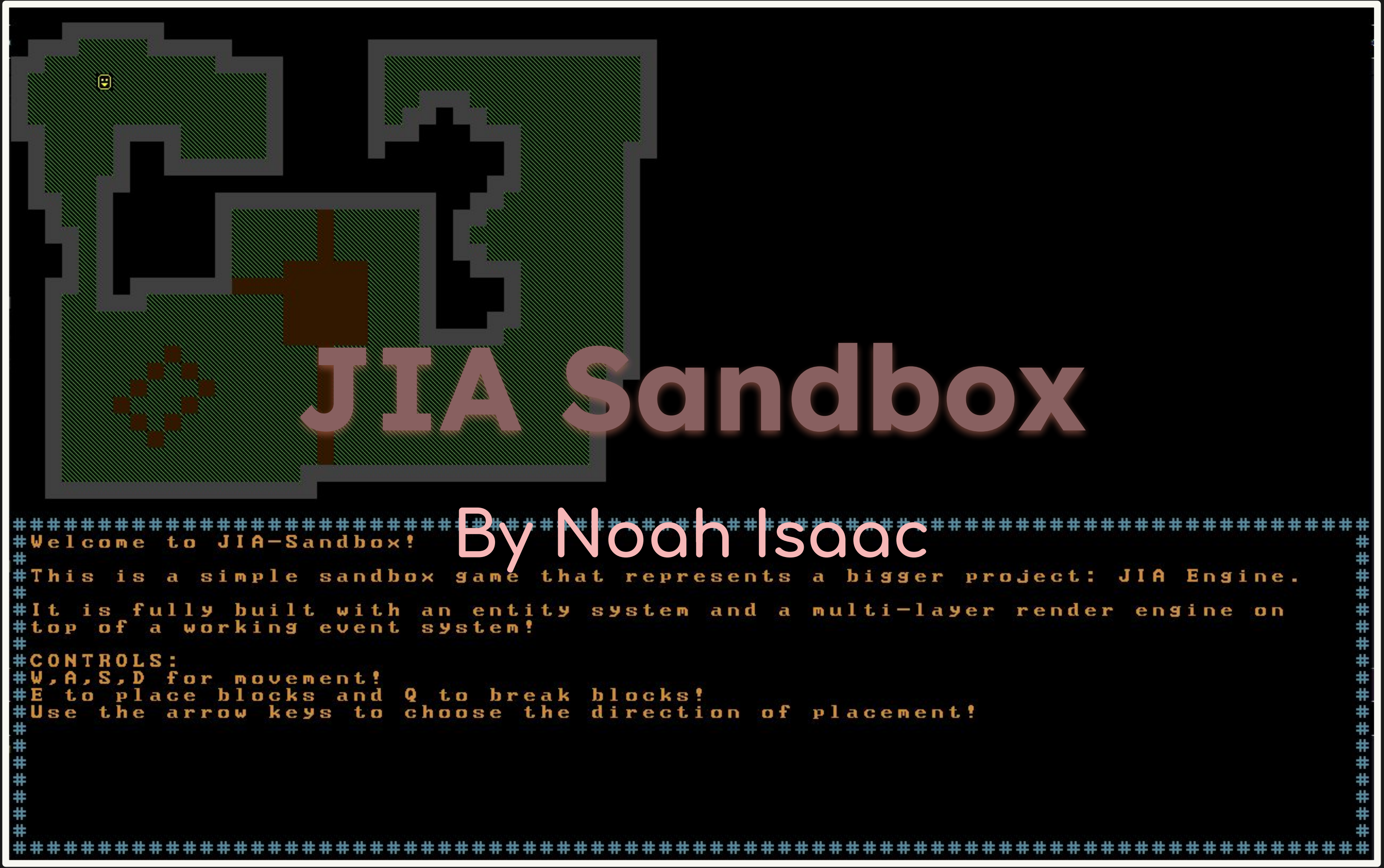
JIA Sandbox

By Noah Isaac

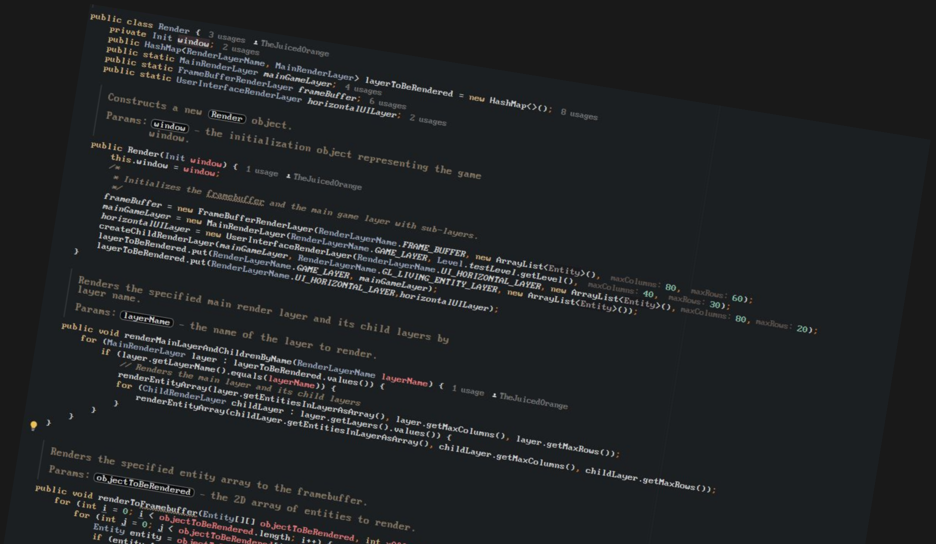


```
#Welcome to JIA-Sandbox!
#
#This is a simple sandbox game that represents a bigger project: JIA Engine.
#
#It is fully built with an entity system and a multi-layer render engine on
#top of a working event system!
#
#CONTROLS:
#W,A,S,D for movement!
#E to place blocks and Q to break blocks!
#Use the arrow keys to choose the direction of placement!
```

