SENG 401 - Presentation 3

Group 11

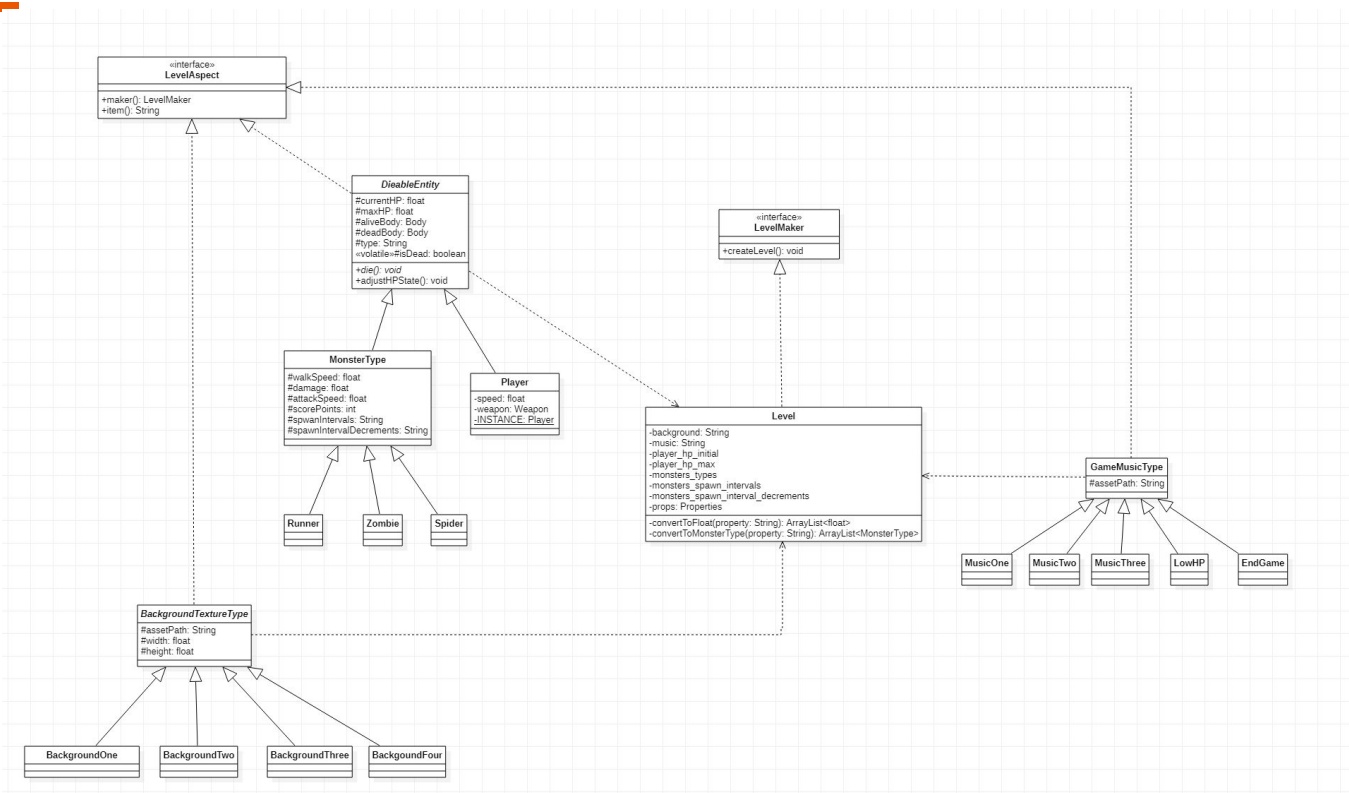
Jay Chuang, Duan Le, Youssef Maksoud, Hunter Coles, Vu Ha(Martin), Ayush Chaudhari, Cloud Chagnon

EverGame - Non-ML

EverGame Design Patterns

- Builder Pattern
  - Justification: We want to separate the construction of the Level object from its representation
  - Solution: Level builder handles construction of all subcomponents (BackgroundTextureType, DieableEntities, GameMusicType) of a level
- Singleton Pattern
  - Justification: The MainScene class is only created once through the lifetime of the game
  - Solution: MainScene becomes Singleton and all its variables static.

