

全國中小學校園 自由軟體應用諮詢中心

ZOPE 進階 講師研習班

教材编輯/黃敏松

內容目錄

1. 系統設定	
1.1. 使用 Apache 作為 Zope 的前端	
1.2. 虛擬主機的設定	5
1.3. 將 Zope 的物件儲存在 Data.fs 之外	10
2. Plone 的進階使用	14
2.1. portal catalog - 內建的搜尋引擎	
2.3. portal workflow - 設定工作流程的工具	
2.3. 使用 Cache Manager	
2.4. 表單驗證和執行流程	
2.5. 使用 MySQL 資料庫	
3. Programming	
3.1. 使用 Script (Python)	
3.2. 使用 External Method	48

1. 系統設定

本章系統設定要說明的範圍不是只在 Zope 內容,而是 Zope 與主機的其他服務相互配合的設定。

1.1. 使用 Apache 作為 Zope 的前端

如果你原本就是使用 Apache 作為 web server 的話, Zope 可以用以下的三 種方式與 Apache 搭配使用:

a) CGI: PCGI, FastCGI

b) Proxy Module

c) Rewrite Module

使用 Apache 作為 Zope 的前端,除了有與原本系統架構結合的優點之外, 還能讓多種不同的環境同時存在。譬如說除了 Zope 外,系統上還想用 PHPMyAdmin(PHP), OpenWebMail(Perl)等等這些不同的自由軟體程式的話, 使用 Apache 作為前端是最方便的方式了。我們要介紹的是使用 Rewrite Module 的方式,將瀏覽的 Request 重導給 Zope。

Apache 的設定檔以 Debian 為例是在 /etc/apache/httpd.conf,我們需要確定 是否已經將 rewrite_module 和 proxy_module 這二個模組載入。我們可以籍 由搜尋下列四行來確認:

LoadModule rewrite_module /usr/lib/apache/1.3/mod_rewrite.so LoadModule proxy_module /usr/lib/apache/1.3/libproxy.so

AddModule mod_rewrite.c AddModule mod_proxy.c

如果這四行的前面被加上「#」井字號當成是註解的話,就要先刪除井字號後存檔,然後重新啓動 Apache 才能使這二個 Module 可以使用。

接下來的是 Rewrite Module 的設定範例:

RewriteEngine On RewriteLog "/var/log/apache/rewrite_log" RewriteLogLevel 9 RewriteRule ^(.*) http://localhost:8080\$1 [P]

「RewriteEngine On」這一行才正式啓動了 Rewrite Modele, 「RewriteLog」設定重導的記錄檔,「RewriteLogLevel」的參數可以從 0~9 ,數字愈大記錄的內容愈詳細,設定正確後可以將數字歸零。最後這個「RewriteRule」就是設定重導的規則。RewriteRule的語法是:

RewriteRule Pattern Substitution [flags]

這裡的 pattern 就是「^(.*)」,「^」是指行首,「.」可以匹配任意字元,「*」是重複前一字元任意次數,「(.*)」將括號內的字元當作一組變數,可 在 Substitution 中使用 \$1 取用。

這 *pattern* 用在這的意思是將 URL 的 path 全部擷取。譬如 ServerName 是 www.abc.com, Request URL 是 http://www.abc.com/Members/song/index_html,這時「(.*)」會符合 「/Members/song/index html」這一整串。

Substitution 是重導的目的地,這裡我們把 Request 重導到本機的 8080 port,「\$1」會把「/Members/song/index_html」這一串加到 URL 的後面來。所以假設 Request URL 是:

http://www.abc.com/Members/song/index_html

經過了這個 RewriteRule:

RewriteRule ^(.*) http://localhost:8080\$1 [P]

就會變成:

http://localhost:8080/Members/song/index_html

在 Flags 的部份我們只用了一個 P,這是代表 Proxy 的意思,就是說我們 直接向重導後的 URL 取回網頁內容,再傳回給 Client 端,如此一來 Client 端並不會知道我們在中間作了代理。在這個範例中 Zope 是啓動在 8080 port,透過 rewrite_module 和 proxy_module 的重導和代理,讓 Client 端不需 要指定 port 來瀏覽網站,完全不會感覺到這中間的有任何的改變。

1.2. 虛擬主機的設定

虛擬主機的設定是讓同一台機器上可以服務多個 Domain name,讓不同的 Domain name 可以有自已的目錄和內容。譬如讓 <u>http://www.abc.com</u>使用 /abc.com 目錄,而 http://www.def.com 則使用 /def.com 這個目錄。

Zope 2.6 有二個物件可以用來設定虛擬主機。一個是舊的 SiteRoots 物件, 這是為了相容性的考量所以還保留著。另一個是新的 Virtual Host Monster 物件,接下來將會以這個物件為主來介紹虛擬主機的設定方式。

設定虛擬主機之前, DNS 的設定要先完成。而在我們的測試中就只以修改系統中的 hosts 這個檔案來達成我們的目的。 Unix-Like 的系統中這個檔案的位置是在 /etc/hosts, Windows 2000 則是在

c:\WINNT\system32\drivers\etc\hosts。我們就設成這樣一行:

127.0.0.1 localhost www.abc.com www.def.com

然後在 ZMI 根目錄新增二個目錄物件,各別取名為 abc.com 和 def.com, 建立時請勾選 Create public interface 的選項:

Add	Folder
A Fold <i>interf</i> The c Folde	der contains other objects. Use Folders to orga ace option creates an index document inside t create user folder option creates a User Folder r.
Id Title	abc.com
THE	Create public interface Create user folder Add

插圖 1-1 新增目錄

接下來在 ZMI 的根目錄中新增一個 Virtual Host Monster 物件,這個物件的 ID 取什麼名字都沒關係,我們就將他取名為 VHM 吧:

🖕 Zope on http://local.hout.8080 - M	onile Fuebad	ald ×		
File Edit View Go Book	narks Lools Help	Y		
😔 • 🛞 - 🧐 🔅 🔞	http://locaBort.8000/manage	. Q		
🚹 Mozilla Fizebird Halp 🚹 User	Support Forum 📋 Flug-in FAQ			
	Logged in as admin	Zope Quick Start 💌 Go		
Root Folder	Add Virtual Host Monster			
* I Control_Panel A Virtual Host Monster changes the URLs generated by all objects within the same Folder, using inform to it in special URL path elements. This is useful if you are using some rewriting tool (Apache or an Ace example) to insert these special elements into your URL.				
def.com To set the protocol ('http:/, 'https') and host ('www.foo.com') portion of generated URLs, insert 'Vent's the protocol, and the host into the path.				
Refresh	Insert "VirtualHostRoot" directly after the name of the Folder that is supposed to be the root of the virtual host.			
	For example, to publish Folder "/foo" as http://www.foo.com/, put a Virtual Host rewrite requests for that UPL to /VirtualHostBase/http/www.foo.com/foo/Vir	Monster in the root folder and tualHostRoot/		
	Values affected include DTML variables starting with URL or BASE, and the absolute	_url() methods of objects.		
	Id [VHM Add			
Dote	I			

插圖 1-2 新增 VHM

接著我們設定 VHM 的 Mappings tab ,指定虛擬主機的目錄,每一行就是一個虛擬主機的設定,設定的格式是:虛機主機/目錄。

Zope - Mozilla Frebrid	A REAL PROPERTY AND ADDRESS OF			
File Edit View Go Book	narks <u>I</u> ools <u>H</u> elp			Y
) 🚯 😒 😒 💿 ا	http://localhost/8080/VHM/manage_e	dit	· Q	
🚹 Mozilla Firebird Help 📋 User	Support Forum 📄 Plug-in FAQ			
	About	Ма	ppings	 1^
😤 Virtual Host Monste	rat / <u>VHM</u>			
this tab if you are using simple virtual hosting in a Zope in this list you are a address to fix things.	Apache or some other front-e bare Zope server. If you plac ikely to regret it , and will pr	nd server to rewrite e the hostname tha obably need to mana	requests. This is only t you use to manage ige Zope using its rav	y for your # IP
Last Modified	2003-12	-29 21:04		100
Each line represents a pa	th mapping for a single host (host/path), or a se	t of hosts (* .host/ p	oath).
www.abc.com/abc.com www.def.com/def.com				
Done				<u> </u>
				11

插圖 1-3 設定 Mappings

然後我們就可以瀏覽這二個虛擬主機了:

因為這裡的 Zope 是啓動在 8080 port 上,所以 URL 後面都加上了 8080,如果是單獨執行 Zope 的機器,可以將 Zope 啓動在 80 port 上。不然我們還

👛 Mozilla Firebird	
<u>File Edit View Go</u> Bookmarks <u>T</u> ools <u>H</u> elp	4
🚱 🕶 🅘 – 😪 👔 🚹 http://www.abc.com:8080/ 💽 🔍	
🕒 Mozilla Firebird Help 📑 User Support Forum 📑 Plug-in FAQ	
abc.com This is Page Template <i>index_html</i> .	
👑 Mozilla Firebird	
<u>File Edit View Go Bookmarks Tools H</u> elp	Y
👝 🖌 🚽 😪 🎲 🚺 http://www.def.com:8080/ 💽 🔍	
🚹 Mozilla Firebird Help 🚹 User Support Forum 📑 Plug-in FAQ	
def.com	
This is Page Template <i>index_html</i> .	
看圖 <i>1-4</i> 虛擬主機	

可以用上一節的方式以 Apache Rewrite Module 來重導 URL。

使用 Apache Rewrite Module 的話,不需要在 Virtual Host Monster 的 Mappings tab 裡頭作設定,只要在 Rewrite Rules 作設定即可,不過還是需要 有一個 Virtual Host Monster 的物件存在於 Zoep 的根目錄裡才行。

使用 Apache 虛擬主機的設定時要注意,一定要將第一個虛擬主機的設定 區塊保留給原來的設定值,也就是說新增的虛擬主機要由第二個虛擬主機的 設定區塊開始寫。我們用同樣的情況來作個設定範例:

```
NameVirtualHost *:80

<VirtualHost *:80>

ServerAdmin Song@song.idv.tw

ServerName localhost

DocumentRoot "C:/Program Files/Apache Group/Apache/htdocs"

</VirtualHost>

<VirtualHost *:80>

ServerAdmin Song@song.idv.tw

ServerName www.abc.com

RewriteEngine On

RewriteEngine On

RewriteLog "C:\Program Files\Apache

Group\Apache\logs\abc_rewrite_log"
```