Project Overview and Use Case

- JAI is an ASCII based game engine built from scratch written in Java using JFrame as its window manager.
- Built with the idea of flexibility through polymorphism and inheritance in mind to allow for an expandable base for future projects.

OVER 8964 LINES OF CODE!



```
public class Render {  3 usages  TheJuicedOrange
    private Init window;  2 usages
    public HashMap<RenderLayerName, MainRenderLayer> layerToBeRendered = new HashMap<>();  8 usages
    public static MainRenderLayer mainGameLayer;  4 usages
    public static FrameBufferRenderLayer frameBuffer;  6 usages
    public static UserInterfaceRenderLayer horizontalUILayer;  2 usages
    Constructs a new Render object.
    Params: window - the initialization object representing the game
            window.
public Render(Init window) {  1 usage  TheJuicedOrange
    this.window = window;
    /*
     * Initializes the framebuffer and the main game layer with sub-layers.
     */
    frameBuffer = new FrameBufferRenderLayer(RenderLayerName.FRAME_BUFFER, new ArrayList<Entity>(),  maxColumns: 80,  maxRows: 60);
    mainGameLayer = new MainRenderLayer(RenderLayerName.GAME_LAYER, Level.testLevel.getLevel(),  maxColumns: 40,  maxRows: 30);
    horizontalUILayer = new UserInterfaceRenderLayer(RenderLayerName.UI_HORIZONTAL_LAYER, new ArrayList<Entity>(), maxColumns: 80, maxRows: 20);
    createChildRenderLayer(mainGameLayer, RenderLayerName.GL_LIVING_ENTITY_LAYER, new ArrayList<Entity>());
    layerToBeRendered.put(RenderLayerName.GAME_LAYER, mainGameLayer);
    layerToBeRendered.put(RenderLayerName.UI_HORIZONTAL_LAYER,horizontalUILayer);
}

    Renders the specified main render layer and its child layers by
    layer name.
    Params: layerName - the name of the layer to render.
public void renderMainLayerAndChildrenByName(RenderLayerName layerName) {  1 usage  TheJuicedOrange
    for (MainRenderLayer layer : layerToBeRendered.values()) {
        if (layer.getLayerName().equals(layerName)) {
            // Renders the main layer and its child layers
            renderEntityArray(layer.getEntitiesInLayerAsArray(), layer.getMaxColumns(), layer.getMaxRows());
            for (ChildRenderLayer childLayer : layer.getLayers().values()) {
                renderEntityArray(childLayer.getEntitiesInLayerAsArray(), childLayer.getMaxColumns(), childLayer.getMaxRows());
            }
        }
    }
}

    Renders the specified entity array to the framebuffer.
    Params: objectToBeRendered - the 2D array of entities to render.
public void renderToFramebuffer(Entity[][] objectToBeRendered, int
    for (int i = 0; i < objectToBeRendered.length; i++) {
        for (int j = 0; j < objectToBeRendered
            Entity entity = ob
            if (entit
```

