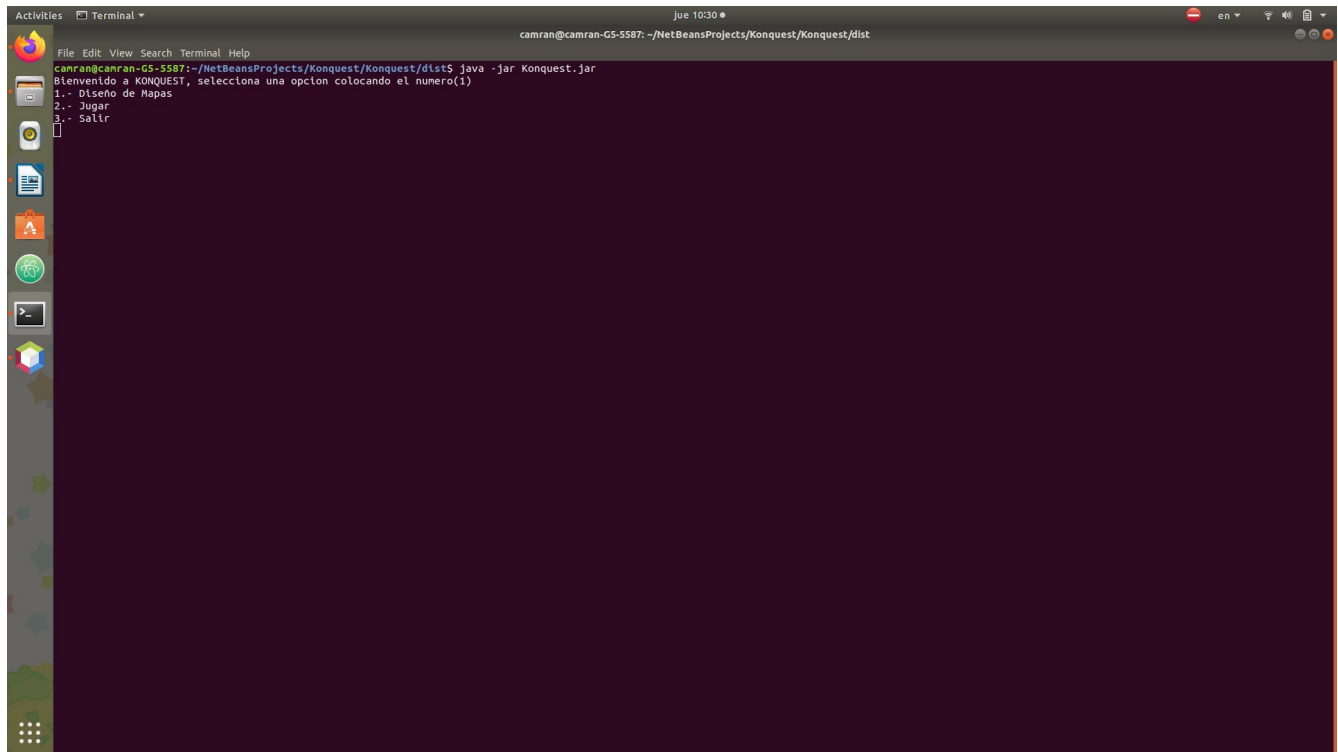
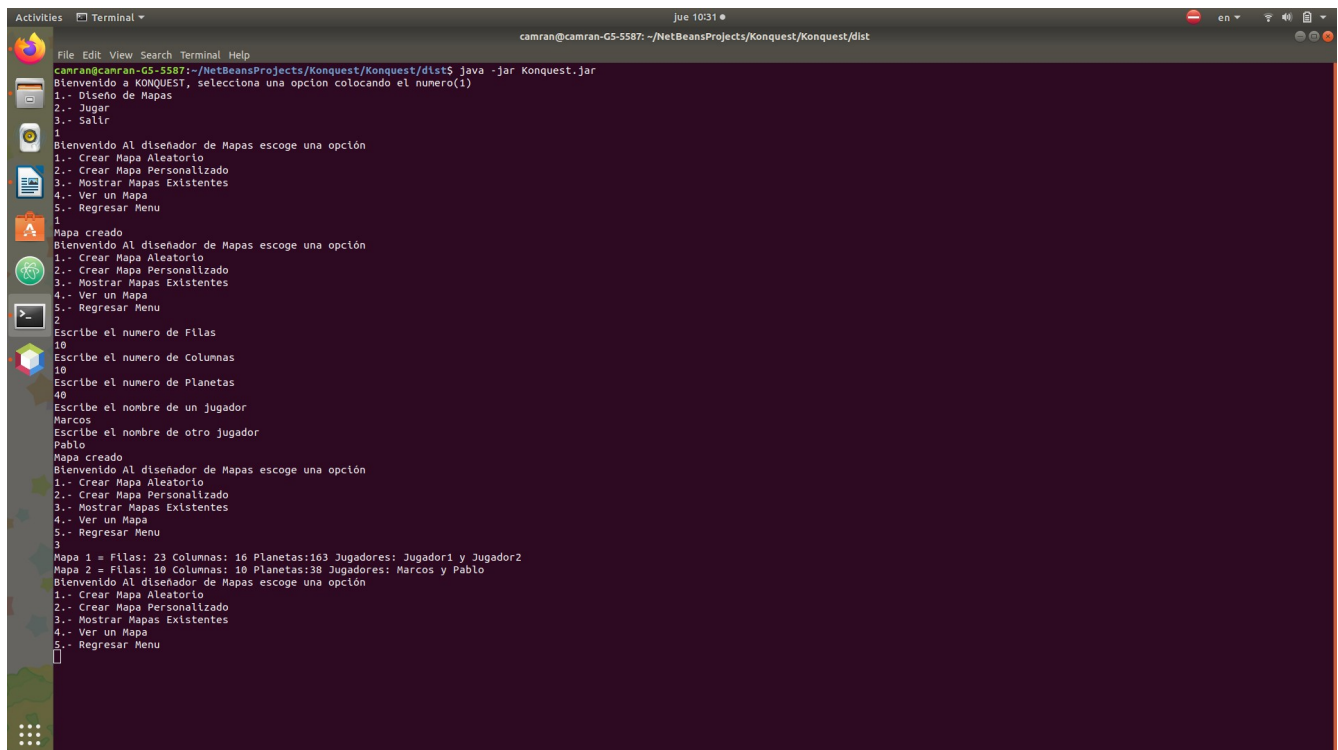


