Найкраща практика. Рецепти для видавничої RDF словників

W3C Working Group Note 28 серпня 2008
Ця версія:
http://www.w3.org/TR/2008/NOTE-swbp-vocab-pub-20080828/

Остання версія:
http://www.w3.org/TR/swbp-vocab-pub/

Попередня версія:
http://www.w3.org/TR/2008/WD-swbp-vocab-pub-20080123/

Редактори:
Дієго Берруета, фонд CTIC
Джон Фіппс, бібліотеки Корнельського університету
Попередній Редактори:
Алістер Міль, STFC Rutherford Appleton Laboratory
Томас Бейкер, Геттінген держави і університетська бібліотека
Ральф Свік, W3C

Анотація

Цей документ описує роботи найкращих рецептів для публікації словників або онтологій в Інтернеті (в RDF Schema і OWL). Особливості кожного рецепту докладно описані, так що дизайнери можуть вибрати рецепт, який найкращим чином відповідає їхнім потребам. Кожен рецепт вводить загальні принципи та приклад конфігурації для роботи з сервером Apache HTTP (який може бути адаптований до інших середовищ). Всі рецепти розроблені у відповідності з архітектурою Інтернету, який в даний час визначений, хоча

відповідні приклади конфігурації були збережені умисне простими.

Статус цього документа

Цей розділ описує статус цього документа на момент його публікації. Інші документи можуть заміняти цей документ. Зі списком поточних публікацій W3C і останної ревізії цієї технічної доповіді можна ознайомитися в W3C технічного індексу доповідей на http://www.w3.org/TR/.

Цей документ був підготовлений Semantic Web розгортання робочої групи (ДСО) на основі попередніх робіт Semantic Web Best Practices і розгортання робочої групи (SWBPD). Ця робота є частиною діяльності W3C Semantic Web.

Цей документ є Приміткою документування деяких найкращих практик. На момент публікації Semantic Web Deployment Working Group (SWD), не має планів щодо подальшої роботи за цим документом. Ця версія усуває ряд зауважень за попередньою версією. Це, однак, не вирішує відомі проблеми, пов'язані зі значення Q' в змісті переговорів, не надає рецептів для публікації словників допомогою RDFa [RDFa] і GRDDL [GRDDL], обидва з яких є визнаними корисними методи для деяких словників.

Коментарі вітаються і можуть бути спрямовані на public-swd-wg@w3.org, будь ласка, додайте текст "коментар" у рядку теми. Всі отримані повідомлення за цією адресою можна переглянути в публічному архіві. Робоча група може відповісти на зауваження, які одночасно доступні. Ми закликаємо спільноту для обговорення аспектів цієї записці в Semantic Web Interest Group список розсилки (громадські архів).

Цей документ був підготовлений групою, що діють в рамках 5 лютого 2004 W3C патентної політики. W3C підтримує публічний список будь-яких патентів, зроблених у зв'язку з результатами роботи групи; ця сторінка містить інструкції для розкриття патентів. Будь-яка особа, якій фактично було відомо про патенти в яких особа вважає, що містить основні претензії (и) повинна розкрити інформацію відповідно до розділу 6 W3C з патентної політиці.

Публікація в якості записки Робочої групи не означає схвалення членів W3C. Це проект документа і може бути змінений, замінені застарілі або інші документи в будь-який час. Недоцільно, щоб привести цей документ як одного, ніж робота.

Зміст

Введення

Цей документ призначений для творців із супроводженням словників у RDFS і OWL (лексика і онтології є взаємозамінними в контексті даної специфікації). Вона забезпечує крок за кроком інструкції для публікації словників в Інтернеті, даючи приклад конфігурації призначений для покриття найбільш поширених випадків. Для отримання додаткової інформації про RDFS і OWL см. [RDFS, RDFPrimer, OWLGuide, OWLFeatures].

Ця "кулінарна книга" дає "рецепти" з описом кроків, необхідних для публікації словника на веб-сервер, і настроїти веб-сервер з підтримкою семантичних веб-додатків. У розділі "Вибір рецепту" містяться рекомендації з вибору найбільш відповідного рецепту в залежності від ситуації і потреб. Після того як рецепт був обраний, читач може стежити за кроками, адаптуючих прикладів конкретної лексики.

Всі ці рецепти дають приклад конфігурації сервера Apache HTTP [APACHE20]. Для тих, хто ще не знайомий з конфігурацією Apache, додавання від конфігурації Apache містить короткий вступ до конфігурації Apache механізмів, що використовуються в прикладах і основні відомості про усунення неполадок.

Хоча наводяться приклади конфігурації для конкретного сервера Apache HTTP, загальні принципи застосовні і до не-Apache, а також середовищ. Робоча група пропонує внески додаткових прив'язок для не-сервера Apache. Консорціум W3C

представив вікі-сторінку для збору цих Non-Apache прив'язки і рекомендації.

Цей документ призначений насамперед для творців і супроводжуючих існуючих словників, які шукають керівництва щоб зрозуміти якого краще всього словника опубліковати в Інтернеті. Вона не призначена для надання докладного і вичерпаного керівництва по вибору відповідних імен URI для позначення нової лексики та її умови складової. Тим не менш, деякі основні відомості про технічні імена URI, включаючи деякі міркування, пов'язані з вибором імен URI для словника, наводиться у розділі, присвяченому іменам URI.

Всі рецепти були розроблені у відповідності з архітектурними принципами World Wide Web, які в даний час зазначені в документі, "Архітектура World Wide Web" [AWWW04]. Для того, щоб переконатися, що це насправді так, то набір мінімальних вимог описаний в кінці цього документу. Ці мінімальні вимоги повинні сформулювати основні вимоги семантичних веб-додатків. Всі рецепти за правильної програми повинні задовольняти мінімальним вимогам. Набір розширених вимог також наводиться. Розширені вимоги покликані формулювати подальші практичні потреби розробників семантичних веб-додатків, таких як надання документації про лексику на веб-сторінки в HTML. Рецепти 3, 4, 5 і 6, якщо вони правильно виконані, повинні задовольняти вимогам Extended.

Для того, щоб задовольнити вимоги Extended, рецепти 3, 4, 5 і 6 налаштування сервера для виконання зміст переговорів. Короткий опис цього процесу наводиться в розділі про зміст переговорів, разом з описом деяких варіантів впоратися з мінливістю розгорнутої поведінки клієнта.

Додаток А описує, як адаптувати шість рецептів представлена в основному тексті документа на спеціальному випадку

визначаються з використанням словників "Persistent URL", або URLs, які дозволено з допомогою PURL послуг, таких як http://purl.org/ [PURL].

Для стислості, обгрунтування кожної з рецептів не описані в цьому документі. Читачі, які бажають розглянути ширше повинні консультуватися URI / Ресурсний Відносини в "Архітектура World Wide Web" [AWWW04], фрагмент ідентифікаторів URI, HTTP [RFC3986] [RFC2616], W3C Interest Group Примітка: "Cool URI, для Semantic Web" [COOLURI], і резолюція W3C з технічної архітектурі групи по колу HTTP разименованіі (AKA "httpRange-14"). [HTTPRANGE14]

Нарешті, слід відзначити, що рецепти, описані в цій «кулінарній книзі» не єдиний спосіб опублікувати словниковий запас або онтології для використання семантичних веб-додатків. RDFa і його двоюрідний брат GRDDL може в найближчому майбутньому забезпечити ефективний метод публікації документів для використання як людини і машини. Але корисне обговорення RDFa і GRDDL виходить далеко за рамки цього документа.

---------- ----------

Вибір рецепта

Вибір рецепта в першу чергу залежить, який тип змісту, якого хочуть надати зі свого словника URI URI, а також класів і властивостей визначається ваш словниковий запас. URI простору імен описані більш докладно у Додатку B; в даному документі, вираз "словника URI" може тлумачитися як "словника імен URI".

Швидкий вибір таблиці

Конфігурація	хеш імен	слеш імен
Машина оброблюваних RDF	рецепт 1	рецепт 2
Машина оброблюваних RDF (використовуючи PURLs)	рецепт 1а	рецепт 2а
RDF і єдиного документа HTML	рецепт 3	рецепт 4
RDF і єдиного документа у форматі HTML (використовуючи PURLs)	рецепт 3а	рецепт 4а
RDF і множинні HTML документи		рецепт 5 або 6 рецептів
RDF і множинних документів HTML (використовуючи PURLs)		рецепт 5а

Проста конфігурація

Найпростіші рецепти налаштувати сервера для забезпечення тільки машинної обробки (PCO) утримання в словнику, URI.

Якщо ви використовуєте хеш імен див. рецепт 1 або, у випадку використання PURLs, рецепт 1а.

Якщо ви використовуєте слеш імен див. рецепт 2 або, у випадку використання PURLs, рецепт 2а.

Розширена конфігурація

Розширена настройка рецептів на ваш сервер для забезпечення як машинної обробки (PCO) і людського розуміння (HTML)

зміст (див. також розділ переговорів зміст нижче). Ці рецепти легко поширити для обслуговування додаткових типів вмісту.

Якщо ви використовуєте хеш імен і хочете забезпечити як RDF і HTML вмісту див. рецепт 3 або, у випадку використання PURLs, рецепт 3а.

Якщо ви використовуєте слеш імен і хочете забезпечити як RDF і HTML змістом, де вміст HTML міститься в одному документі, см. рецепт 4 або, у випадку використання PURLs, рецепт 4а.

Якщо ви використовуєте слеш імен і хочете забезпечити як RDF і HTML змістом, де вміст HTML подається як окремі документи з гіперпосиланнями для кожного класу або майна, з оглядом (наприклад, змісту або індекс) на словника URI см. рецепт 5 або, у випадку використання PURLs, рецепт 5а.

Якщо ви використовуєте слеш імен і хочете забезпечити як RDF і HTML змістом, де вміст HTML подається як окремі документи з гіперпосиланнями для кожного класу або майна, і де RDF подається як обмежений опис кожного класу або власність, див рецепт 6.

---------- ----------

Узгодження вмісту

Коли HTTP-клієнт намагається разименовать URI, він може вказати, який тип (або типи) змісту він хотів би отримувати у відповідь. Він робить це, в тому числі "Приймати: полем в заголовку повідомлення запиту, значення якого дає MIME типів, відповідних пріоритетним типу вмісту. Наприклад, HTTP-клієнт, який віддає перевагу RDF / XML зміст може включати в себе наступні поля в заголовку кожного запиту:

Приймати: застосування / RDF + xml

Аналогічно, HTTP, що клієнт віддає перевагу HTML вміст, такий як веб-оглядач, можуть містити в собі щось подібне до наступного поля в заголовку кожного запиту:

Приймати: застосування / XHTML + XML, текст / html

Це буде прийняте по принципу, що хороша практика HTTP клієнта повинна включати "Прийняти:" поле в заголовку запиту, явним зазначенням тих типів контенту, які можуть бути оброблені.

Коли сервер отримує запит, він може використати значення (и) "Прийняти: поле, щоб вибрати найбільш підходящий відповідь від наявних, намагаючись задовольнити уподобання клієнта, наскільки це можливо. Цей процес є прикладом змісту переговорів [AWWW04].

Рецепти 1 і 2 нижче, не налаштовують сервер для виконання будь-якого змісту переговорів. RDF / XML це єдиний доступний тип подання і надається незалежно від значення "Приймати:" заголовка отриманого від клієнта.

Рецепти 3, 4, 5 і 6 нижче дозволяють зробити настройку сервера для виконання змісту переговорів на основі значення "Приймати:" Поле заголовка, посланих клієнтом. Однак пропоновані приклади конфігурації не беруться за повну специфікацію HTTP у зв'язку зі змістом переговорів. Насправді, хоча приклади розміщення замовлення деяких клієнтських переваг у зміст переговорів, вони не займаються 'Q' метрик. Таким чином, HTTP клієнти, які використовують Q-значений в заголовку Приймати "може отримати несподівані результати.

Примітка редактора: Робоча група розглядає альтернативи, які повністю сумісні зі специфікацією HTTP, зберігаючи при

цьому рецепти якомога простішими. Це відбивається як ПИТАННЯ-58 в процесі випуску робочої групи. Запрошуються коментарі з цього питання, і повинні бути надіслані електронною поштою у ДЗГ робочої групи, починаючи з темою "Коментар: ПИТАННЯ-58".

