MAJ 2021

Pika

19. do 20. OKTOBER 2021
Bohinjska Bistrica, Slovenija
THRIVE CONFERENCE
NOVI DATUM KONFERENCE


THRIVE
CONFERENCE
NOVI
DATUM
KONFERENCE
19. do 20.
OKTOBER 2021
BOHINJSKA BISTRICA, SLOVENIJA
11. konferenca
o modernih IT tehnologijah
Zagotovite si svojo vstopnico zdaj!
thriveconf.com

UVODNIK

Spoštovani,

Prav neverjetno! Že polno leto je minilo, odkar nam je pandemija na glavo obrnila vsakdanje navade, življenje, šolanje in delo. Niti v sanjah si nismo mislili, da bo trajalo dlje kot mesec, dva.

Sedaj vsi upamo, da je najhuje že za nami in bomo, cepljeni ali preboleli, lahko začeli živeti bolj običajno življenje, tako, ki ga tako zelo pogrešamo.

Resnično nas veseli, da ste se v jesenskem, pa tudi v tem pomladnem obdobju, z veliko manj pomisleki in zadržki odločali za udeležbo na live-virtual tečajih. Tako so bili posamezni tečaji celo rekordno zasedeni. Nadvse spodbudno je, da so se opogumila tudi podjetja in spodbudila zaposlene, da nadaljujemo z različnimi »office« delavnicami za njihove – končne uporabnike. In digitalna preobrazba se pospešeno nadaljuje.
Hvala vam za zaupanje in prisrčno dobrodošli še na kateri od prihodnjih delavnic.

- 31.5. - 4.6.2021 20767: Implementing a SQL Data Warehouse - Dejan Sarka
- 31.5. - 4.6.2021 20339-1: Planning and Administering SharePoint 2016 - Robi Vončina
- 31.5. - 4.6.2021 MS030: Office 365 Administrator - Miha Pihler
- 14. - 17.6.2021 AZ104: Microsoft Azure Administrator - Jože Markič
- 28.6. - 2.7.2021 20486D: Developing ASP.NET Core MVC Web Applications - Gašper Rupnik

In ne pozabite, šolanje je lahko za vas BREZPLAČNO, če kotizacijo poravnate z SATV voucherji. V Sloveniji je aktivnih še skoraj za 400.000,00 € SATV, katerim grozi, da bodo propadli. Le še do konca junija 2021 jih imate možnost vnovčiti. Posredujte nam jih čimprej, za udeležbo na tečaju je nato čas še do konca letošnjega leta.
Za pojasnila in podrobnosti se nam oglasite info@kompas-xnet.si ali 01/ 5136 990. Ne dovolite, da se vam denar izmuzne!

Še posebna ugodnost:
Če se v času poletnih počitnic udeležite MLP tečaja pri nas, je lahko v tem času vaš šolar BREZPLAČNO na kateri od počitniških delavnic.

Ste razmišljali o tem, da bi na MLP šolanju pridobljeno znanje preverili na izpitu? Dodajte svoji izobrazbi še certifikat, ki je mednarodno uveljavljen in potrjuje vaše IT kompetence. Microsoft je pripravil t.i. »role-based« certifikate, ki dokazujejo vaše tehnično znanje na 4 ključnih področjih: Azure, Microsoft 365, Dynamics 365 ali Power Platform.
Več podrobnosti >

Če ste spregledali ali pa še niste utegnili, naj vas spomnimo, da si še vedno lahko ogledate posnetke poslovnih in tehničnih predavanj ter okrogle mize s konference "Digitalna preobrazba smo ljudje"

Digitalizacija

Poglobljeno tehnično znanje in pa sodobne tehnološke rešitve so ključ do večje učinkovitosti zaposlenih in uspešnega poslovanja vašega podjetja. Portali za sodelovanje, brezpapirno poslovanje, elektronska dokumentacija in e-arhivi brez dvoma prispevajo tudi svoj delček k ohranjanju našega planeta. Zato so naložbe v digitalizacijo in »zeleno« nedvomno koristne in neobhodne.
Ko boste posodabljali svojo IT infrastrukturo in uvajali učinkovite rešitve, nas povabite k sodelovanju, saj vam lahko v marsičem pomagamo. Naši strokovnjaki imajo poglobljeno tehnično znanje in ogromno izkušenj z malih in velikih projektov, pa tudi postavitve velikih multinacionalk jim niso tuje. Svoje izkušnje bodo z veseljem delili z vami, da boste prišli do optimalne rešitve za vaše okolje.

In to tako s funkcionalnega, kot tudi finančnega vidika.

Ste že slišali, da je Dejan Sarka pred kratkim zaključil že svojo 19 knjigo? Za SQL in podatkovna skladišča je res vrhunski strokovnjak in nekaj njegovih knjig se vedno najde med TOP 10 na Amazonu.

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

Keynote predavatelj:
Jeff Teper - Corporate Vice President Teams, OneDrive, SharePoint; z vzdevkom »oče SharePoint-a«
Vljudno vas vabim, da si pogledate program Thrive konference in se čimprej prijavite. Zgodnje prijave vam prinašajo do 20 % popust.
Ne odlašajte s prijavo na najboljši, tehnično – izobraževalni dogodek s tradicijo in mednarodno udeležbo.

Čuvajte se in ostanite zdravi!
Branka Slinkar

Pika

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

Direktorica
Branka Slinkar

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

KOLOFON

MICROSOFT OFFICE

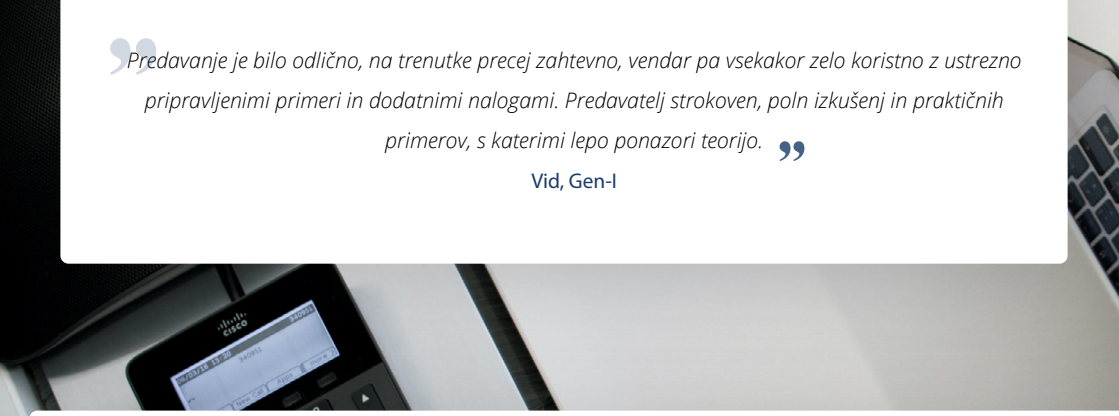
SHAREPOINT

SQL

RAZVOJ

ADMINISTRACIJA

DRUGO



Predavanje je bilo odlično, na trenutke precej zahtevno, vendar pa vsekakor zelo koristno z ustrezno pripravljenimi primeri in dodatnimi nalogami. Predavatelj strokoven, poln izkušenj in praktičnih primerov, s katerimi lepo ponazori teorijo.

Vid, Gen-I

TEČAJI ZA IT STROKOVNJAKE

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

AZ104

Microsoft Azure Administrator

Kdaj: : 14. 6. - 17. 6. 2021
Predava: Jože Markič, MCT

POGLEJ VEČ

20339-1

Planning and Administering SharePoint 2016

Kdaj: : 31. 5. - 4. 6. 2021
Predava: Robi Vončina, MCT

POGLEJ VEČ

DA100

Analyzing Data with Power BI

Kdaj: : 7. 6. - 10. 6. 2021
Predava: Dejan Sarka, MCT

POGLEJ VEČ

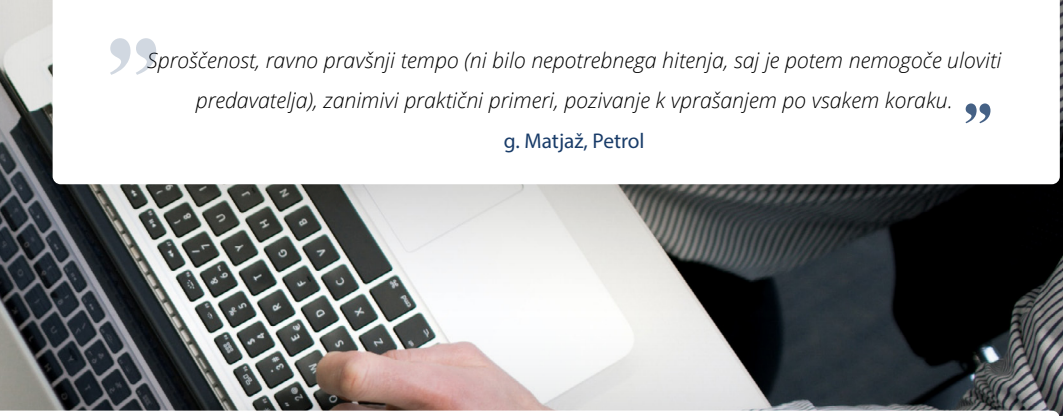
20486D

Developing ASP.NET Core MVC Web Applications

Kdaj: : 28. 6. - 2. 7. 2021
Predava: Gašper Rupnik, MCT

POGLEJ VEČ

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



> *Sproščenost, ravno pravšnji tempo (ni bilo nepotrebnega hitenja, saj je potem nemogoče uloviti predavatelja), zanimivi praktični primeri, pozivanje k vprašanjem po vsakem koraku.*
>
> g. Matjaž, Petrol

TEČAJI ZA UPORABNIKE

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

Uvod v Excel BI

Kdaj: 1. 7. 2021

POGLEJ VEČ

Microsoft Teams

Kdaj: 5. 7. 2021

POGLEJ VEČ

Analize podatkov v Excelu

Kdaj: 26. 5. 2021

POGLEJ VEČ

Office Crash Course

Kdaj: 29. 7. 2021

POGLEJ VEČ

Vrtilne tabele

Kdaj: 30. 7. 2021

POGLEJ VEČ

Office 365 - za uporabnike

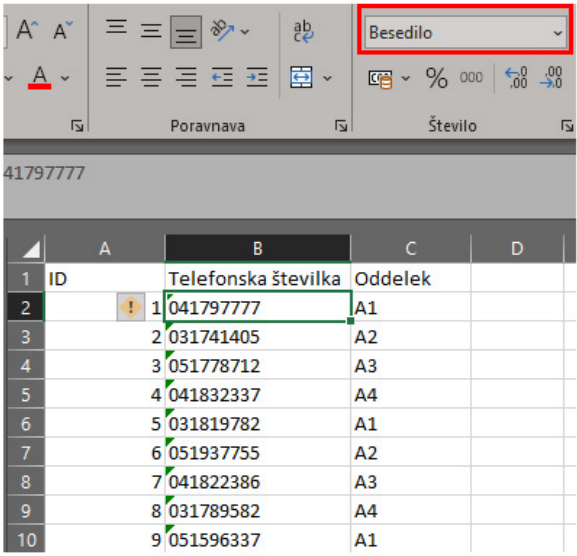
Kdaj: 28. 7. 2021

POGLEJ VEČ

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

Ko se funkcija noče preračunati

Tečajniki mi pogosto postavijo vprašanje, ki se glasi nekako takole: “Ko funkcijo spišem in potrdim, se mi v celici namesto rezultata funkcije pojavi le njeno besedilo. Zakaj?”. Najpogosteje je težava v podatkovnem tipu celice ali kar celega stolpca, v katerega funkcijo vpisujemo. Poglejmo si primer.



