國立台灣師範大學資訊工程研究所

碩士論文

指導教授：葉耀明博士

行動個人化電子書系統設計

Personalized e-book system design

研究生：簡孝倫撰

中華民國 101 年　6 月

國立臺灣師範大學資訊工程研究所
博碩士學位論文

行動個人化電子書系統設計

Personalized e-book system design

經考試合格特此證明

審查教授：葉慶隆

趙明正

指導教授：葉耀明

系主任：李忠謀

中華民國 101 年 6 月

摘要

Web 2.0 是現在這個數位時代一個相當重要的觀念，各式各樣子的資源網站不斷的產生，例如 CNN、蘋果日報、mobile01、Engadget 等等都是資源相當豐富的網站，以往我們都是使用 PC 電腦來做網站的瀏覽，而最近蘋果公司及 Google 將行動裝置帶到一個高峰，讓大家得以在任何時候都可以去吸取這個世界龐大的數位資源。然而要如何讓這些網站變得在行動裝置得以方便的瀏覽就顯得相當重要。

數位資訊爆炸的時代，本研究的重點就是要如何將自己所喜愛的數位資源彙整起來，變成電子書的方式來做瀏覽。Personalized 就是我們所要的觀念，而此研究須要用到 HTML5、PHP、RSS 等技術來結合出一套電子書網頁瀏覽系統，我們不需要去將各個網站一個一個做點選開啟，而是利用簡單的動作就能將所需要的數位資源蒐集起來，並且動態的自動將資源傳送到此系統中，因此我們就更能在任何時間及地點來使用我們所自己制定的閱讀體驗，這樣子的系統能運用在各種情境之中，不管是數位教學，還是閱報系統，電子刊物，甚至社群數位，都可以很方便的在任何時刻地點做閱讀的體驗。

關鍵字: Web 2.0、HTML 5、PHP、RSS、Android、iPad、iphone

ABSTRACT

Personalized e-book system design

Web 2.0 is a very important concept at this digital age, the wide range of digital resources are generated, such as CNN, Apple Daily, mobile01, Engadget, and so on, in the past we are using a PC computer do browse this site, but with the rise of handheld devices, we need to watch these huge digital resources at any location and any moment , however, how to make these sites become to be browsing on mobile devices becomes very important.

Of course, the site are hundred of kinds , how to own the favorite viewing of digital resources exchange it together, and turned into e-book way to browse is the focus of this study, the 「Personalized」 is the concept of what we want, but with such an idea we must use HTML5, PHP, RSS to combine a personal system, we do not need to click and open each site, but an action will be able to digital resources needed to catch, and dynamic and automatic resource sent to this system, this way we are better at any time and place to use the personalized reading experience , in this system , we developed this way can be used in a lot of situations ,whether it is in e-learning, or the newspaper system , electronic publications , and even the digital resource of the social network , it can be a very convenient experience at any moment

keywords: Web 2.0、HTML 5、PHP、RSS、Android、iPad、iphone

致謝

兩年的時光過得非常快，但也讓我在 XML 實驗室學到很多，謝謝指導教授 葉曜明老師從一開始的 meeting 及課程，讓我學到不少。這兩年的期間我們也參加了許許多多的計畫，不管事諮會的或是國科會的計畫，都讓我獲益良多，不只讓我學到專業上面的知識，也讓我學習到人與人之間的相處。感謝陽明正教授以及葉慶隆教授擔任口試委員，給予我許多寶貴的意見以及指導。

另外也感謝實驗室學長的教導，以及同學的鼓勵，感謝阿迪、阿倫、芝華、小胖、小白、明憲在平時給予我們大力的相助，讓我在遇到問題時不會太過於慌張，也讓我學習到許多東西，最後也要感謝父母在平時給予我的陪伴，以及女友適時的陪伴與鼓勵，如果沒有這些人，就不會有這篇論文的產生，讓我不會在忙碌的同時迷失方向，在此獻上最誠摯的祝福與感謝，謝謝你們。

簡孝倫 謹致

國立臺灣師範大學

資訊工程研究所

民國 101 年 6 月

Chapter 1 緒論

1.1 研究背景及動機

在這個網路互動盛行的時代，越來越多的數位資料爆炸產生，這也是我們所謂 WEB 2.0 所帶來的極大影響，很早很早的時候，blog 的誕生，加速了 WEB 2.0 的發展，接著出現了像 youtube 這樣子的影音網站，，隨著數位資源越來越多，網路上也慢慢的出現互動性高的數位網站，慢慢結合出影音 blog 等等的變化，各種新聞網站、數位刊物、各大論壇，都出現了互動性的服務，接著 WEB 2.0 的發展出現了爆炸性的變化，社群網站 (Social network)的誕生，將這些龐大的資源做了交互性的動作，人們開始更能分享所看到的資源，也可以評論他，因此我們開始被動的接收這些資源。

接著，在行動設備及手持系統的盛行下，例如平板電腦及智慧型手機，加上我們擁有龐大的數位資源，我們便可以到處接收資訊，那要如何讓使用者方便的接收資訊，就是本研究的重點。

1.2 研究目的

以往，我們使用電腦瀏覽資料的習慣，都是一個一個去做點選，進入個個網站，然後再這些網站之間做切換，但其實我們不但可以被動接收這些資訊，甚至可以整理成起來讓使用者做觀賞瀏覽。

電子書的出現，正好可以讓這樣子的概念做更棒的實現，ipad 及平板電腦的出現，人們手上拿著的漸漸不在是書本，書本變成電腦的好處不僅僅是不需要帶著厚重的書本，最重要的是其中的數位內容，我們不一定要將寫好的東西放在檔案上，我們可以將龐大的數位資源整理起來然後動態的放在這個所謂個人化的電子書本上，這樣使用只每天起床，看到的書本都是最新，而且是不同的資訊，更重要的是，使用者是被動的接收他，也不用去做個個網站點選，使用者只需要像閱讀書本一樣很輕鬆的接收數位資料。

這樣子的觀念其實很早就有出現,那就是RSS（Really Simple Syndication），但是之前因為社群網站以及電子書不是非常盛行，所以他漸漸的被忽略，所以這樣子的技術，在現在這個數位資料爆炸的時代，是非常有用的，因為他可以幫助我們整合所要觀看的資料，那如果我們運用這樣子的技術，加上電子書的觀賞模式，那會是一個效果非常好的觀賞體驗，而本研究的目的，正勢將龐大資源做整合，運用電子書的方式將他做呈現，甚至可以合併現在所有的社群網站，因為現在我們的使用者，擁有的不一定是一個社群網站帳號，而是多數的社群帳號，若我們將之做合併，這樣使用者就可以向在閱讀書本的同時觀賞到不只是你所訂閱的網站，而是你朋友分享給你的資源，這樣子我們形成的個人化書籍，裡面的數位內容不單單是你想看的,還有朋友推薦給你的,甚至是現在最火紅的話題新聞，不只整合資料，更是被動接收資料，更是提供活的內容給你，正是本研究的重點。

Chapter 2 文獻探討

在本系統中，有許多背景知識是相當重要的，HTML5、PHP、RSS，這些知識尤為重要，本節也會針對這些知識以及文獻多做介紹。

2.1 HTML5

HTML5 早在 2004 年就被提出，而在 2007 年被 W3C 接納，這是一個網路標準，由 HTML 延伸而來，一個網站的主要架構都是由 HTML 所撰寫，而後端資料庫的部份則交給 PHP 等等的程式語言，若整個網站是一棟大樓，那 HTML 就像這棟大樓的鋼筋架構一樣，以往 HTML 對網站所做的設計，就是相當一般的排版，例如左右切割，簡安的資料呈現，但是現在 HTML5 就可以將現今許多的動畫、特效、3D 運算等等較為複雜的外觀呈現，一一做實現，而這樣不僅僅可以讓程式設計師在設計網站時較為方便，呈現出來的網站也較為絢麗，也更可以使網站較為多元，而 HTML5 就很符合本研究所需要的元素，因為我們所要做的就是將數位資源做集合，在做完整的呈現，而「呈現」這個工作，交給 HTML5 來實作是在適合不過了。

HTML5 與之前的 HTML 有著什麼樣的不同呢？真正具體來說，HTML5 添加了許多新的語法標籤，其中包括<video>, <audio>, 和<canvas>標籤，同時也整合了許多 SVG 內容。這些標籤可以讓我們在設計網站外觀的時候更為方便，例如我們要在網站中放置一個高解析影片，我們要做的只是需要放上<video>標籤就可以了，還有許多新的標籤包括<section>, <article>, <header>, 和 <nav>, 這些標籤更可以將文字檔案的內容做更有效的排版。同時也有一些屬性和元素被移除掉了。一些元素，像<a>, <cite>和<menu>被修改，這些標籤則被重新定義及修改。同時

APIs 和 DOM 已經成為 HTML5 中的基礎部分。HTML5 還可以使所有瀏覽器和用戶端程式能夠一致地處理語法錯誤。

尤其在 WEB 2.0 時代中，所謂的 WEB Application 就顯得相當重要了，而什麼是 Web Application 呢？就像是我們平常使用的應用程式，只是將他搬移到瀏覽器上而已， 我們以往看影片，玩遊戲都是透過電腦上的應用程式來做使用，但現在網路盛行的時代，人們不知不覺的，漸漸將這些使用習慣搬移到網站上，我們可以在 YOUTUBE 上看影片，我們可以在網站上玩遊戲，我們更可以開始將網站更為複雜化，讓他看起來更像一個應用程式，而要如何實作這樣子的應用程式，不是用 C 語言也不是用 JAVA 語言，而是使用 HTML5 就可以達到許許多多的應用，例如小畫家這樣子的應用程式，我們使用 HTML5 就可以將他實作出來，而本研究所要設計的系統，基本上也可以稱作一個 WEB Application，而此系統這個應用程式所要做的事情，最主要就是蒐集所有的數位資源，將他們做呈現，要如何更漂亮的呈現，HTML5 就扮演著相當重要的角色。

2.2 RSS (Really Simple Syndication) & ATOM(Atom Syndication Format)

RSS 顧名思義就是將資料做「聚合」，RSS 最早是由一家美國網景公司要在 My.Netscape.Com 上做使用而設計的，接著版本不斷的演化才將之命名為 Rich Site Summary 稱為 RSS，這是較早的版本，接著越來越多的站點對此做支持，並且已經成為一個成功 XML 應用，現在 RSS 擁有一個傳播迅速的技術平台，使得每一個人都可以提供 RSS 訊息，但是此項技術在 HTML4 的時代，幾乎可以說是快要絕種，因為數位資源不夠多，在加上社群網站以及平板電腦的興起，使得這項技

術再次復活，並且可應用的範圍極廣。

而另外一種幾乎相同於 RSS 的格式,它叫做 ATOM（Atom Syndication Format），他也是基於 XML 的一種的文檔格式，他吸取了眾多的 RSS 概念，成為一種替代 RSS 的新形態格式，而 Google 的各項服務正在使用 ATOM 格式，而為何會出現 Atom 這樣子的格式想用來替代 RSS 呢? 首先 Atom 的概念可以說是完全相同於 RSS，但是在結構上有著些許的不同，他相較於 RSS 的標籤形式來的更為嚴謹，

例如: XML 的 description 會包含整個內文和摘要，而 Atom 則用 summary 和 content，來區分出內文和摘要，在呈現的時候會更為整齊及美觀，這對我們的系統的呈現會更有幫助，而 RSS 存在許許多多的標準形式，Atom 則是採用統一的標準，簡單來說，Atom 是統整之前 RSS 的所有概念及內容，統一重新製作的一種新的標準，相較於 RSS，Atom 已經是標準化過後的格式，頻寬消耗更少，複雜性更低，更新來源時會更為快速，未來更多的網站及公司會使用這樣的格式來做為網站的推送來源。

此項技術也是本研究最重要的重點，這項技術有一個很重要的概念,「被動」，以往我們很直覺的認為，使用電腦、智慧型系統，我們是主動的去找尋我們所要的資料，但是 RSS 使得使用者得以被動接收資料，就猶如我們躺在家裡，送報生不斷的送資料最新的報紙一樣，而我不用去外面的超商買自己想要看的報紙，這是一個很重要的概念，因為現在數位資料龐大，我們無法一一去找尋他，我們要讓電腦自動送過來，這也是 RSS 最主要的宗旨，而這樣子的概念對本研究有相當大的幫助，RSS 的主要架構就是傳送 XML 格式的檔案，供使用者觀看，他是一份將網站整理好的 XML 檔案，而我們只要對該網站做訂閱的動作，資料就會很快速的送到你的身邊，在加上我們現在平板電腦的盛行，若對此系統多做包裝，將會是

一個非常棒的應用，而且會實現別於一般 RSS 的閱讀體驗。

市面上有著許許多多的 RSS 閱讀器，一種就是存在客戶端的，但是這種存在客戶端的閱讀器，一旦更換裝置或是更新軟體，就不能觀看自己所要的 RSS。另一種就是所謂的 Web 服務，最為知名的就是 Google Reader，而閱讀器正是用來閱讀 RSS 的一項工具，而現今市面上的閱讀器，幾乎都只是單純地看到摘要等等，無法一次全部觀看，而且都非常不美觀，這無法給我們擁有所謂類似書籍的閱讀體驗，這還是在電腦上使用一個程式的感覺，並不是相當的舒適，也並無針對不同裝置上的解析度去做設計，或是由於螢幕尺寸的不同而去做調整，呈現的大多也只是摘要，若要收看全文，則會跳至另外一個網頁，這樣必需又在連結之間作切換，也無消去廣告的功能，或是針對一些閱讀障礙上的設計，一般的 RSS Reader 是一種方便於電腦使用者將網站做聚合的一個動作，但是並沒有讓使用者擁有所謂的閱讀體驗，因此本研究要創造的是一個讓使用者可以用閱讀的方式，來瀏覽網站上的所有的東西，希望任何網站都能擁有閱讀體驗的系統。

2.3 PHP (Personal Home Page)

PHP 是一種腳本語言，他專門用來處理動態網頁，他在 1995 年被拉斯姆斯·勒多夫發明，此程式語言應用非常廣泛，大部分被使用在網頁伺服器上，此語言也是完全免費，而當中也包含了許多 GUI，他可以在許許多多的伺服器及作業系統極平台上執行，現在常常被用來處理資料庫。

如果 HTML5 是網頁的前端架構，那麼 PHP 就是一個網頁的後端處理，後端處理在本系統上非常非常的重要，如何完整個將 RSS 抓取下來，並解做解析，過濾的動作，就顯得非常的重要，而這些解析 XML、過濾數位資訊的動作，就變得非常

需要 PHP 來實作，並且可以做很完整的處理。

2.4 Web Application

Web application 可以算是一種新興的名詞，由於網路行為越來越發達，我們使用電腦的習慣也慢慢開始改變，看電影、玩遊戲、看新聞、聽音樂、防毒，許許多多的電腦應用，漸漸的轉變到網路上，我們在網站上看電影，也可以在網站上玩遊戲聽音樂，因次漸漸的，我們可能不再需要所謂的應用程式，也就是一般作業系統平台下的程式，而我們現在只需要一個瀏覽器，就可以做到我們所需要電腦做到的事情，Google 所推出的 Chrome 作業系統就是一個非常好的例子，一個瀏覽器，取代所有的作業系統，一打開電腦就是一個瀏覽器，所有事情我們在上面完成，而要在瀏覽器上面的應用程式，我們就稱作 Web Application，一種新型態的應用程式，而他甚至只需要 HTML5 + CSS +javascript 就可以完成，對撰寫程式的設計師來說，設計程式起來也會較為方便。

而這樣子的應用程式就類似本研究的設計，本研究所要設計的系統，基本上也可以稱作一個 Web Application,這有著什麼樣子的好處？就如雲端的概念一樣，我們可以在任何時刻，任何地點，任何平台，任何設備上啟用，我們不需要安裝，我們只需要有一個瀏覽器，就可以使用這項系統，這樣子的概念最符合我們所要求的任和地點任何時刻都可以接收數位資訊的宗旨。

