JSR 184 - tutorial 3

Přidávání, ubírání a kopírování objektů v JSR-184

Cíl tutorialu: - naučit vás přidávat/ ubírat objekty ze scény a kopírovat je

Tak jdeme na to:

V předešlém tutorialu jsme si popsali základní metody pro změnu polohy, rotace a velikosti objektů. Dnes bych vám chtěl popsat pár metod, pomocí kterých se do scény přidávají nebo ubírají objekty a pomocí kterých můžete objekty kopírovat. Na závěru tohoto tutorialu se opět vrátíme k naší hře „packman“ a ukážeme si, jak lze vytvářet úrovně do takovýchto her za pomoci naučených metod.

Třída Group obsahuje metody na přidávání a odebírání objektů ze skupin. A protože je třída World odvozena od třídy Group, tak všechny tyto metody můžeme aplikovat i na „World“ (což je vlastně také taková větší skupina). Pokud do nějaké skupiny něco přidáváme (odebíráme), musí to být odvozeno od třídy Node. To znamená, že můžeme přidávat (odebírat) objekty, jako jsou „Camera“ (kamera), „Group“ (skupina), „Light“ (světlo), „Mesh“ (model) a „Sprite3D“ (sprit). Jak vidíte, do jedné skupiny můžeme přidat další skupinu, do které můžeme přidat další atd…

Přidávání a ubírání

Přidávání a ubírání objektů je velmi jednoduchá záležitost. Přidání nějakého objektu do nějaké skupiny (světa) se provádí pomocí metody „addChild(Node node)“. Možná vám není jasné, k čemu vám je, když máte nějaký objekt ve skupině. Tak například, kdybyste neměli objekty ve vykreslovaném světě, tak by tam samozřejmě nebyli vidět (nebo by v případě světel neovlivňovali okolí atp.). Takže proč mít objekty ve světě je snad jasné. Ale proč je dávat do obyčejných skupin? Důvodů je hned několik. Například, když chci, aby se spolu pohybovalo více objektů, dám si je do skupiny a pohybuji celou skupinou – hýbou se všechny objekty závisle na skupině (samozřejmě můžete pohybovat i každým zvlášť). Ale nejspíš hlavním důvodem, proč přiřazovat objekty skupinám je ten, že kontrola kolize (které se naučíme v jednom z příštích tutorialů) se vždy provádí na skupině a ne na jednotlivých objektech.

Prakticky stejným způsobem, jako se objekty do skupin přidávají, se z nich mohou i odebírat. To se dělá pomocí metody „removeChild(Node node)“. Poté už objekt do skupiny nepatří, tedy skrze tuto skupinu s ním nelze manipulovat ani provádět kontroly kolizí.

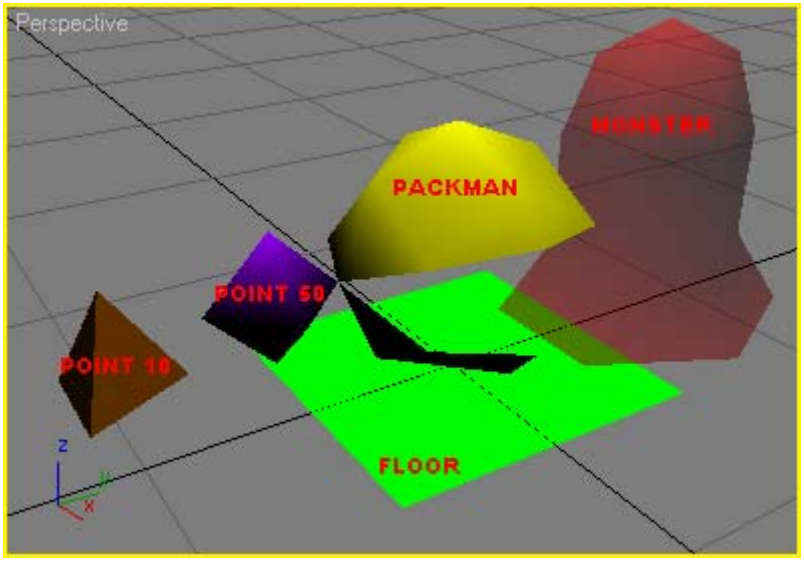
Každá skupina má také možnost zjistit, kolik pod ní spadá objektů. To se dělá pomocí metody „getChildCount()“, která vrací hodnotu typu int, udávající počet objektů spadajících pod tuto skupinu. Ještě bych chtěl poznamenat, že jeden objekt nemůže spadat pod více skupin.