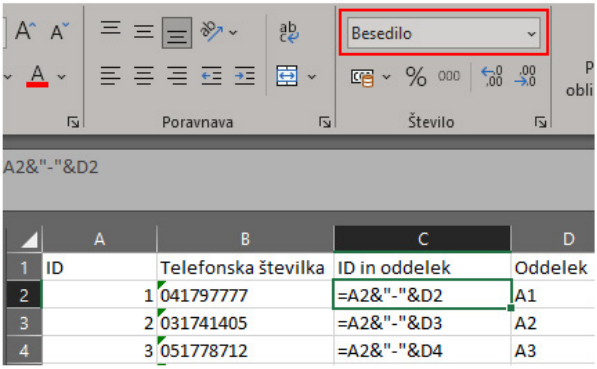
	A	B	C	D
1	ID	Telefonska številka	Oddelek	
2	1	041797777	A1	
3	2	031741405	A2	
4	3	051778712	A3	
5	4	041832337	A4	
6	5	031819782	A1	
7	6	051937755	A2	
8	7	041822386	A3	
9	8	031789582	A4	
10	9	051596337	A1	

V bazi imamo nekaj stolpcev, med njimi tudi stolpec s telefonskimi številkami. Ko smo poštne številke vpisovali, nam ni nikakor uspelo Excel prepričati, da bi si zapomnil tiste vodilne ničle (pri na primer 041, 051 in podobno). Končno smo le našli rešitev – stolpec smo označili in ga oblikovali kot besedilo (te metode sicer ne priporočam, saj nas lahko pripelje ravno do situacije, ki jo opisujem v tem članku, a služi namenu).

Med stolpca "Telefonska številka" in "Oddelek" smo kasneje želeli vpisovati še dodatne informacije, zato smo med ta dva stolpca vrinili nov prazen stolpec. V vrinjenem stolpcu smo želeli izvajati neko funkcijo. Na tem namišljenem primeru naredimo nekaj nesmiselnega, čisto za demonstracijo – poskusimo združiti vsebino stolpcev "ID" in "Oddelek". Kot bomo videli, se funkcija ne bo izvajala.

	A	B	C	D
1	ID	Telefonska številka	ID in oddelek	Oddelek
2	1	041797777	=A2&"-"&D2	A1
3	2	031741405	=A2&"-"&D3	A2
4	3	051778712	=A2&"-"&D4	A3
5	4	041832337	=A2&"-"&D5	A4
6	5	031819782	=A2&"-"&D6	A1
7	6	051937755	=A2&"-"&D7	A2
8	7	041822386	=A2&"-"&D8	A3
9	8	031789582	=A2&"-"&D9	A4

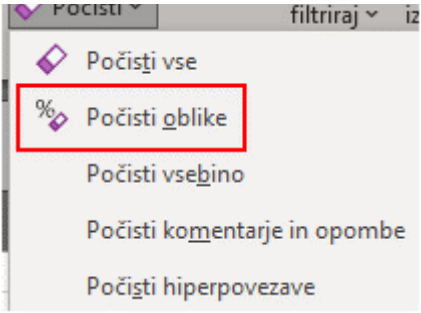
Čemu pride do situacije, ki jo vidimo v stolpcu C. Spomnimo se, da smo stolpec B oblikovali kot besedilo, da Excel vpisanim telefonskim številkam ni pobrisal vodilnih ničel. Ko smo za stolpec B vrinili nov prazen stolpec, je ta novi stolpec prevzel obliko prejšnjega stolpca, torej B.



	A	B	C	D
1	ID	Telefonska številka	ID in oddelek	Oddelek
2	1	041797777	=A2&"-"&D2	A1
3	2	031741405	=A2&"-"&D3	A2
4	3	051778712	=A2&"-"&D4	A3

Kar smo prej s pridom izkoriščali, nam zdaj povzroča težave, saj Excel tudi funkcije ohrani takšne, kot smo jih zapisali in jih ne preračunava. To zagato lahko rešimo v dveh korakih:

1. Stolpec C (ali težavne celice) označimo in z njega pobrišemo oblikovanje (trak Osnovno (Home) – Počisti (Clear) – Počisti oblike (Clear Formats)).



2. Na prvo celico, s težavno funkcijo (v primeru na sliki je to C1) dvokliknemo in funkcijo znova potrdimo s tipko Enter. Funkcijo nato ponovno kopiramo navzdol (dvoklik na spodnji desni vogal celice z delujočo funkcijo). Če je šlo vse po sreči, bi stolpec moral vsebovati delujočo funkcijo.

	A	B	C	D
1	ID	Telefonska številka	ID in oddelek	Oddelek
2	1	041797777	1-A1	A1
3	2	031741405	=A2&"-"&D3	A2
4	3	051778712	=A2&"-"&D4	A3
5	4	041832337	=A2&"-"&D5	A4

C
ID in oddelek
1-A1
2-A2
3-A3
4-A4
5-A1
6-A2
7-A3
8-A4

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

SharePoint 2019 Incoming email

V tokratni številki Pike se po nekaj številkah Pike zopet vračam na On-premise produkt. Predstavil bom, kaj so bile spremembe v verziji SP2019 glede nastavitev za Incoming email.

Uvod

Incoming email ali dohodna pošta je nastavitev v SharePoint On-premise produktu, ki dovoljuje, da nekaterim seznamom ali dokumentnim knjižnicam lahko nastavimo naslov elektronske pošte. S tem dosežemo, da uporabniki lahko pošljejo email na določen naslov in ta nato pristane v knjižnici ali seznamu.

Primeri takih seznamov se uporabljajo v podjetjih različno. V našem primeru, uporabljamo incoming email za nekatero pošto, ki pride od zunanjih pošiljateljev, za lažjo nadaljnjo obdelavo in hranjenje. V drugih podjetjih uporabljajo dohodno pošto za elektronske naslove za monitoriranje sistemov, za klipinge, ki jih dobijo od zunanjih partnerjev, in še bi lahko naštevali…

Nastavitve

Za nastavitev dohodne pošte, potrebno nastaviti:

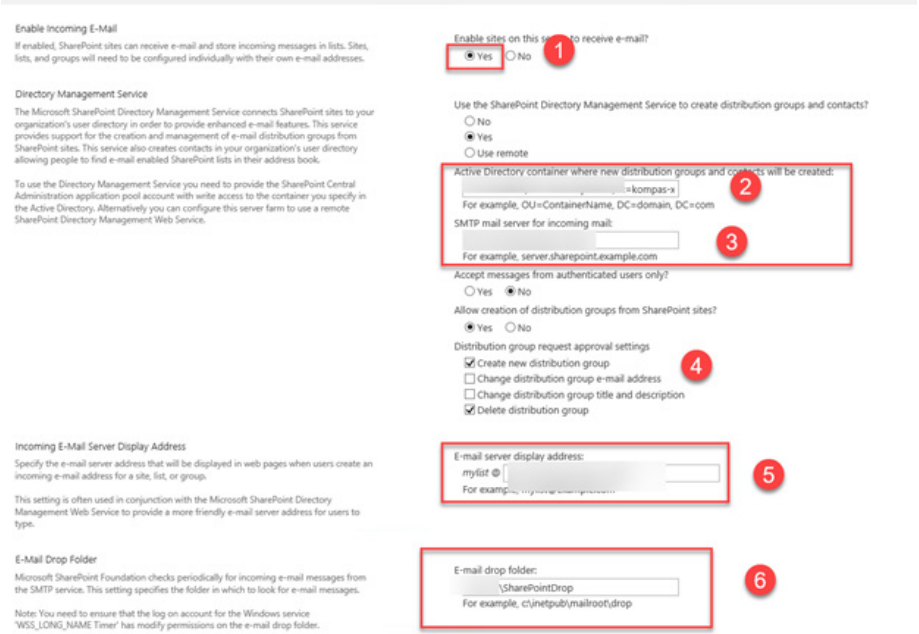
- SharePoint gručo
- Aktivni imenik,
- Poštni strežnik
- Seznam ali knjižnico.

SharePoint

Na SharePoint strežniku je v primerjavi s prejšnjimi različicami, pravzaprav potrebno urediti manj stvari, kot v prejšnjih verzijah, saj so v tej vse odvisnosti od IIS 6.0 odstranjene. To pomeni, da lahko na SP strežniku samo odpremo Centralno administracijo in pod **»System settings«** odpremo nastavitve za **»Incoming email«.**

Na spodnji sliki je predstavljenih 6 nastavitev:

1. Vklopimo nastavitve za dohodno pošto
2. Vpišemo pot do Organizacijske enote v AD-ju, kjer se bodo ustvarili kontakti
3. Vpišemo SMTP mail server za dohodno pošto
4. Določimo, katere akcije so v povezavi z dohodno pošto dovoljene
   a. Npr ustvarjanje distribucijskih skupin
   b. Brisanje distribucijskih,…
5. Nastavitve »domene« za prejemanje pošte v SharePoint
   a. Domena za pošiljanje pošte mora biti druga kot je primarna domena za elektronsko pošto.
   b. Če je domena v našem primeru kompas-xnet.si, potem mora biti »Incoming email« npr. »dohodna.kompas-xnet.si«
6. Email drop folder > mapa od koder SharePoint pošto »pobere« in jo zapiše v ustrezen seznam.



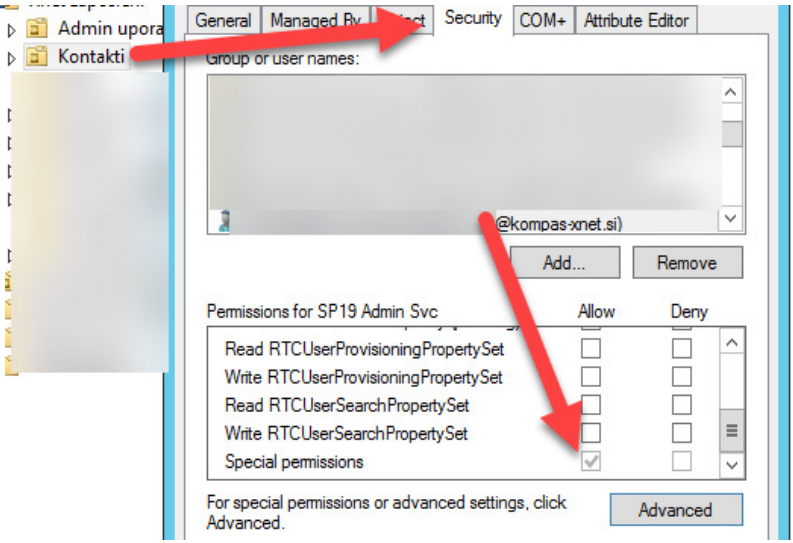
Aktivni imenik

V aktivnem imeniku, je dobra praksa ustvariti nov OU za ustvarjanje kontaktov. Kontakti so ustvarjeni s strani SharePoint Timer Service-a vsakič, ko seznamu ali knjižnici določimo email naslov. Iz tega naslova, moramo tudi dodeliti ustrezne pravice na tem OU-ju.