RewriteLogLevel 9 RewriteRule ^/(.*)
http://localnost.8080/virtualHostBase/http/www.abc.com.80
/abc.com/virtualHostRoot/\$1 [P]
<virtualhost *:80=""></virtualhost>
ServerAdmin Song@song.idv.tw
ServerName www.def.com
RewriteEngine On
RewriteLog "C:\Program Files\Apache
Group\Apache\logs\abc_rewrite_log"
RewriteLogLevel 9
RewriteRule ^/(.*)
http://localhost:8080/VirtualHostBase/http/www.def.com:80
/def.com/VirtualHostRoot/\$1 [P]

(請注意,這裡的 RewriteRule 的部份因為過長而折行,實際上是同一行。) 設定完成後重新啓動 Apache,再試試瀏覽不加 port 的 URL:



插圖 1-5 虛擬主機

Rewrite Module 的用法在上一節介紹過了,這裡主要再解釋一下

RewriteRule 的部份。在 RewriteRule Substitution 前面的部份

http://localhost:8080/

和上一節一樣是重導到本機上的 port 8080,也就是 Zope 啓動的 port。接下來的 VirtualHostBase 之後到 VirtualHostRoot 之間的內容是傳給 VirtualHostMonster 物件的資料,所以 ZMI 的根目錄上一樣要有一個 VirtualHostMonster 的物件, ID 取名為什麼還是一樣沒關係,而且使用 Apache Rewrite Module 的設定法時,VirtualHostMonster 的 Mappings 裡不需要作任何的設定。

VirtualHostBase/http/www.def.com:80

是指定 Request 的 URL,也就是 Base 的 URL,這裡是被指定為 http://www.def.com/,這會影響到網頁內所有連結的 Base。接下來的路徑被 當作虛擬主機的根目錄,而緊接著的就是 VirtualHostRoot 這個特定字:

/def.com/VirtualHostRoot

最後的 /\$1 就是使用 RewriteRule 的 Pattern,上一節已經介紹過了。

這個範例讓二個虛擬主機使用同一個 Zope 裡的不同目錄來存放各自的網頁,也讓機器上的 Web Services 透過 Apache 的統合來保持規劃上的彈性,所以這個方式很適合直接套用到你目前的主機上。

最後,如果想將 Zope 上的某個目錄當作是原有站台的一個目錄,例如說 http://www.def.com 是你原有的站台,現在你想將 Zope 上的 plone 這個目錄 當成是 http://www.def.com/portal 這樣的 URL 時,要如果設定 Rewrite Rule 呢?這個時候有一個新的路徑特定字: _vh_,把他放在 VirtualHostRoot 的後面,隨後再接著你想要他變成怎麼樣的目徑名稱,像現在就該是 _vh_ portal,所以這個 Rule 就是:

RewriteRule ^/portal(.*) http://localhost:8080/VirtualHostBase/http/ www.def.com:80/plone/VirtualHostRoot/_vh_portal\$1 [P]

如果想要放在第二層目錄之下的話,如 http://www.def.com/zope/portal:

RewriteRule ^/portal(.*) http://localhost:8080/VirtualHostBase/http/ www.def.com:80/plone/VirtualHostRoot/_vh_zope/_vh_portal\$1 [P]

1.3. 將 **Zope** 的物件儲存在 **Data.fs** 之外

標準的 ZODB 是儲存在檔名為 Data.fs 的單一檔案,從早期的 LocalFS Product 開始, CMFOptions, ExtFile/ExtImage, ExternalFile, Ape等,都是在想要將 Zope 的物件儲存到其他 Storage 上的努力。這個想法起初是要避免單一的 ZODB 檔案過大,在早期 Linux 上的單一檔案有 2G 大小的限制時,這是 唯一預防的方法。當 Linux 和 Python 都突破了 2G 的限制之後(在 Windows 平台上的 Python 2.1 還是有單一檔案不能超過 2G 的限制),努力的方向改為方便載入檔案,共用儲存體,減少 Zope 負荷等。

這幾個 Products 中以 Ape 的彈性最大,作法最徹底,他將物件和物件的 Properties 都存到外部的儲存體上。目前 Ape 支援的儲存體除了 FileSystem 之外還可以儲存在 MySQL, PostgreSQL 等的 RDB 上,而且還可以依照特殊 的需求自行擴充其他儲存體的實作。

在 Zope 2.6.x 上使用 Ape 需要安裝 DBTab product, Zope 2.7 開始 DBTab 就會直接成為內建的 product 了。 DBTab 讓 ZODB 擁有擴充介面 的功能,而 Ape 則是實作出 Storage 的功能。安裝時,將 DBTab 和 Ape 解壓縮到 Products 目錄之後,再把位在 DBTab 目錄底下的 custom_zodb.py 作個 soft link 到 ZOPE 的安裝目錄,還有在 Ape 目錄底下的 dbtab.conf 也 作 soft link 到 ZOPE 的安裝目錄。(在 Windows 平台則是直接 Copy 即可)

修改 dbtab.conf 可以自訂 Storage 的設定。預設的 dbtab.conf 有二組設定:

[Storage: Main] type=FileStorage file_name=%(CLIENT_HOME)s/Data.fs [Database: Main] mount_paths=/

每組設定都由 [Storage: NAME] 和 [Database: NAME] 二個區段所組成, Storage 區段設定實際的儲存實體參數, Database 區段設定這個 Storage 在 ZODB 中的相關參數。以上的範例設定了一組叫作 Main 的 Storage, type=FileStorage 指定使用單一檔案作為儲存實體,這就和原本一般的 ZODB 的方式一樣。 file_name=%(CLIENT_HOME)s/Data.fs 指定檔案的路 徑和檔名,而 %(CLIENT_HOME)s 用的是 Python 格式化字串的語法,將 CLIENT_HOME 的變數值作為字串填入。如果 Zope 是安裝在 /opt/zope 的 話,上述的 file_name 的設定就等於是:

file_name=/opt/zope/var/Data.fs

Database 區段中只指定了 mount_paths , 這裡將 Main Storage 設定為 ZODB 的根目錄。接下來看第二組的設定:

[Storage: FS] type=apelib.zodb3.storage.ApeStorage factory=apelib.zope2.mapper.createFSMapper basepath=%(CLIENT_HOME)s/mnt

[Database: FS] class=apelib.zodb3.db.ApeDB cache_size=0 mount_paths=/fs container_class=OFS.Folder.Folder

這組設定是將物件儲存在 FileSystem 底下。在 Storage 的區段中, type 換成了 apelib.zodb3.storage.ApeStorage, factory 設定使用 apelib.zope2. Mapper. CreateFSMapper 作為建構元, basepath 設定存放的目錄路徑。

在 database 區段中除了 mount_paths 外,要指定 class, cache_size 和 container_class。可以依主機上配置的記憶體數量來調整 cache_size 的值, 建議先設定為 4000 再依照狀況來作修正。cache_size 是設定 cache 的檔案 數,而非使用的記憶體的大小。container_class 指定建立 mount point 目錄的 建構類別。這個範例中使用的 OFS.Folder.Folder 和在 ZMI 中指定建立 Folder 是一樣的類別,如果這個 mount point 是用在 Plone site 中,而且希望 他和一般在 Plone 中建立的目錄相同, container_class 就要這樣設定:

container_class=Products.CMFPlone.PloneFolder.PloneFolder

再進一步如果想使用 MySQL 作為儲存實體的話,下面是一組設定的範例:

[Storage: MySQL] type=apelib.zodb3.storage.ApeStorage factory=apelib.zope2.mapper.createSQLMapper module_name=MySQLdb kwparams=db:song user:my passwd:my

[Database: MySQL] class=apelib.zodb3.db.ApeDB mount_paths=/my

kwparams 指定連結資料庫的參數。這之前還需要先安裝 mysql-python:

http://sourceforge.net/project/showfiles.php?group_id=22307&package_id=157 75&release_id=102893

然後在 MySQL 建立 DB: song, User: my, 設定完 dbtab.conf 之後重新啓 動 Zope, 然後在 ZMI 中將 mount point 新增進來:



插圖 1-6 新增 Mount Point



插圖 1-7 建立 Mount Point

然後就可以看到 fs 和 my 這二個 Mount Point 的目錄已經建立完成,在這二個目錄中所建立的物件都會儲存到他們各自的 Storage 裡去。

👂 🔸 🌛 – 🤕 👷 🌇 🚹 http://127.0.0.1.8080/menage_mein		· G	
Mozille Fuebard Help 📋 User Support Forum 📋 Plug-in FAQ			
Contents View Properties	Security T	Undo T Ownership T	Find
Folder at /			Help!
	Accelerated	HTTP Cache Manager 💌	Add
Type Name	Size	Last Modified	
🗧 🏭 Control_Panel (Control Panel)		2004-01-13 20:20	
🗂 🛃 acl_users (User Folder)		1998-12-30 10:23	
🗖 🅩 browser_id_manager (Browser Id Manager)		2004-01-13 20:08	
🗆 🔇 error_log		2004-01-13 20:08	
T 🗋 fs		2004-01-13 20:20	
🗖 💊 index_html	1 Kb	2001-01-21 05:39	
T imy		2004-01-14 23:50	
🕝 🕪 session_data_manager (Session Data Manager)		2004-01-13 20:08	
Standard_error_message (Standard Error Message)	1 Kb	2001-01-21 05:39	
🗖 🏠 standard_html_footer (Standard Html Footer)	1 Kb	2001-01-21 05:39	
🗖 🍫 standard_html_header (Standard Html Header)	1 Kb	2001-01-21 05:39	
🗆 🗊 standard_template.pt	1 Kb	2004-01-13 20:08	
e (m) (2.11 (e) (e.11.1			

插圖 1-8 Mount Point 新增完成

2. Plone 的進階使用

本章中我們要介紹幾個 CMF/Plone 的工具,並以實際的例子來講解如何 使用這些工具。

2.1. portal_catalog - 內建的搜尋引擎

Plone 裡的 portal_catalog 工具就是 ZCatalog 的延伸, ZCatalog 是 Zope 內 建的搜尋引擎,他可以讓你分類搜尋 Zope 的所有物件。ZCatalog 有豐富的 詢問介面,你可以執行全文檢索,也可以一次搜尋多個索引欄位。甚至你 可以讓 ZCatalog 保持追蹤被索引物件的改變。ZCatalog 有二種使用模式:

Mass Cataloging

一次將大量匯集的物件編目 (cataloging)

