Welcome to SmartDate

Welcome to version 2.2 of SmartDate, Robelle's date library, which is fully compatible with year 2000 dates. SmartDate's powerful capabilities can be called from your program to process dates and convert them into many formats.

SmartDate solves some of the most common data processing problems: checking, editing, and converting dates. SmartDate also provides you with an easy way to change your applications to support dates into the year 2000. Use SmartDate in all your applications to ensure reliable and fast date handling, in this century or the next.

MPE V versus MPE/iX

SmartDate works on both MPE V and MPE/iX computers. On MPE V, you install SmartDate in the system SL (see the chapter on Installing SmartDate for instructions). On MPE/iX, you run your native mode programs with an XL file: DateXL.Pub.Robelle.

Conditions of Use

SmartDate is a separate product; it is not a component of any other Robelle product. You are licensed to use SmartDate only on a specific set of CPUs. SmartDate cannot be included in software that is being distributed to sites which are not licensed to use SmartDate. Application software developers who are interested in integrating SmartDate into their products should contact Robelle Consulting Ltd.

Fourth-Generation Languages (4GLs)

SmartDate is more difficult to use from 4GLs than from a regular programming language (e.g., COBOL or Pascal). Some 4GLs do a LOADPROC for every call to the SmartDate intrinsics (e.g., Reactor from Speedware). Because the LOADPROC calls to SmartDate are slow, they affect SmartDate's overall performance. Fortunately, most 4GLs have their own date-handling routines that replace the functionality of SmartDate.

Documentation

The user documentation for SmartDate assumes that you have some previous experience with MPE and programming. To print copies of the user manual, run the Printdoc program:

```
:run printdoc.pub.robelle;info="sdate.doc.robelle"
```

Printdoc is menu driven, and it is very easy to use. Printdoc asks you for information; if you are not sure of your answer, you can ask for help by typing a question mark (?). Printdoc supports all types of LaserJet printers and regular line printers.

The user manual is also available as a Microsoft WinHelp file. You can also obtain on-line help by using the Help command of the SmartDate program:

```
:run sdate.pub.robelle
>>h
```

New Features of SmartDate

Here are the highlights of the new features in SmartDate.

New Features in Version 2.2:

- Combining edit function 7, add days to the from date, with a to-date format of 16 showed the wrong day of the week.

New Features in Version 2.1:

- SmartDate will now work with Time Machine and the HP utility Setdate.

Installing SmartDate

There are two versions of SmartDate: SmartDate/V for MPE V systems and for compatibility mode programs on MPE/iX, and SmartDate/iX for native mode programs on MPE/iX.

Depending on your operating system (MPE V or MPE/iX) and the kind of programs that call SmartDate (compatibility mode, native mode, or both), you may end up installing either, both, or neither version. Read all the instructions carefully.

Before using either version of SmartDate, you must build or upgrade the Robelle account and restore all files from the distribution tape.

Installation Procedures

You can install SmartDate on your system in three easy steps. First, build (or upgrade) the Robelle account using a job stream that we provide. Second, restore all Robelle files from tape to disc. Third, stream the installation job stream. All of these steps are easily accomplished if you log on as Manager.Sys.

Even if you already have the Robelle account, the first thing you must do is stream the Robelle job:

```
:hello manager.sys
:file robtape;dev=tape
:restore *robtape;robelle.pub.sys {=reply on console}
:stream robelle.pub.sys            {supply password}
```

This job stream launches a second job, which sends you a message when it has completed. Stay logged on as Manager.Sys and put a password on the Robelle account. If you already had a Robelle account, use the same password.

```
:altacct robelle;pass=pswd {something tricky}
```

Please note that during installation, we add OP capability to the Robelle account. When the SmartDate installation is complete, you have the option to remove OP capability.

Stay logged on as Manager.Sys and restore the Robelle files:

```
:file robtape;dev=tape
:restore *robtape;@.@.robelle     {=reply on console}
```

After the SmartDate files have been restored, complete the installation by streaming the installation job stream:

```
:hello mgr.robelle
:stream install.datejob
```

System SL Installation

This procedure is required for MPE V systems and for compatibility mode programs on MPE/iX.

The SmartDate/V routines are distributed in the file DateUSL.Pub.Robelle. The job stream SysSL.DateJob.Robelle installs the SmartDate/V intrinsics in the system SL; this is the only method for installing SmartDate/V. You also use this job stream to update SmartDate/V when you receive a new version, or to re-install the interface after an update from HP. You will need a small tape to make a new cold load tape that includes the SmartDate/V segments.

Warning: You must have created the Robelle account and restored all files as described in the Installation Procedures section above.

Steps

To install SmartDate/V into the system SL, follow these steps:

1. Ensure that no one uses SmartDate/V during the installation. Also make sure programs that use SmartDate/V are not running. Stop all such jobs and send an operator warning.

```
:showjob
:warn @;please stop for 20 minutes
:abortjob #snnn
```

2. Stream the installation job. If MPE prompts for passwords, supply them.

```
:stream syssl.datejob.robelle
```

3. SmartDate/V uses the Segmenter to add the segments into SL.Pub.Sys. It then requests a tape called "coldload" to create a new cold load tape containing MPE plus the SmartDate/V intrinsics. Mount a tape with a write ring and :Reply. Save this tape and use it for future cold loads.

   If you're installing to an MPE/iX machine, the job does not create a cold load tape. You must create a system load tape manually.

4. If everything goes well, SmartDate/V prints a final message on the console.
5. Please save the job listing for future reference.

Now that SmartDate/V is installed, you should be able to use it in your application programs.

User XL Installation

This is an optional procedure for native mode programs on MPE/iX.

You normally access SmartDate/iX by specifying XL=DateXL.Pub.Robelle when running your program. As a result, there may be no need to perform any installation steps for SmartDate/iX.

The advantage of leaving SmartDate/iX in the Robelle account and always pointing your programs to the XL when you run them, is that when you receive an upgrade of the Robelle account files, SmartDate/iX is automatically upgraded, with no effort on your part. The disadvantage is that you must remember to always put the XL= statement in all your Run commands of programs that use SmartDate/iX. If you don't want to change your Run commands, you can copy SmartDate/iX into your own XL, which is already being searched.

The XL can reside anywhere on your system; it does not have to be in the same account or the same group as your XL file. Here are the commands to install SmartDate/iX in your own XL:

```
:hello user.acct
:linkedit
>xl xl                {we assume the XL already exists}
>copyxl from=sdatexl.pub.robelle; replace
>exit
```

The Replace option is not in all versions of Linkedit. If Replace is not available in your version and you already have SmartDate in your XL file, you need to manually purge the existing modules before copying the new one.

Datexl can successfully be installed into the system XL file, but this is not recommended by HP. Datexl can be combined into an XL file with other Robelle XL files, except Qcompxl (a part of Qedit).

Group or Pub SL Installation

This procedure is for MPE/V systems and for compatibility mode programs on MPE/iX. It is not the recommended or supported procedure. See the SmartDate/V instructions in the previous section.

Pub SL

To install SmartDate/V in a Pub SL, both the account and the Pub group have to have privileged mode capability. The calling program must be installed in the same account (it does not have to be in the Pub group), and it must be run with Lib=P.
Example

Here are the commands to modify an existing account called Dev:

```
:hello mgr.dev,pub
:segmenter
-buildusl $newpass,400,8
-auxusl sdateusl.pub.robelle
-copy segment,smartdate
-buildsl sl,400,8
-addsl smartdate
-exit
```

Group SL

The steps for installing SmartDate/V in a group SL are the same as for the Pub SL, except that you have to use the actual group name instead of the Pub group. Your user programs must be installed in the same group as the SL, and they must be run with Lib=G.

SmartDate Essentials

SmartDate is a set of routines or intrinsics that you call from your program. The SmartDate/V intrinsics are installed in the system SL. The SmartDate/iX intrinsics are installed in an XL. You access SmartDate by calling these intrinsics, just as you would the IMAGE intrinsics. There are two primary intrinsics:

RDTCONVERT: The main routine that does all of the processing.