Da bi lahko vedeli kater račun potrebuje pravice, je najlažje na SP strežniku preveriti, pod katerim računom teče »SharePoint Timer Service (OWSTimer.exe, SPTimerv4)«

SharePoint Insights	Microsoft S...		Disabled	Local Syste...
SharePoint Search Host Controller	Performs h...	Running	Automatic	KOMPAS-X...
SharePoint Server Search 16	Administers...	Running	Automatic	KOMPAS-X...
SharePoint Timer Service	Sends notifi...	Running	Automatic	KOMPAS-X...
SharePoint Tracing Service	Manages tr...	Running	Automatic	Local Service
SharePoint User Code Host	Executes us...		Disabled	KOMPAS-X...

Račun mora imeti delegirane pravice na organizacijsko enoto in sicer za kreiranje, urejanje in brisanje vseh »child« objektov.



Poštni strežnik

V našem primeru uporabljamo Exchange Server in ker je bila na SP2019 odstranjena odvisnost od IIS6, to pomeni, da ne moremo pošte več preusmerjati na pametnega gostitelja, temveč je potrebno na Exchange-u nastaviti »Foreign connector«.

Foreign connector lahko nastavimo samo prek PowerShell ukazov po naslednjem postopku:

New-ForeignConnector -Name "SharePoint Mail" -AddressSpaces "SMTP:dohodna.kompas-xnet.si" -SourceTransportServers "mailserver"

S tem ukazom ustvarimo nov connector za SharePoint mail domeno

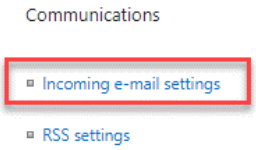
Set-ForeignConnector -Name "SharePoint Mail" -DropDirectory "e:\SpDropMail"

Tu nastavimo kam naj se pošta odlaga.

Zadnji korak na poštnem strežniku je še, da mapo kamor se odlagajo poštne datoteke delimo, in sicer zopet mora imeti tukaj pravice »SharePoint Timer service« račun.

SharePoint seznam ali knjižnica

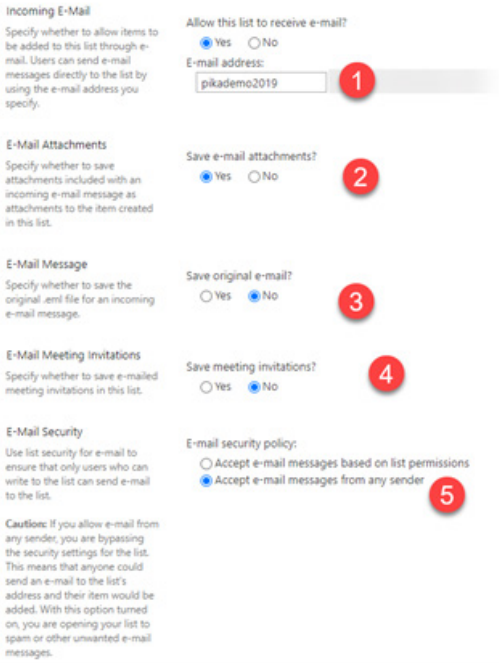
Zadnji korak za prejemanje pošte je, da na SharePoint-u ustvarimo seznam. Meni najljubši so v tem primeru seznami za obvestila, saj se, za funkcionalnost dohodne pošte, obnašajo najbližje nabiralnikom v poštnih klientih.



V nastavitvah novo ustvarjenega seznama imamo nastavitve za dohodno pošto.

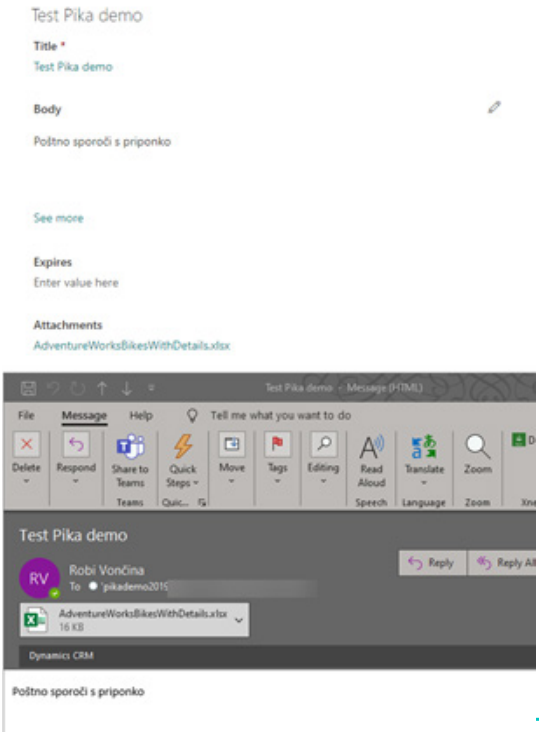
Odprejo se nam bolj podrobne možnosti za nastavitev kaj točno želimo s prejeti sporočili storiti:

1. določimo naslov elektronske pošte za dotični seznam
2. Določimo ali želimo shraniti poslane priloge
3. Določimo ali želimo shraniti tudi originalno email sporočilo
4. Določimo ali želim shraniti tudi vabila na srečanja
5. Določimo kdo lahko pošilja elektronsko pošto na ta seznam, ali se upoštevajo pravice za seznam ali pa se pošta lahko prejme od kogar koli



Če nastavitve delujejo, lahko sedaj naredimo tudi test pošiljanje pošte in v seznam prejmemo elektronsko sporočilo.

S tem so tudi zaključene nastavitve za SharePoint dohodno pošto za on-premise. V naslednji številki, pa pogledamo, kaj bi lahko naredili za enako funkcionalnost a SharePoint Online.



SQL Server Security Part 13: Dynamic Data Masking

Dejan Sarka
MVP, MCT
dsarka@solidq.com

With SQL Server 2016, the Dynamic Data Masking (DDM) feature, you have an additional tool that helps you limit the exposure of sensitive data by masking it to non-privileged users. The masking is done on the SQL Server side, and thus you don't need to implement any changes to applications so they can start using it.

Introduction to Dynamic Data Masking

You define dynamic data masking at the column level. You can obfuscate values from a column in a table by using four different masking functions:

• The default function implements full masking. The mask depends on the data type of the column. A string is masked by changing each character of a string to X. Numeric values are masked to zero. Date and time data type values are masked to 01.01.2000 00:00:00.0000000 (without double quotes). Binary data is masked to a single byte of ASCII value 0.

• The email function masks strings that represent e-mail addresses in the form: aXXX@XXXX.com.

• The random function masks numeric values to a random value in a specified range.

• The partial function uses a custom string to mask character data. You can skip masking some characters at the beginning of the string (prefix) or at the end of the string (suffix).

You must give the users the UNMASK database level permission if you want them to see unmasked data.

For a demo, I will create the same database I used in my previous article, where I described the data classification.

```
USE master;
IF DB_ID(N'DDMDemo') IS NULL
CREATE DATABASE DDMDemo;
GO
USE DDMDemo;
```

The following code creates and populates a demo table using the SELECT INTO statement. It uses the employees from the WideWorldImporters demo database, and adds a randomized salary:

```
SELECT PersonID, FullName, EmailAddress,
        CAST(JSON_VALUE(CustomFields,
'$.HireDate') AS DATE)
 AS HireDate,
 CAST(RAND(CHECKSUM(NEWID()) % 100000
+ PersonID) * 50000 AS INT) + 20000
 AS Salary
INTO dbo.Employees
FROM WideWorldImporters.Application.
People
WHERE IsEmployee = 1;
```

In order to show how DDM works, I need first two users.

```
CREATE USER SalesUser1 WITHOUT LOGIN;
CREATE USER SalesUser2 WITHOUT LOGIN;
```

You must grant the SELECT permission on this table to the two database users, with the help of the following code:

```
GRANT SELECT ON dbo.Employees
TO SalesUser1, SalesUser2;
```

If you execute the following queries, you can see that you, as the dbo user, and both database users you created, can see all of the data:

```
SELECT * FROM dbo.Employees;
EXECUTE (N'SELECT * FROM dbo.Employees')
AS USER = N'SalesUser1';
EXECUTE (N'SELECT * FROM dbo.Employees')
AS USER = N'SalesUser2';
```

Here is the partial result of one of the three previous queries:

```
PersonID  FullName          EmailAddress
HireDate  Salary
--------  --------------    ------------------------------
----------  ------
2         Kayla Woodcock    kaylaw@wideworldim-
porters.com  2008-04-19 45823
3         Hudson Onslow     hudsono@wide-
worldimporters.com 2012-03-05 39344
```

Defining Dynamic Data Masking

The following code adds masking to the columns:

```
ALTER TABLE dbo.Employees ALTER COLUMN
EmailAddress
 ADD MASKED WITH (FUNCTION = 'email()');
ALTER TABLE dbo.Employees ALTER COLUMN
HireDate
 ADD MASKED WITH (FUNCTION = 'default()');
ALTER TABLE dbo.Employees ALTER COLUMN
FullName
 ADD MASKED WITH (FUNCTION = 'partial(1,
"&&&&&", 3)');
ALTER TABLE dbo.Employees ALTER COLUMN
Salary
 ADD MASKED WITH (FUNCTION = 'random(1,
100000)');
GO
```

Now we will try to read the data as one of the regular users, using the following code:

```
EXECUTE (N'SELECT * FROM dbo.Employees')
AS USER = N'SalesUser1';
```

The result for this user is masked, as shown here:

```
PersonID  FullName          EmailAddress
HireDate  Salary
--------  --------------    ------------------------------
----------  ------
2         K&&&&&ock         kXXX@XXXX.com
1900-01-01  57709
3         H&&&&&low         hXXX@XXXX.com
1900-01-01  44627
```

Note that you might get different values for the salary because this column uses the random masking function. Now you can grant the UNMASK permission to the SalesUser1 user, and try to read the data again. This time, the result is unmasked:

```
GRANT UNMASK TO SalesUser1;
EXECUTE (N'SELECT * FROM dbo.Employees')
AS USER = N'SalesUser1';
```

Conclusion

With DDM, I am getting close to the end of the SQL Server security features. In the next article, I will show the limitations of DDM, and what is coming in the next version of SQL Server in this area.