# Manual Técnico

## Demostracion Menu Inicial y Menu Diseño de Mapas



```
Activities Terminal
camran@camran-G5-5587: ~/NetBeansProjects/Konquest/Konquest/dist
camran@camran-G5-5587:~/NetBeansProjects/Konquest/Konquest/dist$ java -jar Konquest.jar
Bienvenido a KONQUEST, selecciona una opción colocando el numero(i)
1.- Diseño de Mapas
2.- Jugar
3.- Salir
1
```



```
Activities Terminal
camran@camran-G5-5587: ~/NetBeansProjects/Konquest/Konquest/dist
camran@camran-G5-5587:~/NetBeansProjects/Konquest/Konquest/dist$ java -jar Konquest.jar
Bienvenido a KONQUEST, selecciona una opción colocando el numero(i)
1.- Diseño de Mapas
2.- Jugar
3.- Salir
1
Bienvenido Al diseñador de Mapas escoge una opción
1.- Crear Mapa Aleatorio
2.- Crear Mapa Personalizado
3.- Mostrar Mapas Existentes
4.- Ver un Mapa
5.- Regresar Menu
1
Mapa creado
Bienvenido Al diseñador de Mapas escoge una opción
1.- Crear Mapa Aleatorio
2.- Crear Mapa Personalizado
3.- Mostrar Mapas Existentes
4.- Ver un Mapa
5.- Regresar Menu
2
Escribe el numero de Filas
10
Escribe el numero de Columnas
10
Escribe el numero de Planetas
40
Escribe el nombre de un jugador
Marcos
Escribe el nombre de otro jugador
Pablo
Mapa creado
Bienvenido Al diseñador de Mapas escoge una opción
1.- Crear Mapa Aleatorio
2.- Crear Mapa Personalizado
3.- Mostrar Mapas Existentes
4.- Ver un Mapa
5.- Regresar Menu
3
Mapa 1 = Filas: 23 Columnas: 16 Planetas:163 Jugadores: Jugador1 y Jugador2
Mapa 2 = Filas: 10 Columnas: 10 Planetas:38 Jugadores: Marcos y Pablo
Bienvenido Al diseñador de Mapas escoge una opción
1.- Crear Mapa Aleatorio
2.- Crear Mapa Personalizado
3.- Mostrar Mapas Existentes
4.- Ver un Mapa
5.- Regresar Menu
1
```

## Forma de Ver Mapa Menu Diseño de Mapas



[illegible][illegible]











```

//Inicializamos el juego
while(true)
    //Accede al menu principal del juego
    choice = handler.getMenu()
    //Opcion para comprobar si se sale del programa
    Si (choice == 3)
        Escribir ("Hasta luego")
        break
    Fin
    handler.choice(choice)

Fin

```

### Inicio del Juego

#### **Clase: Planeta**

Importante clase que se basa la mayoria del programa, que establece el objeto tipo planeta con las variables nombre, estado, tipoPlaneta, porcentajeDeMuerte, valorGalactus, listaGuerreros, listaNaves, listaConstructores

#### **Metodos: Planeta:**

##### Constructor

```

Random random = new Random()
valueGalactus = random.nextvar(401)+100
para (var i=0 i<6 i++)
    name = name + (var) (random.nextvar(26)+65)
Fin

```

##### **Metodo: leftWarriors:**

```

publico booleano leftWarriors(var warriorlength)
Si(warriorlength<warriorList.length)
    Warriors[] warriorAux = new Warriors[warriorlength]
    para (var i=0 i<warriorlength i++)
        warriorAux[i] = warriorList[i]
    Fin
    warriorList = warriorAux
    return true
Fin
return false

```

Fin

Elimina guerreros de la listaGuerreros, y mantiene solo el numero de guerreros indicado por la varibale warriorlength

##### **Metodo: welcomeAllies:**

```

publico void welcomeAllies(Warriors[] warriors)
    var contadora=0
    Warriors[] warriorAux = new Warriors[warriorList.length + warriors.length]

```



```

para (var i=0 i<warriorList.length i++)
    warriorAux[i] = warriorList[i]
Fin
try
Si(warriorList.length >0)
    para (var i=warriorList.length-1 i<warriorAux.length-1 i++)

        warriorAux[i] = warriors[contadora]
        contadora++
    Fin
FinSino
    para (var i=0 i<warriorAux.length-1 i++)

        warriorAux[i] = warriors[contadora]
        contadora++
    Fin
Fin
Fincatch(Throwable e)
    Escribir ("Ocurrio un error")
Fin
Escribir ("Guerreros que se unen: "+warriors.length)
warriorList = warriorAux

```

Fin

Agrega el arreglo de gurreros indicados a lista guerreros

**Metodo: addShip:**

```

public void addShip(Ship ship)
    Ship[] shipAux = new Ship[1 + ships.length]
    para (var i=0 i<ships.length i++)
        shipAux[i] = ships[i]
    Fin
    shipAux[ships.length] = ship
    ships = shipAux

```

Fin

Agrega una nave a la lista

**Metodo: createShip:**

```

public void createShip(Constructors constructor)
    var encontrado=0
    para(var i=0 i<constructorList.length i++)
        Si(constructorList[i].getName() == constructor.getName() && constructorList[i].getState() !=
true)
            Si((valueGalactus - constructorList[i].getShipClass().getPrice()) > 0)
                valueGalactus -= constructorList[i].getShipClass().getPrice()
                constructorList[i].createShip()
                encontrado=1
                break
    Fin
Sino

```

```
        Escribir ("Dineor insuficiente")
    Fin
Fin
Fin
```

```
Si(encontrado==0)
    Escribir ("No se hallo ningun constructor")
FinSino
    Escribir ("Empezando a trabajar")
```

Fin

Envia a un constructor de la lista a empezar a construir una nave

**Metodo: addBuilder:**

```
public void addBuilder(Constructors constructor)
    Constructors[] builderAux = new Constructors[1 + constructorList.length]
    para (var i=0 i<constructorList.length i++)
        builderAux[i] = constructorList[i]
    Fin
    builderAux[constructorList.length] = constructor
    constructorList = builderAux
Fin
```

Agrega un constructor

**Metodo: showShip**

```
public void showShip(var choice)
    Si(choice==0)
        para(var i=0 i<this.ships.length i++)
            Escribir ("Nave "+(i+1)+" ")
            ships[i].getData()
        Fin
    FinSino
        para(var i=0 i<this.ships.length i++)
            Escribir ("Nave "+(i+1)+" ")
            ships[i].getDataOn()
        Fin
    Fin
Fin
```

Obtiene datos de la nave

**Metodo:restart()**

```
public void restart()
    warriorList = new Warriors[0]
    ships = new Ship[1]
    constructorList = new Constructors[1]
    state = null
Fin
```

Reestablece valores

**Metodo: showBuilder**

```
public void showBuilder()  
    para(var i=0 i<this.constructorList.length i++)  
        Escribir ("Constructor "+(i+1)+" ")  
        constructorList[i].getData()  
    Fin  
Fin
```

Muestra datos de los constructores

**Metodo: get"Valor":**

svaraxis general

```
public var getMoney()  
    return valueGalactus
```

Fin

Retorna algun valor del planeta

## Clase Hija: Earth, Fire, Organic, Radiation, Water

Constructor ejemplo Siicacion

```
public Earth()  
    name = name + "EA"  
    typePlanet = "Earth"  
    ships[0] = new Naboo(this)  
    constructorList[0] = new Workman(this)  
Fin
```

Los valores otorgados cambia segun la clase

### Metodos:

#### addWarrior()

#### Svaraxis General

#### Polimorfismo

```
public void addWarrior()  
    Warriors[] warrior = new Mole[random.nextvar(11)+15]  
    Warriors[] warriorListAux  
    warriorListAux = warriorList  
    warriorList = new Mole[warriorListAux.length + warrior.length]  
  
    para (var i=0 i<warrior.length i++)  
        warrior[i] = new Mole(state,deathPercent)  
Fin  
  
    System.arraycopy(warrior, 0, warriorList, 0, warrior.length)  
    System.arraycopy(warriorListAux, 0, warriorList, 0, warriorListAux.length)
```

Fin

Agrega guerreros a la lista el valor depende de la clase con la que se este manejando

#### addGalactus()

#### Svaraxis General

#### Polimorfismo

```
public void addGalactus()  
    valueGalactus += random.nextvar(51)+50
```

Fin

Agrega dinero al planeta

#### restart()

```
public void restart()  
    warriorList = new Warriors[0]  
    ships = new Ship[1]  
    ships[0] = new Naboo(this)  
    constructorList = new Constructors[1]  
    constructorList[0] = new Workman(this)  
    state = null
```

