# ESP 8266, NodeMCU, Lua

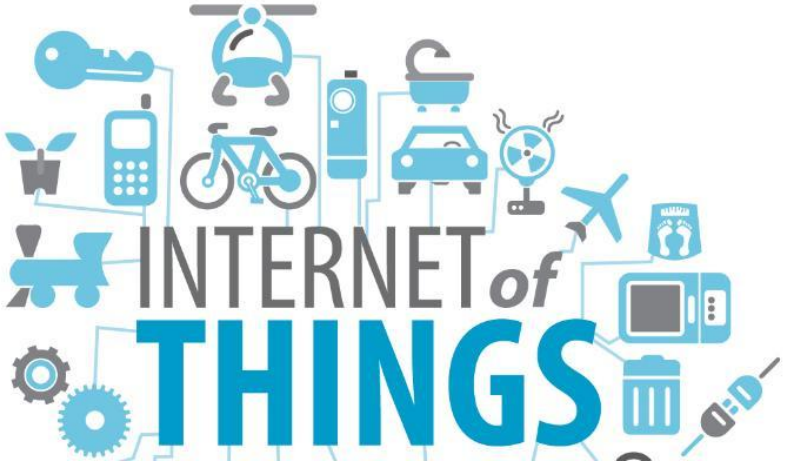
IoT eszköz
és a Lua nyelv


Készítette : Támcsu Péter

# IoT, dolgok internete

Interneten fellelhető adatok keletkezése

- emberek által feltöltött adat
- eszközök által feltöltött adat

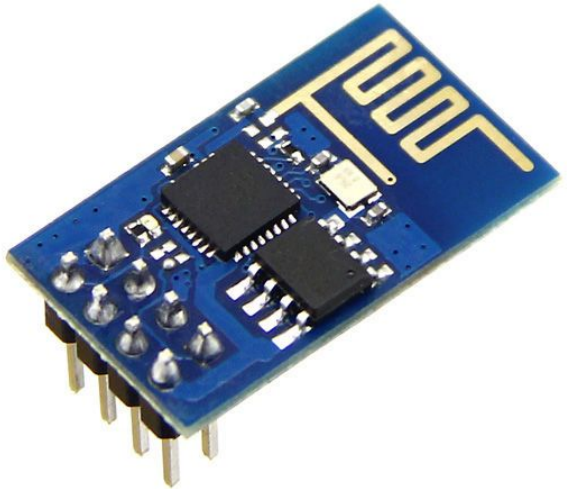



IoT területek

Környezet megfigyelés, infrastruktúra irányítása, gyártás követés, energiahasználat feltérképezése, gyógyászati adatgyűjtés...

# Espressif ESP-8266

- Espressif, olcsó eszköz ami kapcsolódik a WiFi hálózathoz
- kínai gyártó, nagy lökést ad amikor lefordítják angol nyelvre a dokumentációt
- 32 bit RISC CPU (Tensilica Xtensa LX106)
- External flash 512 Kb - 4 Mb
- 16 GPIO, 1 10 bit ADC
- SPI, $I^2C$, UART



# ESP-8266 tipusok

| Modul | Méret | Antenna | GPIO |
|---|---|---|---|
| ESP-01 | 14.3 x 24.8 | PCB antenna | 3 |
| ESP-02 | 14.2 x 14.7 | U-FL csatlakozó | 3 |
| ESP-03 | 17.4 x 12.2 | Kerámia antenna | 7 |
| ESP-04 | 14.7 x 12.1 | nincs | 7 |
| ESP-05 | 14.2 x 14.2 | U-FL csatlakozó | - |
| ESP-06 | 14.2 x 14.7 | nincs | 7 |
| ESP-07 | 22 x 16 | Kerámia + U-FL | 9 |
| ESP-08 | 17 x 16 | nincs | 7 |
| ESP-09 | 10 x 10 | nincs | 6 |
| ESP-10 | 14.2 x 10 | nincs | - |
| ESP-11 | 19.3 x 13 | Kerámia antenna | 2 |
| ESP-12 | 24 x 16 | PCB antenna | 9+ADC |





