FEBRUAR 2022

Pika

IZOBRAŽEVANJA

MOJ PRVI MICROSOFT CERTIFIKAT

str. 32

RAZVOJ

ČE NISI LJUBITELJ CLIENT-SIDE KODE, JE TUKAJ ZATE BLAZOR .NET CORE PLATFORMA

str. 23

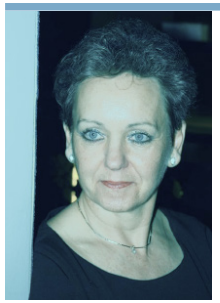
THRIVE CONFERENCE

Pridružite se nam od 24. do 25. maja 2022 V BOHINJSKI BISTRICI

Xnet vaš zanesljiv IT partner

INFRASTRUKTURA | RAZVOJ APLIKACIJ IN SPLETNIH STRANI | OFFICE TEČAJI | SHAREPOINT | ITIL | SQL | BI

www.kompas-xnet.si



Branka Slinkar

Direktorica

Spoštovane in spoštovani,

Odločno smo že zakorakali v leto 2022, zato mi na začetku dovolite, da vam voščim vse dobro. Iz vsega srca vam želim veliko zdravja, sreče, veselja, uspehov in neskončno radostnih trenutkov z vašimi najdražjimi. Naj se vam izpolnijo skrite želje in tudi tisto, o čemer le sanjate. Naj se to nepredvidljivo obdobje pandemije čimprej umiri in zaključi. Naj bo leto 2022 bolj prijazno in radodarno za vse nas.
Hkrati čutim tudi prijetno dolžnost, da se vam iskreno zahvalim za izkazano zaupanje in odlično preteklo sodelovanje.
Verjamem, da ste v naši ekipi prepoznali zaupanja vrednega partnerja, ki stoji za svojimi obljubami in se na nas res lahko zanesete.
Če ravno iščete partnerja za svoje letošnje IT projekte, dajte priložnost našim specialistom, da tudi vam pomagamo do uspešnega zaključka katerega od njih.

»V Skupini SIJ – Slovenski industriji jekla smo v letu 2020 vzpostavili enoten intranetni portal za sedem naših družb, projektu sem se pridružila štiri mesece pred tem. In spoznala Anjo, Robija, Gašperja. Skozi naše tedenske sestanke se je prvi vtis krepil v dolgotrajnejši občutek: vse bo v redu. In je res bilo. Uspešno smo lansirali naš intranet Moj SIJ. ……. (celotna izjava >)
Neprecenljivo je, če delaš z ljudmi, na katere se lahko zaneseš. Hvala celotni ekipi za super sodelovanje.«

Preteklo leto je bilo za nas kar naporno, turbulentno, razgibano a tudi uspešno. Veseli smo, da smo številnim med vami pomagali izkoristiti SATV voucherje, da smo skupaj izvedli lepo število tečajev in zanimivih projektov. Nekatere v živo, večina pa
na daljavo.

Žal pa smo ponovno bili prisiljeni prestaviti Thrive konferenco. Novi datum je 24. – 25. maj, s pred-konferenčnimi delavnicami 23. maja 2022. Prijavite se v času zgodnjih prijav (do konca januarja), ki prinašajo 20 % popust. Preverite zares bogat program in vrhunsko zasedbo mednarodno priznanih strokovnjakov / predavateljev. Poleg Jeffa Teperja
– Microsoft VP, se nam bodo pridružili še 4

Microsoft predstavniki, odgovorni za razvoj in promocijo posameznih produktov. Tako bomo lahko iz prve roke izvedeli tudi kaj o strategijah razvoja in novostih na področju MS tehnologij. Veliko pozornosti bomo posvetili tudi IT varnosti in to z izjemnimi poznavalci. Ne zamudite izjemne priložnosti, da se lahko polna dva dni učite od najboljših. In to za manj kot 500 €. Udeležba na konferenci je vaša najbolj ugodna naložba v nadgradnjo znanja. Prijavite se še danes.

Kot zagotovo že veste, je Microsoft spremenil koncept usposabljanj, ki so sedaj usmerjena v pridobivanje znanj za delovno mesto (»role based«) in ne več na učenje tehnologij. Novi koncept je podprt tudi z ustreznimi tečaji, ki kandidate pripravijo na opravljanje pripadajočih izpitov. Uspešno opravljeni izpiti prinesejo kandidatom različne certifikate, ki izkazujejo njihove strokove kompetence za posamezna področja. Zato naj bo delodajalcem v interesu, da zaposlene motivirajo, naj po usposabljanju pristopijo tudi k opravljanju izpita in tako še formalno potrdijo svoje znanje.

Microsoft je s koncem leta 2021 ukinil številne serverske tečaje, med njimi za Windows Server, System Center, Exchange in SharePoint. Za vse žal ni ustreznega nadomestila. Zato smo si zagotovili vse potrebno, da bomo vsaj v prvem kvartalu letos te tečaje še lahko izvajali, če boste le izrazili dovolj visoko stopnjo zanimanja. Vljudno vas vabimo, da nam sporočite svoj plan in učne potrebe za letos, da bi lahko kar najbolj uskladili vaša pričakovanja in naše zmožnosti. Hvala vam.

Prisrčno vabljeni, da spremljate naše novice in koledar tečajev in se učite od izjemnih strokovnjakov, ki jih imamo v ekipi. Pri nas boste vedno deležni osebnega pristopa in posebne pozornosti, saj nam je dragocen prav vsak posameznik, ki nam izkaže zaupanje. Po tečaju smo na voljo tudi za brezplačno pomoč, imamo priložnostna darila za zveste udeležence, … organiziramo kratka brezplačna šolanja, imate zagotovljene popuste, …in vedno nove ugodnosti za zvestobo.

Naj bo Xnet vaša prva izbira, ko gre za IT rešitve in storitve. Microsoft tehnologije so naša strast.

Čuvajte se in ostanite zdravi!
Branka Slinkar

Katalog SharePoint gradnikov >

Xnet

KAZALO

MICROSOFT OFFICE

10

Power Query – Kako stolpce, ki v celicah vsebujejo več podatkov, razdeliti na vrstice

Klemen Vončina

MCT, Microsoft Office Specialist Master

UI/UX DESIGN

21

UI Trendi za leto 2022

Premzl Urška

RAZVOJ

28

Chrome Orodja za razvijalce - Performance

Domen Gričar

MCSD, MCSA, MCT

SHAREPOINT

14

Microsoft Lists

Robi Vončina

MVP, MCT, MCITP, MCSA, MCTS

RAZVOJ

23

Če nisi ljubitelj client-side kode, je tukaj zate Blazor .NET Core platforma

Gašper Rupnik

MCT, MS, MCSD, MCPS

ADMINISTRACIJA

30

PowerShell kotiček

Aleš Lipušček

MCP, MCTS, MCITP

SQL

17

Optimizing Analytical Queries Part 3: Joins

Dejan Sarka

MVP, MCT

RAZVOJ

25

Xnet gre samozavestno globlje tudi v vode mobilnega razvoja

Gašper Rupnik

MCT, MS, MCSD, MCPS

DRUGO

32

Moj prvi Microsoft certifikat

Petra Militarev

Vodja izobraževanj

Xnet ekipa

Klemen	ZARJI SE JE MALO MUDILO. ČESTITKE!
Aida	POČASI GRE NA BOLJE.
Manca	17. 4. BO POSEBEN DAN, JUHU!!
Robi	PONOSNI STRIC!
Urška	VČASIH SE PA KAJ ZATAKNE ;)
Domen	ZAVZET MENTOR
Andraž	SRČEN IN DELAVEN
Dejan	STATISTIKA MU JE STRAST
Anja	POLICIST JO VABI
Robi, Domen,	NOVE KOMPETENCE, NOVI IZPITI
Gašper, Klemen,	
Andraž, Urška	
Gašper	BO SNEŽILO AL' BOM SPAL?
Petra	ESI, ASI, ILT, MLP, AI, BI, …PA SE ZNAJDI ;)
Mojca	»LETOS VEČERJA BO!«
Aleš	10 ALI 15 M?
Miha	ŽOGOBRC JE ZAHTEVNA REČ
Luka	CELO ŠOLARJE ZNA NAVDUŠIT. BRAVO
Jože	NAZAJ GA ŠE NE MIKA

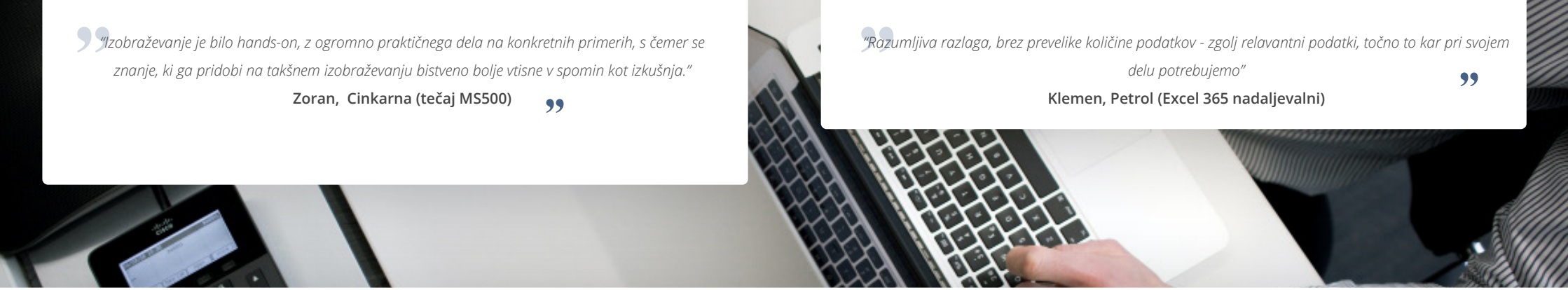
KOLOFON

ISSN: 1408-7863
Kompas Xnet d.o.o.
Stegne 7
1000 Ljubljana

Telefon: 01 5136 990
Fax: 01 5136 999
Email: info@kompas-xnet.si
Web: https://www.kompas-xnet.si

Urednica in oblikovalka
Urška Premzl

Člani uredništva
Aleš Lipušček, Aida Kalender Avdić, Gašper Rupnik, Miha Pihler, Jože Markič, Klemen Vončina, Robert Vončina, Anja Rupnik, Petra Militarev, Dejan Sarka, Andraž Bergant, Manca Gruden



"Izobraževanje je bilo hands-on, z ogromno praktičnega dela na konkretnih primerih, s čemer se znanje, ki ga pridobi na takšnem izobraževanju bistveno bolje vtisne v spomin kot izkušnja."

