Firefox 2の新APIの拡張機能への応用

- Feed Content Access API 編 -

Firefox 2の拡張開発者向け新機能

- MDC (Mozilla Developer Center)
  - 開発者のためのFirefox 2の新機能
    - XUL と拡張の開発者向け
- http://developer.mozilla.org/

プレゼンの内容

- （1）フィードプレビュー機能
  - 機能の概要
  - 仕組み
- （2）フィードコンテントアクセスAPI
  - nsIFeedProcessor
  - nsIFeed, nsIFeedEntry…
- （3）拡張機能への応用
  - 独自のフィードビューアを作る
  - 独自のフィードビューアを登録する
  - Sage風のサイドバーとフィードの更新チェック機能を作る

（1）フィードプレビュー機能

フィードプレビュー機能

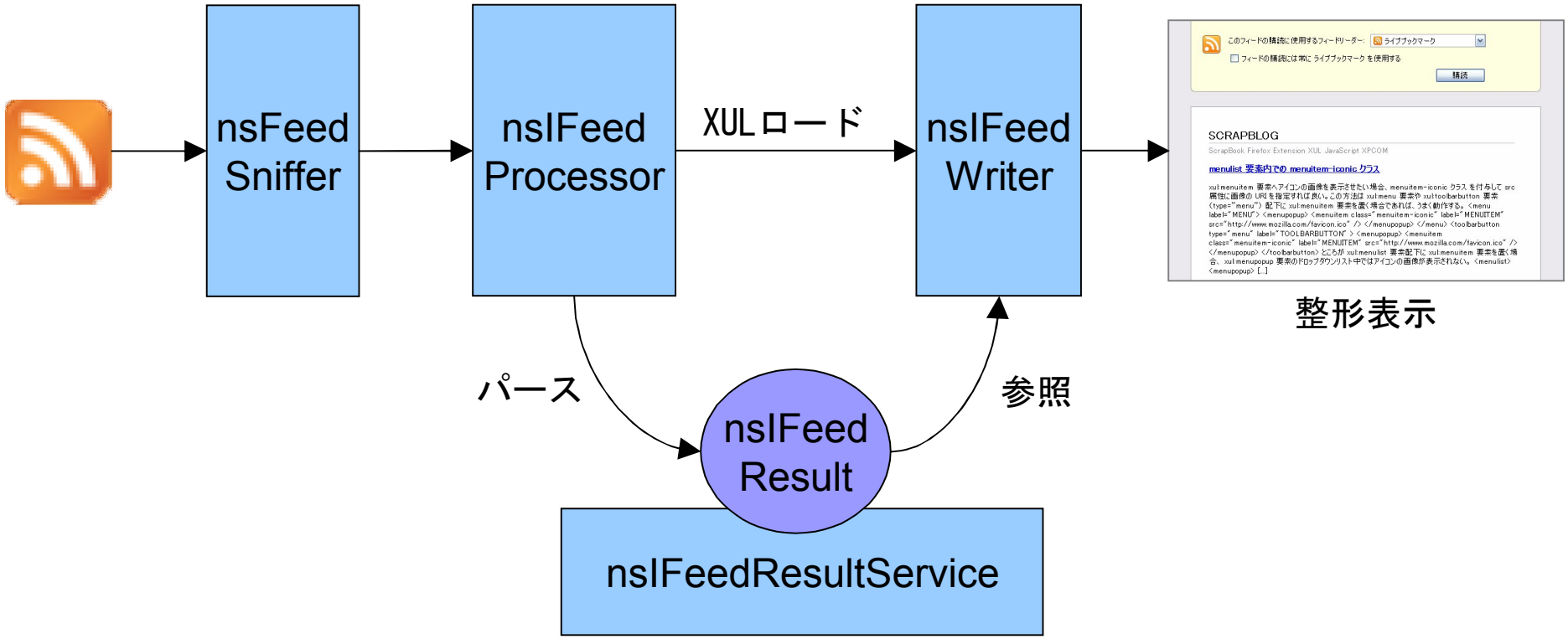
- ブラウザでフィードを読み込むと…
- 自動的にパースして整形表示する
- 購読するためのUIも備える
  - ライブブックマーク
  - 各種Webサービス
  - プログラム


