Les entrées

Les entrées : introduction

- Entrée = interaction de l'utilisateur vers l'application (jeu)
- But : gérer les interactions
  - communes aux différentes plateformes
  - spécifiques à une plateforme

~ Un seul code source

→Libgdx gère les différences

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

Les entrées : introduction

- Desktop
  - Souris
  - Mouvement de la souris (move, hover, etc.)
  - Un seul bouton est pris en compte par Libgdx
  - Clavier
- Android
  - Toucher tactile
  - Multi-touch
  - Clavier générique
  - Touches spécifiques : touche de retour, de menu, de recherche, etc.

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

Les entrées : le texte

- Entrée de texte = boîte de dialogue
- Desktop = boîte de dialogue SWING
- Création de la boîte de dialogue :

```
Gdx.input.getTextInput(ecouteur, "Titre", "texte initial");
```

- L'écouteur représente la réaction du système lorsque l'utilisateur interagit avec la boîte de dialogue
- Interface TextInput Listener

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

Les entrées : le texte

```
public class MonEcouteur implements TextInputListener {
   @Override
   public void input (String text) {
   }
   @Override
   public void canceled () {
   }
}
```

- Input
  - Appelée lorsque l'utilisateur aura entré une chaîne de caractères et appuyé sur OK.
- Canceled
  - Appelée lorsque l'utilisateur aura fermé la boîte de dialogue (sous Desktop) ou appuyé sur le bouton retour (sous Android).

Les entrées : le clavier

- Deux mécanismes
- Le polling = l'interrogation
  - Un évènement clavier déclenché par un utilisateur (en appuyant ou en relâchant une touche du clavier) génère un code qui identifie la touche concernée.
  - Desktop et Android ne possèdent pas les mêmes codes → Libgdx utilise sa propre table de code des touches clavier (classe Keys).
  - Principe : vérifier de façon répété si une touche a été manipulée par l'utilisateur
  - E

```
boolean isAPressed = Gdx.input.isKeyPressed(Keys.A);
```

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

Les entrées : le clavier

- L'écouteur = l'attente/réveil
  - Permet de suivre tous les événements
  - Un évènement clavier déclenche une méthode sur un écouteur
  - Implémenter la classe InputProcessor

```
public class MonEcouteur implements InputProcessor { … }
```

  - Attacher l'écouteur à Libgdx

```
Gdx.input.setInputProcessor(new MonEcouteur ( … ));
```

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

Les entrées : le clavier

```
public class MonEcouteur
      implements InputProcessor {
  @Override
  public boolean keyDown(int codeCle) { … }
  @Override
  public boolean keyTyped(char caractere) { … }
  @Override
  public boolean keyUp(int codeCle) { … }
  …
}
```

- Classe InputProcessor
- keyDown
  - Appelée quand une touche a été appuyée (en paramètre le code de la touche appuyée).
- keyUp
  - Appelée quand une touche a été relâchée (en paramètre le code de la touche relâchée).
- keyTyped
  - Appelée quand un caractère Unicode est généré par une entrée du clavier (en paramètre le caractère généré).
  - A l'appui … même pour les touches spéciales (shift, etc.).

Les entrées : le clavier

- Un exemple :
  - Méthodes non liées au clavier …. (voir plus tard)

```
class MonEcouteur implements InputProcessor
{
  @Override public boolean mouseMoved(int screenX, int screenY) { return false; }

  @Override public boolean scrolled(int arg0) { return false; }

  @Override
  public boolean touchDown(int arg0, int arg1, int arg2, int arg3) { return false; }

  @Override public boolean touchDragged(int arg0, int arg1, int arg2) {
    return false;
  }

  @Override public boolean touchUp(int a, int arg1, int arg2, int arg3) {
    return false;
  }
  …
}
```

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

Les entrées : le clavier

- Un exemple :

```
class MonEcouteur implements InputProcessor
{
  …
  private ArrayList _codes; // liste des codes des touches enfoncées
  private String _car; // dernier caractère entré
  private String _msg; // message à afficher (_codes + _car)

  public MonEcouteur ()
  {
    _codes = new ArrayList (); // la liste des codes est vide
    _car = ""; // pas de caractère tapé au début
    update (); // on met à jour le message
  }

  private void update () {
    _msg = "";
    for (Object c : _codes) {
      _msg += c + " ";
    }
    _msg += _car;
  }

  public String getMsg () {
    return _msg;
  }
}
```

