Portál pre odborné publikovanie ISSN 1338-0087

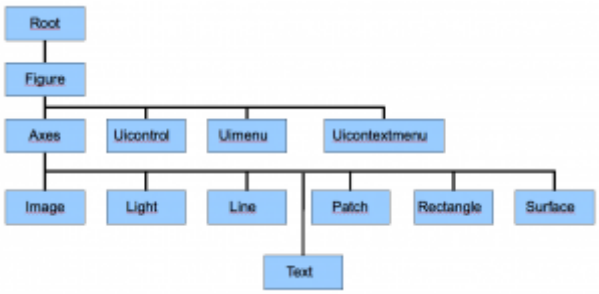
18. Matlab – figure ako objekt

Foltin Martin · MATLAB/Comsol

06.11.2009

Dlhším používaním grafických funkcií v Matlabe, dôjdete k poznaniu že ich možnosti sú pomerne oklieštené. Komplikácie spôsobuje len taká drobnosť ako zmena šírky čiary, zmena farby čiar pri použití príkazu plotyy, prípadne zmena ciachovania mriežky v grafe. Všetky tieto problémy majú riešenie. Musíme sa však na obrázok – *figure* pozrieť ako na objekt. Ako každý objekt, tak aj figure má množstvo vlastností. Práca s vlastnosťami grafických objektov bude práve témou dnešného článku.

Matlab vníma každý obrázok ako objekt. Priebehy, osi, popisy a všetko ostatné v objekte figure je vnímané ako potomok rodiča. Hierarchická štruktúra grafických objektov je na obr. 1.



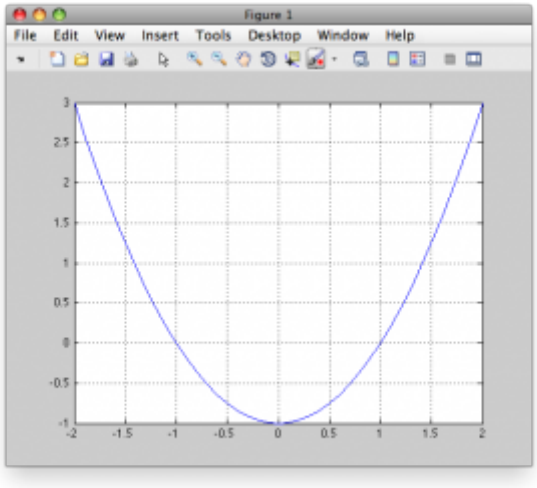
Obr. 1. Hierarchické usporiadanie grafických objektov v objekte Figure

Prečo sa, ale vôbec zaoberáme grafickými objektami? Je to preto, lebo pomocou klasických príkazov nie je možné realizovať niektoré úlohy. Pomocou objektov máme graf plne pod kontrolou a dokážeme meniť jeho vlastnosti. Existujú dve možnosti ako narábať s vlastnosťami grafického objektu. Prvá je pomocou GUI samotného objektu figure. K dispozícii sú všetky vlastnosti. Interaktívne môžeme upravovať graf až do požadovanej podoby. Na príklade si ukážme ako postupovať. Predstavme si, že potrebujeme vykresliť graf funkcie $y=x^2 - 1$, ktorú sme použili v článku o práci s 2D grafmi (vzťah 2). Túto funkciu chceme zobraziť na rovnakom intervale, čiara ma byť hrubšia a má mať bledomodrú farbu. Ak sa pustíme do riešenia úlohy klasickým spôsobom, zistíme, že máme k dispozícii len obmedzený počet farieb (kde bledomodrá chýba) a šírku čiary nemôžeme meniť vôbec.

```
>> x=-2:0.1:2;
>> y=x.^2-1;
```

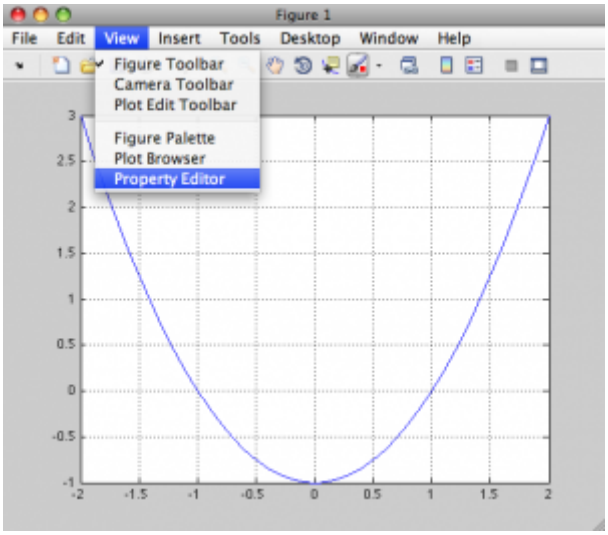
```
>> plot(x,y)
>> grid
```

Výsledkom uvedených príkazov je obr. 2.