Andraž Bergant
andraz.bergant@kompas-xnet.si

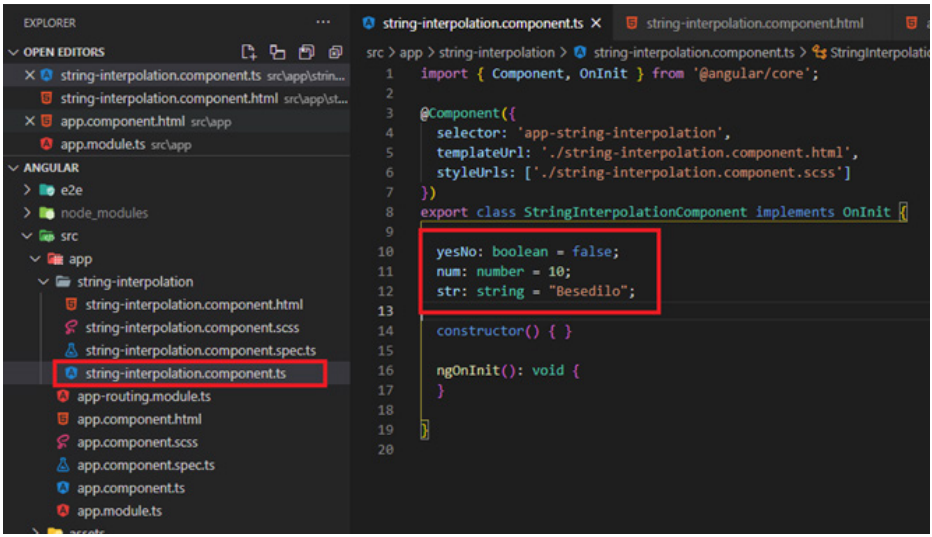
ANGULAR DATA BINDING

Da lahko začnemo uporabljati Angular framework moramo najprej naložiti node.js (https://nodejs.org/en/) in IDE po lastni izbiri (v mojem primeru VS Code (https://code.visualstudio.com/)). Ko uspešno naložimo node.js in izbrani IDE lahko z ukazom npm install -g @angular/cli naložimo še Angular CLI (command-line interface), ki nam bo pomagal pri samem kreiranju projekta, komponent, itd. Projekt kreiramo z ukazom ng new Angular in sledimo čarovniku. Za kreiranje nove komponente uporabimo ukaz ng g c ime-komponente. Kjer je g okrajšava za generate in c za component. CLI komponento avtomatsko doda v app.module v declarations array.

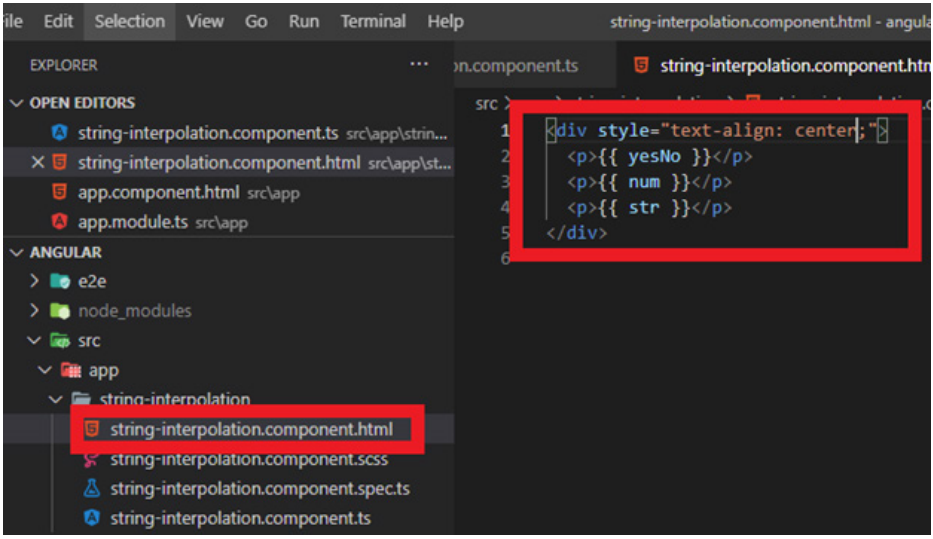
Angular pozna 4 tipe data bindinga in to so String Interpolation, Property Binding, Event Binding in Two-Way Data Binding.

String Interpolation spada pod enosmerne tipe data bindinga, kjer se variable nahajajo med dvema zavitima oklepajema {{ primer }}. Angular potem spremeni spremenljivko primer v string, ki ima vrednost variable primer.

V spodnjem primeru smo zagnali ukaz ng g c string-interpolation in CLI nam je avtomatsko generiral datoteko string-interpolation. Najprej smo v komponento zapisali (slika 1) nato pa v template (slika 2)



Slika 1

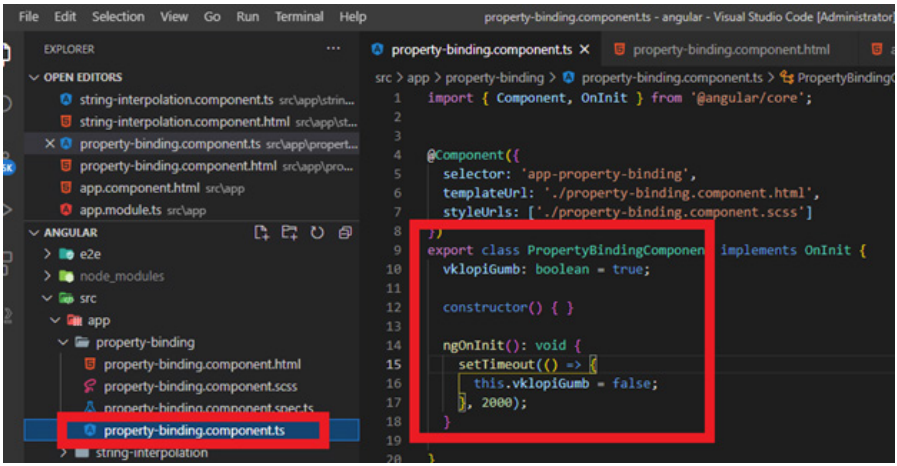


Slika 2

Ko zaženemo aplikacijo z ukazom ng serve, lahko v brskalniku vidimo, kje vse so izpisane vrednosti string!

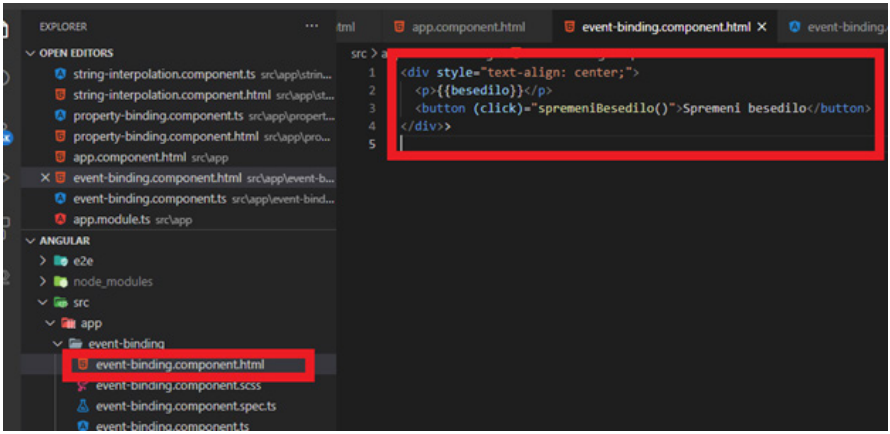
```
▼<app-string-interpolation _ngcontent-niq-c17 _nghost-niq-c16>
  ▼<div _ngcontent-niq-c16 style="text-align: center;">
      <p _ngcontent-niq-c16>false</p>
      <p _ngcontent-niq-c16>10</p>
      <p _ngcontent-niq-c16>Besedilo</p>
    </div>
  </app-string-interpolation>
```

Property Binding je, poenostavljeno rečeno, posodabljanje vrednosti spremenljivke v komponenti (model), ki se prikaže v viewu. Prav tako kot String Interpolation spada pod enosmerne tipe data bindinga. Največja razlika med njima je to, da se String Interpolation uporablja samo s stringi medtem, ko nam Property Binding omogoča, da npr nastavimo lastnost gumba iz disabled=true na disabled=-false pri tem pa ostaneta true in false kot boolean in ne kot string kot smo videli v prvem primeru. Najbolje, da si to pogledamo kar v kodi.



Slika 3

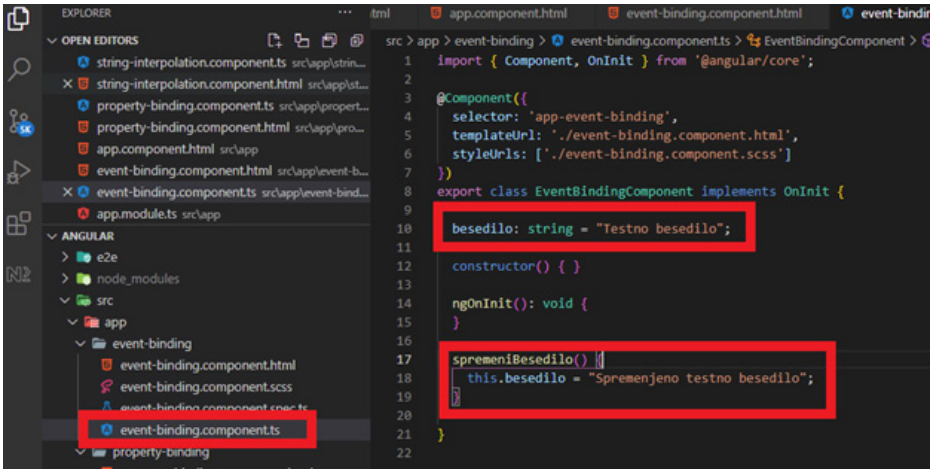
Tako kot pri prvem primeru smo tukaj generirali novo komponento, in jo uporabili v app templateu. Najprej v komponento zapišemo spremnljivko vklopiGumb in jo nastavimo na true, potem v metodo ngOnInit(), zapišemo timeout funkcijo, katera bo čez 2s spremenila vrednost spremenljivke vklopiGumb na false.



Slika 4

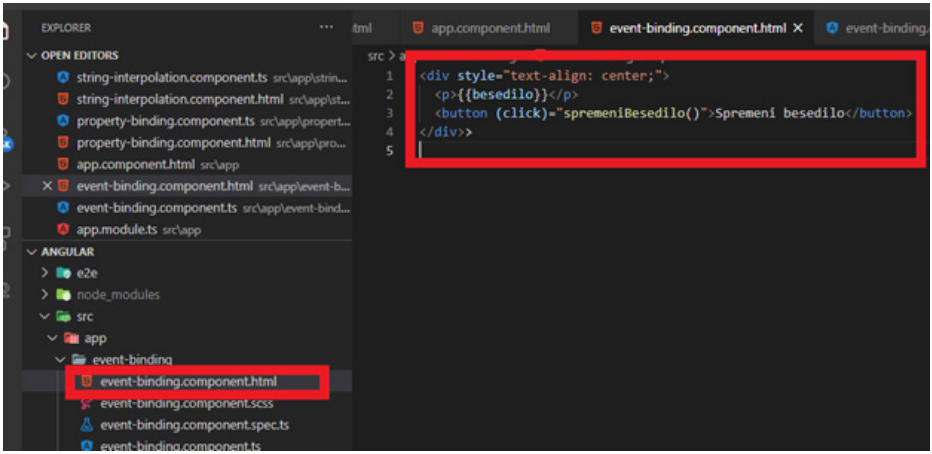
V templateu uporabimo Property Binding tako, da dinamično vežemo HTMLjev atribut disabled na izraz oziroma spremnljivko, katera se nahaja med narekovaji. ! pred spremenljivko true spremeni v false in obratno. Če to zaženete boste ugotovili, da prvi dve sekundi gumb lahko kliknete potem pa ne več.

