

7. Hálózati fogalmak, alapok

Visszatekintés

- Számítógépek, információk, számábrázolás, kódolás
- Felépítés, kliens-szerver szerep,fájlrendszerek
- Alapvető parancsok, folyamatok előtérben, háttérben
- I/O átirányítás, szűrők,reguláris kifejezések
- Változó, parancs behelyettesítés,aritmetikai, logikai kifejezések
- Script vezérlési szerkezetek
- Sed,AWK

Mi jön ma?

- Unix-Linux management alapok!
- Hálózati elemek
- ARPANET projekt
 - Szolgáltatások
- Terminál, FTP, HTTP szolgáltatások
 - Későbbi félév során nagy óraszámban a „Számítógépes hálózatok” tárgyban részletesen tanulnak a hálózati lehetőségekről!

Unix-Linux boot

- Boot sector – BIOS-MBR- UEFI
 - LILO, Op. Rendszer választás
- Kernel betöltése
- Init processz indítása (/sbin/init-initctl, systemd-systemctl)
 - /etc/inittab konfigurációs állomány
 - Alapértelmezett futásszint beállítás- Systemd- Systemctl esetén service-ek
 - Futásszintek 0- rendszeráll, 1- single user mód,
2,3,4,5- normál szintek, 6- reboot, 2- default (multi user)
 - A 2-5 szinteket minden rendszer sajátosan definiálja
 - Mi a teendő az egyes futásszinteknél
 - /etc/init.d/rc szint
 - PL: GETTY indítás-terminál service

Unix-Linux management alapok

- Központi management programok
 - SMIT, YAST,YAST2
- Kézi módosítás
 - /etc könyvtár
 - /etc/hosts, /etc/passwd, /etc/shadow, /etc/services
 - /etc/resolv.conf, /etc/sendmail.conf
 - /etc/inetd.conf Internet server konfiguráció
 - /etc/sendmail.cf
 - /etc/httpd.conf- vagy /etc/apache2

Felhasználó management

- Központi adminisztrációs eszköz
 - Létezik minden Unix rendszeren
 - Kényelmes, könnyű használat
 - Hátrány: nagy létszámnál nehézkes, lassú
 - Megoldás: script használat
- Kézi módszer
 - /etc/passwd, /etc/shadow kézi módosítása
 - Létezik adduser, useradd vagy hasonló nevű parancs.
- Példa: useradd.awk (operációs rendszerek)

Önálló gép vs. hálózat

- Ma már elképzelhetetlen a csak egy önálló számítógép!
 - A NETWORK maga a számítógép!
- A kapcsolati elemek az operációs rendszer alapszolgáltatásai között vannak!
- Többféle hálózati kapcsolódási lehetőségek lehetnek!
 - Soros port
 - Ethernet kártya-RJ45
 - WIFI (kártya)
 - Mobil adatkapcsolat (4G,LTE,...)
 - Bluetooth
 - ...

Network - ARPANET

- A hálózat ősprojektje: ARPANET - *Advanced Research Projects Agency Network, 1960-as évek projektje*
 - Mára az utód projekt: NFSNET
- Csomagkapcsolt hálózat- csomagban van a küldő, a cél címe, a csomag sorszáma, adatok.
- Hálózati kapcsolatok, rendszerek decentralizálása fontos!

ARPANET elemek

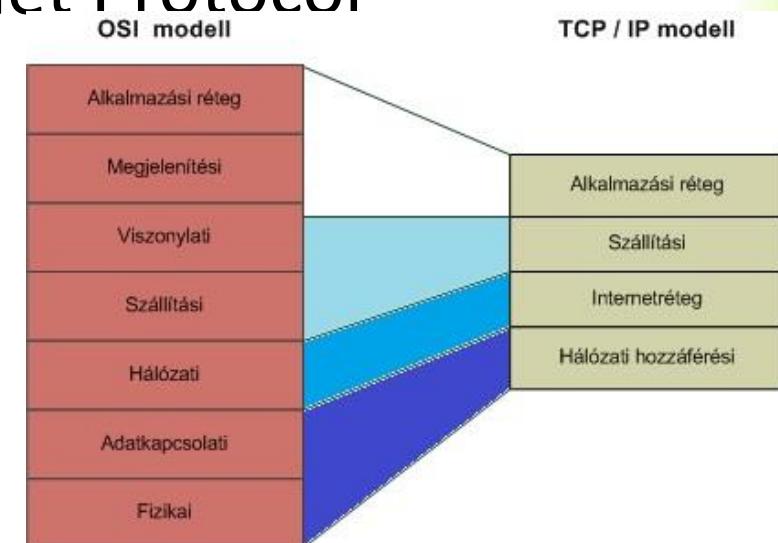
- Network Control Protocol – NCP a kezdeti ARPANET kommunikációs szabvány
- A ma is használt TCP/IP 1983-ban váltotta az NCP-t!
- ARPANET szolgáltatások
 - File transfer (FTP – RFC354, 1971,73)
 - Terminál szolgáltatás (telnet – RFC 137, 1971, RFC854)
 - Erőforrások megosztása (NFS)
 - Üzenetek továbbítása (Mail- RFC524,561 1971,73)
 - Hang továbbítás (NVP) – nem sikeres, ma helyette: VOIP!

Hálózati alapok – OSI modell

- Open Systems Interconnection - ISO/IEC 7498-1.
- 7 réteg – minden réteg a saját feladatát végzi
 - 1. fizikai réteg – ethernet, bluetooth, rs232, stb.
 - 2. adatkapcsolati réteg – PPP,DHCP, L2TP,MAC, stb.
 - 3. hálózati réteg – IPv4,IPv6,AppleTalk,IPSec, stb.
 - 4. szállítási réteg – TCP,UDP
 - 5. munkamenet réteg (session, együttműködés, viszonylati) – SSL, RPC stb.
 - 6. megjelenítési réteg – HTML, CSS
 - 7. alkalmazási réteg – HTTP,SSH,Telnet stb.

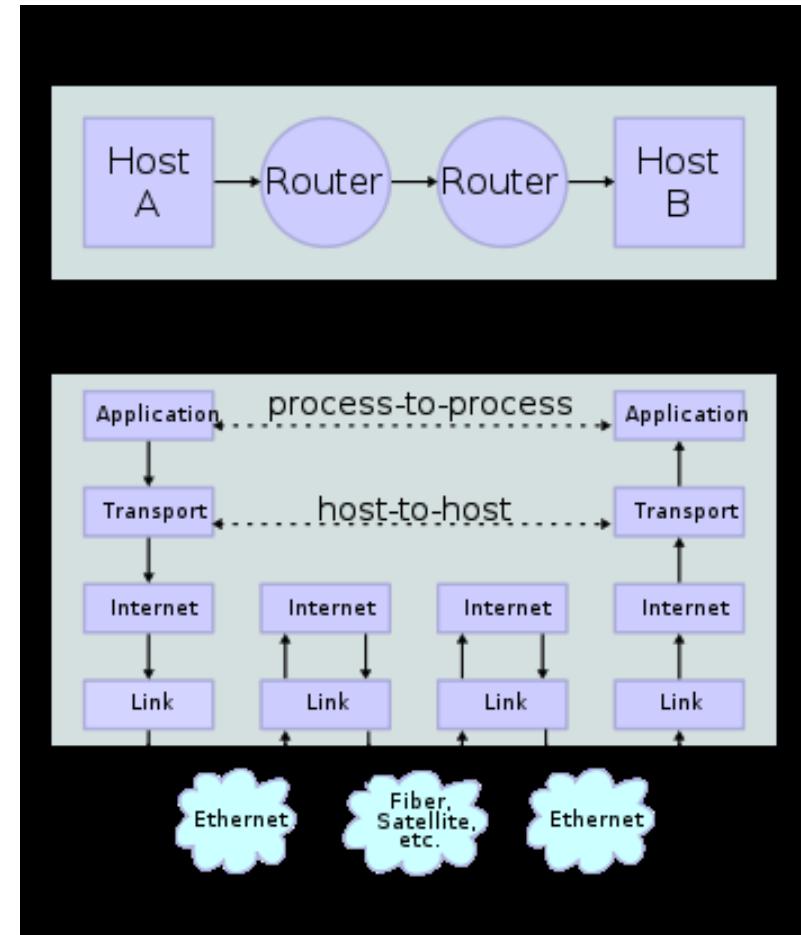
TCP/IP

- A TCP/IP csomagot gyakran „Internet protokoll” csomagként is hívják! – RFC1122,1123,
- TCP- Transmission Control Protocol, IP- Internet Protocol
- A TCP/IP feladatai 4 rétegre oszthatók
 - Alkalmazás réteg
 - Szállítási réteg
 - Internet réteg
 - Hálózati réteg
- TCP/IP leírás: RFC1180- <https://tools.ietf.org/html/rfc1180>



Hálózati kapcsolat példa

- 2 host(gép), 2 útválasztón keresztüléri el egymást!
- Az alkalmazások adatcsatornákba írnak, olvasnak!
- minden más kommunikációs részlet az alkalmazásból nem látszik!



Hálózati topológiák

- Csillag topológia - Ma gyakorlatilag a lokális hálózatok ezt használják! (UTP)
- Fa topológia- a csillagok hierarhikus összekapcsolása
- Gyűrű topológia – Jellemzően gerinc topológiaként használt.
- Sín (busz) topológia – Korábban használt (BNC).
- Lánc topológia – hasonló a sínhöz, de az elem kiesése megszakítja a rendszert!

Hálózati eszközök - SWITCH

- A topológiák végpontjain elhelyezkedő eszközök a számítógépek!
- A csillag topológia központi eleme a kapcsoló, a „switch”!
 - Az azonos lokális hálózatba kötött gépek kapcsolatát biztosítja, OSI 2. adatkapcsolási rétegben van, MAC címek alapján csak a kívánt portra továbbítja a csomagot, ehhez táblázatot használ.
 - Régebben a fizikai rétegben működő Repeater-ek, HUB-ok is használtak voltak!
- Más hálózati kapcsolók is léteznek, Frame Relay, X25 utód, nagy távolságú hálózati kapcsoló, vagy Fibre Chanel ami a SAN rendszerek kapcsolója!

Hálózati eszközök - BRIDGE

- Szintén a 2. réteg eszköze, akár a switch.
- Feladata 2 LAN összekapcsolása!
- Korábban a bridge és switch különböző eszközt jelentett!
- Ma gyakorlatilag a BRIDGE mint eszköz nem kapható!
 - A switch-ek is tudják gyakran ezt a funkciót!
 - Ha az egyszerűbb switch mégse, akkor a router igen!

Hálózati eszközök - ROUTER

- Különböző lokális hálózatok, globális hálózatokat a „router” kapcsol össze!
- A router az OSI modell 3. rétegben helyezkedik el!
 - Döntései alapja: IPv4 (OSI 3. rétegben ismert)
 - A router minden portján egy-egy LAN található, feladata az, hogy egy beérkező csomagról eldöntse, hogy melyik kapcsolódó hálózatba továbbítsa azt, illetve útvonal információk felépítése(routing tábla)!
 - Router protokollok: RIP, OSPF, EIGRP([Enhanced Interior Gateway Routing Protocol](#), CISCO)

IP beállítások

- minden hálózati elem egyedi IP címmel rendelkezik!
- IPv4 protokoll – 4 bájtos(32 bit) IP cím,RFC791
 - 2 részre osztható- Hálózati cím+számítógép(host) azonosító.
 - A osztály: 0.0.0.0-127.255.255.255, 8(1+7)/24 bit
 - B osztály: 128.0.0.0-191.255.255.255, 16(2+14)/16 bit
 - C osztály: 192.0.0.0-223.255.255.255, 24(3+21)/8 bit
 - D osztály: 224.0.0.0-239.255.255.255, 4/28 (groupid), multicast
 - E osztály: 240.255.255.255-247.255.255.255 – foglalt, későbbi használatra.

Számítógép IP címe,neve-DNS

- Ahány hálózati kártya, annyi IP cím él.
- Ha nincs a hálózaton tiltva, a ping ellenőrzi a kapcsolat meglétét!
- DNS- Domain Name Service
 - Név – IP cím társítás
 - szamrend.inf.elte.hu – 157.181.161.38
 - B osztályú cím – elte.hu domain
 - TTL-Time To Live, amíg ez >0 a csomag él!

```
Command Prompt
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\illes>ping szamalap.inf.elte.hu

Pinging szamalap.inf.elte.hu [157.181.161.47] with 32 bytes of data:
Reply from 157.181.161.47: bytes=32 time=5ms TTL=53
Reply from 157.181.161.47: bytes=32 time=8ms TTL=53
Reply from 157.181.161.47: bytes=32 time=5ms TTL=53
Reply from 157.181.161.47: bytes=32 time=5ms TTL=53

Ping statistics for 157.181.161.47:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 5ms, Maximum = 8ms, Average = 5ms
```

IP cím beállítása

- IP – 157.181.161.38 – látható, hogy B osztályú cím!
 - Ettől lehet, hogy még C osztályú alhálózat van!
- Mask: 255.255.255.0 – megadja a LAN (alhálózat)méretét (256)
- Broadcast: 157.181.161.255 – tipikusan a LAN utolsó címe
- Subnet IP: 157.181.161.0 – a LAN első címe, subnet address
- Gateway IP: a kivezető út IP címe (router)
 - /sbin/route -n

IPv4 címek

- Méret: 4 bájt – max. 2^{32} gép
 - Ez nem sok, sőt ma már kevés!
- Mit lehet tenni?
- Kiút: nem „routolható”, lokális címtartományok (RFC1918)
 - A osztály: 10.x.x.x (8/24)
 - B osztály: 172.16.0.0-172.16.255.255 (20/12)
 - C osztály: 192.168.x.x (16/16, 256 darab C osztályú cím)

IPv6

- Születésének fő oka: IPv4 előrelátható szűkössége!
- IPv6 128 bites címeket használ!
 - 8 darab 16 bites szám hexa alakja:
 - 2015:0a0d:0102:1961:0324:fe01:03ab:0405
 - Első 64 bit: subnet prefix
 - Második 64 bit: interfész azonosító
 - Első standard: RFC2640 (1998 dec.)
- IPv4 – IPv6 megfeleltetés
 - 80 bit 0, majd 16 bit 1, utána jön a 32 bit IPv4.

DHCP

- Dynamic Host Configuration Protocol
- IP cím megadása lehet statikus
 - Manuális konfiguráció!
- Dinamikus beállítások megadása(ip,mask,gw)
 - Szerveren: Címtartomány (IP pool) megadás
 - Korlátozások használhatók, pl csak regisztrált (mac cím) gépek kapjanak beállításokat!
 - Statikus IP címek

Szerver elérés

- IP konfiguráció
 - Parancssori lehetőség
 - ifconfig, vagy újabban ip parancs (eth0, lo(opback))
 - Adminisztrációs felület
 - SUSE- yast, IBM-smit, stb.
- Terminál elérés
 - Ssh
 - FTP
 - FTP over SSL
- Webes elérés

FTP(s) vs HTTP(s)

- Mi a különbség a kettő között?
 - File transfer – Hypertext transfer
- A HTTP jóval későbbi szabvány, de ma gyakorlatilag csak ez használt!
- Miért?
 - HTTP segítséggel is letölthetők, feltölthetők állományok!
 - A böngészőben tudunk le és feltölteni!
- FTP – gyorsabb, de telnet-hez hasonlóan külön jogosultság kell!
 - Bináris- szöveges fájl transzfer!

Web publikáció, hitelesítés

- **public_html könyvtár**
 - Módosítható, a httpd.conf állományban
 - Rendszergazda (root) jogosítvány
 - Ha egy könyvtárban van index.html (def. dokumentum), akkor azt adja vissza.
 - Ha nincs index.html, akkor mintegy ftp tartalomjegyzék!
- **Publikus minden, mindenkinek!**

Hitelesítés, jelszó védelem

- Adott könyvtárra érvényes, ha .htaccess fájl létezik a könyvtárban (speciális forma)
- htpasswd, basic, kódolás nincs, apache2 esetén htpasswd2 a név
 - szamrend.inf.elte.hu gépen: /usr/bin könyvtárban
 - Használat: htpasswd2 [-c] filenév usernév
 - -c filenév új állomány lesz
 - Megkérdezi a jelszót, majd a névvel együtt a file-ba rakja kódolva a jelszót
 - Csak első alkalommal kell a –c kapcsoló!

Hitelesítés, védelem II.

- htdigest, MD5 kódolás (apache2 esetén htdigest2)
 - Használat: htdigest2 [-c] filenév azonosító usernév
 - Bőngészőfüggő lehet (IE ?, FireFox ...)
- A „digest” hitelesítéshez a mod_auth_digest modult installálni kell
- A Basic hitelesítés minden szerveren általában engedélyezett!
 - Ha mégsem, akkor az Apache konfigurációs állományban a felhasználó könyvtárakra be kell jelölni az „AllowOverride AuthConfig” lehetőséget!

Apache irodalom

- Apache Web szerver konfiguráció
 - /etc/httpd.conf
 - Vagy újabban: /etc/apache2/ és itt sok kis conf.
- <http://httpd.apache.org/docs/1.3/howto/htaccess.html>
- Google
- Könyvesbolt

.htaccess tartalom adatok

- AuthType Basic
- AuthName "Gyumolcsfa gyujtemeny"
- AuthUserFile /usr/people/XY/public_html/letolt/alma
- Require user alma
- Order deny,allow # sorrend: tiltás, engedélyezés
- Deny from all
- Allow from elte.hu
- allow from 81.82.83.94
- Satisfy any # valamelyik kritériumnak igaznak kell lenni ahoz, hogy hozzáférésünk legyen a tartalomhoz

Példa

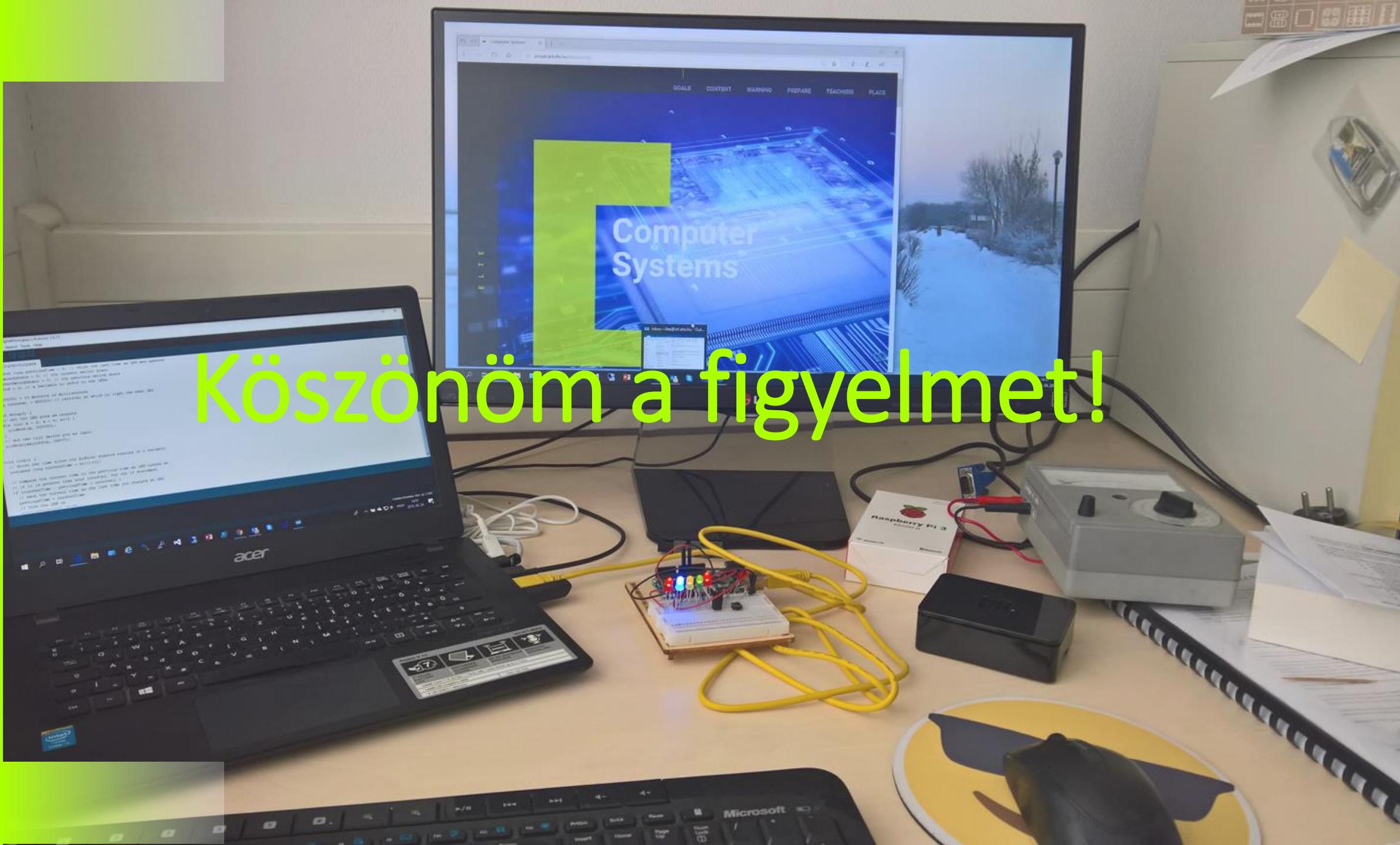
- A legegyszerűbb .htaccess tartalom.
- Ekkor a böngésző kér felhasználói adatokat!

```
ali@os:~/public_html/titkos> cat .htaccess
AuthType Basic
AuthName "Adja meg adatait!"
AuthUserFile
/home/ali/public_html/titkos.pw
Require user alma
```

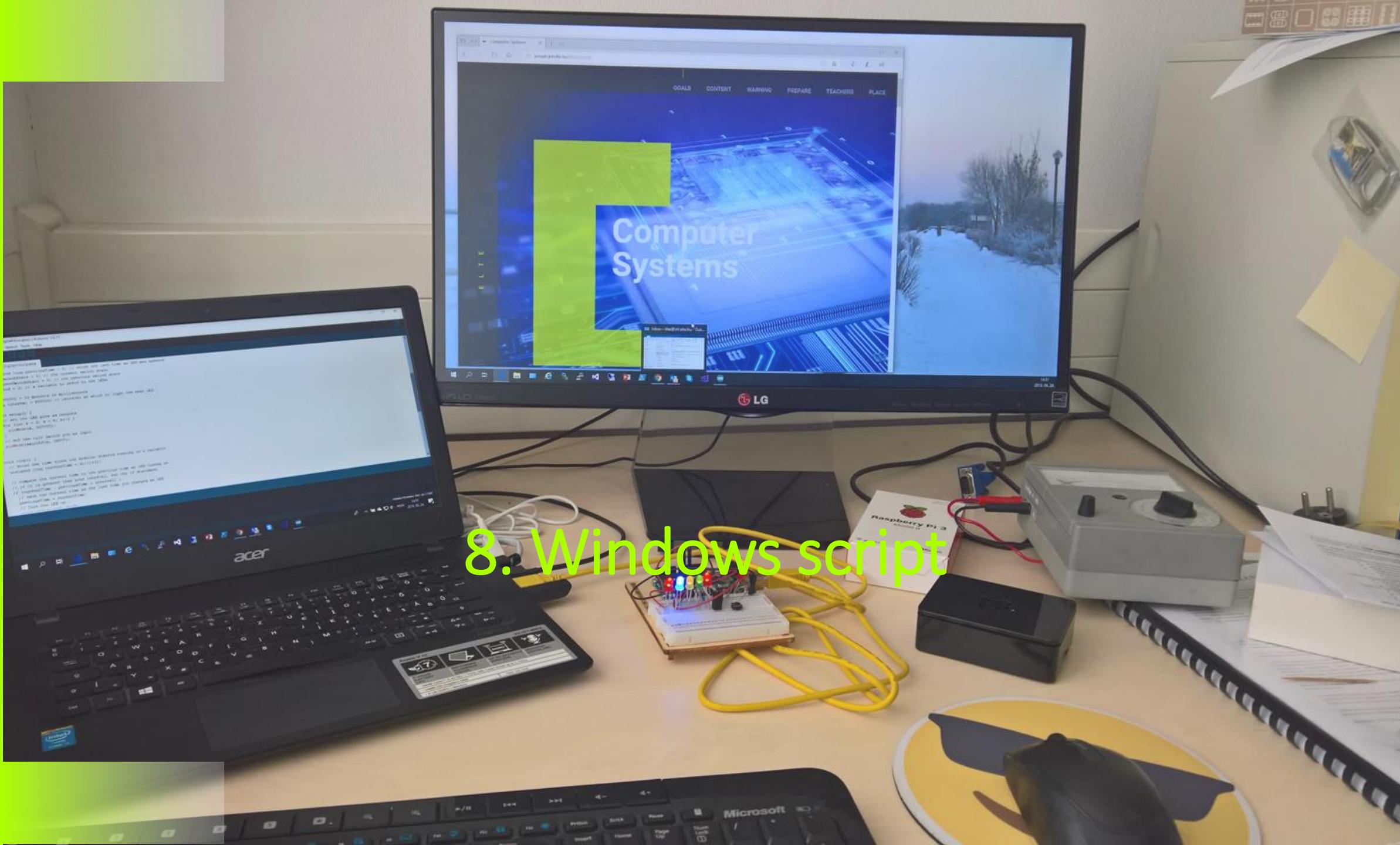
```
ali@os:~/public_html/titkos>
```

Apache config- Virtuális host

- Jelentése: más néven hivatkozunk egy címre
- Httpd.conf
 - Általában /etc könyvtárban
 - Webprogramozáson /etc/apache2 könyvtárban
 - (Suse linux) Nem egy fájl, hanem szét van szedve funkcionális szerint.
 - SSI, CGI jogok
 - Könyvtárra, .shtml kiterjesztés
 - Mod_userdir.conf
 - Virtuális könyvtár
 - Vhosts.d könyvtárban, adott nevű .conf állományok



Köszönöm a figyelmet!



Visszatekintés

- Számítógépek, számábrázolás, kódolás, felépítés, fájlrendszerek
- Alapvető parancsok, folyamatok előtérben, háttérben
- I/O átirányítás, szűrők, reguláris kifejezések
- Változó, parancs behelyettesítés, aritmetikai, logikai kifejezések
- Script vezérlési szerkezetek, SED, AWK
- Hálózati szolgáltatások

Mi jön ma?

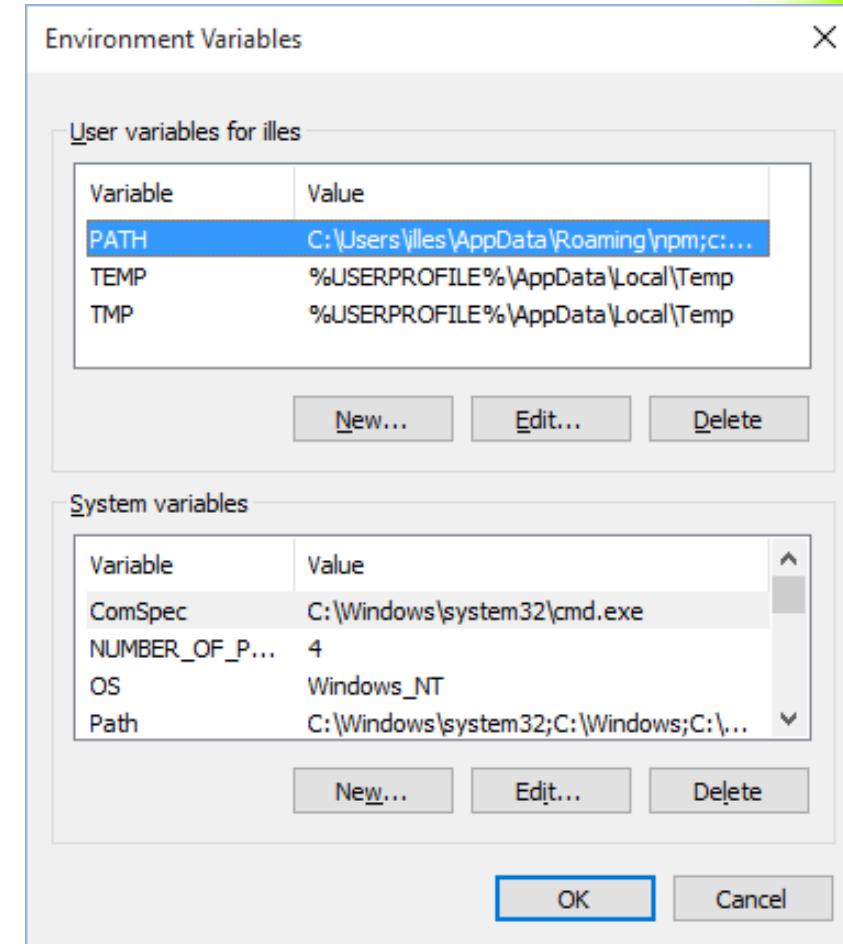
- Windows kiszolgáló
 - Szolgáltatások
- Műveletek automatizálása
 - Batch parancsok
 - Batch hiányosságok
- Windows Script Host
- PowerShell

Windows operációs rendszer

- Múlt: IBM PC DOS – Disk Operating System
 - 86-DOS – egy CP/M klón, Seattle Computer Products
 - A Microsoft ezt megvette 1981-ben.
 - Átnevezéssel: MS-DOS , 1981
- 1984 – Microsoft Windows mint az MS-DOS grafikus kiterjesztése.
- 1993 – 32 bites rendszer, Windows NT
- 1995 – Windows 95, Windows NT 4.0
- Jelen: 2015-2020-2021 – Windows 10,11

Windows kezelőfelület

- Grafikus felület, parancsok indítása – egy ikonra kattintás
 - Start menüpont, task bar, desktop
- Parancssor ablak (cmd) létezik
 - minden parancs, installált program parancssorból is indítható!
- Path környezeti változó 2 részből áll!
 - User path + system path



Van script készítési lehetőség Windows rendszerben?

