MODULE 12: CONSOLE & TIME/DATE

On completion of this module you will be able to work and use time/date to interact with your program and give information to your user.

MODULE 12.1: CONSOLE COMMANDS

Subject Outcome 1: Introduction

MODULE 12.2: TIME

Subject Outcome 1: Sleep

Subject Outcome 2: Time$ and TIMER

Subject Outcome 3: Randomize Timer

Subject Outcome 4: QTimer & Micro Timer (milliseconds)

Subject Outcome 5: Determine difference between 2 timings

Subject Outcome 6: Watch & Stop Watch

Subject Outcome 7: Convert Seconds-Time and Time-Seconds.

MODULE 12.3: DATE

Subject Outcome 1: Date$

Subject Outcome 2: Rectify Date format

Subject Outcome 3: Determine day of week

Subject Outcome 4: Difference between 2 dates and determine age (from ID)

Subject Outcome 5: Calendars

12.1 INTRO: CONSOLE COMMANDS

These are a group of commands meant for console programming (the old DOS style off-screen coding). This book is however about Windows programming therefore I will not go into too much detail, only those commands applicable to what you would need).

These commands mostly have to do with information gathering (date and time), delaying your programming by specified split seconds or to determine random selected numbers.

12.2 TIME

Time is essential to each program, especially when you are building AI into your program whereby the program have to reason and reconsider/evaluate the data and processes to determine the next logical step – such as games. There are a number of timer functions. Each function will be addressed with its purposes and usages.

12.2.1 SLEEP

This function will **pause** the program for the indicated amount of seconds. Very little or no CPU processing will be passed/used. The values are **1=1 (1 value = 1 second).** You may break it down to less than **1 second by added a point (.)** for instance half a second will be **.5** Once the indicated time has passed, the program will continue the next step. This is especially used when loading data (large data files – databases or when having to process large images/video /music files). It will then give the program some time (in most cases 1 second is more than enough) just enough time to process the action without having to buffer.

Commands	Description
$TYPECHECK ON $INCLUDE <RapidQ2.inc> **dim i as single** **declare sub startseconds** CREATE Form AS QFORMex caption="TIMERS":Width = 640:Height = 480:Center **create start as qbutton** **caption="start timer":left=10:top=10:width=150** **onclick=startseconds** **end create** END CREATE SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal	The timer value numerical holder. The sub program that will count in seconds. The button that will start the event to count in seconds.

Commands	Description

sub startseconds
i=0
do
  i++
  start.caption=str$(i)
  sleep 1
loop until i>3
start.caption="Start Again"
end sub
	Start the event.
	Reset the timer numerical variable.
	Start the count
	Count + 1
	Display the second value in the button.
	Wait one second before you continue.
	If the value of the timer numerical variable is less than 3 then continue to count. If it is more, then stop and reset the name of the button.

As you see in the example above, the program will stop for one second before the next command is executed. Remember however, THE ENTIRE PROGRAM IS PAUSED!!!

12.2.2 TIME$, TIMER, MicroTimer

These functions will all indicate times. See the detail to understand the difference.

12.2.2.1 TIME$

Will determine the system's (the current computer) time in format HH:MM:SS whereby HH is hours, MM is minutes and SS being seconds. We will now compile two example programs, one to display the time and one to determine the values of each part HH:MM:SS and then recorded into numerical variables as they are in actual fact, numbers.

Sno	Commands	Description

1	$TYPECHECK ON
	$INCLUDE <RapidQ2.inc>
	declare sub timenow
	CREATE Form AS QFORMex
	  caption="TIMERS":Width = 640:Height = 480:Center
	  create start as qbutton
	  caption="time now":left=10:top=10:width=150
	  onclick=timenow
	  end create
	END CREATE
	SetWindowLong(Form.Handle, -8, 0)
	SetWindowLong(Application.Handle, -8, Form.Handle)
	Form.ShowModal
	
	sub timenow
	  start.caption=time$
	end sub
		When the user click the button, a sub program will be called that will assign the value of TIME$ to the caption of the button.
		
		Display the time.

2	$TYPECHECK ON
	$INCLUDE <RapidQ2.inc>
	declare sub timenow
	dim hour as single:dim minute as single:dim second as single
	CREATE Form AS QFORMex
	  caption="TIMERS":Width = 640:Height = 480:Center
	  create start as qbutton
	  caption="time now":left=10:top=10:width=150
	  onclick=timenow
	  end create
	END CREATE
	SetWindowLong(Form.Handle, -8, 0)
	SetWindowLong(Application.Handle, -8, Form.Handle)
	Form.ShowModal
		This will record the hours.
		This will record the minutes.
		This will record the seconds.

Sno	Commands	Description
	sub timenow **start.caption=time$** hour=val(left$(time$,2)) minute=val(mid$(time$,4,2)) second=val(right$(time$,2)) **showmessage str$(hour)+":"+str$(minute)+":"+str$(second)** **end sub**	Assign the value to the **hour** variable. Assign the value to the **minute** variable and the seconds to **second**. Remember that these variables are numerical, therefore you need to use the **STR$()** command to display them again. **Later on when we start to calculate the timings**, you will need to **convert** them **as numerical variable** to be **able** to **add** or **deduct**. **You cannot do mathematics with strings**.

12.2.2.2 TIMER

Will determine the amount of seconds that elapsed since midnight.

Sno	Commands	Description
1	$TYPECHECK ON $INCLUDE <RapidQ2.inc> randomize timer declare sub checknumber CREATE Form AS QFORMex caption="TIMERS":Width = 640:Height = 480:Center create start as qbutton caption="since midnight":left=10:top=10:width=150 onclick=checknumber end create END CREATE SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal sub checknumber **showmessage str$(timer)** end sub	Display seconds since midnight.

12.2.2.3 MICROTIMER

This is a much faster counter (every 7 milliseconds) than the standard TIMER (every 15 milliseconds) counter. This timer starts it's counter when you switch on the machine. Take note that you need to call the INC file and to activate you need to assign a numerical variable. The usage of the MICROTIMER is to determine the difference between two actions (start the stop-watch and then stop the stop-watch for instance – measured in milliseconds).

