTML:

```
<table bgcolor="blue" border="1">
  <caption align="bottom">Una tabella in stile HTML</caption>

  <tr bgcolor="#FFE080">
    <th>Prima colonna</th>
    <th bgcolor="#FFFF00">Seconda colonna</th>
  </tr>
  <tr align="center">
    <td bgcolor="orange" colspan="2">Cella di tabella che occupa due
      colonne</td>
  </tr>
  <tr>
    <td bgcolor="#00FFC0">Eccomi qui!</td>
    <td align="right" bgcolor="#E0E0E0" rowspan="2" valign="bottom">
      Adesso me ne vado qua!</td>
  </tr>
  <tr>
    <td bgcolor="#FFA0FF">Una riga ancora...</td>
  </tr>
</table>
```

Ed ecco come viene:

Prima colonna	Seconda colonna
Cella di tabella che occupa due colonne	
Eccomi qui!	Adesso me ne vado qua!
Una riga ancora...	

Tabella 3. Una tabella in stile HTML

Non tutti gli elementi e gli attributi di tabella sono supportati dai nostri fogli di stile. Per esempio, non vengono considerate le proprietà specificate negli elementi `col` e `colgroup`. Si possono invece specificare negli elementi `td` e `th`, oppure estendere i fogli di stile!

### Nota

In XMLMind, si possono creare tabelle HTML solo dal menù aperto col pulsante «Aggiungi tabella» sulla toolbar. Dal pannello di edit si possono aggiungere solo normali tabelle DocBook.

## La restituzione di tabelle grandi in PDF

Le tabelle DocBook appartengono ad un gruppo chiamato *elementi formali*. Gli elementi formali sono incluse in indici generati automaticamente (tabelle, figure, ecc.); se un elemento formale non ha un attributo `id`, il foglio di stile gliene attribuisce uno. Inoltre, il modello che genera il file XSL-FO (è una fase intermedia per ottenere il PDF) da ad ogni oggetto formale l'attributo `keep-together.within-page="always"` per impedire che ci siano salti pagina all'interno dell'oggetto. Questo di solito va bene, ma che succede se l'oggetto non ci stà in una pagina? Fino a poco fa, abbiamo usato Apache FOP 0.20.5 per rendere il file XSL-FO in PDF. Questo processore semplicemente ignorava l'attributo `keep-together` se l'oggetto era troppo grande. Ma la versione attuale (Apache FOP 0.93) lo applica *sempre*. Se l'oggetto è troppo grande, il risultato è che viene troncato o distrutto in qualche altro modo per farlo rientrare nella pagina. Questa è una caratteristica, non un problema, per cui non ci si può lamentare.

Ci sono due modi per aggirare il problema se una tabella diventa troppo grande per stare in una pagina:

1. Se la tabella non ha bisogno di un titolo e non interessa che sia inclusa nella lista delle tabelle, si può usare al suo posto una `informaltable`.
2. Si può inserire una *istruzione di elaborazione* all'inizio della tabella:

```
<table frame="all" id="ufb-about-tbl-features">
  <?dbfo keep-together='auto'?>
  <title>Summary of features</title>
```

In XMLMind, si fa così:

1. Mettere il cursore da nel titolo oppure selezionare tutto l'elemento del titolo.
2. Dal menù, selezionare *Modifica -> Elabora Istruzione -> Inserisci Istruzione di Elaborazione Prima*. Una linea in cui il testo sarà verde apparirà sopra il titolo.
3. Digitare `keep-together='auto'` su quella linea.
4. Con il cursore ancora su quella linea, nel menù selezionare *Modifica -> Elabora Istruzione -> Cambia l'Obiettivo dell'Istruzione di Elaborazione...*. Compare una finestra di dialogo.
5. Nella finestra, modificare il testo `target` in `dbfo` e confermare cliccando su OK.

Naturalmente si può fare lo stesso con tabelle più piccole se le si vuole rendere interrompibili. L'istruzione opposta, `<?dbfo keep-together='always'>`, impedisce le interruzioni delle tabelle informali, cioè in `informaltable`. Bisogna accertarsi che l'elemento stia in una pagina prima di farlo, naturalmente.

## Immagini

Per includere un'immagine, si usa un `mediaobject` contenente un `imageobject` che a sua volta contiene un elemento `imagedata`:

```
<mediaobject>
  <imageobject>
    <imagedata align="center" fileref="images/services.png"
      format="PNG" />
  </imageobject>
</mediaobject>
```

Ci si potrebbe meravigliare della necessità di avere tre elementi innestati per includere un'immagine. Ci sono buone ragioni, ma non ve lo dirò ;- ) — non ci interessano. Tutto quello che c'è da sapere è come si fa..

Indipendentemente da dov'è l'immagine relativamente al sorgente DocBook, l'attributo `fileref` deve essere *sempre* del tipo `images/filename.ext`. Questo perchè, sia per l'HTML che per il FO, i file delle immagini sono tutti copiati nella subdirectory `images` della directory di uscita. (Il file FO è una forma intermedia, ed una volta convertita in PDF, l'immagine è inclusa nel file.)

Se il `fileref` non è giusto dal punto di vista del file sorgente, l'immagine non si vedrà in XMLMind. Se da' fastidio, in Linux si può creare un link simbolico alla directory delle immagini mentre in Windows bisogna copiare le immagini nello stesso sottodirettorio del `fileref`, cioè nel nostro caso `images`. Sembra che creare dei link in Windows non risolva il problema. Fate questo solo sulla vostra copia privata, non committate archivi di immagini duplicate al repository! (vedi anche la nota qui sotto).

Un `mediaobject` è formattata come blocco separato. Per fare in modo che l'immagine sia allineata nel testo usare invece un `inlinemediaobject`; gli elementi interni (cioè `imageobject` e `imagedata`) restano gli stessi.

#### Nota per i traduttori

Per i traduttori (ovvero: non fate come me # ehm...) Ogni immagine che non viene modificata o sostituita da una versione tradotta, non va copiata nelle immagini della propria lingua. A partire dal gennaio 2006, gli strumenti di ricostruzione prima cercano un'immagine nella directory della lingua (ad esempio `manual/src/docs/firebirddocs-it/images`), e, se non la trovano, cercano in `manual/src/docs/firebirddocs/images`. In questo modo, usando l'immagine originale, non è necessario sprecare spazio sul CVS duplicandola. La stessa cosa si applica a tutti gli altri insiemi: se un'immagine riferita dalle Note di Rilascio spagnole non si trova in `rlsnotes-es/images`, viene usata, se esiste, quella in `rlsnotes/images`. Questo purtroppo non funziona *attraverso* gli insiemi, cioè da un insieme fare riferimento ad una immagine di un altro insieme.

## Avvertimenti

DocBook dispone di vari marcatori per blocchi di testo come note, avvisi, consigli, ecc. Nel documento risultante tipicamente tali blocchi risultano indentati ed evidenziati con una icona o una parola che esplicita lo scopo. Questi marcatori sono, in ordine alfabetico:

```
<caution>, <important>, <note>, <tip>, e <warning>
```

e significano, nell'ordine: «attenzione», «importante», «nota», «suggerimento», «avviso». Questo è un `<tip>` di esempio; gli altri si usano nello stesso identico modo:

```
<tip>
  <para>Inserendo un elemento di attenzione, importante, nota,
    suggerimento o avviso nel testo, non iniziate con le parole
    attenzione, importante, nota, suggerimento o avviso, perchè
    queste parole sono usualmente inserite in modo automatico
    dal sistema di generazione.
```

```
</para>
</tip>
```

E questo è il risultato:

### Suggerimento

Inserendo un elemento di attenzione, importante, nota, suggerimento o avviso nel testo, non iniziate con le parole «attenzione», «importante», «nota», «suggerimento» o «avviso», perchè queste parole sono usualmente inserite in modo automatico dal sistema di generazione.

