CS1 (6): 状態を更新する機能

脇田建

2014.11.11

Designing Functions with Memory (HtDP 36)

記憶する機能の設計のステップ

- あなたのプログラムはなにかを記憶するのか？
- もし記憶するのなら
  - どのような内容を記憶するのか？
  - プログラムのどの機能がその記憶に関わるのか？

1. 記憶の必要性

- すべての人がプログラミングに熟達していたら記憶は不要かも
  - 「記憶」を関数への引数として渡せばよいから
- でも，みんなが Scala のインタプリタを利用できるわけじゃない

記憶機能が必要となる場合

1. システムが複数の機能を提供する場合（例：住所録）

2. 提供する機能はひとつだが，状況の応じて振舞いが変化する場合（例：信号機）

例１：住所録（複数の機能）

- 住所録へのデータの追加の機能
- 住所録の検索の機能

- この場合の記憶はなに？

- この場合の機能はなに？

例２：クローク（複数の機能）

- 預入：荷物を渡すとタグを返す
- 返却：タグを渡すと荷物を返す

- この場合の記憶はなに？
- この場合の機能はなに？

例３：信号機（履歴に応じた振舞い）

- 赤 → 青 → 黄 → 赤 → 青 → 黄 → 赤 → 青 → 黄 → …

- この場合の記憶は？

- この場合の機能は？

例４：乱数 (scala.math.random)

- random: Double → [0, 1)

```
1. Default
Default | Default
scala>

scala> import scala.math.random
import scala.math.random

scala> List(random, random, random)
res4: List[Double] = List(0.3640285336626812, 0.3417481259521269, 0.6535961091970731)
```

- この場合の記憶は？

- この場合の機能は？

2. 状態変数の同定

- 「記憶」は Scala の**可変変数**によって実装される.

この「可変変数」は *var* 宣言されたものを指している.

val 宣言された**定数**や関数の**引数**と混同しないこと.

In principle, a single variable is enough to implement all memory needs, ...

- 「変数がひとつあれば，十分」(HtDP 36.2)
  - var v1; var v2; ... のかわりに var state = (v1, v2, ...) で代用できるから．
    - とはいえ，これは不自然．自然に表現可能な変数を割り出すことが大切
- 一方で，無闇に多くの変数を導入することは，混乱のもと
- システムの状態をできるだけ簡潔に説明できるものを考えることが大切

住所録についてのデータ解析

住所録を表現する変数

- 住所録の内容
  - Adam: 014-1421-356
  - Eve: 017-3405-08
- こんな内容を表現するのに相応わしいデータ型は？

住所録を表現する変数

- 住所録の内容
  - Adam: 014-1421-356
  - Eve: 017-3405-08
- こんな内容を表現するのに相応わしいデータ型は？
  - List[Symbol, String]

住所録を表現する変数

- 住所録の内容
  - Adam: 014-1421-356
  - Eve: 017-3405-08
- こんな内容を表現するのに相応わしいデータ型は？
  - List[Symbol, String]
  - Map[Symbol, String]

