

FJapaneseCalendar ライブラリー

佐原伸日本フィット株式会社

情報技術研究所

TEL : 03-3623-4683

shin.sahara@jfits.co.jp

平成 18 年 1 月 11 日

概 要

日本の暦（カレンダー）に関わる関数を提供するモジュールである。
FCalendar クラスのサブクラスとして実装している。

0.1 FJapaneseCalendar

日本の暦に関わる関数を定義する。

class *FJapaneseCalendar* is subclass of *FCalendar*

values

public

1.0 *DiffBetweenGMTandJST* = 9;

public

2.0 *DiffBetweenSeirekiAndHoureki* = 1988

GetHolidays は、指定した年 *yyyy* (2000 年以降) の日本の休日の集合を返す。

functions

public

3.0 *GetHolidays* : $\mathbb{Z} \rightarrow \text{Date-set}$

.1 *GetHolidays* (*yyyy*) \triangleq

.2 let *Seijin* = *GetNthDayOfWeekOfMonth* (MON) (2) (1) (*yyyy*),

.3 *Umi* =

.4 if *yyyy* \geq 2003

.5 then *GetNthDayOfWeekOfMonth* (MON) (3) (7) (*yyyy*)

.6 else *DateFromInt* (*yyyy*) (7) (20),

.7 *Keirou* =

.8 if *yyyy* \geq 2003

.9 then *GetNthDayOfWeekOfMonth* (MON) (3) (9) (*yyyy*)

.10 else *DateFromInt* (*yyyy*) (9) (15),

.11 *Taiiku* = *GetNthDayOfWeekOfMonth* (MON) (2) (10) (*yyyy*),

.12 *NationalHoliday* = {

.13 *DateFromInt* (*yyyy*) (1) (1),

.14 *Seijin*,

.15 *DateFromInt* (*yyyy*) (2) (11),

.16 *GetVernalEquinoxInJST* (*yyyy*),

.17 *DateFromInt* (*yyyy*) (4) (29),

.18 *DateFromInt* (*yyyy*) (5) (3),

.19 *DateFromInt* (*yyyy*) (5) (4),

.20 *DateFromInt* (*yyyy*) (5) (5),

.21 *Umi*,

.22 *Keirou*,

.23 *GetAutumnalEquinoxInJST* (*yyyy*),

.24 *Taiiku*,

.25 *DateFromInt* (*yyyy*) (11) (3),

.26 *DateFromInt* (*yyyy*) (11) (23),

.27 *DateFromInt* (*yyyy*) (12) (23)},

.28 *holidayInLieu* = {*d* + 1 | *d* \in *NationalHoliday* · *IsSunday* (*d*)} in

.29 *NationalHoliday* \cup *holidayInLieu*;

GetDataInJST は、日本標準時基準の日付を得る。

public static

4.0 *GetDataInJST* : *Date* \rightarrow *Date*

.1 *GetDataInJST* (*d*) \triangleq

.2 *GetDataInST* (*DiffBetweenGMTandJST*) (*d*);