Поведінка за замовчуванням
Якщо сервер налаштований на виконання змісту переговорів, а 'за замовчуванням' повинно бути вказано - сервер повинен бути в змозі визначити, яку відповідь повинен дати в тих випадках, коли:

Клієнт не включати "Приймати: полем в заголовку повідомлення запиту (наприклад, клієнт не вказати переваги) Значення 'Accept: "на місцях не відповідає жодному з доступних типів вмісту (наприклад, клієнт просить чогось іншого, ніж RDF / XML або HTML).
У рецептах 3, 4, 5 і 6 нижче, RDF / XML налаштований як відповідь за умовчуванням. Це зроблено для зведення до мінімуму впливу на розгорнутих семантичний веб-додатках, в даний час не відправити відповідне 'Accept: "Тема значення полів для утримання RDF. Майте на увазі, що якщо HTML налаштований в якості відповіді за замовчуванням, деяких існуючих Семантичний веб-додатків очікує отримання RDF зміст отримаєте HTML контент, і буде перерва.

Забезпечення 'Accept: "Тема
Розробники та супроводжуючі Семантичних веб-додатків, які очікують у процесі RDF зміст, і що в даний час не надають "Прийняти:" Поле заголовка HTTP в проханнях, повинна планують надати такий заголовок у майбутньому.

Відповідне значення "Приймати: поле виглядає наступним чином:

```
Приймати: застосування / RDF + XML, додаток / XML; 0.5
```

Це Q = 'Q = 0.5' значення в наведеному вище прикладі показує відносну вартість якості зазначеною статтею 14 протокол HTTP 1.1 [RFC2616]. Значення Q замовчуванням ', якщо не вказано ні 1,0. Цей приклад може бути прочитаний як "Я вважаю за краще додаток '/ RDF + XML, надаючи йому за замовчуванням відносного значення якості 1.0, але буде як і раніше, приймаємо заяви / XML, хоча я вважаю, що це буде тільки 50% від вартості застосування / RDF + XML ".

Спеціальна поведінка за замовчуванням для Internet Explorer
На відміну від більшості інших браузерів, Internet Explorer посилає 'Accept: "Заголовки містять Catch-All * / * без посередницьких' Q 'значення. На відміну від цього, Firefox і Safari приймати * / *, Q = 0.5, Opera і приймає * / *, Q = 0,1. Створення стандартної поведінки, що приймає це до уваги, вимагає включення в нього переписаних умов, які грунтуються на значенні 'User-Agent: "Поле заголовка:

```
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
```

Крім того, якщо ви також хочете зберегти 'інтерактивними' в URI, наприклад, ви повинні встановити HTML як відповідь за умовчанням (див. коментарі в прикладах для отримання інструкцій по, як це зробити).

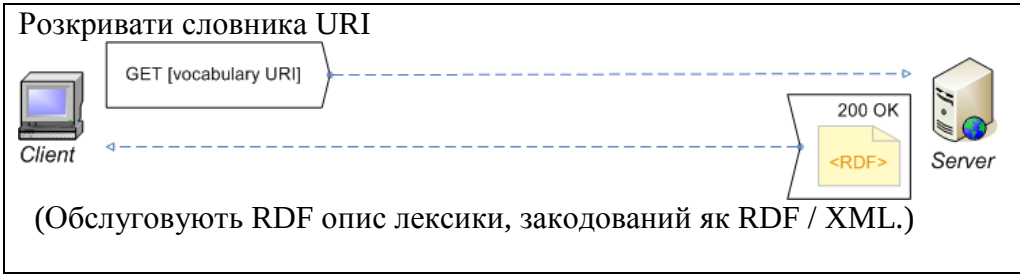
Конфігурація сервера тестування
Ефективно тестування результатів зміст переговорів може стати досить складним для деяких рецептів. З метою полегшення тестування про результати переговорів з утримання одного URI контент-послуги тестування переговорів була створена.

Ця послуга буде просити умови URI від сервера, запустити тести, спеціально призначені для випробувань у відповідь сервера, стосовно специфікацій рецепт, і відображати пройшов / не пройшов доповідь по кожній серії випробувань, а також докладні пояснення своїх висновків .

Вихідний код для тестування послуги також доступний.

Рецепт 1. Мінімальна конфігурація для "хеш імен '
Перейти до прямої: Приклад конфігурації | Тестування конфігурації

Цей рецепт дає **приклад найпростішої конфігурації** для словника, який використовує хеш імен. Рецепт Налаштування сервера для забезпечення машинної обробки (PCO) утримання в словнику, URI, задовольнивши тим самим мінімальним вимогам. Це ілюструється на наступній діаграмі:



Приклад конфігурації
Для словника ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example1defining

класи ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example1 # ClassA
http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example1 # ClassBand

властивості ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example1 # Propa
http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example1 # propB

Крок 1
Створіть файл з ім'ям example1.rdf, яка містить повну RDF / XML сериализация словника. M.I. всі ресурси, певні лексиці описаних в цьому файлі.

Крок 2
Скопіюйте файл example1.rdf / apachedocumentroot / приклади / каталогів на сервері.

Крок 3
Додати такі директиви. Htaccess файл у каталозі / apachedocumentroot / приклади / каталогів на сервері:

```
# Директиви для забезпечення RDF файлах *. служив відповідний тип вмісту,
# Якщо немає в основний конфігурації Apache
AddType застосування / RDF + XML. RDF

# Rewrite установка двигуна
RewriteEngine On
RewriteBase / Приклади

# Правила перезапису служити RDF / XML контент в словнику, URI
RewriteRule ^ example1 $ example1.rdf
```

(NB Якщо файл. Htaccess файл не існує, створіть його).

Перевірка конфігурації

Якщо ця конфігурація працює, вона повинна підтримувати такі взаємодії:

Розкривати словника URI

Тестове повідомлення (замінити правильний шлях і бере свій словниковий запас, URI):

```
GET / examples/example1 HTTP/1.1
Ведучий: example.com
```

Response заголовок повинен містити наступні поля:

```
HTTP/1.x 200 OK
Content-Type: Application / RDF + XML
```

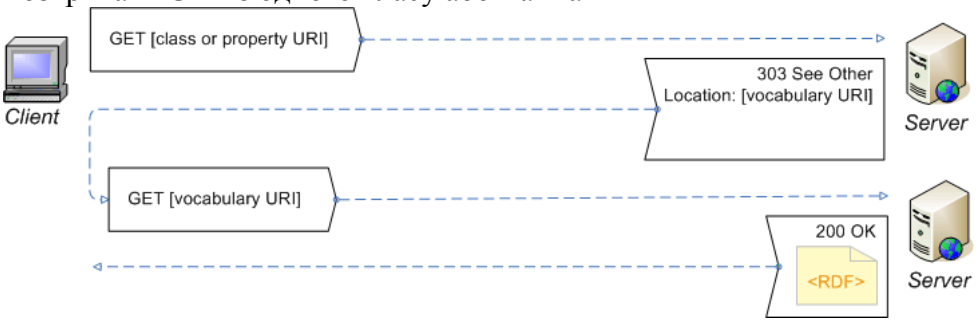
Рецепт 2. Мінімальна конфігурація для "слеш імен '
Перейти до прямої: Приклад конфігурації | Тестування конфігурації

Цей рецепт дає приклад найпростішої конфігурації для лексики, яка використовується як коса риска імен. Рецепт Налаштування сервера для забезпечення машинної обробки (PCO) утримання в словнику, URI, і перенаправляти клієнта до лексики URI з класу і майна URI, задовольнивши тим самим мінімальним вимогам. Про це свідчать наступні діаграми:

Розкривання словника URI

(Обслуговують RDF опису лексики, закодований як RDF / XML.)

Розкривати URI з одного класу або майна



(Перенаправлення клієнта на словника URI.)

Приклад конфігурації
Для словника ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example2/defining
класи ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example2/ClassA

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example2/ClassBand
властивості ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example2/propA

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example2/propB

Крок 1
Створіть файл з ім'ям example2.rdf, яка містить повну RDF / XML сериализация словника. M.I. всі ресурси, певні лексиці описаних в цьому файлі.

Крок 2
Скопіюйте файл example2.rdf / apachedocumentroot / приклади / каталогів на сервері.

Крок 3
Додати такі директиви. Htaccess файл у каталозі / apachedocumentroot / приклади / каталогів на сервері:

```
# Вимкнути MultiViews
Options-MultiViews

# Директиви для забезпечення RDF файлах *. служив відповідний тип вмісту,
# Якщо немає в основний конфігурації Apache
AddType застосування / RDF + XML. RDF

# Rewrite установка двигуна
RewriteEngine On
RewriteBase / Приклади

# Правила перезапису перенаправляти 303 з будь-якого класу або опору URI
RewriteRule ^ example2 /. + Example2 / [R = 303]

# Правила перезапису служити RDF / XML контент в словнику, URI
RewriteRule ^ example2 / $ example2.rdf
```

(NB Якщо файл. Htaccess файл не існує, створіть його).

MultiViews каталозі опцію 'повинна бути відключена для такої конфігурації працювати, і каталог / apachedocumentroot/examples/example2 / не повинен фактично існує на файловій системі сервера.

Перевірка конфігурації
Якщо ця конфігурація працює, вона повинна підтримувати такі взаємодії:

Розкривати словника URI
Тестове повідомлення (замінити правильний шлях і бере свій словниковий запас, URI):

GET / examples/example2 / HTTP/1.1
Ведучий: example.comResponse заголовок повинен містити наступні поля:

HTTP/1.x 200 OK
Content-Type: Application / RDF + xml
Розкривати URI з одного класу або майна
Тестове повідомлення (замінити правильний шлях і приймає для вашого класу або власності, URI):

GET / examples/example2/ClassA HTTP/1.1
Ведучий: example.comResponse заголовок повинен містити наступні поля, з вашого словника URI як значення поля "Розташування":

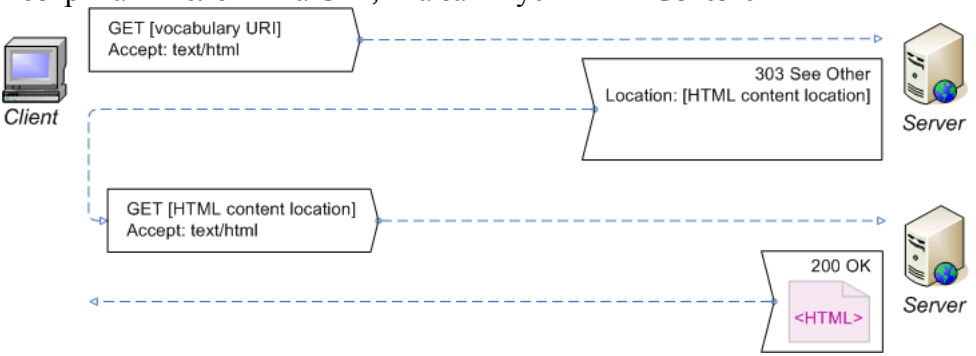
HTTP/1.x 303 Див інше
Розміщення: http://example.com/examples/example2/

Рецепт 3. Розширена конфігурація для "хеш імен '

Перейти до прямої: Приклад конфігурації | Тестування конфігурації | Нотатки

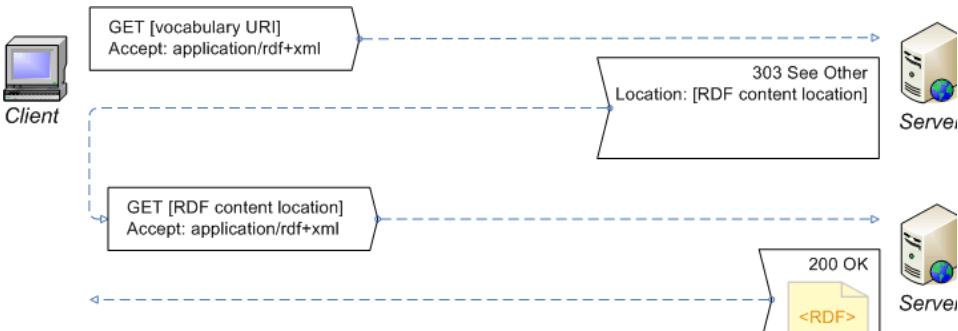
Цей рецепт дає приклад розширеної конфігурації для словника з хеш імен. Рецепт Налаштування сервера для забезпечення або людського розуміння (HTML) або машинної обробки (PCO) утримання в словнику, URI в залежності від того, що просив, задовольнивши тим самим вимоги Extended. Про це свідчать наступні діаграми:

Розкривання словника URI, яка запитує HTML Content



(Перенаправлення клієнта на поточному HTML документацію для словника.)