住所録の初期値は？

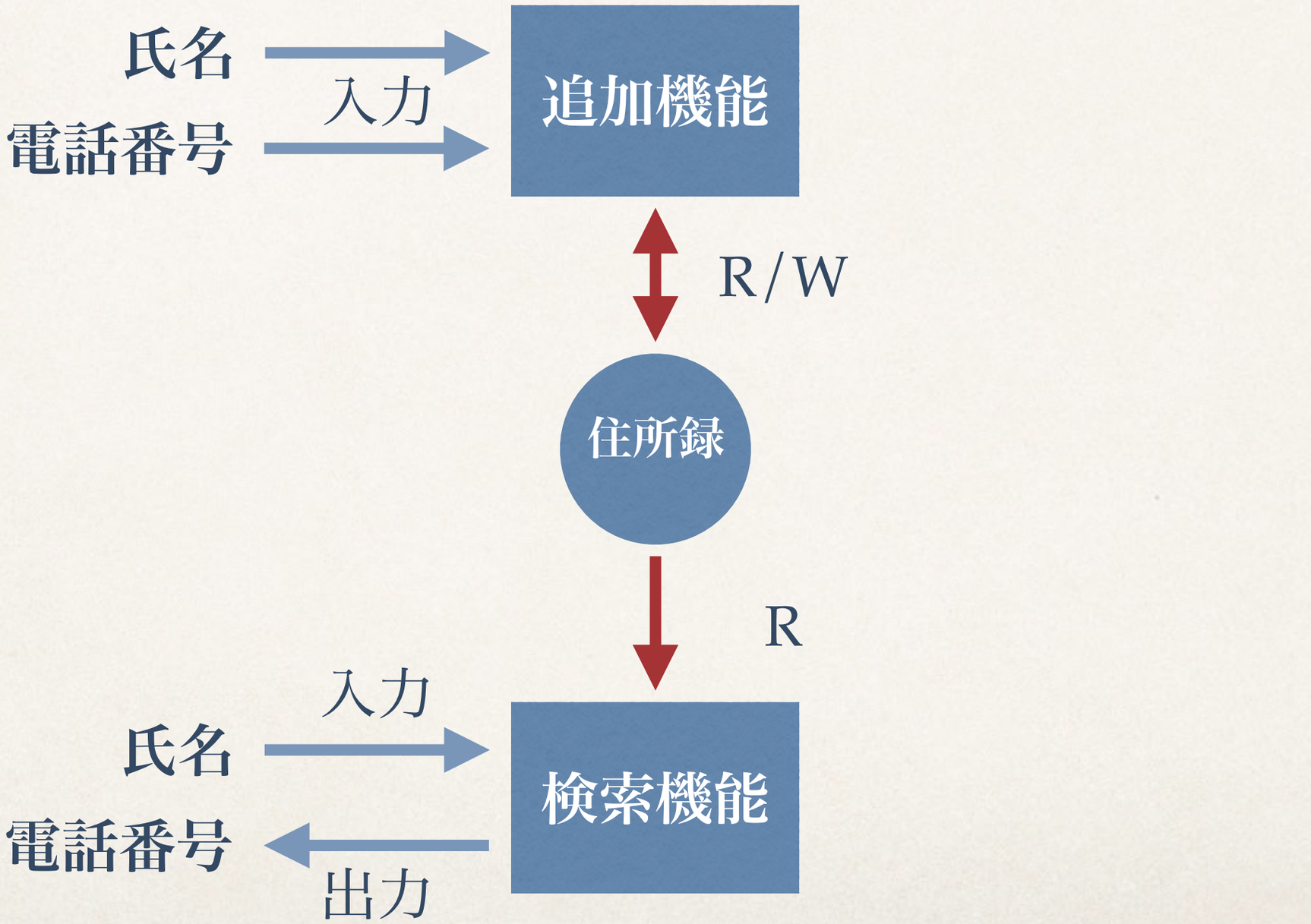
- type AddressBook = List[Symbol, String] のとき

- どんな初期値が適切か考えよう

住所録へはどのような代入が考えられるか？

初期状態	[]
[]に (人2, 電話2) を追加	[('人2, 電話2)]
[('人1, 番号1), …] に (人2, 電話2) を追加	[('人2, 電話2), ('人1, 電話2), …]

住所録を操作する機能の分析



信号機についてのデータ解析

信号機の色を表現する変数

- 信号の状態
  - 赤 → 青 → 黄 → ...
- ある時点での信号: 赤 or 青 or 黄
- こんな内容を表現するのに相応わしいデータ型は？

信号機の色を表現する変数

- 信号の状態
  - 赤 → 青 → 黄 → ...
- ある時点での信号: 赤 or 青 or 黄
- こんな内容を表現するのに相応わしいデータ型は？
  - abstract class Color
    - case class Red() extends Color
    - case class Green() extends Color
    - case class Yellow() extends Color

信号機の色を表現する変数

- 信号の状態
  - 赤 → 青 → 黄 → ...
- ある時点での信号: 赤 or 青 or 黄
- こんな内容を表現するのに相応わしいデータ型は？
  - trait TrafficLight
    - object Green extends TrafficLight
    - object Yellow extends TrafficLight
    - object Red extends TrafficLight

信号機の色を表現する変数

- 信号の状態
  - 赤 → 青 → 黄 → ...
- ある時点での信号: 赤 or 青 or 黄
- こんな内容を表現するのに相応わしいデータ型は？
  - trait TrafficLight
    - Green = 0
    - Yellow = 1
    - Red = 2

信号機の色の初期値は？

- 安全な初期値の原則

"In setting current_color to 'red, we follow a conventional rule of engineering to put devices into their least harmful state when starting it up."

–HtDP 36.3 Functions that initialize memory

信号機ではどのような代入が考えられるか？

状態	次の状態
初期状態	赤
赤	青
青	黄
黄	赤

信号機を操作する代入の分析は？

信号機を表す変数への代入

- current_color = Red()
- current_color = Green()
- current_color = Yellow()

状態更新を伴う機能のデザインレシピ

- データ解析
- 契約 (Contract) / 入出力の働き (Purpose) / 副作用 (Effect)
- 例 (Program Examples) → テストケース
- 雛形 (The Template)
- 定義本体 (The Body)