Kopírování

Kopírování se provádí metodou z třídy Object3D a to metodou „duplicate()“. Takto se vytvoří přesná kopie dané věci. Všimněte si, že kopírovat (duplikovat) nemusíme pouze věci odvozené od třídy Node, ale můžeme kopírovat vše, co je odvozené od třídy Object3D. To znamená, že můžeme kopírovat třeba i celý „svět“. Jen nezapomeňte do závorek před tím napsat, co vlastně kopírujete. Například, kopírujeme-li světlo, zapíšeme to takto: „Light light2 = (Light)light1.duplicate()“.

Zpět k našemu packmanovi

Jak jsem již říkal na začátku tohoto tutorialu, dnes si ukážeme, jak lze jednoduše vytvářet úrovně do takových her, jako je náš packman. Budeme to dělat tak, že si vytvoříme pole, které bude představovat náš level a podle hodnot v něm budeme kopírovat na pozice (které si vypočteme) políčka, po kterých se bude packman pohybovat. 0 bude představovat prázdné pole, 1 bude pole, na kterém bude náš packman začínat, 2 bude představovat pole, na kterém budou začínat „monstra“ (a bude na nich 10 bodové „jídlo“), 3 bude tvořit pole, nad kterým bude při načtení úrovně „jídlo“ (nebo jak tomu chcete říkat) s bodovou hodnotou 10 a 4 bude pole s „jídlem“ za 50 bodů umožňující dočasnou průchodnost monster. Takže si do vašeho *.m3g souboru přidejte zmíněné modely (políčko po kterém bude packman chodit, model monstra a modely 10 a 50 bodového jídla). Na obrázku jsou tyto modely vidět vedle sebe a popsané, abyste věděli, co bude jaký model představovat:



Zjistěte si ID těchto modelů a zapište je takto (jen nejspíš s jinými hodnotami):

```
int floorID = 45;
int monsterID = 36;
int point10ID = 11;
int point50ID = 19;
```

Teď si vytvoříme novou třídu s názvem „levely“, do které budeme ukládat námi vytvořené úrovně. Ještě než vytvoříme samotnou první úroveň, řekneme si, kolik bude mít políček. Pro ulehčení budeme dělat čtvercové úrovně řekněme o velikosti 15x15 polí. Toto číslo si zapíšeme jako proměnnou typu int s názvem „rad“ (jako řad).

```
static int rad = 15;
```

A teď si uděláme level podle toho, jak jsme si to napsali výše. Bude to pole typu int s 15x15 hodnotami (s 225 hodnotami). Nazveme si ho „level1“:

```
static int[] level1 = new int[]
{
	2, 3, 3, 3, 3, 3, 3, 4, 3, 3, 3, 3, 3, 3, 2,
	3, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 3,
	3, 0, 4, 3, 3, 3, 0, 3, 0, 3, 3, 3, 4, 0, 3,
	3, 0, 0, 0, 0, 3, 0, 3, 0, 3, 0, 0, 0, 0, 3,
	3, 3, 3, 3, 0, 3, 0, 3, 0, 3, 0, 3, 3, 3, 3,
	3, 0, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 0, 3,
	3, 3, 3, 3, 0, 3, 0, 3, 0, 3, 0, 3, 3, 3, 3,
	3, 0, 0, 3, 0, 3, 3, 1, 3, 3, 0, 3, 0, 0, 3,
	3, 3, 3, 3, 0, 0, 0, 0, 0, 0, 0, 3, 3, 3, 3,
	0, 3, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0, 3, 0,
	0, 3, 0, 3, 0, 0, 0, 3, 0, 0, 0, 3, 0, 3, 0,
	0, 3, 0, 3, 0, 3, 3, 3, 3, 3, 0, 3, 0, 3, 0,
	0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0,
	0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0,
	0, 4, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 0
};
```

