DUM č. 19 v sadě

28. Inf-4 Jednoduchá hra Had ve Flashi (ActionScript)

Autor: Robert Havlásek

Datum: 08.06.2013

Ročník: 5AV

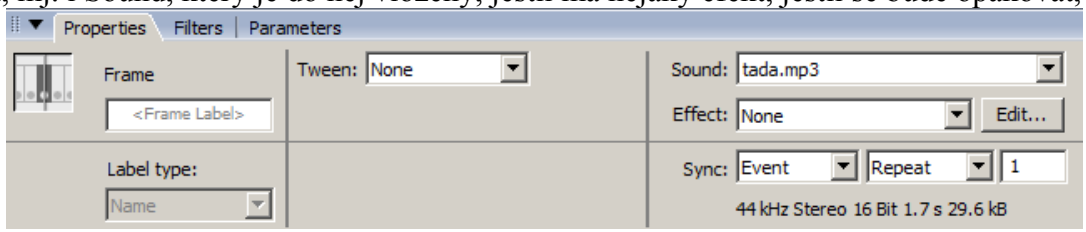
Anotace DUMu: Flash - teorie: Zvuky ve Flashi. Skoky mezi jednotlivými snímky animace uvnitř skriptu.\n

esf evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

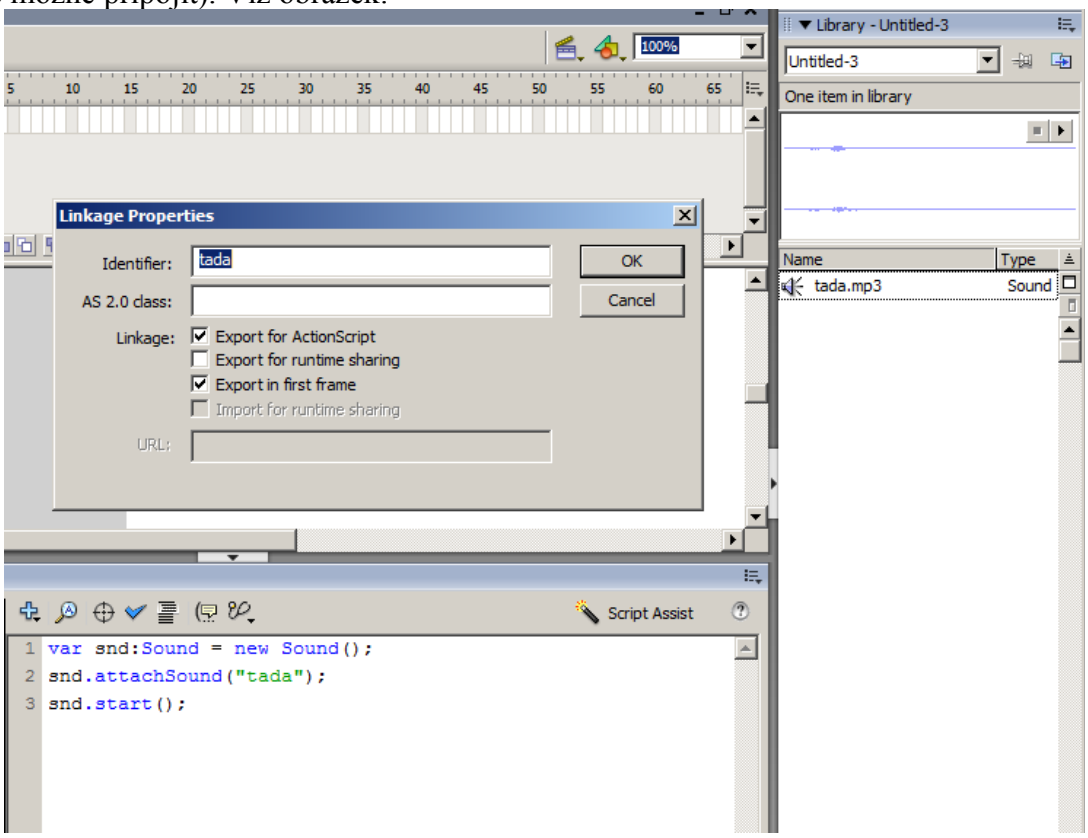
Zvuky ve Flashi

Je několik způsobů, jak dostat do flashové animace zvuk.