Zoran, Cinkarna (tečaj MS500)

"Razumljiva razlaga, brez prevelike količine podatkov - zgolj relavantni podatki, točno to kar pri svojem delu potrebujemo"

Klemen, Petrol (Excel 365 nadaljevalni)

TEČAJI ZA IT STROKOVNJAKE

Počutili se boste kot v učilnici, vendar iz udobja doma/pisarne

AZ800

Administering Windows Server Hybrid Core Infrastructure

Kdaj: : 14.2. - 17.2.2022

Predava: Jože Markič , MCT

POGLEJ VEČ

AZ204

Developing solutions for Microsoft Azure

Kdaj: : 7.3. - 11.3.2022

Predava: Saša Kranjac

POGLEJ VEČ

MD101

Managing Modern Desktops

Kdaj: : 21.3. - 25.3.2022

Predava: Jože Markič, MCT

POGLEJ VEČ

MS 100

Microsoft 365 Identity and Services

Kdaj: : 28.2. - 4.3.2022

Predava: Miha Pihler, MCT

POGLEJ VEČ

Za vse informacije smo vam na voljo na info@kompas-xnet.si ali prek tel.: 01 5136 990

TEČAJI ZA UPORABNIKE

Spoznajte, v živo ali na daljavo, osnovne ali napredne funkcionalnost zbirk programov Microsoft Office, Microsoft Office 365.

Microsoft Project

Kdaj: 14. – 15. 2. 2022

POGLEJ VEČ

Vrtilne tabele

Kdaj: 18. 2. 2022

POGLEJ VEČ

Power Excel

Kdaj: 25. 2. 2022

POGLEJ VEČ

Microsoft Access začetni

Kdaj: 9. – 11. 3. 2022

POGLEJ VEČ

Uvod v Excel BI

Kdaj: 17. – 18. 3. 2022

POGLEJ VEČ

Microsoft Access nadaljevalni

Kdaj: 29. – 31. 3. 2022

POGLEJ VEČ

Klemen Vončina
Microsoft Office Specialist Master, MCT
klemen.voncina@kompas-xnet.si

Power Query – Kako stolpce, ki v celicah vsebujejo več podatkov, razdeliti na vrstice

Pred časom sem bil soočen s sledečim izzivom – kako na podlagi .csv datoteke, ki jo uvozimo v Power Query, zapisati vse podatke v tabelarično obliko. Delal sem popis virtualiziranih strežnikov v našem okolju in med drugim me je zanimala poraba diskovja. V .csv datoteke sem s pomočjo PowerShella naredil izvoz s podatki o virtualiziranih strežnikih, nato sem se lotil uvoza v Power Query. Ob uvozu .csv datoteke v Power Query pa sem opazil, da nekatere celice vsebujejo več kot eno informacijo. Na spodnji sliki lahko vidimo, da na primer stolpec "Column4" v nekaterih celicah vsebuje velikosti več kot enega diska.

Naša želja pa je, da bi v vsaki vrstici imeli podatek o velikosti za vsak disk posebej, kot na sliki spodaj. Poleg tega lahko opazimo tudi, da imajo sedaj stolpci bolj smiselna imena, kar je bilo tudi potrebno spremeniti v poizvedbi.

Žal se do tega rezultata ne da priti z enostavnim klikanjem po ukazih Power Queryja, pač pa je treba spisati nekaj vrstic M kode. V tem članku bomo opisali celoten postopek. Delali bomo v Excelu, postopek pa bi bil praktično enak tudi preko Power BI Desktopa.

Prvi korak je seveda uvoz .csv datoteke s pomočjo Power Queryja. V Excelu se postavimo na trak Podatki (Data), kliknemo na ukaz Iz besedila/CSV (From Text/CSV) in poiščemo datoteko, ki jo želimo uvoziti.

V naslednjem koraku določimo nekatere značilnosti te .csv datoteke, denimo regionalne značilnosti in predvsem ločilo, ki v datoteki med seboj ločuje posamične stolpce. V našem primeru je to podpičje.

Column1	Column2	Column3	Column4
Host	VMName	VHDType	VHDSize
Server-01	Exchange-01	Fixed, Fixed	182536110080, 78383153152
Server-01	SharePoint-01	Fixed, Fixed, Fixed	85899345920, 42949672960, 21474836480
Server-02	SQL-01	Fixed, Fixed, Fixed, Fixed	85899345920, 214748364800, 107374182400, 53687091...

Host	VMName	VHDType	VHDSize
Server-01	Exchange-01	Fixed	182536110080
Server-01	Exchange-01	Fixed	78383153152
Server-01	SharePoint-01	Fixed	85899345920
Server-01	SharePoint-01	Fixed	42949672960
Server-01	SharePoint-01	Fixed	21474836480
Server-02	SQL-01	Fixed	85899345920
Server-02	SQL-01	Fixed	214748364800
Server-02	SQL-01	Fixed	107374182400
Server-02	SQL-01	Fixed	53687091200
Server-02	DC-01	Fixed	53687091200

PowerQueryData.csv

Izvor datoteke: 1250: srednjeevropska (Windows) | Ločilo: Podpičje | Zaznavanje podatkovnega tipa: Na podlagi prvih 200 vrstic

Column1	Column2	Column3	Column4
Host	VMName	VHDType	VHDSize
Server-01	Exchange-01	Fixed, Fixed	182536110080, 78383153152
Server-01	SharePoint-01	Fixed, Fixed, Fixed	85899345920, 42949672960, 21474836480
Server-02	SQL-01	Fixed, Fixed, Fixed, Fixed	85899345920, 214748364800, 107374182400, 536870...
Server-02	DC-01	Fixed	53687091200

Nastavitve spremenimo glede na naše potrebe in kliknemo Pretvori podatke (Transform Data) v spodnjem desnem kotu okna.

Odprl se bo Power Query urejevalnik, kjer bomo glede na uvožene podatke že opazili težave iz prve slike tega članka – neustrezno poimenovane stolpce (Column1, Column2…) ter več kot 1 podatek v nekaterih celicah.

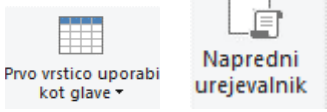
Prve težave se lahko rešimo na zelo enostaven način. Na traku Preoblikuj (Transform) imamo namreč ukaz Prvo vrstico uporabi kot glave (Use First Row as Headers), ki prvo vrstico uvoženih podatkov pretvori v naslove stolpcev.

Problema, ki jih imamo v stolpcih VHDType in VHDSize, pa sta nekoliko bolj kompleksna. Rešili ju bomo tako, da bomo za vsakega od problematičnih stolpcev naredili nov stolpec, ki bo v vsaki celici vseboval seznam vseh vrednosti iz problematičnega stolpca. Opis je morda nerazumljiv, tako da si oglejmo praktično rešitev.

Za začetek si prikažimo trak Prikaži (View) in odprimo Napredni urejevalnik (Advanced Editor).

Zdaj pa bomo v "let" blok kode za vsak problematičen stolpec dodali nov stolpec, ki bo vrednosti iz problematičnega stolpca zapisal kot seznam. To naredimo na sledeč način:

Na začetku vsake vrstice je ime koraka oziroma spremenljivke, temu pa sledi navodilo, kaj naj se zgodi. Kot vidimo je bil vsakič uporabljen ukaz Table.AddColumn, ki na podlagi rezultata prejšnje spremenljivke, npr. #"ChangedType1", naredi nov stolpec, npr. "CustomColumn1". Točno na kakšen način naj se stolpec naredi je zapisano na koncu ukaza. Kot vidimo, je zapisano, naj se pri vsaki vejici in presledku (", ") vsebina stolpca razbije.



Host	VMName	VHDType	VHDSize
Server-01	Exchange-01	Fixed, Fixed	182536110080, 78383153152
Server-01	SharePoint-01	Fixed, Fixed, Fixed	85899345920, 42949672960, 21474836480
Server-02	SQL-01	Fixed, Fixed, Fixed, Fixed	85899345920, 214748364800, 107374182400, 53687091...
Server-02	DC-01	Fixed	53687091200

Če nas zanima rezultat teh dodatnih korakov, lahko v "in" blok vpišemo ime zadnjega koraka oziroma spremenljivke, torej #"Custom2". Rezultat nam bo nekoliko neobičajen, saj bosta nova 2 stolpca v celicah vsebovala "List".

CustomColumn1	CustomColumn2
List	List
List	List
List	List
List	List

Če pa na eno od teh celic kliknemo, bomo v spodnjem delu Power Query okna videli, da celica dejansko vsebuje seznam, v katerem je vsebina celice problematičnega stolpca zapisana po vrsticah.

List
85899345920
214748364800
107374182400
53687091200

Na ta način bi v posamične vrstice razbili problematične podatke iz poljubnega števila stolpcev, za naš članek bosta dovolj že 2. Naslednji korak pa je, da vse te stolpce s seznami združimo v 1 stolpec, ki bo v vsaki celici vseboval tabelo. To zoper storimo s pomočjo naprednega urejevalnika na sledeč način:

```
#"Custom1" = Table.AddColumn(#"Changed Type1", "CustomColumn1", each Text.Split([VHDType], ", ")),
#"Custom2" = Table.AddColumn(#"Custom1", "CustomColumn2", each Text.Split([VHDSize], ", ")),
#"Custom3" = Table.AddColumn(#"Custom2", "CustomColumn3", each Table.FromColumns({[CustomColumn1], [CustomColumn2]}))
```

Spremenljivka #"Custom3" zopet ustvari nov stolpec, vendar je ta narejen tako, da iz vsebine prejšnjih dveh stolpcev ustvarja tabele (Table. FromColumns). Zopet lahko sproti preverimo rezultat tako, da v "in" blok zapišemo ime zadnje spremenljivke, torej #"Custom3". Zopet lahko opazimo, da smo dobili nov stolpec, s tem da ta vsebuje v vsaki celici tabelo. Če na eno od celic kliknemo, si bomo v spodnjem

```
in
    #"Custom3"
```

CustomColumn3
Table
Table
Table
Table

Column1	Column2
Fixed	182536110080
Fixed	78383153152

delu Power Query okna lahko ogledali vsebino tiste tabele.

