Pisanje MEL skriptova

MEL (Maya Embedded Language) interni je skript jezik programa Maya. Mada će reči *pisanje skriptova* prestraviti mnoge umetnike, korišćenje MEL skriptova bitan je deo rada u programu Maya. Da biste u potpunosti ovladali MEL-om, morali biste da proučite knjigu koja je posvećena ovom skript jeziku ili programiranju uopšte. Srećom, dovoljno je da razumete osnove da biste pisali kratke skriptove i mnogo efikasnije radili u programu. Ovaj dodatak poslužiće kao vodič za razumevanje osnovne strukture skript jezika MEL.

Dodatak ćemo zaokružiti tako što ćemo istražiti postupak pisanja skripta. U odeljku „Analiza“, napisaćemo kratak skript koji smo pomenuli u poglavlju 10.

Šta je MEL?

MEL je jezik za pisanje malih, izvršnih programa – skriptova. Pisanje skriptova je jednostavniji oblik programiranja. Za razliku od jezika kao što su C i C++, MEL ne zahteva da skriptovi budu *prevedeni* na mašinski kôd da bi ih računar izvršio. Zapravo, MEL možete smatrati slojem između grafičkog korisničkog okruženja programa Maya i jezika C++. Svaki put kada pritisnete dugme na polici ili odaberete komandu iz menija, poslaćete MEL komandu glavnom programu Maya, a on će potom razgovarati sa operativnim sistemom računara koji kontroliše hardver.

Šta MEL skriptovi rade?

MEL skriptovi mogu da vam pomognu u gotovo svemu što radite u programu Maya. Jedna od najjednostavnijih stvari koje pomoću njih možete uraditi jeste postavljanje nekoliko komandi u niz i njihovo izvršavanje u sekvenci. Na primer, vreme koje je potrebno za pretvaranje NURBS modela u poligone i njihovo raščišćavanje, znatno ćete skratiti ako napravite skript koji će pokrenuti komande za pretvaranje, kombinovanje i stapanje temena, zamrzavanje transformacija i brisanje istorije. Kad god primetite da se neki zadatak ponavlja, kao u navedenom primeru, trebalo bi da razmislite o pisanju MEL skripta koji će taj zadatak obaviti.

Neke od namena MEL skriptova su:

- Ponavljanje prethodnog zadatka
- Pravljenje namenskih kontrola, prozora i delova grafičkog korisničkog okruženja
- Upravljanje sistemima čestica
- Pravljenje, povezivanje i kontrolisanje više atributa (primer za to je skript tmRenderPass koji smo koristili u poglavlju 20)
- Poravnavanje atributa transformacija različitih objekata (objašnjeno je na kraju ovog dodatka)

MEL komande

Da biste u potpunosti ovladali MEL-om, verovatno ćete istražiti i vežbati osnovno programiranje. Odaberite knjigu o jeziku C, C++, Perl, Java, ili nekom drugom programskom jeziku, pa se upoznajte s načinom na koji programiranje funkcioniše, s načinima optimizovanja i pisanja dobrih skriptova.

Ukoliko ne želite da idete toliko daleko, na raspolaganju vam je mnoštvo odličnih knjiga posvećenih pisanju MEL skriptova. Jedan od najboljih izvora za učenje MEL-a nalazi se u Mayinoj ugrađenoj dokumentaciji, u odeljku „MEL Command Reference“. Proučavanje komandi i njihove namene odlično je polazište za učenje MEL-a. Taj deo dokumentacije je rečnik svih MEL komandi dostupnih u programu Maya, s detaljnim opisom opcija za svaku komandu i primerima njihove upotrebe.