Builder Pattern Class Diagram



Builder Pattern Code

```java
public interface LevelAspect{
    public LevelMaker maker();
    public String item();
}

public interface LevelMaker{
    public void create_level();
}
```

```java
public class Level implements LevelMaker{
    private String background;
    private String music;
    private String player_hp_initial;
    private String player_hp_max;
    private String monster_types;
    private String monsters_spawn_intervals;
    private String monsters_spawn_interval_decrements;

    private Properties props;
    private DieableEntity de;
    private BackgroundTextureType bt;
    private GameMusicType gm;

    public Level()
    {
        background = "default_background";
        music = "default_music";
    }

    public void setMusic(String s)
    {
        music = s;
    }

    public void setBackgroundTextureType(String s)
    {
        background = s;
    }

    public void setMonsterTypes(String s)
    {
        monster_types = s;
    }
```

```java
public void setSpawnInterval(String s)
{
    monsters_spawn_intervals = s;
}

public void setSpawnIntervalDec(String s)
{
    spawnIntervalDecrements = s;
}

public void setPlayerHpInitial(String s)
{
    player_hp_initial = s;
}

public void setPlayerHPMax(String s)
{
    player_hp_max = s;
}

@Override
public LevelAspect create_level() {
    BasicLevel level = new BasicLevel();
    level.setGameMusicType(getGameMusicType());
    level.setInitialPlayerHP(getInitialPlayerHP());
    level.setMaxPlayerHP(getMaxPlayerHP());
    de.setLevel(level)
    bt.setLevel(level);
    gm.setLevel(level);
    return level;
}
```

Builder Pattern Code

```java
public abstract class DieableEntity implements LevelAspect
{
    private volatile boolean isDead;
    protected float currentHP;
    protected float maxHP;
    protected Body aliveBody;
    protected Body deadBody;
    protected String type;
    protected Level l;

    public DieableEntity() {

    }

    public DieableEntity(boolean dead, float currentHP, float maxHP, Body aliveBody, Body deadBody, String type, Level l)
    {
        this.isDead = dead;
        this.currentHP = currentHP;
        this.maxHP = maxHP;
        this.aliveBody = aliveBody;
        this.deadBody = deadBody;
        this.type = type;
        this.l = l;
    }

    @Override
    public void create_level();

    @Override
    public String item();
```

```java
public class MonsterType extends DieableEntity {
    private float HP;
    private float walkSpeed;
    private float damage;
    private float attackSpeed;      // in seconds
    private int scorePoints;
    private String spawnIntervals;
    private String spawnIntervalDecrements;

    private MonsterType(boolean dead, float currentHP, float maxHP, Body aliveBody, Body deadBody, String type, Level l,
            float hP, float walkSpeed, float damage,
            float attackSpeed, int scorePoints, String interval, String intervalDec) {
        super(dead, currentHP, maxHP, aliveBody, deadBody, type, l);
        HP = hP;
        this.walkSpeed = walkSpeed;
        this.damage = damage;
        this.attackSpeed = attackSpeed;
        this.scorePoints = scorePoints;
        spawnIntervals = interval;
        spawnIntervalDecrements = intervalDec;
    }
    @Override
    public abstract create_level()
    {
        l.setMonsterTypes(this.type);
        l.setSpawnInterval(this.spawnIntervals);
        l.setSpawnIntervalDec(this.spawnIntervalDecrements);
    }
    @Override
    public String item()
    {
        return type;
    }
```