Za nadaljevanje si zapomnimo, da imajo tudi stolpci teh tabel imena, kot vidimo na sliki zgoraj (Column1, Column2). Naslednji korak je pravzaprav opcijski, vendar smiseln s stališča, da ohranimo le podatke, ki jih potrebujemo. Power Queryju bomo namreč povedali, naj ohrani le zadnji ustvarjeni stolpec (CustomColumn3) ter 2 neproblematična stolpca iz izvorne .csv datoteke, Host ter VMName.

```
    #"Custom1" = Table.AddColumn(#"Changed Type1", "CustomColumn1", each Text.Split([VHDType], ", ")),
    #"Custom2" = Table.AddColumn(#"Custom1", "CustomColumn2", each Text.Split([VHDSize], ", ")),
    #"Custom3" = Table.AddColumn(#"Custom2", "CustomColumn3", each Table.FromColumns({[CustomColumn1], [CustomColumn2]})),
    #"Custom4" = Table.SelectColumns(#"Custom3", {"Host", "VMName", "CustomColumn3"}),
    #"Custom5" = Table.ExpandTableColumn(#"Custom4", "CustomColumn3", {"Column1", "Column2"},{"VHDType","VHDSize"})
in
    #"Custom5"
```


Host	VMName	CustomColumn3
Server-01	Exchange-01	Table
Server-01	SharePoint-01	Table
Server-02	SQL-01	Table
Server-02	DC-01	Table

Če zopet pogledamo rezultat zadnjega koraka, tako, da pod "in" blok zapišemo #"Custom4", bomo videli, da nam res ostanejo le še izbrani stolpci.

Do konca nam ostane le še en korak, in sicer da dobljene objekte v stolpcu CustomColumn3, torej tabele, razširimo v samostojne stolpce in vrstice. To lahko storimo bodisi preko gumba poleg imena stolpca (2 puščici, ki kažeta vsaka v svojo smer), bodisi preko M kode. Ker smo večino stvari v članku naredili preko kode, pa dajmo še zadnji korak. Dodana prednost dela na ta način je tudi to, da lahko v isti sapi še določimo imena novonastalih stolpcev.

Ukaz Table.ExpandTableColumn torej razširi stolpec, ki vsebuje tabele (CustomColumn3), iz tabel vzame stolpca Column1 in Column2, na katere sem opozoril nekaj odstavkov nazaj, in jih preimenuje v VHDType in VHDSize. Na sliki je prikazan tudi "in" blok, kjer je določeno, da bo poizvedba na koncu prikazala rezultat tega zadnjega koraka. Sicer sem sliko s končnim rezultatom naših transformacij v članek dodal že na začetku, pa jo bom zaradi boljše preglednosti vseeno dodal še na to mesto.

Opazimo lahko ne le, da je zdaj vsak zapis kar se tiče velikosti diska (VHDSize) in tipa diska (VHDType) v svoji vrstici, pač pa tudi, da je Power Query v vsaki vrstici samodejno ponovil tudi zapise iz stolpca Host in VMName. Podatki so sedaj pripravljeni na nadaljnje transformacije in obdelavo.

```
#"Custom1" = Table.AddColumn(#"Changed Type1", "CustomColumn1", each Text.Split([VHDType], ", ")),
#"Custom2" = Table.AddColumn(#"Custom1", "CustomColumn2", each Text.Split([VHDSize], ", ")),
#"Custom3" = Table.AddColumn(#"Custom2", "CustomColumn3", each Table.FromColumns({[CustomColumn1], [CustomColumn2]})),
#"Custom4" = Table.SelectColumns(#"Custom3", {"Host", "VMName", "CustomColumn3"})
```

Host	VMName	VHDType	VHDSize
Server-01	Exchange-01	Fixed	182536110080
Server-01	Exchange-01	Fixed	78383153152
Server-01	SharePoint-01	Fixed	85899345920
Server-01	SharePoint-01	Fixed	42949672960
Server-01	SharePoint-01	Fixed	21474836480
Server-02	SQL-01	Fixed	85899345920
Server-02	SQL-01	Fixed	214748364800
Server-02	SQL-01	Fixed	107374182400
Server-02	SQL-01	Fixed	53687091200
Server-02	DC-01	Fixed	53687091200

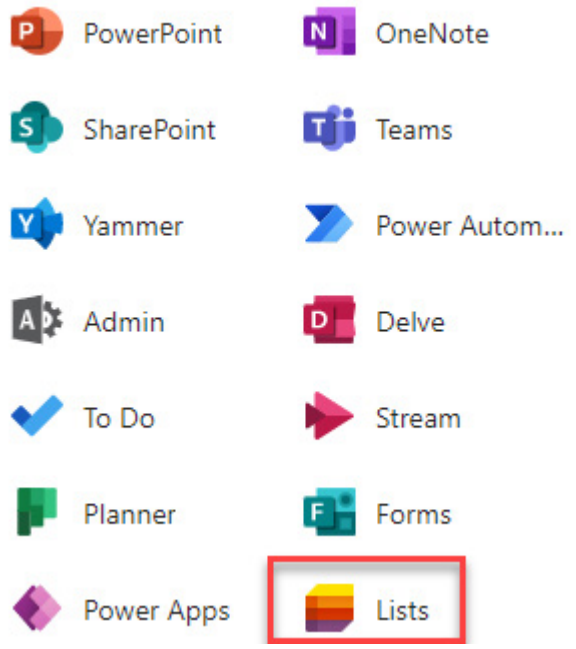
Robi Vončina
MVP, MCT, MCITP, MCSA, MCTS
robi.voncina@kompas-xnet.si

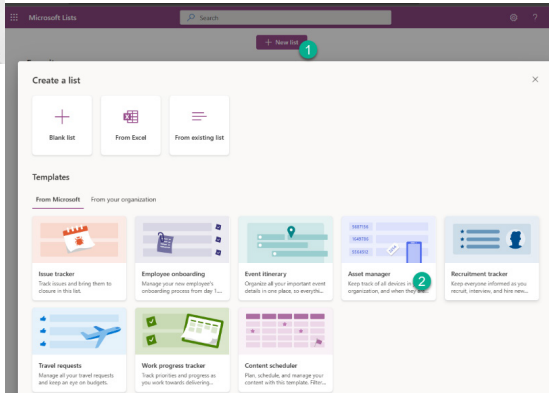
Microsoft Lists

Že pred časom je Microsoft najavil nov Office 365 aplikacijo, ki se imenuje Microsoft Lists. Mnogo vprašanj se poraja, kaj MS Lists sploh je in kako se razlikuje od obstoječih SharePoint seznamov. Na vsa ta vprašanja bom poskusil odgovoriti v tem članku.

Ideja MS Lists je pravzaprav evolucija klasičnih SharePoint seznamov, kjer bi lahko ti seznami bili bolj dinamični, bolj prilagodljivi, boljši za upravljanje in dosegljivi tudi na mobilnih napravah v mobilni aplikaciji kot samostojna entiteta. Če pogledamo s stališča SharePoint seznama in seznama v MS Lists sta to ena in ista zadeva, kar lahko vidimo tudi v sami aplikaciji MS Lists. Pogoj, da lahko uporabljajo MS Lists je, da uporabljamo moderne sezname (nove) in ne klasične predloge, ki so bile značilne za on-premise okolja.

Do MS Lists aplikacije ali domače strani MS Lists lahko pridemo tako, da v brskalniku v »app launcherju« odpremo aplikacijo »MS Lists«, kot je prikazano na sliki. S klikom, se nam odpre nov zavihek v brskalniku, ki nas popelje na OneDrive mesto, kar boste opazili, če pozorno pogledate URL naslov. Na tem mestu imamo samo pregled seznamov, ki smo jih nedavno uporabljali in klik na enega od njih nas pelje na dejansko mesto seznama. Če boste pozorni, boste ugotovili, da se URL naslovu seznama doda »query string« parameter »?env=WebViewList«, ki vpliva na to ali se seznam pokaže kot navaden (SharePoint) seznam ali pa se pokaže kot MS Lists seznam.





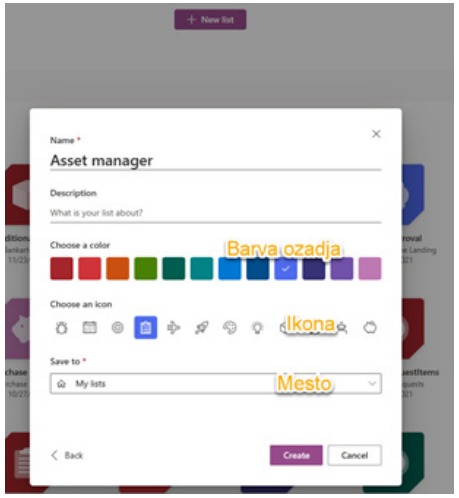
Ena izmed prednosti ustvarjanja seznama na tem mestu je tudi to, da lahko iz centralne točke ustvarimo seznam na praktično katerem koli mestu, kjer imamo zadostne pravice.

S klikom na »New list«, se nam odpre pogovorno okno, kjer nam je proces že poznan in od nas zahteva, da si izberemo vrsto, predlogo seznama, ki bi ga želeli ustvariti.

V naslednjem koraku nam predstavi izgled in strukturo samega lista tako da, če smo z izbiro prepričani, lahko potrdimo s klikom na gumb »Use template«.

Tretji, zadnji korak, pa ponudi nekaj več možnosti. Lahko določimo:

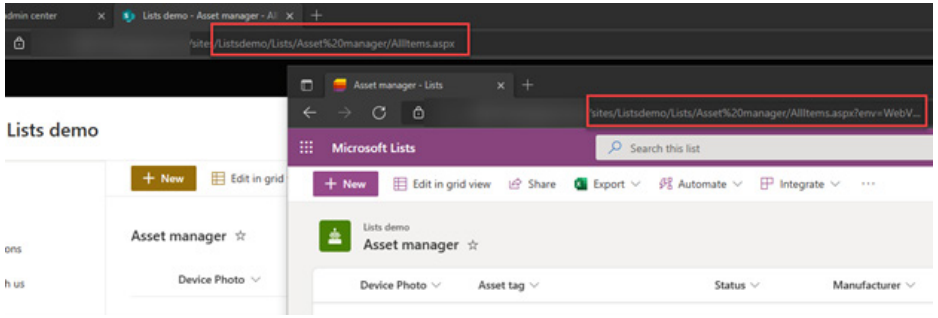
- Naslov seznama
- Opis seznama

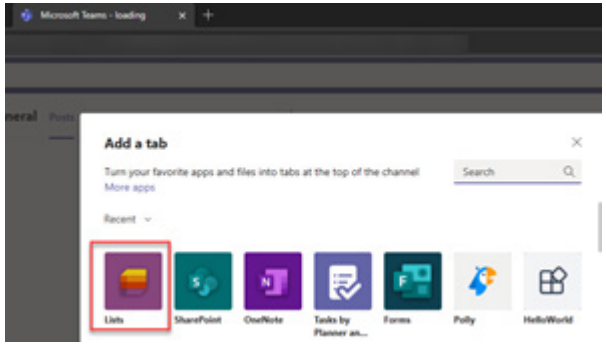


- Barvo ozadja ikone seznama
- Ikono seznama
- Mesto seznama

Za primerjavo, je na spodnji sliki isti seznam. Enkrat je odprt prek MS Lists, v drugem primeru je odprt kot klasičen SharePoint seznam.