Sno	Commands	Description
1	$TYPECHECK ON $INCLUDE <RapidQ2.inc> $include <microtimer.inc> dim a as double:a=microtimer CREATE Form AS QFORMex caption="TIMERS":Width = 640:Height = 480:Center END CREATE **form.caption=str$(microtimer)** SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal	Include the INC file. Assign a numerical variable to the microtimer. Display current value of microtimer as the FORM's caption.

12.2.3 RANDOMIZE TIMER

This function is used to determine a randomly selected number (a quested number) as restricted by the upper and lower boundaries (minimum and maximum values). The RANDOMIZE TIMER command will start the shuffling of numbers. The RND command will actually determine the current number with a maximum setting.

Sno	Commands	Description
1	$TYPECHECK ON $INCLUDE <RapidQ2.inc> randomize timer declare sub checknumber dim whatnum as single CREATE Form AS QFORMex caption="TIMERS":Width = 640:Height = 480:Center create start as qbutton caption="? number":left=10:top=10:width=150 onclick=checknumber end create END CREATE SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal sub checknumber whatnum=rnd(50) start.caption=str$(whatnum) end sub	Start process. This numerical variable will record and hold the value. Press the button to determine the current number. Assign the current value to WHATNUM – the max number value to be allowed is 50.

There is no minimum number to select from. To add this function we need to code a condition so that if the current value is less than whatever we need, redo until current selected value is more.

Sno	Commands	Description
1	$TYPECHECK ON $INCLUDE <RapidQ2.inc> randomize timer declare sub checknumber dim whatnum as single CREATE Form AS QFORMex caption="TIMERS":Width = 640:Height = 480:Center create start as qbutton caption="? number":left=10:top=10:width=150 onclick=checknumber end create END CREATE SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal sub checknumber checkagain: whatnum=rnd(50) if whatnum<10 then goto checkagain start.caption=str$(whatnum) end sub	Return to this line if the value is less than the minimum. If the value is less than 10 then re-check again until the value is more than 10.

12.2.4 QTIMER

These functions work in the **background**. They are established then activated when needed. When they **reach the indicate interval timing**, then **they will activate** (every so many seconds/minutes/hours then do this) a sub program. These functions work in the **background. Take note of the following rules related to QTIMER:**

- You must **redim** and **use unique variables assignments** (both string and numerical) within the entire sub-program. Be careful of using global variables as they will **conflict** values when the sub-program is called whilst still busy within another sub-program.
- I **advise you to only activate** the (.ENABLED=1) just before the SHOWMODAL or during the ONSHOW event sub-program.
- You **may have multiple TIMERS**.
- **De-activate** (enabled=0) the **other timers** when a sub-program is called by any one of the timers.
- Remember this element works in the **background** so the **rest of your program continues as normal**.

Sno	Commands	Description
1	$TYPECHECK ON $INCLUDE <RapidQ2.inc> **$include <microtimer.inc>** **dim a as single:a=microtimer** declare sub checknumber dim timera as qtimer:timera.interval=10 timera.ontimer=checknumber CREATE Form AS QFORMex caption="TIMERS":Width = 640:Height = 480:Center END CREATE SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) timera.enabled=1 Form.ShowModal sub checknumber **form.caption=str$(microtimer)** end sub	**Activate the sub program** **Establish** the timer and **check every 10th of a second**. When a 10th of a second has been **reached, call sub program CHECKNUMBER.** **Start the TIMERA** element. **Display the value** of the **MICROTIMER** as the FORM's caption.

12.2.5 Determine difference between two timings.

This is now some mathematics to be done. If there was a 100 seconds in a minute, then it would have had been easy, however it is not the case. There is 60 seconds in a minute, 60 minutes within an hour and 24 hours in a day. We are not so much worried about the 24 hours as we are not yet busy with calendars.

There are three formulas you need to learn now:

- Difference between two given times (deduct time).
- Adding time (alarm or reminder setting application).
- Difference between a start and stop time.

12.2.5.1 Difference between two timings (deducting).

The principle of deduction is to minus (-). We need to take in consideration the following two factors:

- The TIME$ (hh:mm:ss) is a string value.
- Time elapses at 00:00:00 and turns to 24:59:59.
- The formula: **FROM TIME – TILL TIME = DIFFERENCE**

Sno	Commands	Description

1	$TYPECHECK ON
	$INCLUDE <RapidQ2.inc>
	dim tyda as string:dim tydb as string
	dim hha as single:dim mma as single:dim ssa as single
	dim hhb as single:dim mmb as single:dim ssb as single
	
	declare sub optiona
	declare sub optionb
	
	CREATE Form AS QFORMex
	Caption = "TIME":Width = 640:Height = 480:Center
	create remark as qlabel
	left=10:top=10
	end create
	create opta as qbutton
	left=50:top=80:caption="1st":onclick=optiona
	end create
	create optb as qbutton
	left=300:top=80:caption="2nd":onclick=optionb
	end create
	END CREATE
	SetWindowLong(Form.Handle, -8, 0)
	SetWindowLong(Application.Handle, -8, Form.Handle)
	Form.ShowModal
	
	sub optiona
	tyda="19:24:54":tydb="07:52:58"
	hha=val(left$(tyda,2))
	mma=val(mid$(tyda,4,2))
	ssa=val(right$(tyda,2))
	hhb=val(left$(tydb,2)):mmb=val(mid$(tydb,4,2))
	ssb=val(right$(tydb,2))
	hha=hha-hhb:mma=mma-mmb:ssa=ssa-ssb
	if ssa<0 then
	mma=mma-1:ssa=ssa+60
	end if
	if mma<0 then
	hha=hha-1: mma=mma+60
	end if
		String variables for two timings.
		The numericals variables to hold the values of hour, minute and seconds. HHA is timing 1 and HHB is for timing b.
		The sub programs (option 1 and option 2)
		
		The label will display the result.
		
		This button will call a timing whereby two times (till) is deducted from another (from) that does not elapse the 00:00:00 time.
		This button will call a timing whereby the time will elapse the 00:00:00 time.
		
