

Basi di eZ publish

Questo capitolo descrive i termini di base, i modelli, le strutture e i blocchi di eZ publish. Per capire ed essere in grado di usare eZ publish correttamente, una persona deve conoscere le basi. Questo è importante. Il lettore dovrebbe dedicare un po' di tempo a questo capitolo. E' una buona idea prendere nota della nomenclatura e delle tecniche che sono descritte qui. Questo renderà più semplice apprendere le sorprendenti possibilità di eZ publish.

La struttura interna di eZpublish

Questa sezione descrive la struttura interna di eZ publish presentando una breve panoramica dei differenti strati di software del sistema. Inoltre viene presentata una panoramica e la spiegazione della struttura delle directory.

Kernel, librerie e moduli

eZ publish è una applicazione complessa, completamente object oriented scritta in linguaggio PHP. Il sistema consiste basilarmente di tre parti:

- Moduli
- Kernel
- Librerie

Le librerie

Le librerie sono i principali blocchi di costruzione del sistema. Queste sono classi PHP general purpose altamente riutilizzabili. Le librerie non sono in nessun modo dipendenti dal kernel di eZpublish. Chi cercasse le librerie dovrebbe dare una occhiata alla cartella "lib" nella root directory di una installazione di eZ publish. L'Appendice A contiene una lista completa e una breve descrizione delle librerie disponibili.

Il kernel

Il kernel eZ publish può essere descritto come cuore del sistema. Si prende carico di tutte le funzioni di basso livello come la manipolazione dei contenuti, i workflows, il controllo delle versioni, il controllo degli accessi, ecc. Basilarmente, il kernel è una collezione di "engines", costruiti su e facenti uso delle librerie.

I moduli

Un modulo può essere pensato come una collezione di funzionalità. Può essere descritto come una interfaccia a un "engine" all'interno del kernel. Ogni modulo ha una collezione di funzioni che si prendono carico di vari compiti. L'Appendice B contiene una lista completa e una breve descrizione dei moduli di eZ publish disponibili.

Struttura delle directory

La root directory di eZ publish è suddivisa in più sottodirectories. Ogni sottodirectory è dedicata a una parte specifica del sistema. Le sottodirectory contengono collezioni di files che hanno una correlazione logica. La struttura è pulita e dovrebbe essere semplice da comprendere.

bin	Contiene vari scrips Perl e di shell. Questi script sono principalmente usati per scopi di manutenzione.
design	Contiene tutti i files riguardanti il design come i templates, banners/immagini, fogli di stile, ecc.
doc	Contiene la documentazione e i change logs.

kernel	Contiene tutti i files del kernel come le classi, le viste, i tipi di dati, ecc. Qui è dove risiedono i files del cuore del sistema. Solamente gli esperti dovrebbero mettere il naso in questa parte.
lib	Contiene le librerie general purpose. Queste librerie sono collezioni di classi che svolgono vari compiti. Il kernel fa uso di queste librerie.
settings	Contiene i files di configurazione dinamici, specifici per ogni sito.
share	Contiene files di configurazione statici come code pages, informazioni sulla localizzazione e files di traduzione.
var	Contiene i file della cache e i logs. In aggiunta contiene anche immagini e files (contenuti specifici per sito). Ovviamente questa directory crescerà in dimensioni.

Contenuto e design

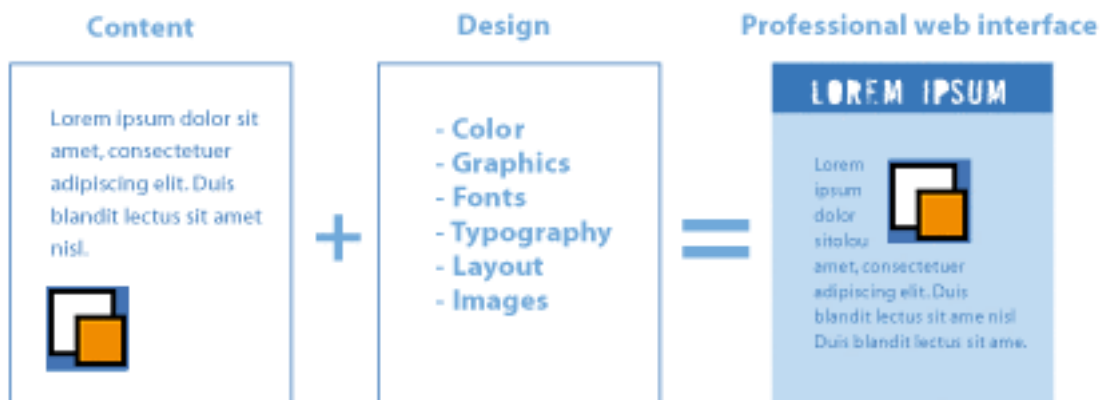
Nel mondo di eZ publish, il contenuto e il design sono separati.

Per contenuto intendiamo informazioni che devono essere organizzate e immagazzinate in una struttura (per un seguente recupero). Potrebbe essere il contenuto di una notizia (titolo, riassunto, testo, immagini), gli attributi di una persona (nome, data di nascita, indirizzo, foto) e così via.

Ovviamente, le informazioni immagazzinate in una struttura di contenuto devono essere presentate a un certo punto, preferibilmente in un modo che sia facilmente comprensibile dagli esseri umani. Mentre contenuto significa “dati grezzi”, il design è il modo in cui i dati vengono presentati visivamente. Quando parliamo di design, siamo parlando delle cose che compongono una bella interfaccia: fogli di stile, banners, immagini, layout, ecc.

La separazione del contenuto dal design

Il seguente diagramma illustra l'elegante separazione del contenuto dal design – e come queste cose possano essere combinate a formare una interfaccia professionale.



Questa distinzione, e l'abilità del sistema di manipolarla elegantemente, è una delle qualità più importanti di eZ publish. La separazione del contenuto dal design apre una vasta rosa di possibilità che non potrebbero essere ottenute altrimenti. Per nominarne qualcuna:

- Autori dei contenuti e designer possono lavorare separatamente senza conflitti.
- I contenuti possono essere pubblicati facilmente in più formati.
- I contenuti possono essere facilmente trasferiti e riutilizzati.