private static

```

5.0  AsStringAux :  $\mathbb{Z} \rightarrow \text{char}^*$ 
.1   AsStringAux (i)  $\triangleq$ 
.2     let str = FInteger‘AsString in
.3     if i  $\geq$  10
.4     then str (i)
.5     else " "  $\frown$  str (i);
    GetJapaneseYearAsString は、平成以後の和暦日付文字列を得る
public static
6.0  GetJapaneseYearAsString : Date  $\rightarrow$   $\text{char}^*$ 
.1   GetJapaneseYearAsString (d)  $\triangleq$ 
.2     let asString = FInteger‘AsString,
.3         JapaneseYear = Year (d) - DiffBetweenSeirekiAndHoureki,
.4         m = Month (d),
.5         aDate = Day (d),
.6         YY = asString (JapaneseYear),
.7         MM = AsStringAux (m),
.8         DD = AsStringAux (aDate) in
.9     YY  $\frown$  MM  $\frown$  DD;
    GetVernalEquinoxInJST は、yyyy 年の日本標準時の春分を得る。
public static
7.0  GetVernalEquinoxInJST :  $\mathbb{Z} \rightarrow \text{Date}$ 
.1   GetVernalEquinoxInJST (yyyy)  $\triangleq$ 
.2     GetDateInJST (GetVernalEquinoxInGMT (yyyy));
    GetSummerSolsticeInJST は、yyyy 年の日本標準時の夏至を得る。
public static
8.0  GetSummerSolsticeInJST :  $\mathbb{Z} \rightarrow \text{Date}$ 
.1   GetSummerSolsticeInJST (yyyy)  $\triangleq$ 
.2     GetDateInJST (GetSummerSolsticeInGMT (yyyy));
    GetAutumnalEquinoxInJST は、yyyy 年の日本標準時の秋分を得る。
public static
9.0  GetAutumnalEquinoxInJST :  $\mathbb{Z} \rightarrow \text{Date}$ 
.1   GetAutumnalEquinoxInJST (yyyy)  $\triangleq$ 
.2     GetDateInJST (GetAutumnalEquinoxInGMT (yyyy));
    GetWinterSolsticeInJST は、yyyy 年の日本標準時の冬至を得る。
public static
10.0 GetWinterSolsticeInJST :  $\mathbb{Z} \rightarrow \text{Date}$ 
.1   GetWinterSolsticeInJST (yyyy)  $\triangleq$ 
.2     GetDateInJST (GetWinterSolsticeInGMT (yyyy))
end FJapaneseCalendar
Test Suite :      vdm.tc
Class :          FJapaneseCalendar

```

Name	#Calls	Coverage
FJapaneseCalendar‘AsStringAux	2	✓
FJapaneseCalendar‘GetHolidays	143	90%
FJapaneseCalendar‘GetDateInJST	298	✓
FJapaneseCalendar‘GetVernalEquinoxInJST	146	✓
FJapaneseCalendar‘GetSummerSolsticeInJST	3	✓
FJapaneseCalendar‘GetWinterSolsticeInJST	3	✓

Name	#Calls	Coverage
FJapaneseCalendar‘GetAutumnaEquinoxInJST	146	✓
FJapaneseCalendar‘GetJapaneseYearAsString	1	✓
Total Coverage		93%

0.2 FJapaneseCalendarT

FCalendar のテストを行う。

class *FJapaneseCalendarT* is subclass of *FJapaneseCalendar*

functions

public static

```
11.0 run : () → ℬ
.1 run ()  $\triangleq$ 
.2 let testcases =
.3 [t1 (), t2 (), t3 (), t4 (), t5 (), t6 (), t7 (), t8 (), t9 (), t10 (),
.4 t11 ()] in
.5 FTestDriver.run (testcases);
```

0.2.1 「GetHolidaysWithinDays」を検査する

```
12.0 t1 : () → FTestDriver.TestCase
.1 t1 ()  $\triangleq$ 
.2 mk-FTestDriver.TestCase
.3 (
.4 "FCalendarT:t1 : \t 「GetHolidaysWithinDays」を検査する",
.5 let d = DateFromInt,
.6 g = GetHolidaysWithinDates (GetHolidays) in
.7 g (d (2004) (4) (28)) (d (2004) (4) (29)) = {d (2004) (4) (29)} ∧
.8 g (d (2004) (4) (28)) (d (2004) (5) (2)) = {d (2004) (4) (29)} ∧
.9 g (d (2004) (4) (28)) (d (2004) (5) (3)) =
{d (2004) (4) (29), d (2004) (5) (3)} ∧
.10 g (d (2004) (4) (28)) (d (2004) (5) (4)) =
{d (2004) (4) (29), d (2004) (5) (3), d (2004) (5) (4)} ∧
.11 g (d (2004) (4) (28)) (d (2004) (5) (5)) =
{d (2004) (4) (29), d (2004) (5) (3), d (2004) (5) (4), d (2004) (5) (5)} ∧
.12 g (d (2004) (4) (29)) (d (2004) (5) (5)) =
{d (2004) (4) (29), d (2004) (5) (3), d (2004) (5) (4), d (2004) (5) (5)} ∧
.13 g (d (2004) (4) (30)) (d (2004) (5) (5)) =
{d (2004) (5) (3), d (2004) (5) (4), d (2004) (5) (5)} ∧
.14 g (d (2004) (1) (1)) (d (2004) (12) (31)) =
.15 {d (2004) (1) (1), d (2004) (1) (12), d (2004) (2) (11), d (2004) (3) (20), d (2004) (4) (29),
.16 d (2004) (5) (3), d (2004) (5) (4), d (2004) (5) (5), d (2004) (7) (19), d (2004) (9) (20), d (2004) (9) (23),
.17 d (2004) (10) (11), d (2004) (11) (3), d (2004) (11) (23), d (2004) (12) (23)} ∧
.18 g (d (2005) (3) (1)) (d (2005) (3) (31)) =
{d (2005) (3) (20), d (2005) (3) (21)});
```