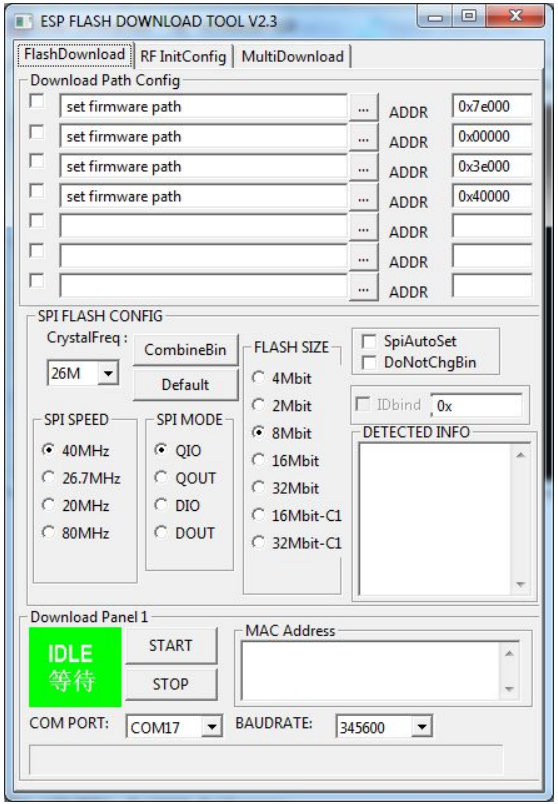




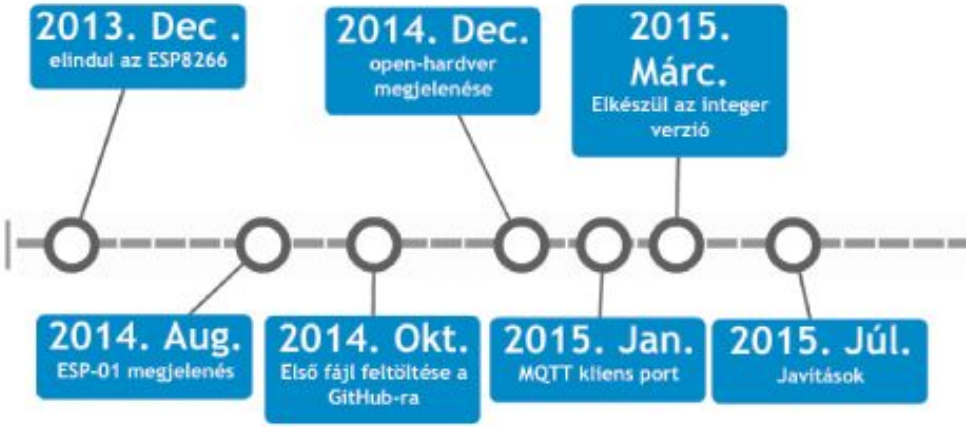


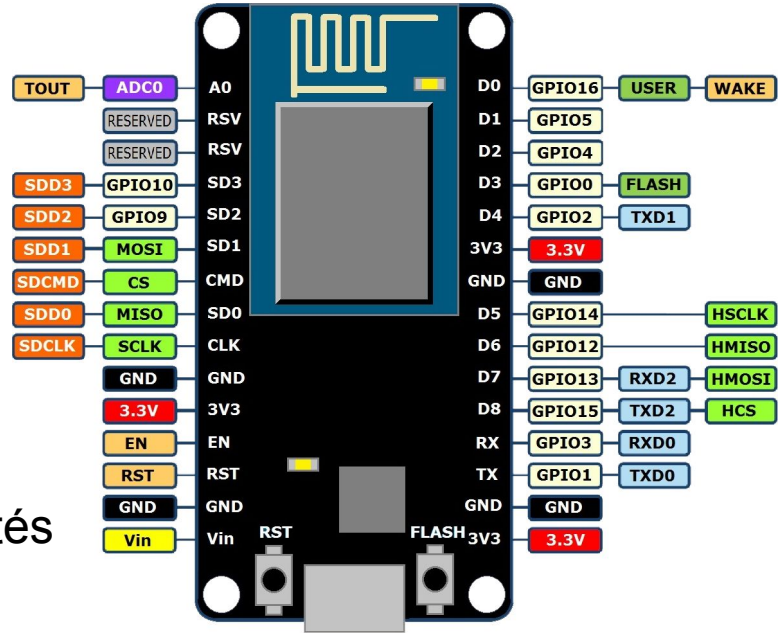

# ESP-8266 bootloaderek

- Eredetileg WIFI soros portnak készült, a firmware Hayes parancsokkal kommunikál a soros porton keresztül egy másik mikrovezérlővel
  pl: `AT+CWMODE=1` WIFI kliens mód
  `AT+CWJAP=ssid,pwd` kapcsolódás az AP-hez
- Arduino IDE-ből is lehet rá fordítani programot, van hozzá kapcsolódó  bootloader-e is.
- NodeMCU firmware,
  LUA programozás lehetséges
- ESP Open SDK gcc-xtensa, lx106-hal
- RTOS SDK



# NodeMCU




- Developer board, egyszerű firmware feltöltés USB-n keresztül.
- Lua futtatása
- EsPlorer IDE, orosz fejlesztő

