Le graphique 2D

Introduction

- Texture = une image décodée (par exemple png) et envoyée au GPU.
- Une géométrie est alors décrite et la texture est appliquée.
  - Il faut indiquer les correspondances de points entre la texture et la géométrie.
- Un rectangle qui est une sous-partie d'une texture est appelé une région (texture region).

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Introduction

- Pour dessiner,
  - la texture est chargée (elle devient courante)
  - la géométrie est indiquée.
- La taille et la position sur l'écran sont déterminées par la géométrie et le viewport OpenGL (voir plus tard).
- Il est courant de dessiner les textures dans des rectangles.
- Il est également courant d'utiliser plusieurs fois la même texture ou plusieurs régions de la même texture.
- Optimisation = regrouper les instructions en paquet.
  - Classe SpriteBatch.

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

SpriteBatch

- SpriteBatch reçoit une texture et les coordonnées de chaque rectangle à tracer.
- Les informations sont collectées mais pas envoyées au GPU.
- Les informations sont envoyées à la fin du SpriteBatch ou à l'arrivée d'une nouvelle texture
- Changer souvent de texture implique un envoi fréquent au GPU.
- Il est donc courant de stocker les petites textures dans une seule grosse texture et d'utiliser des texture region (voir TexturePacker plus tard).
- La texture ne change pas mais la région change

→Les paquets sont plus gros et le code mieux optimisé pour le GPU.

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

SpriteBatch

- Squelette :

```
public class Game implements ApplicationListener {

  private SpriteBatch _batch;

  public void create () {

    _batch = new SpriteBatch();

  }

  public void render () {
    Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT); // Efface l'écran

    batch.begin();
      // Dessins
    batch.end();

  }

  …
}
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Texture

- Prendre en compte une image.
- Le fichier devrait se situer dans « assets ».
- GL 1.0 → dimension en puissance de 2
- Pour passer en GL 2.0 :

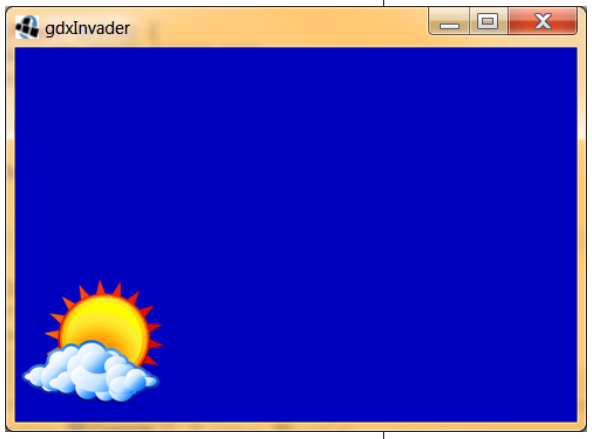
```
package com.me.gdxInvader;

import com.badlogic.gdx.backends.lwjgl.LwjglApplication;

public class Main {
    public static void main(String[] args) {
        LwjglApplicationConfiguration cfg = new LwjglApplicationConfiguration();
        cfg.title = "gdxInvader";
        cfg.useGL20 = false;
        cfg.width = 480;
        cfg.height = 320;

        new LwjglApplication(new GdxInvader(), cfg);
    }
}
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Texture

- Exemple (128x128px) :
  - (0,0) en bas à gauche

```
public class InvaderScreen implements Screen {
  private Texture _texture;
  private SpriteBatch _batch;

  public InvaderScreen() {
     texture = new Texture(Gdx.files.internal("data/soleil.png"));
    _batch = new SpriteBatch();
  }

  @Override public void render(float delta) {
    Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT);
    Gdx.gl.glClearColor(0, 0, .75f, 0);

    _batch.begin();

      _batch.draw(_texture, 0, 0);

    _batch.end();
  }
}
```



http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Texture

- Variantes

Method signature	Description
`draw(Texture texture, float x, float y)`	Draws the texture using the texture's width and height
`draw(Texture texture, float x, float y, int srcX, int srcY, int srcWidth, int srcHeight)`	Draws a portion of the texture.
`draw(Texture texture, float x, float y, float width, float height, int srcX, int srcY, int srcWidth, int srcHeight, boolean flipX, boolean flipY)`	Draws a portion of a texture, stretched to the width and height, and optionally flipped.
`draw(Texture texture, float x, float y, float originX, float originY, float width, float height, float scaleX, float scaleY, float rotation, int srcX, int srcY, int srcWidth, int srcHeight, boolean flipX, boolean flipY)`	This monster method draws a portion of a texture, stretched to thewidth and height, scaled and rotated around an origin, and optionally flipped.
`draw(Texture texture, float x, float y, float width, float height, float u, float v, float u2, float v2)`	This draws a portion of a texture, stretched to the width and height. This is a somewhat advanced method as it uses texture coordinates from 0-1 rather than pixel coordinates.
`draw(Texture texture, float[] spriteVertices, int offset, int length)`	This is an advanced method for passing in the raw geometry, texture coordinates, and color information. This can be used to draw any quadrilateral, not just rectangles.

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Mélange (Blending)

- Par défaut, le blending est actif.
- Les parties transparentes ne sont pas traitées
- Si le blending est désactivé, l'image est affecté par la totalité de la texture.

→Efficacité !

→le Blending devrait être désactivé sauf si nécessaire.

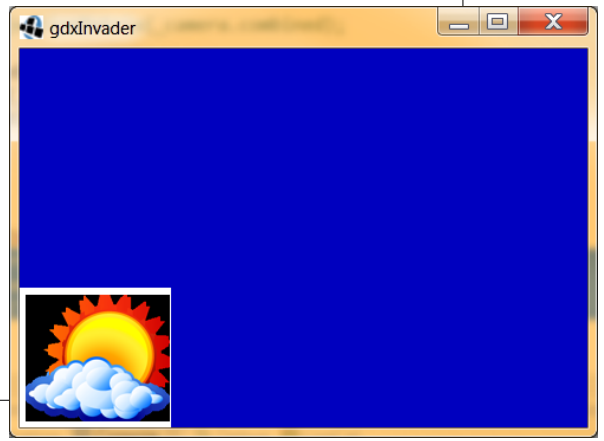
  – Par exemple, affichage du fond d'écran

- Attention : ne pas oublier d'effacer l'écran.

