IZPOPOLNJENA INTERAKTIVNOST

Ta članek je nadaljevanje prejšnjega, v katerem smo obravnavali osnove programskega jezika Flasha, actionscript. V tem članku bomo skozi primere spoznali malce bolj zahtevno uporabo actionscripta, kljub temu pa bo še vedno precej daleč od pravega programiranja in doseganja profesionalnih interaktivnih učinkov, kot jih lahko zasledimo ponekod na internetu. Temeljil bo na konkretnih primerih, saj je tako najlažje predstaviti določene ukaze, ki so začetnim uporabnikom težje razumljivi.

Vlečenje objekta

Izdelali bomo preprost objekt, ki ga z miško lahko vlečemo po prizorišču. To je eden od najučinkovitejših načinov vplivanja in odzivanja uporabnikov na dogodke in hkrati eden osnovnih načinov, ki se uporablja pri večini iger in tudi kot učinkovita učna metoda.

Primer najlažje izvedemo z uporabo primerka gumba. Razlika v primerjavi z dosedanjim obravnavanjem gumbov in njim pripetih akcij v prejšnjih člankih je v tem, da je treba primerek simbola na prizorišču poimenovati. To storimo v oknu z lastnostmi, kot prikazuje slika 1.

V zgornjem primeru smo primerek simbola gumba poimenovali *vlecni_gumb*. To poimenovanje je potrebno zato, da se lahko v programski kodi actionscrtipta pozneje sklicujemo nanj (bolj nazorno bomo videli na primeru). Drugi razlog pa je, da imamo seveda lahko na prizorišču več primerkov istega simbola in jih tako ločimo med seboj. Ker

Slika 1. Poimenovanje primerka simbola.

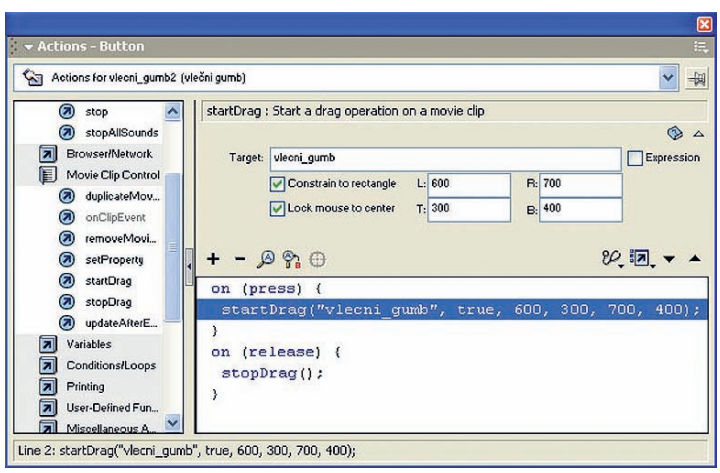
imamo torej opraviti s programskim jezikom actionscripta, je pri poimenovanjih primerkov simbolov treba upoštevati kar nekaj pravil:

✗ imena ne smejo imeti presledkov (napačno: *vlecni gumb*; pravilno: *vlecni_gumb*, *vlecni-gumb*);
✗ imena ne smejo imeti šumnikov (č, š, ž) in drugih posebnih črk abeced različnih jezikov (ć, đ, ö, ü itn.);
✗ imena ne smejo vsebovati posebnih znakov (?, !, *, #, & itn.).

Akciji, s katerima dosežemo vlečenje gumba, sta startDrag in stopDrag. Sta v orodjarni akcij pod Action → Movie Clip Control. Programsko kodo uporabe teh dveh akcij prikazuje slika 2. Tam vidimo, kje v programski kodi smo uporabili ime primerka simbola. Pri tem moramo paziti, da je vlečenje gumba pogojeno z dogodkom *press* (pritisni) in prenehanje vlečenja z dogodkom *release* (izpusti). Če kodo »prevedemo« v nam razumljiv jezik, pomeni: ob pritisku miške na gumb z imenom *vlecni_gumb* tega začni vleči (sledi gibanju miške), ob izpustu miške prenehaj vleči.

Iz slike vidimo tudi, da smo nastavili še dve dodatni možnosti: *Constrain to rectangle* in *Lock mouse to center*. S pomočjo prve nastavljamo štiri mejne vrednosti položaja, do koder se objekt lahko giblje: L – left (levo), R – right (desno), T – top (zgoraj), B – bottom (spodaj). Te vrednosti podajamo v točkah, podobno kot dimenzije dokumenta, kar smo spoznali v enem prvih člankov. Druga nastavitev pa pomeni, da se nam ob kliku miške na gumb ta postavi na sredino miškinega kurzorja.

Skoraj identično lahko vlečemo tudi animirane izrezke (Movie Clip). Primeri dveh vlečnih gumbov in enega animiranega izrezka so na spletni strani Grafičarja.



Slika 2. Programska koda za vlečenje objekta (gumba).

Vpisovanje in izpis podatkov

Drugo zanimivo področje interaktivnosti uporabnika je vnos in izpis podatkov. Nekatera najbolj uporabna področja tega načina uporabe so:

✗ vpraševanje uporabnika po imenu;
✗ postavljanje vprašanj uporabniku in spremljanje njegovih odgovorov;
✗ prikazovanje vsebine, ki se pogosto spreminja.

Na začetku izvedbe takih nalog moramo najprej razjasniti dva osnovna pojma računalniškega programiranja: ime in vrednosti spremenljivke. Z imenom spremenljivke določamo ime, kateremu bomo prirejali različne vrednosti, vrednost spremenljivke pa je trenutna vrednost določene spremenljivke. Zadeva zveni dokaj banalno, vendar iz izkušenj vem, da pri uporabi mešanje teh dveh pojmov ni tako redko. Dva primera prikazuje tabela 1.