このフィードの購読に使用するフィードリーダー:
ライブブックマーク
フィードの購読には常に ライブブックマーク を使用する
購読
SCRAPBLOG
ScrapBook Firefox Extension XUL JavaScript XPCOM
menulist 要素内での menuitem-iconic クラス
xul:menuitem 要素へアイコンの画像を表示させたい場合、menuitem-iconic クラス を付与して src 属性に画像の URI を指定すれば良い。この方法は xul:menu 要素や xul:toolbarbutton 要素 (type="menu") 配下に xul:menuitem 要素を置く場合であれば、うまく動作する。<menu label="MENU"><menupopup><menuitem class="menuitem-iconic" label="MENUITEM" src="http://www.mozilla.com/favicon.ico" /></menupopup></menu><toolbarbutton type="menu" label="TOOLBARBUTTON" ><menupopup><menuitem class="menuitem-iconic" label="MENUITEM" src="http://www.mozilla.com/favicon.ico" /></menupopup></toolbarbutton> ところが xul:menulist 要素配下に xul:menuitem 要素を置く場合、xul:menupopup 要素のドロップダウンリスト中ではアイコンの画像が表示されない。<menulist><menupopup> [..]

疑問

フィードをロードしてから、
フィードプレビューで
整形表示されるまでの間、
何が起こっているのか？

フィード整形表示までの流れ（イメージ）



フィード整形表示までの流れ（1/3）

- ストリーム読み込み開始
  - Content-type: application/rss+xml
  - Content-type: application/atom+xml
  - 誤ったMIMEタイプがセットされたフィードのデータ

  ↓
- nsFeedSniffer登場
  - データがフィードであることを判定
  - 内部的なContent-typeをセット
    「application/vnd.mozilla.maybe.feed」

  ↓

フィード整形表示までの流れ(2/3)

- ストリームの変換(パース)
  - nsIFeedProcessorが非同期にパース
  - 最終的にnsIFeedResultオブジェクトが生成

  ↓
- nsIFeedResultServiceでいったん保存
  - nsIFeedResultオブジェクトを、
  - nsIFeedResultServiceへ追加
  - グローバルな参照が可能になる

  ↓

フィード整形表示までの流れ（3/3）

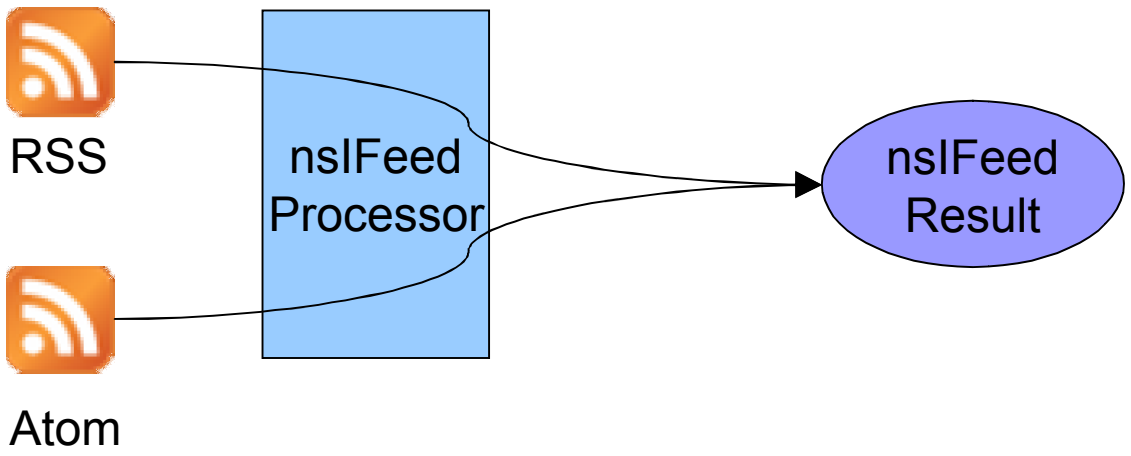
- XULのロード
  - URI「about:feeds」でチャネルを開く
  - その実体はXUL+XHTML（subscribe.xhtml）
  - ↓
- XHTMLで整形表示
  - nsIFeedWriterが、
  - nsIFeedResult型オブジェクトから、
  - 様々なフィード情報を取り出して、
  - XHTMLで整形表示