Si può notare che le parole *attenzione*, *importante* ecc. hanno un aspetto diverso dal resto del testo: come mai? A dir la verità sono state incapsulate con alcuni marcatori speciali (prima con `<sgmltag>`, e poi con `<literal>`) per dare quell'aspetto. Questo però renderebbe l'originale XML alquanto illeggibile, così si è deciso di rimuoverli dal sorgente di esempio.

Si può dare all'avvertimento un opzionale titolo, con «`title`». Non facendolo viene dato un titolo di default, nella lingua del documento.

Per evidenziare una parte di testo da quanto gli sta' intorno, senza attribuirgli uno di questi marcatori, bisogna usare un `<blockquote>`.

## Testate di paragrafo

Volendo creare una testata ad un paragrafo, oppure un titolo senza creare una sottosezione, ci sono alcune possibilità.

### *bridgehead*

Un *bridgehead* è un titolo flottante tra paragrafi, non associato all'inizio di un capitolo o sezione. L'attributo `renderas` determina come sarà l'aspetto finale.

```
<para>Il biliardo all'italiana si gioca oggi sul biliardo
internazionale, che è un tavolo privo di buche di 2,84 m x 1,42 m;
si usano tre biglie e cinque piccoli birilli. Le biglie
destinate ai giocatori sono di colore bianco o giallo,
il pallino di colore rosso. I birilli, alti circa 2,5 cm,
sono di colore bianco ad eccezione di quello centrale, che è
di colore rosso.</para>
```

```
<bridgehead renderas="sect5">Ma come si gioca?</bridgehead>
```

```
<para>I birilli devono essere disposti prima di ogni tiro in posizioni
ben precise, al centro del tavolo, a formare quello che si suole
definire castello. Bisogna colpire con la propria biglia quella
dell'avversario cercando possibilmente di indirizzare quest'ultima
sui birilli, alla cui caduta è attribuito un punteggio.</para>
```

Il sorgente qui sopra ha questo aspetto:

Il biliardo all'italiana si gioca oggi sul biliardo internazionale, che è un tavolo privo di buche di 2,84 m x 1,42 m; si usano tre biglie e cinque piccoli birilli. Le biglie destinate ai giocatori sono di colore bianco o giallo, il pallino di colore rosso. I birilli, alti circa 2,5 cm, sono di colore bianco ad eccezione di quello centrale, che è di colore rosso.

## Ma come si gioca?

I birilli devono essere disposti prima di ogni tiro in posizioni ben precise, al centro del tavolo, a formare quello che si suole definire castello. Bisogna colpire con la propria biglia quella dell'avversario cercando possibilmente di indirizzare quest'ultima sui birilli, alla cui caduta è attribuito un punteggio.

Si è liberi nella scelta del livello di `renderas`, ma la scelta più logica sarebbe normalmente quella del corrente livello di sezione più (almeno) uno.

### *formalpara*

Un `formalpara` è un paragrafo con un titolo. L'aspetto dipende dal foglio di stile, nel nostro caso trasforma il titolo indentando tutto.

```
<formalpara>
  <title>Amor materno:</title>
  <para>Questo è l'amore che ha tua madre per te, da non confondere
    con l'amor paterno, fraterno o altrui.</para>
</formalpara>
```

Il risultato viene visualizzato come segue:

**Amor materno:** Questo è l'amore che ha tua madre per te, da non confondere con l'amor paterno, fraterno o altrui.

Viene aggiunto un punto alla fine del titolo, a meno che non termini già con un carattere di interpunzione.

## Vari elementi in linea

Per concludere la sezione sugli elementi di DocBook, mostriamo ora un certo numero di elementi in linea (*inline elements*). Sono detti «in linea» perchè non interrompono il fluire del testo. Usando ad esempio un elemento `emphasis`:

```
Non chiamarmi <emphasis>mai più</emphasis> ciccione!
```

Il risultato è:

Non chiamarmi *mai più* ciccione!

Le parole «mai più» sono enfatizzate, e comunque prende posto nella frase. Abbiamo già visto altri elementi in linea: i vari tipi di link. Altri elementi, come `table`, `warning`, `blockquote` e `programlisting`, sono sempre mostrati come blocchi e posti separatamente dal testo intorno, perfino se li metti «in linea» nel sorgente XML. Non sorprende che siano chiamati elementi a blocchi o brevemente «blocchi» (*block elements*). I blocchi contengono spesso elementi in linea, ovviamente non è possibile l'inverso.

Bene, cominciamo a vedere qualcosa sugli elementi in linea. Ci saranno esempi, sia sorgenti XML che il risultato in uscita, per molti di essi:

### *filename – command – application – envar*

Si usa il marcatore `filename` per evidenziare nomi di file nel senso più generico possibile. Gli attributi possono opzionalmente indicare se il file è una libreria, un link, una directory, ecc.

```
Mettere il documento nella directory <filename
class="directory">src/docs/firebirddocs</filename>.
```

che si leggerà:

Mettere il documento nella directory `src/docs/firebird/docs`.

`command` e `application` si usano entrambi per programmi eseguibili. `command` si usa per piccoli programmi e comandi interni; il suo contenuto dovrebbe essere il comando esatto così come dato in una linea di comando; `application` è usato di solito per grandi programmi e non necessariamente deve essere il nome dell'eseguibile. Entrambi possono riferire lo stesso programma:

```
Digitare <command>netscape&lt;/command> in una finestra terminale
per lanciare il <application>Netscape Navigator</application>.
```

Questo apparirà così:

Digitare **netscape&** in una finestra terminale per lanciare Netscape Navigator.

`envvar` invece serve per indicare una variabile d'ambiente.

*subscript – superscript*

È immediato intuire cosa fanno:

```
Dopo la scoperta della formula  $e = mc^2$ ,
avevo proprio voglia di un bicchiere di  $H_2O$  !
```

*Output:* Dopo la scoperta della formula  $e = mc^2$ , avevo proprio voglia di un bicchiere di  $H_2O$  !

*varname – constant – database*

L'uso di `varname` (nome di variabile) e `constant` (costante) dovrebbero essere ovvii. Il marcatore `<database>` non è stato previsto solo per i database, ma anche per gli oggetti di database:

```
La tabella <database class="table">COUNTRY</database> ha due campi:
<database class="field">COUNTRY</database> e
<database class="field">CURRENCY</database>.
```

*Risultato:* La tabella COUNTRY ha due campi: COUNTRY e CURRENCY.

*function – parameter – returnvalue*

Anche questi sono autoesplicativi, credo.

```
La funzione <function>log</function> ha due parametri
<parameter>a</parameter> e <parameter>b</parameter>.
```

*Risultato:* La funzione `log` ha due parametri `a` e `b`.

*prompt – userinput – computeroutput*

Si usa `prompt` per una stringa che chiede all'utente di inserire del testo; `userinput` si riferisce al testo inserito dall'utente (non necessariamente ad un prompt!); `computeroutput` è il testo mostrato dal computer:

```
Digitare <userinput>guest</userinput> al prompt
<prompt>login:</prompt> ed il server risponderà con
<computeroutput>Benvenuto, utente guest</computeroutput>.
```

*Risultato:* Digitare **guest** al prompt `login:` ed il server risponderà con `Benvenuto, utente guest`.

### *keycap*

Il testo scritto su di un tasto della tastiera, oppure il suo nome comune:

```
Digitare il tasto <keycap>Del</keycap> per cancellare il messaggio,  
oppure <keycap>SPACE</keycap> per confermarlo.
```

*Risultato:* Digitare il tasto **Del** per cancellare il messaggio, oppure **SPACE** per confermarlo.

