Content

[Project package Classes 3](#_Toc57292225)

[Monster 3](#_Toc57292226)

[As of: 05.11.2020 3](#_Toc57292227)

[Player 4](#_Toc57292228)

[As of: 05.11.2020 4](#_Toc57292229)

[SClassCreature 6](#_Toc57292230)

[As of: 23.11.2020 6](#_Toc57292231)

[As of: 05.11.2020 10](#_Toc57292232)

[As of: 24.10.2020 11](#_Toc57292233)

[Project package Game 12](#_Toc57292234)

[Battle 12](#_Toc57292235)

[As of: 23.11.2020 12](#_Toc57292236)

[Main 15](#_Toc57292237)

[As of: 23.11.2020 15](#_Toc57292238)

[As of: 05.11.2020 15](#_Toc57292239)

[As of: 24.10.2020 15](#_Toc57292240)

[OpenWorldGame 16](#_Toc57292241)

[Release: 26.11.2020 16](#_Toc57292242)

[As of: 23.11.2020 20](#_Toc57292243)

[SaveGame 24](#_Toc57292244)

[Release: 26.11.2020 24](#_Toc57292245)

[As of: 23.11.2020 27](#_Toc57292246)

[Project package GUI 30](#_Toc57292247)

[BattlePanel 30](#_Toc57292248)

[As of: 23.11.2020 30](#_Toc57292249)

[As of: 04.11.2020 33](#_Toc57292250)

[CharCreationPanel 36](#_Toc57292251)

[As of: 23.11.2020 36](#_Toc57292252)

[As of: 04.11.2020 42](#_Toc57292253)

[EndPanel 46](#_Toc57292254)

[As of: 04.11.2020 46](#_Toc57292255)

[As of: 04.11.2020 48](#_Toc57292256)

[HelpPanel 50](#_Toc57292257)

[As of: 23.11.2020 50](#_Toc57292258)

[As of: 29.10.2020 52](#_Toc57292259)

[LoadPanel 54](#_Toc57292260)

[Release: 26.11.2020 54](#_Toc57292261)

[As of: 23.11.2020 58](#_Toc57292262)

[As of: 05.11.2020 62](#_Toc57292263)

[MainWindow 65](#_Toc57292264)

[Release: 26.11.2020 65](#_Toc57292265)

[As of: 23.11.2020 69](#_Toc57292266)

[As of: 05.11.2020 73](#_Toc57292267)

[As of: 04.11.2020 75](#_Toc57292268)

[As of: 29.10.2020 78](#_Toc57292269)

[As of: 24.10.2020 81](#_Toc57292270)

[MenuPanel 83](#_Toc57292271)

[Release: 26.11.2020 83](#_Toc57292272)

[As of: 05.11.2020 86](#_Toc57292273)

[OpenWorldPanel 89](#_Toc57292274)

[As of: 23.11.2020 89](#_Toc57292275)

[SavePanel 92](#_Toc57292276)

[As of: 23.11.2020 92](#_Toc57292277)

[As of: 05.11.2020 95](#_Toc57292278)

[StartPanel 98](#_Toc57292279)

[As of: 29.10.2020 98](#_Toc57292280)

[WindowSizePanel 100](#_Toc57292281)

[As of: 23.11.2020 100](#_Toc57292282)

[As of: 05.11.2020 104](#_Toc57292283)

[Project package Variables 108](#_Toc57292284)

[Variables 108](#_Toc57292285)

[As of: 23.11.2020 108](#_Toc57292286)

# Project package Classes

## Monster

### As of: 05.11.2020

package Classes;  
  
public class Monster extends SClassCreature{  
  
 private String type;  
  
 public String getType() {  
 return type;  
 }  
  
 public void setType(String type) {  
 this.type = type;  
 }  
  
 public void profile(){  
 System.*out*.println("Type: " +type +" Level: " +getLevel());  
 System.*out*.printf("Life: %d Stamina: %d \n", getLife(), getStamina());  
 System.*out*.printf("Damage: %d Experience: %d \n", getDamage(), getExperience());  
 }  
}

## Player

### As of: 05.11.2020

package Classes;  
  
public class Player extends SClassCreature {  
  
 private String name;  
 private String sex;  
 private float levelUpBonus = 1.2f;  
 private int experienceCap;  
  
 public float getLevelUpBonus() {  
 return levelUpBonus;  
 }  
  
 public void setLevelUpBonus(float levelUpBonus){  
 this.levelUpBonus = levelUpBonus;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getSex() {  
 return sex;  
 }  
  
 public void setSex(String sex) {  
 this.sex = sex;  
 }  
  
 public int getExperienceCap() {  
 return experienceCap;  
 }  
  
 public void setExperienceCap(int experienceCap) {  
 this.experienceCap = experienceCap;  
 }  
  
 */\*\*  
 \* A Method to show the Information of a Player  
 \* Will also trigger the levelUP() event  
 \*/* public void profile() {  
 if (getExperience() >= getExperienceCap()) {  
 levelUp();  
 } else if (isDeath()) {  
 System.*out*.println("Status: Dead");  
 }  
  
 System.*out*.println("Name: " + getName() + " Geschlecht: " +getSex() + " Level: " + getLevel());  
 System.*out*.printf("Life: %d(%d) Stamina: %d(%d)\n", getLife(), getMaxLife(), getStamina(), getMaxStamina());  
 System.*out*.printf("Damage: %d Experience: %d(%d)\n", getDamage(), getExperience(), getExperienceCap());  
 }  
  
 private void levelUp() {  
  
 do {  
 setLevel(getLevel() + 1);  
 setExperience(getExperience() - getExperienceCap());  
 setExperienceCap(Math.*round*(getExperienceCap() \* getLevelUpBonus()));  
 setMaxLife(Math.*round*(getMaxLife() \* getLevelUpBonus()));  
 setMaxStamina(Math.*round*(getMaxStamina() \* getLevelUpBonus()));  
 setDamage(Math.*round*(getDamage() \* getLevelUpBonus()));  
 System.*out*.println("Glückwunsch, du bist im Level gestiegen\n");  
 } while (getExperience() >= getExperienceCap());  
 }  
}

## SClassCreature

### As of: 23.11.2020

package Classes;  
  
public class SClassCreature {  
  
 private boolean death = false;  
 private boolean tired = false;  
  
 private String text = "";  
 private String place;  
  
 private int maxLife;  
 private int life;  
 private int maxStamina;  
 private int stamina;  
 private int staminaLoss = 10;  
 private int level;  
 private int experience;  
 private int damage;  
  
 public int getStaminaLoss() {  
 return staminaLoss;  
 }  
  
 public void setStaminaLoss(int staminaLoss) {  
 this.staminaLoss = staminaLoss;  
 }  
  
 public boolean isDeath() {  
 return death;  
 }  
  
 public void setDeath(boolean death) {  
 this.death = death;  
 }  
  
 public boolean isTired() {  
 return tired;  
 }  
  
 public void setTired(boolean tired) {  
 this.tired = tired;  
 }  
  
 public int getMaxLife() {  
 return maxLife;  
 }  
  
 public void setMaxLife(int maxLife) {  
 this.maxLife = maxLife;  
 }  
  
 public int getLife() {  
 return life;  
 }  
  
 public void setLife(int life) {  
 this.life = life;  
 }  
  
 public int getMaxStamina() {  
 return maxStamina;  
 }  
  
 public void setMaxStamina(int maxStamina) {  
 this.maxStamina = maxStamina;  
 }  
  
 public int getStamina() {  
 return stamina;  
 }  
  
 public void setStamina(int stamina) {  
 this.stamina = stamina;  
 }  
  
 public int getLevel() {  
 return level;  
 }  
  
 public void setLevel(int level) {  
 this.level = level;  
 }  
  
 public int getExperience() {  
 return experience;  
 }  
  
 public void setExperience(int experience) {  
 this.experience = experience;  
 }  
  
 public int getDamage() {  
 return damage;  
 }  
  
 public void setDamage(int damage) {  
 this.damage = damage;  
 }  
  
 */\*\*  
 \* Let's the Creature rest, it depends on the place  
 \** ***@param*** *place  
 \** ***@return*** *\*/* public String rest(String place) {  
 this.place = place;  
 float multiplierLife = 1.3f;  
 float multiplierStamina = 1.5f;  
  
 if (this.place.equals("openWorldRest")) {  
 if (isTired()) {  
 setLife(getMaxLife());  
 setStamina(getMaxStamina());  
 setTired(false);  
 text = "Du hast dich ausgeruht und bist wieder fit.\n";  
 } else text = "Du bist ausgeruht.\n";  
 } else{  
 if (Math.*round*(getLife() \* multiplierLife) <= getMaxLife()) {  
 int life = getLife() + Math.*round*(getMaxLife() \* (multiplierLife - 1));  
 setLife(life);  
 } else{  
 setLife(getMaxLife());  
 }  
 if (Math.*round*(getStamina() \* multiplierStamina) <= getMaxStamina()) {  
 int stamina = getStamina() + Math.*round*(getMaxStamina() \* (multiplierStamina-1));  
 setStamina(stamina);  
 } else{  
 setStamina(getMaxStamina());  
 }  
  
 text = "Du hast dich etwas erholt.\n";  
 }  
  
 text += "Life: " + getLife() + "(" + getMaxLife() + ") Stamina: " + getStamina() + "(" + getMaxStamina() + ")\n";  
 text += "\nWas möchtest du machen?";  
  
 return text;  
 }  
  
 public String giveDamage(String nameAttacker) {  
  
 text = nameAttacker +" greift an.\n";  
 setStamina(getStamina() - getStaminaLoss());  
 setTired(true);  
  
 return text;  
 }  
  
 public String receiveDamage(String nameReceiver, int damage) {  
  
 text = nameReceiver +" erleidet " +damage +" Schaden\n";  
  
 setLife(this.getLife() - damage);  
 setTired(true);  
  
 if(this.getLife() <= 0) this.setDeath(true);  
  
 return text;  
 }  
  
 public void receiveExperience(int experience) {  
 experience = getExperience() + experience;  
 setExperience(experience);  
 }  
  
 public String giveCounterAttack(String nameReceiver) {  
 text = "Du konterst den Angriff von " +nameReceiver +"\n";  
 setStamina(getStamina() - Math.*round*(getStaminaLoss()/2.0f));  
 setTired(true);  
  
 return text;  
 }  
  
 public String receiveCounterAttackDamage(String nameReceiver, int damage) {  
  
 text = nameReceiver +" erleidet " +(damage \* 3) +" Schaden\n";  
  
 setLife(this.getLife() - damage \* 3);  
 setTired(true);  
  
 if(this.getLife() <= 0) this.setDeath(true);  
  
 return text;  
 }  
}

### As of: 05.11.2020

package Classes;  
  
public class SClassCreature {  
  
 private boolean death = false;  
 private boolean tired = false;  
  
 private int maxLife;  
 private int life;  
 private int maxStamina;  
 private int stamina;  
 private int level;  
 private int experience;  
 private int damage;  
  
 public boolean isDeath() {  
 return death;  
 }  
  
 public void setDeath(boolean death) {  
 this.death = death;  
 }  
  
 public boolean isTired() {  
 return tired;  
 }  
  
 public void setTired(boolean tired) {  
 this.tired = tired;  
 }  
  
 public int getMaxLife() {  
 return maxLife;  
 }  
  
 public void setMaxLife(int maxLife) {  
 this.maxLife = maxLife;  
 }  
  
 public int getLife() {  
 return life;  
 }  
  
 public void setLife(int life) {  
 this.life = life;  
 }  
  
 public int getMaxStamina() {  
 return maxStamina;  
 }  
  
 public void setMaxStamina(int maxStamina) {  
 this.maxStamina = maxStamina;  
 }  
  
 public int getStamina() {  
 return stamina;  
 }  
  
 public void setStamina(int stamina) {  
 this.stamina = stamina;  
 }  
  
 public int getLevel() {  
 return level;  
 }  
  
 public void setLevel(int level) {  
 this.level = level;  
 }  
  
 public int getExperience() {  
 return experience;  
 }  
  
 public void setExperience(int experience) {  
 this.experience = experience;  
 }  
  
 public int getDamage() {  
 return damage;  
 }  
  
 public void setDamage(int damage) {  
 this.damage = damage;  
 }  
}

### As of: 24.10.2020

package Classes;  
  
public class SClassCreature  
{  
 private int level;  
  
 public void setLevel(int level)  
 {  
 this.level = level;  
 }  
  
 public int getLevel()  
 {  
 return this.level;  
 }  
}

# Project package Game

## Battle

### As of: 23.11.2020

package Game;  
  
import Classes.Monster;  
import Classes.Player;  
  
public class Battle {  
  
 Classes.Monster monster;  
 Classes.Player myPlayer;  
  
 private String monsterType = "Goblin";  
 private String text = "test";  
 private String place = "battle";  
 private boolean recover = false;  
 private int monsterLevel = 1;  
 private int maxLife = 50;  
 private int maxStamina = 30;  
 private int staminaLoss = 8;  
 private int experience = 10;  
 private int damage = 15;  
  
  
 public Monster battle(Player myPlayer) {  
  
 this.monster = new Classes.Monster();  
 this.myPlayer = myPlayer;  
  
 monster.setType(monsterType);  
 monster.setLevel(monsterLevel);  
 monster.setMaxLife(maxLife);  
 monster.setLife(maxLife);  
 monster.setMaxStamina(maxStamina);  
 monster.setStamina(maxStamina);  
 monster.setStaminaLoss(staminaLoss);  
 monster.setExperience(experience);  
 monster.setDamage(damage);  
  
 return monster;  
 }  
  
 public String fight() {  
  
 place = "battle";  
 text = "";  
  
 if(this.myPlayer.getStamina() >= this.myPlayer.getStaminaLoss() && !recover){  
 text += this.myPlayer.giveDamage(this.myPlayer.getName());  
 text += monster.receiveDamage(monster.getType(), this.myPlayer.getDamage());  
 }else if(this.myPlayer.getStamina() <= this.myPlayer.getStaminaLoss() || recover){  
 if(recover) text = "Du erholst dich.\n";  
 else text = "Du hast nicht genug ausdauer.\nDu erholst dich.\n";  
 this.myPlayer.rest(place);  
 recover = false;  
 }  
  
 text += "\n";  
 if(!monster.isDeath()){  
 if(monster.getStamina() >= monster.getStaminaLoss()) {  
 text += monster.giveDamage(monster.getType());  
 text += this.myPlayer.receiveDamage(this.myPlayer.getName(), monster.getDamage());  
 }else  
 {  
 text += monster.getType() + " erholt sich.\n";  
 monster.rest(place);  
 }  
 } else text += monster.getType() + " ist tod.\nDu hast gewonnen.\n";  
  
 if(!this.myPlayer.isDeath() && !monster.isDeath()) text += "\nRestleben: " + this.myPlayer.getLife() + " Restausdauer: " + myPlayer.getStamina();  
  
 return text;  
 }  
  
 public String recover() {  
 recover = true;  
 text = fight();  
  
 return text;  
 }  
  
 public String notEscaped() {  
 place = "battle";  
 text = "Flucht fehlgeschlagen!\n";  
  
 if(monster.getStamina() >= monster.getStaminaLoss()) {  
 text += monster.giveDamage(monster.getType());  
 text += this.myPlayer.receiveDamage(this.myPlayer.getName(), monster.getDamage());  
 }else  
 {  
 text += monster.getType() + " erholt sich.\n";  
 monster.rest(place);  
 }  
  
 if(!this.myPlayer.isDeath() && !monster.isDeath()) text += "\nRestleben: " + this.myPlayer.getLife() + " Restausdauer: " + myPlayer.getStamina();  
  
 return text;  
 }  
  
 public String attackBlocked() {  
 place = "battle";  
 text = "Du blockst den Angriff von " + monster.getType() +"\n";  
  
 if(monster.getStamina() >= monster.getStaminaLoss()) {  
 monster.giveDamage(monster.getType());  
 }else  
 {  
 text += monster.getType() + " erholt sich.\n";  
 monster.rest(place);  
 }  
  
 text += "\nRestleben: " + this.myPlayer.getLife() + " Restausdauer: " + this.myPlayer.getStamina();  
 return text;  
 }  
  
 public String attackCountered() {  
  
 place = "battle";  
 text = "";  
  
 if(this.myPlayer.getStamina() >= this.myPlayer.getStaminaLoss() && !recover){  
 text += this.myPlayer.giveCounterAttack(monster.getType());  
 text += monster.receiveCounterAttackDamage(monster.getType(), this.myPlayer.getDamage());  
 }else if(this.myPlayer.getStamina() <= this.myPlayer.getStaminaLoss() || recover){  
 text = "Du hast nicht genug ausdauer.\nDu erholst dich.\n";  
 this.myPlayer.rest(place);  
 recover = false;  
 }  
  
 text += "\n";  
 if(!monster.isDeath()){  
 text += monster.getType() + " wurde schwer getroffen und rappelt sich auf.";  
 } else text += monster.getType() + " ist tod.\nDu hast gewonnen.\n";  
  
 if(!this.myPlayer.isDeath() && !monster.isDeath()) text += "\nRestleben: " + this.myPlayer.getLife() + " Restausdauer: " + this.myPlayer.getStamina();  
  
 return text;  
 }  
}

## Main

### As of: 23.11.2020

package Game;  
  
import GUI.MainWindow;  
  
public class Main  
{  
 public static void main(String[] args)  
 {  
 OpenWorldGame openWorldGame = new OpenWorldGame();  
 new MainWindow(openWorldGame, 800,600, 1);  
 }  
}

### As of: 05.11.2020

package Game;  
  
import javax.swing.\*;  
  
public class Main  
{  
 public static void main(String[] args)  
 {  
 new GUI.MainWindow(800,600, 1);  
 }  
}

### As of: 24.10.2020

package Game;  
  
public class Main  
{  
 public static void main(String[] args)  
 {  
 new GUI.MainWindow();  
 }  
}

## OpenWorldGame

### Release: 26.11.2020

package Game;  
  
import Classes.Monster;  
import Classes.Player;  
import GUI.MainWindow;  
  
public class OpenWorldGame {  
  
 Classes.Monster monster = new Monster();  
 Game.Battle battle = new Battle();  
 MainWindow mainWindow;  
  
 private Classes.Player myPlayer = new Player();  
  
 private final float levelUpBonus = 1.2f;  
 private final int multiplier = 10;  
 private final int level = 1;  
 private final int experienceCap = 50;  
 private final int experience = 0;  
  
 private String text = "";  
 private String textSaveGame = "";  
  
 public void gameStart() {  
 int width = 800;  
 int height = 600;  
 mainWindow = new MainWindow(this, width, height, 1);  
 }  
  
 public void openWorldGame(String name, String sex, int life, int stamina, int damage, int width, int height) {  
  
 life \*= multiplier;  
 stamina \*= multiplier;  
 int maxLife = life;  
 int maxStamina = stamina;  
  
  
 myPlayer.setName(name);  
 myPlayer.setSex(sex);  
 myPlayer.setLevel(level);  
 myPlayer.setLevelUpBonus(levelUpBonus);  
 myPlayer.setMaxLife(maxLife);  
 myPlayer.setLife(life);  
 myPlayer.setMaxStamina(maxStamina);  
 myPlayer.setStamina(stamina);  
 myPlayer.setExperienceCap(experienceCap);  
 myPlayer.setExperience(experience);  
 myPlayer.setDamage(damage);  
  
 mainWindow = new MainWindow(this, width, height, 4);  
 mainWindow.setGameStart(true);  
 mainWindow.changeTextOpenWorld(myPlayer.profile());  
 }  
  
 //Erholen als nächste Aufgabe!  
  
 public void doRest(String place) {  
  
 if(place.equals("openWorldRest")) mainWindow.changeTextOpenWorld(myPlayer.rest(place));  
 else if(place.equals("battle")){  
 text = battle.recover();  
 mainWindow.changeTextBattle(text);  
 }  
 }  
  
 public boolean getProfile() {  
 if(!myPlayer.isDeath()) mainWindow.changeTextOpenWorld(myPlayer.profile());  
 if(myPlayer.getLevel() >= 3) mainWindow.youWon();  
 return myPlayer.isDeath();  
  