（2）フィードコンテントアクセスAPI

フィードのデータをパースする

nsIFeedProcessor

- フィードのXMLを非同期にパースする
- パース完了時にリスナへ通知される
- RSSやAtomといった形式の差異を吸収



3つのパース方法

- parseFromString
  - 文字列からパースする
- parseFromStream
  - ストリームからパースする
- parseAsync
  - 非同期なストリームからパースする
  - nsIStreamListenerを介する

パース結果の受け取り方

- パース完了時、リスナが呼ばれる
  - nsIFeedResultListener::handleResult()
- パース結果はnsIFeedResult型オブジェクト

パース結果からフィード情報へアクセス

- nsIFeedResult::doc
- nsIFeed型オブジェクトを取得

```
handleResult: functioon(aFeedResult)
{
      var feed = aFeedResult.doc
                  .QueryInterface(Components.interfaces.nsIFeed);
      ...
},
```

フィード情報へのアクセス

フィードのメタ情報を取得

- nsIFeedContainerから取得できる情報
  - フィードのタイトル
  - フィードのホームページ
  - フィードの最終更新日　　など
- nsIFeedから取得できる情報
  - フィードのサブタイトル
  - フィードの画像（RSS2.0のimageタグ）　　など

フィードのタイトルを取得

- nsIFeed::title
- nsIFeedTextConstruct型オブジェクトを返す
  ↓
- nsIFeedTextConstruct::text
  - マークアップされている場合にタグも含んだ文字列を取得
  - `var feedTitle = feed.title.text;`
- nsIFeedTextConstruct::plainText()
  - マークアップされている場合にタグを除いた文字列として取得
  - `var feedTitle = feed.title.plainText();`
- nsIFeedTextConstruct::createDocumentFragment()
  - マークアップされている場合にDocumentFragmentとして取得
  - `var feedTitle = feed.title.createDocumentFragment(node);`

フィードの更新日を取得

- nsIFeedContainer::updated
- RFC822 形式の文字列を返す
- そのままDateオブジェクトを生成することが可能。

```
var feedUpdated = new Date(feed.updated);
```

フィードの画像を取得

- nsIFeedContainer::fieldsを使用

```
var feedImage = feed.fields.getProperty("image")
    .QueryInterface(Components.interfaces.nsIWritablePropertyBag2);

// 画像のURL
feedImage.getPropertyAsAString("url");

// リンク先URL
feedImage.getPropertyAsAString("link");

// タイトル
feedImage.getPropertyAsAString("title");
```

フィードエントリ情報へのアクセス

フィードエントリの配列を取得

- nsIFeed::items
- エントリ(nsIFeedEntry)の配列を返す
- nsIArray::queryElementAt()を使って、個々のエントリへアクセス

```
var entries = feed.items;
for (var i = 0; i < entries.length; i++)
{
   var entry = feed.items.queryElementAt(i,
                Components.interfaces.nsIFeedEntry);
}
```

フィードエントリの情報を取得

- nsIFeedContainerから取得できる情報
  - エントリのタイトル
  - エントリのパーマネントリンク
  - エントリの筆者
  - エントリのカテゴリ　など
- nsIFeedから取得できる情報
  - エントリの本文
  - エントリの要旨
  - エントリの発行日　など

エントリのパーマネントリンクを取得

- nsIFeedContainer::link
- nsIURI型オブジェクトを返す

```
var parmaLink = entry.link.spec;
```

エントリの要旨や本文を取得

- nsIFeedContainer::summary
- nsIFeedContainer::content
- nsIFeedTextConstruct型オブジェクトを返す

```
// 要旨
var summary = entry.summary;
// 本文
var content = entry.content;
// どちらか一方についてDocumentFragmentを生成する
var docFrag = (content || summary).createDocumentFragment(node);
```

エントリの筆者を取得

- nsIFeedContainer::authors
- 筆者(nsIFeedPerson型オブジェクト)の配列を返す
  ↓
- nsIFeedPerson::nameから名前を取得
- nsIFeedPerson::emailからE-mailアドレスを取得
- nsIFeedPerson::uriからホームページURLを取得

```
for (var i = 0; i < entry.authors.length; i++)
{
        var author = entry.authors.queryElementAt(i,
                     Components.interfaces.nsIFeedPerson);
        var authorName = author.name;
        var authorMail = author.email;
        var authorURI  = author.uri.spec;
}
```

