Serendipity definiert eine klare Ansage an die PHP-Weblog-Engine

Serendipity – Underdog mit Biss

Garvin Hicking

Serendipity ist der unterschätzte Underdog der Blog-Szene, der sich vor den populären Konkurrenten wie *WordPress* und *MoveableType* aber nicht zu verstecken braucht. Viele bekannte Namen in der PHP-Szene [1] setzen das System bereits schon seit einer Weile ein. Dieser Artikel kann vielleicht auch Ihnen einen Anreiz zur Re-Dekoration der eigenen vier Webwände geben...

Die Entwicklung der Blog-Engine begann Anfang 2003, initiiert von Jannis Hermanns [2] und damals noch unter dem Namen jBlog. Schnell war in dem größer werdenden Entwicklerumfeld der treffendere Name *Serendipity* aus der Taufe gehoben und soll sowohl das Konzept des Blogsystems verdeutlichen, als auch einer Figur des Films *Dogma* Tribut zollen. *Serendipity* stand von da an für zielgerichtete Bedienung, maximale Flexibilität und ausbaufähige, solide Technik.

Wie die Blogszene selbst hat sich auch das Blogsystem sprunghaft entwickelt und von Version zu Version über die Jahre zahlreiche Innovationen erfahren. So wurde nach und nach die Plugin-Infrastruktur auf ein Event-basiertes Konzept erweitert, das Templatesystem Smarty hielt Einzug in den Kern und Mehrbenutzer- sowie CMS-Fähigkeiten wurden ausgebaut.

Die konstante Entwicklung des Systems gipfelt heute in einem stabilen, Featurekompletten Paket, das auch sicher demnächst noch Kaffee wird kochen können.

Features

Das Aufzählen aller Features würde deutlich den Rahmen sprengen, denn nicht umsonst wird *Serendipity* als beste Weiterentwicklung des geschnitten' Brot bezeichnet. Da man aber gerade zu Web-2.0-Zeiten mit Buzzwords sparsam umgehen sollte, hier kurz die wichtigsten Eigenschaften:

- Web-basierter Installations-Wizard für einfache und fortgeschrittene Konfiguration. Upgrades aller Versionen seit Erscheinen auf die aktuelle Version können ebenfalls mittels des integrierten Update-Managers durchgeführt werden.
- Übersichtliche, funktionale Bedienung mit (optionalen) WYSIWYG-Editoren.
- Integrierte Mediendatenbank (Bilder, PDFs, MP3, ...) mit Rechtemanagement, Keywords, Thumbnail-Erzeugung, dynamischer Verzeichnisnavigation, Layoutmanager zum Einfügen von Dateien.
- Gruppenbasiertes Rollenkonzept mit unbegrenzt vielen Autorengruppen und individuellen Rechten.
- Thread-basierte Kommentare, verschachtelbare Kategorien, Zuweisung

von Einträgen in mehrere Kategorien, Trackback, Pingback, XML-RPC, XHTML 1.1, CSS, RSS, Atom, ...
- Flexible Plugin-API für Seitenleisten- und Ereignisplugins, die effektiv jede Stelle des Systems ohne Eingriffe in den Core-Code erweitern können. Online Plugin-Repository (*Spartacus*) für 1-Click-Installation von mehr als 150 Plugins. Drag'n'Drop-Pluginverwaltung zur Änderung von Seitenleisten-Layouts.
- Dynamisches, Smarty-basiertes Template-Konzept. Strukturelle Änderungen aller Elemente des Frontends sind so mit einzelnen, aufeinander aufbauenden Templatedateien umsetzbar.
- Integration in bestehende Webseiten einfach möglich. *Shared Installation* ermöglicht Betrieb von beliebig vielen Blog-Instanzen mit nur einer Codebase.
- Umfangreiche, konfigurierbare Anti-Spam-Maßnahmen (Captcha, Moderation, Akismet, ...).
- Unterstützung von MySQL(i), PostgreSQL und SQLite.
- OpenSource. Und zwar richtig, nämlich durch Verwendung der BSD-Lizenz. Somit kann *Serendipity* auch in kommerziellen Programmen seinen Einsatz finden.
- Importieren aus zahlreichen anderen Blog-Systemen (WordPress, MoveableType, b2Evo, blogger, ...)

Aber ist das auch besser als ...