Event Binding je kot posodabljanje oziroma pošiljanje vrednosti določene spremnljivke iz viewa komponenti (model). Lep primer tega je klik event. Pa si poglejmo na primeru.



Slika 5

Naredimo novo komponento, v kateri zapišemo spremenljivko z neko vrednostjo. Nato napišemo metodo, ki to vrednost prepiše.



Slika 6

V view zapišemo besedilo z uporabo String Interpolation, potem naredimo gumb, pri katerem uporabimo Event Binding. To se v Angularju naredi z navadnimi oklepaji potem pa se v narekovajih ponavadi kliče neko metodo kot naprimer spremeniBesedilo() ta metoda pa spremeni variablo besedilo.

Pred klikom na gumb:

Testno besedilo

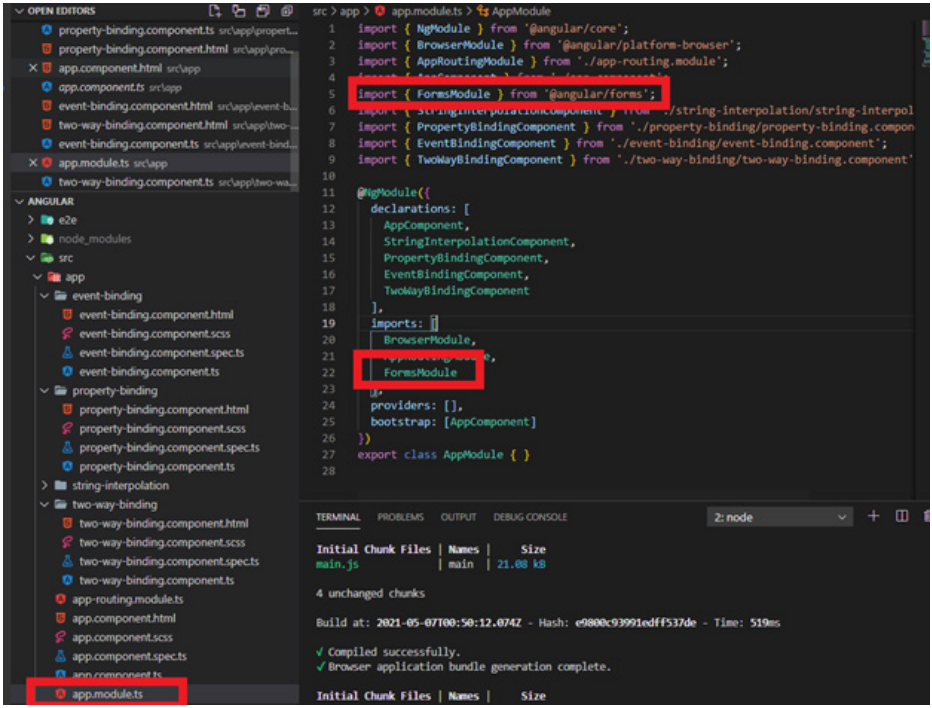
Spremeni besedilo

Po kliku na gumb:

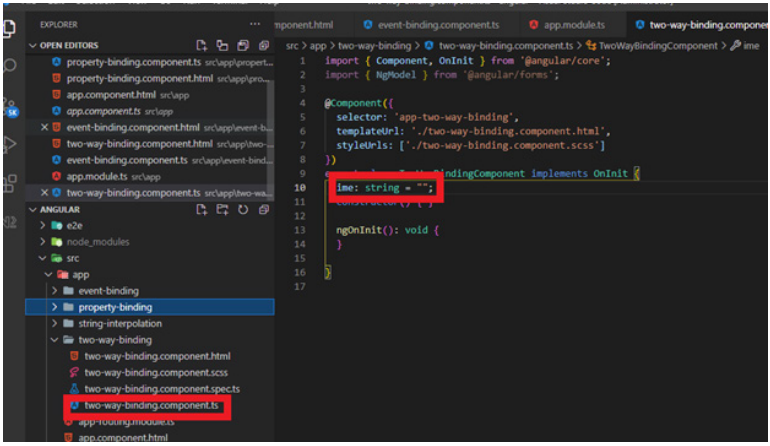
Spremenjeno testno besedilo

Spremeni besedilo

Two-Way Data Binding je kot neka mešanica Property in Event Bindinga. Gre za nenehno sinhroniziranje podatkov iz viewa do komponente in nazaj. To Angular naredi s pomočjo direktive ngModel.ngModel je del Angularejeve knjižnice in je definiran v forms modulu zato moramo importati FormsModule knjižnico v našo app.module.ts datoteko.

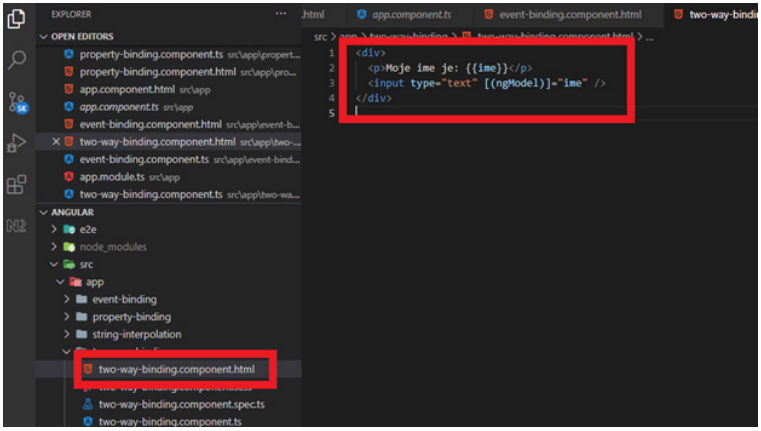


Slika 7



Slika 8

Ko dodamo FormsModule lahko naredimo novo komponento v katero zapišemo spremnljivko ime, ki je tipa string.



Slika 9

V viewu nato uporabimo String Interpolation za izpis spremnljivke ime in pri inputu uporabimo Two-Way Data Binding katerega zapišemo z oglatima in navadnima oklepajema kot [(ngModel)] in potem med narekovaje zapišemo spremenljivko. S tem dosežemo to, da se na začetku v input pošlje vrednost spremenljivke ime (ki je prazen string). Potem pa se vse spremembe, ki jih vpisujemo v input »pošlje« v spremenljivko ime, ta pa se istočasno prikaže zgoraj (z uporabo String Interpolacije). Primer, ko zapišemo v input Janez:

Moje ime je: Janez

Janez

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

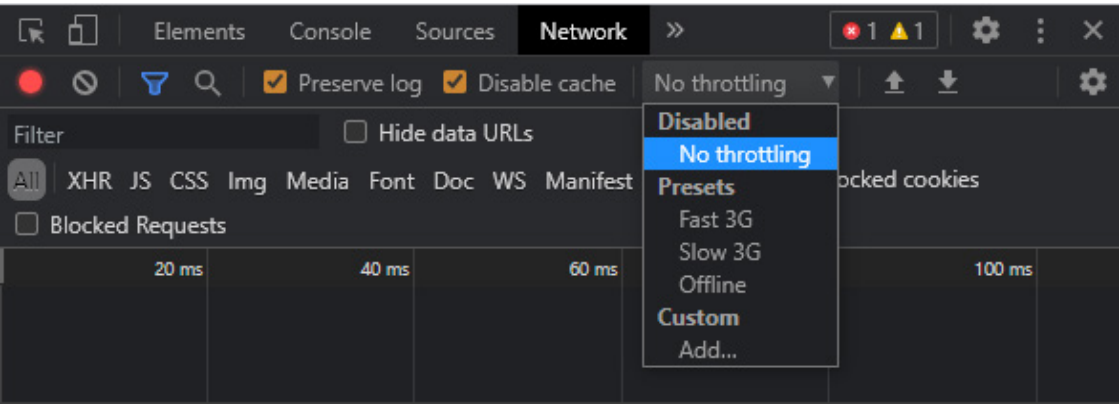
Chrome Orodja za razvijalce - Network

V preteklih člankih sem opisal, kako lahko razvijalska orodja uporabimo za lažje testiranje spletne aplikacije, kako jo oblikujemo in kako testiramo JavaScript funkcije in razhroščujemo kodo. V tem članku si bomo pogledali še, kako preverimo naložene in manjkajoče datoteke, kje in kdaj so se naložile ter koliko časa so se nalagale.

Da preverimo prenašanje datotek uporabimo Network zavihek. Zavihek uporabimo, ko se želimo prepričati, da so se datoteke in sredstva prenesla v brskalnik, kot bi se morala ali da so bile datoteke uspešno poslane na strežnik. Poleg tega si lahko ogledamo lastnosti posamezne datoteke ali klica, kot so čas, HTTP headerje, vsebino in velikost.

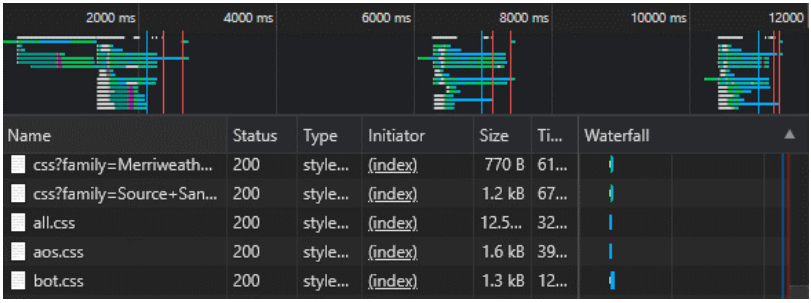
Da se nam prikažejo datoteke, ki so naložene, najprej odpremo razvijalska orodja in zavihek Network, nato ponovno osvežimo spletno stran. Datoteke se bodo zapisovale dokler imamo odprta orodja ali dokler ne ustavimo snemanja.

Pri prikazu rezultatov imamo možnost filtriranja, glede na vrsto datoteke (so prikazani gumbi za izbor, lahko prikažemo vse datoteke, XHR, CSS, JS, slike, medijske datoteke, pisave, dokumente in druge), lahko pa vnesemo filtre po meri, če vpišemo samo besedilo išče po vseh lastnostih, če pa želimo natančneje določiti po kateri lastnosti filtrira, pa lahko zapišemo tudi lastnost (npr. url:https://developer.chrome.com). V primeru, da želimo ohraniti zapise tudi pri večkratni osvežitvi strani si izberemo možnost Preserve log. Nekatere datoteke so shranjene v cachu in če želimo, da se nam zapišejo tudi te, si izberemo možnost Disable cache, kar prepreči caching datotek.