Розкривання словника URI з проханням RDF зміст



(Перенаправлення клієнта на поточному описі RDF лексики.)

Приклад конфігурації
Для словника ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example3defining
класи ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example3 # ClassA
http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example3 # ClassBand
властивості ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example3 # Propa
http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example3 # propB

Крок 1
Створіть файл з ім'ям 2005-10-31.rdf, який містить повну RDF / XML сериализацію лексики, за станом на 2005-10-31 (або що поточна дата). M.I. Всі ресурси визначаються лексикою описаних в цьому файлі, і цей файл є 'моментальним знімком' або 'Версією' лексики.

Крок 2
Створіть файл з ім'ям 2005-10-31.html, який містить HTML зміст документації по всіх класах і властивостях, які визначаються лексичним станом на 2005-10-31 (або що поточна дата). Цей документ може включати розділи по кожному з класів / Властивості документів, у кожному розділі очолюють якір HTML, ім'я якого збігається з ідентифікатором фрагмента з документованих класів чи майна.

Крок 3
Копія 2005-10-31.rdf і 2005-10-31.html в директорію / apachedocumentroot/examples/example3-content / на сервер.

Крок 4
Додати такі директиви. Htaccess файл у каталозі / apachedocumentroot / Приклади / Каталог:

```
# Вимкнути MultiViews
Options-MultiViews

# Директиви для забезпечення RDF файлах *. служить відповідний тип вмісту,
# Якщо немає в основній конфігурації Apache
AddType застосування / RDF + XML. RDF

# Rewrite установка двигуна
RewriteEngine On
RewriteBase / Приклади

# Правила перезапису служить HTML контент із словника URI бажанням
RewriteCond% () HTTP_ACCEPT! Застосування / RDF \ + XML .* (Текст / HTML | Заявка / XHTML \ + XML)
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ + XML [OR]
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
Приклад 3 RewriteRule ^ $ example3-content/2005-10-31.html [R = 303]
# Правила перезапису служить RDF / XML контент в словнику, URI бажанням
RewriteCond% () HTTP_ACCEPT застосування / RDF \ + XML
Приклад 3 RewriteRule ^ $ example3-content/2005-10-31.rdf [R = 303]
```

```
# Вибрати за замовчуванням відповідь
# -------------------------

# Правила перезапису служить RDF / XML контент в
словнику, URI за замовчуванням
Приклад 3 RewriteRule ^ $ example3-content/2005-10-31.rdf [R =
303]

# Правила перезапису служить HTML контент із словника URI
за замовчуванням (для інвалідів)
# (Щоб включити цю операцію, розкоментуйте переписати
правила нижче, і коментар
# З переписати правила безпосередньо вище)
# RewriteRule ^ Приклад 3 $ example3-content/2005-10-31.html
[R = 303]
```

(NB Якщо. Htaccess файл не існує, створіть його).

Перевірка конфігурації

Якщо ця конфігурація працює, вона повинна підтримувати такі взаємодії:

Розкривати словник URI, який запитує HTML Content
Тестове повідомлення (замінює правильний шлях і бере свій словниковий запас, URI):

GET / examples/example3 HTTP/1.1
Host: example.com
Accept: Текст / htmlResponse заголовок повинен містити наступні поля, з вашого HTML Content місці в якості значення поля "Розташування":

HTTP/1.x 303 Див інше

Розміщення: http://example.com/examples/example3-content/2005-10-31.htmlDereference словника URI з проханням RDF зміст
Тестове повідомлення (замінює правильний шлях і бере свій словниковий запас, URI):

GET / examples/example3 HTTP/1.1
Host: example.com
Приймати: застосування / RDF + xmlResponse заголовок повинен містити наступні поля з поточним змістом RDF місця в якості значення поля "Розташування":

HTTP/1.x 303 Див інше
Розміщення: http://example.com/examples/example3-content/2005-10-31.rdfDereference словника URI, за замовчуванням справу
Тестове повідомлення (замінює правильний шлях і бере свій словниковий запас, URI):

GET / examples/example3 HTTP/1.1
Ведучий: example.comResponse заголовок повинен містити наступні поля з поточним змістом RDF місця в якості значення поля "Розташування":

HTTP/1.x 303 Див інше
Розміщення: http://example.com/examples/example3-content/2005-10-31.rdfNotes
Цей приклад використовує дату модифікації версії словника для створення імен файлів. Також було б можливо використовувати номери версій (наприклад, '1 .01 ') замість дат для цієї мети, або яке-небудь конвенції, яка робить можливим проводити різницю між версією словника, а також допомагає відслідковувати історію версій.

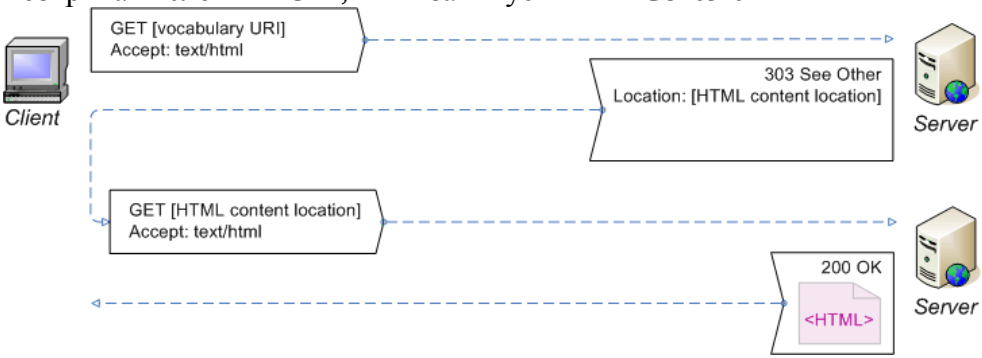
Див також розділ про зміст переговорів.

Рецепт 4. Розширена конфігурація для "слеш імен", використовуючи єдиний документ HTML

Перейти до прямої: Приклад конфігурації | Тестування конфігурації | Нотатки

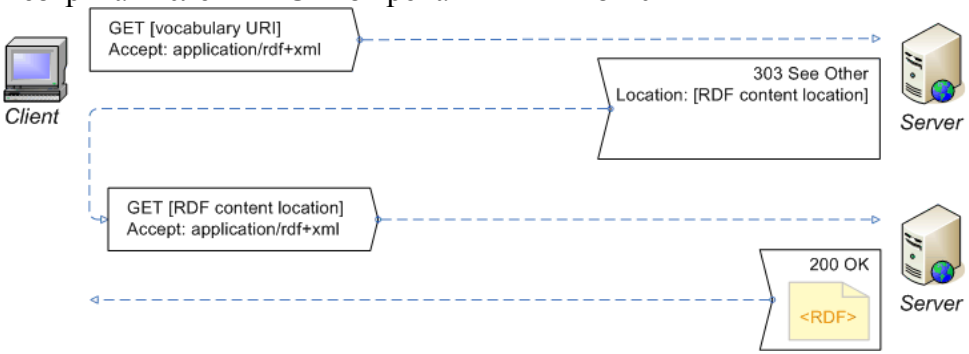
Цей рецепт дає приклад розширеної конфігурації для словника з косою рискою імен. Рецепт Налаштування сервера для забезпечення або людського розуміння (HTML) або машинної обробки (PCO) утримання в словнику, URI, в залежності від того, що вимагалося, і перенаправляти клієнта з класу і власності URI, у відповідних місцях утримання, знову залежно від того, що вимагалося, задовольнивши тим самим вимоги Extended. Документація HTML подається у вигляді окремого файлу. Про таку поведінку свідчать наступні діаграми:

Розкривати словник URI, який запитує HTML Content



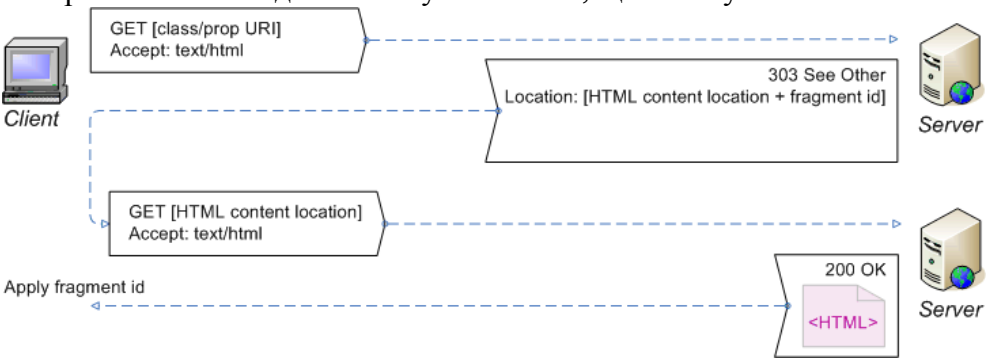
(За рецептом 3, перенаправити клієнта на поточному HTML документацію для словника.)

Розкривати словник URI з проханням RDF зміст



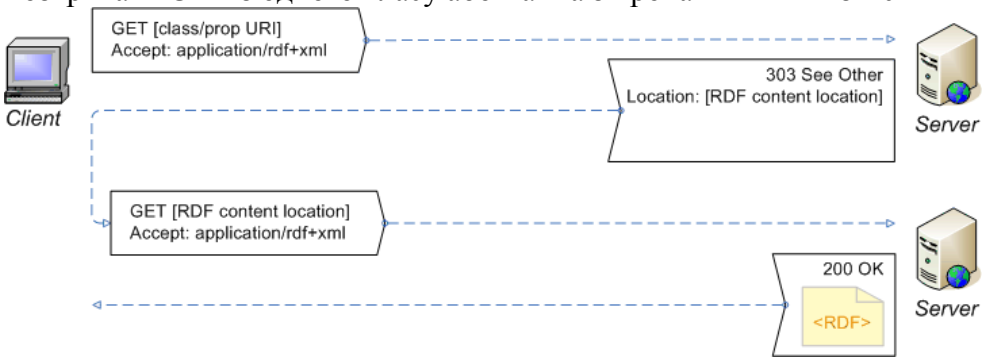
(Рецепт 3, перенаправляє клієнта на поточному описі RDF лексики.)

Розкривати URI з одного класу або майна, що запитує HTML Content



(Перенаправлення клієнта на поточному фрагменті HTML документацію для словника, що мають відношення до класу або майна).

Розкривати URI з одного класу або майна з проханням RDF зміст



(Перенаправлення клієнта на поточному описі RDF лексики.)

Приклад конфігурації
Для словника ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example4/defining
класи ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example4/ClassA

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example4/ClassBand
властивості ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example4/propA

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example4/propB

Крок 1
Створіть файл з ім'ям 2005-10-31.rdf, який містить повну RDF / XML сериалізацію лексики, за станом на 2005-10-31 (або що

поточна дата). M.I. Всі ресурси визначаються лексикою описаних в цьому файлі, і цей файл є 'моментальним знімком' або 'Версією' лексики.

Крок 2
Створіть файл з ім'ям 2005-10-31.html, яка містить HTML зміст документації по всіх класах і властивості лексики визначаються станом на 2005-10-31 (або що поточна дата). Цей документ може включати розділи по кожному з класів / Властивості документами, у кожному розділі очолює якір HTML, ім'я якого збігається з ідентифікатором фрагмента з документованих клас чи майна.

Крок 3
Копія 2005-10-31.rdf і 2005-10-31.html в директорію / apachedocumentroot/examples/example4-content / на сервер.