 // -> als nächstes Spieler gewinnt ab lvl 3  
 }  
  
 public void encounter() {  
 this.battle = new Battle();  
  
 monster = battle.battle(myPlayer);  
  
 text = "Sie begegnen einem " + monster.getType();  
  
 if(monster.getLevel() <= myPlayer.getLevel()) text += "\nSie fühlen sich dem Monster gewachsen.";  
 else text += "\nSie fühlen sich vom Monster eingeschüchtert.";  
  
 text += "\n\nWas möchtest du machen?";  
  
 mainWindow.changeTextBattle(text);  
 }  
  
 public void attack() {  
  
 if(!myPlayer.isDeath() && !monster.isDeath()){  
 text = battle.fight();  
 if(!monster.isDeath()){  
 text += "\n\nWas möchtest du machen?";  
 mainWindow.changeTextBattle(text);  
 } else  
 {  
 myPlayer.receiveExperience(monster.getExperience());  
 text += "\nRestleben: " + myPlayer.getLife() + " Restausdauer: " + myPlayer.getStamina();  
 text += "\nDu hast " + monster.getExperience() + " Erfahrungspunkte erhalten.";  
 endBattle(text);  
 }  
 }  
  
 if(myPlayer.isDeath()){  
 mainWindow.youLost();  
 }  
 }  
  
 public void endBattle(String text) {  
 mainWindow.battleEnd(text);  
 }  
  
 public void tryEscape() {  
 int chance = Math.*toIntExact*(Math.*round*((Math.*random*()) \* 5)) + 1;  
 if (chance <= 4) mainWindow.escaped();  
 else{  
 text = battle.notEscaped();  
 mainWindow.changeTextBattle(text);  
 }  
 }  
  
 public void block() {  
 text = battle.attackBlocked();  
 mainWindow.changeTextBattle(text);  
 mainWindow.monsterAttackBlocked(true);  
 }  
  
 public void counterAttack() {  
 text = battle.attackCountered();  
 mainWindow.monsterAttackBlocked(false);  
 if(monster.isDeath()){  
 myPlayer.receiveExperience(monster.getExperience());  
 text += "\nRestleben: " + myPlayer.getLife() + " Restausdauer: " + myPlayer.getStamina();  
 text += "\nDu hast " + monster.getExperience() + " Erfahrungspunkte erhalten.";  
 text += "\n\nWas möchtest du machen?";  
 endBattle(text);  
 } else{  
 text += "\n\nWas möchtest du machen?";  
 mainWindow.changeTextBattle(text);  
 }  
 }  
  
 public String saveMyGame(int i) {  
 text = myPlayer.getName() +" Level: ";  
 text += myPlayer.getLevel() + " Erfahrung: ";  
 text += myPlayer.getExperience();  
  
 createTextSaveGame();  
  
 SaveGame saveGame = new SaveGame();  
 saveGame.SaveMyGame(textSaveGame, i);  
  
 mainWindow.changeTextLoadPanel(text, i);  
  
 return text;  
 }  
  
 private void createTextSaveGame() {  
 textSaveGame = myPlayer.getName() +";";  
 textSaveGame += myPlayer.getSex() +";";  
 textSaveGame += myPlayer.getLevel() +";";  
 textSaveGame += myPlayer.getMaxLife() +";";  
 textSaveGame += myPlayer.getLife() +";";  
 textSaveGame += myPlayer.getMaxStamina() +";";  
 textSaveGame += myPlayer.getStamina() +";";  
 textSaveGame += myPlayer.getExperienceCap() +";";  
 textSaveGame += myPlayer.getExperience() +";";  
 textSaveGame += myPlayer.getDamage();  
 }  
  
 public void loadMyGame(int i, int width, int height) {  
 SaveGame saveGame = new SaveGame();  
 textSaveGame = saveGame.loadMyGame(i);  
  
 if(textSaveGame=="noGame"){  
 mainWindow = new MainWindow(this, width, height, 9);  
 return;  
 }  
  
 loadMyPlayer(textSaveGame, width, height);  
  
 if(myPlayer.isDeath()) myPlayer.setDeath(false);  
 mainWindow.setGameStart(true);  
 }  
  
 private void loadMyPlayer(String textSaveGame, int width, int height) {  
  
 String[] myPlayerString = textSaveGame.split(";");  
  
 myPlayer.setName(myPlayerString[0]);  
 myPlayer.setSex(myPlayerString[1]);  
 myPlayer.setLevel(Integer.*parseInt*(myPlayerString[2]));  
 myPlayer.setLevelUpBonus(levelUpBonus);  
 myPlayer.setMaxLife(Integer.*parseInt*(myPlayerString[3]));  
 myPlayer.setLife(Integer.*parseInt*(myPlayerString[4]));  
 myPlayer.setMaxStamina(Integer.*parseInt*(myPlayerString[5]));  
 myPlayer.setStamina(Integer.*parseInt*(myPlayerString[6]));  
 myPlayer.setExperienceCap(Integer.*parseInt*(myPlayerString[7]));  
 myPlayer.setExperience(Integer.*parseInt*(myPlayerString[8]));  
 myPlayer.setDamage(Integer.*parseInt*(myPlayerString[9]));  
  
 if(myPlayer.isDeath()) myPlayer.setDeath(false);  
  
 mainWindow = new MainWindow(this, width, height, 4);  
 mainWindow.changeTextOpenWorld(myPlayer.profile());  
 }  
}

### As of: 23.11.2020

package Game;  
  
import Classes.Monster;  
import Classes.Player;  
import GUI.MainWindow;  
  
public class OpenWorldGame {  
  
 Classes.Monster monster = new Monster();  
 Game.Battle battle = new Battle();  
 MainWindow mainWindow;  
  
 private Classes.Player myPlayer = new Player();  
  
 private final float levelUpBonus = 1.2f;  
 private final int multiplier = 10;  
 private final int level = 1;  
 private final int experienceCap = 50;  
 private final int experience = 0;  
  
 private String text = "";  
 private String textSaveGame = "";  
  
 public void openWorldGame(String name, String sex, int life, int stamina, int damage, int width, int height) {  
  
 life \*= multiplier;  
 stamina \*= multiplier;  
 int maxLife = life;  
 int maxStamina = stamina;  
  
  
 myPlayer.setName(name);  
 myPlayer.setSex(sex);  
 myPlayer.setLevel(level);  
 myPlayer.setLevelUpBonus(levelUpBonus);  
 myPlayer.setMaxLife(maxLife);  
 myPlayer.setLife(life);  
 myPlayer.setMaxStamina(maxStamina);  
 myPlayer.setStamina(stamina);  
 myPlayer.setExperienceCap(experienceCap);  
 myPlayer.setExperience(experience);  
 myPlayer.setDamage(damage);  
  
 mainWindow = new MainWindow(this, width, height, 4);  
 mainWindow.changeTextOpenWorld(myPlayer.profile());  
 }  
  
 //Erholen als nächste Aufgabe!  
  
 public void doRest(String place) {  
  
 if(place.equals("openWorldRest")) mainWindow.changeTextOpenWorld(myPlayer.rest(place));  
 else if(place.equals("battle")){  
 text = battle.recover();  
 mainWindow.changeTextBattle(text);  
 }  
 }  
  
 public boolean getProfile() {  
 if(!myPlayer.isDeath()) mainWindow.changeTextOpenWorld(myPlayer.profile());  
 if(myPlayer.getLevel() >= 3) mainWindow.youWon();  
 return myPlayer.isDeath();  
  
 // -> als nächstes Spieler gewinnt ab lvl 3  
 }  
  
 public void encounter() {  
 this.battle = new Battle();  
  
 monster = battle.battle(myPlayer);  
  
 text = "Sie begegnen einem " + monster.getType();  
  
 if(monster.getLevel() <= myPlayer.getLevel()) text += "\nSie fühlen sich dem Monster gewachsen.";  
 else text += "\nSie fühlen sich vom Monster eingeschüchtert.";  
  
 text += "\n\nWas möchtest du machen?";  
  
 mainWindow.changeTextBattle(text);  
 }  
  
 public void attack() {  
  
 if(!myPlayer.isDeath() && !monster.isDeath()){  
 text = battle.fight();  
 if(!monster.isDeath()){  
 text += "\n\nWas möchtest du machen?";  
 mainWindow.changeTextBattle(text);  
 } else  
 {  
 myPlayer.receiveExperience(monster.getExperience());  
 text += "\nRestleben: " + myPlayer.getLife() + " Restausdauer: " + myPlayer.getStamina();  
 text += "\nDu hast " + monster.getExperience() + " Erfahrungspunkte erhalten.";  
 endBattle(text);  
 }  
 }  
  
 if(myPlayer.isDeath()){  
 mainWindow.youLost();  
 }  
 }  
  
 public void endBattle(String text) {  
 mainWindow.battleEnd(text);  
 }  
  
 public void tryEscape() {  
 int chance = Math.*toIntExact*(Math.*round*((Math.*random*()) \* 5)) + 1;  
 if (chance <= 4) mainWindow.escaped();  
 else{  
 text = battle.notEscaped();  
 mainWindow.changeTextBattle(text);  
 }  
 }  
  
 public void block() {  
 text = battle.attackBlocked();  
 mainWindow.changeTextBattle(text);  
 mainWindow.monsterAttackBlocked(true);  
 }  
  
 public void counterAttack() {  
 text = battle.attackCountered();  
 mainWindow.monsterAttackBlocked(false);  
 if(monster.isDeath()){  
 myPlayer.receiveExperience(monster.getExperience());  
 text += "\nRestleben: " + myPlayer.getLife() + " Restausdauer: " + myPlayer.getStamina();  
 text += "\nDu hast " + monster.getExperience() + " Erfahrungspunkte erhalten.";  
 text += "\n\nWas möchtest du machen?";  
 endBattle(text);  
 } else{  
 text += "\n\nWas möchtest du machen?";  
 mainWindow.changeTextBattle(text);  
 }  
 }  
  
 public String saveMyGame(int i) {  
 text = myPlayer.getName() +" Level: ";  
 text += myPlayer.getLevel() + " Erfahrung: ";  
 text += myPlayer.getExperience();  
  
 createTextSaveGame();  
  
 SaveGame saveGame = new SaveGame();  
 saveGame.SaveMyGame(textSaveGame, i);  
  
 mainWindow.changeTextLoadPanel(text, i);  
  
 return text;  
 }  
  
 private void createTextSaveGame() {  
 textSaveGame = myPlayer.getName() +";";  
 textSaveGame += myPlayer.getSex() +";";  
 textSaveGame += myPlayer.getLevel() +";";  
 textSaveGame += myPlayer.getMaxLife() +";";  
 textSaveGame += myPlayer.getLife() +";";  
 textSaveGame += myPlayer.getMaxStamina() +";";  
 textSaveGame += myPlayer.getStamina() +";";  
 textSaveGame += myPlayer.getExperienceCap() +";";  
 textSaveGame += myPlayer.getExperience() +";";  
 textSaveGame += myPlayer.getDamage();  
 }  
  
 public void loadMyGame(int i, int width, int height) {  
 SaveGame saveGame = new SaveGame();  
 textSaveGame = saveGame.loadMyGame(i);  
  
 loadMyPlayer(textSaveGame, width, height);  
 }  
  
 private void loadMyPlayer(String textSaveGame, int width, int height) {  
  
 String[] myPlayerString = textSaveGame.split(";");  
  
 myPlayer.setName(myPlayerString[0]);  
 myPlayer.setSex(myPlayerString[1]);  
 myPlayer.setLevel(Integer.*parseInt*(myPlayerString[2]));  
 myPlayer.setLevelUpBonus(levelUpBonus);  
 myPlayer.setMaxLife(Integer.*parseInt*(myPlayerString[3]));  
 myPlayer.setLife(Integer.*parseInt*(myPlayerString[4]));  
 myPlayer.setMaxStamina(Integer.*parseInt*(myPlayerString[5]));  
 myPlayer.setStamina(Integer.*parseInt*(myPlayerString[6]));  
 myPlayer.setExperienceCap(Integer.*parseInt*(myPlayerString[7]));  
 myPlayer.setExperience(Integer.*parseInt*(myPlayerString[8]));  
 myPlayer.setDamage(Integer.*parseInt*(myPlayerString[9]));  
  
 mainWindow = new MainWindow(this, width, height, 4);  
 mainWindow.changeTextOpenWorld(myPlayer.profile());  
 }  
}

## SaveGame

### Release: 26.11.2020

package Game;  
  
import java.io.\*;  
import java.util.Scanner;  
  
public class SaveGame {  
  
 File saveGame = new File("SaveGame");  
 File saveOne = new File("SaveGame/saveOne.txt");  
 File saveTwo = new File("SaveGame/saveTwo.txt");  
 File saveThree = new File("SaveGame/saveThree.txt");  
  
 public void SaveMyGame(String textSaveGame, int i) {  
  
 if (!saveGame.exists()) saveGame.mkdir();  
  
 switch (i) {  
 case 1:  
 if(!saveOne.exists()) CreateSaveOne();  
 try{  
 OutputStream stream = new FileOutputStream(saveOne);  
 stream.write(textSaveGame.getBytes());  
 stream.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 break;  
 case 2:  
 if(!saveTwo.exists()) CreateSaveTwo();  
 try{  
 OutputStream stream = new FileOutputStream(saveTwo);  
 stream.write(textSaveGame.getBytes());  
 stream.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 break;  
 case 3:  
 if(!saveThree.exists()) CreateSaveThree();  
 try{  
 OutputStream stream = new FileOutputStream(saveThree);  
 stream.write(textSaveGame.getBytes());  
 stream.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 private void CreateSaveOne() {  
  
 try{  
 saveOne.createNewFile();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 private void CreateSaveTwo() {  
  
 try{  
 saveTwo.createNewFile();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 private void CreateSaveThree() {  
  
 try{  
 saveThree.createNewFile();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public boolean SaveGameExists(){  
 boolean exists = FolderExists();  
  
 if(exists) exists = SaveExists();  
  
 return exists;  
 }  
  
 public boolean FolderExists(){  
 boolean exists = false;  
  
 if (saveGame.exists()) {  
 exists = true;  
 }  
  
 return exists;  
 }  
  
 public boolean SaveExists(){  
 boolean exists = false;  
  
 if(saveOne.exists()) exists = true;  
 else if(saveTwo.exists()) exists = true;  
 else if(saveThree.exists()) exists = true;  
  
 return exists;  
 }  
  
 public boolean SaveExists(int i){  
 boolean exists = false;  
  
 switch (i) {  
 case 1: if(saveOne.exists()) exists = true;  
 break;  
 case 2: if(saveTwo.exists()) exists = true;  
 break;  
 case 3: if(saveThree.exists()) exists = true;  
 }  
 return exists;  
 }  
  
 public String loadMyGame(int i) {  
 String text = "noGame";  
  
 if(saveGame.exists()){  
 switch (i) {  
 case 1: if(saveOne.exists()) {  
 text = "";  
 try {  
 Scanner sc = new Scanner(saveOne);  
 while (sc.hasNext()) {  
 text += sc.next();  
 }  
 sc.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
 break;  
 case 2: if(saveTwo.exists()){  
 text = "";  
 try{  
 Scanner sc = new Scanner(saveTwo);  
 while (sc.hasNext()) {  
 text += sc.next();  
 }  
 sc.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
 break;  
 case 3: if(saveThree.exists()){  
 text = "";  
 try{  
 Scanner sc = new Scanner(saveThree);  
 while (sc.hasNext()) {  
 text += sc.next();  
 }  
 sc.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 }  
 return text;  
 }  
}

### As of: 23.11.2020

package Game;  
  
import java.io.\*;  
import java.util.Scanner;  
  
public class SaveGame {  
  
 File saveGame = new File("SaveGame");  
 File saveOne = new File("SaveGame/saveOne.txt");  
 File saveTwo = new File("SaveGame/saveTwo.txt");  
 File saveThree = new File("SaveGame/saveThree.txt");  
  
 public void SaveMyGame(String textSaveGame, int i) {  
  
 if (!saveGame.exists()) saveGame.mkdir();  
  
 switch (i) {  
 case 1:  
 if(!saveOne.exists()) CreateSaveOne();  
 try{  
 OutputStream stream = new FileOutputStream(saveOne);  
 stream.write(textSaveGame.getBytes());  
 stream.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 break;  
 case 2:  
 if(!saveTwo.exists()) CreateSaveTwo();  
 try{  
 OutputStream stream = new FileOutputStream(saveTwo);  
 stream.write(textSaveGame.getBytes());  
 stream.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 break;  
 case 3:  
 if(!saveThree.exists()) CreateSaveThree();  
 try{  
 OutputStream stream = new FileOutputStream(saveThree);  
 stream.write(textSaveGame.getBytes());  
 stream.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 private void CreateSaveOne() {  
  
 try{  
 saveOne.createNewFile();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 private void CreateSaveTwo() {  
  
 try{  
 saveTwo.createNewFile();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 private void CreateSaveThree() {  
  
 try{  
 saveThree.createNewFile();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public boolean SaveGameExists(){  
 boolean exists = FolderExists();  
  
 if(exists) exists = SaveExists();  
  
 return exists;  
 }  
  
 public boolean FolderExists(){  
 boolean exists = false;  
  
 if (saveGame.exists()) {  
 exists = true;  
 }  
  
 return exists;  
 }  
  
 public boolean SaveExists(){  
 boolean exists = false;  
  
 if(saveOne.exists()) exists = true;  
 else if(saveTwo.exists()) exists = true;  
 else if(saveThree.exists()) exists = true;  
  
 return exists;  
 }  
  
 public boolean SaveExists(int i){  
 boolean exists = false;  
  
 switch (i) {  
 case 1: if(saveOne.exists()) exists = true;  
 break;  
 case 2: if(saveTwo.exists()) exists = true;  
 break;  
 case 3: if(saveThree.exists()) exists = true;  
 }  
 return exists;  
 }  
  
 public String loadMyGame(int i) {  
 String text = "noGame";  
  
 if(saveGame.exists()){  
 switch (i) {  
 case 1: if(saveOne.exists()) {  
 text = "";  
 try {  
 Scanner sc = new Scanner(saveOne);  
 while (sc.hasNext()) {  
 text += sc.next();  
 }  
 sc.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
 break;  
 case 2: if(saveTwo.exists()){  
 text = "";  
 try{  
 Scanner sc = new Scanner(saveTwo);  
 while (sc.hasNext()) {  
 text += sc.next();  
 }  
 sc.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
 break;  
 case 3: if(saveThree.exists()){  
 text = "";  
 try{  
 Scanner sc = new Scanner(saveThree);  
 while (sc.hasNext()) {  
 text += sc.next();  
 }  
 sc.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 }  
 return text;  
 }  
}

# Project package GUI

## BattlePanel

### As of: 23.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class BattlePanel {  
  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font TEXT\_AREA\_FONT = new Font("Verdana", Font.*PLAIN*, 25);  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color TEXT\_AREA\_BACKGROUND\_COLOR = Color.*black*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
 private final Color TEXT\_AREA\_FOREGROUND\_COLOR = Color.*blue*;  
  
  
 private final JButton BLOCK = new JButton("Blocken");  
 private final JButton REST = new JButton("Erholen");  
 private final JButton ESCAPE = new JButton("Fliehen");  
  
 private BorderLayout openWorldBorderLayout;  
 private GridLayout owButtonGridLayout;  
 private BoxLayout openWorldBoxLayout;  
  
 private static JButton *ATTACK* = new JButton("Angriff");  
 private JTextArea shownText;  
 private JPanel owStartPanel, owCenterPanel, owShownTextPanel, owButtonPanel, owEndPanel;  
  
 private int windowWidth, windowHeight;  
  
  
 public JPanel BattlePanel(MainWindow.ChoiceHandler cHandler, CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 JPanel openWorldPanel = new JPanel();  
 openWorldBorderLayout = new BorderLayout();  
 openWorldPanel.setLayout(openWorldBorderLayout);  
  
 */\*\*  
 \* The Button to enter the menu  
 \* it's at the PAGE\_START  
 \*/* owStartPanel = new JPanel();  
 owStartPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowHeight = Math.*round*(height \* 0.07f);  
 owStartPanel.setPreferredSize(new Dimension(width, windowHeight));  
  
  
 */\*\*  
 \* The Center of the Panel  
 \* It shows the game texts and gives the Player the options  
 \* explore, rest and profile  
 \* those will trigger the actions  
 \*/* owCenterPanel = new JPanel();  
 owCenterPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 openWorldBoxLayout = new BoxLayout(owCenterPanel, BoxLayout.*Y\_AXIS*);  
 owCenterPanel.setLayout(openWorldBoxLayout);  
  
 //Create the Text area, which will give the text  
 owShownTextPanel = new JPanel();  
 owShownTextPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 shownText = new JTextArea("Testtext");  
 shownText.setBackground(TEXT\_AREA\_BACKGROUND\_COLOR);  
 shownText.setForeground(TEXT\_AREA\_FOREGROUND\_COLOR);  
 shownText.setFont(TEXT\_AREA\_FONT);  
 shownText.setEditable(false);  
 shownText.setBorder(*createEmptyBorder*());  
 windowWidth = Math.*round*(width \* 0.9f);  
 windowHeight = Math.*round*(height \* 0.6f);  
 shownText.setPreferredSize(new Dimension(windowWidth, windowHeight)); //Set the size of the TextArea  
  
 owShownTextPanel.add(shownText);  
  