Fin

Reinicia Valores

**Clase: Warriors**

Variables owner, name, deathFactor, timesConfronted, slots

Constructor

```
this.owner = owner
    this.deathFactor = deathFactor
```

**Metodos:****get"Valor":**

Svaraxis General

```
public var getTimesConfronted()
    return timesConfronted
```

Fin

Devuelve el valor de alguna variable

**changeOwner():**

```
this.owner = owner
```

Cambia propietario

**Clase Hija: Mole,Nemo,Magma,Groot,FisionGuy**

Constructor

Svaraxis General

```
public FisionGuy(var owner, var deathFactor)
    super(owner,deathFactor)
    deathFactor *=1.95
    specialAttack = "Rayo Gamma"
    slots= 4
    name = "Fision Guy"
```

Fin

Los valores otorgados cambia segun la clase

**Clase: Constructors**

variables timeTaked, workedDays, price, soldPrice, state, name, shipClass, planet

Constructor

```
public Constructors(Planets planet)
    this.planet = planet
```

Fin

**Metodos:****get"Valor":**

Svaraxis General

```
public var getTimesConfronted()
    return timesConfronted
```

Fin

Devuelve el valor de alguna variable

**createShip():**

Si(state != true)

```

        state = true
        Escribir ("Empezando a trabajar, me tomara "+ timeTaked + " turnos")
    Fin
    Sino

```

```

        Escribir ("Trabajador ocupado")

```

Empieza con la construccion de una nave, cambiandole el parametro de construccion

**turnPass():**

```

public Ship turnPass()

```

```

    Si (state == true)

```

```

        workedDays+=1

```

```

        Si(workedDays == timeTaked)

```

```

            state = false

```

```

            workedDays = 0

```

```

            Escribir ("Trabajo realizado se ha agregado una nave tipo "+ shipClass.getName() + "\n ala
direccion planeta ")

```

```

            Escribir (planet.getName())

```

```

            return shipClass

```

```

        Fin

```

```

    Fin

```

```

    return null

```

```

Fin

```

Se encola la accion

**getData():**

```

var working="Sin trabajo"

```

```

Si(state==true)

```

```

    working = "Con trabajo"

```

```

Escribir ("Tipo: " + name + " Nave Conocida: "+ shipClass.getName()+" Estado: "+working)

```

```

Si(state == true)

```

```

    Escribir ("Turnos para terminar: "+(timeTaked - workedDays))

```

```

FinSino

```

```

    Escribir ("")

```

```

Fin

```

**Clase Hijas: Workman, paraeman, Architect, Engineer**

Constructor:

```

public Architect(Planets planet)

```

```

    super (planet)

```

```

    timeTaked = 1

```

```

    price = 250

```

```

    soldPrice = 175

```

```

    shipClass = new MillenialFalcon(planet)

```

```

    name = "Arquitecto"

```

```

Fin

```

El valor dado depende de cada clase



## **Clase: Ship**

Variables slots, name, warriorSlots, cost, velocity, state, turns, home, orbit, warriorsAboard, owner

Constructor

```
public Ship(Planets planet)
    this.owner = planet.getOwner()
    orbit = planet
    home = planet
Fin
```

## **Metodos:**

### ***get"Valor":***

Svaraxis General

```
public var getTimesConfronted()
    return timesConfronted
Fin
```

Devuelve el valor de alguna variable

### ***newOwner():***

```
public void newOwner(var owner)
    this.owner = owner
Fin
```

### ***ArriveWarriors():***

```
public boolean ArriveWarriors(Warriors[] warriors)
    warriorSlots = 0
```

```
    para (var i=0 i<warriors.length i++)
        warriorSlots += warriors[i].getSlots()
Fin
```

```
    Si (warriorSlots<=slots)
        warriorsAboard = warriors
        return true
```

Fin

Sino

Escribir ("La cantidad de guerreros enviados supera el limite de "+slots +" espacios de la nave")

```
    warriorsAboard = warriors
    return false
```

Fin

Agrega guerreros a la nave

### ***deleteWarriors()***

```
    warriorsAboard = new Warriors[0]
```

Elimina los guerreros

**returnHome()**

```

public Planets returnHome(Planets[][] planetList)
    WarriorHandler handler = new WarriorHandler()
    return handler.GoPlanet(home, orbit, this, "", 0, handler.getDistance(planetList, home.getName()),
orbit.getName()))
    Fin

```

Retorna la nave a su planeta Origen

**addTurns():**

```

public void addTurns(var turns)
    this.turns = turns
    Fin

```

Agrega Turnos A la nave

**newDestiny():**

```

public void newDestiny(Planets planet)
    this.orbit = planet
    this.state=true
    Fin

```

//Cambia el destino de la nave

**turnPass():**

```

Si (state)
    this.turns -=1
    Si (turns == 0)
        state =false
        return true
    Fin
    Fin
    return false
    Fin

```

Encola la Accion

**Clases Hijas: Naboo, XWing, Millenial Falcon, StarDestroyer:**

Constructor

```

public Naboo(Planets planet)
    super(planet)
    slots = 25
    cost= 40
    velocity = 1.00
    name = "Naboo N-1"
    Fin

```

Fin

El valor dado depende de cada clase

## **Class: Map Handler**

### **Metodos:**

#### **createRandomMap()**

```
public Map createRandomMap()
```

```
    PlanetHandler mkr = new PlanetHandler()
    Planets[][] world = new Planets[random.nextvar(21)+5][random.nextvar(21)+5]

    para (var row=0 row<world.length row++)
        para (var col=0 col<world[0].length col++)
            Si (random.nextvar(2) == 1)
                world[row][col] = mkr.mkWorld(world[row][col])
        Fin
    Fin
```

Retorna un mapa aleatorio

#### **createMap()**

```
public Map createMap(var rows, var cols, var numberPlanets, var player1, var player2)
```

```
    PlanetHandler mkr = new PlanetHandler()
    Planets[][] world = new Planets[rows][cols]
    var tries= numberPlanets
    while(tries != 0)
        para (var row=0 row<world.length row++)
            para (var col=0 col<world[0].length col++)
                Si (random.nextvar(3) == 1)
                    world[row][col] = mkr.mkWorld(world[row][col])
                tries--
            Fin
        Si (tries == 0)
            return new Map(world,player1, player2)
        Fin
    Fin
    return new Map(world,player1, player2)
Fin
```

Retorna un Mapa con parametros definidos

#### **returnPlanets()**

```
public Planets[] returnPlanets(Map map)
```

```
    Planets[] planets
    var counter = 0

    para (var x=0 x<map.getMap().length x++)
        para (var y=0 y<map.getMap()[0].length y++)
            Si(map.getMap()[x][y] != null)
                counter++
        Fin
    Fin
```

```

planets=new Planets[counter]
para (var x=0  x<map.getMap().length  x++)
    para (var y=0  y<map.getMap()[0].length  y++)
        Si(map.getMap()[x][y]!= null)
            planets[counter-1] = map.getMap()[x][y]
            counter--
        Fin
    Fin
Fin

```

```

return planets

```

```

Fin

```

Retorna los planetas del Mapa

**getPlayersPlanets():**

```

public Map getPlayersPlanets(Map map,var Player1, var Player2)

```

```

    Map mapAux=map

```

```

    var planetsTerrapamedPlayer1=2

```

```

    var planetsTerrapamedPlayer2=2

```

```

    var x

```

```

    var y

```

```

    while(true)