Celá tato třída tedy bude vypadat jednoduše takto:

```
public class levely
{
	static int rad = 15;

	static int[] level1 = new int[]
	{
		2, 3, 3, 3, 3, 3, 3, 4, 3, 3, 3, 3, 3, 3, 2,
		3, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 3,
		3, 0, 4, 3, 3, 3, 0, 3, 0, 3, 3, 3, 4, 0, 3,
		3, 0, 0, 0, 0, 3, 0, 3, 0, 3, 0, 0, 0, 0, 3,
		3, 3, 3, 3, 0, 3, 0, 3, 0, 3, 0, 3, 3, 3, 3,
		3, 0, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 0, 3,
		3, 3, 3, 3, 0, 3, 0, 3, 0, 3, 0, 3, 3, 3, 3,
		3, 0, 0, 3, 0, 3, 3, 1, 3, 3, 0, 3, 0, 0, 3,
		3, 3, 3, 3, 0, 0, 0, 0, 0, 0, 0, 3, 3, 3, 3,
		0, 3, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0, 3, 0,
		0, 3, 0, 3, 0, 0, 0, 3, 0, 0, 0, 3, 0, 3, 0,
		0, 3, 0, 3, 0, 3, 3, 3, 3, 3, 0, 3, 0, 3, 0,
		0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0,
		0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0,
		0, 4, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 0
	};
}
```

Nyní se vrátíme k třídě s packmanem. Deklarujeme Meshe vytvořených modelů, každý s počáteční nulovou hodnotou:

```
Mesh floorMesh = null;
Mesh monsterMesh = null;
Mesh point10Mesh = null;
Mesh point50Mesh = null;
```

Dále budeme potřebovat pole těchto Meshů, kam budeme ukládat zkopírované Meshe našich modelů. Hodnoty zde nastavené jsou nejvyšším možným počtem každého objektu (i když třeba monster asi 225 mít nebudete):

```
Mesh[] floor = new Mesh[levely.level1.length];
Mesh[] monster = new Mesh[levely.level1.length];
Mesh[] point10 = new Mesh[levely.level1.length];
Mesh[] point50 = new Mesh[levely.level1.length];
```

A ještě si (trochu do budoucna) připravíme skupiny, do kterých budeme vkládat také tyto Meshe, hlavně kvůli kolizím:

```
Group floorGroup = new Group();
Group monsterGroup = new Group();
Group point10Group = new Group();
Group point50Group = new Group();
```

Do funkce na načtení světa („loadWorld()“) si za kód pro načtení vložíme kód, který přidá tyto skupiny do světa:

```
world.addChild(floorGroup);
world.addChild(monsterGroup);
world.addChild(point10Group);
world.addChild(point50Group);
```

A samozřejmě si někde musíme ve světě vyhledat objekty, které budeme kopírovat. Od toho máme naši funkci „loadObjects()“:

```
public void loadObjects()
{
      cam = world.getActiveCamera();
      packman = (Mesh)world.find(packmanID);
      floorMesh = (Mesh)world.find(floorID);
      monsterMesh = (Mesh)world.find(monsterID);
      point10Mesh = (Mesh)world.find(point10ID);
      point50Mesh = (Mesh)world.find(point50ID);
      world.removeChild(floorMesh);
      world.removeChild(monsterMesh);
      world.removeChild(point10Mesh);
      world.removeChild(point50Mesh);
}
```

Pro správné umístění objektů ještě potřebujeme vědět, jak velká bude naše hrací plocha. A protože máme „hřiště“ 15x15 polí a 1 pole má rozměry 10x10, tak velikost hrací plochy bude 150:

```
int poleVelikost = 150;
```

