LibGDX

Java Game Library

Randy J. Fortier
randy.fortier@uoit.ca
@randy_fortier

Outline

- LibGDX game library
  - Graphics
  - Physics
  - Controls
  - Audio
  - AI

LibGDX

Overview

LibGDX

- Multi-platform:
  - Windows, Mac OS-X, Linux
  - Android
  - iOS
  - Web (HTML5)

LibGDX

- Graphics: Built on OpenGL
- Physics: Box2D
- Controls: Custom
- Audio: Custom
- AI: Custom

LibGDX

Graphics

LibGDX

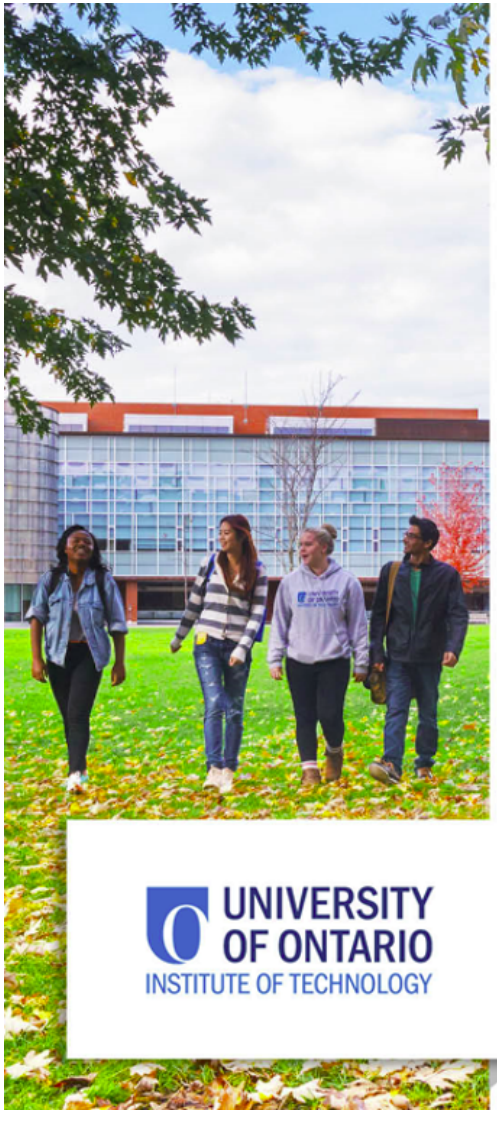
- Textures
  - Loading of image formats
  - Scaling, rotation, translation, flipping

```
Texture playerTexture = new Texture("images/hero.png");
SpriteBatch batch = new SpriteBatch();
float x = 100;
float y = 100;
batch.begin();
batch.draw(playerTexture, x, y);
batch.end();
```

LibGDX

- Textures
  - Drawing only part of an image

```
Texture spriteSheetTexture = new Texture("images/sprite_sheet.png");
SpriteBatch batch = new SpriteBatch();
float x = 100, y = 100;
float centreX = 50, centreY = 50;
float width = 100, height = 100;
float scaleX = 1.0f, scaleY = 1.0f;
float rotation = 1.0f;
float srcX = 0, srcY = 0;
float srcWidth = 100, srcHeight = 100;
boolean flipX = false, flipY = false;
batch.begin();
batch.draw(spriteSheetTexture, x, y, centreX, centreY, width, height
           scaleX, scaleY, rotation,
           srcX, srcY, srcWidth, srcHeight, flipX, false);
batch.end();
```



LibGDX

Physics

LibGDX

- Box2D
  - Linear acceleration and velocity
  - Angular acceleration and velocity
  - Gravity and mass
  - Collisions and transfer of momentum

LibGDX

- World
  - Manages all of the bodies
- Bodies
  - Location
  - Density (size + density -> mass)
  - Shape representation (e.g. box)
  - Dynamic (does it move?)
  - Fixed rotation (does it rotate?)

LibGDX

- World

```
World world = new World(new Vector2(0, -9.8f), false);
```

LibGDX

- Bodies

```
Body physicsBody;

BodyDef bodyDefinition = new BodyDef();
bodyDefinition.type = BodyDef.BodyType.DynamicBody;
bodyDefinition.position.set(x, y);
bodyDefinition.fixedRotation = true;
physicsBody = world.createBody(bodyDefinition);

PolygonShape shape = new PolygonShape();
shape.setAsBox(width, height);
Fixture fixture = physicsBody.createFixture(shape, 1f);
shape.dispose();
```

LibGDX

Controls

LibGDX

- Keyboard

```
KeyboardHandler keyboardHandler = new KeyboardHandler(this);
Gdx.input.setInputProcessor(keyboardHandler);
...
public class KeyboardHandler implements InputProcessor {
    @Override
    public boolean keyDown(int keycode) {
        if (keycode == Input.Keys.LEFT) {
            game.moveLeft();
        } else if (keycode == Input.Keys.RIGHT) {
            game.moveRight();
        }
        ...
    }
    ...
}
```

LibGDX

- Touch screen

```
TouchHandler touchHandler = new TouchHandler(this);
Gdx.input.setInputProcessor(touchHandler);
...
public class TouchHandler implements InputProcessor {
    @Override
    public boolean touchDown(int screenX, int screenY, int pointer,
        game.jump();
        ...
    }
    ...
}
```