```

```

        mapAux=map

```

```

        x=random.nextvar(map.getMap().length)

```

```

        y=random.nextvar(map.getMap()[0].length)

```

```

        Si(mapAux.getMap()[x][y] != null)

```

```

            Si(mapAux.getMap()[x][y].getOwner() == null && planetsTerrapamedPlayer1 != 0)

```

```

                mapAux.getMap()[x][y].addOwner(Player1)

```

```

                mapAux.getMap()[x][y].getShips()[0].newOwner(Player1)

```

```

                mapAux.getMap()[x][y].getBuilders()[0].newOwner(Player1)

```

```

                planetsTerrapamedPlayer1 -=1

```

```

            Fin

```

```

            Sino(mapAux.getMap()[x][y].getOwner() == null && planetsTerrapamedPlayer2 != 0)

```

```

                mapAux.getMap()[x][y].addOwner(Player2)

```

```

                mapAux.getMap()[x][y].getShips()[0].newOwner(Player2)

```

```

                mapAux.getMap()[x][y].getBuilders()[0].newOwner(Player2)

```

```

                planetsTerrapamedPlayer2 -=1

```

```

            Fin

```

```

        Fin

```

```

        Si(planetsTerrapamedPlayer1 ==0 && planetsTerrapamedPlayer2 ==0)

```

```

            break

```

```

    Fin

```

```

    return mapAux

```

*Fin*

*Retorna mapas con Dueño*

**Class: PlanetHandler**

**Metodos():**

**mkWorld():**

```
public Planets mkWorld(Planets planet)
    final var radiation = 0.05
    final var organic=0.15
    final var fire=0.30
    final var water=0.55
```

```
Random random = new Random()
var probability = random.nextvar()
```

```
Si (probability <= radiation)
    planet = new Radiation()
Sino (probability >radiation && probability<= organic)
    planet = new Organic()
Sino (probability >organic && probability <= fire)
    planet = new Fire()
Sino (probability >fire &&probability <= water)
    planet = new Water()
Sino
    planet = new Earth()
```

```
return planet
```

*Fin*

Retorna un planeta con clase Hija

**buyConstructor()**

```
public Planets buyConstructor(Planets planet,Constructors builder)
    Planets planetAux=planet
    var money = planetAux.getMoney()
    Si(money >= builder.getPrice())
        money -= builder.getPrice()
        planetAux.addBuilder(builder)
        Escribir ("Transaccion completada, gracias por comprar un " + builder.getName())
        planetAux.newMoney(money)
    Fin
    Sino
        Escribir ("Transaccion incompleta, se requieren más fondos")
    return planet
    Fin
```

*return planetAux*

*Fin*

Accion de Comprar un constructor, devuelve un Planeta

**sellConstructor():**

*public Planets sellConstructor(Planets planet,Constructors builder)*

*Planets planetAux=planet*

*var money = planet.getMoney()*

*boolean findBuilder=false*

*Constructors[] builders = planet.getBuilders()*

*//Se toma en cuenta si se encuentra el constructor*

*para (var i=0 i<builders.length i++)*

*Si (builder.getName().equalsIgnoreCase(builders[i].getName())) )*

*Si(builders[i].getState() == false)*

*findBuilder=true*

*builders[i] =null*

*break*

*Fin*

*Fin*

*Fin*

*Si(findBuilder)*

*Constructors[] buildersAux = new Constructors[builders.length-1]*

*para (var i=0 i<builders.length i++)*

*Si (builders[i]!= null)*

*buildersAux[i] = builders[i]*

*Fin*

*Fin*

*money += builder.getSoldPrice()*

*planetAux.newMoney(money)*

*planetAux.newBuilder(buildersAux)*

*Escribir ("Vendido")*

*Fin*

*Sino*

*Escribir ("No se hallo el constructor mencionado")*

*return planet*

*Fin*

*return planetAux*

*Fin*

Accion de vender un constructor, devuelve un planeta, NO ELIMINAR TODOS LOS CONSTRUCTORES DE UN PLANETA



**sellShip()**

```
public Planets sellShip(Planets planet, Ship ship)
    Planets planetAux=planet
    var money = planet.getMoney()
    var index
    boolean findShip=false
    Ship[] ships = planet.getShips()

    //Trata de hallar la nave en la lista de naves del planeta
    para (var i=0 i<ships.length i++)
        Si (ship.getName().equalsIgnoreCase(ships[i].getName()) )
            Si(ships[i].getState() == false)
                findShip = true
                ships[i] =null
                break
            Fin
        Fin
    Fin

    Si(findShip)
        Ship[] shipsAux = new Ship[ships.length-1]
        para (var i=0 i<shipsAux.length i++)
            Si (ships[i]!= null)
                shipsAux[i] = ships[i]
            Fin
        Fin
        money += ship.getPrice()
        planetAux.newMoney(money)
        planetAux.newShip(shipsAux)
        Escribir ("Vendido")
    Fin
    Sino
        Escribir ("No se hallo la nave mencionado")
        return planet
    Fin
    return planetAux
Fin
```

**Accion de vender una nave****turnPass()**

```
public Planets turnPass(Planets planet)
    planet.addWarrior()
    planet.addGalactus()
    Ship shipBuild
    para(var index=0 index<planet.getBuilders().length index++)
        shipBuild = planet.getBuilders()[index].turnPass()
        Si(shipBuild != null)
```

```
    planet.addShip(shipBuild)
Fin
Fin
```

```
    return planet
Fin
Accion encolada se ejecuta
```

### **Class: PlayerHandler()**

variables originMap, playableMap, mapHandler, starterMap

#### **Metodos():**

##### **getMenu():**

```
public var getMenu()
    var menu=0
    Escribir ("Bienvenido a KONQUEST, selecciona una opcion colocando el numero(1)")
    Escribir ("1.- Diseño de Mapas")
    Escribir ("2.- Jugar")
    Escribir ("3.- Salir")
    try
        menu = vareger.parsevar(scanner.nextLine())
    Fincatch(Throwable e)
        Escribir ("Opcion incorrecta se debe de escoger solo un numero,\n ejemplo 1\n")
    Fin
    return menu
Fin
Muestra el Menu
```

##### **choice()**

```
public void choice(var menu)
    boolean state=true

    while(state == true)
        switch(menu)
            case 1:
                state = mapDesign(state)
                break
            case 2:
                state = play(state)
                break
        Fin
    Fin
Fin
Escoge que funcion llamar
```