### *sgmltag*

Questo elemento è stato usato moltissimo in questa guida: evidenzia marcatori SGML e XML, elementi, attributi, entità, ecc.:

```
Se riguarda una directory, impostate l'attributo  
<sgmltag class="attribute">class</sgmltag> dell'elemento  
<sgmltag class="element">filename</sgmltag> a  
<sgmltag class="attvalue">directory</sgmltag>.
```

*Risultato:* Se riguarda una directory, impostate l'attributo `class` dell'elemento `filename` a `directory`.

Altri possibili valori per `sgmltag.class` sono: `starttag`, `endtag`, `emptytag`, e `genentity` (per una entità).

### *emphasis – citetitle – firstterm*

Si usa `emphasis` per evidenziare genericamente delle parole, `citetitle` per titoli di libri ecc., e `firstterm` introducendo un nuovo termine o concetto ai lettori:

```
Si usa <firstterm>DocBook XML</firstterm> per la nostra  
documentazione Firebird. Segue una breve introduzione;  
<emphasis>per favore</emphasis>, leggetela attentamente!  
Per avere ulteriori informazioni al riguardo, comprare il  
libro <citetitle>DocBook – The Definitive Guide</citetitle>.
```

*Risultato:* Si usa *DocBook XML* per la nostra documentazione Firebird. Segue una breve introduzione; *per favore*, leggetela attentamente! Per avere ulteriori informazioni al riguardo, comprare il libro *DocBook – The Definitive Guide*.

### *quote – literal*

Usare `quote` per virgolettare in linea (a differenza di `blockquote`). Le virgolette vengono messe automaticamente. Usando `quote` invece di digitare le virgolette (che sarebbe comun que perfettamente legale) ha il vantaggio che si possono cambiare il tipo di virgolettatura globalmente nel foglio di stile volendolo. Inoltre si diversificano per lingua:

```
<para>Queste sono <quote lang="en">virgolette inglesi</quote>  
e queste sono <quote lang="fr">virgolette francesi</quote>.  
Invece sono così <quote lang="it">quelle italiane</quote>.</para>
```

*Risultato:* Queste sono “virgolette inglesi” e queste sono « virgolette francesi ». Invece sono così «quelle italiane».

Non si dovrebbe usare l'attributo `lang` nell'elemento `quote` nei propri documenti. L'attributo `lang` dell'elemento radice assicura che vengano usate le virgolette giuste. Nella traduzione del vostro documento, e quindi nella modifica alla radice dell'attributo `lang`, verranno così automaticamente usate le virgolette del linguaggio della traduzione. Ho usato l'attributo in questo caso per mostrare la differenza, e per assicurarmi che le differenti virgolette rimarranno indipendentemente dalla traduzione.

Un `literal` è una parola o un frammento di testo da prendere alla lettera. È un elemento generico, spesso usato per evidenziare tipograficamente certe parole:

```
L'ultima parola del vocabolario italiano è  
<literal>zuzzurellone</literal>, credo.
```

*Risultato:* L'ultima parola del vocabolario italiano è zuzzurellone, credo.

Ma questi elementi vanno usati sempre e comunque dove è appropriato? Be', facendolo, il documento sarà sicuramente di molto arricchito; sarà più facile trovare i nomi di file, ad esempio, o generare un indice di tutte le applicazioni menzionate. D'altra parte, questi elementi semantici sono talmente tanti (quelli qui elencati sono solo *alcuni*) che applicandoli dovunque abbia senso, si rischia di impazzire prima di finire il documento. Siccome lo sconsigliamo caldamente, se proprio volete impazzire, be', fatelo solo *dopo* aver committato il testo :-)

Così, in generale, è meglio non appesantire troppo con questi elementi in linea; usarli solo quando ha veramente senso e meglio non abusarne.

## Considerazioni finali sugli elementi

È facile notare come nei documenti restituiti (ne state leggendo uno, ameno che non stiate consultando la versione XML) diversi elementi hanno lo stesso aspetto: un `filename`, un `literal` e una `application` possono avere lo stesso preciso aspetto tipografico, così come `emphasis`, `firstterm` e `citetitle`.

All'ora qual'è l'utilità di tutti questi differenti marcatori? Perché non usare solo quelli strettamente necessari, come `emphasis` e `literal`, se devono sembrare ad ogni modo uguali? Ci sono due buonissime ragioni per non farlo:

- Primo, se eliminiamo molti dei nostri marcatori in linea lasciando, ad esempio, solo `emphasis` e `literal`, perdiamo la semantica. Ricordare che DocBook XML è incentrato sulla struttura e la semantica. `firstterm` and `citetitle` possono *sembrare* identici a `emphasis` in stampa, ma non *sono* la stessa cosa. Il sorgente XML lo sa, anche se non lo mostra. Questa è una informazione utile, e non la dobbiamo perdere.
- Secondo, è possibile adattare i propri fogli di stile per ogni tipo di elemento individualmente. Appena si decide che `firstterm` deve avere un diverso aspetto da `citetitle`, possiamo farlo, ma *solo* se sono già diversi i loro marcatori, non se sono entrambe `emphasis` nel sorgente XML.

Questo conclude la sezione su DocBook. Con le informazioni date fin qui, si è in grado di scrivere documenti DocBook XML per il progetto Firebird. Naturalmente, usando un editor XML dedicato, cosa caldamente consigliata, sarebbe meglio anche consultarne la documentazione per imparare ad usarlo, cosa che questa guida non insegna a fare.

## Lingua e stile

Dopo il fiume di informazioni delle sezioni precedenti, si può volgere lo sguardo ad altri aspetti importanti nello scrivere la documentazione: la lingua e lo stile (in questa sezione) ed i diritti d'autore (nella successiva sezione).

## Lingua

La comunità di Firebird è molto variegata, e costituita da persone di diversa lingua madre. Scrivendo il documento in una lingua diversa dalla propria, è possibile fare errori. Ciò non è catastrofico, ma almeno bisogna cercare di ridurre al minimo tali errori. Si possono adottare alcune strategie per aiutarsi nel compito:

- Usare un dizionario! Semplice, efficace, e felicemente non tecnologico.
- Essendo incerti sulla scrittura di una parola, o fra diverse possibili versioni di un modo di dire, si può cercare in rete fra le alternative e controllate le frequenze. Magari seguendo i link dei risultati della ricerca per vedere come viene usata la parola o l'espressione nella lingua viva.
- Far controllare il vostro testo da qualcuno di lingua madre e correggetelo ove necessario.

## Stile

Questa non è una vera e propria guida sullo stile, perchè non saprei proprio come scriverne una. Solo alcune linee guida e suggerimenti:

- Cercare di scrivere col semplice linguaggio di ogni giorno, ovunque possibile. Evitare le parole difficili se esiste un'alternativa più usuale e familiare.
- Evitare le frasi lunghe (oltre le 25 parole) se possibile; specialmente evitare due frasi molto lunghe una dopo l'altra.
- (N.d.T. tradotto *letteralmente*; è bene leggere i suggerimenti per le traduzioni in italiano qui sotto) Fare attenzione ai costrutti con doppie e triple negazioni («Non posso negare che non mi dispiace») e le voci passive («Attenzione deve essere fatta...»). Non è che sono da evitare ad ogni costo, ma possono rendere una frase più difficile da capire. Nel caso eliminare le doppie negazioni («Sono compiaciuto») e usare una voce attiva («Fate attenzione...»).
- Usare liste per elencare un certo numero di oggetti in serie, ad esempio:
  - Un insieme di suggerimenti e trucchetti.
  - Una serie di esempi (come questo).
  - Passi da seguire in una procedura.
  - Soluzioni alternative ad uno stesso problema.

Nel caso ci fossero solo un esiguo numero di termini brevi, invece è meglio usare una frase lineare come: «Mia madre ama tre uomini: John, Dick, e Dave.»