エントリのカテゴリを取得

- nsIFeedContainer::categoriesは現在未実装
- 代わりにnsIFeedContainer::fieldsを使用

```
// 「dc:subject」タグから取得する場合
aEntry.fields.getProperty("dc:subject");

// 「category」タグから取得する場合
var categories = entry.fields.getProperty("categories");
categories.QueryInterface(Components.interfaces.nsIArray);
for (var i = 0; i < categories.length; i++)
{
        var category = categories.queryElementAt(i,
                        Components.interfaces.nsIPropertyBag);
        // カテゴリ名称を取得
        var term = category.getProperty("term");
}
```

（3）拡張機能への応用

第一段階

独自のフィードビューアを作る

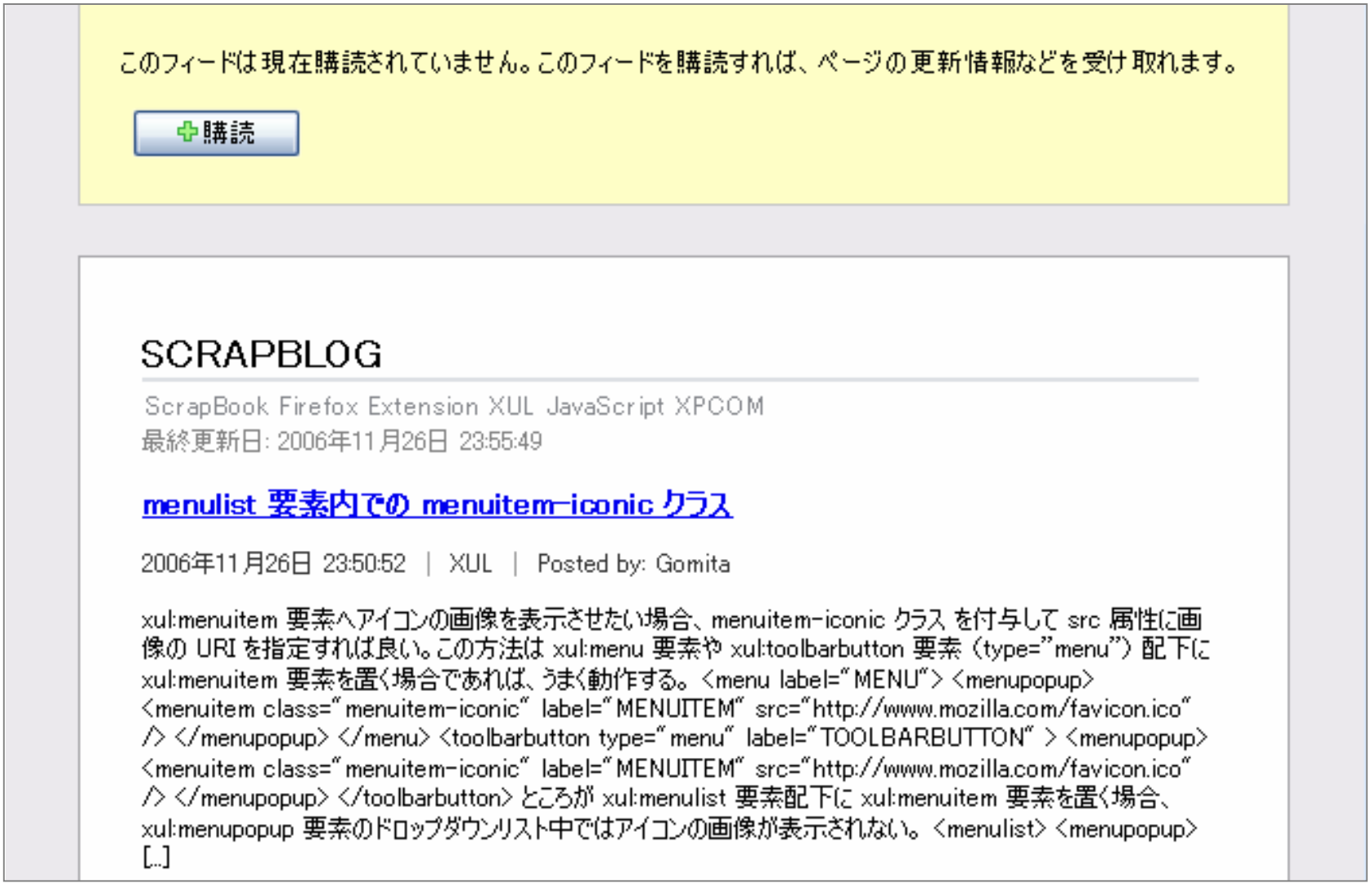
独自のフィードビューア

- 標準のフィードプレビュー機能は物足りない

そこで…

- 独自のフィードビューアをXULで作る
  - 要旨ではなく本文を表示をしたい
  - エントリのカテゴリ名や筆者を表示したい
  - 既読のエントリはグレーで区別したい
  - スタイルシートでカスタマイズ可能したい

完成品のイメージ


このフィードは現在購読されていません。このフィードを購読すれば、ページの更新情報などを受け取れます。
購読
SCRAPBLOG
ScrapBook Firefox Extension XUL JavaScript XPCOM
最終更新日: 2006年11月26日 23:55:49