RDTERROR: A routine to obtain an English-language description for a specific error number.

Control Record
Errors
MPE/iX

SmartDate Control Record

SmartDate requires a special control record. For COBOL, we suggest you place this control record in the Copylib and then copy it into programs that use SmartDate. Typing the control record incorrectly is a common error. Instead of typing the record, we recommend using the file Cobol5.Qlibsrc.Robelle. You can copy the control record directly into your COBOL program with the following Qedit commands (use /Join with EDIT/3000):

```
/add 50.1=cobol5.qlibsrc.robelle 1/20
```

Definition

This is the definition of the control record with the proper initializing values:

```
01  rdt-control.
    05  rdt-from-type   pic s9(4) comp.
    05  rdt-to-type     pic s9(4) comp.
    05  rdt-aux-result  pic s9(4) comp value zeros.
    05  rdt-status      pic s9(4) comp.
```

Error Handling

SmartDate is very careful to edit all dates and to make sure they are valid. We suggest that you check the status result in the control record after all calls to RDTCONVERT.

You can use the RDTERROR routine to obtain an English description of the last error. Note: The error routine expects the entire control record as a parameter, not just the status result. An example call to RDTERROR in COBOL looks like this:

```
if rdt-status <> 0 then
   move spaces to out-buf
   call "RDTERROR" using rdt-control
                         out-buf
                         out-buflen
   display "Date error " out-buf.
```

MPE/iX

To access SmartDate/iX from a native mode program, you must modify the :Run command for your program so that it includes an XL file. The following example shows how you would compile, link, and run a native mode COBOL program that calls SmartDate/iX:

```
:cob74xl main.src
:link
:run $oldpass;xl='sdatexl.pub.robelle'
```

SmartDate Intrinsics

Processing dates is a large part of many data processing applications. In order to successfully use SmartDate, you must understand the basic ideas of how it works. You call RDTCONVERT to convert one date, the *from-date*, to another format, the *to-date*; additional editing during conversion is optional. Before calling RDTCONVERT, you must first set up the rdt-control area with a description of the *from-* and *to-date* formats.

Much of the hard work is done by RDTCONVERT. But you, the applications programmer, must correctly initialize the control record with the proper information. The control record is designed to be very compact so that you can specify a lot of information with just a few simple initialization statements. Understanding the set up procedure for the control record is the key to making SmartDate work for you.

A key benefit of SmartDate is the automatic verification of input date formats. By checking all its input date formats and making sure they are valid, SmartDate completely eliminates the possibility of using invalid formats.

In this chapter we describe the SmartDate intrinsics in alphabetical order. All intrinsics require a control parameter. The condition code is not returned by any SmartDate intrinsic.

RDTCONVERT Intrinsic

Check a *from-date* with an option to convert it to a *to-date*, and an option to edit it during the checking and converting process.

RDTCONVERT *from-date, to-date, rdt-control*

From-Date
To-Date
Rdt-Control
Validating Dates
Cutoff Year
Date Format
Edit Options

From-Date

A date in the format specified by the *from-type* parameter of rdt-control. If the *from-date* is the current date, a dummy parameter must be passed as the *from-date* to act as a placeholder.

To-Date

A date in the format specified by the *to-type* parameter of rdt-control. If no *to-date* is specified, a dummy parameter must be passed as the *to-date* to act as a placeholder.

Rdt-Control

Special control record that must be declared as follows:

```
01  rdt-control.
    05  rdt-from-type   pic s9(4) comp.
    05  rdt-to-type     pic s9(4) comp.
    05  rdt-aux-result  pic s9(4) comp value zeros.
    05  rdt-status      pic s9(4) comp.
```

You must initialize this control buffer before each call to RDTCONVERT. See the chapter on Accessing SmartDate for complete details of possible values in the control record.

Validating Dates

Whenever you call RDTCONVERT, the *from-date* is checked to ensure that it is valid. For character dates, the century, year, month, and day are checked to ensure that they are numeric (date formats that permit non-numeric characters are checked appropriately). All dates are also checked to make sure the month is valid (not less than 1 or greater than 12) and the day is valid for the month. SmartDate correctly calculates leap years and ensures that February 29 is considered valid only in leap years.

You can apply additional date checking by using some of the specific edit options described below. For example, you can ask SmartDate to check whether the *from-date* is not equal to or greater than the current date. Date verification is an important function of SmartDate.

Cutoff Year

If the *from-date* has only a 2-digit year, SmartDate must determine the century component of the date. By default, SmartDate assumes the current century. The century component of a *from-date* with a 2-digit year for a date before December 31, 1999 is assumed to be 19xx. For dates in the 2000s, SmartDate assumes the century to be 20xx. You can change this behavior in two ways.

1. Use the cutoff year edit option.

2. Specify a cutoff year by assigning a value to the RobelleCutoffYear JCW. For example,

```
:setjcw RobelleCutoffYear 50
```

If the 2-digit year in the *from-date* is on or after the cutoff value, 19xx is the assumed century in the *to-date*; otherwise, 20xx is the assumed century.

Date Formats

There is a single set of constants describing the date formats. Some formats are only valid for the *from-date* and some are only valid for the *to-date*. Most are valid for both.

Each format details both a storage container (e.g., a 32-bit integer) and a date type. You must describe the correct format to RDTCONVERT in order to have your date processed correctly. A date in CCYYMMDD format in an 8-byte character field is not the same as a date in CCYYMMDD format in a 32-bit integer.

There are many date formats. Before we present each format individually, here is a summary of all the date formats:

Num	Dec 31, 2000	Format	Data Type	From/To
0	0	Today/None	16-bit	From/To
1	311200	DDMMYY	6-byte	From/To
2	31/12/00	DD/MM/YY	8-byte	From/To
3	001231	YYMMDD	32-bit	From/To
4	51566	Calendar	16-bit	From/To
5	DEC 31, 2000	MMM DD, CCYY	12-byte	To-only
6	31 DEC00	DD MMMYY	8-byte	From/To
7	123100	MMDDYY	6-byte	From/To
8	12/31/00	MM/DD/YY	8-byte	From/To
9	001231	YYMMDD	6-byte	From/To
10	0012	YYMM	16-bit	From/To
11	1200	MMYY	4-byte	From/To
12	DEC 00	MMM YY	6-byte	To-only
13	00/12/31	YY/MM/DD	8-byte	From/To
14	20001231	CCYYMMDD	8-byte	From/To
15	20001231	CCYYMMDD	32-bit	From/To
16	WED, DEC 31, 2000	DDD, MMM DD, CCYY	18-byte	To-only
17	A01231	AAMMDD	6-byte	From/To
18	31122000	DDMM[CC]YY	n-byte	From/To
19	12312000	MMDD[CC]YY	n-byte	From/To
20	20001231	[CC]YYMMDD	n-byte	From/To
21	Dec31 00	MMMDD YY	8-byte	From/To
22	2001231	YYYMMDD	32-bit	To-only
23	200012	CCYYMM	6-byte	From/To
24	200012	CCYYMM	32-bit	From/To
25	2000	CCYY	4-byte	From/To
26	2000	CCYY	32-bit	From/To
27	51615	Powerhouse	16-bit	From/To

Notes:
Format 4: 51566 = 1100100 101101110 = (100) (366)
Format 27: 51615 = 1100100 1100 11111 = (100) (12) (31)

0
1
2
3
4
5

6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

Format 0: Today/None From/To

If this is the *from-type*, RDTCONVERT assumes today's date. If this is the *to-type*, no *to-date* is produced, but all date editing functions are applied to the *from-date*.

Format 1: DDMMYY 6-byte From/To

Character date in day, month, year format.

Format 2: DD/MM/YY 8-byte From/To

Character date in day, month, year format with a delimiter. For the *from-date*, the delimiter can be any special character. If this is the *to-date* format, the delimiter is always a slash.

Format 3: YYMMDD 32-bit From/To

Binary date in year, month, day format.

