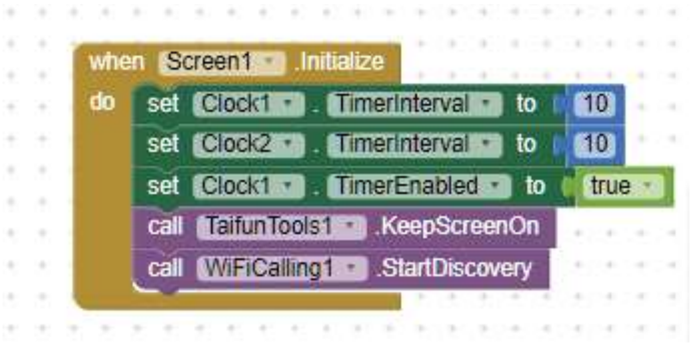
All based on WiFiCalling extension from Kumaraswamy.

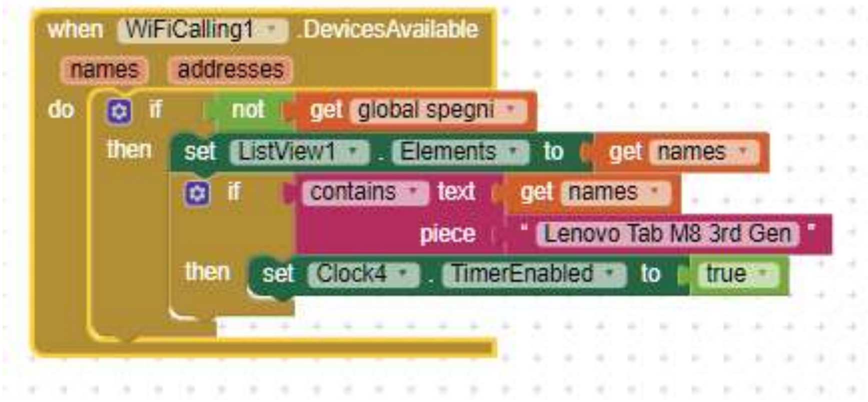
SCREEN 1 INITIALIZE



```
when Screen1 .Initialize
do  set Clock1 . TimerInterval to 10
    set Clock2 . TimerInterval to 10
    set Clock1 . TimerEnabled to true
    call TaifunTools1 .KeepScreenOn
    call WiFiCalling1 .StartDiscovery
```

1. Set 10 ms period of Clock1, used to poll for new data received and enabled
2. Set 10 ms Clock2, parser of buffer received (started once when Message Received event raised). This delay is needed because the data just received cannot be parsed immediately but a delay is required (not pretty clear why, but it works).
3. Keep screen on (Taifun extension)
4. Start Discovery yo find any WiFi peer.

EVENT Devices.Available

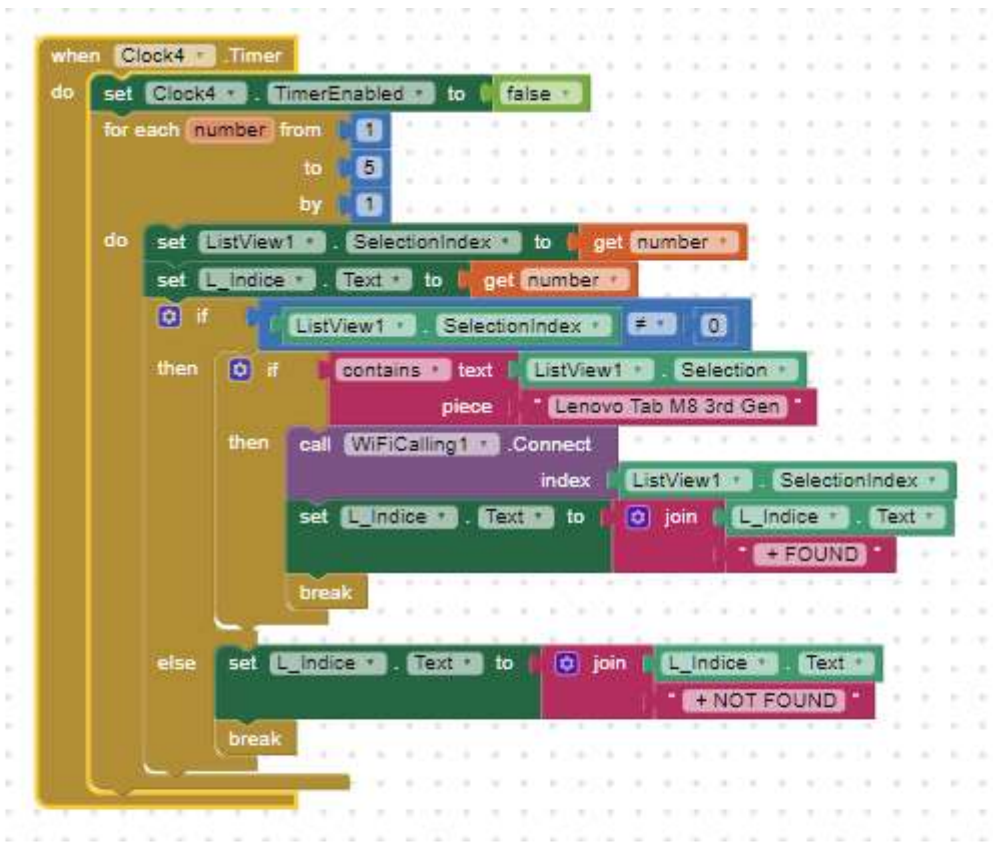


```
when WiFiCalling1 .DevicesAvailable
  names  addresses
do  if not get global spegni
    then set ListView1 . Elements to get names
         if contains text get names
                     piece " Lenovo Tab M8 3rd Gen "
         then set Clock4 . TimerEnabled to true
```