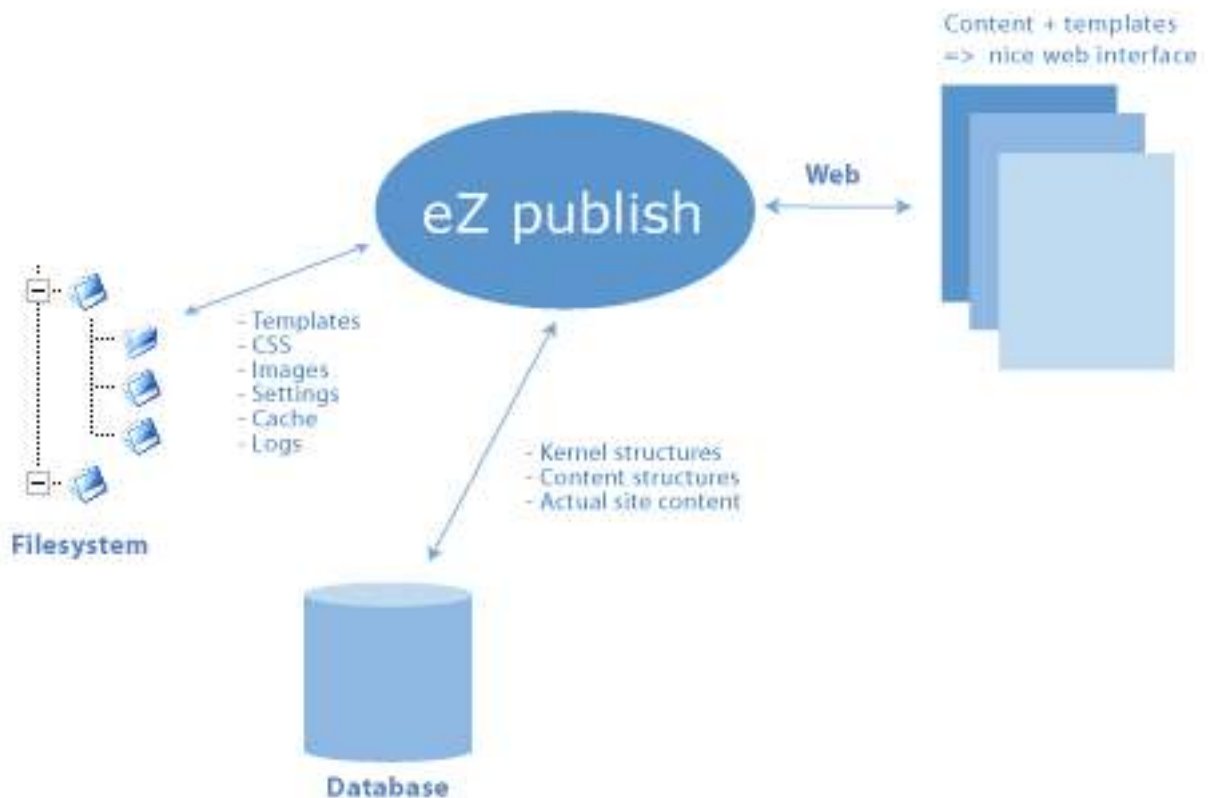
- Cambiamenti e redesign possono essere applicati tramite semplici modifiche.

Templates

eZ publish utilizza i templates come unità fondamentale di design del sito. Un template è fondamentalmente un file HTML esteso che descrive come alcuni particolari tipi di contenuto dovrebbe essere visualizzato. L'estensione è un potente set di operatori e funzioni specifiche per i template di eZ publish. Per esempio, un template può decidere che una pagina dovrebbe apparire con i bordi blue, la barra del titolo del sito in alto, e una manciata di elementi di contenuto nel mezzo della pagina. Quando si accede alla pagina, diventa compito del sistema di content management inserire il contenuto nei punti appropriati del template.

Immagazzinamento

eZ publish struttura e immagazzina i contenuti all'interno di un database. Questo è vero per tutti i contenuti eccetto che per le immagini e dei files binari e di dati. Questi sono immagazzinati nel filesystem (all'interno della directory "var") per guadagnare velocità. Anche tutto ciò che è in relazione col design (files di template, files CSS, immagini non specifiche del contenuto) è immagazzinato sul filesystem.



Management dei contenuti in eZ publish

Ogni file che viene creato, che sia un documento, una transazione finanziaria, un report, una nota, un file video, una immagine, un ordine di acquisto, un back-up di dati o un contatto col cliente è classificato come "contenuto". La regola di un sistema di content management è di organizzare ogni singolo elemento di questi contenuti indipendentemente dalla suo tipo o complessità. Questo fornisce alle organizzazioni una soluzione strutturata, automatizzata e flessibile che permette di distribuire liberamente le informazioni e di aggiornarle istantaneamente per diversi canali di comunicazione come il world wide web, le intranet e diversi tipi di front-end e back-end.

In eZ publish, tutto il contenuto è gestito attraverso una interfaccia grafica chiamata *administration*

interface. Questa interfaccia è descritta in dettaglio nel capitolo “L'interfaccia amministrativa”.

Questa sezione descrive come eZ publish struttura e gestisce i contenuti.

Una nota sulla tecnologia object oriented

Superficialmente, object-oriented significa niente più che guardare il mondo in termini di oggetti. Nella vita reale, la gente è circondata da diversi oggetti: mobili, auto, animali domestici, umani, ecc. Ognuno di questi oggetti ha delle caratteristiche che noi utilizziamo per identificarli. Questo è anche il modo in cui il contenuto è descritto e gestito all'interno di eZ publish, attraverso l'astrazione e gli oggetti.

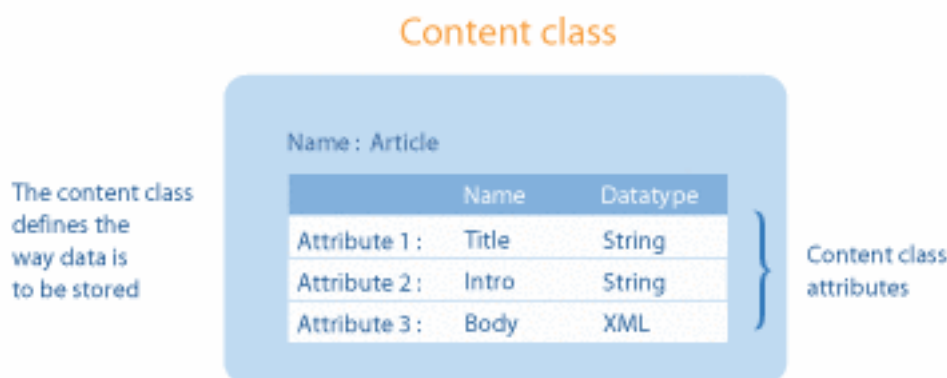
La struttura del contenuto

Diversamente da altri sistemi di content management, eZ publish non fa uso di un modello di contenuto predefinito “one-size-fits-all”. Invece, permette all'amministrazione del sito di creare la propria struttura di contenuti utilizzando un singolo approccio object oriented. Questo rende semplice strutturare, immagazzinare, recuperare e presentare dati comuni. Invece di provare disperatamente a fare entrare i dati in una struttura rigida e predefinita, semplicemente creiamo una struttura che sia adatta ai nostri dati. Comunque, è abbastanza improbabile che tutti abbiano bisogno di creare una struttura diversa. Ecco perché eZ publish viene fornito con un set di strutture e tipi di dati pronti all'uso. E' anche possibile modificare ed estendere le strutture contenute. In altre parole, eZ publish offre sia strutture pronte all'uso che custom. Questa è una delle caratteristiche che rendono eZ publish un sistema estremamente flessibile e di successo.