První možností je, že jej přetažením Drag&Drop z nějakého windowsového okna posuneme do Flashe do plochy nějakého konkrétního frame (tím dojde i k jeho importu do Library) a on při skoku na tento frame začne rovnou hrát. Hraje až do konce nebo do okamžiku, než v časové ose Flash narazí na keyframe. V časové ose dokonce vidíme charakteristickou křivku zvuku. Doporučuji mít pro každý zvuk/skladbu samostatnou vrstvu.
Označíme-li frame s vloženým zvukem kliknutím myší, v panelu Properties vidíme jeho vlastnosti, mj. i Sound, který je do něj vložený, jestli má nějaký efekt, jestli se bude opakovat, atd.:



Druhou možností je, že jej naimportujeme do Library (např. Drag&Drop do okna Library či v menu File-Import-Import to Library), poté jej připravíme pro použití v ActionScriptu (v Library na něj klikneme pravým tlačítkem, zvolíme Linkage… a v následujícím dialogu zatrhneme Export for ActionScript a napíšeme identifikátor, pod kterým jej v ActionScriptu bude možné připojit). Viz obrázek:



Na obrázku vidíme zvuk tada.mp3 naimportovaný do Library a připravený pro použití v ActionScriptu s identifikátorem "tada", dole v kódu je pak zavedena nová proměnná snd (příkazem `var snd:Sound = new Sound();`), do ní připojen zvuk s identifikátorem "tada" (příkazem `snd.attachSound("tada");`) a ten následně přehrán (příkazem `snd.start()`). Zavoláme-li poté v kódu libovolného objektu `_root.snd.start()`, zvuk se pokaždé přehraje.

První i druhá možnost mají výhodu (a občas i nevýhodu) tu, že se zvuky/hudba při kompilaci do .swf rovnou uloží do zkompilovaného souboru. Soubor .swf je tak poměrně velký, ale zase je samostatný, nepotřebuje žádné další externí, vedle ležící soubory.

Třetí možností je využití externího souboru, který přečteme až v okamžiku, kdy se rozhodneme, že jej budeme potřebovat. Pojmem „externí soubor“ lze chápat nejen fyzický soubor, ale i tzv. stream, tedy datový tok stahovaný ze sítě, např. prostřednictvím protokolu http.
V principu existují dvě metody přístupu – nestreamovaný zvuk a streamovaný zvuk. Ten nestreamovaný se celý načte do paměti (celý jeho obsah se získá z disku, ze sítě), pak Flash zavolá událost onLoad a teprve potom jej lze přehrávat, a to i opakovaně. Ten streamovaný stačí začít načítat a můžete jej rovnou začít přehrávat (s rizikem, že bude-li přehrávání rychlejší, Flash situaci vyhodnotí tak, že přehrávání chvíli pozastaví).
Srovnejte streamovanou variantu:

```
var snd:Sound = new Sound();
snd.loadSound("tada.mp3",true); //true znamena streamovany zvuk
snd.start();
```

a nestreamovanou variantu:

```
var snd:Sound = new Sound();
snd.onLoad = function(success:Boolean) {
    if (success) snd.start(0,3);
    }
snd.loadSound("tada.mp3",false); //false znamena nestreamovany zvuk
```

V té nestreamované si povšimněte, že jsme objektu snd zavedli obsluhu události onLoad s parametrem `success` – když se zvuk celý stáhnul v pořádku, přehraj jej `snd.start(0,3);`, tedy od začátku, třikrát po sobě.