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

Les entrées : le clavier

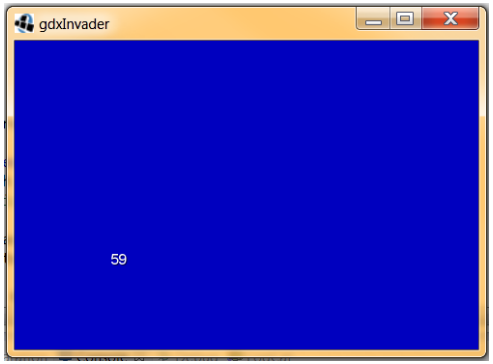
- Un exemple :

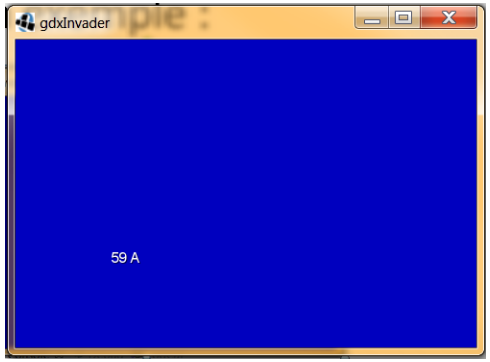
```
class MonEcouteur implements InputProcessor
{
  …
  @Override public boolean keyDown(int codeCle) {
    _codes.add(new Integer (codeCle));
    update ();
    return false;
  }

  @Override public boolean keyUp(int codeCle) {
    _codes.remove(new Integer (codeCle));
    update ();
    return false;
  }

  @Override public boolean keyTyped(char caractere) {
    _car = Character.toString(caractere); //récupérer le caractère
    update ();
    return false;
  }
}
```

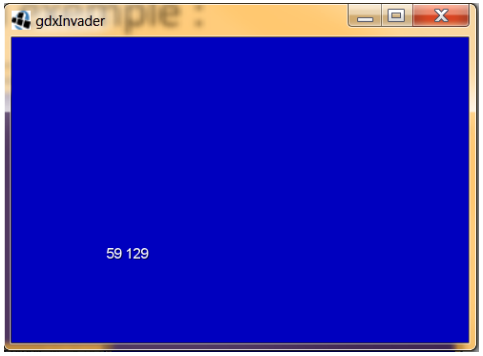
http://www.siteduzero.com/informatique/tutoriels/creez-des-applications-pour-android

Les entrées : le clavier

- Un exemple :
  - Appui / maintien Shift
  - Appui / relâchement 'a'
  - Appui / Maintien Ctrl







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

Les entrées : souris / écran tactile

- Deux mécanismes comme pour le clavier
- Polling = ensemble de méthodes pour vérifier l'état actuel du périphérique d'entrée
  - Principe : vérifier de façon répétée si une entrée avec Souris/Ecran tactile a été effectuée.
  - Le polling est un moyen rapide et facile pour traiter les entrées des utilisateurs.
  - Android : tester si l'écran est touché par un ou plusieurs doigts, etc.
  - Desktop : tester si on a effectué un clic avec la souris, etc.

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

Les entrées : souris / écran tactile

- Android : vérifier si un ou plusieurs doigts sont sur l'écran tactile :

```
boolean estTouche = Gdx.input.isTouched();
```

  - Sous Desktop : estTouche est à vrai si on clique sur l'écran.
- Obtenir les coordonnées d'un clic avec la souris / doigt sur l'écran :

```
float X = Gdx.input.getX();
float Y = Gdx.input.getY();
```

- On peut aussi obtenir la distance qui sépare un premier clic et un deuxième :

```
float deltaX =  Gdx.input.getDeltaX();
float deltaY =  Gdx.input.getDeltaY();
```

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

Les entrées : souris / écran tactile

- Android : multi-touch

```
boolean premierDoigt = Gdx.input.isTouched(0);
boolean deuxiemeDoigt = Gdx.input.isTouched(1);
boolean troisiemeDoigt = Gdx.input.isTouched(2);
```

- Un numéro à chaque doigt qui touche l'écran tactile (le plus petit indice disponible).
- Exemple :
  - Premier doigt touche l'écran → 0
  - Deuxième doigt touche l'écran → 1
  - Troisième doigt touche l'écran → 2
  - Troisième doigt quitte (se lève de) l'écran → 2 se libère
  - Deuxième doigt quitte (se lève de) l'écran → 1 se libère
  - Un nouveau doigt touche l'écran → 1

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

Les entrées : souris / écran tactile

- Android : multi-touch
- Les coordonnées :

```
int premierDoigtX = Gdx.input.getX();
int premierDoigtY = Gdx.input.getY();
int deuxiemeDoigtX = Gdx.input.getX(1);
int deuxiemeDoigtY = Gdx.input.getY(1);
int troisiemeDoigtX = Gdx.input.getX(2);
int troisiemeDoigtY = Gdx.input.getY(2);
```

- Par défaut, l'indice est 0 et correspond à la souris sur Desktop.