Automatic Cataloging

當物件被建立或修改時自動編目

ZCatalog 根據你設定的 indexes 將物件符合的(欄位)資料記入,這些資料 就是當你對 ZCatalog 作搜尋時可被搜尋的內容。 當你在搜尋時,ZCatalog 會將符合你設定條件的物件丟回給你,這些物件並不是原本的物件,而是 ZCatalog 本身的記錄物件, 這些記錄物件會儲存一些參數,這參數叫作 Metadata,就是你在 ZCatalog 的 Metadata tab 中所設定的。

Contents	Catalog	Properties	Indexes	Metadata	Find Objects	Advanced	Undo	Security	Owner
획 ZCatalo	gat /Zo	o/ <u>AnimalC</u>	ataloq						Help!
Index Adde	d (2003-0	02-22 17:45)						
This list def any attribut	ines what es which	indexes the match an in) Catalog v dex in this	vill contain. Iist will ge	When ob t indexed	jects get c	ataloge	d, the valu	ues of
						DateInd	ex	•	Add
Name		I	ndex type		#	objects	La	ast modifie	d
🗖 Principia	SearchSo	ource z	ZCTextInde	эх	0		20	003-02-22	17:45
🗖 bobobas	se_modific	ation_time [DateIndex		0		20	003-02-22	17:43
🗖 id		F	FieldIndex		0		20	003-02-22	17:40
Remov	e index	Reindex	Clear inde	ex Selec	it All				

插圖 2-1 Indexes Tab

ZCatalog 的 indexes 有數種不同的型式,以因應不同的搜尋需求:

ZCTextIndex

如果你需要作全文檢索的話就用這個。他提供布林運算,括號優先運算,通用字搜尋,片語搜尋等功能。

FieldIndex

當你的欄位是特定的值的話可以用這個,他會將整個的內容都存起來。

KeywordIndex

這個 index 可以與 lines 型式的 property 搭配,他可以記錄一個序列的關鍵字,每 個關鍵字都像 FieldIndex 一樣整個被記錄起來,不過只要搜尋條件符合其中一個或 一個以上的關鍵字就會被當作的符合條件的。

PathIndex

搜尋物件的路徑,如果你的物件是依照目錄來作分類的話,可以考慮使用這個。

DateIndexes

類似 FieldIndex 的運作,但有針對 DateTime 的值作最佳化。

DateRangeIndexes

針對時間區段的搜尋所使用的。

TopicIndexes

TopicIndexes 是一個容納 FilteredSets 的地方, FilteredSet 由一個運算式和一組 ZCatalog 內部文件的辨識符。他可以提供一個經過預先運算過的結果列表,讓效能 的表現更好。

TextIndex

舊式的全文檢索用的 index 型式,為了相容性的原因所以還保留著。請使用 ZCTextIndex。

你可以使用 Z Search Interface 來建立簡單的搜尋介面及報表輸出。Z Search Interface 會產生二個 DTML Methods 或是 Page Templates。你只要 指定 ZCatalog,然後輸入搜尋介面及報表的 ID,再選擇你希望產生 DTML Methods 或 Page Templates,就可以讓 Zope 幫你作出基本的搜尋介面及報表檔了。你可以再自行加入你要的功能,或是直接剪貼到其他地方去。

我依照 ZopeBook 的範例作了一個 ZCatalog 叫 AnimalCatalog ,裡頭有一個 Lexicon 叫 zooLexicon ,這是作 ZCTextIndex 必須要的東西。還建了一個 ZCTextIndex 叫作 zooTextIdx ,是以物件的 PrincipiaSearchSource 來作 index , PrincipiaSearchSource 是 DTML Document 及 Method 的一個 API Method ,會傳回 DTML Document/Method 的本文區的內容。再建立二個 Metadata: id, title。

接著使用 Z Search Interface 產生了 Page Template 的搜尋介面 SearchForm 及輸出的報表 SearchResults。 先來看一下搜尋介面 SearchForm 的內容:

Add Search Int	erface	Help!
A Search Interface Search Interface wi displaying the searc	allows you to search Zope databases. T Il create a search-input form and a repo h results.	'he rt for
In the form below, s Methods) to be sea ids of the report an <i>report style</i> indicati	seachable objects are the objects (usua rched. <i>report id</i> and <i>search input id</i> are d search form objects that will be create as the type of report to generate.	lly SQL the ed.
Select one or more searchable objects	AnimalCatalog	
Report Id	SearchReport	
Report Title		
Report Style	Tabular 💌	
Search Input Id	SearchForm	
Search Input Title		
	C Generate DTML Methods	
	🖲 Generate Page Templates	
	Add	

插圖 2-2 建立搜尋介面

```
<html><body>
<form action="SearchResults" method="get">
<h2 tal:content="template/title_or_id">Title</h2>
Enter query parameters:<br>
ZooTextIdx
><tn>uname="zooTextIdx" width=30 value="">
```


form 的 action 直接設定為輸出的報表 SearchResults,因為只作了一個 index 所以只有一個 input, name 就是 zooTextIdx。再來看輸出報表 SearchResults 的內容,因為有點長所以我們分幾個段落來說明:

<html></html>	
<body tal:define="results here/AnimalCatalog;</td><td></td></tr><tr><td>start request/start python:0;</td><td></td></tr><tr><td>batch python:modules['ZTUtils'].Batc</td><td>h(results,</td></tr><tr><td></td><td>size=20,</td></tr><tr><td></td><td>start=start);</td></tr><tr><td>previous python:batch.previous;</td><td></td></tr><tr><td>next python:batch.next"></body>	

tal:define 是用來定義變數用的 ZPT 語法。 在我們定義 results here/AnimalCatalog 時,已經將 SearchForm 傳來的 zooTextIdx 轉給 AnimalCatalog,同時將 AnimalCatalog 搜尋的結果指定給了 results 這個變 數。results 是一個 Recode Object list。

ZTUtils 是 ZPT 的工具,在這裡使用了 Batch 來作批次呈現 results 的事 情。 ZTUtils 還有 math, random, sequence, standard, string 等的 Modules 可 以使用。接著來看下一段:

```
<a href="previous_url"
tal:condition="previous"
tal:attributes="href string:${request/URL0}?start:int=${previous/first}
">
previous <span tal:replace="previous/length">20</span>
results</a>
<a tal:condition="next"
tal:attributes="href string:${request/URL0}?start:int=${next/first}"
href="next_url">next <span tal:replace="next/length">20</span>
results</a>
```

如果有上、下頁的話上面這段才會顯示出來,tal:condition是 ZPT 判斷用的語法:

```
Title
 Id
 Data record id 
<div tal:repeat="result batch" >
<span tal:replace="result/title">title goes here</span>
 <span tal:replace="result/id">id goes here</span>
 <span tal:replace="result/data record id ">
   data record id goes here
  </span>
 </div>
```

tal:repeat 指定使用迴圈。將 results 批次處理過的 batch 一次一個指 定給 result,前面提過 recode object 會帶著 Metadata 的 attributes,所以 tal:replace 可以用 result/title 和 result/id 直接取用這二個 attributes。 tal:replace 是將所在的 HTML Tag 區塊取代掉,包含 Tag 本身:

```
<a tal:condition="previous"
tal:attributes="href string:$ {request/URL0}?start:int=$
{previous/first}"
href="previous_url">
previous_span tal:replace="previous/length">20</span> results
</a>
<a tal:condition="next"
tal:attributes="href string:$ {request/URL0}?start:int=$ {next/first}"
href="next_url">
next <span tal:replace="next/length">20</span> results
```


重複一次上、下頁的判斷和連結。

以下是這個範例的圖示:

SearchForm	Title	Id	Data record id
	Chilean four-eyed fr	og chilean_frog	-1200102376
Inter query parameters: ZooTextIdx frog OR python	Carpet Python	carpet_python	-1200102375
Submit Query			

插圖 2-3 搜尋介面

插圖 2-4 搜尋結果

如果只要作搜尋介面和結果的輸出,DTML Method 或 ZPT 就已經足夠用了,但是有時候會想把搜尋的結果用在其他地方,用 Script 來操作 ZCatalog 是一個好辦法:

return context.AnimalCatalog({'zooTextIdx' : 'python OR frog'})

像上面這行的 Script 會傳回給你像這樣的 recode object list:

[<mybrains instance at a447ca8>, <mybrains instance at 9f8ae28>]

你可以依照需要的 attributes 來設定 ZCatalog 的 Metadata 欄位,然後再 自行取出 recode object 的 attributes 來運用。

對於上述的介紹及範例瞭解之後,我們接著來看 Plone 的 portal_catalog 這個工具。本節一開始提到了 Automatic Cataloging 的方式,只要 Zope 的 物件繼承了 class CatalogAware 就會在新增修改時自動更新 Catalog 中的內 容。而 Plone 的所有 portal_type 物件則是繼承了 class CMFCatalogAware, 這個 class 會在物件被新增修改時更新 portal_catalog 中的內容,所以 Plone 裡的物件會被 portal_catalog 自動建立和修改索引。

底下這個 ZPT 的範例 docs_slot 是用來將最新的文件作成首頁中的一個 Box ,這是由 news_slot 修改來的,其中就是透過 portal_catalog 的查詢來 得到最新的文件:

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-

```
transitional.dtd">
<html xmlns:tal="http://xml.zope.org/namespaces/tal"
xmlns:metal="http://xml.zope.org/namespaces/metal"
i18n:domain="plone">
<body>
<!-- The news box -->
<div metal:define-macro="docsBox"
     tal:define="results python:request.get('docs',
here.portal catalog.searchResults( Type = ['Document'
                                          , 'Wiki Page']
                                  , sort on='Date'
                                 , sort order='reverse'
                                 , sort limit=10
                                 , review state='published'));"
     tal:condition="python:test(template.getId()!='docs' and results, 1,
0)">
  <div class="box">
     <h5 i18n:translate="box docs">Docs</h5>
     <div class="body">
       <tal:block tal:repeat="obj results">
          <div tal:define="oddrow repeat/obj/odd"
             tal:attributes="class python:test(oddrow, 'content even',
'content odd')">
            <a href=""
                tal:attributes="href obj/getURL">
              <img tal:replace="structure python:path('here/%s' %
obj.getIcon)" />
              <span tal:replace="python:test(obj.Title, obj.Title,
obj.getId)"> Extended Calendar Product </span>
            </a>
            <div class="boxDetails"
               tal:content="python:'%s, %s' %(obj.Creator,
here.toPortalTime(obj.modified))">July 7, 08:11</div>
          </div>
       </tal:block>
     </div>
  </div>
</div>
</body>
</html>
```