1. Whenever a new WiFi source is discovered this event is raised and if not already connected (bool “spegni”), fills the listview with the WiFi sources names.
2. If the wanted peer’s name is catched, starts Clock4 that parses the listview one element at a time, to find which one contains the name of the wanted peer. See note below.

➢ A delay is required between the finding of the wanted peer name in the listview loaded by this event before it can be passed to the WiFiCalling.Connect method.

CLOCK4 **(**period 2000 ms)

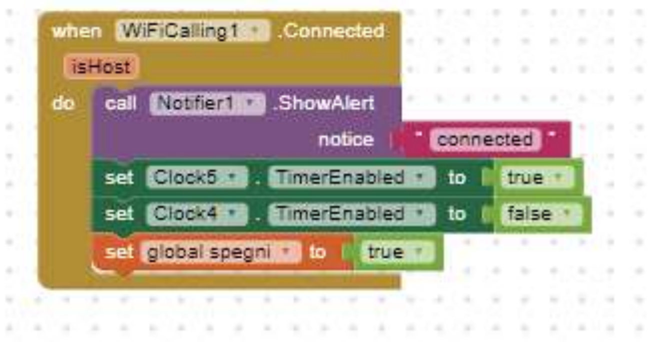


```
when Clock4 .Timer
do  set Clock4 . TimerEnabled to false
    for each number from 1
                       to 5
                       by 1
    do  set ListView1 . SelectionIndex to get number
        set L_Indice . Text to get number
        if  ListView1 . SelectionIndex ≠ 0
        then  if  contains text ListView1 . Selection
                           piece " Lenovo Tab M8 3rd Gen "
              then  call WiFiCalling1 .Connect
                                       index ListView1 . SelectionIndex
                    set L_Indice . Text to join L_Indice . Text
                                                " + FOUND "
                    break
        else  set L_Indice . Text to join L_Indice . Text
                                          " + NOT FOUND "
              break
```

1. Disables itself
2. It parses the listview, one element at a time, to find which one contains the name of the wanted peer. This is required since the WiFiCalling.Connect method wants an index of the listview to connect the relevant peer.
3. Once found, its element number (index) is passed to the WiFiCalling.Connect method and exits from loop. It exits also if no match is found. A label is set accordingly.

- ➢ It is re-enabled if the Discovery method finds a new peer available.
- ➢ The above feature is useful because it retries to connect the selected peer untili it succeeds.
- ➢ Disabled finally by the event Connected. See below.

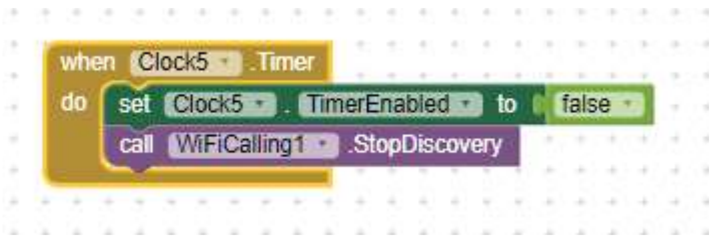
EVENT CONNECTED



```
when WiFiCalling1 .Connected
  isHost
do  call Notifier1 .ShowAlert
                      notice  " connected "
    set Clock5 . TimerEnabled to  true
    set Clock4 . TimerEnabled to  false
    set global spegni to  true
```

1. When the selected peer is connected the Clock4 is disabled and the bool “spegni” that disables further attempts to connect is set.
2. The Clock5 that switches off the Searching method is also triggered. This delay is required otherwise the connection fails (not clear why, but without this delay, it fails).

