DUM č. 8 v sadě

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

Autor: Robert Havlásek

Datum: 27.03.2013

Ročník: 5AV

Anotace DUMu: Flash - příklad: Využití pole pro rozpis zdí. Statické nastavení jednotlivých prvků pole (rozmístění zdí).\nKreslení zdí pomocí duplikování cihly (připravené mimo plochu).\nTestování, zda hlava hada nenarazila do zdi.

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

Tento DUM je součástí sady, v níž programujeme Flashovou hru Had. Pokračujeme úpravou kódu, který vznikl v DUMu č. 6., do Actions plochy navíc přidáme kód plnící pole zdix a zdiy okolními zdmi – z konce DUMu č. 7. Výchozí stav je tedy:

v ploše:

```
var zdix:Array = Array();
var zdiy:Array = Array();
for (i=0; i<=310; i=i+10) {zdix.push(i);zdiy.push(0);}  // horni rada
for (i=0; i<=310; i=i+10) {zdix.push(i);zdiy.push(190);}  // spodni rada
for (i=0; i<=190; i=i+10) {zdix.push(0);zdiy.push(i);}  // leva rada
for (i=0; i<=190; i=i+10) {zdix.push(310);zdiy.push(i);}  // prava rada

function smrt() {
      _root.hlava._x=160;
      _root.hlava._y=100;
      _root.hlava.smer=4;
    }
```

v objektu hlava:

```
onClipEvent(enterFrame)
{ if (smer==0) {_root.hlava._x = _root.hlava._x + 10;}
   if (smer==1) {_root.hlava._y = _root.hlava._y - 10;}
   if (smer==2) {_root.hlava._x = _root.hlava._x - 10;}
   if (smer==3) {_root.hlava._y = _root.hlava._y + 10;}
   if (_root.hlava._x>=320) {_root.hlava._x=0;}
   if (_root.hlava._y>=200) {_root.hlava._y=0;}
   if (_root.hlava._x<0) {_root.hlava._x=310;}
   if (_root.hlava._y<0) {_root.hlava._y=190;}
/*   if ((_root.hlava._x>=320) || (_root.hlava._y>=200) ||
       (_root.hlava._x<0) || (_root.hlava._y<0)) _root.smrt();  */
   }
```

Tvorba zdí

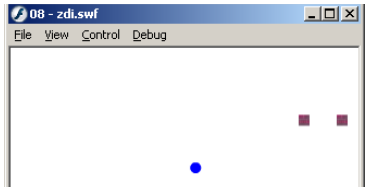
V tomto DUMu použijeme dvě pole, v nichž jsou (ručně nebo cyklem) zadány souřadnice zdí, obrázek zdi do těchto souřadnic duplikujeme a začneme zjišťovat, zda had do zdi nenarazí.

Praktický úkol: V bitmapovém editoru si nakreslete obrázek o rozměrech 10×10 pixelů s několika cihličkami zdi – tak, aby bylo možné cihličky sázet vedle sebe a přitom navazovaly – např.  *. Obrázek naimportujte do animace, vyrobte z něj symbol typu MovieClip, instance bude mít jméno* `vzorzdi`.

Studentům dále předvedeme duplikaci objektu – zavoláním funkce `duplicateMovieClip`, jako parametry zadáváme: původní objekt, nový objekt a hloubku, do níž se nový objekt usadí. Při zjišťování hloubky lze s úspěchem použít funkce `getNextHighestDepth()`, která vrátí nejvyšší hloubku – klon tedy bude nad všemi předchozími objekty. S originálem je pak vhodné pohnout, aby nebyly originál a klon na stejném místě. Vyzkoušíme v Actions plochy:

```
duplicateMovieClip(_root.vzorzdi,"klonzdi",getNextHighestDepth());
_root.vzorzdi._x=300;   // odskočím s originálem
```

Po stisku CTRL+Enter by měl Flash vyrobit klon a s originálem odskočit do souřadnice x=300:



Budeme-li chtít v budoucnu k jednotlivým klonům přistupovat, je vhodnější si je pojmenovat individuálně, například pomocí proměnné:

```
duplicateMovieClip(_root.vzorzdi,"klonzdi["+k+"]",getNextHighestDepth());
k=k+1;
```

Praktický úkol: S využitím polex a poley zduplikujte zeď do míst okolo hrací plochy. Originál zdi můžete ponechat v místě posledního klonu.
Řešení:
```
for (k=0; k<zdix.length; k=k+1){
  _root.vzorzdi._x=zdix[k];
  _root.vzorzdi._y=zdiy[k];
  duplicateMovieClip(_root.vzorzdi,"klonzdi["+k+"]",getNextHighestDepth());
  }
```
Neboli: Proměnnou k necháme běžet od 0 do délky pole zdix (nevčetně), pokaždé přesuneme vzor do místa, kam se má udělat duplikát, a ten uděláme.

Výmaz klonu

Podobně jako `duplicateMovieClip` vyrobí klon, tak jej `removeMovieClip` smaže. Jako parametry zadáváme pouze jméno. Například `removeMovieClip("klonzdi[0]")`.
Funkce `removeMovieClip` umí mazat jen objekty, které vznikly za běhu programu, tedy neumí (v našem případě) smazat `_root.vzorzdi`.
Pozor, navíc má Flash8 poměrně nepříjemný bug: neumí mazat objekty, které jsou „moc vysoko“, čili při tvorbě byla použita moc velká výška (nebo vznikly moc pozdě s použitím `getNextHighestDepth()`). To lze vyřešit tím, že výšku objektu před smazáním prohodíme metodou `swapDepths` – viz později. Nebo to lze vyřešit i tak, že „statické“ objekty budeme kreslit s vyšší hloubkou a ty, jenž mažeme, s nižší hloubkou.

Náhodná funkce

Chceme-li získat náhodné číslo, použijeme funkci `random`, která je součástí knihovny `Math`, voláme ji jako `Math.random(číslo)`, vrátí celé číslo mezi 0 a číslo–1.
Zavoláme-li například `Math.random(5)`, vrátí nám buď `0` nebo `1` nebo `2` nebo `3` nebo `4`.

Náhodné zdi

Máme-li umístit náhodně zeď (s vyloučením již vyrobených okrajových zdí), může její xová souřadnice nabývat hodnot 10, 20, 30, …, 300. Tedy celkem 30 možností. Velmi zajímavá je geneze, jak vznikne vzorec (doporučuji probrat se studenty):
`random(30)` vrací číslo mezi 0 a 29, tedy jednu z třiceti možností
`1+random(30)` vrací číslo mezi 1 a 30
`10+10*random(30)` vrací číslo dělitelné deseti mezi 10 a 300.

Praktický úkol: Kromě existujících zdí okolo vyrobte ještě 20 náhodně rozmístěných zdí uvnitř plochy. Jejich souřadnice uložte jako další čísla do zdix a zdiy.
Řešení:
```
for (k=0; k<20; k=k+1){
  _root.vzorzdi._x=10+10*random(30);
  _root.vzorzdi._y=10+10*random(18);
  zdix.push(_root.vzorzdi._x);
  zdiy.push(_root.vzorzdi._y);
  duplicateMovieClip(_root.vzorzdi,"klonzdi["+(zdix.length-1)+"]",getNextHi
  ghestDepth());
  }
```

Namísto `+k+` v klonování zdi jsme použili `+(zdix.length-1)+`. Ono `+k+` by znamenalo, že budeme tvořit položky pole, které už existují z předchozí tvorby (k jde v tomto druhém cyklu opět od nuly). Na tento fakt obvykle studenti nepřijdou, je lépe jim to říct.

Předchozí kód neřeší dva logické nesmysly: případ, kdy jsou dvě náhodné zdi na stejném místě, a případ, kdy se náhodná zeď umístí na [160,100], tedy na hlavu hada… (A protože je výš než hlava hada, není hlava pod ní vidět.)
Pedagogická poznámka: Na oba nesmysly mohou přijít i sami studenti, stačí zvýšit počet náhodně rozmisťovaných zdí a vyzvat je, aby víckrát zkoušeli program spustit a důsledně sledovali, kde co nastane špatně.