Mélange (Blending)

- Exemple :

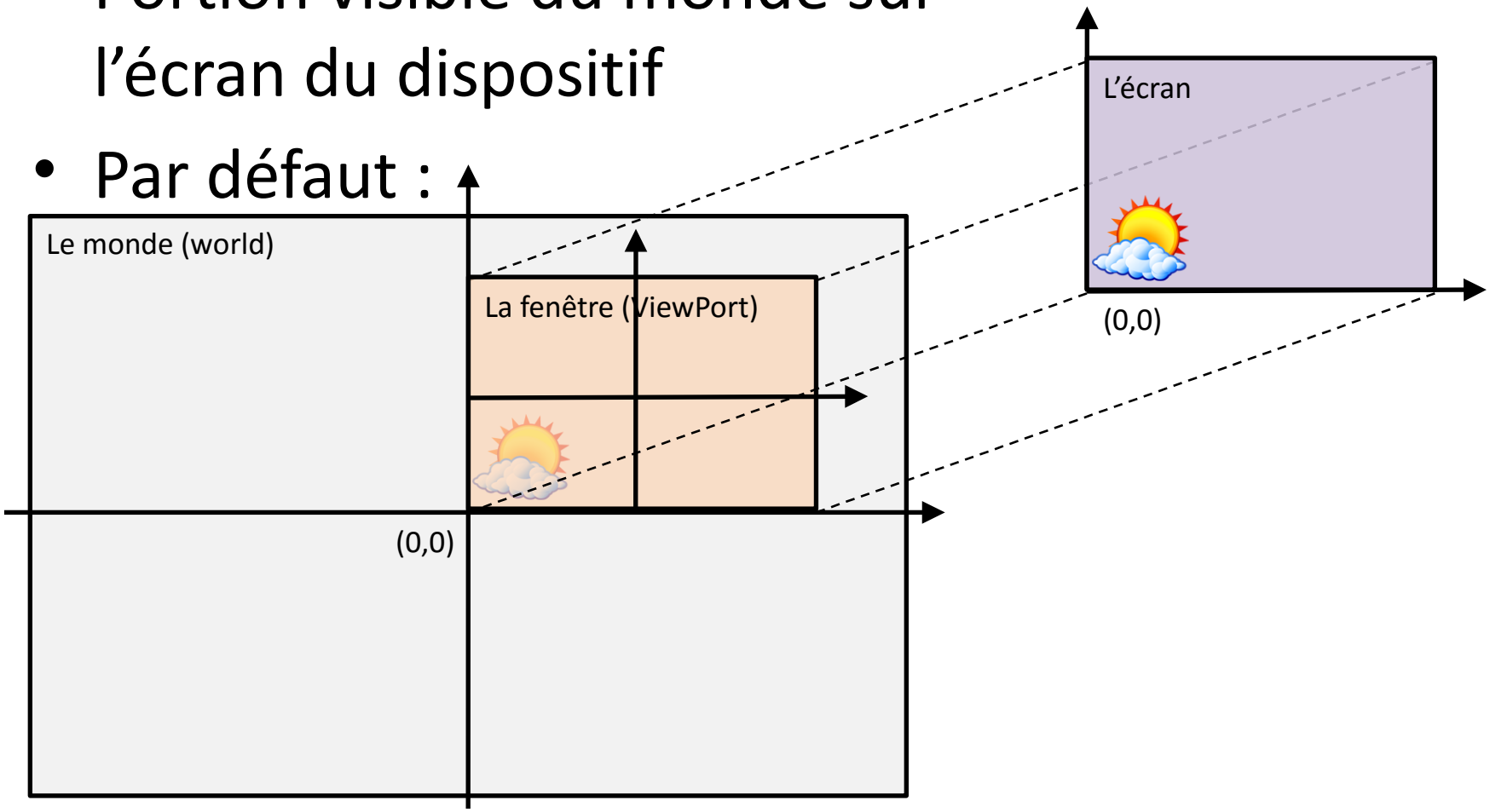
```
public class InvaderScreen implements Screen {
  private Texture _texture;
  private SpriteBatch _batch;

  public InvaderScreen()
  {
    _texture = new Texture(Gdx.files.internal("data/soleil.png"));
    _batch = new SpriteBatch();
  }

  @Override
  public void render(float delta) {
    Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT);
    Gdx.gl.glClearColor(0, 0, .75f, 0);

    _batch.begin();
      _batch.disableBlending();
      _batch.draw(_texture, 0, 0);
      _batch.disableBlending();
    _batch.end();
  }
}
```



http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Viewport

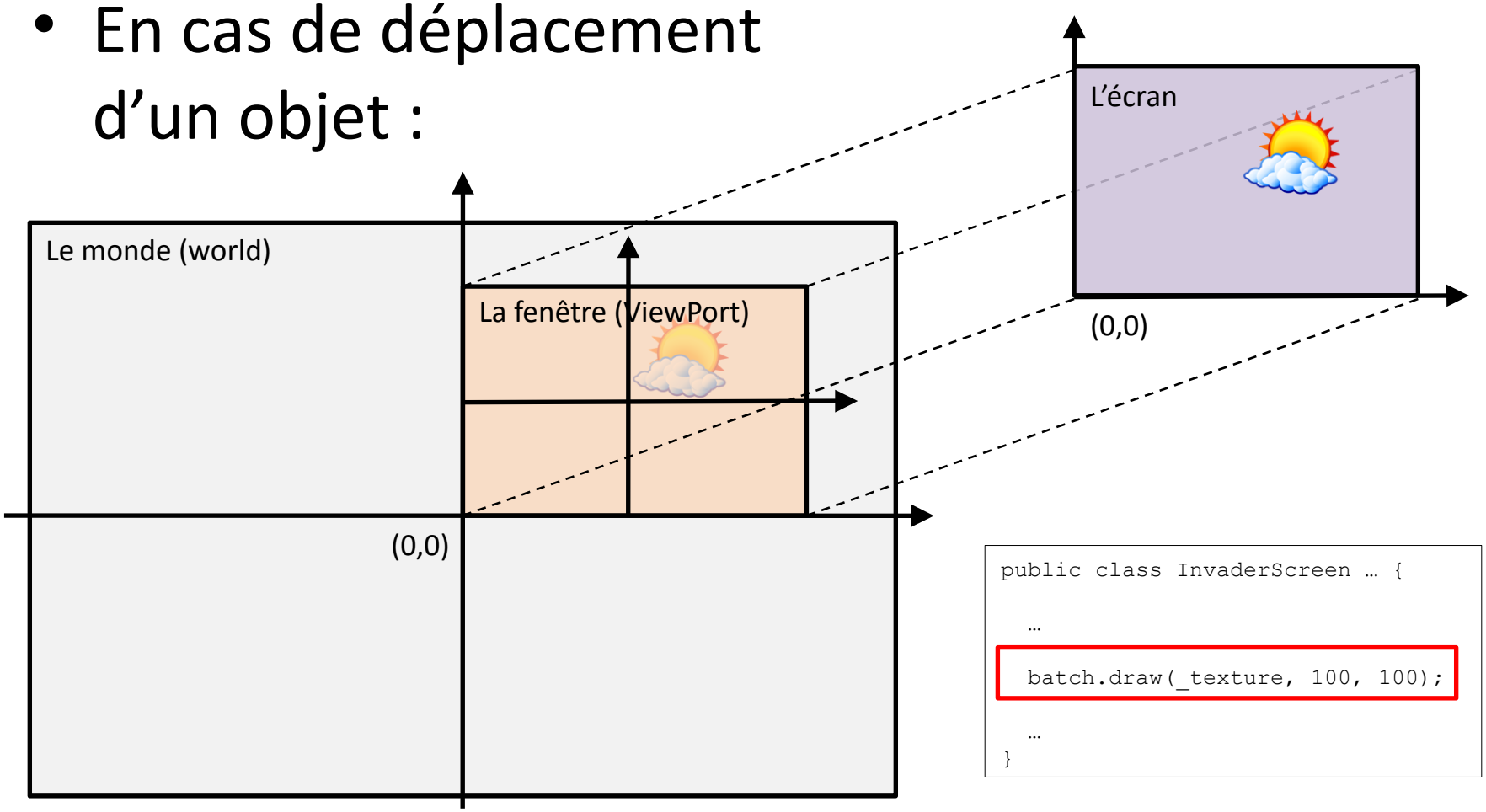
- Portion visible du monde sur l'écran du dispositif
- Par défaut :



http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Viewport

- En cas de déplacement d'un objet :