Format 4: Calendar 16-bit From/To

MPE calendar format. In this format, bits 0 to 6 contain the year and 7 to 15 contain the day. If the *from-date* evaluates to a year beyond 2027, an error is returned if you select this as the *to-date* type (the maximum value that can be held in 7 bits is 127).

Format 5: MMM DD, CCYY 12-byte To-only

Character format that you cannot use as a *from-type*. The month is formatted with a three-letter English abbreviation for the month.

Format 6: **DD MMMYY** **8-byte** **From/To**

Short form of a date that includes a three-letter English abbreviation for the month.

Format 7: MMDDYY 6-byte From/To

Character date in month, day, year format.

Format 8: **MM/DD/YY** **8-byte** **From/To**

Character date in month, day, year format with a delimiter. For the *from-date*, the delimiter can be any special character. If this is the *to-date* format, the delimiter is always a slash.

Format 9: YYMMDD 6-byte From/To

Character date in year, month, day format.

Format 10: YYMM 16-bit From/To

Binary date with only the year and the month. By default, if you are converting from this *from-type* to a *to-type* that includes the day of the month, the first day of the month is used. You can specify the last day of the month by using the RdtLastDay edit option (#9).

Format 11: MMYY 4-byte From/To

Character date with only the year and the month. See format 10 for restrictions on this date type.

Format 12: MMM YY 6-byte To-only

Character date with the month in a three-letter English abbreviation and the year. This format cannot be used as a *from-type*.

Format 13: YY/MM/DD 8-byte From/To

Character date in year, month, day format with a delimiter. For the *from-date*, the delimiter can be any special character. If this is the *to-date* format, the delimiter is always a slash.

Format 14: CCYYMMDD 8-byte From/To

Character date in century, year, month, day format.

Format 15: CCYYMMDD 32-bit From/To

Binary date in century, year, month, day format.

Format 16: DDD, MMM DD, CCYY 18-byte To-only

Character date in day, month, year format. The day, month, and year must be valid and the day number must be valid for the month.

Format 17: AAMMDD 6-byte From/To

A special 6-character date format in which the year is either two numeric digits, or a letter and a number. This format makes it easy to preserve dates in a 6-character container while handling year 2000 dates. Dates beyond December 31, 1999 still collate correctly if they are stored in this format.

By substituting a letter of the alphabet in the first position of the year, we can extend a 6-digit date and make sure the dates collate correctly. For example,

YY of AAMMDD	CCYY
A0 - A9	2000 - 2009
B0 - B9	2010 - 2019
C0 - C9	2020 - 2029

Because letters come before numbers in the collating sequence, you can be sure AAMMDD dates beyond 1999 are ordered correctly.

Format 18: DDMM[CC]YY n-byte From/To

Free-format date in day, month, year format.

If this is the *from-type*, the *from-date* must end in a space, a carriage return, or a binary zero. The date is read left to right. There can be white space or special characters between the day, the month, and the year. The month can be a numeric or a three-letter abbreviation in English (e.g., MAR for March). The century is optional. If the century is not specified, one of the following methods will be applied to determine the century:

1. If the century cutoff edit code is used, the cutoff value is used to determine the century.

2 If the RobelleCutoffYear JCW has been set, its value is used to determine the century.

3. If no century cutoff edit code is specified and there is no RobelleCutoffYear JCW, the century is assumed to be the current century.

If this is the *to-type*, the *from-date* is formatted as DDMMCCYY with no spaces or special characters.

Format 19: MMDD[CC]YY n-byte From/To

Free-format date in month, day, year format. See format 18 for details on processing free-format dates as the *from-type*. If this is the *to-type*, the *from-date* is formatted as MMDDCCYY with no spaces or special characters.

Format 20: [CC]YYMMDD 8-byte From/To

Free-format date in month, day, year format. See format 18 for details on processing free-format dates as the *from-type*. If this is the *to-type*, the *from-date* is formatted as CCYYMMDD with no spaces or special characters.

Format 21: MMMDD YY 8-byte From/To

Character date in month, day, year format. The month is formatted in a three-letter English abbreviation.

Format 22: YYYMMDD 32-bit To-only

The YYYMMDD date type is similar to YYMMDD, except that the first digit denotes the century. If the first digit is a one (1), then the century is 19xx; if the first digit is a two (2), then the century is 20xx. This date format is used by some third-party software packages such as MACS and APS.

Format 23: CCYYMM 6-byte From/To

Character date in century, year, month format.

Format 24: CCYYMM 32-bit From/To

Binary date in century, year, month format.

Format 25: CCYY 4-byte From/To

Character date in century, year format.

Format 26: CCYY **32-bit** **From/To**

Binary date in century, year format.

Format 27: Powerhouse 16-bit From/To

The PHdate format is compatible with the Cognos PowerHouse date format. The date is stored in individual bit groupings with seven bits for the year of the century (bits 0-6), four bits for the month (bits 7-10), and five bits for the day (bits 11-15).

Edit Options

You can specify an edit option as part of the *from-type*. The edit specifications are optional; most people use the aux-result field of the control record to specify extra information.

The *from-type* consists of numbers in this form:

```
xxyy
```

where *xx* is the edit operation and *yy* is the *from-type* of the *from-date*. For example, the *from-type*

```
1015
```

would specify edit option 10 and *from-type* 15 (binary CCYYMMDD).

This is a complete list of the edit options for SmartDate.

1
2
3
4
5
6
7
8
9
10

Option 1: Days between dates (RdtDaysBetween)

The number of days between the *from-date* and the *to-date* is returned in the aux-result field of the control record (*to-date* minus *from-date*). Both the *from-date* and the *to-date* must contain valid dates. No output conversion occurs. An error is returned if the number of days between the dates exceeds 9999.

Option 2: Day of the week (RdtDayOfTheWeek)

Return the day of the week as a number in the aux-result field of the control record. The numeric value corresponds to these days:

1. Sunday
2. Monday
3. Tuesday
4. Wednesday
5. Thursday
6. Friday
7. Saturday

You can combine this edit option with a conversion between the *from-date* and the *to-date*.

Option 3: Date within range of the current date (RdtX6DDMMYY)

Check whether the *from-date* is within a specified number of days before or after today's date. For this edit option, the aux-result field of the control record must contain a special number in the form of

```
bbaa
```

where *bb* checks the number of days before and *aa* checks the number of days after the current date. For example, the value 1020 would check whether the *from-date* is no less than 10 days before today's date and no more than 20 days after. The value 0015 would check whether the *from-date* is not before today's date and not more than 15 days in the future.

Option 4: Date within 20 days before today and 60 days after (Rdt2060)

Check whether the *from-date* is no more than 20 days before today and no more than 60 days after. The aux-result field of the control record is ignored by this edit option.

Option 5: Date on or before today's date (RdtBeforeToday)

Check whether the *from-date* is less than or equal to today's date.

Option 6: Is the from-date greater than the to-date? (RdtFromAfterTo)

Check whether the *from-date* is greater than or equal to the *to-date*.

Option 7: Add days (RdtAddDays)

Use this edit code to add days to a *from-date* and use the result to form a new *to-date*. You can add a positive number of days (to move the date forward) or a negative number of days (to move the date backward). Specify the added number of days in the aux-result field of the control record. The maximum number of days RDTCONVERT can add or subtract is 9,999.

Option 8: Allow zero from-date (RdtAllowZero)

Some applications use a date value of zero to indicate some special date. Use this edit option to force RDTCONVERT to treat a zero date as a valid date. When converting a zero *from-date*, RDTCONVERT chooses a suitable format for the *to-date* based on its type.

Option 9: Last day of the month (RdtLastDay)

Use this edit option to force the day of the *to-date* to the last day of the month for the year and month specified in the *from-date*. This option only works with *to-date* formats that include the day of the month.

Option 10: Apply a century cutoff rule (RdtCutoffYear)

If the 2-digit year in the *from-date* is on or after the cutoff value in the aux-result field of the control record, 19xx is the assumed century in the *to-date*; otherwise, 20xx is the assumed century. This edit function only works with *from-date* types that have 2-digit years. You can also specify a cutoff year by using the RobelleCutoffYear JCW.

