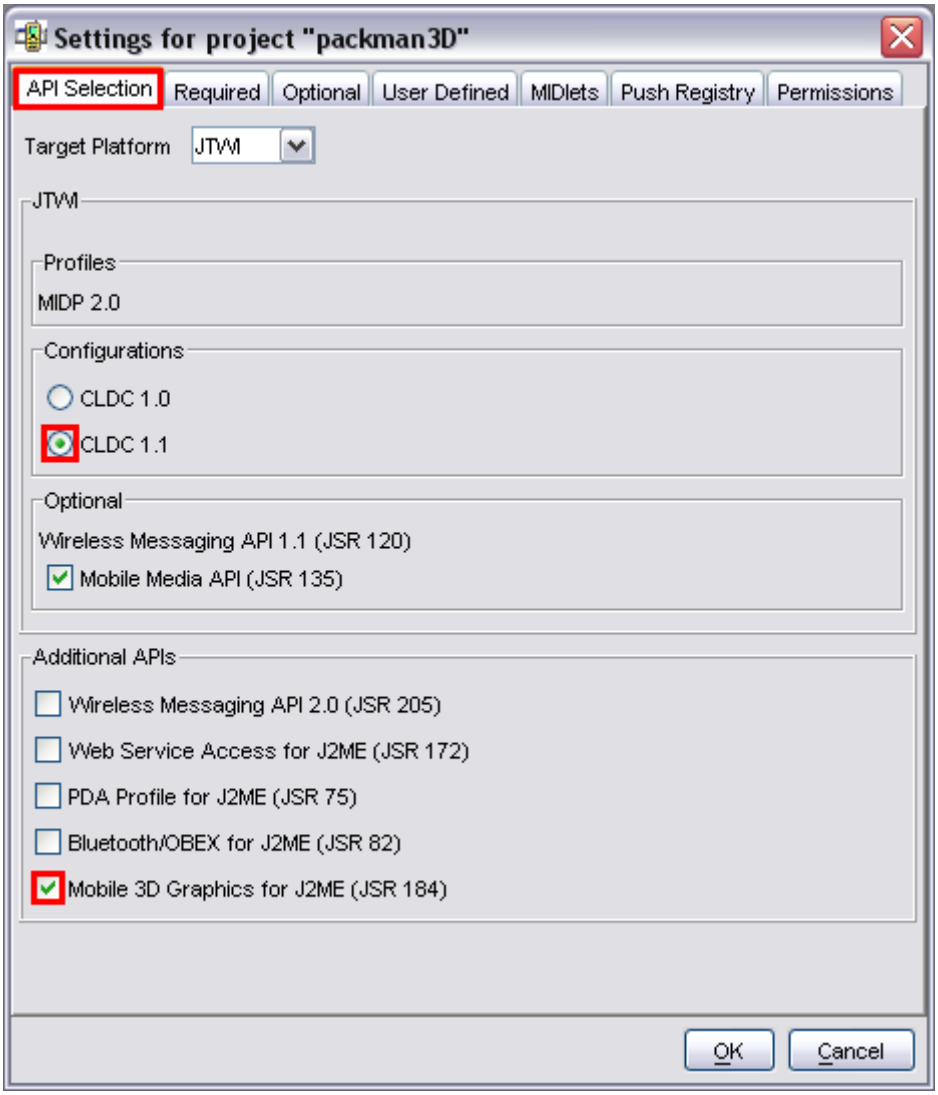
JSR 184 - tutorial 1

Úvod do JSR-184

Cíl tutorialu: - naučit vás načíst *.m3g soubor, vybrat z něho 1 objekt a vykreslit scénu

Tak jdeme na to:

Nejdříve si samozřejmě vytvoříme nový projekt. Já ho nazval „packman3D“ (protože během této série tutorialů bych vám chtěl krok za krokem pomoci naprogramovat hru Packman v 3D a učit vás během toho práci s JSR-184). Navím, jak kdo z vás, ale já pracuji ve WTK. Takže kdo v něm také pracujete, nezapomeňte v nastavení zaškrtnout položku „CLDC 1.1“ a „Mobile 3D Graphics for J2ME (JSR 184)“ (nevím jak to chodí u jiných IDE a spol. tak si musíte zjistit sami zda se někde něco má zaškrtnout…):



Nejdříve si napíšeme hlavní třídu. Ta je velmi jednoduchá:

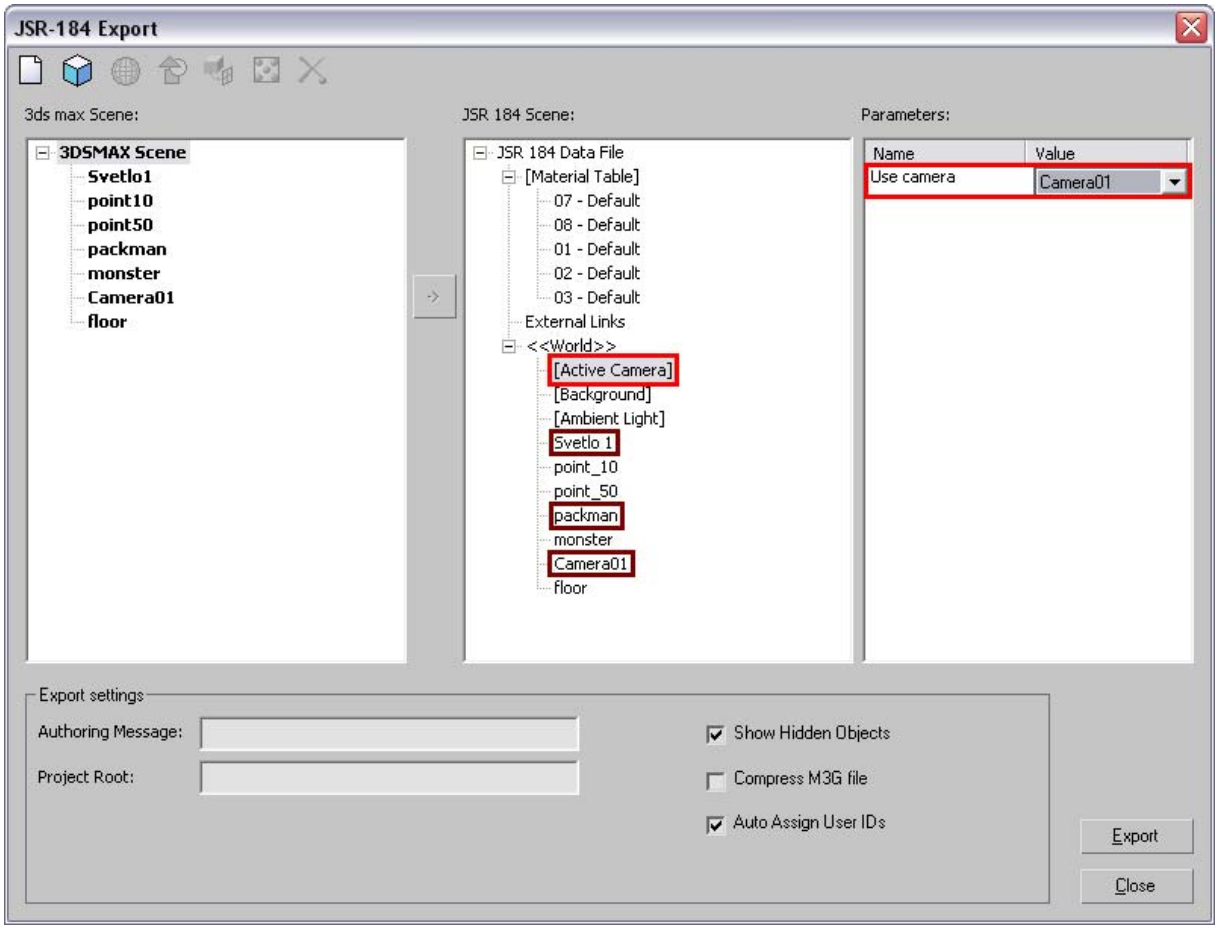
```
import javax.microedition.midlet.MIDlet;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Command;

public class packman3D extends MIDlet implements CommandListener
{
        Display display = null;
        Packman3DCanvas packmanCanvas = new packman3DCanvas();

        Command exit = new Command("Exit", Command.EXIT, 0);

        public void startApp()
        {
                display = Display.getDisplay(this);
                display.setCurrent(packmanCanvas);
                packmanCanvas.addCommand(exit);
                packmanCanvas.setCommandListener(this);
        }

        public void pauseApp()
        {
        }

        public void destroyApp(boolean b)
        {
        }

        public void commandAction(Command c, Displayable d)
        {
                if(c == exit)
                {
                        notifyDestroyed();
                }
        }
}
```

Tuto třídu snad nemusím nijak složitě popisovat, protože předpokládám, že když se chcete pouštět do programování v J2ME 3D, tak zvládáte snad vše obsažené v této třídě. V této třídě jsme prakticky jen získali přístup k displayi a předali jsme jeho ovládání třídě „packman3DCanvas“.