		Assign the TILL and FROM times.
		Determine the hour (pos within string)
		Determine the minute.
		Determine the second.
		Determine values for the TILL time.
		
		Deduct straight (hh till – hh from, etc).
		If the seconds result is less than 0 then fix it: minute-1, add 60 seconds to result.
		
		If the minute result is less than 0 then fix it: hour-1 and add 60 minutes.

Sno	Commands	Description

	if hha<0 then hha=24+hha
	remark.caption=str$(hha)+":"+str$(mma)+":"+str$(ssa)
	end sub
	
	sub optionb
	tyda="03:24:54":tydb="07:45:58"
	hha=val(left$(tyda,2))
	mma=val(mid$(tyda,4,2))
	ssa=val(right$(tyda,2))
	hhb=val(left$(tydb,2))
	mmb=val(mid$(tydb,4,2))
	ssb=val(right$(tydb,2))
	hha=hha-hhb:mma=mma-mmb:ssa=ssa-ssb
	if ssa<0 then
	  mma=mma-1:ssa=ssa+60
	end if
	if mma<0 then
	  hha=hha-1:mma=mma+60
	end if
	if hha<0 then hha=24+hha
	remark.caption=str$(hha)+":"+str$(mma)+":"+str$(ssa)
	end sub
		If the hour has elapsed the 00:00:00 time then fix; deduct 24 hours from the result (remember HHA is negative so in actual fact it is 24+(-result).
		
		With this example you will see how the 00:00:00 is elapsed to determine the difference.
		
		The entire formula is the same …

12.2.5.2 Adding two timings to determine how long result is.

The principle of deduction is to add (+). Remember the TIME$ (hh:mm:ss) is a string value.

Sno	Commands	Description

1	$TYPECHECK ON
	$INCLUDE <RapidQ2.inc>
	dim tyda as string:dim tydb as string
	dim hha as single:dim mma as single:dim ssa as single
	dim hhb as single:dim mmb as single:dim ssb as single
	
	declare sub optiona
	declare sub optionb
	
	CREATE Form AS QFORMex
	  Caption = "TIME":Width = 640:Height = 480:Center
	  create remark as qlabel
	  left=10:top=10
	  end create
	  create opta as qbutton
	  left=50:top=80:caption="1st":onclick=optiona
	  end create
	  create optb as qbutton
	  left=300:top=80:caption="2nd":onclick=optionb
	  end create
	END CREATE
	SetWindowLong(Form.Handle, -8, 0)
	SetWindowLong(Application.Handle, -8, Form.Handle)
	Form.ShowModal
	
	sub optiona
	tyda="07:21:42":tydb="03:10:30"
	hha=val(left$(tyda,2))
	mma=val(mid$(tyda,4,2))
	ssa=val(right$(tyda,2))
		Let's add the 3 hours … to the time being 07:21:42 to determine how long the result will be (this is to determine how long, not new time).

Sno	Commands	Description

	hhb=val(left$(tydb,2))
	mmb=val(mid$(tydb,4,2))
	ssb=val(right$(tydb,2))
	hha=hha+hhb:mma=mma+mmb:ssa=ssa+ssb
	if ssa>=60 then
	ssa=ssa-60
	mma=mma+1
	end if
	if mma>=60 then
	mma=mma-60
	hha=hha+1
	end if
	remark.caption=str$(hha)+":"+str$(mma)+":"+str$(ssa)
	end sub
	
	sub optionb
	tyda="23:58:45":tydb="04:45:58"
	hha=val(left$(tyda,2))
	mma=val(mid$(tyda,4,2))
	ssa=val(right$(tyda,2))
	hhb=val(left$(tydb,2))
	mmb=val(mid$(tydb,4,2))
	ssb=val(right$(tydb,2))
	hha=hha+hhb:mma=mma+mmb:ssa=ssa+ssb
	if ssa>=60 then
	ssa=ssa-60
	mma=mma+1
	end if
	if mma>=60 then
	mma=mma-60
	hha=hha+1
	end if
	remark.caption=str$(hha)+":"+str$(mma)+":"+str$(ssa)
	end sub	

12.2.5.3 Determine new time when you add time/deduct time.

The previous examples determined how long, the next example will determine what is the new time when you deduct or add time. We will use the current time (TIME$) and add/deduct some time to determine the till/from time.

What you need to take in consideration is the format of the result. The result will be in hour, minute and second numerical variables. If you have for instance 7 minutes only, then your result will be 13:7:12. The result (RESULT.CAPTION) will be missing a 0 (zero) to make 13:07:12. We will fix this with condition coding when compiling the RESULT.CAPTION display. Basically if the value of the hours/minutes/seconds are less than 10 then we will add a "0" string to the result. Each part of the result (HH / MM / SS) must be done individually.

Sno	Commands	Description

1	$TYPECHECK ON
	$INCLUDE <RapidQ2.inc>
	dim tyda as string:dim tydb as string
	dim hha as single:dim mma as single:dim ssa as single
	dim hhb as single:dim mmb as single:dim ssb as single
	
	declare sub optiona
	declare sub optionb
	
	CREATE Form AS QFORMex
	Caption = "TIME":Width = 640:Height = 480:Center
	create remark as qlabel
	left=10:top=10
	end create
	create opta as qbutton
	left=50:top=80:caption="deduct":onclick=optiona
	end create
	create optb as qbutton
	left=300:top=80:caption="add":onclick=optionb
	end create
	END CREATE
	SetWindowLong(Form.Handle, -8, 0)
	SetWindowLong(Application.Handle, -8, Form.Handle)
	Form.ShowModal
	
	sub optiona
	tyda=time$
	tydb="03:10:30"
	hha=val(left$(tyda,2))
	mma=val(mid$(tyda,4,2))
	ssa=val(right$(tyda,2))
	hhb=val(left$(tydb,2))
	mmb=val(mid$(tydb,4,2))
	ssb=val(right$(tydb,2))
	hha=hha-hhb:mma=mma-mmb:ssa=ssa-ssb
	if ssa<0 then
	mma=mma-1:ssa=ssa+60
	end if
	if mma<0 then
	hha=hha-1:mma=mma+60
	end if
	if hha<0 then hha=24+hha
	if hha<10 then remark.caption="0"+str$(hha)+":"
	if hha>=10 then remark.caption=str$(hha)+":"
	if mma<10 then
	remark.caption=remark.caption+"0"+str$(mma)+":"
	if mma>=10 then remark.caption=remark.caption+str$(mma)+":"
	if ssa<10 then remark.caption=remark.caption+"0"+str$(ssa)
	if ssa>=10 then remark.caption=remark.caption+str$(ssa)
	end sub
	