Крок 4
Додати такі директиви. Htaccess файл у каталозі / apachedocumentroot / Приклади / Каталог:

```
# Вимкнути MultiViews
Options-MultiViews

# Директиви для забезпечення RDF файлах *. служив відповідний тип вмісту,
# Якщо немає в основний конфігурації Apache
AddType застосування / RDF + XML. RDF

# Rewrite установка двигуна
RewriteEngine On
RewriteBase / Приклади

# Правила перезапису служити HTML контент із словника URI бажанням
```

```
RewriteCond% () HTTP_ACCEPT! Застосування / RDF \ +
XML .* (Текст / HTML | Заявка / XHTML \ + XML)
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ +
XML [OR]
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
RewriteRule ^ example4 / $ example4-content/2005-10-31.html
[R = 303]

# Правила перезапису служити направлено HTML вміст
класу / URI, опора
RewriteCond% () HTTP_ACCEPT! Застосування / RDF \ +
XML .* (Текст / HTML | Заявка / XHTML \ + XML)
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ +
XML [OR]
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
RewriteRule ^ example4 /(.+) example4-content/2005-10-
31.html # $ 1 [R = 303, NE]

# Правила перезапису служити RDF / XML зміст, якщо
просив
RewriteCond% () HTTP_ACCEPT застосування / RDF \ +
XML
RewriteRule ^ example4 / example4-content/2005-10-31.rdf [R
= 303]

# Вибрати за замовчуванням відповідь
# -------------------------

# Правила перезапису служити RDF / XML стандартний
вміст
RewriteRule ^ example4 / example4-content/2005-10-31.rdf [R
= 303]

# Правила перезапису служить HTML стандартний вміст
```

(для інвалідів)
(Щоб включити цю опцію, розкоментуйте два правила перезапису нижче,
I закоментуйте переписані правила безпосередньо вище)
RewriteRule ^ example4 / $ example4-content/2005-10-31.html [R = 303]
RewriteRule ^ example4 /(.+) example4-content/2005-10-31.html # $ 1 [R = 303, NE]

(NB Якщо файл Htaccess файл не існує, створіть його).

Для цієї конфігурації для роботи в MultiViews каталозі опцію 'повинна бути відключена, і каталог / apachedocumentroot/examples/example4 / не повинен фактично існувати на файловій системі сервера.

Перевірка конфігурації

Якщо ця конфігурація працює, вона повинна підтримувати такі взаємодії:

Розкривати словник URI, який запитує HTML Content
Тестове повідомлення (замінює правильний шлях і бере свій словниковий запас, URI):

GET / examples/example4 / HTTP/1.1
Host: example.com
Accept: Текст / htmlResponse заголовок повинен містити наступні поля, з вашого HTML Content місці в якості значення поля "Розташування":

HTTP/1.x 303 Див інше
Розміщення: http://example.com/examples/example4-content/2005-10-31.htmlDereference словника URI з проханням RDF зміст
Тестове повідомлення (замінює правильний шлях і бере свій словниковий запас, URI):

GET / examples/example4 / HTTP/1.1
Host: example.com
Приймати: застосування / RDF + xmlResponse заголовок повинен містити наступні поля, на ваш RDF змісту місці в якості значення поля "Розташування":

HTTP/1.x 303 Див інше
Розміщення: http://example.com/examples/example4-content/2005-10-31.rdfDereference словника URI, за замовчуванням справу
Тестове повідомлення (замінює правильний шлях і бере свій словниковий запас, URI):

GET / examples/example4 / HTTP/1.1
Ведучий: example.comResponse заголовок повинен містити наступні поля, на ваш RDF змісту місці в якості значення поля "Розташування":

HTTP/1.x 303 Див інше
Розміщення: http://example.com/examples/example4-content/2005-10-31.rdfDereference Урі з одного класу або майна, що запитує HTML Content
Тестове повідомлення (замінює правильний шлях і приймає для вашого класу або майна, URI):

GET / examples/example4/ClassA HTTP/1.1
Host: example.com
Accept: Текст / htmlResponse заголовок повинен містити наступні поля, з вашого HTML місце розташування змісту (плюс відповідний ідентифікатор фрагмента) в якості значення поля "Розташування":

HTTP/1.x 303 Див інше

Розміщення: http://example.com/examples/example4-content/2005-10-31.html # ClassADereference URI з одного класу або майна з проханням RDF зміст
Тестове повідомлення (замінити правильний шлях і приймає для вашого класу або майна, URI):

GET / examples/example4/ClassA HTTP/1.1
Host: example.com
Приймати: застосування / RDF + xmlResponse заголовок повинен містити наступні поля, на ваш RDF змісту місці в якості значення поля "Розташування":

HTTP/1.x 303 Див інше
Розміщення: http://example.com/examples/example4-content/2005-10-31.rdfDereference Урі з одного класу або майна, за замовчуванням справу
Тестове повідомлення (замінити правильний шлях і приймає для вашого класу або майна, URI):

GET / examples/example4/ClassA HTTP/1.1
Ведучий: example.comResponse заголовок повинен містити наступні поля, на ваш RDF змісту місці в якості значення поля "Розташування":

HTTP/1.x 303 Див інше
Розміщення: http://example.com/examples/example4-content/2005-10-31.rdf

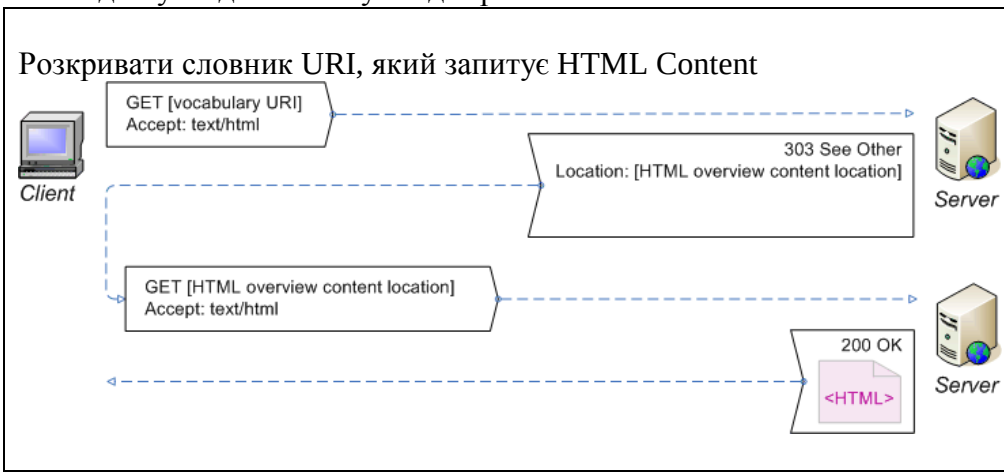
Нотатки

Як рецептом 3, у цьому прикладі використовує дату модифікації версії словника для створення імен файлів. Також було б можливо використовувати номери версій (наприклад, '1 .01 ') замість дат для цієї мети, або яке-небудь конвенції, яка робить можливим проводити різницю між версією словника, а також допомагає відслідковувати історію версій.
Див також розділ про зміст переговорів.

Рецепт 5. Розширена конфігурація для "слеш імен", використовуючи кілька документів HTML

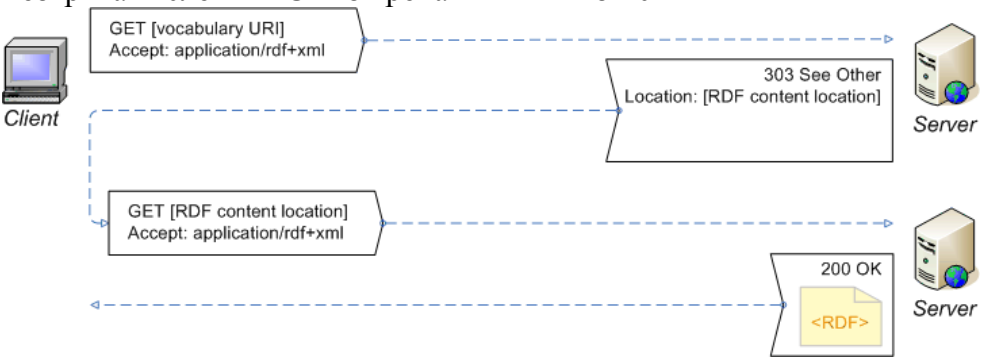
Перейти до прямої: Приклад конфігурації | Тестування конфігурації | Нотатки

Цей рецепт дає приклад розширеної конфігурації для словника з косою рискою імен. Рецепт Налаштування сервера для забезпечення як машинної обробки (PCO) так і людського розуміння (HTML) змістом, залежно від того, що просили, з документацією, HTML приділяється як кілька гіпертекстового документа HTML плюс оглядового документа. Про таку поведінку свідчать наступні діаграми:



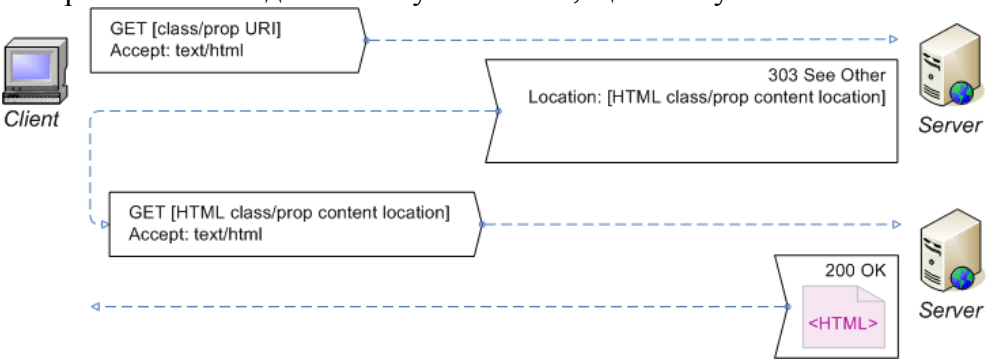
(Перенаправлення клієнта на поточному огляді HTML документацію для словника.)

Розкривати словник URI з проханням RDF зміст



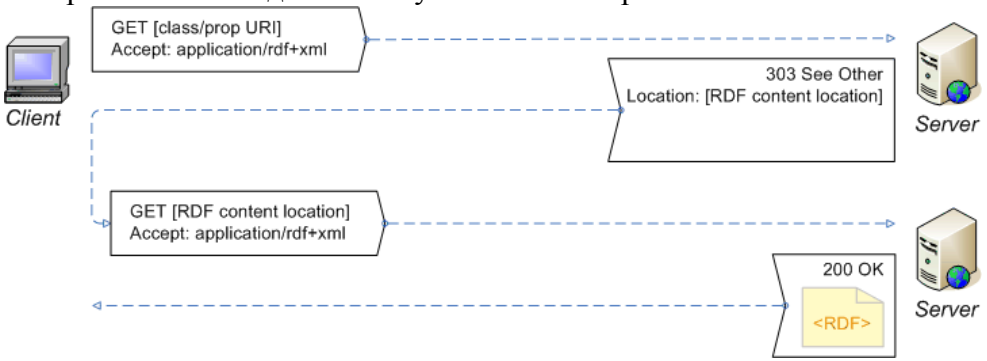
(За станом на 3 рецепт і рецепт 4, перенаправити клієнта на поточному описі RDF лексики.)

Розкривати URI з одного класу або майна, що запитує HTML Content



(Перенаправлення клієнта на поточному HTML документацію для класу або майна).

Розкривати URI з одного класу або майна з проханням RDF зміст



(За станом на 4 рецепт, перенаправляє клієнта на поточному описі RDF лексики.)

Приклад конфігурації

Для словника ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example5/defining

класи ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example5/ClassA

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example5/ClassBand

властивості ...

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example5/propA

http://www.w3.org/2006/07/SWD/recipes/examples-20080421/example5/propB

Крок 1
Створіть файл з ім'ям 2005-10-31.rdf, який містить повну RDF / XML сериализацію лексики, за станом на 2005-10-31 (або що поточна дата). M.I. Всі ресурси визначаються лексикою описаних в цьому файлі, і цей файл є 'моментальним знімком' або 'Версією' лексики.

Крок 2
Копія 2005-10-31.rdf в директорію / apachedocumentroot/examples/example5-content / на сервер.

