OCamlDuceで RSSリーダを

水野 洋樹

RSSって何?

- 更新情報を配信するアレ
- いろんな種類がある。
  - RSS 1.0, RSS 2.0, Atomなどなど...
- 単純化のため、偽RSSを使う

偽RSSの例

```
<rdf>
  <item>
    <title>less is more</title>
    <link>http://example.com/a</link>
    <description>...</description>
  </item>
 <item>
    <title>keep it simple</title>
    <link>http://example.com/b</link>
    <description>...</description>
  </item>
</rdf>
```

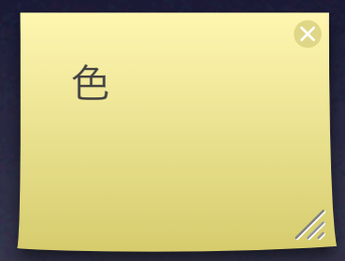
OCamlDuceって何?

- OCaml + CDuce = OCamlDuce
- CDuce
  - XMLを扱える関数型言語
  - OCamlに似ているけど別物
- 追加された機能は{{..}}のように書く

構文上の理由

サンプルデータ

```
let rss = {{ <rdf>[
              <item>[
                <title>"less is more"
                <link>"http://example.com/a"
                <description>"..."
              ]
              <item>[
                <title>"keep it simple"
                <link>"http://example.com/b"
                <description>"..."
              ]
            ]}};;
```



値の書き方

- 例: <foo attr1=”a” attr2=”b”>[1 2 3]
- [...]はシーケンス。リストではない
- 文字列は文字のシーケンス
  - ”abc”=[‘a’ ‘b’ ‘c’]
  - <p>”hoge”のように書ける

ヘテロジーニアスなリスト

RSSの型

```
type rdf = {{ <rdf>
              [<item>[
              <title>String
              <link>String
              <description>String ]+
    ]
}};;
```

型の書き方

- 基本的に値の書き方と似てる
- Int,Stringなどと大文字で始まる

[Int] != int list
[Int*] == int list

- 正規表現が使える
  - [Int*] : Intが任意個続くシーケンス
  - R* , R+ , R? , R1 | R2など
  - _だって使える

itemの取り出し

```
let items (rss : {{rdf}}) : {{[item*]}} =
  match rss with
    {{<rdf>items}} -> items;;

(* 別の方法 *)
let items (rss:{{rdf}}) =
  let {{<rdf>items}} = rss in
    items;;
```

入力の型宣言がかならずいる

実行例

```
# items rss;;
- : {{[ item* ]}} =
{{[ <item>[
      <title>[ 'less is more' ]
      <link>[ 'http://example.com/a' ]
      <description>[ '...' ]
    ]
  <item>[
    <title>[ 'keep it simple' ]
    <link>[ 'http://example.com/b' ]
    <description>[ '...' ]
    ]
  ]}}
```

XMLパターン

- OCmalでパターンが使えるところならどこでも使える、らしい
- 基本的に値の書き方と同じ
- 正規表現が使える
- 欲張りじゃない正規表現もある
  - *?,+?など

itemの変換

- 変換にはmap ... withが便利

```
let trans (rss:{{rdf}}) =
  let xs = items rss in
   {{<body>(
     map xs with
       <item>[<title>title <link>url _*]
        ->
       [<p>[<a href=url>title]]) }};;
```

実行例

```
# trans rss;;
- : {{<body>[ <p>[ <a href=String>String ]* ]}} =
{{<body>[
   <p>[ <a href="http://example.com/a">[ 'less is more' ] ]
   <p>[ <a href="http://example.com/b">[ 'keep it simple' ] ]
   ]}}
```

変換したXMLの出力

- 出力は Ocamlduce.Print.print_xmlを使う
- (string -> unit) -> Ocamlduce.Load.anyxml -> unit

```
let _ =
  Ocamlduce.Print.print_xml
    print_string
    (trans rss);;
```