SAVET ***Kad god pokrenete program Maya, pritisnite taster F1 da biste otvorili ugrađenu dokumentaciju, pa u njoj pronađite odeljak „MEL Command Reference“. Odaberite komandu koja vas zanima, ili bilo koju komandu, i pročitajte šta o njoj piše. Pre ili kasnije, to što čitate dobiće smisao. Tako je i autor ovih redova naučio osnove pisanja MEL skriptova.***

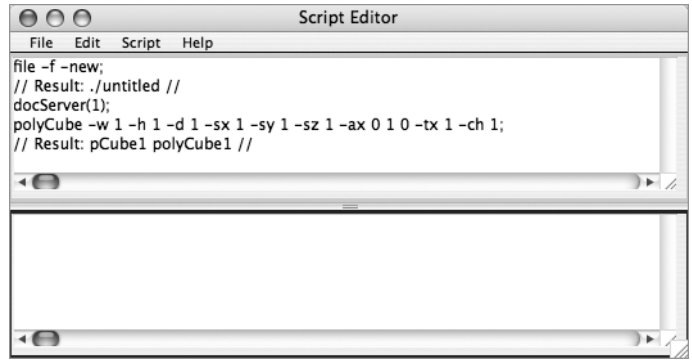
Ako govorite engleski, većina MEL-ovih komandi za vas će imati razumljiva, smislena imena. Na primer, komanda za dodavanje atributa zove se `addAttr` (od *add attribute* – dodaj atribut). Pored toga što uputstvo možete pretraživati po abecednom redu, komande možete pregledati i po kategorijama. U svakom slučaju, imena MEL komandi smišljena su tako da što više liče na engleski jezik.

Script Editor

Najjednostavniji način da pronađete MEL komandu jeste da je izvršite preko nekog menija u glavnom korisničkom okruženju programa, i potom pogledate šta je zabeleženo u Script Editoru (Window | General Editors | Script Editor). Svaki put kada izaberete objekat, otvorite prozor ili izvršite komandu iz menija ili preko dugmadi na polici, akcija će pokrenuti MEL komandu, a ona će biti zabeležena u Script Editoru. Ukoliko ste nekoliko sata radili u programu Maya, možda će vas iznenaditi kada vidite da je u tom prozoru navedena svaka operacija koju ste obavili otkad ste pokrenuli program.

NAPOMENA ***Neke komande se podrazumevano ne ispisuju u Script Editoru. To su obično komande koje učitavaju određene delove korisničkog okruženja. Ukoliko hoćete da vidite i njih, s trake menija Script Editora odaberite Script | Echo All Commands.***

Slika A-1 prikazuje Script Editor i njegov sadržaj pošto je program Maya pokrenut, otvorena ugrađena dokumentacija i napravljena poligonalna kocka. Gornji deo Script Editora, u kom se nalaze zabeležene komande, naziva se History (okno istorije), a donji deo se naziva Input (okno za unos), i u njemu ćete upisivati i izvršavati komande.



SLIKA A-1

Script Editor.

Pogledajmo komande koje su dosad zabeležene i zaključimo šta se dešavalo. U prvom redu piše

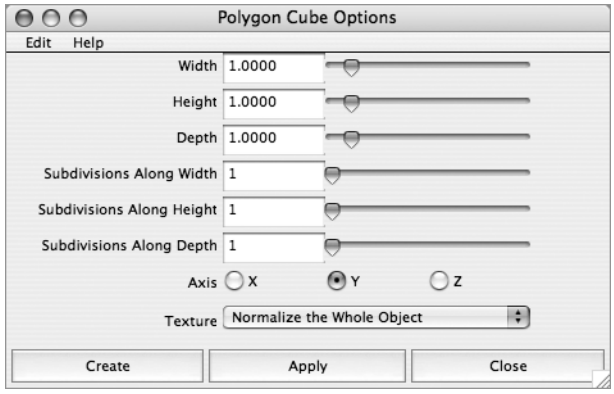
```
file -f -new;
```

Tu je upotrebljena komanda `file`. Ona će napraviti novu scenu u programu Maya, a izvršava se automatski kada pokrenete program. Tekst koji sledi nakon crtice (`-f, -new`) naziva se *indikator* (engl. *flag*). Indikatori su opcije ili parametri za komandu. Kada podesite opcije za komandu u njenom prozoru Options, te opcije ili atributi, pojaviće se kao indikatori u skriptu. Pa šta rade indikatori `–f` i `–new`? Mogli bismo potražiti komandu `file` u uputstvu „MEL Command Reference“, ali pošto smo već u Script Editoru, unećemo komandu za prikazivanje pomoći (`help`) za komandu `file`: u okno na dnu Script Editora upišite sledeće:

```
help file ;
```

Potom pritisnite CTRL-ENTER (CONTROL-RETURN) da biste izvršili komandu. Biće naveden kratak opis komande `file`, a svi dostupni indikatori biće navedeni uz objašnjenje šta oni predstavljaju. Na listi pronađite indikator `–f` i gornji prozor će vam reći da on predstavlja *force*. Precizniju definiciju te oznake potražite u uputstvu „MEL Command Reference". Saznaćete da *force* znači prinudno pokretanje neke akcije (kao što su new, open i save). Indikator `–new` pravi novu scenu nazvanu *untitled*. Najzad, red se završava tačkom i zarezom (;), koji označava kraj komande.