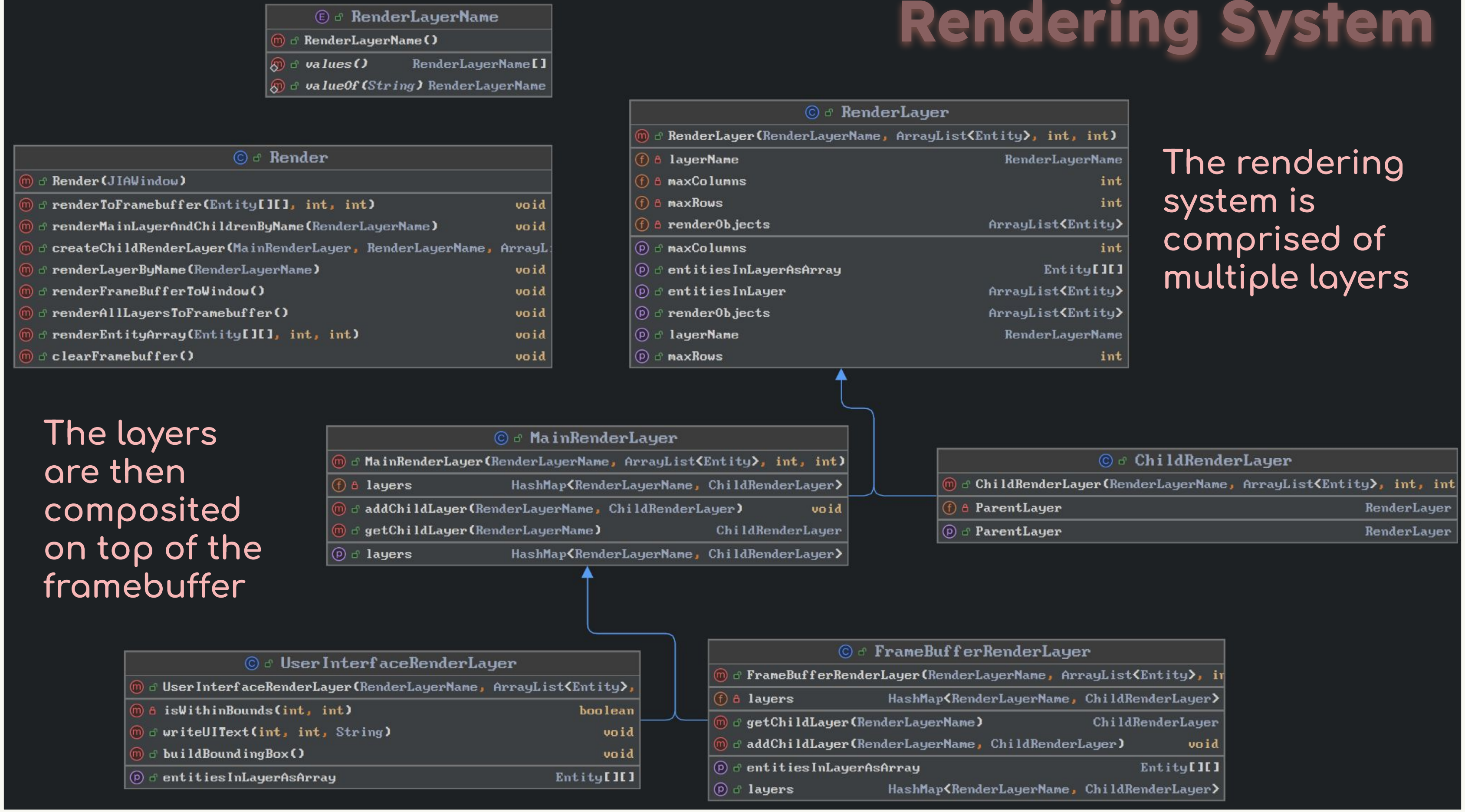
Goals and Struggles

- Goals
  - Written in java
  - Proper structure and flexible design using object oriented programming
  - Featuring a terrible Retro/ASCII style!
  - A new and fancy event system
- Struggles:
  - Rendering arrays decided to erase themselves through side effects
  - Fixing the jank entity and array correlation problem
  - The event system frying itself like an egg after failing to subscribe correctly
  - Framebuffer versues immediate rendering!

Rendering System

The rendering system is comprised of multiple layers

The layers are then composited on top of the framebuffer



```
RenderLayerName
RenderLayerName()
values()                    RenderLayerName[]
valueOf(String)             RenderLayerName
```

```
Render
Render(JIAWindow)
renderToFramebuffer(Entity[][], int, int)                                void
renderMainLayerAndChildrenByName(RenderLayerName)                        void
createChildRenderLayer(MainRenderLayer, RenderLayerName, ArrayL
renderLayerByName(RenderLayerName)                                       void
renderFrameBufferToWindow()                                              void
renderAllLayersToFramebuffer()                                           void
renderEntityArray(Entity[][], int, int)                                  void
clearFramebuffer()                                                       void
```

```
RenderLayer
RenderLayer(RenderLayerName, ArrayList<Entity>, int, int)
layerName                                         RenderLayerName
maxColumns                                        int
maxRows                                           int
renderObjects                                     ArrayList<Entity>
maxColumns                                        int
entitiesInLayerAsArray                            Entity[][]
entitiesInLayer                                   ArrayList<Entity>
renderObjects                                     ArrayList<Entity>
layerName                                         RenderLayerName
maxRows                                           int
```

```
MainRenderLayer
MainRenderLayer(RenderLayerName, ArrayList<Entity>, int, int)
layers                              HashMap<RenderLayerName, ChildRenderLayer>
addChildLayer(RenderLayerName, ChildRenderLayer)                        void
getChildLayer(RenderLayerName)                              ChildRenderLayer
layers                              HashMap<RenderLayerName, ChildRenderLayer>
```