- Igen!
- Elsőszorban adminisztráció megkönnyítésére
 - Hasonlóan a UNIX, Linux alatti lehetőséghez!
- Többféle script létezik MS Windows!
 - Batch program (alapok)
 - Windows Script(ing) Host (VB Script vagy JScript alapú)
 - PowerShell

Batch program alapok

- Az MS DOS rendszer részeként jelent meg!
- Fő feladata: Parancsok összegyűjtése és indítása egy parancs a ‚batch’ parancs segítségével.
- Ma is használt, bár ritkán!
- Szöveges parancsok
- Fájlnév kiterjesztés: .bat
- Megjegyzés: rem
- echo utasítás
- call masik.bat

Batch változók, paraméterek

- Kis-nagybetű azonos!
- Változó definiálás: set a=5
- Összes változó kiírása: set
- Változó értéke: %név%, Pl: %a%
- path parancs, %path% változó
- prompt utasítás
- %1, %9 a batch paraméterei
- %0 a batch program neve
- shift parancs, balra tolja a paramétereket

Batch vezérlési szerkezetek

- Címke definiálás: :címke1
- Ugrás egy címkére: goto címke1
- Elágazás: if [not] feltétel utasítás
 - if errorlevel 5 goto ot
 - Igaz, ha az előző parancs visszatérési értéke nagyobb vagy egyenlő mint 5!!
 - if %a%=="5" goto 5
 - if exist fájlnév goto hat
- Ciklus: for %%változó in (lista) do (utasítás)

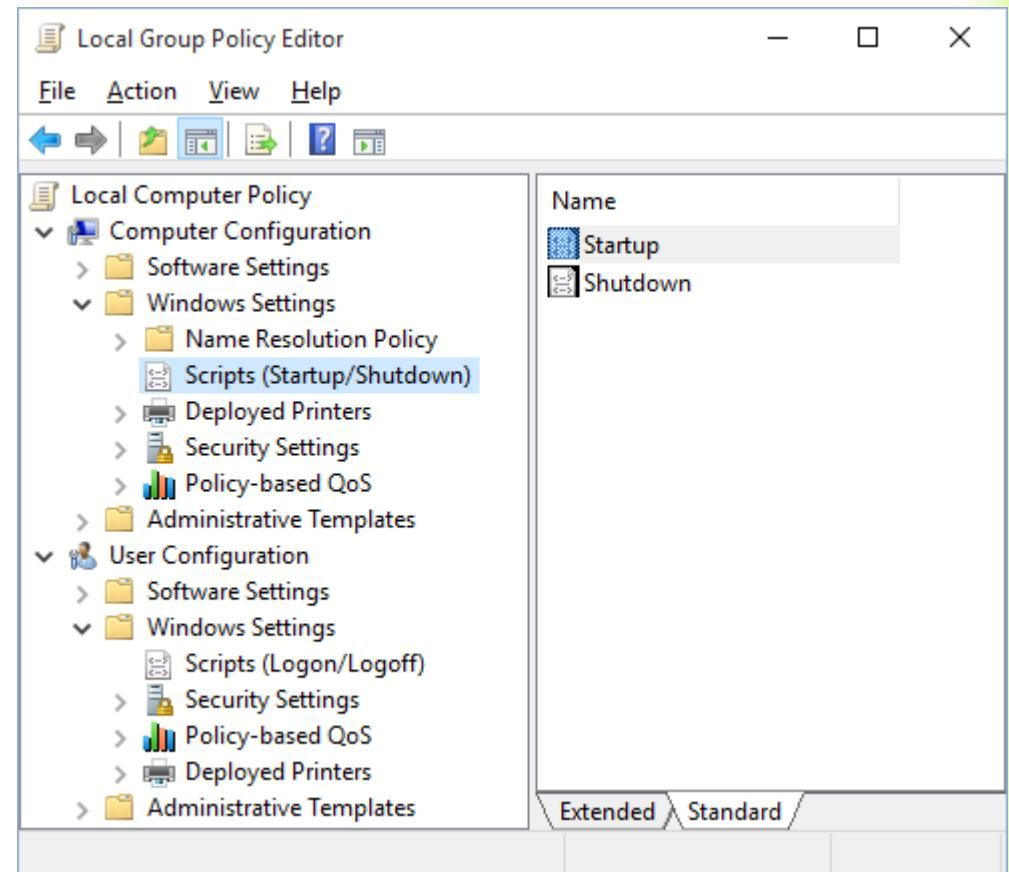
Batch ciklus példa

- Hasonlít a shell script for ciklusához
- XP-ben /f kapcsoló

```
rem for ciklus példa
rem ne írja ki a parancsokat
@echo off
rem %%név formát kell használni!!!
for %%i in (alma korte) do (
    echo %%i
)
rem egy file sorainak első szavait vegyük
for /f %%j in (alma.txt) do (
    echo %%j
)
```

autoexec.bat vs. GPEDIT.MSC

- Feladata: belépéskor automatikusan lefutó parancsok (batch) gyűjteménye!
- Mára helyét átvette a csoportos házirend készítés lehetősége!
 - Windows startup/shutdown
 - User startup/shutdown



Windows Script Host - VBS

- Windows 2000 szerverben jelenik meg – új script nyelv
 - Ami nem új, hiszen az MS Office 97-ben jelent meg mint új makró nyelv!
 - Általános script környezet (Visual Basic Script, JScript bár ez ritkán használt)
 - A mai Office környezet makró nyelve ugyanez!
- .vbs kiterjesztés (.js is lehet, Jscript)
- Leírás: MSDN dokumentáció
 - [https://msdn.microsoft.com/en-us/library/9bbdkx3k\(v=vs.84\).aspx](https://msdn.microsoft.com/en-us/library/9bbdkx3k(v=vs.84).aspx)

WSH példa

- Készíteni kell egy .vbs vagy .js kiterjesztésű parancsállományt!
- A WSH objektumelvű, Wscript a főobjektum!
 - Echo metódus
- Futtatás: Parancssorból vagy egér kattintás

The screenshot shows a Microsoft Notepad window titled "elso.vbs - Notepad". The menu bar includes File, Edit, Format, View, and Help. The main content area contains the following VBScript code:

```
WScript.Echo("Hajrás Fradi!")
```



Mi van a WSH után?

- Miközben a WSH ma is használt script nyelv, a .NET Framework megjelenése után új script eszköz jelenik meg!
- .NET Framework 2.0, 2005
- **PowerShell**

PowerShell

- A Microsoft új generációs script nyelve
 - Batch, VBScript, WSH utód
- Ingyenes, utólag kellett installálni XP, Vista alá, a WS 2008, 2012, Win7,Win8,Win10 része
- Letölthető: <http://www.microsoft.com/> letöltések oldaláról.
Ingyenes!
- Jelenlegi verzió: 3.0 (Windows 8), 4.0 (Win 8.1), 5.0 (Win10)
 - Windows 7 alatt a Powershell 2.0 érhető el, de frissíthető!
 - Windows Management Framework 4 is elérhető Win7 alá!
 - Get-PSSnapin – Megadja a PS verziót!

Powershell elérhetősége - Install

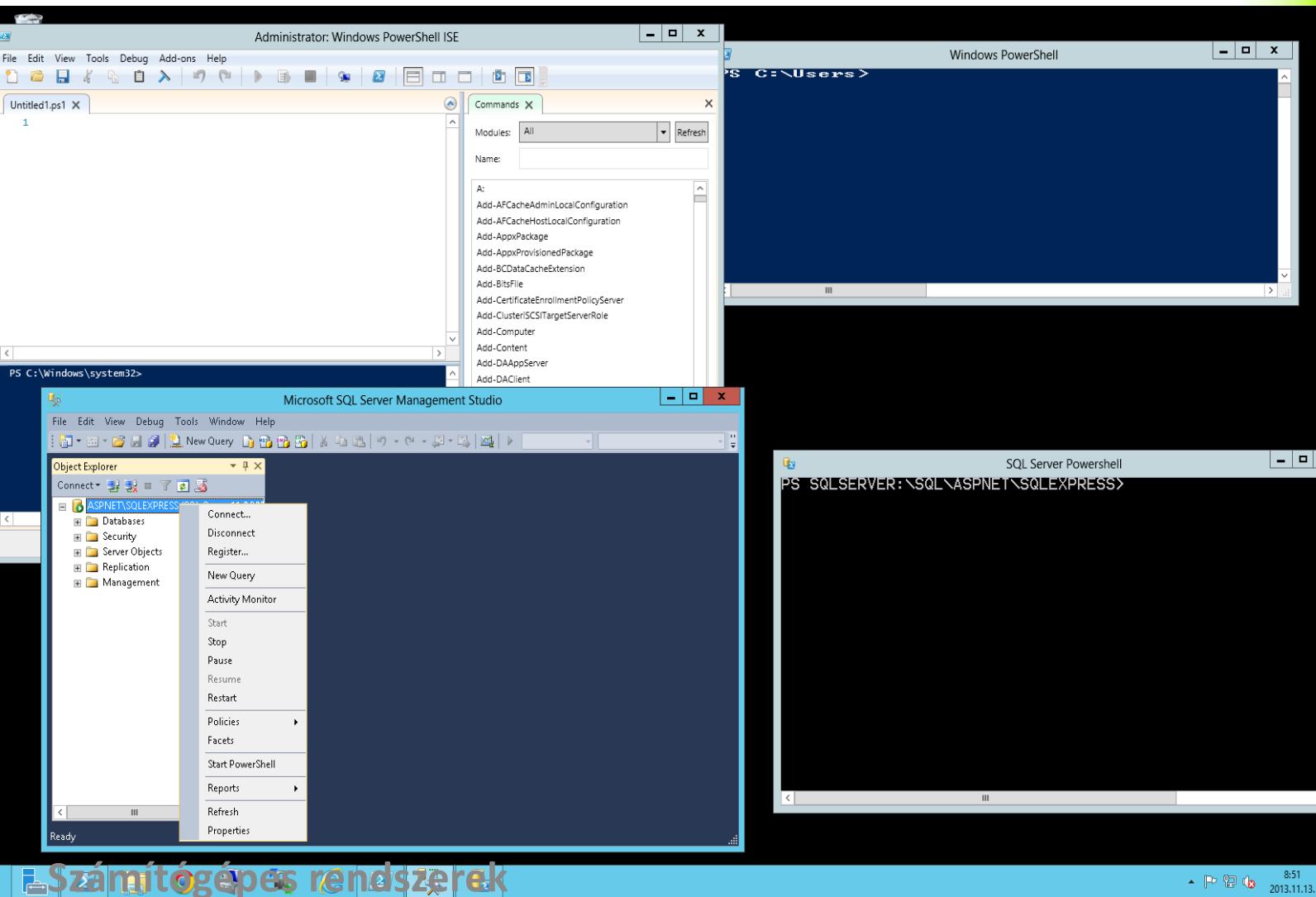
- .NET FrameWork 2.0 szükséges
- Start- Programok- Windows Powershell programcsoporthoz hozzáférhető
- Windows XP alá kell külön installálni!
 - PowerShell 1.0 Documentation Pack
- Dokumentáció: online help
- A parancs egy cmd (command.com) –hoz hasonló karakteres ablakként jelenik meg, vagy grafikus felületen (ISE).

Mire jó a PowerShell?

- Mint a shell script!
- Elsősorban menedzsment célra
 - Első hivatalos MS tanfolyam: Course 6434A (2008)
 - Automating Windows Server® 2008 Administration with Windows PowerShell
 - Általános script programozási környezet Windows operációs rendszer alatt.

Powershell „változatok”

- Alap (Core), minden operációs rendszer része
- A szolgáltatások (MS SQL, Web, Exchange) saját kiegészítést (modult) adnak az alap rendszerhez!



PowerShell (core) ma

- Szokásos grafikus lehetőségek (File, Edit, View)
- Tools – munka ablak font, szín beállítások.
- Debug – Szokásos nyomkövetési lehetőség!

The screenshot shows the Windows PowerShell ISE interface. In the top-left window, the command `get-PSS` is typed, and a dropdown menu lists several suggestions: `Get-NetIPHttpsState`, `Get-PSSession`, `Get-PSSessionConfiguration`, and `Get-PSSnapin`. The `Get-PSSnapin` option is highlighted with a blue selection bar. To the right of the dropdown, a detailed help pane displays the command's synopsis: `Get-PSSnapin [[-Name] <string>] [<CommonParameters>]`. The bottom-left window shows a command prompt with the text `PS C:\Users\illes>`. The bottom-right corner of the interface shows zoom controls at 185%.

PowerShell Debug

- F9- Toggle Breakpoint
 - Szokásos step over, stb.
 - Csak ISE-n belül(F5)!!!
 - Parancsablakban indítva nincs a töréspontnak hatása
- Kiírathatók a script változók!
- Speciális változók:
 - A debugger belső pipeline változói!
 - \$Args
 - \$MyInvocation
 - \$Input
 - \$_
 - \$PSBoundParameters

The screenshot shows the Windows PowerShell ISE interface. The top menu bar includes File, Edit, View, Tools, Debug, Add-ons, and Help. The toolbar below has various icons for file operations. A script named 'egy.ps1' is open in the editor, showing PowerShell code. The code defines a variable '\$hajra', loops from 1 to 10, prints each iteration's index and a string, initializes '\$max=0', sets '\$nev="almás"', and processes directory contents. The line '\$nev="almás"' is highlighted in orange. The bottom pane shows the PowerShell session output:

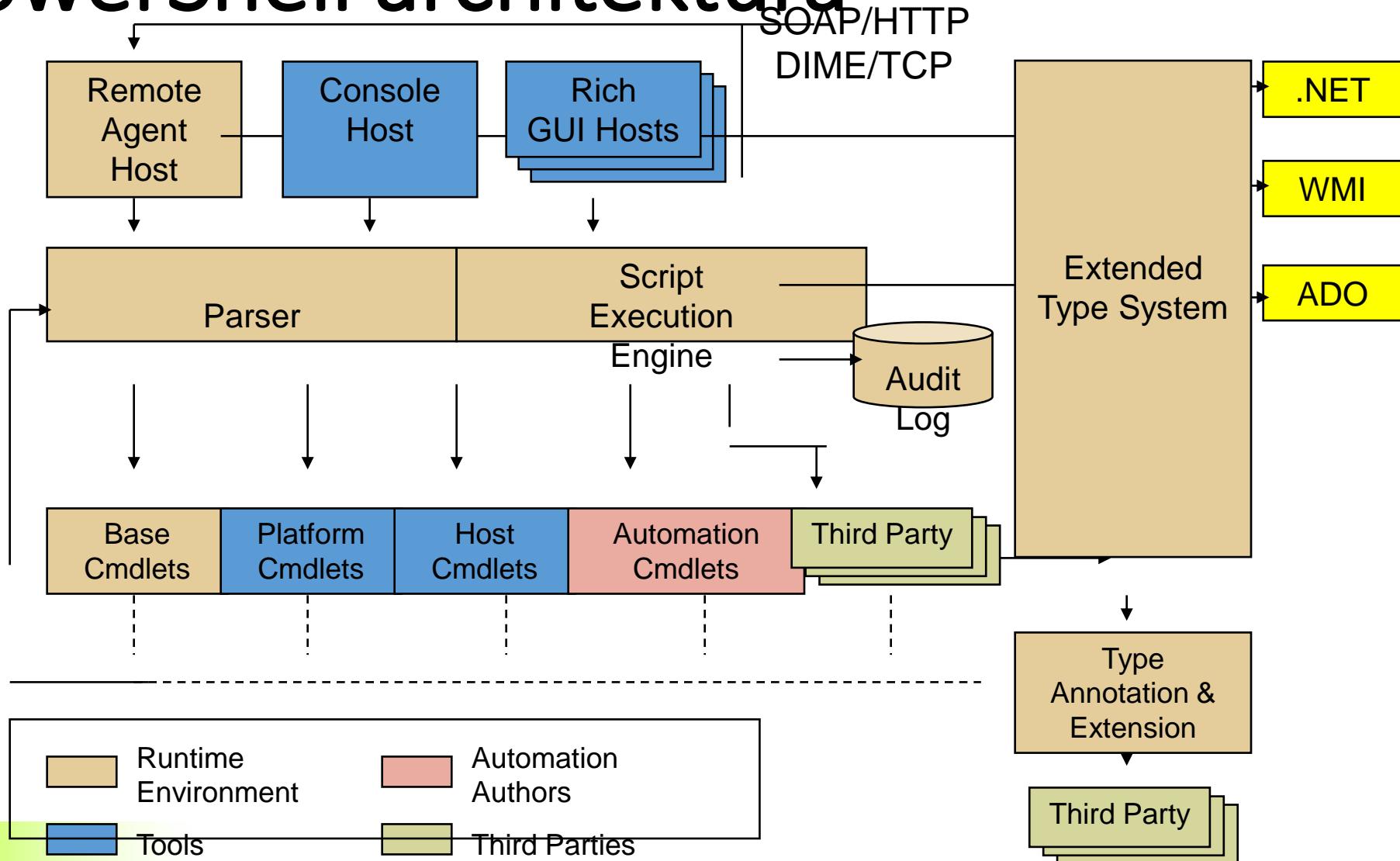
```
[DBG]: PS C:\Users\illes> $max
0

[DBG]: PS C:\Users\illes> $i
10

[DBG]: PS C:\Users\illes>
```

The status bar at the bottom indicates 'Completed' and 'Ln 45 Col 28'. The bottom right corner shows '185%'. The title bar says 'Windows PowerShell ISE (x86)'.

PowerShell architektúra

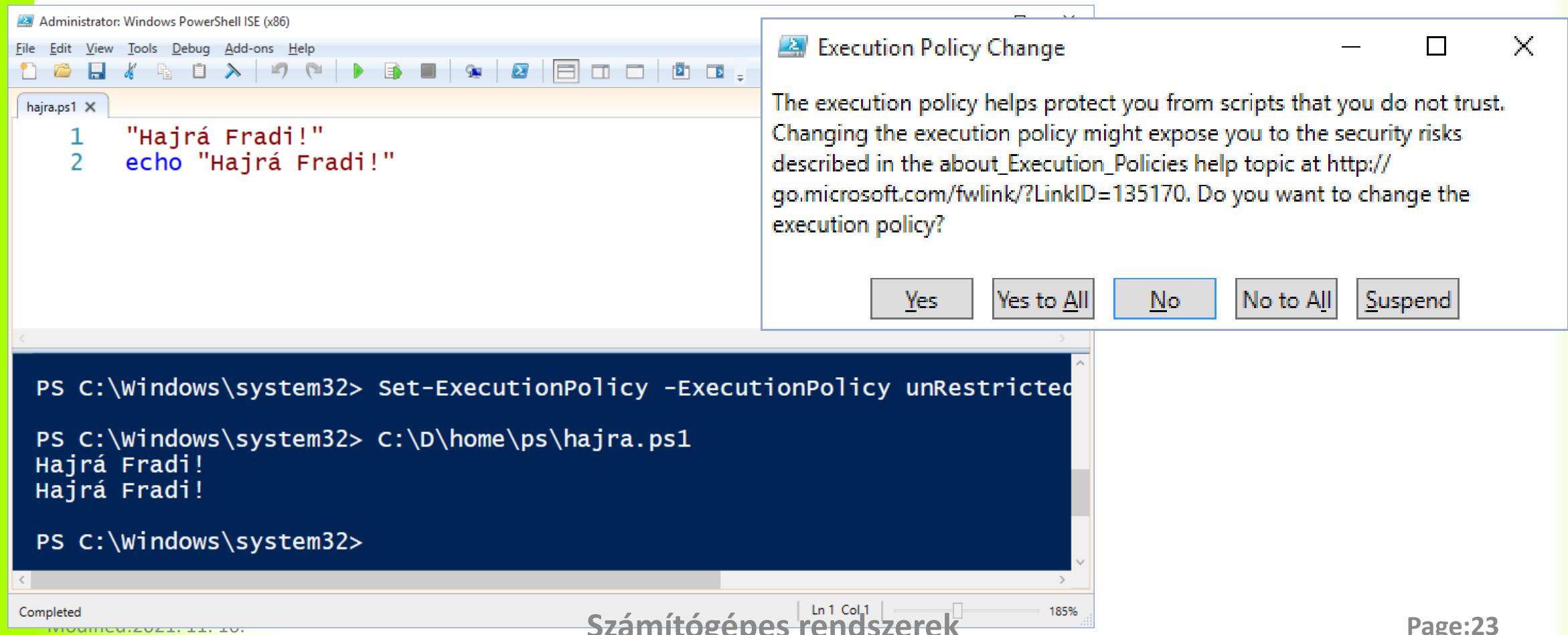


Biztonságos script végrehajtás

- Get-ExecutionPolicy
- Set-ExecutionPolicy –ExecutionPolicy UnRestricted
 - Alapértelmezett: Restricted – nem engedélyezett a futtatás!
 - Lehetséges policy értékek: Allsigned, Remotesigned,Bypass
 - Unrestricted esetén a letöltött, nem aláírt script esetén rákérdez, Bypassnál ezt sem!
 - -scope process vagy currentuser vagy localmachine
 - Remotesigned: Internetről letöltött állományok esetén csak akkor futtatja, ha megbízható partner írta alá.
- Fontos! PowerShell indítása: Run As Administrator!

PS script futtatás

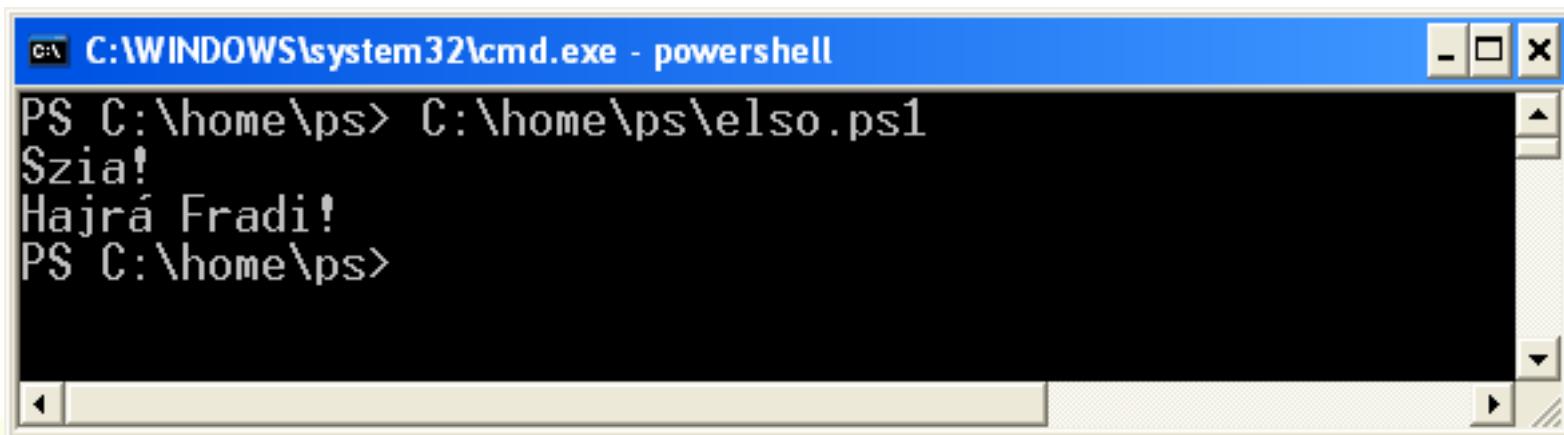
- Fájlnév kiterjesztés: ps1



PS script futtatása

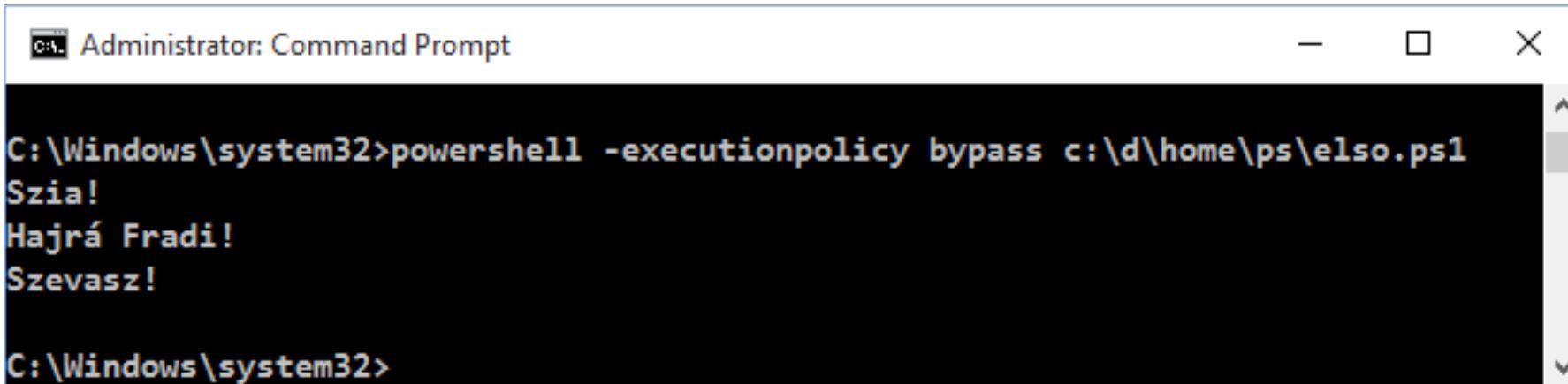
- Parancs futtatás, teljes útvonal beírásával
- Ha helyköz van egy könyvtárban, akkor az & jelet írjuk a parancs elő, és "" között legyen a parancs.
 - &"c:\alma fa\jonatán.ps1"

```
# Megjegyzés  
echo Szia!  
#  
Write-Host "Hajrá Fradi!"
```



PS script futtatás parancssorból

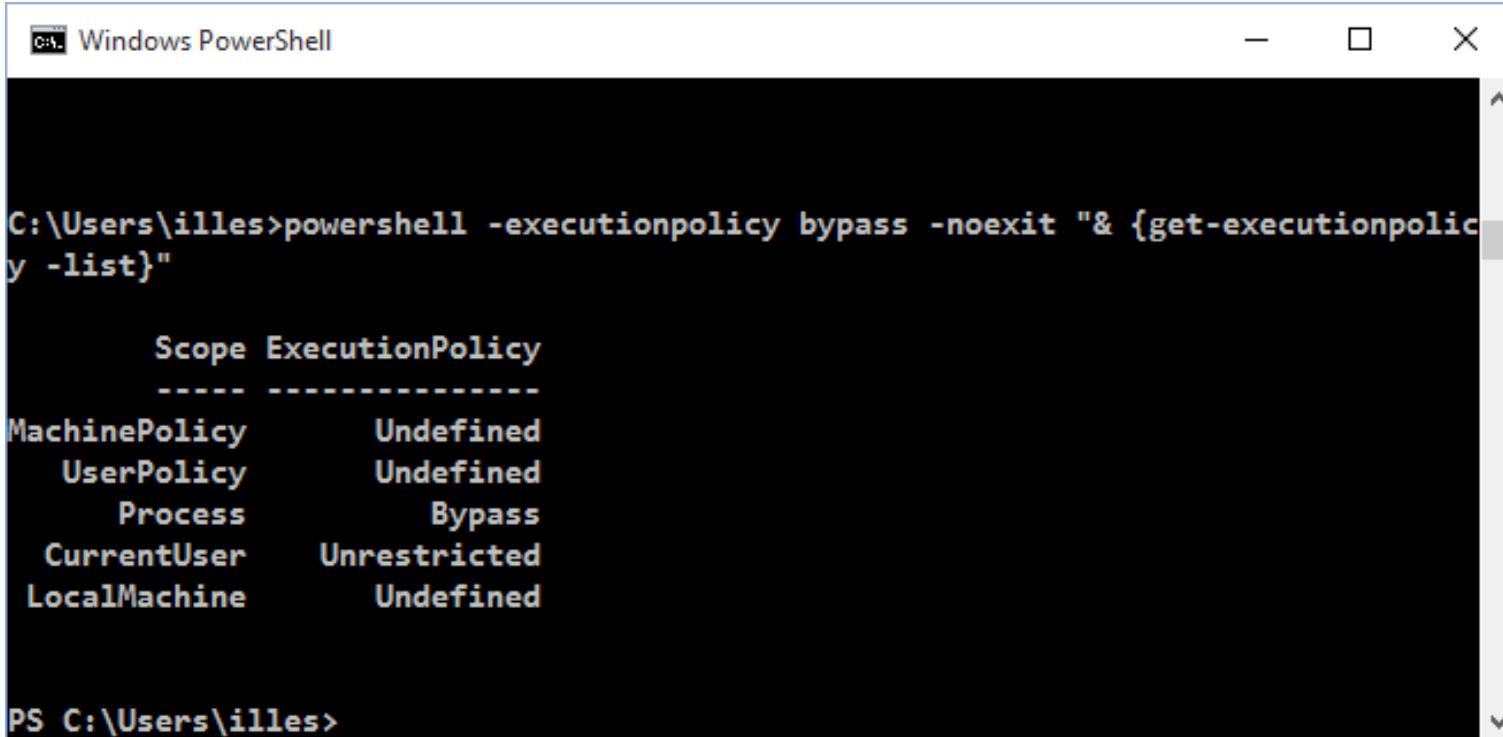
- Meg kell adni a scriptet mint paramétert!
- Meg kell adni az executionpolicy paramétert! (ha nincs beállítva!)



The screenshot shows an 'Administrator: Command Prompt' window. The command entered is `C:\Windows\system32>powershell -executionpolicy bypass c:\d\home\ps\elso.ps1`. The output displayed is:
Szia!
Hajrá Fradi!
Szevasz!
C:\Windows\system32>

PS parancs futtatása

- PS paraméter: "& {parancs}"
- -noexit nem lép ki a PS-ből



A screenshot of a Windows PowerShell window titled "Windows PowerShell". The command entered is:

```
C:\Users\illes>powershell -executionpolicy bypass -noexit "& {get-executionpolicy -list}"
```

The output shows the current execution policies:

Scope	ExecutionPolicy
MachinePolicy	Undefined
UserPolicy	Undefined
Process	Bypass
CurrentUser	Unrestricted
LocalMachine	Undefined

At the bottom, the prompt is PS C:\Users\illes>

PowerShell parancssor

- Ha konstanst írunk, azt az értelmező próbálja egy típushoz illeszteni.
- Ha mászt nem mondunk, az alap művelet: write-host
 - Azaz egy konstans leírása automatikusan annak képernyőre írását jelenti!