### **statusPlaying()**

```
private boolean statusPlaying(User[] user)
```

```
    PlanetHandler planetHandler = new PlanetHandler()
```

```
    WarriorHandler warriorHandler = new WarriorHandler()
```

```
    var turno=0
```

```
    while(true)
```

```
        Escribir ("\nTurno de " + user[turno].getPlayer())
```

```
        playableMap = user[turno].showActions()
```

```
        Si(turno==0)
```

```
            turno=1
```

```
        Sino
```

```
            turno=0
```

```
        para(var x=0 x<playableMap.getMap().length x++)
```

```
            para(var y=0 y<playableMap.getMap()[0].length y++)
```

```
                Si(playableMap.getMap()[x][y] != null)
```

```
                    playableMap.getMap()[x][y] = planetHandler.turnPass(playableMap.getMap()[x]
```

```
[y])
```

```
                Fin
```

```
            Fin
```

```
        Fin
```

```
        para(var x=0 x<playableMap.getMap().length x++)
```

```
            para(var y=0 y<playableMap.getMap()[0].length y++)
```

```
                Si(playableMap.getMap()[x][y] != null)
```

```
                    para(var index=0 index<playableMap.getMap()[x][y].getShips().length index++)
```

```
                        para(var x1=0 x1<playableMap.getMap().length x1++)
```

```
                            para(var y1=0 y1<playableMap.getMap()[0].length y1++)
```

```
                                Si(playableMap.getMap()[x1][y1] != null && playableMap.getMap()[x1][y1]!
```

```
=playableMap.getMap()[x][y])
```

```
                                    Si(playableMap.getMap()[x1][y1].getName() == playableMap.getMap()
```

```
[x][y].getShips()[index].getOrbit().getName())
```

```
                                    Si(playableMap.getMap()[x][y].getShips()[index].turnPass())
```

```
                                        Planets planeta = playableMap.getMap()[x1][y1]
```

```
                                            playableMap.getMap()[x1][y1] =
```

```
warriorHandler.closeCombatReturnPlanet(planeta,playableMap.getMap()[x][y].getShips()[index])
```

```
                                            playableMap.getMap()[x][y].getShips()[index] =
```

```
warriorHandler.closeCombatReturnShip(planeta,playableMap.getMap()[x][y].getShips()[index])
```

```
                                            playableMap.getMap()[x][y] = playableMap.getMap()[x]
```

```
[y].getShips()[index].returnHome(playableMap.getMap())
```

```

        Fin
    Fin
    Fin
    Fin
    Fin
    Fin
    Fin
    Fin
    Si( (user[0].checkVictory(playableMap.getNumberPlanets()))
        break
    Fin
    Sino((user[1].checkVictory(playableMap.getNumberPlanets()))
        break
    Fin
    Fin
    Fin
    return false
    Fin

```

#### Motor de encolamiento y turnos del juego

##### **play()**

```

private boolean play(boolean state)
    var menuX = 0
    var mapSelected
    Map originMap = null
    playableMap = null
    var[] player = new var[2]
    Escribir ("Bienvenido al Menu Jugar, selecciona una opcion colocando el numero (1)")
    Escribir ("1.- Iniciar Juego")
    Escribir ("2.- Salir")
    try
        menuX = vareger.parsevar(scanner.nextLine())
        Si (menuX == 1)
            Escribir ("Selecciona un Mapa para jugar")

            Si(starterMap == true)
                para (var index=0 index<this.originMap.length index++)
                    Escribir ("Mapa "+(index+1)+" = ")
                    this.originMap[index].getData()
                Fin

            FinSino
                Escribir ("Sin mapas disponibles para jugar, Crea Uno")
                return true
            Fin

        mapSelected = vareger.parsevar(scanner.nextLine())

```

```

try
    originMap = (Map) this.originMap[mapSelected-1].clone()
Fin catch (CloneNotSupportedException ex)
    Escribir ("error al clonar")
Fin
Escribir (originMap.tovar())
player = originMap.getPlayers()
this.playableMap = mapHandler.getPlayersPlanets(originMap, player[0], player[1])
User[] user = new User[2]
user[0] = new User(player[0],playableMap)
user[1] = new User(player[1],playableMap)
statusPlaying(user)
FinSino
    state = false
Fin
Fincatch(Throwable e)
    Escribir ("Error Escogiendo la opcion")

Fin
return state
Fin

```

Muestra el menu de jugar y se vareractua

### **mapDesign()**

```

private boolean mapDesign(boolean state)
    var menu1=0
    var rows
    var cols
    var numberPlanets
    var menuX = 0
    var player1
    var player2
    Escribir ("Bienvenido Al diseñador de Mapas escoge una opción")
        Escribir ("1.- Crear Mapa Aleatorio")
        Escribir ("2.- Crear Mapa Personalizado")
        Escribir ("3.- Mostrar Mapas Existentes")
        Escribir ("4.- Ver un Mapa")
        Escribir ("5.- Regresar Menu")
    try
        menu1 = vareger.parsevar(scanner.nextLine())
    Fincatch(Throwable e)
        Escribir ("Opcion incorrepta se debe de escoger solo un numero,\n ejemplo 1\n")
    Fin

    switch(menu1)
        case 1:
            this.originMap = concatenate(originMap,mapHandler.createRandomMap())
            Escribir ("Mapa creado")
        break

```

case 2:

```
try
    Escribir ("Escribe el numero de Filas")
    rows = vareger.parsevar(scanner.nextLine())
    Escribir ("Escribe el numero de Columnas")
    cols = vareger.parsevar(scanner.nextLine())
    Escribir ("Escribe el numero de Planetas")
    numberPlanets = vareger.parsevar(scanner.nextLine())
    Si(numberPlanets > rows * cols)
        Escribir ("Error Creando el Mapa")
        break
    Fin
    Escribir ("Escribe el nombre de un jugador")
    player1 = scanner.nextLine()
    Escribir ("Escribe el nombre de otro jugador")
    player2 = scanner.nextLine()
```

```
        this.originMap = concatenate(originMap,mapHandler.createMap(rows,
cols,numberPlanets, player1, player2))
        Escribir ("Mapa creado")
    Fincatch(Throwable e)
        Escribir ("Error creando el Mapa")
    Fin
break
```

case 3:

```
Si(starterMap == true)
    para (var index=0 index<originMap.length index++)
        Escribir ("Mapa "+(index+1)+" = ")
        originMap[index].getData()
    Fin
    Fin
    Sino
        Escribir ("Sin mapas disponibles, Crea Uno")
    break
```

case 4:

```
Si(this.starterMap == true)
    Escribir ("Escribe el numero de mapa que quieres ver")

    try
        menuX = vareger.parsevar(scanner.nextLine())
        originMap[menuX-1].prvarMap("", "")
        originMap[menuX-1].getData()
        Fincatch(Throwable e)
            Escribir ("Mapa no encontrado")
        Fin
    FinSino
```



*Escribir ("No hay mapas para mostrar, Crea uno")*

*break*  
*default:*  
*state=false*

*Fin*

*return state*

*Fin*

Devuelve el menu de diseño de mapa e vareractua con ella

### **concatenete()**

*private Map[] concatenate(Map[] originMap, Map map)*

*Map[] mapAux = new Map[originMap.length+1]*

*Si( starterMap == true)*

*para (var i=0 i<originMap.length i++)*

*mapAux[i] = originMap[i]*

*Fin*

*mapAux[originMap.length] = map*

*Fin*

*Sino*

*originMap[0] = map*

*starterMap =true*

*return originMap*

*Fin*

*return mapAux*

*Fin*

Une dos vectores

### **Class: WarriorHandler**

#### **Metodos()**

#### **GoPlanet()**

*public Planets GoPlanet(Planets planetOrigin,Planets planetTarget,Ship ship,var warriorClass,var numberWarriors, var distance)*

*Planets planetAux=planetOrigin*

*var contadorGuerreros=0*

*var indexShip=0*

*Si(planetOrigin == null || planetTarget == null)*

*Escribir ("Escogiste un planeta Vacio")*

*return planetOrigin*

*Fin*

*para(var index=0 index<planetAux.getShips().length index++)*

*Si(planetAux.getShips()[index].getName()==ship.getName())*

*Si(planetAux.getShips()[index].getState() != true)*

*indexShip=index*

*break*

*Fin*

```

Fin
Si(index == planetAux.getShips().length-1)
    Escribir ("No se encontro la Nave")
    return planetOrigin
Fin
Fin