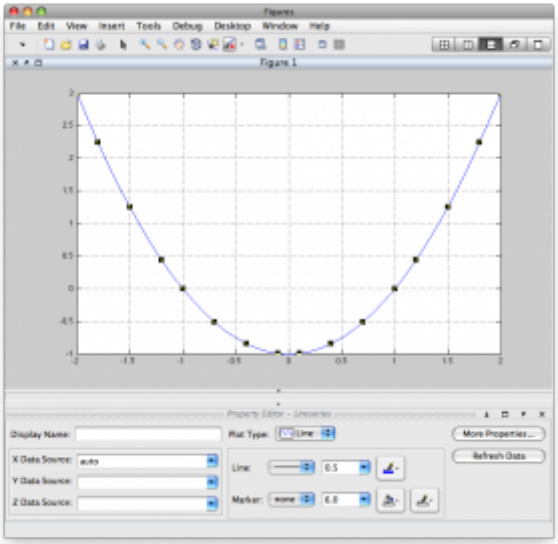
Obr. 2. Priebeh funkcie $y=x^2-1$

Pre zmenu atribútov vybraného objektu v grafe zvolíme z menu okna Figure 1 položku View/Property Editor (obr. 3.)



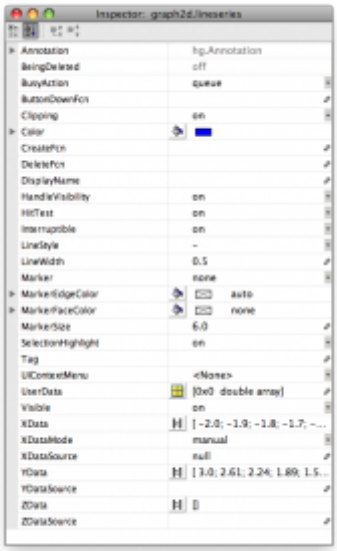
Obr. 3. Výber Property Editor

Po kliknutí na modrú čiaru, ktorá predstavuje funkčné hodnoty funkcie, máme ďalšie možnosti nastavenia grafu (obr. 4.).



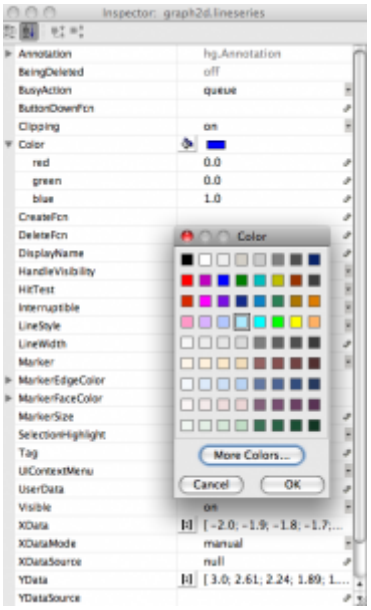
Obr. 4. Property Editor – Lineseries

V tejto situácii môžeme meniť aj šírku čiary, farbu, prípadne typ. Taktiež môžeme zmeniť značky na grafe. Za pozornosť, ale stojí tlačidlo **More Properties...** . Po jeho stlačení sa nám otvorí okno **Inspector**, kde máme možnosť meniť všetky vlastnosti čiary. Či už je to farba, ale šírka. Naviac tu nájdeme samotné dáta, ktoré tvoria graf. Za povšimnutie stojí, že mená vlastností sú uvedené v ľavej časti tabuľky a ich hodnoty v pravej. Práve mená sú dôležité pre zmenu vlastnosti z príkazového riadku. Tento postup si ukážeme v druhej časti dnešného dielu.