傳入參數給 portal_catalog 時,那些參數名稱就是在 portal_catalog 的 indexes tab 中的 index 的名稱,像上例中的 Type = ['Document', 'Wiki Page'] 使用 list 格式可以指定給參數多個值。參數 sort_on 可以指定使用那一個 index 來排序, sort_order 指定遞升或遞減的排序,預設爲遞升排序,可以使 用 deascending 或 reverse 指定使用遞減排序,如 sort_order='reverse'。 sort limit 指定傳回蒐尋結果的數量。

docs_slot 建立完成後我們可以透過設定首頁的 right_slots 或 left_slots Properties 將這個 docs_slot 加入到頁面裡:

here/docs_slot/macros/docsBox

用 docs_slot 的範例修改 Type 的值就可以作出不同物件的 Box 來,再依 照不同的 Type 修改 sort_on, sort_order,這樣就可以自製出你自已的 Box。

2.3. portal_workflow - 設定工作流程的工具

工作流程就是完成工作的一連串的動作,在不同的組織中通常都會有自已的一個或多個工作流程。所以工作流程的工具必須要可以依照需求來調整。 在 Plone/CMF 中提供了一個工具 - portal_workflow 用來管理文件的發佈 流程。



插圖 2-5 文件的發佈流程

我們可以看到上圖中共有四個狀態 (States),狀態間有說明和箭頭連結, 這是指在狀態間變換時的動作和路徑;在不同的狀態時可以設定文件不同的 瀏覽和修改內容的權限;在狀態變換的動作上可以設定允許不同的角色來執

Discussion Ite	m (Default)
Document	Default
Event	(Default)
Favorite	(Default)
File	(Default)
Folder	folder workflow
Image	(Default)
Link	(Default)
News Item	(Default)
Topic	folder workflow
(Default) Change	plone woddflow

插圖 2-6 指定 Workflow

行;在某個需要特定角色審查狀態的物件,可以在特定角色登入時顯示提醒的文字和連結。這些就是 portal_workflow 工具所提供的功能。

在 portal_workflow 裡可以建立不同的 Workflow 設定,並且可以指定每 種物件要使用哪一種 Workflow 或是不使用。Plone 預設的物件除了 Folder 和 Topic 使用 folder_workflow 外,其他的物件均是使用 plone_workflow。



插圖 2-7 Workflows

要新增或修改 Workflow 的設定要到 portal_workflow 的 Contents tab 頁面, Plone 預設有二個 workflow: plone_workflow, folder_workflow

在這裡可以新增一個 Workflow,也可以點選一個現有的 Workflow 進入 修改設定。進入現有的 Workflow 時會看到 **Properties** tab 的頁面,在 Properties tab 中只有 Title 的欄位可以修改。

我們再進入 **States** tab 頁面,這裡可以看到四種 **States**: pending, private, published, visible。如果新增 Plone site 時 site type 選擇 Default Plone,則 在這裡就會看到 visible 的前面會有個「*」星號。這是表示 visible 的物件 建立的時候初始的 state 就是 visible。如果新增 Plone site 時 site type 選擇 了 Private Plone site 的話,星號就會在 private 的前面,也會多一個 state 叫 作 public,而且 states 的 **Permissions** 設定也不大一樣。在 **States** tab 這裡 也可以重設物件初始的 state。

我們再切換到 Workflow 的 Transitions tab 來看,這裡的每一個 transition 就是一個 States 之間轉換的路線。transition 設定了轉換之後的 state 和可以



插圖 2-8 Workflow States tab



插圖 2-9 Workflow Transitions tab

執行轉換動作的角色。也可以指定在轉換前、後執行 Script 來作其他的事。

Workflow 裡可以設定一些變數,用來記錄 state 轉換時的相關資料。這些資料可以設定是否儲存。這是在 Variables tab 頁面裡設定的。

Workflow Variables at /plone/portal_workflow/plone_workflow,	<u>variables</u>
🗖 action The last transition	
Available to catalog No	
Stored in status: Yes	
Default expr transition/getId nothing	
🗖 actor The ID of the user who performed the last transition	
Available to catalog No	
Stored in status: Yes	
Default expr. user/getId	
Comments Comments about the last transition	
Available to catalog No	
Stored in status: Yes	
Default expr python:state_change.kwargs.get('comment', '	1)
review_history Provides access to workflow history	
Available to catalog No	
Stored in status: No	
Default expr. state_change/getHistory	
Requires permission Request review of Review portal conten	τ
🗆 time Time of the last transition	
Available to catalog No	
Stored in status: Yes	

插圖 2-10 Workflow Variables tab

在 plone_workflow 中, pending 的物件是需要具有 Reviewer 角色的成員 才能夠執行 publish 的 transition,將他轉變成 published 的狀態,或是執行 reject 的 transition 將他退回到 visible 的狀態。所以,我們需要通知具有 Reviewer 角色的成員有 pending 中的物件需要他來審核。 在 Worklists tab 裡就可以設定這樣的通知清單,這會透過 portal_action 工具通知登入的相關 角色成員。

Proper	es States Transition	us 🛛 Variables 🗍 W	orklists Scripts Per	nissions 🚺 Unde
Worklists at	plone/portal_workflov	v/plone_workflow	/worklists	
reviewer_qu	ue Reviewertasks			
Catalog	natches review_state =	pending		
Kequire	permission Review port	al content		
inkme Delete				
id a work	list			
	Add			

插圖 2-11 Workflow Worklists tab

2.3. 使用 Cache Manager

Cache 是一個暫時儲存經常存取物件的空間,我們會使用 Cache 只有一個 原因,就是速度。

Zope 有一些動態內容物件是需要經過計算後才能得到頁面的輸出結果, 例如 ZPT, DTML 或是 Script(Python),在每次被瀏覽時都需要被計算。如 果是簡單的計算那沒有問題,怕的是非常複雜的 ZPT, DTML 或 Script 需要 花很多時間處理,或是需要透過存取遠端主機才能取得資料,這也需要一些 時間。如果在 ZPT, DTML 或 Script 裡使用大量的迴圈計算,或是呼叫很多 的 Script(Python) 等等,會用到大量的運算時間,我們稱之為「昂貴」的。

使用 Cache 將這些「昂貴」的 ZPT 或 Script 的運算結果儲存起來重覆使用,這樣可以大幅度的提升網站的反應速度。第一位瀏覽這些「昂貴」的頁面的人會碰到緩慢的回應,但是當計算的結果被 Cache 記住之後,隨後的瀏覽者直接取用 Cache 的內容就不需要再經過計算,網站的回應會非常的快速。

使用 Cache 的機制也有些需要考慮的因素,像是 Cache 資料的生命週期,如果週期太長到無法呈現正確而即時的資料的話,這不會是我們所要的。還有個人化的資訊並不能提供給其他人,所以這部份的資料並不適合使用 Cache 的機制。

透過設定 Zope 的 Cache 規則可以排除上述的問題, Cache 規則可以設定 哪些的物件要被 Cache。 Zope 使用 Cache Manager 物件來設定 Cache 的 原則。Zope 提供二種 Cache Manager 物件: Accelerated HTTP Cache Manager, RAM Cache Manager。

Accelerated HTTP Cache Manager 實際上並沒有 Cache 住計算的結果,而 是透過修改 HTTP 回應的標頭,告訴外部的 Cache 主機 (如: Squid)哪些的 內容是需要 cache 的。這裡不會討論 Squid 的架設,但是我們來看一下 Accelerated HTTP Cache Manager 改變了什麼部份。

我們可以直接使用 telnet 來取得 HTTP 回應,這樣可以直接看到所有回應的內容,包括標頭。在新增 Cache Manager 之前,我們先來看看上一節的範例 docs_slot 的 HTTP header:

telnet 127.0.0.1 8080 Trying 127.0.0.1... Connected to localhost (127.0.0.1). Escape character is '^]'. GET http://127.0.0.1:8080/plone/docs_slot HTTP/1.1 HTTP/1.1 200 OK Server: Zope/(Zope 2.6.3 (binary release, python 2.1, win32-x86), python 2.1.3,win32) ZServer/1.1b1 Date: Sun, 18 Jan 2004 11:46:11 GMT Content-Type: text/html Etag: Content-Length: 691

使用 telnet 連結本機的 8080 port 之後就進入等待輸入的畫面,我們如上 般輸入 GET http://127.0.0.1:8080/plone/docs_slot HTTP/1.1 之後要按二下 Enter 鍵,因為輸入最後是以空白行作為結尾的確認。然後我們就會得到一 連串的回應,最開頭的部份就是標頭,標頭之後空一行接著的才是網頁的內 容。上述只列出標頭的部份。

接下來我們新增一個 Accelerated HTTP Cache Manager 在 Plone site 的根目錄裡,輸入 Id 後按下 *Add* 鍵就完成新增的動作。接下來要選擇要 Cache 的物件,這部份要在 cache manager 的 Associate tab 中設定:

-	Propert	ies Y	Statistics	Associate
MACC	elerated HTTP C	ache Mana	ger at /plone/	cacheAccelerated
Select v settings	which objects sho " permission are s	uld be cach hown.	ed using this ca	che manager. Only tho:
	💊 Members/	/index_html(Member list)	
	D portal_ski	ns/custom/ Select All	docs_slot	
Locate Of the	cacheable obje type(s):	Cts: C All Accelera Browser CMF Ca CMF Co	C Associated w ed HTTP Cache Mana Id Manager ching Policy Manager re Content	vith this cache manage
		IM Sea	rch subfolders	

插圖 2-12 加入物件到 cache manager

先搜尋出所有可被 cache 的物件, 勾選要 cache 的物件後按下 Save Changes 鍵儲存設定。然後我們再來看看 HTTP header 的改變:

<pre># telnet 127.0.0.1 8080 Trying 127.0.0.1 Connected to localhost (127.0.0.1). Escape character is '^]'. GET http://127.0.0.1:8080/plone/docs_slot HTTP/1.1</pre>
HTTP/1.1 200 OK Server: Zope/(Zope 2.6.3 (binary release, python 2.1, win32-x86), python 2.1.3,win32) ZServer/1.1b1 Date: Sun, 18 Jan 2004 11:48:05 GMT Content-Type: text/html Expires: Sun, 18 Jan 2004 12:48:05 GMT Cache-Control: max-age=3600 Etag: Content-Length: 691

可以看到經過 Accelerated HTTP Cache Manager 的設定後多出了二行的標 頭,指定了資料期限和 Cache 的秒數。

接下來在 Plone site 的根目錄裡新增一個 RAM Cache Manager ,然後在他的 Associate tab 中找出要加入 Cache 的物件:

