REFACTORING DOCUMENT

Refactoring Type	Refactoring Class	Explanation	Screenshot

1. Remove Deadcode	GameEngine	switch case remove deploy and show	Before:
			`case "deploy":`
			`  break;`
			
			`case "show":`
			`  if(d_MapDone==true) {`
			`  showAllPlayerWithArmies();`
			`  d_CpView.setCommandAcknowledgement("\n");`
			`  }else{`
			`  d_CpView.setCommandAcknowledgement("The M`
			`  }`
			`  break;`

2. Move Code Logic	GameModel	Move logic of setting the continent list from startup phase to assign reinforcement	Before:
			`public void startUpPhase() throws Exception {`
			`  if(getAllPlayers().size()>1) {`
			`  d_PlayerQueue.addAll(getAllPlayers());`
			`  List<Country> l_CountryList = new ArrayList<>();`
			`  l_CountryList = (List<Country>) getSelectedMap().getCountryList().clone();`
			`  while (l_CountryList.size() > 0) {`
			`  Random l_Random = new Random();`
			`  int l_Index = l_Random.nextInt(l_CountryList.size());`
			`  setPlayerId(d_PlayerQueue.remove());`
			`  getPlayerId1().addCountry(l_CountryList.get(l_Index));`
			`  d_PlayerQueue.add(d_PlayerID);`
			`  l_CountryList.remove(l_Index);`
			`  }`
			`  for (Player l_Player : getAllPlayers()) {`
			`  l_Player.setContinentsList();`
			`  }`
			
			`  assignReinforcementArmies();`
			After:
			`public void assignReinforcementArmies() throws Exception {`
			`  int l_ContinentValue=0;`
			`  for (Player l_Player : getAllPlayers()) {`
			`  l_Player.setContinentsList();`
			`  }`
			`  if(getAllPlayers().size()>0) {`
			`  for (Player l_Player : getAllPlayers()) {`

3. Substitute Algorithm	Player	Instead of using go to label and comparison we have used .contains() method and flag	Before:
			`public void setContinentsList() {`
			`ArrayList <Continent> l_MapContinents = d_GameModelNew.getSelectedMap().getContinentList();`
			`for(Continent l_MapContinent : l_MapContinents) {`
			`int l_Flag=0;`
			`outerloop:`
			`for(Country l_CountryOfContinent : l_MapContinent.getCountryList()) {`
			`for(Country l_CountryOfPlayer: d_Countries) {`
			`if(!(l_CountryOfPlayer==l_CountryOfContinent)) {`
			`l_Flag =1;break outerloop;`
			`}`
			`}`
			`}`
			`if(l_Flag==0) {`
			`d_Continents.add(l_MapContinent);`
			`}`
			`}`
			`}`
			After:
			`public void setContinentsList() {`
			`ArrayList <Continent> l_MapContinents = d_GameModelNew.getSelectedMap().getContinentList();`
			`for(Continent l_MapContinent : l_MapContinents) {`
			`int l_Flag = 0;`
			`for(Country l_Country : l_MapContinent.getCountryList()) {`
			`if(!d_Countries.contains(l_Country))`
			`l_Flag=1;`
			`}`
			`if(l_Flag==0)`
			`d_Continents.add(l_MapContinent);`
			`}`
			`}`

4. Add Field	Country	Introducing Country owner and creating respective getter setter methods	Before:
			`public class Country {`
			`private static int D_Count = 0;`
			`int d_ID;`
			`String d_Name;`
			`String d_ContinentName;`
			`ArrayList<String> d_Neighbors;`
			`int d_NoOfArmies;`
			After:

			`public class Country {`
			`private static int D_Count = 0;`
			`int d_ID;`
			`String d_Name;`
			`String d_ContinentName;`
			`ArrayList<String> d_Neighbors;`
			`int d_NoOfArmies;`
			`private Player d_Owner;`

5. Remove Method
Remove Field
	Player	Removing field and methods related to Player colour	Before:
			`public class Player {`
			`private String d_PlayerName="";`
			`private int d_PlayerId;`
			`private String d_PlayerColor = "";`
			`private int d_Armies;`
			`private int d_TempArmies;`
			`private int d_ResultInteger;`
			`private ArrayList<Country> d_Countries = new ArrayList<Country>();`
			`private Queue<Order> d_Order = new LinkedList<Order>();`
			`private ArrayList<Continent> d_Continents = new ArrayList<Continent>();`
			`private String d_Result="";`
			`private String d_StringOrder="";`
			`private GameModelNew d_GameModelNew;`
			After:
			`public class Player {`
			`private String d_PlayerName="";`
			`private int d_PlayerId;`
			`private int d_Armies;`
			`private ArrayList<Country> d_Countries = new ArrayList<Country>();`
			`private Queue<Order> d_Order = new LinkedList<Order>();`
			`private ArrayList<Continent> d_Continents = new ArrayList<Continent>();`
			`private String d_Result="";`
			`private String d_StringOrder="";`
			`private GameModelNew d_GameModelNew;`
			`private ArrayList<String> d_Cards = new ArrayList<String>();`
			`private ArrayList<Player> d_NegotiatedPlayers = new ArrayList<Player>();`
			`private boolean d_AtleastOneBattleWon=false;`