	sub optionb
	tyda=time$:tydb="04:45:58"
	hha=val(left$(tyda,2))
	mma=val(mid$(tyda,4,2))
	ssa=val(right$(tyda,2))
	hhb=val(left$(tydb,2))
	mmb=val(mid$(tydb,4,2))
	ssb=val(right$(tydb,2))
	hha=hha+hhb:mma=mma+mmb:ssa=ssa+ssb
		The first time (from) will be allocated the current TIME$ value – actual time. We will deduct 3 hours, 10 minutes and 30 seconds (keep on clicking the button to see the result).
		
		If the 00:00:00 has been elapsed, then fix the result.
		Here we will compile the result set by step to add the missing "0" if the result is less than 10.
		
		We will add 04 hours, 45 minutes and 58 seconds to the current time.

Sno	Commands	Description

	if ssa>=60 then
	ssa=ssa-60
	mma=mma+1
	end if
	if mma>=60 then
	mma=mma-60
	hha=hha+1
	end if
	if hha>=24 then
	hha=hha-24
	end if
	if hha<10 then remark.caption="0"+str$(hha)+":"
	if hha<10 then remark.caption="0"+str$(hha)+":"
	if hha>=10 then remark.caption=str$(hha)+":"
	if mma<10 then
	remark.caption=remark.caption+"0"+str$(mma)+":"
	if mma>=10 then remark.caption=remark.caption+str$(mma)+":"
	if ssa<10 then remark.caption=remark.caption+"0"+str$(ssa)
	if ssa>=10 then remark.caption=remark.caption+str$(ssa)
	end sub	

12.2.5.4 Deduct between a start and end time in milliseconds (Deduct Timings)

If you wish to calculate the difference between two times (stop-watch for athletics for instance), the best and most accurate method is to use the MILLISECOND method. This method will add milliseconds to your result and you will not have the 00:00:00 elapse situation (should the race be longer than 24 hours or lapse the 00:00:00 time).

Sno	Commands	Description
1	$TYPECHECK ON $INCLUDE <RapidQ2.inc> $include <microtimer.inc> dim a as double:a=microtimer dim calc4 as double:dim calc1 as double dim tyda as double:dim a5 as string declare sub starta declare sub monitor dim timera as qtimer:timera.enabled=0 timera.interval=10:timera.ontimer=monitor CREATE Form AS QFORMex Caption = "TIME":Width = 640:Height = 480:Center create remark as qlabel left=10:top=10 end create create opta as qbutton left=50:top=80:caption="start":onclick=starta end create END CREATE SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal sub starta tyda=microtimer:timera.enabled=1 opta.caption="00.00:0000"	Include MicroTimer. Activate the micro timer. Use the DOUBLE value for the numerical variables. Start the timer. Display the counter in milliseconds. We will use a timer to monitor the counter – remember the ENABLE=1 will only be activated once the counter starts (start button). The button is pressed to start. Record the current milliseconds and enable the QTIMER. Reset the caption.

Sno	Commands	Description

	do
	loop until (microtimer-tyda)>0.00350
	end sub
	
	sub monitor
	timera.enabled=0
	calc4=(microtimer-tyda)
	if calc4<3600 then a5="00."
	if calc4>=3600 then
	calc1=calc4/3600
	calc1=floor(calc1)
	if calc1>=10 then a5=str$(calc1)+"."
	if calc1<10 then a5="0"+str$(calc1)+"."
	calc4=calc4-(floor(calc1)*3600)
	end if
	if calc4<60 then
	a5=a5+"00:"
	end if
	if calc4>=60 then
	calc1=calc4/60
	calc1=floor(calc1)
	if calc1>=10 then a5=a5+str$(calc1)+":"
	if calc1<10 then a5=a5+"0"+str$(calc1)+":"
	calc4=calc4-(floor(calc1)*60)
	end if
	if calc4<10 then a5=a5+"0"+str$(calc4)
	if calc4>=10 then a5=a5+str$(calc4)
	if mid$(a5,9,1)="." then a5=delete$(a5,9,1)
	opta.caption=left$(a5,11)
	timera.enabled=1
	end sub
		This is required to start the timer and pause the program for a 0.003 second to prevent classes or crash of the processes.
		
		Determine the NOW millisecond and deduct it from the START millisecond. All of the following coding is to determine the Hours, Minutes, Seconds and Milliseconds. Use as is.
		
		Remember all the variables used within this sub-program may not be used within another sub program.
		
		And here you have a stop-watch program. Later on we will add a splitter (record as each athlete completes the race for instance).

12.2.6 Watch and Stop-Watch

You now know and understand timers used by Windows programming. Now let's code a complete watch and stop-watch program. A watch is simple as it displays only HH:MM:SS with an interval of actual second for the timer.

The stop-watch is a bit more complicated with the added millisecond and ability to record each entry as the "racers" completes the race (etc.)

We will add a START button that will start the stop-watch. The watch's result will be displayed by the QLABEL.

Once you press the START button, the SPLIT button will be enabled. Whenever somebody completes the race, click the split button to record that time.

