26 tip: Objekty a concurrency 2 – swizzling

Keď sa tento seriál pred niekoľkými rokmi začínal, jeho prvý diel bol venovaný zabezpečeniu izolácie inštancie objektov na exkluzívny prístup a opisu príslušných funkcií API. Nedávno sa mi však stalo, že ma tento diel dostihol. Jeden zo zákazníkov začal mať problémy v aplikácii, hoci dôsledne používal exkluzívne zámky na editovanie inštancií svojich objektov. Keď po príčine pátral dôslednejšie, zistil (a priznám sa, že som to sám nevedel), že zámky aplikované na inštancie objektov majú veľmi nepríjemnú vlastnosť, že sa nešíria prostredníctvom odkazov (swizzling) na ďalšie prepojené objekty.

A to bol práve zdroj problémov. V aplikácii sa nachádzalo niekoľko miest, kde bola dvoma procesmi zároveň modifikovaná rovnaká inštancia objektu. Raz priamo, druhýkrát prostredníctvom odkazu práve z inej, pripojenej triedy. Ako toto môže nastať a ako takéto situácie ošetriť, to vám ukážeme v nasledujúcom modelovom príklade. Po stiahnutí, rozbalení a nainštalovaní projektu môžeme testovať. Otvoríme si dve konzoly Caché a prepneme sa do testovacieho názvového priestoru, pripravíme si (ALE ZATIAĽ nespustite!) príkazy:

```
// konzole 1
>w ##class(User.Test).GoObjectB()

// konzole 2
>w ##class(User.Test).GoObjectA(0)
```

Najskôr spustíme proces na konzole 1 a počas 10 sekúnd aj proces na konzole 2 a sledujeme výstupy. To, čo uvidíme, môže vyzerať na prvý pohľad čudne, ale je to v poriadku, proces v konzole 2 sa snaží otvoriť novú inštanciu objektu User.B, a len čo sa mu to podarí (po tom, čo proces v konzole 1 uvoľní zámok), načíta ho z DISKU a zmení. Medzitým proces v konzole 2 vykoná opätovné načítanie inštancie z disku a zobrazí výsledok. V ideálnom prípade budú hodnoty oboch inštancií konzistentné (v praxi sa nám totiž kód oboch konzol môže vykonať rôzne rýchlo a vrátiť potom tomu zodpovedajúce – od uvedeného obrázka odlišné – výsledky).

A teraz zavoláme

```
// konzole 1
>w ##class(User.Test).GoObjectB()

// konzole 2
>w ##class(User.Test).GoObjectA(1)
```

Výsledok bude odlišný – rovnaká inštancia objektu vracia rôzne hodnoty v každom procese, ako ukazuje súhrnný obrázok č. 1.

Kritický kód uvádzam ešte separátne: takto treba vždy ošetriť prístup k odkazovaným objektom, ak ich chceme izolovať.

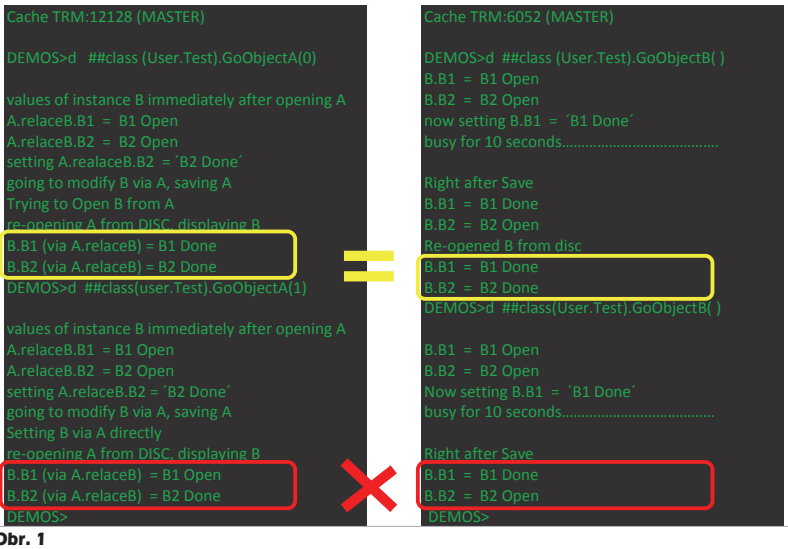
```
        #dim oB as User.B
        w !,"Trying to Open B from A"
        set oB=oA.relaceB.%Open(oA.re-
laceB.%Oid(),4,.sc)
        $$$THROWONERROR(sc,sc)
        set oB.B2 = "B2 Done"
```

Rozdiely v správaní zámkov pri priamom prístupe a pri použití swizzlingu môžu, ako sme si ukázali, mať veľké dôsledky pre aplikáciu. Žiaľ, dokumentácia k tomuto správaniu nie je dostatočne podrobná.

Tento príklad sa vám môže zdať pritiahnutý za vlasy, ale v praxi možno na takú kombináciu dvoch procesov pracujúcich nad rovnakou inštanciou raz priamo a raz nepriamo ľahko naraziť. Ak napríklad vykonávate účtovný zápis, aktualizujete zostatky na jednotlivých účtoch analytickej evidencie v účtovnej knihe, ale aj meníte zostatky na príslušných účtoch syntetickej evidencie. No a ak sa niekto rozhodne zmeniť čokoľvek na syntetickom účte v okamihu realizácie účtovnej operácie, je zamiesené na nešťastie, pravda, iba ak nebudete dbať na uvedené odporúčania.

DAN KUTÁČ, InterSystems

Ďalšie zaujímavé diely seriálu Tipy a triky s Caché, ktoré vychádzajú už od roku 2005, nájdete na stiahnutie tu: **http://www.intersystems.cz/education/university/serialy/tipyTrikyCache.html**



Obr. 1