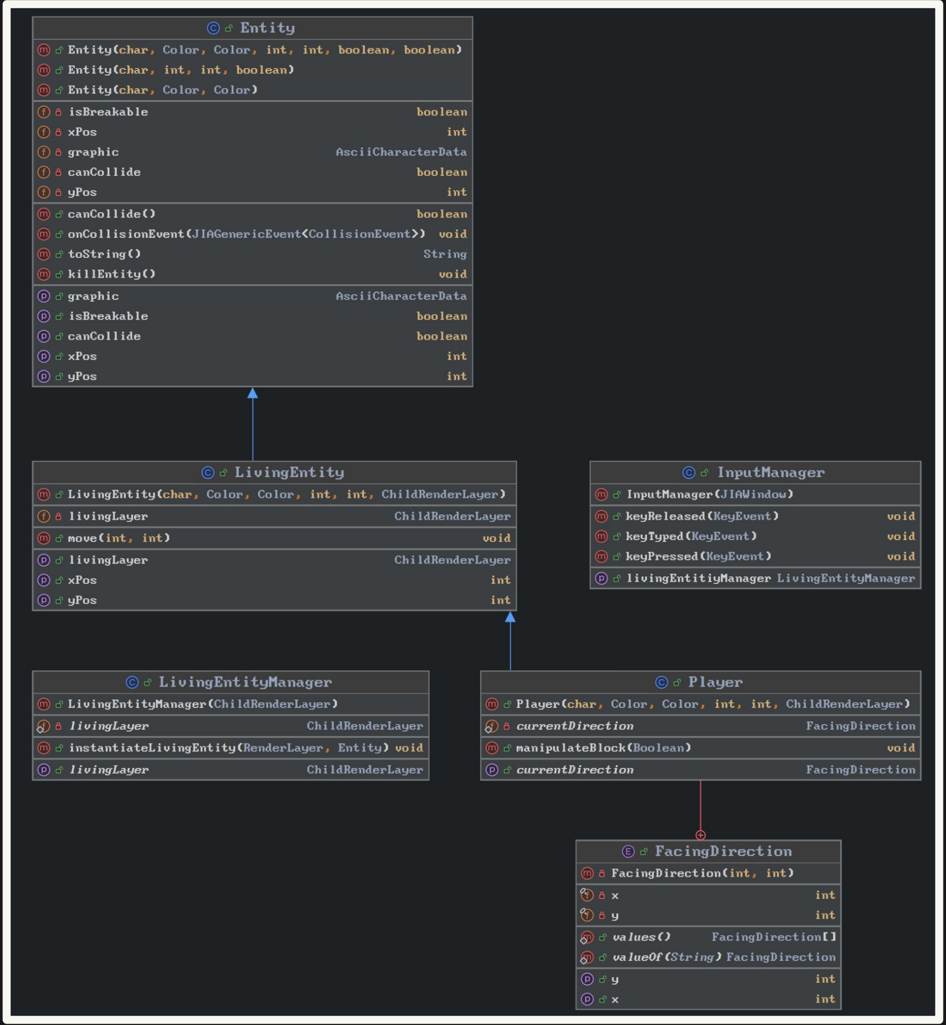
```
ChildRenderLayer
ChildRenderLayer(RenderLayerName, ArrayList<Entity>, int, int
ParentLayer                                                  RenderLayer
ParentLayer                                                  RenderLayer
```

```
UserInterfaceRenderLayer
UserInterfaceRenderLayer(RenderLayerName, ArrayList<Entity>,
isWithinBounds(int, int)                                     boolean
writeUIText(int, int, String)                                void
buildBoundingBox()                                           void
entitiesInLayerAsArray                                       Entity[][]
```

```
FrameBufferRenderLayer
FrameBufferRenderLayer(RenderLayerName, ArrayList<Entity>, in
layers                              HashMap<RenderLayerName, ChildRenderLayer>
getChildLayer(RenderLayerName)                              ChildRenderLayer
addChildLayer(RenderLayerName, ChildRenderLayer)                        void
entitiesInLayerAsArray                                            Entity[][]
layers                              HashMap<RenderLayerName, ChildRenderLayer>
```