Ještě budeme do budoucna potřebovat vědět, kolik je v úrovni monster (aby nemusel kód probíhat v cyklu 225 krát když bude stačit jenom 2x). Proto si vytvoříme proměnnou (int) s názvem „pocetMonster“ a počáteční hodnotou 0. Jak bude kód procházet naší úrovní a narazí na pole s monstry tak se hodnota této proměnné o 1 zvýší.

```
Int pocetMonster = 0;
```

Z konstruktoru zavoláme metodu (kterou si nyní vytvoříme) pro načtení levelu s parametrem pole typu int (námi vytvořené úrovně ze třídy „levely“):

```
loadRoom(levely.level1);
```

A nyní ta slibovaná metoda na „postavení“ našeho levelu.. Nejtěžší pro vás zřejmě bude se zorientovat v metodě „setTranslation“, protože jsem tam zahrnul trošku složitější výpočty pro správné umístění daného objektu na správné souřadnice (ale když se jim budete chvíli věnovat tak se v nich snat zorientujete):

```
public void loadRoom(int[] level)
{
	for(int i = 0; i < level.length; i++)
	{
		if(level[i] == 1)
		{
			floor[i] = (Mesh)floorMesh.duplicate();
			floor[i].setTranslation(((-
10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-10*(int)(i-
(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
			floorGroup.addChild(floor[i]);
			packman.setTranslation(((-
10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-10*(int)(i-
(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
		}
		else if(level[i] == 2)
		{
			floor[i] = (Mesh)floorMesh.duplicate();
			floor[i].setTranslation(((-
10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-10*(int)(i-
(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
			floorGroup.addChild(floor[i]);
			monster[pocetMonster] = (Mesh)monsterMesh.duplicate();
			monster[pocetMonster].setTranslation(((-
10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-10*(int)(i-
(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
			monsterGroup.addChild(monster[pocetMonster]);
			pocetMonster++;
			point10[i] = (Mesh)point10Mesh.duplicate();
			point10[i].setTranslation(((-
10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-10*(int)(i-
(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
			point10Group.addChild(point10[i]);
		}
		else if(level[i] == 3)
		{
			floor[i] = (Mesh)floorMesh.duplicate();
			floor[i].setTranslation(((-
10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-10*(int)(i-
(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
			floorGroup.addChild(floor[i]);
			point10[i] = (Mesh)point10Mesh.duplicate();
			point10[i].setTranslation(((-
10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-10*(int)(i-
(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
			point10Group.addChild(point10[i]);
		}
		else if(level[i] == 4)
		{
```

```
                floor[i] = (Mesh)floorMesh.duplicate();
                floor[i].setTranslation(((-
10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-10*(int)(i-
(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
                floorGroup.addChild(floor[i]);
                point50[i] = (Mesh)point50Mesh.duplicate();
                point50[i].setTranslation(((-
10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-10*(int)(i-
(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
                point50Group.addChild(point50[i]);
            }
        }
}
```

Nyní se už náš packman docela obstojně pohybuje po úrovni (nepočítaje to, že se může ohybovat i mimo cestu), kde jsou rozmístěná jak pole, tak body a monstra, ale kamera je stále na jednom místě, což je pro takovouto hru vcelku nepraktické. Proto si ještě před koncem dnešního tutorialu ukážeme, jak udělat pohled (a pohyb) kamery „z třetí osoby“, u kterého využijeme to, co jsme se dnes naučili. Nejprve si vytvoříme 2 pomocné skupiny (camG a camGG), které nám usnadní manipulaci s kamerou:

```
Group camGG = new Group();
Group camG = new Group();
```

Pak budeme potřebovat proměnnou (float) s aktuální rotací kamery (zpočatku 0):

```
Float camRotZ = 0.0f;
```

Na konci metody „loadWorld“ si do světa přidáme skupinu „camGG“:

```
world.addChild(camGG);
```

A ve funkci „loadObjects“ si hned za načtením kamery tuto kameru odebereme ze světa, protože ji budeme dávat do jiné skupiny (pro snažší manipulaci) a jak jsme si již říkali, 1 objekt nemůže být ve více skupinách. Kameru dáme pod skupinu „camG“ a tuto skupinu dáme pod „camGG“:

```
world.removeChild(cam);
camGG.addChild(camG);
camG.addChild(cam);
```