 //Create the buttons area, which will give the player some options  
 owButtonPanel = new JPanel();  
 owButtonGridLayout = new GridLayout(1, 3);  
 owButtonPanel.setLayout(owButtonGridLayout);  
  
 //Create the button for the attack  
 *ATTACK*.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 *ATTACK*.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 *ATTACK*.setFocusPainted(false);  
 *ATTACK*.setBorder(*createEmptyBorder*());  
 *ATTACK*.setFont(BUTTON\_FONT);  
 *ATTACK*.addActionListener(cHandler);  
  
 //Create the button for the block  
 BLOCK.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 BLOCK.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 BLOCK.setFocusPainted(false);  
 BLOCK.setBorder(*createEmptyBorder*());  
 BLOCK.setFont(BUTTON\_FONT);  
 BLOCK.addActionListener(cHandler);  
  
 //Create the button for resting  
 REST.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 REST.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 REST.setFocusPainted(false);  
 REST.setBorder(*createEmptyBorder*());  
 REST.setFont(BUTTON\_FONT);  
 REST.addActionListener(cHandler);  
  
 //Create the button for escape  
 ESCAPE.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 ESCAPE.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 ESCAPE.setFocusPainted(false);  
 ESCAPE.setBorder(*createEmptyBorder*());  
 ESCAPE.setFont(BUTTON\_FONT);  
 ESCAPE.addActionListener(cHandler);  
  
 //Add buttons to the owButtonPanel  
 owButtonPanel.add(*ATTACK*);  
 owButtonPanel.add(BLOCK);  
 owButtonPanel.add(REST);  
 owButtonPanel.add(ESCAPE);  
  
  
 //Add Panels to the owCenterPanel  
 owCenterPanel.add(owShownTextPanel);  
 owCenterPanel.add(owButtonPanel);  
  
  
 */\*\*  
 \* The Page ends as a blank Panel  
 \* it's at the PAGE\_END  
 \*/* owEndPanel = new JPanel();  
 owEndPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowHeight = Math.*round*(height \* 0.1f);  
 owEndPanel.setPreferredSize(new Dimension(width, windowHeight));  
  
 */\*\*  
 \* Adding everything together  
 \* return the openWorldPanel  
 \*/* openWorldPanel.add(owStartPanel, BorderLayout.*PAGE\_START*);  
 openWorldPanel.add(owCenterPanel, BorderLayout.*CENTER*);  
 openWorldPanel.add(owEndPanel, BorderLayout.*PAGE\_END*);  
  
 return openWorldPanel;  
 }  
  
 public void setText(String text){  
 shownText.setText(text);  
 }  
  
 public static void counter(boolean isBlocked) {  
 if(isBlocked) *ATTACK*.setText("Konter");  
 else *ATTACK*.setText("Angriff");  
 }  
}

### As of: 04.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class BattlePanel {  
  
 private final Font MENU\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 15);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font TEXT\_AREA\_FONT = new Font("Verdana", Font.*PLAIN*, 25);  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color TEXT\_AREA\_BACKGROUND\_COLOR = Color.*black*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
 private final Color TEXT\_AREA\_FOREGROUND\_COLOR = Color.*blue*;  
  
  
 private final JButton ATTACK = new JButton("Angriff");  
 private final JButton BLOCK = new JButton("Blocken");  
 private final JButton REST = new JButton("Erholen");  
 private final JButton ESCAPE = new JButton("Fliehen");  
  
 private BorderLayout openWorldBorderLayout;  
 private GridLayout owButtonGridLayout;  
 private BoxLayout openWorldBoxLayout;  
  
 private JTextArea shownText;  
 private JPanel owStartPanel, owCenterPanel, owShownTextPanel, owButtonPanel, owEndPanel;  
  
 private int windowWidth, windowHeight;  
  
 public JPanel BattlePanel(CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 JPanel openWorldPanel = new JPanel();  
 openWorldBorderLayout = new BorderLayout();  
 openWorldPanel.setLayout(openWorldBorderLayout);  
  
 */\*\*  
 \* The Button to enter the menu  
 \* it's at the PAGE\_START  
 \*/* owStartPanel = new JPanel();  
 owStartPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowHeight = Math.*round*(height \* 0.07f);  
 owStartPanel.setPreferredSize(new Dimension(width, windowHeight));  
  
  
 */\*\*  
 \* The Center of the Panel  
 \* It shows the game texts and gives the Player the options  
 \* explore, rest and profile  
 \* those will trigger the actions  
 \*/* owCenterPanel = new JPanel();  
 owCenterPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 openWorldBoxLayout = new BoxLayout(owCenterPanel, BoxLayout.*Y\_AXIS*);  
 owCenterPanel.setLayout(openWorldBoxLayout);  
  
 //Create the Text area, which will give the text  
 owShownTextPanel = new JPanel();  
 owShownTextPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 shownText = new JTextArea("Testtext");  
 shownText.setBackground(TEXT\_AREA\_BACKGROUND\_COLOR);  
 shownText.setForeground(TEXT\_AREA\_FOREGROUND\_COLOR);  
 shownText.setFont(TEXT\_AREA\_FONT);  
 shownText.setEditable(false);  
 shownText.setBorder(*createEmptyBorder*());  
 windowWidth = Math.*round*(width \* 0.9f);  
 windowHeight = Math.*round*(height \* 0.6f);  
 shownText.setPreferredSize(new Dimension(windowWidth, windowHeight)); //Set the size of the TextArea  
  
 owShownTextPanel.add(shownText);  
  
 //Create the buttons area, which will give the player some options  
 owButtonPanel = new JPanel();  
 owButtonGridLayout = new GridLayout(1, 3);  
 owButtonPanel.setLayout(owButtonGridLayout);  
  
 //Create the button for the attack  
 ATTACK.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 ATTACK.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 ATTACK.setFocusPainted(false);  
 ATTACK.setBorder(*createEmptyBorder*());  
 ATTACK.setFont(BUTTON\_FONT);  
  
 //Create the button for the block  
 BLOCK.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 BLOCK.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 BLOCK.setFocusPainted(false);  
 BLOCK.setBorder(*createEmptyBorder*());  
 BLOCK.setFont(BUTTON\_FONT);  
  
 //Create the button for resting  
 REST.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 REST.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 REST.setFocusPainted(false);  
 REST.setBorder(*createEmptyBorder*());  
 REST.setFont(BUTTON\_FONT);  
  
 //Create the button for escape  
 ESCAPE.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 ESCAPE.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 ESCAPE.setFocusPainted(false);  
 ESCAPE.setBorder(*createEmptyBorder*());  
 ESCAPE.setFont(BUTTON\_FONT);  
 ESCAPE.addActionListener(e -> cardLayout.show(cardPanel, "4"));  
  
 //Add buttons to the owButtonPanel  
 owButtonPanel.add(ATTACK);  
 owButtonPanel.add(BLOCK);  
 owButtonPanel.add(REST);  
 owButtonPanel.add(ESCAPE);  
  
  
 //Add Panels to the owCenterPanel  
 owCenterPanel.add(owShownTextPanel);  
 owCenterPanel.add(owButtonPanel);  
  
  
 */\*\*  
 \* The Page ends as a blank Panel  
 \* it's at the PAGE\_END  
 \*/* owEndPanel = new JPanel();  
 owEndPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowHeight = Math.*round*(height \* 0.1f);  
 owEndPanel.setPreferredSize(new Dimension(width, windowHeight));  
  
 */\*\*  
 \* Adding everything together  
 \* return the openWorldPanel  
 \*/* openWorldPanel.add(owStartPanel, BorderLayout.*PAGE\_START*);  
 openWorldPanel.add(owCenterPanel, BorderLayout.*CENTER*);  
 openWorldPanel.add(owEndPanel, BorderLayout.*PAGE\_END*);  
  
 return openWorldPanel;  
 }  
}

## CharCreationPanel

### As of: 23.11.2020

package GUI;  
  
import Game.OpenWorldGame;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class CharCreationPanel {  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 50);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font TEXT\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font TEXT\_FIELD\_FONT = new Font("Verdana", Font.*PLAIN*, 25);  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JTextField title = new JTextField("Charakter erstellen");  
 private final JButton PLAY\_BUTTON = new JButton("Play");  
 private final JButton mButton = new JButton("M");  
 private final JButton wButton = new JButton("W");  
 private final JButton plusButton1 = new JButton("+");  
 private final JButton plusButton2 = new JButton("+");  
 private final JButton plusButton3 = new JButton("+");  
 private final JButton minusButton1 = new JButton("-");  
 private final JButton minusButton2 = new JButton("-");  
 private final JButton minusButton3 = new JButton("-");  
 private final JLabel nameLabel = new JLabel("Name:");  
 private final JLabel pointsLeftLabel = new JLabel("Übrige Punkte:");  
 private final JLabel sexLabel = new JLabel("Geschlecht:");  
 private final JLabel lifeLabel = new JLabel("Leben:");  
 private final JLabel staminaLabel = new JLabel("Ausdauer");  
 private final JLabel damageLabel = new JLabel("Schaden");  
  
 private BorderLayout charCreationBorderLayout;  
 private GridLayout charCreationGridLayout;  
  
 private JPanel nameField, sexField, pointsLeftField, dummyField, lifeField, damageField, staminaField;  
 private JTextField nameTextField, pointsLeftTextField, lifeTextField, staminaTextField, damageTextField;  
  
  
 private String name = "John";  
 private String sex = "m";  
 private int pointsLeft = 3;  
 private int life = 10;  
 private int stamina = 10;  
 private int damage = 10;  
  
 public JPanel charCreationPanel(JFrame baseFrame, CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 JPanel charCreationPanel = new JPanel();  
 charCreationBorderLayout = new BorderLayout();  
 charCreationPanel.setLayout(charCreationBorderLayout);  
  
 */\*\*  
 \* PAGE\_START  
 \* Title of the Card  
 \*/* title.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 title.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 title.setEditable(false);  
 title.setFont(TITLE\_FONT);  
 title.setHorizontalAlignment(SwingConstants.*CENTER*);  
 title.setBorder(*createEmptyBorder*());  
 title.setPreferredSize(new Dimension(width,Math.*round*(height \* 0.2f)));  
  
  
 */\*\*  
 \* Center in the Border Layout  
 \* Gives the Player the possibility to change the Charaktersetup  
 \*/* JPanel centerCharCreationPanel = new JPanel();  
 charCreationGridLayout = new GridLayout(4, 1);  
 centerCharCreationPanel.setLayout(charCreationGridLayout);  
 centerCharCreationPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 //NameField including nameTextField and nameText to change the name of the charakter  
 nameField = new JPanel();  
 nameField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 nameLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 nameLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 nameLabel.setFont(TEXT\_FONT);  
  
 nameTextField = new JTextField(name);  
 nameTextField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 nameTextField.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 nameTextField.setFont(TEXT\_FIELD\_FONT);  
 nameTextField.setPreferredSize(new Dimension(100,50));  
  
 nameField.add(nameLabel);  
 nameField.add(nameTextField);  
  
 //sexField including Buttons to change the sex  
 sexField = new JPanel();  
 sexField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 sexLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 sexLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 sexLabel.setFont(TEXT\_FONT);  
  
 mButton.setFont(BUTTON\_FONT);  
 mButton.addActionListener(e ->{  
 sex = "m";  
 mButton.setBackground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 mButton.setForeground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 mButton.setEnabled(false);  
 wButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 wButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 wButton.setEnabled(true);  
 if(name.equals("Jane")); nameTextField.setText("John");  
 });  
 wButton.setFont(BUTTON\_FONT);  
 wButton.addActionListener(e ->{  
 sex = "w";  
 mButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 mButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 mButton.setEnabled(true);  
 wButton.setBackground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 wButton.setForeground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 wButton.setEnabled(false);  
 if(name.equals("John")); nameTextField.setText("Jane");  
 });  
  
 sexField.add(sexLabel);  
 sexField.add(mButton);  
 sexField.add(wButton);  
  
 //pointsLeftField including pointsLeftLabel and pointsLeftTextField to inform about points left  
 pointsLeftField = new JPanel();  
 pointsLeftField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 pointsLeftLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 pointsLeftLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 pointsLeftLabel.setFont(TEXT\_FONT);  
  
 pointsLeftTextField = new JTextField("" +pointsLeft);  
 pointsLeftTextField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 pointsLeftTextField.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 pointsLeftTextField.setFont(TEXT\_FIELD\_FONT);  
 pointsLeftTextField.setSize(50,50);  
 pointsLeftTextField.setEditable(false);  
 pointsLeftTextField.setBorder(*createEmptyBorder*());  
  
 pointsLeftField.add(pointsLeftLabel);  
 pointsLeftField.add(pointsLeftTextField);  
  
 //lifeField including points and plusButton and minusButton  
 lifeField = new JPanel();  
 lifeField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 lifeLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 lifeLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 lifeLabel.setFont(TEXT\_FONT);  
  
 lifeTextField = new JTextField("" +life);  
 lifeTextField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 lifeTextField.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 lifeTextField.setFont(TEXT\_FIELD\_FONT);  
 lifeTextField.setSize(50,50);  
 lifeTextField.setEditable(false);  
 lifeTextField.setBorder(*createEmptyBorder*());  
  
 plusButton1.setFont(BUTTON\_FONT);  
 plusButton1.addActionListener(e -> {  
 if (pointsLeft >= 1) {  
 minusButton1.setEnabled(true);  
 pointsLeft--;  
 pointsLeftTextField.setText("" +pointsLeft);  
 life++;  
 lifeTextField.setText("" +life);  
 if (pointsLeft == 0){  
 plusButton1.setEnabled(false);  
 plusButton2.setEnabled(false);  
 plusButton3.setEnabled(false);  
 }  
 }  
 });  
 minusButton1.setFont(BUTTON\_FONT);  
 minusButton1.addActionListener(e -> {  
 if(life >= 9){  
 plusButton1.setEnabled(true);  
 plusButton2.setEnabled(true);  
 plusButton3.setEnabled(true);  
 pointsLeft++;  
 pointsLeftTextField.setText("" +pointsLeft);  
 life--;  
 lifeTextField.setText("" +life);  
 if(life==8) minusButton1.setEnabled(false);  
 }  
 });  
  
 lifeField.add(lifeLabel);  
 lifeField.add(lifeTextField);  
 lifeField.add(plusButton1);  
 lifeField.add(minusButton1);  
  
 //damageField including points and plusButton and minusButton  
 damageField = new JPanel();  
 damageField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 damageLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 damageLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 damageLabel.setFont(TEXT\_FONT);  
  
 damageTextField = new JTextField("" +damage);  
 damageTextField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 damageTextField.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 damageTextField.setFont(TEXT\_FIELD\_FONT);  
 damageTextField.setSize(50,50);  
 damageTextField.setEditable(false);  
 damageTextField.setBorder(*createEmptyBorder*());  
  
 plusButton2.setFont(BUTTON\_FONT);  
 plusButton2.addActionListener(e -> {  
 if (pointsLeft >= 1) {  
 minusButton2.setEnabled(true);  
 pointsLeft--;  
 pointsLeftTextField.setText("" +pointsLeft);  
 damage++;  
 damageTextField.setText("" +damage);  
 if (pointsLeft == 0){  
 plusButton1.setEnabled(false);  
 plusButton2.setEnabled(false);  
 plusButton3.setEnabled(false);  
 }  
 }  
 });  
 minusButton2.setFont(BUTTON\_FONT);  
 minusButton2.addActionListener(e -> {  
 if(damage >= 9){  
 plusButton1.setEnabled(true);  
 plusButton2.setEnabled(true);  
 plusButton3.setEnabled(true);  
 pointsLeft++;  
 pointsLeftTextField.setText("" +pointsLeft);  
 damage--;  
 damageTextField.setText("" +damage);  
 if(damage==8) minusButton2.setEnabled(false);  
 }  
 });  
  
 damageField.add(damageLabel);  
 damageField.add(damageTextField);  
 damageField.add(plusButton2);  
 damageField.add(minusButton2);  
  
 //staminaField including points and plusButton and minusButton  
 staminaField = new JPanel();  
 staminaField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 staminaLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 staminaLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 staminaLabel.setFont(TEXT\_FONT);  
  
 staminaTextField = new JTextField("" +stamina);  
 staminaTextField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 staminaTextField.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 staminaTextField.setFont(TEXT\_FIELD\_FONT);  
 staminaTextField.setSize(50,50);  
 staminaTextField.setEditable(false);  
 staminaTextField.setBorder(*createEmptyBorder*());  
  
 plusButton3.setFont(BUTTON\_FONT);  
 plusButton3.addActionListener(e -> {  
 if (pointsLeft >= 1) {  
 minusButton3.setEnabled(true);  
 pointsLeft--;  
 pointsLeftTextField.setText("" +pointsLeft);  
 stamina++;  
 staminaTextField.setText("" +stamina);  
 if (pointsLeft == 0){  
 plusButton1.setEnabled(false);  
 plusButton2.setEnabled(false);  
 plusButton3.setEnabled(false);  
 }  
 }  
 });  
 minusButton3.setFont(BUTTON\_FONT);  
 minusButton3.addActionListener(e -> {  
 if(stamina >= 9){  
 plusButton1.setEnabled(true);  
 plusButton2.setEnabled(true);  
 plusButton3.setEnabled(true);  
 pointsLeft++;  
 pointsLeftTextField.setText("" +pointsLeft);  
 stamina--;  
 staminaTextField.setText("" +stamina);  
 if(stamina==8) minusButton3.setEnabled(false);  
 }  
 });  
  
 staminaField.add(staminaLabel);  
 staminaField.add(staminaTextField);  
 staminaField.add(plusButton3);  
 staminaField.add(minusButton3);  
  
 //dummyField  
 dummyField = new JPanel();  
 dummyField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 //Add the Fields to the centerCharCreationPanel  
 centerCharCreationPanel.add(nameField);  
 centerCharCreationPanel.add(sexField);  
 centerCharCreationPanel.add(pointsLeftField);  
 centerCharCreationPanel.add(dummyField);  
 centerCharCreationPanel.add(lifeField);  
 centerCharCreationPanel.add(damageField);  
 centerCharCreationPanel.add(staminaField);  
  
 */\*\*  
 \* PAGE\_END  
 \* PLAY\_BUTTON setup  
 \*/* PLAY\_BUTTON.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 PLAY\_BUTTON.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 PLAY\_BUTTON.setFont(BUTTON\_FONT);  
 PLAY\_BUTTON.setFocusPainted(false);  
 PLAY\_BUTTON.addActionListener(e -> playGame(baseFrame, width, height));  
 PLAY\_BUTTON.setBorder(*createEmptyBorder*());  
 PLAY\_BUTTON.setPreferredSize(new Dimension(width,Math.*round*(height \* 0.1f)));  
  
  
 */\*\*  
 \* Adding everything together  
 \* return the charCreationPanel  
 \*/* charCreationPanel.add(title, BorderLayout.*PAGE\_START*);  
 charCreationPanel.add(centerCharCreationPanel, BorderLayout.*CENTER*);  
 charCreationPanel.add(PLAY\_BUTTON, BorderLayout.*PAGE\_END*);  
  
 return charCreationPanel;  
 }  
  
 private void playGame(JFrame baseFrame, int width, int height) {  
  
 if(sex.equals("m")) sex = "männlich";  
 else if(sex.equals("w")) sex = "weiblich";  
  
 name = nameTextField.getText();  
  
 OpenWorldGame openWorldGame = new OpenWorldGame();  
 openWorldGame.openWorldGame(name, sex, life, stamina, damage, width, height);  
  
 baseFrame.dispose();  
 }  
}

### As of: 04.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class CharCreationPanel {  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 50);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font TEXT\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font TEXT\_FIELD\_FONT = new Font("Verdana", Font.*PLAIN*, 25);  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JTextField title = new JTextField("Charakter erstellen");  
 private final JButton PLAY\_BUTTON = new JButton("Play");  
 private final JButton mButton = new JButton("M");  
 private final JButton wButton = new JButton("W");  
 private final JButton plusButton1 = new JButton("+");  
 private final JButton plusButton2 = new JButton("+");  
 private final JButton plusButton3 = new JButton("+");  
 private final JButton minusButton1 = new JButton("-");  
 private final JButton minusButton2 = new JButton("-");  
 private final JButton minusButton3 = new JButton("-");  
 private final JLabel nameLabel = new JLabel("Name:");  
 private final JLabel pointsLeftLabel = new JLabel("Übrige Punkte:");  
 private final JLabel sexLabel = new JLabel("Geschlecht:");  
 private final JLabel lifeLabel = new JLabel("Leben:");  
 private final JLabel staminaLabel = new JLabel("Ausdauer");  
 private final JLabel damageLabel = new JLabel("Schaden");  
  
 private BorderLayout charCreationBorderLayout;  
 private GridLayout charCreationGridLayout;  
  
 private JPanel nameField, sexField, pointsLeftField, dummyField, lifeField, damageField, staminaField;  
 private JTextField nameTextField, pointsLeftTextField, lifeTextField, staminaTextField, damageTextField;  
  
 private int pointsLeft = 3;  
 private int life = 10;  
 private int stamina = 10;  
 private int damage = 10;  
  