Sledeći red u skriptu govori koji je rezultat komande `file`, a u našem primeru to je scena nazvana *untitled*. Rezultat je samo povratna informacija kojom program javlja šta se desilo – ona ne izvršava komandu. Red koji počinje dvostrukom kosom crtom (//) naziva se *komentar* (engl. *comment*) i on se ne izvršava. Komentari se često koriste za dokumentovanje, objašnjavanje ili komentarisanje skripta.



SLIKA A-2

Prozor Options sa opcijama komande.

Komanda `docServer` u sledećem redu, otvara ugrađenu dokumentaciju. Nakon komande za pravljenje poligonalne kocke, `polyCube`, sledi mnogo indikatora i njihovih vrednosti. Opet biste mogli u Script Editoru iskoristiti komandu `help` ili potražiti komandu `polyCube` u ugrađenoj dokumentaciji. Otkrili biste da indikatori predstavljaju sve opcije komande koje ćete videti i u prozoru Options, kao što je prikazano na slici A-2.

Kao što vidite, mada su dostupne mnoge opcije, MEL komande se prilično jednostavno čitaju. Sada biste u okno za unos mogli da postavite nekoliko komandi u niz i da ih izvršite bez previše muke.

Korišćenje MEL skriptova

Jedna od najlepših strana MEL-a jeste to što programu možete dodati alatke i opcije, ne čekajući da kompanija Alias izda sledeću verziju softvera. Međutim, mnogi među nama više vole da koriste alatke nego da ih prave. Zajednica korisnika programa Maya velikodušno je sastavila biblioteku MEL skriptova koje možete besplatno preuzeti sa adrese *http://www.highend3d.com*. Pre nego što rešite da napravite novu alatku, posetite ovu Web prezentaciju i pogledajte da li možete pronaći MEL skript koji radi ono što vam treba. Vrlo je verovatno da ga je neko već napisao i da ga možete preuzeti, instalirati i koristiti.

Još jedan dobar izvor MEL skriptova jeste Web prezentacija kompanije Alias (*http://www.alias.com*). Alias nudi i mnogo DVD-ova sa obukom koji sadrže MEL skriptove. Među njima su diskovi „Integrating a Creature Rig into a Production Pipeline and Fast Animation Rigs" sa desetinama izvanrednih skriptova koji će vam pomoći u opremanju i animiranju likova. Autor diskova, Jason Schliefer, nudi mnogo MEL skriptova koje možete preuzeti s njegove Web prezentacije, *http://www.jonhandhisdog.com*.

Gde su skriptovi smešteni?

Upravo ste posetili prezentaciju Highend3d.com, pronašli MEL skript koji vam treba i preuzeli ga. I šta sad? Uobičajen način korišćenja MEL skriptova jeste da ih postavite u poddirektorijum za skriptove na svom disku. Kada instalirate program Maya, nastaju tri takva poddirektorijuma.

Jedan se nalazi u istom direktorijumu u kom je i program Maya. U njemu ćete pronaći još nekoliko poddirektorijuma sa stotinama skriptova. Ne zaboravite da je celo korisničko okruženje programa Maya programirano na MEL-u. Skriptovi u tim poddirektorijumima upravljaju korisničkim okruženjem. Ako ne morate da ga menjate, najbolje je da izbegavate taj poddirektorijum.

Drugi poddirektorijum sa skriptovima nalazi se na lokaciji Documents and Settings\<*vaše korisničko ime*>\My Documents\Maya\Scripts (na Macintoshu putanja je /Users/<*vaše korisničko ime*>/Library/Preferences/Alias/Maya/Scripts). On je dobro mesto da smestite skriptove u razvoju ili skriptove koje ste sami učitali.