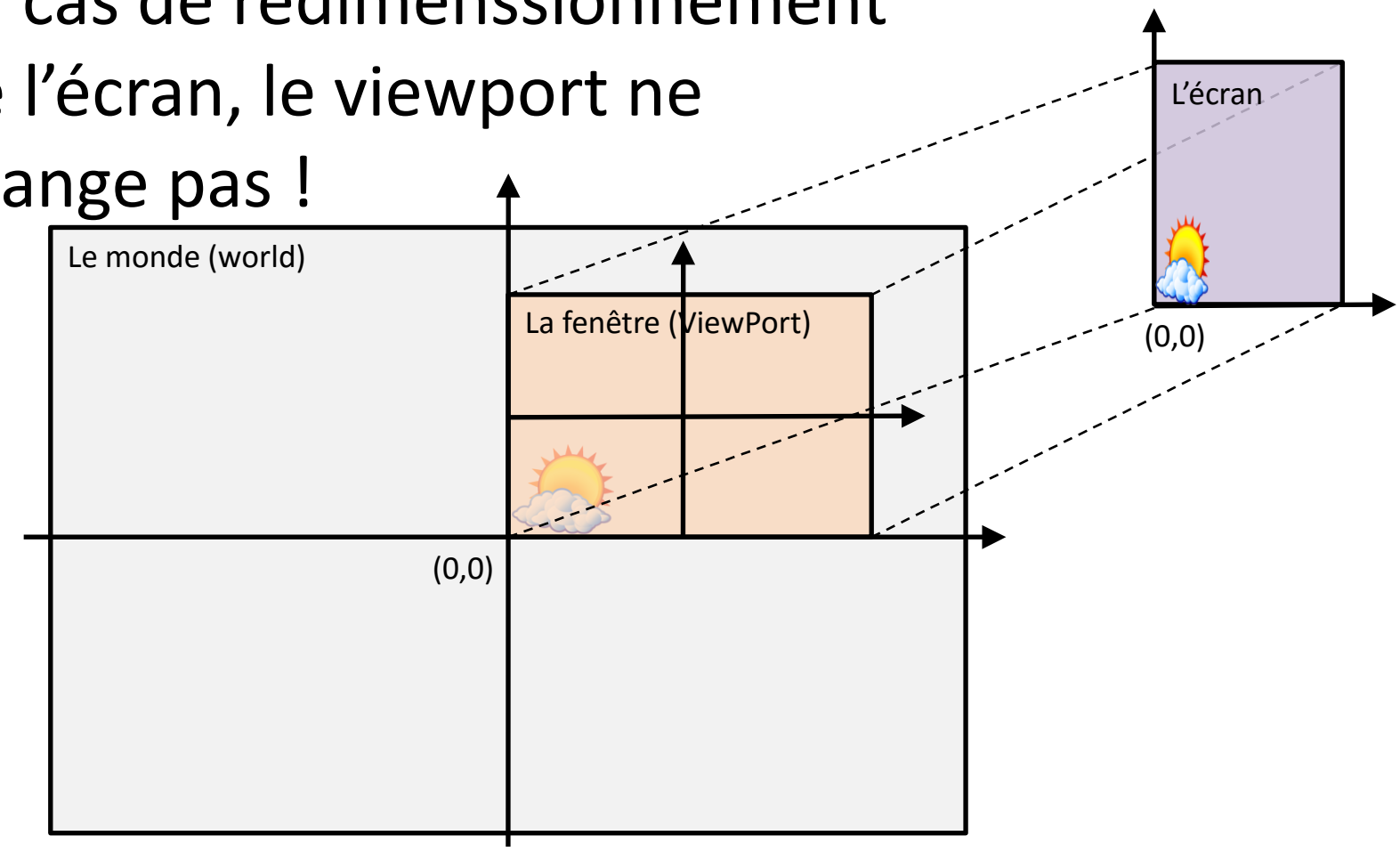



```
public class InvaderScreen … {

  …

  batch.draw(_texture, 100, 100);

  …
}
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Viewport

- En cas de redimenssionnement de l'écran, le viewport ne change pas !



http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Viewport

- Gérer son propre ViewPort = Camera
  - OrthographicCamera
- Création de la caméra :

```
public class InvaderScreen implements Screen {
…
   private OrthographicCamera _camera;

   public InvaderScreen()
   {
      …
      _camera=new OrthographicCamera();
   }


   …
}
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Viewport

- Dimensionner le ViewPort
  - resize

```
public class InvaderScreen implements Screen {
  …

  @Override
  public void resize(int width, int height) {
    _camera.setToOrtho(false, width, height);
    _camera.position.set (width / 2, height / 2, 0);
    _camera.update();
  }
  …
}
```