	Properties	Statistics	Associate
➡ RAM Cacl	ne Manager at ,	/plone <mark>/cacheRAM</mark>	
Select which settings" perr	objects should b nission are showr	e cached using this ca n.	ache manager. Only those
	Members/inde	_html(Member list)	
Locate cach	portal_skins/ci Save Changes Sel eable objects:	ect All	with this cache manager
Of the type(s):	All Accelerated HTTP Cache Man Browser Id Manager CMF Caching Policy Manager CMF Core Content	
		Search subfolders	

插圖 2-13 加入物件到 RAM Cache Manager

接著我們再使用 telnet 的方式測試一下:

telnet 127.0.0.1 8080
Trying 127.0.0.1...
Connected to localhost (127.0.0.1).
Escape character is '^]'.
GET http://127.0.0.1:8080/plone/docs_slot HTTP/1.1

HTTP/1.1 200 OK Server: Zope/(Zope 2.6.3 (binary release, python 2.1, win32-x86), python 2.1.3,win32) ZServer/1.1b1 Date: Sun, 18 Jan 2004 11:50:03 GMT Content-Type: text/html Etag: 1Content-Length: 691

從上面的標頭可以看到和沒有設定 Cache 之前是一樣的,再看看 RAM Cache Manager 的 Statistics tab 的顯示:

Propertie	s T S	itatistics T As	sociate	Undo T O	wnership T	Security
➡ RAM Cache M	lanag	er at /plone	/ <u>cache</u> l	RAM		Help!
Memory usage is cached data. The accessed entries made using the <i>re</i>	appro: cach since cent	ximate. It is b e is cleaned u the last clear <i>hits</i> counter.	ased on ip by rem iup opera	the pickleo noving the ation. The	d value of 1 least frequ determinat	the Jently Jion is
Path	Hits	Recent Hits	Misses	Memory	Views	Entries
/plone/docs_slot	0	0	1	961	<default></default>	1
/plone/docs_slot	0	0	1	961	<default></default>	1

插圖 2-14 RAM Cache 的使用統計

從上圖中可以看到被瀏覽的次數和被 Cache 的實體數。被 Cache 的實體 依照 Properties tab 中 Request Variables 的設定,同一個檔案可以儲存不只 一個的實體,然後依據當次瀏覽的 Request Variables 丟出適合的實體,如果 沒有適合的實體時,就會重新計算動態內容後回應給瀏覽端並儲存成另一個 實體。實體數的上限可以在 Threshold entries 中設定。

➡ RAM Cache Manager	at /plone/ <u>cacheRAM</u>	Help
Title		
REQUEST variables	HTTP_ACCEPT_LANGUAGE	
Threshold entries	1000	
Maximum age of a cache entry (seconds)	3600	
Cleanup interval (seconds)	300	
	Save Changes	

插圖 2-15 設定 Cache Manager Properties

使用 RAM Cache Manager 就像用「空間」來換取「時間」,使用時要注意記憶體的使用量,不要無限制的將物件加入。

在 Plone site 中使用的 slot box 是用到 ZPT 的 macro , 這部份是沒有辦 法使用 Cache Manager 的,所以要將 slot box 加入到 Cache Manager 中的 話需要修改 slot box 的寫法和 Properties 中 right_slots 的設定。

以 docs_slot 的範例來修改,要將 <html> <body> </body> </html>等 Tag 去掉,直接當成首頁的一個部份引用進來。不把他視為 macro 引用時, Zope 會先計算出頁面的結果,然後才將結果插入到首頁來。底下就是修改 後的 docs_slot,為了區別,我們將修改後的檔案命名為 docs_box:

```
<div metal:define-macro="docsBox"
tal:define="results python:request.get('docs',
here.portal_catalog.searchResults( Type=['Document', 'Wiki Page']
, sort_on='Date'
, sort_order='reverse', sort_limit=10
, review_state='published'));"
tal:condition="python:test(template.getId()!='docs' and results, 1,
0)">
<div class="box">
<h5 i18n:translate="box docs">Docs</h5>
```

```
<div class="body">
     <tal:block tal:repeat="obj results">
    <div tal:define="oddrow repeat/obj/odd"
         tal:attributes="class python:test(oddrow, 'content even',
'content odd')">
     <a href="" tal:attributes="href obj/getURL">
        <img tal:replace="structure python:path('here/%s' %
obj.getIcon)" />
        <span tal:replace="python:test(obj.Title, obj.Title, obj.getId)">
        Extended Calendar Product </span>
     </a>
     <div class="boxDetails"
           tal:content="python:'%s, %s' %(obj.Creator,
here.toPortalTime(obj.modified))">July 7, 08:11</div>
     </div>
     </tal:block>
   </div>
   </div>
</div>
```

修改完 docs_box 之後就可以將 docs_box 加入到 RAM Cache Manager 裡。 然後在 Plone site 根目錄 Properties tab 中的 right_slots 加入一行:

portal/docs_box

如此就可以用 RAM Cache Manager 將 slot box 計算出來的結果 Cache 在記憶體中,加快 Plone site 的回應速度。

Propertie	:5	Statistics As	sociate	Undo	Ownership	Security
➡ RAM Cache M	anage	er at /plone/	cacheRA	M		Help!
A CONTRACT OF A DESCRIPTION OF A DESCRIP		the second s	CONTRACTOR AND	the second s	the state of the second st	the second
Memory usage is a data. The cache i entries since the l recent hits count	approx s clear ast cle ar.	imate. It is ba: ned up by remi eanup operatio	sed on th oving the n. The de	e pickled v least frequ sterminatio	alue of the Jently acce n is made u	cached ssed sing the
Memory usage is a data. The cache i entries since the l <i>recent hits</i> count(Path	approx s clear ast cle ar. Hits	imate. It is ba: ned up by remo anup operatio Recent Hits	sed on th oving the n. The de Misses	e pickled v least frequ eterminatio Memory	alue of the Jently acce n is made u Views	cached ssed sing the Entries

插圖 2-16 docs_box 使用統計

Click 上圖 Path 欄位的「/plone/docs_box」會進到 docs_box 的 Cache tab 畫面,在這裡可以按下 Invalidate 鍵來更新 Cache 的內容,也可以選擇要使 用那一個 Cache Manager。

Page Template at	/plone/docs_box	Help
ache this object usi	ng: cacheRAM	
Save Changes	(None)	
and a second	cacheRAM	
ache Settings	cacheAccelerated	
Invalidate		

插圖 2-17 更新 Cache 的內容

2.4. 表單驗證和執行流程

Plone 提供一個工具 portal_form 和 portal_navigation 可以作表單的驗證和 表單執行的流程控制。要使用表單的驗證前先要在 portal_properties / form properties 中設定要驗證的 ZPT 名稱和其驗證的 Script 名稱。

Prope		
Plone Property Sheet at 🏼 /	plone/portal_properties/ <u>form_</u>	properties
Properties allow you to assig Changes".	n simple values to Zope objects. T	o change pr
Name	Value	Туре
title	Form Properties	string
🗆 link_edit_form	validate_id,validate_link_edit	string
🗆 newsitem_edit_form	validate_id,validate_newsitem_edit	string
🗖 document_edit_form	validate_id,validate_document_edit	string
🗖 image_edit_form	validate id validate image edit	string

插圖 2-18 表單驗證 Script 的設定

我們看 document_edit_form 對應的是 validate_id, validate_document_edit 。validate_id 是個通用的 Script ,可以檢查物件的 Id 是否符合規則。validate_document_edit 就是針對 document_edit_form 的表單欄位作驗證,同時也可以對其他的事項作驗證。像 validate_document_edit 在表單欄位上只針對 title 欄位作驗證,然後也對於上傳的檔案作檢查。

Properties	Undo 📔 Owne
Plone Property Sheet at /plone/portal_pr	operties/ <u>navigation_properties</u>
Properties allow you to assign simple values to Changes".) Zope objects. To change property
Name	Value
title	Navigation Properties
🗖 default.link_edit_form.failure	link_edit_form
default.link_edit_form.success	script:link_edit
🗖 default.link_edit.failure	link_edit_form
default.link_edit.success	action:view
<pre>default.document_edit.success</pre>	action:view
🗖 default.document_edit.failure	action:edit
<pre>default.document_edit_form.success</pre>	script:document_edit
🗖 default.document_edit_form.failure	document_edit_form
□ default.newsitem edit.success	action view

插圖 2-19 執行流程的設定

執行流程的部份在 portal_properties/navigation_properties 裡設定。設定的 規則是:

Name	Value
[type].[action].[states]	[next thing to do]

每一個執行結果有二種可能的 state: success or failure, Value 就是設定不同的 state 時要執行的下一步的動作。上圖中可以看到:

Name	Value
default.document_edit_form.success	script:document_edit
default.document_edit_form.failure	document_edit_form

當 document_edit_form 執行成功時就接著執行 script:document_edit , 當執 行失敗時又回到 document_edit_form 的頁面。Value 的值可以是:

```
PAGE_TEMPLATE
使用 ZPT 的網頁顯示目前的內容
action:ACTION
執行目前物件的 ACTION
script:SCRIPT
執行 SCRIPT 在目前的內容
url:URL
重導到指定的 URL
```

當 document_edit_form 執行成功後接著執行 script: document_edit 時,又可以有二個 states,所以可以再設定 default.document_edit.success 和 failure 的 Value 來決定下一個步驟。

要讓表單可以使用 Plone 的表單驗證和流程控制,在撰寫 ZPT 時有二點 要注意:

<form>的 action 須要指定 here.portal_form_url(template.id),底下是 document_edit_form ZPT 的片段:

<form class="group" name="edit_form"

```
action="document_edit"
method="post"
enctype="multipart/form-data"
tal:attributes="action python:here.portal_form_url(template.id)">
```

表單內必須含有一個 hidden 欄位, name = "form_submitted", value = template/id:

<input type="hidden" name="form_submitted" value="1" tal:attributes="value template/id" />

這二點都符合才能使用 portal_form 工具。下一節中的範例就會用到 portal_form 的工具來處理 validation 和 navigation。

2.5. 使用 MySQL 資料庫

Zope 可以透過資料庫聯接介面使用關聯式資料庫。目前在 Zope 裡可以使用的關聯式資料庫包括: Oracle, DB2, PostgreSQL, SAP DB, Sybase, SQLServer, Interbase/Firebird, Informix, Gadfly, 和本節的主題: MySQL。