- È meglio non adoperare troppi punti esclamativi o punti di domanda. Sono proprio brutti!!!! Non siete d'accordo???
- **Per i traduttori dall'inglese in italiano - aggiunto nella sola versione italiana.** Basandomi sulla mia (invero misera) esperienza, le linee guida che posso dare a chi si accingerà ad aiutarmi a tradurre dall'inglese in italiano questi manuali, si basano su queste poche note. Ricordarsi che la seconda persona plurale inglese non è consigliabile tradurla letteralmente (troppo colloquiale): in italiano esistono varie alternative. Usare ad esempio l'infinito: («Use lists to enumerate» diventa «Usare le liste per elencare»), oppure una frase impersonale («Don't overuse exclamation marks» si può tradurre con «È meglio non adoperare troppi punti esclamativi»). Altre volte aiuta il gerundio («If you write your documentation» eccolo tradotto in «Scrivendo

il documento»). Altra cosa difficile da evitare in un buon italiano sono le rime: meno ce ne sono, migliore e più leggibile risulta il testo; provare per credere. Le rime possono essere in fine parola ma anche all'inizio, e sono più difficili da stanare: si trovano rileggendo ad alta voce. Confrontando questo documento in inglese ed in italiano, si possono trovare anche altre soluzioni: talvolta è stato necessario girare o cambiare tutta una frase perchè una sua traduzione troppo letterale non mi «suonava» proprio.

## Il blocco dello scrittore.

Talvolta si conosce quello che si vuol dire, e ci sono anche le parole giuste, ma non si riesce a cominciare la frase: proprio non si riesce a farle *andare*. È veramente frustrante e può proprio bloccare il procedere del testo per molti minuti. E la frustrazione risulta ancora maggiore sapendo cosa si vuol dire ai lettori, ma apparentemente non si è in grado di scrivere una frase decente. Dopo una serie di esperienze da panico, ho sviluppato la seguente strategia (anche se non credo di essere stato il primo):

1. Scrivere ciò che si deve dire in frasi sparse e frammentarie senza preoccuparsi né dello stile né che il tutto potrà sembrare orrendo. Scrivere proprio quello che si deve dire al lettore ed assicurarsi che ci sia tutto e nel giusto ordine. Se si nota, facendo questo, di essere insicuri di qualcosa, evidenziarlo proprio in quel punto. Fatelo in modo che escano dal testo intorno, ad esempio <<tipo questo>> oppure !QUALCOSA DEL GENERE!

Questo può trasformare il testo in qualcosa come:

CVS significa Concurrent Versions System (<<controllare!>>). Obiettivo: gestire le versioni del software. Si può usare da soli o in gruppo. Ci vuole un client CVS per usarlo. Un client CVS è un programma con cui si accede ad una repository (<<spiegare il termine?>>) di un CVS. Per sapere se un CVS è installato nel proprio sistema, digitare «cvs» sulla linea di comando. Se non c'è, vai a questo sito per scaricarlo ... [ecc., ecc.]

2. Gestire prima ogni cosa in evidenza. Cioè *Controllare* se CVS davvero significa Concurrent Versions System (giusto). *Decidere* se davvero conviene spiegare il termine «CVS repository» a quel punto (e si dovrebbe farlo).
3. A questo punto, ripassando il paragrafo, provare a rendere il testo più scorrevole dovunque possibile. Ci sono possibilità che sia molto più facile del previsto!
4. Se sembra ancora un po' grezza, non importa: meglio grezza e chiara che scoordinata ed incomprensibile. Magari, non si sa mai, ripassandoci in un secondo tempo, si riesca a renderla più carina.

Questo sistema per me funziona bene e, come si dice, bloccandovi qualcosa, «tentar non nuoce».

## Problemi di diritti d'autore

Ogni volta che si parla di copyrights e diritti d'autore, viene voglia di dire «ma che noia!», ma è una cosa importante e va letta attentamente.

### Nota di traduzione

Questa parte della guida è stata genericamente tradotta ma non è stata nè interpretata nè confrontata con la legislazione italiana e comunitaria vigenti: scusate, ma non è la mia materia. I lettori sono pregati di comunicarmi eventuali discrepanze si cercherà di correggere il tiro. Ad ogni modo, per quanto riguarda la licenza d'uso e le note relative, fa fede l'originale del testo in inglese.

## Usare il materiale scritto da altri

Scrivendo i propri testi, si possono consultare tutti i tipi di altra documentazione esistenti – e si dovrebbe, desiderando ottenere il miglior risultato possibile. Ogni informazione trovata in manuali pubblicamente disponibili di terzi, guide utente, manuali d'istruzione, ecc. può essere usato nel proprio documento, ma è importante non confondere *informazione* con *testo alla lettera*. Il testo del lavoro altrui non può essere copiato ed incollato così com'è nel proprio documento, a meno che non ci sia un esplicito permesso del detentore dei diritti che ci permetta di farlo.

Volendo usare uno scritto altrui, bisogna controllare le note sui diritti d'autore che lo accompagnano. Se non c'è, il lavoro è automaticamente protetto dalla convenzione di Berna e pertanto è *illegale* copiarlo, anche in minima parte. Questo è vero anche quando il documento è disponibile liberamente! Anche se non è obbligatorio pagare per avere un documento non significa che si possono liberamente copiare parti del suo contenuto e ripubblicarle come proprie nel proprio documento.

## I manuale di Interbase della Borland

La documentazione della versione 6 beta di Interbase, sebbene di pubblico dominio, non sono parte del pacchetto Interbase che è stato reso open-source nel Luglio del 2000. Abbiamo chiesto a Borland molte volte di poter usare questi documenti «come se fossero parte del Licenza Pubblica Interbase» (InterBase Public License), ma non si sono mai curati di rispondere. Pertanto si è liberi di usare quell'insieme di documenti come sorgente delle informazioni, ma non si deve copiarne letteralmente il testo.

## I documenti di PostgreSQL

PostgreSQL è un altro database open source abbastabza diffuso, che somiglia in molti punti a Firebird, anche se esistono molte differenze. In funzione della documentazione che si intende scrivere, può essere utile basarsi su quella già esistente per PostgreSQL. È bene notare che, usando il materiale PostgreSQL, è OBBLIGATORIO include la loro nota sui diritti d'autore nel documento!

Qui si trova la homepage della documentazione PostgreSQL:

<http://www.postgresql.org/docs/>

La licenza d'uso e più recente si trova attualmente a:

<http://www.postgresql.org/about/licence>

Una cosa carina della documentazione di PostgreSQL è che è scritta in DocBook, proprio come la nostra. Tuttavia usano DocBook SGML e non XML, così potrebbero essere necessari un po' di aggiustamenti. I sorgenti DocBook SGML possono essere trovati qui:

<http://developer.postgresql.org/cvsweb.cgi/pgsql-server/doc/src/sgml/>

Oppure si può fare il check out dell'intero albero CVS, documenti e tutto quanto il resto. Per istruzioni, andate a:

<http://developer.postgresql.org/docs/postgres/cvs.html>

## Proprietà dei diritti d'autore e PDL

Contribuendo al sottoprogetto documentazione di Firebird, il lavoro sarà incluso nella repository di dominio pubblico su SourceForge. Nel gennaio del 2005, il gruppo di lavoro sulla documentazione decise di rilasciare quanto sviluppato sotto la *Public Documentation License* (PDL, licenza di documentazione pubblica). La licenza PDL significa che si mantengono i propri diritti d'autore, ma si rinuncia a certi diritti lasciandoli alla collettività:

- *Uso libero*: chiunque può usare e distribuire il lavoro, sia gratis che a pagamento, purchè lasci intatte le note della licenza d'uso allegate.
- *Diritti di modifica*: chiunque può modificare e redistribuire il lavoro, purchè ogni versione modificata mantenga la stessa licenza PDL, e resti intatta la licenza originale documentando tutte le modifiche.
- *Lavori più grandi*: tutti possono incorporare il documento (originale o modificato) in un documento più grande. Significa che il lavoro più grande non deve essere necessariamente interamente rilasciato sotto la PDL, ma le clausole della licenza devono essere completate con la licenza PDL per le parti relative.