  - Le ViewPort change à chaque changement de taille d'écran

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Viewport

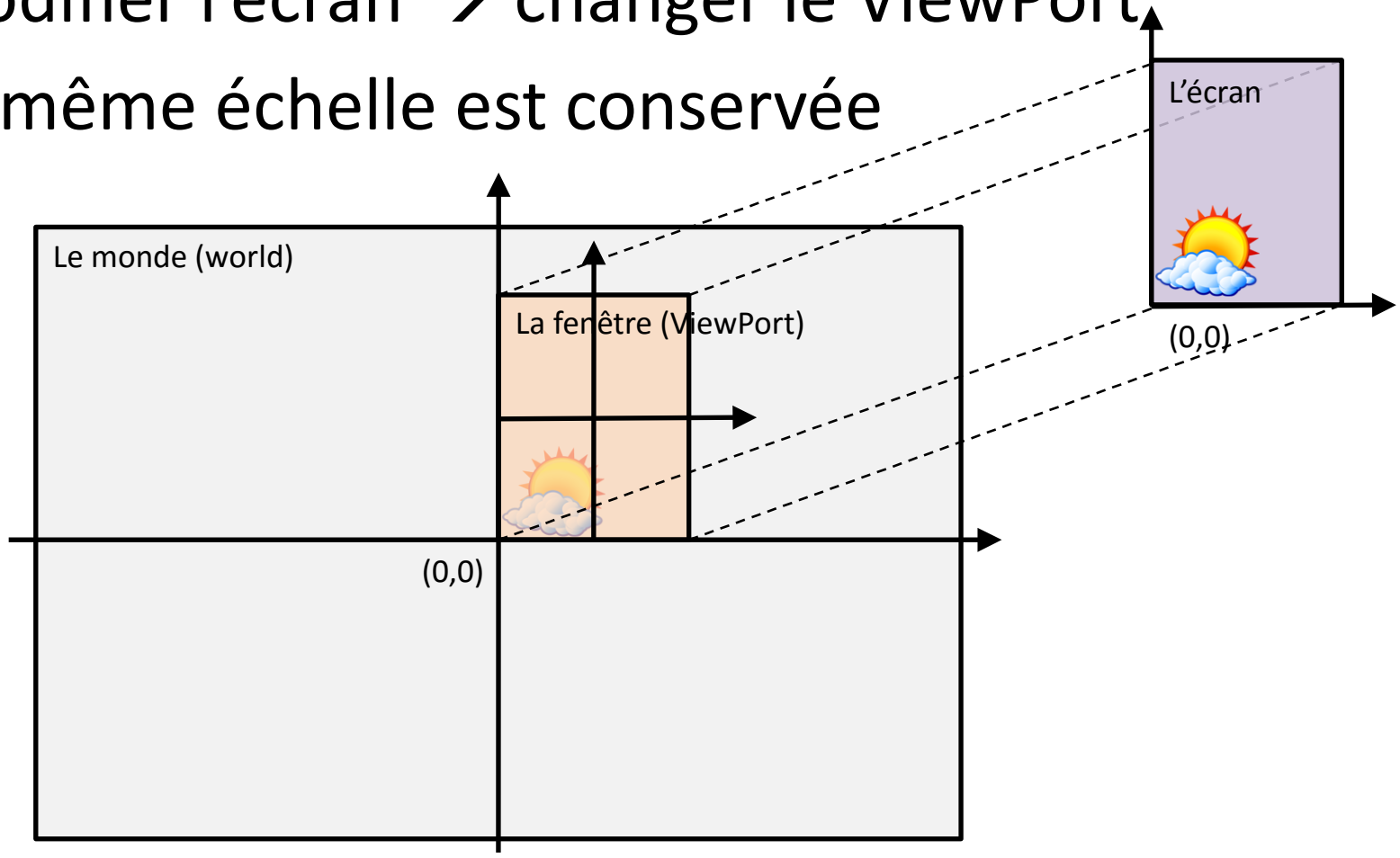
- Prise en compte de la caméra
  - render

```
public class InvaderScreen implements Screen {
  …

  @Override
  public void render(float delta) {
    Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT);
    Gdx.gl.glClearColor(0, 0, .75f, 0);

    _batch.setProjectionMatrix(_camera.combined);

    _batch.begin();
      _batch.draw(_texture, 0, -10);
    _batch.end();
  }

  …
}
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Viewport

- Modifier l'écran → changer le ViewPort
- La même échelle est conservée

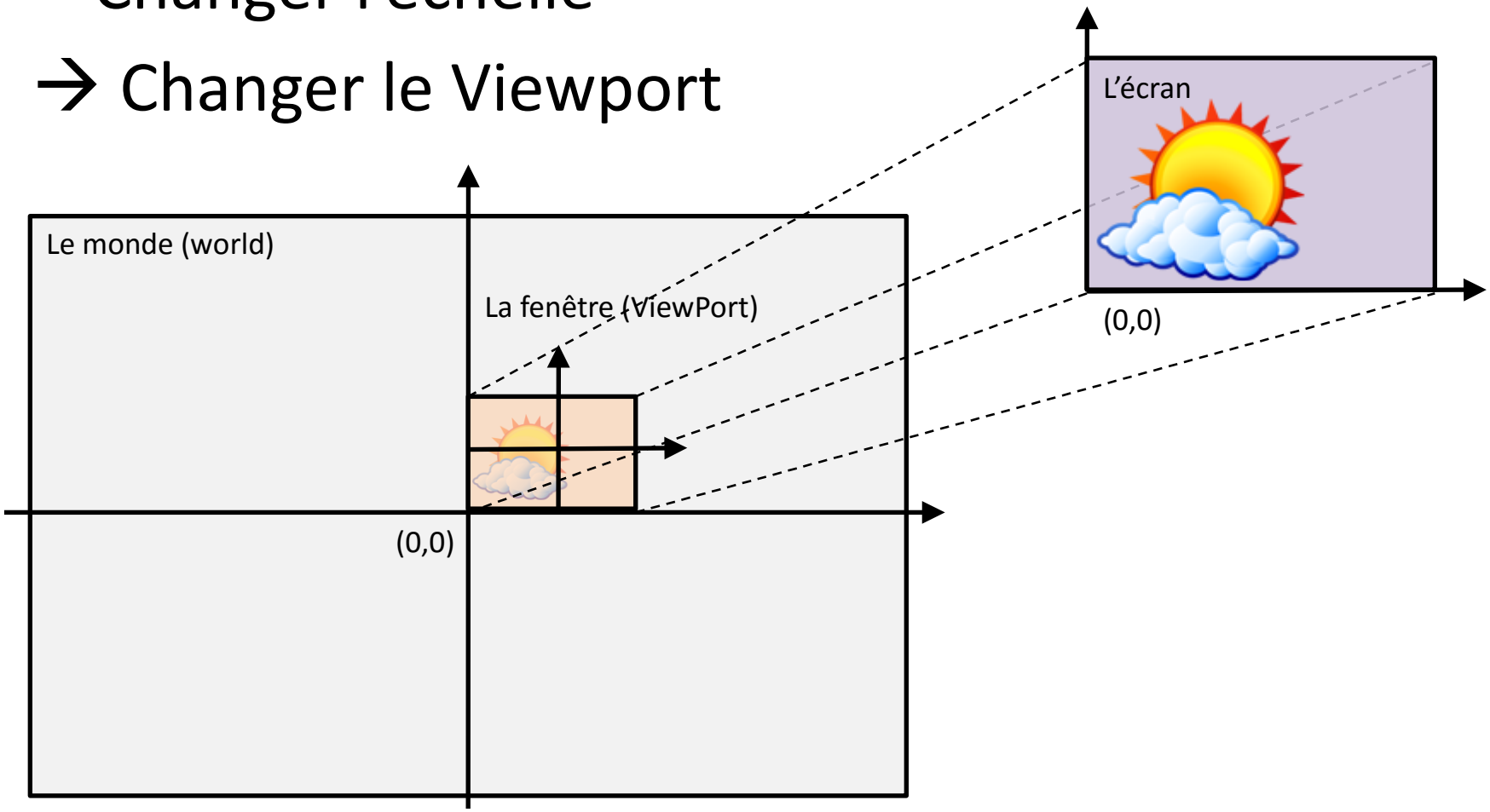


http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Viewport

- Changer l'échelle

→ Changer le Viewport

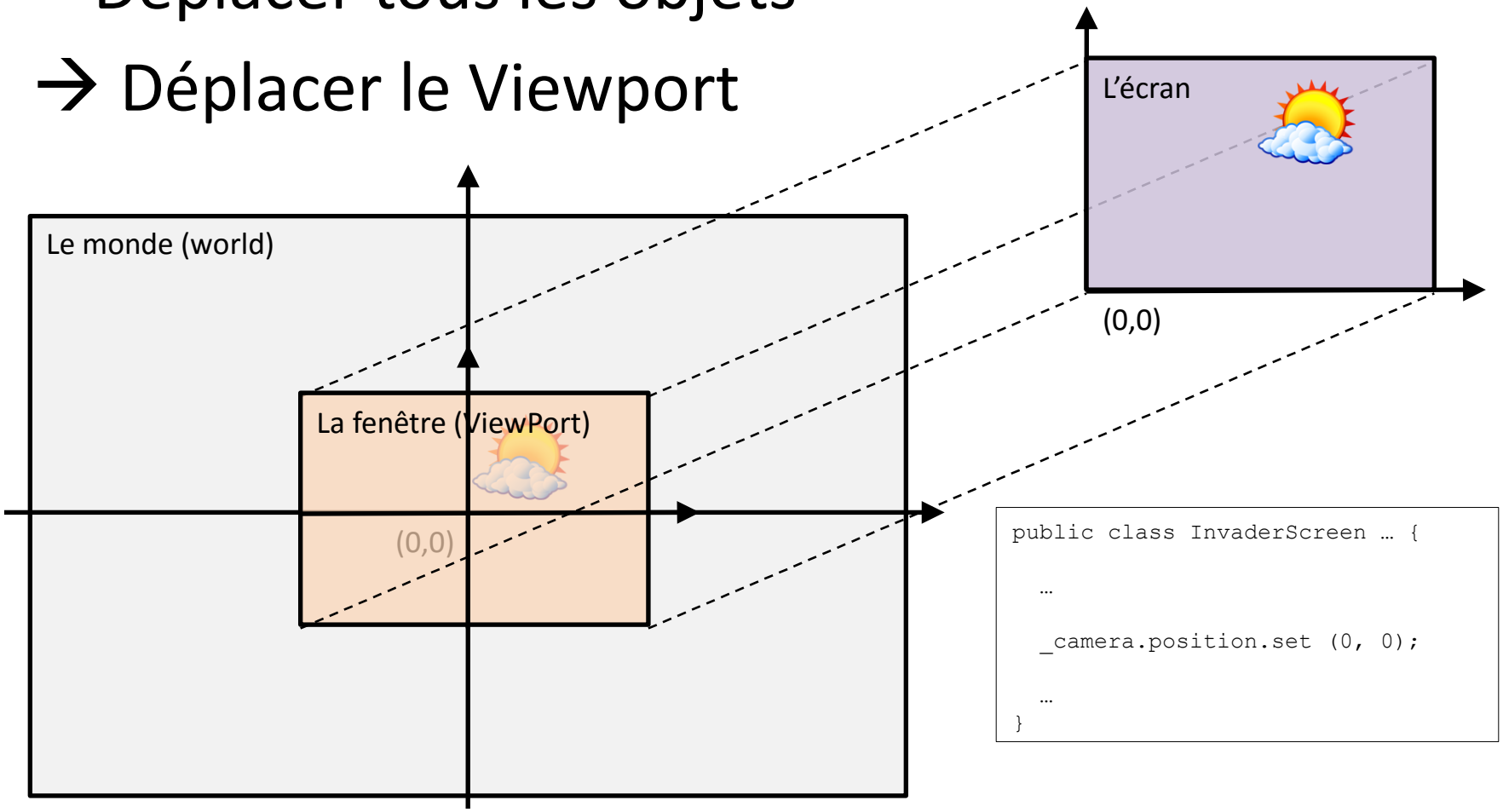


http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Viewport

- Déplacer tous les objets

→ Déplacer le Viewport