LibGDX

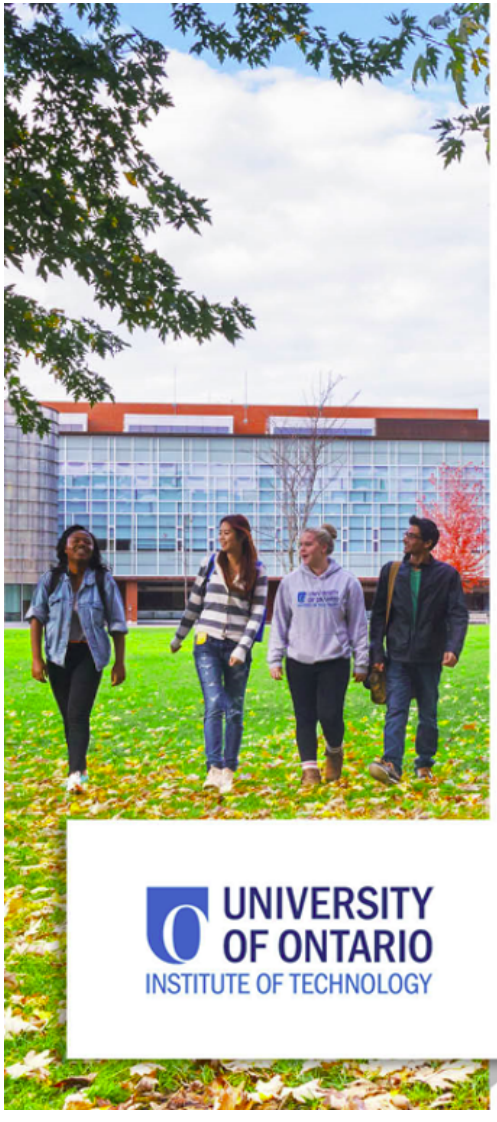
- Gamepad/controllers

```
ControllerHandler controllerHandler = new ControllerHandler(this);
Controllers.addListener(controllerHandler);
...
public class ControllerHandler implements ControllerListener {
    @Override
    public boolean buttonDown(Controller controller, int buttonCode)
        if (buttonCode == 0) {
            game.jump();
        }
        ...
    }
    ...
}
```

LibGDX

- Accelerometer (for mobile)

```
if (Gdx.input.isPeripheralAvailable(Input.Peripheral.Accelerometer))
    float accelY = Gdx.input.getAccelerometerY();
    System.out.println("accelY = " + accelY);
    if (accelY > 1f) {
        moveRight();
    } else if (accelY < -1f){
        moveLeft();
    }
}
```



LibGDX

Audio

LibGDX

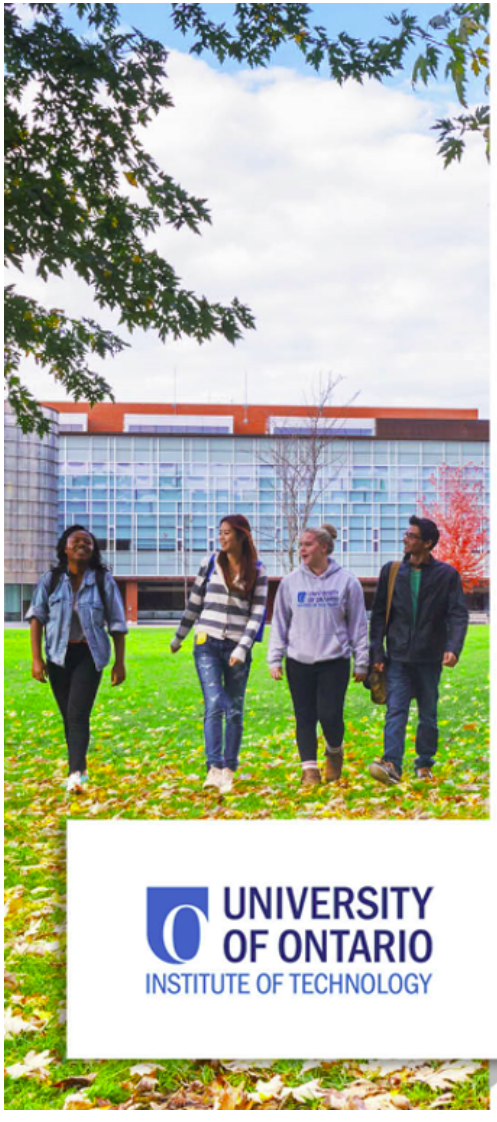
- Sounds
  - Many media types supported (e.g. .wav, .mp3, .ogg)

```
Sound sound = Gdx.audio.newSound(Gdx.files.internal("effect.wav"));
...
sound.play();
```

LibGDX

- Music
  - Buffered for larger audio streams

```
Music music = Gdx.audio.newMusic(Gdx.files.internal("music.mp3"));
music.setLooping(true);
music.play();
```



LibGDX

Artificial Intelligence

LibGDX

- Steering behaviour
  - Seek - attempts to crash into a target
  - Flee - moves away from a target
  - Pursue - attempts to intercept a target
  - Evade - attemps to avoid interception
  - Arrive - attempts to stop at a target
  - Face - attempts to face toward a target
  - Wander - random motion
  - FollowPath - follow a prescribed path
  - Interpose - predict future position and aim to intersect with this path
- Formation motion

LibGDX

- Steerable

```
public class FollowEntity extends SteerableAdapter {
    public void update(float delta) {
        behaviour.calculateSteering(steeringOutput);

        if (!steeringOutput.linear.isZero()) {
            Vector2 force = steeringOutput.linear.scl(delta);
            body.applyForceToCenter(force, true);
        }
    }
    ...
}
```

LibGDX

- Arrive

```
Arrive<Vector2> arrive = new Arrive<>(followerEntity, followedEntity
arrive.setTimeToTarget(0.01f);
arrive.setArrivalTolerance(2f);
arrive.setDecelerationRadius(200f);
followerEntity.setBehaviour(arrive);
```

Wrap-Up

- In this section we learned about:
  - LibGDX
    - Graphics
    - Physics (Box2D)
    - Controls
    - Audio
    - AI