Bei allen Vorzügen des Systems sollte man dennoch neutral beim Vergleich mit anderen etablierten Systemen am Markt bleiben. Der Blogger an sich ist ja ein possierliches, individuelles Wesen und tut sich daher bei der Wahl seines Systems ähnlich schwer wie bei der Wahl der Religion.

Der weiten Verbreitung und monolithischen Popularitätskost von *WordPress* entgegensetzen kann *Serendipity* vor allem eine zielgerichtete, zentrale Community. Dort sind die Entwickler noch am Ohr der Benutzer und können den Code genau in die Richtung entwickeln, die von den Benutzern gewünscht wird. Ganz ohne visionäre Marketinggedanken [8]. Zudem kann *Serendipity* durch sein zentrales, leicht zugängliches Plugin-Archiv auf stabile und größtenteils offiziell unterstützte Plugins zurückgreifen. Bevor man 10 Pluginlösungen für ein Tagging-Plugin ausprobiert und mit keinem richtig glücklich wird, gibt es hier ein umfangreiches Tagging-Plugin, das alle Geschmäcker befriedigen kann. Und wenn nicht, dann kann es zukünftig leicht gemeinsam durch die Community verbessert werden.

Weiterhin ist *Serendipity* mit einer zweckmäßigeren, abstrakteren Codebase ausgestattet. PHPDoc-Funktionskommentare, aufgeräumte Verschachtelung und Benutzung von Objekt-Orientierung in der Plugin-API lassen das Entwicklerherz höher schlagen. Die Trennung von Layout und Code im Templatebereich macht Änderungen fühlbar angenehmer. Unüberschaubare PHP-Konstrukte, Plugin-Hacks und Modifikationen an Dateien des Sourcecodes sucht man hier vergebens.

Woran es *Serendipity* noch mangelt, ist eine ausführlichere Dokumentation und auch mehr Beteiligung der Nutzer an Plugin- und Template-Entwicklung. Für die Zukunft ist neben der OpenID-Integration (bereits in Arbeit) auch die stärkere Integration von PEAR-Channels, Unit-Tests, Performance-Tuning und Workflow-Integration gewünscht – wer sich also schon immer einmal an einem Open-Source-Projekt messen wollte und etwas beitragen möchte, darf sich in der einfachen Welt der *Serendipity*-Entwicklung gerne jederzeit beteiligen. [3]

Für den geneigten PHP-Entwickler ist *Serendipity* eine sehr gute Anlaufstelle, um sich mit Plugin-API basierten Systemen näher zu befassen und selbst Nutznießer der angebotenen Flexibilität und Erweiterbarkeit zu werden. Wer nutzt als Entwickler schon gerne ein System Out-of-the-Box, ohne sich selbst in der Konfiguration und Gestaltung des Blogs ausdrücken zu können? Genau dies war schon immer die Grundanforderung bei den Entwicklern – am Ende möchte man ein System individuell so nutzen, als hätte man es selber entwickelt. Und dieses Gefühl, mit einem soliden Framework in der Hinterhand, möchte *Serendipity* vermitteln.

Neuerungen von Serendipity

Wer das System schon länger kennt, aber vielleicht auch schon mit früheren Versionen beinahe wunschlos glücklich war, dürfte vielleicht an den Neuerungen der letzten Versionen interessiert sein.

Version 1.1 bot nach der Konsolidierungsphase und Logo-Umgestaltung der Vorgängerversion erstmals wieder umfangreiche neue Features. Beispielsweise die komplett überarbeitete Mediendatenbank, die nun mittels Smarty-Templating individuell anpassbar und mit automatischer Datei-Synchronisierung und AJAX-basierter Verzeichnisnavigation noch einfacher benutzbar wurde.

Der AJAX-Trend wurde natürlich auch bei *Serendipity* aufgegriffen, aber nur dezent an den Stellen eingesetzt, wo er wirklich Verbesserung bringen konnte. So zum Beispiel in der Verwaltung der Plugins, deren Reihenfolge nun per einfachem Drag and Drop änderbar wurde – und die Möglichkeit, dass Templates eine beliebige Anzahl von Seitenleisten definieren können sowie andere templatespezifische Optionen konfigurierbar machen.