La cosa carina della PDL è che attribuisce gli stessi diritti e restrizioni sull'uso della documentazione che le licenze IPL e IDPL (cioè le licenze del codice di Firebird) rilasciano sull'uso del codice sorgente di Firebird. Per il testo completo della licenza, seguite i link nella nota di licenza alla fine; il sorgente DocBook è in `src/docs/firebirddocs/licenses.xml`

## Applicare la PDL al proprio lavoro

Per rilasciare il proprio lavoro sotto la PDL, bisogna aggiungere un elemento `appendix` intitolato *License Notice*, con questo testo:

The contents of this Documentation are subject to the Public Documentation License Version 1.0 (the «License»); you may only use this Documentation if you comply with the terms of this License. Copies of the License are available at <http://www.firebirdsql.org/pdfmanual/pdl.pdf> (PDF) and <http://www.firebirdsql.org/manual/pdl.html> (HTML).

The Original Documentation is `_TITLE OF THE WORK_`.

The Initial Writer of the Original Documentation is `_INITIAL AUTHOR'S NAME_`.

Copyright (C) `_YEAR(S)_`. All Rights Reserved. Initial Writer contact(s): `_EMAIL OR OTHER CONTACT ADDRESS(ES)_`.

Ovviamente tutto ciò che è stato scritto `_COME QUESTO_` deve essere sostituito. Se non siete l'autore originale, dovete lasciare la nota originale intatta e aggiungere quanto segue:

Contributor(s): `_NAME(S) + SHORT DESCRIPTION (COUPLE OF WORDS) OF CONTRIBUTION_`.

Portions created by `_CONTRIBUTOR'S NAME_` are Copyright (C) `_YEAR(S)_`. All Rights Reserved. Contributor contact(s): `_EMAIL OR OTHER CONTACT ADDRESS(ES)_`.

Ci possono essere più sezioni di contributi in queste note della Licenza.

## Aggiungere una cronologia

Se il contributo è più consistente di un semplice cambio o aggiunta «una tantum», si può aggiungere prima o dopo la License Notice una *appendix* chiamata *cronologia documento* (che sarebbe l'elenco cronologico delle revisioni e modifiche al documento o, in inglese, *Document History*). Se c'è già quel tipo di appendice, vi si aggiunge comunque la descrizione dei propri contributi. Notare che anche se c'è già la cronologia, va aggiunta comunque una sezione dei contributi alle note di Licenza, ma si può mettere semplicemente un «vedi nella lista delle revisioni» al posto della breve descrizione.

Anche l'autore iniziale del documento deve includere una cronologia nella prima versione del documento, che serva da punto iniziale per le revisioni future. Nell'esempio che segue notate l'uso del primo elemento `revision`.

Punto chiave della cronologia è l'elemento `revhistory` con i suoi figli:

```
<revhistory>
  <revision>
    <revnumber>1.0</revnumber>
    <date>12 sett 2005</date>
    <authorinitials>PV</authorinitials>
    <revdescription>
      <para>Prima versione</para>
    </revdescription>
  </revision>
  <revision>
    <revnumber>1.1</revnumber>
    <date>5 dic 2005</date>
    <authorinitials>PV</authorinitials>
    <revdescription>
      <para>Aggiunte informazioni su COALESCE</para>
      <para>Corretti alcuni errori di scrittura</para>
    </revdescription>
  </revision>
</revhistory>
```

Si prega di abbreviare sempre il nome del mese nell'elemento `date`, poichè la colonna della data nel PDF è abbastanza stretta.

L'esempio di cronologia che segue (come risultato, non è il sorgente!) usa un elemento `revhistory`. Notare il riferimento all'albero CVS: si è legalmente obbligati ad identificare e datare tutte le modifiche. Ma poichè già il CVS lo fa, si può semplicemente avvisare l'utente e dargli una cronologia meno particolareggiata e più snella nel documento.

La cronologia precisa è registrata nel modulo `manual` nel nostro albero CVS; vedere a [http://sourceforge.net/cvs/?group\\_id=9028](http://sourceforge.net/cvs/?group_id=9028)

### Diario delle Revisioni

1.0	2003	IBP	Prima pubblicazione della Guida introduttiva rapida.
1.x	giu 2004	IBP	Donata al progetto Firebird dalla IBPhoenix.
2.0	2004	PV	Riportata al livello di Firebird 1.0 Aggiunta la sezione Classic confronto Superserver.

Riorganizzata e corretta la Tabella delle locazioni disco (Disk Locations Table).  
Aggiunte (nuove) immagini di schermate.  
Aggiornata e completata con le informazioni sui servizi nel Pannello di Controllo.  
Aggiunti altri esempi su «Espressioni con valori NULL».  
Altre piccole correzioni e aggiunte.

Apredo il sorgente DocBook di questa guida (`src/docs/firebirddocs-it/docwriting-howto-it.xml`) con l'editor XML preferito, si può facilmente copiare sia la cronologia sia le note di Licenza ed incollarle in un nuovo documento. Attenzione a non copiare gli esempi sopra, ma vanno copiati proprio le appendici alla fine del documento e poi si possono far aderire, modificandole, alle esigenze del proprio lavoro.

## Un avviso sui diritti d'autore (copyright) posto all'inizio

Le note di licenza e la cronologia sono posti entrambi alla fine del documento. Per manifestare i propri diritti d'autore fin dall'inizio, si può includere una breve nota di copyright nelle `xxxinfo` iniziali del testo, ad esempio così:

```
<bookinfo>
  <title...
  <author...
  <edition...
  <copyright>
    <year>2003</year>
    <year>2004</year>
    <holder>Tootsie Griff</holder>
  </copyright>
</bookinfo>
```

Questa nota non sostituisce la nota di Licenza e/o la cronologia, è semplicemente un optional extra.

## Inserire l'intera Pubic Documentation License

Invece di dare l'indirizzo internet (cioè l'URL), si può anche inserire l'intera PDL così com'è nel documento. Questo può essere utile specialmente quando il lavoro è un book oppure un lungo article e ci si aspetta (o si spera) che molta gente lo stampi e ne distribuisca le copie. Su documenti brevi la PDL intera sarebbe un po' troppo pesante, ma ognuno è libero di fare come crede.

Il sorgente DocBook della PDL si trova in `src/docs/firebirddocs/licenses.xml`. Notare che solo la sezione con la il testo della licenza (includere le generiche note di licenza) appartiene alla vera propria PDL. L'introduzione non fa parte della licenza.

Includendo la PDL nel proprio documento, si devono riempire le parti lasciate in bianco della sezione 5.2 della licenza stessa. Si potrebbero anche lasciare così come sono (se il vostro nome è nelle note di licenza), oppure riempire solo «the Initial Writer» e/o «the Copyright holder».