# Lua scriptnyelv

- Brazil fejlesztés 1993-ban készült az 1.0 a Brazil Katolikus Teológiai Egyetemen, Rio de Janeiro-ban
- 2006 -ban jelent meg az 5.1 (2012. Február 5.1.5)
- szkript nyelv
- beágyazható más programnyelvbe ami legtöbbször C vagy C++ (Delphi, C#, Python ...)
- önálló programnyelvként is használható, REPL
- játékfejlesztők használják (WoW, Corona)
- egyszerű pascal szerű szintaktika
- alternatívák, LLVM lua-jit, eLua
- small footprint, a forrás is “kevés” sor (5.3.1 23 000)
- bytecode-ra fordít és azt futtatja (luac)
- dinamikus típus kezelés

```
pin = 1
gpio.mode(pin,gpio.OUTPUT)
gpio.write(pin,gpio.HIGH)
gpio.mode(pin,gpio.INPUT)
print(gpio.read(pin))
```

# Lua nyelv, típusok, foglalt szavak

## Típusok

- nil
- boolean
- number (double)
- string
- function
- table (asszociatív tömb)
- userdata

## Foglalt szavak

| | | | |
|---|---|---|---|
| and | break | do | else |
| elseif | end | false | for |
| function | if | in | local |
| nil | not | or | repeat |
| return | then | true | until |
| while | | | |

```lua
-- Példa 1
n=11
n=11.12
n=0x11+12
s="szia"
s="'szia'"
s=s .. '"szia"'
print(s)
--> 'szia'"szia"
s="\"szia\nhello\""
print(s)
--> "szia
--> hello"
s="szia" .. " hello"
print(s)
--> szia hello
b=true
print( type(b) )
m=true and 11
--> boolean
print( type(m) )
--> number
n=nil or "hanull"
l=m==11 and "I" or "N" -- (m==11) ? "I" : "N"
```

# Lua nyelv, vezérlő szerkezetek, függvények

**Vezérlő szerkezetek**

ciklusok: while, repeat, for
elágazás: if elseif else (nincs case)
megszakítás: break (continue nincs)

**Függvények használata**

csak function van (nincs procedure)
mivel típus, egy változó is megkaphatja
… nyitott paraméter

```lua
-- Példa 2
i=1
while i<3 do print(i) ; i=i+1 end
print("*",i)
repeat print(i) ; i=i-1 until i==0
print("**",i)
for k in function () i=i+1 ; return i<=2 and i or nil end do print(k) end
print("***",i)
for i=1,2 do print(i) end
print("****",i)
```

```lua
-- Példa 3
i=2
if i==1 then print("egy")
elseif i==2 then print("kettő")
else print("egyéb")
end
```

```lua
-- Példa 4
myprint=function(param)
    print("This is my print function -  ##",param,"##")
end

function add(num1,num2,functionPrint)
    result = num1 + num2
    functionPrint(result)
end
myprint(10)
add(2,5,myprint)
```

# Lua nyelv, table

**Egy és több dimenziós tömb**

tomb={1,2,3} -- egydimenziós
print(tomb[1])

tomb={{1,2,3},{1,2,3}}
print(tomb[1][1]) -- többdimenzós

**Asszociatív tömb**

tomb={ert1=1,ert2=2}
print(tomb[“ert1”])
print(tomb.ert1)

```lua
-- Példa 5
array={"Lua","Tutorial"}
for i=1,2 do print(array[i]) end
--> Lua
--> Tutorial

array={}
for i=1,3 do
    array[i]={}
    for j=1,3 do array[i][j] = i*j end
end
print(array[1][1],array[1][2],array[1][3])
--> 1   2   3
print(array[2][1],array[2][2],array[2][3])
--> 2   4   6
print(array[3][1],array[3][2],array[3][3])
--> 3   6   9

-- Példa 6
mytable = {}
mytable[1] = "Lua"
mytable["wow"] = "Tutorial"
print("mytable Element az index 1 is ",mytable[1])
--> mytable Element az index 1 is Lua
print("mytable Element az index wow is ",mytable["wow"])
--> mytable Element az index wow is Tutorial
print("mytable Element az index wow is ",mytable.wow)
--> mytable Element az index wow is Tutorial
```

# Lua nyelv, tömb bejárása

**Bejárás for ciklussal**
tömb esetén
for i=1,2 do tomb[i] end
for i,v in ipairs(tomb) do end
a sorrend számítható

asszociatív tömb esetén
for k,v in pairs(tomb) do end
a sorrend véletlenszerű

for ciklus működése
for érték,... in func() do end

```
-- Példa 7
tomb={11,22}
for i=1,2 do print(tomb[i]) end
--> 11
--> 22

for i,v in ipairs(tomb) do print(i,v) end
--> 1    11
--> 2    22

tomb={ert1=11,ert2=22}
for k,v in pairs(tomb) do print(k,v) end
--> ert1    11
--> ert2    22

func=function()
        local i=0
        return function()
            if i<2 then i=i+1 ; return i,i*2
            else return end
        end
    end
for v1,v2 in func() do print(v1,v2) end
--> 1    2
--> 2    4
```