RDTERROR Intrinsic

Format an English error message into a buffer and return the length of the message in the buffer based on the value in the control buffer.

RDTERROR *rdt-control, buffer, len*

Rdt-Control
Buffer
Len
Error Messages

Rdt-Control

The control buffer used in a previous call to RDTCONVERT. The control buffer must not be changed between the RDTCONVERT call and the call to RDTERROR.

Buffer

A character array into which RDTERROR puts the error message. Note: The buffer must be big enough to hold the largest possible error message.

Len

The length of the message returned in the buffer. This is a 16-bit integer that must be passed by reference.

SmartDate Error Messages

RDTCONVERT always returns a result in the rdt-status field of the rdt-control record. When the call to RDTCONVERT is successful, a zero is returned. Otherwise, an error number is returned. You can use RDTERROR to obtain an English description of the error.

This is a complete list of the RDTCONVERT error numbers and their associated RDTERROR error messages.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22

1: Invalid Day

The day in the *from-date* is invalid. The day is invalid when it is greater than 31, not numeric, or it exceeds the number of days in the month.

2: Invalid Month

The month in the *from-date* is invalid. The month is invalid when it is greater than 12 or not numeric, or the ASCII version of the month does not match one of the three-letter English abbreviations for the month.

3: Invalid Year

SmartDate supports years from 1583 to 2583. This error occurs if the year in the *from-date* is either not numeric or outside the range of valid years, including results from date arithmetic. For specific date formats (e.g., calendar), this error can also be returned when the year of the *from-date* exceeds the possible values for the year of the *to-date*.

4: Bad Calendar Date

The MPE calendar date format has special restrictions. The year cannot exceed 2027, and the day of the year cannot exceed 365 (366 in leap years).

5: Day Not in Month

The day of the month is either less than one or greater than the number of days in the month of the specified year (e.g., 29 as the day for a date in February of a non-leap year).

6: Features Not Implemented

Some features of SmartDate have not been implemented. This error should not occur. If it does, please contact Robelle Consulting Ltd.

7: Invalid Date From-Type

The *from-type* in the control record is not one of the valid SmartDate date types.

8: Invalid Date To-Type

The *to-type* in the control record is not one of the valid SmartDate date types.

9: Edit Check Failed

One of the checking operations specified by the edit option failed. This is usually caused by the *from-date* not falling within the edit option constraints. It is also returned by edit option #6 (RdtFromAfterTo) when the *from-date* is not greater than the *to-date*.

10: Invalid Edit Option

The edit option in the control record is not one of the valid SmartDate edit options.

11: Invalid Special Character

If the *from-type* specifies a date format with special characters as delimiters between the day, month, and year, this error is returned when SmartDate doesn't find any of these special delimiters in the *from-date*.

12: Not Used

This error number should not be returned. If it is, please report the example to Robelle Consulting Ltd.

13: Not Used

This error number should not be returned. If it is, please report the example to Robelle Consulting Ltd.

14: Not Used

This error number should not be returned. If it is, please report the example to Robelle Consulting Ltd.

15: Not Used

This error number should not be returned. If it is, please report the example to Robelle Consulting Ltd.

16: Too Many Days

The maximum number of days between any two dates supported by SmartDate is 9,999. If you attempt to add or subtract more than this number of days (edit option #7), or if the difference between dates is more than 9,999 (edit option #1), this error is returned.

17: Missing Termination Character for Free-Form Date

If you specify a free-form date format as the *from-type* and the *from-date* does not end in a space, a carriage return, or a binary zero, this error is returned.

18: Invalid Cutoff Year

This error is returned for two reasons:

1. The cutoff year specified in the aux-result field of the control record is less than one or greater than 99.

2. The format of the *from-date* does not allow a cutoff year to be applied (e.g., the format already includes the century).

19: SmartDate has expired

SmartDate has expired because it was only available for a given number of days. Contact Robelle Consulting Ltd. about purchasing SmartDate or extending its expiry date if you are already licensed for SmartDate.

20: SmartDate not authorized for this HPSUSAN

The HPSUSAN number of your HP 3000 computer has not been authorized for SmartDate. This often happens when you upgrade your HP 3000. Contact Robelle Consulting Ltd. about changing your authorization or purchasing SmartDate for your specific CPU.

21: Call to FINDJCW failed

A call to the MPE FINDJCW intrinsic failed. Please report this error to Robelle Consulting Ltd.

22: Call to PUTJCW failed

A call to the MPE PUTJCW intrinsic failed. Please report this error to Robelle Consulting Ltd.

Examples

The rich functionality of SmartDate is packed into a single routine called RDTCONVERT. By combining different values in the control record, many different date operations can be performed. Rather than showing examples in a specific programming language, we have shown them in pseudocode, which we assume you are able to translate into your specific programming language. We also include complete COBOL and Pascal examples later in this documentation.

Constants

To make your programs and our examples more readable, you should always declare a set of constants for SmartDate date formats and edit options. We use only symbolic constants in all the examples.

Date Format
Edit Options

Date Format Constants

The following constants are for the various date formats. For character dates, we use X*n*, where *n* is the number of characters in the date. For binary dates, we use I16 for 16-bit integers and I32 for 32-bit integers.

```
set RdtToday             to 0
set RdtNone              to 0
set RdtX6DDMMYY          to 1
set RdtX8DD-MM-YY        to 2
set RdtI32YYMMDD         to 3
set RdtI16Calendar       to 4
set RdtX12MMMDDCCYY      to 5
set RdtX8DDMMMYY         to 6
set RdtX6MMDDYY          to 7
set RdtX8MM-DD-YY        to 8
set RdtX6YYMMDD          to 9
set RdtI16YYMM           to 10
set RdtX4YYMM            to 11
set RdtX6MMMYY           to 12
set RdtX8YY-MM-DD        to 13
set RdtX8CCYYMMDD        to 14
set RdtI32CCYYMMDD       to 15
set RdtX18DDDMMDDCCYY    to 16
set RdtX6AAMMDD          to 17
set RdtXnDDMMCCYY        to 18
set RdtXnMMDDCCYY        to 19
set RdtXnCCYYMMDD        to 20
set RdtX8MMMDDYY         to 21
set RdtI32YYYMMDD        to 22
set RdtX6CCYYMM          to 23
set RdtI32CCYYMM         to 24
set RdtX4CCYY            to 25
set RdtI32CCYY           to 26
set RdtI16Powerhouse     to 27
```

Edit Options

These are the constant values for the SmartDate edit options:

```
set RdtDaysBetween       to  100
set RdtDayOfTheWeek      to  200
set RdtDateRange         to  300
set Rdt2060              to  400
set RdtBeforeToday       to  500
set RdtFromAfterTo       to  600
set RdtAddDays           to  700
set RdtAllowZero         to  800
set RdtLastDay           to  900
set RdtCutoffYear        to 1000
```

Using the Current Date

This example initializes a date with today's date. The *to-date* is a 32-bit integer in CCYYMMDD format (i.e., date type format 15). The DummyDate parameter is a placeholder; it will never be examined by RDTCONVERT because the *from-date* is today's date.

```
set RdtFromType to RdtToday
set RdtToType   to RdtI32CCYYMMDD

RdtConvert(DummyDate,
           CurrentDate,
           RdtControl)

if RdtStatus <> 0
   ReportDateError
```

Using the Current Date Plus Ten Days

This example initializes a date that is ten days in the future. As in the previous example, the *to-date* is a 32-bit integer in CCYYMMDD format.

```
set RdtFromType  to RdtAddDays + RdtToday
set RdtToType    to RdtI32CCYYMMDD
set RdtAuxResult to 10

RdtConvert(DummyDate,
           CurrentPlusTen,
           RdtControl)

if RdtStatus <> 0
   ReportDateError
```

Validate a Date

If you wanted, for example, to check a user-specified date to see whether it was valid, use the zero format for the *to-type*. In this example, the DummyDate variable is again used as a placeholder, but this time it holds the place of the *to-date*. Because the *from-date* is not being converted to any *to-type*, the *to-date* will never be used by RDTCONVERT.

```
set RdtFromType  to RdtX8CCYYMMDD
set RdtToType    to RdtNone

RdtConvert(InputDate,
           DummyDate,
           RdtControl)

if RdtStatus <> 0
   ReportDateError
```