```
public class InvaderScreen … {

  …

  _camera.position.set (0, 0);

  …
}
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Sprite

- Regrouper dans un seul objet
  - La texture
  - La région
  - La géométrie
- Attention
  - On associe des données modèle avec des données vue
  - Cela va à l'encontre d'une approche MVC

```
private Sprite sprite;
...
texture = new Texture(Gdx.files.internal("image.png"));
sprite = new Sprite(texture, 20, 20, 50, 50);
sprite.setPosition(10, 10);
sprite.setRotation(45);
...
batch.begin();
sprite.draw(batch);
batch.end();
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

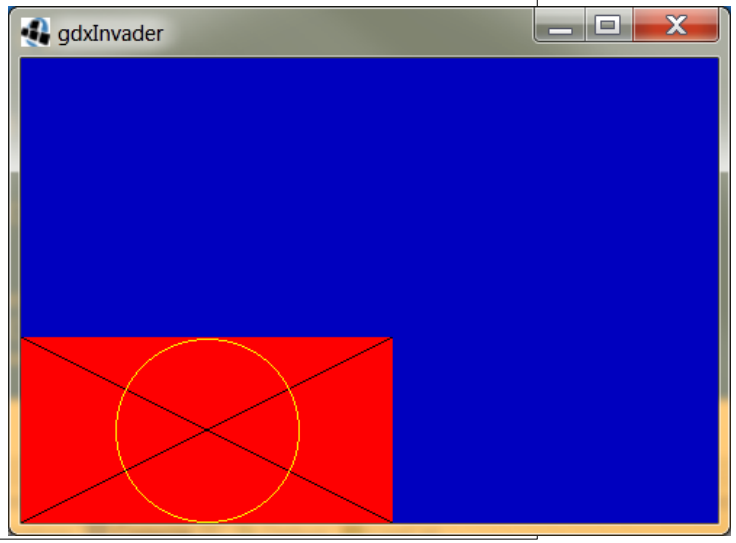
Pixmap

- Préparer une image
  - pour être stockée en mémoire.
  - Pour être téléchargée dans le GPU sous la forme d'une texture.
- On peut dessiner des lignes, des rectangles, etc. et même dessiner pixel par pixel
- Création à partir
  - d'un tableau d'octets contenant les données d'image codées en tant que JPEG , PNG ou BMP
  - d'un fichier
  - de ses dimensions et son format.

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Pixmap

```
public class InvaderScreen implements Screen {

  private Texture _texture;
  private SpriteBatch _batch;
  private Pixmap _pixmap;

  public InvaderScreen() {
    _batch = new SpriteBatch();

    _pixmap = new Pixmap(256,128, Pixmap.Format.RGBA8888);
    _pixmap.setColor(Color.RED);
    _pixmap.fill();
    _pixmap.setColor(Color.BLACK);
    _pixmap.drawLine(0, 0, _pixmap.getWidth()-1, _pixmap.getHeight()-1);
    _pixmap.drawLine(0, _pixmap.getHeight()-1, _pixmap.getWidth()-1, 0);
    _pixmap.setColor(Color.YELLOW);
    _pixmap.drawCircle(_pixmap.getWidth()/2, _pixmap.getHeight()/2, _pixmap.getHeight()/2 - 1);

    _texture = new Texture(_pixmap);

    _pixmap.dispose();
  }

  @Override public void render(float delta) {
    Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT);
    Gdx.gl.glClearColor(0, 0, .75f, 0);