2.5 Electronic book

電子書顧名思義,就是書本以電子的形式來呈現,也很多人將他稱之為 e-book，他必須透過一些特別的硬體極軟體來做閱讀，硬體的部份想當然而就是要以輕薄

為主，而軟體的部份就要使用特殊的閱讀器來做閱讀，但其實早在很早以前，電子書就有出現了，但是卻慢慢的沒落下來，因為硬體部分不夠讓閱讀體現達到比紙本更好，所以使用的人就慢慢下降，直到最近的 ipad 以及其他平板電腦，如亞馬遜的 Kindle fire，在加上數位資源越來越多，觀看者也就越來越多，紙本出版商也漸漸投入這個市場，始這個市場像滾雪球般，越滾越大。

市場越大，投入的廠商越多，最頭痛的問題就莫過於格式了，現在市面上沒有一個非常標準的統一格式，市面上諸如，PDF、EPUB、MOBI、kindle，格式太多太多，甚至現在制定出一個比較通用的標準，EPUB 3，但是其中還是有著許多標準問題，甚至現在，市面上出了許多自己自定的格式，然後包裝成 Application，在軟體平台販售，甚至現在出現了許多使用 HTML5 制定的電子書，而這樣子的電子書也就很符合我們所要的格式，他其實不能算是一種格式，而是一種呈現方式，我們使用網頁來當做電子書來呈現，他的好處就是我們只要有瀏覽器的設備上都可以閱讀，他不只是在平板上，也可以在智慧型行動系統，個人電腦上做觀看，我們不用傭有特殊的閱讀器，也不需要特別的平板電腦，只要我們設定好解析度，我們便可以在任何平台上觀看，電子書的形式百百種，選擇對自己的系統有幫助的格式是最為重要，尤其在這個還沒有統一標準格式的時代中，再著，若使用 HTML5 來製作電子書，好處就是可以加入動態的元素，而不是寫死的動畫及圖片文字在其中，我們可以動態放入新的數位資料，就猶如本研究所要做到的概念一樣，別於其他格式，能在其中做動態的處理是在本研究的系統中是相當重要的。

2.6 Internet in a magazine

說到雜誌，本研究重點放在電子書，雜誌便是電子書一個很重要的數位刊物，

而我們喜愛閱讀雜誌，是因為其中有很多文字與圖片，並且有漂亮的排版，看起來是非常舒服，而雜誌的內容呢？雜誌可以分為許多種類，財經雜誌、八卦雜誌、數位雜誌、新聞雜誌、時尚雜誌等等，這些雜誌分門別類有著許多類型，雜誌在傳統上的編輯者，不外乎就是記者、編輯、作家等等，而現在我們有著新型態的數位雜誌,內容就是我們現在擁有的龐大數位資源, 提供內容者可以是你的朋友、你的家人、你喜愛的明星、作家等等。

由於是數位雜誌的關係，我可以動態呈現這份雜誌，我們每天看到的封面都會是不同的，可能是你朋友分享的一張圖片，可能是 CNN 頭條的照片，可能是一個論壇最熱門的話題，這些都是我們每天上網所要去找尋的資訊，但是我們可以動態的將他呈現在你的電子雜誌上面，他絕對是一份新型態的體驗，一本雜誌，多種內容，我們不用去購買其他雜誌，只要一份，每天的內容是動態的在改變，以往雜誌的作者都是編輯、記者，但是現在可以是你的朋友，若你有著一群喜愛其腳踏車的朋友，那這本雜誌勢必會成為一本腳踏車的專門雜誌，因為你的朋友喜愛腳踏車，分享的事物新聞，多半都會跟腳踏車有關，這種雜誌對本研究有著非常重大的影響，首先他達到了「聚合」的觀念，再來也有了「被動」的概念，他所要使用的也是我們需要的RSS，簡單來說這樣子的系統可以分成兩大部分，前端與後端,後端資源可以有CNN的Feed、facebook、twitter、各大網站的RSS feed。

舉個很不同的瀏覽方式，一般我們瀏覽網站或著是社群網站，大部分的事件都是由時間或著是熱門程度來排序，而現在這種數位雜誌的排序方式會有著很大的不同，他所呈現的，是「你要的東西」,「你喜愛的東西」，而且也不一定是由上而下的排序方式，而是類似雜誌的排版方式，這樣子的閱讀體驗會讓你看到一些你平常使用電腦所看不到或是不會去看的內容，朋友的相簿，可以變成一本寫真

雜誌，Yahoo 新聞可以讓他變成一本新聞雜誌，癮科技的 RSS feed 可以讓他變成一本數位科技雜誌，重點是，他還可以將這些做整合，讓你可以在各種喜好中去做翻閱，每一頁都會是不同領域的體驗。

這種雜誌也有著許多的呈現方式及製作方式，大致可以分為兩種，第一種就是用 Appliaction 的方式來做呈現,那麼就需要用到 Object-C 或著是 Android 這樣子的好處就是我們可以量身訂做，針對某些的硬體設備去做調整，例如 ipad 等等，也可以做更多的變化及美化，但是缺點就是無法在任何平板電腦或是行動設備上去做觀賞，例如 ipad 就無法在 android 的平板電腦上使用，第二種就是使用 HTML5 等網路程式語言來實作，嚴格來說，這其實太算不是一種應用程式，應該算是一個網頁但是，他又具備應用程式該有的應用，而這種類似 Web Application 的形態，好處就是擁有 Everywhere、Anytime 的特性，在加上任何擁有瀏覽器的行動設備上，我們都可以去做觀賞體驗，各有各的好處，而這種新型態的數位刊物絕對會是未來的一股新潮流，也會帶動整個電子書型態的轉變。

2.7 Ipad

一個由蘋果公司所推出的平板電腦，在 2010 年 1 月 27 號發佈，他被蘋果公司制定在介於智慧型手機及筆記型電腦間的電子產品，其實類似的產品早已經一大堆，但是此產品的推出，竟然造就成了平板電腦的旋風，使得其他廠商不斷投入這塊大餅，而人們也漸漸開始將貧版電腦帶在身邊，由於 ipad 體積非常輕薄，讓人很方便的帶在身上，除此之外，他也提供許多非常方便的應用，例如行事曆、ipod、照片、遊戲中心等等，最重要的是他提供了 app store 這樣子的軟體商店供使用者購買軟體，由於軟體量豐富，也是由全世界的程式設計師所撰寫然後再

上面販賣，所以軟體量可以是相當的大，然後其中也是內建了蘋果公司自有的瀏覽器 safari，在加上觸控的功能，讓瀏覽網頁在平板電腦上變得更為方便，除了 app store 以外，其中一個非常新型態的的平台叫做 iBook，他與 appstore 不同所提供的資源不是軟體，而是書本，千百萬種的書本在 ibooks 上面販售，提供使用者下載，而使用者也可以自己制定書本在上面販售上架，該系統所使用的書本格式則是最知名的 epub 格式，這也使得大家更容易能用平板電腦來看電子書籍。

接著蘋果在 2011 年 3 月 2 號發佈了 ipad2，他除了更快的處理器，體積也變得更為輕薄，可以說是非常非常的薄，使用者拿在手上可以說是毫無負擔，也因此造成的大賣，也加入了攝影機始的平板的功能更加多元，然後這更造成了平板電腦的風潮，人們不但可以在裡面編輯照片，可以秀照片影片給大眾看，玩遊戲，辦公等等，而其中新的作業系統版本 iOS5，除了保留原有的 ibooks 系統，更增加了所謂的 newsstand，書報攤系統，書報攤系統是蘋果最新推出的系統他不同於 ibooks 系統的，就是 ibooks 是使用者主動去找尋書籍，而 newsstand 則是被動的接收書報雜誌，這也非常類似本研究所要表達的概念，使用者訂閱好自己所喜愛的書報雜誌，這樣使用者每天起床，打開 ipad 的書報攤，接收的就會是最新的報紙，以及最新一期的雜誌週刊，但是書報攤的格式就不是非常統一，所以目前造成了撰寫者的一些困擾，導致書報攤上，幾乎都是英文的報章雜誌，鮮少出現中文的，而且炫則也較為少，但是相信在未來這樣子的系統絕對會成為平板電腦的主要潮流，ibooks 及 newsstand 使得平板電腦更像書本，也更像一本活的書本，本研究也希望能仿造出這樣子的概念來實作出動態內容的電子書刊，這也是本研究所期望的結果。

2.8 Format of the ebook

由於平板電腦以及 ipad 的興起，市面上越來越多的電子書格式不斷推出，最廣為人知的當然就是 epub3，而 epub3 現在被蘋果公司廣為應用在 ibook 上面，算是電子書比較通用的格式。電子書的形式有太多種，它可以是一個網頁擷取而成的，也有可能是紙本掃描的，甚至他是一個應用程式，但是只要我們有處在閱讀體驗上，其實都可以稱之為電子書，最知名的大廠 adobe 他則是推出了屬於自己的電子書格式 Adobe Folio，此格式吸引了多數的數位內容出版商用此格式來出版他們的電子書，而這個格是必須使用他們自家的軟體 Adobe Digital Magazine Suite 來編輯，這對出版商在編寫上多了一些許的限制，而這個格式現在有個致命的缺點，就是 publisher-branded Content Viewer，的觀看模式，他的意思就是『一本刊物一個 APP』，假設今天，天下雜誌出了 10 本刊物，那就使用者就必須擁有 10 個 app 在桌面上，台灣現今最大的男人雜誌，男人幫過去就是如此，而她現在轉到蘋果公司的 newstand，而解決這樣的問題，由此可見，太多太多的格式，造成了出版商的出版方式太過於不同，也造成了他們的問題，必須不斷的轉戰各大廠商的格式而讓自己有一個最佳的出版方式。

再來就是現在最為流行的 Application，其實他不能算是一種格式，因為他本身其實不算是電子書，而是一個程式，只是他讓使用者在使用這個程式的時候，讓使用者擁有再閱讀的體驗，所以我們其實也把他歸類在格式之中，因為現今許多的出版商使用此格式來出刊他們的刊物，這有一個好處，不用受制於格式的限制，因為他其實就只是一個程式而已，例如 SPIN 這家公司，或是 Flipboard，這些都是所謂的 APP 格式，由於是 APP 關係，所以在這其中我們可以加入許多功能

是其他電子書所沒有的，不只在排版上可以更為複雜，更可以做多項功能的呈現，例如分享、互動、遊戲、匯入匯出等等，這也是本研究上想要達成的部分。再來最一般的格式，就是台灣最多出版商所使用的格式 PDF，PDF 是一個非常單純的文件格式，而出版商可以將此格式的書本，放在網站，或是自行製作的 APP 來供使用者下載，PDF 是最為簡單也最不會有爭議的格式，但是此格式，不能讓使用者體會到所謂電子書動態排版，或是互動的情境，在閱讀此格式的書籍的時候，就僅僅單純是在閱讀一份掃描後的紙本書籍，所以並無其他更多的變化，雖然變化很少，也不能讓使用者有在閱讀上有新的體驗，但是對於那些沒有特別資訊部門的出版商而言，這個格式便是最為省錢也是最方便的方法。

最後，由於 HTML5 的出現，讓電子書的格式又增加了一項新的選擇，何謂 HTML5 格式? 簡單來說，就是讓我們一直都在瀏覽的網頁，用電子書的排版方式，讓使用者在瀏覽網頁的時候，擁有閱讀書籍的體驗及方便，HTML5 有著什麼樣的好處呢? 現在 HTML5 的出現，使得在製作網頁可以擁有更多的變化，例如影音、互動、排版等等，在實做這些功能時，HTML5 讓他變得更為簡單，再來，在上述我們提到的各種格式，都必須擁有自己的閱讀平台，就連最簡單的 PDF，那也依定要有 PDF Reader 才可閱讀，但是 HTML5 呢 ？ 不需要擁有這些特有平台，只要我們的裝置上，擁有瀏覽器，那我們就可以閱讀這項書籍，再加上，網頁可以針對個平台去做解析度的調整，所以我們不只在平板上可以閱讀書籍，電腦甚至手機上都可以閱讀，今天，假設我拿到一台不是我自己的平板電腦，那們我即使要閱讀我想閱讀的書籍，只需要我登入帳號之後，就可以進行閱讀，這有點像是所謂雲端的概念，我們的書籍其實都存在於我們的伺服器，而不是裝置之中，假設今天我們沒有網路，但是透過 HTML5 的離線閱讀，我們更可以在沒有網路的環境之下，閱讀曾經開啟

過的書籍，現今越來越多的出版商開始採用此格式來出版書籍，例如:Aside Magazine 等等。

2.9 適應性網頁設計 (Responsive Web Design)

此概念是由 Ethan Marcotte 在 2010 年提出來的，最早的概念是硬體如何因應環境改變，而做出適當的回應，例如冷氣如何因應外面的溫度做出最適當的調整，而 Ethan Marcotte 應用此概念，認為網頁可以套用此概念，第一個原因就是由於現在的設備越來越多，尤其是行動設備，而行動設備螢幕尺寸非常多種，那麼要如何讓網頁去對不同設備，去做不同解析度，或是使用者介面的調整，讓使用者可以很方便的閱讀網頁，就是此概念最主要的目的，而要達到這樣的目的，有非常多種方式，最普遍的就是利用瀏覽器來對網頁去做調整，而不需要開發者去對每一個設備去實作每一個網頁，這是相當麻煩的，或是開發者利用 CSS 去針對每一種設備去配與不同的框架，這樣網頁偵測到不同的設備就會給予不同的 CSS，這樣子使用者在觀賞網頁的時候，就會看到一個真正『量身訂做』的網頁，這個所謂的『量身訂做』，其實不單單就是將字體放大或是將框架放大或是縮小，而是將整個畫面內的整個範圍作出對這個螢幕尺寸最為方便的使用者介面，這對於本系統也是相當重要的概念，因為我們希望可以在各種設備上都能執行我們的系統。

2.10 Instapaper

Instapaper 是一個在 2007 年開發 2008 年推出的一款網路服務，它讓你用非常簡便的方式將網路想要稍後閱讀的網站儲存起來，等到你有更多時間的時候，

可以在此平台專心地做閱讀，而此平台在 2008 年的時候推出第一個在 iphone 上面的 application，2009 年的時候在 Amazon 上的 kindle 上推出，2010 年在 ipad 上推出專屬於 ipad 的 application,2011 年將社群服務整合在 instapaper 之中，讓使用者可以方想自己認為不錯的文章。

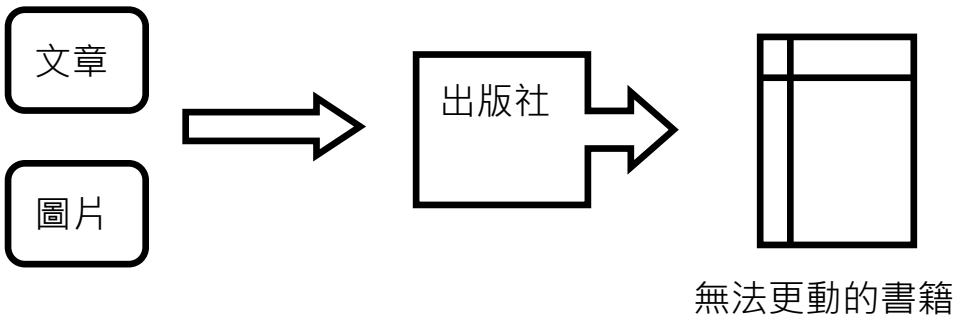
而 instapaper 的宗旨就是『Instapaper facilitates easy reading of long text content』，意思就是希望可以幫助使用者更簡單的閱讀長篇文章，由於現在的人生活忙碌，常常忽略或是無法閱讀好文章，所以 instapaper 的想法就是，希望可以在忙碌的生活中，將好文章保存起來，等到像是夜深人靜時，起床梳洗後，或是等候通勤時，可以好好將這些保存起來的文章，好好閱讀，讓使用者可以在忙碌的空閒，好好來閱讀這些好文章。

