Key Concepts

Game Entities

Application Listener

Game

Screen

Stages

Actors

Game Entities

Libgdx has several game entities that can be utilized to create a very **well organized** game scene

Application Listener

ApplicationListener is the root of the **Game** class, which implements the LibGdx life cycle methods **create**, **resize**, **render**, **pause**, **resume**

Game class

Game classes can be used as the **brain** of your game, it contains an **active screen** object that can be set to point to the screen you want to **show**

Game class

void setScreen(**Screen** s) 1

Use this method to set the **active** screen in your Game

Screen

Screen objects are an **interface** that can be implented by the to display a screen in your game

class Main **implements** Screen

Screen

For example, a screen object can be used to make a **menu** screen, **level 1** screen, **level 2** screen **pause** screen, **exit** screen, etc

menu | **level1** | **level2** | **pause**

Screen

Once a **screen** is created, it can be **set** to the game's main screen

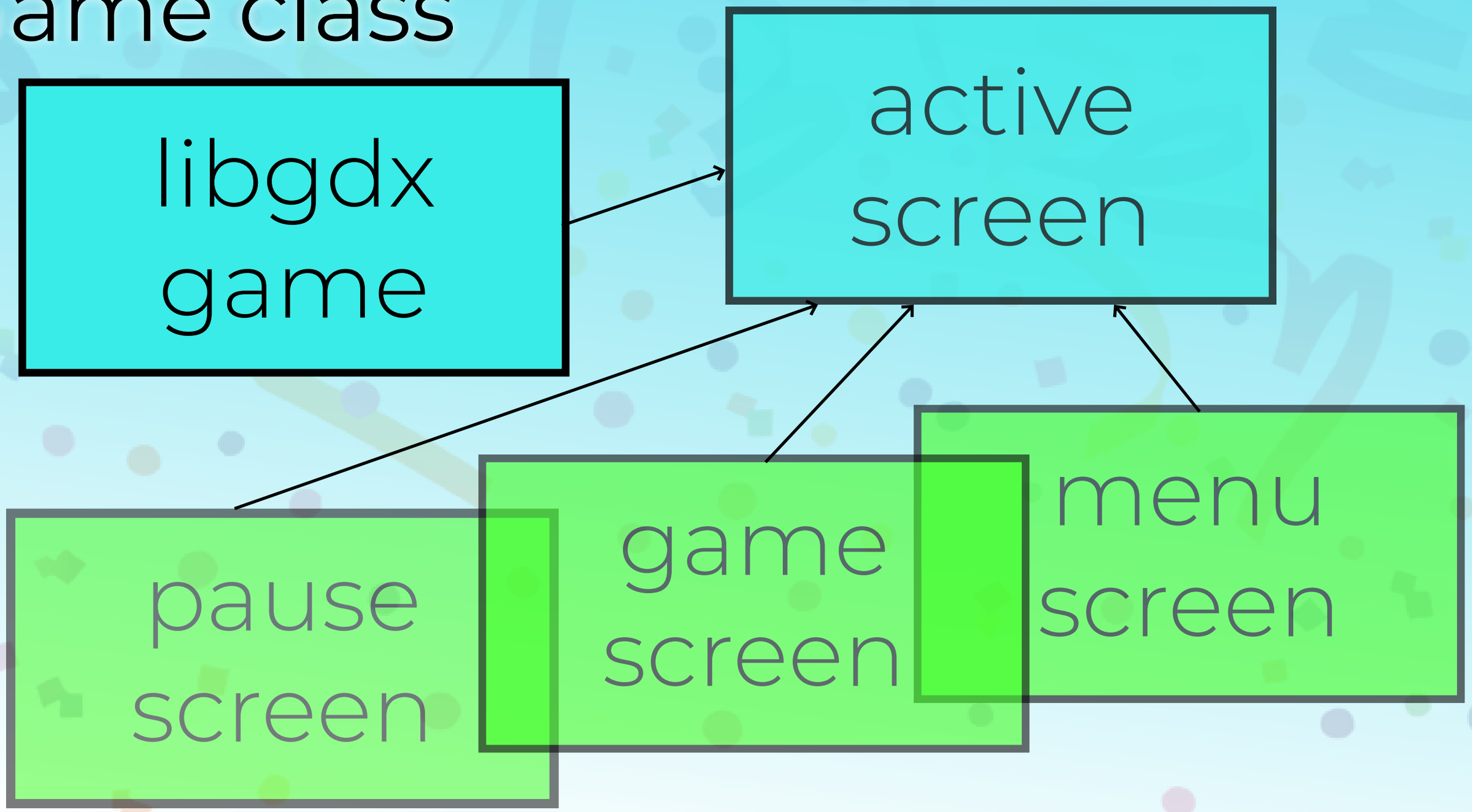
```
if (gameStarted) {
    setScreen(mainScreen);
}
```