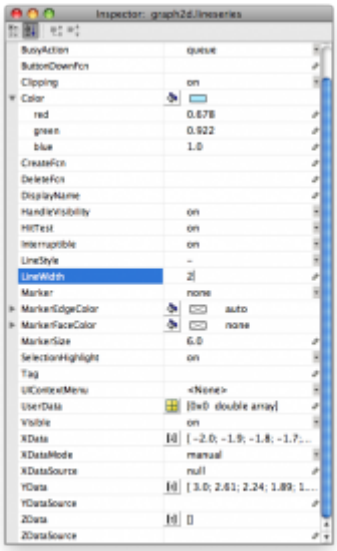
Obr. 5. Inspector graph2d.lineseries

Zmenu farby z prednastavenej tmavomodrej na bledomodrú docielime tak, že vyhľadáme vlastnosť **Color**. Farbu môžeme buď vybrať z palety farieb, alebo definovať ako trojzložkovú farbu RGB (zastúpenie farby je z intervalu 0-1) (obr. 6).



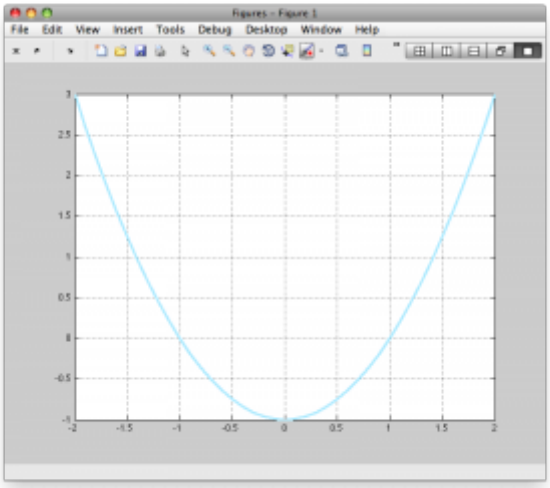
Obr. 6. Zmena farby čiary

Ďalšiu vlastnosť, ktorú plánujeme zmeniť je šírka čiary. Tú meníme vlastnosťou LineWidth. Postačuje zmeniť prednastavenú hodnotu 0,5 na dajme tomu 2 (obr. 7).



Obr. 7. Zmena šírky čiary – vlastnosť LineWidth

Zatvorením okna Inspector a Property Editor sa dostávame k výslednému grafu.



Obr. 8. Výsledný obrázok

Týmto postupom môžeme interaktívne meniť aj ďalšie vlastnosti grafu, ktoré nájdeme prehľadne usporiadané v oknách Property Editor, resp. Inspector. Ako ale postupovať, ak chcem vytvoriť program v ktorom k vykresleniu takéhoto grafu príde automaticky? Práve teraz sa dostávame k priamej práci s grafickými objektami. Počas vzniku ľubovoľného grafického objektu dochádza k vytvorení jedinečného identifikátora objektu. V Matlabe sa nazýva **handle**. Pomocou handle dokážeme pristupovať k danému grafickému objektu a vieme meniť jeho vlastnosti pomocou príkazov. Doposiaľ sme sa nestretli s tým, že by mali grafické funkcie výstupnú hodnotu. Po dôkladnom preštudovaní helpu k takýmto príkazom zistíte, že návratová hodnota grafických funkcií je práve ukazovateľ na novovzniknutý grafický objekt – handles. Náš príklad by sme teda mohli modifikovať takto.

```
>> x=-2:0.1:2;
>> h=plot(x,y);
>> grid
```

Výsledkom bude rovnaký graf, ako v predošlom prípade. Rozdiel je len v tom, že máme premennú **h**, ktorá je handle na daný graf.

Teraz, keď máme odkaz na grafický objekt, tak si ukážeme ako narábať s jeho vlastnosťami. Najdôležitejšími príkazmi sú **get** a **set**. Ako sa dá očakávať, príkazom get zisťujeme aktuálnu hodnotu zvolenej vlastnosti a príkazom set meníme hodnotu vlastnosti. Príkaz get má ešte jednu funkciu. Ak nezadáme žiadnu vlastnosť a ako parameter zvolíme len handle príslušného objektu, jej návratová hodnota bude zoznam všetkých vlastností objektu.

```
>> get(h)
DisplayName: ''
Annotation: [1x1 hg.Annotation]
Color: [0 0 1]
LineStyle: '-'
LineWidth: 0.5000
Marker: 'none'
MarkerSize: 6
```

```
MarkerEdgeColor: 'auto'
MarkerFaceColor: 'none'
XData: [1x41 double]
YData: [1x41 double]
ZData: [1x0 double]
BeingDeleted: 'off'
ButtonDownFcn: []
Children: [0x1 double]
Clipping: 'on'
CreateFcn: []
DeleteFcn: []
BusyAction: 'queue'
HandleVisibility: 'on'
HitTest: 'on'
Interruptible: 'on'
Selected: 'off'
SelectionHighlight: 'on'
Tag: ''
Type: 'line'
UIContextMenu: []
UserData: []
Visible: 'on'
Parent: 170.0027
XDataMode: 'manual'
XDataSource: ''
YDataSource: ''
ZDataSource: ''
```