Tipi di dati

Un tipo di dato è l'entità più piccola di immagazzinamento possibile. eZ publish è fornito con una collezione di tipi di dati fondamentali che possono essere usati per costruire strutture di contenuto potenti e complesse. In aggiunta, è possibile creare tipi di dati personalizzati per necessità specifiche. Alcuni dei tipi di dati di default sono: stringhe di testo; testo XML, immagini, file binari, prezzi, ecc. L'Appendice C contiene una lista completa dei tipi di dati interni.

La classe contenuto



La classe contenuto è semplicemente una definizione di una struttura dati arbitraria. Una classe contenuto non contiene nessun dato. Come menzionato prima, eZ publish è fornito con un set di classi contenuto predefinite pronte all'uso come “folder”, “articolo”, “account utente”, “immagine”, ecc. Queste classi contenuto sono disegnate attentamente per coprire la maggior parte dei compiti di un CMS. In aggiunta, è possibile creare classi contenuto personalizzate utilizzando l'interfaccia amministrativa. Le classi contenuto (sia quelle personalizzate che quelle fornite) possono essere

facilmente modificate e/o estese in qualsiasi momento. L'Appendice D contiene una lista completa delle classi contenuto fornite.

Attributi della classe contenuto

Ogni classe contenuto è fatta di attributi, quindi una collezione di attributi definisce la struttura dati di una classe contenuto. Un attributo può essere una stringa di testo, un intero, un float, ecc. Il tipo di attributo è determinato dal tipo di dato che viene scelto per quello specifico attributo. In altre parole: ogni classe contenuto è fatta da uno o più tipi di dato. Le classi contenuto possono essere costruite e modificate facilmente usando l'interfaccia amministrativa.

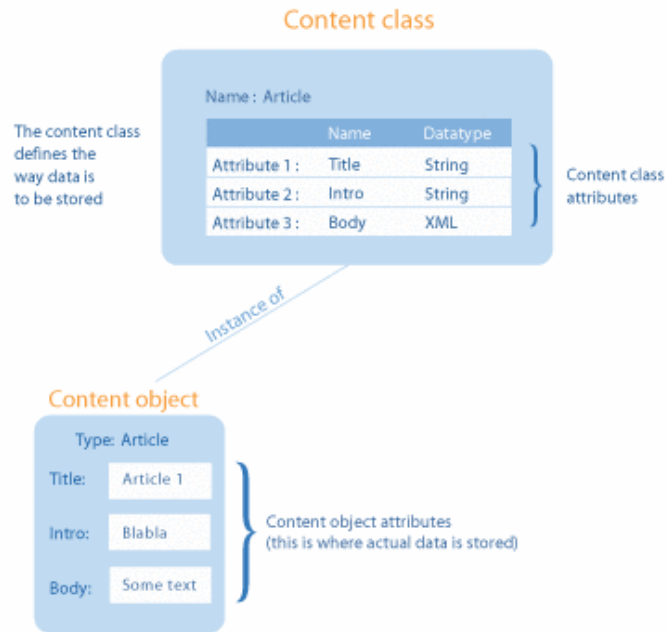
Esempio

Assumiamo di voler immagazzinare informazioni sulla frutta. Le informazioni che vogliamo immagazzinare sono il nome del frutto, il suo peso e il suo colore. Possiamo semplicemente creare e costruire una classe contenuto chiamata "Frutto" con tre attributi: nome, peso e colore. Il tipo di dato stringa può essere usato per rappresentare il nome e il colore mentre l'attributo peso può essere rappresentato utilizzando il tipo intero:

Classe contenuto: Frutto		
	Nome attributo	Tipo di dato
Attributo 1:	Nome	Testo
Attributo 2:	Peso	Intero
Attributo 3:	Colore	Testo

L'oggetto contenuto

Un oggetto contenuto è una istanza di una classe contenuto. Una classe contenuto definisce semplicemente la struttura dell'oggetto contenuto utilizzando gli attributi della classe contenuto. E' l'oggetto contenuto stesso che contiene i dati/informazioni. Una volta che una classe contenuto è definita, diversi oggetti contenuto di quel tipo possono essere creati. Molteplici oggetti dello stesso tipo sono usati per immagazzinare dati/informazioni simili, per esempio: molteplici oggetti articolo sono utilizzati per immagazzinare differenti notizie.

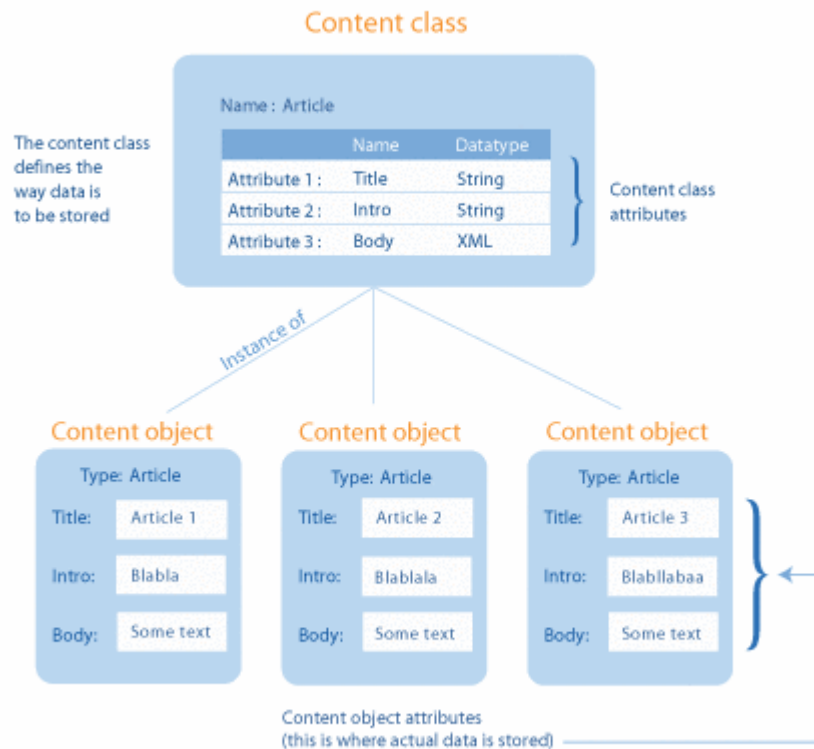


Attributi oggetto contenuto

Ogni oggetto contenuto consiste di diversi attributi indipendenti. Questi attributi sono definiti da attributi della classe contenuto. All'interno di un oggetto contenuto, tutti i dati sono incapsulati in uno o più attributi dell'oggetto contenuto.