Additional Checking

In the previous example, RDTCONVERT checked whether the *from-date* was valid. In this example, we also check whether the *from-date* is within a range of days from the current date. To do this, we use the RdtDateRange edit option, in which we must specify the number of days before and after today's date. The format for this option is

```
bbaa
```

where *bb* is the number of days before and *aa* is the number of days after the current date. For this example, we assume that we want the *from-date* to be no less than today's date and no more than 60 days in the future. The RdtDateRange for this check is 0060 (zero days back and 60 days ahead).

```
set RdtFromType  to RdtDateRange + RdtX8CCYYMMDD
set RdtToType    to RdtNone
set RdtAuxResult to 0060

RdtConvert(InputDate,
           DummyDate,
           RdtControl)

if RdtStatus <> 0
   ReportDateError
```

Convert a Date

SmartDate includes a powerful feature that lets you convert from one date format to another. While doing conversions, SmartDate ensures that the *from-date* is valid. In this example, we assume the user has entered a date in YYMMDD format. Internally, we store dates in CCYYMMDD 32-bit format (this takes little storage space and ensures that internal dates collate correctly). This example then converts the user-specified date to our internal format, validating it at the same time:

```
set RdtFromType  to RdtX6YYMMDD
set RdtToType    to RdtI32CCYYMMDD

RdtConvert(InputDate,
           DBDate,
           RdtControl)

if RdtStatus <> 0
   ReportDateError
```

Using a Cutoff Year in Date Conversions

To ensure that all dates are correct, you should always use 4-digit years (e.g., 2001). Unfortunately, many users prefer to enter only 2-digit years. In order to ensure that dates entered after December 31, 2000 are correct, you need to use a cutoff rule for processing 2-digit dates.

This example is the same as the previous one, except that a century cutoff rule is applied using the year 1950. In this example, RDTCONVERT assumes 19xx for the century if the year is equal to or greater than 50 (i.e., 500101 is treated as January 1, 1950), and it assumes 20xx for the century when the year is less than 50 (i.e., 010101 is treated as January 1, 2001).

```
set RdtFromType  to RdtX6YYMMDD + RdtCutoffYear
set RdtToType    to RdtI32CCYYMMDD
set RdtAuxResult to 50

RdtConvert(InputDate,
           DBDate,
           RdtControl)

if RdtStatus <> 0
   ReportDateError
```

Date Program

SmartDate includes a program for demonstration and verification. This program is called SDate.Pub.Robelle.

The date program allows you to enter dates in a variety of SmartDate formats, to store dates in different internal formats, and to do date arithmetic and date comparisons. After each command, the A and B registers (described below) are formatted into the selected print format and displayed on $stdlist.

The date program is command-driven. The command set is very simple, but like SmartDate itself, these commands can be combined in numerous ways.

On-Line Help

The Help command provides on-line help on both the Date program and the SmartDate intrinsics. To obtain help, type "h" at the command prompt:

```
>>h
```

If you wish to obtain help only about the SmartDate intrinsics, use the Intrinsic keyword with the Help command:

```
>>h intrinsic
```

Registers

The date program uses five internal registers; use these registers to store dates or date formats. The name of the register is used in many of the commands.

A
B
I
P
Current Date

A Register

The A register holds a date. By default, the date is stored in date format 15 (32-bit CCYYMMDD). You can change the format of the A register with the "=" command.

B Register

The B register is another date register. Its default format is the same as the A register. The A and B registers do not have to have the same date format.

I Register

The I register is the input register. You set the I register to the date format in which you want to enter dates. By default, the I register is set to format 14 (8-byte CCYYMMDD).

P Register

The P register is the print register. You set the P register to the date format in which you want to print dates. By default, the P register is set to format 5 (12-byte MMM DD, CCYY).

Current Date

You can refer to the current date with an asterisk (*). In the following example, the current date is assigned to the A register:

```
A=*
```

Commands

When you run the date program, it prompts you with ">>." You enter a command by typing its name and parameters. You can enter commands in uppercase or lowercase letters.

A
B
=
Question

A Command

The A command assigns a value to the A register.

B Command

The B command assigns a value to the B register.

= Command

The "=" command is used to assign date formats to specific registers. The various forms of this command are:

=An	Assign date format *n* to the A register.
=Bn	Assign date format *n* to the B register.
=D+	Print the day of the week when displaying dates.
=D-	Do not print the day of the week.
=In	Select the input date format.
=Pn	Specify the print date format.

? Command

The "?" command is used to answer questions about dates. The various forms of this command are:

?A<=* Is the A register less than or equal to today?

?A>B Is the A register greater than the B register?

?A-B Number of days between the A and B registers.

?A:5/10 Is the A register between 5 days before and 10 days after today's date?

Examples

These examples demonstrate the powerful capabilities of SmartDate.

Assignment
Questions
Input Format
Internal
Weekday

Assignment

We start by showing you how to assign dates to the A or B register.

```
a=*-1          {A is assigned yesterday's date}
a=19970327     {assign a specific date}
a=a+1          {add one day to A}
a=b            {assign B to A}
a=b+10         {add 10 days to B and assign the result to A}
b=a-10         {subtract 10 days from A and assign the result to B}
a=b@           {set A to last day of the month in B}
```

Questions

The following examples show you how to answer questions about the dates in the A and B registers.

`?a<b`	{Is A less than B?}
`?b-a`	{number of days between dates}
`?b<=*`	{Is B less than or equal to today?}
`?a:30/60`	{Is A 30 days before or 60 days after today's date?}

Input Format

This example changes the input date format used by the date program. We specify the DDMMCCYY format, instead of the default CCYYMMDD format, and then we assign a date to the B register.

```
=i18
a=27031997      {notice the DDMMCCYY form}
```

Internal Format

You can change the internal storage format of the A and B registers. However, be careful when you do this because some internal formats cannot represent all possible dates. For example,

```
=a3
a=20010327
```

puts the date 010327 in A register, which SmartDate displays as March 7, 1901. Notice that the century has changed from the one you specified. That is why it's good practice to always give your internal dates enough room for the century and the year.

Weekday

When you run the date program, it always displays the day of the week for the A and B register dates. You can disable this feature with

```
=d-
```

Examples of Calling SmartDate

This chapter contains working examples of COBOL and Pascal source code that call SmartDate. You can copy the examples from the manual by typing them from scratch, which would be tedious and error-prone. The best way to copy the examples is to take them from the on-line SmartDate documentation file, which is stored in Robelle's Qedit format. If you have Qedit, just Text a copy of SDate.Doc.Robelle and extract the parts that you need.

If you don't have Qedit, use the Qcopy program to copy the documentation file into a new file with a non-Qedit format; then use your favorite text editor to edit the new file.

```
:run qcopy.qlib.robelle
>from=sdate.doc.robelle;to=mydate.source;new
>exit
```

Cobol Example
Pascal Example

COBOL Example

The following example shows you how to call a COBOL program from SmartDate. This program prompts for a database, a dataset, and a field. It reads the entire dataset, and checks each record to make sure the field you specified has a valid MM/DD/YY date format.

