

Sequence

佐原伸

2005年12月31日

1 Introduction

SBCalendar ライブラリ。

2 SBCalendar

2.1 責任

証券会社システムの暦クラス。

2.2 概要

国・会社・休日ベース区分を考慮した、トレードワンシステムの暦処理のスタブである。土曜日や年末年始を休日として扱う。

class *SBCalendar* is subclass of *JapaneseCalendar*

values

- 1.0 *Rcsid* = "\$Id: SBCalendar.vpp, v 1.2 2005/11/22 06:54:49 vdmtools Exp \$";
- 2.0 *io* = new *IO* ()

instance variables

- .1 public 基準日という日 : [*Date*] := nil ;
- .2 public 会社基準日写像 : [*char** \xrightarrow{m} *Date*] := { \mapsto } ;
- .3 public システムの時刻 : [*Time*] := nil ;
- .4

functions

public

- 3.0 暦を得る : () \rightarrow *Calendar*
- .1 暦を得る () \triangleq
- .2 **self** ;

public static

- 4.0 限月が正当 : *char** \rightarrow \mathbb{B}
- .1 限月が正当 (限月) \triangleq
- .2 文字列から日付を得る (限月 \curvearrowright "01") \neq false ;

public static

- 5.0 権利行使日を得る : *char** \rightarrow *Date*
- .1 権利行使日を得る (限月) \triangleq
- .2 let 限月の月初日 = 文字列から日付を得る (限月 \curvearrowright "01"),
- .3 指定年 = 限月の月初日.年 (),
- .4 指定月 = 限月の月初日.月 () in
- .5 第 *n* 指定曜日を得る (指定年, 指定月, 2, 金).過去の平日を得る ()
- .6 pre 限月が正当 (限月) ;

2.2.1 信用期日を得る

信用取引の決済日（期日）を得る。弁済期限とは、信用建玉に対して当社がお客様に信用を供与する期限をいいます。弁済期限は、現在のところ6ヶ月のみを取扱っています。弁済期限が6ヶ月であるということは、信用建玉の建日（信用建玉が約定した日）の6ヶ月目応答日が信用期日となり、この日を超えて建玉を保有することは法律で禁じられています。信用期日が休日の場合には、直近の前営業日が信用期日となります。

public static

```
6.0 信用期日を得る : Date → Date
.1 信用期日を得る (aDate) △
.2   let 暦 = new SBCalendar (),
.3     mk-(年, 月) = 暦.正則月を得る (aDate.年 (), aDate.月 () +
6),
.4     月末日 = 暦.月末日を求める (年, 月),
.5     日 = aDate.日 (),
.6     期日候補 =
.7         if 月末日.日 () < 日
.8         then 月末日
.9         else 暦.整数三つ組から日付を得る (年, 月, 日) in
.10    期日候補.過去の平日を得る ()
```

```

.11 pre aDate.平日か?() post let 暦 = new SBCalendar (),
.12     mk-(年, 月) = 暦.正則月を得る (aDate.年 (), aDate.月 ()+
6),
.13     月末日 = 暦.月末日を求める (年, 月),
.14     日 = aDate.日 (),
.15     期日候補 =
.16     if 月末日.日 () < 日
.17     then 月末日
.18     else 暦.整数三つ組から日付を得る (年, 月, 日) in
.19     RESULT. = (期日候補.過去の平日を得る ()) ∧
.20     if  $\forall day \in \{1, \dots, 期日候補.日 ()\}$  .
.21     暦.お休みか? (暦.整数三つ組から日付を得る (年, 月
, day))
.22     then let mk-(-, 前月) = 暦.正則月を得る (年, 月 - 1) in
.23     RESULT.月 () = 前月
.24     else RESULT.月 () = 月;

public static
7.0 日付が空か?: [Date] →  $\mathbb{B}$ 
.1 日付が空か?(日付)  $\triangleq$ 
.2 日付 = nil ;

public static
8.0 システム日付: () → Date
.1 システム日付 ()  $\triangleq$ 
.2 今日 ()

operations
public
9.0 休日集合を設定する:  $\mathbb{Z} \xrightarrow{o} ()$ 
.1 休日集合を設定する (年)  $\triangleq$ 
.2 let 日本の暦 = new JapaneseCalendar (),
.3     日本の休日集合 = 日本の暦.休日集合を得る (年),