    _batch.begin();
      _batch.draw(_texture, 0, 0);
    _batch.end();
  }
  …

}
```



http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Pixmap

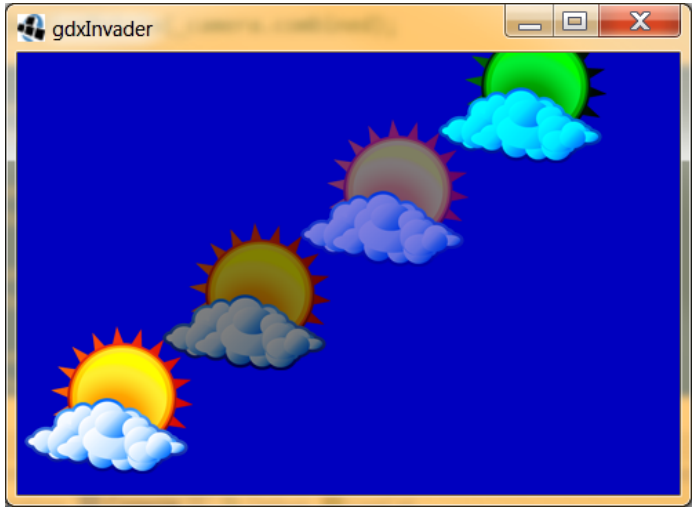
- Les différents formats possibles :
  - RGBA8888 - "True Color". Chaque composante (R, G, B, A) est un octet de 8 bits pour obtenir des couleurs de haute qualité avec transparence.
  - RGBA4444 – Similaire au précédent mais avec seulement 4bits par composante. Cela permet de meilleures performances et une consommation mémoire moins importante.
  - RGB565 – Les composantes rouges et bleues sont représentées sur 5bits. Le vert l'est sur 6bits car l'oeil humain perçoit généralement plus de nuances de vert.
  - LuminanceAlpha – Image à niveau de gris avec une composante de transparence. Les couleurs à niveau de gris ont les trois composantes RGB égales. Ainsi (R=127, G=127, B=127, A=255) est représenté par LuminanceAlpha: (L=127, A=255). Chaque composante utilise 8 bits.
  - Alpha – Image particulière qui ne stocke que la transparence sur 8 bits.
  - Intensity – Image particulière avec un seul canal. Par exemple, une couleur (L=127) sera équivalente à (R=127, G=127, B=127, A=127).

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Tinting

- Méthode setColor (SpriteBatch, Sprite)
  - Modifier la couleur
  - Par défaut Color.WHITE (1,1,1,1)

```
_batch.begin();
  _batch.setColor (Color.WHITE);
  _batch.draw(_texture, 0, 0);

  _batch.setColor(.5f, .5f, .5f, 1f);

  _batch.draw(_texture, 100, 75);

  _batch.setColor(1f, 1f, 1f, .5f);
  _batch.draw(_texture, 200, 150);

  _batch.setColor(0f, 1f, 1f, 1f);
  _batch.draw(_texture, 300, 225);
_batch.end();
```

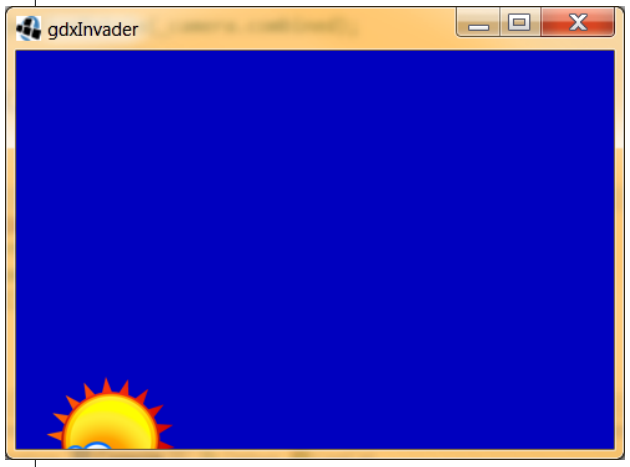


http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

TextureRegion

- Prendre en compte une partie d'une texture
- Créer la texture et ensuite créer la région

```
public class InvaderScreen implements Screen {
  private Texture _texture;
  private TextureRegion  region;
  private SpriteBatch _batch;

  public InvaderScreen() {
    _texture = new Texture(Gdx.files.internal("data/soleil.png"));
    _region = new TextureRegion (
                          _texture,
                          0, 0,
                          _texture.getWidth(),
                          _texture.getHeight() /2);
    …
  }