Prav tako lahko obstoječe sezname MS Lists dodamo v MS Teams kanale in na enak način, kot bi ustvarjali na SharePoint strani, lahko urejamo vse podatke in nastavitve tudi tukaj. Ko izberemo možnost, da bi radi dodali Lists app kot dodaten zavihek v Teamse, nas nato vpraša ali bi želeli ustvariti nov seznam ali pa bi dodali že obstoječega. Pri dodajanju seznama

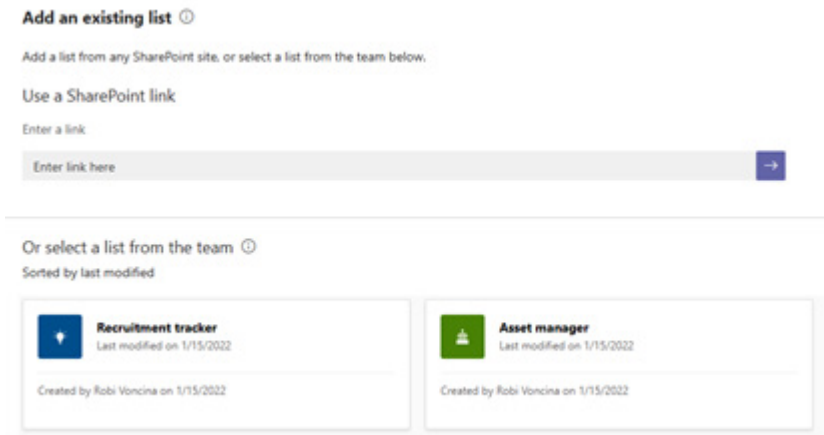




nismo omejeni samo na trenutno mesto, temveč nam dovoli tudi dodajanje iz drugih mest prek linka.

Trenutno so MS Lists še v relativno zgodnji fazi in lahko v prihodnosti pričakujem veliko novosti, ki bodo na voljo v MS Lists aplikaciji/pogledu, ne pa več v SharePoint seznamih. Za enkrat lahko rečemo, da sta to eni in isti zadevi, le da MS Lists predstavlja pogled s šminko in dostop do posameznega modernega seznama prek aplikacije.

Za dodatne informacije se lahko obrnete name prek elektronske pošte na naslov robi.voncina@kompas-xnet.si.



Optimizing Analytical Queries Part 3: Joins

Dejan Sarka
MVP, MCT
dsarka@solidq.com

SQL Server executes a query by a set of physical operators. Because these operators iterate through rowsets, they are also called iterators. There are different join operators, because when performing joins, SQL Server uses different algorithms. SQL Server supports three basic algorithms: nested loops join, merge join, and hash join.

Join Types

The nested loops algorithm is a very simple and, in many cases, efficient algorithm. SQL Server uses one table for the outer loop, typically the table with the fewest rows. For each row in this outer input, SQL Server seeks matching rows in the second table, which is the inner table. SQL Server uses the join condition to find the matching rows. The join can be a non-equijoin, meaning that the equality operator does not need to be part of the join predicate. If the inner table has no supporting index to perform seeks, then SQL Server scans the inner input for each row of the outer input. This is not an efficient scenario. A nested loop join is efficient when SQL Server can perform an index seek in the inner input.

Merge join is a very efficient join algorithm. However, it has its own limitations. It needs at least one equijoin predicate and sorted inputs from both sides. This means that the merge join should be supported by indexes on both tables involved in the join. In addition, if one input is much smaller than another, then the nested loop join could be more efficient than a merge join.

In a one-to-one or one-to-many scenario, a merge join scans both inputs only once. It starts by finding the first rows on both sides. If the end of the input is not reached, the merge join checks the join predicate to determine whether the rows match. If the rows match, they are added to the output. Then the algorithm checks the next rows from the other side and adds them to the output until they match the predicate. If the rows from the inputs do not match, then the algorithm reads the next row from the side with the lower value from the other side. It reads from this side and compares the row to the row from the other side until the value is bigger than the value from the other side. Then it continues reading from the other side, and so on. In a many-to-many scenario, the merge join algorithm uses a worktable to put the rows from one input side aside for reuse when duplicate matching rows are received from the other input.

If none of the input is supported by an index and an equijoin predicate is used, then the hash join algorithm might be the most efficient one. It uses a searching structure named a hash table. This is not a searching structure you can

build, like a balanced tree used for indexes. SQL Server builds the hash table internally. It uses a hash function to split the rows from the smaller input into buckets. This is the build phase. SQL Server uses the smaller input for building the hash table because SQL Server wants to keep the hash table in memory. If it needs to get spilled out to tempdb on disk, then the algorithm might become much slower. The hash function creates buckets of approximately equal size.

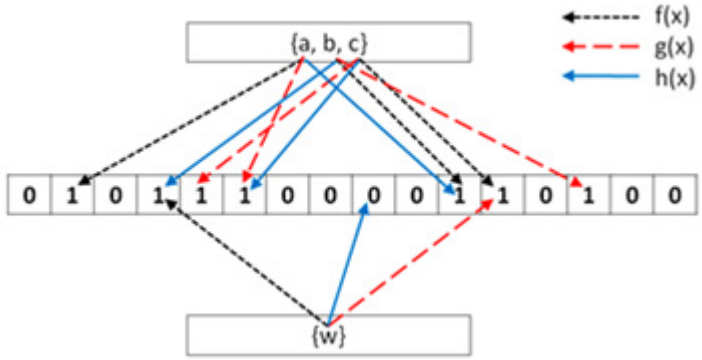
After the hash table is built, SQL Server applies the hash function on each of the rows from the other input. It checks to see which bucket the row fits. Then it scans through all rows from the bucket. This phase is called the probe phase.

A hash join is a kind of compromise between creating a fully balanced tree index and then using a different join algorithm and performing a full scan of one side of the input for each row of the other input. At least in the first phase, a seek of the appropriate bucket is used. You might think that the hash join algorithm is not efficient. It is true that in single-thread mode, it is usually slower than the merge and nested loop join algorithms, which are supported by existing indexes.

Bitmap Filtered Hash Join

SQL Server can split rows from the probe input in advance. It can push the filtering of rows that are candidates for a match with a specific hash bucket down to the storage engine. This kind of optimization of a hash join is called a bitmap filtered hash join. It is typically used in a data warehousing scenario, in which you can have large inputs for a query that might not be supported by indexes. In addition, SQL Server can parallelize query execution and perform partial joins in multiple threads. In data warehousing scenarios, it is not uncommon to have only a few concurrent users, so SQL Server can execute a query in parallel. Although a regular hash join can be executed in parallel as well, a bitmap filtered hash join is even more efficient because SQL Server can use bitmaps for early elimination of rows not used in the join from the bigger table involved in the join.

In the bitmap filtered hash join, SQL Server first creates a bitmap representation of a set of values from a dimension table to prefilter rows to join from a fact table. A bitmap filter is a bit array of m bits. Initially, all bits are set to 0. Then SQL Server defines k different hash functions. Each one maps some set elements to one of the m positions with a uniform random distribution. The number of hash functions k must be smaller than the number of bits in array m. SQL Server feeds each of the k hash functions to get k array positions with values from dimension keys. It set the bits at all these positions to 1. Then SQL Server tests the foreign keys from the fact table. To test whether any element is in the set, SQL Server feeds it to each of the k hash functions to get k array positions. If any of the bits at these positions are 0, the element is not in the set. If all are 1, then either the element is in the set, or the bits have been set to 1 during the insertion of other elements. The following figure shows the bitmap filtering process:



Bitmap filtering

In the preceding figure, the length m of the bit array is 16. The number k of hash functions is 3. When feeding the hash functions with the values from the set {a, b, c}, which represents dimension keys, SQL Server sets bits at positions 2, 4, 5, 6, 11, 12, and 14 to 1 (starting numbering positions at 1). Then SQL Server feeds the same hash functions with the value w from the smaller set at the bottom {w}, which represents a key from a fact table. The functions set bits at positions 4, 9, and 12 to 1. However, the bit at position 9 is set to 0. Therefore, the value w is not in the set {a, b, c}. If all of the bits for the value w were set to 1, this could mean either that the value w is in the set {a, b, v} or that this is a coincidence.

Bitmap filters return so-called false positives. They never return false negatives. This means that when you declare that a probe value might be in the set, you still need to scan the set and compare it to each value from the set. The more false positive values a bitmap filter returns, the less efficient it is. Note that if the values in the probe side in the fact table are sorted, it will be quite easy to avoid the majority of false positives.