6. Using Iterator	GameModelNew	We were using for loop which is not a good	Before:

		practise while removing an object from the list. So, iterator makes sure that concurrent exception is not thrown.	`public void removePlayer(String p_PlayerName) throws Exception {`
			`  Player l_CurrentPlayer;`
			`  boolean l_PlayerFound = false;`
			`  for (Player l_Player:d_PlayerList) {`
			`  l_CurrentPlayer = l_Player;`
			`  if (l_CurrentPlayer.getPlayerName().equalsIgnoreCase(p_Pla`
			`  l_PlayerFound = true;`
			`  d_PlayerList.remove(d_PlayerList.indexOf(l_Player));`
			`  }`
			After:
			`public void removePlayer(String p_PlayerName) throws Exception {`
			`  Iterator<Player> l_Iterator = this.d_PlayerList.iterator();`
			`  boolean l_PlayerFound = false;`
			`  while(l_Iterator.hasNext()) {`
			`  Player l_TempPlayer = l_Iterator.next();`
			`  if(l_TempPlayer.getPlayerName().equalsIgnoreCase(p_Playe`
			`  l_PlayerFound = true;`
			`  l_Iterator.remove();`

7. Move Method	GameEngine	Moving editPlayer method from GameEngine to Player Controller	Before: GameEngine.java GameEngine CommandListener d_CpView d_GameModelNew d_MapController d_PlayerController d_PlayerList GameEngine(CommandPrompt, assignCountries() : void editPlayer(String, String) : String After:

			PlayerController d_AllCards d_CpView d_GameEngine d_GameModelNew d_LEB d_OrderAcknowledgment d_Players d_Rand PlayerController(GameModelNew checkTheWinner() : void clearNegotiatedPlayerList() : void editPlayer(String, String) : String
8. Substitute Algorithm	GameModelNew	Startup phase will add owner of the country when country is assigned	`l_CountryList.get(l_Index).setCountryOwnerPlayer(getPlayerId1());`
9. Improving User Experience	GameEngine, CommandPrompt	Printing Owner and armies assigned in same line, And Scaling the size of the command prompt so that the user need not have to scroll much.	Before: Continent: africa Country: kenya -->Owner: zeal -->Armies deployed: 3 --> Borders : india,pakistan,mosambique,sydney,mel, Country: mosambique -->Owner: raj -->Armies deployed: 0 --> Borders : kenya,sydney,india,pakistan,mel, After:

			Continent: africa Country: kenya-->Owner: zeal-->Armies deployed: 0 --> Borders : india,pakistan,mosambique,sydney,mel, Country: mosambique-->Owner: raj-->Armies deployed: 0 --> Borders : kenya,sydney,india,pakistan,mel,
10. Push Down Method	Order	Pushing down the deploy and execute order from order class to deploy class	Before: Order d_CountryName d_ExecuteResult d_GameModelNew d_NoOfArmies d_Order Order(String, GameModelNew) deploy() : void execute() : void After: Deploy.java Deploy d_Country d_NumArmies d_Player Deploy(Player, Country, int) execute() : void

11. Hide Method	Continent.java	setContinnetID method isn't used by other classes or is used only	Before:
			`public void setContinentID(int p_ContinentID) {`
			`d_ID = p_ContinentID;`
			`}`
			After:

		inside its own class hierarchy.	`private void setContinentID(int p_ContinentID) {`
			`  d_ID = p_ContinentID;`
			`}`

12. Rename Method	GameModelTest	The name of a method doesn't explain what the method does.	Before: testIssueOrder() : void testIssueOrder1() : void testIssueOrder2() : void After: testNonAdjacentTerritory() : void testSourceTargetNeighbours() : void testSourceTargetTerritory() : void

13. Replace magic number with symbolic constant	GameModelNew	Replace this number with a constant that has a human-readable name explaining the meaning of the number.	Before:
			`l_ArmyCount= Math.max(l_ArmyCount, 3);`
			After:
			`l_ArmyCount= Math.max(l_ArmyCount, D_MINARMIES);`

14. Command Pattern	Order, Player, GameEngine	The Command class must be the Order class, the Invoker Class is the Player, and the Client class is the GameEngine.	

15. State Pattern	GameEngine	State pattern to implement the phases of the application	Before:
			`private boolean d_MapDone = false;`
			`private boolean d_StartUpDone = false;`
			`private boolean d_AssignCountriesDone = false;`

		instead of using control variable	After: `private Phase d_GamePhase;`