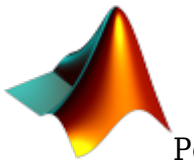
Portál pre odborné publikovanie ISSN 1338-0087

9. Matlab – cell

Foltin Martin · MATLAB/Comsol

17.07.2009



Postupne sa prepracovávame v našom seriáli dátovými typmi v Matlabe. Dnes nadviažeme na predošlý diel v ktorom sme si vysvetlili štruktúry a sústredíme sa na neobvyklý dátový typ *CELL – bunka*. Tento formát je pravdepodobne špecifický len pre Matlab. Doposiaľ som sa s ekvivalentom nestretol v žiadnom inom programovacom jazyku.

V čom je teda bunka špecifická ? Jedná sa o pole podobné matici, alebo vektoru. Zásadným rozdielom je, že prvkami v matici sú čísla. V bunke môže byť prvkom ľubovolný dátový typ. Bunka teda môže obsahovať číslo, maticu, vektor, štruktrúru, text a dokonca aj bunku. Môže tak vznikať komplikovaný dátový objekt s množstvom informácií rôzneho typu. Bunka cell, podobne ako matica, môže mať viac rozmerov (odkaz na matice). Na rozdiel od matíc sa jednotlivé prvky adresujú v zložených zátvorkách. Pomocou nich môžeme ak objekt typu cell vytvoriť.

Deklarovanie cell

Deklarovanie premennej, ktorá bude z triedy cell môžeme dvoma spôsobmy. Prvý je využitie príkazu **cell** s parametrami, ktoré predstavujú rozmer bunky. Bunku s menom A a rozmerom 2×2 teda deklarujeme takto :

```
>> A=cell(2,2)
A =
[] []
[] []


>> whos
Name Size  Bytes  Class Attributes
A    2x2   16     cell
```

V tomto okamihu máme k dispozícii bunku A, ktorá je prázdna.

Druhý spôsob je pre Matlab typický. Jedná sa o deklarovanie priradením. Nie je teda ani v tomto prípade explicitná deklarácia. Postačuje prvé priradenie a Matlab okamžite vyhradí pamäť pre zvolený typ. Hodnoty, ktoré priraďujeme do bunky sú uzavreté v

zložených zátvorkách.

```
>> B={[1] [1 2;3 4]; 'Posterus' [4 5 6] }
B =
[ 1] [2x2 double]
'Posterus' [1x3 double]


>> whos
Name Size  Bytes Class Attributes
A    2x2   80    cell
B    2x2   320   cell
```

Týmto príkladom sme demonštrovali variabilitu triedy cell. Bunka B obsahuje skalár, maticu, text a vektor.

Priradenie do premenných z triedy cell
Bunky môžeme plniť naraz (tak ako sme ukázali pri deklarovaní bunky B), alebo môžeme plniť postupne jednotlivé položky.

```
>> A(1,1)={1};
>> A(1,2)={'textovy retazec'};
>> A(2,1)={magic(4)}
A =
[ 1] 'textovy retazec'
[4x4 double] []
```

Pozrime sa pozornejšie na tento príklad. Na ľavej strane priradenie indexujeme bunku ako obyčajnú maticu. Na strane pravej dávame jednotlivé prvky do zložených zátvoriek. Práve zápis do zložených zátvoriek spôsobí, že aj číslo 1 (na pozícii A(1,1)) je vnímané ako bunka. Existuje aj iná možnosť zápisu. Na ľavej strane indexujeme bunku A ako položky bunky a na pravú stranu zapisujeme priamo dáta bez pretypovania.

```
>> C{1,1}=1;
>> C{1,2}='textovy retazec';
>> C{2,1}=magic(4)
C =
[ 1] 'textovy retazec'
[4x4 double] []
```

Bunky A a C majú rovnaký obsah.

Pre zväčšenie bunky postačuje opäť len priradenie.

```
>> C{3,3}=[2 5 4]'
C =
[ 1] 'textovy retazec' []
[4x4 double] [] []
[] [] [3x1 double]
```

Ako sme už spomenuli, bunka môže mať aj viac rozmerov. Stačí ak priradíme na danú pozíciu ľubovoľné dáta.

```
>> C{3,3,2}='text v 3. rozmere'
C(:,:,1) =
[ 1] 'textovy retazec' []
[4x4 double] [] []
[] [] [3x1 double]


C(:,:,2) =
[] [] []
[] [] []
[] [] 'text v 3. rozmere'
```

