DUM č. 12 v sadě

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

Autor: Robert Havlásek

Datum: 06.04.2013

Ročník: 5AV

Anotace DUMu: Flash - příklad: Náhodně generované jablko. Test, jestli jablko nevygenerujeme do hlavy, do těla hada či do zdi.\nPokus s více jablky.

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

Náhodně generované jablko

Ano, mohli bychom jablko nakreslit dopředu, při návrhu plochy, a pak jej pouze posunout na vhodné místo. Umíme-li ale vyrobit prázdný MovieClip v programu a do něj červené kolečko nakreslit (viz předchozí DUM č. 11 na poslední straně), nic nám nebrání vyrobit jej za běhu programu.
Pokud navíc celý kód snězení jablka napíšeme rovněž do kódu jablka, je možné jablek vyrobit víc, na různých místech, každé bude reagovat na snězení stejně…

Na konec kódu plochy vložíme a mírně upravíme kód z posledního příkladu DUMu č. 11, který zní:

```
this.createEmptyMovieClip("circle2_mc", 2);
circle2_mc.beginFill(0xFF0000);
drawCircle(circle2_mc, 100, 100, 100);
circle2_mc.endFill();
function drawCircle(mc:MovieClip, x:Number, y:Number, r:Number):Void {
    mc.moveTo(x+r, y);
    mc.curveTo(r+x, Math.tan(Math.PI/8)*r+y, Math.sin(Math.PI/4)*r+x,
Math.sin(Math.PI/4)*r+y);
    mc.curveTo(Math.tan(Math.PI/8)*r+x, r+y, x, r+y);
    mc.curveTo(-Math.tan(Math.PI/8)*r+x, r+y, -Math.sin(Math.PI/4)*r+x,
Math.sin(Math.PI/4)*r+y);
    mc.curveTo(-r+x, Math.tan(Math.PI/8)*r+y, -r+x, y);
    mc.curveTo(-r+x, -Math.tan(Math.PI/8)*r+y, -Math.sin(Math.PI/4)*r+x,
-Math.sin(Math.PI/4)*r+y);
    mc.curveTo(-Math.tan(Math.PI/8)*r+x, -r+y, x, -r+y);
    mc.curveTo(Math.tan(Math.PI/8)*r+x, -r+y, Math.sin(Math.PI/4)*r+x,
-Math.sin(Math.PI/4)*r+y);
    mc.curveTo(r+x, -Math.tan(Math.PI/8)*r+y, r+x, y);
}
```

Namísto circle2_mc navrhuji použít jméno jablko1, do hloubky 41 (od hloubky 50 začínají zdi), střed nakreslíme relativně na [5,5], poloměr bude 5 (nikoliv 100). Funkci drawCircle lze napsat těsně nad funkci smrt();
Tedy po úpravách:

```
this.createEmptyMovieClip("jablko1", 41);
jablko1.beginFill(0xFF0000);
drawCircle(jablko1, 5, 5, 5);
jablko1.endFill();
```

Aktuálně jablko neuvidíme, protože je schované v pozici [0,0], kde MovieClip vznikl, za zdí.
Pro generování jablka použijeme podobnou náhodnou funkci jako u zdí:

```
jablko1._x=10+10*random(30);
jablko1._y=10+10*random(18);
```

Test, zda jablko není ve zdi

Jablko usadím na náhodnou pozici. Zjistím, jestli náhodou nebyla obsazená dřívější zdí, když ano, tak znovu jablko usadím na náhodnou pozici, znovu zjistím…
Jako nejvhodnější cyklus pro tento typ situace se jeví do-while cyklus (jeho syntaxe viz DUM č. 9, str. 2). Použijeme tradičně proměnnou obsazeno a cyklus procházející všechny zdi:

```
do {
  jablko1._x=10+10*random(30);
  jablko1._y=10+10*random(18);
  var obsazeno=0;
  for (m=0; m<zdix.length; m=m+1)
```

```
    if ((jablko1._x==zdix[m])&&(jablko1._y==zdiy[m]))
      {obsazeno=1;break;}
  } while (obsazeno==1);
```

Pedagogická poznámka: Chceme otestovat, zda se jablko opravdu nikdy netrefí do zdi? Jednoduše zvýšíme počet náhodných zdí z 20 třeba na (dramatických) 500… ☺

Test, zda jablko není v hlavě ani v těle

Do již hotového do-while cyklu napíšeme kromě for-cyklu procházející všechny zdi též for-cyklus procházející celé tělo:

```
  for (m=0; m<_root.hlava.telox.length; m=m+1)
    if ((jablko1._x==_root.hlava.telox[m])&&(jablko1._y==_root.hlava.telo
y[m]))
      {obsazeno=1;break;}
```

Hlavu zvlášť testovat nemusíme, její pozice v dané chvíli odpovídá telox[0], teloy[0], které se v cyklu rovněž testují.
Studenti mohou namítnout, že had stejně na začátku hry stojí v pozici [150,100], proč tedy zkoumat celé tělo, když stačí testnout tuto pozici… Odpověď je jednoduchá: Kód pro generování jablka tvoříme univerzálně, i pro situaci, kdy bylo jablko snězeno, had je roztažený po ploše a vyrábíme jablko nové… V přespříštím DUMu se nám to bude hodit.