V druhé (a pro dnešek také poslední) třídě se konečně dostáváme k problematice 3D javy. Tuto třídu jsme si nazvali „packman3Dcanvas“ a jak již z názvu vyplývá, tato třída rozšiřuje Canvas (tzv. „plátno“ na které budeme vykreslovat). Mimo knihovny „javax.microedition.lcdui.*;“ (kterou budeme samozřejmě potřebovat pro vykreslování) budeme ještě potřebovat knihovnu „javax.microedition.m3g.*;“ která se v J2ME stará o 3D. Z knihovny „lcdui“ konkrétně budeme potřebovat jen „Canvas“ a „Graphics“, z knihovny „m3g“ budeme dnes potřebovat „World“, „Camera“, „Graphics3D“, „Loader“, „Mesh“ a „Object3D“. Proč je budeme potřebovat atd. si povíme níže. Nyní bude začátek naší třídy vypadat 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;
```

Nejdříve si však musíme vytvořit nějaký *.m3g soubor. V čem si jej vytvoříte nechám zcela na vás. Já osobně používám 3D studio max r.8 se zabudovaným exportérem do *.m3g (doporučuji). Můžete však použít i jiné 3D editory a k nim vytvářené pluginy. Hlavní co potřebujeme je, aby se v naší scéně nacházel alespoň 1 objekt, dále tam musí být alespoň 1 světlo (jinak by naše scéna byla celá černá a nic bychom neviděli) a nakonec ještě 1 kamera (v mé scéně toho je o něco víc ale toho si nemusíte všímat, rovnou jsem si vytvořil i objekty, které budeme používat v příštích tutorialech). Pro naše účely nezapomeňte během exportu nastavit tuto kameru jako aktivní:



Samozřejmě toto všechno se dá vytvářet i přes kód, ale to není předmětem dnešního tutorialu. Tuto jednoduchou scénu si tedy vyexportujte do souboru s názvem „packman.m3g“ a uložte ho do složky „res“ u naší aplikace (jiné IDE a spol. mají možná jiný název složky pro zdrojové soubory než je „res“ ve WTK tak tam to zkrátka dejte).

Základy m3g

Ještě než se dostaneme přímo k našemu 3D programování, musíme si deklarovat pár věcí a rovnou si zběžně vysvětlíme, co jaká vlastně znamená. Zaprvé si musíme říct, co se bude starat o vykreslování naší scény. Jako v 2D javě na to máme třídu „Graphics“, v 3D javě to je „Graphics3D“. To si pojmenujeme „g3d“ a ze začátku bude hodnota tohoto nulová. Tedy:

```
Graphics3D g3d = null;
```

Potom samozřejmě budeme potřebovat něco, do čeho se načte námi vytvořená scéna. V JSR-184 se tomu říká „svět“, tedy „World“. Ten si pojmenujeme jednoduše „world“ a jeho hodnota ze začátku bude také nulová, tedy:

```
World world = null;
```

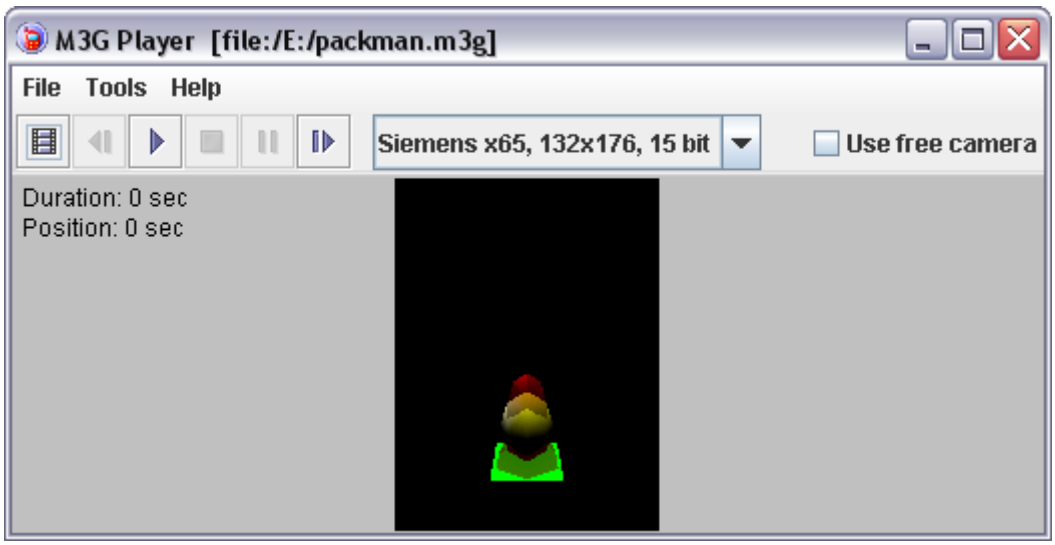
Ještě časem budeme potřebovat manipulovat s kamerou ve scéně (i když to nebude předmětem dnešního tutorialu, mysleme trochu dopředu). Kamera v JSR-184 se nazývá (docela logicky) „Camera“ tak naši kameru si pojmenujeme „cam“, zpočátku s nulovou hodnotou:

```
Camera cam = null;
```

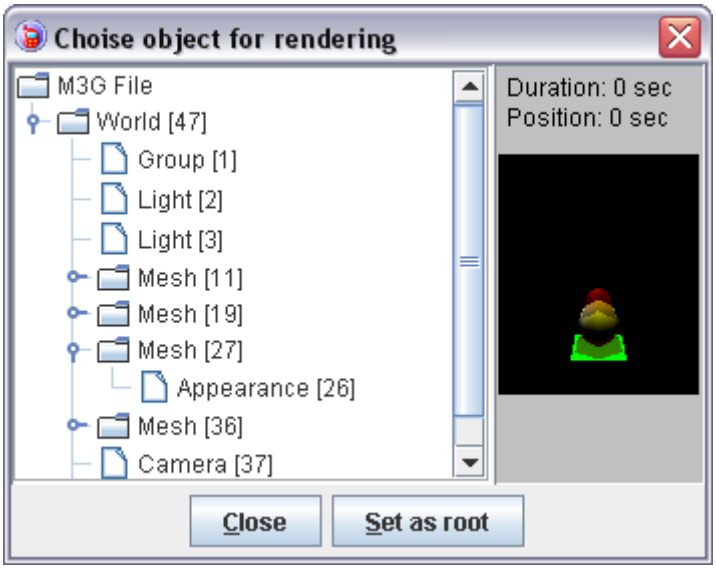
Nakonec pro náš první tutorial budeme ještě potřebovat získat z našeho světa ten objekt, který jsme si předtím vytvořili v nějakém 3D editoru. Takovému objektu se v JSR-184 říká „Mesh“. A protože jsem si říkal, že bychom během naší série tutorialů mohli udělat jednoduchou 3D hru packman, tak jsem si tento mesh nazval „packman“ a ze začátku bude mít také nulovou hodnotu:

```
Mesh packman = null;
```

Ve „světech“ v 3D J2ME se objekty hledají pomocí tzv. „ID“, což je takové identifikační číslo každého objektu, kamery, světla a všeho podobného v *.m3g souboru. ID každé součásti m3g souboru se dá zjistit u některých exportérů již během exportu nebo jej lze zjistit v programech na prohlížení *.m3g souborů, jako je například v 3D studiu max r.8 zabudovaný program M3G player:



Otevřete-li v něm váš m3g soubor, můžete se podívat, jakému objektu je přiřazeno jaké ID:



3ds také během exportu do *.m3g generuje soubor *.html ve kterém jsou v tabulce napsána všechna ID všech součástí scény. Nějakým z těchto způsobů si zjistěte, jaké ID má námi hledaný objekt a jeho hodnotu zapište do kódu jako „int“ se jménem „packmanID“. U mě má hodnotu 27 tak to zapíši takto:

```
int packmanID = 27;
```

Nyní konečně přichází na řadu JSR-184 v praxi. Abychom mohli s naší scénou v kódu nějak manipulovat a vykreslovat ji, musíme si ji nejdříve do našeho kódu nějak načíst. Já osobně na to používám metodu „loadWorld(String path)“ kterou jsem někde opsal a zčásti si ji poupravil a ta se mi zdá asi nejvhodnější:

```
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);
	}
}
```

Samozřejmě tuto metodu si odněkud musíme zavolat, a to nejlépe z konstruktoru této třídy, který bude vypadat asi takto:

```
public packman3DCanvas()
{
      setFullScreenMode(true);
      loadWorld("/packman.m3g");
      loadObjects();
}
```

Nejdříve jsme nastavili, aby naše budoucí hra byla přes celý display, poté jsme zavolali výše zmíněnou metodu na načtení vytvořené scény s parametrem typu String, který udává cestu k uložené scéně a nakonec jsme zavolali metodu „loadObjects()“, kterou si nyní popíšeme. Jak jsem již říkal, v JSR-184 se vybírají objekty pomocí tzv. ID. Toto budeme uplatňovat v této metodě. Nejdříve získáme naši kameru pomocí metody „getActiveCamera()“, která vrací právě aktivní kameru v našem světě a potom přijde na řadu nalezení našeho objektu (packmana). Pokud chceme ve „světě“ vyhledat něco podle ID (což je většinou), uplatňujeme na náš svět metodu „find(int userID)“(do závorek před to zapisujeme, co chceme hledat; v našem případě Mesh). Stejně tak by se dala najít i naše kamera, jen by se to zapisovato „cam = (Camera)world.find(ID kamery)“ a nebyla-li by ve scéně nastavena jako aktivní, mohli bychom to udělat pomocí metody „setActiveCamera(Camera cam)“ (to jen tak pro zajímavost). Naše metoda tedy teď bude vypadat asi takto:

```
public void loadObjects()
{
      cam = world.getActiveCamera();
      packman = (Mesh)world.find(packmanID);
}
```

Rendering

Teď si ukážeme, jak naši scénu vykreslíme. To většinou provádíme v metodě pro vykreslování, kterou má Canvas jako povinnou, tedy „paint(Graphics g)“:

```
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();
}
```

Nejdříve musíme získat instanci třídy Graphics3D - „Graphics3D.getInstance()“, potom nastavíme cíl na co a jak budeme vykreslovat - „bindTarget(Object target, boolean depthBuffer, int hints)“, vykreslíme námi vytvořený svět – „render(World world)“ a nakonec cíl „uvolníme“ – „releaseTarget()“. Pak už jen budeme potřebovat, aby se display překresloval – „repaint()“.

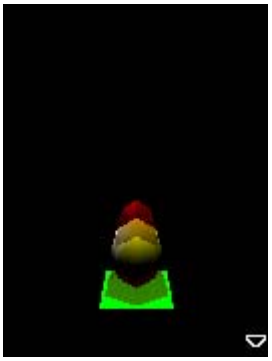
Rád bych se zastavil u příkazu „bindTarget(…..)“ a jeho parametrů. Jeho prvním parametrem je, na co chceme scénu vykreslit. Jelikož vykreslujeme přímo na Canvas, nastavíme sem Graphics, které ovládá Canvas. Mohli bychom sem ale také nastavit obrázek na který vykreslíme scénu a s ním pak můžeme dělat efekty, jako třeba ho nastavit místo textury jinému objektu a mít tak vlastně více světů v sobě. Druhým parametrem je, jestli chceme povolit „hloubkový buffer“. Ještě jsem se nesetkal s případem, kde by u něho bylo nastaveno „false“ (jenom u jednoho tutorialu). Tvoří to pak efekt „nerozměrovosti“. Nakonec, můžete si to vyzkoušet sami. Posledním parametrem je kvalita vykreslované scény. Já ji nastavil na 0, protože se zatím nevyrábí moc telefonů, co by s přijatelnou rychlostí zvládali vykreslovat na plné detaily nebo spíše to nedokáží vůbec. Parametry mohou být: ANTIALIAS, TRUE_COLOR, DITHER, OVERWRITE (nemohu přesně říct co jaký dělá v praxi; nemám možnost to vyzkoušet).
Tyto parametry se dají spojovat pomocí operátoru „|“ (např. Graphics3D.ANTIALIAS | Graphics3D.TRUE_COLOR). Pokud běží aplikace s některým z těchto parametrů na zařízení, které jej nepodporuje (těch je v dnešní době většina), mělo by jej to ignorovat.

Nyní by třída „packman3DCanvas“ měla 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;

public class packman3DCanvas extends Canvas implements Runnable
{
	Graphics3D g3d = null;

	World world = null;
	Mesh packman = null;
	Camera cam = null;

	int packmanID = 27;

	public packman3DCanvas()
	{
		setFullScreenMode(true);
		loadWorld("/packman.m3g");
		loadObjects();
	}

	public void run()
	{
		try {
			Thread.sleep(30);
		} 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);
		}
	}
```

```
        public void loadObjects()
        {
                cam = world.getActiveCamera();
                packman = (Mesh)world.find(packmanID);
        }

        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();
                }
        }
}
```

Když si nyní naši „hru“ zkompilujete a pustíte, měli byste vidět scénu, jakou jste vyexportovali z vámi použitého 3D editoru.



Pokud vidíte pouze černou obrazovku, může to být způsobeno např. špatným umístěním kamery či světla, nebo se třeba nepodařilo nalézt kameru (nebyla aktivní)…

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.