```

```

Si (planetAux.getShips()[indexShip].getState() == false)

```

```

    var time=0
    Si (distance!=0)
        time = (var) (distance/ship.getVelocity())
        Si (distance/ship.getVelocity() % 1 >0.50)
            time +=1
        Fin
    Fin

```

```

    Sino
        Escribir ("Ya estas en este planeta")
        return planetOrigin
    Fin

```

```

    Fin
    Warriors[] planetWarriors = planetAux.getWarriors()
    Warriors[] shipWarriors = new Warriors[numberWarriors]
    para(var i=0 i<numberWarriors i++)
        //trycatch para ver que los guerreros son insuficientes
        Si(planetWarriors[i].getName().equalsIgnoreCase(warriorClass))
            shipWarriors[i] = planetWarriors[i]
            contadorGuerreros++
        Fin
    Fin

```

```

    Si(contadorGuerreros!= numberWarriors)
        Escribir ("Cantidad de guerreros solicitados insuficientes")
        Escribir ("No se ha podido enviar la Nave")
        return planetOrigin
    Fin

```

```

    Si( planetAux.getShips()[indexShip].ArriveWarriors(shipWarriors))

```

```

        Si(warriorClass != "")
            planetAux.leftWarriors(planetAux.getWarriors().length - numberWarriors)
            planetAux.getShips()[indexShip].newDestiny(planetTarget)
            planetAux.getShips()[indexShip].addTurns(time)
            Escribir ("Se ha enviado la nave " + planetAux.getShips()[indexShip].getName() + " de " +
planetAux.getShips()[indexShip].getOwner() + " Hacia ")
            planetTarget.getData()
            return planetAux
        Fin

```

```

    Sino
        planetAux.leftWarriors(planetAux.getWarriors().length - numberWarriors)
        planetAux.getShips()[indexShip].newDestiny(planetOrigin)
        planetAux.getShips()[indexShip].addTurns(time)
        Escribir ("Se ha enviado la nave " + planetAux.getShips()[indexShip].getName() + " de " +
planetAux.getShips()[indexShip].getOwner() + " Hacia ")
        planetTarget.getData()
        return planetAux
    Fin

```

Fin

Fin

```

    Escribir ("No se ha podido enviar la Nave")
    return planetOrigin

```

Fin

Devuelve un planeta y cambia los parametros de una de sus naves dependiendo de los parametros insertados en la funcion

### **getDistance()**

```

public var getDistance(Planets[][] planetList, var origin, var destiny)
    var time
    var xi=0
    var yi=0
    var xf=0
    var yf=0
    var distance=0
    para (var x=0 x<planetList.length x++)
        para (var y=0 y<planetList[0].length y++)
            Si(planetList[x][y] != null)
                Si (planetList[x][y].getName().equalsIgnoreCase(origin))
                    xi=y+1
                    yi=x+1
                Fin
                Sino(planetList[x][y].getName().equalsIgnoreCase(destiny))
                    xf=y+1
                    yf=x+1
                Fin
            Fin
        Fin
    Fin
    distance = Math.sqrt(Math.pow((xf-xi),2) + Math.pow((yf-yi),2))
    Si (distance <0)
        distance *= -1

    time = (var) (distance/1)
    Si (distance/1 % 1 >0.50)
        time +=1

```

```

    Fin
    Escribir ("Turnos requeridos para la nave Naboo N-1 = "+ time )
    time = (var) (distance/1.25)
    Si (distance/1.25 % 1 >0.50)
        time +=1
    Fin

```

```

    Escribir ("Turnos requeridos para la nave X-Wing = "+ time )
    time = (var) (distance/1.50)
    Si (distance/1.50 % 1 >0.50)
        time +=1
    Fin

```

```

    Escribir ("Turnos requeridos para la nave Millenial Falcon = "+ time )
    time = (var) (distance/1.75)
    Si (distance/1.75 % 1 >0.50)
        time +=1
    Fin

```

```

    Escribir ("Turnos requeridos para la nave Star Destroyer = "+ time )

```

```

    return distance

```

```

Fin

```

Obtiene la distancia separadas por dos planetas, utilizando la paramula de la distancia y ademas muestra cuanto se demorara segun cada tipo de nave

### **closeCombatReturnPlanet()**

```

public Planets closeCombatReturnPlanet(Planets planet, Ship ship)

```

```

Si (ship.getOwner() != planet.getOwner())
    Warriors[] resident= planet.getWarriors()
    Warriors[] visitor = ship.getWarriors()
    var residentBattles = 0
    var visitorBattles = 0
    para (var i = 0   i < resident.length   i++)
        Si(resident[i] != null)
            residentBattles++
    Fin
    Fin
    para (var i = 0   i < visitor.length   i++)
        Si(visitor[i] != null)
            visitorBattles++
    Fin
    Fin
    residentBattles--
    visitorBattles--
    Si(resident.length != 0 || visitor.length!=0)

```

while(true)

```
Si(resident[residentBattles].getdeathFactor() > visitor[visitorBattles].getdeathFactor())
    visitor[visitorBattles] = null
    visitorBattles--
    resident[residentBattles].addTimesConfronted()
Fin
Sino (resident[residentBattles].getdeathFactor() < visitor[visitorBattles].getdeathFactor())
    resident[residentBattles] = null
    residentBattles--
    visitor[visitorBattles].addTimesConfronted()
Fin
```

```
Si (resident[residentBattles].getTimesConfronted() == 2)
    resident[residentBattles] = null
    residentBattles--
Fin
Sino (visitor[visitorBattles].getTimesConfronted() == 2)
    visitor[visitorBattles] = null
    visitorBattles--
Fin
```

```
Si (residentBattles == 0)
    Warriors[] survivours = new Warriors[visitorBattles+1]

    para (var i=0 i<survivours.length i++)
        Si(visitor[i]!=null)
            survivours[i] = visitor[i]
    Fin
Fin
```

```
planet.addOwner(ship.getOwner())
planet.leftWarriors(0)
planet.welcomeAllies(survivours)
ship.deleteWarriors()
Escribir ("Se ha conquistado el planeta "+planet.getName())
break
Fin
Sino(visitorBattles == 0)
    ship.deleteWarriors()
    planet.leftWarriors(residentBattles+1)
    Escribir ("No se ha podido conquistar el planeta "+planet.getName())
    break
Fin
```

```
Fin
Fin
Si(resident.length == 0)
```

```

        Si(visitor.length!=0)
        planet.welcomeAllies(visitor)
        ship.deleteWarriors()
        Escribir ("Se ha conquistado el planeta "+planet.getName())
    Fin
FinSino(visitor.length == 0)
Si(resident.length!=0)
    Escribir ("No se ha podido conquistar el planeta"+planet.getName())
FinSino
    planet.addOwner(ship.getOwner())
    Escribir ("Se ha conquistado el planeta "+planet.getName())
Fin
Fin
Fin
Sino
    planet.welcomeAllies(ship.getWarriors())
    ship.deleteWarriors()
Fin
return planet
Fin

```

Realiza la accion de combatir entre los guerreros de un planeta enemigo y los guerreros abordo de la nave invasora, retorna el planeta

### **closeCombatReturnShip()**

Mismo Algoritmo que el anterior pero devuelve una clase nave

### **Class: Mapa**

variables map, player1, player2

Constructor

```

public Map(Planets[][] map, var player1, var player2)
    this.map = map
    this.player1 = player1
    this.player2 = player2
Fin