要在 Zope 內使用 MySQL 資料庫,有二個東西要先安裝。一個是 Python Module: MySQLdb,一個是 Zope Product: ZMySQLDA。在 SourceForge上 的首頁是: <u>http://sourceforge.net/projects/mysql-python/</u>這裡就可以同時抓到 MySQLdb 和 ZMySQLDA 了。在 Zope 2.6.3, Python 2.1.3, MySQL 4.0.17 的 搭配上,安裝 MySQLdb (mysql-python) 0.92 和 ZMySQLDA 2.0.8 的版本是 可以正常運作的。安裝完成後重新啓動 Zope。

在 Zope 中用來建立 MySQL 資料庫連結的物件叫作 Z MySQL Database Connection,建立了資料庫連結物件後就可以使用 Z SQL Method 的物件透 過他來存取 MySQL 的資料了。

Contents View	Properties Security Undo Owne	ership Find
Folder at /son	g	Help!
	Select type to add	▼ Add
	SiteRoot User Folder	
Type Name	Version Virtual Host Monster	ified
🔲 这 acl_users (U	Vocabulary	05
🔲 🖓 add-board (¥	Z Gadfly Database Connection Z MvSOL Database Connection	
	Z SQL Method	
Database Co	Z Search Interface ZCatalog	17
E Bindoy html (Zope Tutorial	• 21

插圖 2-20 新增 Z MySQL Database Connection

新增 Z MySQL Database Connection 有三個地方要輸入, Id 和 Titel 可以 隨你的意思來決定, Enter a Database Connection String 的部份就得照 MySQL 上所設定的來填寫了。MySQLDA 可以在 MySQL 的 Database 裡新 增、修改、刪除 Table, Record, 但是無法新增、修改、刪除 DB,所以你必 須先在 MySQL 裡新增一個 DB 然後設定好 user 的權限及密碼,將 Database user password 填入 Enter a Database Connection String。 如圖中下 方的說明, Connection String 的語法是:

[+/-]database[@host[:port]] [user [password [unix_socket]]]

所以當 MySQL 不是在同一台機器上時,可以用 Database@host:port user password 來建立 Z MySQL Database Connection。輸入後按下 <u>Add</u> 按鍵來新 增這個 Connection。如果出現 Error 請參照 Error Value 來修正你的錯誤, 通常會是 MySQL user 權限的問題。

Add Z MySQL	Database Connection
Id	connect_song
Title	Z MySQL Database Connection
Connection String $\frac{1}{2}$	database user passwo.
Connect immediately	
	Add
¹ Connection Strings	
The connection str of the form:	ing used for Z MySQL Database Connection is
[+/-]database[@	nost[:port]] [user [password [unix_socket]]]
or typically:	
database user p	assword

插圖 2-21 新增 Z MySQL DC 輸入畫面

新增完成後,可以看到這一個物件:

Contents View Properties Securi	ty 🕇 Undo	Ownership Find
🗎 Folder at /song		Help!
Select type to ac	łd	▼ Add
Type Name	Size	Last Modified
🔲 度 acl_users (User Folder)		2001-07-05 01:32
☐ 公 add-board (新增流言)	1 Kb	2001-07-23 08:00
□ 🗐 connect_song (Z MySQL Database Connection)		2001-07-17 08:37

插圖 2-22 Z MySQL DC Icon

你可以點選這個物件來編輯他,如下圖:



插圖 2-23 編輯 Z MySQL DC

可以看到目前的狀態是 The database connection is open. 下面有一個 <u>Close</u> <u>Connection</u> 的按鍵,可以關閉與資料庫的連結。如果你要修改 Connection String 可以到 Properties tab 中去修改,請見下圖:

Status Properties	Test Security Undo Ownership Browse
Z MySQL Databa	ase Connection at <u>/song</u> /connect_song
Edit Z MySQI (connected)	Database Connection
Id <i>Title</i>	connect_song Z MySQL Database Connection
Database Connection String	database user passwd
Connect immediatel y	
	Change

插圖 2-24 編輯 Z MySQL DC Properties

在 MySQL 資料庫管理的部份,我們使用了 MySQL Control Center (aka mysqlcc),這是一個 GPL 的軟體,他有 Linux(x86, glibc 2.2, 2.3) 和 Windows (95/98/2k/xp) 和 Source 可以下載,下載的 URL 是:

http://www.mysql.com/downloads/mysqlcc.html

下一個範例中我們要在 Plone 建立一個 ZPT 作為報名表單,透過 Portal_form 工具作 validation 和 navigation 的控制,並將報名資料儲存在

MySQL 資料庫中。

我們用 mysqlcc 先建立了下一個範例的資料庫和使用者,使用者和密碼我 們都定為 zope,這個密碼稍後在 Z MySQL Database Connection 中會用到:

MySQL Control Center 0.9.4-beta - [Console Manager]	
🔄 Console Options HotKeys <u>Wi</u> ndow <u>H</u> elp	B×
File View Action F Create Database	? ×
Bater new Database name	
MySQL Servers / Pyzope	
Broot@127.0.0.1:330	Cancel
Tables	
pyzope	
Tables Username zope	Allow access to 🖉
Host localhost	- Global Privileges
	- 🗗 🔲 mysql
Tables Privilemen	(🗹 🌒 ругоре
	🕂 - 🗌 🛯 song
	🔄 🦾 🗋 🚺 test
🖸 mot@% Insert	
groot@local Update	
- Song@locs Delete	
······································	
with GRAN I option	
User created sur 🗯 Delete User	Add Close
SYou don't have	
U[root@127/U.0.1:3306] Querying MySQL Server for Table information in database: p Messages / SOL Debug /	yzope
I (THERE) (DEDITE)	1

插圖 2-25 新增資料庫和使用者

然後將 PyReg.sql 匯入到 pyzope 資料庫中,這是一個 registry Table 的結構。PyReg.sql 可以在教材包中找到。使用 mysqlcc 匯入 PyReg.sql 的方式 是先點選 pyzope 資料庫然後按下圖示功能鍵「SQL」,再將 PyReg.sql 的 內容貼入到 Query 1 tab 欄位中,按下「!」即可執行 SQL Code 新增出 registry Table 在 pyzope 資料庫中。

接下來在教材包中有一個 PyReg.zexp 的檔案,將他 copy 到 \$Zope/import 的目錄中,然後到 ZMI 的 /plone/portal_skins/custom 中按下 Import/Export 按鍵將 PyReg.zexp 匯入,這會在 custom 目錄中新增出 PyReg 的目錄,裡 面有三個 ZPT,一個是報名表單: PyzopeRegForm,一個顯示報名清單的 網頁: PyzopeRegList,和顯示報名者報名資料的網頁: PyzopeRegPerson 。有一個 Z MySQL Database Connection: MDC_pyzope。五個 Z SQL Method,用來處理資料的新增,修改,搜尋等動作。三個 Script(Python)用 來預先處理表單內容後呼叫 Z MySQL Method 執行資料的增修動作,和作 為 validation 的檢查之用。匯入 PyReg.zexp 請注意,他是在 UTF-8 的環境制作的,所以匯入的環境也要是 UTF-8 才行,不然畫面就會有亂碼。

Contents View Properties	Security
Folder at /plone/portal_skins/custom/PyReg	
Type Name	Size
🗖 🗊 MDC_pyzope (Z MySQL Database Connection)	
🗖 📾 M_getgroupclass_registry	
🗖 🎝 M_insert_registry	
🗖 📾 M_listbyclass_registry	
🗖 뤍 M_person_registry	
🗖 💼 M_update_disable_registry	
🗖 🍓 PyzopeReg (Personalization Handler.)	2 Kb
🗖 🍓 PyzopeRegDisable	1 Kb
🗖 🗊 PyzopeRegForm (Register Form)	18 Kb
🗖 🗊 PyzopeRegList (Register List)	5 Kb
🗖 🗊 PyzopeRegPerson (Register Person)	2 Kb
🗖 🧶 validate_PyzopeRegForm (Validates Pyzope Register Form) 1 Kb
Rename Cut Copy Delete Import/Export Se	lect All

在 portal_skins 中想讓新增的目錄可以像其他的目錄一般可以在 plone site 中的任何地方被呼叫使用的話,必須在 portal_skins 的 Properties tab 將這個 路徑加到你希望的 skin path 中才行。

Contents	T	Properties	T	Viev
/ CMF Skins Tool at /pl	one/port	al_skins		
Properties changed. (2004	-01-24 13	3:30)		
Skin selections				
Name			Laye	ers (i
🗆 Plone Autumn	custom, cus	tom/PyReg, ar	chexam	ple,arc
🗆 Plone Core	custom, cus	tom/PyReg, ar	chexam	ple, arc.
🗆 Plone Core Inverted	custom, cus	tom/PyReg, ar	chexam	ple,arc
	laustant au	ton DrDon or	al ava	-10 000

插圖 2-27 porta;_skins Properties tab

接著我們要將 PyzopeRegForm 的 validation 和 navigation 設定好。如本節

插圖 2-26 匯入的 PyReg 目錄

先前所提到的方式,設定 portal_properties/form_properties。

1ionn	[validate_sendto	string
🗂 synPropertiesForm	validate_synPropertiesForm	string
🗖 PyzopeRegForm	validate_PyzopeRegForm	string
	Save Changes Delete	

插圖 2-28 新增 form_property

default.PyzopeRegForm.success	script: PyzopeReg		
🗖 default.PyzopeRegForm.failure	PyzopeRegForm		
🗖 default.PyzopeReg.success	PyzopeRegForm		
🗖 default.PyzopeReg.failure	PyzopeRegForm	PyzopeRegForm	
	Save Changes	Delete	

插圖 2-29 新增 navigation_property

設定完成後就可以測試報名表單了:

) Portal	— 希名贾料 ————————————————————————————————————
	姓名 🛛
λ	黄敏松
用者名葉	服務單位 / 學校 ■
1	ossacc
> 登入	戰稱 / 身份 •
	boy
忘記您的密碼?	
and the second second	₩終電話 -
\$P\$19月-首?	02-8257-0395
8 20	E-mail .
入您的 E-Mail Address	👍 song@song.idv.tw
奇雄恐(首称)弦] 目級 近修改 004-01-16	上課日期 第一梯次報名已載止。 ④ 第二梯次 8/16 (六), 8/17 (日)
	●●● 「7回君・ 1.諸問導一個 Tag 是用来畫表格?