  @Override public void render(float delta) {
    …
    _batch.draw(_region, 0, 0);
    …
  }
}
```



http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

TextureRegion

- Variantes

Method signature	Description
`draw(TextureRegion region, float x, float y)`	Draws the region using the width and height of the region.
`draw(TextureRegion region, float x, float y, float width, float height)`	Draws the region, stretched to the width and height.
`draw(TextureRegion region, float x, float y, float originX, float originY, float width, float height, float scaleX, float scaleY, float rotation)`	Draws the region, stretched to the width and height, and scaled and rotated around an origin.

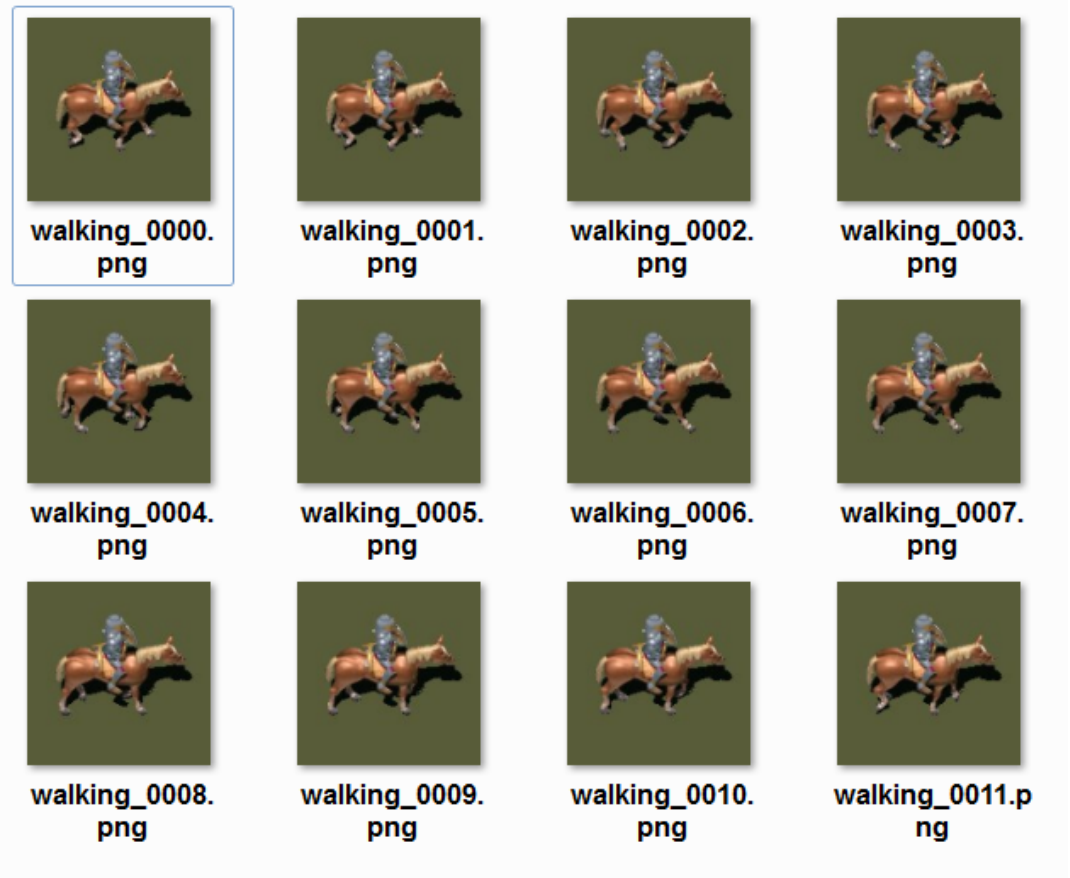
Texture Atlases

- Simplifier l'usage de TextureRegion
- Classe TextureAtlas
- Accéder aux texture stockées sous forme de régions dans une image unique
- Créer l'image unique qui va contenir les différentes textures

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

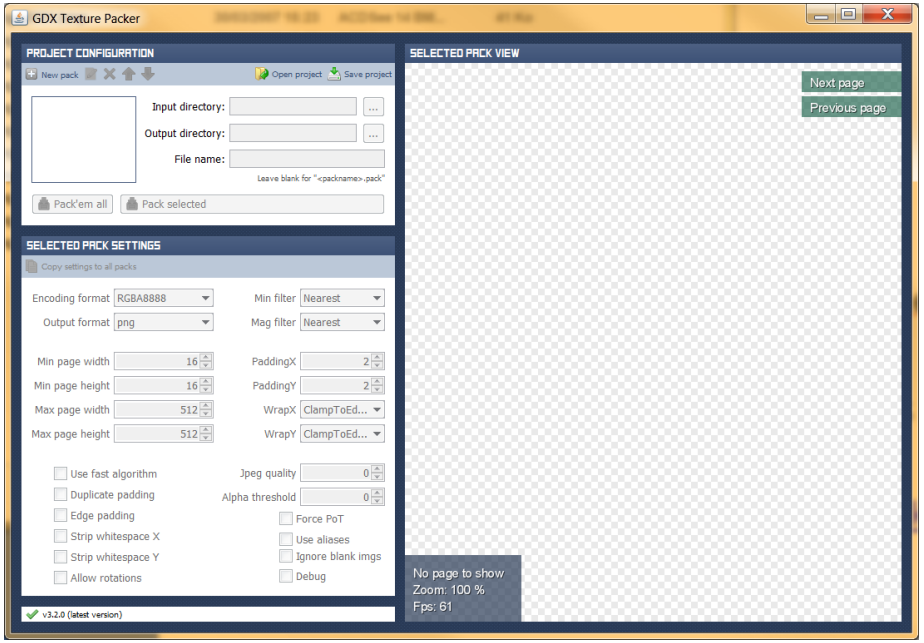
Texture Atlases

- Les images : http://www.reinerstilesets.de/



Texture Atlases

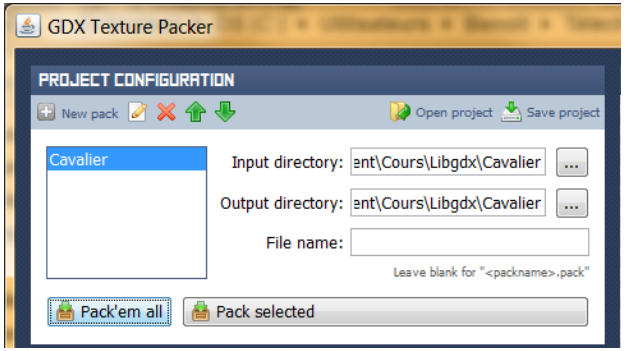
- Plusieurs outils
  - http://www.codeandweb.com/texturepacker
  - https://code.google.com/p/libgdx-texturepacker-gui/
- Ce dernier ne demande pas d'installation
- Exécuter gdx-texturepacker



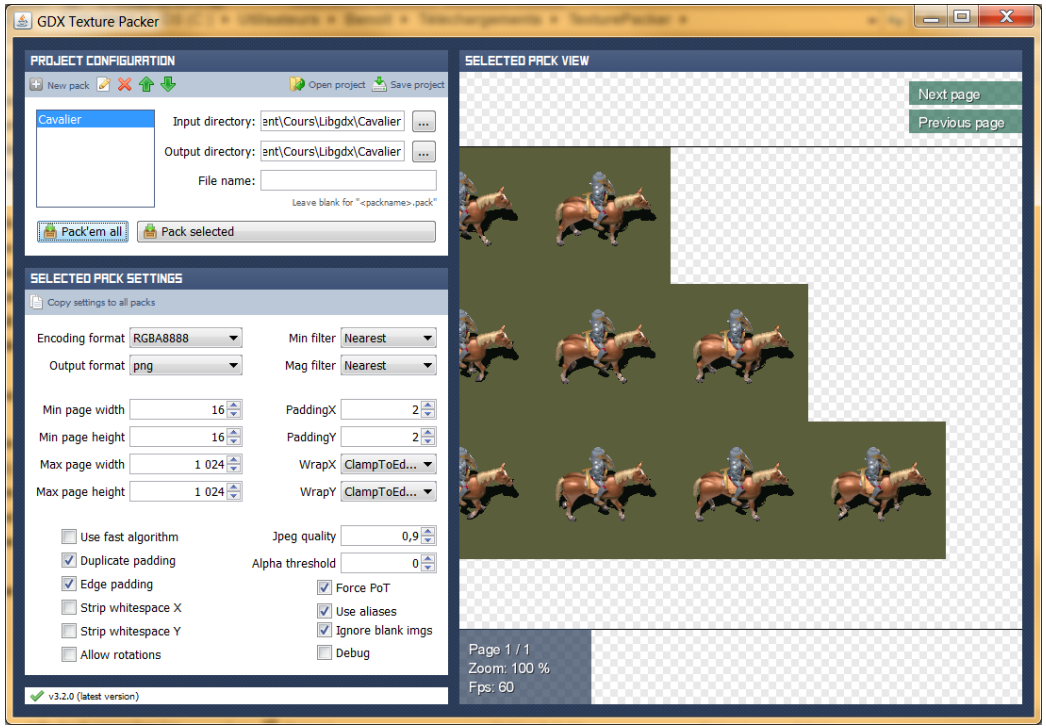
http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Texture Atlases

- Créer un pack :



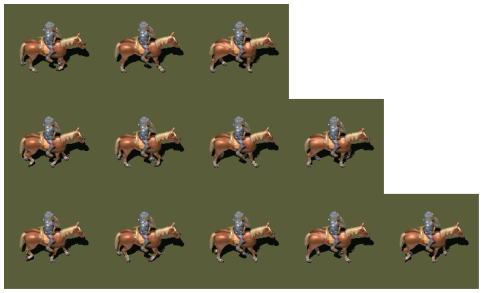
- Pack'em all :



http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Texture Atlases

- Dans le répertoire de sortie :
  - Un fichier Cavalier.png



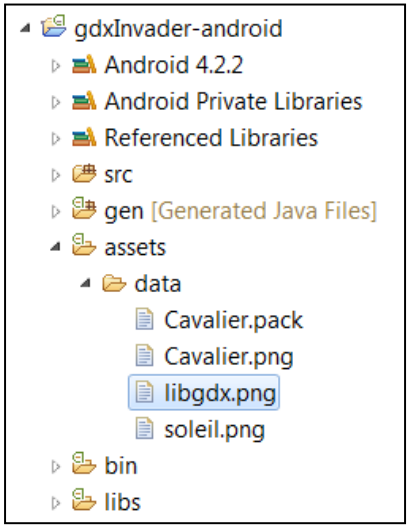
  - Un fichier Cavalier.pack
    - Fichier texte avec les positions des textures dans l'image
    - Et les index … car les noms des fichiers étaient nom_n°

```
Cavalier.png
format: RGBA8888
filter: Nearest,Nearest
repeat: none
walking
  rotate: false
  xy: 1, 293
  size: 144, 144
  orig: 144, 144
  offset: 0, 0
  index: 0
walking
  rotate: false
  xy: 1, 147
  size: 144, 144
  orig: 144, 144
  offset: 0, 0
  index: 1
walking
  rotate: false
  xy: 147, 293
…
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Texture Atlases