Sno	Commands	Description
1	$TYPECHECK ON $INCLUDE <RapidQ2.inc> **$include <microtimer.inc>** **dim a as double:a=microtimer** **dim calc4 as double:dim calc1 as double** **dim tyda as double:dim tydb as double** **dim a5 as string** **declare sub starta** **declare sub monitor** **declare sub monitor2** **declare sub splitnow** **dim timera as qtimer:timera.enabled=0** **timera.interval=10:timera.ontimer=monitor** **dim timerb as qtimer:timerb.enabled=0** **timerb.interval=1000:timerb.ontimer=monitor2** CREATE Form AS QFORMex Caption = "TIME":Width = 500:Height = 400:Center create remark as qlabel left=10:top=10 end create **create opta as qbutton** **left=50:top=30:caption="START":onclick=starta** **end create** **create optb as qbutton** **left=150:top=30:caption="Split":onclick=splitnow** **enabled=0** **end create** **create splitlist as qlistbox** **left=50:top=60:width=400:height=200** **end create** END CREATE SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) timerb.enabled=1 Form.ShowModal **sub starta** **tyda=microtimer:timera.enabled=1** **opta.caption="00.00:0000":** optb.enabled=1 **do** **loop until (microtimer-tyda)>0.00350** **end sub** sub monitor2 remark.caption=time$ end sub sub monitor timera.enabled=0 calc4=(microtimer-tyda) if calc4<3600 then a5="00." if calc4>=3600 then calc1=calc4/3600 calc1=floor(calc1) if calc1>=10 then a5=str$(calc1)+"."	TimerA is for the stop watch. TimerB is for the standard watch. Start the stop watch. Record the stop watch time. It will only be enabled once the user clicks the START button. Every split will be recorded here. START button was pressed. Enable the split button. Update the watch. Stop watch is updated.

Sno	Commands	Description

	if calc1<10 then a5="0"+str$(calc1)+":"
	calc4=calc4-(floor(calc1)*3600)
	end if
	if calc4<60 then
	a5=a5+"00:"
	end if
	if calc4>=60 then
	calc1=calc4/60
	calc1=floor(calc1)
	if calc1>=10 then a5=a5+str$(calc1)+":"
	if calc1<10 then a5=a5+"0"+str$(calc1)+":"
	calc4=calc4-(floor(calc1)*60)
	end if
	if calc4<10 then a5=a5+"0"+str$(calc4)
	if calc4>=10 then a5=a5+str$(calc4)
	if mid$(a5,9,1)=":" then a5=delete$(a5,9,1)
	opta.caption=left$(a5,11)
	timera.enabled=1
	end sub
	
	sub splitnow
	timera.enabled=0
	splitlist.additems opta.caption+" at time: "+time$+" diff: "+str$(microtimer-tydb)
	tydb=microtimer
	timera.enabled=1
	end sub
		Display the stop watch's time, the current time and difference between the last split and this split.

12.2.6 Converting time

You need to understand two conversions; seconds as time and time as seconds. Here are the formulas for both conversions:

Sno	Commands	Description
1	$include "rapidq2.inc" $TYPECHECK ON **dim calc1 as single** **dim calc2 as single** **dim calc3 as single** **dim calc4 as single** **dim a as string** dim font as qfont:font.name="arial":font.addstyles(fsbold) declare sub checktimenow create form as qformex left=5:top=5:caption="FITMAN - Convert Sec to Time":width=200:borderstyle=4:height=70 FormStyle=fsstayontop **create ab as qedit** **left=65:top=5:inputmask="######":width=45** **onkeyup=checktimenow** **end create** **create abc as qlabel** **left=60:top=30:font=font** **end create** end create form.showmodal sub checktimenow calc4=**val(ab.text)** if calc4<=3599 then a="00." if calc4>=3600 then	Convert seconds as time. As you type the seconds, it will auto convert as time. This is the entire formula to convert to time.

Sno	Commands	Description

	calc1=calc4/3600
	calc1=floor(calc1)
	if calc1>=10 then a=str$(calc1)+"."
	if calc1<=9 then a="0"+str$(calc1)+"."
	calc4=calc4-(floor(calc1)*3600)
	end if
	if calc4<=59 then
	a=a+"00."
	end if
	if calc4>=60 then
	calc1=calc4/60
	calc1=floor(calc1)
	if calc1>=10 then a=a+str$(calc1)+"."
	if calc1<=9 then a=a+"0"+str$(calc1)+"."
	calc4=calc4-(floor(calc1)*60)
	end if
	if calc4<=9 then a=a+"0"+str$(calc4)
	if calc4>=9 then a=a+str$(calc4)
	abc.caption=a
	end sub
		The result is string variable **A.**

2	$include "rapidq2.inc" $TYPECHECK ON **dim calc1 as single** **dim calc2 as single** **dim calc3 as single** **dim calc4 as single** **dim a as string** dim font as qfont:font.name="arial":font.addstyles(fsbold) declare sub checktimenow create form as qformex left=5:top=5:caption="FITMAN - Convert Time to Seconds":width=200:borderstyle=4:height=70 FormStyle=fsstayontop **create ab as qedit** **left=65:top=5:inputmask="##-##-##":width=52** **onkeyup=checktimenow** **end create** **create abc as qlabel** **left=60:top=30:font=font** **end create** end create form.showmodal sub checktimenow **calc4=0** **calc4=val(left$(ab.text,2))*60*60** **calc4=calc4+(val(mid$(ab.text,4,2))*60)** **calc4=calc4+(val(right$(ab.text,2)))** **abc.caption=str$(calc4)** **end sub**	**Converting time to seconds.** As you type the time, it will convert to seconds. The formula to convert time as seconds. The value is determined as **CALC4**.

12.3 DATE

This function will work with a date and calendars. The only true problem that might occur is the format. By default Windows format for a data is mm-dd-yyyy and not dd-mm-yyyy as the RSA standard requires it. Don't worry about this, we will fix this with coding.

12.3.1 DATE$

This function will determine and display/assign a date (to a string variable or numerical variables as year, month and day). The format is a string format. Take note that the format is mm-dd-yyyy.

Sno	Commands	Description
1	$TYPECHECK ON $INCLUDE <RapidQ2.inc> CREATE Form AS QFORMex Caption = "Form":Width = 640:Height = 480:Center **CREATE Label1 AS QLABEL** **Caption = "Date":Left = 86:Top = 58** **END CREATE** END CREATE **label1.caption=date$** SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal	We will assign the value of **DATE$** to the **QLabel**.

12.3.2 Fix date format

As seen the date format is mm-dd-yyyy and we need it to be dd-mm-yyyy. To do this you simply need to code the following fixed coding (you may however alter the names of the variables).