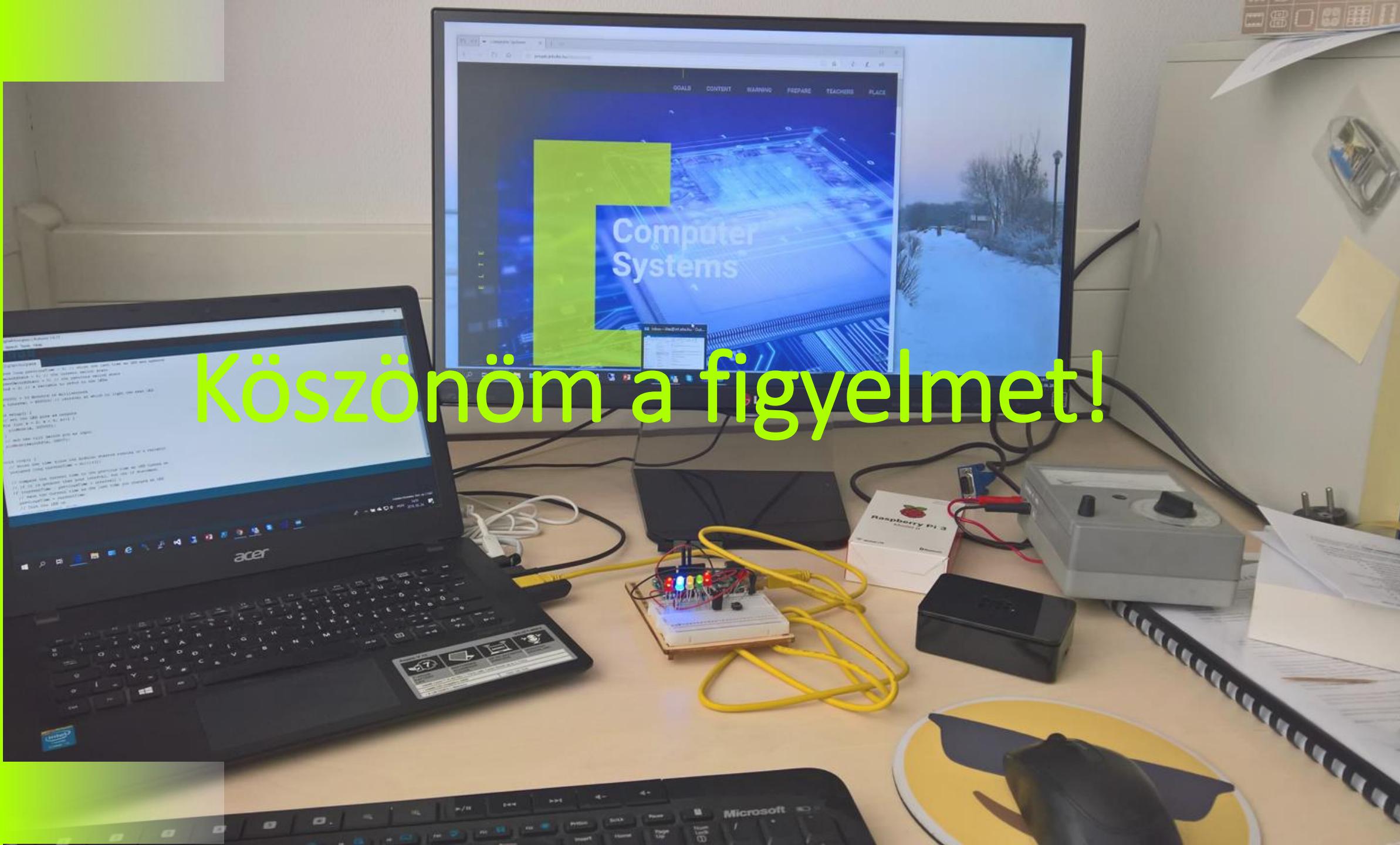
```
PS C:\Users\illes> fradi
The term 'fradi' is not recognized as a cmd in. At line:1
char:5 + fradi <<<
PS C:\Users\illes> "fradi"
fradi
PS C:\Users\illes> 5
5
PS C:\Users\illes> 'F'
F
PS C:\Users\illes> F
The term 'F' is not recognized as a cmdlet, At line:1char:1
+ F <<<
```

PS parancsok formája

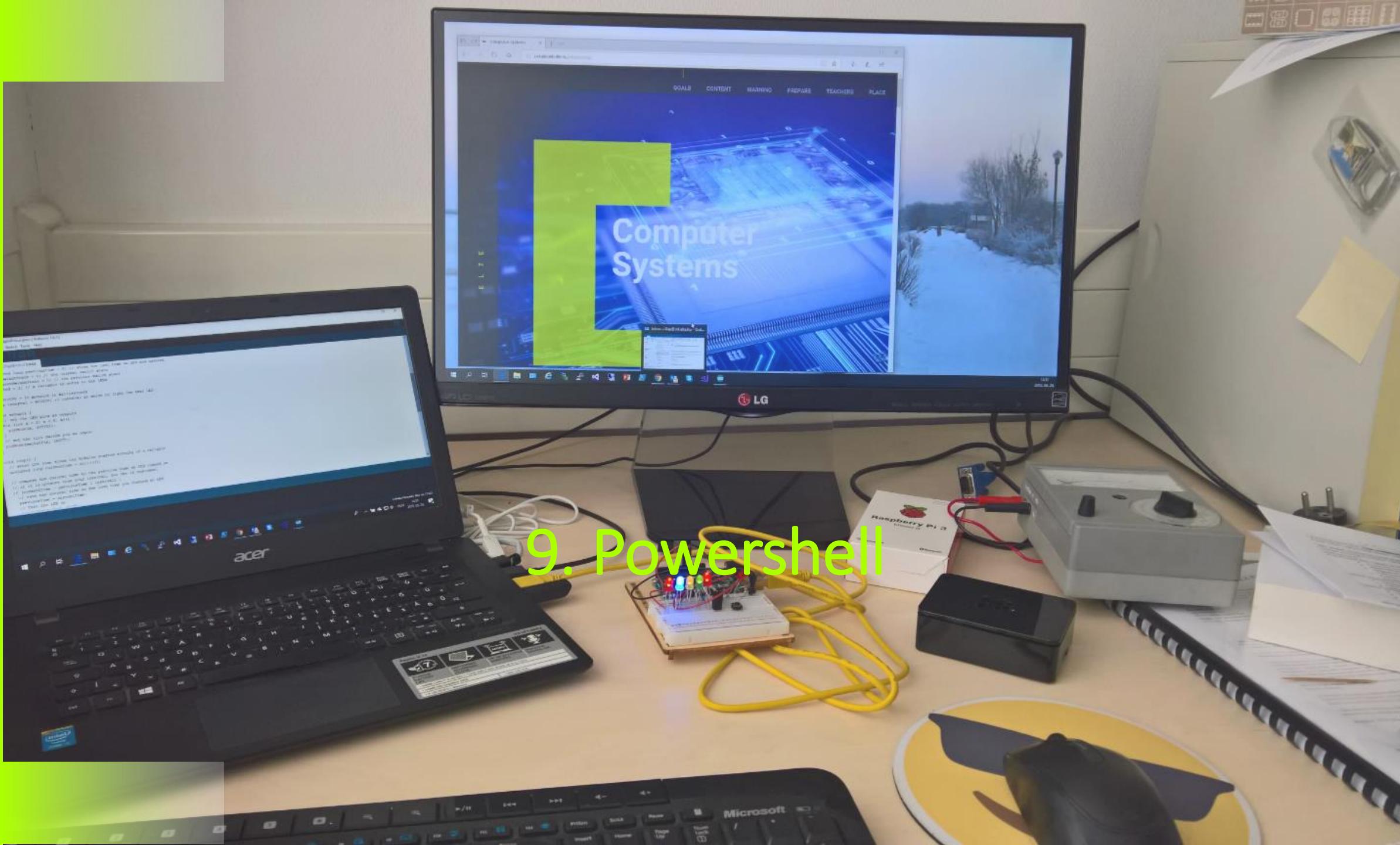
- Két nagy család (kezdetben, ma több😊)
 - Get-parancs család, Set-parancs család
- PS parancs : commandlets- cmdlets
 - Beépített parancsok + kiegészítések.
 - „Hagyományos parancsok” (echo) is használhatók!
- A Tab billentyű kiegészíti a parancsot
- Megjegyzés: #
 - Többsoros megjegyzés: <##>
 - Script elején ha szerepel benne .Description, .Syntax,.Synopsis, .Examples blokk valamelyike, akkor a get-help script ezt veszi alapul!
- Kis-nagybetűre nem érzékeny

PowerShell Help

- A segítség parancsa: Get-Help
- Alább csak a parancsok szintakszisa érhető el!
- Update-Help parancs a lokális gépre installálja a dokumentációt!
 - Hasznosabb az online help!
- <https://docs.microsoft.com/hu-hu/powershell/>
 - PS 5.x
- Magyar nyelvű help (PS 2.0 pdf)



Köszönöm a figyelmet!



Visszatekintés

- Számítógépek, számábrázolás, kódolás,felépítés, fájlrendszerek
- Alapvető parancsok, folyamatok előtérben, háttérben
- I/O átirányítás, szűrők,reguláris kifejezések
- Változó, parancs behelyettesítés,aritmetikai, logikai kifejezések
- Script vezérlési szerkezetek,Sed,AWK
- Batch, WSH
- PS áttekintés

Mi jön ma?

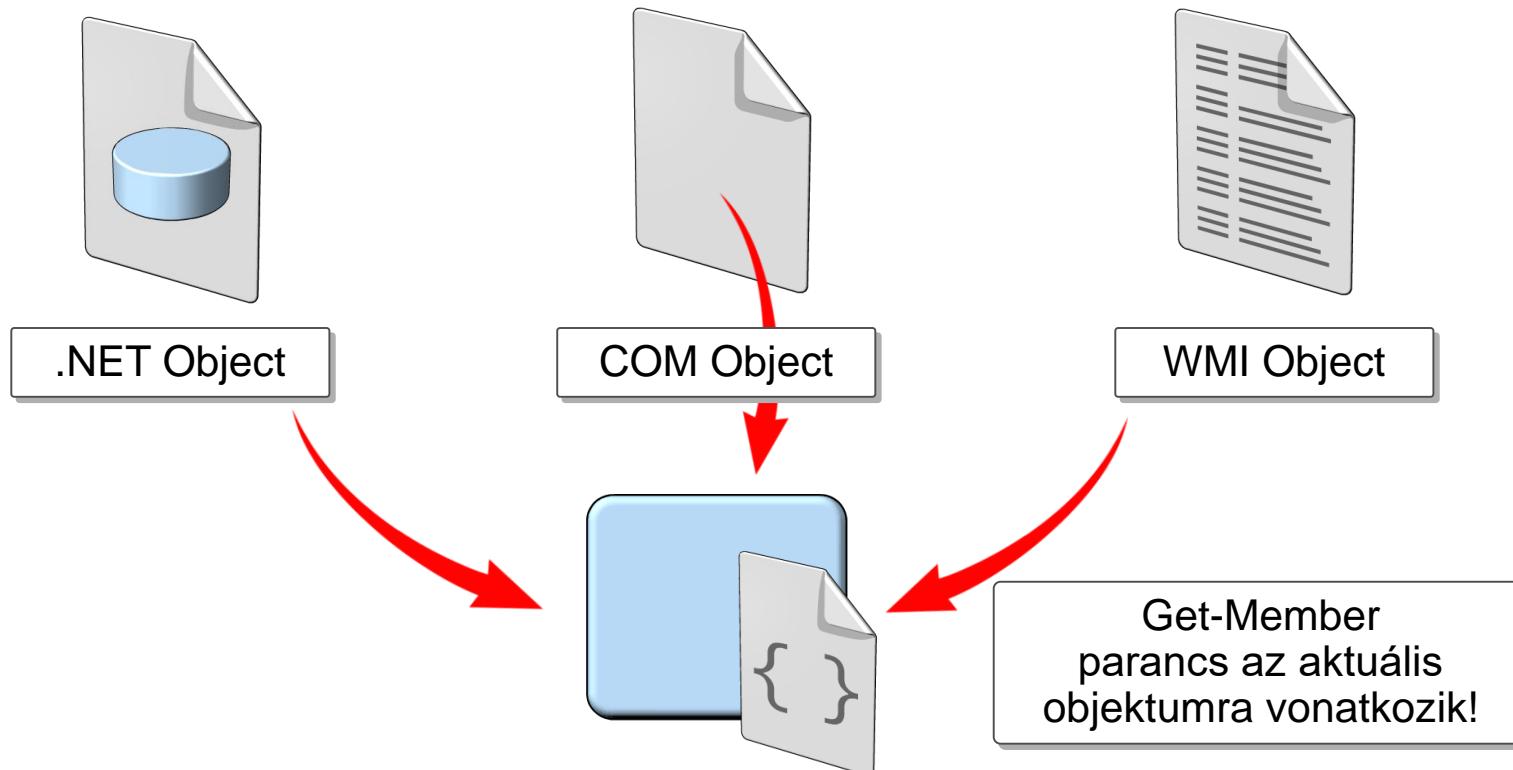
- PowerShell alapok
- PowerShell nyelvi elemek
- ...

A PowerShell objektum orientált

- get-date – eredmény: 20xx. november y...
- „Cső, cső hátán”: get-date | get-member
 - A get-date objektum a get-member bemenetére kerül majd az objektum mezőit kapjuk.
 - -inputobject paraméter sok cmdlet-nél él
 - „fradi” | get-member
- get-date | get-member –membertype method
 - get-date metódusait kapjuk meg, hasonlóan property-t, tulajdonságokat is lekérhetünk.
- (get-date).month, day, ...ticks

NET, COM, WMI Objektumok mint könyvtárak

- PowerShell a .NET-et natív módon használja. A COM, WMI objektumokat is közvetlenül elér.



Alap PowerShell parancsok

- alias , kiírja a definiált rövidítéseket
 - Unix-hoz hasonló parancsok
- gcm – Get-Command, kiírja parancsokat
- echo – Write-Output, képernyőre írás, egyszerű, pipe használat
 - Write-Host – [Console]::WriteLine, előtér, háttérszín állítás
- Get-Help – rövid leírás parancsokról
 - Update-Help – letölți és installálja a helpet!
 - Get-Help –full Write-Host
 - set-alias gh get-help
- ps – Get-Process, futó processzek kiírása
 - Sleep – Start-Sleep, várakozás

Drive vs. Más adatforrás

- dir, ls – Get-ChildItem, könyvtár tartalom
 - Get-ChildItem c:\users\teszt*.* -include *.c,*.cpp # c, cpp kiterjesztés csak
- Get-PsDrive – Powershell adatforrások
- Cd könyvtár váltás – Set-Location parancs
 - Cd c:
 - Cd hklm:
 - Cd alias:
 - Dir- kilistáztatja az aktuális drive, adatforrás tartalmát
- Get-Location – pwd alias parancsa

Fontosabb fájl kezelő parancsok

- New-Item – fájl vagy könyvtár létrehozása
 - New-Item –itemtype directory almadir #mkdir dos parancs megfelelője!
- Copy-Item forrás cél [–recurse] # másolás (Alias: cp)
- Remove-Item – fájl, könyvtár törlése (Alias: rm, rmdir, del,...)
- Move-Item – fájl, könyvtár mozgatás (Alias: mv)
- Rename-Item – átnevezés (Alias: ren)
- Get-Item – fájl, könyvtár, reg.kulcs visszaadás
 - Get-item \$(c:\users).LastWriteTime, Get-item hkcu:\software |get-member
- Test-Path fájl vagy könyvtár vagy reg.kulcs, #létezik-e?

PowerShell parancsok felépítése, paraméterek

- PowerShell parancs felépítés: Ige-főnév
 - PL: Get-Command
- Paraméterek megadása jellemzően: -név érték
 - Érték lehet: szám, szöveg, dátum
 - PI: Get-Command –Verb write
- History – F7 előző parancsok
 - felfelenyíl, előző parancs
- Profile:Dokumentumok\WindowsPowerShell könyvtárban:
 - Microsoft.PowerShell_profile.ps1
 - profile.ps1 - Ezt csak az ISE hajtja végre!

PowerShell változók

- \$név=érték, kötelező a \$ jel a definiáláskor is
 - Pl: \$f=„fradi”; echo \$f
- egy sorba több parancs írható, ; az elválasztó
- Támogatott fontosabb alap típusok:

Adattípus	Értelmezése	Példa
[int]	Egész szám(32bit)	-273, -1, 0, 10, 42
[byte]	8-bit, bájt	0, 1, ..., 254, 255
[boolean]	Logikai	\$false, \$true
[char]	Karakter(16 bit uni.)	a, b, c, 1, 2, 3, !, #
[string]	Szöveg	“FTC”
[single]	32 bites valós	2.3e-1, 3.1415,...
[datetime]	Idő	April 1, 2008

PowerShell változók használata

- Magunk is megadhatjuk (típuskényszerítés):
 - [int] \$d=6.2e-4; echo \$d # 0, \$d egész lesz
 - \$s= [string] 65; echo \$s # 65 szövegként
 - \$s1=[string] [char] 65; echo \$s1 # A
 - \$i=[int] "65"; echo \$i # 65 szövegként
- Ha nem jelölünk semmit, az értelmező eldönti a típusát
 - \$d=6.2e-4; echo \$d # 0,00062, valós lesz

PowerShell változók definiálása parancs segítségével

- Set-Variable –Name alma –value „jonatán” –option constant
 - Konstans definiálás
 - Egy leírás adható a –description paraméterrel
 - Get-Variable alma
- Clear-Variable alma # alma létezik, csak tartalma nincs.
- Remove-Variable alma # alma nem létezik

Változók láthatósága I.

- Egy környezetben definiált változókat, a környezetében használhatjuk, az ebből származó függvény, script látja!
 - Ha azonos nevűt definiálunk egy scriptben, függvényben, akkor alapból a lokálisat látjuk
- Ezen lehet segíteni a get-variable –scope paraméterrel.
 - Get-variable alma –scope 0 # aktuális környezet
 - Get-variable alma –scope 1 # szülő környezet
 - Get-variable alma –scope 2 # nagyszülő környezet
 - Stb.

Változók láthatósága II.

- Általános változó definíciós forma:
 - \${[scope:]név vagy \${név}}
 - Ha a scope elmarad, aktuális helyen (script, függvény) használható lesz a változó.
 - A scope lehet, global, local, private, script.
 - \$global:x=5 # globális lesz az x változó, mindenhol látható
 - \$script:y=6 # a teljes scriptben használható lesz
 - \$local:z=„MTK” # lokálisan és a gyerek blokkokban látható
 - dir \$env:ProgramFiles # az env drive eleme
 - A környezeti változók érhetők el ezen keresztül!
 - \$env:Path += „;d:\tmp”

Aritmetikai műveletek PowerShell-ben

- $+,-,*,/,%$ (maradék)- alapműveletek
 - Nem kell külön parancsot kiadni, mint pl. az expr!
 - `$a= 32*3; echo $a # 96`
 - `$a=„alma”; $f=„fa”; $c=$a + $f; echo $c #almafa`
 - `$a= „125” + „2”; echo $a # 1252!`
 - `$a= 12 + „4”; echo $a # 16`
 - automatikusan konvertálja a „4”-et
- Értékkadások: `=, +=, -=, *=, /=, %=`
- Post növelés, csökkenés: `$a++, $b--`
- Bitműveletek: `-band, -bor, -bxor, -bnot, -shl, -shr`

Még több művelet

- A PowerShell mögött a .NET FrameWork áll.
 - Az összes típus, double, decimal stb. elérhető
 - Nem csak alaptípusok
 - Példa: [System.IO.DirectoryInfo]\$home=Get-Item D:\home
- Bármely statikus tulajdonság, metódus elérés operátora a ::
 - [DateTime]::Now # aktuális dátum
- Teljes Math osztály is rendelkezésre áll
 - [math]::pi
 - [math]::sin(2), Stb.
- Konverzió: [system.convert]::toint32(„32”)
- Stb. ,Net FrameWork teljes könyvtár használat

PowerShell változók összefoglalás

- \$csapat=„Fradi”
- Automatikus típusmegadás, de felülbírálható
 - [int]\$a=„alma” # ez hibát ad persze
- minden alapművelet rendelkezésre áll! + .NET
- Érdekesség: \$b=\$csapat*5 #ez ok, „Fradi ötször”
 - \$c=5*\$csapat #hiba!!, „Fradi” nem lesz egész!!!
- Adatmegadás parancssal
 - Set-Variable –name a –value „körte” –option constant
- Szöveges parancs végrehajtás operátora: &
 - \$dir=„dir”; &\$dir

Változó behelyettesítés

- `$a=„alma”`
- „\$afa” # eredmény üres
- „\${a}fa” # eredmény: almafa
- „piros\$a” #pirosalma
- A \$ karakter semlegesítése: `
 - „`\$a változó értéke: \$a”
 - Reguláris kifejezésben a \ karakter használandó semlegesítésre!
- Parancs behelyettesítés forma külön nincs!
 - `$könyvtárlista=dir` # Nincs szükség a `dir` formára!

Szövegek, behelyettesítés

- A „” között lévő szövegben lévő változó behelyettesítésre kerül!
- AZ ,’ közötti nem: ,Ez nincs behelyettesítve: \$a’
- Hasonlóan Unix-hoz, lehet: "echo '\$i vége'"
 - Itt (is) \$i behelyettesítésre kerül!!!!
- Powershellben nincs input átirányítás (<,<<)
- Van helyette többsoros szöveg: @” ...több sor is lehet... ”@
 - Közte a változók behelyettesítésre kerülnek!
 - @’ Több sor is lehet.... ’@ # nincs változó behelyettesítés

PowerShell tömbök I.

- Változók gyakori elnevezése: skalár, egy adatot tartalmaz, pl: \$adat=„alma”
- Több adatot tartalmazó „skalár”: tömb
- Definiálás: \$tömb="alma","körte","barack"
 - Teljes változat: \$tömb=@("alma","körte","barack")
 - Elemek elérése a 0 indextől!
 - echo \$tömb[1] # körte
 - echo \$tömb[1..2] # körte barack
 - Egy elem nem csak egyszerű (skalár) lehet, hanem tömb is: \$tömb[2]=@(2,3,4); echo \$tömb[2][1] # 3

PowerShell tömbök II.

- A tömb is valójában objektum. A tömbök hossza a Length tulajdonsággal érhető el.
 - echo \$tömb.Length
- Új elem hozzáadása: \$tömb1+=6;
- Összes tömbelem kiírása: \$tömb (azonos az echo \$tömb utasítással)
- Tömböket összefűzhetünk: + jel segítségével
 - \$tömb1=2,3,4,5
 - \$tömb+=\$tömb1
 - echo \$tömb[3] # 2

PowerShell tömb műveletek

- -contains : Tartalmazás (-notcontains)
 - 1,2,3,4 –contains 3 # True
- -eq, -ne Eredmény az összes elem ami egyenlő, (nem egyenlő) adott értékkel
 - 1,2,3,4 –ne 3 # 1,2,4
- -lt, -gt Eredmény az összes elem ami kisebb, (nagyobb) adott értéknél
 - 1,2,3,4 –lt 3 # 1,2
- -le, -ge Kisebb, nagyobb vagy egyenlő
- -join, -split, -csplit (case split, kis-nagybetű)
- Stb.

PowerShell asszociatív tömbök

- \$atömb=@{„kulcs”=„érték”; ...}
 - \$at=@{a=4;b=5} # Elemek között a ; !!!!!!
- Elem elérés: \$at[a] vagy \$at.a
- Elem értékkadás: \$at[a]=10
- Új elem hozzáadás: \$at+=@{c=11}
- Asszociatív tömb kiírása: \$at

```
PS C:\home\ps> $at
Name          Value
----          -----
a              10
b              6
c              11
```

.NET FrameWork tömbök

- System.Collections a különböző adatszerkezetek névtere:
 - \$t=new-object system.collections.arraylist
 - \$t1= [system.collections.arraylist] (2,3,4)
 - \$t1.add(10)
 - \$t1.contains(3) # igaz
 - \$t1.insert(2,20) # 3 után lesz a 20!
 - \$t1.sort()
 - Stb.

Elágazás PowerShellben

- Összehasonlító operátorok, mint a tömböknél.
 - -eq, -ne, -gt, -lt, -le, -ge
 - -not, -and, -or, -xor logikai tagadás, és, vagy
 - Szövegnél: -ceq, Kis, nagybetű különböző, -ieq nem különböző,
 - -like *, ?, [ab.] karakterek, -match reg. kif. használat
- If utasítás:
 - if (kif) {utasítás} [elseif (kif1)] else {utasítás}

```
$a=3
if ($a -gt 2)
{ Write-Host "A" $a "nagyobb mint 2." }
else
{ Write-Host "Nem nagyobb mint 2." }
```

Többirányú elágazás

- switch utasítás - .net nyelkekhez hasonló
 - alágakba nem kell break

```
# switch példa
#
# Beolvasás utasítás: read-host
$a=Read-Host -prompt "Írja be a kedvenc gyümölcsét "
switch ($a)
{
    "alma"      { "a értéke: "+$a } # write-output röv.
    "barack"    { "a értéke: "+$a }
    "szőlő"     { "a értéke: "+$a }
    "szilva"    { "a értéke: "+$a }
    "körte"     { "a értéke: "+$a }
    default     { "a ismeretlen számomra: "+$a}
}
```

PowerShell Ciklusok I

- for – gyakorlatilag mint C,...stb –ben
 - Mindig kell a ciklusmag köré: {}
 - Pl: for(\$i=0;\$i -lt 5;\$i++) {echo \$i}
- foreach(\$i in tömb) {ciklusmag}

```
#  
$t=2,3,4,5  
foreach($i in $t)  
{  
    write-host $i  
}
```

PowerShell Ciklusok II.

- **foreach – Foreach-Object, szűrő**

- Egy sorba több parancs: ;
- Több sorba egy parancs: `
- AWK-hoz hasonló begin, end blokk

```
# egy sor folytatása új sorban: ` karakter, fontos!!!
#
get-process|foreach-object ` 
    -begin { Write-Host "Elkezdtem a Get-Process feldolgozást!" } ` 
    -process { write-host $_.name -foreground green } ` 
    -end { Write-Host "Befejeztem a Get-Process feldolgozást!" }
```

PowerShell Ciklusok III.

- While ciklus: mint C-ben, do...while kif.– amíg igaz
- Until ciklus: do ...until kif.; Amíg a kifejezés hamis!

```
$a=5  
while ($a -gt 0)  
{  
    write-host $a  
    $a--  
}
```

```
do  
{  
    "Egyszer belépek a ciklusba biztosan!"  
    write-host $a  
    $a--  
} while ($a -gt 0)
```

```
$a=0  
do {  
    "Egyszer belépek a ciklusba biztosan!"  
    write-host $a  
    $a++  
} until ($a -gt 3)
```

Script szerkezet – Foreach minta

- Nem csak a foreach blokk állhat 3 blokkból!
- Egy script is három blokkot tartalmazhat:
 - Begin { utasítások} # A script elején egyszer végrehajtódik
 - Process { utasítások } # A script törzse, minden „csöves” objektumra végrehajtódik!
 - End { utasítások } # A script végén egyszer végrehajtódik
- A foreach utasítástól függetlenül a „csöves” kapcsolaton keresztül adatokat feldolgozó scriptek gyakori formája!
 - Példa: csoparam.ps1

Példa: csoparam.ps1

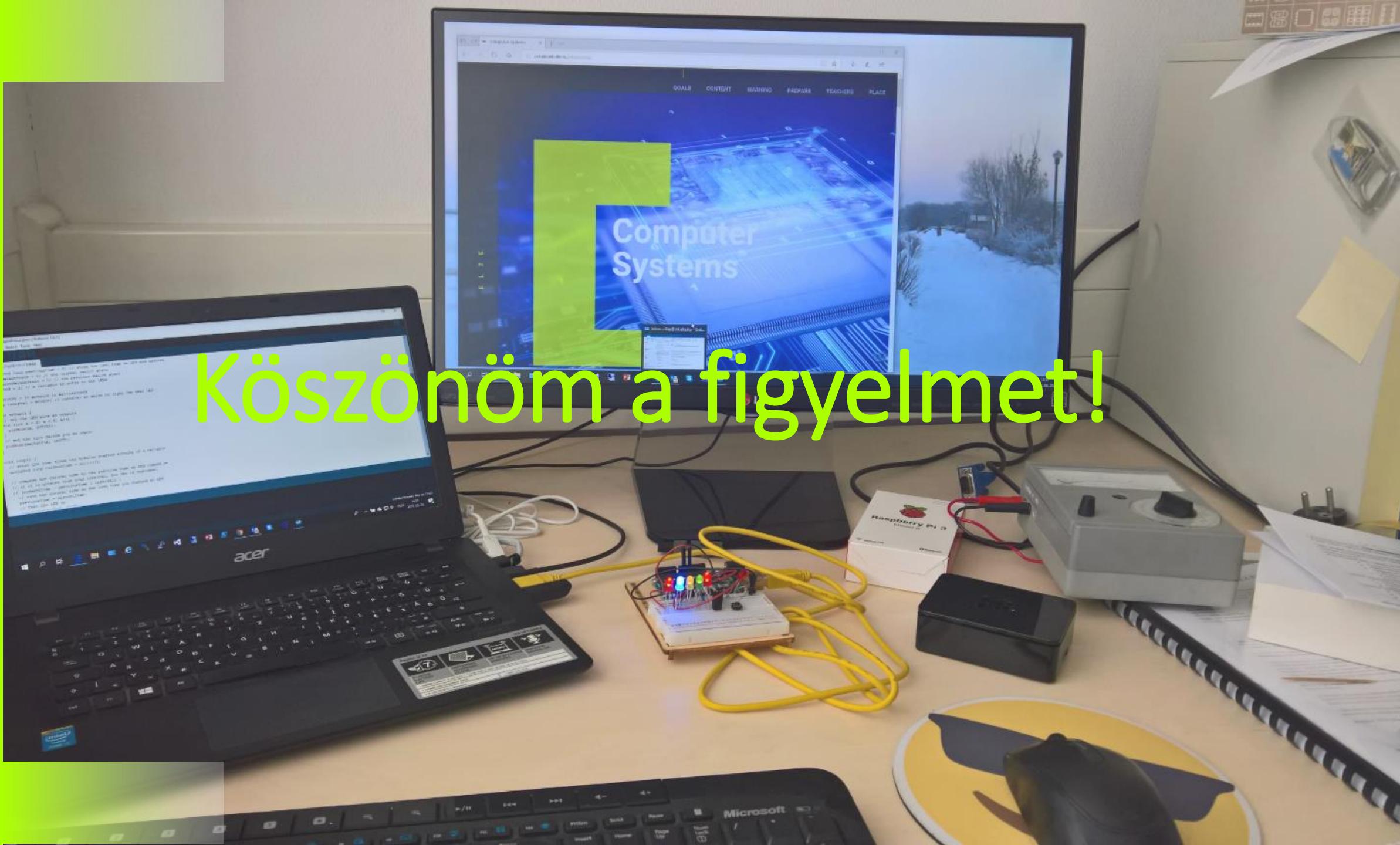
```
BEGIN {
    foreach( $i in $Args )
        { Write-Host "Begin blokk a paraméterek $i" -F red }
    Write-Host "Begin blokkban Pipeline: $_" -Fore red
}

PROCESS {
    foreach( $i in $Args )
        { Write-Host "Process blokk a paraméterek: $i" -F white }
    Write-Host "Process blokkban Pipeline: $_" -F white
    $_.GetType().Name # Objektum típusának neve
}

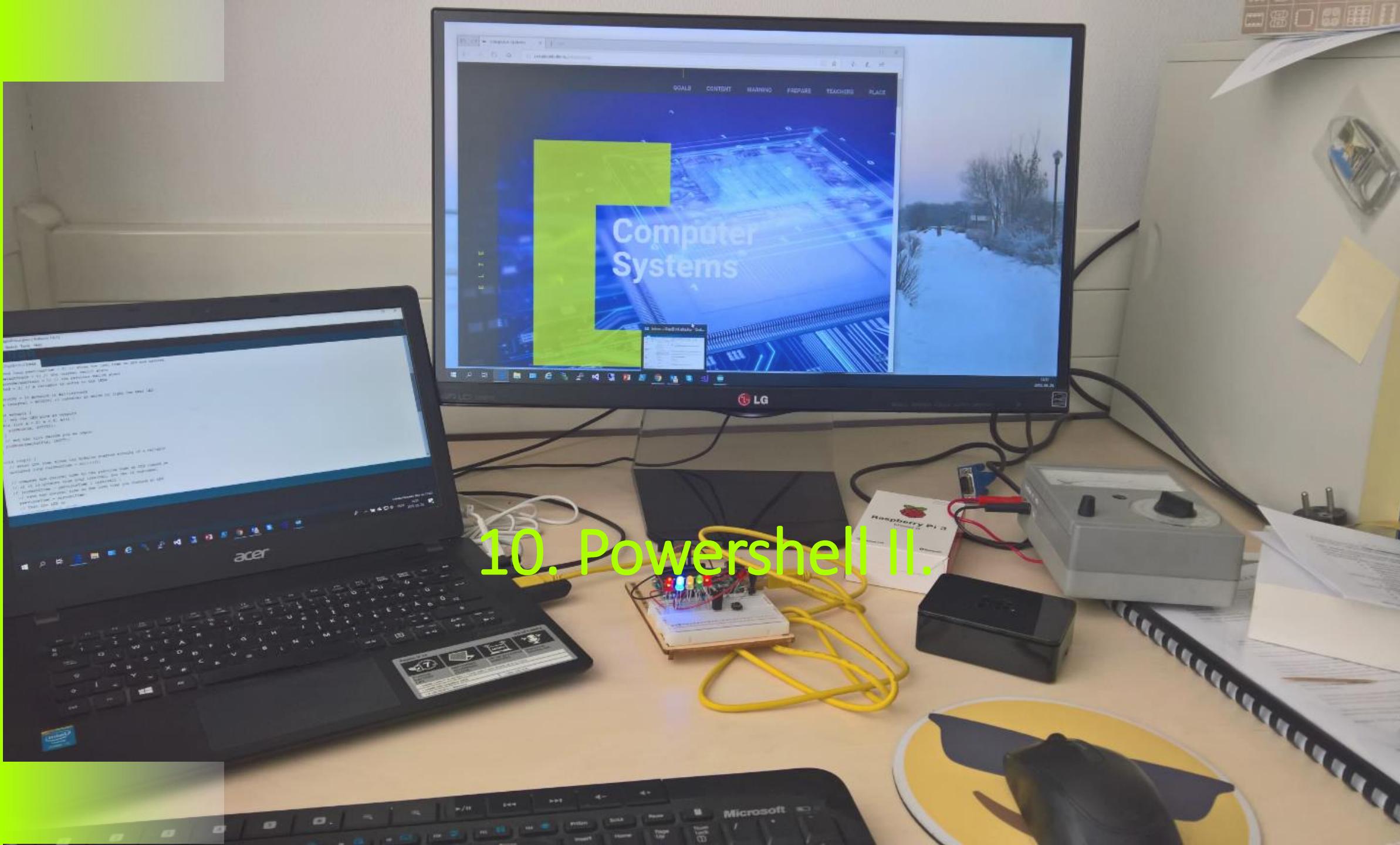
END {
    foreach( $i in $Args )
        { Write-Host "End blokk a paraméterek: $i" -F green }
    Write-Host "End blokkban Pipeline: $_" -F green
}
```

```
PS C:\d\home\ps> 34,35 | .\csoparam.ps1 3 4
Begin blokkban a paraméterek 3
Begin blokkban a paraméterek 4
Begin blokkban a paraméterek 5
Begin blokkban Pipeline:
Process blokkban a paraméterek: 3
Process blokkban a paraméterek: 4
Process blokkban a paraméterek: 5
Process blokkban Pipeline: 34
Int32
Process blokkban a paraméterek: 3
Process blokkban a paraméterek: 4
Process blokkban a paraméterek: 5
Process blokkban Pipeline: 35
Int32
End blokkban a paraméterek: 3
End blokkban a paraméterek: 4
End blokkban a paraméterek: 5
End blokkban Pipeline: 35

PS C:\d\home\ps>
```



Köszönöm a figyelmet!