menulist 要素内での menuitem-iconic クラス
2006年11月26日 23:50:52 | XUL | Posted by: Gomita
xul:menuitem 要素へアイコンの画像を表示させたい場合、menuitem-iconic クラス を付与して src 属性に画像の URI を指定すれば良い。この方法は xul:menu 要素や xul:toolbarbutton 要素 (type="menu") 配下に xul:menuitem 要素を置く場合であれば、うまく動作する。<menu label="MENU"> <menupopup> <menuitem class="menuitem-iconic" label="MENUITEM" src="http://www.mozilla.com/favicon.ico" /> </menupopup> </menu> <toolbarbutton type="menu" label="TOOLBARBUTTON" > <menupopup> <menuitem class="menuitem-iconic" label="MENUITEM" src="http://www.mozilla.com/favicon.ico" /> </menupopup> </toolbarbutton> ところが xul:menulist 要素配下に xul:menuitem 要素を置く場合、xul:menupopup 要素のドロップダウンリスト中ではアイコンの画像が表示されない。<menulist> <menupopup>
[..]

おおまかな処理の流れ

- XULをロード
  ↓
- XMLHttpRequestでフィードのURLへ要求
  ↓
- 回答されたXMLを文字列として取得
  ↓
- nsIFeedProcessor::parseFromString()でパース
  ↓
- nsIFeedResultオブジェクトからフィードやエントリの各種情報を取得
  ↓
- XHTMLにて整形表示