- **Ms3year=right$(date$,4)**
- **Ms3month=left$(date$,2)**
- **Ms3day=mid$(date$,4,2)**

Each variable will hold the value of the three parts of a date (year, month and day).

Sno	Commands	Description
1	$TYPECHECK ON $INCLUDE <RapidQ2.inc> **dim rightdate as string** CREATE Form AS QFORMex Caption = "Form" Width = 640:Height = 480:Center CREATE Label1 AS QLABEL Caption = "Date" Left = 86:Top = 58 END CREATE END CREATE **rightdate=mid$(date$,4,2)+"-"+left$(date$,2)+"-"+right$(date$,4)** label1.caption=rightdate SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal	This string variable will re-order the DATE$ result into the correct format. Formula to correct format and then assign to **RIGHTDATE** string variable.

12.3.3 Determine day of week

It is crucial for a lot of database programs to be able to determine the day of the week. This appear to be a lot of coding, however you may simply copy the DATEV (and variable creations) coding. The values of the EDIT boxes will be replaced by your own element values or variable values.

Sno	Commands	Description
1	$TYPECHECK ON $include "rapidq2.inc" DECLARE SUB datev DIM font AS QFONT:font.Name = "arial" **DIM y AS SINGLE 'last 2 digits of year** **DIM d AS SINGLE 'day of the month** **DIM mc AS SINGLE 'month code** **DIM l AS SINGLE 'leap year** **DIM cc AS SINGLE 'century** **DIM ccc AS SINGLE 'century code** **DIM f AS SINGLE** DIM dayofweek AS STRING CREATE form AS QFORM Height = 200 : width = 300 : center:caption="DAY" CREATE labelday AS QLABEL Left = 20 : top = 22 : caption = "Day" : font = font : font.Size = 9 : font.AddStyles{fsBold} END CREATE **CREATE dayedit AS QEDIT** **Left = 65 : top = 20 : width = 50** **END CREATE** CREATE dayeg AS QLABEL Left = 140 : top = 25 : caption = "eg. 1, 2, 20" : font = font : font.Size = 7 END CREATE CREATE labelmonth AS QLABEL Left = 20 : top = 52 : caption = "Month" : font = font : font.Size = 9 : font.AddStyles{fsBold} END CREATE **CREATE monthedit AS QEDIT** **Left = 65 : Top = 50 : width = 50** **END CREATE** CREATE montheg AS QLABEL Left = 140 : top = 55 : caption = "eg. 1=Jan, 2=Feb, etc." : font = font : font.Size = 7 END CREATE CREATE labelyear AS QLABEL Left = 20 : Top = 82 : caption = "Year" : font = font : font = font : font.Size = 9 : font.AddStyles{fsBold} END CREATE **CREATE yearedit AS QEDIT** **Top = 80 : left = 65 : width = 50** **END CREATE** CREATE yeareg AS QLABEL Left = 140 : top = 85 : caption = "eg. 2001, 2002, etc." : font = font : font.Size = 7 END CREATE CREATE button AS QBUTTON Top = 120 : left = 20 : Caption = "Day of the week" : width = 120 onclick = datev : font = font : font = font : font.Size = 10 : font.AddStyles{fsBold} END CREATE END CREATE form.ShowModal	This is where the actual day will be determined. All these variables are required to calculate the complex formula. This is the final result variable. Enter the day. Enter the month. Enter the year. Call the sub-routine to do calculations

Sno	Commands	Description

```
SUB datev
   y = VAL(RIGHT$(yearedit.Text , 2))
   d = VAL(dayedit.Text)
   cc = VAL(LEFT$(yearedit.Text , 2))
   IF FRAC((VAL(yearedit.Text)) / 4) <> 0 THEN l = 0
   IF FRAC((VAL(yearedit.Text)) / 4) = 0 THEN l = - 1

   IF l = STR$(0) THEN
      IF VAL(monthedit.Text) = 3 THEN
         d = (VAL(STR$(d)) + 1)
      END IF
   END IF
   IF l = STR$(0) THEN
      IF VAL(monthedit.Text) = 4 THEN
         d = (VAL(STR$(d)) + 1)
      END IF
   END IF
   IF l = STR$(0) THEN
      IF VAL(monthedit.Text) = 5 THEN
         d = (VAL(STR$(d)) + 1)
      END IF
   END IF
   IF l = STR$(0) THEN
      IF VAL(monthedit.Text) = 6 THEN
         d = (VAL(STR$(d)) + 1)
      END IF
   END IF
   IF l = STR$(0) THEN
      IF VAL(monthedit.Text) = 7 THEN
         d = (VAL(STR$(d)) + 1)
      END IF
   END IF
   IF l = STR$(0) THEN
      IF VAL(monthedit.Text) = 8 THEN
         d = (VAL(STR$(d)) + 1)
      END IF
   END IF
   IF l = STR$(0) THEN
      IF VAL(monthedit.Text) = 9 THEN
         d = (VAL(STR$(d)) + 1)
      END IF
   END IF
   IF l = STR$(0) THEN
      IF VAL(monthedit.Text) = 10 THEN
         d = (VAL(STR$(d)) + 1)
      END IF
   END IF
   IF l = STR$(0) THEN
      IF VAL(monthedit.Text) = 11 THEN
         d = (VAL(STR$(d)) + 1)
      END IF
   END IF
   IF l = STR$(0) THEN
      IF VAL(monthedit.Text) = 12 THEN
         d = (VAL(STR$(d)) + 1)
      END IF
   END IF
   IF VAL(monthedit.Text) = 01 THEN mc = 1
   IF VAL(monthedit.Text) = 02 THEN mc = 4
   IF VAL(monthedit.Text) = 03 THEN mc = 4
   IF VAL(monthedit.Text) = 04 THEN mc = 0
```

Assign values of the QEDIT fields to the variables for processing.