Eine Verbesserung der Plugin-API bringt nun auch die Möglichkeit, die Ausführung gewisser Plugins für definierte Autorengruppen zu verhindern, und auch bestimmte Textformatierungen nur auf konfigurierte Artikel anwenden zu können. Auch RSS-Feeds können nun mittels Smarty-Templating individualisiert werden (ebenso wie Feeds nun auch logingeschützt per HTTP-Authentication abgerufen werden können). Die Integration von PEAR wurde verbessert, und vorhandene Komponenten auf dem Server können genutzt werden. Auch bei der SQL-Performance wurde einiges untersucht und verbessert.

Für Profis kam die Möglichkeit hinzu, alternative Templating-Prozessoren wie XSLT oder PHP zu nutzen. Damit fiel eine weitere Hürde von (unverstandenen) Smarty-Hassern, Spaß an *Serendipity* finden zu können.

Hervorzuheben ist, dass seit dem Bestehen des Projekts größter Wert auf Rückwärts-Kompatibilität gelegt wurde. Selbst Templates und Plugins der ersten Version sind in heutigen Versionen noch nutzbar, alle Updates der vergangenen Versionen waren immer nicht-destruktiv, notwendige Code-Änderungen wurden automatisch durchgeführt.

Serendipity 1.2

Nachdem das letzte Major Release im Dezember 2006 stattfand, wird Version

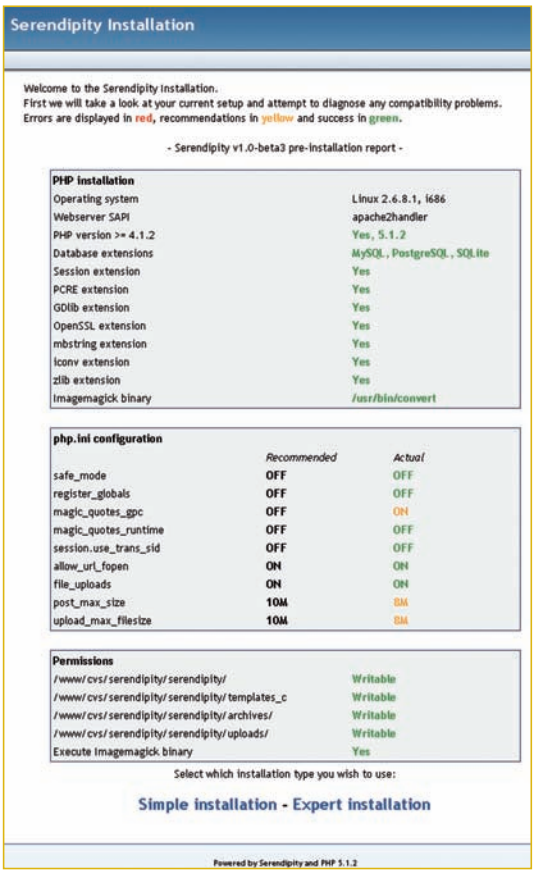


Abb. 1: Der Installationsmanager erstellt eine Analyse des Systems

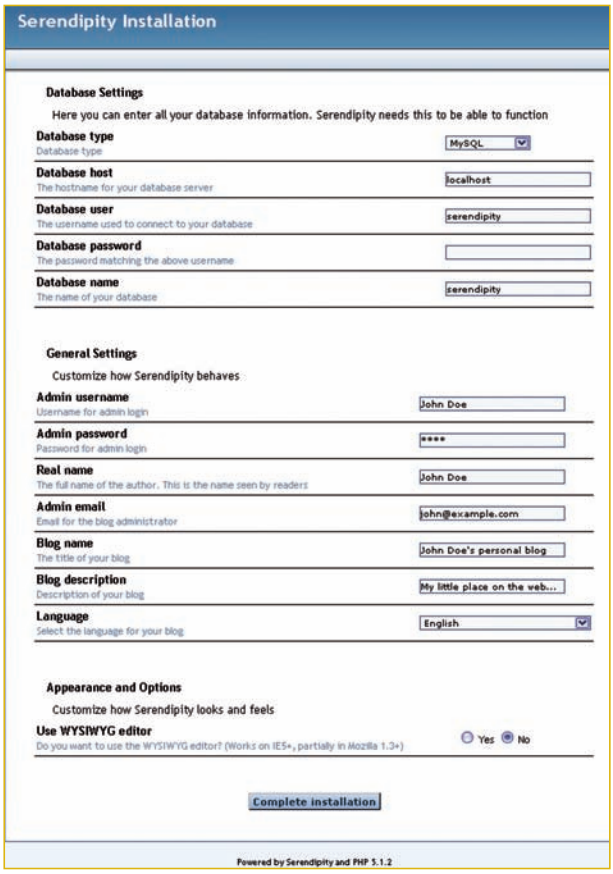


Abb. 2: Datenbankanbindung über eine übersichtliche Eingabemaske

1.2 auch nicht mehr lange auf sich warten lassen. Die derzeit aktuelle 1.2-Alpha3-Version bietet bereits jetzt folgende Neuerungen:

- Verbesserte Konnektivität mit dem Online-Repository Spartacus, verbesserte Erkennung von Firewall-Problemen.
- Technisch verbesserter Login-Workflow, so dass Plugins bereits vor dem Login in Zukunft Aktionen ausführen können (z.B. OpenID-Integration).
- Neue RSS-Varianten können nun auch direkt von Plugins ausgegeben werden (z.B. iTunes podcast feeds)
- Verbesserte Speichernutzung, mehrere Usability-Tweaks, feinere Anti-Spam Maßnahmen, Bugfixe für gewisse Permalink-Konstellationen

Wie man sieht, liegt das Augenmerk derzeit bei weiteren Änderungen für gesteigerte Flexibilität. Viele zusätzliche Features von *Serendipity* werden inzwischen vordergründig als Plugins entwickelt, daher lohnt sich ein Blick auf das Online-Plugin Repository [4].

Installation

Wer sich bis zu dieser Stelle noch zurückhalten konnte und *Serendipity* nicht direkt selbständig installiert hat, sollte spätestens jetzt tätig werden.

Benötigt wird ein Webserver (*Apache*, *lighttpd*, *IIS*) mit mindestens PHP 4.3.1 und einem Datenbanksystem der Wahl (*MySQL(i)* > 3.23, *SQLite*, *PostgreSQL*). In der Datenbank sollte bereits eine leere Datenbank mit beliebigem Namen existieren und ein Benutzeraccount für diese Datenbank eingerichtet worden sein. Bei Windows-Servern ist besonders darauf zu achten, dass der session.save_path für PHP-Sessions korrekt konfiguriert ist.

Nach dem Download des aktuellen Releases [5] (oder über diese Heft-CD) sollte die heruntergeladene Datei entpackt werden und in das Zielverzeichnis des Webservers unterhalb des Document-Roots gespeichert werden (z.B. */www/httpdocs/serendipity/*). Beim Hochladen und Entpacken sollte darauf geachtet werden, dass die Verzeichnisse *templates_c*, *uploads* und *archives* sowie das Serendipity-Stammverzeichnis für den Webserver schreibbar sind. Nach der Installation kann für das Stammverzeichnis das Schreibrecht wieder entzogen werden, da es nur für die Erstellung der Dateien *.htaccess* und *serendipity_config_local.inc.php* wichtig ist.

Der automatische Installationswizard kann dann über den Browser z.B. via *http://127.0.0.1/serendipity/* aufgerufen werden. Der Wizard zeigt erst eine Analyse des Systems und dessen Einstellungen sowie etwaiger Hinweise oder Fehlermeldungen (vergleiche Abb. 1). Am Ende der Seite kann man auf den Link *Einfache Installation* klicken, um fortzufahren.

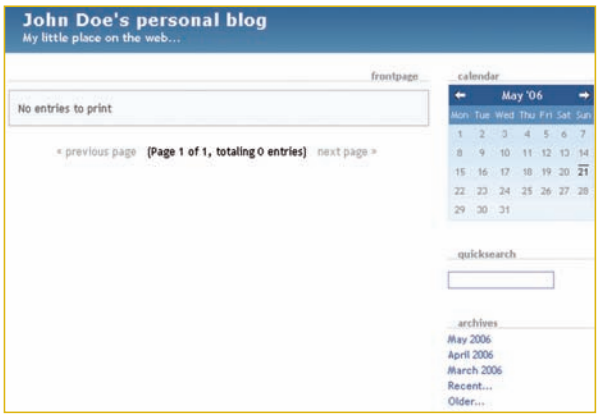
Im ersten Abschnitt der Folgeseite (siehe Abb. 2) müssen die Zugänge der erwähnten Datenbank eingetragen werden, im zweiten Abschnitt können Sie den Loginnamen und den Titel des Blogs festlegen. Nach der Installation können Sie sämtliche Werte auch über die Konfiguration nochmals ändern.

