テキスト処理 第10回 (2006-06-27)
treemapレポート説明

田中哲
産業技術総合研究所
情報技術研究部門
akr@isc.senshu-u.ac.jp
http://staff.aist.go.jp/tanaka-akira/textprocess/

レポート

- map と同様に要素にブロックを適用するが、フラットとは限らずネストしているかもしれない配列を扱うメソッド treemap を定義せよ
- def treemap(obj) ... end
- ユニットテストを提供するので、実装したらテストして確認すること
- ∠切 2006-06-27 16:20
- IT's class
- 拡張子が txt なテキストファイルを望む

実装1

```
def treemap(obj, &b)
  if obj.respond_to? :each
    result = []
    obj.each {|v| result << treemap(v, &b) }
    result
  else
    yield obj
  end
end
```

配列だったら各要素について
再帰して結果をまとめる

配列でなかったら
b に通しておしまい

実装2

```
def treemap(obj, &b)
  if obj.respond_to? :map
    obj.map {|v| treemap(v, &b) }
  else
    yield obj
  end
end
```

配列だったら
(map があったら)
map を使う

他の案

- yield でなく b.call を使う

Duck Type

- each や map の存在を確かめるのは厳密には配列かどうか検査しているわけではない
- メソッドの存在で調べると、メソッドが存在すれば、配列でなくても動くという利点がある
- メソッドの存在でオブジェクトを分類するやりかたを、Duck Type という
- アヒルのように鳴くならそれはアヒルである、というイメージ
- メソッド名と動作の一貫性が必要
  - 同じメソッド名で違う動作をすると困る
  - 違うメソッド名で同じ動作をすると困る

ざっと眺めた結果

- できている率がかなりあがった感じ
- treemap(1) {lvl v} が [1] になるような、入力と出力がずれているのがあった
  - 再帰の終端 (今回は配列でない場合) のケースから考えて記述する
- ぜんぶの要素をたどらないといけないので treemap は break しない