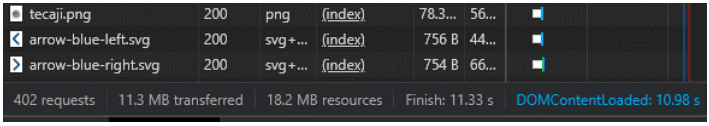


Dodatno lahko preverimo kako se nalagajo datoteke pri počasnejših povezavah. Imamo možnost simulacije 3G, počasnejšega 3G in nalaganje strani brez povezave. Ta funkcionalnost je priročna, ko želimo optimizirati za spletne aplikacije, ki se bodo izvajale na terenu in nimajo dostopa do spleta ali imajo omogočeno le slabo povezavo.

Na vrhu prikaza, pod filtri, imamo časovnico, grafični prikaz nalaganja posameznih datotek. Časovnico lahko približamo ali oddaljimo in izberemo samo en odsek ter s tem prikažemo samo tiste datoteke, ki so v izbranem času.



V spodnjem delu je seznam vseh datotek, ki so se prenesle. Vsaka vrstica predstavlja drugo datoteko. Za vsako datoteko lahko pregledamo lastnosti status (predstavlja http kodo. Pri uspešnem prenosu se izpiše status 200, ostali statusi so med 100 in 199 za informativna obvestila, med 200 in 299 so uspešne klice, med 300 in 399 so za preusmeritve, med 400 in 499 so napake na strani brskalnika in med 500 in 599 so napake na strežniku), tip vira oziroma vrsta datoteke, initiator (kje je bil narejen klic oziroma zahteva za datoteko, to je lahko glavna html datoteka ali skripta, ki zahteva dodatne vire), čas v katerem se je vir naložil zapisan v milisekundah in waterfall – grafični prikaz kako in kdaj se je vir nalagal. Ob kliku na posamezno lastnost se nam prikažejo dodatne informacije, ob kliku na ime datoteke se pokažejo splošni podatki o klicu in odgovoru, ob kliku na initiator nas preusmeri v zavihek Source na vrstico kode, kjer je bil izveden klic in ob kliku na waterfal se pokažejo natančnejše informacije o klicu, kdaj je bil izveden, koliko časa je trajal. Pod seznamom datotek je tudi povzetek, kjer je zapisano število vseh datotek, velikost in koliko časa je trajalo da so se naložile.



Network zavihek je zelo uporaben, ko želimo preveriti nalaganje datotek oziroma ali ko želimo preveriti uspešnost izvajanja klicev. Prav tako je zelo uporaben, ko želimo optimizirati strani za počasnejše povezave in s tem izboljšati uporabnikovo izkušnjo.

Powershell kotiček

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

V PowerShellu poznamo koncept takoimenovanih politik izvajanja (execution policies), s katerimi skušamo zagotoviti bolj varno okolje za administracijska opravila.

Le-te definirajo omejitve, pod katerimi se izvajajo naše skripte. To so Restricted, AllSigned, RemoteSigned in Unrestricted. Powershell se privzeto izvaja na najvarnejši, Restricted, način, torej deluje samo kot interaktivna lupina, ki ne omogoča izvajanja skript. AllSigned dopušča izvajanje skript , a morajo le-te biti podpisane. RemoteSigned dopušča izvajanje skript, pridobljeni prek maila,brskalnikov ipd, Unrestricted pa omogoča neomejeno izvajanje skript.

Vsaka od njih lahko deluje tudi na različnih območjih: MachinePolicy in LocalMachine politiki izvajanja nastavljena z GP za vse uporabnike UserPolicy in CurrentUser za trenutnega uporabnika in Process za trenutni Powershell proces.

Tu se bomo danes ukvarjali z možnostjo, ki jo ponuja AllSigned.

Podpisovanje nam omogoča, da skripti preskrbi identiteto izvora in omogoča detekcijo morebitne zlonamerne spremembe kode.

Digitalno podpisovanje skripte zahteva uporabo ustreznega certifikata za podpisovanje. Tu imamo dve možnosti. Prva je uporaba certifikata znane in zanesljive certifikacijske avtoritete, druga pa kreiranje samopodpisanih certifikatov s strani uporabnika. Če bomo podpisali kodo za interno uporabo, bo tudi samopodpisan certifikat povsem v redu, pa še zastonj je. Seveda pa bo moral biti nameščen na vsakem računalniku, kjer se bo skripta izvajala.

Za samopodpise smo dosedaj uporabljali program Makecert.exe, in je bil del Microsoft .NET Framework SDK oziroma Microsoft Windows Platform SDK. Ponavadi se nahajal v direktoriju C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\.

Po novem pa imamo v PowerShellu cmdlet New-Self-SignedCertificate

PS> New-SelfSignedCertificate -Subject TestPodpisa -Type CodeSigning -CertStoreLocation cert:\CurrentUser\my

Uspešnost kreiranja preverimo z

PS>Get-ChildItem -Path Cert:\CurrentUser\My | ? Subject -EQ "TestPodpisa"

Če sedaj skušamo podpisati skripto

PS>Set-AuthenticodeSignature -FilePath .\skripta.ps1 -Certificate $cert

..pridelamo napako, saj naš certifikat ni zaupanja vreden, saj se nahaja v Personal shrambi.

```
SignerCertificate                         Status                                 Path
-----------------                         ------                                 ----
F45684658A40795DDEB99EC113F462E7A9EF75CB  UnknownError                           skripta.ps1
```

Zato ga z

Move-Item -Path $cert.PSPath -Destination "Cert:\CurrentUser\Root"

Premaknemo v Root shrambo.

Skripta pa je pridobila podpisni blok v obliki

 # SIG # Begin signature block

 # MIIFTwYJKoZIhvcNAQcCoIIFQDCCBTwCAQExCzA-JBgUrDgMCGgUAMGkGCisGAQQB

…..odvisno od tipa enkripcije, a ponavadi ne manj kot 30 vrsrtic :)

 # qzaGtue6oTDofiZOcJcnBoI3MQ==

 # SIG # End signature block

```
PS C:\Users\ales\Documents> Get-AuthenticodeSignature .\skripta.ps1

    Directory: C:\Users\ales\Documents

SignerCertificate                         Status                                 Path
-----------------                         ------                                 ----
F45684658A40795DDEB99EC113F462E7A9EF75CB  Valid                                  skripta.ps1
```

Tomaž Borštnar
CCSI, RHCE, RHCIA, SCE

Virtualizacija

Kontejnerji niso modna muha, temveč pomembno orodje sedanjosti in prihodnosti!

Virtulizacija je že zelo zrela tehnologija, katere korenine segajo v 60. leta prejšnjeg stoletja, ko je bila to pomembna prednost velikih (mainframe) sistemov. S pomočjo virtualizacije so bolj učinkovito uporabili (drage) kapacitete, ki so bile na razpolago. Virtualizacija je postala zelo razširjena tudi v širšem okolju proti koncu prejšnjega stoletja npr. podjetje VMWare je bilo ustanovljeno leta 1998. Že leto kasneje so izdali VMWare Workstation in v 2001 tudi strežniški inačici GSX Server in ESX Server. S tem je postala ta tehnologija široko dostopna vsem vrstam uporabnikom.

Zakaj je bila virtualizacija tako uspešna? Predvsem zaradi izjemno povečane fleksibilnosti ter boljše izrabe strojnih kapacitet. V teh časih nam je samoumevno, da nimamo več toliko računalnikov kot včasih, saj s pomočjo virtualizacije poganjamo praktično vse operacijske sisteme kot navidezne računalnike. Prav tako nam je samoumevno, da uporabljamo posnetke stanja (snapshot), kjer shranimo vmesna stanja teh navideznih računalnikov, saj se tako lahko hitro vrnemo na prejšnje stanje v primeru neuspešne namestitve, posodobitve, ipd. Marsikdo celo misli, da je to varnostna kopija, a temu ni tako, saj je še vedno vezano na konkretni sistem in če nam ta crkne, nimamo ničesar. Super lastnost je možnost premikanja navideznih računalnikov, a ima tudi to v resnici samo delno prednost, saj nam ne pomaga kaj dosti pri zagotavljanju 100% razpoložljivosti.

Virtualizacija nam v večini primerov ni prihranila kaj dosti ali nič pri stroških nakupa/licenciranja programske opreme, prav tako ni kaj dosti pomagala pri zmanjševanju nekaterih operativnih stroškov npr. shranjevanja podatkov. Prihranila nam je prostor v strežniških sobah ter strošek elektrike in delno strošek upravljanja. Zakaj samo delno strošek upravljanja? Pred virtualizacijo smo imeli manjše število namestitev operacijskih sistemov, a so hkrati imeli tudi veliko različnih aplikacij nameščenih. administratorji s(m)o se bali sprememb, saj je lahko vsaka sprememba na operacijskem sistemu ter nameščenih knjižnicah bistveno vplivala na delovanje APLIKACIJ.

V resnici je virtualizacija olajšala samo upravljanje operacijskih sistemov – navideznih računalnikov, bore malo je pripomogla k

upravljanju aplikacij, ki so v resnici »KRUH« vsake resne organizacije. Na produkcijskih sistemih je bila tipično naloga administratorjev, da so skrbeli za operacijski sistem in aplikacije, zato je bila cenjena stabilnost – pred novostmi in prepogosto tudi pred VARNOSTNIMI POPRAVKI. Ker je bila stabilnost tako cenjena, je bil administrator v tem smislu ves čas med kladivom in nakovalom, saj so razvijalci želeli okolje, ki se hitro prilagaja, vodstvo jim je pogosto pritrjevalo, a je hkrati spet pričakovalo praktično 100% razpoložljivost APLIKACIJ. Polna razpoložljivost virtualizacije je relativno enostavna, saj vsi resni sistemi omogočajo prenos živega navideznega računalnika med fizičnimi strežniki. A vse to ne pomeni kaj dosti, ko je treba npr. POSODOBITI operacijski sistem ali aplikacijo.

Sedaj je verjetno večini že jasno, da virtualizacija sama po sebi ne rešuje 100% razpoložljivosti, niti upravljanja aplikacij. Za kaj takega potrebujemo drugačna orodja...

Kontejnerji

Ena od alternativ virtualizaciji so kontejnerji. Za razliko od navideznih računalnikov, ki potrebujejo strojno opremo, kjer teče hipervizor, ki omogoča delovanje navideznih računalnikov, je tu navidezno slabša izolacija, saj je kontejner dejansko skupina enega ali več procesov, ki si delijo jedro (kernel) operacijskega sistema. Skupno jedro pomeni, da kontejner ne rabi svojega navideznega računalnika, zato se lahko zelo hitro zažene in porabi precej manj strojnih kapacitet. Tak kontejner je lahko sistemski kontejner, ki ima v sebi običajne servise operacijskega sistema npr. oddaljeno prijavo, logiranje in podobno; ali pa je aplikacijski kontejner, ki vsebuje samo komponente, ki so nujne za delovanje ene aplikacije.

Zgodovina kontejnerjev

Začetnik kontejnerjev je bil Unix V7, kjer so leta 1979 uvedli možnost omejitve procesa na posamezni imenik – chroot funkcionalnost. Leta 2000 je bilo to bistveno izboljšano v FreeBSD jails mehanizmu, ki je postavil temelje sistemskim in aplikacijskim kontejnerjem. Linux je kmalu (2001) sledil s pomočjo VServer tehnologije ter kasneje tudi Solaris Zones (2004). Te tehnologije so se aktivno razvijale npr. Linux LXC ter končno v 2013, ko smo dobili Docker kontejnerje kot evolucijo vsega tega omenjenega razvoja.