Nach einem Klick auf *Installation abschließen* werden Zugangsdaten eingerichtet, die Tabellen erstellt und Konfigurationsdateien gespeichert. Einen Klick später befindet man sich dann schon, wie in Abbildung 3 zu sehen, im eigenen Blog.

Sollte bei der Installation ein Fehler aufgetreten sein, hilft natürlich ein Blick in die Serendipity-Dokumentation [6] oder in das Forum [3]. Übliche Fehlerquellen sind nicht anwendbare *.htaccess* Regeln, fehlende Schreibrechte oder fehlende Zugriffsrechte des Datenbankbenutzers.

Ein Klick auf die Administrationsoberfläche führt schließlich zum eigentlichen Herzen von *Serendipity*.

Abb. 3: Der eigene Blog kurz nach der Installation



Serendipity benutzen

Um sämtliche Funktionen des Systems hier zu beschreiben, fehlen sowohl der Platz als auch die Notwendigkeit. *Serendipity* ist in fast all seinen Oberflächen auf Eindeutigkeit und schnelle Zugänglichkeit getrimmt, so dass sich nahezu alle Optionen selbst erklären. Und wenn dennoch etwas unklar ist, helfen die Dokumentation [6] und die Community [3] natürlich stets gerne weiter.

Serendipity erweitern

Damit auch die Techniker und potentiellen Liebhaber des Blogs noch etwas Nützliches erfahren, gibt es an dieser Stelle noch eine kleine Einführung in den Rohbau des Codes.

Die zentralen Funktionsbibliotheken befinden sich im Verzeichnis *include*. Alle Funktionen sind mittels PHPDoc-Kommentaren größtenteils ausreichend dokumentiert und liegen zur Benutzung durch Plugins o.ä. bereit.

Plugin API

In der *plugin_api.inc.php* werden alle Objekt-Klassen für die Ableitung eines eigenen Plugins definiert.

1. serendpity_plugin_api

Diese meist statisch aufgerufene Klasse enthält zentrale Verwaltungsmethoden

der Plugins, also z.B. das Laden eines Plugins, die Introspection oder das Auflisten aller installierten Plugins.

Die zentrale Stelle der Plugin_API für den Aufruf von Plugins ist die Methode *hook_event()*. Sie wird an zahlreichen Stellen des Serendipity-Codes wie folgt aufgerufen:

```
serendipity_plugin_api::hook_event('frontend_
                                    display', $entry);
```

Dieser Aufruf im Code führt dazu, dass die Plugin-API jedes einzelne Plugin befragt, ob es sich für den gerade ausgeführten Event zuständig fühlt. Ist dies der Fall, wird eine eigene Methode des Plugins aufgerufen (*event_hook()*), die mit den Event-Daten (referenzierte Variable) operieren kann.

Sollte einmal der Bedarf eines neuen Event-Hooks für neuartige Plugins bestehen, kann man ihn mit nur einer Zeile im Sourcecode hinzufügen. Eine Liste der bisher bestehenden Hooks findet sich in der Dokumentation [7].

2. serendipity_property_bag

Diese Klasse ist ein zentraler Datenhaltungscontainer für Plugins, mittels dessen das Plugin Metadaten zur späteren Auswertung angegeben kann. Dort werden z.B. eine Versionsnummer, der Pluginautor, genutzte Event-Hooks etc. spezifiziert.

3. serendipity_plugin

Die einfache Form eines Plugins ist ein sogenanntes Seitenleistenplugin. Dies kann nur Ausgaben im Frontend ausführen und hat sonst keinerlei „aktiven" Inhalt. Somit ist es geeignet für Flickr-Badges, Currently-Listening-Informationen oder eine Liste der aktuellsten Einträge.

4. serendipity_event

Eine auf *serendipity_plugin* aufbauende Klasse dient einem Ereignis-Plugin dazu, an den oben erwähnten Event-Hooks gewisse Operationen auszuführen. Die eigentliche Aktion eines Plugins wird mittels einer großen *event_hook()*-Methode dispatcht.