Kot vidimo, vrednosti spremenljivke niso vedno številskega tipa, ampak so lahko tudi opisne. S spremenljivkami lahko v osnovi počnemo dvoje:

✗ določanje ali spreminjanje njihovih vrednosti;
✗ ugotavljanje in preverjanje njihovih vrednosti.

V prvem primeru v oknu z lastnostmi pri nastavitvah lastnosti teksta nastavimo možnost *Input Text* za vnašanje vrednosti, v

TABELA 1. SPREMENLJIVKI IN NJUNE VREDNOSTI

IME SPREMENLJIVKE	VREDNOST SPREMENLJIVKE
Hitrost	50 km/uro
Barva mize	Rdeča

drugem primeru pa možnost *Dynamic Text* za izpis vrednosti. Primer vnašanja in izpisovanja imena je na spletni strani Grafičarja.

Primer je izveden v dveh ključnih sličicah. V prvi je vnosno polje z nastavitvijo *Input Text*. To polje določimo kot spremenljivko z imenom *vpis_ imena*. Torej se tej spremenljivki ob vpisu imena priredi vrednost, to pa je ime, ki smo ga vpisali.

V drugi ključni sličici pa je polje za izpis, ki ima nastavitev *Dynamic Text*, temu polju pa smo dali ime *izpis*. V drugi celici smo nato vstavili akcijo določanja vrednosti spremenljivke: *izpis = "Pozdravljeni, " + vpis_imena + "!";* kar pomeni, da se v spremenljivki z imenom *izpis* priredi vrednost *"Pozdravljeni, " + vpis_imena + "!"*. To je v bistvu sestavljanje niza besed. Kar je v narekovajih, se normalno izpiše, kot vrednost spremenljivke *vpis_imena* pa se izpiše ime, ki smo ga vpisali ob vpisu imena.

Priznam, da je za začetnike to malce težje razumljivo, izkušenim programerjem pa je zelo poznano, ker je to ena izmed osnov programiranja s spremenljivkami. Slika 3 prikazuje primere opisanega postopka vnašanja in izpisovanja imena.

Spodaj slika 3.
Primer vpisovanja in izpisovanja imena.

Prvi del slike je prikaz prve ključne sličice in nastavitev lastnosti vnosnega polja, drugi del prikaz druge ključne sličice nastavitev lastnosti izpisnega polja, tretji del pa prikazuje akcijo izpisa v drugi ključni sličici.

Nastavljanje in spreminjanje lastnosti objektom

Tretji zanimiv primer je nastavljanje in spreminjanje lastnosti določenim objektom. Na spletni strani Grafičarja je primer nastavljanja in spreminjanja prosojnosti objekta, njegovo pomikanje in spreminjanje velikosti.

Podobno kot v prejšnjih dveh primerih smo primerek simbola na prizorišču poimenovali, v našem primeru kar *krog*. Nato gumbom pripenjamo akcije, s katerimi temu krogu spreminjamo lastnosti. Pri spreminjanju prosojnosti lahko nastavimo prosojnost *alpha* na 50 odstotkov in jo spreminjamo v korakih po deset, pri pomikanju objekta spreminjamo vrednosti koordinat x in y danega objekta za deset enot, pri spreminjanju višine in širine pa to spreminjamo prav tako po deset enot. Akcije za opisani primer prikazuje tabela 2.

V tabeli vidimo, da smo uporabili lastnosti SetProperty, ki je pod Action → Movie Clip Control. V oknu z akcijami nato v spustnem seznamu lahko izbiramo med osnovnimi lastnostmi, kot so višina, širina, obrat, prosojnost itn.

Povzetek

V tem članku smo skozi posamezne primere spoznali malce bolj zahtevno uporabo actionscripta, ki uporabniku omogoča večjo interaktivnost z izdelki, narejenimi v Flashu. Poudariti moram, da je to še vedno osnovna uporaba actionscripta, saj je pri izdelavi zahtevnih projektov, ki vsebujejo pravo uporabnikovo vplivanje na rezultat (razne spletne igre, izdelane v Flashu), koda actionscripta dolga več strani. Vendar smo kljub temu v tem članku spoznali osnove premikanja objekta po prizorišču, tehnike vpisovanja in izpisovanja podatkov in osnove nastavljanja in spreminjanja lastnosti.

Primeri, povezani s tem člankom, so na spletni strani *www.delo.si/graficar* (*začasno v rubriki ZADNJA ŠTEVILKA, kasneje v rubriki oz. v oknu ARHIV/Grafičar 2007/Grafičar 3/2007*).

Andrej ISKRA
Univerza v Ljubljani

TABELA 2. AKCIJE NASTAVITEV PROSOJNOSTI IN PREMIKANJE OBJEKTA

NASTAVLJANJE PROSOJNOSTI	PREMIKANJE OBJEKTA	SPREMINJANJE ŠIRINE IN VIŠINE

on (release) {
setProperty("krog", _alpha, "50");
}
	on (release) {
	krog._y -= 10;
	}
		on (release) {
		krog._width += 10;
		}

on (release) {
krog._alpha -= 10;
}
	on (release) {
	krog._y += 10;
	}
		on (release) {
		krog._width -= 10;
		}

on (release) {
krog._alpha += 10;
}
	on (release) {
	krog._x -= 10;
	}
		on (release) {
		krog._height += 10;
		}

	on (release) {
	krog._x += 10;
	}
		on (release) {
		krog._height -= 10;
		}