插圖 2-30 輸入報名資料

導覽	
🗿 Portal	Registered have been saved.
登入	PyZope 課程報名
使用者名稱	您好,在報名截止之後,我們會依照報名先後順序及測驗分數來決定學員的名單,並在 http://zelp.zope.org.tw/Training/roster公佈名單。
老嘴	第一梯次於 07/10 公告學員名單,
	第二梯次於 08/07 公告學員名單。
◆登入	謝謝您的報名。
。忘記您的落碼?	Cheers,
- nar iste des skum	PyZope Training Team

插圖 2-31 報名完成

這個範例中,很多流程都會回到 PyzopeRegForm 這個 ZPT 來,所以這個 ZPT 作了許多的 tal:condition 的檢查,可以參考一下他的寫法。其次, Script: PyzopeReg 將資料寫入資料庫後,會寄出一封確認信給報名者。這點 要確認 plone 目錄下的 MailHost 物件所設定的 mail server 可以使用,才可 以正確的寄出確認信。

Z SQL Method 的物件我們還沒有加以說明。他主要就是以 SQL 語法來控制資料庫,他是透過 Database Connection 物件對資料庫作存取,所以當資料庫變更時只要更新 Database Connection 的物件,不需要修改 Z SQL Method 物件,甚至是將 MySQL 換成 PostgreSQL 也不用。

Z SQL Method 物件可以使用 DTML 的語法來支援變數式的 SQL 語法, DTML 有一組針對 SQL 的 tags 可以使用更完整而安全的格式作出更複雜的 SQL 運算。請見 <u>Zope Book Appendix A: DTML Reference</u>: http://zope.org/Documentation/Books/ZopeBook/2_6Edition/AppendixA.stx

3. Programming

在 Zope 中可以使用 Script(Python) 物件來撰寫 python code,用在邏輯處理的工作上,這是和用來作呈現工作的 ZPT 作為比對。將邏輯和呈現的部份分開來,在各自的處理上會有很大的幫助。

為了安全的因素, Script(Python) 在使用上有很多限制。這時可以使用 External Method, 他必須撰寫在安裝 Zope 的檔案系統中,所以必定是擁有 主機上的帳號和權限的人才能將檔案儲存在主機裡, Zope 用這個方式來過 濾安全的疑慮。

3.1. 使用 Script (Python)

我們新增一個 Script(Python) 物件叫作 *hello*,在 *parameter list* 中輸入: name="World",在主欄位中輸入:

return "Hello %s." % name

這個作法如同 Python 定義一個 function :

def hello(name="World"): return "Hello %s." % name

itle		
arameter List	name="World"	
ound Names ast Modified	context, container, script, traverse_subpath 2004-01-25 11:12	
return "Hello	%s." % name	

插圖 3-1 新增 hello Script

存檔之後就可以到 Test tab 來測試執行 hello Script。因為我們有設定一個參數 name,所以可以看到 Test tab 中有一個 name 的 input 欄位等待輸入。如果不輸入的話就會使用我們設定的預設值 "World"。

Edit	indings T Test T Proxy T Hi	story 🕇 Undo 🕇 Ownership 🕇 Securi
抱 Script (Pytho	n) at / <u>hello</u>	He
Enter variable val to view the result	es to pass to the script in of the call.	the form below and click run scr
Parameter	Value	
name		
Run Script		

插圖 3-2 測試 hello Script

保留 name value 欄位空白,按下 <u>Run Script</u> 鍵, Zope 應該會回應你 "Hello World."。可以回頭在 name value 欄位中輸入你的名字試試。

除了在 Test tab 執行之外, Script 還可以直接由 Web 介面呼叫執行,就 是說由瀏覽器直接以 URL 指定來執行。例如:

http://127.0.0.1:8080/hello

這樣就可以看到 Zope 回應 "Hello World.",也可以直接傳參數進去:

http://127.0.0.1:8080/hello?name=Script

這樣就會看到 Zope 回應 "Hello Script."。也可以由其他的物件來呼叫 Script,例如由 DTML 來呼叫時,可以用:

<dtml-var hello> #要傳參數進去的話就用:

<dtml-var expr="hello(name='Script')">

如果要由其他的 Script(Python) 來呼叫的話,可以用:

context.hello() # 或是

context.hello(name='Script')

如果要由 ZPT 中使用 Script 的話,可以用:

<div tal:replace="here/hello" /> # 或是

<div tal:replace="python: here.hello(name='Script')" />

呼叫、執行 Script 時要記得 <u>URL Traversal</u> 和 <u>Acquisition</u> 的特性。同時 利用這個特性我們可以將 Script 執行在某個物件上。我們先建一個 Script 名稱為 get_name, *Parameter list* 保持空白,內容塡入:

return context.title_and_id()

然後用瀏覽器瀏覽這些 URL:

http://127.0.0.1:8080/Control_Panel/get_name # Control Panel (Control Panel)

http://127.0.0.1:8080/temp_folder/get_name
Temporary Folder (temp_folder)

http://127.0.0.1:8080/acl_users/get_name # User Folder (acl_users)

http://127.0.0.1:8080/standard_html_header/get_name
Standard Html Header (standard_html_header)

你可以看到 get_name 執行在不同的物件上就會取出不同的 Title 和 Id 來, 這也稱作由這些物件來呼叫 Script。在這裡我們用了 context 這個預設的變 數, context 表示呼叫他的物件本身。 Script 有一組特殊的變數,當 Script 被呼叫時就會建立:

Context

這變數指向呼叫 Script 的物件

Container

這變數指向容納 Script 的目錄物件

Script

這變數指向 Script 物件本身

Namespace

只有由 DTML 呼叫時才會定義這個變數,他會指向 DTML 呼叫時的 namespace

Subpath

這是個進階的變數,只有當透過 Web 呼叫 Script ,而且 URL 在 Script 之後還有 其他的物件路徑時才會將這些路徑切割放入一個 list 中,由左向右依序放置。其他 時候這只是一個空的 list 。

當我們需要取得使用者輸入的表單資料時,我們可以在 context 裡找到一個 REQUEST 物件,他記錄了 Zope 的 web request。例如我們有一個 Script 叫作 feed,我們用這樣的 URL 瀏覽:

Zoo/LargeAnimals/hippo/feed?food_type=spam

這時在 feed 中我們可以用 context.REQUEST.food_type 來取得 food_type 的變數值。由表單傳來的變數可以用一樣的方法取得。另外一個取得 REQUEST 的方法是,在 Script 的 parameter list 中輸入 REQUEST,這樣 作的話 Zope 會自動將 HTTP request 傳給 REQUEST 這個變數,如此我們 就可以用 REQUEST.food_type 來取得 food_type 的變數值。

在 Script 中很多 Python 的 build-in functions 是不能使用的,像 open 就不能使用,這是防止直接由 Web 存取主機的檔案系統。

這些 build-in functions 是可以如同在 Python 環境中一般使用的: None, abs, apply, callable, chr, cmp, complex, delattr, divmod, filter, float, getattr, hash, hex, int, isinstance, issubclass, list, len, long, map, max, min, oct, ord, repr, round, setattr, str, tuple.

*range*和 *pow*用法是一樣的,但大小受到限制。這是預防過大的數字或序列會造成 Denial of Service 的攻擊。一些 DTML 的 functions 也是可用的, 如 DateTime 和 test。

要比較二個物件是否為同一型態, type 被 same_type 取代。用 type 時:

if type(foo) == type([]):
 return "foo is a list"

使用 same_type 比較之後如果是同型態的物件,則傳回 true :

if same_type(foo, []):

return "foo is a list"

Script 的限制是為了預防傷害,包括禁止存取 Zope 內部的物件、對 Zope 物件的不當修改、傷及 Zope 本身的執行程序、和存取 Zope 所在的主機。 這些限制是用 Script 只能執行受限的事情來達成。

迴圈限制

Script 不能執行無限迴圈,如果 Script 執行一個很大數字的迴圈, Zope 會引發一個 錯誤。不論是 for 或 while 迴圈都被限制。這是避免因為一個無限迴圈造成 Zope 本 身停止回應。

Import 限制

Script 不能隨心所欲的 import Packages 和 Modules ,只能夠 import *Products.PythonScripts.standard* utility module , *AccessControl* module , *string* , *random* , *math* , *sequence* 。還有一些經由 Product 作者特別指定的 Modules , 可以在 Zope Book Appendix B, <u>API Reference</u> 中找到更多的訊息。(或是看 Products 目錄中的 PythonScripts 裡的 README.txt , 會提到如何讓 Script 可以 Import 更多 的 Modules。)

使用限制

當 Script 使用物件時也會受 Zope 安全原則的限制。換句話說,當使用者呼叫 Script 存取某一物件時, Zope 會檢查使用者是否有權限存取這個物件。所有可被執 行的物件都可以在 Proxy tab 中設定 Proxy Roles,讓執行時的角色參照這個角色。 再則,不能存取 Id 開頭為底線的物件, Zope 認定這是內部的物件。最後,雖然可 以在 Script 中定義 class,但是不能正常使用他。因為 __init__()不能被執行。

寫入限制

不能直接使用 Script 改變物件的屬性,只能呼叫適當的 Zope API 方法(method)來作。

我們說了這麼多 Script 使用上的限制,這些都是為了安全的考量。如果想要在 Zope 中使用 Python 而且免去這些種種的限制的話,可以使用 External Method。

3.2. 使用 External Method

當你需要讀取主機檔案系統上的資料,或是想使用一些進階的模組如 regular expressions 或是圖形處理等,就可以用 External Method。

撰寫、編輯 External Method 必需要在 Zope 安裝目錄的 Extensions 目錄中,或是在已安裝的 Products 目錄的 Extensions 目錄中,或是在 Zope 的 INSTANCE_HOME 的目錄中。

讓我們來新增一個檔案叫作 *Example.py* 在 Zope 安裝目錄的 Extensions 目錄中,內容寫入:

def hello(name="World"): return "Hello %s." % name

然後在 ZMI 的根目錄底下新增一個 External Method 物件叫作 *exthello*, Module Name 填入 Example, Function Name 填入 hello。新增完成後會在 ZMI 上看到多了一個物件,點選他會進入 Properties tab,如下圖:

4	Properties	Test	Undo
💖 External Me	thod at / <u>exthello</u>		
Id	exthello		
Title			
Module Name	Example		
Function Name	hello		
	Save Changes		

插圖 3-3 External Method Properties tab

你可以按下 Test tab 來執行這支 External Method ,也可如同呼叫 Script (Python) 物件一般經由 Web, ZPT, Script, DTML 等來呼叫他。

使用 External Method 的主要原因就是要使用 Script (Python) 受到限制的 功能,像是存取主機的檔案系統或網路,載入在 Script 不被允許的模組等。 所以我們接下來的範例就是以 Script 配合 External Method 將網頁擷取回 Plone site 建立成 Document 物件。