Relazioni fra tipi di dati, classi e oggetti

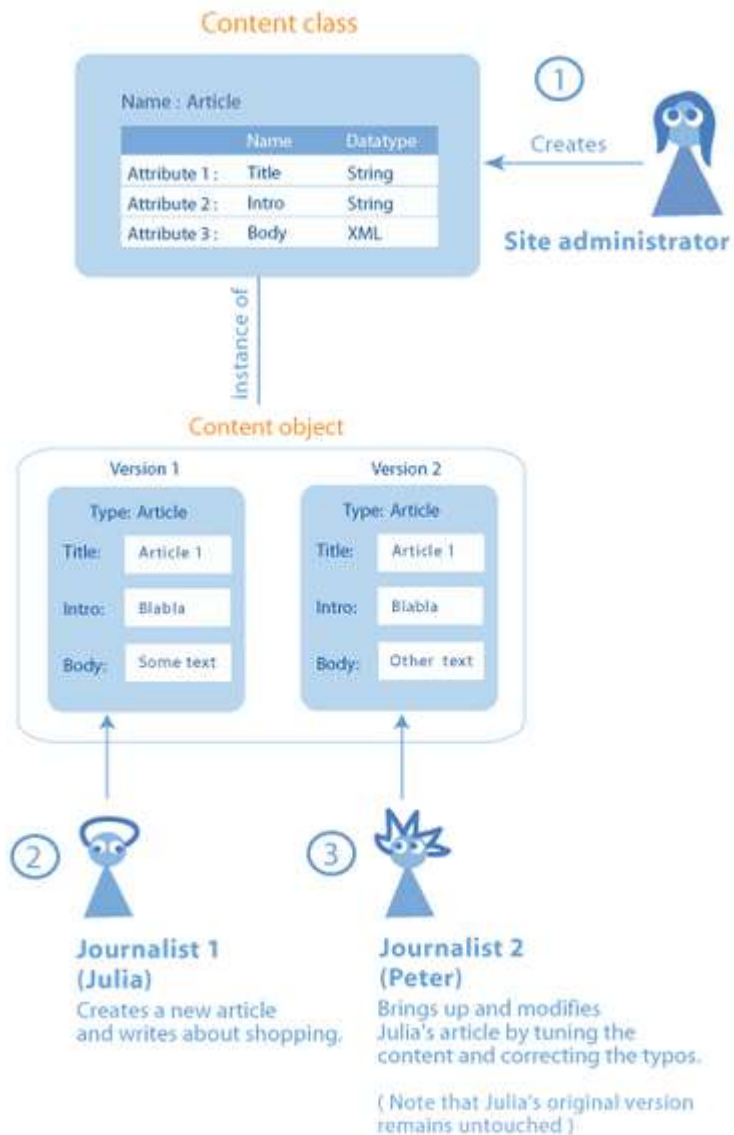
Il diagramma seguente riassume e illustra la relazione fra tipi di dato, una classe contenuto e una collezione di oggetti contenuto. Gli oggetti contenuto sono istanze della stessa classe contenuto; il che fondamentale significa che sono dello stesso tipo, ma contengono differenti contenuti/dati/informazioni.



Versioni degli oggetti contenuto

Il contenuto che immagazzinato all'interno di un oggetto contenuto, può esistere in una o più versioni. Ogni volta che il contenuto è modificato, viene creata una nuova versione. La vecchia versione rimane uguale. Questa è la modalità con cui eZ publish tiene traccia delle modifiche apportate da vari utenti. Il sistema di versioni permette all'utente di annullare e rifare le modifiche ecc. Per evitare che il database sia riempito da versioni vecchie inutilizzate, l'amministratore può settare limitazioni sulle versioni. E' possibile settare il numero massimo di versioni per ogni classe contenuto.

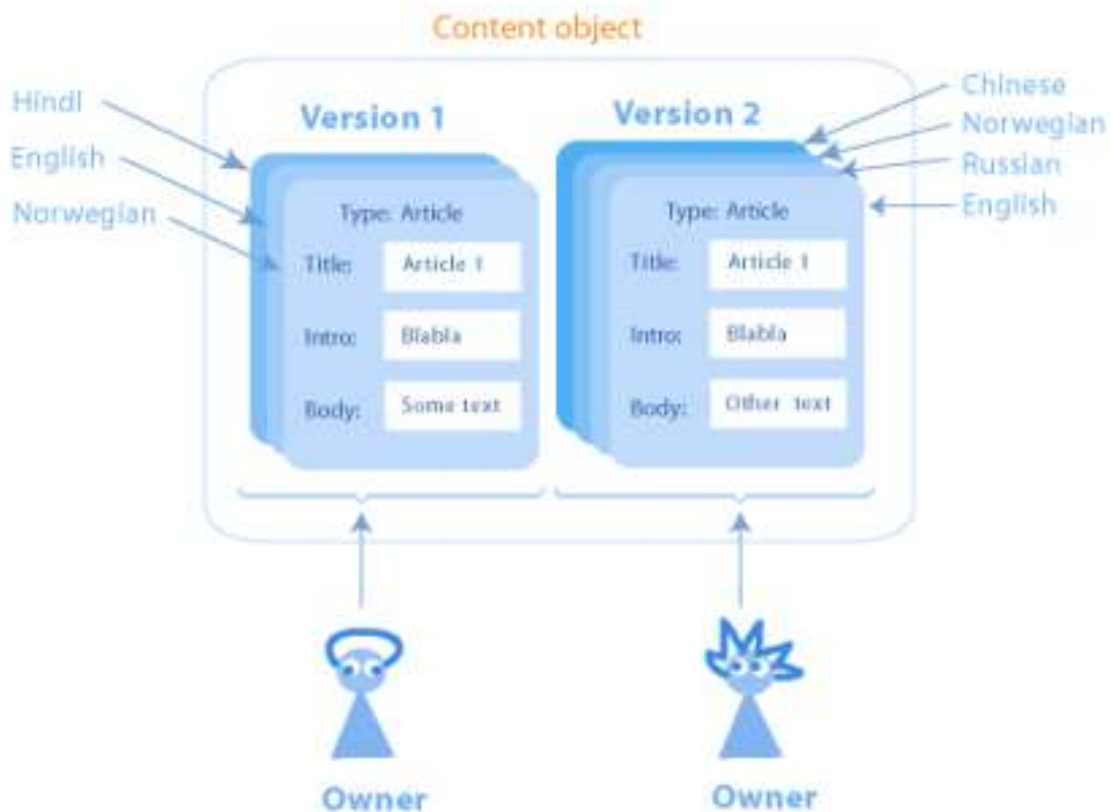
Ogni versione di un oggetto contenuto può appartenere a un utente diverso. Solo una versione può essere considerata "pubblicata", questa è chiamata versione corrente. Usando giornalisti e articoli come esempio, il seguente diagramma presenta una semplice illustrazione di come funziona il sistema di versioni.