Uvedený zoznam vlastností sme mohli vidieť v Inspectore. Ak potrebujeme zistiť hodnotu vybranej vlastnosti, tak jej názov použijeme ako druhý parameter. Dajme tomu, že chceme zistiť aktuálnu šírku čiary.

```
>> get(h,'LineWidth')
ans =
0.5000
```

Máme možnosť nahliadnuť aj do dát ktoré graf tvoria.

```
>> Xgraf=get(h,'Xdata');
>> Ygraf=get(h,'Ydata');
```

Týmto zápisom sme z grafu odčítali všetky funkčné hodnoty.

Pomocou príkazu set teraz zmeníme šírku čiary a farbu. Prvý parameter je ukazovateľ na objekt, nasleduje názov vlastnosti a nová hodnota.

```
>> set(h,'LineWidth',2)
```

Bledomodrá farba má v 8bitovom RGB zápise hodnotu R=173, G=235, B=255. Matlab však spracúva hodnoty normované na interval 0-1. Preto zmena farby bude vyzerať

takto :

```
>> set(h,'Color',[173 235 255]/255)
```

Získali sme rovnaký graf ako v predošlom prípade. Výsledok sme dosiahli len pomocou príkazov, čo predstavuje značnú výhodu. Príkazy je možné zapísať do programu v ktorom sa len pomocou zmeny niekoľkých parametrov môže docieliť iný výsledný obrázok. Taktiež je možné opakovať sekvenciu príkazov s rôznymi dátami a tak dosiahnuť značnej časovej úspory. Práca v interaktívnom prostredí s myšou si totiž vyžaduje individuálny prístup ku každému novému obrázku a nie je možné ju viacnásobne aplikovať.

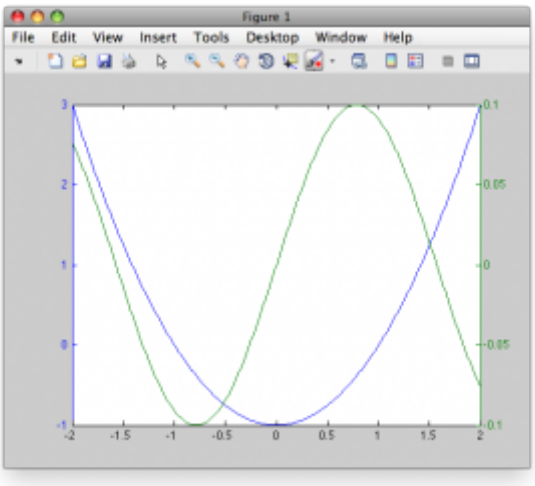
Často využívanou grafickou funkciou je príkaz plotyy. Príkaz je vhodný na vykreslenie dvoch priebehov s rôznym ciachovaním osí. Prehľadne sa tak dajú zobraziť priebehy, ktoré používajú rádovo iné jednotky. Po čase zistíme, že možnosti modifikácie grafu sú mizivé. Bez práce s objektami v grafe neide mimo elementárnych vecí nastaviť prakticky nič. Náhľadom do nápovedy zistíme, že funkcia plotyy vracia dokonca hodnoty 3.

```
[AX,H1,H2] = PLOTYY(...)
```

Ukazovateľ AX je vlastne vektor dvoch hodnôt a ukazuje na prvú a druhu os grafu (Axis). Ukazovatele H1 a H2 identifikujú dvojicu čiar v grafe.

```
>> x=-2:0.01:2;
>> y1=x.^2-1;
>> y2=0.1*sin(2*x);
>> [ax,h1,h2]=plotyy(x,y1,x,y2)
ax =
170.0017 172.0011
h1 =
171.0021
h2 =
173.0016
```

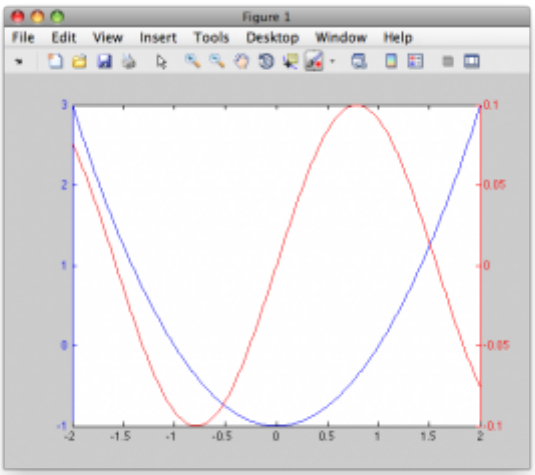
Výsledkom bude graf (obr. 9)