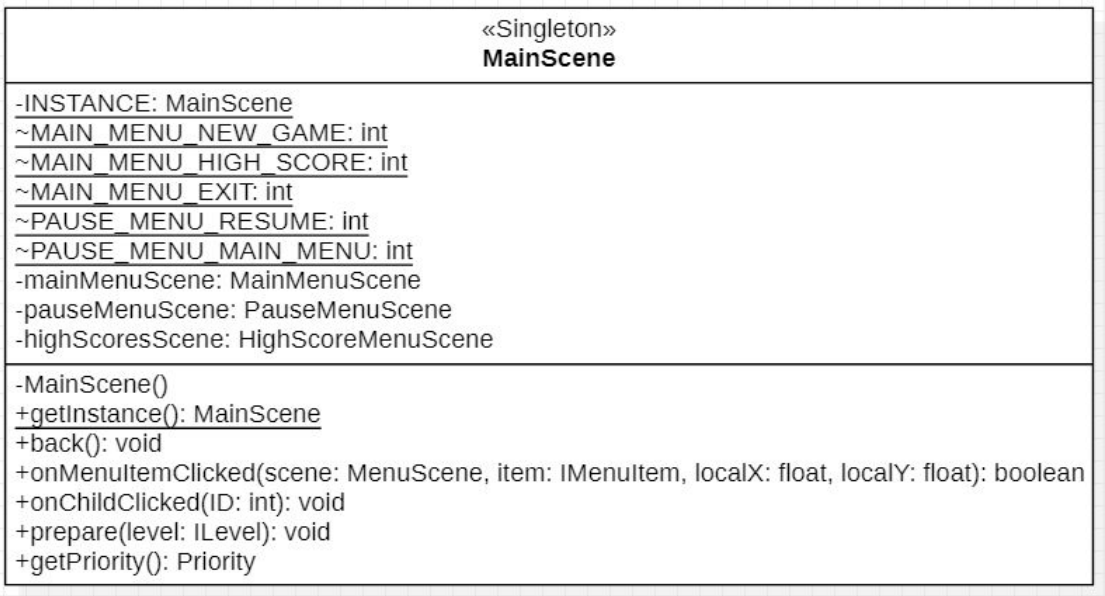
```java
public class Spider extends MonsterType
{

    public Spider(boolean dead, float currentHP, float maxHP, Body aliveBody, Body deadBody, String type, Level l,
    float hP, float walkSpeed, float damage,
    float attackSpeed, int scorePoints, String interval, String intervalDec){
        super(dead, currentHP, maxHP, aliveBody, deadBody, type, l, attackSpeed, scorePoints, interval, intervalDec);
    }

    @Override
    public void create_level()
    {
        l.setMonsterTypes(this.type);
        l.setSpawnInterval(this.spawnIntervals);
        l.setSpawnIntervalDec(this.spawnIntervalDecrements);
    }

    @Override
    public String item()
    {
        return type;
    }
```

Builder Pattern Code

```java
public abstract class GameMusicType implements LevelAspect
{
    protected String assetPath;
    protected Level l;

    private GameMusicType(String assetPath, Level l) {
        this.assetPath = assetPath;
        this.l = l;
    }

    public String getAssetPath() {
        return assetPath;
    }

    @Override
    public void create_level();

    @Override
    public String item();

}
```

```java
public class MusicOne extends GameMusicType
{
    private MusicOne(String assetPath, Level l) {
        super(assetPath, l);
    }

    public String getAssetPath() {
        return assetPath;
    }

    public void setLevel(Level l)
    {
        this.l = l;
    }

    @Override
    public void create_level()
    {
        l.setMusic(this.assetPath);
    }

    @Override
    public String item()
    {
        return assetPath;
    }
}
```