- Ajouter ces deux fichiers au projet dans Eclipse



- Charger le fichier pack pour créer l'atlas

```
public class InvaderScreen implements Screen {

  private TextureAtlas _spriteSheet;

  public InvaderScreen() {
    …
    _spriteSheet = new TextureAtlas(Gdx.files.internal("data/Cavalier.pack"));
  }

  …
}
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Texture Atlases

- Accéder aux Sprite à l'intérieur de l'atlas :
  - Nom du fichier d'origine
  - CreateSprite
    - Retourne le premier Sprite dont le nom correspond
  - CreateSprites
    - Retourne un array de Sprites dont le nom correspond
    - En respectant les index

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Texture Atlases

- Extraire les Sprites de l'atlas :

```
…
import com.badlogic.gdx.utils.Array;

public class InvaderScreen implements Screen {

  private SpriteBatch _batch;
  private TextureAtlas _spriteSheet;
  private Array<Sprite> _skeleton;

  public InvaderScreen() {
    _batch = new SpriteBatch();
    _spriteSheet = new TextureAtlas(Gdx.files.internal("data/Cavalier.pack"));

    _skeleton = _spriteSheet.createSprites("walking");
  }

  …
}
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Texture Atlases

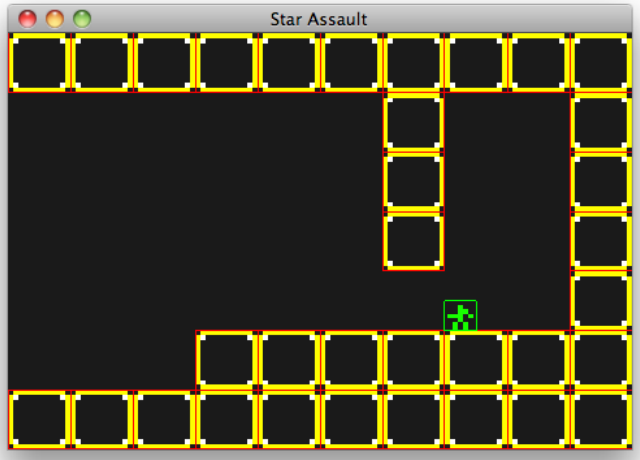
- Utiliser et animer les Sprites :

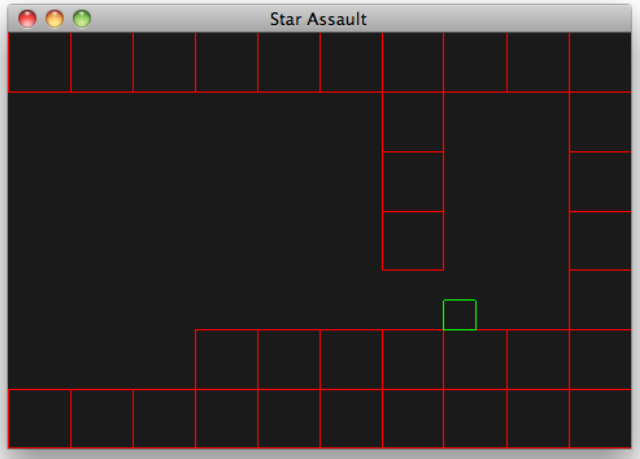
```
public class InvaderScreen implements Screen {

  private int _cpt = 0;
  private float _elapsedTime = 0;
  private final float _frameLength = .075f;

  @Override public void render(float delta) {
    Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT);
    Gdx.gl.glClearColor(0, 0, .75f, 0);

    _elapsedTime += delta;
    while (_elapsedTime > _frameLength) {
      _elapsedTime -= _frameLength;
      _cpt = (_cpt + 1) % _skeleton.size;
    }

    _batch.begin();
      _skeleton.get(_cpt).draw (_batch);
    _batch.end();
  }

  …
}
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

ShapeRenderer

- Afficher des
  - Points,
  - Lignes,
  - Cercles,
  - Rectangles,
  - Etc.
- Utile pour le debug …





http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

ShapeRenderer

- Les commandes sont “batchées”
- Exemple classique d’usage :

```
camera.update();
shapeRenderer.setProjectionMatrix(camera.combined);

shapeRenderer.begin(ShapeType.Line);
shapeRenderer.setColor(1, 1, 0, 1);
shapeRenderer.line(x, y, x2, y2);
shapeRenderer.rect(x, y, width, height);
shapeRenderer.circle(x, y, radius);
shapeRenderer.end();

shapeRenderer.begin(ShapeType.Filled);
shapeRenderer.setColor(0, 1, 0, 1);
shapeRenderer.rect(x, y, width, height);
shapeRenderer.circle(x, y, radius);
shapeRenderer.end();
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Les collisions

- Méthode Simple : utiliser la classe Intersector
- Cette classe fournit différentes méthodes statiques pour tester
  - Distances
  - Intersections
  - Superposition
- Entre différentes géométries
  - Triangles, Cercles, Rectangles, etc.

Les collisions

- Pour obtenir le rectangle intersecté :

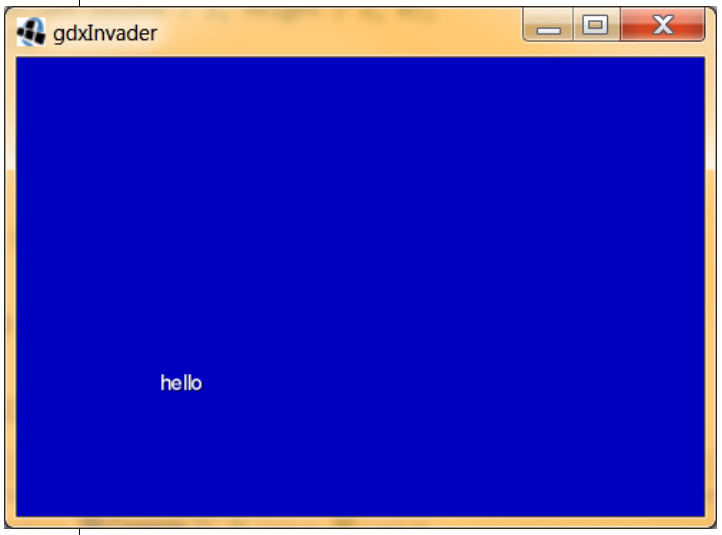
```
public static boolean intersect(
                  Rectangle r1,
                  Rectangle r2,
                  Rectangle intersection)
{
    if (!r1.overlaps(r2)) {
        return false;
    }

    float x = Math.max(r1.x, r2.x);
    float y = Math.max(r1.y, r2.y);
    float width = Math.min(r1.x + r1.width, r2.x + r2.width) - x;
    float height = Math.min(r1.y + r1.height, r2.y + r2.height) - y;
    intersection.set(x, y, width, height);

    return true;
}
```

http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Le Texte

- Classe BitmapFont
  - Par défaut une police Arial 15 est disponible

```
public class InvaderScreen implements Screen {
  private SpriteBatch _batch;
  private BitmapFont _font;

  public InvaderScreen() {
    _batch = new SpriteBatch();
    _font = new BitmapFont();
  }

  @Override public void render(float delta) {
    Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT);
    Gdx.gl.glClearColor(0, 0, .75f, 0);

    _batch.begin();
      _font.draw (_batch, "hello", 100, 100);
    _batch.end();
  }
  …
}
```



http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android