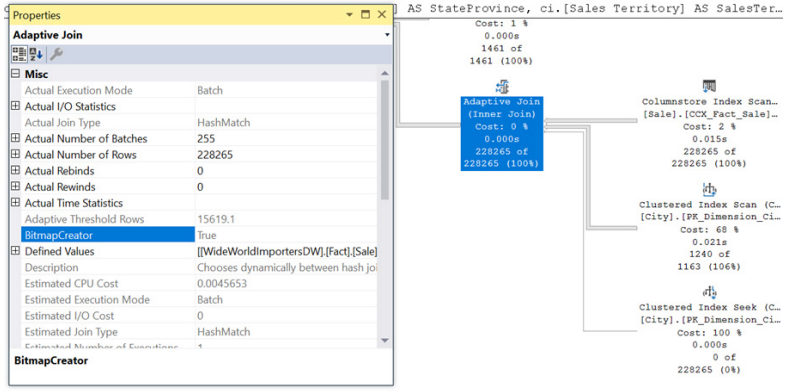
Before starting with the code, let's make sure that the WideWorldImportersDW database compatibility mode is set to 150, the SQL Server 2019 compatibility mode, with the following code.

```
USE master;
ALTER DATABASE WideWorldImportersDW SET
COMPATIBILITY_LEVEL = 150;
GO
```

The following query reads the data from the tables introduced earlier in this section and implements star schema-optimized bitmap filtered hash joins:

```
USE WideWorldImportersDW;
SELECT cu.[Customer Key] AS CustomerKey,
cu.Customer,
 ci.[City Key] AS CityKey, ci.City,
 ci.[State Province] AS StateProvince, ci.[Sales
Territory] AS SalesTeritory,
 d.Date, d.[Calendar Month Label] AS Calen-
darMonth,
 d.[Calendar Year] AS CalendarYear,
```

```
 s.[Stock Item Key] AS StockItemKey, s.[Stock Item] AS Product, s.Color,
      e.[Employee Key] AS EmployeeKey, e.Employee,
      f.Quantity, f.[Total Excluding Tax] AS TotalAmount, f.Profit
FROM Fact.Sale AS f
  INNER JOIN Dimension.Customer AS cu
    ON f.[Customer Key] = cu.[Customer Key]
  INNER JOIN Dimension.City AS ci
    ON f.[City Key] = ci.[City Key]
  INNER JOIN Dimension.[Stock Item] AS s
    ON f.[Stock Item Key] = s.[Stock Item Key]
  INNER JOIN Dimension.Employee AS e
    ON f.[Salesperson Key] = e.[Employee Key]
```




```
 ON f.[Delivery Date Key] = d.Date;
```

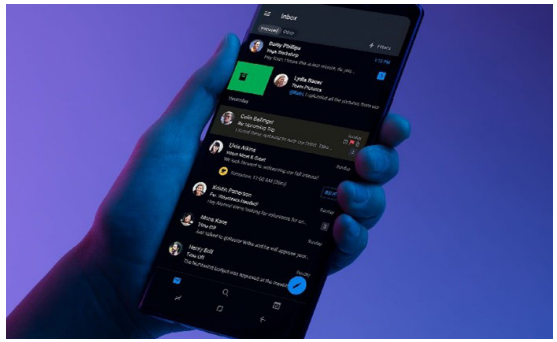
The following figure shows a part of the execution plan. Please note that you can get a different execution plan based on differences in resources available to SQL Server to execute this query. You can see the Adaptive Join operator and part of the properties for this operator. You can note that the property Actual Join Type is HashMatch, and the property BitmapCreator is equal to True.

Bitmap Create operator in the execution plan
You could also check the XML execution plan and find there that the bitmap filter was created, like the following piece of this plan shows.
<AdaptiveJoin BitmapCreator="true" Optimized="false" WithUnorderedPrefetch="true">
Conclusion
This article introduced join types and focused specifically on the bitmap filtered hash join. Although hash join might not be the first choice for transactional systems, it is usually very efficient for analytical queries, especially if it is executed in parallel. You will learn more about analytical queries optimization in the forthcoming articles.

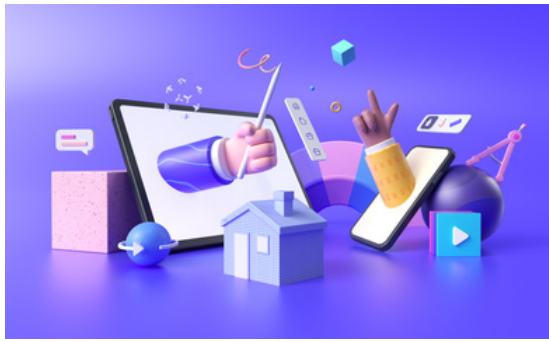
UI Trendi za leto 2022

V tem članku bomo pogledali top UI (user interface) trende za leto 2022, kaj vpliva na spremembe in zakaj bi jih morali uporabljati.

Premzl Urška
urska.premzl@kompas-xnet.com



(Vir: lindaojo.com)



(Vir:dribbble.com)

1. Temni način

Če je bil trend v letu 2021 belo ozadje z veliko belih prostorov, se temni način ali temna tema vračata kot eden najpopularnejši trendov UI designa 2022.

Temni način ni nekaj novega. Več aplikacij in spletnih mest uporabnikom omogoča preklapljanje med svetlim in temnim načinom z visokim kontrastom.

Zakaj izbrati temno temo (svetlo besedilo na temnem ozadju) v letu 2022? Temna tema s kontrastnimi barvami zagotavlja boljšo vidljivost za nekatere slabovidne uporabnike. Poleg tega temna tema zmanjša tudi porabo energije za mobilne telefone z zasloni OLED.

Če še niste razmišljali o tem, da bi dodali temni način za svoje aplikacije in spletna mesta, je verjetno čas, da začnete.

2. 3D in animacije

Potrebe po ustvarjanju 3D animacij se bodo povečale! Animacije so bile včasih ovira za mnoge oblikovalce in razvijalce, saj lahko upočasnijo hitrost nalaganja spletnih mest ali aplikacij in zahtevajo veliko podatkov.

Vendar pa vzlet tehnologije 5G zagotavlja večjo hitrost prenosa podatkov. Z večjo hitrostjo lahko uporabniki dostopajo do datotek, dokumentov in aplikacij, vključno s 3D animacijami, brez dolgega čakanja.

Tudi v Windows 11 je dodal v svoje teme nova 3D ozadja.

3. Optimizacija za različne velikosti zaslonov in naprav

Vsakodnevno delo oblikovalca ni le ustvarjanje nabora ljubkih ikon in grafik vendar mora oblikovalec ustvariti izgled, ki se prilagaja različnim velikostim zaslona. Poleg običajnih prelomnih točk 480px, 768px, 1024px in 1280px morajo oblikovalci poskrbeti za mobilne naprave, tablice, majhne zaslone, zložljive naprave, prenosne računalnike, namizne računalnike, velike zaslone, izjemno velike

zaslone in TV.

Od leta 2022 naprej, bo potrebno optimiziranja in prilagajanja za še več velikosti zaslonov. Ta trend uporabniškega vmesnika se bo verjetno ohranil zaradi več izdelkov in naprav, od zaslonov v avtomobilu do naprav AR in VR.

4. Priprava aplikacij za velike zaslone

Kot da vse velikosti zaslonov niso dovolj, Google pravi, da bo potrebno aplikacije prilagoditi za velike zaslone, zlasti za namizne računalnike. Google je posodobil tudi Material Design za pomoč pri oblikovanju aplikacij za vse velikosti.

Oblikovalci aplikacij smo se pri oblikovanju aplikacij večinoma osredotočali na mobilne telefone. Vendar je kot eden izmed trendov UI designa 2022 prilagajanje svojih dizajnov za večje zaslone, kot so zložljivi telefoni in namizni računalniki.

5. Zmanjševanje kompleksnosti

Super-aplikacija je bila nekoč popularna samo v Aziji. Tencentov WeChat je bil med prvimi v super-aplikaciji, ki v eni aplikaciji združuje storitve, kot so sporočila, družbeni mediji, plačila, e-trgovina, dostava hrane, taksi in še veliko več.

Poleg tega vidimo, da številna podjetja na globalni sceni začenjajo ustvarjati super-aplikacije. Google je na primer integriral več aplikacij za Gmail, Google Meet in Google Hangout Chat v eno samo aplikacijo pod Gmailom. Twitter dodaja "Spaces", medtem ko YouTube dodaja "Shorts" konkurenčnemu Tiktoku.

Podobno so številne večje aplikacije od začetka pandemije COVID-19 v svoje aplikacije dodale tudi funkcije sodelovanja. Vse te dodatne funkcije in funkcionalnosti so povečale zapletenost aplikacij. Predvsem pa so te zapletenosti povzročile zmedo in padce uporabniške izkušnje za uporabnike.

Leta 2022 se bo od oblikovalcev zahtevalo, da čim bolj zmanjšamo kompleksnost aplikacij. Medtem ko podjetja še naprej dodajajo vse več ponudb in storitev v svoje aplikacije, bomo oblikovalci dolžni ohraniti izglede aplikacij čiste in intuitivne.

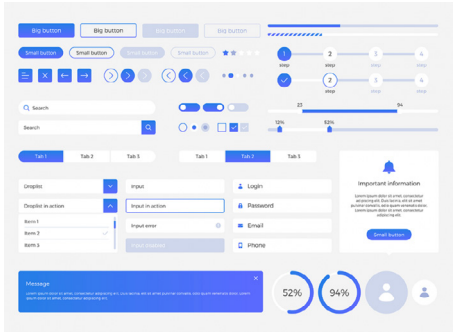
6. Sistem oblikovanja in knjižnica v programu Figma

Sistem oblikovanja omogoča oblikovalcem, da hitro ustvarijo izdelek, pri čemer se dosledno držijo podobe podjetja (CGP, barve, gumbi, ikone, sence itd,...). Ustvari se skupna knjižnica, ki jo uporabljajo vsi v podjetju.

Sistem oblikovanja je bil nekoč stvar samo za večja podjetja, saj je vzdrževanje sistema oblikovanja zamudno, še posebej, če se ne bo pogosto uporabljal.

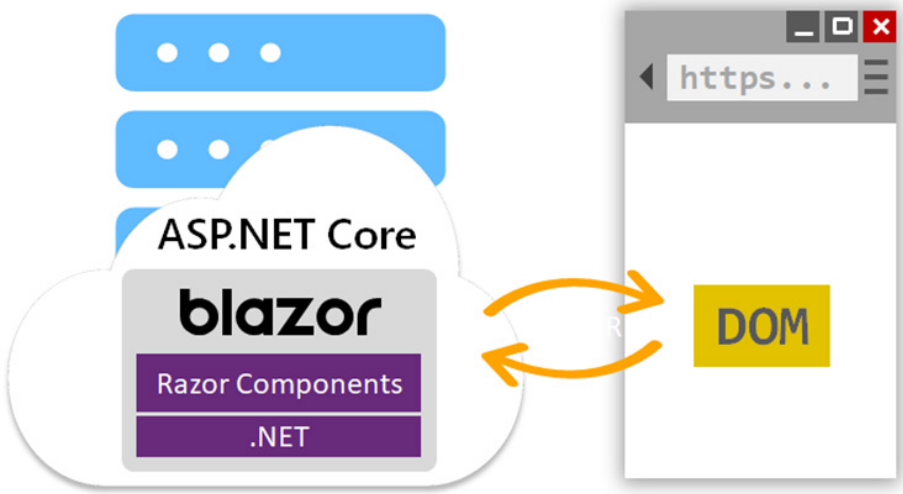
Vendar pa bodo z razpoložljivostjo cenovno dostopnih, enostavnih za uporabo in razširljivih orodij, kot je Figma, tudi manjša podjetja začela ustvarjati lastne sisteme oblikovanja od leta 2022 dalje. Vsi v podjetju bodo imeli koristi od skupnega oblikovalskega sistema, od oblikovalske ekipe, ekipe za izdelke/storitve do marketinške ekipe.