Obr. 9. Graf získaný pomocou plotyy

Ak sa rozhodneme zmeniť farbu druhého priebehu, tak musíme opäť siahnuť po objektoch. Tentokrát ale musíme meniť farbu čiary, ale aj farbu popisu pravej y osi.

```
>> set(h2,'Color',[1 0 0]);
>> set(ax(2),'YColor',[1 0 0]);
```



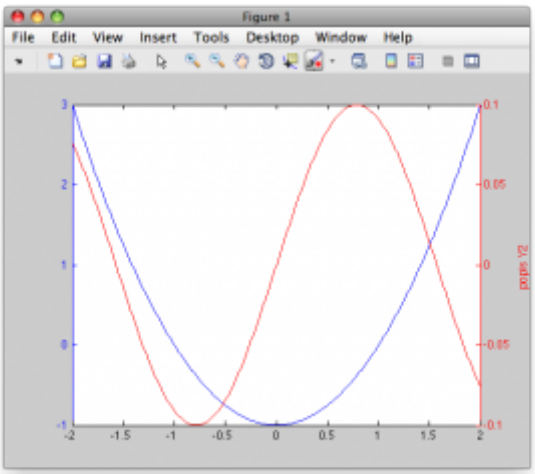
Obr. 10. Zmenená farba druhého priebehu

Pridávanie popisu osí tiež nie je v grafoch typu plotyy triviálna úloha. Nielenže musíme siahnuť po objektoch, ale musíme dokonca hľadať potomkov objektu na ktorý ukazuje AX. Na popis osí odkazuje vlastnosť **YLabel**. Ukazovateľ teda získame príkazom get.

```
>> YL=get(ax(2),'YLabel');
```

V premennej YL máme uložený ukazovateľ na popis osí. Vieme teda získať aj zoznam všetkých vlastností tohto objektu. Vlastnosť String predstavuje popis. Zmeniť ju môžeme príkazom set.

```
>> set(YL,'String','popis Y2');
```

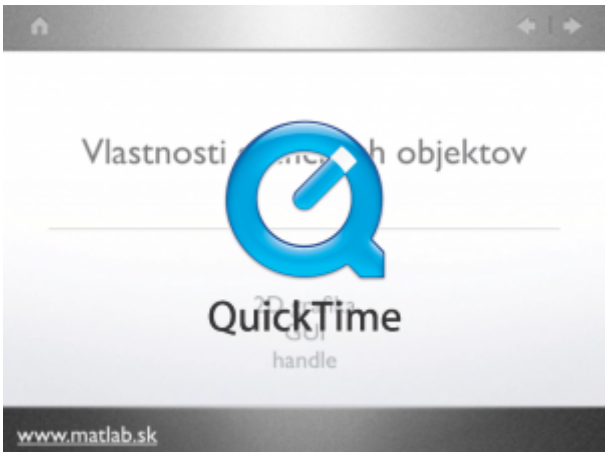


Obr. 11. Zmenený popis pravej y-ovej osi

Ak pri tvorbe obrázku z dajakého dôvodu neuložíme handler do premennej, tak máme možnosť využiť príkazy na ich detekciu. Jedná sa o trojicu príkazov GCF, GCA a GCO. Ich význam je zrejmý z názvu.

```
GCF – Get handle to current figure
GCA – Get handle to current axis
GCO – Get handle to current object
```

Na záver článku uvádzame ešte video v ktorom sú zopakované elementárne postupy pri práci s grafickými objektami.



Video. Práca s grafickými objektami

Predošlé články venované grafickým funkciám nám zabezpečili dostatočné vedomosti v práci s grami. Dnes sme si ukázali pokročilejšie nástroje, ktoré nám umožnia prispôsobiť graf presne našim potrebám. Pomocou objektov a zmeny ich vlastností dokážeme realizovať aj jednoduché animácie. O tom sa, ale budeme baviť v niektorej z ďalších častí.

Literatúra

1. Matlab 7 – Graphics, The MathWorks, http://www.mathworks.com/access/helpdesk/help/pdf_doc/matlab/graphg.pdf, 24.9.2009