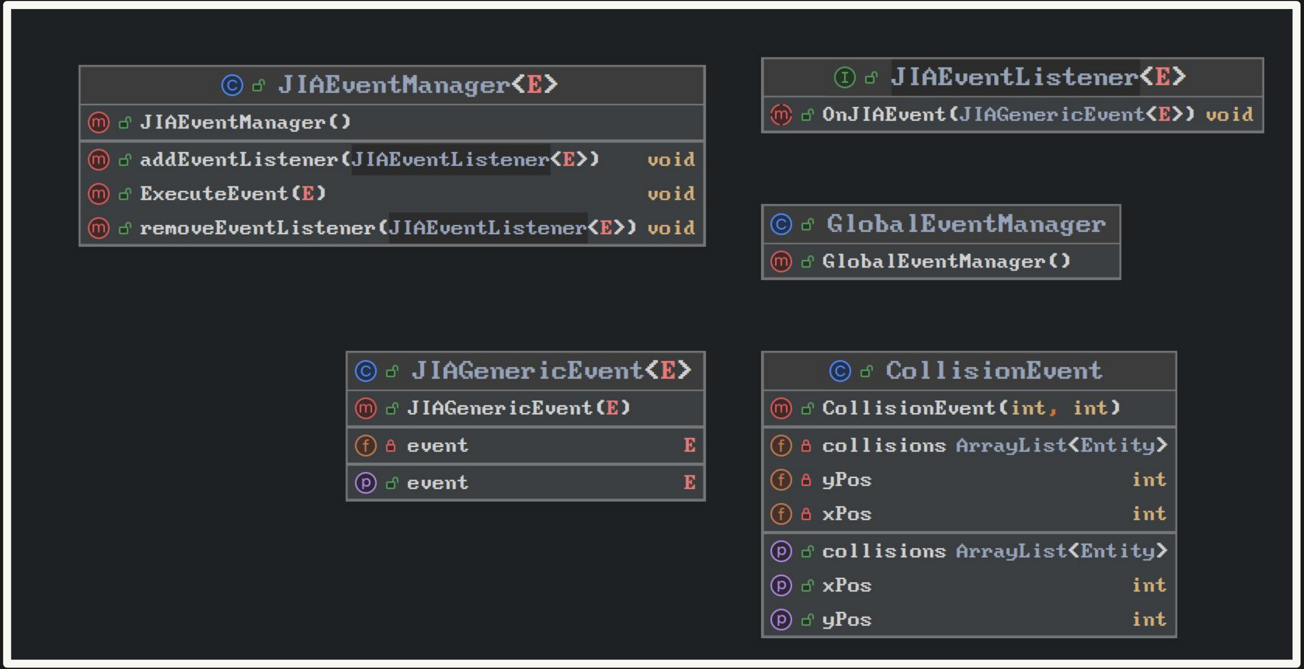
Entity System

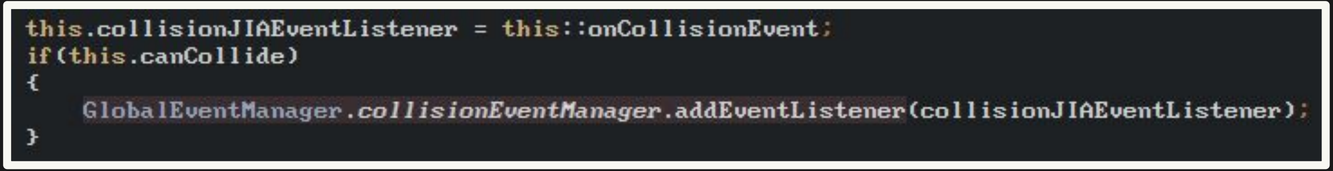
Everything rendered on screen is an entity. They contain a wide variety of functionality from collision to color selection.


Entity
Entity(char, Color, Color, int, int, boolean, boolean)
Entity(char, int, int, boolean)
Entity(char, Color, Color)
isBreakable boolean
xPos int
graphic AsciiCharacterData
canCollide boolean
yPos int
canCollide() boolean
onCollisionEvent(JIAGenericEvent<CollisionEvent>) void
toString() String
killEntity() void
graphic AsciiCharacterData
isBreakable boolean
canCollide boolean
xPos int
yPos int
LivingEntity
LivingEntity(char, Color, Color, int, int, ChildRenderLayer)
livingLayer ChildRenderLayer
move(int, int) void
livingLayer ChildRenderLayer
xPos int
yPos int
InputManager
InputManager(JIAWindow)
keyReleased(KeyEvent) void
keyTyped(KeyEvent) void
keyPressed(KeyEvent) void
livingEntitiyManager LivingEntityManager
LivingEntityManager
LivingEntityManager(ChildRenderLayer)
livingLayer ChildRenderLayer
instantiateLivingEntity(RenderLayer, Entity) void
livingLayer ChildRenderLayer
Player
Player(char, Color, Color, int, int, ChildRenderLayer)
currentDirection FacingDirection
manipulateBlock(Boolean) void
currentDirection FacingDirection
FacingDirection
FacingDirection(int, int)
x int
y int
values() FacingDirection[]
valueOf(String) FacingDirection
y int
x int

Event System

- JIAEventListener is a functional Interface which defines the signature of an event callback
- Anything can register a callback to an event by passing a method reference to a JIAEventManager
- In the code snippet below we are subscribing an entity to the collision event manager!
- JIAGenericEvent is <u>GENERIC</u>!!!! It holds data passed to the callback.