Řešení situace, kdy je náhodná zeď na stejném místě, jako hlava

Postačí jednoduchá podmínka před pamatováním souřadnic a duplikací:

```
if ((_root.vzorzdi._x==160)&&(_root.vzorzdi._y==100)) k=k-1
   else {zdix.push…
```

Studenti, povšimněte si příkazu `k=k-1`, který říká: „Tento k-tý případ je špatně, vraťme k o jedno zpět a zkusme to znovu v dalším kole cyklu.“

Řešení situace, kdy jsou náhodné zdi dvě na stejném místě

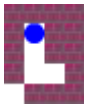
Zjištění, zda na generovaném místě už nějaká zeď existuje, nejde udělat jednou jednoduchou podmínkou. Potřebujeme cyklus procházející všechny existující zdi (od 0 do zdix.length–1); navíc zavedeme proměnnou obsazeno, která na začátku bude 0 (nepravda) a zjistíme-li v průběhu, že dané místo už zdí obsazené je, dáme jí hodnotu 1:

```
  var obsazeno=0;
  for (m=0; m<zdix.length; m=m+1)
     if ((_root.vzorzdi._x==zdix[m])&&(_root.vzorzdi._y==zdiy[m]))
        obsazeno=1;   // najdeme-li v prubehu cyklu zed na stejnem miste
  if ( (obsazeno==1) or
     ((_root.vzorzdi._x==160)&&(_root.vzorzdi._y==100)) ) {k=k-1}
    else {  zdix.push…
```

Pedagogická poznámka: Budeme-li chtít funkčnost cyklu a podmínky pro „obsazeno“ otestovat, je nejlepší omezit množství generovaných zdí např. na 4 a upravit jednu souřadnici na generování čísel mezi 0, 10 a 20, například (podžluceny jsou testovací změny):

```
for (k=0; k<4; k=k+1){
  _root.vzorzdi._x=10+10*random(30);
  _root.vzorzdi._y=  10*random(3);
  var obsazeno=0;
  for (m=0; m<zdix.length; m=m+1)
    if ((_root.vzorzdi._x==zdix[m])&&(_root.vzorzdi._y==zdiy[m]))
      obsazeno=1;
  if ( (obsazeno==1) or
     ((_root.vzorzdi._x==160)&&(_root.vzorzdi._y==100)) ) {k=k-1}
    else {  zdix.push(_root.vzorzdi._x);
            zdiy.push(_root.vzorzdi._y);
            duplicateMovieClip(_root.vzorzdi,"klonzdi["+(zdix.length-1)+"]"
            ,getNextHighestDepth());}
  }
```

Spustím-li tento kód opakovaně, musí být v horní části vidět vždy přesně čtyři cihličky, bez ohledu na fakt, že se některé z nich nejprv chybně vygenerují do nulové výšky (již existujících obvodových cihel).

Výše uvedené techniky nicméně stejně nezabrání náhodné situaci typu  nebo  atp. Jediný způsob, jak takové situaci zamezit, je použití grafových vyhledávacích algoritmů, které ověří, že z pozice hlavy hada je každé bílé místo dostupné… A v jistých případech ani ověření dostupnosti nestačí, například, vyrobíme-li uličku, v níž se had nemůže otočit…
Pedagogická poznámka: Studenti – programátoři – toto je velmi zajímavé téma na maturitní program…

