"Talktimer" a BASIC program speaking timer for the iPhone

When I started flying, my first transmitter lacked a timer, so landing "out" or running my batteries to low (I fly electric...so far) were typical occurrences. I strapped a $3 Walmart kitchen timer to my wrist, but it only beeped when time was up and I had no sense of how far along in a flight I was at a given moment. Even the more advanced transmitter I have now only beeps once a minute until times up. Somewhere along the line I joined the smart phone world and discovered a BASIC programming tool that would run as an "App" on my phone. I've written a very small program("Talktimer") that will announce elapsed time in minutes. The program takes a user input of some value of minutes and announces the time at this input value, twice that time, and then every minute thereafter. So for example, on a maiden flight where I might want to limit flight time to around 5 minutes, I will input "2" minutes so I get a verbal time message at 2 minutes, 4 minutes and then every minute after that.

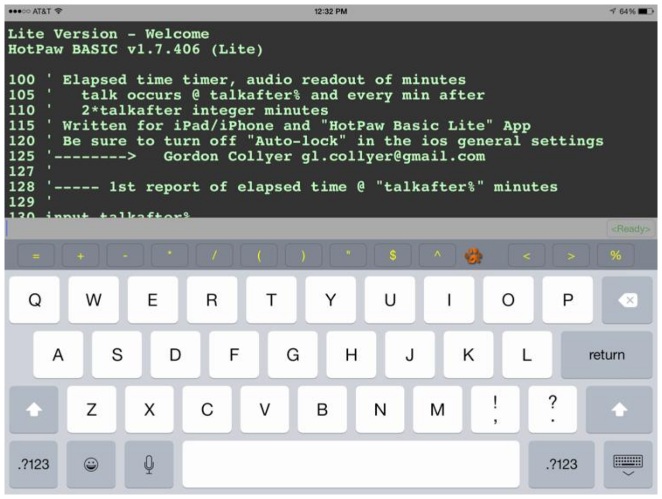
The tool I use is the "Hot Paw" BASIC interpreter for the iPhone. The "lite" version of this App is available free on the Apple App store. To use the program, load and run the App on your iPhone. The program I have written is listed below and may be cut and pasted into the command line bar where the App expects text entry. Type "run" in the command line, hit "return" and a "?" will indicate that a time value may be entered. After that, the running time in minutes will be displayed and an audio elapsed time message will be announced as indicated above. Using the Hot Paw Lite App, the program can be saved to your phone's memory by typing: save "program1.bas". It can be reloaded by typing: load "program1.bas". Be sure to disable the "Auto-lock" feature in the iPhone's "General" settings so that the iPhone won't turn-off and stop the program. The "Hot Paw" web site has more information on the programming App and the BASIC programming language version that is uses. For the younger of the readers, the BASIC programming language began life as a tool for teaching programming, back when the equivalent of the processor in you iPhone would fill a space of several hundred square feet, require a "sign-up" sheet entry and a few minutes waiting in line in order to use the computer for 10 to 15 minutes at a time.

Below is the program listing that can be cut and pasted into the Hot Paw App, and a sequence of screen shots that illustrate running the program.

Questions are welcome, my e-mail is : "gl.collyer@gmail.com".