Skriptovi koje ćete često koristiti u programu treba da budu smešteni na lokaciji Documents and Settings\<*vaše korisničko ime*>\My Documents\Maya\6.0\scripts (na Macu: Users/<*vaše korisničko ime*>/Library/Preferences/ Alias/Maya/6.0/Scripts). Kad god pozovete skript, Maya pregleda ovaj poddirektorijum, pronalazi skript i izvršava procedure koje se u njemu nalaze.

Definicija ***Procedura*** **(engl.** ***procedure*****)** Korisnički definisana funkcija slična Mayinim ugrađenim funkcijama. Objašnjene su u odeljku „Procedure".

Izvršavanje MEL skriptova

Kada preuzmete skript, dobro bi bilo da ga otvorite u programu za obradu teksta i pročitate uputstva koja su mu možda dodata. Ta uputstva će vam reći šta skript radi, kako se koristi (na primer, šta treba da bude izabrano pre nego što izvršite skript), i kako se izvršava. Ponekad su skriptovi pisani *modularno*. Drugim rečima, nekoliko skriptova je potrebno da bi se izvršila procedura. U slučaju skripta tmRenderPass, poseban skript nazvan tmRenderPassUI.mel pravi prozor koji potom poziva procedure u skriptu tmRenderPass.mel.

Dobro dokumentovani skriptovi će vam reći šta treba da uradite da biste ih izvršili. Da biste pokrenuli skript, na komandnu liniju ili u okno Input prozora Script Editor unesite ime komande koja vam treba. Ukoliko skript nema dokumentaciju, verovatno je u pitanju skript globalne procedure.

Većina skriptova koji se preuzimaju sa Interneta jesu *skriptovi globalnih procedura* (engl. *global procedure scripts*). To znači da procedure u tim skriptovima mogu biti pozivane s bilo kog mesta: iz skripta, preko dugmeta na polici ili nekako drugačije. Skript globalne procedure mora dobiti ime po globalnoj proceduri. Ukoliko se procedura zove tmRenderSpec, skript će se zvati tmRenderSpec.mel. Kada na komandnu liniju ili u Script Editor upišete **tmRenderSpec**, Maya će pregledati sve putanje skriptova tražeći ime skripta koje odgovara komandi. Ukoliko pronađe skript, ona će deklarisati sve globalne MEL procedure iz te datoteke i izvršiti ih. U većini slučajeva, najjednostavnije je da na polici napravite dugme sa imenom komande da biste je mogli koristiti bez traženja imena.

Pisanje MEL skriptova

Ranije u ovom dodatku govorili smo o MEL komandama i predstavili povezivanje nekoliko komandi u niz. Takav način rada vam može uštedeti vreme u nekim situacijama, ali kada postanete veštiji u korišćenju programa Maya, bez sumnje ćete morati da odete i korak dalje. Nailazićete na specifične probleme ili pitanja vezana za proces rada koje obično možete rešiti pomoću MEL-a. Na primer, skript tmRenderPass omogućava određen proces vizuelizovanja koji bi zahtevao dosta vremena ako biste ga podešavali ručno, u okruženju programa Maya. U ovom odeljku objasnićemo neke od osnovnih elemenata MEL skriptova.

Sintaksa

Svaki jezik ima svoju sintaksu. U engleskom se svaka rečenica završava tačkom. U MEL-u, komanda se završava tačkom i zarezom (**;**). Savladavanje interpunkcije ili posebnih znakova spada među prve stvari koje morate uraditi. Ukoliko je sintaksa MEL skripta netačna, Maya će vratiti poruku o grešci. Zato je bitno da koristite ispravnu sintaksu dok pišete skript. Sledeća tabela prikazuje uobičajene znakove i daje njihove opise.

//	Dvostruka kosa crta označava komentar.
;	Tačka i zarez se koristi za označavanje kraja komande.
()	Zagrade služe za grupisanje matematičkih izraza.
[]	Uglaste zagrade se koriste za postavljanje indeksa niza u zagradu.
{}	Vitičaste zagrade se koriste za grupisanje komandi i nizova.
" "	Navodnici znače da tekst koji u njima stoji treba da bude tretiran kao tekstualni niz za komandnu liniju, te da eventualni posebni znakovi koji se u njemu koriste ne treba da utiču na izvršavanje skripta.