Při grafickém znázornění by to vypadalo asi takto:

```
World world
    '-----> Group camGG
                '-----> Group camG
                            '-----> Camera cam
```

Takže budeme-li pohybovat (rotovat) skupinou „camGG“, přemístění (rotace) se aplikuje i na „camG“ i „cam“. Pro pohyb kamery budeme potřebovat znát souřadnice packmana, takže si vytvoříme nové pole typu float s názvem „packmanPos“ o 3 hodnotách:

```
float[] packmanPos = new float[3];
```

Polohu našeho packmana zjistíme (jak jsme si říkali v 2. Tutorialu) metodou „getTranslation(float[] xyz)“, tak si do metody „pohybPackmana“ tuto metodu přidáme:

```
packman.getTranslation(packmanPos);
```

Nyní si do metody „run()“ přidáme volání metody „pohybKamery()“, která bude vypadat docela jednoduše:

```
public void pohybKamery()
{
	camGG.setTranslation(packmanPos[0], packmanPos[1], packmanPos[2]);
	camRotZ -= camRotZ/6;
	camG.setOrientation(camRotZ, 0, 0, 1);
}
```

Skupina „camGG“ se v této metodě stará o to, aby se pohybovala společně s packmanem (tím pádem se tak pohybuje i skupina „camG“ i naše kamera) a skupina „camG“ se natáčí podle proměnné „camRotZ“, která se plynule přibližuje nule (tedy vyrovnává se za packmana). Ovšem aby to bylo takto možné, musíme si při změně směru packmana nastavit rotaci „camGG“ stejnou jako rotaci packmana a s proměnnou „camRotZ“ tedy musíme udělat opačnou operaci (aby byla natočena jako předtím pouze aby bylo možné hodnotu opět vyrovnávat na 0). Takto bude vypadat upravená metoda pro kontrolu stisku kláves:

```
protected void keyPressed(int keyCode)
{
	if(keyCode == KEY_NUM4 || keyCode == getKeyCode(Canvas.LEFT))
	{
		if(smerPackmana <= 4)
		{
			smerPackmana = smerPackmana>1 ? smerPackmana-1 : 4;
		}
		else
		{
			smerPackmana = smerPackmana==10 ? 4 : (smerPackmana-
10)/10 ;
		}

		packman.preRotate(90, 0, 0, 1);
		camGG.preRotate(90, 0, 0, 1);
		camRotZ -= 90;
		camG.setOrientation(camRotZ, 0, 0, 1);
	}
	else if(keyCode == KEY_NUM6 || keyCode == getKeyCode(Canvas.RIGHT))
	{
		if(smerPackmana <= 4)
		{
			smerPackmana = smerPackmana<4 ? smerPackmana+1 : 1;
		}
		else
		{
```

```
                smerPackmana = smerPackmana==40 ? 1 :
(smerPackmana+10)/10 ;
            }

            packman.preRotate(-90, 0, 0, 1);
            camGG.preRotate(-90, 0, 0, 1);
            camRotZ += 90;
            camG.setOrientation(camRotZ, 0, 0, 1);
      }
      else if(keyCode == KEY_NUM8 || keyCode == getKeyCode(Canvas.DOWN))
      {
            if(smerPackmana <= 4)
            {
                  smerPackmana *= 10;
            }
            else
            {
                  switch(smerPackmana)
                  {
                        case 10:
                              smerPackmana = 3;
                        break;

                        case 20:
                              smerPackmana = 4;
                        break;

                        case 30:
                              smerPackmana = 1;
                        break;

                        case 40:
                              smerPackmana = 2;
                        break;
                  }
                  if(camRotZ >= 0)
                  {
                        packman.preRotate(180, 0, 0, 1);
                        camGG.preRotate(180, 0, 0, 1);
                        camRotZ -= 180;
                        camG.setOrientation(camRotZ, 0, 0, 1);
                  }
                  else
                  {
                        packman.preRotate(-180, 0, 0, 1);
                        camGG.preRotate(-180, 0, 0, 1);
                        camRotZ += 180;
                        camG.setOrientation(camRotZ, 0, 0, 1);
                  }
            }
      }
      else if(keyCode == KEY_NUM2 || keyCode == getKeyCode(Canvas.UP))
      {
            smerPackmana = smerPackmana>4 ? smerPackmana/10 : smerPackmana;
      }
}
```