Osebno sem dolga leta pridno uporabljal FreeBSD jails kot zelo dobro metodo ločevanja aplikacij, saj je imel vsak tak kontejner (jail) svoje uporabnike, svoje procese in podatke, ki so si delili isto strojno opremo ter isto jedro (kernel) operacijskega sistema. Za razliko od virtualizacije je bilo tu zelo malo izgube hitrosti aplikacij – od 0 do 2%! Tak sistem se je zelo hitro zagnal in je omogočal, da so bile na sistemu aplikacije, ki se med seboj niso videle in niti motile in so zelo hitro delovale. Edina slabost v primerjavi z virtualizacijo je bila nezmožnost poganjanja drugih operacijskih sistemov na isti strojni opremi, a tudi to je kasneje FreeBSD dodal s funkcionalnostjo Bhyve.

Že omenjeni Docker je leta 2013 izdal svoj sistem za kontejnerizacijo z istim imenom. Osebno nikoli nisem bil veliki ljubitelj njihove programske opreme, saj so me razvadili drugi ponudniki (predvsem FreeBSD in Solaris), a jim je potrebno priznati, da so opravili FANTASTIČNO promocijsko delo, ker so izobrazili ogromno razvijalcev, da je to dober in uporaben način za pakiranje in distribucijo programske opreme! Do tedaj so razvijalci običajno uporabljali aplikacijske pakete (packages) ali celo kar cele imenike shranjene v eni datoteki – običajno ZIP ali TAR. Prednost paketa je bila, da je bilo tako enostavno dodati ali odstraniti aplikacijo, a je bilo vezano na konkretno distribucijo operacijskega sistema in ni bilo enotnosti. Pri imenikih je bila stvar navidezno bolj prenosljiva, a manj praktična.

Zasnova kontejnerskih sistemov

Arhitektura tehnologije je bila enostavna – v osnovi je bil to sistem odjemalec-strežnik, kjer je bil odjemalec enostavna aplikacija, ki je komunicirala s strežnikom - lokalno ali na daljavo. Strežnik je bil velik proces, ki je potreboval veliko pravic na sistemu za delovanja. Strežnik je potreboval kontejnersko sliko (container image), kjer so bile notri vse potrebne datoteke za zagon. To sliko smo lahko dobili preko posebnih distribucijskih kanalov – registrov za kontejnerske slike – container image registry. Seveda so poskrbeli, da je bilo kmalu na voljo ogromno teh slik za uporabnike, da so lahko začeli delati s to tehnologijo.

V praksi je pomenilo, da je lahko kdorkoli začel uporabljati to tehnologijo s pomočjo nekaj ukazov za namestitev Dockerja in zagon kontejnerja neposredno iz Interneta. Ni več bilo potrebno gledati, če imamo pravo distribucijo sistema, prave aplikacije in podobno. V primeru zahtevnejših aplikacij je to BISTVENO poenostavilo in olajšalo uporabo aplikacij, saj je glavna prednost kontejnerske slike ravno to, da je notri VSE POTREBNO za zagon in uporabo. In kako HITER je ta zagon! Če je notri samo aplikacija, ko jo potrebujemo, se lahko kontejner zažene v SEKUNDAH in ne v minutah kot je bolj tipično za virtualke.

Dodatne prednosti kontejnerjev

S čim še je Docker prepričal uporabnike v svojo vizijo? S poenostavljenim upravljanjem omrežnih nastavitev, saj je lahko uporabnik v večini primerov preprosto zagnal kontejner in je aplikacija že bila na voljo znotraj istega sistema. V primeru, da je bilo potrebno kaj več, je v večini primerov zadoščalo odprtje enih TCP/UDP vrat in že je bila aplikacija na voljo tudi drugim uporabnikom na omrežju! Prav tako je bilo enostavno narediti svoja zasebna omrežja za posamezne aplikacije in podobno.

Prav tako je bilo enostavno pripraviti svoje kontejnerske slike – s pomočjo zagona druge slike in shranjenimi spremembami ter preko uporabe že obstoječih slik (base image), ki so bile samo še dopolnjenes pomočjo mehanizma Dockerfile. Dockerfile je ime

datoteke, kjer s pomočjo precej enostavnih ukazov pripravimo svojo kontejnersko sliko – iz nič ali s pomočjo obstoječe slike, ki jo dopolnimo s svojimi spremembami.

Dobre prakse

Na primer, če imamo aplikacijo, ki potrebuje točno določeno distribucijo Linuxa npr. Red Hat Enterprise Linux 7 ali Ubuntu 18 LTS, je dovolj, da poiščemo sliko, ki to že vsebuje in je po možnosti celo uradna slika distribucije (npr. Red Hat Universal Base Image – UBI), saj v tem primeru ni potrebno, da sami skrbimo za varnost te slike, ampak to počne skrbnik, ki to verjetno bolj obvlada kot mi. Naša naloga je samo, da ponovno zgeneriramo svojo sliko, ko vidimo, da je bila izvirna slika posodobljena zaradi varnostnih lukenj! Na ta način si lahko bistveno izboljšamo upravljanje in varnost aplikacij, ki so nam pomembne.

Naslednja pomembna stvar so stare aplikacije zaradi katerih smo bili do sedaj prisiljeni uporabljati navidezne računalnike, kjer je bil pogosto nameščen operacijski sistem, ki se ni posodabljal oz morda nima več podpore za posodobitve. V večini primerov je mnogo boljša opcija, da take aplikacije premaknemo v kontejnerske slike, ki se jih da enostavneje upravljati in zamenjati kot navidezne računalnike, a so hkrati notri SAMO tiste komponente, ki so potrebne za aplikacijo! S tem pomembno zmanjšamo potrebo po strojnih kapacitetah in hkrati bistveno izboljšamo varnost, saj imamo manj ranljivih točk kot pri navideznem računalniku. V takem primeru lahko izkoristimo tudi prednosti novejšega jedra operacijskega sistema – predvsem optimizacije delovanja in varnosti!

Do sedaj sem omenjal predvsem prednosti za razvijalce, zato je prav, da se omenijo tudi prednosti za druge skupine, ki so pomembne.

Katere so prednosti za administratorje in vodstvo?

Že prej sem omenil, da je namestitev zelo enostavna – za Linux kontejnerje potrebujemo Linux z jedrom, ki podpira kontejnerje ter sistem za upravljanje le teh npr. docker, podman, etc. Namestitev je v večini primerov samo en ukaz, zato je začetek res enostaven. Malce bolj zahtevno je, ko imamo zahtevnejšo aplikacijo pravilno segmentirano v več kontejnerjev, ko postane pomemben vrstni red zagona, pravilno omrežje in podobno. Za take primere se super obneseta docker-compose in podman-compose, ki sta namenjena ravno temu, da se pravilni vzpostavi in odstrani okolje za aplikacijo – pravo omrežje, s podporo za shranjevanje pomembnih podatkov izven kontejnerjev ter odprtje vrat za zunanji dostop. S pomočjo docker-compose in podman-compose je življenje zelo enostavno – za zagon uporabimo ukaz UP in za pravilno zaustavitev uporabimo ukaz DOWN. Obe aplikaciji poskrbita tudi za posodobitve slik, če so javno na voljo. Skratka, zelo enostavno in uporabno.

Varnost oz ko ni vse samo rožnato

Na začetku razcveta Docker kontejnerjev je bilo veliko govora o tem, da ta segmentacija ni ustrezno varnostno podprta, a se je sčasoma to precej spremenilo in danes je lahko tak sistem precej bolj varen od marsikaterega fizičnega ali navideznega strežnika, saj uporablja več nivojev zaščite, ki vsi skupaj poskrbijo za boljšo varnost glavnega sistema in samih kontejnerjev. Prva zaščita je že sama segmentacija, saj so aplikacije v kontejnerjih med seboj izolirane. Naslednji nivo je varnostni modul v samem Linux jedru – SELinux ali AppArmor, ki sta namenjena temu, da ujameta/preprečite poskuse pobega izven omejenega okolja, ki je na voljo kontejnerjem. Še vedno pa je možnost izrabe povezave med gostiteljem in kontejnerjem – preko sistemskih klicev (syscalls), ki so način kako lahko aplikacija komunicira z jedrom gostitelja. Naj povem, da je točno to tudi pogosto največji varnostni problem pri virtualizaciji, česar se večina ne zaveda.

Zelo pomemben modul je tudi cgroups oz Control Groups, ki med drugim lahko zagotavlja tudi to, da aplikacije v kontejnerjih ne morejo uporabi več kapacitet kot smo jim jih dodelili! Na ta način lahko omejimo porabo procesorja, pomnilnika, dela z diskom in podobno. To je del Linux jedra že 14 let, a se še vedno premalo uporablja tudi brez kontejnerjev.

Pomembna lastnost uporabe kontejnerskih slik je, da so tipično precej manjše od virtualk, se hitro prenesejo in da zasedajo minimalno prostora, saj v primeru, da poganjamo več kontejnerjev iz istih slik, se ta slika shrani samo enkrat in se potem preslika tolikokrat kot je potrebno. Na ta način se da prihraniti ogromno prostora, saj za poganjanje 100 kontejnerjev iz iste slike, ne potrebujemo tolikokratno količino prostora. Hitri diskovni prostor je zelo drag in ga lahko na ta način ogromno prihranimo.

Prav tako je velikost teh slik zelo majhna – male slike imajo lahko tudi samo 5 MB! Da, pravilno ste prebrali – Alpine Linux slika ima samo 5 MB in vsebuje minimalni sistem, ki že zadošča ogromno aplikacijam! Običajne velikosti slik se gibljejo med 50 in 200MB. Le redko je potrebno več! Če to primerjamo s slikami za navidezne računalnike, kjer je večina teh slikah v GB ali celo desetinah GB in se tipično med seboj ne morejo deliti za večkratno uporabo! Kdor kopuje drage diskovne sisteme, si lahko hitro izračuna, da lahko samo na račun take optimizacije prihranimo precej denarja.