10. Powershell II.

Visszatekintés

- Számítógépek, számábrázolás, kódolás, fájlrendszerek
- Alapvető parancsok, folyamatok, reguláris kifejezések
- Változó, parancs behelyettesítés, aritmetikai, logikai kifejezések
- Script vezérlési szerkezetek, Sed, AWK
- Batch, WSH
- PS áttekintés, Powershell változók, műveletek
- Alapvető Powershell parancsok, vezérlési szerkezetek

Mi jön ma?

- PowerShell függvények
- PowerShell könyvtári elemek
- minden ami fontos.....

Függvények PowerShellben

- `function név(par) { fvtörzs }`
 - Bárhol elhelyezkedhet, de csak a definíció után használható!
 - eredmény: `return` utasítás
 - `$lastexitcode` változó (utoljára végrehajtott külső program visszatérési értéke)
 - Hívás: `név` 5 vagy `név(5)`
 - Több paraméter:
 - `név1 $x $y`

```
function nfaktor($n)
{
    $f=1
    for($i=2;$i -le $n;$i++) {$f*=$i}
    return $f
}
echo "N faktoriális"
nfaktor(5) vagy nfaktor 5
```

Klasszikus ill. paraméter blokk

Klasszikus
paraméter megadás:

MATIKA

PARAMéter blokk
megadás

-
ELTE

```
function global:Where-BigProcess(  
    [int]    $meret = 200MB,  
    [String] $property = "VirtualMemorySize",  
    [String] $formatString = "Total {2} = {1}"  
) { body }
```

```
function global:Where-BigProcess {  
    PARAM(  
        [int]    $meret = 200MB,  
        [String] $property = "VirtualMemorySize",  
        [String] $formatString = "Total {2} = {1}"  
    )  
    body  
}
```

Függvény paraméterek

- Megadásuk jellemzően a klasszikus minta szerint:
 - function alma(\$név,\$méret) { ...} vagy param blokk
- Alapértelmezett adatok jelzése:
 - \$név="zoli", mint pl. C#-ban
 - Kötelezően adandó paraméter:
 - function alma([parameter(Mandatory=\$true)] \$név) { ...}
 - Korábban: alma(\$név=\$(throw "Kerek nevet!!")) { ...}
- Változószámú paraméteres függvény
 - Mint .NET-ben, \$args tömb

Függvény paraméterek megadása

- Function alma(\$név,\$ár,\$szín) {...}
- Pozíció szerinti megadás:
 - Hívás: alma „jonatán” 150 „piros”;
- Név szerinti megadás:
 - Hívás: alma –név golden –szín sárga
 - Az árat nem adtuk meg, üres lesz!
 - Rövid forma: alma –n golden –s sárga
- Keverhetjük a két megadási módot!
 - Pl: alma –s zöld zöld 250
 - Lehetőleg ne használjuk ezt a kevert módot!

Függvény, változó szint módosítás (dot sourcing)

- Lehet függvényen belül is függvényt definiálni. Belső függvény nem hívható közvetlenül!
 - Pontos indítás: . Fv
 - Azt eredményezi, hogy ezentúl az Fv-n belüli függvények is láthatók, használhatók közvetlenül.
- Függvény lokális változó nem látható kívül.
 - Pontos indítás: . Fv
 - Lokális változó kívül is látszik!
- Óvatosan a használattal!!

Függvény közvetlen használat

- Ez példa az előző „szint” módosító . használatra!
 - „Dot-Sourcing” operátor
 - Shell script használat esetén is ugyanez!

The screenshot shows the Windows PowerShell ISE interface. The script file 'negy.ps1' contains the following code:

```
Windows PowerShell ISE
File Edit View Debug Help
negy.ps1* X
1 param ($szám=$(throw "Adj meg egy számot!"))
2 # négyzet fv.
3 function negyzet($x)
4 {
5     return $x*$x
6 }
7 $n=negyzet($szám)
8 "A paraméterként megadott $szám négyzete: {0}" -f $n
9
```

The output pane shows the execution of the script:

```
PS D:\home\ps> ./negy.ps1 6
A paraméterként megadott 6 négyzete: 36

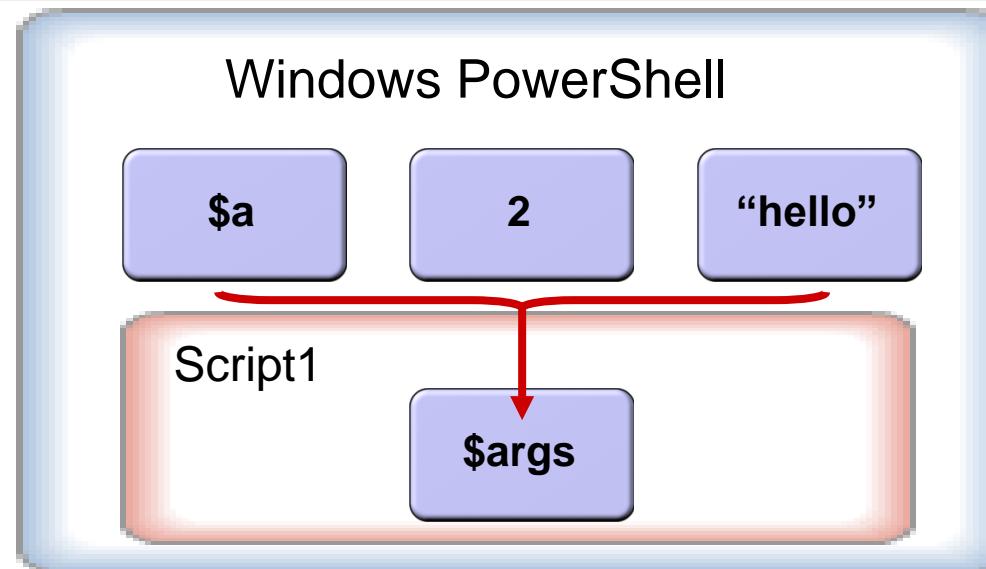
PS D:\home\ps> negyzet(7)
49
```

The bottom status bar indicates 'Completed'.

Script paraméterek mint tömb

```
# script file argtest.ps1
foreach( $i in $args ){"Paraméterek {0:D};" -f $i }
```

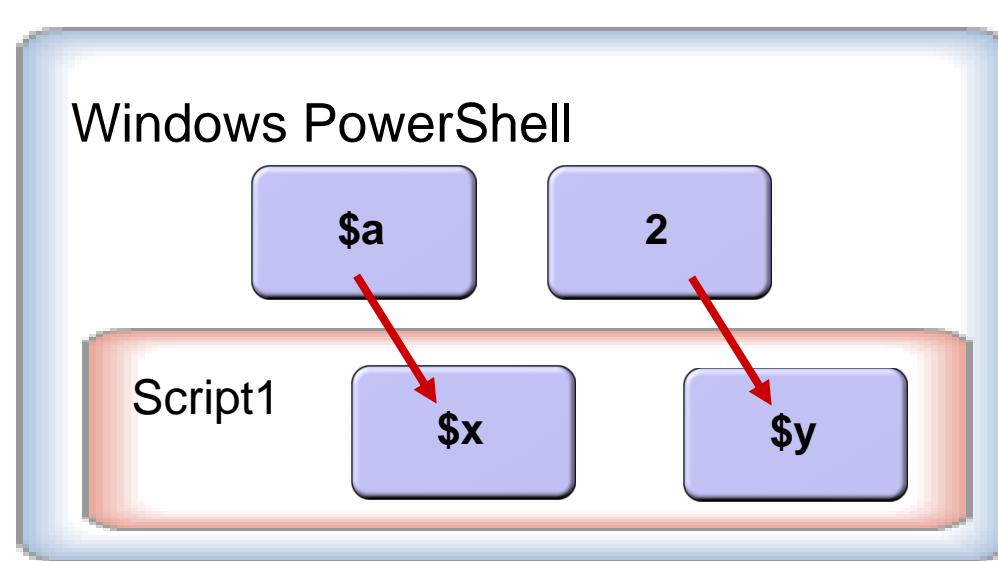
```
PS> $a = 10
PS> .\argtest.ps1 $a 2 "hello"
```



Nevesített script paraméterek

```
# nevesített paraméterek  
param( $x, $y )  
“A `\$x={0}” -f $x  
“A `\$y={0}” -f $y
```

```
PS> $a = 10  
PS C:\> .\paramtest.ps1 $a 2
```



Nevesített és normál paraméterek közös használata

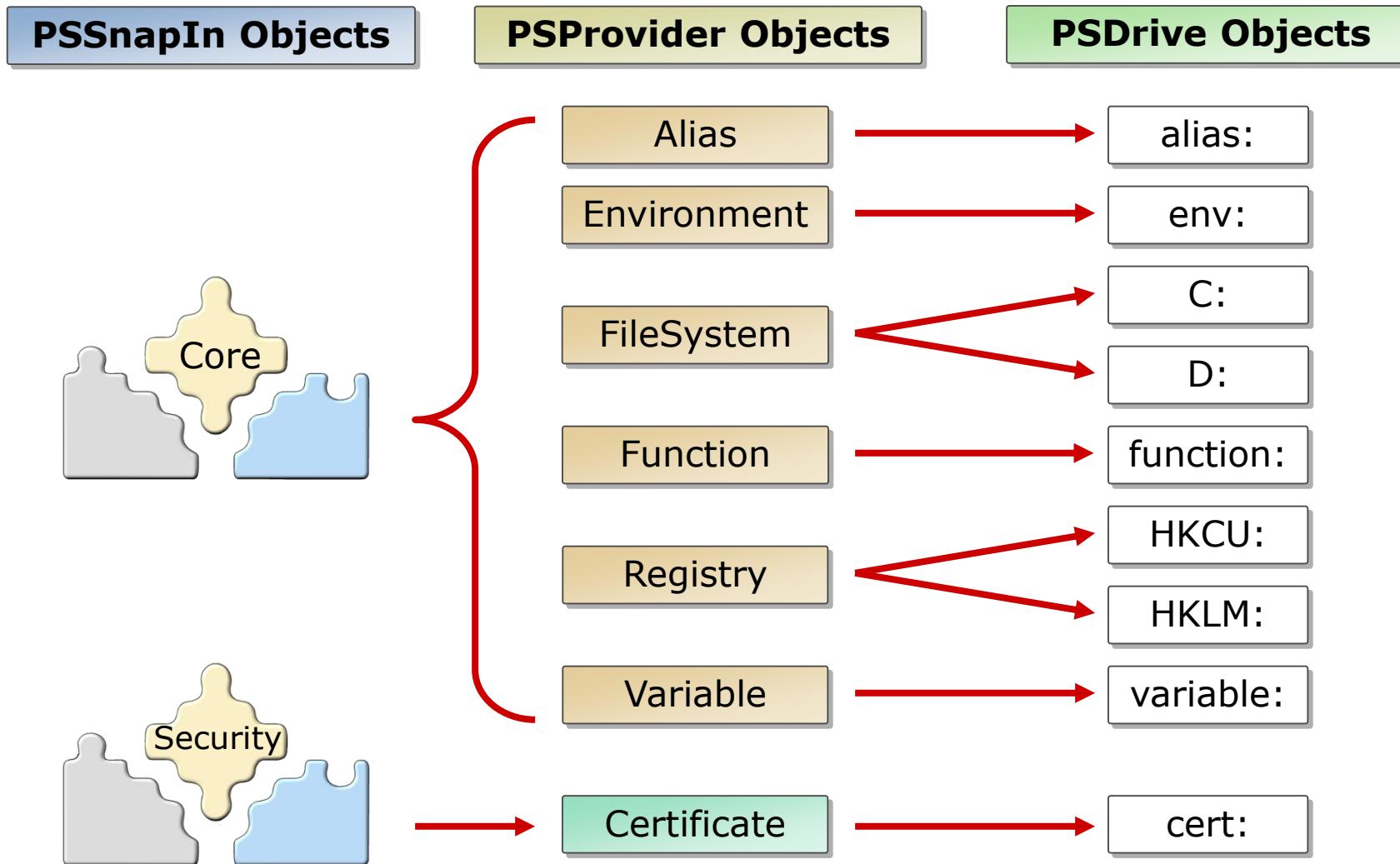
- Lehet keverni a nevesített és normál paraméterekeket.

```
#  
param($x,$y)  
write-output $args.length  
# write-output $args.count  
# ez ugyanaz mint az előző  
write-output $x  
write-output $y  
write-output "A 2. paramétertől kezdve:"  
foreach( $i in $args )  
{"A script paraméterek sorban {0:D};"-f $i }
```

PowerShell (fontosabb) belső változói

- `$_` - aktuális csővezeték objektum, (foreach)
- `$?` - előző parancs eredmény státusza, logikai
- `$home` - felhasználó home könyvtára
- `$$` - előző parancssor utolsó szava
- `$^` - előző parancssor első szava
- `$host` - aktuális kiszolgáló (nem egy név!!)
- `$myinvocation` – aktuális futtatási információk
- `$pshome` - PS install könyvtár
- `$profile` – felhasználó profile fájl neve

PS forrás-Drive



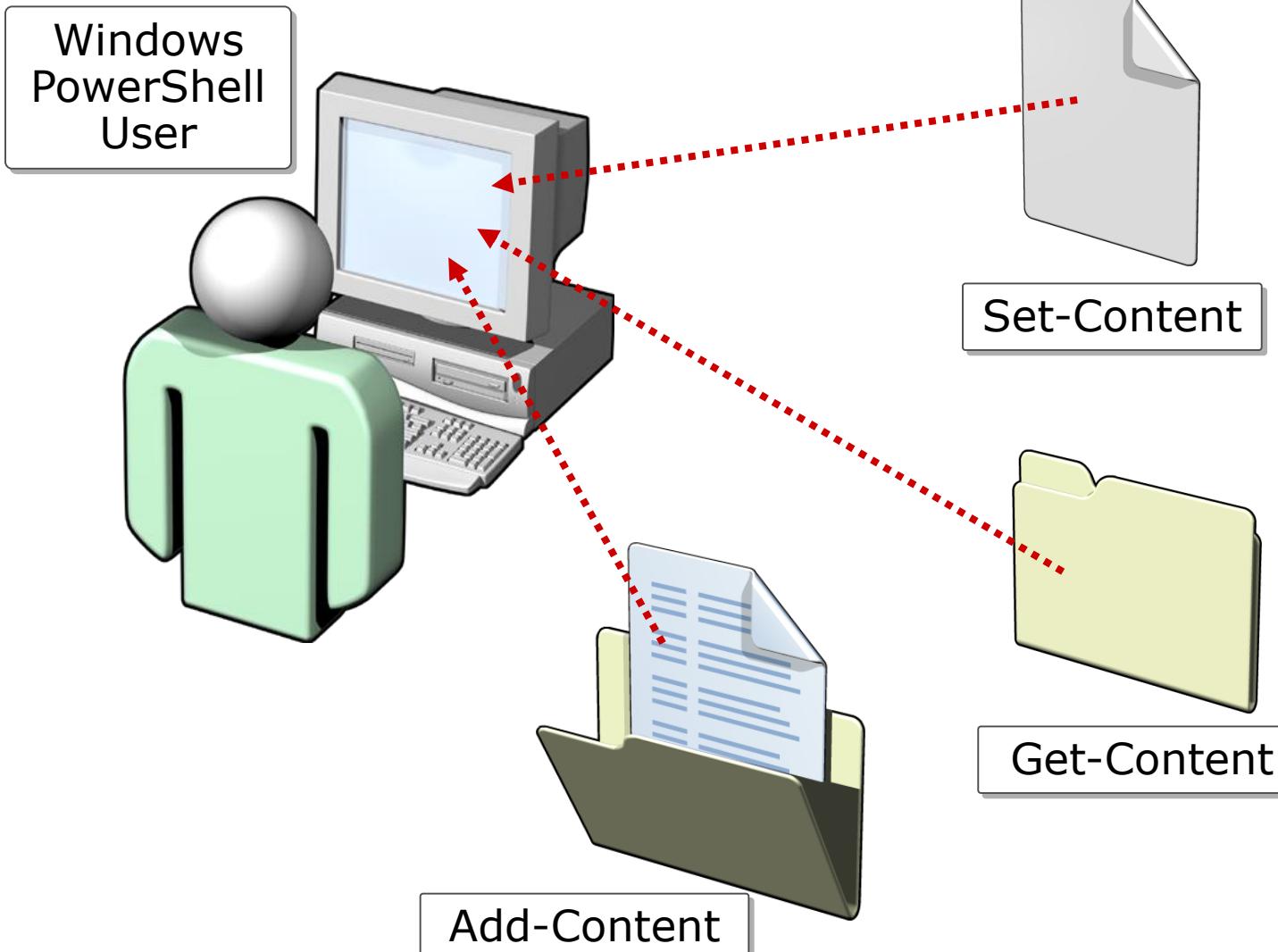
Adatforrások, Provider-ek

- dir parancs valójában a Get-ChildItem, az aktuális adatforrás elemeit adja meg.
- Milyen források vannak?
 - Get-PSDrive, Get-PSProvider
- Hogy tudunk váltani?
 - Set-Location, pl: set-location alias:\
 - Cd hklm:
 - dir – Get-ChildItem mire vonatkozik?
 - set-location d:\home

Output átirányítás (fájl létrehozás)

- „Hajrá Fradi!” >fradi.txt # felülírás, új fájl
 - „Kukába” >\$null
 - Del fradi1.txt 2> \$null # hibakimenet máshova
 - 1> ez nincs, helyette simán >
- Get-Content fradi.txt # PS típus
 - Cat fradi.txt # unix típus
 - Type fradi.txt # dos típus
- Append: >>
 - „Pápa-FTC 0:5” >>fradi.txt # Ha nincs fájl, létrehozza!
- Nincs < vagy << átirányítás!

Fájlok elérése



Példa fájlok használatára

- `dir|set-content dir.txt` # csak a fájl nevek kerülnek bele, miért? (a dir elemei objektumok)
- `dir|out-string|set-content dir1.txt` # teljes tábla kiírásra kerül
 - `dir|out-file dir2.txt` # a teljes táblaszerű kiírás
- `dir|out-printer` # az alapértelmezett nyomtatóra nyomtatunk
- `dir|export-csv dir.csv; import-csv dir.csv`
- `dir|export-clixml dir.xml; import-clixml dir.xml`
- `Out-null` # mint a `/dev/null`

Adatok szűrése- Where parancs

- Adatok megadása:
 - Pipeline
 - Paraméter segítséggel: -inputobject
- Where-Object { szűrőblokk }
 - A szűrőblokk logikai igaz érték esetén átengedi a szűrőn az adott objektum elemet. (logikai operátorok: -gt, -lt, -eq, stb)
 - PL: dir | where-object { !\$_.PSIsContainer}
 - Nem könyvtárak listázása.
 - \$_ Pipe elem aktuális értéke (objektuma).

Where-Object – foreach példa

- A foreach két változata where-object –tel
 - A where a unix grep-hez hasonlít

```
$list = Get-ChildItem -Recurse |where-object  
{!$_.PSIsContainer} #könyvtárakat is megnézzük  
foreach ( $file in $list )  
{ $név = $file.name; $size = $file.length  
    write-output "A $név fájl mérete: $size byte."  
}  
  
Get-ChildItem -Recurse | where-object  
{!$_.PSIsContainer } |ForEach-Object {  
    $név = $_.name; $size = $_.length  
    write-output "A $név fájl mérete: $size  
byte." }
```

Reguláris kifejezések PowerShellben

Karakter	Jelentés	Példa
.	Tetszőleges karakter	.o.th
[xyz]	Egy a felsoroltak közül	[CMRS]andy
[x-z]	Egy az intervallumból	[A-Z]eramy
^	Szöveg kezdet	^Subject:
\$	Szöveg vége	meeting\$
*	0 vagy több ismétlése az előzőnek	w.*s
+	1 vagy több az előzőből	[MZ]+any
?	0 vagy 1 előző	[MZ]?any
\	Speciális karakter a következő	Try\\$

-like és -match operátor példák

```
PS> gci -r | where-object { $_.name -match "\.x[m][ls]" }  
... # reg. kif, az XML, XLS, XMS, XLL fájlok.
```

```
PS> get-process | where-object { $_.name -match "ss$" }  
... # összes system service processz.
```

```
PS> $p = Get-Content planes.txt;  
     $p -match "a[ie]ro?plane"  
... # sorok az airplane, aeroplane, szavakkal.
```

```
PS> $p -like "*plane*"  
... # sorokban a plane bent van.
```

Rendezés – Sort PowerShellben

- Sort-Object [tulajdonság] –paraméterek
 - Ha tulajdonság adott, akkor a szerint rendez
 - Pl: length
 - Paraméterek: -unique, -casesensitive, -descending, -culture név, Lásd: Get-Culture, Set-Culture parancsok
 - Ha nincs semmi paraméter, tulajdonság, akkor alapértelmezésként a teljes objektumot, név szerint, növekvő sorrendben, kis-nagybetű különbséget nem figyelembe véve rendez!
 - PL: dir | sort

Select-Object - Kiválogatás

- Objektum jellemzőket válogat ki
 - Pl: get-process|select-object processname,Id
 - Kiválasztjuk a processzek nevét, azonosítóját.
- Fontosabb paraméterei: -first 4, -last 5, -unique
- Példa: (saját hashtábla kifejezés írható)

```
$ get-process|sort-object processname|select-object -first 5  
$  
$ $p = get-process | select-object ProcessName,@{Name=„Kezdő idő”;  
Expression = {$_StartTime}}  
$ $p
```

Objektumműveletek (Measure-Object)

- Measure-Object, átlag, összeg stb.
 - Get-content dir.txt | measure-object -line –word –char
- Objektum egy tulajdonsága alapján végzi a műveleteket.

```
PS C:\P> dir|measure-object -Property length -sum -Average -Maximum -  
Minimum  
Count : 9  
Average : 3666,66666666667  
Sum : 33000  
Maximum : 11459  
Minimum : 120  
Property : length  
PS C:\P>
```

Processz kezelés

- Get-Process
 - `ps | Where-Object {$_.handles -gt 500}`
- Processz befejezés
 - `$p = Get-Process powershell`
 - `$p.kill` megadja a kill alakját!
 - `$p.kill()` # A PowerShellnek vége...
 - `ps | stop-process –whatif` #mi történne, ha ...
 - `ps | where-object {$_ .name –like „s*” } # fájlnév`
 - `ps | where-object {$_ .name –match „s*” } # reguláris kifejezés illesztés`

Futtatás háttérben (PS 2.0)

- Start-Job –scriptblokk {start-sleep 10}
- Get-job # megkapjuk a futók listáját
- Remove-job –id szám #törölés
- Stop-job –id szám #megállítás
- Invoke-Command: Parancs futtatás helyi vagy távoli gépen
 - Enable-PSRemoting –force
 - Először engedélyezni kell a PS session kezelést, ha ezt használjuk!
 - Invoke parancsot vagy egy PSSession-ben, vagy direktben egy gépen (-computer) futtathatunk.

Szerviz parancsok, indítás, megállítás...

```
PS C:\> Get-Command -Noun Service
```

CommandType	Name	Definition
Cmdlet	Get-Service	Get-Service [[-Name] <String[]>] [-Co...
Cmdlet	New-Service	New-Service [-Name] <String> [-Binary...
Cmdlet	Restart-Service	Restart-Service [-Name] <String[]> [-...
Cmdlet	Resume-Service	Resume-Service [-Name] <String[]> [-P...
Cmdlet	Set-Service	Set-Service [-Name] <String> [-Displa...
Cmdlet	Start-Service	Start-Service [-Name] <String[]> [-Pa...
Cmdlet	Stop-Service	Stop-Service [-Name] <String[]> [-For...
Cmdlet	Suspend-Service	Suspend-Service [-Name] <String[]> [-...

Hitelesítés objektum

- Get-Credential
 - \$c= Get-Credential alma
 - #alma felhasználónévhez
 - \$c=get-credential alma
 - write-host "Felhasználói név: "+ \$c.username
 - write-host "Felhasználói jelszó: "+ \$c.password
 - \$c=Get-Credential
 - Hitelesítést gyakran használunk a Get-WmiObject utasításnál
 - PL: Get-WmiObject Win32_DiskDrive –computername szerver1 –credential \$c

PowerShell bővítmények (SNAPIN)

- A PowerShell moduláris felépítésű
 - `gcm | Where-Object {$_ .name -match "PSSnapin" } # Get, Add, Remove`
- `Get-PSSnapin #` megadja az aktuális modulok listáját
- `Add-PSSnapin Webadministration`
- `Remove-PSSnapin Webadministration`
- Előtte persze az IIS webadmin snapint installálni kell!

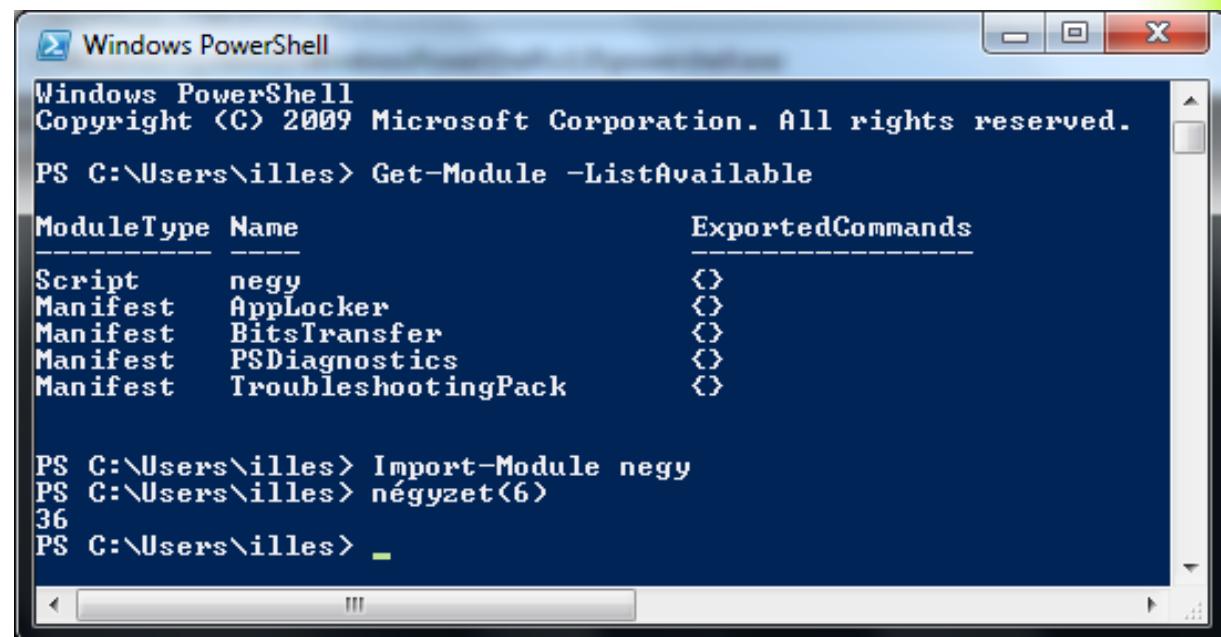
Powershell modulok

- A SNAPIN bővíthetőség egy bináris formátum, telepíteni kell először.
- A modul a PS 2.0-ban jelent meg, forráskódú
 - Get-Module – milyen modulok érhetők aktuálisan el
 - Get-Module –ListAvailable #összes elérhető modul listázás
 - Import-Module név # adott modul betöltése
 - Hasznos függvények, álnevek, változók definíójának gyűjtőhelye.
 - \$env:PSModulePath

```
PS D:\home\ps> $env:PSModulePath
C:\Users\illes\Documents\WindowsPowerShell\Modules;C:\Windows\system32\WindowsPowerShell\v1.0\Modules\
```

Saját modul (Script modul)

- 1. .psm1 kiterjesztés a saját modulnak.
- 2. A könyvtár neve azonos a fájl névvel, ezt helyezzük a „My Documents\WindowsPowerShell\Modules könyvtárba!
- 3. Import-Module negy.psm1



The screenshot shows a Windows PowerShell window titled "Windows PowerShell". The command "Get-Module -ListAvailable" is run, displaying a table of available modules:

ModuleType	Name	ExportedCommands
Script	negy	○
Manifest	AppLocker	○
Manifest	BitsTransfer	○
Manifest	PSDiagnostics	○
Manifest	TroubleshootingPack	○

Subsequent commands "Import-Module negy" and "négyszet(6)" are run, followed by the number "36" and the prompt "PS C:\Users\illes>".

WMI

- Windows Management Instrumentation
 - Infrastruktúra kezelés
- WMI Tools (külön kell installálni)
- WMI osztályok, névterek
 - Get-WmiObject -Class __Namespace -Namespace root
- **Get-WmiObject –list # wmi osztályok listája**
 - Get-WmiObject Win32_Diskdrive
 - Get-WmiObject Win32_NetworkAdapter
 - Stb...

Active Directory (PS 1.0)

- ADSI – Active Directory Service Interface
- ADSI Providers
 - WinNT : NT4 PDC, BDC, és lokális felhasználók
 - LDAP : Win2000 óta az AD-k ezzel mennek
 - NDS : Novell Directory Services
 - Pl: \$a=[ADSI]"LDAP://dc=alma,dc=fa"
 - \$u=\$a.create(„organizationalunit”,“TesztUnit”)
 - \$u.setInfo();
- Stb....

Active Directory (PS 2.0)

- Windows 2008 R2 serverhez megjelent ez a modul.
- Installálni kell először mint win2008 részt:
 - Active Directory for Windows PowerShell
- Ezután importáljuk:
 - Import-module activedirectory
- Kapunk sok új parancsot:
 - Get-command –module activedirectory

További lehetőségek I.