Singleton Class Diagram



Singleton Code

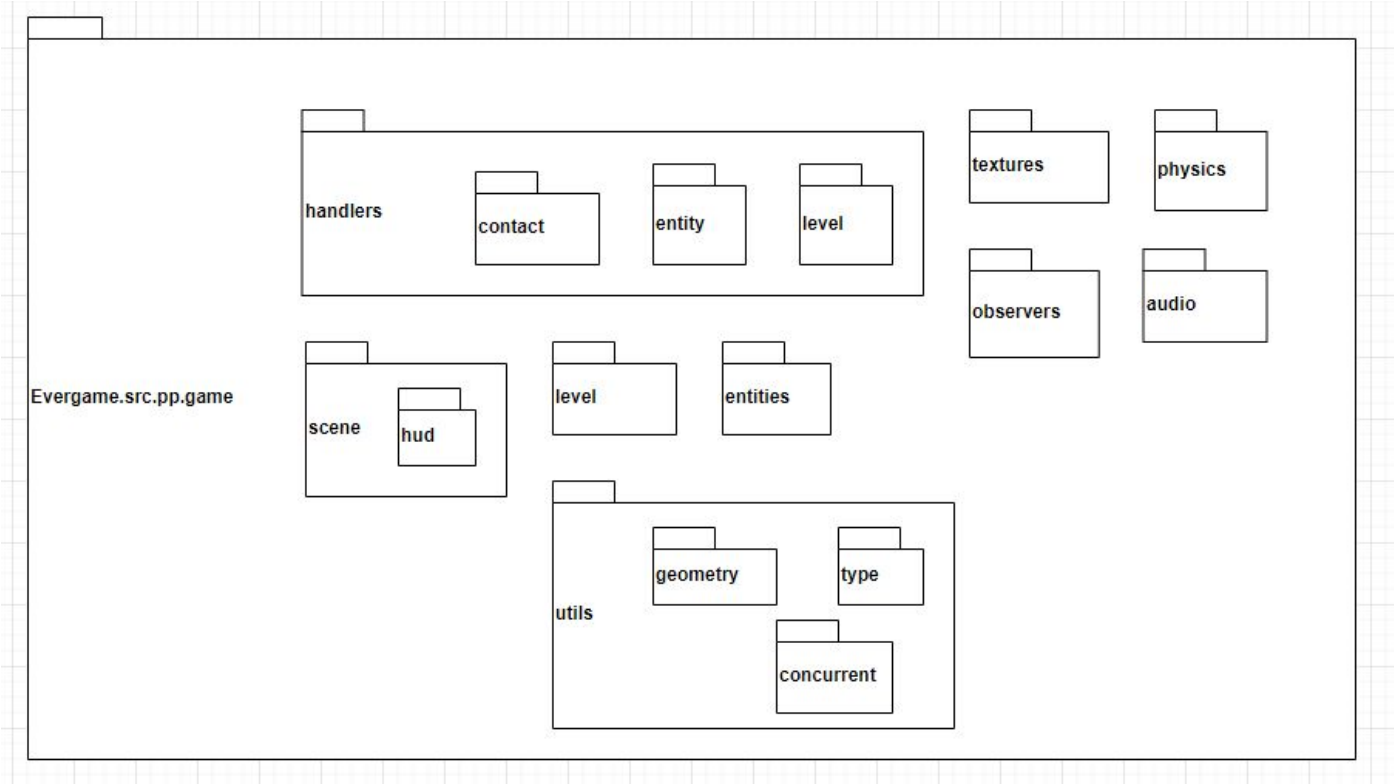
Class MainScene:

```
public class MainScene extends Scene implements IOnMenuItemClickListener, IChildClickListener, IPreparable {

    private static MainScene INSTANCE;

    static final int MAIN_MENU_NEW_GAME = 0x0;
    static final int MAIN_MENU_HIGH_SCORES = 0x1;
    static final int MAIN_MENU_EXIT = 0x2;

    static final int PAUSE_MENU_RESUME = 0x3;
    static final int PAUSE_MENU_MAIN_MENU = 0x4;

    private MainMenuScene mainMenuScene;
    private PauseMenuScene pauseMenuScene;
    private HighScoresMenuScene highScoresScene;

    private MainScene() {
        Game.getGameInstance().addPreparable(this);

        mainMenuScene = new MainMenuScene(this);
        pauseMenuScene = new PauseMenuScene(this);
        highScoresScene = new HighScoresMenuScene(this);

        GameScene.setChildClickListener(this);

        setChildScene(mainMenuScene);
    }

    public static Scene getInstance() {
        if(INSTANCE == null) INSTANCE = new MainScene();
        return INSTANCE;
    }
```

Class Game:

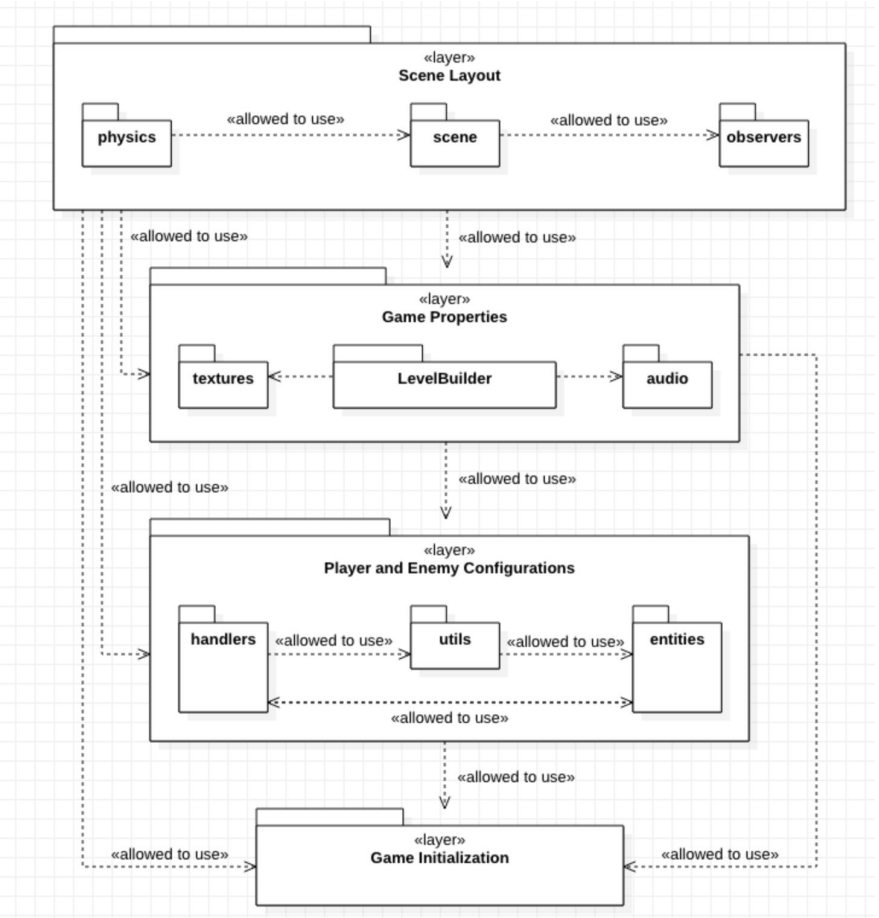
```
@Override
public void initialize() {
    AudioHolder.getInstance().initialize();
    TextureHolder.getInstance();
    scene = MainScene.getInstance();
    initializePreparables();
}
```