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

Les entrées : souris / écran tactile

- Android : multi-touch
- Les coordonnées :

```
int premierDoigtX = Gdx.input.getX();
int premierDoigtY = Gdx.input.getY();
int deuxiemeDoigtX = Gdx.input.getX(1);
int deuxiemeDoigtY = Gdx.input.getY(1);
int troisiemeDoigtX = Gdx.input.getX(2);
int troisiemeDoigtY = Gdx.input.getY(2);
```

- Par défaut, l'indice est 0 et correspond à la souris sur Desktop.

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

Les entrées : souris / écran tactile

- Desktop : savoir quel bouton de la souris est enfoncé

```
boolean boutonGauche = Gdx.input.isButtonPressed(Input.Buttons.LEFT);
boolean boutonDroit  = Gdx.input.isButtonPressed(Input.Buttons.RIGHT);
boolean boutonMilieu = Gdx.input.isButtonPressed(Input.Buttons.MIDDLE);
```

- Android : seul le bouton gauche est émulé
  - Un doigt sur l'écran = bouton gauche

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

Les entrées : souris / écran tactile

- Par écouteurs
  - Comme pour le clavier, classe InputProcessor

```
class MonEcouteur implements InputProcessor
{
  …

  @Override public boolean mouseMoved(int screenX, int screenY) { return false; }

  @Override public boolean scrolled(int arg0) { return false; }

  @Override
  public boolean touchDown(int arg0, int arg1, int arg2, int arg3) { return false; }

  @Override public boolean touchDragged(int arg0, int arg1, int arg2) {
    return false;
  }

  @Override public boolean touchUp(int a, int arg1, int arg2, int arg3) {
    return false;
  }
}
```

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

Les entrées : souris / écran tactile

- TouchDown
  - Un doigt se pose sur l'écran (Android) / un bouton la souris a été pressé (desktop).
  - Coordonnées, indice du pointeur et le bouton de la souris.
- touchUp
  - Un doigt a été levé de l'écran(Android) / un bouton de la souris a été relâché (Desktop).
  - Coordonnées, indice du pointeur et le bouton de la souris.
- touchDragged
  - Un doigt glisse sur l'écran (Android) / la souris se déplace avec un bouton appuyé (Desktop).
  - Coordonnées et indice du pointeur (pour le bouton, il faut utiliser le polling).
- touchMoved
  - La souris se déplace sur l'écran sans bouton appuyé (Desktop uniquement).
- Scrolled
  - La molette de la souris est tournée (Desktop uniquement).

Les entrées : les gestes

- Android
- Interface GestureListener / Classe GestureDetector
- Détection « automatique » de certains gestes
  - Appuyer/toucher (tap) : l'utilisateur touche l'écran et relève le doigt en restant dans une zone carrée spécifiée autour de la position initiale.
  - Pan (défiler) : l'utilisateur fait glisser un doigt sur l'écran. Le détecteur indique la position touchée et différence (le delta) entre les différentes positions.
  - Balayer (fling) : l'utilisateur fait glisser un doigt sur l'écran, puis le soulève.
  - Zoom : l'utilisateur met deux doigts sur l'écran et les déplace. Le détecteur renvoie la distance initiale et la nouvelle distance entre les doigts.
  - Pincer/dézoomer (pintch) : Similaire au zoom, sauf que le détecteur renvoie la position des doigts au lieu de la distance. Ce geste est utile pour détecter d'autres gestes.

```
Gdx.input.setInputProcessor(new MonEcouteur ( … ));
```

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

Les entrées : les gestes

```
interface GestureListener {
  abstract public boolean touchDown(float x, float y, int pointer, int button);
  abstract public boolean tap(float x, float y, int count, int button);
  abstract public boolean longPress(float x, float y);
  abstract public boolean fling(float velocityX, float velocityY, int button);
  abstract public boolean pan(float x, float y, float deltaX, float deltaY);
  abstract public boolean zoom(float initialDistance, float distance);
  abstract public boolean pinch(Vector2 initialPointer1, Vector2 initialPointer2,
            Vector2 pointer1, Vector2 pointer2);
}
```

```
class MonEcouteur implements GestureListener {
  @Override boolean touchDown(float x, float y, int pointer, int button) { … }
  @Override public boolean tap(float x, float y, int count, int button);
  @Override public boolean longPress(float x, float y);
  @Override public boolean fling(float velocityX, float velocityY, int button);
  @Override public boolean pan(float x, float y, float deltaX, float deltaY);
  @Override public boolean zoom(float initialDistance, float distance);
  @Override public boolean pinch(Vector2 initialPointer1, Vector2 initialPointer2,
            Vector2 pointer1, Vector2 pointer2);
}
```

```
GestureDetector gd;
Gd = new GestureDetector (new MonEcouteur (…));
Gdx.input.setInputProcessor(gd);
```