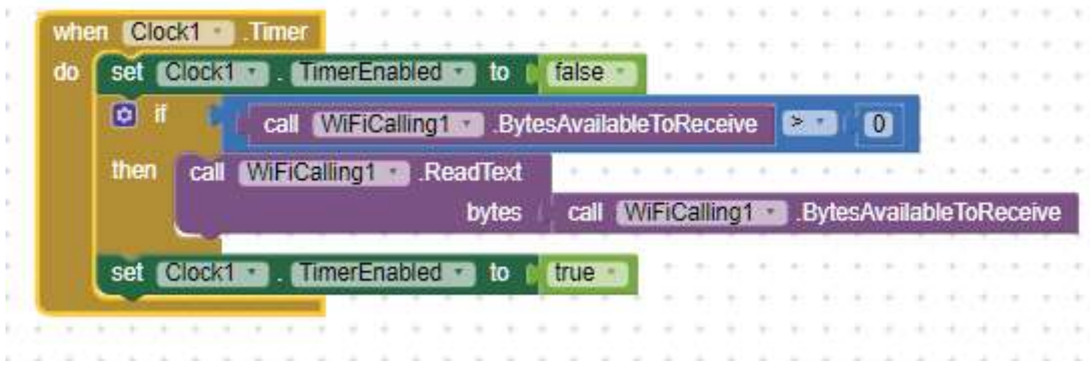
CLOCK5 (period 2000 ms)



```
when Clock5 .Timer
do  set Clock5 . TimerEnabled to  false
    call WiFiCalling1 .StopDiscovery
```

1. Fires only once. Stops the Discovery method, once connected successfully to the wanted peer.

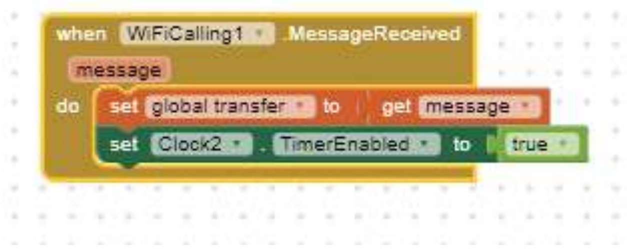
CLOCK1 (period 10 ms)



```
when Clock1 .Timer
do  set Clock1 . TimerEnabled to  false
    if    call WiFiCalling1 .BytesAvailableToReceive  >  0
    then  call WiFiCalling1 .ReadText
                          bytes  call WiFiCalling1 .BytesAvailableToReceive
    set Clock1 . TimerEnabled to  true
```

1. Disables itself
2. Cheks for data in incoming buffer. If any, it reads all the buffer, till the end
3. Re-enables itself