- IIS szerver kezelése
 - IIS hozzáadja a kezelés parancsokat
 - aspnet.inf.elte.hu

The screenshot shows the Windows PowerShell ISE interface. On the left is a code editor window titled "aspnet_user.ps1" containing PowerShell script code. On the right is a "Commands" palette with a search bar set to "Modules: All" and "Name: web". The palette lists various cmdlets related to web management, such as Get-WebRequest, Get-Website, Get-WebsiteState, Get-WebURL, Get-WebVirtualDirectory, Install-PswaWebApplication, Invoke-WebRequest, New-WebApplication, New-WebAppPool, New-WebBinding, New-WebFtpSite, New-WebGlobalModule, New-WebHandler, New-WebManagedModule, New-WebServiceProxy, New-Website, New-WebVirtualDirectory, Publish-BCWebContent, Remove-WebApplication, and Remove-WebAppPool. At the bottom of the interface, a PowerShell prompt shows the command "get-help Uninstall-PswaWebApplication" being run.

```
Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
aspnet_user.ps1 X
139 # $acl=get-acl $útvonal
140 #Add this access rule to the ACL
141 #$acl.SetAccessRule($rule)
142 #Write the changes to the object
143 #set-acl $útvonal $acl
144 $útvonal+" jogosítvány állítása!"
145 full_control($név) # így csak 1 paramétert szeret
146 # webapplication létrehozása
147 New-WebApplication -Site "Default Web Site" -Name $név -PhysicalPath $útvonal
148 New-WebApplication -Site "Default Web Site" -Name $név+_service" -PhysicalPath
149 }
150 else
151 {
152     "Adminisztrátor jog kell a futtatáshoz!"
153 }
154

+ CategoryInfo          : ResourceUnavailable: () [Get-Help], HelpNotFoundException
+ FullyQualifiedErrorId : HelpNotFound,Microsoft.PowerShell.Commands.GetHelpCommand

PS C:\Users\illes\Documents> get-help Uninstall-PswaWebApplication
NAME
    Uninstall-PswaWebApplication
SYNOPSIS
    Uninstalls a web application, [Web Application Name] <string> [-WebSiteName <string>] [-Certificate] [-WhatIf] [-Confirm] [<CommonParameters>]
```

Számítógépes rendszerek

További lehetőségek II.

- Exchange szerver kezelése
 - Exchange Management Shell-As administrator
 - Get-Excommand
 - New-ManagementRoleAssignment –Role „Mailbox Import Export” –user admin
 - New-MailboxImportRequest –mailbox usernév –FilePath <\\gépnév\share\user1.pst>
 - ...
- SQL szerver kezelése
 - Get-help sqlserver

Csomagoljunk be ...

- A shell script csomagol mintájára!
- Használat, csomagolás:
 - csomagol.ps1 fájl1 fájl2 ... >csomi.ps1
- Kicsomagolás: csomi.ps1

```
# csomagoló: csomagol.ps1
# Használat: csomagol.ps1 fájl1
fájl2 ...
"#Csomagoljunk!"
foreach($i in $args)
{
    "echo $i"
    "@"
    Get-Content $i # cat
    ""@ >$i"
    "echo '$i vége!''"
}
"#Csomagolás vége!"
```

Kicsomagolunk ..

```
PS C:\d\home\ps> cp .\csomi.ps1 csomi
```

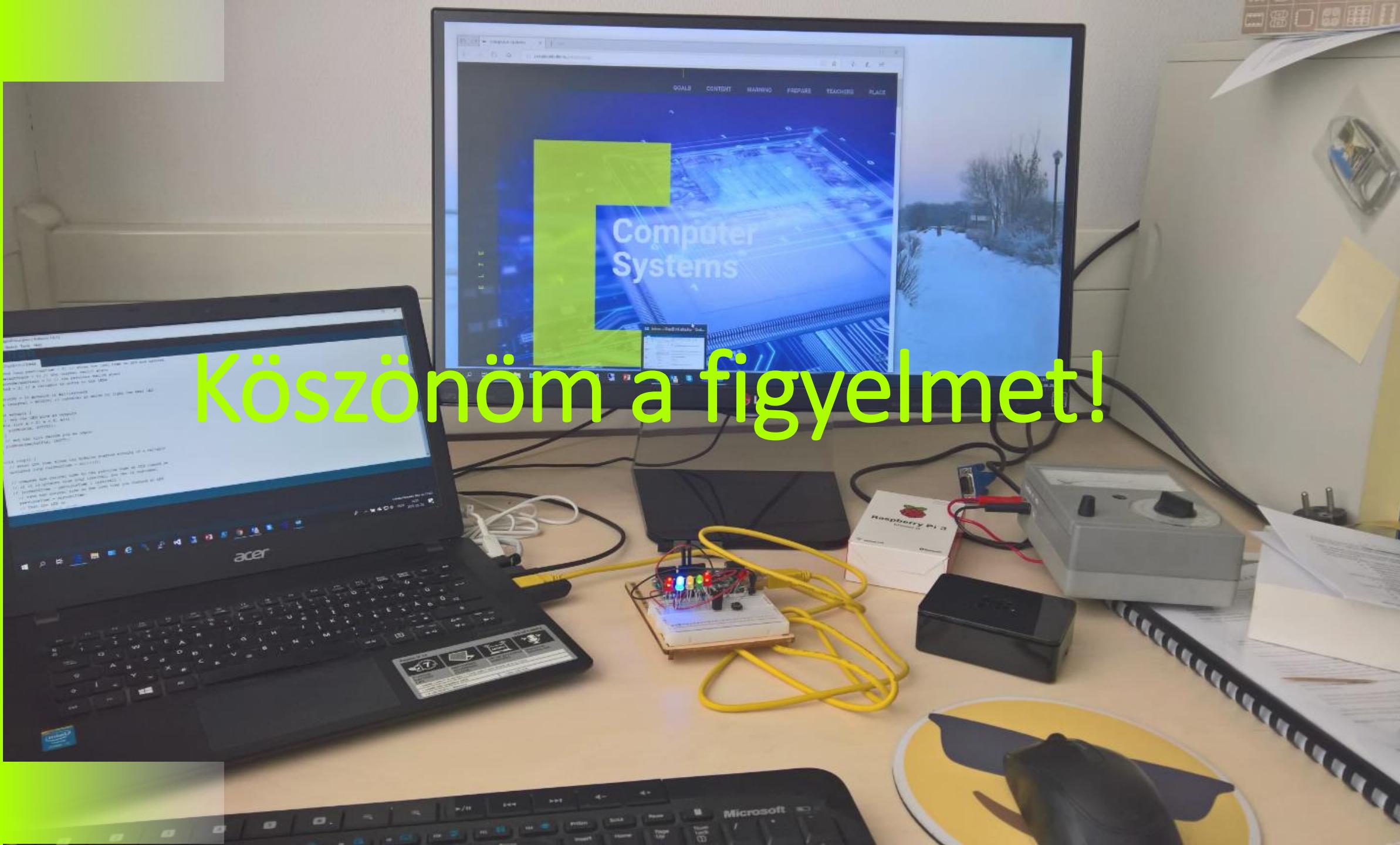
```
PS C:\d\home\ps> cd csomi
```

```
PS C:\d\home\ps\csomi> .\csomi.ps1
.\fradi.ps1
.\fradi.ps1 vége!
.\hajra.ps1
.\hajra.ps1 vége!
```

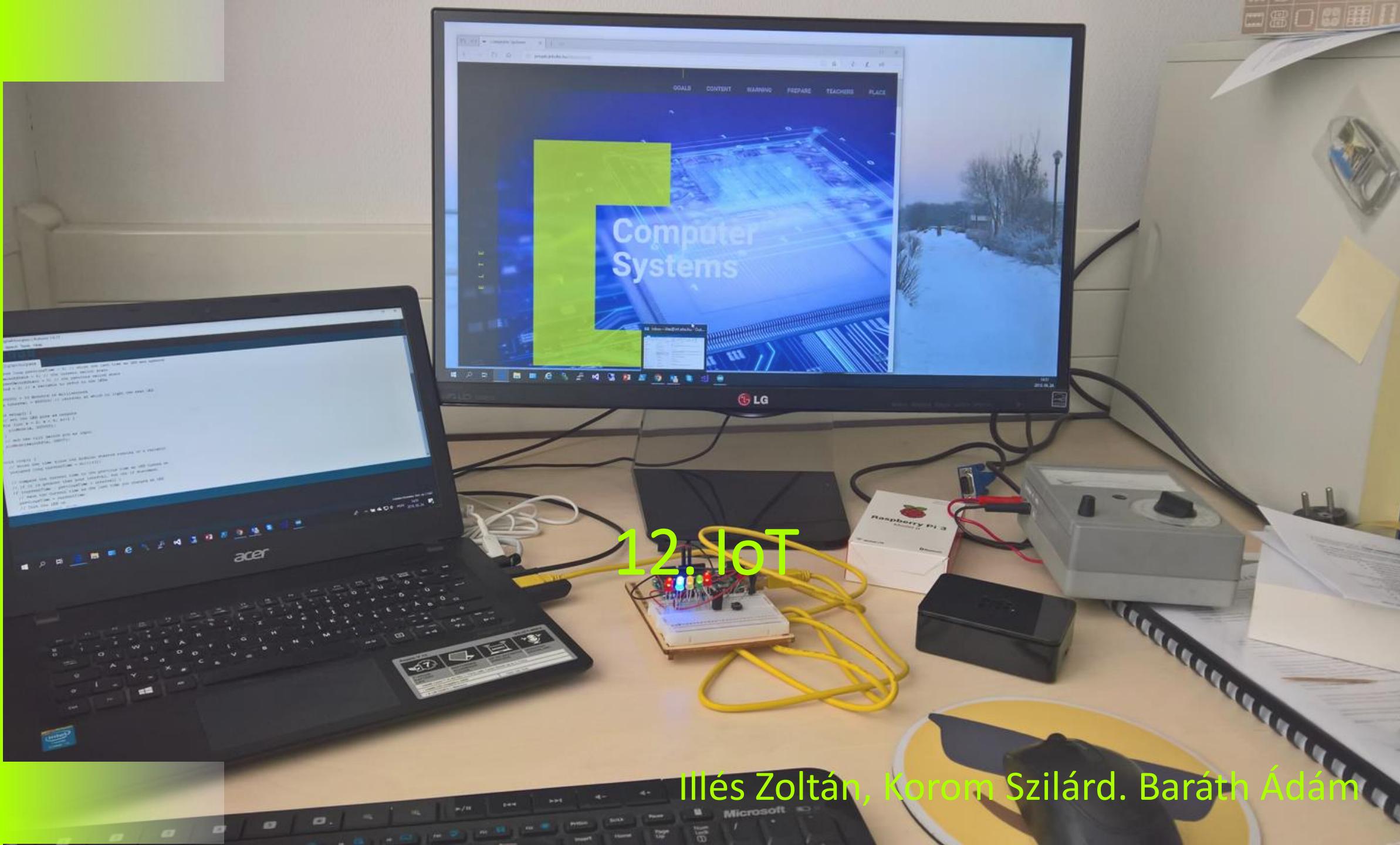
```
PS C:\d\home\ps\csomi> ls
```

```
Directory: C:\d\home\ps\csomi
```

Mode	LastWriteTime	Length	Name
----	-----	-----	-----
-a---	2015. 11. 16.	8:52	526 csomi.ps1
-a---	2015. 11. 16.	8:53	120 fradi.ps1
-a---	2015. 11. 16.	8:53	76 hajra.ps1



Köszönöm a figyelmet!



12. IoT

Illés Zoltán, Korom Szilárd, Baráth Ádám

Visszatekintés

- Számítógépek, számábrázolás, kódolás,felépítés, fájlrendszerek
- Alapvető parancsok, folyamatok, szűrők
- Változó, parancs behelyettesítés,aritmetikai, logikai kifejezések
- Script vezérlési szerkezetek,Sed,AWK
- Hálózatok alapvető jellemzői
- Powershell
- RSA alapok

Mi jön ma?

- Kis teljesítményű számítógépek
- Internet of Things - IoT

Nagy számítógépek

- Nagy számítási kapacitás biztosítása
 - Nagyon sok processzor
 - Sok memória
 - Nagy háttértár (Osztott fájlrendszer)
 - Adatbányászat, szimulációs feladatok
 - <https://www.top500.org>
 - 2019 június, első: **Summit (Kína)**
 - Magok száma: 2,414,592

Klasszikus számítógépek

- 1-2 processzoros alaplap
- Processzoronként 2-12 mag.
- Személyi számítógépekben 4-32GB memória, kis szerverekben 8-512GB.
- Háttértár: SSD előretörés, 120GB-1TB SSD, 1-16 TB
- Érintőkijelzők

Mobilizáció

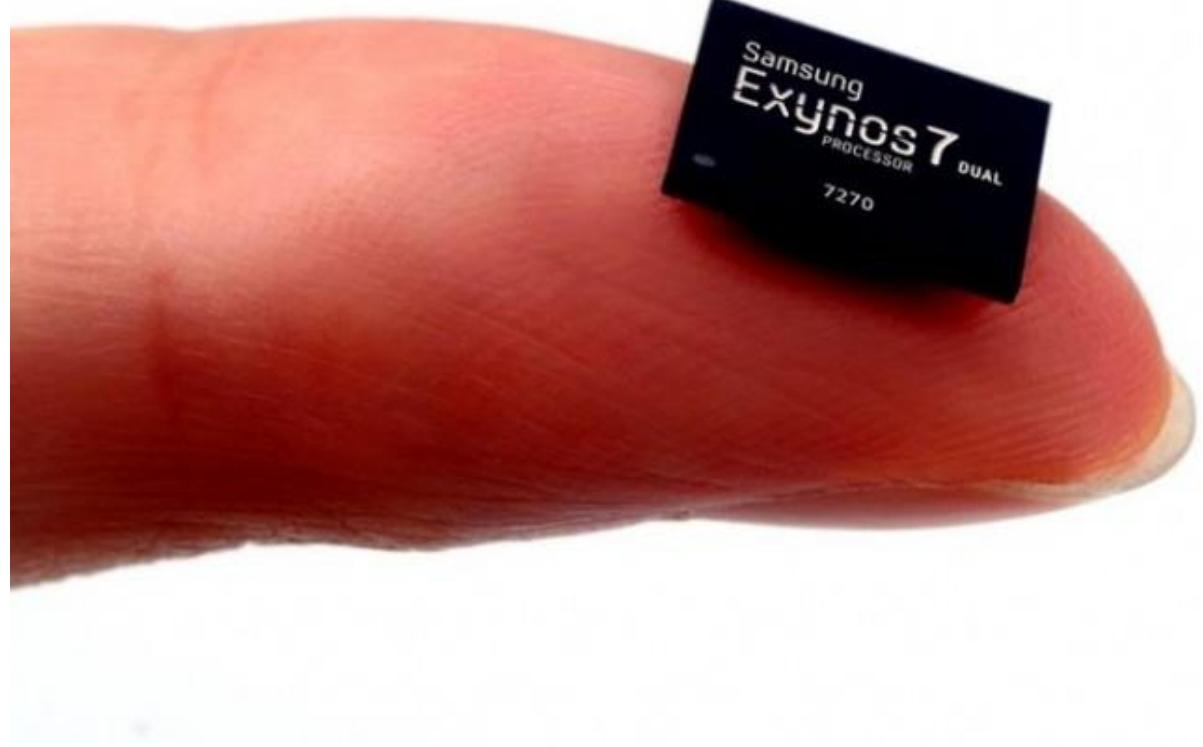
- Mobil telefon – okos telefon – legújabb számítógép helyettes?
 - Mobil operációs rendszerek
 - 4-8 mag, 2-8 GB RAM, 16-128GB tárhely
 - Ezen mobil magok teljesítménye nem azonos a desktop, notebook társakhoz viszonyítva!
 - Dokkolási lehetőség
 - Jelenleg is létezik!, így klasszikus munkahellyé kezd válni

Miniatürizáció

- 2019 : 5 nanométeres technológia bejelentése!
- indul a 7 nanométeres tömeggyártás.
 - Viselhető processzor- viselhető ipar (wearable industry)
- SoC – System on Chip, minden bele!
- Jellemzően az alábbiakat tartalmazzák:
 - CPU – 4, 8 maggal (Cortex-A53 4 mag)
 - LTE modem, WIFI, Bluetooth, memória (DRAM, NAND)

SoC a valóságban

- Exynos 7270 dual
 - 2 darab CortexA53
 - Kb. 1x1 cm
 - Kb. 1 mm vastag
 - Teljes számítógép!



Internet of things



IoT mögött

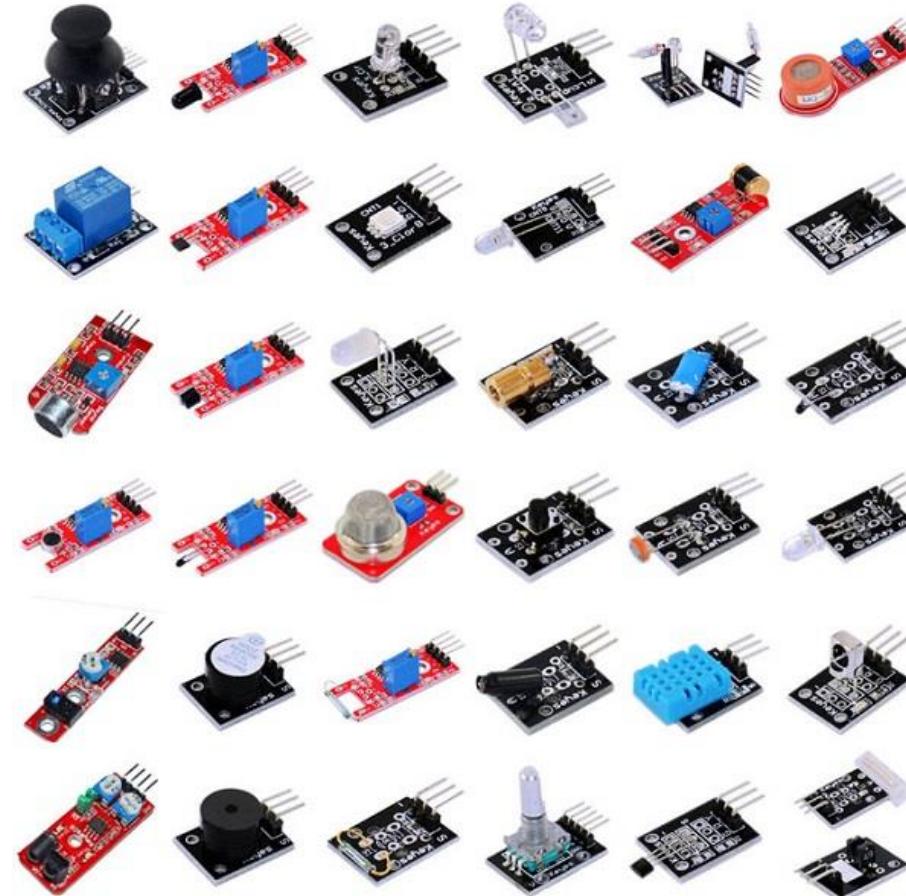
- Beágyazott rendszerek
- Valós idejű rendszerek
- Os
 - Nincs OS
 - Real-time OS
 - Ubuntu
 - Egyéb



Mi az IoT?

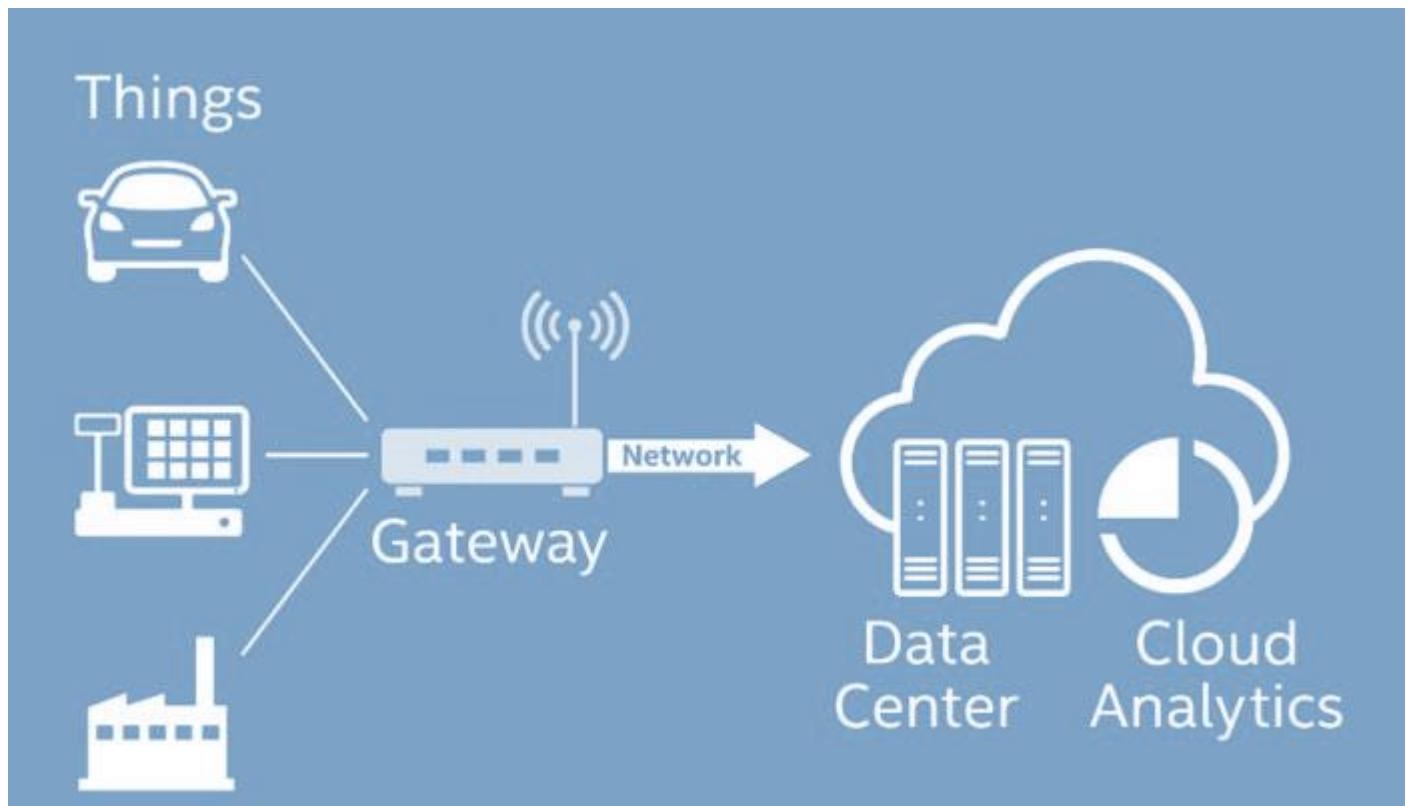


Szenzorok



Hálózat és internet

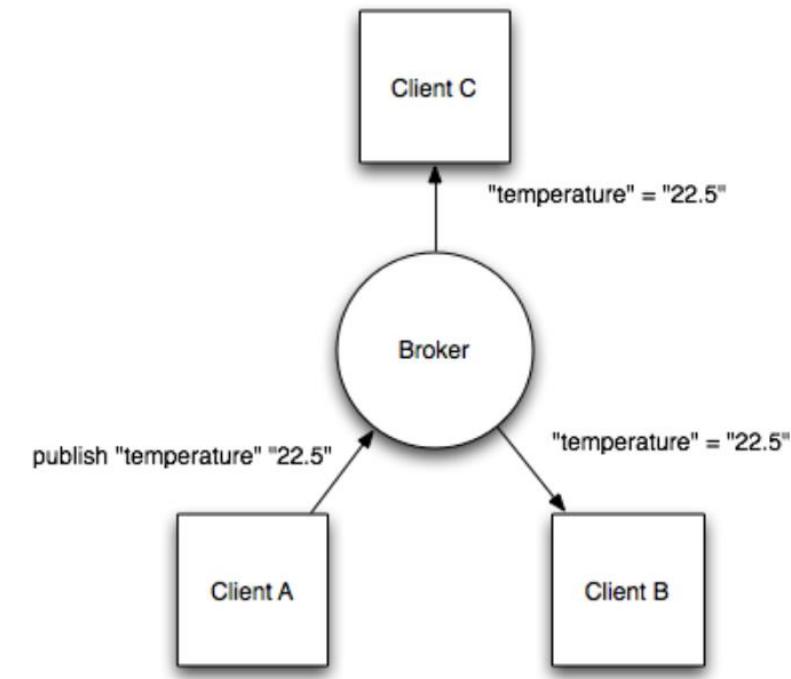
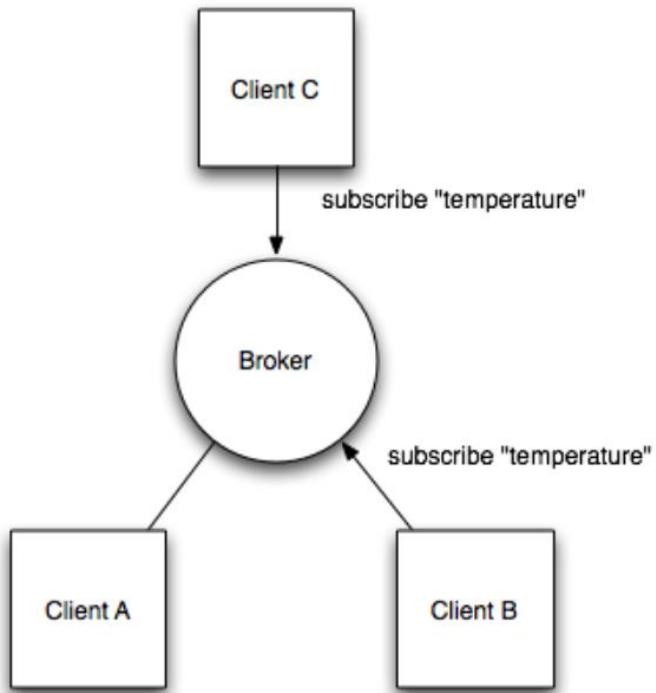
- Protokollok
 - MQTT
 - CoAP
 - XMPP
- IoT gateway



MQTT

- MQTT = Message Queueing Telemetry Transport
- Nyílt, ingyenes
- publish/subscribe (one-to-many) modell
- A „publisher” nem tudja ki „subscribed”
- Üzenet orientált
- Kliens + Broker

MQTT



MQTT vs HTTP

	MQTT	HTTP
Design orientation	Data centric	Document centric
Pattern	Publish/subscribe	Request/response
Complexity	Simple	More complex
Message size	Small, with a compact binary header just two bytes in size	Larger, partly because status detail is text-based
Service levels	Three quality of service settings	All messages get the same level of service
Extra libraries	Libraries for C (30 KB) and Java (100 KB)	Depends on the application (JSON, XML), but typically not small
Data distribution	Supports 1 to zero, 1 to 1, and 1 to <i>n</i>	1 to 1 only

Forrás: <http://www.tmit.bme.hu/sites/default/files/2017-04-04-IoT.pdf>

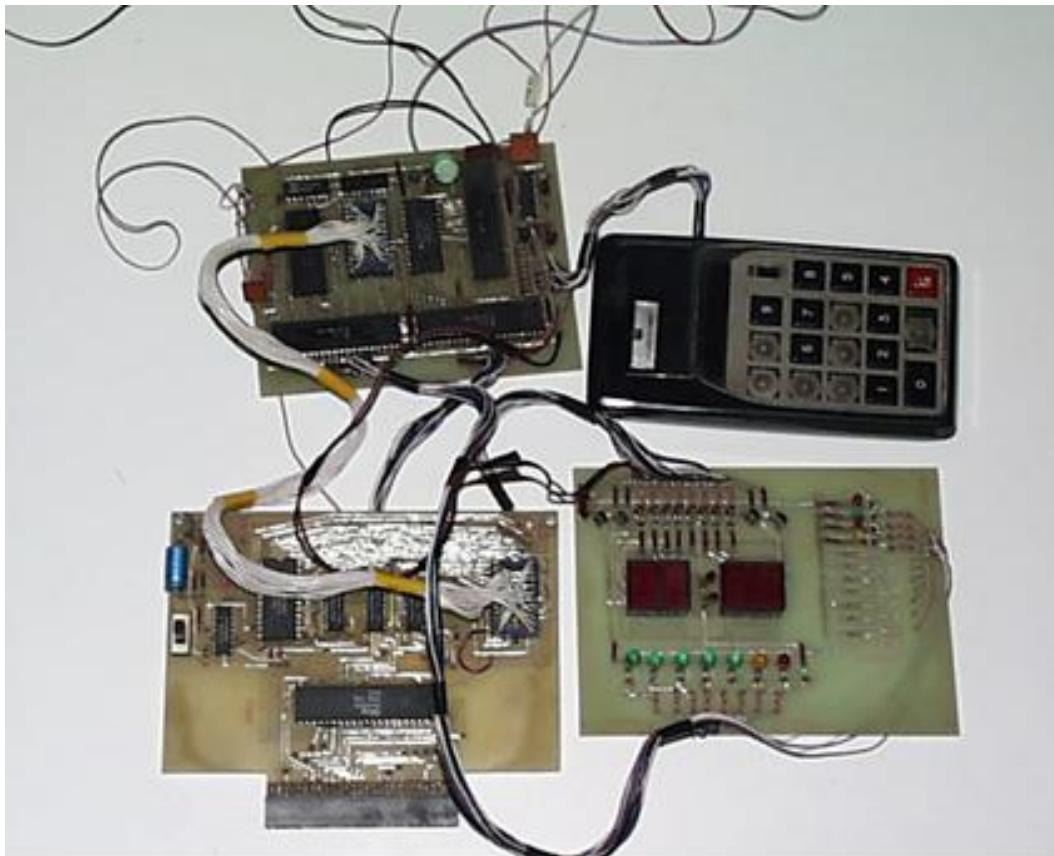
CoAP

- CoAP = RFC 7252 Constrained Application Protocol
- Mint a HTTP, dokumentum átviteli protokoll
- Csomagok kisebbek
- Multicast
- UDP
- REST modell
 - Szerver <-> Kliens
 - URL
 - GET,POST,PUT,DELETE

XMPP

- XMPP = Extensible Messaging and Presence Protocol
- XML alapú üzenetküldtés
- Request/Response
- Skálázható
- Google Talk

Boards (Vezérlők) - tegnap



Boards, vezérlők - ma

- „Ready to use” vezérlők, kiegészítve bemeneti-kimeneti csatlakozással.
- Talán a két leggyakrabban használt:
 - Arduino család
 - https://en.wikipedia.org/wiki/List_of_Arduino_boards_and_compatible_systems
 - Raspberry PI család
 - https://en.wikipedia.org/wiki/Raspberry_Pi

Arduino Uno

- Főbb jellemzők:
 - Atmega328P vezérlő, 16MHz, 8 bites
 - 6 analog input
 - 14 digital input/output, ebből lehet 6 PWM.
 - Pulse-Width Modulation, analóg output szimulálására!
 - 32kb flash RAM, 1kb EEPROM, 2kb SRAM (static RAM, változóknak)
 - A 32kb tartalmazza a program kódot, ebből kb. 2 kb bootloader
 - A bootloader elsősorban a PC-USB kapcsolatért (upload) felel!
- Mire elég ez?