2.11 Wordpress blog

Wordpress 是一個使用 PHP 來開發的 Blog 平台，只要使用者有 PHP 及 MySQL 資料庫的平台，可以架設一個完全屬於自己的 blog 網站，由於他是一個免費的開元項目，所以使用者可以自行的在上面做編輯，自從 2003 年開發到現在，網路上排名前一百萬的網站裡面有 12%是用 Wordpress 來做編輯的,而全球就有將近 3000 萬的網站是由此架構來做建構的，他不斷推出許多的功能，而使用者也可以自行在此架構中加入許多插件，而本研究希望可以在系統中加入此元素，來讓使用者可以達到書籍分享及暫存的概念。

Chapter 3 系統概念及架構

3.1 系統概念



圖表 3-1 系統概念圖

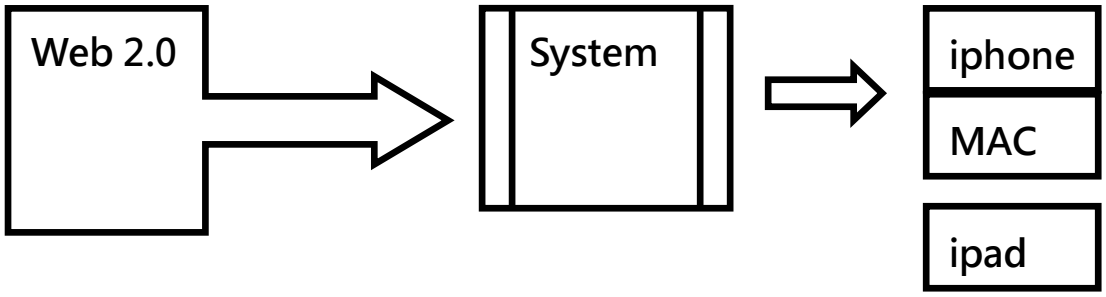
書籍，早在很久很久以前，就是人們的一種娛樂以及吸收知識的一種管道，而書本的內容可以說是相當多樣，書的內容是人制定的，這是絕對不會改變的，現在時代不斷的改變，現在絕對是一個資訊大爆炸的時代，尤其演變至今，成為一個所謂的 WEB 2.0 的時代，而這樣的時代，數位資訊越來越多，Youtube、facebook、CNN、各大論壇新聞、維基百科，這些都是 WEB 2.0 之下的產物，這是非常龐大的資源，但是我們要如何去運用這樣子的資源呢？

平板電腦的誕生，造就了電子書的興起，電子書的興起對整個紙本刊物的廠商做了一個非常大的翻動，由於人們開始喜愛閱讀電子書本，使得紙本書商不得不投入此一個市場中，這樣也使得電子書的種類越來越多，在加上現在的平板電腦越來越為輕薄，更使得讓人們越來越願意帶一個平板電腦在身邊，帶著平板電腦到處閱讀已經是非常習以為常的事情了，那麼要如何讓人們可以使用「平板電腦」去運用這一些龐大的 WEB 2.0 資源呢？這就成為了本研究的重點概念。

首先，WEB 2.0 的數位資源產物就像是一個沒有整理過的龐大書堆，我們首先要做的就勢將這龐大的書堆一一做整理，然後在做一個「過濾」的動作，過濾出使用者所想要觀看的數位資料，這樣子一本一本的書籍就會顯得整齊許多。

再來，我們所要做的就是將這些「資源」，傳送到使用者手中，但是一般這種資源，我們都可以在電腦上觀看，而觀看的習慣，就是我們「主動」去這些「書堆」中找尋自己想要的資料，但是這種主動找尋的動作會顯得相當麻煩，因為本研究的概念是希望使用者可以帶著平板電腦，在外面何時何地都可以方變得瀏覽自己想要看的資料，而這種「主動找尋」的動作，在外面就會顯得非常麻煩，因為我們不一定有時間或是空間可以讓雙手在平板電腦中轉換網頁，或是選擇自己要的網頁，所以如果今天，資料能像送報生一樣，早上自動送到自己的家，那會顯得方便很多很多，這種就是將「主動找尋」的動作轉換為「被動接收」，而這樣子被動接收資料，要如何在本研究中做到，那就是使用 RSS，因為 RSS 的概念也與此類似，只是之前這項概念沒有應用在平板電腦的電子書中，而我們希望，這樣子的概念可以在平板中得以應用。

知道如何傳送資料之後，接著我們要做的動作，就是將這些資料做呈現，當然，我們希望可以將這些資料用一種書本的方式做呈現，而這樣子就需要用到 HTML5 這樣子的程式語言，去包裝 WEB 2.0 的數位資料，讓他看起來像是書籍，看起來就只是在閱讀書籍而已，但其實我們，所在觀看的是平常我們在觀看的網站，只是他經過整理並且很有歸納的在使用者面前做呈現，而我們也希望，這些資料可以增加到社群網站的功能，因為如此以來，提供數位資源的就不再是自己，而是你和你的朋友家人等等，這樣子就會使本系統成為一個嶄新的電子書型態。

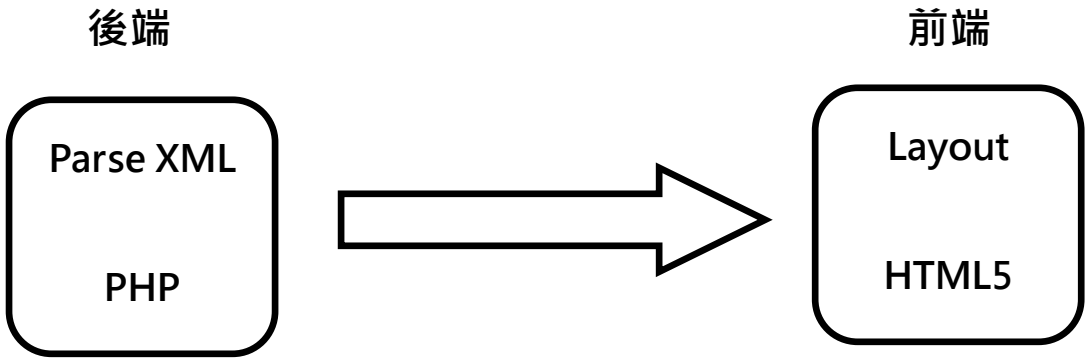


圖表 3-2 系統架構圖

3.2 系統架構

首先我們先對於整體架構大致做個劃分，本系統大致上分為前端與後端，如圖所示，後端的作用就是做 XML 的 parsing，將 Web2.0 的 RSS 資料抓取下來，這樣的好處是使用者可以被動的接收資訊，並且大部分的網站上面都可以用本系統來做瀏覽，我們也可以接收 Podcast XML，讓本系統更豐富化。前端的作用就是如何將收集到的資料做一個很好的呈現，這是相當重要的課題，而本系統是使用 HTML5 來作呈現，希望能讓使用者可以很方便的同時瀏覽多個網站。

架構上我們分為幾大部分: 系統入口、截取資源、呈現、分享。

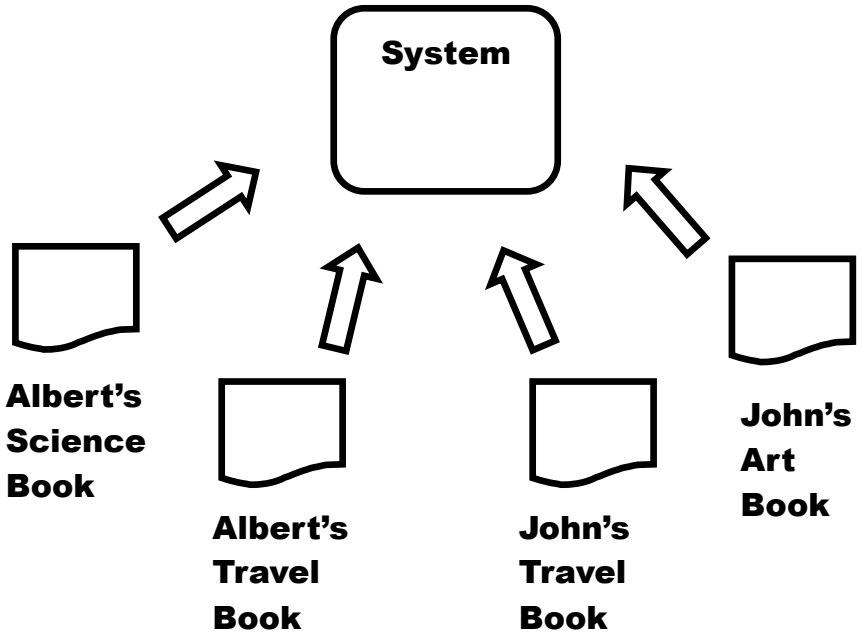


圖表 3-3 系統架構圖

3.2.1 入口

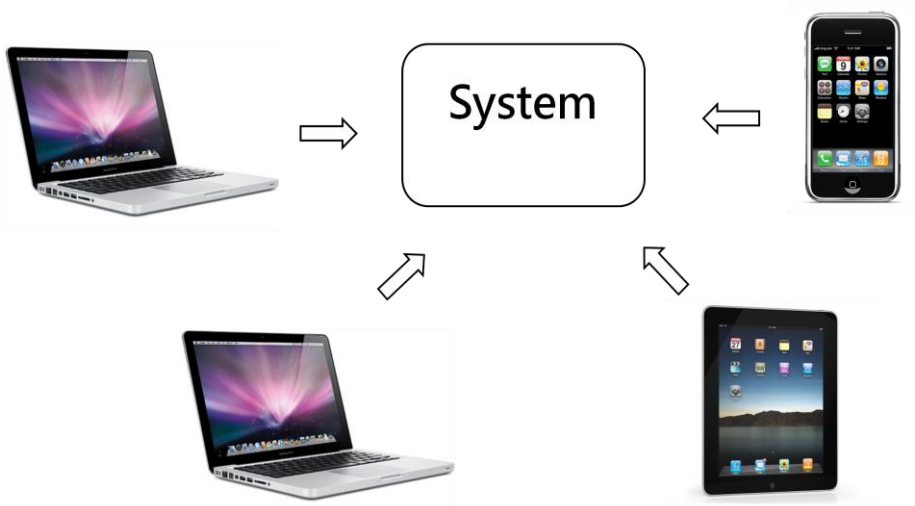
首先本系統的入口，必須要有一個帳戶控制(Account)，為什麼呢?因為本系統的訴求是個人化的電子書系統，因為講求的是個人化，個人化的意思就是其中

的內容我可以照自己的方式來做，但是，使用本系統不可能是一個人，所以我們必須要來儲存每一個人的個人化資料，如此一來，帳戶控制就顯得相當重要，每一個帳戶代表的是一個人，也可以是一個人他所致定的多本書籍，例如今天我們想要的是一本科學雜誌，那我們就可以加入像 Engadget、mobile01、數位時代，這樣子的科學來源，然後用一個帳戶做存取，取名為 Science，然後我們現在想要一本旅遊雜誌，那我們就可以加入類似愛旅遊，奇摩旅遊，許多食記旅遊部落格，然後將此帳戶取名為 Travel，這樣子以此類堆，我們一個人就可以擁有多個帳戶，而每一個帳戶代表的就是一本書籍，而這是一種方式，讓個人化更為多元，也更好整理，而帳戶控制最主要的方式，也就是希望可以給所有人使用，而系統要如何判別你是誰，給你屬於你自己的數位資源，就必須要由帳戶來做控制。



圖表 3-4 帳戶概念說明

除了上敘說到的分類書籍的概念，另外一種就是類似雲端的概念，其實我們的系統資料大可以存在 Client 端,但是這樣變成只有當我們取得這項設備的時候，我們才能觀看我們曾經訂閱的數位資料，又或許我們整理了許多資料要在電腦上觀看但是我們訂閱的來源都在 ipad 上面，這樣我們在電腦上使用系統的時候，就是須重新自己再訂閱一次，但是我們的系統可能更多，例如:iphone、ipad、PC、mac 等等，那這樣我們在使用系統時，就會顯得非常麻煩，所以本系統選擇利用帳戶的控制而不存在於 Client 端也是由於這個原因，雖然 Client 端讀取一定比 Sever 端更為快速，但是方便使用者在各個設備上來瀏覽共同的來源資料，我們選擇了 Sever 端，也就是 Account 的控制，這樣假設我們在 ipad 上面增加了許許多多的數位資料來源，我們回到家使用電腦時，看到的也會是剛剛所看到的來源資料，而又假設，你今天到了親朋好友家，使用別人家的電腦，你一樣可以使用屬於自己的電子書系統，繼續閱讀剛剛所沒看完的文章。



圖表 3-5 雲端概念

3.2.2 蒐集網路資源

加入來源的方式有很多，而最主要的方式就是找尋該網站 RSS 的來源網址，然後直接打入系統加入，當然這是最原始的方式，但一般來說，RSS 的來源網址都非常的長，不但要複製，還必須到系統上面做貼上加入的動作，實在不是非常人性化，所以本研究想出了一個方式，希望在瀏覽網站的同時，就可直接將來源加入到自己的系統，由於本系統的瀏覽平台就是我們一般電腦或是智慧型手持系統都會有的瀏覽器，而瀏覽器有一個很重要的功能，就是書籤功能，而這項書籤功能，也就是我們一般熟知的「我的最愛」，也就是將自己喜歡的網站家到所謂的書籤列中，那麼我們現在做一個本系統的書籤，然後其中放置 JavaScript 的程式碼，只要瀏覽到喜愛的網站，或是想要加入的網站，只需要在登入系統之後，瀏覽到該網站，直接點選這項書籤，就可以把此網站加入到你的個人化系統之中，這會是相當方便的，這項功能會進而省去了一堆複製貼上網址的複雜動作，使的本系統更為人性化，也更好操作。

而一個網站可能會有許許多多的 RSS 來源，例如維基百科，他的 RSS 來源就分為，最近更新、最熱門等等，而最麻煩的就是使用者必須去尋找這個網站的 RSS 來園網址在哪裡，今天我們只需要進到這個網站，做點擊 Bookmark 的動作，這樣子 Bookmark 就會自動去尋找此網站的 RSS 來源，甚至是 Atom 來源，然後系統列出來給使用者知道，以提供使用者去做選擇，這樣就可以讓使用者很方便的加入自己喜愛的來源。

圖表 3-6 Bookmark 概念

3.2.3 呈現

當用盡各種方法擷取到我們所要的數位資源之後，最重要的就是要如何呈現出我們所要的樣子，由於這是一本電子書，所以我們希望在呈現上可以更為多元，而不要再像網頁瀏覽的方式，有許許多多的連結，讓使用者眼花撩亂，我們要給使用者的是書籍閱讀的體驗，裡面就只有數位內容相關的資料以及所要觀看內容，而這些我們打算使用 javascript+CSS3+HTML5，讓使用者有很棒的閱讀體驗，而此系統有一個很大的重點，就是我們希望使用者不只是在平板上面可以使用，在電腦，在手機，在任何手持設備上都能使用，這就必須要使用 jQuery 來偵測解析度，以及設備，或著是利用瀏覽器本身的特性去做修改，這樣子，使用者就可以在任何設備上使用我們的系統，而且他會是最佳觀賞的狀態，詳細的細節，在第四章我們會有更多的描述。

3.2.4 稍後閱讀及分享紀錄

剛剛我們大致上把系統分為兩部分，前端與後端，而將這兩個部分連結再一起，則整個系統會從蒐集數位資料開始，然後由 PHP 去做抓取送到本系統，本系統在使用 HTML5 去作呈現，接著使用各項的瀏覽器及手持行動系統去做觀看數位資料，而本系統會加入幾項分享的功能，例如 Instapaper 稍後觀賞這項功能，如果使用者想將這篇文章或是網站要做稍後閱讀的動作，只需要使用這項功能，接著可以使用 Instapaper 這個 Application 在各項手持系統上去觀賞，另一項功能就是 Wordpress 由於現在許多人創立了屬於自己的 Wordpress 的專屬 blog，而透過這項功能，使用者可以分享自己喜愛的網站或是文章等數位內容。