## Note dei traduttori

Tradurre un documento è un tipo di modifica particolare. In qualità di traduttori è necessario:

- Elencarsi come contributori nelle note di licenza, con una descrizione alla nota del tipo "Traduzione in italiano". Si possono tradurre le note di licenza nella propria lingua volendolo, ma si possono lasciare anche solo in inglese o metterle in entrambe le lingue.
- Aggiungere un elemento `revision`, nella lingua tradotta, alla cronologia nella `revhistory`. Per quanto riguarda il numero di revisione, il `revnumber`, si usa il numero di revisione dell'originale tradotto, seguito da un trattino ed il codice del linguaggio, ad esempio «2.0-es» o «1.1-fr»:

```
<revhistory>
  ...revisioni precedenti...
  <revision>
    <revnumber>1.1</revnumber>
    <date>5 dic 2005</date>
    <authorinitials>PV</authorinitials>
    <revdescription>
      <para>Aggiunte alcune informazioni su COALESCE</para>
      <para>Corretti alcuni errori di battitura</para>
    </revdescription>
  </revision>
  <revision>
    <revnumber>1.1-fr</revnumber>
    <date>13 dic 2007</date>
    <authorinitials>UM</authorinitials>
    <revdescription>
      <para>Traduzione in italiano</para>
    </revdescription>
  </revision>
</revhistory>
```

- Aggiungere un elemento `othercredit` nelle `xxxinfo` all'inizio del documento, così:

```
<articleinfo>
  <title>Guida sul NULL in Firebird</title>
  <author>
    <firstname>Paul</firstname>
    <surname>Vinkenoog</surname>
  </author>
  <othercredit>
    <firstname>Umberto</firstname>
    <surname>Masotti</surname>
    <contrib>Traduzione in italiano</contrib>
  </othercredit>
  <edition>22 luglio 2007 - Versione del documento 2.0-it</edition>
</articleinfo>
```

L'elemento `contrib` contiene la stessa informazione della descrizione nelle note di licenza, ma deve essere sempre nella lingua del documento finale.

Notare anche nell'elemento `edition` la versione del documento, verificate sempre che sia *identica* nella cronologia!

## Tradurre la PDL

Non c'è da tradurre la PDL vera e propria, ma, nel caso:

-

Aggiungerla come un documento indipendente all'insieme dei documenti della lingua, in un book denominato *Licenses* (ma traducete «Licenses» nella *nuova* lingua).

- Nella traduzione dell'introduzione alla PDL, va spiegato che la sola versione inglese ha valore legale ed va accluso un link all'originale inglese.
- ogniqualvolta nelle note di licenza c'è un link alla PDL tradotta, va messo anche un link alla PDL originale e va chiarito che solo quest'ultima ha valore legale.

Si può opzionalmente includere la PDL tradotta al documento stesso, se si ritiene che tutto quel fagottone non crei alcun problema.

## Aggiungere il vostro documento al modulo del manuale

Finito il testo, e verificato che si ricostruisce correttamente, si può aggiungerlo al modulo del manuale. Per il primo contributo al progetto della documentazione ci si accorda con uno dei coordinatori per una prima revisione, oppure si può mettere la versione HTML su un sito per discuterne nella lista. Poi, e dopo che sono state fatte tutte le possibili correzioni, il documento può essere depositato nel modulo. Avendo i diritti di commit, si può farlo autonomamente; senza questi diritti bisogna che venga fatto da uno dei coordinatori.

### *Richiedere i diritti di deposito*

Per ottenere i diritti di deposito, è prima necessario avere un accesso a SourceForge. Per averlo, bisogna registrarsi a <http://sourceforge.net/account/register.php> (purtroppo, finora, il sistema di registrazione è solo in inglese). Poi si invia un messaggio alla lista firebird-docs indicando il vostro nome utente e chiedendo di essere aggiunto al progetto Firebird. Il responsabile del sottoprogetto del manuale e altri amministratori del progetto Firebird seguono la lista: valuteranno pertanto la richiesta di ingresso. Come regola generale si dovrebbe chiedere i diritti di scrittura solo *dopo* il primo contributo, perchè chi decide ha bisogno di una base sulla quale farsi un'opinione.

Le seguenti frasi al momento hanno lo stesso significato, pertanto:

- Essere un membro del progetto.
- Avere i diritti di deposito.
- Avere accesso in lettura e scrittura alla repository del progetto.

### *Cosa fare e cosa no, avendo i diritti di deposito*

Una volta accettato come membro del progetto, si ha diritti di scrittura a tutta la repository del progetto, non solo al modulo del manuale. Non esiste nessuna barriera tecnica per impedire il deposito di modifiche ad altri moduli, come il modulo centrale di `firebird2`, o perfino al modulo `CVSROOT` dove sono conservati importanti informazioni del progetto.

È chiaro che certe cose *non devono* passare neanche per l'anticamera del cervello. Vanno seguite le seguenti regole:

- Non depositare *mai* in altri moduli a meno che i responsabili di quei particolari moduli non l'abbiano chiesto esplicitamente.
- Depositare solo lavori nel modulo del manuale che siano relativi ad un compito assegnato esplicitamente a voi. Perfino in quel caso, è buona cosa preannunciare le modifiche e le aggiunte nella mailing list, per dare la possibilità agli altri componenti di fare eventuali commenti. In fondo, questo è un lavoro collettivo.
- Ritenendo di dover aggiungere un nuovo documento o cartella, non fatelo direttamente, ma prima proponetelo alla lista.

In pratica, le cose potrebbero essere un pochino più semplici di quanto detto, specialmente riguardo al proprio lavoro. Non desideriamo che la gente si senta costretta, e certamente non si ha la voglia di chiedere il permesso di modificare la più piccola cosa del proprio lavoro. Si desidera cooperare con gente responsabile, e desideriamo sapere cosa tutti stiamo facendo. Inoltre, mantenersi in contatto l'un l'altro è piacevole ed insieme si è in grado di fare tutto.

## **Depositare il proprio lavoro**

Pur essendo membri del progetto, si può depositare il proprio lavoro da una copia locale solo se è stata estratta col proprio nome utente SF. Se si sta' ancora lavorando con una copia estratta in modo anonimo, bisogna prima fare un'estrazione SSH (SSH checkout), e poi rimettere le proprie modifiche. Solo allora si riesce a farne il deposito (commit). Le istruzioni dettagliate sono nella guida [Come ricostruire i manuali di Firebird](#) nel si siano dimenticati i particolari di come fare un'estrazione SSH.

Se è passato molto tempo dall'ultimo checkout o dall'ultimo aggiornamento, meglio farne un'altro di controllo prima di depositare. Questo rende la copia locale sincronizzata con il deposito remoto e riduce la possibilità di conflitti.

Dopo questi controlli, non appena si è pronti per depositare, si va nella directory del manuale, e, usando un CVS a linea di comando, si digita il comando:

```
cvs update -d [ solo se si vuole fare un aggiornamento preventivo ]
```

```
cvs add path/to/mydocument.xml [ solo se si deve aggiungere un documento nuovo non ancora in CVS ]
```

```
cvs commit -m "Breve messaggio informativo "
```

Dopo il -m, e all'interno degli apicini doppi, si può mettere un breve messaggio relativo a questo deposito, ad esempio "Aggiunte nuove funzioni all'API Reference" or "Corretti errori nella guida isql".

Quando richiesto bisogna inserire la propria password SF, e tutte le modifiche fatte, incluse quelle nelle sotto-cartelle, saranno depositate definitivamente. Il programma CVS sa quale server contattare; questo perchè le varie informazioni sono memorizzate nelle cartelle CVS che sono state create nell'estrazione, cioè durante il checkout.

Usando un altro programma CVS, com'è ovvio, fate riferimento alla sua documentazione.

### **Importante**

Dopo aver aggiunto un nuovo documento, bisogna fare comunque un commit distinto dall'aggiunta. Questo vale per il CVS a linea di comando e per molti altri (anche se non tutti) programmi CVS.

## Publicare i documenti sul sito web di Firebird

Prima di pubblicare i documenti, è necessario ricostruire i file in formato HTML e PDF. Questo è documentato nella guida [Come fare i manuali di Firebird](#). Per proseguire si assume che si siano fatti con successo i file in formato HTML e PDF.