Крок 3
Створюйте файли ClassA.html ClassB.html propA.html propB.html кожна з яких містить HTML змісту документації, що відноситься до класу або майна з відповідними місцевими іменами, як в 2005-10-31 (або що поточна дата). Створіть файл index.html, який містить HTML документацію про зміст самого словника, з гіперпосиланнями на всіх класах або майна документації.

Крок 4
Копія ClassA.html ClassB.html propA.html propB.html і index.html в директорію / apachedocumentroot/examples/example5-content/2005-10-31-docs / на сервер.

Крок 5
Додати такі директиви. Htaccess файл у каталозі / apachedocumentroot / Приклади / Каталог:

```
# Вимкнути MultiViews
Options-MultiViews

# Директиви для забезпечення RDF файлах *. служать відповідному типу вмісту,
# Якщо немає в основний конфігурації Apache
```

```
AddType застосування / RDF + XML. RDF

# Rewrite установка двигуна
RewriteEngine On
RewriteBase / Приклади

# Правила перезапису 1: служити HTML контент з імен URI, якщо просив
RewriteCond% () HTTP_ACCEPT! Застосування / RDF \ + XML .* (Текст / HTML | Заявка / XHTML \ + XML)
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ + XML [OR]
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
RewriteRule ^ Example5 / $ example5-content/2005-10-31-docs/index.html [R = 303]

# Правила перезапису 2: служити HTML вмісту з класу або опору URI, якщо просив
RewriteCond% () HTTP_ACCEPT! Застосування / RDF \ + XML .* (Текст / HTML | Заявка / XHTML \ + XML)
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ + XML [OR]
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
RewriteRule ^ Example5 /(.+) example5-content/2005-10-31-docs / $ 1.html [R = 303]

# 3 правила перезапису: служити RDF змісту просив
RewriteCond% () HTTP_ACCEPT застосування / RDF \ + XML
RewriteRule ^ Example5 / example5-content/2005-10-31.rdf [R = 303]

# Вибрати за замовчуванням відповідь
# -------------------------
```

```
# 4 правила перезапису: служити RDF / XML стандартний вміст
RewriteRule ^ Example5 / example5-content/2005-10-31.rdf [R =
303]

# Правила перезапису служити HTML стандартний вміст (для
інвалідів)
# (Щоб включити цю опцію, розкоментуйте два правила
перезапису нижче,
# I закомментуйте переписані правила безпосередньо вище)
# RewriteRule ^ Example5 / $ example5-content/2005-10-31-
docs/index.html [R = 303]
# RewriteRule ^ Example5 /(.+) example5-content/2005-10-31-docs
/ $ 1.html [R = 303]
```

(NB Якщо. Htaccess файл не існує, створіть його).

Перевірка конфігурації

Якщо ця конфігурація працює, вона повинна підтримувати такі взаємодії:

Розкривати словник URI, який запитує HTML Content
Тестове повідомлення (замінює правильний шлях і бере свій словниковий запас, URI):

```
GET / examples/example5 / HTTP/1.1
Host: example.com
Accept: Текст / htmlResponse заголовок повинен містити
наступні поля, з вашого HTML огляд місця утримання в якості
значення поля "Розташування":

HTTP/1.x 303 Див інше
Розміщення: http://example.com/examples/example5-content/2005-
10-31-docs/index.html
```

Розкривати словник URI з проханням RDF зміст
Те ж, що рецепт 4.

Розкривати словник URI, за замовчуванням справу
Те ж, що рецепт 4.

Розкривати URI з одного класу або майна, що запитує HTML Content
Тестове повідомлення (замінити правильний шлях і приймає для вашого класу або майна, URI):

GET / examples/example5/ClassA HTTP/1.1
Host: example.com
Accept: Текст / htmlResponse заголовок повинен містити наступні поля, з розташуванням HTML змісту для даного класу або майна, як значення поля "Розташування":

HTTP/1.x 303 Див інше
Розміщення: http://example.com/examples/example5-content/2005-10-31-docs/ClassA.htmlDereference Урі з одного класу або майна з проханням RDF зміст
Те ж, що рецепт 4.

Розкривати URI з одного класу або майна, за замовчуванням справу
Те ж, що рецепт 4.

Нотатки
Див також розділ про зміст переговорів.

Як і рецепт 3, у цьому прикладі використовує дату модифікації версії словника для створення імен файлів. Також було б можливо використовувати номери версій (наприклад, '1 .01 ') замість дат для цієї мети, або якої-небудь конвенції, яка робить можливим проводити різницю між версією словника, а також допомагає відслідковувати історію версій.

Якщо у вас є каталог посилань індексів та MultiViews включений в каталог / apachedocumentroot/examples/example5-content/2005-10-31-docs /, то ви можете замінити і переписати правила 1 і 2 з одного переписати правила:

RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ + XML [OR]
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
RewriteRule ^ Example5 /(.*) example5-content/2005-10-31-docs / $ 1 [R = 303] Ця конфігурація особливо підходить для використання документації, породжених плагін для OWLDoc протеже.

Рецепт 6. Розширена конфігурація для "слеш імен", використовує кілька документів HTML і довідкову службу

Цей рецепт дає приклад розширеної конфігурації для словника з косою рискою імен. Рецепт Налаштування сервера для забезпечення як машинної обробки (PCO) так і людського розуміння (HTML) змісту, залежно від того, що просили, з документацією, HTML приділяється як кілька гіпертекстового документа HTML плюс оглядового документа, зміст RDF час поширюватися через службу таких запитів, які клієнти можуть отримати частковий опис RDF лексики в міру необхідності.

Незважаючи на той факт, що цей рецепт акцій майже всі функції рецепту 5, використання запиту послуги або сценарію для видобування RDF змісту робить його мабуть, найскладнішим з розширених конфігурацій і представляє найбільшу загрозу для забезпечення безпосередньо використаних рецептів. Тому ми вирішили представити цей рецепт як набір пропонованих моделей здійснення, а не "просто виконайте рецептом чисел". Крім того, ми не

описуємо конкретну модель реалізації в тому ж рівні деталізації, зайнятих в інших рецептах. Отже, деякі знання веб програмування, необхідного для здійснення цього рецепта.

План 1: Використання логіки додатка

Ця модель спирається на деяку логіку додатків, розгорнутих у веб-сервері. Ця логіка може бути тонким шаром, який просто перенаправляє запити (або діє як проксі-сервер) на третій-сервера веб-учасника (див. приклад DBPedia нижче), або товстий шар, що завантажується джерелом даних RDF і переводить HTTP-запитів в API дзвінки на виконання запитів.

Є два варіанти введення серверної логіки:

1.Скрипт на стороні сервера.
Невизначений стороні сервера-скрипта Мови (PHP, Python, Perl, Ruby) мають RDF API для прив'язки і з магазинами RDF, і може бути використана, щоб написати простий скрипт, що запитує файли RDF RDF або потрійний магазин і повертає відповідну частину.

- Плюси: простота розгортання (багато хостингові сервери мають підтримку для одного з цих Мов)
- Мінуси: веб-майстрам, як очікується, написати (можливо, спеціальної) Script

2.Java Servlet (або еквівалент).

- Плюси: Java надає непогану підтримку RDF, SPARQL і RDF потрійний магазини
- Мінуси: у суперважкій рішення важких для розгортання (необхідний контейнер сервлетів)

Приклади реалізації (дійсні для обох варіантів 1 і 2).

DBPedia сервер використовується у прикладі як контент-провайдера:

RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ +

```
XML
RewriteRule ^ Example6 /(.+) http://dbpedia.org/page/ $ 1 [R = 303]

RewriteCond% () HTTP_ACCEPT застосування / RDF \ + XML
RewriteRule ^ Example6 /(.+) http://dbpedia.org/data/ $ 1 [R = 303]

RewriteRule ^ Example6 /(.+) http://dbpedia.org/data/ $ 1 [R = 303]
```

Цей приклад 'тонких' здійснення шару і переписати правила прості:

Прохання про HTML дані направляються на адресу версія HTML експортувала сервлет DBPedia;
Запити про надання даних RDF (або без "Acccept:" Тема після повернення RDF поведінки за замовчуванням) передаються в URL даних RDF.
Складна частина реалізації рецепту 6 з допомогою застосування логіки щоб правильно застосовувати HTTP Content-переговори з нуля. Хоча більшість веб Мови сценаріїв (PHP, Python і т.п.) і рамки забезпечують доступ до значення заголовків HTTP і, таким чином, до "Accept: заголовок", вибравши відповідний тип повертається далеко від тривіального. Заголовком 'Accept: "може містити маски і Q-значень, тому регулярних виразів і простих функцій порівняння рядків недостатньо. Існує вікі-сторінка з покажчиками на узгодження змісту бібліотек для різних мов програмування. Робоча група пропонує внески додаткових бібліотек і структур, які підтримують зміст переговорів.

План 2: Перенаправлення на кінцеву точку SPARQL допомогою Apache

Ця схема не передбачає написання будь-якої логіки додатків. Замість того, запити HTTP-перенаправлення допомогою Apache mod_rewrite. Особливо добре ця техніка підходить для упаковки існуючої кінцевої точки SPARQL. Ми експлуатували той факт, що багато в кінці SPARQL експорт HTTP прив'язок.

- Плюси: легкий, не вимагає програмування
- Мінуси: SPARQL кінцевою точкою для словника має бути доступно десь

Приклади реалізації:

```
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \
+ XML
RewriteRule ^ Example6 /(.+) http://dbpedia.org/ $ 1 [R = 303]

RewriteCond% () HTTP_ACCEPT застосування / RDF \ +
XML
RewriteRule ^ Example6 /(.+)
http://dbpedia.org/sparql?query=DESCRIBE+
<http://dbpedia.org/$1> [R = 303]

RewriteRule ^ Example6 /(.+)
http://dbpedia.org/sparql?query=DESCRIBE+
<http://dbpedia.org/$1> [R = 303]
```

У цьому прикладі, коли клієнт запитує HTML зміст, запит перенаправляється на зовнішній веб-сервер. Тим не менше, RDF запити обробляються по-різному. Запит URI таких як http://example.org/example6/property/birthplace перенаправляється на результат виконання DESCRIBE <http://dbpedia.org/property/birthplace> вирок щодо кінцевих DBPedia SPARQL. У результаті графа RDF, яка описує ресурсів (у даному конкретному випадку, власність, яка пов'язує людей до місця, де вони народилися).

Тематичні дослідження

- Джошуа Тауберер оголосив, що він піддається великим набором даних RDF перепису населення США. Існує

SPARQL доступних кінцевих точок, а URI, є dereferencable шляхом переписування URL (див., наприклад, http://www.rdfabout.com/rdf/usgov/geo/us)

- D2R Server обгортання реляційних баз даних і забезпечує SPARQL та пов'язані дані інтерфейси до нього. Останній приклад перенаправлення 303 і зміст переговорів з метою обслуговувати як HTML і RDF описів ресурсів. З іншого боку, забезпечує Pubby пов'язані інтерфейс даних існуючої кінцевої точки SPARQL. Більш детальна інформація у розділі 5 'Cool URI, для Semantic Web' [COOLURI].

Потреби

У цьому розділі спроби сформулювати вимоги і очікування Семантичних веб-додатків і програм додатків по відношенню до поведінки HTTP словників, класів і властивостей позначається HTTP URI (наприклад, URI з http:URI простору). Він покликаний служити еталоном, за яким на прикладі конфігурації наведені в рецептах вище, можуть бути перевірені.

Мінімальні вимоги

M1. "Авторитетних" RDF опису словника, клас або майна позначається HTTP URI може бути отриманий через покажчик URI про те, що лексика, клас чи майна.
HTTP клієнт може отримати "авторитетних" RDF опису лексики, клас, або майна, виконавши HTTP GET запиту проти URI про те, що лексика, клас чи майна. Опис RDF повертається як відповідь HTTP, зміст якого тип зареєстрованих MIME типів вмісту RDF (в даний час тільки додаток '/ RDF + XML').