0.2.2 「GetHolidaysWithinDatesNotSunday」を検査する

```

13.0  t2 : () → FTestDriverc TestCase
.1    t2 ()  $\triangle$ 
.2    mk-FTestDriverc TestCase
.3      (
.4        "FCalendarTct2 : \t 「GetHolidaysWithinDatesNotSunday」を検
査する",
.5        let d = DateFromInt,
.6          g = GetHolidaysWithinDatesNotSunday (GetHolidays) in
.7          g (d (2004) (1) (1)) (d (2004) (12) (31)) =
.8          {d (2004) (1) (1), d (2004) (1) (12), d (2004) (2) (11), d (2004) (3) (20), d (2004) (4) (29),
.9            d (2004) (5) (3), d (2004) (5) (4), d (2004) (5) (5), d (2004) (7) (19), d (2004) (9) (20), d (2004) (9) (23),
.10           d (2004) (10) (11), d (2004) (11) (3), d (2004) (11) (23), d (2004) (12) (23)} ∧

.11          g (d (2005) (3) (1)) (d (2005) (3) (31)) = {d (2005) (3) (21)} ∧
.12          g (d (2006) (1) (1)) (d (2006) (1) (31)) =
{d (2006) (1) (2), d (2006) (1) (9)};

```

0.2.3 「GetHolidaysWithinDatesAsSunday」を検査する

```

14.0  t3 : () → FTestDriverc TestCase
.1    t3 ()  $\triangle$ 
.2    mk-FTestDriverc TestCase
.3      (
.4        "FCalendarTct3 : \t 「GetHolidaysWithinDatesAsSunday」を検
査する",
.5        let d = DateFromInt,
.6          g = GetHolidaysWithinDatesAsSunday (GetHolidays) in
.7          g (d (2004) (1) (1)) (d (2004) (12) (31)) = {} ∧
.8          g (d (2005) (3) (1)) (d (2005) (3) (31)) = {d (2005) (3) (20)} ∧
.9          g (d (2006) (1) (1)) (d (2006) (1) (31)) = {d (2006) (1) (1)};

```

0.2.4 「GetNumberOfHolidaysWithinDates」を検査する

```

15.0  t4 : () → FTestDriverc TestCase
.1    t4 ()  $\triangle$ 
.2    mk-FTestDriverc TestCase
.3    (
.4      "FCalendarTct4 : \t 「GetNumberOfHolidaysWithinDates」を検
査する",
.5      let d = DateFromInt,
.6          g = GetNumberOfHolidaysWithinDates (GetHolidays) in
.7      g (d (2004) (4) (28)) (d (2004) (4) (29)) = 1 ∧
.8      g (d (2004) (4) (28)) (d (2004) (5) (2)) = 1 ∧
.9      g (d (2004) (4) (28)) (d (2004) (5) (3)) = 2 ∧
.10     g (d (2004) (4) (28)) (d (2004) (5) (4)) = 3 ∧
.11     g (d (2004) (4) (28)) (d (2004) (5) (5)) = 4 ∧
.12     g (d (2004) (4) (29)) (d (2004) (5) (5)) = 4 ∧
.13     g (d (2004) (4) (30)) (d (2004) (5) (5)) = 3 ∧
.14     g (d (2004) (1) (1)) (d (2004) (12) (31)) = 15 ∧
.15     g (d (2005) (3) (1)) (d (2005) (3) (31)) = 2 ∧
.16     g (d (2006) (1) (1)) (d (2006) (1) (31)) = 3);