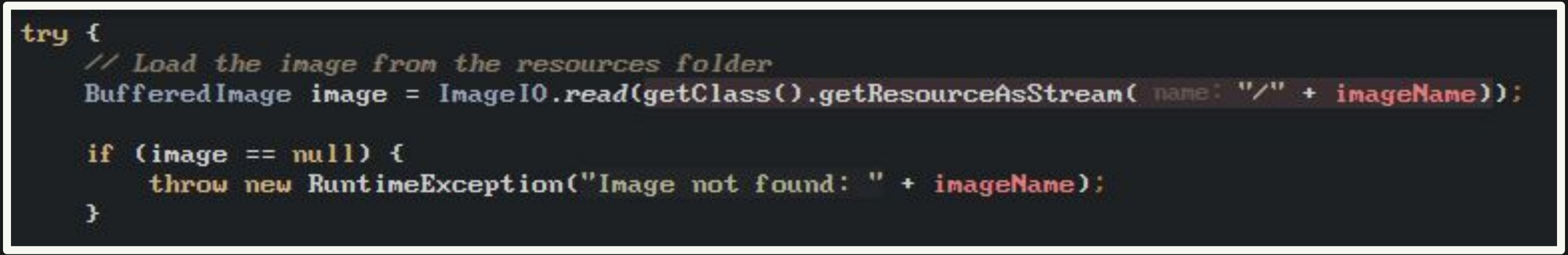




```
this.collisionJIAEventListener = this::onCollisionEvent;
if(this.canCollide)
{
    GlobalEventManager.collisionEventManager.addEventListener(collisionJIAEventListener);
}
```

Error Handling

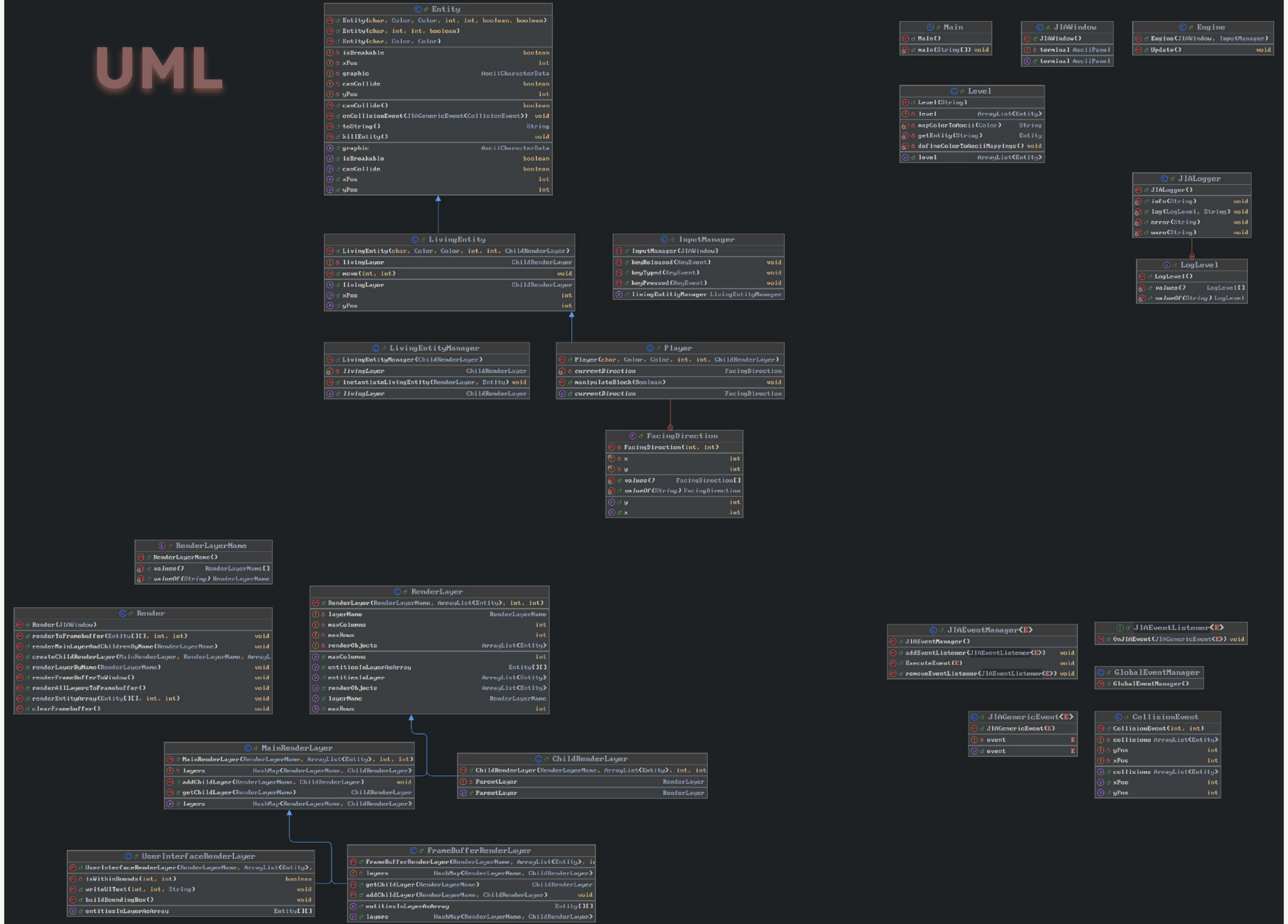
- Standard, best practice use of null checks and try/catch blocks!
- Jia also has a custom logger built for debugging and custom insight into the program!



```java
try {
    // Load the image from the resources folder
    BufferedImage image = ImageIO.read(getClass().getResourceAsStream( name: "/" + imageName));

    if (image == null) {
        throw new RuntimeException("Image not found: " + imageName);
    }
```