Il sistema di versioni interno è un utile tool che può essere usato per controllare le versioni di ogni tipo di contenuto (indipendentemente dal tipo/forma/struttura/ecc.). Fondamentalmente fornisce un sistema di controllo delle versioni integrato.

Supporto per linguaggi multipli

In aggiunta al sistema di versioni, eZ publish fornisce anche una potente soluzione multi linguaggio. Può essere pensato come la terza dimensione di un oggetto contenuto. Ogni versione di qualsiasi contenuto arbitrario può esistere in diversi linguaggi. Questa è una funzionalità estremamente pulita. Fondamentalmente fornisce una soluzione multilingua generica, incorporata che può essere usata con ogni tipo di contenuto. Il diagramma sotto illustra come la soluzione multilingua è inserita nel sistema di versioni.

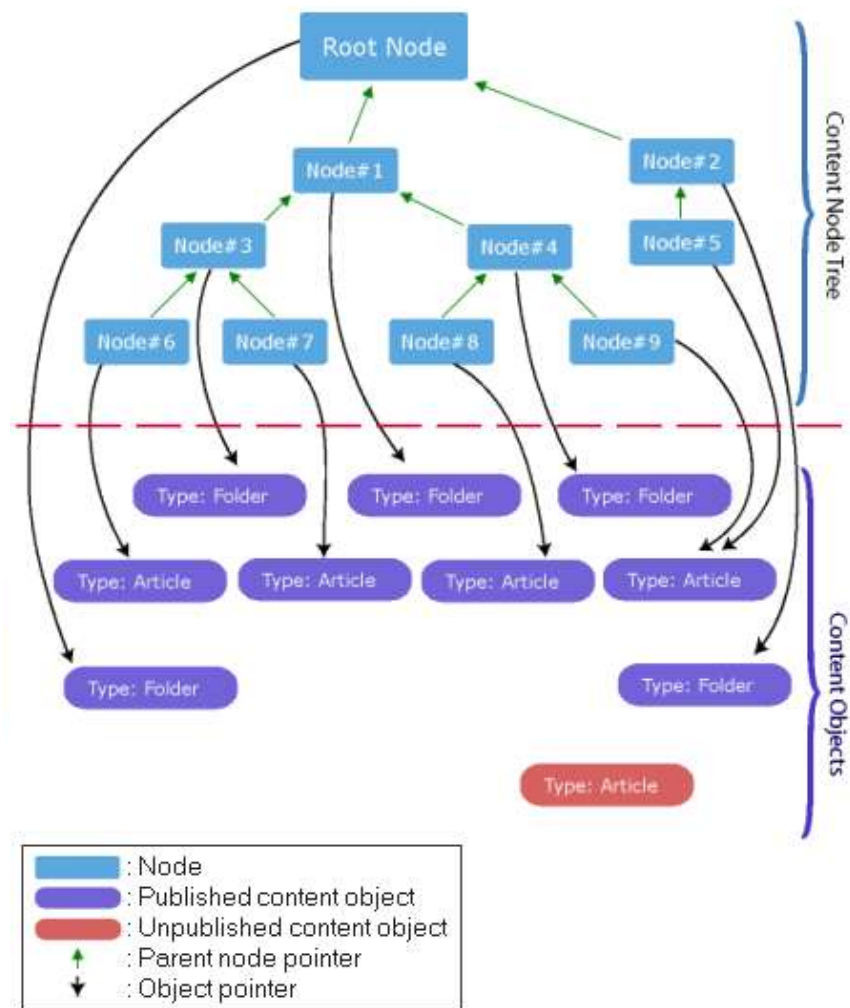


Oggetti, nodi e l'albero dei nodi di contenuto

Un oggetto contenuto è una istanza di una classe contenuto. Quando il sistema è in uso, i nuovi oggetti contenuto sono creati al volo. Per esempio, quando viene creato un nuovo articolo, viene creato un nuovo oggetto contenuto per immagazzinare il contenuto dell'articolo (l'oggetto si prenderà carico anche di immagazzinare tutte le diverse versioni e linguaggi dell'articolo). Ovviamente, gli oggetti contenuto non possono essere oggetti volanti nello spazio. Devono essere organizzati e strutturati in qualche modo. Qui è dove i nodi e l'albero dei nodi di contenuto entra in gioco.

Un nodo può essere pensato come un incapsulamento di un oggetto contenuto. L'albero dei nodi di contenuto è fatto di nodi. È una rappresentazione gerarchica di tutti i contenuti che appartengono a un sito. In altre parole, questo è il meccanismo che è usato per organizzare gli oggetti contenuto presenti nel sistema. Un modo tipico di categorizzare contenuti all'interno dell'albero è quello di fare uso di contenitori (per esempio folders) sotto i quali i contenuti rilevanti sono piazzati (praticamente nello stesso modo in cui sono organizzati in un filesystem).

Usando l'interfaccia di amministrazione per gestire folders, articoli, files, immagini, ecc. l'utente interagisce con l'albero dei nodi di contenuto. L'albero può essere facilmente usato per generare una mappa del sito. Il diagramma seguente illustra la relazione fra gli oggetti contenuto e i nodi.



Ogni foglia nell'albero dei contenuti è un nodo. Come minimo, l'albero consiste di un nodo, che è chiamato il nodo radice (root). Questo è il folder radice sotto il quale tutti i contenuti vengono piazzati. Ogni nodo (eccetto che per il nodo radice) ha due puntatori, uno che punta al nodo genitore (parent) e uno che punta a un oggetto contenuto. Un nodo può puntare solamente a un nodo genitore e un oggetto contenuto. Comunque, un oggetto contenuto può essere puntato da diversi nodi, che significa che lo stesso oggetto potrà comparire in differenti locazioni all'interno dell'albero. Un oggetto contenuto non pubblicato (draft) non ha un nodo associato ad esso. In altre parole solamente gli oggetti pubblicati compaiono nell'albero dei nodi di contenuto. Gli oggetti contenuti archiviati inoltre, perderanno il loro nodo e scompariranno dall'albero. Grazie all'incapsulamento in nodi degli oggetti, ogni tipo di oggetto può essere piazzato in qualsiasi locazione nell'albero (ma rimarrà presente nel sistema). Il numero di livelli nell'albero è illimitato. Chiaramente, questa è una soluzione estremamente flessibile e generica che può essere utilizzata per strutturare qualsiasi tipo di contenuto personalizzato.