Pro přehlednost celý kód plochy:

```
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+"]",50+getNextHighestDept
h());}
// 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;break;}
  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)+"]",ge
tNextHighestDepth());}
  }
```

```
// TVORBA JABLKA:
this.createEmptyMovieClip("jablko1", 41);
jablko1.beginFill(0xFF0000);
drawCircle(jablko1, 5, 5, 5);
jablko1.endFill();
do {
  jablko1._x=10+10*random(30);
  jablko1._y=10+10*random(18);
  var obsazeno=0;
  for (m=0; m<zdix.length; m=m+1)
    if ((jablko1._x==zdix[m])&&(jablko1._y==zdiy[m]))
      {obsazeno=1;break;}
  for (m=0; m<_root.hlava.telox.length; m=m+1)
    if ((jablko1._x==_root.hlava.telox[m])&&(jablko1._y==_root.hlava.telo
y[m]))
      {obsazeno=1;break;}
  } while (obsazeno==1);

function drawCircle(mc:MovieClip, x:Number, y:Number, r:Number):Void {
  mc.moveTo(x+r, y);
  mc.curveTo(r+x, Math.tan(Math.PI/8)*r+y, Math.sin(Math.PI/4)*r+x,
Math.sin(Math.PI/4)*r+y);
  mc.curveTo(Math.tan(Math.PI/8)*r+x, r+y, x, r+y);
  mc.curveTo(-Math.tan(Math.PI/8)*r+x, r+y, -Math.sin(Math.PI/4)*r+x,
Math.sin(Math.PI/4)*r+y);
  mc.curveTo(-r+x, Math.tan(Math.PI/8)*r+y, -r+x, y);
  mc.curveTo(-r+x, -Math.tan(Math.PI/8)*r+y, -Math.sin(Math.PI/4)*r+x,
-Math.sin(Math.PI/4)*r+y);
  mc.curveTo(-Math.tan(Math.PI/8)*r+x, -r+y, x, -r+y);
  mc.curveTo(Math.tan(Math.PI/8)*r+x, -r+y, Math.sin(Math.PI/4)*r+x, -
Math.sin(Math.PI/4)*r+y);
  mc.curveTo(r+x, -Math.tan(Math.PI/8)*r+y, r+x, y);
  } // konec funkce drawCircle

function smrt() {
      _root.hlava._x=160;
      _root.hlava._y=100;
      _root.hlava.smer=4;
      for (var m=0;m<_root.hlava.telox.length;m++)
```

v ploše:

Pro šikovné studenty navíc

Jak bychom vyrobili třeba 5 stejných jablek? Celý dnešní kód na tvoření jablka umístíme do cyklu (např. for (j=1; j<=5; j++), jablka budou v hloubkách hloubek 41 až 45, tedy 40+j. Lehce komplikovaný je přístup k jablkům – použijeme funkci `getInstanceAtDepth(40+j)`, jejíž výstup umístíme do nové proměnné jablkoobj; kdekoliv v kódu používáme jablko1, místo něj použijeme jablkoobj.
Precizně zpracováno to bude takto:

```
  // TVORBA 5 JABLEK:
  for (j=1; j<=5; j++){
  this.createEmptyMovieClip("jablko"+j, 40+j);
  jablkoobj=getInstanceAtDepth(40+j);
  jablkoobj.beginFill(0xFF0000);
  drawCircle(jablkoobj, 5, 5, 5);
  jablkoobj.endFill();
```

```
  do {
    jablkoobj._x=10+10*random(30);
    jablkoobj._y=10+10*random(18);
    var obsazeno=0;
    for (m=0; m<zdix.length; m=m+1)
      if ((jablkoobj._x==zdix[m])&&(jablkoobj._y==zdiy[m]))
        {obsazeno=1;break;}
               for (m=0; m<_root.hlava.telox.length; m=m+1)
      if ((jablkoobj._x==_root.hlava.telox[m])&&(jablkoobj._y==_root.hlav
a.teloy[m]))
        {obsazeno=1;break;}
    } while (obsazeno==1);
  } // of for-j-1-5
```

Kód „Pro šikovné studenty navíc“ není součástí hry Had, jde proti její logice, ale chápu jej jako zajímavý podnět. Obvykle jej ukazuji jen na dataprojektoru a vystavím na sdíleném disku, v celé skupině ho netvoříme. Občas se najde student, který pokračuje s více jablky.