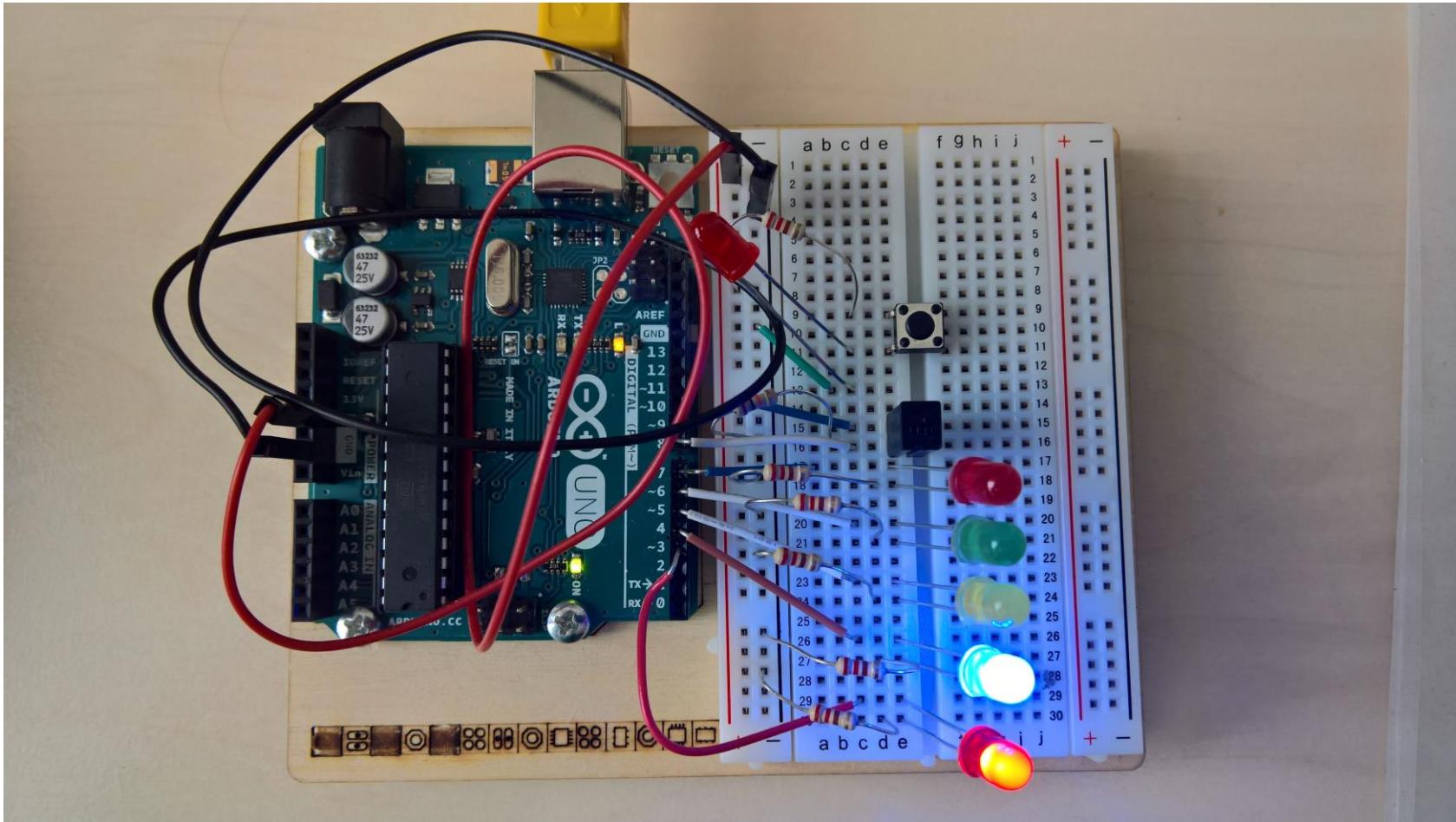
Arduino programozása

- Jellemző modell: nincs operációs rendszer, csak egy célfeladat kódja kerül a memóriába!
- Programozható C nyelven vagy gépi kódban.
- Program szerkezet:
 - Setup függvény – egyszer, induláskor lefut.
 - Loop függvény – ez hajtódik végre örökké
- Arduino IDE – letölthető, jelenlegi verzió: 1.6.13
 - Arduino.cc/en/Main/Software

Arduino példa

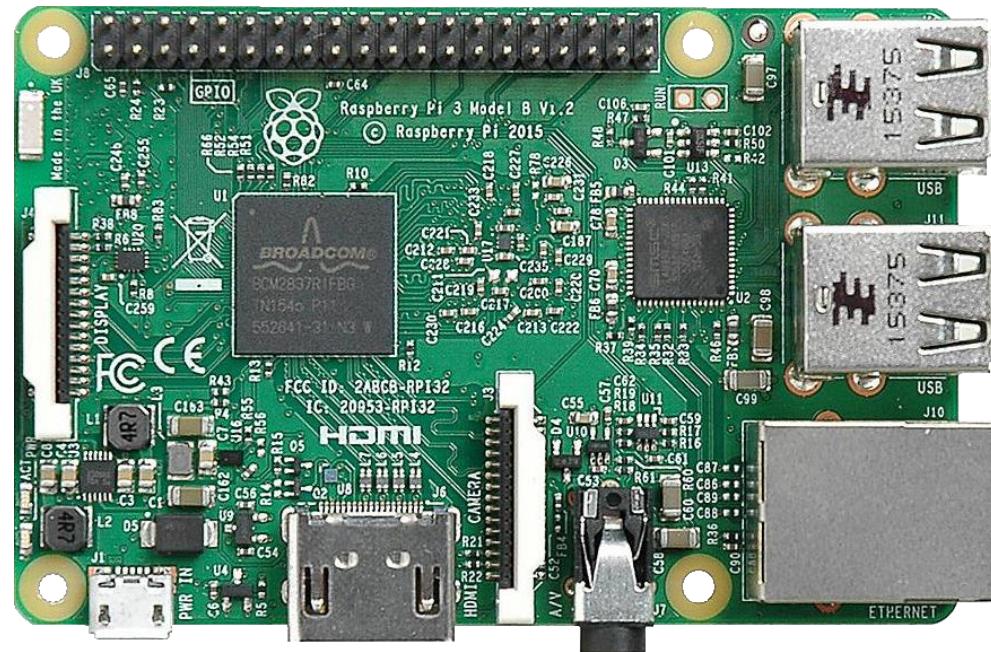
- Ledes „homokóra”
- Setup függvény:
 - pinMode állítás (input,output): pinMode(3,OUTPUT)
- Loop függvény:
 - digitalWrite(led, HIGH); // adott portra 5 volt, LOW – 0 volt
 - X=digitalRead(led);
 - T=millis(); // idő millisec-ben

Konkrétan



Raspberry Pi

- Bankkártya méretű
- 2012 óta a legnépszerűbb mini PC
- Tanítási módszert kínál
 - Programozás
 - Hardware fejlesztés
- <http://hackster.io>



Raspberry PI 3 - Hardware

- Processzor ARM A53 (64bit)
- RAM 1GB
- GPIO 40 pin (analóg + digitális)
- Portok HDMI, Jack, micro SD, 4 USB, Ethernet,Wifi,Bluetooth
- OS Linux alapok VS Windows 10 IoT

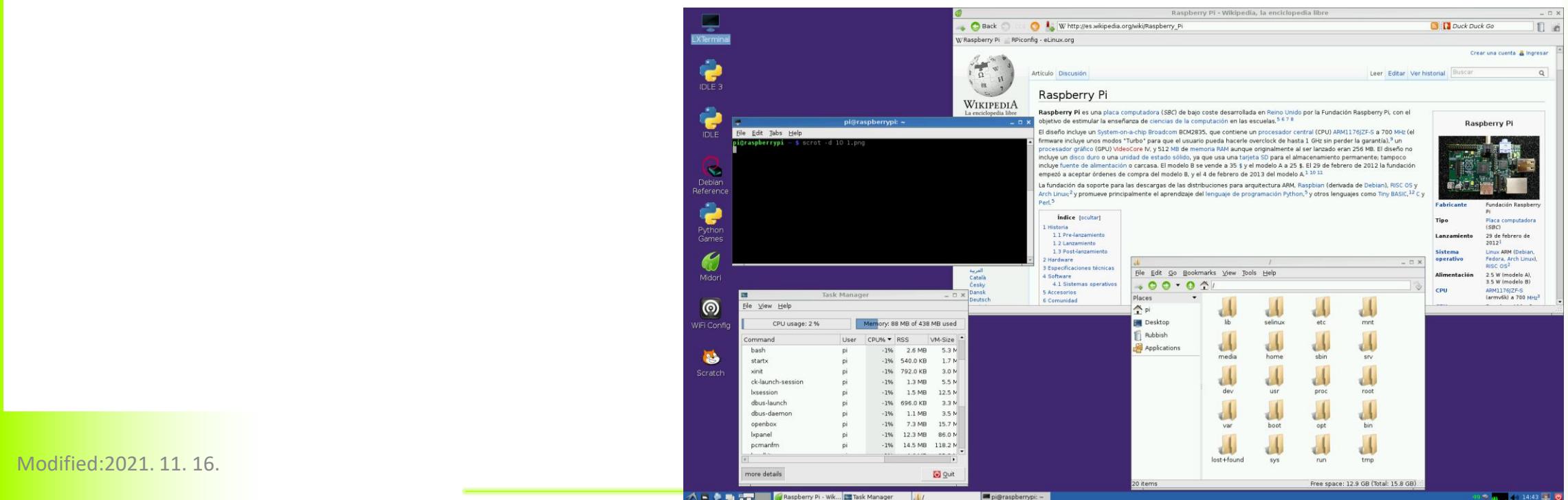
Raspberry Pi 4

- Processzor 1.5GHz quad-core ARM
- RAM 2/4GB
- GPIO 40 pin (analóg + digitális)
- Portok 2xmicro-HDMI,Jack, micro SD,
USB: 2× USB 3.0 és 2× USB 2.0 portok,
Ethernet
Wifi, Bluetooth 5.0



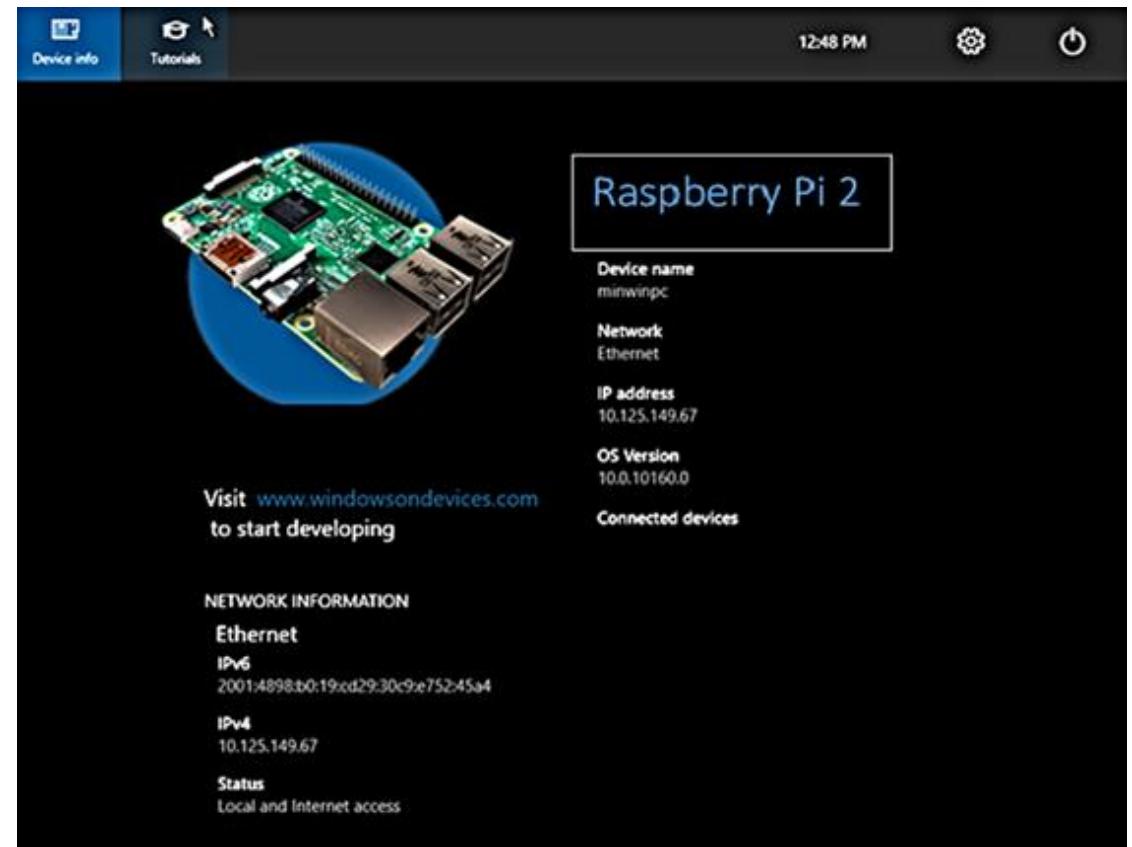
Raspberry PI Raspbian

- Operációs rendszer az SD kártyán
- Linux alapokon (Például Raspbian)
 - Fejlesztés: C / C++ / Python / Node.js



• Raspberry PI (Windows 10 IoT)

- Fejlesztés .NET környezetben
- C#, JavaScript, F#, C++, stb...
- Universal Windows Platform
- Microsoft Azure



Windows 10 IoT

- Nem valós idejű operációs rendszer
 - “közel valós idejű”
- Windows 10 IoT for industry devices
 - Desktop Shell, Win32 apps, Universal apps and drivers, Minimum: 1 GB RAM, 16 GB storage, X86/x64
- Windows 10 IoT for mobile devices
 - Modern Shell, Mobile apps, Universal apps and drivers, Minimum: 512 MB RAM, 4 GB storage, ARM
- Windows 10 IoT Core
 - Universal Apps and Drivers, No shell or MS apps, Minimum: 256MB RAM, 2GB storage, X86/x64 or ARM

Windows? 10? IoT?

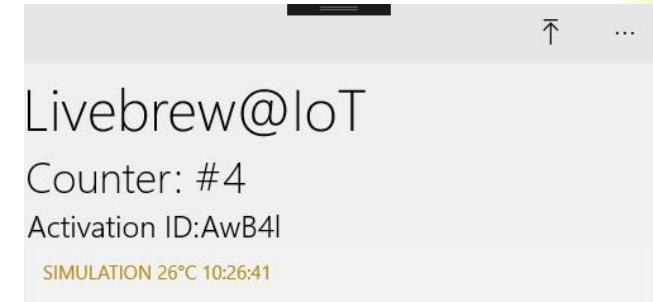
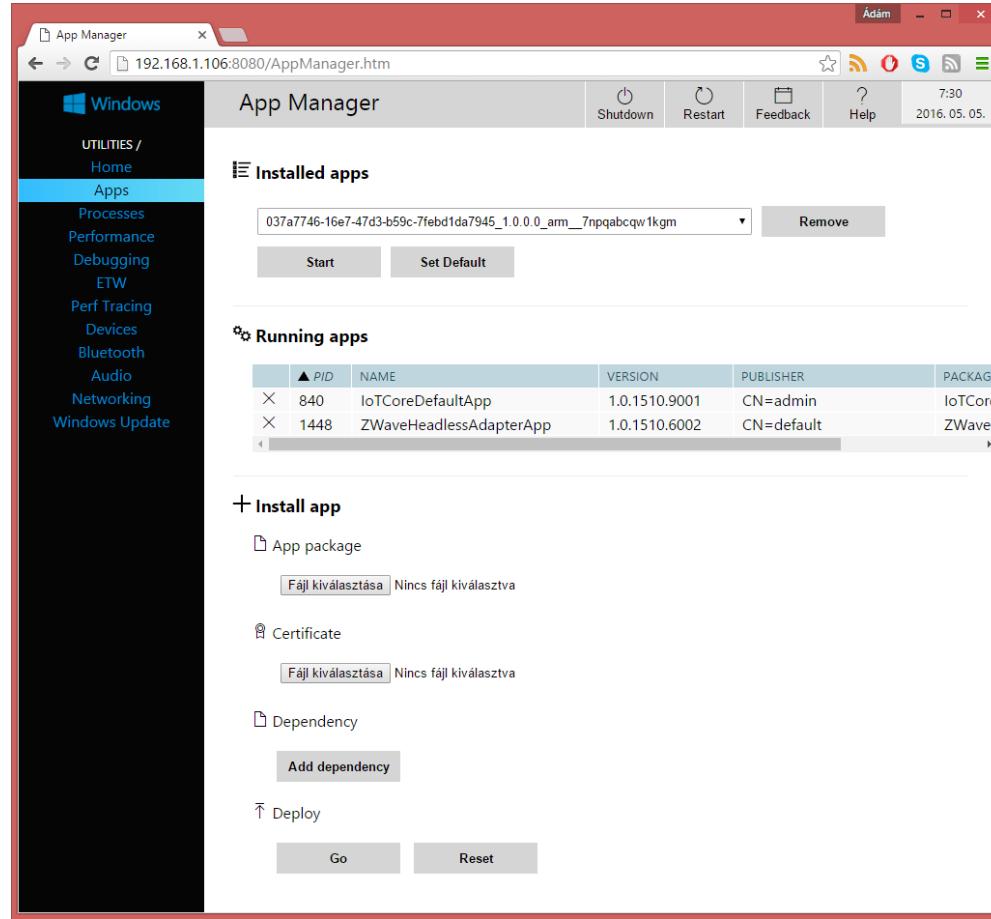
- Igen, ez jó (az iparnak)
 - Nem, nem lesz kék halál
 - Biztonság (pl: BitLocker)
 - Folyamatos támogatottság
 - “It just works”
- Mire is jó?
 - Ipari alkalmazások futtatásához (pl: sörfőzés)
 - Vékony kliens appok -> Felhő
 - Bitlocker -> azonosító kulcsokat nem tudják ellopni

LiveBrew – főzés felügyelete

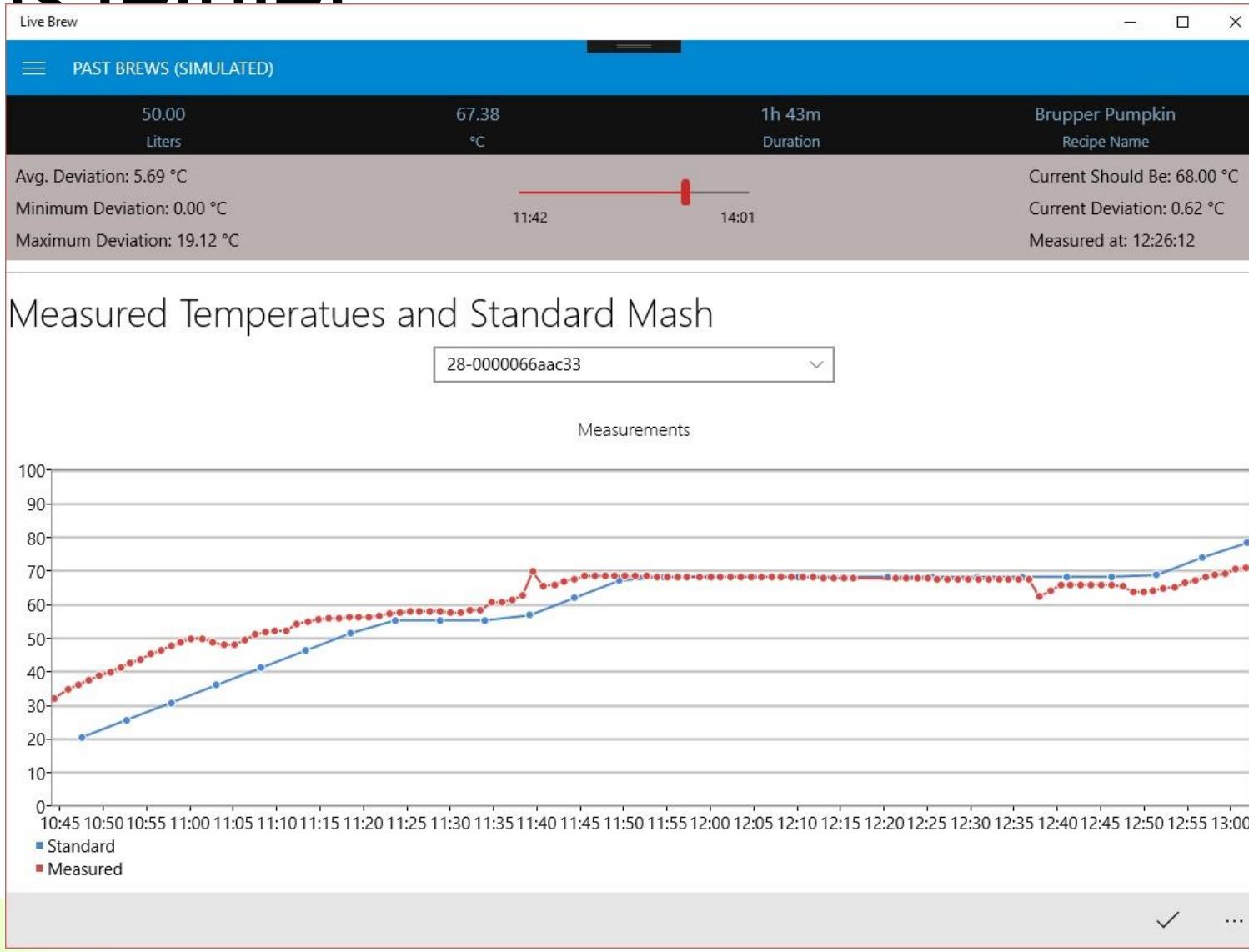
- Raspberry PI 2
 - Tápegység, SD kártya, Board, Wifi modul,
 - One Wire Pi Plus shield + DS18B20 hőmérő szenzorok



Rendszer telepítése SD kártyára



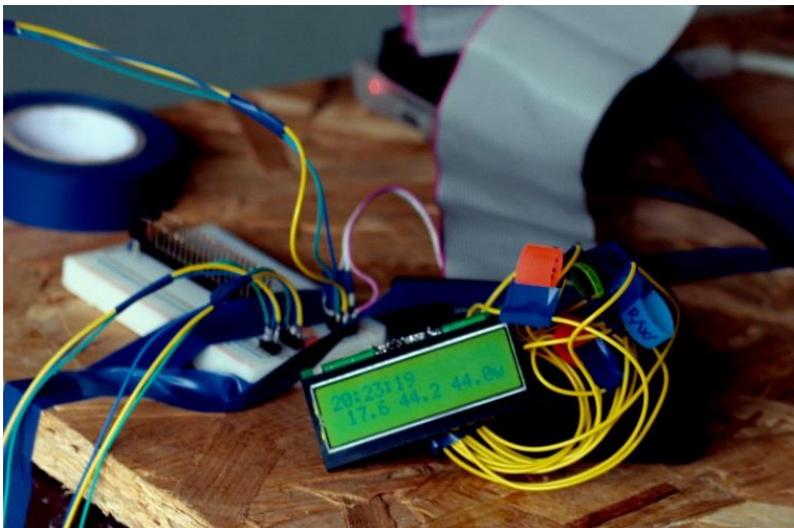
Kliens felület



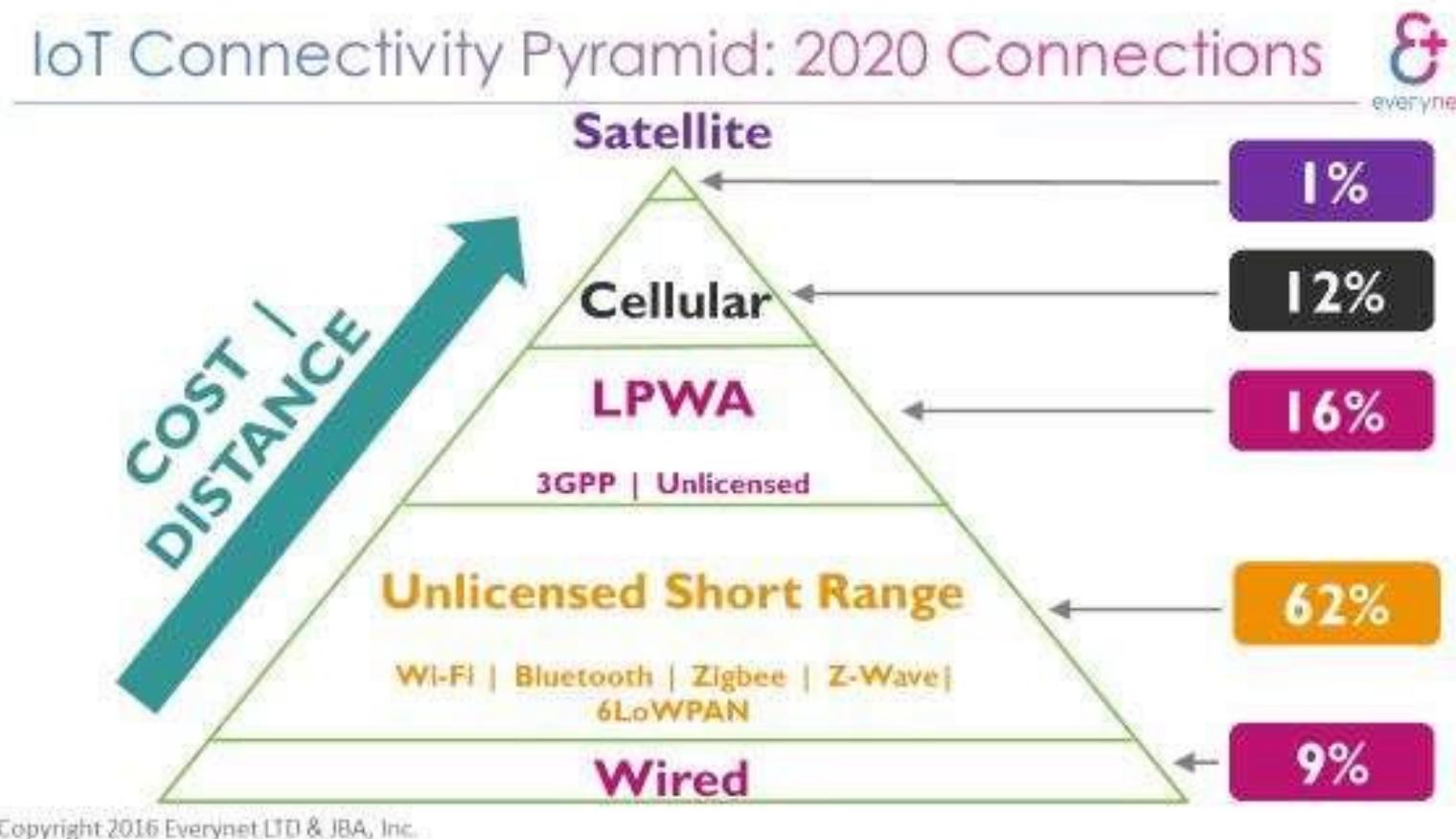
Megvalósítás

- Adatbázis tervezése
- Azure oldali konfiguráció
- IoT készülék rögzítse a hőmérsékletet és azt „felküldje a felhőbe”.
- Értesítések megvalósítása
- Refaktorálás
- Felület véglegesítése
- Hibák keresése, javítása

- Élőben a rendszer tesztelése:



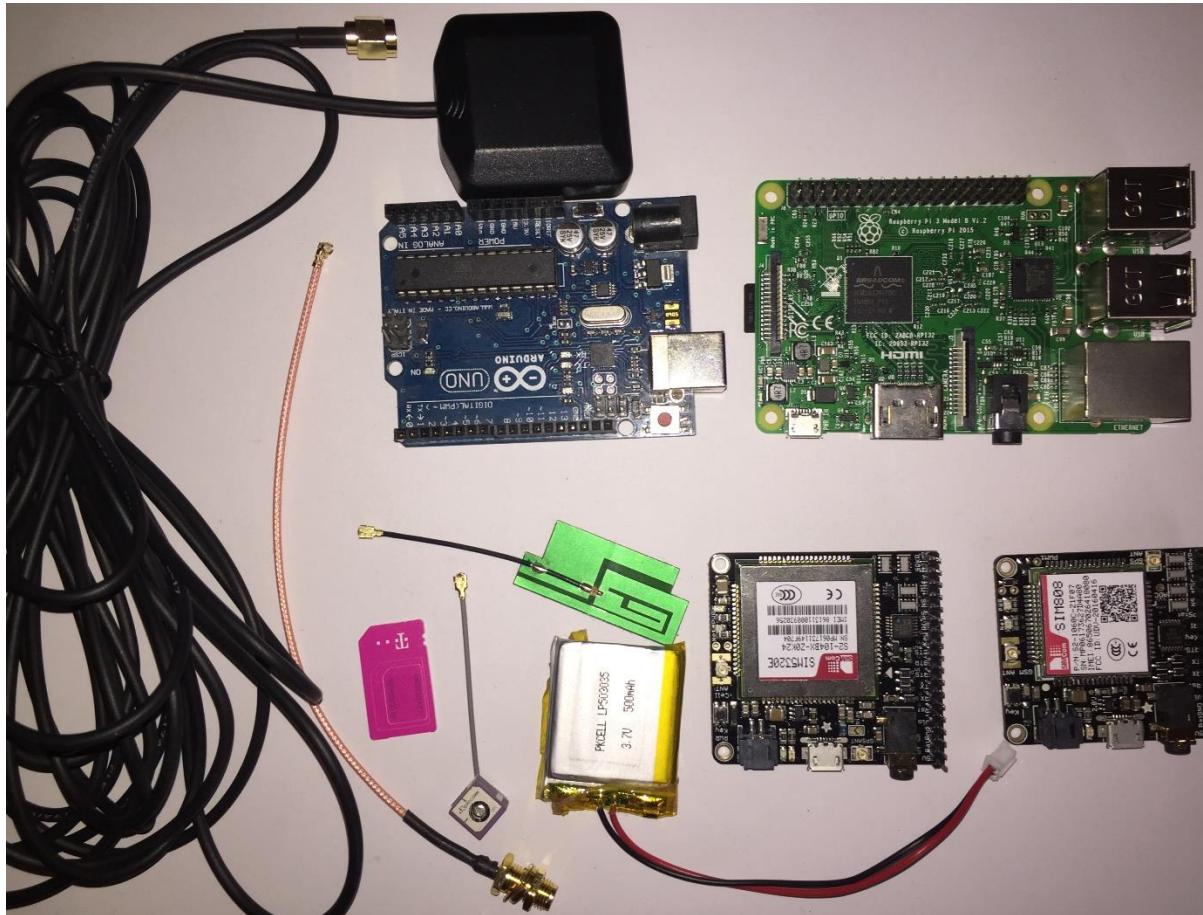
Hálózatprobléma



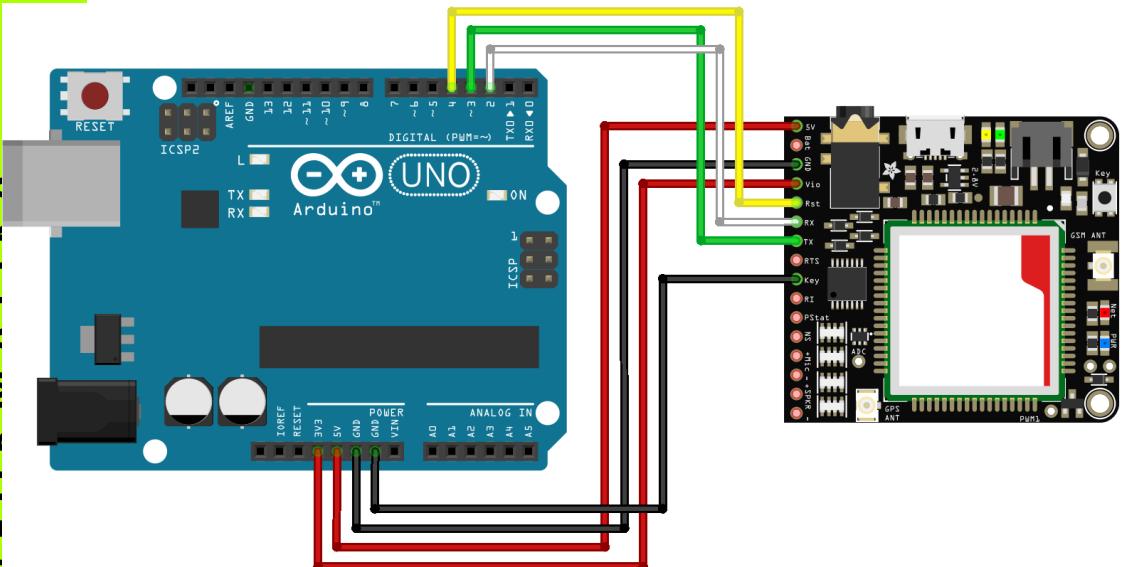
Hálózatprobléma



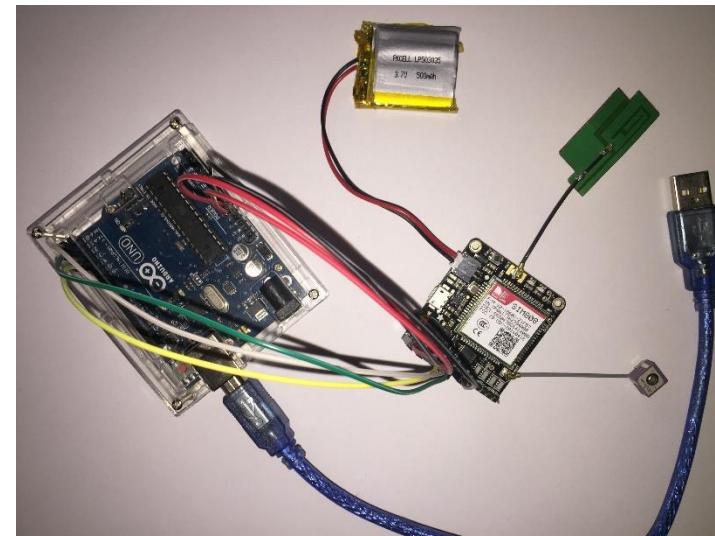
Hálózatprobléma



Hálózatprobléma

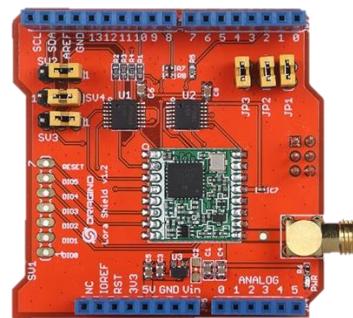


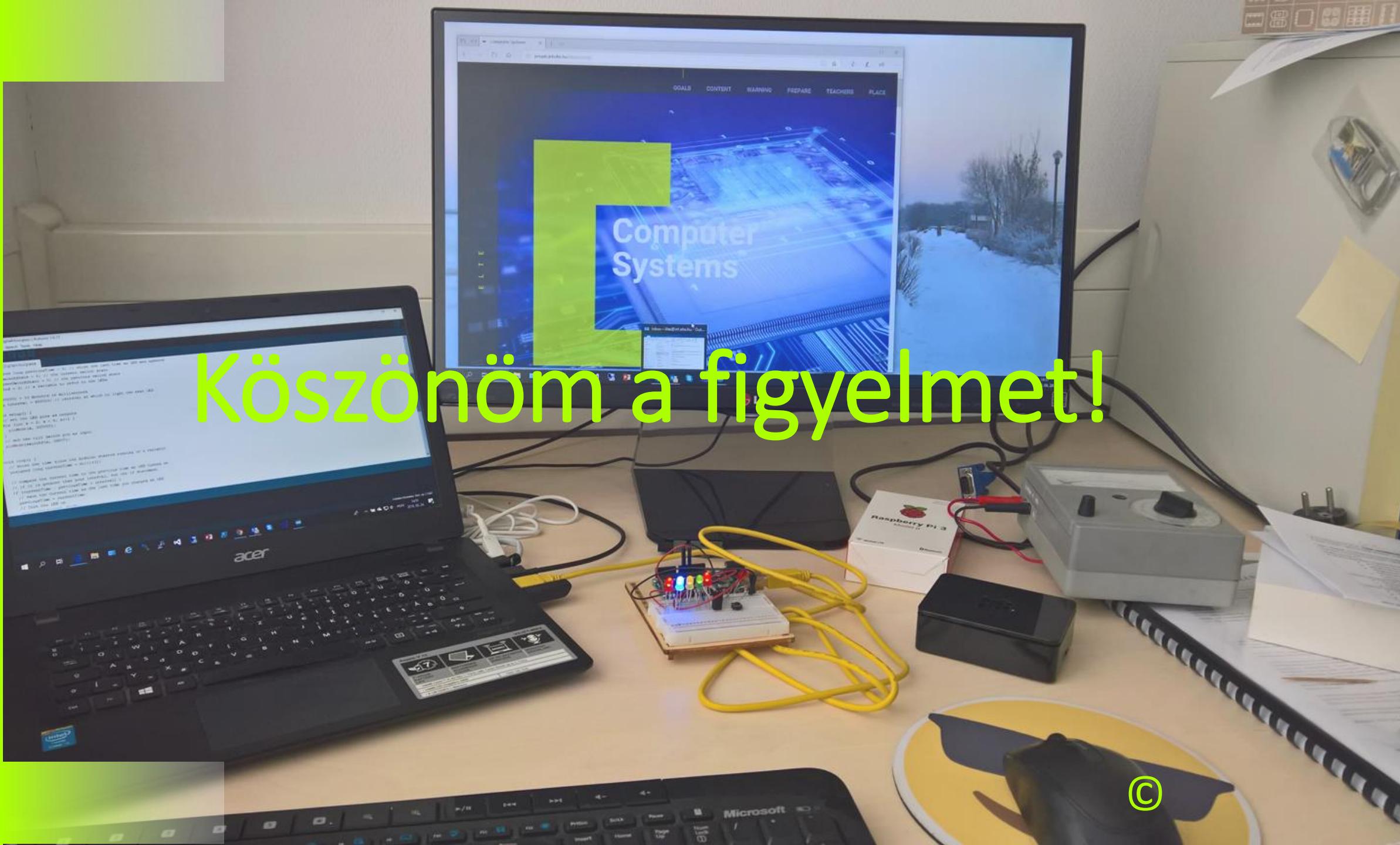
fritzing

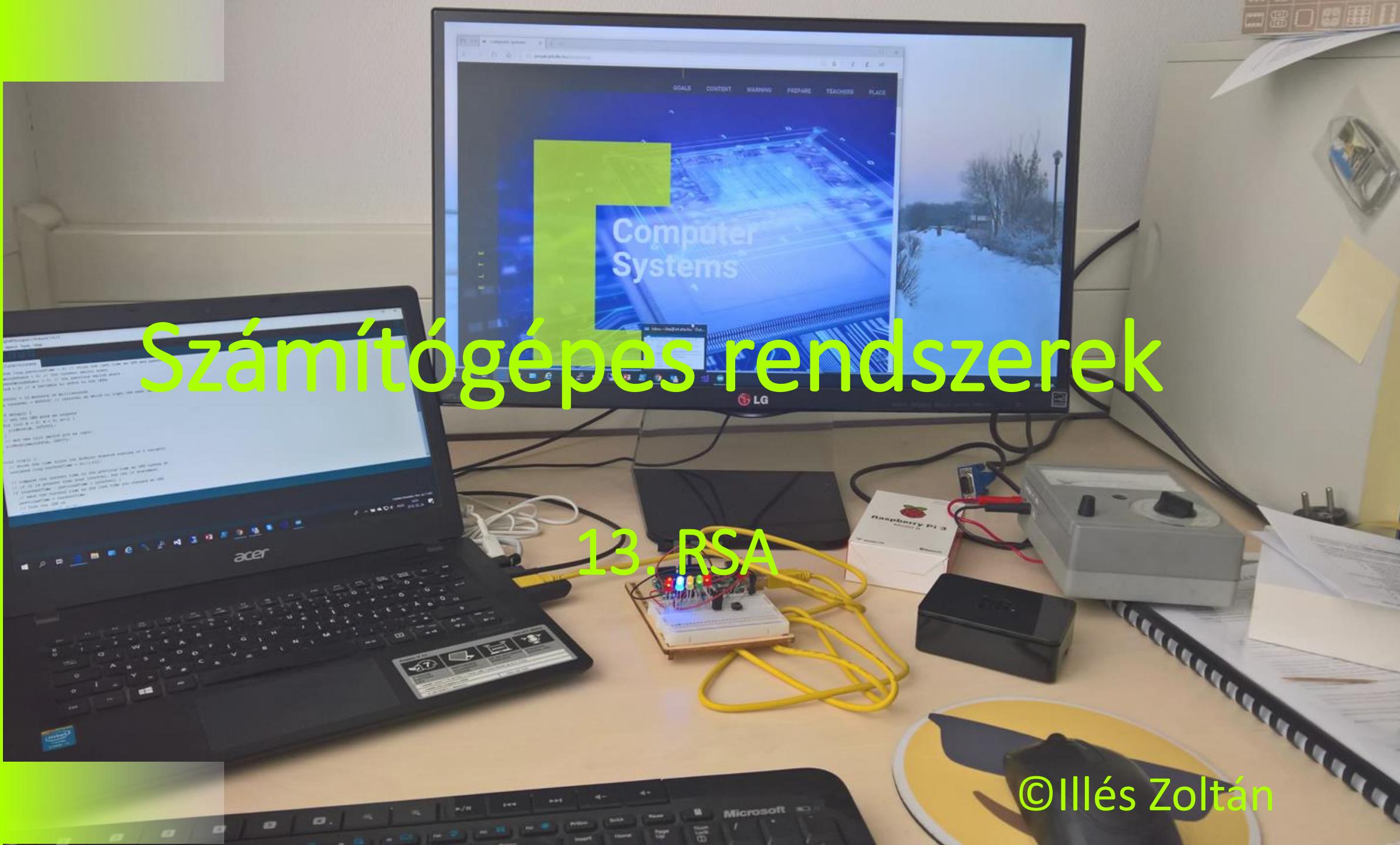


Hálózatprobléma

- USB Modem
- Kiegészítő panel
- Wigi, Ethernet







Számítógépes rendszerek

13. RSA

© Illés Zoltán

Visszatekintés

- Számítógépek, számábrázolás, kódolás,felépítés, fájlrendszerek
- Alapvető parancsok, folyamatok előtérben, háttérben
- I/O átirányítás, szűrők,reguláris kifejezések
- Változó, parancs behelyettesítés,aritmetikai, logikai kifejezések
- Script vezérlési szerkezetek,Sed,AWK
- Hálózatok jellemzői
- Powershell

Mi jön ma?

- Kódolás – Titkosítás
- Szimmetrikus – Aszimmetrikus titkosítás
- Terminál kapcsolódás
 - Windows Terminál – SSH
 - PUTTY - SSH
- RSA – Egyszerűen

Kódolás - Titkosítás

- Fontos: A számítógép (jelenleg) csak számokat tárol a memóriában!
- Ahhoz, hogy szöveget kapjunk, kódtáblára van szükség. PL. ASCII
 - $41h(65) \rightarrow A$, $4Ch(76) \rightarrow L$, $4Dh(77) \rightarrow M$, $41h(65) \rightarrow A$
- Amíg általános kódtáblákat használunk, egyszerű szöveges ábrázolásról van szó.
- Ha módosított, speciális kódtábláról, akkor titkosításról beszélünk.

Alap ASCII kódtábla

ASCII Code Chart																
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	!	"	#	\$	%	&	'	()	*	+	,	-	.	/	
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	-
6	~	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

- Módosítható ez „saját használatra”?
 - Miért ne?- Ekkor nem kódolás - Titkosítás

Szimmetrikus – Aszimmetrikus titkosítás

- Hogyan hozhatunk létre speciális kódtáblákat?- Sokféle módszer ismert
- Legegyszerűbb változat: módosított karakter tábla
- Kicsit rafináltabb, pl. megadott könyv alapján, a küldött számsorozat megadja a könyv karaktereit, ami lesz a valódi üzenet.
- Ma általánosan elterjedt, egy kulcs alapján valamelyen matematikai módszert használva
- Legegyszerűbb: XOR
- Mind szimmetrikus –egy kulcs
- Aszimmetrikus – 2 kulcs.

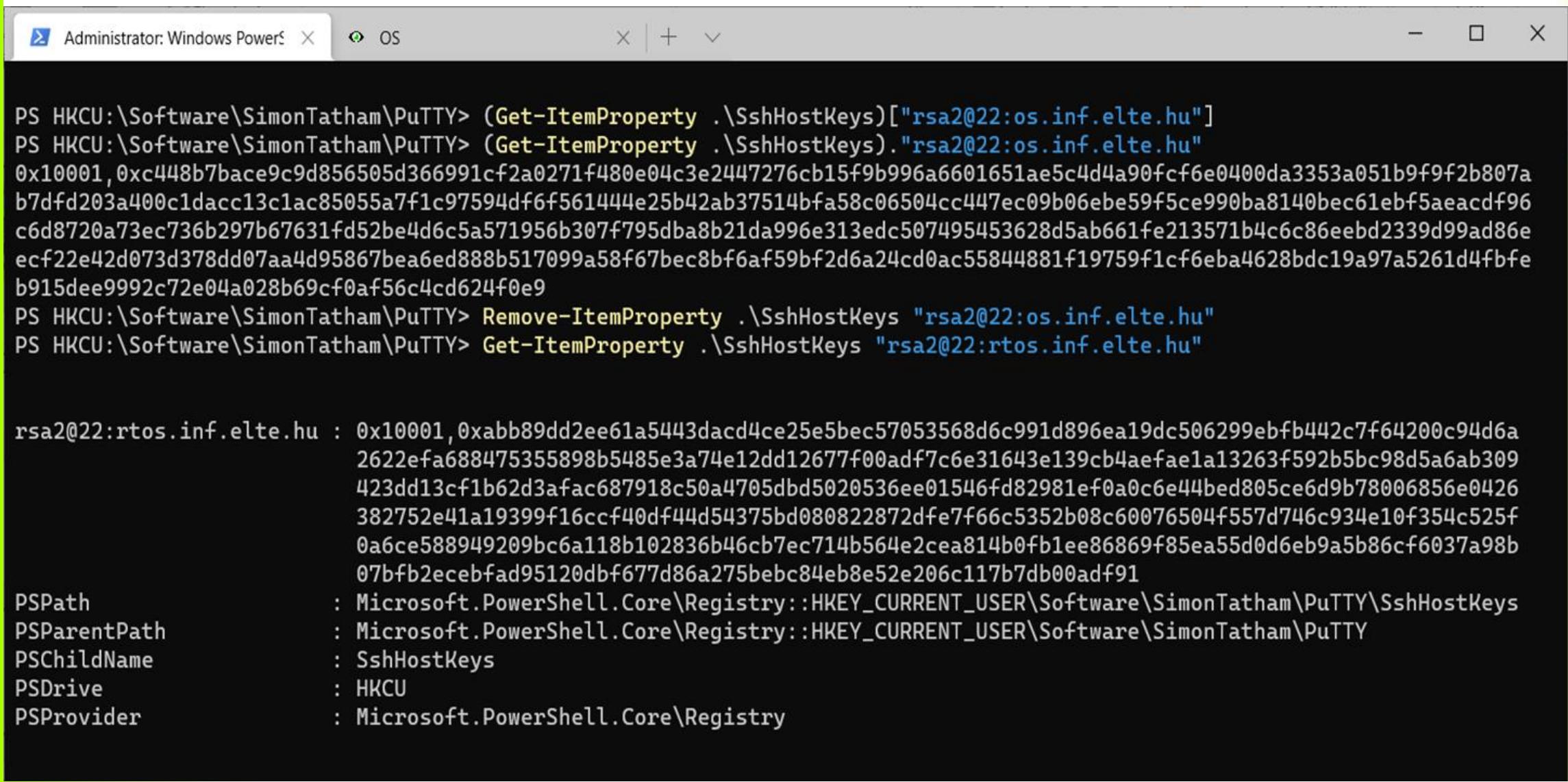
Encryption cipher selection policy:

AES (SSH-2 only)
ChaCha20 (SSH-2 only)
3DES
-- warn below here --
DES
Blowfish

Terminál kapcsolat – Mit használ?

- Windows Terminal
 - Users\.ssh\known_hosts tárolja az ismert host kulcsokat
 - PUTTY: Registry: HKCU\Software\SimonTatham\PuTTY
 - Módosítás
 - Regedit vagy
 - PS

Registry HKCU kulcs



A screenshot of a Windows PowerShell window titled "Administrator: Windows PowerShell". The window shows the following command sequence:

```
PS HKCU:\Software\SimonTatham\PuTTY> (Get-ItemProperty .\SshHostKeys)[\"rsa2@22:os.inf.elte.hu\"]  
PS HKCU:\Software\SimonTatham\PuTTY> (Get-ItemProperty .\SshHostKeys).\"rsa2@22:os.inf.elte.hu\"  
0x10001,0xc448b7bace9c9d856505d366991cf2a0271f480e04c3e2447276cb15f9b996a6601651ae5c4d4a90fcf6e0400da3353a051b9f9f2b807a  
b7dfd203a400c1dacc13c1ac85055a7f1c97594df6f561444e25b42ab37514bfa58c06504cc447ec09b06ebe59f5ce990ba8140bec61ebf5aeacf96  
c6d8720a73ec736b297b67631fd52be4d6c5a571956b307f795dba8b21da996e313edc507495453628d5ab661fe213571b4c6c86eebd2339d99ad86e  
ecf22e42d073d378dd07aa4d95867bea6ed888b517099a58f67bec8bf6af59bf2d6a24cd0ac55844881f19759f1cf6eba4628bdc19a97a5261d4fbfe  
b915dee9992c72e04a028b69cf0af56c4cd624f0e9  
PS HKCU:\Software\SimonTatham\PuTTY> Remove-ItemProperty .\SshHostKeys "rsa2@22:os.inf.elte.hu"  
PS HKCU:\Software\SimonTatham\PuTTY> Get-ItemProperty .\SshHostKeys "rsa2@22:rtos.inf.elte.hu"  
  
rsa2@22:rtos.inf.elte.hu : 0x10001,0xabbb89dd2ee61a5443dacd4ce25e5bec57053568d6c991d896ea19dc506299ebfb442c7f64200c94d6a  
2622efa688475355898b5485e3a74e12dd12677f00adf7c6e31643e139cb4aefae1a13263f592b5bc98d5a6ab309  
423dd13cf1b62d3afac687918c50a4705dbd5020536ee01546fd82981ef0a0c6e44bed805ce6d9b78006856e0426  
382752e41a19399f16ccf40df44d54375bd080822872dfe7f66c5352b08c60076504f557d746c934e10f354c525f  
0a6ce588949209bc6a118b102836b46cb7ec714b564e2cea814b0fb1ee86869f85ea55d0d6eb9a5b86cf6037a98b  
07bfb2ecebfad95120dbf677d86a275bebc84eb8e52e206c117b7db00adf91  
  
PSPPath : Microsoft.PowerShell.Core\Registry::HKEY_CURRENT_USER\Software\SimonTatham\PuTTY\SshHostKeys  
PSParentPath : Microsoft.PowerShell.Core\Registry::HKEY_CURRENT_USER\Software\SimonTatham\PuTTY  
PSChildName : SshHostKeys  
PSDrive : HKCU  
PSPProvider : Microsoft.PowerShell.Core\Registry
```

Mitől vagyunk biztonságban?

RSA algoritmus

avagy: a minden napok során sem
csak az összeadást használjuk a
matematika órán tanultakból

(lásd: boltban, pénztárnál)

Oszthatóság

Mi a közös bennük?

4; 7; 10; 13; 16; 19; 22

14; 5; 17; 8; 11; 20; 23

3; 66; 9; 12; 135; 18; 6

Oszthatóság definíciója:

$$a, b \in N \quad a|b \iff \exists x \in N \exists \quad a \cdot x = b$$

a, b természetes számok esetén azt mondjuk, hogy az **a** osztója a **b**-nek (vagy **b** osztható **a**-val), ha található olyan **x** természetes szám, hogy **a · x = b**

3-mal osztva 1 a maradék

3-mal osztva 2 a maradék

3-mal osztva 0 a maradék

maradékosztályok

Maradékosztályok

Vizsgáljuk meg a maradékosztályokat!

a) 5; 8; 11; 14; 17; 20; 23

$$8 - 5 = 3$$

$$7 - 4 = 3$$

$$11 - 8 = 3$$

$$10 - 4 = 6$$

b) 4; 7; 10; 13; 16; 19; 22

$$17 - 8 = 9$$

$$19 - 7 = 12$$

c) 3; 66; 9; 12; 135; 18; 6

$$6 - 3 = 3$$

Tehát például:

$$12 - 6 = 6$$

$$19 \equiv 7 \pmod{3}$$

$$18 - 6 = 12$$

Vagyis ugyan abban
a maradékosztályban
vannak mod 3

Def.: Ha az m ($\neq 0$) egész osztja az $a - b$ különbséget, akkor azt mondjuk,
hogy az a szám **kongruens** b -vel modulo m (\rightarrow ugyan abban a
maradékosztályban vannak mod m)

Jelölés: $a \equiv b \pmod{m}$

Kongruencia tulajdonságai

1. $a \equiv b \pmod{m} \Rightarrow b \equiv a \pmod{m}$ és $a - b \equiv 0 \pmod{m}$
2. $a \equiv b \pmod{m}$ és $b \equiv c \pmod{m} \Rightarrow a \equiv c \pmod{m}$
3. $a \equiv b \pmod{m}$ és $c \equiv d \pmod{m} \Rightarrow a \cdot c \equiv b \cdot d \pmod{m}$
4. $a \equiv b \pmod{m}$ és $c \equiv d \pmod{m} \Rightarrow a \cdot c \equiv b \cdot d \pmod{m}$
5. $a \equiv b \pmod{m}$ és $d|m$ és $d > 0 \Rightarrow a \equiv b \pmod{d}$
6. f egész együtthatós polinom és $a \equiv b \pmod{m} \Rightarrow f(a) \equiv f(b) \pmod{m}$
7. ha $(a; m) = 1$ akkor: $a \cdot x \equiv a \cdot y \pmod{m} \Leftrightarrow x \equiv y \pmod{m}$
m-hez relatív prímmel szorozhatok, oszthatok

4.tul. $\Rightarrow a \equiv b \pmod{m}$ és $a \equiv b \pmod{m} \Rightarrow a^2 \equiv b^2 \pmod{m}$
 $\Rightarrow \dots \Rightarrow a^n \equiv b^n \pmod{m}$

Hatványozhatom a kongruenciát!



Később még szükség lehet rá ☺

Állítás:

legyen $(a;m)=1$ és $(x;m)=1$ és $(y;m)=1$

továbbá $x \not\equiv y \pmod{m}$ $\Rightarrow a \cdot x \not\equiv a \cdot y \pmod{m}$

Ezek nem feltétlenül
vannak ugyanabban
a maradékosztállyban

Bizonyítás:

$x \not\equiv y \pmod{m}$ jelenti: $m \nmid (x-y)$

ekkor: $m \nmid a \cdot (x-y) = a \cdot x - a \cdot y$, tehát

$a \cdot x \not\equiv a \cdot y \pmod{m}$



Ha két szám nem kongruens egymással, akkor m-hez
relatív prímmel szorozva őket, továbbra sem lesznek
egymással kongruensek!

Maradékrendszer



Nézzük a következő számokat: 3; 4; 5



Egy másik számhármas: 33; 16; 26

Ezek a számok teljes maradékrendszeret alkotnak mod 3

Def: $x_1; x_2; \dots; x_m$ teljes maradékrendszer mod m , ha tetszőleges y egész számhoz pontosan egy olyan x_j található, amelyre $y \equiv x_j \pmod{m}$

φ függvény

Euler-féle φ függvény: $\varphi(m)$ az m -nél nem nagyobb, m -hez relatív prím pozitív egészek száma

Például: $m = 24$ hozzá relatív prímek: 1; 5; 7; 11; 13; 17; 19; 23

$$\varphi(24) = 8$$

$$m = 7$$

hozzá relatív prímek: 1; 2; 3; 4; 5; 6

$$\varphi(7) = 6$$

Mennyi $\varphi(11)$, $\varphi(13)$, $\varphi(23)$? (10, 12, 22)

redukált maradékrendszer
mod 24

Fontos! Ha p prím, akkor $\varphi(p) = p - 1$

Redukált maradékrendszer: a teljes maradékrendszerből csak azokat az elemeket hagyjuk meg, amelyek m -hez relatív prímek

Egy kis feladat

Legyen $m = p_1 \cdot p_2$ $\varphi(m) = ?$

$m = 15 = 3 \cdot 5$ hozzá relatív prímek: 1; 2; 4; 7; 8; 11; 13; 14

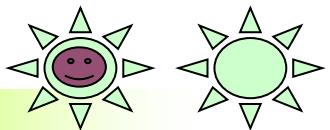
$$\varphi(15) = 8$$

$\varphi(3) = 2$; $\varphi(5) = 4$ és $2 \cdot 4 = 8$ Véletlen?

Állítás: $\varphi(m) = (p_1 - 1) \cdot (p_2 - 1)$

$$\varphi(m) = (p_1 - 1) \cdot (p_2 - 1) = p_1 \cdot p_2 - p_1 - p_2 + 1$$

$p_1 \cdot p_2$ db számból p_1 db p_2 többszörösét és p_2 db p_1 többszörösét vonjuk ki, de a $p_1 \cdot p_2$ -t így kétszer is kivontuk.



Euler tétele

Tétel: ha $(a; m) = 1$, akkor $a^{\varphi(m)} \equiv 1 \pmod{m}$

Biz.:

$r_1; r_2; \dots; r_{\varphi(m)}$ redukált maradék rendszer mod m

$a \cdot r_1; a \cdot r_2; \dots; a \cdot r_{\varphi(m)}$ is redukált maradék rendszer mod m

(ugyan annyi elem van itt is, ott is, ill. itt van szükség a tételre!)

$$r_j \equiv a \cdot r_k \pmod{m}$$

$$r_1 \cdot r_2 \cdot \dots \cdot r_{\varphi(m)} \equiv a \cdot r_1 \cdot a \cdot r_2 \cdot \dots \cdot a \cdot r_{\varphi(m)} \pmod{m} \quad (4. tul.)$$

$$1 \equiv a^{\varphi(m)} \pmod{m} \quad (7. tul. alapján, mert (r_i; m) = 1)$$

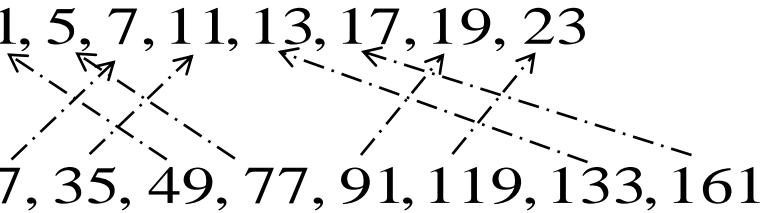
$$m = 24$$

$$\varphi(24) = 8$$

$$(24; 7) = 1$$

$$\Rightarrow 1, 5, 7, 11, 13, 17, 19, 23$$

$$\Rightarrow 7, 35, 49, 77, 91, 119, 133, 161$$



Fermat tétele

$$\begin{aligned} p \text{ prím és } (a; p) = 1 &\Rightarrow a^{p-1} \equiv 1 \pmod{p} \\ &\Rightarrow a^p \equiv a \pmod{p} \end{aligned}$$

Biz.: Az előző téTEL következménye

Eszerint ha $a < p$, akkor ha a -t p -edik hatványra emeljük, majd a kapott értéket elosztjuk p -vel, az osztási maradékként éppen a -t kapjuk vissza.

Egyesítsük a tételeket!

$$m := p_1 \cdot p_2 \quad \text{akkor} \quad \varphi(m) = (p_1 - 1) \cdot (p_2 - 1)$$

$$(a; m) = 1$$

$$a^{\varphi(m)} \equiv 1 \Rightarrow a^{(p_1-1)(p_2-1)} \equiv 1 \pmod{m}$$

$$a^{\varphi(m)+1} \equiv a \Rightarrow a^{(p_1-1)(p_2-1)+1} \equiv a \pmod{m}$$

$b \in \mathbb{Z}$ esetén:

$$(a^{\varphi(m)})^b \equiv 1^b = 1 \Rightarrow a^{b \cdot (p_1-1)(p_2-1)} \equiv 1 \pmod{m}$$

$$a^{b \cdot \varphi(m)+1} \equiv a \Rightarrow a^{b \cdot (p_1-1)(p_2-1)+1} \equiv a \pmod{m}$$

4.tul. $\Rightarrow a \equiv b \pmod{m} \Rightarrow a^n \equiv b^n \pmod{m}$

Folytatás

$$m := p_1 \cdot p_2 \quad \text{akkor} \quad \varphi(m) = (p_1 - 1) \cdot (p_2 - 1)$$

$$(a; m) = 1, \quad b \in \mathbb{Z} \quad \text{esetén} \quad a^{b \cdot \varphi(m)+1} \equiv a \pmod{m}$$

$$b \cdot \varphi(m) + 1 := \alpha \cdot \beta, \quad \text{ahol} \quad \alpha, \beta \in \mathbb{N}$$

$$a^{\alpha \cdot \beta} = (a^\alpha)^\beta \equiv a \pmod{m}$$

2. tulajdonság alapján a^α helyett bármilyen vele mod m kongruens számot is írhatok a számolás során

(α, m), (β, m) lesznek a kulcsok, a pedig a titkosítandó szám (relatív prím m-hez)!

Alkalmazzuk az eddigieket!

$$m := 3 \cdot 5 = 15$$

$$(a; m) = 1 \quad \text{és} \quad a < m \quad \Rightarrow \quad a \in \{1; 2; 4; 7; 8; 11; 13; 14\}$$

$$\varphi(m) = 8$$

$$\varphi(m) + 1 = 9 = 3 \cdot 3 \qquad \text{Nem jó, ugyan az lenne a titkos és a nyilvános kulcs}$$

$$2 \cdot \varphi(m) + 1 = 17 \qquad \text{Nem jó, mert prím}$$

$$3 \cdot \varphi(m) + 1 = 25 = 5 \cdot 5 \qquad \text{EZ sem jó!}$$

$$4 \cdot \varphi(m) + 1 = 33 = 3 \cdot 11 \qquad \text{Végre!}$$

Titkosítunk!

Legyen a nyilvános kulcs: (11;15)

Ekkor a titkos kulcs: (3;15)

Ne felejtsük el: $a \in \{1; 2; 4; 7; 8; 11; 13; 14\}$

Például a titkosítandó számsor: 2 4 8 7

Kódolás

A titkosítandó szám: 2 4 8 7

A nyilvános kulcs: (11;15)

$$2^{11} = 2048 = 136 \cdot 15 + 8$$

$$4^{11} = 4194\,304 = 279\,620 \cdot 15 + 4$$

$$8^{11} = 8\,589\,934\,592 = 572\,662\,306 \cdot 15 + 2$$

$$7^{11} = 1977\,326\,743 = 131\,821\,782 \cdot 15 + 13$$

Az eredmény: 8 4 2 13

Dekódolás:

A titkos üzenet: 8 4 2 13

A titkos kulcs: (3;15)

$$8^3 = 512 = 34 \cdot 15 + 2$$

$$4^3 = 64 = 4 \cdot 15 + 4$$

$$2^3 = 8 = 0 \cdot 15 + 8$$

$$13^3 = 2197 = 146 \cdot 15 + 7$$

Tehát az eredeti üzenetet visszakaptuk: 2 4 8 7

Kódtáblázat:

1	A
2	E
3	É
4	G
5	K
6	R
7	S
8	T
9	Z
10	?

Privát kulcs: (7;187) (Ezzel dekódolunk!)