UML


Entity
Entity(char, Color, Color, int, int, boolean, boolean)
Entity(char, int, int, boolean)
Entity(char, Color, Color)
isBreakable boolean
xPos int
graphic AsciiCharacterData
canCollide boolean
yPos int
canCollide() boolean
onCollisionEvent(JIAGenericEvent<CollisionEvent>) void
toString() String
killEntity() void
graphic AsciiCharacterData
isBreakable boolean
canCollide boolean
xPos int
yPos int
LivingEntity
LivingEntity(char, Color, Color, int, int, ChildRenderLayer)
livingLayer ChildRenderLayer
move(int, int) void
livingLayer ChildRenderLayer
xPos int
yPos int
InputManager
InputManager(JIAWindow)
keyReleased(KeyEvent) void
keyTyped(KeyEvent) void
keyPressed(KeyEvent) void
livingEntitiyManager LivingEntityManager
LivingEntityManager
LivingEntityManager(ChildRenderLayer)
livingLayer ChildRenderLayer
instantiateLivingEntity(RenderLayer, Entity) void
livingLayer ChildRenderLayer
Player
Player(char, Color, Color, int, int, ChildRenderLayer)
currentDirection FacingDirection
manipulateBlock(Boolean) void
currentDirection FacingDirection
FacingDirection
FacingDirection(int, int)
x int
y int
values() FacingDirection[]
valueOf(String) FacingDirection
y int
x int
Main
Main()
main(String[]) void
JIAWindow
JIAWindow()
terminal AsciiPanel
terminal AsciiPanel
Engine
Engine(JIAWindow, InputManager)
Update() void
Level
Level(String)
level ArrayList<Entity>
mapColorToAscii(Color) String
getEntity(String) Entity
defineColorToAsciiMappings() void
level ArrayList<Entity>
JIALogger
JIALogger()
info(String) void
log(LogLevel, String) void
error(String) void
warn(String) void
LogLevel
LogLevel()
values() LogLevel[]
valueOf(String) LogLevel
RenderLayerName
RenderLayerName()
values() RenderLayerName[]
valueOf(String) RenderLayerName
Render
Render(JIAWindow)
renderToFramebuffer(Entity[][], int, int) void
renderMainLayerAndChildrenByName(RenderLayerName) void
createChildRenderLayer(MainRenderLayer, RenderLayerName, ArrayL
renderLayerByName(RenderLayerName) void
renderFrameBufferToWindow() void
renderAllLayersToFramebuffer() void
renderEntityArray(Entity[][], int, int) void
clearFramebuffer() void
RenderLayer
RenderLayer(RenderLayerName, ArrayList<Entity>, int, int)
layerName RenderLayerName
maxColumns int
maxRows int
renderObjects ArrayList<Entity>
maxColumns int
entitiesInLayerAsArray Entity[][]
entitiesInLayer ArrayList<Entity>
renderObjects ArrayList<Entity>
layerName RenderLayerName
maxRows int
MainRenderLayer
MainRenderLayer(RenderLayerName, ArrayList<Entity>, int, int)
layers HashMap<RenderLayerName, ChildRenderLayer>
addChildLayer(RenderLayerName, ChildRenderLayer) void
getChildLayer(RenderLayerName) ChildRenderLayer
layers HashMap<RenderLayerName, ChildRenderLayer>
ChildRenderLayer
ChildRenderLayer(RenderLayerName, ArrayList<Entity>, int, int
ParentLayer RenderLayer
ParentLayer RenderLayer
UserInterfaceRenderLayer
UserInterfaceRenderLayer(RenderLayerName, ArrayList<Entity>,
isWithinBounds(int, int) boolean
writeUIText(int, int, String) void
buildBoundingBox() void
entitiesInLayerAsArray Entity[][]
FrameBufferRenderLayer
FrameBufferRenderLayer(RenderLayerName, ArrayList<Entity>, i
layers HashMap<RenderLayerName, ChildRenderLayer>
getChildLayer(RenderLayerName) ChildRenderLayer
addChildLayer(RenderLayerName, ChildRenderLayer) void
entitiesInLayerAsArray Entity[][]
layers HashMap<RenderLayerName, ChildRenderLayer>
JIAEventManager<E>
JIAEventManager()
addEventListener(JIAEventListener<E>) void
ExecuteEvent(E) void
removeEventListener(JIAEventListener<E>) void
JIAEventListener<E>
OnJIAEvent(JIAGenericEvent<E>) void
GlobalEventManager
GlobalEventManager()
JIAGenericEvent<E>
JIAGenericEvent(E)
event E
event E
CollisionEvent
CollisionEvent(int, int)
collisions ArrayList<Entity>
yPos int
xPos int
collisions ArrayList<Entity>
xPos int
yPos int