Source

```
$control nolist
$control source,errors=10
 identification division.
 program-id.    testdate.
 author.        Robyn Rennie, Robelle Consulting Ltd.
 date-written.  March 24th, 1997.
 security.      Copyright Robelle Consulting Ltd 1991-1998.
 remarks.

*************************************************************
*                                                           *
*                testdate - test valid date values.         *
*                                                           *
*  version:  1.0                                            *
*  purpose:                                                 *
*                                                           *
*  General-purpose program to test for valid dates.         *
*  - this program edits a date field in the format MMDDYY   *
*    checking for valid date values, adds 150 days to the   *
*    date and converts it into a date in the format         *
*    CCYYMMDD                                               *
*                                                           *
*************************************************************

 environment division.
 configuration section.
 source-computer.  hp-3000 series 927.
 object-computer.  hp-3000 series 927.
 special-names.
     top is new-page.

 input-output section.
 file-control.
     select line-printer assign to "LINEPRT".

 data division.
 file section.
 fd  line-printer
     data record is line-record.
 01  line-record            pic x(132).

$page "CONSTANTS"
 working-storage section.
 01  true-value             pic x value "T".
 01  false-value            pic x value "F".

$page "VARIABLES"
 01  line-count             pic s9(4) comp.
 01  page-no                pic s9(4) comp.

 01  input-line                  pic x(80).
     88  answer-spaces           value spaces.

 01  end-of-db-set-flag          pic x value "F".
     88  end-of-db-set           value "T".

 01  in-record-count             pic s9(9) comp.
```

```
 01  in-error-count              pic s9(9) comp.

 01  display-number              pic -zzz,zz9.

$page "image area"

 01  image-area.
     05  db-all-list                pic x(2) value "@ ".
     05  db-same-list               pic x(2) value "* ".
     05  db-null-list               pic s9(4) comp value 0.
     05  db-dummy-arg               pic s9(4).
     05  db-password                pic x(8).
     05  db-mode0                   pic s9(4) comp value 0.
     05  db-mode1                   pic s9(4) comp value 1.
     05  db-mode2                   pic s9(4) comp value 2.
     05  db-get-serial  redefines db-mode2 pic s9(4) comp.
     05  db-mode3                   pic s9(4) comp value 3.
     05  db-rewind-set  redefines db-mode3 pic s9(4) comp.
     05  db-get-backwards redefines db-mode3 pic s9(4) comp.
     05  db-mode4                   pic s9(4) comp value 4.
     05  db-get-direct  redefines db-mode4 pic s9(4) comp.
     05  db-mode5                   pic s9(4) comp value 5.
     05  db-get-chained redefines db-mode5 pic s9(4) comp.
     05  db-mode6                   pic s9(4) comp value 6.
     05  db-get-previous redefines db-mode6 pic s9(4) comp.
     05  db-mode7                   pic s9(4) comp value 7.
     05  db-get-keyed   redefines db-mode7 pic s9(4) comp.
     05  db-status-area.
         10  db-cond-word           pic s9(4) comp.
             88  db-stat-ok             value zeros.
             88  db-end-of-chain        value 15.
             88  db-begin-of-chain      value 14.
             88  db-no-entry            value 17.
             88  db-end-file            value 11.
             88  db-begin-file          value 10.
         10  db-stat2               pic s9(4) comp.
         10  db-stat3-4             pic s9(9) comp.
         10  db-chain-length        pic s9(9) comp.
             88  db-empty-chain     value zeros.
         10  db-stat7-8             pic s9(9) comp.
         10  db-stat9-10            pic s9(9) comp.
$page "db- variables"

 01  db-base.
     05  filler                 pic x(2)  value spaces.
     05  db-name                pic x(26).

 01  db-set                     pic x(16).

 01  db-list                    pic x(80).

 01  db-buffer.
     05  date-field             pic x(08).

$page "Smartdate variables"
 01  rdt-control.
     05  rdt-from-type              pic s9(4) comp.
```

```
     05  rdt-to-type                pic s9(4) comp.
     05  rdt-aux-result             pic s9(4) comp value zeros.
     05  rdt-status                 pic s9(4) comp.

 01  out-buf                        pic x(80).
 01  out-buflen                     pic s9(4) comp value zeros.

 01  new-date                       pic x(18) value spaces.

 01  rdt-edit-codes.
     05  RdtDaysBetween             pic s9(4) comp value 100.
     05  RdtDayOfTheWeek            pic s9(4) comp value 200.
     05  RdtDateRange               pic s9(4) comp value 300.
     05  Rdt2060                    pic s9(4) comp value 400.
     05  RdtBeforeToday             pic s9(4) comp value 500.
     05  RdtFromAfterTo             pic s9(4) comp value 600.
     05  RdtAddDays                 pic s9(4) comp value 700.
     05  RdtAllowZero               pic s9(4) comp value 800.
     05  RdtLastDay                 pic s9(4) comp value 900.
     05  RdtCutoffYear              pic s9(4) comp value 1000.


 01  rdt-from-to-type.
     05  RdtToday                   pic s9(4) comp value 0.
     05  RdtNone                    pic s9(4) comp value 0.
     05  RdtX6DDMMYY                pic s9(4) comp value 1.
     05  RdtX8DD-MM-YY              pic s9(4) comp value 2.
     05  RdtI32YYMMDD               pic s9(4) comp value 3.
     05  RdtI16Calendar             pic s9(4) comp value 4.
     05  RdtX12MMMDDCCYY            pic s9(4) comp value 5.
     05  RdtX8DDMMMYY               pic s9(4) comp value 6.
     05  RdtX6MMDDYY                pic s9(4) comp value 7.
     05  RdtX8MM-DD-YY              pic s9(4) comp value 8.
     05  RdtX6YYMMDD                pic s9(4) comp value 9.
     05  RdtI16YYMM                 pic s9(4) comp value 10.
     05  RdtX4YYMM                  pic s9(4) comp value 11.
     05  RdtX6MMMYY                 pic s9(4) comp value 12.
     05  RdtX8YY-MM-DD              pic s9(4) comp value 13.
     05  RdtX8CCYYMMDD              pic s9(4) comp value 14.
     05  RdtI32CCYYMMDD             pic s9(4) comp value 15.
     05  RdtX18DDDMMDDCCYY          pic s9(4) comp value 16.
     05  RdtX6AAMMDD                pic s9(4) comp value 17.
     05  RdtXnDDMMCCYY              pic s9(4) comp value 18.
     05  RdtXnMMDDCCYY              pic s9(4) comp value 19.
     05  RdtXnCCYYMMDD              pic s9(4) comp value 20.
     05  RdtX8MMMDDYY               pic s9(4) comp value 21.
     05  RdtI32YYYMMDD              pic s9(4) comp value 22.
     05  RdtX6CCYYMM                pic s9(4) comp value 23.
     05  RdtI32CCYYMM               pic s9(4) comp value 24.
     05  RdtX4CCYY                  pic s9(4) comp value 25.
     05  RdtI32CCYY                 pic s9(4) comp value 26.

$page "[00] MAINLINE"
 procedure division.
 00-main.

    perform 05-get-parameters
```

```
       thru 05-get-parameters-exit.

    if not answer-spaces then
        perform 10-open-database
           thru 10-open-database-exit
         move false-value        to end-of-db-set-flag
         move zeros              to in-record-count
         move zeros              to in-error-count
         perform 20-read-set
            thru 20-read-set-exit
           until end-of-db-set or in-error-count > 500
         move in-record-count to display-number
         display "Total records = ", display-number
         move in-error-count  to display-number
         display "Total errors  = ", display-number.



    perform 90-close-base
       thru 90-close-base-exit.

 00-main-exit.  goback.
$page "[05] get-parameters"
*
* Prompt for the database, password, dataset and field list.
*
 05-get-parameters.

     perform 05-10-get-database.

     if not answer-spaces then
         perform 05-20-get-password
         if not answer-spaces then
             perform 05-30-get-dataset
             if not answer-spaces then
                 perform 05-40-get-field-list.

 05-get-parameters-exit.  exit.

 05-10-get-database.
     move spaces                 to input-line.
     display "Enter Database Name".
     accept input-line.
     move input-line             to db-name.

 05-20-get-password.
     move spaces                 to input-line.
     display "Enter Database Password".
     accept input-line.
     move input-line             to db-password.

 05-30-get-dataset.
     move spaces                 to input-line.
     display "Enter Dataset Name (must be upper-case)".
     accept input-line.
     move input-line             to db-set.
```

```
 05-40-get-field-list.
     move spaces                 to input-line.
     display "Enter @ or Field List (must be upper-case)".
     accept input-line.
     move input-line             to db-list.

$page "[10] init-testread"
*
*  Open the database.
*

 10-open-database.

     call "DBOPEN" using db-base
                         db-password
                         db-mode5
                         db-status-area.
     if not db-stat-ok then
         perform 99-fatal-error.

 10-open-database-exit.  exit.
$page "[20] read-set"
*
*  Read all the records in the dataset.
*

 20-read-set.

     perform 20-10-get-next.

     if not end-of-db-set then
         add 1 to in-record-count
         perform 20-20-check-dates.

 20-read-set-exit.  exit.

 20-10-get-next.
     call "DBGET" using db-base
                        db-set
                        db-mode2
                        db-status-area
                        db-list
                        db-buffer
                        db-dummy-arg.
     if db-end-file then
         move true-value               to end-of-db-set-flag
     else
     if not db-stat-ok then
         perform 99-fatal-error.

 20-20-check-dates.
*  Check field for invalid date values add 150 days to the date
*  and convert from MM/DD/YY to CCYYMMDD

     add RdtAddDays to RdtX8MM-DD-YY giving rdt-from-type
     move RdtX8CCYYMMDD                  to rdt-to-type
     move 150                            to rdt-aux-result
```

```
     move 0                                to rdt-status.

     call "RDTCONVERT" using date-field
                             new-date
                             rdt-control
     if rdt-status of rdt-control <> 0 then
        add 1 to in-error-count
        move spaces to out-buf
        move zeros  to out-buflen
        call "RDTERROR" using rdt-control
                              out-buf
                              out-buflen
        display date-field "  " out-buf.

$page "[90] close-base"
*
*  Cleanup by closing the database.
*

 90-close-base.
     call "DBCLOSE" using db-base
                          db-dummy-arg
                          db-mode1
                          db-status-area.
     if not db-stat-ok then
         perform 99-fatal-error.

 90-close-base-exit.  exit.

$page "[99] fatal-error"

 99-fatal-error.

     call "DBEXPLAIN" using db-status-area.

     goback.
```