Sno	Commands	Description
	IF VAL(**monthedit.Text**) = 05 THEN mc = 2 IF VAL(**monthedit.Text**) = 06 THEN mc = 5 IF VAL(**monthedit.Text**) = 07 THEN mc = 0 IF VAL(**monthedit.Text**) = 08 THEN mc = 3 IF VAL(**monthedit.Text**) = 09 THEN mc = 6 IF VAL(**monthedit.Text**) = 10 THEN mc = 1 IF VAL(**monthedit.Text**) = 11 THEN mc = 4 IF VAL(**monthedit.Text**) = 12 THEN mc = 6 IF VAL(STR$(cc)) = 16 THEN ccc = 6 IF VAL(STR$(cc)) = 17 THEN ccc = 4 IF VAL(STR$(cc)) = 18 THEN ccc = 2 IF VAL(STR$(cc)) = 19 THEN ccc = 0 IF VAL(STR$(cc)) = 20 THEN ccc = 6 IF VAL(STR$(cc)) = 21 THEN ccc = 4 f = ((INT(VAL(STR$(y)) / 4)) + (VAL(STR$(d))) + (VAL(STR$(mc))) + (VAL(STR$(l))) + (VAL(STR$(ccc))) + (VAL(STR$(y)))) MOD 7 IF f = 0 THEN dayofweek = "Saturday" IF f = 1 THEN dayofweek = "Sunday" IF f = 2 THEN dayofweek = "Monday" IF f = 3 THEN dayofweek = "Tuesday" IF f = 4 THEN dayofweek = "Wednesday" IF f = 5 THEN dayofweek = "Thursday" IF f = 6 THEN dayofweek = "Friday" button.caption=dayofweek END SUB	Determine the actual day here. Assign the result to the button.

Thanks to **Bruce Williams for this formula (all possible complications were handled during the finalization of this formula.*

Sno	Commands	Description
1	$TYPECHECK ON FUNCTION dow$(d$ AS STRING) AS STRING DIM a AS INTEGER dim leap as single DIM y AS INTEGER DIM m AS INTEGER DIM DOW AS INTEGER DIM dd$ AS STRING leap=0 if frac(val(right$(date$,4))/4)<>0 then leap=1 dd$="SunMonTueWedThuFriSat" print date$ a = (14 - VAL(MID$(d$,1,2))) / 12 : y = VAL(MID$(d$,7,4)) - a a = (14 - VAL(MID$(d$,1,2))) / 12 : y = VAL(MID$(d$,7,4)) - a m = VAL(MID$(d$,1,2)) + 12 * (a - 2) dow = ((VAL(MID$(d$,4,2)) + y + y/4 - y/100 + y/400 + (31 * m)/12) MOD 7) dow$=MID$(dd$,(dow*3)+1,3) END FUNCTION FUNCTION dow(d$ AS STRING) AS INTEGER DIM a AS INTEGER DIM y AS INTEGER DIM m AS INTEGER DIM dd AS INTEGER a = (14 - VAL(MID$(d$,1,2))) / 12 : y = VAL(MID$(d$,7,4)) - a m = VAL(MID$(d$,1,2)) + 12 * (a - 2) dow = (VAL(MID$(d$,4,2)) + y + y/4 - y/100 + y/400 + (31 * m)/12) MOD 7 END FUNCTION dim d$ as string:dim d% as single D$ = dow$(DATE$) D% = dow(DATE$) SHOWMESSAGE d$+" - "+STR$(d%)	This is another formula to determine the date using a FUNCTION command. To call this function you need to use the command DOW(date$) in format mm-dd-yyyy The result is held by D$ (string variable) and day of week (D%) numerical variable.

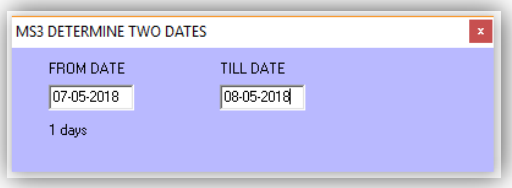
12.3.4 Difference between two dates and age (from ID)

It is important for databases to be able to determine the difference between two dates. In addition, most information is build according to a user/client's ID number. We will use the ID number to determine his/her age for further processing.

12.3.4.1 Determine days between two dates

We will create two QEDIT fields to enter two dates that will then be deducted from each other. We will use a complex formula to determine the total days of each date field, then simply deduct to determine the days in difference. Use the formula as is.

Sno	Commands	Description
1	$TYPECHECK ON $INCLUDE <RapidQ2.inc> dim nname as string dim hha as single:dim hhb as single dim mma as single:dim mmb as single dim ssa as single:dim ssb as single dim dae1 as double:dim dae2 as double declare sub checkform10 declare sub flab1acti2b CREATE Form AS QFORMex visible=0:caption="MS3 DETERMINE TWO DATES" width=400:height=130:left=10:top=10:color=16759225:borderstyle=4 create halab1 as qlabel left=30:top=10:caption="FROM DATE":cursor=-21 showhint=1:hint="click here for today ..." **onclick=flab1acti2b** end create create halab2 as qlabel left=170:top=10:caption="TILL DATE" end create **create haedt1 as qedit** **left=30:top=30:inputmask="##-##-####"** **showhint=1:hint="dd-mm-yyyy":width=70:onkeyup=checkform10** **end create** **create haedt2 as qedit** **left=170:top=30:inputmask="##-##-####"** **showhint=1:hint="dd-mm-yyyy":width=70:onkeyup=checkform10** **end create** create resultime1c as qlabel left=30:top=60:caption="[result difference]" end create END CREATE SetWindowLong(Form.Handle, -8, 0) SetWindowLong(Application.Handle, -8, Form.Handle) Form.ShowModal **sub flab1acti2b** nname=mid$(date$,4,2)+"-"+left$(date$,2)+"-"+right$(date$,4) haedt1.text=nname **end sub** sub checkform10 hha=val(right$(haedt1.text,4)):mma=val(mid$(haedt1.text,4,2)):ssa=val(left$(haedt1.text,2))	Alter date to be today's date if you click on the label. The from date. The till date.