  
  
 public JPanel CharCreationPanel(CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 JPanel charCreationPanel = new JPanel();  
 charCreationBorderLayout = new BorderLayout();  
 charCreationPanel.setLayout(charCreationBorderLayout);  
  
 */\*\*  
 \* PAGE\_START  
 \* Title of the Card  
 \*/* title.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 title.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 title.setEditable(false);  
 title.setFont(TITLE\_FONT);  
 title.setHorizontalAlignment(SwingConstants.*CENTER*);  
 title.setBorder(*createEmptyBorder*());  
 title.setPreferredSize(new Dimension(width,Math.*round*(height \* 0.2f)));  
  
  
 */\*\*  
 \* Center in the Border Layout  
 \* Gives the Player the possibility to change the Charaktersetup  
 \*/* JPanel centerCharCreationPanel = new JPanel();  
 charCreationGridLayout = new GridLayout(4, 1);  
 centerCharCreationPanel.setLayout(charCreationGridLayout);  
 centerCharCreationPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 //NameField including nameTextField and nameText to change the name of the charakter  
 nameField = new JPanel();  
 nameField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 nameLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 nameLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 nameLabel.setFont(TEXT\_FONT);  
  
 nameTextField = new JTextField("John");  
 nameTextField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 nameTextField.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 nameTextField.setFont(TEXT\_FIELD\_FONT);  
  
 nameField.add(nameLabel);  
 nameField.add(nameTextField);  
  
 //sexField including Buttons to change the sex  
 sexField = new JPanel();  
 sexField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 sexLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 sexLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 sexLabel.setFont(TEXT\_FONT);  
  
 mButton.setFont(BUTTON\_FONT);  
 wButton.setFont(BUTTON\_FONT);  
  
 sexField.add(sexLabel);  
 sexField.add(mButton);  
 sexField.add(wButton);  
  
 //pointsLeftField including pointsLeftLabel and pointsLeftTextField to inform about points left  
 pointsLeftField = new JPanel();  
 pointsLeftField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 pointsLeftLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 pointsLeftLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 pointsLeftLabel.setFont(TEXT\_FONT);  
  
 pointsLeftTextField = new JTextField("" +pointsLeft);  
 pointsLeftTextField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 pointsLeftTextField.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 pointsLeftTextField.setFont(TEXT\_FIELD\_FONT);  
 pointsLeftTextField.setSize(50,50);  
 pointsLeftTextField.setEditable(false);  
 pointsLeftTextField.setBorder(*createEmptyBorder*());  
  
 pointsLeftField.add(pointsLeftLabel);  
 pointsLeftField.add(pointsLeftTextField);  
  
 //lifeField including points and plusButton and minusButton  
 lifeField = new JPanel();  
 lifeField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 lifeLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 lifeLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 lifeLabel.setFont(TEXT\_FONT);  
  
 lifeTextField = new JTextField("" +life);  
 lifeTextField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 lifeTextField.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 lifeTextField.setFont(TEXT\_FIELD\_FONT);  
 lifeTextField.setSize(50,50);  
 lifeTextField.setEditable(false);  
 lifeTextField.setBorder(*createEmptyBorder*());  
  
 plusButton1.setFont(BUTTON\_FONT);  
 minusButton1.setFont(BUTTON\_FONT);  
  
 lifeField.add(lifeLabel);  
 lifeField.add(lifeTextField);  
 lifeField.add(plusButton1);  
 lifeField.add(minusButton1);  
  
 //damageField including points and plusButton and minusButton  
 damageField = new JPanel();  
 damageField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 damageLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 damageLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 damageLabel.setFont(TEXT\_FONT);  
  
 damageTextField = new JTextField("" +damage);  
 damageTextField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 damageTextField.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 damageTextField.setFont(TEXT\_FIELD\_FONT);  
 damageTextField.setSize(50,50);  
 damageTextField.setEditable(false);  
 damageTextField.setBorder(*createEmptyBorder*());  
  
 plusButton2.setFont(BUTTON\_FONT);  
 minusButton2.setFont(BUTTON\_FONT);  
  
 damageField.add(damageLabel);  
 damageField.add(damageTextField);  
 damageField.add(plusButton2);  
 damageField.add(minusButton2);  
  
 //staminaField including points and plusButton and minusButton  
 staminaField = new JPanel();  
 staminaField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 staminaLabel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 staminaLabel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 staminaLabel.setFont(TEXT\_FONT);  
  
 staminaTextField = new JTextField("" +stamina);  
 staminaTextField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 staminaTextField.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 staminaTextField.setFont(TEXT\_FIELD\_FONT);  
 staminaTextField.setSize(50,50);  
 staminaTextField.setEditable(false);  
 staminaTextField.setBorder(*createEmptyBorder*());  
  
 plusButton3.setFont(BUTTON\_FONT);  
 minusButton3.setFont(BUTTON\_FONT);  
  
 staminaField.add(staminaLabel);  
 staminaField.add(staminaTextField);  
 staminaField.add(plusButton3);  
 staminaField.add(minusButton3);  
  
 //dummyField  
 dummyField = new JPanel();  
 dummyField.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 //Add the Fields to the centerCharCreationPanel  
 centerCharCreationPanel.add(nameField);  
 centerCharCreationPanel.add(sexField);  
 centerCharCreationPanel.add(pointsLeftField);  
 centerCharCreationPanel.add(dummyField);  
 centerCharCreationPanel.add(lifeField);  
 centerCharCreationPanel.add(damageField);  
 centerCharCreationPanel.add(staminaField);  
  
 */\*\*  
 \* PAGE\_END  
 \* PLAY\_BUTTON setup  
 \*/* PLAY\_BUTTON.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 PLAY\_BUTTON.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 PLAY\_BUTTON.setFont(BUTTON\_FONT);  
 PLAY\_BUTTON.setFocusPainted(false);  
 PLAY\_BUTTON.addActionListener(e -> cardLayout.show(cardPanel, "4"));  
 PLAY\_BUTTON.setBorder(*createEmptyBorder*());  
 PLAY\_BUTTON.setPreferredSize(new Dimension(width,Math.*round*(height \* 0.1f)));  
  
  
 */\*\*  
 \* Adding everything together  
 \* return the charCreationPanel  
 \*/* charCreationPanel.add(title, BorderLayout.*PAGE\_START*);  
 charCreationPanel.add(centerCharCreationPanel, BorderLayout.*CENTER*);  
 charCreationPanel.add(PLAY\_BUTTON, BorderLayout.*PAGE\_END*);  
  
 return charCreationPanel;  
 }  
}

## EndPanel

### As of: 04.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class EndPanel {  
  
 private final Font START\_FONT = new Font("Verdana", Font.*BOLD*, 125); //Custom made Font  
 private final Font START\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
 private final Color TITLE\_WON\_COLOR = new Color(1,50,32);  
 private final Color TITLE\_LOST\_COLOR = new Color(139,0,0);  
  
 private BorderLayout mainWindowLayout;  
 private GridLayout returnButtonFlowLayout;  
 private JPanel endWindow, pageStartPanel, titlePanel, returnButtonPanel;  
 private JTextField title;  
 private JButton returnButton;  
  
 private int panelHeight;  
  
 public JPanel EndPanel(CardLayout cardLayout, JPanel cardPanel, int width, int height) {  
 endWindow = new JPanel();  
 endWindow.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 mainWindowLayout = new BorderLayout();  
 endWindow.setLayout(mainWindowLayout);  
  
 //Set the blank PAGE\_START  
 pageStartPanel = new JPanel();  
 pageStartPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 panelHeight = Math.*round*(height \* 0.07f);  
 pageStartPanel.setPreferredSize(new Dimension(width, panelHeight));  
  
 //Set the title  
 titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 title = new JTextField("YOU WON!");  
 title.setBorder(*createEmptyBorder*()); //delete the border of the title TextField  
 title.setEditable(false);  
 title.setFont(START\_FONT);  
 title.setForeground(TITLE\_WON\_COLOR); //a Dark Green as RGB  
 title.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanel.add(title);  
  
 */\*\*  
 \* Set the returnButtonPanel  
 \* With the returnButton you get back to the first Page  
 \*/* returnButtonPanel = new JPanel();  
 returnButtonFlowLayout = new GridLayout(4, 1);  
 returnButtonPanel.setLayout(returnButtonFlowLayout);  
 returnButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 //return Button  
 returnButton = new JButton("Neues Spiel");  
 returnButton.setFont(START\_BUTTON\_FONT);  
 returnButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 returnButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 returnButton.setBorder(*createEmptyBorder*());  
 returnButton.setFocusPainted(false);  
 returnButton.addActionListener(e -> cardLayout.show(cardPanel, "1"));  
 returnButtonPanel.add(returnButton);  
  
 endWindow.add(pageStartPanel, BorderLayout.*PAGE\_START*);  
 endWindow.add(returnButtonPanel, BorderLayout.*PAGE\_END*);  
 endWindow.add(titlePanel, BorderLayout.*CENTER*);  
 return endWindow;  
 }  
  
 public void lost() {  
 title.setText("YOU LOST!");  
 title.setForeground(TITLE\_LOST\_COLOR);  
 }  
}

### As of: 04.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class EndPanel {  
  
 private final Font START\_FONT = new Font("Verdana", Font.*BOLD*, 125); //Custom made Font  
 private final Font START\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
 private final Color TITLE\_WON\_COLOR = new Color(1,50,32);  
 private final Color TITLE\_LOST\_COLOR = new Color(139,0,0);  
  
 private BorderLayout mainWindowLayout;  
 private GridLayout returnButtonFlowLayout;  
 private JPanel endWindow, pageStartPanel, titlePanel, returnButtonPanel;  
 private JTextField title;  
 private JButton returnButton;  
  
 private int panelHeight;  
  
 public JPanel ENDPanel(CardLayout cardLayout, JPanel cardPanel, int width, int height) {  
 endWindow = new JPanel();  
 endWindow.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 mainWindowLayout = new BorderLayout();  
 endWindow.setLayout(mainWindowLayout);  
  
 //Set the blank PAGE\_START  
 pageStartPanel = new JPanel();  
 pageStartPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 panelHeight = Math.*round*(height \* 0.07f);  
 pageStartPanel.setPreferredSize(new Dimension(width, panelHeight));  
  
 //Set the title  
 titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 title = new JTextField("YOU WON!");  
 title.setBorder(*createEmptyBorder*()); //delete the border of the title TextField  
 title.setEditable(false);  
 title.setFont(START\_FONT);  
 title.setForeground(TITLE\_WON\_COLOR); //a Dark Green as RGB  
 title.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanel.add(title);  
  
 */\*\*  
 \* Set the returnButtonPanel  
 \* With the returnButton you get back to the first Page  
 \*/* returnButtonPanel = new JPanel();  
 returnButtonFlowLayout = new GridLayout(4, 1);  
 returnButtonPanel.setLayout(returnButtonFlowLayout);  
 returnButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 //return Button  
 returnButton = new JButton("Neues Spiel");  
 returnButton.setFont(START\_BUTTON\_FONT);  
 returnButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 returnButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 returnButton.setBorder(*createEmptyBorder*());  
 returnButton.setFocusPainted(false);  
 returnButton.addActionListener(e -> cardLayout.show(cardPanel, "1"));  
 returnButtonPanel.add(returnButton);  
  
 endWindow.add(pageStartPanel, BorderLayout.*PAGE\_START*);  
 endWindow.add(returnButtonPanel, BorderLayout.*PAGE\_END*);  
 endWindow.add(titlePanel, BorderLayout.*CENTER*);  
 return endWindow;  
 }  
}

## HelpPanel

### As of: 23.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class HelpPanel {  
  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font TEXT\_FONT = new Font("Verdana", Font.*PLAIN*, 20);  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JButton START\_BUTTON = new JButton("START");  
 private final JButton BACK\_BUTTON = new JButton("ZURÜCK");  
  
 private BorderLayout helpPanelBorderLayout;  
 private GridLayout helpPanelGridLayout;  
  
 private JPanel helpTitelPanel, helpTextPanel, buttonPanel;  
 private JTextArea helpText;  
 private JTextField helpTitel = new JTextField("Hilfe");  
  
 private final float widthHeightMultiplier = 0.75f;  
  
 public JPanel HelpPanel(int width, int height, CardLayout cardLayout, JPanel cardPanel) {  
  
 JPanel helpPanel = new JPanel();  
 helpPanelBorderLayout = new BorderLayout();  
 helpPanel.setLayout(helpPanelBorderLayout);  
 helpPanel.setBackground(Color.*cyan*);  
  
 //Seitentitel  
 helpTitelPanel = new JPanel();  
 helpTitelPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 helpTitelPanel.setLayout(new FlowLayout(FlowLayout.*LEFT*));  
 helpTitel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 helpTitel.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 helpTitel.setFont(TITLE\_FONT);  
 helpTitel.setBorder(*createEmptyBorder*());  
 helpTitelPanel.add(helpTitel);  
  
 //Den Start und ZurückButton richtig setzen und übergabe zu neuen Seiten einstellen  
 buttonPanel = new JPanel();  
 helpPanelGridLayout = new GridLayout(1,2);  
 buttonPanel.setLayout(helpPanelGridLayout);  
  
 START\_BUTTON.setFont(BUTTON\_FONT);  
 START\_BUTTON.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 START\_BUTTON.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 START\_BUTTON.setFocusPainted(false);  
 START\_BUTTON.setBorder(*createEmptyBorder*());  
 START\_BUTTON.addActionListener(e -> cardLayout.show(cardPanel, "3"));  
 BACK\_BUTTON.setFont(BUTTON\_FONT);  
 BACK\_BUTTON.setBorder(*createEmptyBorder*());  
 BACK\_BUTTON.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 BACK\_BUTTON.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 BACK\_BUTTON.setFocusPainted(false);  
 BACK\_BUTTON.addActionListener(e -> cardLayout.show(cardPanel, "1"));  
  
 buttonPanel.add(BACK\_BUTTON);  
 buttonPanel.add(START\_BUTTON);  
  
 //Textfeld erstellen. Text wird später hinzugefügt  
 helpTextPanel = new JPanel();  
 helpTextPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 helpText = new JTextArea(writeText());  
 helpText.setEditable(false);  
 helpText.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 helpText.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 helpText.setFont(TEXT\_FONT);  
 width = Math.*round*(width \* widthHeightMultiplier);  
 height = Math.*round*(height \* widthHeightMultiplier);  
 helpText.setPreferredSize(new Dimension(width, height));  
  
 //Zeilenumbrüche  
 helpText.setLineWrap(true);  
 helpText.setWrapStyleWord(true);  
  
 helpTextPanel.add(helpText);  
  
 helpPanel.add(helpTitelPanel, BorderLayout.*PAGE\_START*);  
 helpPanel.add(helpTextPanel, BorderLayout.*CENTER*);  
 helpPanel.add(buttonPanel, BorderLayout.*PAGE\_END*);  
  
 return helpPanel;  
 }  
  
 public String writeText() {  
 String text = "";  
  
 text = "Willkommen beim Spiel Growth.\n\n";  
 text += "Es handelt sich um ein Textbasiertes RPG, bei dem der Spieler erstellt werden kann.\n";  
 text += "Der Name kann nach dem Wählen des geschlechts geändert werden.\n";  
 text += "Gespielt wird über die Buttons.\n";  
 text += "Es gibt verschiedene, deren Bezeichnungen die jeweilige Aktion ausführen.\n";  
 text += "Beim Erreichen von Level 3 gewinnt der Spieler.\n";  
 text += "Wenn das Leben auf oder unter 0 fällt verliert der Spieler\n";  
 text += "Beim Ändern der Bildschirmgröße wird das Spiel zurückgesetzt.\n";  
 text += "Sollte der Spielstand nicht gespeichert sein, geht dieser verloren.\n\n";  
 text += "Nun viel Spaß mit dem Spiel!";  
  
 return text;  
 }  
}

### As of: 29.10.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
public class HelpPanel {  
  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font TEXT\_FONT = new Font("Verdane", Font.*PLAIN*, 15);  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JButton START\_BUTTON = new JButton("START");  
 private final JButton BACK\_BUTTON = new JButton("ZURÜCK");  
  
 private BorderLayout helpPanelBorderLayout;  
 private GridLayout helpPanelGridLayout;  
  
 private JPanel buttonPanel;  
 private JTextArea helpText;  
  
 public JPanel HelpPanel(CardLayout cardLayout, JPanel cardPanel) {  
 JPanel helpPanel = new JPanel();  
 helpPanelBorderLayout = new BorderLayout();  
 helpPanel.setLayout(helpPanelBorderLayout);  
 helpPanel.setBackground(Color.*cyan*);  
  
 //Den Start und ZurückButton richtig setzen und übergabe zu neuen Seiten einstellen  
 buttonPanel = new JPanel();  
 helpPanelGridLayout = new GridLayout(1,2);  
 buttonPanel.setLayout(helpPanelGridLayout);  
  
 START\_BUTTON.setFont(BUTTON\_FONT);  
 START\_BUTTON.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 START\_BUTTON.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 START\_BUTTON.setFocusPainted(false);  
 START\_BUTTON.addActionListener(e -> cardLayout.show(cardPanel, "3"));  
 BACK\_BUTTON.setFont(BUTTON\_FONT);  
 BACK\_BUTTON.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 BACK\_BUTTON.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 BACK\_BUTTON.setFocusPainted(false);  
 BACK\_BUTTON.addActionListener(e -> cardLayout.show(cardPanel, "1"));  
  
 buttonPanel.add(BACK\_BUTTON);  
 buttonPanel.add(START\_BUTTON);  
  
 //Textfeld erstellen. Text wird später hinzugefügt  
 helpText = new JTextArea();  
 helpText.setEditable(false);  
 helpText.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 helpText.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 helpText.setFont(TEXT\_FONT);  
  
 helpPanel.add(helpText, BorderLayout.*CENTER*);  
 helpPanel.add(buttonPanel, BorderLayout.*PAGE\_END*);  
  
 return helpPanel;  
 }  
  
 public void writeText(String text) {  
 helpText.setText(text);  
 }  
}

## LoadPanel

### Release: 26.11.2020

package GUI;  
  
import Game.OpenWorldGame;  
import Game.SaveGame;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class LoadPanel {  
  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 35); //Custom made Font  
 private final Font BACK\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JLabel loadTitle = new JLabel("LADEN");  
 private final JButton backButton = new JButton("Zurück");  
  
 private JButton load1Button = new JButton("Spielstand 1");  
 private JButton load2Button = new JButton("Spielstand 2");  
 private JButton load3Button = new JButton("Spielstand 3");  
  
 private BorderLayout menuWindowLayout;  
 private GridLayout chooseButtonGridLayout;  
 private FlowLayout backButtonFlowLayout, titlePanelFlowLayout;  
  
 private JPanel loadPanel, titlePanel, centerWindowPanel, backButtonPanel;  
  
 private boolean load = false;  
 private int topPanelHeight, bottomPanelHeight;  
  
 public JPanel LoadPanel(OpenWorldGame openWorldGame, JFrame baseFrame, CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 loadPanel = new JPanel();  
 loadPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 menuWindowLayout = new BorderLayout();  
 loadPanel.setLayout(menuWindowLayout);  
  
 */\*\*  
 \* Add the title  
 \* BorderLayout PAGE\_START  
 \*/* titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanelFlowLayout = new FlowLayout();  
 titlePanelFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 titlePanel.setLayout(titlePanelFlowLayout);  
 topPanelHeight = Math.*round*(height \* 0.07f);  
 titlePanel.setPreferredSize(new Dimension(width, topPanelHeight));  
  
 loadTitle.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 loadTitle.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 loadTitle.setFont(TITLE\_FONT);  
 loadTitle.setBorder(*createEmptyBorder*());  
 loadTitle.setSize(100, topPanelHeight);  
  
 titlePanel.add(loadTitle);  
  