Promenljive

Promenljive su kamen temeljac bilo kog programskog jezika. One se koriste za privremeno čuvanje informacije koja će se koristiti bilo gde u skriptu. Da bi program Maya mogao da razlikuje promenljivu od imena nekog drugog objekta ili čvora u sceni, na početak imena promenljive postavlja se znak $.

Promenljive mogu sadržati različite vrste podataka. To može biti ceo broj, tekst ili vektor. (Pogledajte poglavlje 18 u kom se nalaze definicije različitih vrsta promenljivih.) Pre nego što upotrebite promenljivu, bitno je da je deklarišete u skriptu tako što ćete je imenovati i označiti vrstu podataka koje sadrži. Evo primera deklarisanja promenljive:

```
float $tmVal = 5 ;
```

Prethodni kôd govori da je promenljiva nazvana `$tmVal` broj s pokretnim zarezom i da je jednaka broju *5*. Pošto je promenljiva definisana, možete je koristiti u matematičkoj jednačini. U sledećem redu koristićemo komandu `print` koja će programu reći da rezultat prikaže u oknu History prozora Script Editor i na traci s povratnim informacijama:

```
print ($tmVal + 1) ;
//Result : 6
```

Umesto definisanja promenljive pomoću konstantnog broja, mogli biste je definisati i pomoću MEL komande:

```
string $tmMaterials[] = `ls –mat` ;
```

U ovom primeru, promenljivu `$tmMaterials` definisali smo kao vrstu podataka „string“ (što znači da može da sadrži tekst). Uglaste zagrade [] nakon imena promenljive znače da promenljiva može sadržati listu objekata. Promenljiva je potom definisana komandom `` `ls –mat` `` koja će nabrojati sve materijale u sceni.

Uslovni iskazi

Dok pišete skriptove, često ćete hteti da obavite izvesnu funkciju samo ako je ispunjen određen uslov: ako „ovo“ uradi „ono“ inače uradi „ono drugo“. Evo primera:

```
if ($x > 1){
print "X je veće od 1";
} else {
print "X nije veće od 1";
}
```

Procedure

Procedure omogućavaju da često korišćen kôd objedinite u jednu funkciju koja se može izvršiti unošenjem njenog imena. Na primer, možda imate skript koji izvodi nekoliko različitih zadataka, ali svaki zadatak zahteva otkrivanje materijala koji postoje u sceni (kao raniji primer promenljive). Umesto da u svakom zadatku pišete kôd za nabrajanje svih materijala, napravićete proceduru koja će raditi samo to: pronalaziće i nabrajaće materijale u sceni i vratiće njihova imena u vidu liste. Potom možete pozvati tu proceduru u svakom zadatku.

Prvo ćemo deklarisati proceduru iskazom `proc`:

```
proc string[] getMaterials()
{
string $materials[] = `ls -mat`;
return $materials;
}
```

Ova procedura će vratiti niz vrednosti koji sadrži imena materijala u sceni. Umesto da ceo ovaj kôd upisujete u svaki skript kom je ta informacija potrebna, možete jednostavno pozvati proceduru `getMaterials()`.

Postoje dve vrste procedura, lokalne i globalne. Procedure su podrazumevano prepoznaju samo lokalno. To znači da, ako imate proceduru napisanu na MEL-u, ona može biti pozivana samo unutar tog MEL skripta. Ukoliko biste ime procedure upisali u Script Editor, Maya bi vratila poruku o grešci. Ako ispred procedure dopišete reč *global*, ona će biti poznata celom programu Maya. To znači da je možete pozivati iz Script Editora, preko ikonice na polici, ili čak preko drugog MEL skripta. Kad god je moguće, koristite lokalne procedure, a globalne upotrebljavajte samo kada će ih korisnik pozivati iz Script Editora ili Mayinog korisničkog okruženja.

Analiza: korišćenje MEL-a za olakšavanje prelaska između IK i FK

U poglavlju 10 ste naučili kako da podesite ruku koja se može prebacivati između normalne i inverzne kinematike. Mada taj odeljak objašnjava podešavanje skeleta i kontrola za prebacivanje, izostavljen je jedan detalj koji je animatorima neophodan ukoliko hoće da položaj i orijentacija IK skeleta odgovara FK skeletu. Time se sprečava poskakivanje ruke prilikom prelaska između dve kinematike.