Pascal Example

Calling SmartDate from Pascal is more difficult than from other languages because Pascal has tighter type-checking on procedure parameters. The following example is a complete Pascal/iX program that prompts for a date in an MMDDYY format. The program then converts this date to a variety of formats and displays the results. Note that the $check_actual_parm$ setting remains at 2 for the entire program. This is done to prevent parameter type mismatches between SmartDate and Pascal.

Source

```
$list off$
$hp3000_16$
{ Sample program showing how to call Robelle's SmartDate routines
  in Pascal
}
program example(input,output);

const
  RdtSunday    = 1;
  RdtMonday    = 2;
  RdtTuesday   = 3;
  RdtWednesday = 4;
  RdtThursday  = 5;
  RdtFriday    = 6;
  RdtSaturday  = 7;

  RdtToday           =  0;
  RdtNone            =  0;
  RdtX6ddmmyy        =  1;
  RdtX8dd_mm_yy      =  2;
  RdtI32yymmdd       =  3;
  RdtI16calendar     =  4;
  RdtX12mmmddccyy    =  5;
  RdtX8ddmmmyy       =  6;
  RdtX6mmddyy        =  7;
  RdtX8mm_dd_yy      =  8;
  RdtX6yymmdd        =  9;
  RdtI16yymm         = 10;
  RdtX4mmyy          = 11;
  RdtX6mmmyy         = 12;
  RdtX8yy_mm_dd      = 13;
  RdtX8ccyymmdd      = 14;
  RdtI32ccyymmdd     = 15;
  RdtX18dddmmmddccyy = 16;
  RdtX6aammdd        = 17;
  RdtXNddmmccyy      = 18;
  RdtXNmmddccyy      = 19;
  RdtXNccyymmdd      = 20;
  RdtX8mmmddyy       = 21;
  RdtI32yyymmdd      = 22;
  RdtI32ccyymm       = 23;
  RdtX6ccyymm        = 24;
  RdtI32ccyy         = 25;
  RdtX4ccyy          = 26;

  RdtDaysBetween     =  100;
  RdtDayOfTheWeek    =  200;
  RdtDateRange       =  300;
  Rdt2060            =  400;
  RdtBeforeToday     =  500;
  RdtFromAfterTo     =  600;
  RdtAddDays         =  700;
  RdtAllowZero       =  800;
  RdtLastDay         =  900;
  RdtCutoffYear      = 1000;

  RdtErrorBadDay        =  1;
```

```
  RdtErrorBadMonth      =  2;
  RdtErrorBadYear       =  3;
  RdtErrorBadJulian     =  4;
  RdtErrorDayInMonth    =  5;
                  { value 6 is unused }
  RdtErrorFromtype      =  7;
  RdtErrorTotype        =  8;
  RdtErrorFailed        =  9;
  RdtErrorBadEditnum    = 10;
  RdtErrorMissingDelim  = 11;
  RdtErrorSuprdate2_1   = 12;
  RdtErrorSuprdate2_2   = 13;
  RdtErrorSuprdate2_3   = 14;
  RdtErrorSuprdate2_4   = 15;
  RdtErrorTooFar        = 16;
  RdtErrorFreeFormat    = 17;

type
   TypeRdtConvert = record
      case shortint of
        1: (x6 : packed array[1..6] of char);
        2: (x8 : packed array[1..8] of char);
        3: (x12: packed array[1..12] of char);
        4: (x18: packed array[1..18] of char);
        5: (xn : packed array[1..80] of char);
        6: (i16: shortint);
        7: (i32: integer);
   end;

   TypeRdtControl = record
      fromtype: shortint;
      totype  : shortint;
      aux     : shortint;
      result  : shortint;
   end;

   line_type = packed array[1..80] of char;

{-------------------------------------------------------------}
type
   dayname_array = packed array[1..12] of char;
   weekday_array = array[1..7] of dayname_array;

const
   weekday_name = weekday_array[
     'Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday',
     'Saturday' ];

var
   inbuf, outbuf: line_type;

{-------------------------------------------------------------}
$check_formal_parm 2$
$check_actual_parm 2$

procedure rdtconvert( var fromdate,
                          todate     : TypeRdtConvert;
```

```
                      var rdtcontrol : TypeRdtControl);
  external spl;

procedure RdtError(   var rdtcontrol : TypeRdtControl;
                      var buf        : line_type;
                      var len        : integer);
  external spl;

$check_formal_parm 3$

{-------------------------------------------------------------}
function get_record( var buf: line_type ): boolean;
begin
  writeln;
  prompt('Enter a date in mmddyy form: ');
  get_record := false;
  readln( buf );
  if buf<>'' then
     get_record := true;
end;


function validate_date( var inbuf, outbuf: line_type ): boolean;
var
  fromdate,
  todate     : TypeRdtConvert;
  rdtcontrol : TypeRdtControl;
  len        : integer;
begin
  validate_date := false;
  outbuf := ' ';

  strmove( 6, inbuf,1, fromdate.x6,1 );
  rdtcontrol.fromtype := RdtX6mmddyy;
  rdtcontrol.totype   := RdtNone;
  rdtconvert( fromdate, todate, rdtcontrol );

  if rdtcontrol.result=0 then
    validate_date := true
  else
    RdtError( rdtcontrol, outbuf, len );
end;


function convert_date( var inbuf, outbuf: line_type ): boolean;
var
  fromdate,
  todate     : TypeRdtConvert;
  rdtcontrol : TypeRdtControl;
  len        : integer;
begin
  convert_date := false;
  outbuf := ' ';

  strmove( 6, inbuf,1, fromdate.x6,1 );
  rdtcontrol.fromtype := RdtX6mmddyy;
  rdtcontrol.totype   := RdtX12mmmddccyy;
```

```
  rdtconvert( fromdate, todate, rdtcontrol );

  if rdtcontrol.result=0 then
  begin
    strmove( 12, todate.x12,1, outbuf,1 );
    writeln( 'You entered ', outbuf:12 );
    convert_date := true;
  end
  else
    RdtError( rdtcontrol, outbuf, len );
end;


function get_weekday( var inbuf, outbuf: line_type ): boolean;
var
  fromdate,
  todate     : TypeRdtConvert;
  rdtcontrol : TypeRdtControl;
  len        : integer;
begin
  get_weekday := false;
  outbuf := ' ';

  strmove( 6, inbuf,1, fromdate.x6,1 );
  rdtcontrol.fromtype := RdtX6mmddyy + RdtDayOfTheWeek;
  rdtcontrol.totype   := Rdtnone;
  rdtconvert( fromdate, todate, rdtcontrol );

  if rdtcontrol.result=0 then
  begin
    writeln( 'That day is a ', weekday_name[ rdtcontrol.aux ] );
    get_weekday := true;
  end
  else
    RdtError( rdtcontrol, outbuf, len );
end;


function add_days( var inbuf, outbuf: line_type ): boolean;
var
  fromdate,
  todate     : TypeRdtConvert;
  rdtcontrol : TypeRdtControl;
  len        : integer;
begin
  add_days := false;
  outbuf := ' ';

  strmove( 6, inbuf,1, fromdate.x6,1 );
  rdtcontrol.fromtype := RdtX6mmddyy + RdtAddDays;
  rdtcontrol.totype   := RdtX18dddmmmddccyy;
  rdtcontrol.aux      := 100;
  rdtconvert( fromdate, todate, rdtcontrol );

  if rdtcontrol.result=0 then
  begin
    strmove( 18, todate.x18,1, outbuf,1 );
```

```
    writeln( '100 days from that date, it will be ', outbuf:18 );
    add_days := true;
  end
  else
    RdtError( rdtcontrol, outbuf, len );
end;


function compare_days( var inbuf, outbuf: line_type ): boolean;
var
  fromdate,
  todate     : TypeRdtConvert;
  rdtcontrol : TypeRdtControl;
  len        : integer;
begin
  compare_days := false;
  outbuf := ' ';

  strmove( 6, inbuf,1, fromdate.x6,1 );
  todate.x6 := '010180';
  rdtcontrol.fromtype := RdtX6mmddyy + RdtFromAfterTo;
  rdtcontrol.totype   := RdtX6mmddyy;
  rdtconvert( fromdate, todate, rdtcontrol );

  if rdtcontrol.result=0 then
  begin
    writeln( 'That date is on or after Jan 1, 1980' );
    compare_days := true;
  end
  else if rdtcontrol.result=RdtErrorFailed then
  begin
    writeln('That date is before Jan 1, 1980' );
    compare_days := true;
  end
  else
    RdtError( rdtcontrol, outbuf, len );
end;


function last_day( var inbuf, outbuf: line_type ): boolean;
var
  fromdate,
  todate     : TypeRdtConvert;
  rdtcontrol : TypeRdtControl;
  len        : integer;
begin
  last_day := false;
  outbuf := ' ';

  rdtcontrol.fromtype := RdtToday + RdtLastDay;
  rdtcontrol.totype   := RdtX18dddmmmddccyy;
  rdtconvert( fromdate, todate, rdtcontrol );

  if rdtcontrol.result=0 then
  begin
    strmove( 18, todate.x18,1, outbuf,1 );
    writeln( 'Your current month-end is ', outbuf:18 );
```

```
    last_day := true;
  end
  else
    RdtError( rdtcontrol, outbuf, len );
end;


function assume_century( var inbuf, outbuf: line_type ): boolean;
var
  fromdate,
  todate     : TypeRdtConvert;
  rdtcontrol : TypeRdtControl;
  len        : integer;
begin
  assume_century := false;
  outbuf := ' ';

  strmove( 6, inbuf,1, fromdate.x6,1 );
  rdtcontrol.fromtype := RdtX6mmddyy + RdtCutoffYear;
  rdtcontrol.totype   := RdtX12mmmddccyy;
  rdtcontrol.aux      := 10; { 00..09 are year 2000 }
  rdtconvert( fromdate, todate, rdtcontrol );

  if rdtcontrol.result=0 then
  begin
    strmove( 12, todate.x12,1, outbuf,1 );
    writeln( 'With a century cutoff of 10, the date entered is ',
             outbuf:12 );
    assume_century := true;
  end
  else
    RdtError( rdtcontrol, outbuf, len );
end;

{-------------------------------------------------------------}

procedure report_error( var outbuf: line_type );
begin
  writeln( '** Error: ', outbuf );
end;

begin
   while get_record(inbuf) do
   begin
      if validate_date(inbuf,outbuf) then
      begin
      if not convert_date(inbuf, outbuf)
         then report_error( outbuf );
      if not get_weekday(inbuf, outbuf)
         then report_error( outbuf );
      if not add_days(inbuf, outbuf)
         then report_error( outbuf );
      if not compare_days(inbuf, outbuf)
         then report_error( outbuf );
      if not last_day(inbuf, outbuf)
         then report_error( outbuf );
      if not assume_century(inbuf, outbuf)
```

```
         then report_error( outbuf );
      end
      else
         report_error( outbuf );
   end;
end.
```