Screen

Only **one** screen object can be **active** at a time

setScreen(**gameScreen**);

Game class


libgdx
game
active
screen
pause
screen
game
screen
menu
screen

Screen

The **show()** method is called when this screen is the **current active screen** in the game

The **hide()** method is called when this screen is **hidden**

Stages

A stage is a scene graph containing a **hierarchy** of actors

You can **add** or **remove** actors from the Stage

Stages

A stage object has a built-in **camera**, **SpriteBatch**, and **root node** used for handling **drawing** of actors **added**

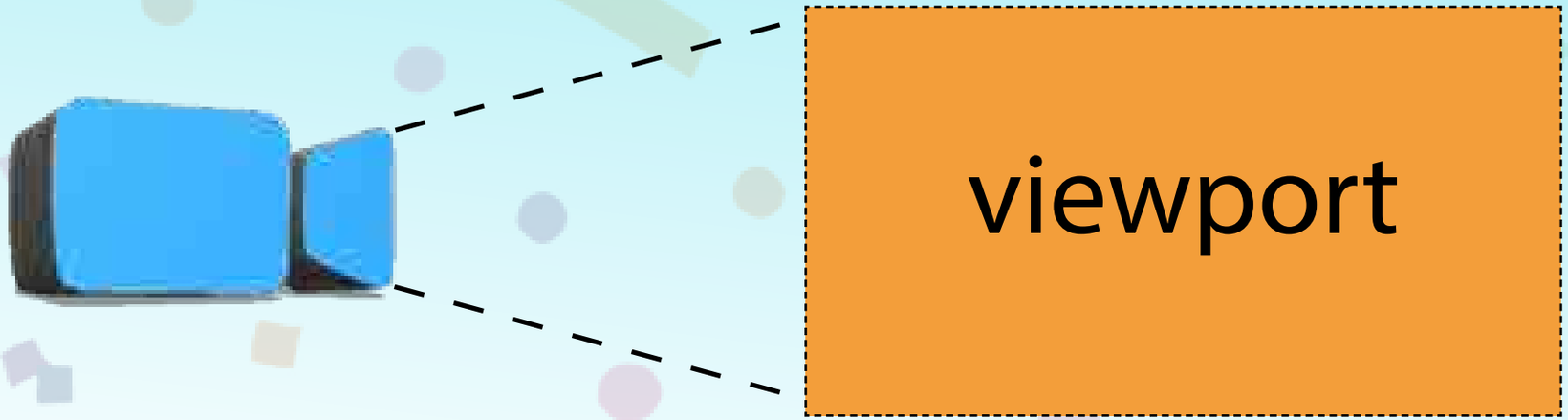
Add **actors** to the root

Stages

Stages handles **input events**, and hold **coordinates** as to where actors are placed within the scene

Camera

The camera can be **accessed** and **modified** through **stage** objects



Actors

An actor is a 2D scene **graph node**

It has a **position**, **orig**

https://images-na.ssl-images-amazon.

Assets Folder

Saves **images** and **audio** files into the assets folder to help create new actors

Actors

Actors can be placed **individually** onto a stage, or together on the stage as a **group** or **party** or **elements**

Actors

Actors can be drawn using the **draw()** method

Pass the **SpriteBatch** and **alpha** values to this function

Actors

Actors can be updated based on time using the **act()** method

Pass the **delta time value**

Groups

Group actors together using a **Group** node

Bundling actors and calling functions will affect every actor within this group

Groups

```
group1 = new Group();


group1.addActor(actor);
group1.act(dt);
```

Example Game

Actor

Actor

screen

Camera

Example Game

In this example, the screen has a **stage** holding **actors**, that are both in groups

The **camera** object belongs to the **stage**