先從教材包的 Import 目錄中將 PageGet.zexp 複製到 \$Zope/import 目錄中; 再將 getPage.py 複製到 \$Zope/Extensions 目錄中。然後到 Plone site (ZMI) 中的 portal_skins/custom 目錄裡把 PageGet.zexp 匯入,就會得到一個 PageGet 目錄,目錄中有一個 ZPT:getHtmlPage.pt,一個 Script(Python): getHtmlPage.py 和二個 External Method:getPage,getFile。

匯入目錄到 custom 後首先要先增 skins 的路徑,如此這個目錄才能被使 用到。然後我們希望 ZPT: getHtmlPage.pt 可以出現在 Folder 物件的 Action tab 上,如此我們就可以在目錄物件中擷取遠端的網頁回來。這時候我們要 新增一個 Action 在 portal_actions:

Name	: PageGet
Id	: PageGet
Action	: string: \${folder_url}/getHtmlPage.pt
Condition	:
Permission	: Add portal content
Category	: folder
Visible?	: V

新增完成後在目錄模式中就可以看到多了一個 PageGet 的 Action tab ,按 下這個 PageGet tab 就會看到 ZPT: getHtmlPage.pt 的網頁畫面:

song s	ноте			
Home page	e area tha	t contains	the items cre	ated and col
這目錄沒有預	設網頁,因7	「會以項目列	表顯示,如果您要	要在這目錄建立了
建立預設文	件			
2. 建立預設文 到上二局	锌			
3.建立預設文 到上一層	神			

插圖 3-4 PageGet action tab

取得遠端網	र्भ अम्ब स्व	
	石斛貝	
請輸入您要取得網	頁的 URL∘	
諸輸入URL:	ttp://www.zope.org.tw/inde	x_html
	Get	

插圖 3-5 getHtmlPage.pt 的畫面

我們來說明一下 ZPT: getHtmlPage.pt 的內容:

...[略] <div metal:fill-slot="main"> <div tal:condition="python: not (here.portal_membership.checkPermission('Add portal contents', here)) "> </div>

上面這段檢查目前的使用者是否有新增物件的權限,沒有的話就要求登入。

```
...[續]

<div class="Desktop">

<form action="getHtmlPage.py" method="POST">

<strong> 請輸入 URL : </strong>

</d>

</d>

<input name="url" type="text" size="50" value="http://">

<input name="url" type="text" size="50" value="http://">

<input name="url" type="text" size="50" value="http://">
```

上面這段建立了一個輸入網址的欄位和確認鍵的表單,然後呼叫 getHtmlPage.py 來處理,接著來看 getHtmlPage.py:

```
req = container.REQUEST
res = req.RESPONSE
cdurl = context.absolute_url()
page = context.getPage(cdurl, req['url'])
page['type'] = 'Document'
```

變數 page 儲存了 External Method : getPage 的傳回值,他傳回一個 dict 包括了取回的網頁內容, Id,和網頁內容分析後所得到的 Title,圖檔網址 list 和檔案網址 list。另外多加了一個 type 到 page 裡。

```
if hasattr(context, page['id']):
  for i in range(1, 20):
    pageid = page['id'] + '-' + str(i)
    if not hasattr(context, pageid):
        page['id'] = pageid
        break
```

上面的判斷式是用來確保待會要新增的物件 Id 不要重複。

context.invokeFactory(type_name=page['type'],id=page['id'])
pagei = getattr(context, page['id'])
pagei.setMetadata({'Title':page['title']})
pagei.manage_editDocument(text=page['body'], text_format='html')

第一行的 invokeFactory 用來新增物件,接下來的幾行將內容填入,到這 裡已經將網頁擷取回來並建立 Document 物件完成了。

filedirname = 'FilesDir'
if not hasattr(context, filedirname):
 context.invokeFactory(type_name='Folder', id=filedirname)
filedir = getattr(context, filedirname)

FilesDir 是要存放網頁裡的圖檔或檔案的目錄,先檢查看看目錄是否存在,如果沒有就建立他,然後取得目錄物件為 filedir 變數。

newdirname = page['id']
filedir.invokeFactory(type_name='Folder', id=newdirname)
filefd = getattr(filedir, newdirname)

在 FilesDir 目錄中建立和網頁同名的目錄,分別存放網頁裡的圖和檔案。

```
imgdb = page['images']
if imgdb:
imgurls = imgdb.keys()
for imgurl in imgurls:
    image = context.getFile(imgurl)
    if image['id'] in filefd.objectIds():
        image['id']=image['id']+'-1'
    filefd.invokeFactory(type_name='Image', id=image['id'])
    imgobj = getattr(filefd, image['id'])
    imgobj.edit(file=image['file'])
```

上面開始處理圖檔的部份,將 getPage 傳回的圖檔清單一一由 External Method:getFile 取回,然後新增 Image 物件儲存他。

```
filedb = page['files']
if filedb:
    fileurls = filedb.keys()
    for fileurl in fileurls:
        file = context.getFile(fileurl)
        if file['id'] in filefd.objectIds():
            file['id']=file['id']+'-1'
        filefd.invokeFactory(type_name='File', id=file['id'])
        fileobj = getattr(filefd, file['id'])
        fileobj.edit(file=file['file'])
```

然後是處理檔案物件的部份。

```
return res.redirect(pagei.absolute_url()+'/metadata_edit_form')
```

最後完成時重導到新增網頁物件的屬性編輯的頁面。

以上大部份的工作在 External Method 處理,包括擷取網頁,分析網頁, 擷取圖和檔案等。在 \$Zope/Extensions/getPage.py 檔案裡有二個 functions 和一個 class。Function:getPage 用 urlopen 將網頁抓回來之後,利用 class: MyHTMLParser 分析網頁的內容,然後將連結等修改成新的網址,再檢查網 頁的編碼,如果需要的話就作轉碼的動作。第二個 Function:getFile 就只是 單純的將檔案抓回來而且。以下是 getPage.py 的內容:

from urllib import urlopen from string import find, rfind, strip, replace from sgmllib import SGMLParser from urlparse import urlparse, urljoin import re

```
def getPage(self, curl, url, **args):
    parser = MyHTMLParser()
    rfile = urlopen(url)
    self.alldata = ""
    self.rhost = ""
```

```
newdir = ""
  images = \{\}
  files = \{\}
  try:
     self.alldata = rfile.read()
     parser.feed(self.alldata)
     parser.close()
  except:
     return rfile, self.alldata, parser
  rfile.close()
  id = urlparse(url)[2]
  id = id[rfind(id, '/')+1:]
  title = parser.title
  imagedb = parser.imageurls
  hrefdb = parser.hrefurls
  filedb = parser.fileurls
  self.rhost = urlparse(url)[1]
  nwdir = 'FilesDir/' + id + '/'
  marktxt = '<!-- pyGet -->'
  if find(self.alldata, marktxt) > -1:
     self.alldata = self.alldata[find(self.alldata, marktxt)+len(marktxt):
rfind(self.alldata, marktxt)]
  if curl[-1] != '/':
     curl = curl + '/'
  def chdata(self, preurl, owurl):
     owp = urlparse(owurl)
     if owp[1] == self.rhost or owp[1] == ":
       owid = owp[2][rfind(owp[2], '/')+1:]
       newurl = preurl + owid
        self.alldata = re.compile("[\=\"]("+owurl+")[\s\"]").sub(newurl,
self.alldata)
  if imagedb:
     for image in imagedb.keys():
        imageurl = urljoin(url, image)
       images[imageurl] = imagedb[image]
        chdata(self, nwdir, image)
  if hrefdb:
```

```
for href in hrefdb.keys():
        hrefurl = urljoin(url, href)
        chdata(self, curl, href)
  if filedb:
     for file in filedb.keys():
        fileurl = urljoin(url, file)
        files[fileurl] = filedb[file]
        chdata(self, nwdir, file)
  if parser.encoding and parser.encoding.lower() != 'utf-8':
     self.alldata = unicode(self.alldata, parser.encoding.lower()).encode
('utf-8')
     title = unicode(title, parser.encoding.lower()).encode('utf-8')
     self.alldata = replace(self.alldata, 'charset='+parser.encoding,
'charset=utf-8')
  return {'id': id, 'title': title, 'body': self.alldata, 'images': images, 'files':
files}
def getFile(fileurl):
  rfile = urlopen(fileurl)
  file = rfile.read()
  rfile.close()
  fileid = urlparse(fileurl)[2]
  fileid = fileid[rfind(fileid, '/')+1:]
  return {'id':fileid, 'file':file}
class MyHTMLParser(SGMLParser):
  def init (self):
     SGMLParser. init (self)
     self.title = ""
     self.data = ""
     self.imageurls = {}
     self.hrefurls = {}
     self.fileurls = {}
     self.encoding = ""
  def handle data(self, data):
```

```
if data:
     self.data = self.data + data
def start title(self, attributes):
  self data = ""
def end title(self):
  self.title = self.data
def do meta(self, attributes):
  for key, value in attributes:
     if key.lower() == "content":
        try:
          self.encoding = value.split(';')[1].split('=')[1].strip()
        except:
          pass
def do body(self, attributes):
  self.img attr(attributes, 'background')
def do img(self, attributes):
  self.img attr(attributes, 'src', 'lowsrc')
def do table(self, attributes):
  self.img attr(attributes, 'background')
def do td(self, attributes):
  self.img attr(attributes, 'background')
def do th(self, attributes):
  self.img attr(attributes, 'background')
def do tr(self, attributes):
  self.img attr(attributes, 'background')
def do a(self, attributes):
  self.href attr(attributes, 'href')
def do link(self, attributes):
  self.href attr(attributes, 'href')
  self.file attr(attributes, 'rel')
```

```
def img_attr(self, attributes, *args):
     alt = "
     for name, value in attributes:
       if name == 'alt': alt = value
       if name in args:
          if value: value = strip(value)
          if value:
             self.imageurls[value] = {}
             self.imageurls[value]['statu'] = 0
             self.imageurls[value]['alt'] = alt
  def href attr(self, attributes, *args):
     for name, value in attributes:
        if name in args:
          if value: value = strip(value)
          if value:
             self.hrefurls[value] = {}
             self.hrefurls[value]['statu'] = 0
  def file attr(self, attributes, *args):
     for name, value in attributes:
       if name in args:
          if value: value = strip(value)
          if value:
             self.fileurls[value] = {}
             self.fileurls[value]['statu'] = 0
if name == ' main ':
  getPage(url="http://www.zope.org.tw/Members/song/index html")
```