(Vir:rossul.com)



Gašper Rupnik
MCT, MS, MCSD, MCPS
gasper.rupnik@kompas-xnet.si

Če nisi ljubitelj client-side kode, je tukaj zate Blazor .NET Core platforma



Aprila 2020 smo z eno izmed naših strank sklenili dogovor o strokovnem sodelovanju ter pričeli z zelo zanimivim projektom: prepis ene izmed njihovih glavnih poslovnih aplikacij iz desktop okolja v spletno aplikacijo. Vemo namreč, da je trend zadnjih nekaj let tak, da popularnost in uporabnost desktop / namiznih aplikacij namreč upada, vse skupaj pa se seli v splet – kot interna aplikacija ali del intraneta (npr SharePoint) ali pa kot globalna aplikacija.

Desktop aplikacija je bila že več kot 15-20 let nazaj spisana v Delphi programskem jeziku, zato je bila definitivno potrebna prenove. Ker stranka, kateri smo pri temu procesu strokovno pomagali, ni hotela uporabljati client-side tehnologije razvoja spletnih aplikacij, smo morali ubrati drugo pot. Zato smo za ta projekt izbrali **Blazor.** Vemo, da moderna spletna aplikacija ne more obstajati brez client-side kode. Blazor pa omogoča izdelavo moderne, iteraktivne spletne aplikacije z uporabo **C#** server side kode namesto **JavaScript** client-side kode. Blazor aplikacija je sestavljena iz spletnih komponent, implementiranih v

C#, HTML in CSS jezikih. Tako je client-side in server-side koda spisana v C# jeziku.

Kako je to mogoče? Blazor lahko poganja tvojo C# client-side kodo direktno v brskalniku, preko uporabe **WebAssembly-a.** Druga opcija pa je poganjanje Blazor client-side logike direktno na **server**-ju. Nek dogodek na spletni strani sproži akcijo oz. klic na server preko uporabe **SignalR** dvosmernega protokola. Ko se logika na serverju izvede, le ta nazaj spletni strani, ki se izvaja znotraj brskalnika klienta, pošlje podatke / ukaz / nalogo, kar povzroči, da se DOM (oblika oz vsebina spletne strani) ustrezno prilagodi.

Stvar je več kot zanimiva, stabilna, uporabna, v določenih primerih še bolj kot client-side koda spisana v JavaScriptu. Naj omenim, da je v trenutni izdaji Pike tudi članek na temu .NET MAUI frameworka (univerzalnega frameworka za izdelavo mobilnih aplikacij), kjer lahko uporabljaš prav tako ta, zgoraj omenjeni **Blazor.**

Omeniti velja, da smo v tem projektu uporabili tudi **Telerik** komponente, ki so ene izmed najbolj razširjenih že izdelanih komponent, ki obstajajo na spletu in omogočajo lažji in hitrejši razvoj. Telerik je vedno v koraku s časom in seveda tako vsebuje komponente tudi za vse najnovejše frameworke – kot je tudi Blazor.

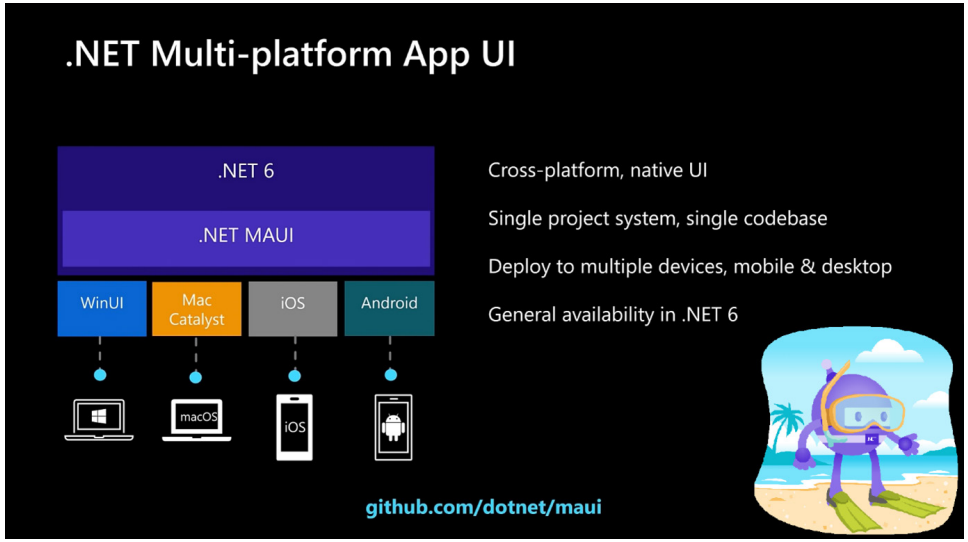
Tekom razvoja zgoraj omenjene aplikacije smo prišli do ogromno zanimivih problemov, dejstev, omejitev in seveda tudi rešitev, saj je projekt ravnokar dobro splavil v produkcijsko uporabo. Naj omenim še enkrat, da gre za večjo, eno izmed najbolj pomembnih poslovnih aplikacij stranke, kjer smo mi kot Xnet sodelovali kot zunanji poslovni partner za podporo in pomoč pri programiranju ter pri reševanju večjih problemov, zagat, težav, strukturnih odločitev.

Zato bom par naslednjih številk Pike izkoristil za to, da najbolj pereče in zanimive probleme tudi predstavim, ter mogoče še komu približam to platformo, ki bi lahko rekli da je »the way to go«, če nisi ljubitelj JavaScript jezika, pa bi rad bil v stiku z časom razvoja modernih aplikacij, kjer vemo, da se vse stvar selijo na splet, ki ima same prednosti.

Gašper Rupnik
MCT, MS, MCSD, MCPS
gasper.rupnik@kompas-xnet.si

Xnet gre samozavestno globlje tudi v vode mobilnega razvoja

Septembra 2021 smo pričeli z izdelavo mobilne aplikacije za eno izmed naših strank, kjer smo kot platformo, na kateri bomo razvijali univerzalno aplikacijo (tako za Android kot za iOS operacijski sistem), vzeli novo (v tem trenutku še ne uradno izdano) Microsoftovo platformo, ki temelji na novem **.NET 6.0** Frameworku - .**NET Multi-platform App UI** oz skrajšano **.NET MAUI**.



Stranka si je namreč želela univerzalno aplikacijo za katerokoli mobilno platformo, sami pa smo za izdelavo le te vzeli novo, zgoraj omenjeno Microsoftovo platformo. Razlogov za tako odločitev je bilo več. Prvi izmed njih je bil ta, da smo podjetje, ki se ukvarja v 99,9% z razvojem na Microsoftovih tehnologijah – Microsoftovem tehnološkem stacku. Drug razlog je ta, da smo hoteli izbrati čimbolj nov framework, najnovejšo tehnologijo, saj smo s tem tako mi kot stranka na varni strani, da bo stvar ostala živa ter jo bo možno nadgrajevati in uporabljati čim dlje časa. Zavedamo se namreč, kakšna je situacija v svetu v preteklosti s takimi in podobnimi frameworki za izdelavo univerzalnih aplikacij. Svet mobilnih operacijskih sistemov se namreč razvija s svetlobno hitrostjo in temu morajo

vse te univerzalne platforme slediti z enako hitrostjo, sicer lahko hitro prideš do tega, da si enostavno ostal »out-of-support« in svoje aplikacije ne moreš več oddati na store ali jo zgolj posodobiti. Druga opcija je seveda vedno najbolj zanesljiva – spišeš mobilno aplikacijo v native okolju vsakega operacijskega sistema -> torej za Android seveda v jeziku Java, za iOS pa v jeziku Swift kot ena izmed native opcij. Seveda se tudi s tem ukvarjamo, ampak seveda to potegne za sabo 1x večje stroške za stranko, kot posledica 1x več dela za razvijalsko ekipo, pa seveda tudi sam izgled aplikacije v tem primeru ni nikoli točno 1:1 (Android:iOS). To so torej glavni trije razlogi, zakaj smo pri tem projektu šli v razvoj v novi t.i. **.NET MAUI** platformi.

Seveda pa ta odločitev ni prinesla samo prednosti, torej samo dobrih točk, ampak seveda tudi nekaj manj dobrih, katere je bilo potrebno ob tem prebroditi ali pa se z njimi trenutno še vedno ukvarjamo. Odločitev o razvoju v tej platformi smo s stranko sprejeli že v prvi polovici leta 2021, ko je bila le-ta še v zelo zgodnji fazi razvoja, nestabilna, z veliko mero še nerazvitih funkcionalnosti. Takrat je bila s strani Microsofta podana informacija, da bo platforma zaživela v stable/release verziji v septembru 2021, kar je bilo tako za nas kot za stranko sprejemljivo, saj je bil plan, da naša aplikacija zaživi – pride na store, ravno v tem času, torej zgodaj jeseni.

Začeli smo z razvojem designa aplikacije, za kar je poskrbela naša odlična designer-ka **Urška Premzl**. Tako kot vsaka stvar, ki pride izpod njenih rok, je bila grafična predloga tudi tokrat na nivoju. Njeni designi niso nikoli »developer friendly« ali drugače povedano enostavni za razvoj. Seveda vsak, ki se ukvarja z razvojem aplikacij ve, da sta si izraza »developer friendly« in »user friendly« ravno obratna -> torej če želiš spisati aplikacijo, ki bo uporabniku čimbolj enostavna, sofisticirana, pregledna in učinkovita, bo to seveda prineslo več dela programerju, več zapletov, več izven standardnih postopkov in komponent. Ker smo v tem primeru izbrali platformo, ki še ni bila niti v celoti razvita, je seveda v njej teh komponent manj, imamo za sam izgled aplikacije manj možnosti, kar pomeni da imamo malo zvezane roke. Tukaj nam je že v začetku, kot zelo dobra rešitev prišel prav nam zelo dobro znan **Blazor**, ki smo ga predhodno že do potankosti spoznali pri enem izmed drugih projektov, ko smo razvijali kompleten, večji spletni sistem, spisan v platformah **.NET Core MVC Blazor**, ki je prav tako predstavljen v tokratni izdaji Pike.