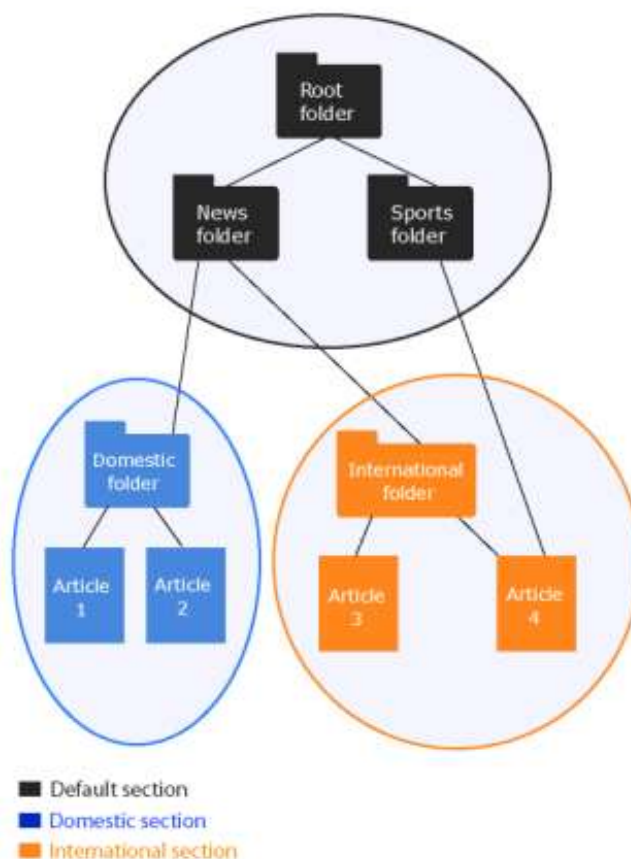
Locazioni

Nella descrizione del posizionamento di un oggetto, la sezione precedente ha fatto uso del termine "locazione".

Un oggetto può essere assegnato a diversi nodi, quindi apparire in molteplici posti nell'albero dei contenuti. In altre parole, un oggetto può avere diverse *locazioni* all'interno dell'albero dei contenuti. Una locazione è fondamentalmente un sinonimo per un nodo (l'incapsulamento di un oggetto).

Sezioni

Come menzionato nelle precedenti sezioni, l'albero dei nodi di contenuto è una rappresentazione ad albero gerarchico di tutti i contenuti che appartengono a un sito. Un modo tipico di categorizzare contenuti all'interno dell'albero è di fare uso di contenitori (ad esempio folders) sotto i quali gli oggetti contenuto rilevanti vengono piazzati (praticamente nello stesso modo in cui sono organizzati in un filesystem). In aggiunta a questa struttura gerarchica, l'albero può essere diviso in sezioni logiche. Le sezioni sono fondamentalmente utilizzate per segmentare l'albero dei contenuti.



L'utilizzo delle sezioni semplifica:

- L'utilizzo di templates personalizzati
- Il controllo/limitazione dell'accesso ai contenuti

L'uso delle sezioni è implementato a livello di oggetto. Ogni oggetto può appartenere a una sola sezione. Di default, un oggetto appartiene alla prima sezione, quella di default. L'interfaccia amministrativa semplifica l'utilizzo delle sezioni dando la possibilità di realizzare il sezionamento a livello di nodo. Ogni volta che viene cambiata l'assegnazione della sezione di un nodo, l'assegnazione della sezione di tutti i nodi figli di quel nodo sarà cambiata a sua volta. Ogni volta che viene pubblicato un oggetto, eredita automaticamente la sezione della locazione principale dell'oggetto.

Gestione del sito in eZ publish

eZ publish è in grado di far girare più siti all'interno della stessa installazione. Un sito può essere raggiunto in modi differenti. Vari design possono essere associati alle differenti parti di un sito. Questa sezione contiene una breve spiegazione su come eZ publish gestisce i siti e le interfacce dei siti.

Sito

Il termine “sito” comprende tutto ciò che appartiene a un particolare sito:

- Settaggi
- Un database contenente la struttura dei contenuti e i contenuti stessi
- I files relativi ai contenuti
- I files relativi al design

Interfaccia del sito

Il contenuto di un sito può essere visualizzato (e modificato) in vari modi. Questo è possibile grazie all'uso di interfacce del sito multiple. Una interfaccia del sito fondamentale determina che particolare design utilizzare quando si accede al sito in qualche modo. Come minimo, ogni sito ha due interfacce: una interfaccia di amministrazione e una interfaccia utente. Un esempio tipico è che l'amministratore del sito usa l'interfaccia di amministrazione per costruire e modificare il sito. Gli utenti devono solo accedere all'interfaccia utente, che visualizza semplicemente il contenuto in una bella forma. Un sito può avere diverse interfacce differenti.

Accesso al sito

Per distinguere fra siti e interfacce dei siti, eZ publish fa uso di una soluzione generica chiamata accesso al sito. Il settaggio di un accesso al sito fondamentale definisce due cose:

- Come eZ publish riconosce il sito/interfaccia al quale si accede
- Settaggi di configurazione che si sovrapporranno ai defaults

I settaggi più significativi sono quelli del database e del design. Aggiungendo accessi al sito, è possibile creare nuovo siti e/o nuove interfacce a un sito esistente.

URLs in eZ publish

eZ publish è in grado di gestire due tipi di URLs:

- URLs di sistema
- URLs virtuali

URLs di sistema

Un URL di sistema è un semplice URL di eZ publish che contiene diverse informazioni sul contenuto/funzionalità al quale si fa accesso. Un URL di sistema può apparire come:

<http://www.example.com/content/view/full/44>

URLs virtuali

Un URL virtuale è una versione semplificata di un arbitrario URL di sistema. Un URL virtuale è più carino, più semplice da ricordare e qualche volta più corto del corrispondente URL di sistema.