RSA kódolt üzenet:

83; 162; 83; 46; 36; 162; 83; 83; 175

162; 64; 181; 46; 36; 46; 181; 64; 162; 83;
162; 180; 150; 162

Mi lehet a publikus kulcs?(Ezzel kódoltunk!)

- Kis segítség
 - $(7,187)$
 - $(23,187)$

$$187 = 17 \cdot 11$$

$$\varphi(187) + 1 = 16 \cdot 10 + 1 = 161$$

$$161 = 7 \cdot 23$$

$$2 \cdot \varphi(187) + 1 = 2 \cdot 16 \cdot 10 + 1 = 321$$

$$321 = 3 \cdot 107$$

$$3 \cdot \varphi(187) + 1 = 3 \cdot 16 \cdot 10 + 1 = 481$$

$$481 = 13 \cdot 37$$

$$4 \cdot \varphi(187) + 1 = 4 \cdot 16 \cdot 10 + 1 = 641$$

$641 = \text{prím}$

Feltörhető? – Kis számok esetén...igen

$$m = 15 \quad \alpha = 11 \quad \beta = ?$$

$$15 = 3 \cdot 5$$

$$\varphi(15) = 2 \cdot 4 = 8$$

$$b \cdot 8 + 1 = 11 \cdot \beta$$

Rövid próbálgatás után:

$$b = 4 \quad \beta = 3$$

$$m = 527 \quad \alpha = 13 \quad \beta = ?$$

$$527 = 17 \cdot 31$$

$$\varphi(527) = 16 \cdot 30 = 480$$

$$b \cdot 480 + 1 = 13 \cdot \beta$$

Rövid próbálgatás után:

$$b = 1 \quad \beta = 37$$

Feltörhető?

- $M = P_1 * P_2 - P_1, P_2$ 1024, 2048 bites

$$m := p_1 \cdot p_2 \quad \text{akkor} \quad \varphi(m) = (p_1 - 1) \cdot (p_2 - 1)$$

$$(a; m) = 1, \quad b \in \mathbb{Z} \quad \text{esetén} \quad a^{b \cdot \varphi(m) + 1} \equiv a \pmod{m}$$

$$b \cdot \varphi(m) + 1 := \alpha \cdot \beta, \quad \text{ahol} \quad \alpha, \beta \in \mathbb{N}$$

$$a^{\alpha \cdot \beta} = (a^\alpha)^\beta \equiv a \pmod{m}$$

(α, m), (β, m) lesznek a kulcsok, a pedig a titkosítandó szám (relatív prím m -hez)!

- A módszer egyszerű, de a számolásigény óriási, évek.

Mit jelent az elnevezés? - RSA



Ronald L. Rivest



Adi
Shamir



Leonard
Adleman



Köszönöm a figyelmet!

© Illés Zoltán

SzámRend – kérdezz-felelek

1. **Milyen jellemző paraméterei vannak egy mai asztali számítógépnek?**
Processzor, memória, háttértár, alaplap tulajdonságai, hűtés típusa stb.
2. **Mi a cache szerepe a mikroprocesszorban?**
Cache: gyorsítótár. Különböző órajelű egységek sebessékgülönbségeinek kompenzációja.
Pl.: gyorsabb órajelű egység küld adatot lassabbra: az adatok átmenetileg a cache-ben tárolódnak, így a gyorsabb egységek nem kell a lassabbra várni.
3. **Milyen a kettes komplexumos számábrázolás?**
Fixpontos számábrázolási módszer. A pozitív számokat hagyományos módon alakítjuk át kettes számrendszerbe, míg a negatívokat a konverzió után bitenként negáljuk, majd hozzáadunk 1-et.
4. **Mit tud az UTF-8 kódolásról? Mire jó?**
[8-bit Unicode Transformation Format] Változó hosszúságú UNICODE karakterkódolási eljárás. Bármilyen UNICODE karaktert képes reprezentálni, ugyanakkor visszafelé kompatibilis a 7 bites ASCII szabvánnyal.
5. **Soroljon fel legalább 3 memóriatípust!**
 - RAM
 - ROM
 - Flash
 - Cache
6. **Soroljon fel olyan hétköznapi eszközöket, amelyekben „számítógép” van!**
Okostelefon, okosóra, légkondicionálók, autók, házimozi-hifi rendszerek, közlekedési lámpák stb.
7. **Mi a különbség a szerver és egy kliensgép között?**
Szerver: nagy teljesítményű hardware vagy software, ami a következőket biztosít(hat)ja más számítógépek (kliensgépek) számára egy hálózaton keresztül:
 - tárolt adatok elérése
 - szerver által előállított adatok elérése
 - a szerver hardware erőforrásainak kihasználása (pl. nyomtató, processzor)
 - egyéb szolgáltatások eléréseKliens: hálózaton keresztül kapcsolódik a szervergéphez, igénybe veszi annak szolgáltatásait.
8. **Soroljon fel legalább 3 operációs rendszert!**
Windows, Unix, Linux (+ disztrók), MacOS, iOS, Android, DOS stb.
9. **Mi a különbség az SSH és a Telnet kapcsolat között?**
Az SSH kapcsolat biztonságos (titkosított) kommunikációs csatornát épít ki a kliens és a szerver között, titkosított adatátvitellel dolgozik. (A Telnet nem titkosít...)

10. Milyen szolgáltatásai vannak egy mai operációsrendszernek?

Programok futtatása, be-/kivitel, szerkeszthető fájlrendszer, hibakeresés/-kezelés, erőforrás elosztás, védelem, hálózati kapcsolat, kommunikáció, kliens-szerver megoldások, közös és osztott háttértárak, több felhasználói fiók stb.

11. Mit ért Shell alatt? Nevezzen meg legalább kettőt!

Rendszerhéj: felület, amely segítségével a felhasználó kommunikálhat a kernellel (rendszermag). Parancssori: Bash, POSIX shell, PowerShell, Command Prompt. Grafikus: (nem vettük, de pl Windows meg minden, ami nem ~~~ DOS)

12. Mi az alias? Hol találkozott vele?

Shell parancs: egy szó helyettesítése egy másikkal. Többnyire parancsok helyettesítésére. pl. PowerShell

13. Milyen UNIX fájlrendszer jellemzőket tud megemlíteni?

A fájlrendszer kiindulópontja a „root” („/”); a fájlok és könyvtárak rendszere fát alkot; hierarchikus szerkezetű; kétféle bejegyzés: könyvtár, fájl; eszközök is rendelkeznek fájlnévvel; linkelés: fájlra (vagy annak tartalmára) mutató bejegyzés

14. Milyen fájlrendszeret ismer?

FAT32, exFAT, NTFS, ext2, ext3, ext4.

15. Milyen fájlnév konvenciókat ismer UNIX-ban?

Kisbetűs, ékezet nélküli elnevezések; elválasztásnak „-” vagy „_”; rejtett fájlok: „.”-tal kezdődnek; különleges karakterek használhatóak, de nem javallottak

16. Milyen fájl jellemzőket ismer UNIX-ban?

A fájl a rendszerben egy strukturálatlan bájt-sorozat (nincs fájlvég-jel). A rendszer nyilvántartja a fájl hosszát, utolsó módosítás időpontját, a fájlrendszer hány pontjáról hivatkozunk erre a fájlra stb.

17. Magyarázza el a UNIX-ban lévő alapvető hozzáférési jogosultsági rendszert!

Minden könyvtárbejegyzés vagy fájl (különböző típus), vagy könyvtár. minden bejegyzés jogosultságai 3 csoportot alkotnak: a tulajdonos/felhasználó jogosultságai (u), a csoport jogosultságai (g), bárki/mindenki jogosultságai (o). minden csoportnak 3 féle jogosultsága lehet: olvasási (r), írási (w), futtatási (x). Módosítás: chmod parancs.

18. Milyen kiegészítő jogokat ismer UNIX-ban?

setuid: a futtható állomány a fájl tulajdonosának jogosultságaival fut.

setgid: a futtható állomány a fájl csoportjának jogosultságaival fut.

sticky bit: a könyvtár fájljai csak a tulajdonos által módosíthatóak.

19. Milyen célt szolgál a UNIX-folyamatok prioritása?

A UNIX rendszer folyamatai felváltva futnak, gyors váltásokkal. A prioritás-rendszer segítségével a rendszer hosszabb futásidőt oszt ki a fontosabb folyamatoknak.

20. Mit tud az idézőjelekről a UNIX rendszerben?

””: stringek deklarálása; különleges karaktereket (pl. változónevek) behelyettesíti.
’’: minden karakter saját magát jelenti; nincs behelyettesítés.
` `: parancs behelyettesítése.

21. Mit jelent az stdin, stdout?

stdin: standard input; alapértelmezett bemenet: billentyűzet
stdout: standard output; alapértelmezett kimenet: monitor
(stderr: standard error output; alapértelmezett hibakimenet: monitor)

22. Hány szűrő kell egy csővezetékhez?

Kettő, vagy több.

23. Mondjon példát arra, hol használhat reguláris kifejezéseket?

Csővezeték: pl grep parancs, sed parancs stb.

24. Mi az az ASCII kódtábla?

8 (eredetileg 7) bites karakterkódolási tábla, az angol ábécé kis- és nagybetűit, számokat, írásjeleket, és néhány vezérlőkaraktert tartalmaz.

25. Mik azok a környezeti változók?

Olyan globális változók, amelyek befolyásolhatják különböző folyamatok futását (pl. PATH, TEMP)

26. Adja meg, hogy UNIX-ban milyen típusú(ak) lehet(nek) a változó tartalma(k)!

UNIX-ban minden változó szöveges típusú.

27. Mit jelent a parancsbehelyettesítés?

Mikor egy parancs kimenetét egy szövegbe szeretnénk visszahelyettesíteni, akkor a szövegen belül, ` -k közé írjuk a parancsot. Futáskor lefut a parancs is, a végeredmény pedig bekerül a szövegbe (változóba stb.).

28. Sorolja fel, hogy milyen műveletek (aritmetikai, logikai) léteznek UNIX shellben!

Alapvetően csak szövegösszefűzés van közvetlenül a shellben. Más (pl. aritmetikai, logikai) műveletekhez parancsokra van szükség (pl. expr, bc, test).

29. Melyik shell utasításnak van befejezési eredménye?

Mindegyik shell utasításnak van befejezési eredménye: 0 = sikeresen lefutott, 1 = sikertelen.

30. Hogyan implementálják a logikai értékeket a UNIX shellben?

”Test” parancs segítségével (vagy ”[]”): lefut-e az adott parancs. Ezért 0 = igaz, 1 = hamis.

31. Hogyan készíthetünk összetett logikai kifejezést UNIX shell scriptben?

”Test” parancs –a vagy –o operátoraival egymás után fűzünk több kiértékelést. A precedenciát módosíthatjuk a ”\(|” és ”\)|” karakterekkel.

32. Lehet-e paramétereket kezelő függvényeket definiálni UNIX alatt?

Igen, C-szerű függvényeket lehet definiálni, amely a paramétereket shell script-hez hasonló módon kezeli; visszatérését a ”return” kulcsszóval lehet beállítani.

33. Tudja-e (és ha igen, hogyan) futtatni a végrehajtási jogosultság nélküli shell scriptet?

Igen, a script tartalmát át kell adni az adott shellnek, és azt futtatni. (sh -v, bár nem tudom, ez mit csinál)

34. Mi az IFS?

[Internal Field Separator] Alapértelmezett elválasztójel a shellben (alapesetben szóköz/tabulátor/soremelés); értékadással megváltoztatható (lehet több karakter).

35. Milyen feladatokat tud elvégezni a SED-del?

Komplex behelyettesítések, cserék (első, utolsó, összes elem cseréje; csere regex alapján stb.)

36. Írja le általánosan egy SED parancs szintakszisát!

sed [paraméter] [cím] s/minta/új_minta/[jelző]
([] = opcionális)

37. Mi a különbség a SED használatában a " és a ' idézőjel használata között?

": változók, különleges karakterek behelyettesítésre kerülnek.

': nem történik behelyettesítés.

38. Jellemesse az AWK lehetőségeit!

C nyelvi lehetőségeket ad; tipikus szűrő, pótolja a shell/sed hiányosságait; soronkénti szövegszerkesztést tesz lehetővé.

39. Adj meg, hogy milyen parancsblokkok találhatók AWK-ban!

3 parancsblokk:

Első sor előtti inicializálás: BEGIN { }

Soronkénti feldolgozás: { }

Utolsó utáni inicializálás: END { }

40. Használható-e az AWK aritmetikai feladatok megoldására?

Igen, C-ből ismert aritmetikai jelek működnek AWK-ban, ill. van numerikus függvénykészlete.

41. Mi az MBR és mi a feladata?

Master Boot Record (~Fő indítórekord): A merevlemez egyik particionált része, ahonnan az operációs rendszer betöltődik. A számítógép bekapcsolása után az alaplap itt keresi, és ide adja át a vezérlést.

42. Írja le a UNIX-LINUX boot folyamatot!

6 lépésből áll:

1. Bios

- a. Az alapvető be- és kimeneti folyamatokat biztosítja
- b. Megkeresi, betölti és lefuttatja a boot betöltő programot (MBR)

2. MBR

- a. Lásd: 41. kérdés

3. GRUB

- a. Grand Unified Bootloader ~ A főbb bootfolyamatokat ez indítja el
- b. Kialakítja a kezdeti RAM-ot
- c. Futtatja a Kernel-t

4. Kernel ~ rendszermag
 - a. Kiosztja a feladatoknak, hogy a hardvereket mennyire használhatják.
 - b. ~ multiplexálás
 - c. Átadja a vezérlést az init-nek
 5. Init
 - a. /etc/inittab néven van elmentve
 - b. Eldönti a Linux futásának a szintjét
 - c. 0: megáll, 6: reboot (nem érdemes)
 6. Futásszintű programok
 - a. Átadja a vezérlést a felhasználónak a további műveletekhez
- 43. Írjon le legalább egy UNIX-LINUX management lehetőséget!**
File manager: Felhasználói felületet biztosít a felhasználónak a fájl – és mappakezeléshez a hierarchikus rendszerben és az alapvető fájlkezelő műveleteket is elvégzi (másolás, átnevezés std.)
- 44. Milyen hálózati kapcsolódási lehetőségeket ismer?**
Fix: kábelalapú, Ideiglenes: WiFi, 3g, 4g, cellás, műholdas
- 45. Mit ért csomagkapcsolt hálózat alatt?**
Nincs előre kiépített út a kommunikációra, továbbá a küldeni kívánt adatot csomagokra bontják, és ezek külön-külön kerülnek elküldésre.
- 46. Mit ír le az OSI modell?**
[Open Systems Interconnection Reference Model] A számítógépek kommunikációjához szükséges hálózati protokollt határozza meg.
- 47. Nevezzen meg hálózati topológiákat!**
Centralizált: Egy központi felületen (szervergépen, buszon) történik a kommunikáció.
(csillag, gyűrű, teljes, fa)
Decentralizált: Nincsen meg az a központi felület, amely a kommunikációt gyorsítaná, ezért a kiépítése vagy nem túl hatékony, vagy költséges (Sín, busz)
Hibrid: előző kettő részenkénti alkalmazása (csillag-sín)
- 48. Mi a feladata egy switch-nek?**
A hálózat felépítésében a kapcsolás és útválasztás feladatát, valamint a portok összekapcsolását a switch, mint hálózati eszköz valósítja meg.
- 49. Mi a feladata a routernek?**
Hálózatok összekapcsolása és az ezeken történő adatfolyamatok irányítása, lebonyolítása.
- 50. Hogyan jellemzné az IPv4 címeket?**
IP címek 32 bites szerinti felírása, bájtonként felírva egy 0 és 255 közti egész számmal, ponttal elválasztva.
- 51. Hol találkozik a DNS-sel az informatikában?**
[Domain Name System] Számítógépes hálózatoknál, pl. internet
- 52. Mi az a DHCP?**
[Dynamic Host Configuration Protocol] Dinamikus állomáskonfigurációs protokoll, ez osztja ki a lokális IP-t, hogy ne kelljen manuálisan konfigurálni.

53. Milyen szerver elérési módozatokat ismer?

SSH, TELNET, HTTP, FTP.

54. Mire szolgál a HTTP protokoll?

Állományok feltöltése, letöltése, főként HTML dokumentumokat.

55. Mi történik, ha a public_html könyvtárban nincs index.html fájl?

Kilistázza a mappát és onnan lehet letölteni a tartalmát, vagy ha az privát, akkor hibát jelez a böngésző.

56. Hogyan lehet jelszóval védeni egy weben lévő könyvtárat?

.htaccess és .htpasswd fájllal.

57. Mit ért virtuális host alatt?

Egy webcímre más néven hivatkozunk.

58. Mit jelent az SSI vagy CGI jog a webszerverekben?

A webszerver dinamikus interfészeket (SSI, CGI) futtathat, az azokhoz szükséges paramétereiket (pl. stdio, környezeti változók) elérheti.

59. Milyen Windows script lehetőségeket ismer? Van egyáltalán?

Igen: Batch, WSH [Windows Scripting Host], PowerShell

60. Mi biztosítja PowerShell-ben az „autoexec.bat” szerepét?

gpedit.msc (ötletem sincs, hogy kéne értelmezni a kérdést, ennyit letem a diákok közt)

61. Hogyan biztosítják PowerShell alatt a biztonságos script futtatását?

ExecutionPolicy segítségével, amely alapértelmezetten “Restricted” - azaz nem engedélyez futtatást. Átállítás: Set-ExecutionPolicy –ExecutionPolicy Unrestricted. Lehetséges policy értékek: Allsigned, Remotesigned, Bypass.

62. Milyen a PowerShell parancsok felépítése?

Ige-[Modul]Főnév alakúak a parancsok. (pl: Get-Help)

Ige: minden műveletet hajt végre (pl: Get, Set, New, Remove)

Modul: megmondja milyen modulban található a parancs. Nem minden parancs tartalmaz modulnevet. Nem mindenhol szükséges.

Főnév: Milyen adaton vagy dolgon hajt végre valamit a parancs.

63. Soroljon fel PowerShell-ben legalább két változóláthatósági formát!

- global: mindenhol látható az egész PS-ben
- local: függvény vagy szűrő hatókörében lesz látható. Miután a függvény befejezte a futást a benne lévő változók elvesznek.
- script: a teljes scriptben látható lesz
- private: Olyan lokális változó amihez a „gyerek” környezetek nem férnek hozzá.

Egy környezetben definiált változókat, a környezetében használhatjuk, az ebből származó függvény, script látja. Azonos nevű esetén a lokálisat látjuk alapból. Változót definiálhatunk a scope-jával együtt: \$[scope:]név

64. Hogyan irányítjuk át PowerShell-ben az output-ot? Lehet?

Új fájl, felülírás: „Hello” > szoveg.txt

Hozzáfűzés: „szia” >> szoveg.txt

Nincs < vagy << fajta átirányítás!

65. Hol és mire használható a dot sourcing?

Függvény, változó szintjének a módosítása. Mivel függvényen belül is definiálható függvény és az nem hívható közvetlenül.

pl: ..\UtilityFunctions.ps1 futtatás után az UtilityFunctions script függvényei és változói használhatóvá válnak a jelenlegi scopeban is.

66. Mit jelent a PowetShell függvények nevesített paraméterezi lehetősége?

Van lehetőségünk elnevezni egy script paramétereit, és így nem az \$args változóba kerülnek.

```
param( $x, $y )
"A '$x={0}" -f $x
"A '$y={0}" -f $y
```

67. Mi a különbség a mikroprocesszor és a mikrokontroller között?

A mikrokontroller átalában egyetlen lapka ami tartalmaz egy processzor magot, adatmemóriát, programot és programozható ki/bemente perifériákat. Általában valamilyen konkrét feladatra optimalizált cél-számítógép.

A mikroprocesszor (CPU), néha kiegészítve memória vezérlővel. Bináris jeleket fogad és dolgoz fel. Más alkatrészekkel együtt szokták használni. Általános felhasználású.

68. Mi a „Harvard architektúra” legfontosabb jellemzője?

Egy számítógép-felépítési elv, amelyben a programkód és az adatok külön, fizikailag elkülönített útvonalakon közlekednek a processzor felé.

69. Milyen operációs rendszerben lehet 128 bites egész számot definiálni?

64 bites operációs rendszerben.

70. Mire használható a lebegőpontos számábrázolás?

Lehetővé teszi a valós számok tárolását és kezelését, véges tárhely esetében.

71. Mit jelent az aszimmetrikus kódolás?

Az aszimmetrikus kódolás azt jelenti, hogy az adatok kódolásához és dekódolásához két külön kulcsra van szükség egy helyett. Ez a két kulcs a publikus és a privát kulcs: a publikussal titkosítunk, és a priváttal lehet dekódolni (ehhez szükséges a publikus is).

72. Hány számot takar az RSA publikus vagy privát kulcsa?

Mind a kettő 2 számot takar (egy prímszámot – hatványkitevő a képletben, ez különbözik kulcsunként – és egy másikat, amire nézve azonos maradékosztályba kerül a kódolt adat – ez azonos a privát és publikus kulcsnál)

73. Mit jelent a bináris FTP lehetősége?

Az üzenetküldő minden fájlt bájtonként küld el a vevőnek, a vevő a bájtfolyamot tárolja.

74. Mit jelent a szöveges FTP? Létezik egyáltalán?

Igen, szöveges fájlok küldésére alkalmas. Az adatot szükség esetén átalakítják 8 bites ASCII karakterkódra, a vevő fordított módszerrel dekódol.

75. Hogyan irányíthatja át a szabványos bemenetet PowerShellben?

Sehogy: nincs input átirányítás (<, <<) PowerShell-ben.

76. Mi helyettesíti a "here input" funkciót PowerShellben?

Read-Host –Prompt (nem vagyok biztos ebben, a „here input”-ról semmit nem találtam)

77. Mire használható a profile.ps1 állomány? Van a UNIX-ban megfelelője?

Globális PowerShell profil létrehozására, gyakran használt változókat, alias-okat és függvényeket lehet vele definiálni. UNIX megfelelője: \$HOME/.profile

78. Mit értünk PowerShell modulon?

Hasznos függvények, alias-ok, változók definíciójának gyűjtőhelyét.

79. Elég-e a core PowerShell modul a registry módosításához? Miért?

Igen, mivel a Core adatforrásai közé tartozik a Registry is.

80. Hogyan használhatja PowerShellben a parancsbehelyettesítést?

A „\$(parancs)” szintakszissal.

81. Hogyan készíthet ciklust SED scriptben?

Nincs ciklus SED scriptben

82. Jellemzően milyen állományokat talál az /etc könyvtárban?

Konfigurációs állományokat és a rendszer számára fontos adatbázisokat tartalmaz.

83. Mire szolgál a hálózati csomagok TTL adata?

A TTL [Time To Live] adat az adott csomag élettartama. Ha eléri a 0-t, a csomag törlődik. Olyan adatcsomagoknál hasznos, amelyek elkeverednek, nem jutnak el a vevőhöz.

84. Mit mutat meg a "Netmask"?

A LAN (alhálózat) méretét.

85. Mit értünk "nem routolható" IP címen?

Erre az IP-re a router nem tud adatot küldeni, pl. privát hálózatok.

86. Mire szolgál a "gateway"?

Ez az IP egy kivezető út, pl. /sbin/route -n.

87. Mi az "ARPANET" és milyen lehetőségeket teremtett?

[Advanced Research Projects Agency Network] A '60-as évek projektje. Csomagkapcsolt hálózat. Megteremtette: TCP/IP, FTP, Mail, NVP (Hangtovábbítás, kezdetleges).

88. Mi az IPv6? Miért van rá szükségünk?

128 bites (8 db 16 bites szám) címeket használó IP cím. Az egyre több internetre kapcsol készülék miatt az IPv4 modell előreláthatóan nem lesz elég, hogy az összeset megcímelje.

89. Mondjon példát a "setuid" bit hasznosságára!

Egy parancs a tulajdonosának jogosultságaival fussen.

90 Mire jó a "sticky bit"?

UNIX-ban könyvtárak jellemzője: könyvtárban csak saját fájl törölhető.

91 Milyen célt szolgál az ACL használata Unix-Linux rendszerben?

Hogy a hagyományos jogosultságrendszeren túl más jogokat is tudunk rendelkezésre bocsátani

92 Létezik Windows rendszerben az ACL lehetősége?

Igen.

93 Mire használhatjuk a setfacl vagy getfacl parancsokat Linux rendszer alatt?

Kiterjesztett jogosultságrendszerbeli (ACL) jogosultságokat állít be (setfacl) és ír ki (getfacl)

94 minden fájlrendszerben hasonló módon (pl setfacl) lehet ACL jogokat állítani?

Igen

95 Mi az analóg- digitális jelek közti alapvető különbség?

Az digitális jelek kvantáltak, időben és értékben diszkrétek (meghatározott értelkekkel vesznek fel). Az analóg jelek nem kvantáltak, tetszőleges értéket felvehetnek tetszőleges időben.

96 Mi az adat, cím, vezérlő sín feladata?

Címsín: eszközök címzését szolgálja, szélessége 32 vagy 64 bit.

Adatsín: a processzor adatokat küld vagy fogad ezen keresztül. 32 vagy 64 bit.

Vezérlő sín: a processzor vezérlőjeleket küld vagy fogad ezen keresztül. (min. 10-15)

97 Hogyan készít szűrőt UNIX illetve Powershell alatt? Lehet?

Igen, egy parancs után "|"-t írunk, majd ezt szűrőparancsok (grep, cut stb.) követnek.

98 Mi a lényegi különbség a UNIX ls és a PowerShell Get-ChildItem parancsának eredménye között?

ls string értéket ad vissza, Get-ChildItem objektumtömböt

ls mappákon és fájlokon dolgozik, Get-ChildItem képes pl. a registryben is navigálni

99 Unix vagy Windows PS környezetben tud használni reguláris kifejezést?

Unix shellekben a különböző parancsok támogatják, pl grep, sed

PowerShellben a string osztály támogatja a -split, -match és -replace metódusokban

100 Milyen speciális jelentése van annak, ha Unix rendszerben egy fájlnév .-tal kezdődik?

A fájl rejttetnek minősül, ls nem jeleníti alapból, ahogy a fájlkezelők sem.

101 Mikor használhatóak jól a reguláris kifejezésekben létrehozható csoportok és miért?

Amikor összetett keresésben pl. csak azt tudjuk, hogy 1 db betűt/számot/karaktert keresünk, de nem tudjuk, mi az.

x db karakter esetén: [a-z]{x}, ha betű, [a-zA-Z]{x}, ha kis- vagy nagybetű, [0-9]{x}, ha szám stb.

102 Mi a lényegi különbség a UNIX shell és a PowerShell csővezetéken áthaladó adatok között?

A UNIX shell csővezetékén szöveges adatok haladnak át, a PowerShell-én pedig objektumok.

103 Milyen eszköztárral rendelkezünk UNIX és PowerShell szkriptek írásához?

UNIX script: vi, vim, nano
PowerShell: PowerShell ISE

104 Milyen kiterjesztésűnek kell lennie egy PowerShell és egy shell szkriptnek?

PowerShell: .ps, .ps1
shell script: pl. .sh (nem kötelező kiterjesztés)

105 Lehet paramétere egy szűrőnek? Ha igen, adjon meg egy tetszőleges példát, ha nem, miért nem!

Igen, pl. cut -f1 -d";", vagy wc -w/-c stb.

106 Mi a különbség az stdout, stderr csatorna között? Létezik PowerShellben?

stdout: alapértelmezett kimenet; sikeresen lefutott programok ide írnak ki eredményt.
stderr: alapértelmezett hibakimenet; hibaüzenetek ide kerülnek kiírásra.
Igen, létezik PS-ben (Write-Output – Write-Error).

107 Mi a "probléma" az egyes komplexensű számábrázolással?

0-nak van egy pozitív és negatív alakja is: redundáns.

108 Hogy lehet az stdin csatornát átirányítani Powershellben?

PowerShell-ben nincs input átirányítás.

109 Mire jó a SED? Mi a leggyakrabban használt parancsa?

A SED szöveges adatfolyamban komplex behelyettesítésekre, cserékre jó. Leggyakoribb parancsa a 's/minta/új_minta/'.

110 Lehet-e egy SED scriptben shell scriptet hívni? Miért?

Nem lehet, mert a SED csak behelyettesítéseket végez. (az indoklás saját interpretáció)

111 Lehet-e shell scriptből SED scriptet hívni? Miért?

Igen, lehet, mert a shell script tud másik scriptet futtatni, és sed parancsokat is végrehajt.

112 Mi dönti el Unix rendszer alatt, hogy a script fájl milyen script?

He egy shell szkriptek legelső sora a #! (shebang) karakterszorozattal kezdődik, a mögötte álló (teljes útvonalú) értelmező kapja a fájlt futtatásra.

```
#!/bin/sh  
#!/usr/bin/bash  
#!/usr/bin/python3
```

113 Mit értünk az alatt, hogy egy processzor például 10 nanométeres technológiájú?

A nanométer technológiájú nódus egykor a processzor kapuinak, tranzisztorainak méretére utalt.

Manapság marketing nevek a különböző gyártási technológiák között, és a név nem írja le az áramkör komponenseinek tényleges méretét

Kisebb MOSFET méretű processzorok energiahatékonyabbak, és potenciálisan gyorsabbak.

114 Mit jelent, hogy egy processzor 32 vagy 64 bites?

A memória címbusz szélességét írja le.

32 bites processzor 32 bit széles címeket tud lekérdezni, így ~3.5GB memóriát tud megcímzni, míg egy 64 bites ~16EB memóriát.

Minden modern processzor 64 bites, a mobilprocesszorok is.

115 Hogyan tud AWK scriptet készíteni? Tud egyáltalán?

Igen: #!/usr/bin/env awk -f

116 Mit értünk IoT eszközökön, lehetőségeken? Lehet ezeket az eszközöket programozni?

[Internet of Things] Internetkapcsolattal rendelkező mikrokontrollerek építhetők a minden nap használati tárgyakba, így azok távolról vezérelhetők, illetve különböző adatok lekérdezhetők róluk.

117 Mi az AWK program BEGIN és END blokkjának a szerepe?

Mind kettő egyszer fut le: a BEGIN az elején, az END a végén. Emiatt használható arra, hogy pl a BEGIN blokkban kiírjuk a feladat címét, azt elvégezzük, majd az END blokkban kiírjuk az összegzését.

118 Lehet-e karakter kódtáblával adatokat, szöveget titkosítani?

Lehet. Más kódtáblán más karaktert jelöl ugyan az a szám.

119 Hogyan készíthet SED scriptben pl for ciklust?

(:) -al létrehozott címekhez ugrik a vezérlés a b parancsal, feltétel teljesülésénél ugrik a t parancsal. Goto-hoz hasonló.

120 Mi a különbség a PowerShell for ciklusa és Foreach-Object utasítása között?

A ForEach-Object parancs végrehajt egy utasítást az adott objektum összes elemén. Az objektumot meg lehet adni csővezetéken keresztül vagy az InputObject paraméterrel.

Ez a dokumentum az ELTE IK Programtervező
Informatikus hallgatói által készített, **nem hivatalos**
segédanyag a Számítógépes Rendszerek c. tárgyhoz.
Használat saját felelősségre!

Készítők:

Ambrus-Dobai Márton, Bahrami Benedek Attila,
Gortka Bence, Hadházi Dávid, Hamrik Szabin,
Csépán Botond

2019 © ELTE IK