Sno	Commands	Description
	hhb=val(right$(haedt2.text,4)):mmb=val(mid$(haedt2.text,4,2)):ssb=val(left$(haedt2.text,2)) dae1=floor((365*hha)+(hha/4)-(hha/100)+(hha/400)+ssa+((153*mma+8)/5)) dae2=floor((365*hhb)+(hhb/4)-(hhb/100)+(hhb/400)+ssb+((153*mmb+8)/5)) resultime1c.caption=str$(dae2-dae1)+" days" end sub  	Allocate variables. HHA year from MMA month from SSA day from ... **Formula to determine total days for FROM and TILL date. Deduct the two results.**

12.3.4.2 Determine age from ID

RSA ID numbers consists out of 13 numbers of which the first 6 numbers your birthday (starting with year, month then day).

Sno	Commands	Description

1	$TYPECHECK ON
	$INCLUDE <RapidQ2.inc>
	dim jaar as double:dim maand as double:dim dag as double
	dim tester as string:dim datum as string
	datum = MID$(DATE$, 4 , 2) + "-" + LEFT$(DATE$, 2) + "-" + RIGHT$(DATE$, 4)
	declare sub checkform10
	CREATE Form AS QFORMex
	visible=0:caption="MS3 DETERMINE AGE":
	width=400:height=130:left=10:top=10
	color=16759225:borderstyle=4
	create halab1 as qlabel
	left=30:top=10:caption="ID":cursor=-21
	end create
	create idnumberx as qedit
	left=30:top=30:width=120:onkeyup=checkform10
	maxlength=13
	end create
	create resultime1c as qlabel
	left=30:top=60:caption="(age)"
	end create
	END CREATE
	SetWindowLong(Form.Handle, -8, 0)
	SetWindowLong(Application.Handle, -8, Form.Handle)
	Form.ShowModal
	
	sub checkform10
	tester = "0"
	IF VAL(MID$(datum , 9 , 2)) < VAL(LEFT$(idnumberx.text , 2)) THEN tester = "19" + LEFT$(idnumberx.text , 2)
	IF VAL(MID$(datum , 9 , 2)) > VAL(LEFT$(idnumberx.text , 2)) THEN tester = "20" + LEFT$(idnumberx.text , 2)
	jaar = VAL(RIGHT$(datum , 4)) - VAL(tester)
	maand = VAL(MID$(datum , 4 , 2)) - VAL(MID$(idnumberx.text , 3 , 2))
	IF maand < 0 THEN jaar = jaar - 1
	IF maand = 0 THEN
	dag = VAL(LEFT$(datum, 2)) - VAL(MID$(idnumberx.text , 5 , 2))
	IF dag < 0 THEN jaar = jaar - 1
	END IF
	resultime1c.caption=str$(jaar)
	end sub
		The variables required to do the complex mathematics.
		
		The EDIT box to receive the ID number.
		
		The result will be displayed using this QLABEL.
		
		Use this formula as is, ensure to use your correct element that whereby the ID number will be entered/recorded.
		
		The rest are standard.

12.3.5 Calendar

You may write a calendar program to be called via your program or code a calendar to be displayed as part of your program. Take note however that I do not advise you to use a build in calendar as the INC clashes with other INC files. Test your program continuously to see if the INC files don't class.

Sno	Commands	Description
1	$TYPECHECK ON $include "rapidq2.inc" dim cc as double CONST WS_CAPTION = &HC00000:CONST WS_CHILD = &H40000000 CONST WS_VISIBLE = &H10000000:CONST WS_BORDER = &H800000 CONST WS_SYSMENU = &H80000:CONST WS_THICKFRAME = &H40000 CONST WS_SIZEBOX = WS_THICKFRAME CONST WS_CLIPCHILDREN = &H2000000:CONST WS_MINIMIZEBOX = &H20000 CONST WS_MAXIMIZEBOX = &H10000:CONST WS_EX_DLGMODALFRAME = 1 CONST WS_EX_NOPARENTNOTIFY = 4:CONST WS_EX_TOPMOST = 8 CONST WS_EX_ACCEPTFILES = &H10:CONST WS_EX_TRANSPARENT = &H20 CONST WS_EX_MDICHILD = &H40:CONST WS_EX_TOOLWINDOW = &H80 CONST WS_EX_WINDOWEDGE = &H100:CONST WS_EX_CLIENTEDGE = &H200 CONST WS_EX_CONTEXTHELP = &H400:CONST WS_EX_RIGHT = &H1000 CONST WS_EX_LEFT = 0:CONST WS_EX_RTLREADING = &H2000 CONST WS_EX_LTRREADING = 0:CONST WS_EX_LEFTSCROLLBAR = &H4000 CONST WS_EX_RIGHTSCROLLBAR = 0:CONST WS_CLIPSIBLINGS=&H4000000 CONST WS_EX_CONTROLPARENT = &H10000:CONST WS_EX_STATICEDGE = &H20000 CONST WS_EX_APPWINDOW = &H40000:CONST WM_DESTROY = 2 TYPE INITC dwSize AS INTEGER:dwICC AS INTEGER END TYPE DIM icex AS INITC CONST ICC_DATE_CLASSES=&H100 ICEx.dwSize = 8:ICEx.dwICC = ICC_DATE_CLASSES DECLARE FUNCTION InitCommonControlsEx LIB "COMCTL32" _ ALIAS "InitCommonControlsEx" (ICEx AS INITC) AS WORD DECLARE FUNCTION UpdateWindow LIB "USER32" ALIAS "UpdateWindow" _ (hWnd AS LONG) AS LONG create form as qformex end create DIM Handle AS LONG:DIM Calendar1 AS LONG Handle=Form.Handle:CC=InitCommonControlsEx(ICEx) Calendar1 = CreateWindowEx(WS_EX_TOOLWINDOW, "SysMonthCal32", _ 0, WS_CHILD OR WS_VISIBLE OR WS_CLIPCHILDREN OR WS_BORDER OR _ WS_CLIPSIBLINGS, 0, 0, 200, 200, Handle, 0, 0, 0) UpdateWindow(Handle) Form.SHOWMODAL	All of these coding is required to add a calendar to your Window Form. Create your window Assign the calendar to your window (FORM.HANDLE). **The 0,0,200,200 is the position (0,0) and size (200,200)**
2	Run "c:\ms3\mybis\cal.exe"	**Calling a Calendar. This is a duel comparison calendar** that is stored within the c:\ms3\MYBIS folder. Simply add to your program's folder and call it using the RUN command.