Ця поведінка за замовчуванням у випадку тій чи іншій формі змісту переговорів була впроваджена для цих URI. M.I. HTTP

GET запит без "Прийняти:" Поле заголовка приведуть у відповідь з типом вмісту 'Application / RDF + XML ", який виконує сериализацию набору операторів RDF, у тому числі тих заяв, які представляють собою" авторитетних "RDF Опис в зазначений ресурс.

N.B. це розумно за спробу разименовать URI з власності RDF або класу призведе до опису RDF більше, ніж просто це майно або класу.

M2. Поведінка HTTP URI позначають RDFS / OWL лексику, класової приналежності або власності, не призводить до непослідовності в трактуванні природи позначений ресурс.
В даний час архітектура веб-додатків дозволяють робити висновки про характер того чи іншого ресурсу позначимо через HTTP URI, на основі наступного:

(I) код HTTP відповіді отримано при разименованіі URI (див. резолюцію TAG тема 'httpRange-14' [HTTPRANGE14]),

і ...

(II) Якщо URI містить ідентифікатор фрагмента, тип змісту (и) наявні подання [AWWW04].

З огляду на ці обмеження, для кожного HTTP URI позначають RDFS / OWL лексика, класової приналежності або власності, коло можливих відповідей на HTTP-запити щодо цієї URI не призведе програми робити які-небудь висновки несумісні.

Додаткові вимоги

Ці вимоги є продовженням мінімальним вимогам.

E1. 'Людиночітаємість документації про лексику RDF, класової приналежності або власності, позначається HTTP

URI, може бути отриманий через покажчик URI про те, що лексика, клас чи майна.
Клієнт HTTP такого як веб-браузера може отримати 'людське розуміння документації, що відноситься до RDFS / OWL лексики, класової приналежності або власності, виконавши HTTP GET запиту проти URI про те, що лексика, клас або майна, із зазначенням "Прийняти:" Заголовки відповідні потрібний тип вмісту у запиті.

E2. Програми здатні розрізняти 'версії' про лексику.
Словники змінюються з плином часу як властивості або класах додається або їх описів в редакційному змінилася. Додатки повинні таким чином провести різницю між послідовними 'знімками' зі словників з плином часу. Щоб бути точним, що така "версія" є описом нерухомості - тобто, набір RDF заяви про нерухомості, - а не саме майно, URI якого не зміниться.

Конвенцій в загальному користуванні для розрізнення послідовні Описи включають використання версії номер рядка або дати рядка в імені файлу (наприклад, 1.01.rdf або 2005-10-31.rdf) або у шляхах (наприклад, http://dublincore.org/2008/01/14/dcelements.rdf # Title). Слід зазначити, що в даний час не існує загальноприйнятих конвенцій для використання дати або версії номер рядка таким чином.

Подяки
Приклади спроба переганяти елементи передової практики в даний час розгорнуті Семантичний веб-словниками, особливо Dublin Core Metadata умови, один одному онтології, і SKOS Core Vocabulary. Всі ті, хто зробив внесок у розвиток цієї практики були вдячні.

Список літератури

Дана бібліографія, як: BibTeX | BibTeXML | RDF

APACHE20
Apache HTTP Server версії 2.0 Документація, Apache Software Foundation.
Наявність на http://httpd.apache.org/docs/2.0/

AWWW04
Архітектура World Wide Web, Volume One, Ян Якобс і Norman Walsh, World Wide Web Consortium, W3C Рекомендація грудня 2004 року.
Наявність на http://www.w3.org/TR/2004/REC-webarch-20041215/

COOLURI
Cool URI, для Semantic Web, Лев Sauermann, DFKI GmbH; Річард Cyganiak Вільного університету Берліна. W3C Interest Group Примітка березня 2008
Наявність на http://www.w3.org/TR/cooluris/

FOAF
FOAF Словник за специфікаціями, Ден Brickley і Ліббі Міллером.
Наявність на http://xmlns.com/foaf/0.1/

GRDDL
Gleaning опису ресурсів з діалектів Мови (GRDDL), Ден Конноллі, World Wide Web Consortium, W3C рекомендація вересня 2007
Наявність на http://www.w3.org/TR/grddl/

HTTPRANGE14

Разименованіе HTTP URI, Риса Льюіса, Volantis системи ТОВ
Наявність на http://www.w3.org/2001/tag/doc/httpRange-14/2007-10-04/HttpRange-14.html

W3C Технічна архітектура Групи резолюцію по колу HTTP разименованіі (AKA "httpRange-14").
Наявність на http://lists.w3.org/Archives/Public/www-tag/2005Jun/0039.html

OWLFeatures
OWL Web Ontology Language Огляд, Дебора Л. МакГіннесс і Франк Ван Harmelen, World Wide Web Consortium, W3C Рекомендація лютого 2004 року.
Наявність на http://www.w3.org/TR/2004/REC-owl-features-20040210/

OWLGuide
OWL Web Ontology Language Guide, Майкл К. Сміт, Кріс Welty і Дебора Л. Макгіннесс, World Wide Web Consortium, W3C Рекомендація лютого 2004 року.
Наявність на http://www.w3.org/TR/2004/REC-owl-guide-20040210/

PURL
Введення Стійкі Uniform Resource локатори, Кейт Шейфер, Вайбель Стюарт, Ерік липня і Джон Fausey, 6565 Франц Road, Дублін, Огайо 43017-3395: OCLC Інтернет Computer Library Center, Inc, 1996.
Наявність на http://purl.oclc.org/docs/inet96.html

RDFa
RDFa в XHTML: Синтаксис і обробка, Б. Adida, M. Birbeck, С. Маккаррон, С. Пембертон, World Wide Web Consortium, W3C Кандидат Рекомендації червня 2008
Наявність на http://www.w3.org/TR/2008/CR-rdfa-syntax-20080620/

RDFPrimer
RDF Primer, Франк Маноло і Ерік Міллер, World Wide Web Consortium, W3C Рекомендація лютого 2004 року.
Наявність на http://www.w3.org/TR/2004/REC-rdf-primer-20040210/

RDFS
RDF Словник мови опису 1.0: RDF Schema, Dan Brickley і P. Гуха, World Wide Web Consortium, W3C Рекомендація лютого 2004 року.
Наявність на http://www.w3.org/TR/2004/REC-rdf-schema-20040210/

RFC2616
Hypertext Transfer Protocol - HTTP/1.1, R. Fielding, J. Геттіс, J. Mogul, H. Frystyk, Л. Masinter, П. Лич, Т. Бернерс-Лі, Internet Engineering Task Force, RFC (2616), червень 1999 .
Доступно за адресою: http://www.ietf.org/rfc/rfc2616.txt

RFC3986
Uniform Resource Identifier (URI): Generic Syntax, Т. Бернерс-Лі, Р. Філдінг, Л. Masinter, Internet Engineering Task Force, RFC (3986), січень 2005 року.
Доступно за адресою: http://www.ietf.org/rfc/rfc3986.txt

SKOS
SKOS Core Словник за специфікаціями, Alistair Мила і Дана Brickley, World Wide Web Consortium, W3C Working Draft, листопад 2005 року.
Наявність на http://www.w3.org/TR/2005/WD-swbp-skos-core-spec-20051102/

XMLNAMES

Простори імен в XML, Тім Брей, Дейв Холландер і Ендрю профана, World Wide Web Consortium, W3C Рекомендація січня 1999 року.
Наявність на http://www.w3.org/TR/1999/REC-xml-names-19990114
/

Додаток A. словники, які використовують для іменування URI
PURLs ('Стійкі URL ") є URI, з http://purl.org/ просторі URI. Дзюрчить підтримуються службою PURL резолюції, яка дозволяє зареєстрованим власникам доменів PURL для перенаправлення HTTP-запити щодо PURL для довільного URL ресурсу. Зареєстровані власники дзюрчить не може налаштувати центральний сервер PURL, ніж інші, щоб вказати URL переадресації для кожного Purl.

Коли центральний сервер PURL спочатку було розроблено в 1990-х років стандартна реакція HTTP сервера на запит відносно PURL було отримано відповідь Кодексу 302 ("тимчасово переїхала"). Веб-архітектура перетворилася з тих пір, і технічною архітектурою Group (TAG) від W3C прийняло рішення про те, що для такого перенаправлення на RDF клас і імена властивостей, код відповіді 303 ("Інші"), повинні бути повернуті (див. TAG резолюції по httpRange-14 [HTTPRANGE14]).

Як PURL сервери використовують 302 код відповіді і в даний час немає можливості налаштувати їх на використання 303 кодів відповіді, існуючих словників з http://purl.org слеш імена серверів не в суворій відповідності до чинного рекомендацій TAG. Ці справи розглядаються в наступні рецепти.

Примітка: Під час цієї робочої проект пишеться (січень 2008), оновлена до служби PURL триває. Ми очікуємо цього

оновлення будуть розглянуті TAG висновок за httpRange-14 і 303 переадресацію.
(див.http://www.oclc.org/news/releases/200669.htm).

Рецепти 1а. | Рецепт 2а. | Рецепт 3а. | Рецепт 4а. | Рецепт 5а.

Рецепти 1а. Мінімальна конфігурація для хеш імен PURL '
Цей рецепт дає приклад конфігурації, який відповідає мінімальним вимогам для словника з хеш імен в просторі http://purl.org/ URI. Тільки машинної обробки (PCO) зміст подається в простір імен URI.

Для словника ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example1adefining
класи ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example1a # ClassA
http://purl.oclc.org/NET/SWD/recipes/examples-A/example1a # ClassBand
властивостями ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example1a # Propa
http://purl.oclc.org/NET/SWD/recipes/examples-A/example1a # propB

Крок 1.Створіть файл з ім'ям example1a.rdf, яка містить повну RDF / XML сериализация словника. M.I. всі ресурси, певні лексиці описаних в цьому файлі.

Крок 2

Копія example1a.rdf в директорію / apachedocumentroot / приклади / на сервер, з якого ви хотіли б служити зміст (в даному прикладі сервер www.w3.org).

Крок 3

Встановити такі PURL:

PURL: http://purl.org/NET/SWD/recipes/examples-A/example1a
URL:
http://www.w3.org/2006/07/SWD/recipes/examples/example1a.rdf

Нотатки

Якщо сервер вже налаштований для роботи з файлами з розширенням. RDF розширення як зміст тип програми / RDF + XML тоді вам не доведеться нічого робити далі. Якщо це не так, то вам необхідно додати наступну директиву:

```
AddType застосування / RDF + XML. Rdf
```

на основні файлу конфігурації Apache, або якщо ви не маєте доступу до цих, в кожному каталозі файлів конфігурації (. Htaccess) в каталозі / apachedocumentroot / приклади / на сервер.

Рецепти 2a. Мінімальна конфігурація для слеш імен PURL '

Цей рецепт дає приклад конфігурації, який відповідає мінімальним вимогам для словника з косою риси імен в просторі http://purl.org/ URI. Тільки машинної обробки (PCO) зміст подається в простір імен URI.

N.B. За станом на дату цього робочого проекту, цей приклад не в суворій відповідності до резолюції мітки на httpRange-14 [HTTPRANGE14], тому що purl.org сервери використовують 302 код переадресації, а не 303.

Для словника ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example2a/defining
класи ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example2a/ClassA

http://purl.oclc.org/NET/SWD/recipes/examples-A/example2a/ClassBand
властивості ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example2a/propA

http://purl.oclc.org/NET/SWD/recipes/examples-A/example2a/propB

Крок 1.Створіть файл з ім'ям example2a.rdf, яка містить повну RDF / XML сериализация словника. M.I. всі ресурси, певні лексиці описаних в цьому файлі.

Крок 2
Копія example2a.rdf в директорію / apachedocumentroot / приклади / на сервер, з якого ви хотіли б служити зміст (в даному прикладі сервер www.w3.org).

Крок 3
Установка переадресації після часткового PURL:

PR PURL: http://purl.oclc.org/NET/SWD/recipes/examples-A/example2a/
Кореневий URL:
http://www.w3.org/2006/07/SWD/recipes/examples/example2a/
Крок 4

Додати такі директиви. Htaccess файл у каталозі / apachedocumentroot / приклади / каталогів на сервері:

```
# Вимкнути MultiViews
Options-MultiViews

# Директиви для забезпечення RDF файлах *. служив відповідний тип вмісту,
# Якщо немає в основний конфігурації Apache
AddType застосування / RDF + XML. RDF

# Rewrite установка двигуна
RewriteEngine On
RewriteBase / Приклади

# Правила перезапису служити RDF / XML змісту з усіх частково перенаправлені URI,
RewriteRule ^ example2a / example2a.rdf
```

(NB Якщо файл. Htaccess файл не існує, створіть його).