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

Les entrées : comment les mixer ?

```
class MonEcouteurClavier implements InputProcessor {
   …
}
```

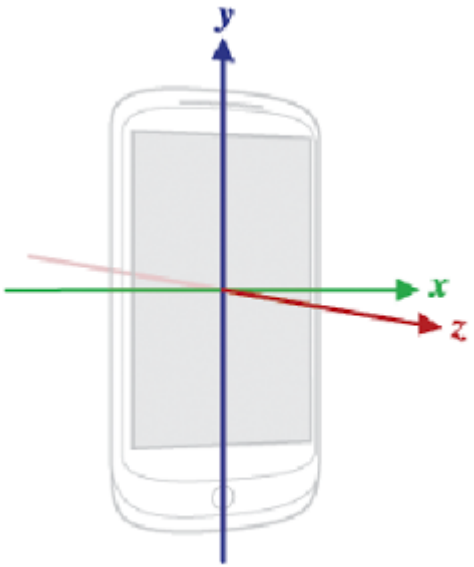
```
class MonEcouteurGeste implements GestureListener {
   …
}
```

```
InputProcessor ip;
ip = new MonEcouteurClavier (…));

GestureDetector gd;
gd = new GestureDetector (new MonEcouteurGeste (…));

InputMultiplexer im = new InputMultiplexer();

im.addProcessor(ip);
im.addProcessor(gd);

Gdx.input.setInputProcessor(im);
```

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

Les entrées : l'accéléromètre

- Accéléromètre = mesure de l'accélération sur trois axes.
- L'accélération est mesurée en mètre par seconde au carré.
- Les valeurs se situe dans l'intervalle 10m/s^2 et -10m/s^2



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

Les entrées : l'accéléromètre

- Vérifier la disponibilité de l'accéléromètre

```
boolean disponible;
disponible = Gdx.input.isPeripheralAvailable(Peripheral.Accelerometer);
```

- Android : activer l'accéléromètre

```
public class MainActivity extends AndroidApplication {
  @Override
  public void onCreate(Bundle savedInstanceState) {
    …
    cfg.useAccelerometer = true;
    …
  }
  …
}
```

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

Les entrées : l'accéléromètre

- Orientation native

```
Orientation nativeOrientation = Gdx.input.getNativeOrientation();
```

- Orientation par rapport à l'orientation native
  - 0, 90, 180 ou 270

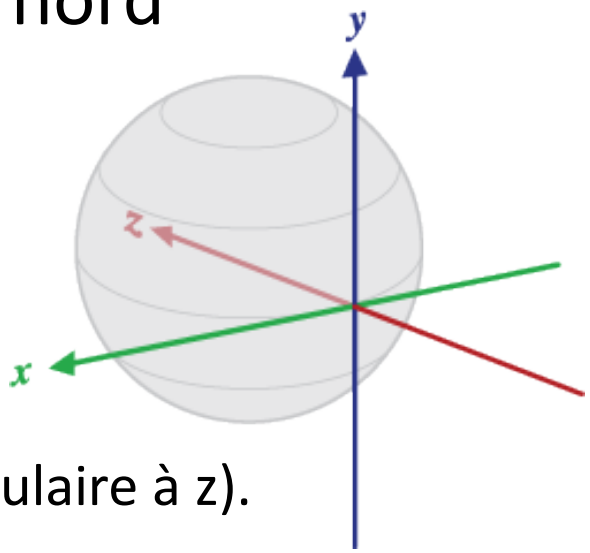
```
int orientation = Gdx.input.getOrientation();
```

- Lecture de l'accélération
  - Polling

```
float accX,accY,accZ;
accX = Gdx.input.getAccelerometerX();
accY = Gdx.input.getAccelerometerY();
accZ = Gdx.input.getAccelerometerZ();
```

Les entrées : le compas

- Compas = capteur magnétique qui mesure l'orientation par rapport au pôle nord
- Angles = degrés
  - Azimuth : angle autour de z
    - z pointe vers le centre de la terre.
  - Pitch : angle autour de x
    - x pointe vers l'ouest et est perpendiculaire à z).
  - Roll : angle autour de y
    - y pointe vers le pôle nord et est orthogonal aux deux autres.



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

Les entrées : le compas

- Vérifier la disponibilité de l'accéléromètre

```
boolean compassAvail;
compassAvail = Gdx.input.isPeripheralAvailable(Peripheral.Compass);
```

- Lecture de l'orientation
  - Polling

```
float azimuth = Gdx.input.getAzimuth();
float pitch = Gdx.input.getPitch();
float roll = Gdx.input.getRoll();
```