Decomposition View

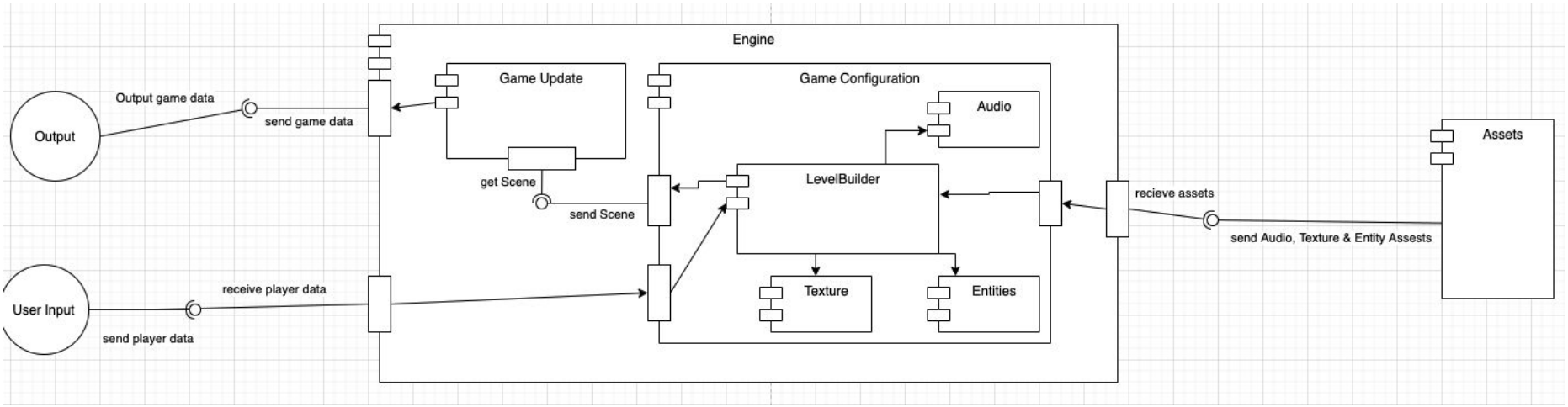


Updated Layered View

- Small architecture change in layer 2
- LevelBuilder handles construction of levels
- Previous implementation, submodules responsible for level building all depended on each other



Updated C&C View



- Creation of Complex object is hidden from client.
- Complex Object “Level” creates many small objects to create the overall Level.
- Creation of small objects is internalized in LevelBuilder.
- Objects with similar functionality all within the LevelBuilder Module.

6.867 Final Project - ML

6.867 Final Project Planned Design Patterns

- Builder Pattern
  - Justification: Creation of levels was to complicated and all sub components are made separately
  - Solution: Builder class handles the creation of Levels, and the creation of the small components
- Singleton Pattern
  - Justification: Level creation depends on lots of variables and data, all of which is used by the sub components of the level
  - Solution: Encapsulate the information regarding the level in a singleton instance, and allow the level components to access the singular instance of level attributes

Questions