# Lua nyelv, láthatóság

ha nincs local deklaráció, akkor globális a változó, kivétel a for ciklusnál
minden esetben a globál változó elérhető _G tömbben
i=1
_G.i
>1
vagy
_G[“i”]
>i

```
-- Példa 8
foo = 1
function bar() foo=2 end
bar() ; print(foo)
--> 2

foo = 1
if true then local foo=2 end
print(foo)
--> 1

foo = 1
function bar() local foo=2 end
bar() ; print(foo)
--> 1

i=1
for i=1,2 do end
print(i)
--> 1
```

# Lua nyelv, osztály helyett table

a : operátor átadja első értéknek a változót ami bal oldalon áll
ha function deklarációban szerepel a : (kettőspont) akkor a self lokális változóba helyezi az első paramétert
Osztaly:new(1,2,3)
megegyezik az alábbival
Oszaly.new(Osztaly,1,2,3)

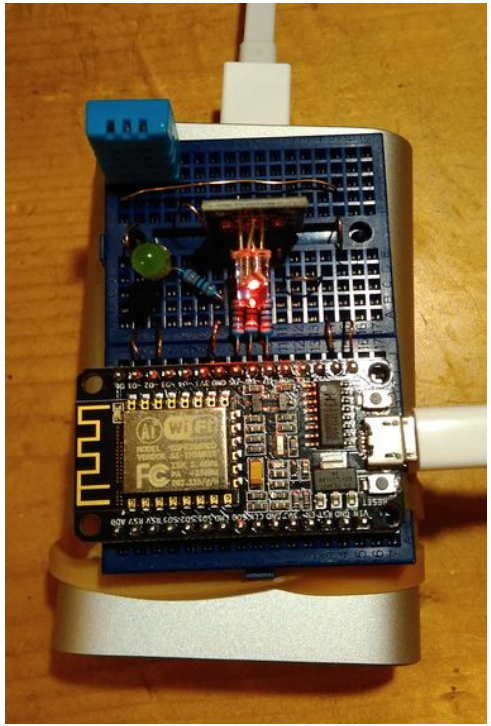
```lua
-- Példa 9
Osztaly={}
Osztaly.__index=Osztaly
Osztaly.atlag=function (self)
                return (self.x + self.y + self.z) /3 end
function Osztaly:new(x,y,z) -- The constructor
    return setmetatable({x=x,y=y,z=z},Osztaly)
end

function Osztaly:osszead()
    return self.x + self .y + self.z
end

o=Osztaly:new(1,2,3)
print(o:osszead())
--> 6
print(o:atlag())
--> 2
```

# Lua, belső modulok

- coroutine
- debug
- file
- io
- math
- os
- package
- string
- table

```lua
-- Példa 10
print(os.clock())
--> 196.967
print(math.floor(1.23))
--> 1
print(math.floor(-1.23))
--> -2
print(table.concat({1,2,3},"\t"))
--> 1    2    3
print(string.upper("szia"))
--> SZIA
print(string.sub("szia",1,-2))
--> szi
print(string.gsub("sziaszia","[ia]","_"))
--> sz__sz__    4
print(string.match("szia","^s") and "I" or "N")
--> I
print(string.format("a szám : %d, a szöveg: %s",12,"szia"))
--> a szám : 12, a szöveg: szia
print(string.find("sziaszia","a"))
--> 4    4
print(string.find("sziaszia","a",5))
--> 8    8
```

# NodeMCU példa, hőmérsékletváltozás jelző



```
-- Példa 10
homerseklet=(function(pin)
                return function()
                    local status,temp,hum=dht.read11(pin)
                    if status==dht.OK then return temp else return nil end
                end end)(1)
rgbled=(function(rpin,gpin,bpin)
                gpio.mode(rpin, gpio.OUTPUT) ; gpio.mode(gpin, gpio.OUTPUT)
                gpio.mode(bpin, gpio.OUTPUT)
                return function(ert)
                    gpio.write(rpin, bit.band(ert,1)~=0 and 1 or 0)
                    gpio.write(gpin, bit.band(ert,2)~=0 and 1 or 0)
                    gpio.write(bpin, bit.band(ert,4)~=0 and 1 or 0)
                end end)(5,6,7)
elozo=homerseklet() ; szamlalo=0
tmr.alarm(0,1000,1,function()
        local hom=homerseklet()
        if not hom then rgbled(0)
        else local dt=hom-(elozo or 0)
            if dt==0 and szamlalo==0 then rgbled(2) -- zöld
            elseif dt>0 then szamlalo=5 ; rgbled(1) -- piros
            elseif dt<0 then szamlalo=5 ; rgbled(4) -- kék
            end if szamlalo>0 then szamlalo=szamlalo-1 end elozo=hom
        end
    end)
```