### *Dare un nome al file PDF*

Il sistema di ricostruzione da un nome automatico a ciascun file prendendolo dall'ID dell'elemento principale che il documento DocBook contiene. I nomi dei file in formato HTML multipagina non vanno cambiati, questi file servono per essere utilizzati in rete, e cambiare un solo nome file potrebbe distruggere tutti i link contenuti nelle altre pagine. Ma i file PDF sono spesso scaricati dal lettore, ed avere nomi come `qsg2.pdf` o `ubusetup.pdf` in una directory di download oppure sul proprio desktop non aiuta *per niente* a riconoscerli come manuali di Firebird. Pertanto ecco qui alcune linee guida per dare un nome ai file PDF:

- Assicurarsi che il nome contenga la parola `Firebird`, possibilmente all'inizio;
- Farlo assomigliare al titolo del documento, magari accorciandolo;
- Usare i trattini («-») per separare fra loro le parole;
- Se diventa troppo lungo, eliminare parole come «manuale», «guida», «Come-fare-per» ecc., a meno che possa provocare confusione;
- Usare la propria lingua (quella del documento), ma solo caratteri ASCII (niente accenti, ecc.)
- Se (e *solo se*) applicando le precedenti regole risultasse un nome file già esistente in un'altra lingua, aggiungergli la lingua (o una sua abbreviazione).

Per esemplificare queste linee guida, ecco alcuni nomi file esistenti:

- `Firebird-2.0-QuickStart.pdf`
- `Firebird-Security.pdf`
- `MSSQL-to-Firebird.pdf`
- `Firebird-Generator-Guide.pdf`
- `Firebird-nbackup.pdf`
- `Firebird-2.0-Schnellanleitung.pdf`
- `Firebird-1.5-Arranque.pdf`
- `Firebird-et-Null.pdf`
- `Firebird-nbackup-fr.pdf`
- `Firebird-su-Ubuntu.pdf`
- `Firebird-nbackup-nl.pdf`
- `Guia-Escrita-Firebird.pdf`
- `Firebird-1.5-BystryjStart.pdf`
- `Firebird-Perehod-s-MSSQL.pdf`

### *HTML unica pagina*

Se e quando si pubblicheranno files in formato HTML singola pagina, per intenderci quelli fatti con **build monohtml**, daremo loro lo stesso nome del corrispondente file PDF, ma con l'estensione `.html`, naturalmente.

## Depositare il PDF

Avendo i diritti di accesso in scrittura sul server web di Firebird, ci si deve collegare con un client SFTP a `web.firebirdsql.org` e depositare i file propriamente rinominati in:

- `/srv/www/htdocs/pdfmanual` (documenti in inglese)
- `/srv/www/htdocs/pdfmanual/it` (documenti in italiano)
- `/srv/www/htdocs/pdfmanual/fr` (documenti in francese)
- `/srv/www/htdocs/pdfmanual/ja` (documenti in giapponese)
- ecc.

Le note di rilascio invece vanno in:

- `/srv/www/htdocs/rlsnotes`
- `/srv/www/htdocs/rlsnotes/it`
- ecc.

Senza i diritti di accesso al server, si deve chiedere a qualcun altro se può farlo per voi, oppure, se si è membri del progetto, chiedere un nome utente ed una password sul server.

## Depositare i file HTML multipagina

Bisogna assicurarsi di depositare tutti i file necessari:

1. i file HTML che insieme formano il manuale o i manuali.
2. il file del foglio di stile `firebirddocs.css` (se è stato modificato dall'ultimo deposito).
3. tutte le immagini che sono state modificate o aggiunte nel sottodirettorio `images`.

Per far funzionare tutti i link, potrebbe essere necessario ricostruire e depositare tutto il libro (`book`) padre del documento creato o aggiornato, o addirittura l'intero insieme (`set`). Ad ogni modo, va depositato il tutto in:

- `/srv/www/htdocs/manual` (Documenti in Inglese)
- `/srv/www/htdocs/manual/it` (Documenti in Italiano)
- `/srv/www/htdocs/manual/fr` (Documenti in francese)
- e così via...

### Avvertimento

Se le pagine in questione appartengono ad un altro insieme base che non sia il default `firebirddocs` (cioè ad esempio a `papers` oppure a `rlsnotes`) non vanno messe in quelle directory. Non sonostate ancora definite delle regole precise, ma gli HTML multipagina generati da insiemi differenti non dovrebbero essere mischiati. Se si rischia di nascere una situazione del genere, meglio parlarne prima nella lista `firebird-docs`.

## Aggiornare l'indice della documentazione di Firebird

L'indice della documentazione di Firebird (Firebird Documentation Index) che è in <http://www.firebirdsql.org/?op=doc> è uno script PHP che preleva la maggior parte del suo contenuto dalle informazioni contenute in file di

dati nel server. Dopo aver aggiornato file esistenti che sono già nell'indice, è già tutto a posto, a meno che non si sia cambiato il nome del file. Nel caso invece in cui si sia creato un nuovo documento o una nuova traduzione, questa va aggiunta all'indice. Ecco come si fa:

*Quando si crea un documento ex-novo*

1. Cercare nell'indice della documentazione quale categoria meglio si confà al documento. Le categorie sono al momento indicate con la testata a caratteri arancione.
2. Connettersi al server, cambiare la directory corrente a `/srv/www/htdocs/doc` e cercare i file che cominciano con `Cat_`. Aprire quello della categoria desiderata.
3. Leggere le istruzioni in cima al file, e se lo avete fatto già altre 100 volte prima, controllate che non sia cambiato niente.
4. Creare una nuova sezione cominciando con il titolo del documento in inglese, seguito da una linea per ogni versione disponibile che deve avere un formato speciale. Per un nuovo documento, la sezione assomiglia a questa:

```
Firebird Uninstallation Howto
en:/manual/fb-uninstall.html
en:/pdfmanual/Firebird-Uninstall.pdf
```

Ogni linea di versione inizia con il codice del linguaggio, seguito dal carattere di due punti («:»), infine c'è l'URL di un file. Per i documenti che sono sul nostro server, l'URL è semplicemente il percorso «assoluto» dalla radice del server. Le singole sezioni sono separate da linee vuote. In ogni categoria della pagina web dell'indice della documentazione Firebird, i documenti sono ordinati come le singole sezioni nel file. L'ordine delle linee di versione all'interno di una sezione è invece irrilevante.

5. Salvare il file. Se l'avete modificato nel vostro computer, ridepositatelo sul server. Aggiornando la pagina dell'indice della documentazione di Firebird nel vostro navigatore internet, verificare se il documento è elencato esattamente dove si desidera e se i link funzionano bene: notare che va controllato se i link sono nelle giuste colonne (HTML nella colonna centrale, PDF ed ogni altra cosa nell'ultima a destra).
6. Lo script PHP fa un buon lavoro nel determinare il tipo di documento, ma ci sono casi in cui sbaglia. Se succede, aggiungere il tipo di file tra parentesi graffe subito dopo l'URL nella linea della versione:

```
en:http://www.ibphoenix.com/main.nfs?a=ibphoenix&page=ibp_60_sqlref{html}
```

7. Quando tutto è a posto, si può confermare la categoria aggiornata nel CVS. Se si è fatta l'estrazione completa del modulo `web` di Firebird da SourceForge, si troveranno i file delle categorie (ed altro ancora) nella directory `web/website/doc`. Si deve usare il proprio nome utente e password SourceForge per prelevare, altrimenti non si riuscirà ad aggiornare le modifiche. Sempre nel manuale [Come fare i manuali di Firebird](#) è descritto come si fa a lavorare con la CVS.

*Quando si traduce un documento esistente in un nuovo linguaggio, o gli si aggiunge un nuovo tipo di documento*

1. Consultare l'indice della documentazione per verificare a quale categoria appartiene il documento. Le categorie sono indicate con testate a caratteri arancione al momento.
2. Collegarsi al server, e cambiare la directory di lavoro a `/srv/www/htdocs/doc` cercando i file che iniziano con `Cat_`. Aprire quello che corrisponde alla categoria desiderata.
3. Trileggere le istruzioni all'inizio del file.