  
 */\*\*  
 \* Check if there are saveFiles  
 \* each is individually checked  
 \*/* SaveGame saveGame = new SaveGame();  
  
 load = saveGame.FolderExists();  
 if(load){  
 for (int i = 1; i <= 3; i++) {  
 load = saveGame.SaveExists(i);  
 if(load) changeText(i, "Spielstand " +i);  
 else changeText(i, "Kein Spielstand");  
 }  
 } else{  
 load1Button.setText("Kein Spielstand");  
 load2Button.setText("Kein Spielstand");  
 load3Button.setText("Kein Spielstand");  
 }  
  
  
 */\*\*  
 \* Add the different Buttons  
 \* load1Button, load2Button, load3Button  
 \* BorderLayout CENTER  
 \*/* centerWindowPanel = new JPanel();  
 centerWindowPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 chooseButtonGridLayout = new GridLayout(3, 1);  
 centerWindowPanel.setLayout(chooseButtonGridLayout);  
  
 //Add the load1Button  
 load1Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 load1Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 load1Button.setFont(BUTTON\_FONT);  
 load1Button.setBorder(*createEmptyBorder*());  
 load1Button.setFocusPainted(false);  
 load1Button.addActionListener(e -> {  
 openWorldGame.loadMyGame(1, width, height);  
 baseFrame.dispose();  
 });  
  
 //Add the load2Button  
 load2Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 load2Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 load2Button.setFont(BUTTON\_FONT);  
 load2Button.setBorder(*createEmptyBorder*());  
 load2Button.setFocusPainted(false);  
 load2Button.addActionListener(e -> {  
 openWorldGame.loadMyGame(2, width, height);  
 baseFrame.dispose();  
 });  
  
 //Add the load3Button  
 load3Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 load3Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 load3Button.setFont(BUTTON\_FONT);  
 load3Button.setBorder(*createEmptyBorder*());  
 load3Button.setFocusPainted(false);  
 load3Button.addActionListener(e -> {  
 openWorldGame.loadMyGame(3, width, height);  
 baseFrame.dispose();  
 });  
  
 //Add the three buttons above to the panel  
 centerWindowPanel.add(load1Button);  
 centerWindowPanel.add(load2Button);  
 centerWindowPanel.add(load3Button);  
  
  
 */\*\*  
 \* Add the backButton  
 \* BorderLayout PAGE\_END  
 \*/* backButtonPanel = new JPanel();  
 backButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButtonFlowLayout = new FlowLayout();  
 backButtonFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 backButtonPanel.setLayout(backButtonFlowLayout);  
 bottomPanelHeight = Math.*round*(height \* 0.07f);  
 backButtonPanel.setPreferredSize(new Dimension(width, bottomPanelHeight));  
  
 backButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 backButton.setFont(BACK\_BUTTON\_FONT);  
 backButton.setBorder(*createEmptyBorder*());  
 backButton.setSize(100,topPanelHeight);  
 backButton.setFocusPainted(false);  
 backButton.addActionListener(e -> cardLayout.show(cardPanel, "7"));  
  
 backButtonPanel.add(backButton);  
  
 */\*\*  
 \* Add the three main Panels, title, center and back together  
 \* return the menuPanel  
 \*/* loadPanel.add(titlePanel, BorderLayout.*PAGE\_START*);  
 loadPanel.add(backButtonPanel, BorderLayout.*PAGE\_END*);  
 loadPanel.add(centerWindowPanel, BorderLayout.*CENTER*);  
  
 return loadPanel;  
 }  
  
 public void changeText(int i, String text) {  
  
 switch(i) {  
 case 1: load1Button.setText(text);  
 break;  
 case 2: load2Button.setText(text);  
 break;  
 case 3: load3Button.setText(text);  
 }  
 }  
}

### As of: 23.11.2020

package GUI;  
  
import Game.OpenWorldGame;  
import Game.SaveGame;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class LoadPanel {  
  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 35); //Custom made Font  
 private final Font BACK\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JLabel loadTitle = new JLabel("LADEN");  
 private final JButton backButton = new JButton("Zurück");  
  
 private JButton load1Button = new JButton("Spielstand 1");  
 private JButton load2Button = new JButton("Spielstand 2");  
 private JButton load3Button = new JButton("Spielstand 3");  
  
 private BorderLayout menuWindowLayout;  
 private GridLayout chooseButtonGridLayout;  
 private FlowLayout backButtonFlowLayout, titlePanelFlowLayout;  
  
 private JPanel loadPanel, titlePanel, centerWindowPanel, backButtonPanel;  
  
 private boolean load = false;  
 private int topPanelHeight, bottomPanelHeight;  
  
 public JPanel LoadPanel(OpenWorldGame openWorldGame, JFrame baseFrame, CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 loadPanel = new JPanel();  
 loadPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 menuWindowLayout = new BorderLayout();  
 loadPanel.setLayout(menuWindowLayout);  
  
 */\*\*  
 \* Add the title  
 \* BorderLayout PAGE\_START  
 \*/* titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanelFlowLayout = new FlowLayout();  
 titlePanelFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 titlePanel.setLayout(titlePanelFlowLayout);  
 topPanelHeight = Math.*round*(height \* 0.07f);  
 titlePanel.setPreferredSize(new Dimension(width, topPanelHeight));  
  
 loadTitle.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 loadTitle.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 loadTitle.setFont(TITLE\_FONT);  
 loadTitle.setBorder(*createEmptyBorder*());  
 loadTitle.setSize(100, topPanelHeight);  
  
 titlePanel.add(loadTitle);  
  
  
 */\*\*  
 \* Check if there are saveFiles  
 \* each is individually checked  
 \*/* SaveGame saveGame = new SaveGame();  
  
 load = saveGame.FolderExists();  
 if(load){  
 for (int i = 1; i <= 3; i++) {  
 load = saveGame.SaveExists(i);  
 if(load) changeText(i, "Spielstand " +i);  
 else changeText(i, "Kein Spielstand");  
 }  
 } else{  
 load1Button.setText("Kein Spielstand");  
 load2Button.setText("Kein Spielstand");  
 load3Button.setText("Kein Spielstand");  
 }  
  
  
 */\*\*  
 \* Add the different Buttons  
 \* load1Button, load2Button, load3Button  
 \* BorderLayout CENTER  
 \*/* centerWindowPanel = new JPanel();  
 centerWindowPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 chooseButtonGridLayout = new GridLayout(3, 1);  
 centerWindowPanel.setLayout(chooseButtonGridLayout);  
  
 //Add the load1Button  
 load1Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 load1Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 load1Button.setFont(BUTTON\_FONT);  
 load1Button.setBorder(*createEmptyBorder*());  
 load1Button.setFocusPainted(false);  
 load1Button.addActionListener(e -> {  
 openWorldGame.loadMyGame(1, width, height);  
 baseFrame.dispose();  
 });  
  
 //Add the load2Button  
 load2Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 load2Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 load2Button.setFont(BUTTON\_FONT);  
 load2Button.setBorder(*createEmptyBorder*());  
 load2Button.setFocusPainted(false);  
 load2Button.addActionListener(e -> {  
 openWorldGame.loadMyGame(2, width, height);  
 baseFrame.dispose();  
 });  
  
 //Add the load3Button  
 load3Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 load3Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 load3Button.setFont(BUTTON\_FONT);  
 load3Button.setBorder(*createEmptyBorder*());  
 load3Button.setFocusPainted(false);  
 load3Button.addActionListener(e -> {  
 openWorldGame.loadMyGame(3, width, height);  
 baseFrame.dispose();  
 });  
  
 //Add the three buttons above to the panel  
 centerWindowPanel.add(load1Button);  
 centerWindowPanel.add(load2Button);  
 centerWindowPanel.add(load3Button);  
  
  
 */\*\*  
 \* Add the backButton  
 \* BorderLayout PAGE\_END  
 \*/* backButtonPanel = new JPanel();  
 backButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButtonFlowLayout = new FlowLayout();  
 backButtonFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 backButtonPanel.setLayout(backButtonFlowLayout);  
 bottomPanelHeight = Math.*round*(height \* 0.07f);  
 backButtonPanel.setPreferredSize(new Dimension(width, bottomPanelHeight));  
  
 backButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 backButton.setFont(BACK\_BUTTON\_FONT);  
 backButton.setBorder(*createEmptyBorder*());  
 backButton.setSize(100,topPanelHeight);  
 backButton.setFocusPainted(false);  
 backButton.addActionListener(e -> cardLayout.show(cardPanel, "7"));  
  
 backButtonPanel.add(backButton);  
  
 */\*\*  
 \* Add the three main Panels, title, center and back together  
 \* return the menuPanel  
 \*/* loadPanel.add(titlePanel, BorderLayout.*PAGE\_START*);  
 loadPanel.add(backButtonPanel, BorderLayout.*PAGE\_END*);  
 loadPanel.add(centerWindowPanel, BorderLayout.*CENTER*);  
  
 return loadPanel;  
 }  
  
 public void changeText(int i, String text) {  
  
 switch(i) {  
 case 1: load1Button.setText(text);  
 break;  
 case 2: load2Button.setText(text);  
 break;  
 case 3: load3Button.setText(text);  
 }  
 }  
}

### As of: 05.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class LoadPanel {  
  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 35); //Custom made Font  
 private final Font BACK\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JLabel loadTitle = new JLabel("LADEN");  
 private final JButton load1Button = new JButton("Spielstand 1");  
 private final JButton load2Button = new JButton("Spielstand 2");  
 private final JButton load3Button = new JButton("Spielstand 3");  
 private final JButton backButton = new JButton("Zurück");  
  
 private BorderLayout menuWindowLayout;  
 private GridLayout chooseButtonGridLayout;  
 private FlowLayout backButtonFlowLayout, titlePanelFlowLayout;  
  
 private JPanel loadPanel, titlePanel, centerWindowPanel, backButtonPanel;  
  
 private int topPanelHeight, bottomPanelHeight;  
  
 public JPanel LoadPanel(CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 loadPanel = new JPanel();  
 loadPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 menuWindowLayout = new BorderLayout();  
 loadPanel.setLayout(menuWindowLayout);  
  
 */\*\*  
 \* Add the title  
 \* BorderLayout PAGE\_START  
 \*/* titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanelFlowLayout = new FlowLayout();  
 titlePanelFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 titlePanel.setLayout(titlePanelFlowLayout);  
 topPanelHeight = Math.*round*(height \* 0.07f);  
 titlePanel.setPreferredSize(new Dimension(width, topPanelHeight));  
  
 loadTitle.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 loadTitle.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 loadTitle.setFont(TITLE\_FONT);  
 loadTitle.setBorder(*createEmptyBorder*());  
 loadTitle.setSize(100, topPanelHeight);  
  
 titlePanel.add(loadTitle);  
  
 */\*\*  
 \* Add the different Buttons  
 \* load1Button, load2Button, load3Button  
 \* BorderLayout CENTER  
 \*/* centerWindowPanel = new JPanel();  
 centerWindowPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 chooseButtonGridLayout = new GridLayout(3, 1);  
 centerWindowPanel.setLayout(chooseButtonGridLayout);  
  
 //Add the load1Button  
 load1Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 load1Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 load1Button.setFont(BUTTON\_FONT);  
 load1Button.setBorder(*createEmptyBorder*());  
 load1Button.setFocusPainted(false);  
  
 //Add the load2Button  
 load2Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 load2Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 load2Button.setFont(BUTTON\_FONT);  
 load2Button.setBorder(*createEmptyBorder*());  
 load2Button.setFocusPainted(false);  
  
 //Add the load3Button  
 load3Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 load3Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 load3Button.setFont(BUTTON\_FONT);  
 load3Button.setBorder(*createEmptyBorder*());  
 load3Button.setFocusPainted(false);  
  
 //Add the three buttons above to the panel  
 centerWindowPanel.add(load1Button);  
 centerWindowPanel.add(load2Button);  
 centerWindowPanel.add(load3Button);  
  
  
 */\*\*  
 \* Add the backButton  
 \* BorderLayout PAGE\_END  
 \*/* backButtonPanel = new JPanel();  
 backButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButtonFlowLayout = new FlowLayout();  
 backButtonFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 backButtonPanel.setLayout(backButtonFlowLayout);  
 bottomPanelHeight = Math.*round*(height \* 0.07f);  
 backButtonPanel.setPreferredSize(new Dimension(width, bottomPanelHeight));  
  
 backButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 backButton.setFont(BACK\_BUTTON\_FONT);  
 backButton.setBorder(*createEmptyBorder*());  
 backButton.setSize(100,topPanelHeight);  
 backButton.setFocusPainted(false);  
 backButton.addActionListener(e -> cardLayout.show(cardPanel, "7"));  
  
 backButtonPanel.add(backButton);  
  
 */\*\*  
 \* Add the three main Panels, title, center and back together  
 \* return the menuPanel  
 \*/* loadPanel.add(titlePanel, BorderLayout.*PAGE\_START*);  
 loadPanel.add(backButtonPanel, BorderLayout.*PAGE\_END*);  
 loadPanel.add(centerWindowPanel, BorderLayout.*CENTER*);  
  
 return loadPanel;  
 }  
}

## MainWindow

### Release: 26.11.2020

package GUI;  
  
import Game.OpenWorldGame;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
public class MainWindow {  
  
 GUI.OpenWorldPanel openWorldPanel = new GUI.OpenWorldPanel();  
 GUI.BattlePanel battlePanel = new GUI.BattlePanel();  
 GUI.LoadPanel loadPanel;  
 Game.OpenWorldGame openWorldGame;  
 ChoiceHandler cHandler = new ChoiceHandler();  
 GUI.EndPanel endPanel;  
  
 //Variablen  
 private JFrame baseFrame;  
 private Container baseContainer;  
 private JPanel cardPanel;  
  
 //Panels for the different cards/windows  
 private JPanel startWindow, helpWindow, charCreationWindow, openWorldWindow, endWindow, battleWindow,  
 menuWindow, saveWindow, loadWindow, resizeWindowWindow;  
 private CardLayout cardLayout;  
  
 public boolean isGameStart() {  
 return gameStart;  
 }  
  
 public void setGameStart(boolean gameStart) {  
 this.gameStart = gameStart;  
 }  
  
 private boolean gameStart = false;  
  
  
 public MainWindow(OpenWorldGame openWorldGame, int width, int height, int actualCard) {  
  
 this.openWorldGame = openWorldGame;  
  
 baseFrame = new JFrame();  
 baseFrame.setSize(width,height);  
 baseFrame.setLocationRelativeTo(null); //Start location in the middle of the screen  
 baseFrame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 baseFrame.getContentPane().setBackground(Color.*white*);  
 baseFrame.setLayout(null); //Deactivate the standard Layout  
 baseFrame.setResizable(false); //Deactivates the ability to resize the Window.  
 baseContainer = baseFrame.getContentPane();  
 baseContainer.setLayout(new FlowLayout());  
  
  
 /\*  
 Create the cards as tests with different colors  
 \*/  
 cardLayout = new CardLayout();  
 cardPanel = new JPanel();  
 cardPanel.setLayout(cardLayout);  
  
 GUI.StartPanel startPanel = new GUI.StartPanel();  
 startWindow = startPanel.StartPanel(cardLayout, cardPanel);  
 GUI.HelpPanel helpPanel = new GUI.HelpPanel();  
 helpWindow = helpPanel.HelpPanel(width, height, cardLayout, cardPanel);  
 GUI.CharCreationPanel charCreationPanel = new GUI.CharCreationPanel();  
 charCreationWindow = charCreationPanel.charCreationPanel(baseFrame, cardLayout, cardPanel, width, height);  
 openWorldWindow = openWorldPanel.openWorldPanel(cHandler,cardLayout, cardPanel, width, height);  
 endPanel = new GUI.EndPanel();  
 endWindow = endPanel.EndPanel(cardLayout, cardPanel, width, height);  
 battleWindow = battlePanel.BattlePanel(cHandler ,cardLayout, cardPanel, width, height);  
 GUI.MenuPanel menuPanel = new GUI.MenuPanel();  
 menuWindow = menuPanel.MenuPanel(this, cardLayout, cardPanel, width, height);  
 GUI.SavePanel savePanel = new GUI.SavePanel();  
 saveWindow = savePanel.SavePanel(openWorldGame, cardLayout, cardPanel, width, height);  
 loadPanel = new GUI.LoadPanel();  
 loadWindow = loadPanel.LoadPanel(openWorldGame, baseFrame, cardLayout, cardPanel, width, height);  
 GUI.WindowSizePanel windowSizePanel = new GUI.WindowSizePanel();  
 resizeWindowWindow = windowSizePanel.WindowSizePanel(openWorldGame, baseFrame, cardLayout, cardPanel, width, height);  
  
 cardPanel.add(startWindow, "1");  
 cardPanel.add(helpWindow, "2");  
 cardPanel.add(charCreationWindow, "3");  
 cardPanel.add(openWorldWindow, "4");  
 cardPanel.add(endWindow, "5");  
 cardPanel.add(battleWindow, "6");  
 cardPanel.add(menuWindow, "7");  
 cardPanel.add(saveWindow, "8");  
 cardPanel.add(loadWindow, "9");  
 cardPanel.add(resizeWindowWindow, "10");  
  
 baseContainer.add(cardPanel);  
  
 //Set the given card visible  
 cardLayout.show(cardPanel, "" +actualCard);  
  
 //Make the window visible  
 baseFrame.setVisible(true);  
 }  
  
 public void changeTextOpenWorld(String text) {  
 openWorldPanel.setText(text);  
 }  
  
 public void changeTextBattle(String text) {  
 battlePanel.setText(text);  
 }  
  
 public void youLost() {  
 endPanel.lost();  
 cardLayout.show(cardPanel, "5");  
 setGameStart(false);  
 }  
  
 public void youWon() {  
 cardLayout.show(cardPanel, "5");  
 setGameStart(false);  
 }  
  
 public void battleEnd(String text) {  
 cardLayout.show(cardPanel, "4");  
 openWorldPanel.setText(text);  
 }  
  
 public void escaped() {  
 cardLayout.show(cardPanel, "4");  
 openWorldPanel.setText("Du bist erfolgreich entkommen.\n\nWas möchtest du machen?");  
 }  
  
 public void monsterAttackBlocked(boolean isBlocked) {  
 if(isBlocked) BattlePanel.*counter*(true);  
 else BattlePanel.*counter*(false);  
 }  
  
 public void changeTextLoadPanel(String text, int i) {  
 loadPanel.changeText(i, text);  
 }  
  
 public class ChoiceHandler implements ActionListener {  
 @Override  
 public void actionPerformed(ActionEvent event) {  
 String yourChoice = event.getActionCommand();  
  
 switch (yourChoice) {  
 case "Ausruhen":  
 openWorldGame.doRest("openWorldRest");  
 break;  
 case "Erkunden":  
 int chance = Math.*toIntExact*(Math.*round*((Math.*random*()) \* 5)) + 1;  
 if (chance <= 4){  
 openWorldGame.encounter();  
 cardLayout.show(cardPanel, "6");  
 } else changeTextOpenWorld("Das Erkunden war ereignislos.\n\nWas möchtest du machen?");  
 break;  
 case "Profil":  
 boolean dead = openWorldGame.getProfile();  
 if(dead){  
 youLost();  
 cardLayout.show(cardPanel, "5");  
 }  
 break;  
 case "Angriff":  
 openWorldGame.attack();  
 break;  
 case "Konter":  
 openWorldGame.counterAttack();  
 break;  
 case "Blocken":  
 openWorldGame.block();  
 break;  
 case "Erholen":  
 openWorldGame.doRest("battle");  
 break;  
 case "Fliehen":  
 openWorldGame.tryEscape();  
 break;  
 }  
 }  
  
 }  
}

### As of: 23.11.2020

package GUI;  
  
import Game.OpenWorldGame;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
public class MainWindow {  
  
 //Variablen  
 private JFrame baseFrame;  
 private Container baseContainer;  
 private JPanel cardPanel;  
  
 //Panels for the different cards/windows  
 private JPanel startWindow, helpWindow, charCreationWindow, openWorldWindow, endWindow, battleWindow,  
 menuWindow, saveWindow, loadWindow, resizeWindowWindow;  
 private CardLayout cardLayout;  
  
 GUI.OpenWorldPanel openWorldPanel = new GUI.OpenWorldPanel();  
 GUI.BattlePanel battlePanel = new GUI.BattlePanel();  
 GUI.LoadPanel loadPanel;  
 Game.OpenWorldGame openWorldGame;  
 ChoiceHandler cHandler = new ChoiceHandler();  
 GUI.EndPanel endPanel;  
  
 int width;  
 int height;  
  
 public MainWindow(OpenWorldGame openWorldGame, int width, int height, int actualCard) {  
  
 this.openWorldGame = openWorldGame;  
  
 this.width = width;  
 this. height = height;  
  
 baseFrame = new JFrame();  
 baseFrame.setSize(width,height);  
 baseFrame.setLocationRelativeTo(null); //Start location in the middle of the screen  
 baseFrame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 baseFrame.getContentPane().setBackground(Color.*white*);  
 baseFrame.setLayout(null); //Deactivate the standard Layout  
 baseFrame.setResizable(false); //Deactivates the ability to resize the Window.  
 baseContainer = baseFrame.getContentPane();  
 baseContainer.setLayout(new FlowLayout());  
  