実際に完成品を見てみよう

さらに…

スタイルシートを変更すると
Internet Explorer 7 風になる

SCRAPBLOG

ScrapBook Firefox Extension XUL JavaScript XPCOM

最終更新日: 2006年11月26日 23:55:49

ツール

- ソースの表示
- すべて表示
- 未読のみ表示

menulist 要素内での menuitem-iconic クラス

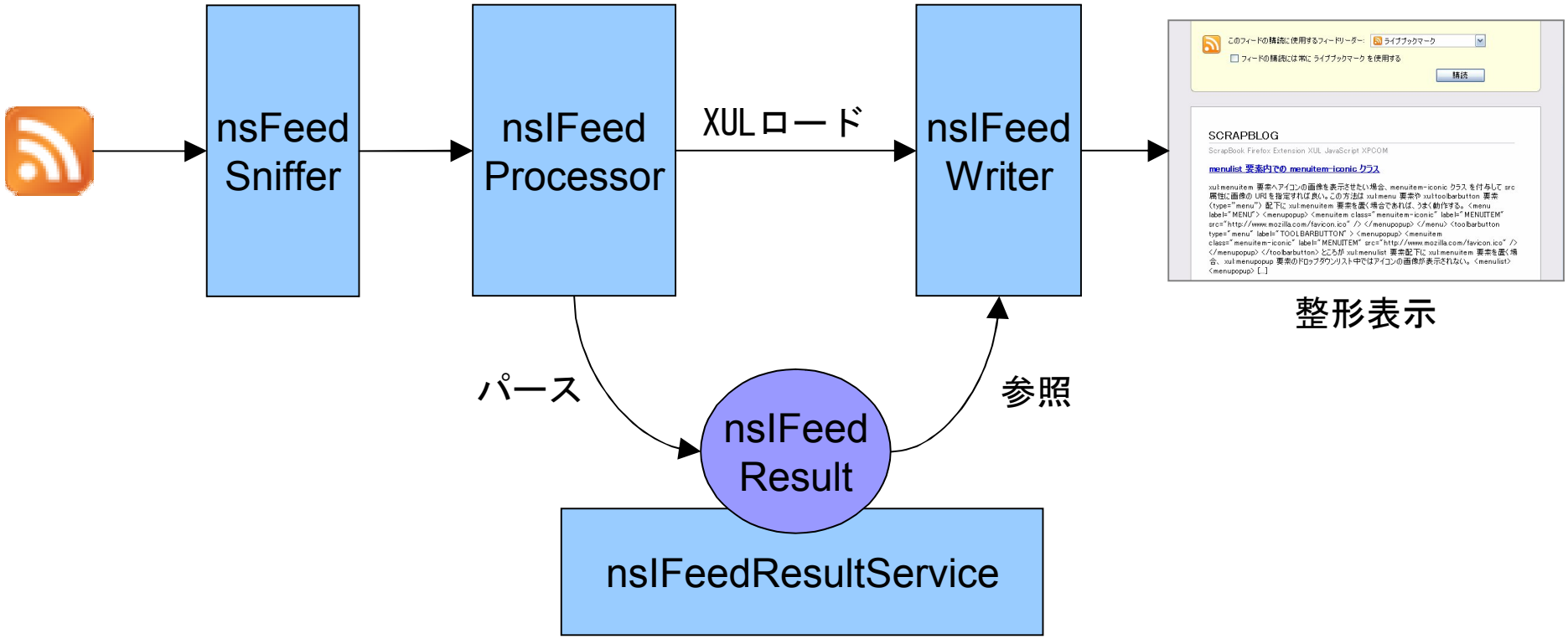
2006年11月26日 23:50:52 | XUL | Posted by: Gomita

xul:menuitem 要素へアイコンの画像を表示させたい場合、menuitem-iconic クラス を付与して src 属性に画像の URI を指定すれば良い。この方法は xul:menu 要素や xul:toolbarbutton 要素 (type="menu") 配下に xul:menuitem 要素を置く場合であれば、うまく動作する。<menu label="MENU"><menupopup><menuitem class="menuitem-iconic" label="MENUITEM" src="http://www.mozilla.com/favicon.ico" /></menupopup></menu><toolbarbutton type="menu" label="TOOLBARBUTTON" ><menupopup><menuitem class="menuitem-iconic" label="MENUITEM" src="http://www.mozilla.com/favicon.ico" /></menupopup></toolbarbutton> ところが xul:menulist 要素配下に xul:menuitem 要素を置く場合、xul:menupopup 要素のドロップダウンリスト中ではアイコンの画像が表示されない。<menulist><menupopup> [..]