EVENT Message.Received

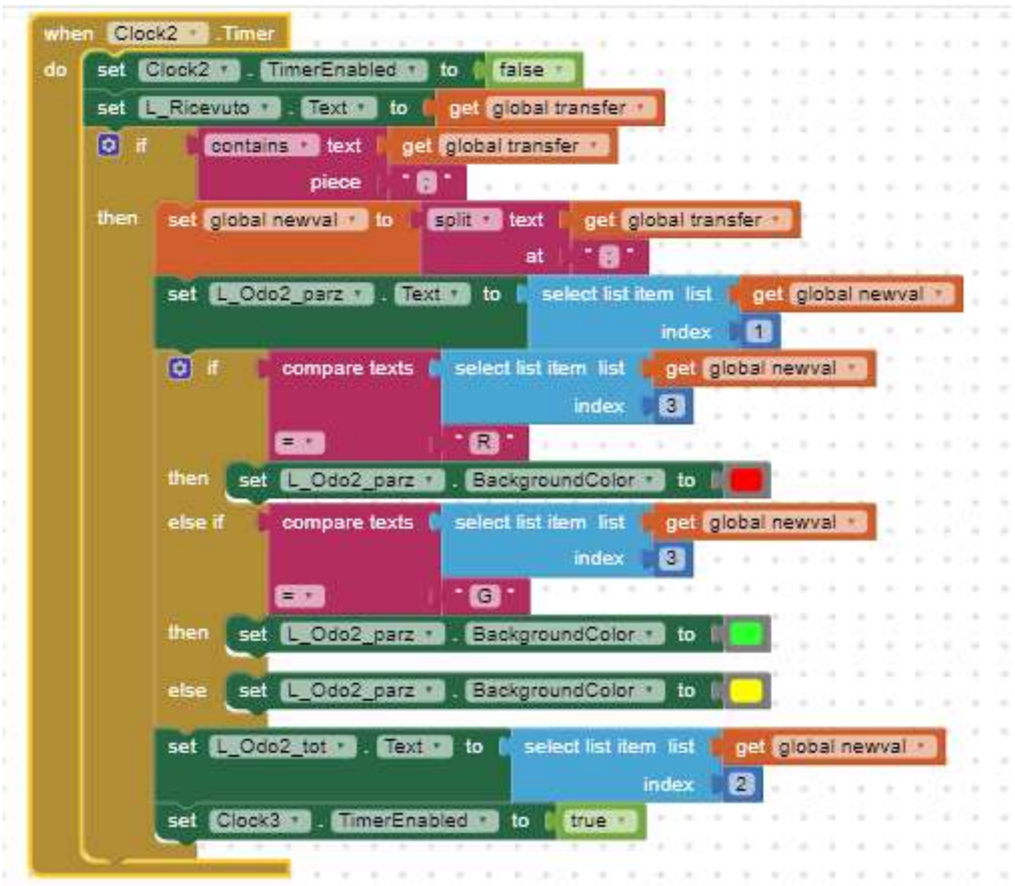


```
when WiFiCalling1 .MessageReceived
  message
do  set global transfer to  get message
    set Clock2 . TimerEnabled to  true
```

- Event raised when a complete message has been received.

1. Moves the read buffer to a working variable
2. Triggers the Clock2 to parse the incoming buffer (see below)

CLOCK2 ((period 10ms)

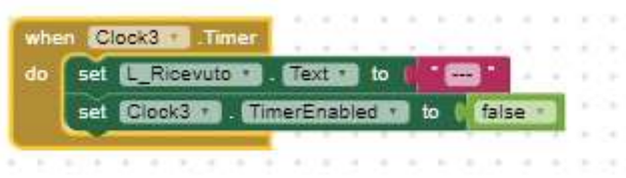


```
when Clock2 .Timer
do  set Clock2 . TimerEnabled to  false
    set L_Ricevuto . Text to  get global transfer
    if   contains text  get global transfer
                  piece  " ; "
    then  set global newval to  split text  get global transfer
                                      at  " ; "
          set L_Odo2_parz . Text to  select list item list  get global newval
                                                      index  1
          if   compare texts  select list item list  get global newval
                                               index  3
                      =     " R "
          then  set L_Odo2_parz . BackgroundColor to  🟥
          else if  compare texts  select list item list  get global newval
                                                 index  3
                      =     " G "
          then  set L_Odo2_parz . BackgroundColor to  🟩
          else  set L_Odo2_parz . BackgroundColor to  🟨
          set L_Odo2_tot . Text to  select list item list  get global newval
                                                     index  2
          set Clock3 . TimerEnabled to  true
```

- Triggered by EVENT Message.Received
- Any time the incoming buffer has been filled and read by Clock1, it can operate.

1. It parses the content of the read buffer in order to find the requested data (to be customized on apps).
2. If the read buffer does not contain the expected data, it starts a CLOCK3 to cancel the previous label and to set it to “---” (just dashes).
3. Disables itself

CLOCK3 ((period 2000ms)



```
when Clock3 .Timer
do  set L_Ricevuto . Text to " --- "
    set Clock3 . TimerEnabled to false
```

- Triggered by Clock2 when the incoming buffer does not contain the acceptable data.

1. Cancels the last shown data and puts some dashes in place
2. Disables itself.

FLOW (in a nutshell)

1. Start the clock for receiving
2. Start discovery
3. Acquire any new peer
4. If the peer name is the wanted one, start the timer to create the index by parsing the listview (without the need to pick manually)
5. When found the name in the list, pass the relevant index to the Connection method.
6. Once connected (it might need several attempts granted by the repetition of the Clock4) start a timer to stop the Discovery of further peers.
7. When the Clock4 elapses, it really stops any new peer acquisition.
8. The data acquisition is demanded to the Message.Received event (while Clock1 receives the raw buffer) which enables Clock2 to parse the buffer.