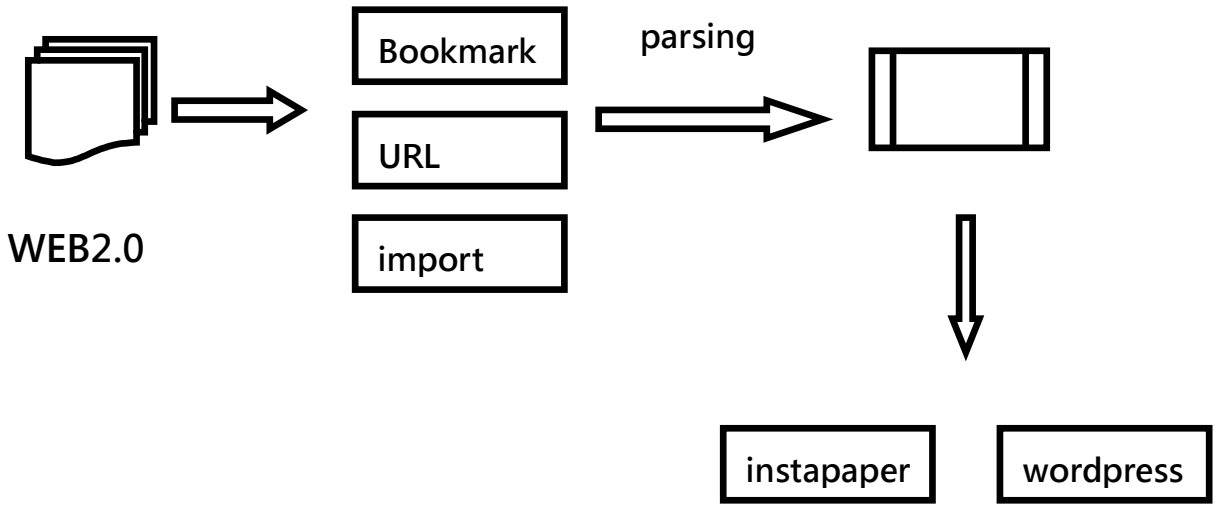
3.2.5 無障礙控制

使用者中一定會有身心障礙不方便的，而本系統希望在上面做一些無障礙的控制以方便使用者使用，例如語音系統或是盤快速鍵(Access Key)等等，以方便使用者使用，若有一些肢體上不方便的使用者不方便使用平板電腦或是滑鼠的使用者，可以利用鍵盤快速鍵，翻閱你想看的文章，進行切換的動作，或是在視覺障礙上面有不方便的使用者，可以利用語音系統來聽閱其中的內容。

3.2.6 離線閱讀

我們希望在使用本系統的時候，不只是在連線的時候可以使用，而是在離線的時候也可以閱讀曾經閱讀過的文章或是圖片，而且可以很正常流暢的使用本系統，而離線瀏覽這個機制，剛好在 HTML5 中有定義一項 manifest 宣告機制，而這個宣告機制就可以將宣告機制在網頁中的以運作，在第四章我們會有更詳細的介紹，大體說來，離線機制就是必須使頁面中所使用的網頁、圖片先行快取於系統之中，而在本系統中，我們使用了帳戶做為入口的控制，而離線機制就是僅存在本地系統之中，換言之，就是當你在使用 ipad 等設備的時候，你可以閱讀你曾經在『此設備』閱讀過的文章會存在設備之中，而如果要觀看你在其他設備看過的文章則必須連線登入帳戶才可行。

3.3 整體運作流程



圖表 3-7 整體運作

首先在本系統中 主要可以分為三個步驟

『擷取數位資料』、『編排及呈現畫面』、『稍後閱讀及分享紀錄文章』

擷取數位資料

我們主要分為三種方法:URL 網址、bookmark、匯入。

第一種方法是最為基本的一種方法，就是記下該網頁的網址，然後加入本系統之中，而第二種方法是本系統最為方便的方法，在你瀏覽網頁的同時，若你喜歡該網頁的內容，那就可以將它加入為自己的電子書來源，該 bookmark 是一串 javascript 的指令，他會抓取該網頁中的 RSS 或是 ATOM 檔案，並且列表出來給使用者知道，該網頁有哪些來源可供使用，然後供使用者來做選擇，在來就是使用者或許曾經自己在 google reader 中有已經整理好的來源檔案，而本系統可以直接作匯入的動作。

編排及呈現畫面

在這個部分也是本系統最為重要的一個部分，當我們手上握有自己的數位來源之後，最重要的就是如何呈現他，我們要使用的就是 HTML5+CSS3+javascript，我們必須將呈現出來的來源檔案作加工，讓他更成為我們要的電子書概念，然而不只在編排上面，我們還要對各種設備去做控制，例如觸控的操作，手勢等等，還有針對各個設備的解析度去作畫面的的排版，而在畫面的編排上有很多方法，我們也加入許多的 javascript 動畫、全螢幕等等，讓它更像是一個應用程式，而不是一個單純的網頁，但他擁有了許多網頁在雲端上的優點，在第四章我們會有更詳細的介紹。

稍後閱讀及分享紀錄文章

在我們的系統上閱讀電子書之後，我們可能會想要分享，會有時候沒時間觀看，或是不能專心閱讀，那麼我們加入了兩個元素，第一個『稍後閱讀』，常常我們在上班或是在上課，看到很喜歡的文章卻無法很專心的看完，而我們希望能讓使用者可以在沒有干擾的時候可以將之前覺得很棒的文章，拿出來閱讀，而且是極佳的閱讀介面，所以我們結合了 instapaper 的功能，讓使用者在某個設備上，若看到喜愛的文章，可以將此加入 instapaper，那麼這篇文章就會很即時的下載到有安裝 instapaper 的設備中，之後即使在沒有網路的情況下，我們也可以在悠閒有空的時候，作車的時候等等，拿出手機或是平板，觀看自己覺得很棒的文章，而 instapaper 也有著筆記紀錄或是分享文章等等的功能，使用者也可以多加利用。

接著『分享紀錄文章』，我們是希望使用者，在觀看到喜愛的文章之後，不只可以作稍後閱讀的動作，更可以將此文章存在自己的部落格中，並且加入許多自己的見解等等，然後發布出去讓親朋好友知道，這不但可以分享文章，更可以蒐

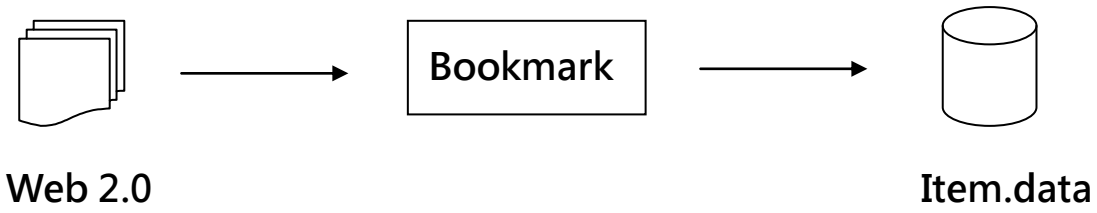
集自己所喜愛的文章在自己的部落格之中，所以使用者如果看到喜愛的文章，除了在本系統可以作稍後閱讀的動作將文章存於自己的手機或是其他設備之中，更可以存在所謂雲端的部落格之中，除了稍後閱讀、分享、紀錄等等，也有著備份的功能在其中，詳細的說明我們會在第四章作詳細的描述。

Chapter 4 系統設計與開發

4.1 系統主要畫面設計

4.1.1 加入來源

Bookmark



圖表 4-1 Bookmark 架構

我們將這個 bookmark 做成一個書籤的形式，就像平常我們點選書籤這樣，當我們瀏覽到覺得不錯的網頁的時候，我們只需要點選這個 bookmark 書籤，就可以將此網頁加入為我們的數位內容來源，而這個 bookmark 簡單來說，是一串 javascript 指令碼，如下:

```
javascript:
void(sp=window.open('http://140.122.184.35/albert/admin/subscribe.php?url='+
escape(document.location)
```

而這串指令碼的主要動作，是將目前的網頁 URL 放入 subscribe.php

這個 PHP 網頁中儲存，然後將此 URL 當作變數，並且將此 URL 中的來源抓取出來，儲存在 items.data 中，使得以後登入該帳戶的時候，可以抓取出該帳戶的來源，而 subscribe.php 會去找尋該 URL 中的來源以加入該系統中，下面的例子是當中的一段程式碼

```
try {
            $feed = $_POST['url'];
            $discovered = self::get_discovered($feed);
            if (count($discovered) > 1) {
                return self::choose_page($feed, $discovered);
            }
            if (!empty($discovered))
                $feed = $discovered[0];
            $result = Feeds::get_instance()->add( $_POST['url'],
$_POST['name'] );
        }
        catch( Exception $e ) {
            return self::sub_page($e->getMessage());
        }
        return self::success_page($result);
```

當中前面會有些判斷式，判斷 bookmark 中有無抓取到該網頁的 URL 一但抓取到之後，則執行上續的程式碼，首先會將 URL 存入我們的$feed 變數中，接著搜尋此 URL 有無 RSS 或 ATOM 的來源，而他可能不只一個，只要數量大於一則更新我們的來源 item 欄位，最後則將使用者選擇的來源資料回傳給 success_page 這個函數處理。

URL

然而除了 bookmark 的加入方式之外，使用者也可以直接將已知的 URL 直接打入系統中，然後直接傳入 PHP 網頁中讀取:

```
<h2><?php _e('Add Feed'); ?></h2>
    <fieldset id="required">
        <div class="row">
            <labelfor="add_url"><?php_e('Feed address (URL)'); ?>:</label>
            <input type="text" name="add_url" id="add_url" />
        </div>
    </fieldset>
```

例如編輯 URL、刪除 URL、替來源命名等等:

```
<thead>
        <tr>
        <th><?php _e('Feed Name'); ?></th>
        <th><?php _e('URL'); ?></th>
        <!--<th><?php _e('Category'); ?></th>-->
        <?php do_action('admin-feeds-infocol-description'); ?>
        <!--<th class="change-col"><?php _e('Edit Feed'); ?></th>-->
        <th class="remove-col"><?php _e('Remove Feed'); ?></th>
        <?php do_action('admin-feeds-actioncol-description'); ?>
        </tr>
    </thead>
```

而我們在上述的控制中加入了刪除 URL 的機制，而我們怕使用者將曾經加入過的來源網址誤刪，所以我們在刪除的同時加入了回復機制:

圖表 4-2 回復設計架構

如上圖在按下 Delete 的同時，系統會先備份一份來源的相關資料，若使用者誤刪之後，可以按下 Undo 來將之前儲存的資料回復過去，以避免誤刪。

```
$removed = $this->feeds[$id];
$removed = apply_filters('feed-delete', $removed);
$cache = new DataHandler(get_option('cachedir'));
```

先將 ID 等等的資訊備份到另一個 remove 變數中，以利回復。

4.1.2 插件設定(plug-in)

Wordpress & instapaper

首先，本系統為了以後可以方便加入自己想要的額外功能，所以將分享及稍後閱讀的功能分離出來，放在插件這個部分，日後，如果想要加入新的功能，只需要在這裡做編輯即可，然後這個區塊，我們會顯示該系統中提供的功能，下面是大致上的 table 架構:

```
<form action="settings.php" method="post">
    <fieldset id="plugins">
        <table class="item-table">
            <thead>
                <tr>
                    <th scope="col"><?php _e('Plugin') ?></th>
                    <th scope="col"><?php _e('Description') ?></th>
                </tr>
            </thead>
            <tbody>
<?php
```

進行開或關閉的操作，因為一旦數量過多，可能會造成頁面讀取過慢，而影響了閱讀的體驗。

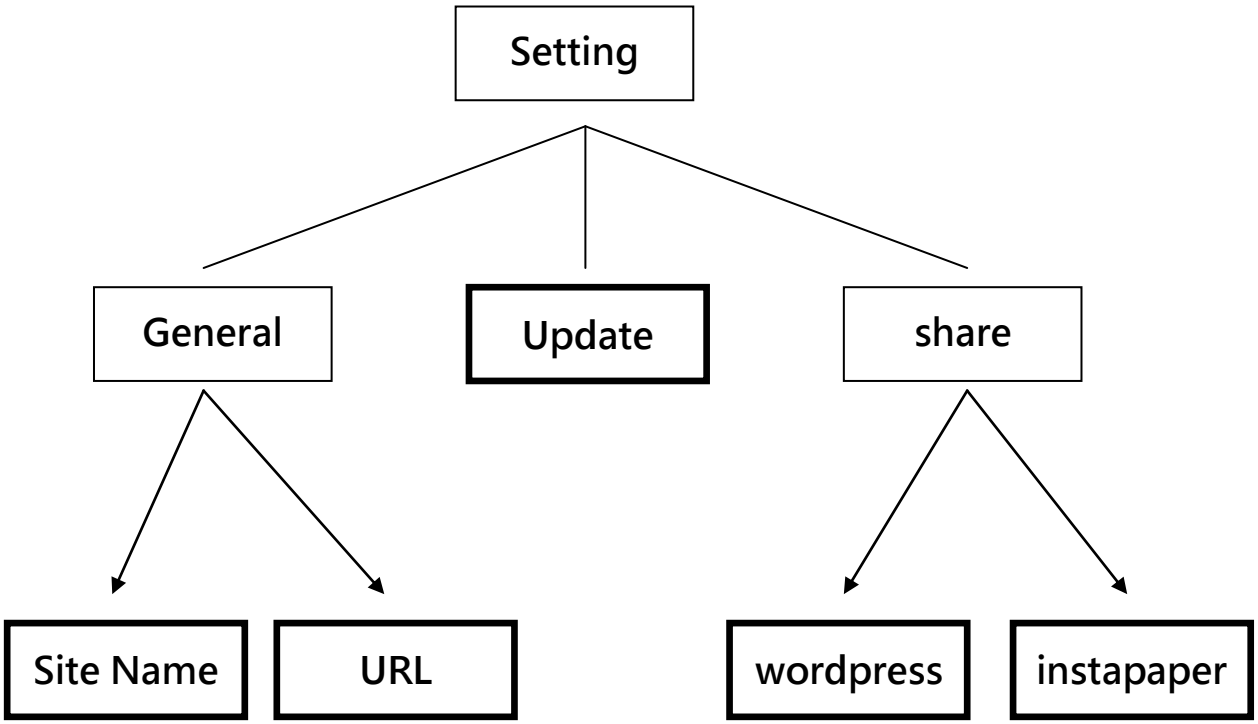
```
<tr class="<?php echo $class ?>">

<td class="plugin-name"><span class="name">

<?php echo $plugin->name ?>

</span>

<p class="plugin-actions"><?php echo $actions ?></p></td>

<td class="plugin-desc"><?php echo $plugin->description ?>

<p><?php echo implode(' | ', $info) ?></p></td>

</tr>
```

我們可以看到當中有一個 class 我們命名為『plugin-actions』，而這就是我們拿來作為 plug-in 開關的類別，當我們將插件設為 activate

就將此 PHP 檔案設為啟動，建立連線，如下：

```
if ($type === 'activate') {
        Lilina_Plugins::activate($plugin);
        header('HTTP/1.1 302 Found', true, 302);
    header('Location:get_option('baseurl')  .'admin/plugins.php?activate
d=1');
        die();
    }
    else {
        Lilina_Plugins::deactivate($plugin);
        header('HTTP/1.1 302 Found', true, 302);
    header('Location:get_option('baseurl') .'admin/plugins.php?deactiva
ted=1');
        die();
    }
```

4.1.3 系統設定



圖表 4-3 系統設定樹狀圖

這個部分，是整個系統的總體設定的區塊，架構大致如上，裡面包含了系統名稱、伺服器位置、自動更新網址、分享系統設定，只要是有關系統的任何設定我們都擺在這裡。

系統名稱

在這裡，使用者可以將自己的系統設成自己所要的名稱，而在使用本系統的時候，左上角的首頁按鈕就會是自己設定的使用者名稱。

伺服器位置

這個部分，是開發者的部分，如果開發者要更動伺服器位置時，必須將此 URL 設為伺服器的 IP 位置，而避免使用者不小心去更動到，我們要設定時必須到 Setting,php 的設定檔案中去設定，在這個部分中相當的重要，如果 URL 設定的不正確，javascript 將會無法執行，而影響整個系統的執行。