Um ein Plugin zu erstellen, muss man folgende Dinge tun:

- Ein Unterverzeichnis in der Namenskonvention *serendipity_(event|plugin)_meinname* im Verzeichnis *plugins* erstellen.2.
- In dieses Verzeichnis muss eine Datei gleichen Namens (mit *.php*-Endung) gespeichert werden.4.
- Der Inhalt der Datei definiert nur eine Plugin-Klasse (gleichen Namens wie des Verzeichnisses). Diese muss eine Ableitung der *serendipity_event* oder *serendipity_plugin*-Klasse sein und mindestens die notwendigen Methoden *introspect()* und *event_hook()* implementieren.

Viele Beispiele für solche Plugins gibt es bereits in der Standard-Installation von Serendipity, „Learning by Doing" ist hier sozusagen das praktischste Konzept. Da sich ein Plugin üblicherweise nur auf eine Datei (zzgl. optionaler Sprachincludes und Hilfsdateien) beläuft, ist es meist sehr einfach zu warten und schnell entwickelt.

Smarty-Funktionen

Ein Template besteht aus einer Ansammlung von Smarty **.tpl* Dateien und liegt in einem Unterverzeichnis des *templates*-Verzeichnisses. Wenn ein Template die Standard-Dateien des Systems benutzen will, lässt es die entsprechende Datei einfach weg und sie wird automatisch aus dem *default*-Verzeichnis geladen. So kann ein Template auch schon einzig aus einer *style.css* Datei bestehen, wenn am eigentlichen HTML-Layout nichts geändert werden soll.

Die zentrale Datei *index.tpl* enthält das Basislayout für jede Frontend-Seite des Blogs. Je nach angeforderter Seite (dispatcht durch die *index.php*) werden unterschiedliche Template-Dateien im Baukastenprinzip zusammengesteckt. Fordert man z.B. eine einzelne Eintragsseite an, sind folgende Templates involviert: *index.tpl*, *entries.tpl*, *comments.tpl* und *commentform.tpl*.

In jeder der Dateien kann man mit beliebigem Smarty-Markup eingreifen, oder auch nur HTML-Änderungen vornehmen. Für individuelle Funktionsanpassungen gibt es erweiterte Smarty-Funktionen, die *Serendipity* bereitstellt. Diese befinden sich in der Datei *include/functions_smarty.inc.php*. Die Funktion *serendipity_smarty_init()* ist dort für die Instanzierung des Smarty-Objektes zuständig und gibt einige Standard-Variablen vor. Innerhalb der Funktion kann man auch leicht erkennen, welche Möglichkeiten man als Entwickler hat um das *$serendipity['smarty']*-Objekt selbständig zu verändern.

Um in einem eigenenständigen Template (oder z.B. in der Seitenleiste des Blogs) Einträge darzustellen, kann man sich der Smarty-Funktion *{serendipity_fetchPrintEntries}* bedienen, die ihrerseits auch Smarty-Templates parst und zurückliefert.

Beliebige Ausgaben von Plugins lassen sich mittels der Funktionen *{serendipity_hookPlugin}* (für Ereignis-Plugins) oder *{serendipity_showPlugin}* (für Seitenleisten-Plugins) an beliebigen Stellen im Template positionieren.

Ausblick

Mit diesem System-Überblick ist die Stärke des Serendipity-Blogs hoffentlich erläutert und Ihre Kreativität angestachelt worden. Als Mit-Entwickler von *Serendipity* hoffe ich, dass Sie den Spaß am Bloggen durch Individualität und mit einer soliden Codebasis neu entdecken können.

Garvin Hicking ist als Web-Entwickler bei der Faktor-E GmbH in Bonn tätig und ist leitender Entwickler von Serendipity.

Links & Literatur

[1] (nach Haarfarbe und -länge sortiert)
Sara Golemon, http://blog.libssh2.org/
Ilia Alshanetsky, http://ilia.ws/
Tobias Schlitt, http://schlitt.info/
George Schlossnagle, www.schlossnagle.org/~george/blog/
Kristian Köhntopp, http://blog.koehntopp.de/
Sebastian Bergmann, www.sebastian-bergmann.de/
Rasmus Lerdorf, http://toys.lerdorf.com/

[2] Jannis Hermanns, www.jannis.to/

[3] Serendipity Forum, http://board.s9y.org/

[4] Spartacus - Plugin/Theme Online Repository, http://spartacus.s9y.org

[5] Serendipity Download, www.s9y.org/12.html

[6] Serendipity Dokumentation, www.s9y.org/

[7] Plugin API Dokumentation, www.s9y.org/116.html

[8] www.waxy.org/archive/2005/03/30/wordpres.shtml