Indexovanie premenných z triedy cell

Doposiaľ sme si ukázali ako deklarovať a ako napĺňať premenné z triedy cell. Dôležité, ale je aj vedieť získať dáta z takejto premennej a následne ich spracovať. Indexovanie v rámci premennej cell je obdobné ako pri maticiach. Prvý index zodpovedá riadkom, druhý stĺpcom a ostatné sú radené do hyperpriestoru, ktorý sa len ťažko dá predstaviť. Na rozdiel od indexovania matíc, v bunkách máme dve možnosti ako osloviť určitý prvok. V prvom prípade zapíšeme súradnice prvku do okrúhlych zátvoriek. Teraz bude vrátená premenná typu cell.

```
>> C(1,1,1)
ans =
[1]


>> whos
Name Size   Bytes   Class Attributes
C    3x3x2  576     cell
ans  1x1    68      cell
```

Druhá možnosť je zapísať súradnice prvku do zátvoriek zložených. V tomto prípade bude vrátená hodnota v príslušnom dátovom type ako bola zapísaná do bunky.

```
>> C{1,1,1}
ans =
1


>> whos
Name Size   Bytes Class Attributes
C    3x3x2  576   cell
ans  1x1    8     double
```

Aj v triede cell môžeme využiť operátor **:** s rovnakým významom ako pri maticiach. Opäť ale treba rozlišovať či zapíšeme symbol **:** do zložených, alebo okrúhlych zátvoriek.

```
>> C(:,:,1)
ans =

[ 1] 'textovy retazec' []
[4x4 double] [] []
[] [] [3x1 double]


>> C{:,:,1}
ans =
1
ans =
16 2 3 13
5 11 10 8
9 7 6 12
4 14 15 1
ans =
[]
ans =
textovy retazec
ans =
[]
ans =
[]
ans =
[]
ans =
[]
ans =
2
5
4
```

Predviedli sme si ako sa dostať k dátam, ktoré sú uložené v bunkách. Významnú úlohu zohrávajú práve zátvorky ktoré použijeme. Ak by sme potrebovali konvertovať dáta z bunky do matice, môžeme využiť príkaz **cell2mat**. Vhodnou voľbou zátvoriek však dokážeme tento príkaz plne nahradiť.

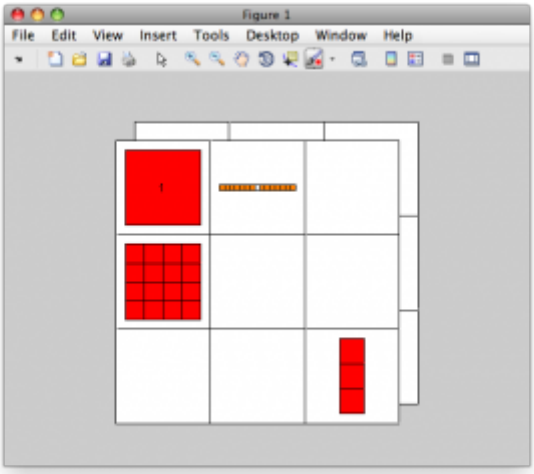
```
>> cell2mat(C(2,1,1))
ans =
16 2 3 13
5 11 10 8
9 7 6 12
4 14 15 1


>> whos
Name  Size   Bytes  Class Attributes
C    3x3x2   576    cell
ans   4x4    128    double
```

Grafické zobrazenie buniek

Ako už asi každý pochopil, premenná z triedy cell môže byť značne koplikovaná a neprehľadná. Pomocť nám môže príkaz **cellplot**. Tento príkaz zobrazí premennú tak aby bolo zrejmé aké dáta sa nachádzajú na jednotlivých poliach. Samozrejme prehladne sa dá zobraziť len bunka 2D. V 3D bunke sú jednotlivé vrstvy radené za seba, takže vidíme len prvú vrstvu. To isté platí aj pre viacrozmerné bunky.

```
>> cellplot(C)
```



Obr. 1 štruktúra C zobrazená príkazom cellplot

Záver

Trieda cell je vhodná na vytváranie zložitých dátových štruktúr. Je dobré ju poznať a aktívne využívať, lebo v istých situáciách ušetrí množstvo programátorskej práce. Trieda cell sa dá využiť pri ukladaní nehomogénnych dát, alebo v špeciálnych prípadoch keď matica nemá rovnaký počet riadkov resp. stĺpcov. Podrobnejšie inforáciek tejto zaujímavej triede nájdete v nápovede Matlabu v kapitole Programming Fundamentals/Classes/Cell Arrays.