Pro zpřehlednění situace uvádím aktuální funkční kód generující okolní i náhodné zdi:

v ploše:

```
var zdix:Array = Array();
var zdiy:Array = Array();
var klonzdi:Array = Array();
// TVORBA POLE ZDI OKOLO:
for (i=0; i<=310; i=i+10) {zdix.push(i);zdiy.push(0);}  // horni rada
for (i=0; i<=310; i=i+10) {zdix.push(i);zdiy.push(190);}  // spodni rada
for (i=0; i<=190; i=i+10) {zdix.push(0);zdiy.push(i);}  // leva rada
for (i=0; i<=190; i=i+10) {zdix.push(310);zdiy.push(i);}  // prava rada
// FYZICKE KLONOVANI ZDI OKOLO:
for (k=0; k<zdix.length; k++){
  _root.vzorzdi._x=zdix[k];
  _root.vzorzdi._y=zdiy[k];
  duplicateMovieClip(_root.vzorzdi,"klonzdi["+k+"]",getNextHighestDepth());}
// NAHODNE ZDI A JEJICH KONTROLA:
for (k=0; k<20; k=k+1){
  _root.vzorzdi._x=10+10*random(30);
  _root.vzorzdi._y=10+10*random(18);
  // KONTROLA OBSAZENOSTI:
  var obsazeno=0;
  for (m=0; m<zdix.length; m=m+1)
    if ((_root.vzorzdi._x==zdix[m])&&(_root.vzorzdi._y==zdiy[m]))
      obsazeno=1;
  if ( (obsazeno==1) or
     ((_root.vzorzdi._x==160)&&(_root.vzorzdi._y==100)) ) {k=k-1}
    else {  zdix.push(_root.vzorzdi._x);
            zdiy.push(_root.vzorzdi._y);
            duplicateMovieClip(_root.vzorzdi,"klonzdi["+(zdix.length-1)+"]",
            getNextHighestDepth());}
  }

function smrt() {
      _root.hlava._x=160;
      _root.hlava._y=100;
      _root.hlava.smer=4;
    }
```

Testování, zda hlava hada nenarazila do zdi

Test, zda hlava hada nenarazila do zdi, budeme psát do kódu objektu hlava, hned pod pohyb. Použijeme podobný cyklus jako u kontroly obsazenosti (viz výše). Ponechal bych i původní názvy proměnných (`obsazeno`, `m`). Zdi nebudeme porovnávat se souřadnicemi vzoru nové zdi, ale se souřadnicemi hlavy hada.
Oproti výše uvedenému kódu se změní také oslovení pole zdix (musíme je oslovovat jako `_root.zdix`, protože pole zdix není definováno v aktuálním objektu, ale v ploše). Rovněž smrt() musíme z hlavy volat jako `_root.smrt()`.

```
var obsazeno=0;
for (m=0; m<_root.zdix.length; m=m+1)
  if ((_root.hlava._x==_root.zdix[m])&&(_root.hlava._y==_root.zdiy[m]))
    obsazeno=1;
if (obsazeno==1) _root.smrt();
```

Aktuální kód v objektu hlava:

v objektu hlava:

```
on (keyPress "<Left>") {smer=2;}
on (keyPress "<Right>") {smer=0;}
on (keyPress "<Up>") {smer=1;}
on (keyPress "<Down>") {smer=3;}

onClipEvent(enterFrame)
 { if (smer==0) {_root.hlava._x = _root.hlava._x + 10;}
   if (smer==1) {_root.hlava._y = _root.hlava._y - 10;}
   if (smer==2) {_root.hlava._x = _root.hlava._x - 10;}
   if (smer==3) {_root.hlava._y = _root.hlava._y + 10;}
   if (_root.hlava._x>=320) {_root.hlava._x=0;}
   if (_root.hlava._y>=200) {_root.hlava._y=0;}
   if (_root.hlava._x<0) {_root.hlava._x=310;}
   if (_root.hlava._y<0) {_root.hlava._y=190;}
   // KONTROLA NARAZENI HLAVY DO ZDI:
   var obsazeno=0;
   for (m=0; m<_root.zdix.length; m=m+1)
     if ((_root.hlava._x==_root.zdix[m])&&(_root.hlava._y==_root.zdiy[m]))
       obsazeno=1;
   if (obsazeno==1) _root.smrt();
/*   if ((_root.hlava._x>=320) || (_root.hlava._y>=200) ||
       (_root.hlava._x<0) || (_root.hlava._y<0)) _root.smrt();*/
   }
```