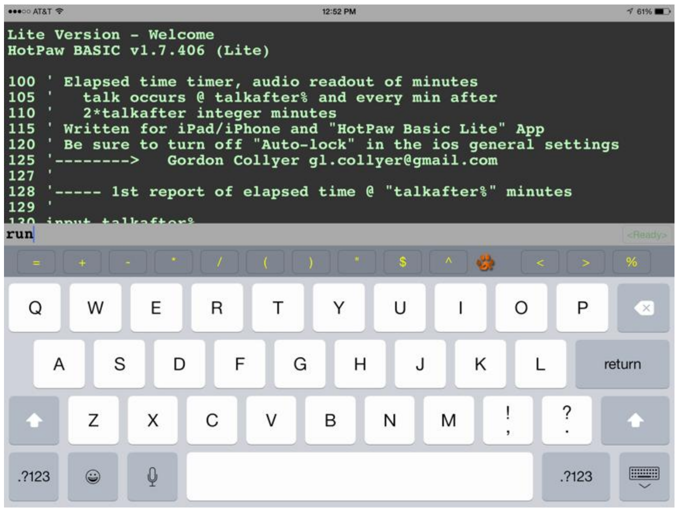
Copy and paste this text, including line numbers into the “iPhone Notes” App to create a version in a simple text format. Then copy and paste this text into the Hot Paw command line window (light grey bar under the large dark grey window). The program should immediately show up in the large dark grey program listing window.

```
100 ' Elapsed time timer, audio readout of minutes
105 '   talk occurs @ talkafter% and every min after
110 '   2*talkafter integer minutes
115 ' Written for iPad/iPhone and "HotPaw Basic Lite" App
120 ' Be sure to turn off "Auto-lock" in the ios general settings
125 '-------->   Gordon Collyer gl.collyer@gmail.com
127 '
128 '----- 1st report of elapsed time @ "talkafter%" minutes
129 '
130 input talkafter%
135 elapse% = 60
140 start% = timer
145 print 0,time$
150 ' loops until 60 sec have elasped
155 while now < start%+60
160     now = timer
165     wend
170 min% = elapse%/60
172 '
173 '---report input minute, 2x input minute and the each
minute after
174 '
175 if (min% = talkafter%) or (min% >= 2*talkafter%) then
180     tout$ = str$(min%) : fn say("t plus") : fn say(tout$)
185     fn say("minutes")
190     endif
195 print min%,time$
200 elapse% = elapse%+60
205 start% = nowhere
210 goto 155
```



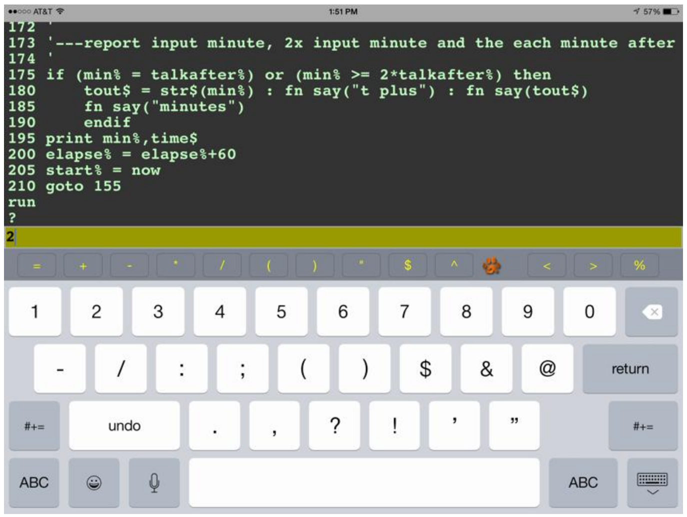
```
Lite Version - Welcome
HotPaw BASIC v1.7.406 (Lite)

100 ' Elapsed time timer, audio readout of minutes
105 '   talk occurs @ talkafter% and every min after
110 '   2*talkafter integer minutes
115 ' Written for iPad/iPhone and "HotPaw Basic Lite" App
120 ' Be sure to turn off "Auto-lock" in the ios general settings
125 '-------->   Gordon Collyer gl.collyer@gmail.com
127 '
128 '----- 1st report of elapsed time @ "talkafter%" minutes
129 '
```