  
 /\*  
 Create the cards as tests with different colors  
 \*/  
 cardLayout = new CardLayout();  
 cardPanel = new JPanel();  
 cardPanel.setLayout(cardLayout);  
  
 GUI.StartPanel startPanel = new GUI.StartPanel();  
 startWindow = startPanel.StartPanel(cardLayout, cardPanel);  
 GUI.HelpPanel helpPanel = new GUI.HelpPanel();  
 helpWindow = helpPanel.HelpPanel(width, height, cardLayout, cardPanel);  
 GUI.CharCreationPanel charCreationPanel = new GUI.CharCreationPanel();  
 charCreationWindow = charCreationPanel.charCreationPanel(baseFrame, cardLayout, cardPanel, width, height);  
 openWorldWindow = openWorldPanel.openWorldPanel(cHandler,cardLayout, cardPanel, width, height);  
 endPanel = new GUI.EndPanel();  
 endWindow = endPanel.EndPanel(cardLayout, cardPanel, width, height);  
 battleWindow = battlePanel.BattlePanel(cHandler ,cardLayout, cardPanel, width, height);  
 GUI.MenuPanel menuPanel = new GUI.MenuPanel();  
 menuWindow = menuPanel.MenuPanel(cardLayout, cardPanel, width, height);  
 GUI.SavePanel savePanel = new GUI.SavePanel();  
 saveWindow = savePanel.SavePanel(openWorldGame, cardLayout, cardPanel, width, height);  
 loadPanel = new GUI.LoadPanel();  
 loadWindow = loadPanel.LoadPanel(openWorldGame, baseFrame, cardLayout, cardPanel, width, height);  
 GUI.WindowSizePanel windowSizePanel = new GUI.WindowSizePanel();  
 resizeWindowWindow = windowSizePanel.WindowSizePanel(openWorldGame, baseFrame, cardLayout, cardPanel, width, height);  
  
 cardPanel.add(startWindow, "1");  
 cardPanel.add(helpWindow, "2");  
 cardPanel.add(charCreationWindow, "3");  
 cardPanel.add(openWorldWindow, "4");  
 cardPanel.add(endWindow, "5");  
 cardPanel.add(battleWindow, "6");  
 cardPanel.add(menuWindow, "7");  
 cardPanel.add(saveWindow, "8");  
 cardPanel.add(loadWindow, "9");  
 cardPanel.add(resizeWindowWindow, "10");  
  
 baseContainer.add(cardPanel);  
  
 //Set the given card visible  
 cardLayout.show(cardPanel, "" +actualCard);  
  
 //Make the window visible  
 baseFrame.setVisible(true);  
 }  
  
 public void changeTextOpenWorld(String text) {  
 openWorldPanel.setText(text);  
 }  
  
 public void changeTextBattle(String text) {  
 battlePanel.setText(text);  
 }  
  
 public void youLost() {  
 endPanel.lost();  
 cardLayout.show(cardPanel, "5");  
 }  
  
 public void youWon() {  
 cardLayout.show(cardPanel, "5");  
 }  
  
 public void battleEnd(String text) {  
 cardLayout.show(cardPanel, "4");  
 openWorldPanel.setText(text);  
 }  
  
 public void escaped() {  
 cardLayout.show(cardPanel, "4");  
 openWorldPanel.setText("Du bist erfolgreich entkommen.\n\nWas möchtest du machen?");  
 }  
  
 public void monsterAttackBlocked(boolean isBlocked) {  
 if(isBlocked) BattlePanel.*counter*(true);  
 else BattlePanel.*counter*(false);  
 }  
  
 public void changeTextLoadPanel(String text, int i) {  
 loadPanel.changeText(i, text);  
 }  
  
 public class ChoiceHandler implements ActionListener {  
 @Override  
 public void actionPerformed(ActionEvent event) {  
 String yourChoice = event.getActionCommand();  
  
 switch (yourChoice) {  
 case "Ausruhen":  
 openWorldGame.doRest("openWorldRest");  
 break;  
 case "Erkunden":  
 int chance = Math.*toIntExact*(Math.*round*((Math.*random*()) \* 5)) + 1;  
 if (chance <= 4){  
 openWorldGame.encounter();  
 cardLayout.show(cardPanel, "6");  
 } else changeTextOpenWorld("Das Erkunden war ereignislos.\n\nWas möchtest du machen?");  
 break;  
 case "Profil":  
 boolean dead = openWorldGame.getProfile();  
 if(dead){  
 youLost();  
 cardLayout.show(cardPanel, "5");  
 }  
 break;  
 case "Angriff":  
 openWorldGame.attack();  
 break;  
 case "Konter":  
 openWorldGame.counterAttack();  
 break;  
 case "Blocken":  
 openWorldGame.block();  
 break;  
 case "Erholen":  
 openWorldGame.doRest("battle");  
 break;  
 case "Fliehen":  
 openWorldGame.tryEscape();  
 break;  
 }  
 }  
  
 }  
}

### As of: 05.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
public class MainWindow {  
  
 //Variablen  
 private JFrame baseFrame;  
 private Container baseContainer;  
 private JPanel cardPanel;  
  
 //Panels for the different cards/windows  
 private JPanel startWindow, helpWindow, charCreationWindow, openWorldWindow, endWindow, battleWindow,  
 menuWindow, saveWindow, loadWindow, resizeWindowWindow;  
 private CardLayout cardLayout;  
  
  
 public MainWindow(int width, int height, int actualCard) {  
  
 baseFrame = new JFrame();  
 baseFrame.setSize(width,height);  
 baseFrame.setLocationRelativeTo(null); //Start location in the middle of the screen  
 baseFrame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 baseFrame.getContentPane().setBackground(Color.*white*);  
 baseFrame.setLayout(null); //Deactivate the standard Layout  
 baseFrame.setResizable(false); //Deactivates the ability to resize the Window.  
 baseContainer = baseFrame.getContentPane();  
 baseContainer.setLayout(new FlowLayout());  
  
  
 /\*  
 Create the cards as tests with different colors  
 \*/  
 cardLayout = new CardLayout();  
 cardPanel = new JPanel();  
 cardPanel.setLayout(cardLayout);  
  
 GUI.StartPanel startPanel = new GUI.StartPanel();  
 startWindow = startPanel.StartPanel(cardLayout, cardPanel);  
 GUI.HelpPanel helpPanel = new GUI.HelpPanel();  
 helpWindow = helpPanel.HelpPanel(cardLayout, cardPanel);  
 GUI.CharCreationPanel charCreationPanel = new GUI.CharCreationPanel();  
 charCreationWindow = charCreationPanel.CharCreationPanel(cardLayout, cardPanel, width, height);  
 GUI.OpenWorldPanel openWorldPanel = new GUI.OpenWorldPanel();  
 openWorldWindow = openWorldPanel.OpenWorldPanel(cardLayout, cardPanel, width, height);  
 GUI.EndPanel endPanel = new GUI.EndPanel();  
 endWindow = endPanel.EndPanel(cardLayout, cardPanel, width, height);  
 GUI.BattlePanel battlePanel = new GUI.BattlePanel();  
 battleWindow = battlePanel.BattlePanel(cardLayout, cardPanel, width, height);  
 GUI.MenuPanel menuPanel = new GUI.MenuPanel();  
 menuWindow = menuPanel.MenuPanel(cardLayout, cardPanel, width, height);  
 GUI.SavePanel savePanel = new GUI.SavePanel();  
 saveWindow = savePanel.SavePanel(cardLayout, cardPanel, width, height);  
 GUI.LoadPanel loadPanel = new GUI.LoadPanel();  
 loadWindow = loadPanel.LoadPanel(cardLayout, cardPanel, width, height);  
 GUI.WindowSizePanel windowSizePanel = new GUI.WindowSizePanel();  
 resizeWindowWindow = windowSizePanel.WindowSizePanel(baseFrame, cardLayout, cardPanel, width, height);  
  
 cardPanel.add(startWindow, "1");  
 cardPanel.add(helpWindow, "2");  
 cardPanel.add(charCreationWindow, "3");  
 cardPanel.add(openWorldWindow, "4");  
 cardPanel.add(endWindow, "5");  
 cardPanel.add(battleWindow, "6");  
 cardPanel.add(menuWindow, "7");  
 cardPanel.add(saveWindow, "8");  
 cardPanel.add(loadWindow, "9");  
 cardPanel.add(resizeWindowWindow, "10");  
  
 baseContainer.add(cardPanel);  
  
 //Set the given card visible  
 cardLayout.show(cardPanel, "" +actualCard);  
  
 //Make the window visible  
 baseFrame.setVisible(true);  
 }  
}

### As of: 04.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
public class MainWindow {  
  
 //Konfiguration  
 private final byte UPPER\_CARD\_LIMIT = 10; //Ändern der Namenskonvention wegen dem Konstanten wert  
 private final byte LOWER\_CARD\_LIMIT = 1;  
  
 //Variablen  
 private JFrame baseFrame;  
 private Container baseContainer;  
 private JPanel testCardLayout, cardPanel;  
  
 //Panels for the different cards/windows  
 private JPanel startWindow, helpWindow, charCreationWindow, openWorldWindow, endWindow, battleWindow,  
 menuWindow, saveWindow, loadWindow, resizeWindowWindow;  
 private CardLayout cardLayout;  
 private int width = 800;  
 private int height = 600;  
 private byte currentCard = 1;  
  
 public MainWindow() {  
  
 baseFrame = new JFrame();  
 baseFrame.setSize(800, 600);  
 baseFrame.setLocationRelativeTo(null); //Start location in the middle of the screen  
 baseFrame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 baseFrame.getContentPane().setBackground(Color.*white*);  
 baseFrame.setLayout(null); //Deactivate the standard Layout  
 baseFrame.setResizable(false); //Deactivates the ability to resize the Window.  
 baseContainer = baseFrame.getContentPane();  
 baseContainer.setLayout(new BorderLayout());  
  
  
 /\*  
 Create the testPanel for switching Cards  
 \*/  
 testCardLayout = new JPanel();  
 testCardLayout.setLayout(new FlowLayout());  
 testCardLayout.setSize(width, height / 10);  
 JButton back = new JButton("Zurück");  
 back.setBackground(Color.*black*);  
 back.setForeground(Color.*white*);  
 back.addActionListener(e -> { //verborgener ActionListener() ersetzt durch lambda  
 if (currentCard > LOWER\_CARD\_LIMIT) {  
 currentCard--;  
 } else currentCard = UPPER\_CARD\_LIMIT;  
  
 cardLayout.show(cardPanel, "" + (currentCard));  
 });  
 back.setFocusPainted(false);  
  
 JButton next = new JButton("Weiter");  
 next.setBackground(Color.*black*);  
 next.setForeground(Color.*white*);  
 next.addActionListener(e -> { //verborgener ActionListener ersetzt durch lambda  
 if (currentCard < UPPER\_CARD\_LIMIT) {  
 currentCard++;  
 } else currentCard = LOWER\_CARD\_LIMIT;  
  
 cardLayout.show(cardPanel, "" + (currentCard));  
 });  
 next.setFocusPainted(false);  
  
  
 testCardLayout.add(back);  
 testCardLayout.add(next);  
  
 baseContainer.add(testCardLayout, BorderLayout.*PAGE\_START*);  
  
 /\*  
 Create the cards as tests with different colors  
 \*/  
 cardLayout = new CardLayout();  
 cardPanel = new JPanel();  
 cardPanel.setLayout(cardLayout);  
 GUI.StartPanel startPanel = new GUI.StartPanel();  
 startWindow = startPanel.StartPanel(cardLayout, cardPanel);  
 //startWindow.setBackground(Color.blue);  
 GUI.HelpPanel helpPanel = new GUI.HelpPanel();  
 helpWindow = helpPanel.HelpPanel(cardLayout, cardPanel);  
 //helpWindow.setBackground(Color.cyan);  
 helpPanel.writeText("Hallo dies ist ein Test!!");  
 GUI.CharCreationPanel charCreationPanel = new GUI.CharCreationPanel();  
 charCreationWindow = charCreationPanel.CharCreationPanel(cardLayout, cardPanel, width, height);  
 GUI.OpenWorldPanel openWorldPanel = new GUI.OpenWorldPanel();  
 openWorldWindow = openWorldPanel.OpenWorldPanel(cardLayout, cardPanel, width, height);  
 GUI.EndPanel endPanel = new GUI.EndPanel();  
 endWindow = endPanel.ENDPanel(cardLayout, cardPanel, width, height);  
 GUI.BattlePanel battlePanel = new GUI.BattlePanel();  
 battleWindow = battlePanel.BattlePanel(cardLayout, cardPanel, width, height);  
 menuWindow = new JPanel();  
 menuWindow.setBackground(Color.*magenta*);  
 saveWindow = new JPanel();  
 saveWindow.setBackground(Color.*red*);  
 loadWindow = new JPanel();  
 loadWindow.setBackground(Color.*yellow*);  
 loadWindow.add(new JTextField("Spielstand laden!"));  
 resizeWindowWindow = new JPanel();  
 resizeWindowWindow.setBackground(Color.*orange*);  
  
 cardPanel.add(startWindow, "1");  
 cardPanel.add(helpWindow, "2");  
 cardPanel.add(charCreationWindow, "3");  
 cardPanel.add(openWorldWindow, "4");  
 cardPanel.add(endWindow, "5");  
 cardPanel.add(battleWindow, "6");  
 cardPanel.add(menuWindow, "7");  
 cardPanel.add(saveWindow, "8");  
 cardPanel.add(loadWindow, "9");  
 cardPanel.add(resizeWindowWindow, "10");  
  
 baseContainer.add(cardPanel, BorderLayout.*CENTER*);  
  
 //Make the window visible  
 baseFrame.setVisible(true);  
 }  
}

### As of: 29.10.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
public class MainWindow {  
  
 //Konfiguration  
 private final byte UPPER\_CARD\_LIMIT = 10; //Ändern der Namenskonvention wegen dem Konstanten wert  
 private final byte LOWER\_CARD\_LIMIT = 1;  
  
 //Variablen  
 private JFrame baseFrame;  
 private Container baseContainer;  
 private JPanel testCardLayout, cardPanel;  
  
 //Panels for the different cards/windows  
 private JPanel startWindow, helpWindow, charCreationWindow, openWorldWindow, endWindow, battleWindow,  
 menuWindow, saveWindow, loadWindow, resizeWindowWindow;  
 private CardLayout cardLayout;  
 private int width = 800;  
 private int height = 600;  
 private byte currentCard = 1;  
  
 public MainWindow() {  
  
 baseFrame = new JFrame();  
 baseFrame.setSize(800, 600);  
 baseFrame.setLocationRelativeTo(null); //Start location in the middle of the screen  
 baseFrame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 baseFrame.getContentPane().setBackground(Color.*white*);  
 baseFrame.setLayout(null); //Deactivate the standard Layout  
 baseFrame.setResizable(false); //Deactivates the ability to resize the Window.  
 baseContainer = baseFrame.getContentPane();  
 baseContainer.setLayout(new BorderLayout());  
  
  
 /\*  
 Create the testPanel for switching Cards  
 \*/  
 testCardLayout = new JPanel();  
 testCardLayout.setLayout(new FlowLayout());  
 testCardLayout.setSize(width, height / 10);  
 JButton back = new JButton("Zurück");  
 back.setBackground(Color.*black*);  
 back.setForeground(Color.*white*);  
 back.addActionListener(e -> { //verborgener ActionListener() ersetzt durch lambda  
 if (currentCard > LOWER\_CARD\_LIMIT) {  
 currentCard--;  
 } else currentCard = UPPER\_CARD\_LIMIT;  
  
 cardLayout.show(cardPanel, "" + (currentCard));  
 });  
 back.setFocusPainted(false);  
  
 JButton next = new JButton("Weiter");  
 next.setBackground(Color.*black*);  
 next.setForeground(Color.*white*);  
 next.addActionListener(e -> { //verborgener ActionListener ersetzt durch lambda  
 if (currentCard < UPPER\_CARD\_LIMIT) {  
 currentCard++;  
 } else currentCard = LOWER\_CARD\_LIMIT;  
  
 cardLayout.show(cardPanel, "" + (currentCard));  
 });  
 next.setFocusPainted(false);  
  
  
 testCardLayout.add(back);  
 testCardLayout.add(next);  
  
 baseContainer.add(testCardLayout, BorderLayout.*PAGE\_START*);  
  
 /\*  
 Create the cards as tests with different colors  
 \*/  
 cardLayout = new CardLayout();  
 cardPanel = new JPanel();  
 cardPanel.setLayout(cardLayout);  
 GUI.StartPanel startPanel = new GUI.StartPanel();  
 startWindow = startPanel.StartPanel(cardLayout, cardPanel);  
 //startWindow.setBackground(Color.blue);  
 GUI.HelpPanel helpPanel = new GUI.HelpPanel();  
 helpWindow = helpPanel.HelpPanel(cardLayout, cardPanel);  
 //helpWindow.setBackground(Color.cyan);  
 helpPanel.writeText("Hallo dies ist ein Test!!");  
 charCreationWindow = new JPanel();  
 charCreationWindow.setBackground(Color.*darkGray*);  
 charCreationWindow.add(new JTextField("Charakter erstellen!"));  
 openWorldWindow = new JPanel();  
 openWorldWindow.setBackground(Color.*gray*);  
 endWindow = new JPanel();  
 endWindow.setBackground(Color.*green*);  
 battleWindow = new JPanel();  
 battleWindow.setBackground(Color.*lightGray*);  
 menuWindow = new JPanel();  
 menuWindow.setBackground(Color.*magenta*);  
 saveWindow = new JPanel();  
 saveWindow.setBackground(Color.*red*);  
 loadWindow = new JPanel();  
 loadWindow.setBackground(Color.*yellow*);  
 loadWindow.add(new JTextField("Spielstand laden!"));  
 resizeWindowWindow = new JPanel();  
 resizeWindowWindow.setBackground(Color.*orange*);  
  
 cardPanel.add(startWindow, "1");  
 cardPanel.add(helpWindow, "2");  
 cardPanel.add(charCreationWindow, "3");  
 cardPanel.add(openWorldWindow, "4");  
 cardPanel.add(endWindow, "5");  
 cardPanel.add(battleWindow, "6");  
 cardPanel.add(menuWindow, "7");  
 cardPanel.add(saveWindow, "8");  
 cardPanel.add(loadWindow, "9");  
 cardPanel.add(resizeWindowWindow, "10");  
  
 baseContainer.add(cardPanel, BorderLayout.*CENTER*);  
  
 //Make the window visible  
 baseFrame.setVisible(true);  
 }  
}

### As of: 24.10.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
public class MainWindow {  
  
 private JFrame baseFrame;  
 private Container baseContainer;  
 private JPanel testCardLayout, cardPanel;  
 //Panels for the different cards/windows  
 private JPanel startWindow, helpWindow, charCreationWindow, openWorldWindow, endWindow, battleWindow,  
 menuWindow, saveWindow, loadWindow, resizeWindowWindow;  
 private CardLayout cardLayout;  
 private int width = 800;  
 private int height = 600;  
 private int currentCard = 1;  
 private final int upperCardLimit = 10;  
 private final int lowerCardLimit = 1;  
  
 public MainWindow() {  
  
 baseFrame = new JFrame();  
 baseFrame.setSize(800,600);  
 baseFrame.setLocationRelativeTo(null); //Start location in the middle of the screen  
 baseFrame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 baseFrame.getContentPane().setBackground(Color.*white*);  
 baseFrame.setLayout(null); //Deactivate the standard Layout  
 baseContainer = baseFrame.getContentPane();  
 baseContainer.setLayout(new BorderLayout());  
  
  
 /\*  
 Create the testPanel for switching Cards  
 \*/  
 testCardLayout = new JPanel();  
 testCardLayout.setLayout(new FlowLayout());  
 testCardLayout.setSize(width, height/10);  
 JButton back = new JButton("Zurück");  
 back.setBackground(Color.*black*);  
 back.setForeground(Color.*white*);  
 back.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 if(currentCard > lowerCardLimit){  
 currentCard--;  
 } else currentCard = upperCardLimit;  
  
 cardLayout.show(cardPanel, "" +(currentCard));  
 }  
 });  
 back.setFocusPainted(false);  
  