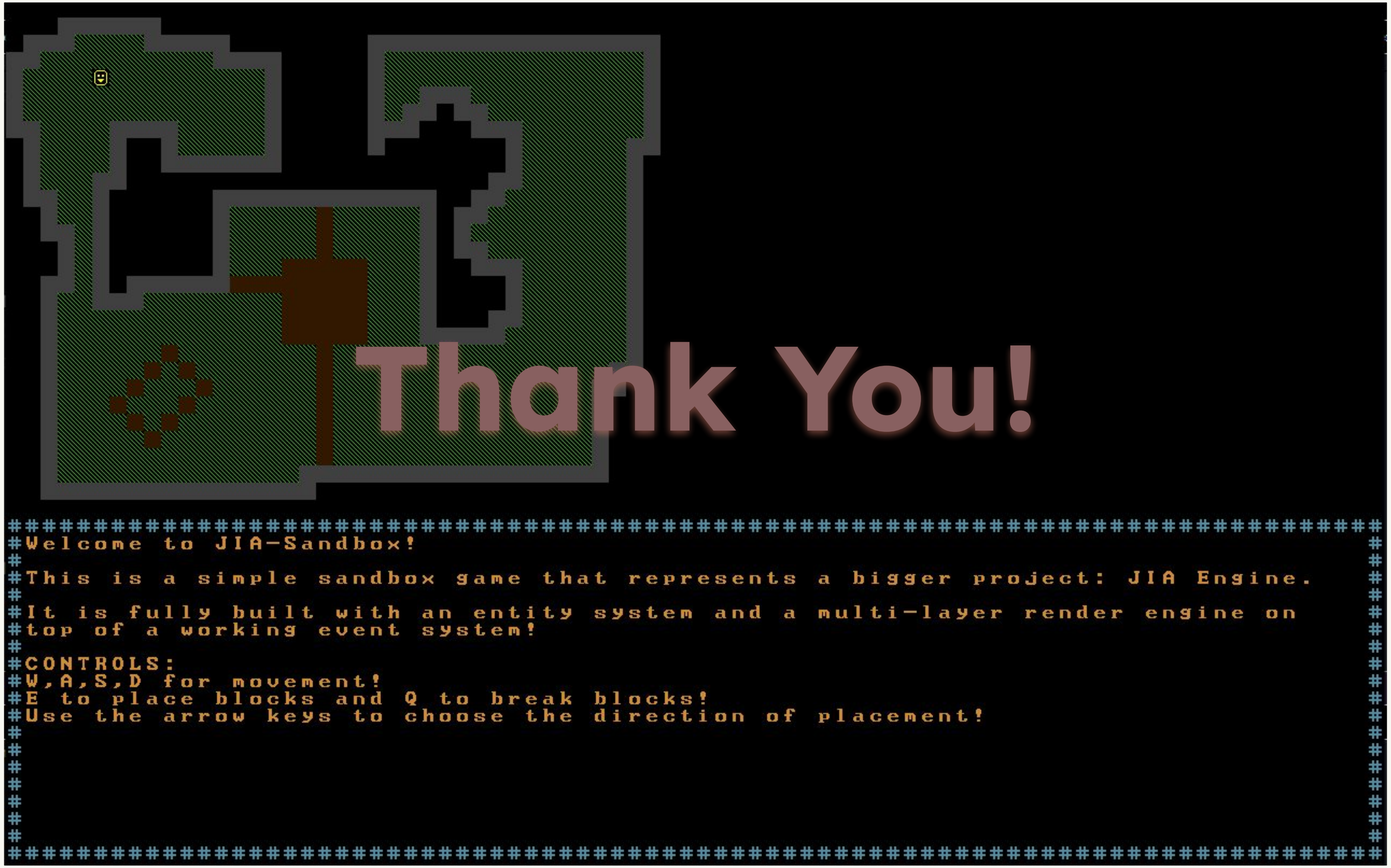
How to Use And Run

Controls:
W,A,S,D to move
Q and E to place and break blocks
And optionally use arrow keys for facing direction

How to run:

```
./gradlew build
java -jar ./build/libs/JIA-1.0-SNAPSHOT.jar
```

Or load with your Gradle-Integrated IDE of choice (such as IntelliJ) and click Run!



Thank You!

```
################################################################################
#Welcome to JIA-Sandbox!                                                      #
#                                                                              #
#This is a simple sandbox game that represents a bigger project: JIA Engine.  #
#                                                                              #
#It is fully built with an entity system and a multi-layer render engine on   #
#top of a working event system!                                                #
#                                                                              #
#CONTROLS:                                                                     #
#W,A,S,D for movement!                                                         #
#E to place blocks and Q to break blocks!                                      #
#Use the arrow keys to choose the direction of placement!                      #
#                                                                              #
#                                                                              #
#                                                                              #
#                                                                              #
#                                                                              #
#                                                                              #
################################################################################
```