プログラミング言語論 #14

Practices

2018-07-23


OUTLINE JavaScript での部分適用の実例

部分適用

Haskell	自動的．全ての関数は高階関数
C 言語	できない
JavaScript	できない．ただし，変換関数を使うとできる．

JavaScript によるグラフ描画

処理の流れ

1. html と JavaScript のロード終了時にデータ取得関数実行
2. 可視化対象データ 1 をロード：終了時にイベント発生 (2E)
3. 可視化対象データ 2 をロード：終了時にイベント発生 (3E)

```
1  google.setOnLoadCallback(runQuery);    // (1)
2  var q1 = new google.visualization.Query(url1);  // (2)
3  var q2 = new google.visualization.Query(url2);  // (3)
4  q1.send(handleQueryResponse1);          // (2E)のためのハンドラ登録
5  q2.send(handleQueryResponse2);          // (3E)のためのハンドラ登録
```

send の引数は 1 引数関数でなければならない

ロード終了に対応するハンドラ

ハンドラ関数

```
1  function handleQueryResponse1(response) {
2    var data = response.getDataTable();                    // 形式変換
3    var area = document.getElementById("gchart1");         // 描画領域
4    var opts = {"title":"graph1"};                       // タイトル設定
5    var chart = new google.visualization.LineChart(area); // グラフ
6    chart.draw(data, opts);                                  // 描画
7  }
```

handleQueryResponse2 は，3 行目が"gchart2"，4 行目が"graph2".

ほとんど同じ内容を送信するのは無駄：パケットを減らしたい

汎用 3 引数ハンドラ関数

```
1  function handleQueryResponse( e, t, response) {
2    var data = response.getDataTable();
3    var area = document.getElementById( e);
4    var opts = {"title": t};
5    var chart = new google.visualization.LineChart(area);
6    chart.draw(data, opts);
7  }
8  q1.send(handleQueryResponse);          // ???
```

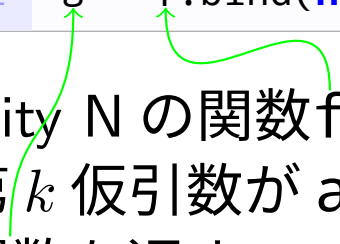
ハンドラは 1 引数関数でなければならない： arity の不一致

JavaScript における部分適用

Haskell では引数の数が足りなければ自動的に部分適用になるが，JavaScript では部分適用実行関数 bind を用いる．

JavaScript での部分適用実行関数 bind

```
1  g = f.bind(null, a1, a2, ..., ak)   // 第1引数のnullはおまじない
```



arity N の関数fに bind を実行させると関数 f の第 1，第 2，$\cdots$ 第 k 仮引数が a1, a2, $\cdots$，ak に部分適用された arity (N-k) の関数を返す

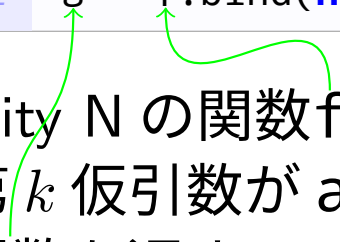
OOPL としての正確な説明では bind は関数ではなく関数クラスが持つメソッドというべき

JavaScript における部分適用

Haskell では引数の数が足りなければ自動的に部分適用になるが，JavaScript では部分適用実行関数 bind を用いる．

JavaScript での部分適用実行関数 bind

```
1  g = f.bind(null, a1, a2, ..., ak)   // 第1引数のnullはおまじない
```



arity N の関数fに bind を実行させると関数 f の第 1，第 2，$\cdots$ 第 k 仮引数が a1, a2, $\cdots$，ak に部分適用された arity (N-k) の関数を返す

OOPL としての正確な説明では bind は関数ではなく関数クラスが持つメソッドというべき

ハンドラの定義は一つだけ：パケット最小

```
1  q1.send(handleQueryResponse.bind(null, "gchart1", "graph1"));
2  q2.send(handleQueryResponse.bind(null, "gchart2", "graph2"));
3  q3.send(...
```

追補

- 別案 1：グローバル変数でタイトルを指定する→ × どちらのデータが先に到着するか不明
- 別案 2：高階関数を利用
  - 2 つの文字列を受け取ると，対象領域，タイトルを固定した関数を生成して返す高階関数

    javascript で関数を返す構文（あるいは無名関数）を説明していないので詳細は省略

  ◯ ただし，実はこれは bind を自分で定義することに等しい
- 別案 3： 汎用ハンドラを

```
1  function handleQueryResponse( e, t, response) {
```

ではなく

```
1  function handleQueryResponse(response, e, t) {
```

と定義してしまった場合→ × bind 時に大変困る

追補の追補

なので Haskell にはこういう関数が用意されている

ライブラリ関数 flip

```
1  :type flip
2  flip :: (a -> b -> c) -> b -> a -> c
```