Ves ta čas se nismo dotaknili težave, ko nam en strežnik ne zadošča več. Želimo uporabljati kontejnerje, a ne želimo tega delati ročno, potrebujemo avtomatizacijo, že omenjeno visoko razpoložljivost in še marsikaj drugega. Leta 2014 je Google najavil, da bo izdal odprtokodno verzijo sistema za masovno uporabo kontejnerjev, ki bo baziral na njihovih desetletnih izkušnjah masovne uporabe kontejnerjev. Tako smo dobili Kubernetes (krajše k8s), ki je kmalu postal de facto standard za upravljanje aplikacij in je

danes eno najboljših ogrodij za ta namen! Kot zanimivost naj povem, da so že v istem letu (2014) k projektu pristopili vsi veliki – Red Hat, IBM, Microsoft in drugi. V letu 2015 smo tako dobili Kubernetes 1.0, ki se od takrat še vedno razvija skoraj s svetlobno hitrostjo, saj vsake 3 mesece dobimo novo verzijo! Kubernetes je nekaj posebnega iz večih razlogov:

- je odprtokodni projekt z najširšo podporo
- zasnovan je bil za največje sisteme – ena gruča ima lahko do 5000 strežnikov! A hkrati lahko dela tudi samo na enem!
- podpira mehanizme za preverjanje delovanja aplikacij
- omogoča avtomatsko reševanje težav gruče in aplikacij npr. v primeru izpada aplikacije ali strežnika se ta lahko ponovno zažene nekje druge. Brez ročnega posega!
- omogoča 100% razpoložljivost aplikacije, če je pravilno zasnovana in nastavljena!
- podpira enotne vmesnike za uporabo in upravljanje – API
- omogoča natančno razčlenitev pravic uporabnikom
- uporabniki so lahko ljudje in aplikacije
- odlično podpira DEVOPS/DEVSECOPS prakse npr. 12 Factor Apps
- zasnovan je bil kot splošni sistem za upravljanje aplikacij, zato podpira kontejnerje in tudi več vrst virtualizacije

Ravno to zadnje je tista pomembna lastnost, ki že sedaj bistveno vpliva na IT sektor, saj je čedalje manj potreb po hkratni uporabi večih tehnologij za namen poganja aplikacij. Kubernetes bo pomembno vplival na tržni delež nekaterih že uveljavljenih ponudnikov, če se ne bodo primerno prilagodili novimu izzivalcu.

Skratka, Kubernetes je fantastično okolje, ko prerastemo zmožnosti enega računalnika. Prav tako je pomemben del rešitev, ko moramo preiti na boljši način razvijanja, testiranja in uporabljanja aplikacij. Sem omenil, da je Kubernets prava »cloud native« rešitev? In da je pomemben razbijalec zaprtih sistemov?

A več o tem kdaj drugič...

Tomaž Borštnar je strokovnjak z dolgoletnimi izkušnjami iz IT področja. Sodeloval je pri postavitvi največjih internetnih projektov v Sloveniji – SiOL, Arnes, Voljatel, etc. Bil je član začetnega uredniške odbora revije Monitor. Vodil je oddajo o računalništvu na Radiu Študent. Postavil in ustanovil je sistem Med. Over.Net, ki ga je dolga leta tudi vodil. V zadnjih letih izobražuje doma in v tujini ter dela na zanimivih projektih s sodobnimi tehnologijami. Je tudi uradni predavatelj za tečaje Red Hat, Suse, EnterpriseDB in Cisco ter nosilec številnih IT certifikatov.

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

Sredstva za razvoj kompetenc zaposlenih

SATV - Software Assurance Training Voucher

Uporabniki Microsoft licenčnih pogodb ste imeli ugodnost programa Software Assurance (SA). Žal Microsoft ukinja ugodnost voucherjev (SATV - Software Assurance Training Voucher) s katero ste se lahko brezplačno udeležili številnih MOC izobraževanj. Po 30. juniju 2021 ne bo več možno kreirati voucherjev na izbrane tečaje, kljub temu, da vam na pogodbi morda piše njena veljavnost do leta 2023. V kolikor imate še neuporabljene SATV, obrnite se na našo ekipo in pomagali vam bomo, da jih boste pravočasno aktivirali in izkoristili.

Voucherji so vnovčljivi v našem izobraževalnem centru na MOC tečajih - seznam tečajev z upravičenimi voucherji. 1 dan izobraževanja je 1 voucher/oseba.

Tehnična znanja tako poglobite IT strokovnjaki kot razvijalci, ne da bi to vplivalo na vaš izobraževalni proračun.

Nekaj predlogov:

31.5. - 4.6.2021 Windows Server 2019 Administration - Luka Manojlovič

31.5. - 4.6.2021 Office 365 Administrator - Miha Pihler

28.6. - 2.7.2021 20486 Developing ASP.NET Core MVC Web Applications -Gašper Rupnik

7.6.- 10.6.2021 Analyzing Data with Power – Dejan Sarka

7. – 11.6. 2021 Administering System Center Configuration Manager – Jože Markič

21.6. – 23.6. 2021 SharePoint Online for Administrators – Robi Vončina

ASI - sredstva za razvoj kompetenc starejših zaposlenih

V okviru projekta Celovita podpora podjetjem za aktivno staranje delovne sile – ASI ste lahko podjetja pridobila sredstva za usposabljanja svojih starejših zaposlenih, s poudarkom na usposabljanjih iz digitalnih veščin. Skupna okvirna višina razpoložljivih nepovratnih namenskih sredstev JR ASI 2021 je 4.7 milijona EUR. Namen razpisa je bil spodbuditi podjetja k učinkovitemu upravljanju starejših zaposlenih in krepitvi njihovih kompetenc s ciljem podaljševanja njihove delovne aktivnosti, ob tem, da morajo vključeni starejši zaposleni izpolnjevati starostni pogoj, da so stari 50 let ali več na dan objave razpisa.

V podjetju Kompas Xnet smo strokovnjaki na področju računalniškega izobraževanja. Za vas imamo pripravljena številna usposabljanja na področju digitalnih kompetenc. Pomagali vam bomo definirati cilje usposabljanj in pripravili

vašim potrebam prilagojena izobraževanja.
Nekaj predlogov:
Office 365 - za uporabnike
Microsoft Teams
Uvod v Excel BI
Microsoft Excel začetni

Vavčerji za digitalizacijo

Vavčerji za digitalizacijo so spodbuda za mikro, mala in srednja podjetja z možnostjo **60% sofinanciranja** ključnih področij digitalizacije. Javne pozive za vavčerje za digitalizacijo je objavil Slovenski podjetniški sklad (SPS). Razpisani so bili štirje vavčerji na temo:

Dvig digitalnih kompetenc : Namen vavčerja je spodbuditi ciljne skupine k zagotovitvi ustreznih znanj zaposlenih in vodstvenega kadra za ključna področja digitalizacije.

Digitalni marketing: Namen vavčerja je uvajanje digitalnega marketinga, s čimer se bo povečala njihova konkurenčnost, dodana vrednost oz. prihodki od prodaje.

Priprave digitalne strategije

Namen vavčerja je spodbuditi ciljne skupine k pripravi digitalne strategije z namenom digitalne transformacije podjetij.

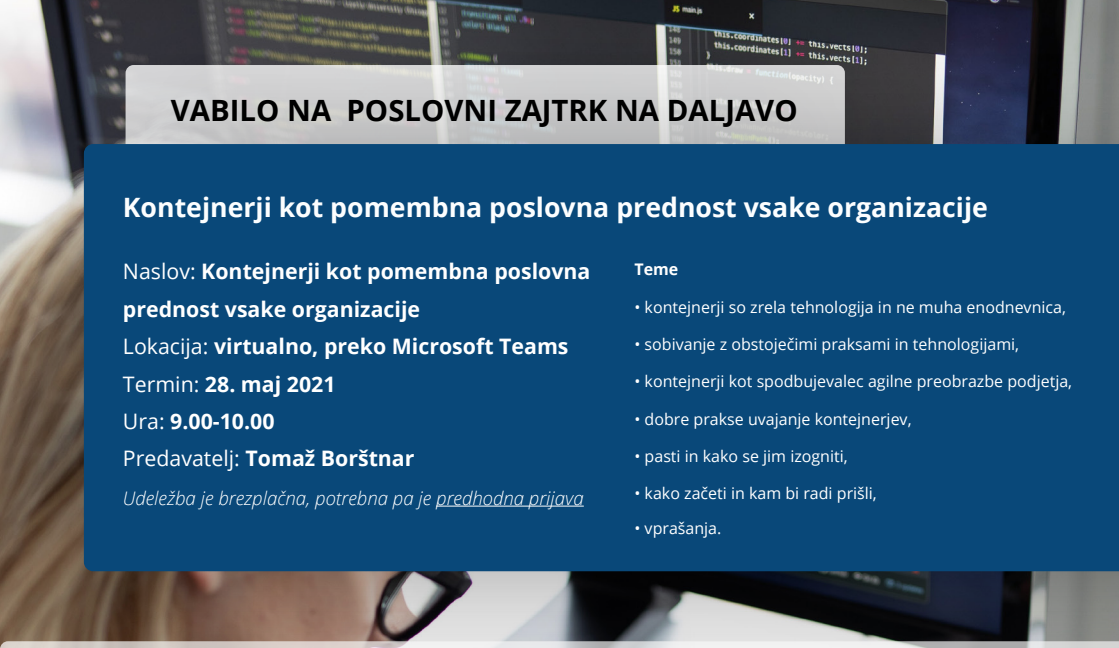
Kibernetska varnost

Namen vavčerja je spodbuditi MSP k zagotavljanju povečanja kibernetske varnosti, s čimer se bo povečala njihova konkurenčnost, dodana vrednost oz. prihodki od prodaje.

Podjetje Kompas Xnet d.o.o. je vpisan v katalog strokovnjakov DIH Slovenija in smo lahko vaš zunanji izvajalec.

Nekaj predlogov:
Digitalizacija dokumentnega sistema in timskega dela
Usposabljanje za digitalno delovno mesto

Za dodatne informacije nam pišite na info@kompas-xnet.si ali pokličite na: 01 5136 993.



VABILO NA POSLOVNI ZAJTRK NA DALJAVO

Kontejnerji kot pomembna poslovna prednost vsake organizacije

Naslov: **Kontejnerji kot pomembna poslovna prednost vsake organizacije**
Lokacija: **virtualno, preko Microsoft Teams**
Termin: **28. maj 2021**
Ura: **9.00-10.00**
Predavatelj: **Tomaž Borštnar**

Udeležba je brezplačna, potrebna pa je predhodna prijava

Teme

- kontejnerji so zrela tehnologija in ne muha enodnevnica,
- sobivanje z obstoječimi praksami in tehnologijami,
- kontejnerji kot spodbujevalec agilne preobrazbe podjetja,
- dobre prakse uvajanje kontejnerjev,
- pasti in kako se jim izogniti,
- kako začeti in kam bi radi prišli,
- vprašanja.

Ali želite vedeti, s katerim treningom boste kmalu lahko s podatki ravnali kot profesionalec?

Power BI je trenutno najbolj popularna zbirka orodij in storitev za pridobivanje informacij iz podatkov. S pomočjo Power BI lahko kreiramo enostavna poročila, bolj zahtevne interaktivne vizualizacije, pa tudi zahtevne analize z uporabo metod podatkovne znanosti.

Analyzing Data with Power BI

V tečaju bomo obravnavali različne metode in najboljše prakse, ki so v skladu s poslovnimi in tehničnimi zahtevami za modeliranje, vizualizacijo in analizo podatkov s Power BI.

POGLEJ VEČ

Data Science with R and SQL Server

Tečaj je namenjen naprednim metodam analize podatkov. Sem spadajo statistične analize, rudarjenje po podatkih, strojno učenje ali kot v zadnjem času imenujemo vse skupaj s poenotenim imenom, data science.

POGLEJ VEČ

Oglejte si našo celotno ponudbo usposabljanja za upravljanje podatkov! Še niste prepričani, katero izobraževanje ustreza vašim ciljem? Vedno smo pripravljeni odgovoriti na vaša vprašanja in se z vami pogovoriti o primernem treningu.