Type “run” in the command line



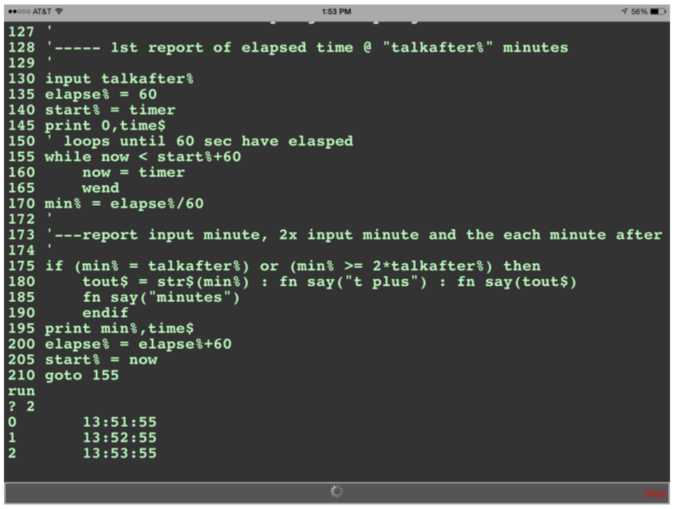
```
Lite Version - Welcome
HotPaw BASIC v1.7.406 (Lite)

100 ' Elapsed time timer, audio readout of minutes
105 '   talk occurs @ talkafter% and every min after
110 '   2*talkafter integer minutes
115 ' Written for iPad/iPhone and "HotPaw Basic Lite" App
120 ' Be sure to turn off "Auto-lock" in the ios general settings
125 '-------->   Gordon Collyer gl.collyer@gmail.com
127 '
128 '----- 1st report of elapsed time @ "talkafter%" minutes
129 '
run
```

Input a time in minutes (whole minutes).



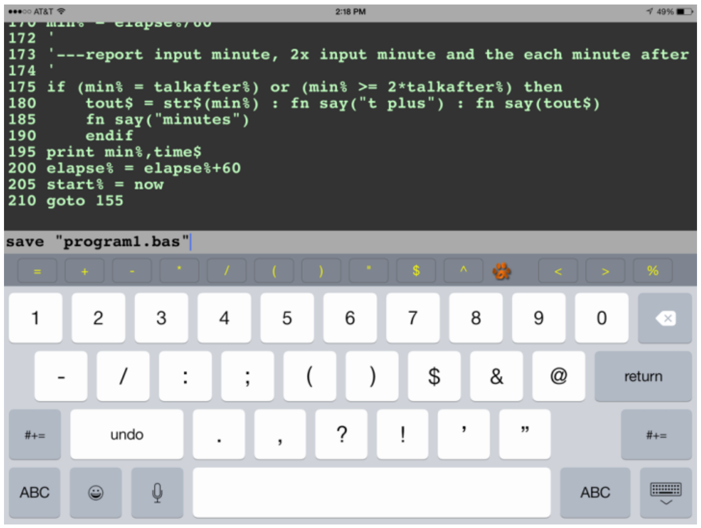
```
172 '
173 '---report input minute, 2x input minute and the each minute after
174 '
175 if (min% = talkafter%) or (min% >= 2*talkafter%) then
180     tout$ = str$(min%) : fn say("t plus") : fn say(tout$)
185     fn say("minutes")
190     endif
195 print min%,time$
200 elapse% = elapse%+60
205 start% = now
210 goto 155
run
?
2
```

The display will show each elapsed minute, pending the audio messages.



```
127 '
128 '----- 1st report of elapsed time @ "talkafter%" minutes
129 '
130 input talkafter%
135 elapse% = 60
140 start% = timer
145 print 0,time$
150 ' loops until 60 sec have elasped
155 while now < start%+60
160     now = timer
165     wend
170 min% = elapse%/60
172 '
173 '---report input minute, 2x input minute and the each minute after
174 '
175 if (min% = talkafter%) or (min% >= 2*talkafter%) then
180     tout$ = str$(min%) : fn say("t plus") : fn say(tout$)
185     fn say("minutes")
190     endif
195 print min%,time$
200 elapse% = elapse%+60
205 start% = now
210 goto 155
run
? 2
0       13:51:55
1       13:52:55
2       13:53:55
```

The program can be saved in the iPhones memory.



```
172 '
173 '---report input minute, 2x input minute and the each minute after
174 '
175 if (min% = talkafter%) or (min% >= 2*talkafter%) then
180     tout$ = str$(min%) : fn say("t plus") : fn say(tout$)
185     fn say("minutes")
190     endif
195 print min%,time$
200 elapse% = elapse%+60
205 start% = now
210 goto 155

save "program1.bas"
```