4. Cercare la sezione del documento e aggiungere la linea (o le linee) per la versione aggiunta, per esempio:

```
Firebird Uninstallation Howto  
en:/manual/fb-uninstall.html  
en:/pdfmanual/Firebird-Uninstall.pdf  
fr:/manual/fr/fb-uninstall-fr.html  
fr:/pdfmanual/fr/Deinstaller-Firebird.pdf
```

L'ordine delle linee di versione è irrilevante all'interno di una sezione, ma il titolo dev'essere in cima.

5. I passi 5, 6, e 7 sono identici a quelli specificati nel caso di un nuovo documento.

## Appendice A: Cronologia

La cronologia completa è memorizzata nel modulo del manuale nell'albero del CVS; vedere [http://sourceforge.net/cvs/?group\\_id=9028](http://sourceforge.net/cvs/?group_id=9028) (in inglese).

### Diario delle Revisioni

0.1	17 gen 2004	PV	Prima bozza incompleta pubblicata col titolo <i>Writing Documentation for Firebird</i> (aka <i>Firebird Docwriting Howto</i> ).
0.2	27 gen 2004	PV	Prima versione completa. (Pubblicata nel CVS 31 gen 2004)
1.0	8 mar 2004	PV	Prime versione ufficiale nel sito Firebird.
1.1	26 feb 2005	PV	<p><i>Le seguenti modifiche si sono accumulate tra il marzo 2004 ed il febbraio 2005:</i></p> <p>Cambiato il titolo inglese in <i>Firebird Docwriting Guide</i>.          Aggiunta sezione sui documenti PostgreSQL.          Aggiunta nota sulle contribuzioni non in DocBook.          Spiegato il termine «XML ben costruito».          Condensata la lista dei benefici di DocBook.          Modificate le raccomandazioni sugli elementi section rispetto a sectN.          Eliminato xref ed altre cose raramente usate dalla lista degli elementi; aggiunto invece procedure.          Aggiornate le informazioni sul literallayout a spazio variabile.          Aggiunta sezione sulla PDL e su come includere le note di Licenza e la cronologia.          Tanti piccoli miglioramenti.          Aggiunta la cronologia ed il numero di revisione.          Pubblicato il lavoro sooto la Public Documentation License.</p>
1.1.1	8 apr 2005	PV	Aggiunto paragrafo relativo agli elementi titleabbrev.
1.2	10 feb 2006	PV	<p>Cambiati tutti gli elementi &lt;sectN&gt; della struttura sorgente in &lt;section&gt;.</p> <p>Cambiati tutti i links alla docbuildhowto in ulinks, in quanto i due articoli saranno in PDF distinti, d'ora in poi.</p> <p><i>Caratteristiche di DocBook XML</i>: rimosso un avviso su «plaintext». Aggiunta la nota su XSLT.</p> <p><i>Strumenti per scrivere in DocBook XML</i>: divisa in due parti; avviso riguardo problemi con UTF-8 e ConText; aggiunte informazioni su SciTE; aggiunto avviso sul salvare in 8-bit; modificato il primo paragrafo sugli strumenti dedicati all'XML; aggiunto Oxygen; rimosso Altova Authentic; aggiornato/modificato Altova XMLSpy information.</p> <p><i>Writing your DocBook doc</i>: reintitolato <i>Impostare il vostro documento DocBook</i>; cambiato secondo par.; spostato un paragra-</p>

fo in *Gli elementi che si usano più di frequente*, cambiandogli il link alla sezione «Elementi gerarchici» in testo normale.

*Creare il documento*: cambiata l'introduzione; aggiornato l'esempio al documento master; aggiunta nota su UTF-16; aggiunta informazione sulla disposizione dei file appartenenti ad insiemi di base alternativi.

*Scrivere il testo*: piccole modifiche al primo e all'ultimo paragrafo.

*Gli elementi che si usano più di frequente*: portata a livello esterno, preceduta da *Impostare il vostro documento DocBook*; varie modifiche ai suggerimenti iniziali sui paragrafi normali; separato *Elementi gerarchici* in sottosezioni, aggiunto/modificato un SACCO di roba; aggiunta sottosezione sulle tabelle HTML; rifatta la sezione su «quote - literal»; aggiunta sezione sulle immagini e le testate di paragrafo.

*Non-DocBook aspects of the writing process*: sparito, tutte le sezioni modificate e promosse al livello superiore, il cui primo paragrafo è in *Lingua e stile*.

*Copyrights*: rinominato *Problemi di diritti d'autore* e aggiunto un paragrafo iniziale.

*Respecting others' copyrights*: rinominato *Usare il materiale scritto da altri*. Separato in due il primo paragrafo e modificato. Il paragrafo relativo ai documenti Borland è spostato in una propria sottosezione. Il testo riguardante PostgreSQL prende il titolo *I documenti di PostgreSQL* diventando una sottosezione di *Usare il materiale scritto da altri*.

*Applicare la PDL al proprio lavoro*: parecchie modifiche, reorganizzazione delle varie sottosezioni ed aggiunte.

*Depositare il proprio lavoro*: include la linea di comando **cvs add** e la nota «Importante» che bisogna depositare comunque, dopo aver aggiunto un nuovo file.

1.2.1	11 mag 2006	PV	Corretto un marcatore iniziale nell'esempio bridgehead (rimosso /).
1.2.1-it	21 ott 2006	UM	Prima versione della traduzione in italiano.
1.2.2	25 gen 2007	PV	In <i>Gli elementi che si usano più di frequente</i> , segnalata l'opzione <code>title</code> per gli avvisi. Spostate le istruzioni ai traduttori per la gestione delle immagini in una nota.
1.3	5 mag 2007	PV	<p>In <i>Argomenti esposti in questa guida</i>: Aggiunto un nuovo elemento all'ultima lista.</p> <p>In <i>Elementi gerarchici: Link</i>, Tolta una nota riguardo lo spostamento dei punti caldi (corretto in FOP 0.93).</p> <p>In <i>Listati, schermate, vista letterale ed esempi</i>, Tolta una nota sul <code>literallayout</code> non a spazio fisso. Un <code>example</code> racchiuso in un blockquote.</p> <p>In <i>Tabelle HTML</i>, assegnato un id; incluso <i>Istruzioni di Elaborazione</i> in un <code>firstterm</code>.</p> <p><i>La restituzione di grandi tabelle nei PDF</i>: nuova sezione</p> <p>In <i>Stile</i>, rese più leggibili alcune frasi (anche in italiano).</p> <p><i>Pubblicare i documenti sul sito web di Firebird</i>: Nuova sezione.</p>

Note di licenza: modificato 2003-2006 in 2003-2007.

1.3-it      7 mag 2007      UM      Seconda versione italiana.

## Appendice B: Note di licenza

Il contenuto di questo documento è soggetto alla Public Documentation License Version 1.0 (la «Licenza»); si può usare questo documento solo se si concorda con i termini della Licenza. Copie della Licenza sono disponibili (in inglese) a <http://www.firebirdsql.org/pdfmanual/pdl.pdf> (PDF) ed in <http://www.firebirdsql.org/manual/pdl.html> (HTML).

Il documento originale è intitolato *Firebird Docwriting Guide*.

L'autore iniziale del documento originale è: Paul Vinkenoog.

Copyright © 2003–2007. Tutti i diritti riservati. Per contattare l'autore: paul at vinkenoog dot nl.

L'autore della versione italiana del documento è: Umberto Masotti.

La traduzione italiana è soggetta a Copyright ©2006–2007. Tutti i diritti riservati. Per contattare l'autore della versione italiana: umasotti at users dot sourceforge dot net.