自動更新網址

這個自動更新網址，不算是設定的一部分，只是系統要告訴使用者一段網址，而這一段網址會呼叫我們寫好的 update method，這一段網址對使用者的用意，是希望讓使用者建立一個很快速的更新書籤，使用者如果來源數眾多，則必須要時常更新，也必須花費一些時間，因為系統會將資料存到本地端 Client，所以使用者在瀏覽網頁的同時，可以不時地去點擊這個網址，讓他直接在背景更新來源內容。

Instapaper 帳戶設定

Instapaper 的使用方式，必須在進入時，輸入一組帳號密碼，而本系統則必須先記錄使用者使用的 instapaper 的帳號密碼，在使用者點擊 instapaper 這項功能時，我們就會直接將此紀錄的帳號密碼，匯入 instapaper 的 api 中，然後將資料送到 instapaper 中。

Wordpress 帳戶設定

Wordpress 是世界上知名的部落格系統，而在這裡，我們需要使用者的 Blog 的 URL，一旦使用者想要分享此數位資源，點擊 wordpress 按鈕之後，我們就會將資料送到 blog 中，並且跳換頁面，讓使用者發佈該資料。

4.2 系統運作方式

4.2.1 擷取數位資源

單一網站，各種來源

一般我們在加入來源的時候，最討厭的就是找不到此網站是否有無 RSS，就算有，也不知道放在哪裡，所以前面所設計的 Bookmark 功能，我們更加他強化，我們用一個陣列$discovered 存入所找到的來源項目，在出現來源選單的時候，我們將他列出來給使用者選擇，並且列出此來源的名稱，如下:

```
foreach ($discovered as $feed) {
echo '<li><input type="radio" name="url" value="' . $feed['file']->url .
'" /> ' . $feed['title'] . '</li>';
        }
```

首先 foreach 會對陣列 $discovered 做迴圈，並將目前所指元素的值放到 $feed 變數裡，然後陣列裡的指標會跟著移到下一個元素的位置。

設定來源名稱

有些網站，他原本預設的名字非常的龍長，而有些則因為編碼的問題，成出現了亂碼，當然，我們為了美觀，我們希望可以讓使用者，也可以自己命名來源網站的名稱，

```
$("<p class='hideshow'>
<span><?php_e('Showadvanced') ?></span></p>")
.insertBefore(".optional").click(function () {
$(this).siblings(".optional").show();
$(this).hide();
});
```

原來的 Feed 名稱，而我們設定了一個 Showadvanced，一旦使用者點擊了這個選項，就會呼叫 optional 這個 class，而這個 class 則會將使用者所輸入的名稱覆蓋掉原本的名稱，如果 input 是空的，則系統就會採取原本的名子。

```
<fieldset id="optional" class="optional">
<p>
<label for="name"><?php _e('Name'); ?>:</label>
<input type="text" name="name" id="name" class="input input_small" />
</p>
<p class="sidenote"><?php _e('If no name is specified, it will be taken from the feed'); ?></p>
</fieldset>
```

擷取數位資源相關訊息

```
<div id="item">
<div id="heading"><h2 class="item-title"><a />
</h2><p class="item-meta">
<span class="item-source">From <a /></span>.
<span class="item-date">Posted
<abbr /></span>
<span class="item-author"> by <a /></span>.</p></div>
<iframe id="item-content" class="framed" /></div>
```

發佈的作者，發佈的時間，而這其中較為麻煩的就是發佈的時間，我們必須用一個變數與方法來儲存與計算他，如下我們設定了一個負責轉換日期的變數$conversions，好讓我們在呈現的時候更可以讓使用者方便的清楚知道日期及時間

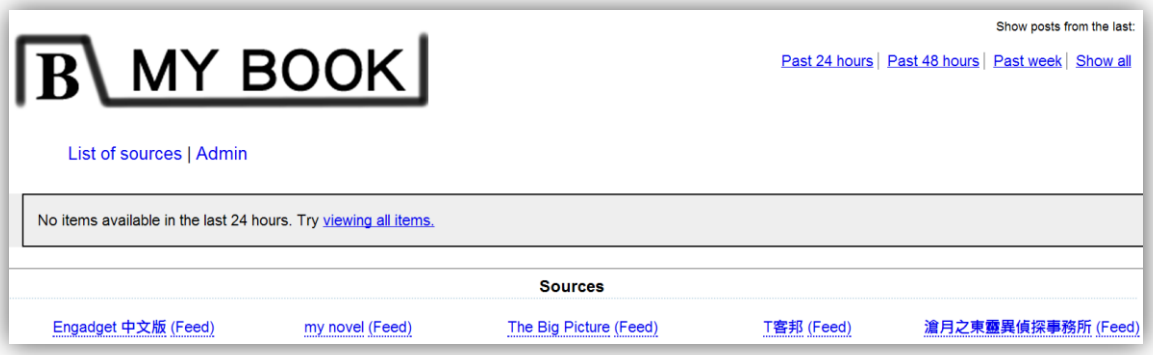
```
var delta        = new Date() - this;
var conversions = {
millisecond: 1, // ms      -> ms
      second: 1000,
      minute: 60,
      hour:    60,
      day:     24,
      month:   30,
      year:    12
};
```

```
for (var key in conversions) {
      if(delta < conversions[key]) {
            break;
      } else {
            units = key;
            delta = delta / conversions[key];
      }
}
```

Delta 是我們用來儲存日期的變數，而計算後我們呈現出來的樣子會是 『From “來源名稱” Posted “時間” ago by “作者” 』。

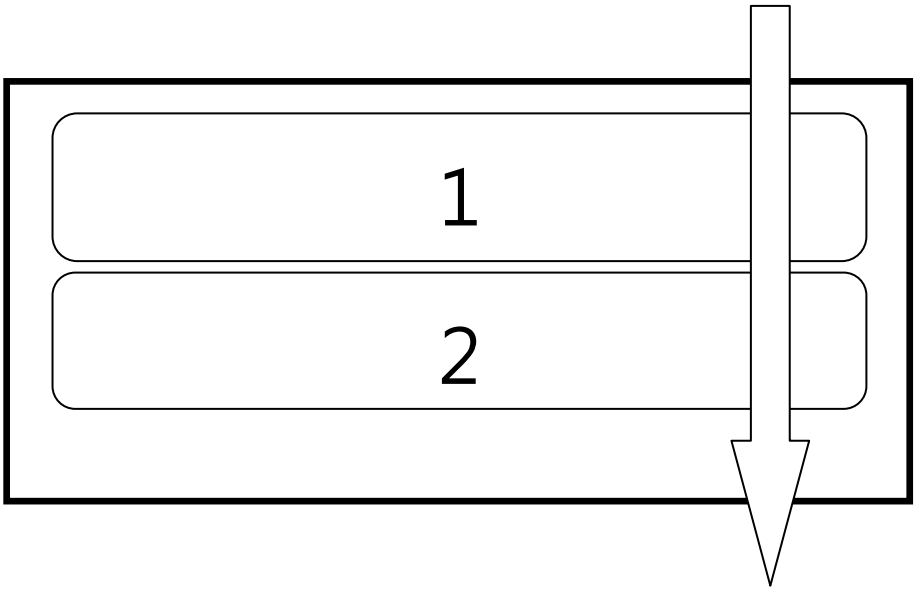
4.2.2 呈現

呈現數位內容及來源各項訊息



圖表 4-4 來源呈現

如畫面呈現的 下面我們列出來源的網站名稱，然後其他地方有時間排序以及內容呈現的地方，我們是希望使用者可以在一個畫面內都不用做切換的動作，就可以一直瀏覽所要看的資料，



圖表 4-5 瀏覽概念圖

如上述畫面所示，我們一個一個將抓到 RSS 內容，呈現在一個畫面上面，並排列下來

圖表 4-6 來源呈現畫面

只要點擊其中一個標題，畫面就會向下拉動，而不會有畫面更換的問題，使用者只會在同一個畫面中瀏覽。

圖表 4-7 系統瀏覽畫面

雖然我們可以玩正呈現我們所要的數位內容，但是這樣還是太過於簡陋，所以之

後我們會再貼家更多的使用者介面，

呈現排序

圖表 4-8 排序畫面

這麼多數位資源，那今天我們要怎麼排序，我們抓取到的來源資訊，當中有我們計算後的時間，而我們將這個時間最為排序的依據，

```
<div id="times">
<p><?php _e('Show posts from the last:'); ?></p>
<ul>
<li><a href="index.php?hours=24"><?php printf(_r('Past %d hours'), 24) ?></a></li>
<li><a href="index.php?hours=48"><?php printf(_r('Past %d hours'), 48) ?></a></li>
<li><a href="index.php?hours=168"><?php _e('Past week') ?></a></li>
<li class="last"><a href="index.php?hours=-1"><span><?php _e('Show all') ?></span></a></li>
</ul>
</div>
```

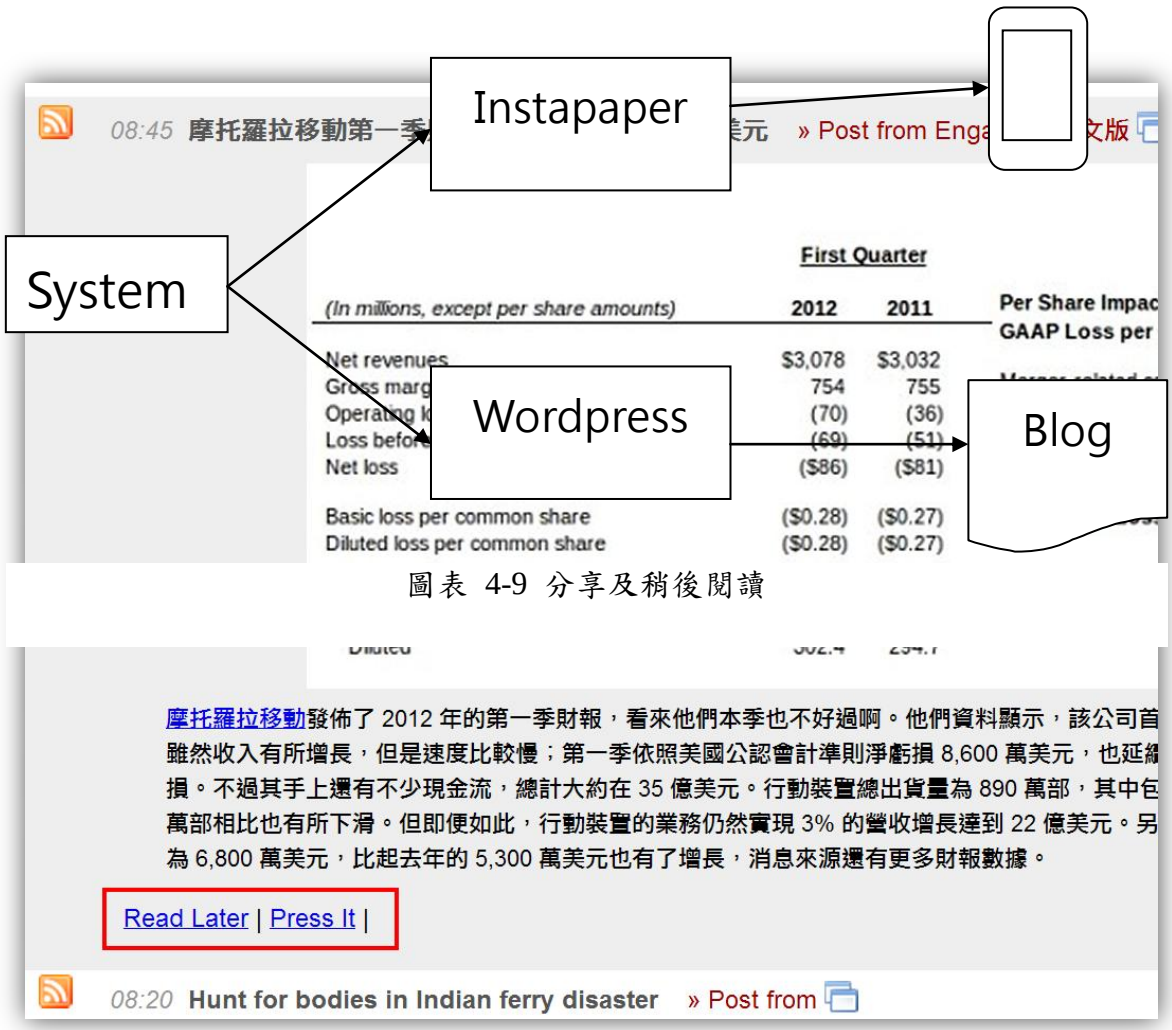
而如上述所示，我們選擇不同的時間 那麼系統就會依照該排續將數位內容呈現，

例如 最近 1 小時、24 小時、這一個禮拜等等，那麼使用者

如果要觀看今天的內容也只需要選擇該排序方式即可，而關於這個時間排序的方式，可以由開發者很簡單的修改他，也可以自訂排序方式。

4.2.3 分享及稍後閱讀

我們希望可以在方便閱讀的同時增加一些互動的功能，如下所示，每一篇抓取到的數位內容，我們在下面加入兩個功能:『Read Later』和『Press It』



圖表 4-9 分享及稍後閱讀

圖表 4-10 分享及稍後閱讀畫面

Wordpress 功能設計

首先我們來看到 Wordpress 的部分，我們所要做的動作就是將你所選擇的數位內容傳入你的部落格發佈文章的頁面，除了原本你所加入的數位資源之外，還可以自行編輯其他編輯的動作，例如筆記 ，做重點等等。

```
function pressit_options() {
    register_option('pressit_wpurl', 'pressit_validate');
    add_option_section
    ('pressit_section', 'Press It Options', 'pressit_section_header');
    add_option_field
    ('pressit_wpurl_field', 'WordPress URL', 'pressit_field', 'pressit_section',
array(
        'label_for' => 'pressit_wpurl',
            ));
}
```

傳入 option_field 函數中處理，這樣我們就可以把使用者所選擇的數位資源，不管是 RSS 還是 Atom 都可以傳入你所在的 wordpress 的 blog 位置。

Instapaper 功能設計

由於使用者不可能隨時隨地都可以很專心的閱讀自己所喜愛的文章或數位內容，所以我們加入了 Instapaper 這項稍後閱讀的功能，而這個 instapaper 的官方就有釋出 API 供大家使用，而我們把它加入了此系統，讓使用者可以在通車或是較為方便的時候，也可以用小小的手機來觀賞文章，而 instapaper 當中也有著不少的功能，例如分享、編輯，字體放大，儲存等等，它的好處就是，一但當你將此數位內容加入 instapaper 之中此系統就會將資料送入，而且是存在手機裡面的，所以使用者不用擔心網路的問題，就可以好好的來閱讀文章。

```
$response = HTTP::post("https://www.instapaper.com/api/add", array(), $data, array('redirects' => 2));
```

首先我們必須呼叫 instapaper 的 API 來做 add 的動作。

```
$data = array(
        'username' => get_option('instapaper_user', ''),
        'password' => get_option('instapaper_pass', ''),
        'url' => $item->permalink,
        'title' => apply_filters( 'the_title', $item->title ),
            );
```

接著我們設定一個 data 變數來儲存許多 instapaper 需要的資訊，例如帳戶名稱及密碼等等。然後系統便會啟動 instapaper 的 api 將資訊傳入，一但你按下稍後閱讀的按鈕之後，下次打開 iphone 或是 ipad 之後，你就會看到你所加入的資料了。

4.3 閱讀介面設計

從上述我們可以看到較為陽春的介面，而我們現在所談到的便是 web Application 的概念，它本身的性質是一個網頁，但是由於應用程序非常的多，所以她其實看起來就像是一個應用程式，一旦放大到全螢幕之後，使用者就不會有在瀏覽網頁的感覺，而是一個很方便的應用程式，霄我們要使用的語言除了 HTML5+CSS3 之外，我們也會大量的應用到 javascript 以及 jQuery。

4.3.1 Web Application

適應性網頁設計

由於我們的概念是可以讓此系統在多種設備上都可以執行，但是各種設備的螢幕大小及解析度不同,那我們要如何針對每一種設備去讓使用者有很好的使用體驗，這裡我們只需要 CSS3 的一些概念就可以完成。