Namísto lokálního souboru lze v loadSound použít i URL adresu, např.

```
snd.loadSound("http://www.gml.cz/tada.mp3",false);
```

či

```
snd.loadSound("http://www.gml.cz/tada.mp3",true);
```

Doplnění zvuku do hry Had při snězení bonusového jablka

Nahoru do kódu plochy (např. za přiřazení textcas) přidáme

```
var snd:Sound = new Sound();
snd.attachSound("tada");
```

a do obsluhy bonusového jablka přidáme `_root.snd.start();`. Tedy:

```
if ((jablek % pocetjablekprobonus) == 0)
                            {zivoty++;_root.textzivoty='Životy: '+zivoty;
                             _root.snd.start();}
```

Více snímků v animaci

Do této chvíle jsme celou hru Had kreslili v jediném frame. Celá animace „nestála na místě“, ale neustále znovu dokola opakovala všechny svoje frame, tedy tento jediný.
Přidáme-li do naší animace další frames (třeba další dva, jeden před hru a jeden za hru), budou se danou rychlostí (defaultně 12 fps) provádět všechny tři framy postupně, čili onEnterFrame ve hře se bude volat čtyřikrát za sekundu. Prostředí hry přitom bude „blikat“ spolu s dalšími nakreslenými framy.
Chceme-li animaci zastavit, aby nerotovaly všechny tři framy, ale jen druhý, musíme v ploše druhého snímku, jenž má rotovat na místě, zavolat funkci `stop()`. To způsobí, že se sice nejprve zobrazí první, pak druhý a pak se začne cyklicky animovat pouze druhý snímek.

Podobně, budeme-li chtít si v prvním snímku s uživatelem „povídat“ (např. TextInputem číst jeho jméno, tlačítkem vyzývat ke spuštění hry, pomocí komponenty NumericStepper nastavovat obtížnost, atp.), zavoláme i v prvním snímku `stop()`.

Sdílení informací mezi snímky

Je důležité zmínit, že samotná plocha je všem framům společná – můžeme v nich využívat společné proměnné, společné funkce. Rovněž lze jako společné využívat všechny objekty, které v obou daných framech existují. Jestliže ale například druhý frame vyrobíme v časové ose jako Blank Keyframe (nebo při návrhu druhého keyframu komponenty smažeme), komponenty prvního už využívat nelze.

Čili situace „uživatel nejprv něco vyplní a pak podle toho začne hrát“ lze řešit buď tak, že komponenty z prvního snímku ve druhém ponecháme, pouze je zneviditelníme, nebo lépe tak, že v prvním snímku hodnoty, jež chceme přenést dál, přiřadíme do speciálních nových proměnných plochy a druhý snímek bude Blank Keyframe.

Skoky mezi snímky

Chceme-li z jednoho snímku skočit na jiný, lze na to použít několik různých funkcí vázaných k ploše:
`nextFrame()` … skočí na následující frame a chová se stejně (byla-li animace stopnutá, zůstane stopnutá; nebyla-li, poběží dál).
`prevFrame()` … skočí na předchozí frame a chová se stejně
`gotoAndPlay()` … skočí na frame, jehož číslo dostane jako parametr (počítají se na časové ose od nuly). Namísto čísla lze použít i jméno framu, pokud mu je v panelu Properties předtím zadáme. Bez ohledu na to, zda byla či nebyla animace stopnutá, se nyní rozběhne a po framu, na něhož jsme skočili, provádí i další framy v pořadí za ním.
Funkci `gotoAndPlay()` je možné použít nejen pro skok na frame, ale i pro skok do jiné scény na konkrétní frame, pokud jméno scény zadáme jako první parametr.
`gotoAndStop()` … podobně jako gotoAndPlay() i gotoAndStop skočí na konkrétní snímek, ale zůstane na něm stát a provádí opakovaně jenom jej.