Ker smo torej razvojna ekipa, ki se v veliki meri ukvarja z razvojem v najnovejših spletnih tehnologijah tudi vemo, da se razvoja v spletnih tehnologijah vsekakor splača učiti oz držati. Le tako imaš določeno zagotovilo, da če v njem pišeš danes, boš lahko pisal tudi čez nekaj let.

Na podlagi vsega navedenega, smo seveda izbrali to opcijo, saj imamo v hiši nekaj dobrih **front-end** razvijalcev. **Domen Gričar in Andraž Bergant** imata spletno programiranje designa v mezincu. Tako smo si implementacijo Urškinih idej v .NET MAUI platformo olajšali s pomočjo frameworka Blazor, ki omogoča, da mobilno aplikacijo izdelamo na tehnološkem stacku spletnih aplikacij. Torej uporabili smo platformo **.NET MAUI Blazor.**

Omenil pa nisem še ene pomembne stvari - .NET MAUI platforma je naslednik zelo znane **Xamarin** platforme, zato se poznavalci slednje lahko kar hitro udomačijo tudi v novi platformi. Seveda sam potek razvoja ni bil tako rožnat, kot bi si človek želel, ampak le to je bilo tudi potihem pričakovati, glede na to da smo šli na pota platforme, ki še ni bila uradno niti do konca razvita (v tistem trenutku sredi marca bi lahko rekli, da ni bila niti na polovici), kaj šele primerna za nek končni stable produkt. Ker smo nekateri ljudje po naravi taki, da vedno raje tvegamo, kot pa bivamo v coni udobja, sploh, če to prinaša tudi prednosti, se radi naučimo nekaj novega in nismo rutinski. Tudi naša ekipa je učenje nove tehnologije sprejela kot izziv in z zelo veliko mero motivacije. S tem smo dobili le še dodaten zagon, veter v jadra, ob začetku novega projekta.

Seveda se je potrebno zavedati, da na spletu ni bilo takrat prav nobene dokumentacije, v resnici je tudi dan-danes ni dosti več kot takrat, da bi programer lažje spoznal in se udomačil v novi .NET MAUI platformi. Zato bi lahko rekli, da je bil ta projekt kar ena velika raziskovalna naloga, polna mejnikov, pregrad in neznank. Seveda pa na koncu, ko smo vse skupaj pripeljali do implementacije pri naročniku, tudi uspešna naloga, na katero smo upravičeno ponosni.

Naj omenim še to, da platforma še vedno ni v release verziji. Microsoft je izzid platforme prestavil iz septembra 2021 na konec Q1 leta 2022. Sami smo projekt pripeljali do konca tako, da smo si pomagali z znanjem in izkušnjami iz raznih drugih frameworkov, ter ga poskusili implicirati tukaj. Seveda je velika prednost v tem, da vse skupaj temelji na **.NET Core**, ki je že v osnovi res lepo zastavljen, lepo grajen, modularno razširljiv in lahko stvari razmeroma enostavno implementiraš oz rešiš. Vsekakor pomagajo tudi izkušnje in poznavanje predhodne platforme Xamarin, iz katere se je .NET MAUI tudi razvil. Veliko pripomore tudi to, da je .NET MAUI odprtokodni projekt, kjer lažje vidiš tudi v njegovo drobovje, preko GitHub kanalov pomagaš razvojni ekipi, se z njimi posvetuješ in jim sporočaš, kaj še ne dela in kako bi moralo delovati ter seveda predlagaš določene funkcionalnosti.

Vse skupaj smo pripeljali do točke, ko je potrebno aplikacijo objaviti v Storu. Aplikacijo smo uspešno poslali v Google Play Store. Manjka pa nam le še to, kako aplikacijo poslati v Apple Store. Za to seveda aktivno sodelujemo z Microsoftom in sedaj čakamo, da bo stvar speljana z Microsoftove strani tako, da bo objavo mogoče stabilno tudi izvesti.

Ker imamo glede razvoja v .NET MAUI platformi še veliko povedati. Zato bo v naslednjih izdajah Pike sledilo nekaj bolj tehničnih člankov na to temo. Do takrat pa lep pozdrav.

Predlagane novosti CSS za leto 2022

Domen Gričar
MCSD, MCSA, MCT
domen.gricar@kompas-xnet.si

CSS se konstantno spreminja in izboljšuje. Nove funkcionalnosti in načini oblikovanja so na voljo skozi vse leto. Ker pa se hitro spreminja ni vedno takoj implementiran v vseh brskalnikih, nekaterih stvari pa se sploh ne implementira. Nekatere funkcionalnosti se lahko testira tako, da v razvijalskih orodjih vklopimo eksperimentalne funkcije, za nekatere funkcionalnosti pa je treba počakati, da jih brskalnik dokončno implementira. V tem članku bom predstavil nekaj zanimivih novosti, ki so bile predlagane za implementacijo tekom leta 2022.

Ena izmed novosti so kaskadni sloji za določanje prioritete nalaganja slogov. Kaskadni sloji bodo imeli prioriteto pred natančnejšim zapisom, kar pomeni, da bo mogoče natančno definirane selektorje povoziti z manj natančnimi sloji.

```
@layer base-layer {
  div#id {
    color: red;
  }
}
@layer theme-layer {
  div.class {
    color: green;
  }
}
div {
  color: blue;
}
```

V zgornjem primeru višji sloj povozi barvo, čeprav je manj natančno določen (običajno id prepiše class). Selektorji, ki pa niso v slojih pa povozijo selektorje v slojih, kljub manj natančnim opredelitvam. Kaskadni sloji so predvideni že v začetku leta 2022, v Chrome verziji 99 in Firefox verziji 97.

Predlaganih je tudi nekaj izboljšav na področju barv. Na voljo sta novi funkciji color-mix(), ki omogoča, da zmešamo svojo barvo na podlagi drugih in color-contrast(), ki omogoča za brskalnik iz seznama sam izbere barvo z najprimernejšim kontrastom.

```
color: color-mix(in lch, purple 80%, plum 80%)
color: color-contrast(wheat vs tan, sienna, var(--myAccent), #d2691e)
```

Dodana je tudi relativna sintaksa za barve, ki nam omogoča, da prilagodimo obstoječe barve.

```
--bg-color: blue;
background: rgb(from var(--bg-color) r g b / 80%);
```

Še ena funkcionalnost, ki bi bila lahko kmalu implementirana in bi lahko delovala v vseh brskalnikih je pseudoclass :has(). Ker je CSS kaskaden (selektorje se lahko izbira le navzdol), se ne da izbrati starša na podlagi nižje uvrščenega elementa. S pomočjo :has() pa bo mogoče izbrati tudi nadrejeni element na podlagi podrejenega. V primeru je izbran le tisti h2, ki vsebuje element span.

```
h2:has(span) {
}
```

Za prilagajanje vsebine glede na velikost brskalnika lahko uporabimo Viewport enote. Poznamo vh (za višino) in vw (za širino). V nekaterih primerih pa brskalniki narobe preračunavajo enote, zato so predlagali natančnejše enote. Viewport enote za manjše velikosti so svh in svw (ne upoštevajo iskalnikov in navigacij naprav) in za cele zaslone lvh in lvw. Predlagane pa so tudi dinamične enote dvh in dvw, ki se samodejno prilagajajo prikazu. V večini brskalnikov enote še ne delujejo, v Safariju pa so na voljo kot predogled.

Nekaj predlaganih izboljšav je tudi za media queryje, ena izmed njih so zapisi razponov. Trenutno je pri media queryjih treba zapisati vsako lastnost posebej, s pomočjo uporabe logičnih operatorjev pa se lahko zapis skrajša in poenostavi.

```
@media (min-width: 400px) and (max-width: 1000px) {
}

@media (400px <= width <= 1000px) {
}
```

Za prilagajanje vsebine glede na velikost naprav lahko uporabimo media query zapis, ta pa ne pomaga pri preverjanju velikosti nadrejenih elementov. Container query pa omogoča, da oblikujemo elemente na podlagi velikosti in oblike nadrejenih elementov. S pomočjo container queryjev lahko oblikujemo glede na velikost celotnega elementa, glede na velikost v vrstici ali glede na velikost po višini. Preverimo lahko tudi oblikovanje in položaj elementov.

```
@container (inline-size > 20em) {
}
```

Pri media query zapisu moramo pravilno določiti zapise, sicer se med sabo povozijo. Večinoma so ti zapisi enostavni, kadar pa je treba preveriti za več različnih podprtosti (supports), hitro pride do daljši in zapletenih zapisov. Pogoj @when/else, bi lahko znatno skrajšal zapis. Ker pa je bil predlagan konec decembra, je verjetno, da še dolgo ne bo na voljo.

```
@when media(width >= 600px) {
} @else {
}
```

To je seznam le nekaj predlaganih funkcionalnosti, celoten seznam je dostopen na GitHubu na povezavi https://github.com/w3c/csswg-drafts/. Čeprav so nekatere funkcionalnosti zelo zanimive in lahko precej olajšajo delo, bo treba počakati na njihovo implementacijo. Tudi, če bodo potrjene, pa ni nujno, da bodo delovale v vseh brskalnikih.

Aleš Lipušček
MCP, MCTS, MCITP
ales.lipuscek@kompas-xnet.si

Powershell kotiček

Tokrat bomo omenili par dobrih praks, ki jih razvijalci sicer dobro poznajo, ne pa nujno tudi tisti, katerim je PowerShell eno od osnovnih orodij.

Če le imamo čas, to je, če ravno ne rešujemo krizne situacije, si je vredno vzeti čas in premisliti pristop k problemu. Nato rešitev realiziramo v obliki psevdo kode oz. komentarjev, ki jih potem zamenjamo z ustreznimi koraki/postopki v PowerShellu.

Če se le da, ne izumljajmo tople vode. Premislimo, če smo že sami reševali/kodirali katerega od postopkov, pregledati se splača tudi module powershell skupnosti. Nato se lotimo kodiranja sami a imejmo ob tem v mislih možnost ponovne uporabe tega, kar pišemo. Stavke, katerih pomen ni očiten na prvi pogled (tj. npr. $a=1 :P) opremimo s kratkimi, a smiselnimi komentarji.