首先，我們將系統分為三大類:電腦、平板、手機。

而這三大類我們又可以將解析度分別作調整，

電腦 :1024px

平板 :640px-1024px

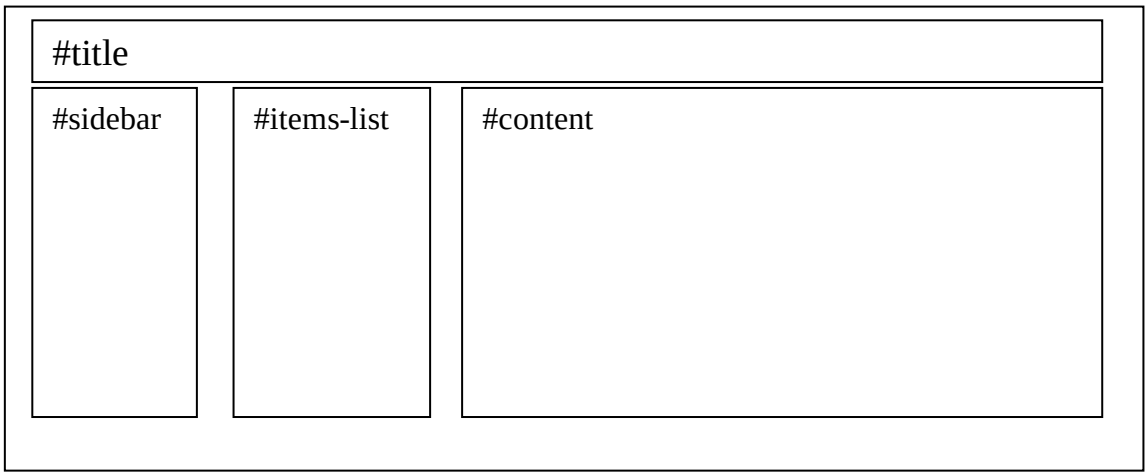
手機 :640px

```
@media only screen and (min-width: 640px) and (max-width: 1028px) {
#category{
! float: left;
! font-size: 200%;
! min-height: 76%;
}
```

如上述的 CSS3 ，這是針對平板的一個 CSS3 程式碼,一但系統偵測到是平板電腦，我們就會套用此 CSS 檔案來讓他達到最佳的效果，而其他設備則以此類推，就可以在每個設備達到適應性的效果

使用者介面設計

上述我們可以看到使用的介面是相當陽春的，並且使用方式是不斷的向下滑動，現在我們希望使用者可以清楚的看到自己有哪些來源，並且大致上有哪些文章，所以我們將系統頁面做了改進，並且套入 css3 及 javascript。



圖表 4-11 使用者介面架構圖

如上，我們將螢幕切割成三大部分，第一塊是來源(sidebar)，第二塊(items-list)是文章標題，第三塊(content)則是要呈現的數位內容。而套用後的樣子會如圖所示。

如上圖所示，我們可以看見兩個紅框，我們在這裡加入了 javascript，而這兩個拉桿的用途就是方便使用者隨自己的喜好，可以自行決定三個區塊的大小分別要多少，如果使ㄨ用者不想看到標題和來源項目，那使用者就可以將之拉至最旁邊，這樣整個畫面就只會有使用者所要閱讀的內容。

```
    $.resizeHandle = {
        drag: function (event) {
            event.data.element.css({
                width:
        Math.max(event.pageX - event.data.posX + event.data.width, 0)
            });

            if (typeof event.data.callback ! = 'undefined') {
                event.data.callback.call();
            }
            return false;
        },
        stop: function (event) {
            event.data.element.css('opacity', event.data.opacity);
            $(document).unbind('mousemove.resizer',
$.resizeHandle.drag).unbind('mouseup.resizer', $.resizeHandle.stop);
        }
    };
```

和減的動作，使用者就可以來自行訂定操作頁面的大小。

觸控設計

由於我們必須要在多種介面上執行，所以執行的方式也會不一樣，就例如，電腦的操作介面是滑鼠，而平板及手機的操作介面則變成了手指，這當中如果要讓自己的網頁可以在各個設備上執行，則必須有所轉換，在觸控操作時，自然不會有 mousedown、mousemove、mouseup 這些事件，所以我們必須另外定義了 touchstart、touchmove、touchend 等觸控平台的專屬事件，但其實如果我們一開始已經為了滑鼠寫好了程式碼，我們之後只需要一段 javascript 就可以將此事件

做轉換，這樣不管是在任何平台上，我們都可以執行。

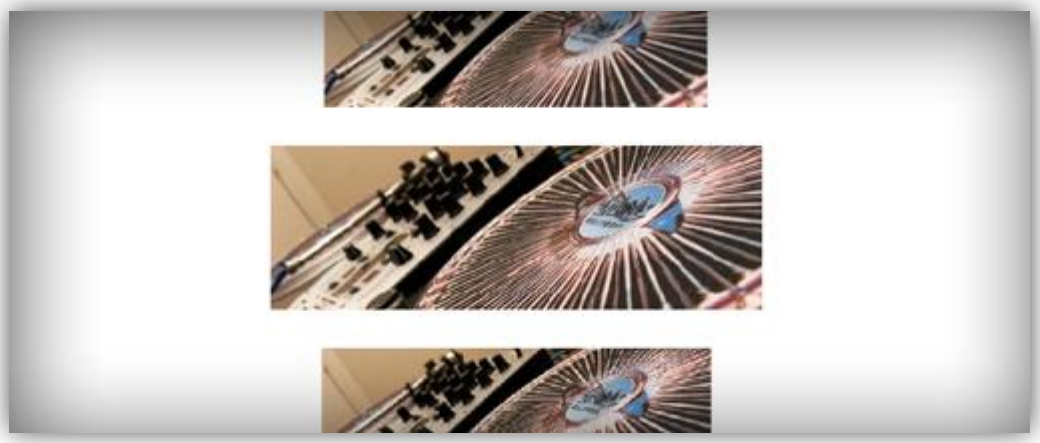
```
var mouseEventTypes = {
    touchstart : "mousedown",
    touchmove : "mousemove",
    touchend : "mouseup"
};
for (originalType in mouseEventTypes) {
    document.addEventListener(originalType, function(originalEvent) {
        event = document.createEvent("MouseEvents");
        touch = originalEvent.changedTouches[0];
        event.initMouseEvent
        (mouseEventTypes[originalEvent.type], true, true,
                window, 0, touch.screenX, touch.screenY, touch.clientX,
                touch.clientY, touch.ctrlKey, touch.altKey, touch.shiftKey,
                touch.metaKey, 0, null);
        originalEvent.target.dispatchEvent(event);
    });
}
```

一開始我們的系統，是無法在平板上做流暢的操作，由於觸發事件的不同，而現在如上述的程式碼，我們可以看到，我們又分別定義了三種觸發事件 touchstart、touchmove、touchend，並且做轉換，這樣下來，我們在平板上也可以做 javascript 的多項動畫操作，例如需要用到 mousemove 的 resize 功能。

4.3.2 外觀調整

介面架構整理好之後，我們希望可以讓系統看起來更為美觀，並且加入一些動畫操作等等，而其實動畫操作在 HTML5+CSS3 上面便可以很簡單的就加入開發系統，而不需要用到任何一點 javascript 的程式碼，下面幾個是我們想加入的動畫操作，而我們只需要 CSS3 便可以簡單完成。

Stack & Grow



圖表 4-12 CSS3

左邊列表上，我們可以看到很多標題列表以及來源列表，這一塊塊的區塊，我們可以套用這個 CSS，讓每一塊區塊在選擇時可以更為明顯。

```
#sidebar{
    height: 100px;
    width: 300px;
    margin: 15px 0;
    -webkit-transition: all 1s ease;
    -moz-transition: all 1s ease;
    -o-transition: all 1s ease;
}
```

如上面的 CSS 程式碼，這是我們在還沒選擇時會呈現的狀態，而動畫時間我們設定為一秒。

```
#side-select {
    height: 133px;
    width: 400px;
    margin-left: -5px;
}
```

Fade In and Reflect

圖表 4-13 CSS3

這是一個讓圖片或是選項可以出現反射效果的 CSS3，而從來源抓取到的文字及圖片，我們也可以用這項 CSS 去做調整，這樣看起來會美觀很多。

```
#item-content a img {
            border-color: #deb887;
            border-width:thick;
            -webkit-box-reflect: below 0px -webkit-gradient(linear, left top, left bottom, from(transparent), color-stop(.7, transparent), to(rgba(0,0,0,0.4)));
        }
```

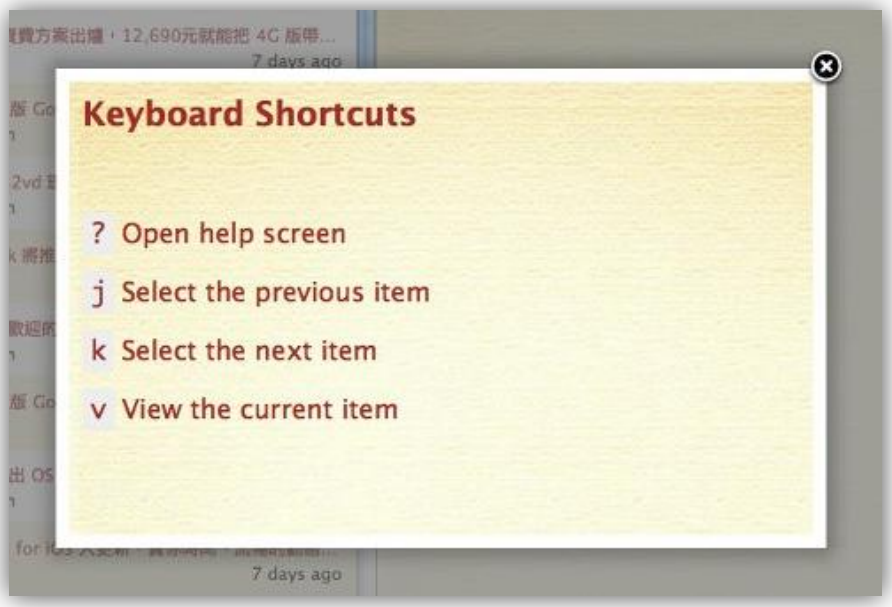
這是 CSS3 的程式碼，我們只需要上述藍色這一行，就可以為樸片製造出反射的效果

圖表 4-14 倒影呈現

這樣看起來就更美觀了。

4.3.3 無障礙控制

鍵盤快速鍵



圖表 4-15 鍵盤快速鍵畫面

在這邊，我們希望有障礙的使用者，無法方便的使用滑鼠，所以我們希望提供鍵盤的快速鍵讓使用者可以方便的使用，而鍵盤快速鍵包含了上下頁、呼叫使用說明、更新等等，而要製作鍵盤快速鍵，就必須要將鍵盤的控制對應到程式碼之中，這必須使用到 jQuery。

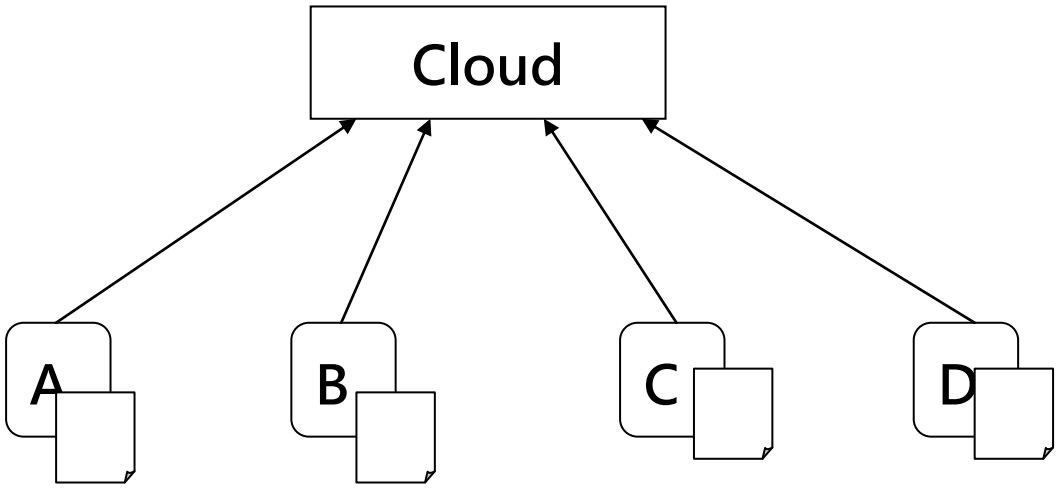
```
jQuery(document).ready(function (a) {
    $("a[hotkey]").each(function () {
        $.hotkey($(this).attr("hotkey"), $(this).attr("href"))
    });
    $(document).bind("keydown.hotkey", function (c) {
        if (!$(c.target).is(":input")) {
            if (c.ctrlKey || c.altKey || c.metaKey) return true;
            c = c.shiftKey ? c.keyCode : c.keyCode + 32;
            if (c = $.hotkeys.cache[c]) {
                $.isFunction(c) ? c.call(this) : (window.location = c);
                return false
            }
        }
    })
```

則會執行相關的動作並且回傳 true 值。，

Voice Ove 語音調用

由於我們主要測試的平台還是 ipad 及 iphone，所以我們直接將系統內的 Voice Over 拿來使用，由於他原先的設計，就是可以分段落來唸出所點選的文字，而我們在內容呈現，剛好是一頁一頁，所以只要使用者點選，內容的框架欄位，系統便會自動將整篇內容唸出來，這絕對會對視障的使用者帶來極大的方便。

4.3.4 離線閱讀



圖表 4-16 離線閱讀儲存圖

如圖所示，假設有 A、B、C、D 設備，那在使用本系統的時候，一般都是連到我們的的伺服器端，伺服器端存有我們的帳戶資料，例如來源項目，Instapaper 的帳號密碼等等，而幻想有一天我們在沒有網路的時候該怎麼辦，這也是一般雲端概念最為詬病的一個部分，就是離現時幾乎所有系統會停擺，而關於這個問題，我們使用的 HTML5 的離線規格，讓這個系統在離現時也可以很方便的使用。

而如圖所示，每一位使用者，在使用本系統的時候，只要是使用者曾經瀏覽過的資料，或是加入的來源項目，會存在每一個設備之中，所以每一位使用者，只要是你曾經在此設備瀏覽過的資料，都會存在設備之中，換言之，一

旦連上網路，不同的是惡被看到的內容會是相同的，但是在離現時，我們會根據此設備曾經瀏覽過的紀錄，而呈現給使用者觀賞，這就必須使用到 HTML5 的離線機制，這樣子，使用者只要曾經觀看過的資料，都會存在於系統之中。

HTML5 的強大功能，未來只用 HTML 與 Java Script 便可以實做出許多的網頁版本的應用軟體也就是所謂的 Web Application，在行動裝置如 iPhone,ipad 上已經有許多軟體實做。其中一項我們要來拿實作的的功能就是離線 Web Apps 的應用。離線網頁的第一個問題，是必須使頁面中所使用的各項網頁、圖片先行快取於系統中。欲達到此目的，規格中定義了一個 manifest 宣告機制，你必須在此檔案中宣告所有要快取的檔案列表。規格中亦定義了一組 Application Cache API，使你知道目前快取狀態。若要作離線程式，第一個要判斷的可能還是網路連線狀態，雖說有時會是電腦依然接在網路上，但是卻連不上網路的狀態，但是若系統可以自行告知目前連線狀態，就可以省下不停確認連線狀態的白工。

接著我們來看看 HTML5 的離線規格，主要分為三部分

1. 在 html 標籤裡面指定 manifest 檔案
2. 在 manifest 檔案裡面指定離線資源
3. API 及相關的事件

在 html 標籤上設定 manifest 屬性

```
<html lang="zh-TW" manifest="test642.manifest">
```

在離線應用上，HTML5 制定了 ApplicationCache 介面：

```
interface ApplicationCache {
   // update status
   const unsigned short UNCACHED = 0;
   const unsigned short IDLE = 1;
   const unsigned short CHECKING = 2;
   const unsigned short DOWNLOADING = 3;
   const unsigned short UPDATEREADY = 4;
   const unsigned short OBSOLETE = 5;
   readonly attribute unsigned short status;
   void update();
   void swapCache();
      attribute Function onchecking;
      attribute Function onerror;
      attribute Function onnoupdate;
      attribute Function ondownloading;
      attribute Function onprogress;
};
```