Нотатки

У наведеному вище рецепт одного переписати правила внутрішнього перенаправлення. Це дозволяє мінімізувати кількість зовнішніх (наприклад, HTTP) перенаправляє що беруть участь в разименовиваем дій. Однак, ви могли б також застосовувати це правило перезапису як зовнішня переадресація, замінивши вищезгадане правило наступного змісту:

```
# Правила перезапису служити RDF / XML змісту з усіх частково перенаправлені URI,
RewriteRule ^ example2a / example2a.rdf [R = 303]
```

Це створює додаткові перенаправлення HTTP в разименовиваем дій, але, можливо, робить її більш зрозумілою клієнтові, що спроби разименовиваем Лексика, клас чи властивість URI, все в кінцевому підсумку на те ж місце, і воно

робить поточної реалізації PURL сумісні з httpRange TAG-14 роздільною здатністю [HTTPRANGE14].

Можливо також, щоб уникнути будь-якої конфігурації сервера шляхом створення окремого URI для кожного сорту і майна лексика все ж посилання на URL (а не часткової PURL Redirect). Однак, якщо зміст згодом могли бути переміщені, кожна PURL необхідно буде оновлюватися - громіздким і непрактичним завдання для середніх і великих розмірів онтологій.

Рецепти 3a. Розширена конфігурація для хеш імен PURL '
Цей рецепт дає приклад конфігурації, який задовольняє вимогам для розширеного словника з хеш імен в просторі http://purl.org/ URI. Обидва машинної обробки (PCO) і людського розуміння (HTML) зміст подається в простір імен URI.

Для словника ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example3adefining

класи ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example3a # ClassA

http://purl.oclc.org/NET/SWD/recipes/examples-A/example3a # ClassBand

властивості ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example3a # Propa

http://purl.oclc.org/NET/SWD/recipes/examples-A/example3a # propB

Крок 1.
Створіть файл з ім'ям 2005-10-31.rdf, яка містить повну RDF / XML сериализация лексики, за станом на 2005-10-31 (або що поточна дата). M.I. всі ресурси визначаються лексикою описаних в цьому файлі, і цей файл є 'моментальним знімком' або 'Версією' лексики.

Крок 2
Створіть файл з ім'ям 2005-10-31.html, який містить HTML зміст документації по всіх класах і властивості визначаються лексикою станом на 2005-10-31 (або що поточна дата). Цей документ може включати розділи по кожному з класів / Властивості документами, у кожному розділі очолюють якір HTML, ім'я якого збігається з ідентифікатором фрагмента з документованих клас чи майна.

Крок 3
Копія 2005-10-31.rdf і 2005-10-31.html в директорію / apachedocumentroot/examples/example3a-content / на сервер, з якого ви хотіли б служити зміст (в даному прикладі сервер www.w3. Org).

Крок 4
Додати такі директиви. Htaccess файл у каталозі / apachedocumentroot / Приклади / Каталог:

```
# Вимкнути MultiViews
Options-MultiViews

# Директиви для забезпечення RDF файлах *. служив
відповідний тип вмісту,
# Якщо немає в основний конфігурації Apache
AddType застосування / RDF + XML. RDF

# Rewrite установка двигуна
RewriteEngine On
RewriteBase / Приклади

# Правила перезапису, щоб переконатися, ми служимо HTML
контент з імен URI, якщо просив
RewriteCond% () HTTP_ACCEPT! Застосування / RDF \ + XML
.* (Текст / HTML | Заявка / XHTML \ + XML)
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ +
XML [OR]
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
RewriteRule ^ example3a $ example3a-content/2005-10-31.html [R
= 303]

# Правила перезапису, щоб переконатися, ми діємо в RDF /
XML контент з імен URI за замовчуванням
RewriteRule ^ example3a $ example3a-content/2005-10-31.rdf [R =
303]
```

Крок 5
Встановіть наступні PURL:

PURL: http://purl.oclc.org/NET/SWD/recipes/examples-A/example3a

URL:
http://www.w3.org/2006/07/SWD/recipes/examples/example3a
Нотаткі
Тому що ми не можемо налаштувати сервер для PURL змісту переговорів, у цьому прикладі Налаштування сервера для виконання переговорів після 302 переадресацію з сервера Purl.

Рецепти 4a. Розширена конфігурація слеш імен PURL'', одного документа HTML
Цей рецепт дає приклад конфігурації, який задовольняє вимогам для розширеного словника з косою рискою імен в просторі http://purl.org/ URI. Обидва машинної обробки (PCO) і людського розуміння (HTML) зміст подається в простір імен URI. Документація HTML подається у вигляді окремого файлу.

N.B. цей приклад не в суворій відповідності до резолюції мітки на httpRange-14 [HTTPRANGE14], тому що purl.org сервери використовують 302 код переадресації, а не 303.

Для словника ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example4a/defining
класи ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example4a/ClassA

http://purl.oclc.org/NET/SWD/recipes/examples-A/example4a/ClassBand
властивості ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example4a/propA

http://purl.oclc.org/NET/SWD/recipes/examples-A/example4a/propB

Крок 1.
Створіть файл з ім'ям 2005-10-31.rdf, який містить повну RDF / XML сериализацию лексики, за станом на 2005-10-31 (або що поточна дата). M.I. всі ресурси визначаються лексикою описаних в цьому файлі, і цей файл є 'моментальним знімком' або 'Версією' лексики.

Крок 2
Створіть файл з ім'ям 2005-10-31.html, який містить HTML зміст документації по всіх класах і властивості визначаються лексикою станом на 2005-10-31 (або що поточна дата). Цей документ може включати розділи по кожному з класів / Властивості документами, у кожному розділі очолюють якір HTML, ім'я якого збігається з ідентифікатором фрагмента з документованих клас чи майна.

Крок 3
Копія 2005-10-31.rdf і 2005-10-31.html в директорію / apachedocumentroot/examples/example4a-content / на сервер, з якого ви хотіли б служити зміст (в даному прикладі сервер www.w3. Org) ..

Крок 4
Додати такі директиви. Htaccess файл у каталозі / apachedocumentroot / Приклади / Каталог:

```
# Вимкнути MultiViews
Options-MultiViews

# Директиви для забезпечення RDF файлах *. служив
відповідний тип вмісту,
```

```
# Якщо немає в основний конфігурації Apache
AddType застосування / RDF + XML. RDF

# Rewrite установка двигуна
RewriteEngine On
RewriteBase / Приклади

# Правила перезапису служити HTML контент з імен URI,
якщо просив
RewriteCond% () HTTP_ACCEPT! Застосування / RDF \ + XML
.* (Текст / HTML | Заявка / XHTML \ + XML)
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ +
XML [OR]
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
RewriteRule ^ example4a / $ example4a-content/2005-10-31.html
[R = 303]

# Правила перезапису служити направлено HTML вміст класу
/ URI, опора
RewriteCond% () HTTP_ACCEPT! Застосування / RDF \ + XML
.* (Текст / HTML | Заявка / XHTML \ + XML)
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ +
XML [OR]
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
RewriteRule ^ example4a /(.+) example4a-content/2005-10-
31.html # $ 1 [R = 303, NE]

# Правила перезапису служити RDF / XML контент з імен URI
за замовчуванням
RewriteRule ^ example4a / example4a-content/2005-10-31.rdf [R
= 303]
```

(NB Якщо. Htaccess файл не існує, створіть його).

Крок 5
Установка переадресації після часткового PURL:

PR PURL: http://purl.oclc.org/NET/SWD/recipes/examples-A/example4a/

URL Root:
http://www.w3.org/2006/07/SWD/recipes/examples/example4a/
Нотатки
Ця конфігурація буде найбільш відповідним для Dublin Core Умови метаданих.

Рецепти 5a. Розширена конфігурація слеш імен PURL ', використовуючи кілька документів HTML
Цей рецепт дає приклад конфігурації, який задовольняє вимогам для розширеного словника з косою рискою імен в просторі http://purl.org/ URI. Обидва машинної обробки (PCO) і людського розуміння (HTML) зміст подається в простір імен URI з документацією HTML приділяється як кілька гіпертекстового документа HTML плюс оглядовому документі.

N.B. цей приклад не в суворій відповідності до резолюції мітки на httpRange-14 [HTTPRANGE14], тому що purl.org сервери використовують 302 код переадресації, а не 303.

Для словника ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example5a/defining

класи ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example5a/ClassA

http://purl.oclc.org/NET/SWD/recipes/examples-A/example5a/ClassBand
властивості ...

http://purl.oclc.org/NET/SWD/recipes/examples-A/example5a/propA

http://purl.oclc.org/NET/SWD/recipes/examples-A/example5a/propB

Крок 1.
Створіть файл з ім'ям 2005-10-31.rdf, яка містить повну RDF / XML сериализация лексики, за станом на 2005-10-31 (або що поточна дата). M.I. всі ресурси визначаються лексика описаних в цьому файлі, і цей файл є 'моментальний знімок' або 'Версія' лексики.

Крок 2
Копія 2005-10-31.rdf в директорію / apachedocumentroot/examples/example5a-content / на сервер, з якого ви хотіли б служити зміст (в даному прикладі сервер www.w3.org).

Крок 3
Створюйте файли ClassA.html ClassB.html propA.html propB.html кожна з яких містить HTML змісту документації, що відноситься до класу або майна з відповідними місцевими іменами, як в 2005-10-31 (або що поточна дата). Створіть файл index.html, яка містить HTML документацію про зміст самого словника, з гіперпосиланнями на всім класом або майну документації.

Крок 4
Копія ClassA.html ClassB.html propA.html propB.html і index.html в директорію / apachedocumentroot/examples/example5a-content/2005-10-31-

docs / на сервер, з якого ви хотіли б служити зміст (в цьому прикладі сервер www.w3.org).

Крок 5

Додати такі директиви. Htaccess файл у каталозі / apachedocumentroot / Приклади / Каталог:

```
# Вимкнути MultiViews
Options-MultiViews

# Директиви для забезпечення RDF файлах *. служив відповідний тип вмісту,
# Якщо немає в основний конфігурації Apache
AddType застосування / RDF + XML. RDF

# Rewrite установка двигуна
RewriteEngine On
RewriteBase / Приклади

# Правила перезапису служити HTML контент з імен URI, якщо просив
RewriteCond% () HTTP_ACCEPT! Застосування / RDF \ + XML .* (Текст / HTML | Заявка / XHTML \ + XML)
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ + XML [OR]
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
RewriteRule ^ example5a / $ example5a-content/2005-10-31-docs/index.html [R = 303]

# Правила перезапису служити HTML вмісту з класу або опору URI, якщо просив
RewriteCond% () HTTP_ACCEPT! Застосування / RDF \ + XML .* (Текст / HTML | Заявка / XHTML \ + XML)
RewriteCond% () HTTP_ACCEPT Текст / HTML [OR]
RewriteCond% () HTTP_ACCEPT застосування / XHTML \ + XML [OR]
```

```
RewriteCond% (HTTP_USER_AGENT) ^ Mozilla / .*
RewriteRule ^ example5a /(.+) example5a-content/2005-10-31-
docs / $ 1.html [R = 303]

# Правила перезапису служити RDF / XML контент з імен
URI за замовчуванням
RewriteRule ^ example5a / example5a-content/2005-10-31.rdf [R
= 303]
```

(NB Якщо. Htaccess файл не існує, створіть його).