To je z dnešního tutorialu už všechno. Celý kód třídy „packman3Dcanvas“ by měl po dnešním tutorialu vypadat asi takto:

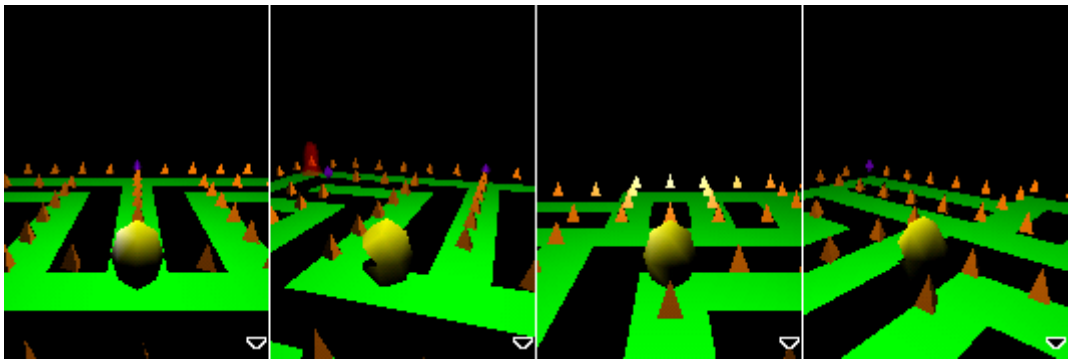
```
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Graphics;
import javax.microedition.m3g.World;
import javax.microedition.m3g.Camera;
import javax.microedition.m3g.Graphics3D;
import javax.microedition.m3g.Loader;
import javax.microedition.m3g.Mesh;
import javax.microedition.m3g.Object3D;
import javax.microedition.m3g.Group;

public class packman3DCanvas extends Canvas implements Runnable
{
        Thread myThread = null;
    boolean running = false;

        Graphics3D g3d = null;

        World world = null;

        Camera cam = null;
        Group camGG = new Group();
        Group camG = new Group();
        float camRotZ = 0.0f;

        Mesh packman = null;
        Mesh floorMesh = null;
        Mesh monsterMesh = null;
        Mesh point10Mesh = null;
        Mesh point50Mesh = null;
        Mesh[] floor = new Mesh[levely.level1.length];
        Mesh[] monster = new Mesh[levely.level1.length];
        Mesh[] point10 = new Mesh[levely.level1.length];
        Mesh[] point50 = new Mesh[levely.level1.length];
        Group floorGroup = new Group();
        Group monsterGroup = new Group();
        Group point10Group = new Group();
        Group point50Group = new Group();

        int pocetMonster = 0;

        int packmanID = 27;
        int floorID = 45;
        int monsterID = 36;
        int point10ID = 11;
        int point50ID = 19;

        int smerPackmana = 30;   // 1 = x-, 2 = y+, 3 = x+, 4 = y-, >4 = stoji
        float rychlostPackmana = 2.5f;
        int[][] smeryPohybu = new int[][] {{-1, 0, 0}, {0, 1, 0}, {1, 0, 0}, {0, -1, 0}};
        float[] packmanPos = new float[3];


        int poleVelikost = 150;

        public packman3DCanvas()
        {
                setFullScreenMode(true);
                loadWorld("/packman.m3g");
                loadObjects();
                loadRoom(levely.level1);

                myThread = new Thread(this);
       running = true;
                myThread.start();
        }

        public void run()
        {
       while(running) {
                        pohybPackmana();
                        pohybKamery();
                        try {
                                Thread.sleep(40);
                        } catch(Exception e) {
                                System.out.println(e);
                        }
                }
        }

        public void loadWorld(String path)
        {
                try
                {
```

```
                        Object3D[] buffer = Loader.load(path);
                        for(int i = 0; i < buffer.length; i++)
                        {
                                if(buffer[i] instanceof World)
                                {
                                        world = (World)buffer[i];
                                        break;
                                }
                        }
                }
                catch (Exception e)
                {
                        System.out.println(e);
                }

                world.addChild(floorGroup);
                world.addChild(monsterGroup);
                world.addChild(point10Group);
                world.addChild(point50Group);
                world.addChild(camGG);
        }

        public void loadObjects()
        {
                cam = world.getActiveCamera();
                world.removeChild(cam);
                camGG.addChild(camG);
                camG.addChild(cam);
                packman = (Mesh)world.find(packmanID);
                floorMesh = (Mesh)world.find(floorID);
                monsterMesh = (Mesh)world.find(monsterID);
                point10Mesh = (Mesh)world.find(point10ID);
                point50Mesh = (Mesh)world.find(point50ID);
                world.removeChild(floorMesh);
                world.removeChild(monsterMesh);
                world.removeChild(point10Mesh);
                world.removeChild(point50Mesh);
        }

        public void loadRoom(int[] level)
        {
                for(int i = 0; i < level.length; i++)
                {