在一般的使用狀況下，通常只要設定好 manifest 檔案就可以離線運作了，複雜的應用程式，可能還需要搭配 locationStorage、sessionStorage 還有 Web SQL 等等方式，來存放應用程式需要的資料。這樣一來只要在系統中設定好 HTML5 的 manifest 之後，就可以使系統擁有離線的閱讀的功能了。

Chapter 5 個人化電子書系統測試

5.1 取得數位資源

在介紹完整個系統的設計開發重點之後，本章節將實際做測試，並且將各部分的細節畫面呈現出來，第一個部分將呈現本系統擷取資源的方式，接著是各項的設定，再來是畫面的呈現，最後將各個設備上呈現的畫面呈現出來。

5.1.1 Bookmark

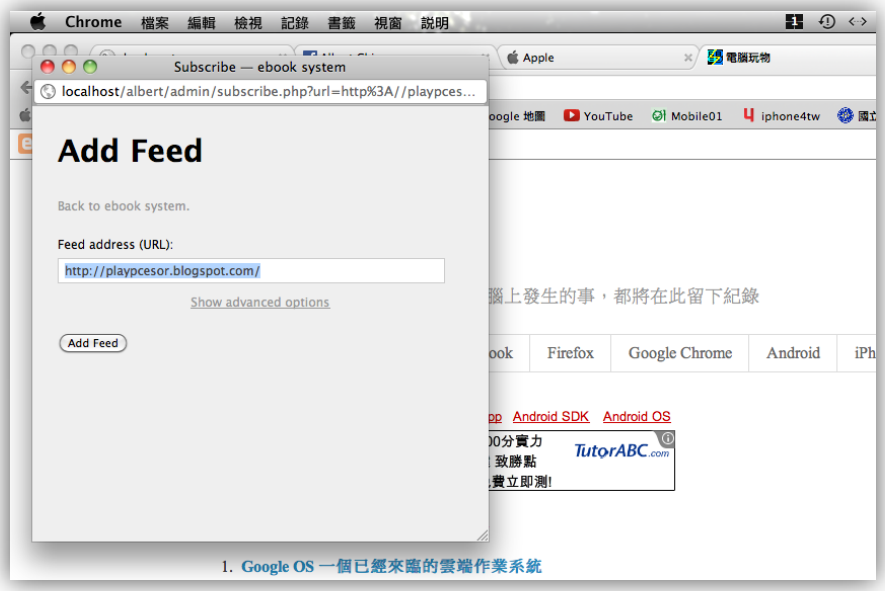
使用者一開始若瀏覽到喜愛的網站或是文章，便可以使用這個 Boomkark 來加入來源，如圖上面的 Subscribe 將之拖移到上方的書籤之後，以後要加入點擊這個書籤就可以了。

圖表 5-1 Bookmark 實作

本系統最大特色就是只要有支援 RSS 或是 ATOM 的網站，就可以加入。

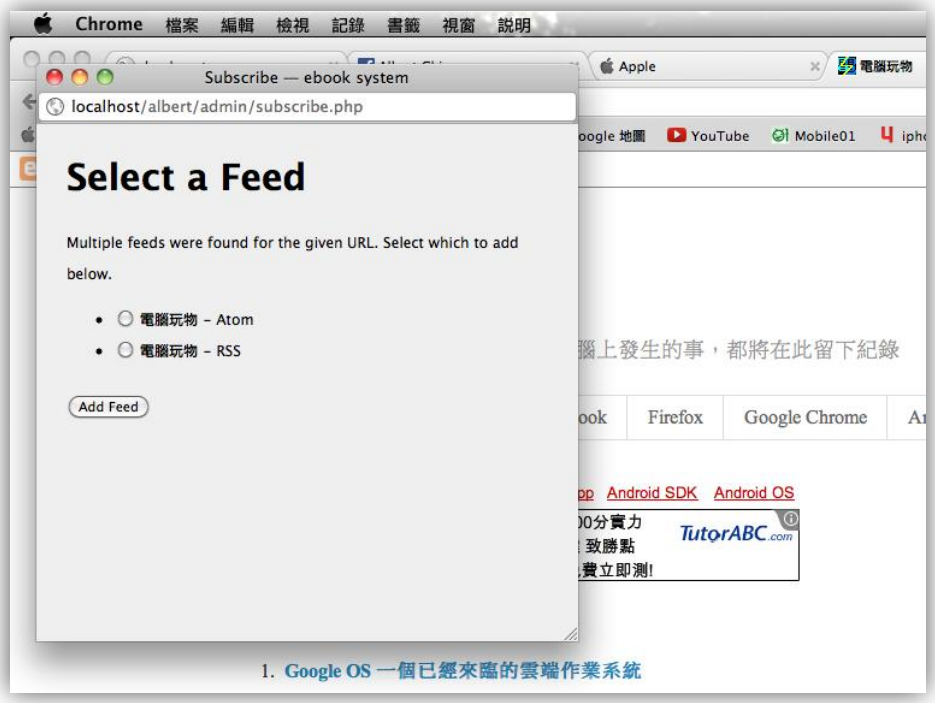
兩種格式

首先我們示範一個擁有兩種格式的網站『電腦玩物』，如下圖，當進入這個網站之後，點擊 bookmark 就可以加入:



圖表 5-2 bookmark 實作

點擊之後，系統便會出現詢問畫面。



圖表 5-3 bookmark 實作 2

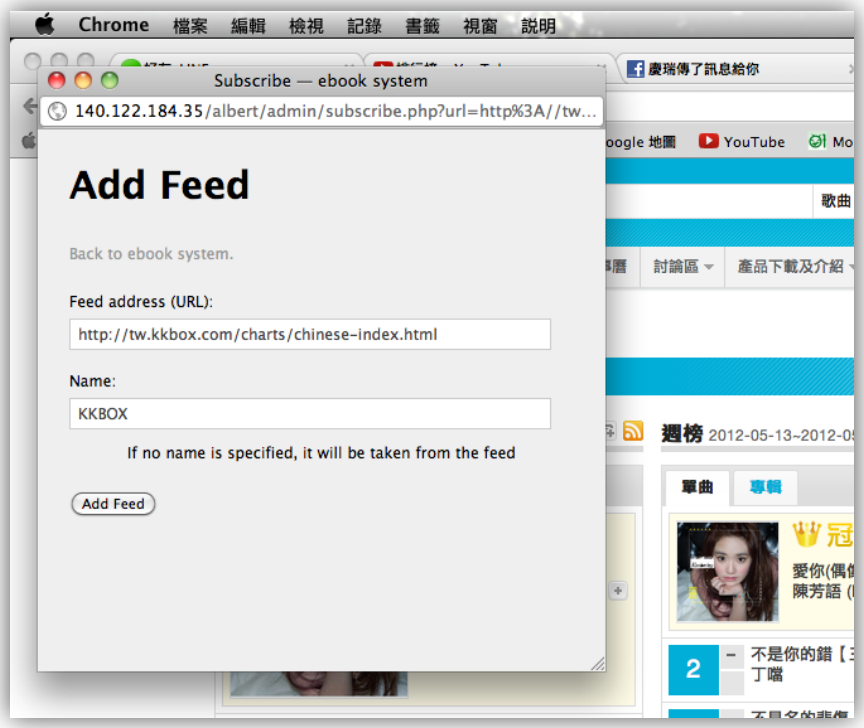
若此網站有多種來源格式，本系統都有支援，如圖系統便會詢問使用者所要加入的來源格式。

多種來源

接下來我們示範當一個網站若存在著多個RSS來源，Bookmark是如何運作的，這邊我們示範KKBOX的官方網站：

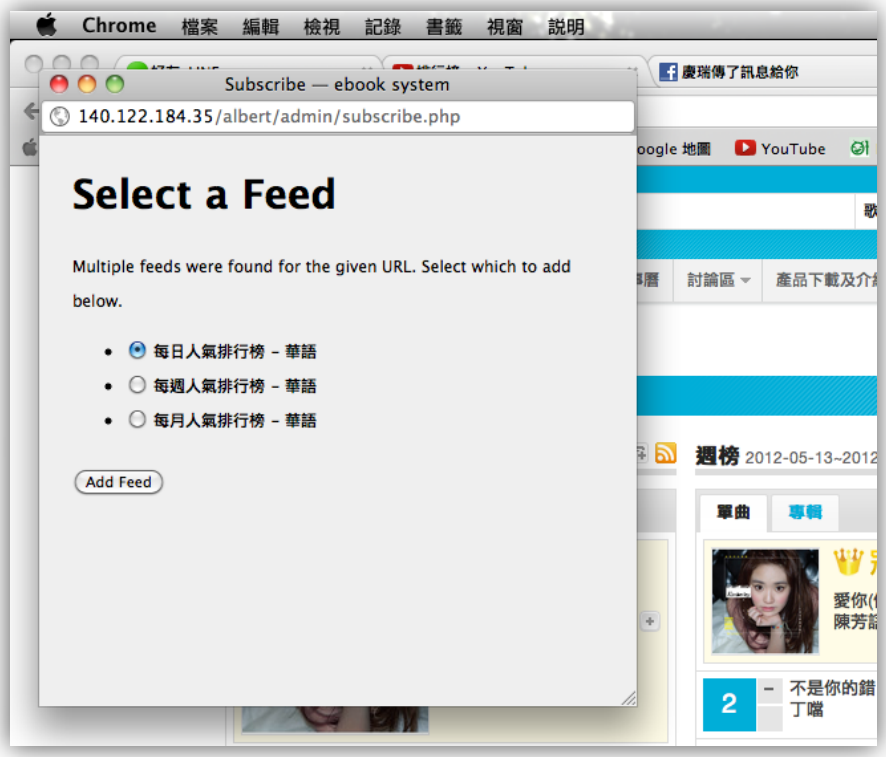
圖表 5-4 多種來源

如圖所示我們可以看到該網站有兩個RSS的來源。



圖表 5-5 多種來源 2

系統出現詢問畫面，接著下面也可以讓使用者命名來源所要的名字。



圖表 5-6 多種來源 3

如畫面呈現，系統便會抓取該網站有幾個來源，並且列出來給使用者選擇。

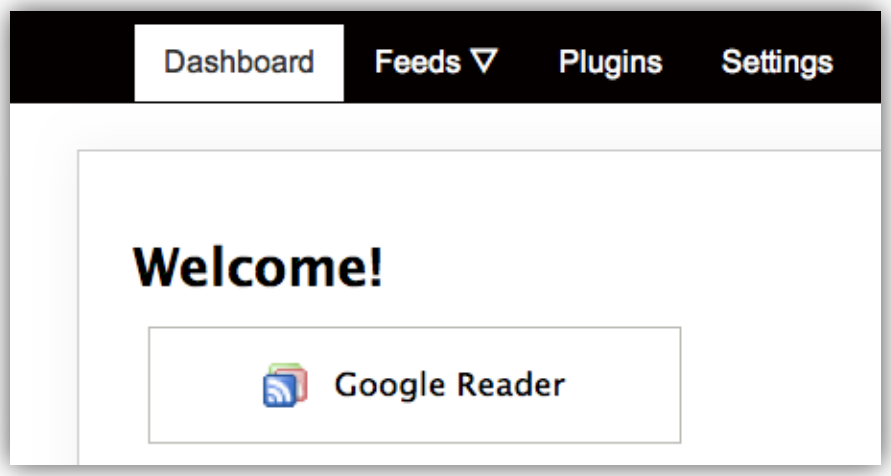
5.1.2 URL

URL 這是最簡單的也最原始的加入方式，就是將網址直接輸入在系統中，讓系統直接做加入的動作。

圖表 5-7 URL 畫面

5.1.3 Import

假設使用者以前曾經使用 Google Reader，那麼使用者就可以透過此系統的匯入功能，直接做匯入的動作。



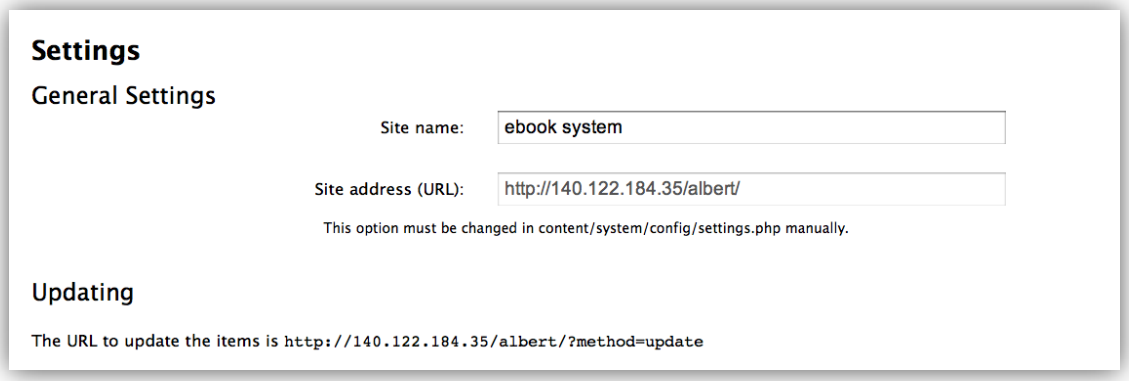
圖表 5-8 匯入畫面

5.2 各項設定

系統上有著許多的功能，而這些功能便有很多變數需要設定，如圖:

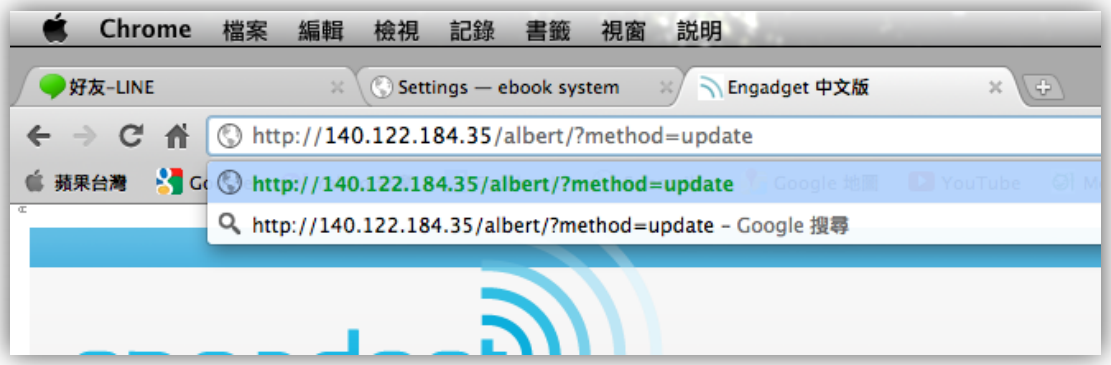
設定

第一項是系統的名子，再來是系統位置，最後則是更新網址，系統會告訴你的更新網址是多少，便可以讓使用者可以快速地在想更新的時候更新。



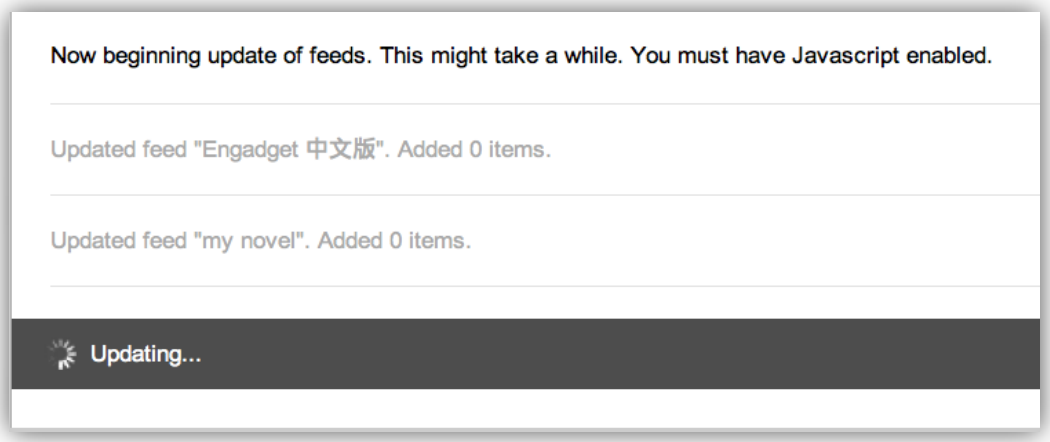
圖表 5-9 設定畫面

在這裡示範一下更新網址的使用畫面



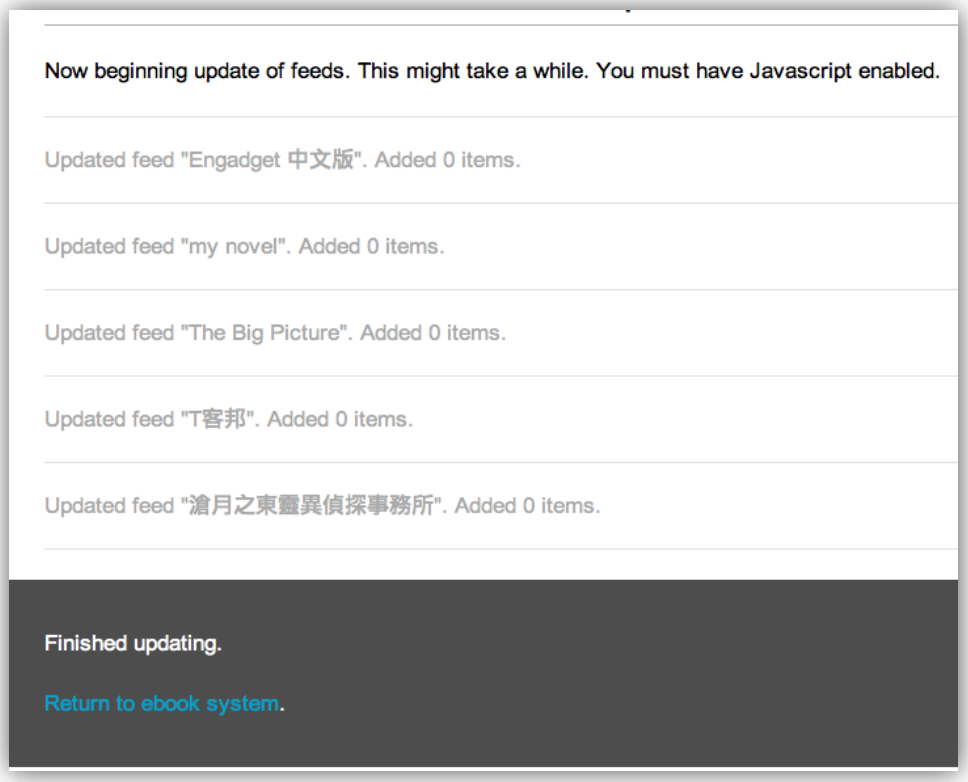
圖表 5-10 更新網址實作

這個更新網址，可以直接貼上，也可以做成書籤的形式直接點選。



圖表 5-11 更新網址實作 2

接著畫面會出現更新的畫面顯示有哪些來源需要更新。



圖表 5-12 更新網址實作 3

更新完成的畫面。

Wordpress

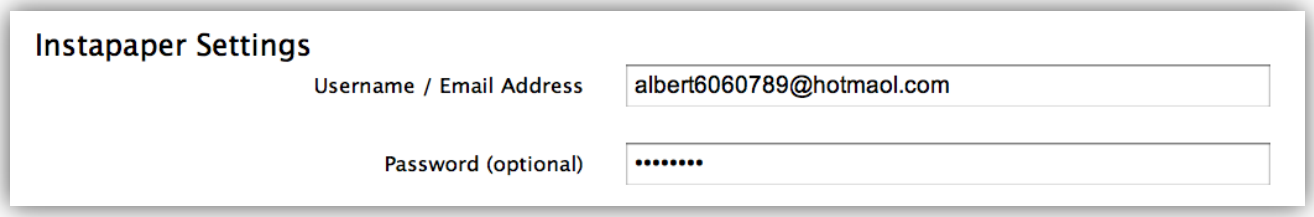
由於 wordpress 是一個部落格的系統架構，所以使用者會有它自己的網址以及帳號密碼，所以使用者必須在這裡設定，之後便可以讓系統直接做發布的動作。

圖表 5-13 Wordpress 設定

如圖，我們只要將網址列出即可以讓系統自動發布想要的文章。

Instapaper

而 Instapaper 也是需要帳戶及密碼來登入，所以使用者必須告知系統，自己的 instapaper 帳號及密碼，這樣系統即可以快速的將資料送到有裝載 instapaper 的設備。



圖表 5-14 instapaper 設定

5.3 系統畫面呈現

5.3.1 第一種瀏覽方式

一開始對於系統畫面呈現有一個很初步的呈現，我們也將這個畫面呈現出來，如圖所示。

圖表 5-15 第一種風格

一開始，是一行一行的將來源項目呈現出來，並且照排序方式排序出來。

圖表 5-16 第一種風格 2

點選項目之後，畫面會下拉，將內容呈現出來。

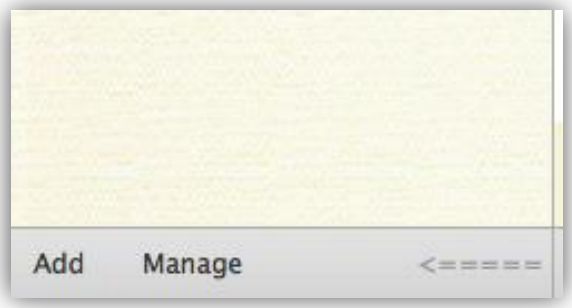
5.3.2 第二種瀏覽方式

由於上述第一種只是初步的樣式，所以經過 CSS、HTML5、Javascript 的加工之後，我們有著更棒的使用者介面，下面我們一一呈現。

最上方系統會有幾個圖示，分別是 Help、Refresh、Logout，而這些都是系統的基本功能，方便使用者使用這項系統。

來源分類

系統會將所收集到的來源做分類，若要做加入更改，只要點擊畫面中的 Add、manage 即可。



標題呈現

而中間的部分是標題的呈現，每個來源的文章，都會有他自己所謂的標題，

為了方便使用者知道他所在瀏覽的內容有哪寫，方便做個總攬，我們也列出了標題的欄位，而其中也有著 Reload 的功能讓使用者可以自己控制更新的時間。

標題的的列出，除了標題以外，還有來源網站，以及發布的時間。

數位內容畫面呈現

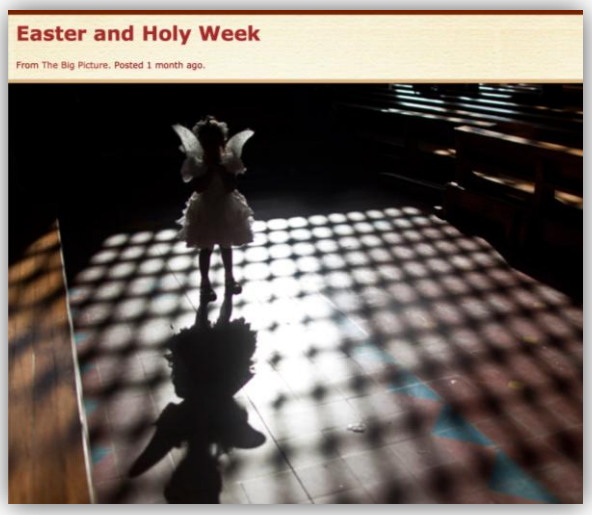
在這個部分，就是本系統最重點的部分，本體內容的呈現，這麼有著許多小細節，我們一一呈現。

字體、排版、倒影、邊框

圖表 5-17 第二種風格畫面

上面可以看到我們在標題用了粗體字，下面有著發佈時間、作者、發佈網站，店選之後會直接連到來源網站。

我們也可以很清楚看到，圖片有著倒影的效果。



圖表 5-18 第二種風格畫面 2

系統會調整圖片的大小，讓畫面變得非常美觀。

圖表 5-19 第二種風格畫面 3

圖片我們加上了邊框，加上倒影的功能，讓他更為美觀。

圖表 5-20 文繞圖

我們也更改了字體，行距，以及邊框的距離，還有讓畫面有著文繞圖的效果。

分享及稍後閱讀

Read later

這邊我們來示範稍後閱讀的流程

當我們點選了 Read later 之後，系統便會將這篇文章載到有裝載 instapaper 的裝置中。

圖表 5-21 instapaper 展示

假設我們對這篇文章有興趣，那們我們點選這篇下面的 Read Later。

圖表 5-22 instapaper 展示 2

如圖所示，他直接就傳至 iphone 的裝置中，而這是可以離線瀏覽的。

該篇文章在 Iphone 上看起來的子會是這樣。

Press it

接下來就是分享的部分，如果你看到一篇很喜歡的文章想要在上面做標記，或是加入自己的心得，那麼這個 press it 的功能就是讓使用者在分享的同時還可以寫心得到自己的部落格上，而發佈的文章是連帶圖片的。

圖表 5-23 press it 展示

假設看到一篇喜愛的文章，使用者點選下方的 press it 之後便會跳到發佈的畫面。

圖表 5-24 press it 展示 2

如圖，我們會跳到發佈的畫面，我們可以看到這篇分享的文章是連帶圖片的，完全沒有更改的讓使用者分享，也可以在上面加入自己的心得。

圖表 5-25 press it 展示 3

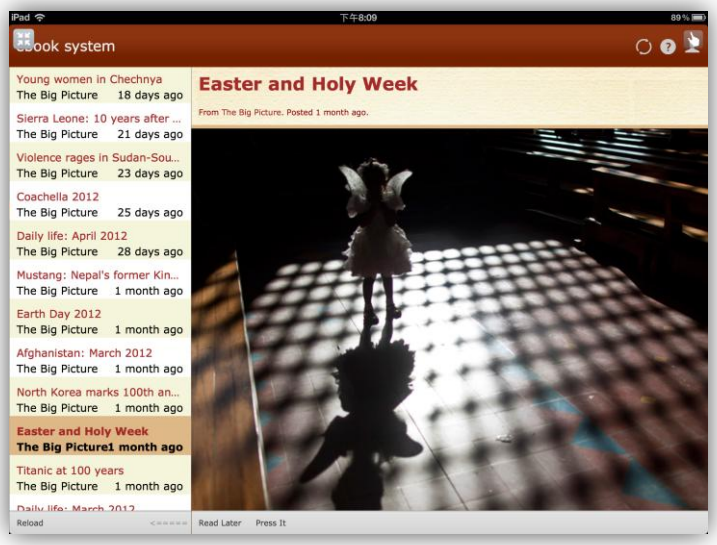
如圖所示，就會發佈到屬於自己的 Word press。

螢幕畫面拖動

由於 IPAD 的螢幕大小與電腦有所差異，為了讓使用者有更好的閱讀體驗，我們在不同設備上有著些微不同的呈現。

圖表 5-26 畫面拖動展示

這是在電腦上畫面的呈現。即使左邊的標題及來源列很寬，還是可以很清楚的看到右邊的內容。



圖表 5-27 移除功能列畫面

這是在 IPAD 上面呈現的畫面，使用者可以自行決定要不要拿掉這些來源列或是標題列，這樣子在 ipad 上面也可以有很好的閱讀體驗。

鍵盤快速鍵

系統在無障礙方面也做了一點些功能，如使用者可能不方便使用滑鼠，所以我們設計了鍵盤的快速鍵，一旦使用者進入了這個系統，只要按 J、K 兩個按鈕，就可以做上下頁的動作，而其實，我們真的只需要這兩個動作就可以了。

圖表 5-28 鍵盤快速鍵

Chapter 6 結論與未來展望

6.1 結論

本研究的概念，主要是來自於WEB2.0龐大資源的啟發，而希望使用者可以用很簡單的方式，來瀏覽這個龐大的資源，而我們用到電子書的概念，也整合了許多的技術，融合成一個新形態的電子書系統，而該系統的資料來源都是 WEB2.0，下面是本文的重點:

- 使用者只須要用很績單的方式就可以將數位資源加入系統。
- 使用者可以將『任何』有RSS或ATOM來源的網站，用本系統來觀賞。
- Web Application的應用上，使用者在使用本系統就像在使用應用程式一樣。
- 由於設計的本質是網站，所以在任何只要有瀏覽器的設備上，就可以使用本系統。
- 帳戶登入的方式，可以讓使用者在任何地方，任何設備上，使用自己個人化的資源。
- 分享功能可以讓使用者在閱讀文章之後，可以做好筆記或是心得，在發佈到自己的部落格。
- 稍後閱讀的功能，可以讓使用者在離線的時候，好好的來閱讀想要之後閱讀的文章。
- 無障礙的設計上，使用者只需要兩個按鍵，就可以完全的來使用本系統，並不會有操作上的困難。
- 使用者介面的加強，讓使用者在瀏覽數位資源的時候，不是像看網站一樣，而是有閱讀書籍的體驗，去除網站不必要的廣告、投票雜項等等。

作為一個閱讀體驗的系統，將來也可以用在教學上面，匹如說老師申請一個帳號加入自己作業或是講義的來源，這樣學生只需要登入之後，就可以觀看到講義內容及作業，而且是相當美觀的的。

本研究的重點還是希望使用者可以在混亂複雜的網路世界，用很簡單的方式來閱讀體驗，並且與其他系統不同的是，在任何設備，任何地方、任何時間，都可以很方便的使用本系統，加上離線規範的使用，

可以讓使用者在沒有網路的時候，觀看以前所過的資料，主旨就是簡單、方便、分享，來貫徹這個複雜的網路環境。

6.2 未來展望

- 本研究在系統的呈現上盡了許多的努力，而在呈現方面會是這個研究最大的重點，但若是可以在呈現畫面上能有翻閱書籍的動畫，會更為美觀，以及有更好的閱讀體驗，這是未來更可以做到的。
- 由於本研究的抓取資料的方式，是抓取網站的 RSS，一旦網站沒有 RSS 來源，那就無法使用本系統，希望未來本系統可以更加入不同於 RSS 的資料擷取方式，
- 由於本系統為了方便，使用了文檔當作資料庫，這樣做較為方便，但是資料容納個可能不夠，未來或許可以使用 SQL 這樣子的資料庫，便可以容納更多更多的資料。
- 本系統在排版上面是較為簡單的，之後也可加入封面，或是更多樣的排版方式，讓使用者有更好的閱讀體驗。

Chapter 7 參考文獻

[1]. M Pilgrim (2010) - Dive into HTML5

[2]. I Hickson - Editor's Draft(2010) - HTML5 Web Messaging

[3]. Yu Ping, Kostas Kontogiannis, Terence C. Lau(2004) - Transforming Legacy Web Applications to the MVC Architecture

[4]. World Wide Web Consortium - World Wide Web Consortium, czerwiec, (2010) - HTML5: A vocabulary and associated APIs for HTML and XHTML. W3C Working Draft

[5]. M David(2010) - HTML5: Designing Rich Internet Applications

[6]. Johan Harjono(2010) - Building smarter web applications with HTML5

[7]. S Pfeiffer(2010) - The Definitive Guide to HTML5 Video

[8]. C Murphy(2010) - Beginning Html5 and Css3: Next Generation Web Standards

[9]. JJ Garrett - Adaptive path(2005) - Ajax: A new approach to web applications

[10]. ARSS Board(2007) - RSS 2.0 Specification

[11]. Shelly Powers(2010) - Javascript Cookbook

[12]. Eric Meyer(2011) - Smashing CSS

[13]. Mark Pilgrim(2010) - HTML5 Up and Running

[14]. C Lynch (2011) - Wordpress-Preparing for your Wordpress site

[15]. S Macdonald - English in Aotearoa,(2010) - Using Wordpress: the Perspective of a Novice

[16]. K Rogers(2012) - Small Business of the Day: Convert to WordPress

[17]. Hill, J.; Cook, M.(2010) - Wordpress and a first-year seminar program

[18]. C Lynch(2011) - Wordpress-Setting up a Database

[19]. J Stamm - Brain(2011) - Ebook reader comparison for avid readers. Amazon Kindle or Nook vs iPad