```

0.2.5 「GetNumberOfDayOff」を検査する

```

16.0  t5 : () → FTestDriverc TestCase
.1    t5 ()  $\triangle$ 
.2    mk-FTestDriverc TestCase
.3    (
.4      "FCalendarTct5 : \t 「GetNumberOfDayOff」を検査する",
.5      let d = DateFromInt,
.6          g = GetNumberOfDayOff (GetHolidays) in
.7      g (d (2004) (4) (28)) (d (2004) (4) (29)) = 1 ∧
.8      g (d (2004) (4) (28)) (d (2004) (5) (2)) = 2 ∧
.9      g (d (2004) (4) (28)) (d (2004) (5) (3)) = 3 ∧
.10     g (d (2004) (4) (28)) (d (2004) (5) (4)) = 4 ∧
.11     g (d (2004) (4) (28)) (d (2004) (5) (5)) = 5 ∧
.12     g (d (2004) (4) (29)) (d (2004) (5) (5)) = 5 ∧
.13     g (d (2004) (4) (30)) (d (2004) (5) (5)) = 4 ∧
.14     g (d (2005) (3) (1)) (d (2005) (3) (31)) = 5 ∧
.15     g (d (2006) (1) (1)) (d (2006) (1) (31)) = 7);

```

0.2.6 「GetNumberOfDayOff1」を検査する

```

17.0  t6 : () → FTestDriver' TestCase
.1    t6 ()  $\triangle$ 
.2    mk-FTestDriver' TestCase
.3    (
.4      "FCalendarT't6 : \t 「GetNumberOfDayOff1」を検査する",
.5      let d = DateFromInt,
.6          g = GetNumberOfDayOff1 (GetHolidays) in
.7      g (d (2004) (4) (28)) (d (2004) (4) (29)) = 1 ∧
.8      g (d (2004) (4) (28)) (d (2004) (5) (2)) = 2 ∧
.9      g (d (2004) (4) (28)) (d (2004) (5) (3)) = 3 ∧
.10     g (d (2004) (4) (28)) (d (2004) (5) (4)) = 4 ∧
.11     g (d (2004) (4) (28)) (d (2004) (5) (5)) = 5 ∧
.12     g (d (2004) (4) (29)) (d (2004) (5) (5)) = 4 ∧
.13     g (d (2004) (4) (30)) (d (2004) (5) (5)) = 4 ∧
.14     g (d (2005) (3) (1)) (d (2005) (3) (31)) = 5 ∧
.15     g (d (2006) (1) (1)) (d (2006) (1) (31)) = 6);

```

0.2.7 休日を考慮した加減算 (+-1) を検査する

```

18.0  t7 : () → FTestDriver' TestCase
.1    t7 ()  $\triangle$ 
.2    mk-FTestDriver' TestCase
.3    (
.4      "FCalendarT't7 : \t 休日を考慮した加減算 (+-1) を検査する",
.5      let d = DateFromInt,
.6          n = BusinessDateToFuture (GetHolidays),
.7          p = BusinessDateToPast (GetHolidays) in
.8      n (d (2004) (4) (29)) = d (2004) (4) (30) ∧
.9      p (d (2004) (4) (29)) = d (2004) (4) (28) ∧
.10     n (d (2004) (5) (1)) = d (2004) (5) (6) ∧
.11     n (d (2004) (5) (2)) = d (2004) (5) (6) ∧
.12     p (d (2004) (5) (5)) = d (2004) (4) (30));