                        if(level[i] == 1)
                        {
                                floor[i] = (Mesh)floorMesh.duplicate();
                                floor[i].setTranslation(((-10*(int)(i/levely.rad))+(poleVelikost/2))-5,
((-10*(int)(i-(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
                                floorGroup.addChild(floor[i]);
                                packman.setTranslation(((-10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-
10*(int)(i-(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
                        }
                        else if(level[i] == 2)
                        {
                                floor[i] = (Mesh)floorMesh.duplicate();
                                floor[i].setTranslation(((-10*(int)(i/levely.rad))+(poleVelikost/2))-5,
((-10*(int)(i-(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
                                floorGroup.addChild(floor[i]);
                                monster[pocetMonster] = (Mesh)monsterMesh.duplicate();
                                monster[pocetMonster].setTranslation(((-
10*(int)(i/levely.rad))+(poleVelikost/2))-5, ((-10*(int)(i-(i/levely.rad)*levely.rad))+(poleVelikost/2))-5,
0);
                                monsterGroup.addChild(monster[pocetMonster]);
                                pocetMonster++;
                                point10[i] = (Mesh)point10Mesh.duplicate();
                                point10[i].setTranslation(((-10*(int)(i/levely.rad))+(poleVelikost/2))-5,
((-10*(int)(i-(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
                                point10Group.addChild(point10[i]);
                        }
                        else if(level[i] == 3)
                        {
                                floor[i] = (Mesh)floorMesh.duplicate();
                                floor[i].setTranslation(((-10*(int)(i/levely.rad))+(poleVelikost/2))-5,
((-10*(int)(i-(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
                                floorGroup.addChild(floor[i]);
                                point10[i] = (Mesh)point10Mesh.duplicate();
                                point10[i].setTranslation(((-10*(int)(i/levely.rad))+(poleVelikost/2))-5,
((-10*(int)(i-(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
                                point10Group.addChild(point10[i]);
                        }
                        else if(level[i] == 4)
                        {
                                floor[i] = (Mesh)floorMesh.duplicate();
                                floor[i].setTranslation(((-10*(int)(i/levely.rad))+(poleVelikost/2))-5,
((-10*(int)(i-(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
                                floorGroup.addChild(floor[i]);
                                point50[i] = (Mesh)point50Mesh.duplicate();
```

```
                                point50[i].setTranslation(((-10*(int)(i/levely.rad))+(poleVelikost/2))-5,
((-10*(int)(i-(i/levely.rad)*levely.rad))+(poleVelikost/2))-5, 0);
                                point50Group.addChild(point50[i]);
                        }
                }
        }

        public void pohybPackmana()
        {
                packman.getTranslation(packmanPos);
                if(smerPackmana >= 1 && smerPackmana <= 4 )
                {
                        packman.translate(smeryPohybu[smerPackmana-1][0]*rychlostPackmana,
smeryPohybu[smerPackmana-1][1]*rychlostPackmana, smeryPohybu[smerPackmana-1][2]*rychlostPackmana);
                }
        }

        public void pohybKamery()
        {
                camGG.setTranslation(packmanPos[0], packmanPos[1], packmanPos[2]);
                camRotZ -= camRotZ/6;
                camG.setOrientation(camRotZ, 0, 0, 1);
        }

        public void paint(Graphics g)
        {
                g3d = Graphics3D.getInstance();
                try {
                        g3d.bindTarget(g, true, 0);
                        g3d.render(world);
                } catch(Exception e) {
                        System.out.println(e);
                } finally {
                        g3d.releaseTarget();
                }
                repaint();