```

### **get"Valor":**

Svaraxis General

```

public Planets[][] getMap()
    return map
Fin

```

Obtiene Algun valor de variable o clase

### **getNúmerPlanets()**

```

public var getNumberPlanets()
    var counter=0
    Planets[] planets
    para (var x=0  x<map.length  x++)

```



```

    para (var y=0 y<map[0].length y++)
        Si(map[x][y]!= null)
            counter++
    Fin
Fin

```

```

return counter

```

```

Fin

```

Retorna el tottal de planetas del Mapa

**prvarMap()**

```

public void prvarMap(var jugador1, var jugador2)

```

```

    var espaciosBlancos=0

```

```

    var oracion

```

```

    var size

```

```

    para (var x=0 x<map[0].length x++)

```

```

        Si(map.length>=9)

```

```

            Escribir ("XXXXX "+ (var) (65+x) +" XXXXXX+ ")

```

```

        FinSino

```

```

            Escribir ("XXXXX "+(var) (65+x)+" XXXXXXXX+ ")

```

```

    Fin

```

```

Escribir ("")

```

```

para(var x=0 x<map.length x++)

```

```

    Escribir ((x+1))

```

```

    para(var y=0 y<map[0].length y++)

```

```

        Si(map[x][y]!= null)

```

```

            oracion = "Nombre: " + map[x][y].getName()

```

```

        Si(x<=9)

```

```

            Si(map[x][y].getOwner() != null)

```

```

                Si (map[x][y].getOwner().equalsIgnoreCase(jugador1))

```

```

                    Escribir ("\033[31mNombre:" + map[x][y].getName() + "\033[0m X")

```

```

                Fin

```

```

                Sino

```

```

                    Escribir ("\033[33mNombre:" + map[x][y].getName() + "\033[0m X")

```

```

                Fin

```

```

            Fin Sino

```

```

                Escribir ("Nombre:" + map[x][y].getName() + " X")

```

```

            Fin

```

```

    Fin

```

```

    Sino

```

```

        Si(map[x][y].getOwner() != null)

```

```

            Si (map[x][y].getOwner().equalsIgnoreCase(jugador1))

```

```

                Escribir ("\033[31mNombre:" + map[x][y].getName() + "\033[0mX")

```

```

            Fin

```

```

            Sino

```

```

        Escribir ("\033[33mNombre:" + map[x][y].getName() + "\033[0mX")
    Fin
Fin Sino
    Escribir ("Nombre:" + map[x][y].getName() + "X")
Fin
Fin
Fin
Sino
    Si(x>=9 && y == 0)
        Escribir ("          X")
    Fin
    Sino
        Escribir ("          X")
    Fin
Fin
Fin
Escribir ("\nX")
para(var y=0  y<map[0].length  y++)
    Si(map[x][y]!= null)
        oracion = "Dueño:"+map[x][y].getOwner() + " X"
        Si(oracion.length() > 16)
            size = map[x][y].getOwner()
            Si(size.length()>8)
                oracion="Dueño:"
                para(var i=0  i<8  i++)
                    oracion += size.varAt(i)
            Fin
        Fin
    Fin
    Sino (oracion.length()<16)
        espaciosBlancos = 16-oracion.length()
        oracion = "Dueño:"+map[x][y].getOwner()
        para (var i=0  i<espaciosBlancos  i++)
            oracion += " "
        Fin
    Fin
    Sino (oracion.length()==16)
        oracion = "Dueño:"+map[x][y].getOwner()
    Si(map[x][y].getOwner() != null)
        Si (map[x][y].getOwner().equalsIgnoreCase(jugador1))
            Escribir ("\033[31m"+oracion+ "\033[0m X")
        Fin
    Sino
        Escribir ("\033[33m"+oracion+ "\033[0m X")
    Fin
Fin Sino
    Escribir (oracion+" X")
Fin
Fin

```

```

        Sino
            Escribir ("          X")
        Fin
    Fin

    Escribir (" ")
    para (var d=0 d<map[0].length d++)
        Escribir ("XXXXXXXXXXXXXXXXXXXX")
    Fin
    Escribir (" ")
Fin

```

Fin

Imprime el Mapa

**restart()**

public void restart()

```

    para(var i = 0 i < map.length i++)
        para(var j = 0 j < map[0].length j++)
            Si(map[i][j] != null) map[i][j].restart()
    Fin

```

Fin

Fin

Reinicia Valores

**Class: Usuario**

Variables player, shipList, map,

Constructor

public User(var player, Map map)

    this.player = player

    this.map = map

Fin

### **get"Valor"()**

Svaraxis General

```
public var getPlayer()  
    return player
```

Fin

Retorna el valor de alguna variable o clase

### **showActions()**

```
public Map showActions()  
    var choice  
    var indexRows1 = 0  
    var indexCols1  
    var indexRows2 = 0  
    var indexCols2  
    var warriorsSent = 0  
    var indexvar = 0  
    var shipUse = null  
    var warriorsUse = null  
    var oracion = ""  
    boolean state= true  
    Scanner scanner = new Scanner(System.in)  
    WarriorHandler warriorHandler = new WarriorHandler()  
    PlanetHandler planetHandler = new PlanetHandler()  
    Constructors constructor = null  
    var[] planeta  
    while(state == true)  
        try  
            Escribir ("Presiona Cualquier Tecla Para Continuar")  
            scanner.nextLine()  
            map.prvarMap(map.getPlayers()[0], map.getPlayers()[1])  
            Escribir ("\nAcciones "+player+" , porfavor elige una")  
            Escribir ("1.- Obtener distancia")  
            Escribir ("2.- Ver planeta")  
            Escribir ("3.- Consulta de Flota")  
            Escribir ("4.- Envio de Flota")  
            Escribir ("5.- Construir Nave")  
            Escribir ("6.- Tienda")  
            Escribir ("7.- Ver Posesiones")  
            Escribir ("8.- Terminar")  
            choice = vareger.parsevar(scanner.nextLine())  
  
            Escribir ("\n\n\n\n")  
            switch(choice)  
                case 1:  
                    getDistance()  
                    break  
  
                case 2:  
                    //Permite ver las caracteristicas del planeta
```

*Escribir ("Escribe el planeta que quieres ver de la siguiente manera, A1")*  
*planeta = scanner.nextLine().toCharArray()*

*indexCols1 = planeta[0] - 65*  
*oracion = ""*  
*para(var i=1 i<planeta.length i++)*

*Si(i<planeta.length)*  
*oracion += planeta[i]*  
*indexvar = i+1*  
*Fin*

*Fin*  
*indexRows1 = vareger.parsevar(oracion) -1*  
*Si(map.getMap()[indexRows1][indexCols1] != null)*  
*Si(map.getMap()[indexRows1][indexCols1].getOwner() == null || map.getMap()*  
*[indexRows1][indexCols1].getOwner().equalsIgnoreCase(player))*  
*map.getMap()[indexRows1][indexCols1].getData()*  
*FinSino*  
*Escribir ("No puedes ver el planeta enemigo")*  
*FinSino*  
*Escribir ("Es el espacio")*  
*break*

*case 3:*  
*this.getShipOnGoing()*  
*break*

*case 4:*  
*Escribir ("Indica el planeta Origen, cantidad de guerreros, tipo de guerreros, tipo de nave*  
*a usar, planeta destino ejemplo A1,35,Mole,Naboo N-1,G4")*  
*Escribir ("Tipo de Guerreros: Mole(1 espacio), Nemo(1 espacio), Magma(2 espacios),*  
*Groot(3 espacios), Fision Guy(4 espacios)")*  
*Escribir ("Tipo de Naves: Naboo N-1(25 Espacios), X-Wing(42 Espacios), Millenial*  
*Falcon(58 espacios), StarDestroyer(80 espacios)")*  
*Escribir ("")*  
*planeta = scanner.nextLine().toCharArray()*