```

0.2.8 休日を考慮した加減算を検査する


```

19.0  t8 : () → FTestDriver ` TestCase
.1    t8 ()  $\triangle$ 
.2    mk-FTestDriver ` TestCase
.3    (
.4      "FCalendarT:t8:\t 休日を考慮した加減算を検査する",
.5      let d = DateFromInt,
.6          n = BusinessDateToFuture (GetHolidays),
.7          p = BusinessDateToPast (GetHolidays),
.8          a = AddBusinessDays (GetHolidays),
.9          s = SubtractBusinessDays (GetHolidays) in
.10     n (d (2004) (4) (28)) = a (d (2004) (4) (28)) (0) ∧
.11     p (d (2004) (4) (30)) = s (d (2004) (4) (30)) (0) ∧
.12     n (d (2004) (4) (29)) = d (2004) (4) (30) ∧
.13     n (d (2004) (5) (1)) = d (2004) (5) (6) ∧
.14     p (d (2004) (5) (6)) = d (2004) (5) (6) ∧
.15     p (d (2004) (5) (5)) = d (2004) (4) (30) ∧
.16     s (d (2004) (5) (6)) (1) = d (2004) (4) (30) ∧
.17     a (d (2004) (5) (1)) (1) = d (2004) (5) (7) ∧
.18     s (d (2004) (5) (1)) (-1) = d (2004) (4) (30) ∧
.19     a (d (2004) (5) (6)) (-1) = d (2004) (5) (6) ∧
.20     s (d (2004) (5) (6)) (1) = d (2004) (4) (30) ∧
.21     s (d (2004) (5) (6)) (6) = d (2004) (4) (22) ∧
.22     a (d (2004) (4) (22)) (6) = d (2004) (5) (6));

```

0.2.9 休日の判定を検査する

```

20.0  t9 : () → FTestDriver ` TestCase
.1    t9 ()  $\triangle$ 
.2    mk-FTestDriver ` TestCase
.3    (
.4      "FCalendarT:t9:\t 休日の判定を検査する",
.5      let d = DateFromInt,
.6          h = IsHoliday (GetHolidays),
.7          f = IsDayOff (GetHolidays) in
.8      h (d (2004) (4) (29)) ∧
.9      h (d (2004) (5) (2)) = false ∧
.10     h (d (2004) (5) (3)) ∧
.11     h (d (2004) (5) (4)) ∧
.12     h (d (2004) (5) (5)) ∧
.13     h (d (2004) (5) (6)) = false ∧
.14     f (d (2004) (5) (2)) ∧
.15     f (d (2004) (5) (9)) ∧
.16     f (d (2005) (3) (19)) = false ∧
.17     f (d (2005) (3) (20)) ∧
.18     f (d (2005) (3) (21)) ∧
.19     f (d (2005) (3) (22)) = false);

```

0.2.10 「和暦日付文字列を得る」を検査する

```
21.0  t10 : () → FTestDriver`TestCase
.1    t10 ()  $\triangle$ 
.2      mk-FTestDriver`TestCase
.3      (
.4        "FCalendarT`t10 : \t 「和暦日付文字列を得る」を検査する",
.5        GetJapaneseYearAsString (DateFromInt (2004) (2) (29))      =
"16 229");
```

0.2.11 春分・夏至・秋分・冬至を検査する

```
22.0  t11 : () → FTestDriver`TestCase
.1    t11 ()  $\triangle$ 
.2      mk-FTestDriver`TestCase
.3      (
.4        "FCalendarT`t11 : \t 春分・夏至・秋分・冬至を検査する",
.5        let d = DateFromInt in
.6        GetVernalEquinoxInJST (2004) = d (2004) (3) (20) ∧
.7        GetVernalEquinoxInJST (2003) = d (2003) (3) (21) ∧
.8        GetVernalEquinoxInJST (2020) = d (2020) (3) (20) ∧
.9        GetSummerSolsticeInJST (2004) = d (2004) (6) (21) ∧
.10       GetSummerSolsticeInJST (2003) = d (2003) (6) (22) ∧
.11       GetSummerSolsticeInJST (2020) = d (2020) (6) (21) ∧
.12       GetAutumnalEquinoxInJST (2004) = d (2004) (9) (23) ∧
.13       GetAutumnalEquinoxInJST (2003) = d (2003) (9) (23) ∧
.14       GetAutumnalEquinoxInJST (2020) = d (2020) (9) (22) ∧
.15       GetWinterSolsticeInJST (2004) = d (2004) (12) (22) ∧
.16       GetWinterSolsticeInJST (2003) = d (2003) (12) (22) ∧
.17       GetWinterSolsticeInJST (2020) = d (2020) (12) (21))
end FJapaneseCalendarT
```