 JButton next = new JButton("Weiter");  
 next.setBackground(Color.*black*);  
 next.setForeground(Color.*white*);  
 next.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 if(currentCard < upperCardLimit){  
 currentCard++;  
 } else currentCard = lowerCardLimit;  
  
 cardLayout.show(cardPanel, "" +(currentCard));  
 }  
 });  
 next.setFocusPainted(false);  
  
  
 testCardLayout.add(back);  
 testCardLayout.add(next);  
  
 baseContainer.add(testCardLayout, BorderLayout.*PAGE\_START*);  
  
 /\*  
 Create the cards as tests with different colors  
 \*/  
 cardLayout = new CardLayout();  
 cardPanel = new JPanel();  
 cardPanel.setLayout(cardLayout);  
 startWindow = new JPanel();  
 startWindow.setBackground(Color.*blue*);  
 helpWindow = new JPanel();  
 helpWindow.setBackground(Color.*cyan*);  
 charCreationWindow = new JPanel();  
 charCreationWindow.setBackground(Color.*darkGray*);  
 openWorldWindow = new JPanel();  
 openWorldWindow.setBackground(Color.*gray*);  
 endWindow = new JPanel();  
 endWindow.setBackground(Color.*green*);  
 battleWindow = new JPanel();  
 battleWindow.setBackground(Color.*lightGray*);  
 menuWindow = new JPanel();  
 menuWindow.setBackground(Color.*magenta*);  
 saveWindow = new JPanel();  
 saveWindow.setBackground(Color.*red*);  
 loadWindow = new JPanel();  
 loadWindow.setBackground(Color.*yellow*);  
 resizeWindowWindow = new JPanel();  
 resizeWindowWindow.setBackground(Color.*orange*);  
  
 cardPanel.add(startWindow, "1");  
 cardPanel.add(helpWindow, "2");  
 cardPanel.add(charCreationWindow, "3");  
 cardPanel.add(openWorldWindow, "4");  
 cardPanel.add(endWindow, "5");  
 cardPanel.add(battleWindow, "6");  
 cardPanel.add(menuWindow, "7");  
 cardPanel.add(saveWindow, "8");  
 cardPanel.add(loadWindow, "9");  
 cardPanel.add(resizeWindowWindow, "10");  
  
 baseContainer.add(cardPanel, BorderLayout.*CENTER*);  
  
 //Make the window visible  
 baseFrame.setVisible(true);  
 }  
}

## MenuPanel

### Release: 26.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class MenuPanel {  
  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 35); //Custom made Font  
 private final Font BACK\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JLabel menuTitle = new JLabel("MENÜ");  
 private final JButton saveButton = new JButton("Speichern");  
 private final JButton loadButton = new JButton("Laden");  
 private final JButton windowSizeButton = new JButton("Fenstergröße");  
 private final JButton backButton = new JButton("Zurück");  
  
 private BorderLayout menuWindowLayout;  
 private GridLayout chooseButtonGridLayout;  
 private FlowLayout backButtonFlowLayout, titlePanelFlowLayout;  
  
 private JPanel menuPanel, titlePanel, centerWindowPanel, backButtonPanel;  
  
 private int topPanelHeight, bottomPanelHeight;  
  
 public JPanel MenuPanel(MainWindow mainWindow, CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 menuPanel = new JPanel();  
 menuPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 menuWindowLayout = new BorderLayout();  
 menuPanel.setLayout(menuWindowLayout);  
  
 */\*\*  
 \* Add the title  
 \* BorderLayout PAGE\_START  
 \*/* titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanelFlowLayout = new FlowLayout();  
 titlePanelFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 titlePanel.setLayout(titlePanelFlowLayout);  
 topPanelHeight = Math.*round*(height \* 0.07f);  
 titlePanel.setPreferredSize(new Dimension(width, topPanelHeight));  
  
 menuTitle.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 menuTitle.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 menuTitle.setFont(TITLE\_FONT);  
 menuTitle.setBorder(*createEmptyBorder*());  
 menuTitle.setSize(100, topPanelHeight);  
  
 titlePanel.add(menuTitle);  
  
 */\*\*  
 \* Add the different Buttons  
 \* SaveButton, loadButton, windowSizeButton  
 \* BorderLayout CENTER  
 \*/* centerWindowPanel = new JPanel();  
 centerWindowPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 chooseButtonGridLayout = new GridLayout(3, 1);  
 centerWindowPanel.setLayout(chooseButtonGridLayout);  
  
 //Add the saveButton  
 saveButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 saveButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 saveButton.setFont(BUTTON\_FONT);  
 saveButton.setBorder(*createEmptyBorder*());  
 saveButton.setFocusPainted(false);  
 saveButton.addActionListener(e -> cardLayout.show(cardPanel, "8"));  
  
 //Add the loadButton  
 loadButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 loadButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 loadButton.setFont(BUTTON\_FONT);  
 loadButton.setBorder(*createEmptyBorder*());  
 loadButton.setFocusPainted(false);  
 loadButton.addActionListener(e -> cardLayout.show(cardPanel, "9"));  
  
 //Add the windowSizeButton  
 windowSizeButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowSizeButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 windowSizeButton.setFont(BUTTON\_FONT);  
 windowSizeButton.setBorder(*createEmptyBorder*());  
 windowSizeButton.setFocusPainted(false);  
 windowSizeButton.addActionListener(e -> cardLayout.show(cardPanel, "10"));  
  
 //Add the three buttons above to the panel  
 centerWindowPanel.add(saveButton);  
 centerWindowPanel.add(loadButton);  
 centerWindowPanel.add(windowSizeButton);  
  
  
 */\*\*  
 \* Add the backButton  
 \* BorderLayout PAGE\_END  
 \*/* backButtonPanel = new JPanel();  
 backButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButtonFlowLayout = new FlowLayout();  
 backButtonFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 backButtonPanel.setLayout(backButtonFlowLayout);  
 bottomPanelHeight = Math.*round*(height \* 0.07f);  
 backButtonPanel.setPreferredSize(new Dimension(width, bottomPanelHeight));  
  
 backButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 backButton.setFont(BACK\_BUTTON\_FONT);  
 backButton.setBorder(*createEmptyBorder*());  
 backButton.setSize(100,topPanelHeight);  
 backButton.setFocusPainted(false);  
 backButton.addActionListener(e -> {  
 if(mainWindow.isGameStart()) cardLayout.show(cardPanel, "4");  
 else cardLayout.show(cardPanel, "1");  
 });  
  
 backButtonPanel.add(backButton);  
  
 */\*\*  
 \* Add the three main Panels, title, center and back together  
 \* return the menuPanel  
 \*/* menuPanel.add(titlePanel, BorderLayout.*PAGE\_START*);  
 menuPanel.add(backButtonPanel, BorderLayout.*PAGE\_END*);  
 menuPanel.add(centerWindowPanel, BorderLayout.*CENTER*);  
  
 return menuPanel;  
 }  
}

### As of: 05.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class MenuPanel {  
  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 35); //Custom made Font  
 private final Font BACK\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JLabel menuTitle = new JLabel("MENÜ");  
 private final JButton saveButton = new JButton("Speichern");  
 private final JButton loadButton = new JButton("Laden");  
 private final JButton windowSizeButton = new JButton("Fenstergröße");  
 private final JButton backButton = new JButton("Zurück");  
  
 private BorderLayout menuWindowLayout;  
 private GridLayout chooseButtonGridLayout;  
 private FlowLayout backButtonFlowLayout, titlePanelFlowLayout;  
  
 private JPanel menuPanel, titlePanel, centerWindowPanel, backButtonPanel;  
  
 private int topPanelHeight, bottomPanelHeight;  
  
 public JPanel MenuPanel(CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 menuPanel = new JPanel();  
 menuPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 menuWindowLayout = new BorderLayout();  
 menuPanel.setLayout(menuWindowLayout);  
  
 */\*\*  
 \* Add the title  
 \* BorderLayout PAGE\_START  
 \*/* titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanelFlowLayout = new FlowLayout();  
 titlePanelFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 titlePanel.setLayout(titlePanelFlowLayout);  
 topPanelHeight = Math.*round*(height \* 0.07f);  
 titlePanel.setPreferredSize(new Dimension(width, topPanelHeight));  
  
 menuTitle.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 menuTitle.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 menuTitle.setFont(TITLE\_FONT);  
 menuTitle.setBorder(*createEmptyBorder*());  
 menuTitle.setSize(100, topPanelHeight);  
  
 titlePanel.add(menuTitle);  
  
 */\*\*  
 \* Add the different Buttons  
 \* SaveButton, loadButton, windowSizeButton  
 \* BorderLayout CENTER  
 \*/* centerWindowPanel = new JPanel();  
 centerWindowPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 chooseButtonGridLayout = new GridLayout(3, 1);  
 centerWindowPanel.setLayout(chooseButtonGridLayout);  
  
 //Add the saveButton  
 saveButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 saveButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 saveButton.setFont(BUTTON\_FONT);  
 saveButton.setBorder(*createEmptyBorder*());  
 saveButton.setFocusPainted(false);  
 saveButton.addActionListener(e -> cardLayout.show(cardPanel, "8"));  
  
 //Add the loadButton  
 loadButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 loadButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 loadButton.setFont(BUTTON\_FONT);  
 loadButton.setBorder(*createEmptyBorder*());  
 loadButton.setFocusPainted(false);  
 loadButton.addActionListener(e -> cardLayout.show(cardPanel, "9"));  
  
 //Add the windowSizeButton  
 windowSizeButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowSizeButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 windowSizeButton.setFont(BUTTON\_FONT);  
 windowSizeButton.setBorder(*createEmptyBorder*());  
 windowSizeButton.setFocusPainted(false);  
 windowSizeButton.addActionListener(e -> cardLayout.show(cardPanel, "10"));  
  
 //Add the three buttons above to the panel  
 centerWindowPanel.add(saveButton);  
 centerWindowPanel.add(loadButton);  
 centerWindowPanel.add(windowSizeButton);  
  
  
 */\*\*  
 \* Add the backButton  
 \* BorderLayout PAGE\_END  
 \*/* backButtonPanel = new JPanel();  
 backButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButtonFlowLayout = new FlowLayout();  
 backButtonFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 backButtonPanel.setLayout(backButtonFlowLayout);  
 bottomPanelHeight = Math.*round*(height \* 0.07f);  
 backButtonPanel.setPreferredSize(new Dimension(width, bottomPanelHeight));  
  
 backButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 backButton.setFont(BACK\_BUTTON\_FONT);  
 backButton.setBorder(*createEmptyBorder*());  
 backButton.setSize(100,topPanelHeight);  
 backButton.setFocusPainted(false);  
 backButton.addActionListener(e -> cardLayout.show(cardPanel, "4"));  
  
 backButtonPanel.add(backButton);  
  
 */\*\*  
 \* Add the three main Panels, title, center and back together  
 \* return the menuPanel  
 \*/* menuPanel.add(titlePanel, BorderLayout.*PAGE\_START*);  
 menuPanel.add(backButtonPanel, BorderLayout.*PAGE\_END*);  
 menuPanel.add(centerWindowPanel, BorderLayout.*CENTER*);  
  
 return menuPanel;  
 }  
}

## OpenWorldPanel

### As of: 23.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class OpenWorldPanel {  
  
 private final Font MENU\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 15);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font TEXT\_AREA\_FONT = new Font("Verdana", Font.*PLAIN*, 25);  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color TEXT\_AREA\_BACKGROUND\_COLOR = Color.*black*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
 private final Color TEXT\_AREA\_FOREGROUND\_COLOR = Color.*blue*;  
  
 private final JButton MENU = new JButton("MENÜ");  
 private final JButton EXPLORE = new JButton("Erkunden");  
 private final JButton REST = new JButton("Ausruhen");  
 private final JButton PROFILE = new JButton("Profil");  
  
 private BorderLayout openWorldBorderLayout;  
 private FlowLayout owMenuFlowLayout;  
 private GridLayout owButtonGridLayout;  
 private BoxLayout openWorldBoxLayout;  
  
 private JTextArea shownText;  
 private JPanel owMenuPanel, owCenterPanel, owShownTextPanel, owButtonPanel, owEndPanel;  
  
 private int windowWidth, windowHeight;  
  
 public JPanel openWorldPanel(MainWindow.ChoiceHandler cHandler , CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 JPanel openWorldPanel = new JPanel();  
 openWorldBorderLayout = new BorderLayout();  
 openWorldPanel.setLayout(openWorldBorderLayout);  
  
 */\*\*  
 \* The Button to enter the menu  
 \* it's at the PAGE\_START  
 \*/* owMenuPanel = new JPanel();  
 owMenuPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 owMenuFlowLayout = new FlowLayout();  
 owMenuFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 owMenuPanel.setLayout(owMenuFlowLayout);  
 windowHeight = Math.*round*(height \* 0.07f);  
 owMenuPanel.setPreferredSize(new Dimension(width, windowHeight));  
  
 MENU.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 MENU.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 MENU.setFont(MENU\_BUTTON\_FONT);  
 MENU.setFocusPainted(false);  
 MENU.addActionListener(e -> cardLayout.show(cardPanel, "7"));  
  
 owMenuPanel.add(MENU);  
  
 */\*\*  
 \* The Center of the Panel  
 \* It shows the game texts and gives the Player the options  
 \* explore, rest and profile  
 \* those will trigger the actions  
 \*/* owCenterPanel = new JPanel();  
 owCenterPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 openWorldBoxLayout = new BoxLayout(owCenterPanel, BoxLayout.*Y\_AXIS*);  
 owCenterPanel.setLayout(openWorldBoxLayout);  
  
 //Create the Text area, which will give the text  
 owShownTextPanel = new JPanel();  
 owShownTextPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 shownText = new JTextArea("Testtext");  
 shownText.setBackground(TEXT\_AREA\_BACKGROUND\_COLOR);  
 shownText.setForeground(TEXT\_AREA\_FOREGROUND\_COLOR);  
 shownText.setFont(TEXT\_AREA\_FONT);  
 shownText.setEditable(false);  
 shownText.setBorder(*createEmptyBorder*());  
 windowWidth = Math.*round*(width \* 0.9f);  
 windowHeight = Math.*round*(height \* 0.6f);  
 shownText.setPreferredSize(new Dimension(windowWidth, windowHeight)); //Set the size of the TextArea  
  
 owShownTextPanel.add(shownText);  
  
 //Create the buttons area, which will give the player some options  
 owButtonPanel = new JPanel();  
 owButtonGridLayout = new GridLayout(1, 3);  
 owButtonPanel.setLayout(owButtonGridLayout);  
  
 //Create the button for the exploration  
 EXPLORE.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 EXPLORE.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 EXPLORE.setFocusPainted(false);  
 EXPLORE.setBorder(*createEmptyBorder*());  
 EXPLORE.setFont(BUTTON\_FONT);  
 EXPLORE.addActionListener(cHandler); //e -> cardLayout.show(cardPanel, "6")  
  
 //Create the button for resting  
 REST.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 REST.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 REST.setFocusPainted(false);  
 REST.setBorder(*createEmptyBorder*());  
 REST.setFont(BUTTON\_FONT);  
 REST.addActionListener(cHandler);  
  
 //Create the button for profile  
 PROFILE.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 PROFILE.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 PROFILE.setFocusPainted(false);  
 PROFILE.setBorder(*createEmptyBorder*());  
 PROFILE.setFont(BUTTON\_FONT);  
 PROFILE.addActionListener(cHandler);  
  
 //Add buttons to the owButtonPanel  
 owButtonPanel.add(EXPLORE);  
 owButtonPanel.add(REST);  
 owButtonPanel.add(PROFILE);  
  
  
 //Add Panels to the owCenterPanel  
 owCenterPanel.add(owShownTextPanel);  
 owCenterPanel.add(owButtonPanel);  
  
  
 */\*\*  
 \* The Page ends as a blank Panel  
 \* it's at the PAGE\_END  
 \*/* owEndPanel = new JPanel();  
 owEndPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowHeight = Math.*round*(height \* 0.1f);  
 owEndPanel.setPreferredSize(new Dimension(width, windowHeight));  
  
 */\*\*  
 \* Adding everything together  
 \* return the openWorldPanel  
 \*/* openWorldPanel.add(owMenuPanel, BorderLayout.*PAGE\_START*);  
 openWorldPanel.add(owCenterPanel, BorderLayout.*CENTER*);  
 openWorldPanel.add(owEndPanel, BorderLayout.*PAGE\_END*);  
  
 return openWorldPanel;  
 }  
  
 public void setText(String text){  
 shownText.setText(text);  
 }  
}

## SavePanel

### As of: 23.11.2020

package GUI;  
  
import Game.OpenWorldGame;  
import Game.SaveGame;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class SavePanel {  
  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 35); //Custom made Font  
 private final Font BACK\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JLabel saveTitle = new JLabel("SPEICHERN");  
 private final JButton backButton = new JButton("Zurück");  
  
 private JButton save1Button = new JButton("Spielstand 1");  
 private JButton save2Button = new JButton("Spielstand 2");  
 private JButton save3Button = new JButton("Spielstand 3");  
  
 private BorderLayout menuWindowLayout;  
 private GridLayout chooseButtonGridLayout;  
 private FlowLayout backButtonFlowLayout, titlePanelFlowLayout;  
  
 private JPanel savePanel, titlePanel, centerWindowPanel, backButtonPanel;  
  
 private boolean load = false;  
 private int topPanelHeight, bottomPanelHeight;  
  
 public JPanel SavePanel(OpenWorldGame openWorldGame, CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 savePanel = new JPanel();  
 savePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 menuWindowLayout = new BorderLayout();  
 savePanel.setLayout(menuWindowLayout);  
  
 */\*\*  
 \* Add the title  
 \* BorderLayout PAGE\_START  
 \*/* titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanelFlowLayout = new FlowLayout();  
 titlePanelFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 titlePanel.setLayout(titlePanelFlowLayout);  
 topPanelHeight = Math.*round*(height \* 0.07f);  
 titlePanel.setPreferredSize(new Dimension(width, topPanelHeight));  
  
 saveTitle.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 saveTitle.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 saveTitle.setFont(TITLE\_FONT);  
 saveTitle.setBorder(*createEmptyBorder*());  
 saveTitle.setSize(100, topPanelHeight);  
  
 titlePanel.add(saveTitle);  
  
 */\*\*  
 \* Check if there are saveFiles  
 \* each is individually checked  
 \*/* SaveGame saveGame = new SaveGame();  
  
 load = saveGame.FolderExists();  
 if (load) {  
 for (int i = 1; i <= 3; i++) {  
 load = saveGame.SaveExists(i);  
 if(load) changeText(i, "Spielstand " +i);  
 else changeText(i, "Kein Spielstand");  
 }  
 } else{  
 save1Button.setText("Kein Spielstand");  
 save2Button.setText("Kein Spielstand");  
 save3Button.setText("Kein Spielstand");  
 }  
  
  
 */\*\*  
 \* Add the different Buttons  
 \* SaveButton, loadButton, windowSizeButton  
 \* BorderLayout CENTER  
 \*/* centerWindowPanel = new JPanel();  
 centerWindowPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 chooseButtonGridLayout = new GridLayout(3, 1);  
 centerWindowPanel.setLayout(chooseButtonGridLayout);  
  
 //Add the save1Button  
 save1Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 save1Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 save1Button.setFont(BUTTON\_FONT);  
 save1Button.setBorder(*createEmptyBorder*());  
 save1Button.setFocusPainted(false);  
 save1Button.addActionListener(e -> {  
 String text = openWorldGame.saveMyGame(1);  
 save1Button.setText(text);  
 });  
  
 //Add the save2Button  
 save2Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 save2Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 save2Button.setFont(BUTTON\_FONT);  
 save2Button.setBorder(*createEmptyBorder*());  
 save2Button.setFocusPainted(false);  
 save2Button.addActionListener(e -> {  
 String text = openWorldGame.saveMyGame(2);  
 save2Button.setText(text);  
 });  
  
 //Add the save3Button  
 save3Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 save3Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 save3Button.setFont(BUTTON\_FONT);  
 save3Button.setBorder(*createEmptyBorder*());  
 save3Button.setFocusPainted(false);  
 save3Button.addActionListener(e -> {  
 String text = openWorldGame.saveMyGame(3);  
 save3Button.setText(text);  
 });  
  
 //Add the three buttons above to the panel  
 centerWindowPanel.add(save1Button);  
 centerWindowPanel.add(save2Button);  
 centerWindowPanel.add(save3Button);  
  
  
 */\*\*  
 \* Add the backButton  
 \* BorderLayout PAGE\_END  
 \*/* backButtonPanel = new JPanel();  
 backButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButtonFlowLayout = new FlowLayout();  
 backButtonFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 backButtonPanel.setLayout(backButtonFlowLayout);  
 bottomPanelHeight = Math.*round*(height \* 0.07f);  
 backButtonPanel.setPreferredSize(new Dimension(width, bottomPanelHeight));  
  
 backButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 backButton.setFont(BACK\_BUTTON\_FONT);  
 backButton.setBorder(*createEmptyBorder*());  
 backButton.setSize(100,topPanelHeight);  
 backButton.setFocusPainted(false);  
 backButton.addActionListener(e -> cardLayout.show(cardPanel, "7"));  
  
 backButtonPanel.add(backButton);  
  