URLs virtuali sono spesso chiamati “URLs carini (nice)” o “URL aliases”. Un URL virtuale può apparire come: http://www.example.com/recent_news

Esempio

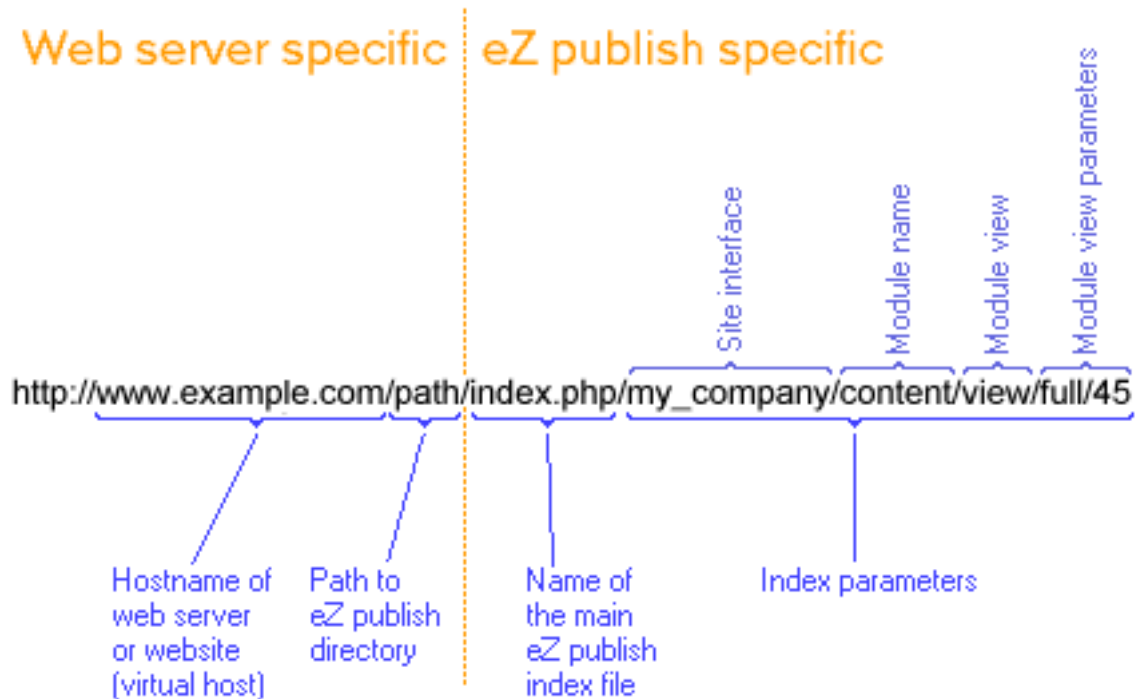
La pagina dei links di un sito può essere raggiunta in due modi:

- Usando un URL di sistema: <http://www.example.com/content/view/full/46>
- Usando un URL virtuale: <http://www.example.com/links>

Le sottosezioni seguenti daranno ulteriori spiegazioni sui due tipi di URLs.

URLs di sistema

A prima vista, un URL di sistema può sembrare caotico e intimidire, In realtà, non è il caso. Tutti gli URLs di sistema seguono una struttura elegante e ben definita. L'illustrazione seguente spiega le varie parti di un tipico URL di sistema di eZ publish:



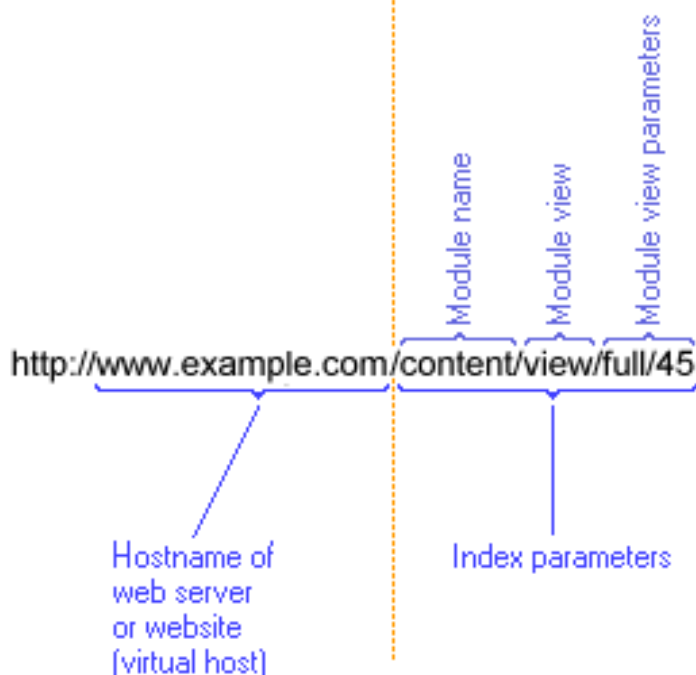
L'URL nella illustrazione sopra riprende la maggior parte delle installazioni di eZ publish con una configurazione di base/default.

Il web server può essere configurato per non mostrare il file index; in questo caso, "index.php" semplicemente scomparirà da tutte le richieste.

La configurazione di accesso di default di eZ publish è *URL*, che significa che il nome dell'interfaccia del sito è incorporata nelle URL di richiesta stesse. Se la configurazione dell'accesso al sito è *hostname*, allora il nome del sito sarà molto probabilmente incorporato nel nome dell'host (forse in forma di sottodominio). Se la configurazione dell'accesso al sito è *port*, allora un carattere ":" e un numero di porta saranno aggiunti al nome dell'host. Negli ultimi due casi (*hostname* e *port*), il nome dell'interfaccia del sito non sarà inclusa nell'URL.

Il diagramma sotto illustra una versione abbreviata (index nascosto, modo di accesso non-URL) dell'URL nell'esempio precedente.