Крок 6
Встановити наступні параметри переадресації Часткова URI:

PR PURL: http://purl.oclc.org/NET/SWD/recipes/examples-A/example5a/

Кореневий URL:
http://www.w3.org/2006/07/SWD/recipes/examples/example5a/

Додаток Б: URI простору імен
URI, який ідентифікує ваш словниковий запас згадується тут у якості словника імен URI або просто словника URI (або онтологія URI якості словника і онтології тут використовуються як взаємозамінні). Наприклад, наступний URI ідентифікує SKOS Core Vocabulary:

http://www.w3.org/2004/02/skos/
класи ...
і наступні URI ідентифікує FOAF онтології:

http://xmlns.com/foaf/0.1/

Словники що використання 'Hash імен'

SKOS Core [SKOS] є прикладом словника, який використовує хеш імен. Це неофіційний вираз, який посилається на яких будуються URI, для класів і властивостей у лексиці. У цьому випадку URI, для класів і властивостей будуються шляхом додавання першого хеш ('#') характеру, а потім 'Місцева назва' на словника URI. 'Місцева назва' є рядком символів, що однозначно визначає, що клас або майна в рамках лексики, відомої також як "ідентифікатор фрагмента '[AWWW04] (місцева назва повинна бути юридична [XML-NS] знак NCName).

Наприклад, наступний URI, визначити клас і майна з словника SKOS Core:

http://www.w3.org/2004/02/skos/core # Концепції
http://www.w3.org/2004/02/skos/core # prefLabel

Словники, що використовують 'слеш імен'

FOAF [FOAF] є прикладом лексики, яка використовується коса риска імен. Будуються Знову ж таки, це неофіційне вираз, що відноситься до того, в якому URI, для класів та властивостей, визначених у лексиці. У цьому випадку словника URI кінці з косою рискою характеру ('/'), і URI, класів і властивостей будуються шляхом додавання 'Місцевої назви' класу або майна безпосередньо у лексиці URI. Знову ж таки, "місцевим назвою 'є рядком символів, що однозначно визначає, що клас або майна в рамках лексики, і повинні бути правові [XML-NS] знак NCName.

Наприклад, наступний URI, визначити клас і майна з словника FOAF:

http://xmlns.com/foaf/0.1/Person

http://xmlns.com/foaf/0.1/maker

Примітка, що лексика якого URI кінці з косою риси характеру, не обов'язково використовувати косу риску імен. Вона могла б використовувати хеш імен, наприклад, лексика http://example.org/myvocabulary/ може визначити класи http://example.org/myvocabulary/ # Poo і http://example.org/myvocabulary/ # Bar.

Обидва хеш імен та слеш імен підтримуються в архітектурі Інтернету. Однак певні дії потрібні від веб-сервера, які відрізняються від цих двох варіантів. Оскільки обидва прохання, отримані від сервера і відповідей, що повертаються сервером різні в кожному конкретному випадку, механіка створення сервера HTTP задовольнити деякі або всі вимоги, наведені нижче, також відрізняються, і, отже, ці випадки розглядаються окремо .

Словники, які використовують інші типи імен

Читачі повинні усвідомлювати третій тип словника URI обговорюється на момент написання: URI, заснованих на 303-Redirect послуг, таких як http://thing-described-by.org.
Хоча простіше реалізувати, ніж підходи, описані в цьому документі, 303-Redirect підхід до цих пір не виконані через стабільного, опубліковані RDF словників і не використовується ні в одному з наступних рецептів. Додаток C описує цей підхід більш докладно.

Деякі міркування при виборі імен URI

Цей документ призначений для авторів і супроводжуючі існуючі словники. Належного керівництва з вибору кращих URI простору імен для тієї чи іншої ситуації, виходить за рамки цього документа. Проте, рецепти Наведені тут робити припущення і залучати компроміси щодо функціональних можливостей, тому деякі міркування, пов'язані з вибором імен

URI описані в цьому розділі. Якщо ви вже вибрали імен URI, перейдіть до Вибір рецепта.

URI імен вибрати для вашого словникового запасу має бути веб-адреса (URI), до якої у Вас є доступ на запис. Інші, які використовують свій словниковий запас, будуть чекати щоб мати можливість розйменувати обидва словника URI себе, а також URI, властивостей і класів, визначених свій словниковий запас. Вибір імен URI є одним з основних рішень Ви робите на початку розробки свіго словникового запасу.

Хоча RDF дозволу простору імен для початку будь-якого правомірного схема URI, найкращої практики для Semantic Web є використання схеми URI, яка може бути вирішена будь-яким клієнтом без необхідності використання додаткових плагінів або конфігурації установки клієнта. HTTP URI схема є найбільш відомою з них, і це визнається всіма веб-клієнтами. Цей документ орієнтований виключно на словники імен з ім'я якого починається з HTTP:.

Найкраща практика підказує, що всі словники RDF використовували хеш імен або слеш імен (див. вище). Який ви виберете, залежить частково від того, який ви очікуєте великий ваш словниковий запас, щоб знати, як часто Ви плануєте додати нові терміни (наприклад, властивостей і класів), і як ви очікуєте користувачам отримувати доступ до інформації про окремих термінів у свій словниковий запас.

Для малих словників, вона може бути найбільш зручною для обслуговування всього словникового запасу в одному веб-доступу. Така лексика, як правило, використовує хеш імен та веб-доступ (наприклад, HTTP GET запит) на будь-який термін, в лексиці повернеться одному ресурсі інформацію з описом всіх цих термінів у словнику.

Словникового запасу, що є великим, до якого очікуються додатки часто, або що визначає більше даних, ніж типовий додаток користувачеві потрібно отримати доступ в один час, повинна бути організована таким чином, поступово Більш докладно про умови в словнику може бути відновлена через кілька Веб-доступ. Повний опис всіх умов може бути розділена між багатьма інформаційними ресурсами, або може управлятися через запит послуги (наприклад, [SPARQL] - див рецепт 6). В такій лексиці, як правило, використовується коса риска імен, яка передбачає можливість того, що доступ до веб-на будь-який термін, в лексиконі може повертати інформацію головним чином про тільки що один термін. (Така конфігурація не має можливості словника, який використовує хеш імен, бо механіки протокол HTTP.)

Більш детальне обговорення питань, пов'язаних з проектуванням URI для Semantic Web можна знайти в цих документах:

- "Cool URI, для Semantic Web" [COOLURI]
- TAG знаходження: "об'єднання ресурсів за допомогою просторів імен"

---------- ----------

Додаток C. Словник URI, заснованих на 303-перенаправити

URI, такого типу утворюються шляхом URI додавання з описових ресурсів, а рядок запиту до бази URI з 303-Redirect послуг, таких як http://thing-described-by.org.

Домен речі описані-by.org делегує повноваження з визначення значення такого запиту URI до домену цитується у рядок запиту (наприклад, частина після знаку питання).

В принципі, то, можна монетою http://thing-described-by.org?http://example.org/foo

URI в якості ідентифікатора для словника Foo. HTTP GET запиту проти URI для словника Фу, або у відношенні майна або класу в лексиці Фу, призведе код відповіді 303, що відповідає другому з двох мінімальних вимог, сформульованих у цьому документі для публікації RDF словників. Якщо, крім того, http://example.org/foo URI було визначено авторитетний опис RDF для словника, а сервер за умови, що описи були повернутися MIME тип правильно визначити її як таку, то використання HTTP : / / речі описані-by.org? http://example.org/foo
можна було б сказати, щоб вони відповідали першій з мінімальною вимогою.

Додаток D: Настройка Apache

Сервер Apache HTTP [APACHE20] налаштована директивами письмового або всередині основний конфігураційний файл Apache (т.д. звичайно "httpd.conf") або внутрішньо-файли в каталозі конфігурації (обично. Htaccess ').В наведених тут рецептах припустимо, що ви не маєте доступу до основної конфігурації Apache файлів і тому ви повинні використовувати конфігурацію за допомогою каталогу. Htaccess файли на поставку конфігураційних директив. Ви можете знайти більш детальну інформацію про використання. Htaccess файли в Apache Tutorial:. Htaccess файлів.

Якщо у вас є доступ на запис в основний конфігурації Apache файлів, можливо, дати конфігураційних директив прямо туди, тому що при кожному каталозі конфігурації може негативно позначитися на продуктивності сервера і вимагає більш ретельної конфігурації каталозі рівня безпеки см. Apache Підручник: Коли (не) для використання. Htaccess файлів. Слід зазначити, що AllowOverride є каталогом рівні Директиви та її ефективності та безпеки побічні ефекти звичайно не виходить за межі дерева каталогів, в яких його дії.

Примітка: Конфігурація Наведені тут були протестовані на сервер Apache HTTP версії 2.0.46 тільки.

Конфігурація сервера

З метою підтримки такого використання файлів в каталозі конфігурації, сервер повинен бути налаштований на вирішення певних перекриває для каталогів, які ви використовуєте. Відміняє необхідності:

```
AllowOverride FileInfo OptionsIn
```

Крім того, переконайтеся, що ваша версія Apache був зібраний з модулем mod_rewrite, або що mod_rewrite Dynamic Shared Object (DSO) був встановлений і наступні рядки у файлі конфігурації Apache:

```
AddModule mod_rewrite.cLoadModule rewrite_module модулі / mod_rewrite.so
```

Коли у вас виникли проблеми з отриманням рецептів на роботу, це може бути тому, що необхідно перевизначити директивами не зазначені в основний конфігураційний файл Apache або mod_rewrite недоступний.

Тестування. Htaccess розбір

Щоб перевірити ваш сервер буде розбирати. Htaccess файли в заданій директорії, завантажувати URL, який звертається до файлу в каталозі, який ви хочете перевірити в веб-браузері і переконайтеся, що файл подається неправильно. Наступний створити (або змінити). Htaccess файл у цьому каталозі, який містить наступний рядок:

```
InvalidDirective Here
```

Перезавантажити URL в адресному рядку браузера. Якщо сервер розбору.

Htaccess файлів у цьому каталозі, сервер повинен повернути сторінці "Internal Server Error" породжених unparsable

InvalidDirective. Якщо сторінка перемальовує нормально, то вам потрібно попросити свого системного адміністратора включити. Htaccess файли в директорії, які ви будете використовувати з рецептами. Коли ви переконаєтеся, що сервер розбору Вашого. Htaccess файли, не забудьте видалити InvalidDirective лінію. Htaccess файл у цьому каталозі.

Mod_rewrite встановлений для тестування

Для перевірки mod_rewrite встановлено, замінити InvalidDirective рядок з першого тесту з цим:

```
RewriteEngine On
```

Перезавантажити URL в адресному рядку браузера. Якщо mod_rewrite не встановлено, або ви не маєте достатньо прав, щоб нехтувати її. Htaccess в цьому каталозі, то сервер повинен повернути сторінці "Internal Server Error". Директива AllowOverride повиненна бути встановлена у FileInfo (або всі), щоб дозволити Вам необхідно включити mod_rewrite в. Htaccess файлу.

Установка RDF / XML типу вмісту

Відповідний тип вмісту для відбування RDF / XML змісту додаток '/ RDF + XML ", як визначено в RFC3870. Сервер Apache можна налаштувати на розпізнавання файлів з розширенням. RDF і обслуговувати їх з відповідним типом змісту, додавши наступну директиву. Htaccess файлу:

```
AddType застосування / RDF + XML. Rdf
```

Ці директиви також можуть застосовуватися на глобальному рівні сервера у файлі конфігурації сервера. У цьому випадку воно може бути безпечно вилучено з рецептів.

Альтернативи правила перезапису

Приклад конфігурацій, описаних в цьому документі, здійснювати узгодження змісту з допомогою правила перезапису (mod_rewrite) для перенаправлення запитів на

найбільш відповідне подання. Цей механізм простий і дозволяє більшу гнучкість щодо розташування HTML і RDF / XML документів, які можуть навіть знаходитися в різних серверах.

Є декілька випадків, в яких умови приклад конфігурації не належним чином регулювати зміст переговорів. Зокрема, що запит з "Приймати:" заголовок, який містить Q-значення не можуть бути розібрані до правил перезапису, а сервер може реагувати з поданням відрізняється від переважно. Хоча це, безумовно, можливо настроїти сервер Apache обробляти такі запити, ця конфігурація не може бути виконана в рамках обмежень. Htaccess файлів. Це виходить за рамки цього документа, щоб забезпечити повний приклад конфігурації сервера для обробки цих запитів. Зацікавлені читачі, можливо, забажає звернутися до змісту переговорів з типом карт в Apache.