 */\*\*  
 \* Add the three main Panels, title, center and back together  
 \* return the menuPanel  
 \*/* savePanel.add(titlePanel, BorderLayout.*PAGE\_START*);  
 savePanel.add(backButtonPanel, BorderLayout.*PAGE\_END*);  
 savePanel.add(centerWindowPanel, BorderLayout.*CENTER*);  
  
 return savePanel;  
 }  
  
 private void changeText(int i, String text) {  
  
 switch (i) {  
 case 1 -> save1Button.setText(text);  
 case 2 -> save2Button.setText(text);  
 case 3 -> save3Button.setText(text);  
 }  
 }  
}

### As of: 05.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class MenuPanel {  
  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 35); //Custom made Font  
 private final Font BACK\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JLabel menuTitle = new JLabel("MENÜ");  
 private final JButton saveButton = new JButton("Speichern");  
 private final JButton loadButton = new JButton("Laden");  
 private final JButton windowSizeButton = new JButton("Fenstergröße");  
 private final JButton backButton = new JButton("Zurück");  
  
 private BorderLayout menuWindowLayout;  
 private GridLayout chooseButtonGridLayout;  
 private FlowLayout backButtonFlowLayout, titlePanelFlowLayout;  
  
 private JPanel menuPanel, titlePanel, centerWindowPanel, backButtonPanel;  
  
 private int topPanelHeight, bottomPanelHeight;  
  
 public JPanel MenuPanel(CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 menuPanel = new JPanel();  
 menuPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 menuWindowLayout = new BorderLayout();  
 menuPanel.setLayout(menuWindowLayout);  
  
 */\*\*  
 \* Add the title  
 \* BorderLayout PAGE\_START  
 \*/* titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanelFlowLayout = new FlowLayout();  
 titlePanelFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 titlePanel.setLayout(titlePanelFlowLayout);  
 topPanelHeight = Math.*round*(height \* 0.07f);  
 titlePanel.setPreferredSize(new Dimension(width, topPanelHeight));  
  
 menuTitle.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 menuTitle.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 menuTitle.setFont(TITLE\_FONT);  
 menuTitle.setBorder(*createEmptyBorder*());  
 menuTitle.setSize(100, topPanelHeight);  
  
 titlePanel.add(menuTitle);  
  
 */\*\*  
 \* Add the different Buttons  
 \* SaveButton, loadButton, windowSizeButton  
 \* BorderLayout CENTER  
 \*/* centerWindowPanel = new JPanel();  
 centerWindowPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 chooseButtonGridLayout = new GridLayout(3, 1);  
 centerWindowPanel.setLayout(chooseButtonGridLayout);  
  
 //Add the saveButton  
 saveButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 saveButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 saveButton.setFont(BUTTON\_FONT);  
 saveButton.setBorder(*createEmptyBorder*());  
 saveButton.setFocusPainted(false);  
 saveButton.addActionListener(e -> cardLayout.show(cardPanel, "8"));  
  
 //Add the loadButton  
 loadButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 loadButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 loadButton.setFont(BUTTON\_FONT);  
 loadButton.setBorder(*createEmptyBorder*());  
 loadButton.setFocusPainted(false);  
 loadButton.addActionListener(e -> cardLayout.show(cardPanel, "9"));  
  
 //Add the windowSizeButton  
 windowSizeButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowSizeButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 windowSizeButton.setFont(BUTTON\_FONT);  
 windowSizeButton.setBorder(*createEmptyBorder*());  
 windowSizeButton.setFocusPainted(false);  
 windowSizeButton.addActionListener(e -> cardLayout.show(cardPanel, "10"));  
  
 //Add the three buttons above to the panel  
 centerWindowPanel.add(saveButton);  
 centerWindowPanel.add(loadButton);  
 centerWindowPanel.add(windowSizeButton);  
  
  
 */\*\*  
 \* Add the backButton  
 \* BorderLayout PAGE\_END  
 \*/* backButtonPanel = new JPanel();  
 backButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButtonFlowLayout = new FlowLayout();  
 backButtonFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 backButtonPanel.setLayout(backButtonFlowLayout);  
 bottomPanelHeight = Math.*round*(height \* 0.07f);  
 backButtonPanel.setPreferredSize(new Dimension(width, bottomPanelHeight));  
  
 backButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 backButton.setFont(BACK\_BUTTON\_FONT);  
 backButton.setBorder(*createEmptyBorder*());  
 backButton.setSize(100,topPanelHeight);  
 backButton.setFocusPainted(false);  
 backButton.addActionListener(e -> cardLayout.show(cardPanel, "4"));  
  
 backButtonPanel.add(backButton);  
  
 */\*\*  
 \* Add the three main Panels, title, center and back together  
 \* return the menuPanel  
 \*/* menuPanel.add(titlePanel, BorderLayout.*PAGE\_START*);  
 menuPanel.add(backButtonPanel, BorderLayout.*PAGE\_END*);  
 menuPanel.add(centerWindowPanel, BorderLayout.*CENTER*);  
  
 return menuPanel;  
 }  
}

## StartPanel

### As of: 29.10.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class StartPanel {  
  
 private final Font START\_FONT = new Font("Verdana", Font.*BOLD*, 125); //Custom made Font  
 private final Font START\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Font HELP\_BUTTON\_FONT = new Font("Verdana", Font.*PLAIN*, 15); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
 private final Color TITLE\_COLOR = new Color(1,50,32);  
  
 private BorderLayout mainWindowLayout;  
 private FlowLayout helpButtonFlowLayout;  
 private GridLayout startSafeButtonFlowLayout;  
 private JPanel startWindow, helpButtonPanel, titlePanel, startButtonPanel;  
 private JTextField title;  
 private JButton helpButton, startButton, loadButton;  
  
 private boolean load = false;  
  
 public JPanel StartPanel(CardLayout cardLayout, JPanel cardPanel) {  
 startWindow = new JPanel();  
 startWindow.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 mainWindowLayout = new BorderLayout();  
 startWindow.setLayout(mainWindowLayout);  
  
 //Set the Button for the help function  
 helpButton = new JButton("Hilfe");  
 helpButton.setSize(50, 10);  
 helpButton.setFont(HELP\_BUTTON\_FONT);  
 helpButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 helpButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 helpButton.setFocusPainted(false);  
 helpButton.addActionListener(e -> cardLayout.show(cardPanel, "2"));  
 helpButtonPanel = new JPanel();  
 helpButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 helpButtonFlowLayout = new FlowLayout();  
 helpButtonFlowLayout.setAlignment(FlowLayout.*RIGHT*);  
 helpButtonPanel.setLayout(helpButtonFlowLayout);  
 helpButtonPanel.add(helpButton);  
  
 //Set the title  
 titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 title = new JTextField("GROWTH");  
 title.setBorder(*createEmptyBorder*()); //delete the border of the title TextField  
 title.setEditable(false);  
 title.setFont(START\_FONT);  
 title.setForeground(TITLE\_COLOR); //a Dark Green as RGB  
 title.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanel.add(title);  
  
 */\*\*  
 \* Set the StartButtonPanel  
 \* The Load Button is deactivated as long as there is no safe-data  
 \*/* startButtonPanel = new JPanel();  
 startSafeButtonFlowLayout = new GridLayout(4, 1);  
 startButtonPanel.setLayout(startSafeButtonFlowLayout);  
 startButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
  
 //Start Button  
 startButton = new JButton("START");  
 startButton.setSize(50,50);  
 startButton.setFont(START\_BUTTON\_FONT);  
 startButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 startButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 startButton.setBorder(*createEmptyBorder*());  
 startButton.setFocusPainted(false);  
 startButton.addActionListener(e -> cardLayout.show(cardPanel, "3"));  
 startButtonPanel.add(startButton);  
  
 //LoadButton  
 loadButton = new JButton("LADEN");  
 loadButton.setFont(START\_BUTTON\_FONT);  
 loadButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 loadButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 loadButton.setBorder(*createEmptyBorder*());  
 loadButton.setFocusPainted(false);  
 loadButton.addActionListener(e -> cardLayout.show(cardPanel, "9"));  
 if(load == false)loadButton.setVisible(false);  
 else loadButton.setVisible(true);  
 startButtonPanel.add(loadButton);  
  
 startWindow.add(helpButtonPanel, BorderLayout.*PAGE\_START*);  
 startWindow.add(startButtonPanel, BorderLayout.*PAGE\_END*);  
 startWindow.add(titlePanel, BorderLayout.*CENTER*);  
 return startWindow;  
 }  
}

## WindowSizePanel

### As of: 23.11.2020

package GUI;  
  
import Game.OpenWorldGame;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class WindowSizePanel {  
  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 30); //Custom made Font  
 private final Font BACK\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JLabel windowSizeChooseButton = new JLabel("Fenstergröße");  
 private final JButton size1Button = new JButton("800 x 600");  
 private final JButton size2Button = new JButton("1200 x 900");  
 private final JButton size3Button = new JButton("1680 x 1050");  
 private final JButton size4Button = new JButton("1920 x 1080");  
 private final JButton backButton = new JButton("Zurück");  
  
 private BorderLayout windowSizeWindowLayout;  
 private GridLayout chooseButtonGridLayout;  
 private FlowLayout backButtonFlowLayout, titlePanelFlowLayout;  
  
 private JPanel windowSizePanel, titlePanel, centerWindowPanel, backButtonPanel;  
  
 private int topPanelHeight, bottomPanelHeight;  
  
  
 public JPanel WindowSizePanel(OpenWorldGame openWorldGame, JFrame baseFrame, CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 windowSizePanel = new JPanel();  
 windowSizePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowSizeWindowLayout = new BorderLayout();  
 windowSizePanel.setLayout(windowSizeWindowLayout);  
  
 */\*\*  
 \* Add the title  
 \* BorderLayout PAGE\_START  
 \*/* titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanelFlowLayout = new FlowLayout();  
 titlePanelFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 titlePanel.setLayout(titlePanelFlowLayout);  
 topPanelHeight = Math.*round*(height \* 0.07f);  
 titlePanel.setPreferredSize(new Dimension(width, topPanelHeight));  
  
 windowSizeChooseButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowSizeChooseButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 windowSizeChooseButton.setFont(TITLE\_FONT);  
 windowSizeChooseButton.setBorder(*createEmptyBorder*());  
 windowSizeChooseButton.setSize(100, topPanelHeight);  
  
 titlePanel.add(windowSizeChooseButton);  
  
 */\*\*  
 \* Add the different Buttons  
 \* sizeXButtons  
 \* BorderLayout CENTER  
 \*/* centerWindowPanel = new JPanel();  
 centerWindowPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 chooseButtonGridLayout = new GridLayout(4, 1);  
 centerWindowPanel.setLayout(chooseButtonGridLayout);  
  
 //Add the size1Button (800 x 600)  
 size1Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 size1Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 size1Button.setFont(BUTTON\_FONT);  
 size1Button.setBorder(*createEmptyBorder*());  
 size1Button.setFocusPainted(false);  
 size1Button.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 baseFrame.setVisible(false);  
  
 SwingUtilities.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 new GUI.MainWindow(openWorldGame, 800, 600,1);  
 }  
 });  
 }  
 });  
  
 //Add the size2Button (1200 x 900)  
 size2Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 size2Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 size2Button.setFont(BUTTON\_FONT);  
 size2Button.setBorder(*createEmptyBorder*());  
 size2Button.setFocusPainted(false);  
 size2Button.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 baseFrame.setVisible(false);  
  
 SwingUtilities.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 new GUI.MainWindow(openWorldGame, 1200, 900,1);  
 }  
 });  
 }  
 });  
  
 //Add the size3Button (1680 x 1050)  
 size3Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 size3Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 size3Button.setFont(BUTTON\_FONT);  
 size3Button.setBorder(*createEmptyBorder*());  
 size3Button.setFocusPainted(false);  
 size3Button.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 baseFrame.setVisible(false);  
  
 SwingUtilities.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 new GUI.MainWindow(openWorldGame,1680, 1050 ,1);  
 }  
 });  
 }  
 });  
  
 //Add the size3Button (1920 x 1080)  
 size4Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 size4Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 size4Button.setFont(BUTTON\_FONT);  
 size4Button.setBorder(*createEmptyBorder*());  
 size4Button.setFocusPainted(false);  
 size4Button.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 baseFrame.setVisible(false);  
  
 SwingUtilities.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 new GUI.MainWindow(openWorldGame,1920, 1080,1);  
 }  
 });  
 }  
 });  
  
  
 //Add the three buttons above to the panel  
 centerWindowPanel.add(size1Button);  
 centerWindowPanel.add(size2Button);  
 centerWindowPanel.add(size3Button);  
 centerWindowPanel.add(size4Button);  
  
  
 */\*\*  
 \* Add the backButton  
 \* BorderLayout PAGE\_END  
 \*/* backButtonPanel = new JPanel();  
 backButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButtonFlowLayout = new FlowLayout();  
 backButtonFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 backButtonPanel.setLayout(backButtonFlowLayout);  
 bottomPanelHeight = Math.*round*(height \* 0.07f);  
 backButtonPanel.setPreferredSize(new Dimension(width, bottomPanelHeight));  
  
 backButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 backButton.setFont(BACK\_BUTTON\_FONT);  
 backButton.setBorder(*createEmptyBorder*());  
 backButton.setSize(100,topPanelHeight);  
 backButton.setFocusPainted(false);  
 backButton.addActionListener(e -> cardLayout.show(cardPanel, "7"));  
  
 backButtonPanel.add(backButton);  
  
 */\*\*  
 \* Add the three main Panels, title, center and back together  
 \* return the menuPanel  
 \*/* windowSizePanel.add(titlePanel, BorderLayout.*PAGE\_START*);  
 windowSizePanel.add(backButtonPanel, BorderLayout.*PAGE\_END*);  
 windowSizePanel.add(centerWindowPanel, BorderLayout.*CENTER*);  
  
 return windowSizePanel;  
 }  
}

### As of: 05.11.2020

package GUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
import static javax.swing.BorderFactory.*createEmptyBorder*;  
  
public class WindowSizePanel {  
  
 private final Font TITLE\_FONT = new Font("Verdana", Font.*BOLD*, 30); //Custom made Font  
 private final Font BACK\_BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25);  
 private final Font BUTTON\_FONT = new Font("Verdana", Font.*BOLD*, 25); //Custom made Font  
 private final Color MAIN\_WINDOW\_BACKGROUND\_COLOR = Color.*white*;  
 private final Color MAIN\_WINDOW\_FOREGROUND\_COLOR = Color.*black*;  
  
 private final JLabel windowSizeChooseButton = new JLabel("Fenstergröße");  
 private final JButton size1Button = new JButton("800 x 600");  
 private final JButton size2Button = new JButton("1200 x 900");  
 private final JButton size3Button = new JButton("1680 x 1050");  
 private final JButton size4Button = new JButton("1920 x 1080");  
 private final JButton backButton = new JButton("Zurück");  
  
 private BorderLayout windowSizeWindowLayout;  
 private GridLayout chooseButtonGridLayout;  
 private FlowLayout backButtonFlowLayout, titlePanelFlowLayout;  
  
 private JPanel windowSizePanel, titlePanel, centerWindowPanel, backButtonPanel;  
  
 private int topPanelHeight, bottomPanelHeight;  
  
  
 public JPanel WindowSizePanel(JFrame baseFrame, CardLayout cardLayout, JPanel cardPanel, int width, int height){  
  
 windowSizePanel = new JPanel();  
 windowSizePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowSizeWindowLayout = new BorderLayout();  
 windowSizePanel.setLayout(windowSizeWindowLayout);  
  
 */\*\*  
 \* Add the title  
 \* BorderLayout PAGE\_START  
 \*/* titlePanel = new JPanel();  
 titlePanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 titlePanelFlowLayout = new FlowLayout();  
 titlePanelFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 titlePanel.setLayout(titlePanelFlowLayout);  
 topPanelHeight = Math.*round*(height \* 0.07f);  
 titlePanel.setPreferredSize(new Dimension(width, topPanelHeight));  
  
 windowSizeChooseButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 windowSizeChooseButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 windowSizeChooseButton.setFont(TITLE\_FONT);  
 windowSizeChooseButton.setBorder(*createEmptyBorder*());  
 windowSizeChooseButton.setSize(100, topPanelHeight);  
  
 titlePanel.add(windowSizeChooseButton);  
  
 */\*\*  
 \* Add the different Buttons  
 \* sizeXButtons  
 \* BorderLayout CENTER  
 \*/* centerWindowPanel = new JPanel();  
 centerWindowPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 chooseButtonGridLayout = new GridLayout(4, 1);  
 centerWindowPanel.setLayout(chooseButtonGridLayout);  
  
 //Add the size1Button (800 x 600)  
 size1Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 size1Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 size1Button.setFont(BUTTON\_FONT);  
 size1Button.setBorder(*createEmptyBorder*());  
 size1Button.setFocusPainted(false);  
 size1Button.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 baseFrame.setVisible(false);  
  
 SwingUtilities.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 new GUI.MainWindow(800, 600,10);  
 }  
 });  
 }  
 });  
  
 //Add the size2Button (1200 x 900)  
 size2Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 size2Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 size2Button.setFont(BUTTON\_FONT);  
 size2Button.setBorder(*createEmptyBorder*());  
 size2Button.setFocusPainted(false);  
 size2Button.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 baseFrame.setVisible(false);  
  
 SwingUtilities.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 new GUI.MainWindow(1200, 900,10);  
 }  
 });  
 }  
 });  
  
 //Add the size3Button (1680 x 1050)  
 size3Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 size3Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 size3Button.setFont(BUTTON\_FONT);  
 size3Button.setBorder(*createEmptyBorder*());  
 size3Button.setFocusPainted(false);  
 size3Button.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 baseFrame.setVisible(false);  
  
 SwingUtilities.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 new GUI.MainWindow(1680, 1050 ,10);  
 }  
 });  
 }  
 });  
  
 //Add the size3Button (1920 x 1080)  
 size4Button.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 size4Button.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 size4Button.setFont(BUTTON\_FONT);  
 size4Button.setBorder(*createEmptyBorder*());  
 size4Button.setFocusPainted(false);  
 size4Button.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 baseFrame.setVisible(false);  
  
 SwingUtilities.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 new GUI.MainWindow(1920, 1080,10);  
 }  
 });  
 }  
 });  
  
 //Add the three buttons above to the panel  
 centerWindowPanel.add(size1Button);  
 centerWindowPanel.add(size2Button);  
 centerWindowPanel.add(size3Button);  
 centerWindowPanel.add(size4Button);  
  
  
 */\*\*  
 \* Add the backButton  
 \* BorderLayout PAGE\_END  
 \*/* backButtonPanel = new JPanel();  
 backButtonPanel.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButtonFlowLayout = new FlowLayout();  
 backButtonFlowLayout.setAlignment(FlowLayout.*LEFT*);  
 backButtonPanel.setLayout(backButtonFlowLayout);  
 bottomPanelHeight = Math.*round*(height \* 0.07f);  
 backButtonPanel.setPreferredSize(new Dimension(width, bottomPanelHeight));  
  
 backButton.setBackground(MAIN\_WINDOW\_BACKGROUND\_COLOR);  
 backButton.setForeground(MAIN\_WINDOW\_FOREGROUND\_COLOR);  
 backButton.setFont(BACK\_BUTTON\_FONT);  
 backButton.setBorder(*createEmptyBorder*());  
 backButton.setSize(100,topPanelHeight);  
 backButton.setFocusPainted(false);  
 backButton.addActionListener(e -> cardLayout.show(cardPanel, "7"));  
  
 backButtonPanel.add(backButton);  
  
 */\*\*  
 \* Add the three main Panels, title, center and back together  
 \* return the menuPanel  
 \*/* windowSizePanel.add(titlePanel, BorderLayout.*PAGE\_START*);  
 windowSizePanel.add(backButtonPanel, BorderLayout.*PAGE\_END*);  
 windowSizePanel.add(centerWindowPanel, BorderLayout.*CENTER*);  
  
 return windowSizePanel;  
 }  
}

# Project package Variables

## Variables

### As of: 23.11.2020

package Variables;  
  
public class Variables {  
  
 public final String START\_PANEL = "1";  
  
 */\*\*  
 \* Ausprobieren ob ich Variablen Öffentlich machen kann und welche  
 \*/*}