        }

        protected void keyPressed(int keyCode)
        {
                if(keyCode == KEY_NUM4 || keyCode == getKeyCode(Canvas.LEFT))
                {
                        if(smerPackmana <= 4)
                        {
                                smerPackmana = smerPackmana>1 ? smerPackmana-1 : 4;
                        }
                        else
                        {
                                smerPackmana = smerPackmana==10 ? 4 : (smerPackmana-10)/10 ;
                        }

                        packman.preRotate(90, 0, 0, 1);
                        camGG.preRotate(90, 0, 0, 1);
                        camRotZ -= 90;
                        camG.setOrientation(camRotZ, 0, 0, 1);
                }
                else if(keyCode == KEY_NUM6 || keyCode == getKeyCode(Canvas.RIGHT))
                {
                        if(smerPackmana <= 4)
                        {
                                smerPackmana = smerPackmana<4 ? smerPackmana+1 : 1;
                        }
                        else
                        {
                                smerPackmana = smerPackmana==40 ? 1 : (smerPackmana+10)/10 ;
                        }

                        packman.preRotate(-90, 0, 0, 1);
                        camGG.preRotate(-90, 0, 0, 1);
                        camRotZ += 90;
                        camG.setOrientation(camRotZ, 0, 0, 1);
                }
                else if(keyCode == KEY_NUM8 || keyCode == getKeyCode(Canvas.DOWN))
                {
                        if(smerPackmana <= 4)
                        {
                                smerPackmana *= 10;
                        }
                        else
                        {
                                switch(smerPackmana)
                                {
                                        case 10:
                                                smerPackmana = 3;
                                        break;

                                        case 20:
                                                smerPackmana = 4;
```

```
                                        break;

                                        case 30:
                                                smerPackmana = 1;
                                        break;

                                        case 40:
                                                smerPackmana = 2;
                                        break;
                                }
                                if(camRotZ >= 0)
                                {
                                        packman.preRotate(180, 0, 0, 1);
                                        camGG.preRotate(180, 0, 0, 1);
                                        camRotZ -= 180;
                                        camG.setOrientation(camRotZ, 0, 0, 1);
                                }
                                else
                                {
                                        packman.preRotate(-180, 0, 0, 1);
                                        camGG.preRotate(-180, 0, 0, 1);
                                        camRotZ += 180;
                                        camG.setOrientation(camRotZ, 0, 0, 1);
                                }
                        }
                }
                else if(keyCode == KEY_NUM2 || keyCode == getKeyCode(Canvas.UP))
                {
                        smerPackmana = smerPackmana>4 ? smerPackmana/10 : smerPackmana;
                }
        }
}
```

Když nyní hru spustíte, nebude již náš svět tak prázdný jako předtím a packman se po něm bude schopen pohybovat následován kamerou snímající jej z pohledu třetí osoby. V našem případě to bude vypadat asi takto:



Pokud budete mít nějaké potíže, naleznete v tutorialu chybu, budete mít nějaký nápad na vylepšení, budete se chtít na něco zeptat, budete se chtít pochlubit svou prací, budete se chtít k tutorialu vyjádři (ať už kladně či záporně) nebo cokoli podobného, budu velmi rád, když napíšete do fóra, popřípadě přímo mně na mail nebo na ICQ.

Kompletní zdrojový kód tohoto tutorialu se všemi soubory naleznete ke stažení v sekci Download.