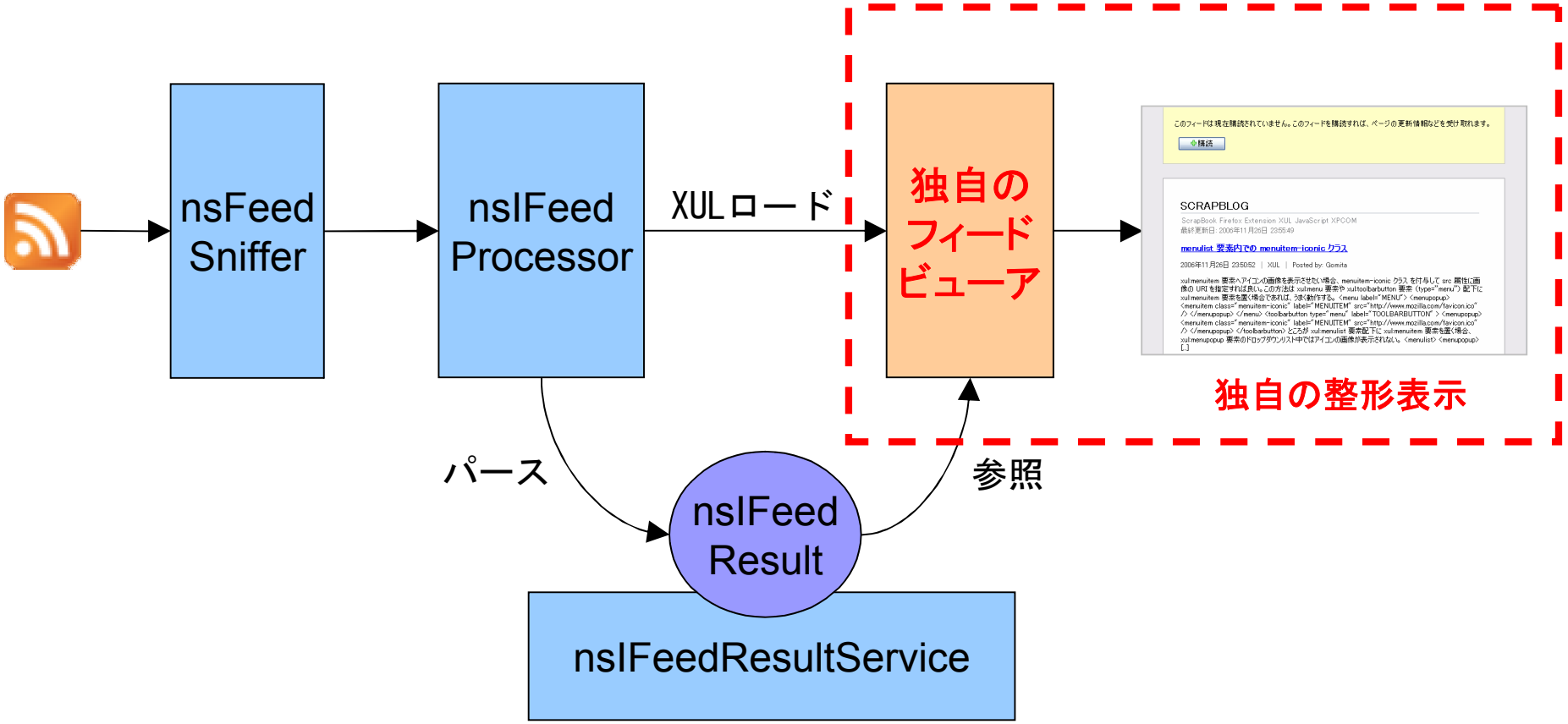
第二段階

独自のフィードビューアを
フィードリーダとして
登録する

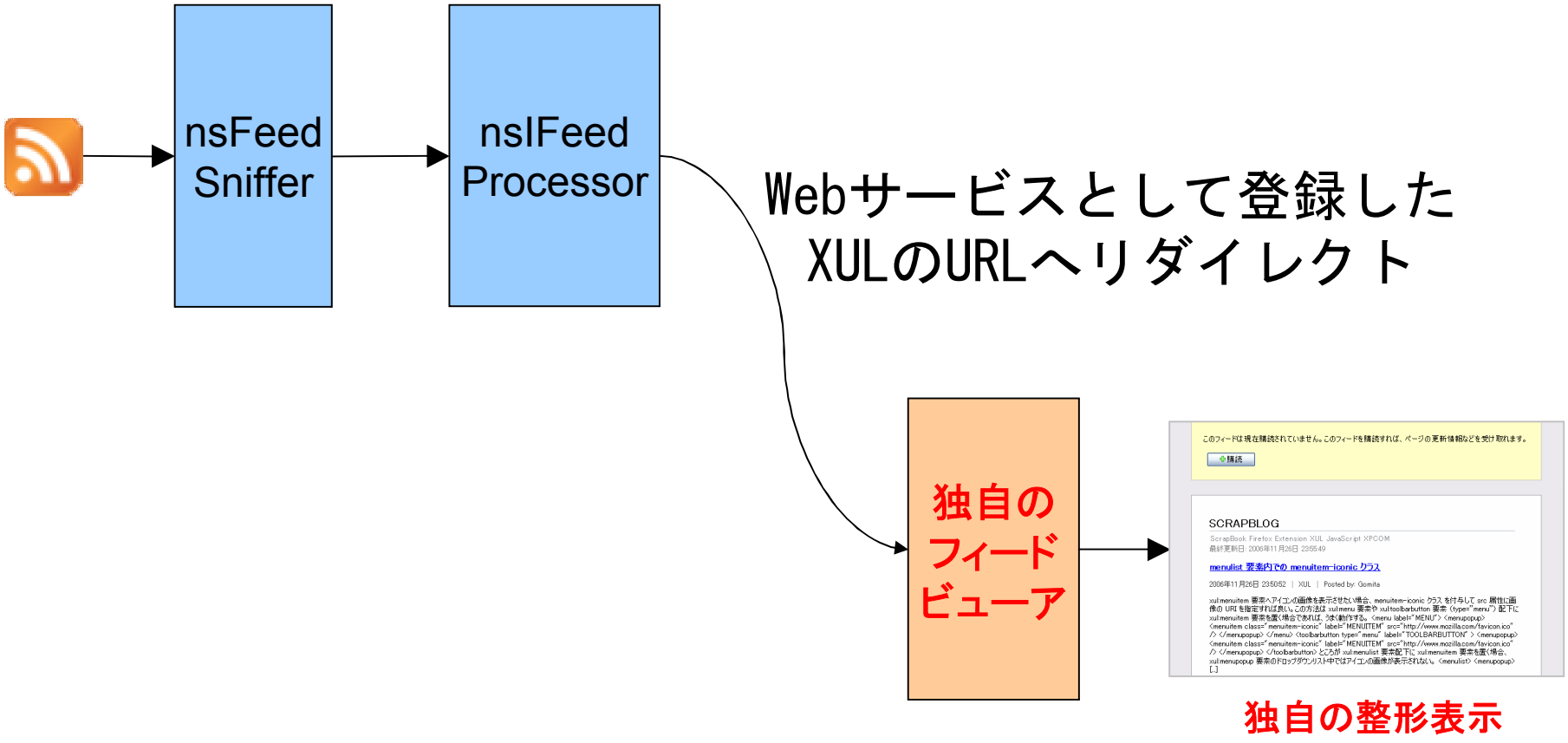
もう一度おさらい


nsFeed Sniffer
nsIFeed Processor
XULロード
nsIFeed Writer
パース
nsIFeed Result
参照
nsIFeedResultService
SCRAPBLOG
整形表示

本当はこうしたかった…


nsFeed Sniffer
nsIFeed Processor
XULロード
独自のフィードビューア
SCRAPBLOG
独自の整形表示
パース
参照
nsIFeed Result
nsIFeedResultService

でも、こうするしかなさげ…



まずは準備

- 「exfeed:」プロトコルで、
  フィードビューアを起動できるようにする
  - exfeed:http://www.example.com/rss.xml

独自プロトコルの実装

- XPCOMのクラスを定義
  - @mozilla.org/network/protocol;1?name=exfeed
- nsIProtocolHandlerインタフェースを実装
  - nsIProtocolHandler::newChannel()
    XULのURI（chrome://…）でチャネルを開く
  - nsIURI::originalURI
    ロケーションバーのURL表示を「http://...」へ変更

実際にexfeedプロトコルを試してみよう

フィードリーダの選択肢に加える

- 以下の3つの設定値
- 新しいフィードリーダの候補を追加可能
  - browser.contentHandlers.types.6.title
    =「Extended Feed Viewer」
  - browser.contentHandlers.types.6.type
    =「application/vnd.mozilla.maybe.feed」
  - browser.contentHandlers.types.6.uri
    =「exfeed:%s」


このフィードの購読に使用するフィードリーダー:
ライブブックマーク
フィードの購読には常に ライブブックマーク を
ライブブックマーク
アプリケーションを選択...
My Yahoo!
Google
Bloglines
はてな RSS
livedoor Reader
goo RSS リーダー
Extended Feed Viewer
えむもじら
Mozilla,Firefox,Thunderbird関連の情報を提供して
リマインダ:Firefox Developers Conference
Firefox Developers Conference 2006 は土曜日で

独自のフィードビューアを常用する

ようやく…

標準のフィードプレビューは
独自のフィードビューアへと
置き換わった。

では、実際にフィードを表示させてみよう。

第三段階

Sage風サイドバーと更新チェック機能を作る

サイドバーの特長

- ブックマーク統合型
- 同時に4つの更新チェック
- 「未読あり」を強調表示
- フィードの未読・既読状態はmozStorageで別管理

実際にサイドバーを動かしてみよう