Večkrat uporabljene dele kode zapakirajmo v funkcije in nato kličimo le-te, namesto da kopiramo/lepimo vsepovprek. Tako kopirana koda ima namreč grdo navado, da rabi enega ali več popravkov oz nadgradnjo in potem izgubljamo čas in živce z iskanjem vseh pojavnosti le-te. Funkcije naj uporabljajo parametre za vse vrednosti, ki bi jih sicer "zapekli". Pri tem uporabljajmo validacijo in kontrolo vhodnih vrednosti in skušajmo v funkcijo ne spustiti vrednosti, ki je nismo pričakovali. Določimo tudi tip vhodnim parametrom (in po možnosti tudi izhodne "vrednosti" naše funkcije (tj OutputType)). Funkcije naj bodo eno opravilne in naj služijo enemu samemu namenu. Vsekakor pa ni narobe, če podpirajo pipo. V večjih projektih zna priti prav, če jih uredimo po abecednem redu, saj jih bomo tako lažje našli.

Interaktivne, večkrat uporabljene funkcije pa zmeraj shranimo v PowerShell profil.

Vedno poskrbimo, da nam morebitne napake ne bodo nekontrolirano "bezljale" po našem okolju. Ujeti moramo prav vsako od njih in poskrbeti za odpravo problema ali pa izvajanje skripte zaključiti na dostojen način. Kot vemo, Powershell pozna dva tipa napak, od katerih "lažji" tip napake sproducira samo kopico rdečega teksta na zaslonu a izvajanje se kljub temu nadaljuje. Najbolje je, da se na ta tip ne zanašamo in če se le da, sforsiramo pojav "trdih" napak, saj le te lažje polovimo/kontroliramo. Le to dosežemo z ErrorAction parametrom ali globalno avtomatično spremenljivko $errorActionPreference. Sledenje toku poteka naše skripte in lovljenje nepredvidenih napak je pri večjih skriptah podvig zase, zato si pri tem pomagamo z logiranjem aktivnosti v datoteko in/ali več tokovi, ki so nam pri tem na voljo: debug, verbose, information in progress. Write-progress je večinoma namenjen končnemu uporabniku, da mu vizualno pokažemo, da skripta še dela, še posebej, če izvajanje zahteva več časa, uporabnik pa nima nobenega drugega indica kaj se s skripto dogaja. Z uporabo write-information toka oznanjamo osnovna statusna sporočila o aktivnosti skripte, write-verbose lahko uporabimo za podrobnejše razčlenitve (na primer vstope v if/then bloke ali pa začetke in konce funkcij, vrednosti spremenljivk na določenih mestih izvajanja).

Največjo uslugo si naredimo, ko začnemo spremenljivkam dajati imena, ki v kontekstu skripte nekaj pomenijo. Tako bo koda bolj berljiva pa tudi v primeru nujne intervencije/popravkov bomo iz smiselnih imen lažje razbrali, kaj se v določenem primeru z vrednostmi dogaja.
Vedno hranimo konfiguracijske vrednosti kot so uporabniška imena, imena računalnikov, spletni naslovi, api ključi…ločeno od kode pa naj si bo to v obliki JSON, xml, csv ali pa sql baza.

Izogibajmo se uporabi aliasov v skriptah saj zmanjšujejo preglednost kode, hkrati pa se s tem izognemo neljubim primerom, ko okrajšave na različnih sistemih pomenijo različne stvari.
Regularni izrazi so čudovito orodje dokler je ideja, kaj in na kakšen način z njimi rešujemo še sveža. Ko sestavimo kakšnega kompleksnega mu vedno dodajmo še vrstico komentarja, v kateri kasnejšemu sebi ali drugim razložimo, kakšne ujemajoče se nize od njega pričakujemo.

Opustimo uporabo privzete spremenljivke $?, saj vse prej kot pripomore k berljivosti kode. Marsikdo bo, ko bo prebral članek do tu, uporabil google, da si osveži znanje, saj njeno ime ni intuitivno.
Izognimo se uporabi Invoke-expression ukaza, če nimamo stoprocentne kontrole nad njenim vhodnim parametrom, saj bo PowerShell izvedel karkoli se nahaja v nizu v trenutnem kontekstu, pa čeprav je nek »šaljivec« vanj vpisal npr. »Remove-Item -Path 'C:\' -Recurse -Force«

Ne shranjujmo občutljivih informacij, posebej še gesel, zasebnih ključev, api ključev v berljivi obliki direktno v skripti. Če ne drugega, jih shranimo in beremo z export/import-clixml, ki jih spotoma en/dekriptira.
Ko je skripta končana ali pa popravljena po nekem obdobju uporabe, vedno preverimo, če se v njej nahaja del kode, ki ni bila in ne bo nikdar uporabljena, in le to odstranimo.
S tem seveda še nismo izčrpali vseh možnosti (kot recimo vgrajene pomoči, kriptografsko podpisovanje skript, idr) a več o tem v prihodnjih člankih.

Moj prvi Microsoft certifikat

Petra Militarev
Vodja izobraževanj
petra.militarev@kompas-xnet.si

Pred nami je novo leto in z njim sveži izzivi in načrti za odličen uspeh, okrepitev strokovnih kompetenc ter višje znanje. Da organizacije na svoji poti k uspešnosti potrebujejo tudi izobraževalne rešitve, ki zagotavljajo pravočasen, ugoden in ciljno usmerjen prenos znanja, ni potrebe poudarjati. Veseli me, da se številna podjetja strateško zavedate pomembnosti učenja in da vam vlaganje v znanje in v razvoj kompetenc zaposlenih, tudi na področju informacijskih tehnologij, postaja del kulture podjetja.

V preteklem letu smo s številnimi podjetji uspešno sodelovali, pisali dobre zgodbe in se vam na tem mestu tudi iskreno zahvaljujem za zaupanje. V skladu z vašimi projekti in vizijo smo skupaj načrtovali potrebna znanja, pripravili ustrezne učne poti, izpeljali usposabljanja in vam pomagali pri dosegi posameznih Microsoft certifikacij. In s sodelovanji nadaljujemo.

Kaj pričakovati…

O tem, kako je Microsoft krepko prevetril tehnološke rešitve in temu dodal nova usposabljanja, nove izpite in certifikacije, smo že pisali. Letos bodo na našem IT izobraževalnem programu v fokusu aktualna področja:

- Microsoft 365
- Power Platform
- Security, Compliance and Identity
- Dynamics 365
- Azure

Vsako področje vključuje spekter usposabljanj, od osnov do experta in specializacije.

Sklop Microsoft 365:

- Microsoft 365 Fundamental (MS-900)
- Microsoft 365 Modern Desktop Administrator (MD-100, MD-101)
- Microsoft 365 Messaging Administrator (MS-203)
- Microsoft 365 Security Administrator (MS-500)
- Microsoft 365 Developer (MS-600)
- Microsoft 365 Teams Administrator (MS-700)
- Microsoft 365 Enterprise Administrator (MS-100, MS-101)
- Microsoft 365 Teams Voice Engineer (MS-720)

Sklop Power platform:

- Power Platform Fundamentals (PL-900)
- Microsoft Power Platform App Maker (PL-100)
- Microsoft Power Platform Functional Consultant (PL-200)
- Microsoft Power Platform Developer (PL-400)
- Data Analyst (DA-100)
- Microsoft Power Platform Solution Architect (PL-600)

Sklop Microsoft Security, Compliance, and Identity:

- Security, Compliance, and Identity Fundamentals (SC-900)
- Azure Security Engineer (AZ-500)
- Microsoft 365 Security Administrator (MS-500)
- Information Protection Administrator (SC-400)
- Identity and Access Administrator (SC-300)
- Security Operations Analyst (SC-200)

Sklop Dynamics 365:

Customer engagement apps

- Dynamics 365 Fundamentals (CRM) (MB-910
- Dynamics 365 Sales Functional Consultant (PL-200, MB-210)
- Dynamics 365 Marketing Functional Consultant (PL-200, MB-220)
- Dynamics 365 Customer Service Functional Consultant (PL-200, MB-230)
- Dynamics 365 Field Service Functional Consultant (PL-200, MB-240)

Sklop Azure:

- Azure Fundamentals (AZ-900)
- Azure Data Fundamentals (DP-900)
- Azure AI Fundamentals (AI-900)
- Azure Administrator (AZ-104)
- Azure Security Engineer (AZ-500)
- Azure Stack Hub Operator (AZ-600)
- Azure Developer (AZ-204)
- Azure Database Administrator (DP-300)
- Azure Data Engineer (DP-203)
- Data Analyst (DA-100)
- Azure AI Engineer (AI-102)
- Azure Data Scientist (DP-100)
- Azure Network Engineer (AZ-700)
- Windows Server Hybrid Administrator (AZ-800 + AZ-801)
- Azure Solutions Architect (AZ-303 + AZ-304/ AZ305)
- DevOps Engineer (AZ-400)
- Azure Virtual Desktop (AZ-140)

Na programu ostaja sklop izobraževanj s področja SharePoint (Online in On-premise). Dodali pa smo že tudi nove seminarje za razvijalce ali administratorje baz podatkov ter razvijalce aplikacij in sistemov za poslovno inteligenco:

- Napredno analiziranje podatkov z jezikom Transact-SQL
- SQL Server onstran klasične uporabe
- ZagRabite R
- Python za analizo podatkov
- Notranjost baz podatkov, optimizacija poizvedb in transakcije na strežniku SQL Server
- Varnost v strežniku SQL Server
- SQL Server in Azure SQL Database za začetnike (razvijalci, ki se ukvarjajo s podatki

Moj prvi Microsoft certifikat

Letos bomo namenili posebno pozornost tudi opravljanju izpitov in dosegi ustreznih certifikacij. Zakaj ne bi, v kolikor se že usposobite, svoje znanje potrdili tudi z opravljenim izpitom? Tisti, ki niste še nikoli pristopili k opravljanju Microsoft izpitov in niste prepričani kaj pričakovati, se lahko na povezavi seznanite z vrstami vprašanj, ki jih vidite na izpitu in kako krmariti po uporabniškem vmesniku (čeprav vprašanja niso prava, gre za občutek izpitov, tako da se lahko z njim seznanite, preden opravljate izpit).

Skozi leto bomo za vas pripravljali različne kampanje, ki vas bodo opolnomočile in tudi opogumile k opravljanju izpitov. Ne odlašajte. Pridružite se nam lahko že na prvih:

- Postanite Windows Server Hybrid Administrator Associate
- Pridobite si naziv Microsoft 365 Certified: Modern Desktop Administrator Associate

Naša ekipa ima pripravljeno vrsto rešitev, ki udeležencem povečajo učinek izobraževanja. Priporočamo, da se povežemo čimprej! Za več informacij ali predstavitev in prilagoditev programov, nas prosim pokličite na 01 5136 993 ali nam pišite na info@kompas-xnet.si in z veseljem vam bomo pomagali na vaši razvojni poti.