How to Contact Robelle

In the United States, in Canada, and in places not listed below, contact us at the following address:

Robelle Consulting Ltd.
Unit 201, 15399-102A Ave.
Surrey, B.C. Canada V3R 7K1

Toll-free: 1-888-ROBELLE
: (1-888-762-3553)
Phone : (604) 582-1700
Fax : (604) 582-1799

E-mail : info@robelle.com
E-mail : support@robelle.com
Web : www.robelle.com

For our international distributors listing, note that the phone and fax numbers shown are for out-of-country dialing.

Europe

Czech Republic, Slovak Republic

ASW Praha s.r.o.
Attention: Jiri Nemec
Phone: 420 2 723 305
Fax: 420 2 723 305
E-mail: aswltd@mbox.vol.cz

France, Belgium

ARES
Attention: Renee Belegou
Phone: 33 1 69 86 60 24
Fax: 33 1 69 28 19 18
E-mail: rbelegou@ares.fr
Web: www.ares.fr

Germany

SWS SoftWare Systems GmbH
Attention: Daniela Wieland
Phone: 49 7621 689 190
Fax: 49 7621 689 191
E-mail: info@sws.ch
Web: www.sws.ch

The Netherlands, Belgium

Samco Automation b.v.
Attention: Marius Schild
Phone: 31 13 521 5655
Fax: 31 13 528 8815
E-mail: marius@samco.nl
Web: www.samco.nl

Scandinavia

Ole Nord AB
Attention: Ole Nord
Phone: 46 8 623 00 50
Fax: 46 8 35 42 45
E-mail: info@olenordab.se
Web: www.olenordab.se

Switzerland, Austria

SWS SoftWare Systems AG
Attention: Daniela Wieland
Phone: 41 31 981 0666
Fax: 41 31 981 3263
E-mail: info@sws.ch
Web: www.sws.ch

United Kingdom, Ireland

Robelle Consulting
Attention: Clive Oldfield
Phone: 44 171 473 2558
Fax: 44 171 473 2558

E-mail: robelle_oldfield@msn.com

Africa

Saudi Arabia, United Arab Emirates, Kuwait, Oman,

Saudi Information Technology
Attention: Abdulmohsen Al-Hobayb
Phone: 966 1 477 4555
Fax: 966 1 478 1451

South Africa

Synergy Computing (Pty) Ltd
Attention: Paul Howard
Phone: 27 21 685 7809
Fax: 27 21 685 7927
E-mail: synergy@synergy.co.za

Asia and Australia

Australia, New Zealand

Facer Solutions
Attention: Kathy Ewart
Phone: 61 2 9484 3979
Fax: 61 2 9484 5709
E-mail: kathye@facer.com.au
Web: www.facer.com.au

Hong Kong

SCS Computer Systems Ltd.
Attention: Steven Lai
Phone: 852 2609 1338
Fax: 852 2607 3042

Singapore, Malaysia

ST Computer Systems & Services Limited
Attention: Toh Tiau Hong
Phone: 65 441 2688
Fax: 65 441 2811
E-mail: tohth@stcs.com.sg
Web: www.stcs.com.sg

North America

Mexico

Infosistemas Financieros SA de CV
Attention: Anita De Urquijo
Phone: 52 5 813 1325
Fax: 52 5 813 3026
E-mail: ifanita@infosel.net.mx