Web server specific eZ publish specific

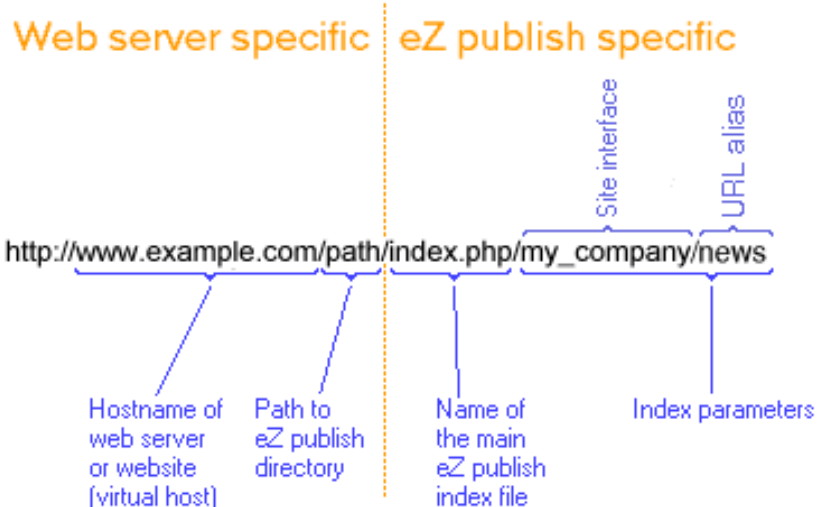


URLs virtuali

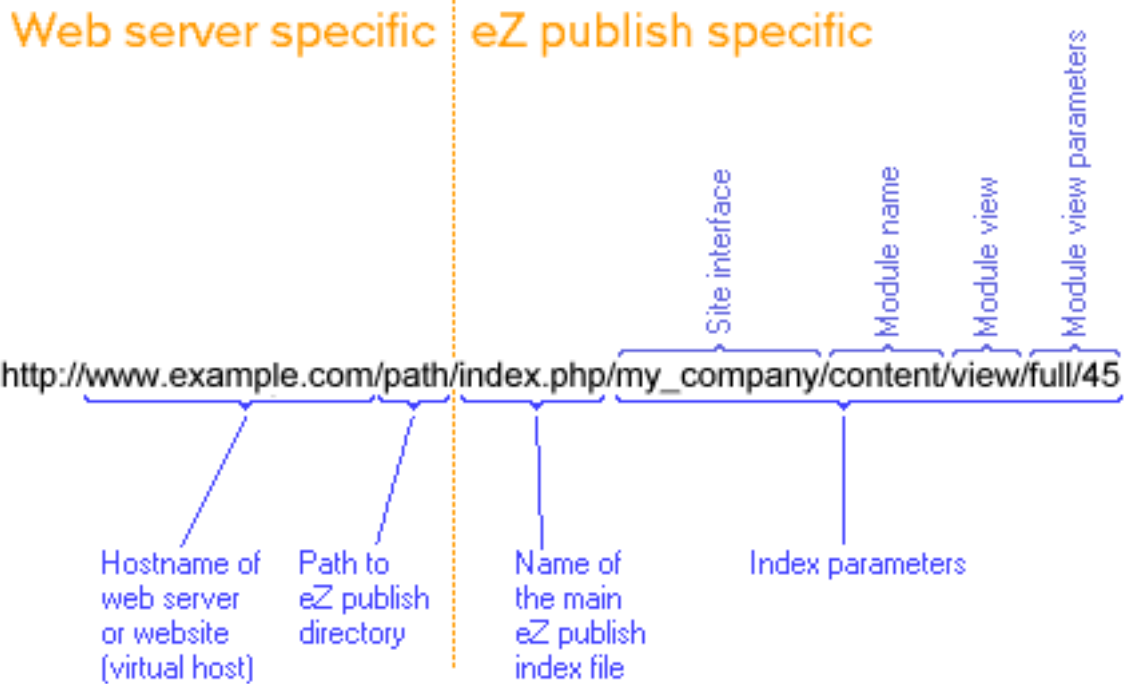
Un URL virtuale è semplicemente un alias di un arbitrario URL di sistema. Un URL virtuale è più carino, più semplice da ricordare e qualche volta più corto del corrispondente URL di sistema. URL virtuali sono spesso chiamati “URLs carini (nice)” o “URL aliases”.

Ogni nodo di contenuto è raggiungibile utilizzando un URL virtuale (eZ publish genera automaticamente aliases per gli oggetti contenuto che sono pubblicati). L'URL virtuale di un nodo (un oggetto contenuto pubblicato) è una versione semplificata del nome del nodo. Il nome del nodo è convertito in lettere minuscole, gli spazi sono rimpiazzati con degli underscores e così via. Gli URL virtuali (e le redirezioni, ecc.) possono essere anche aggiunte manualmente utilizzando l'interfaccia amministrativa.

L'illustrazione seguente spiega le varie parti di un URL virtuale di eZ publish:



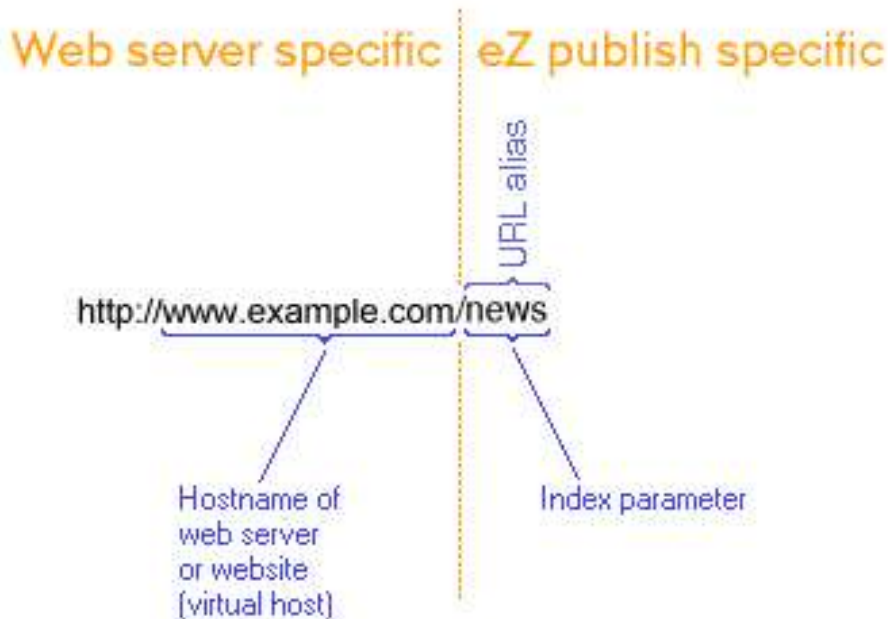
L'URL nella illustrazione sopra riprende la maggior parte delle installazioni di eZ publish con una configurazione di base/default.



Il web server può essere configurato per non mostrare il file index; in questo caso, “index.php” semplicemente scomparirà da tutte le richieste.

La configurazione di accesso di default di eZ publish è *URL*, che significa che il nome dell'interfaccia del sito è incorporata nelle URL di richiesta stesse. Se la configurazione dell'accesso al sito è *hostname*, allora il nome del sito sarà molto probabilmente incorporato nel nome dell'host (forse in forma di sottodominio). Se la configurazione dell'accesso al sito è *port*, allora un carattere “:” e un numero di porta saranno aggiunti al nome dell'host. Negli ultimi due casi (*hostname* e *port*), il nome dell'interfaccia del sito non sarà inclusa nell'URL.

Il diagramma sotto illustra una versione abbreviata (index nascosto, modo di accesso non-URL) dell'URL virtuale/aliased/niced nell'esempio precedente.



Sommario

TODO