Dođite do rešenja

Kako biste ovaj problem mogli rešiti skriptom? Kada se suočite sa ovakvim problemima, prvi korak je da vidite kakav je to proces – zapitajte se „koji su koraci uključeni u ručno obavljanje ovog zadatka"? Najbolji način da to uradite jeste da sve neophodne akcije obavite u glavnom okruženju programa Maya.

Kako biste ovaj posao uradili ručno? Mogli biste pomeriti i priljubiti IK kontrolu ručnog zgloba uz FK kontrolu ručnog zgloba, ali biste time rešili problem poklapanja samo translacija, ne i rotacija. Treba da otkrijete kako biste to mogli uraditi skriptom. Prianjanje omogućava alatki Move da priljubi objekte uz određene komponente, pa ga ne možemo koristiti u skriptu. Kopiranje vrednosti translacije i rotacije ne bi radilo zato što će se transformacije svakog objekta blago razlikovati u zavisnosti od njihovih roditeljskih objekata.

Dobro rešenje bi bilo da IK kontrolu ograničite na kretanje FK kontrole ručnog zgloba pomoću usmerivača Point i Orient i otkrijete koje su vrednosti transformacije IK kontrole na tom položaju. Potom biste kopirali te vrednosti, obrisali usmerivač i vratili kopirane vrednosti.

Pošto ćete pisati skript, dobro bi bilo da napravite kopiju IK kontrole i ograničite je na FK kontrolu ručnog zgloba. Tako ćete ciljne vrednosti transformacije lako smestiti u promenljivu. Pošto završite zadavanje tih vrednosti za originalnu IK kontrolu, možete obrisati celu kopiju. Time ćete rešiti problem otkrivanja usmerivača koje treba kopirati.

Napišite skript

Prođite kroz sledeći skript. Za svaki red dat je komentar koji objašnjava proces. Neki redovi su prelomljeni zbog širine strane.

```
//Pomera armCTRL uz LT_Wrist_FK
//Ovaj red pravi kopiju kontrole LT_WristCTRL i smešta
//objekat u promenljivu nazvanu $dup.
$dup = `duplicate "LT_ WristCTRL"` ;
//Komanda select se koristi za biranje objekta LT_Wrist_FK
//i promenljive $dup navedenim redom
select "LT_Wrist_FK" $dup[0] ;
//Promenljiva $dup je usmerivačima Point i Orient usmerena na LT_Wrist_FK
pointConstraint ;
orientConstraint ;
//Translacija za $dup smeštena je u promenljivu nazvanu
//$pos, a rotacije su smeštene u promenljivu $rot
$pos = `getAttr ($dup[0] + ".t")` ;
$rot = `getAttr ($dup[0] + ".r")` ;
//Vrednosti u $pos koriste se za zadavanje atributa translacije
//i rotacije za kontrolu LT_WristCTRL
setAttr ("LT_WristCTRL" + ".t") $pos[0] $pos[1] $pos[2];
setAttr ("LT_WristCTRL" + ".r") $pos[0] $pos[1] $pos[2] ;
//Promenljiva $dup i njeni usmerivači obrisani su iz scene.
delete $dup[0] ;
//Prikazaćemo rezultat da bismo znali da je postupak uspeo
print ("//Result:"+" LT_WristCTRL"+" to " +"LT_Wrist_FK" + "\n");
// završeno


//Pomera IKPole uz FKPole
$dup = `duplicate "FKPole"`;
select $dup[0] "FKPole";
pointConstraint;
$pos = `getAttr ($dup[0] + ".t")`;
setAttr ("LT_ElbowCTRL" + ".t") $pos[0] $pos[1] $pos[2];
delete $dup[0];
print ("// Result: " + "LT_ElbowCTRL" + " moved to the position
of " + "FKPole" + "\n");
// završeno
```

Kada napište ovaj skript, možete ga izabrati u Script Editoru i prevući na policu sa ostalim kontrolama lika koje na njoj već postoje. Tako ćete brzo izvršiti skript dok animirate lik.