```

```

.4      TR1 の休日集合 = {
.5          日本の暦.整数三つ組から日付を得る (
年, 1, 2),
.6          日本の暦.整数三つ組から日付を得る (
年, 1, 3),
.7          日本の暦.整数三つ組から日付を得る (
年, 12, 29),
.8          日本の暦.整数三つ組から日付を得る (
年, 12, 30),
.9          日本の暦.整数三つ組から日付を得る (
年, 12, 31)},
.10     土曜日集合 = 日本の暦.ある年の指定曜日集合を得る (年, 土) in
.11     休日集合写像 := 休日集合写像  $\sqcup$  {年  $\mapsto$  日本の休日集合  $\cup$ 
TR1 の休日集合  $\cup$  土曜日集合};
public
10.0    基準日を読み込む : char*  $\xrightarrow{o}$  [Date]
.1      基準日を読み込む (ファイル名)  $\triangle$ 
.2      let mk- (結果, mk- (y, m, d)) = io.freadval[ $\mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$ ] (ファイル
名) in
.3      if 結果
.4      then return 整数三つ組から日付を得る (y, m, d)
.5      else let - = io.echo ("Can't read BaseDay's data file.") in
.6          return nil ;
public
11.0    基準日 : ()  $\xrightarrow{o}$  Date
.1      基準日 ()  $\triangle$ 
.2      if 基準日という日 = nil
.3      then return 基準日を読み込む (homedir  $\curvearrowright$  "/temp/BaseDay.txt")
.4      else return 基準日という日;
public
12.0    ファイルから読み込む基準日 : char*  $\xrightarrow{o}$  Date
.1      ファイルから読み込む基準日 (ファイル名)  $\triangle$ 
.2      if 基準日という日 = nil
.3      then return 基準日を読み込む (ファイル名)
.4      else return 基準日という日;

```

```

public
13.0 基準日を設定する :  $Date \xrightarrow{o} ()$ 
    .1 基準日を設定する (日付)  $\triangle$ 
    .2 基準日という日 := 日付;
public
14.0 会社基準日 :  $char^* \xrightarrow{o} Date$ 
    .1 会社基準日 (会社コード)  $\triangle$ 
    .2 ( if 会社基準日写像 = nil
    .3     then 会社基準日を設定する(会社コード, 基準日 ());
    .4     return 会社基準日写像 (会社コード)
    .5 );
public
15.0 会社基準日を設定する :  $char^* \times Date \xrightarrow{o} ()$ 
    .1 会社基準日を設定する (会社コード, 日付)  $\triangle$ 
    .2 会社基準日写像 := 会社基準日写像 † {会社コード ↦ 日付 };
public
16.0 システム時刻を読み込む :  $() \xrightarrow{o} [Time]$ 
    .1 システム時刻を読み込む ()  $\triangle$ 
    .2 let mk-(結果, now) = io.freadval[Time] (homedir ^ "/temp/SystemTime.txt") in
    .3 if 結果
    .4 then return now
    .5 else let - = io.echo ("Can't read System Time data file.") in
    .6 return nil ;
public
17.0 システム時刻 :  $() \xrightarrow{o} Time$ 
    .1 システム時刻 ()  $\triangle$ 
    .2 if システムの時刻 = nil
    .3 then システム時刻を読み込む()
    .4 else return システムの時刻;
public
18.0 システム時刻を設定する :  $Time \xrightarrow{o} ()$ 
    .1 システム時刻を設定する (時刻)  $\triangle$ 
    .2 システムの時刻 := 時刻;
public

```

```

19.0 SBCalendar : ()  $\rightarrow$  SBCalendar
.1 SBCalendar ()  $\triangle$ 
.2 ( グリニッジ標準時との差を設定する(日本標準時とグリ
ニッジ標準時との差);
.3 return self
.4 )
end SBCalendar
Test Suite : vdm.tc
Class : SBCalendar

```

Name	#Calls	Coverage
SBCalendar'基準日	3	√
SBCalendar'暦を得る	0	0%
SBCalendar'会社基準日	2	69%
SBCalendar'限月が正当	3	√
SBCalendar'システム日付	1	√
SBCalendar'システム時刻	1	85%
SBCalendar'日付が空か？	2	√
SBCalendar'信用期日を得る	32	32%
SBCalendar'基準日を設定する	2	√
SBCalendar'基準日を読み込む	2	65%
SBCalendar'権利行使日を得る	2	86%
SBCalendar'休日集合を設定する	37	√
SBCalendar'SBCalendar	38	√
SBCalendar'会社基準日を設定する	2	√
SBCalendar'システム時刻を設定する	1	√
SBCalendar'システム時刻を読み込む	0	0%
SBCalendar'ファイルから読み込む基準日	1	80%
Total Coverage		61%