*indexCols1 = planeta[0] - 65*  
*oracion = ""*  
*para(var i=1 i<planeta.length i++)*  
*Si(planeta[i] != ',')*  
*oracion += planeta[i]*  
*FinSino*  
*indexRows1 = vareger.parsevar(oracion) -1*  
*indexvar = i+1*  
*break*  
*Fin*

Fin

oracion=""

para(var index=indexvar index<planeta.length index++)

Si(planeta[index] != ',')

oracion += planeta[index]

Fin

Sino

warriorsSent = vareger.parsevar(oracion)

indexvar = index+1

oracion = ""

break

Fin

Fin

para(var index= indexvar index<planeta.length index++)

Si(planeta[index] != ',')

oracion += planeta[index]

Fin

Sino

warriorsUse = oracion

indexvar = index+1

oracion = ""

break

Fin

Fin

para(var index= indexvar index<planeta.length index++)

Si(planeta[index] != ',')

oracion += planeta[index]

Fin

Sino

shipUse = oracion

indexvar = index+1

oracion = ""

break

Fin

Fin

indexCols2 = planeta[indexvar]-65

para(var i=indexvar+1 i<planeta.length i++)

oracion += planeta[i]

Fin

indexRows2 = vareger.parsevar(oracion)-1

Si(map.getMap()[indexRows1][indexCols1].getOwner().equalsIgnoreCase(player))

Escribir ("planeta origen: "+map.getMap()[indexRows1][indexCols1].getName())

Escribir ("Cantidad guerreros: "+ warriorsSent)

```

Escribir ("Tipo Guerrero: "+ warriorsUse )
Escribir ("Tipo de Nave: "+shipUse)
Escribir ("Destino: " + map.getMap()[indexRows2][indexCols2].getName())

```

```

    map.getMap()[indexRows1][indexCols1] = warriorHandler.GoPlanet(map.getMap()
[indexRows1][indexCols1], map.getMap()[indexRows2][indexCols2], this.getShipClass(map.getMap()
[indexRows1][indexCols1], shipUse), this.getWarriorsClass(map.getMap()[indexRows1][indexCols1],
warriorsUse).getName(), warriorsSent, warriorHandler.getDistance(map.getMap(), map.getMap()
[indexRows1][indexCols1].getName(), map.getMap()[indexRows2][indexCols2].getName()))
    Fin

```

```

    break
case 5:

```

```

    Escribir ("Se requiere indicar el planeta, y el constructor , ejemplo A1,Obrero")
    Escribir ("Tipos de constructores: Obrero(40 Galactus), Maestro de Obra(50 Galactus),
Arquitecto(70 Galactus), Ingeniero(100 Galactus)")
    planeta = scanner.nextLine().toCharArray()
    indexCols1 = planeta[0] - 65
    oracion = ""
    para(var i=1 i<planeta.length i++)
        Si(planeta[i] != ',')
            oracion += planeta[i]
        FinSino
        indexRows1 = vareger.parsevar(oracion)-1
        indexvar = i+1
        break
    Fin
Fin

```

```

Si(map.getMap()[indexRows1][indexCols1].getOwner().equalsIgnoreCase(player))
    oracion = ""
    para(var i=indexvar i<planeta.length i++)
        oracion += planeta[i]
    Fin

```

```

Si(oracion.equalsIgnoreCase("Obrero"))
    constructor = new Workman(map.getMap()[indexRows1][indexCols1])
FinSino(oracion.equalsIgnoreCase("Maestro de Obra"))
    constructor = new paraeman(map.getMap()[indexRows1][indexCols1])
FinSino(oracion.equalsIgnoreCase("Arquitecto"))
    constructor = new Architect(map.getMap()[indexRows1][indexCols1])
FinSino(oracion.equalsIgnoreCase("Ingeniero"))
    constructor = new Engineer(map.getMap()[indexRows1][indexCols1])
FinSino
    Escribir ("Instruccion mal escrita, vuelve a varentar")

```

```

    map.getMap()[indexRows1][indexCols1].createShip(constructor)
FinSino
    Escribir ("No es tu planeta")

```

```

        break
    case 6:
        try
            Escribir ("Bienvenido a la tienda")
            Escribir ("Escoge una opcion")
            Escribir ("1.- Comprar Constructor")
            Escribir ("2.- Vender Constructor")
            Escribir ("3.- Vender nave")
            choice = vareger.parsevar(scanner.nextLine())

            switch(choice)
            case 1:
                Escribir ("Para comprar un constructor indique el planeta, y el tipo de constructor
que desea, ejemplo A1,Obrero")
                Escribir ("Tipos de constructores: Obrero(50 Galactus), Maestro de Obra(100
Galactus), Arquitecto(250 Galactus), Ingeniero(300 Galactus)")
                planeta = scanner.nextLine().toCharArray()
                indexCols1 = planeta[0] - 65
                oracion = ""
                para(var i=1 i<planeta.length i++)
                    Si(planeta[i] != ',')
                        oracion += planeta[i]
                    FinSino
                    indexRows1 = vareger.parsevar(oracion)-1
                    indexvar = i+1
                    break
                Fin
            Fin
            Si(map.getMap()[indexRows1][indexCols1].getOwner().equalsIgnoreCase(player))
                oracion = ""
                para(var i=indexvar i<planeta.length i++)
                    oracion += planeta[i]
                Fin

                Si(oracion.equalsIgnoreCase("Obrero"))
                    constructor = new Workman(map.getMap()[indexRows1][indexCols1])
                FinSino(oracion.equalsIgnoreCase("Maestro de Obra"))
                    constructor = new paraeman(map.getMap()[indexRows1][indexCols1])
                FinSino(oracion.equalsIgnoreCase("Arquitecto"))
                    constructor = new Architect(map.getMap()[indexRows1][indexCols1])
                FinSino(oracion.equalsIgnoreCase("Ingeniero"))
                    constructor = new Engineer(map.getMap()[indexRows1][indexCols1])
                FinSino
                    Escribir ("Instruccion mal escrita, vuelve a varentar")

                map.getMap()[indexRows1][indexCols1] =
planetHandler.buyConstructor(map.getMap()[indexRows1][indexCols1], constructor)
                FinSino
                    Escribir ("No es tu planeta")

```



*break*

*case 2:*

*Escribir ("Para vender un constructor indique el planeta, y el tipo de constructor que desea vender, ejemplo A1,Obrero")*

*Escribir ("Tipos de constructores: Obrero(40 Galactus), Maestro de Obra(70 Galactus), Arquitecto(175 Galactus), Ingeniero(200 Galactus)")*

*planeta = scanner.nextLine().toCharArray()*

*indexCols1 = planeta[0] - 65*

*oracion = ""*

*para(var i=1 i<planeta.length i++)*

*Si(planeta[i] != ',')*

*oracion += planeta[i]*

*FinSino*

*indexRows1 = vareger.parsevar(oracion)-1*

*indexvar = i+1*

*break*

*Fin*

*Fin*

*Si(map.getMap()[indexRows1][indexCols1].getOwner().equalsIgnoreCase(player))*

*oracion = ""*

*para(var i=indexvar i<planeta.length i++)*

*oracion += planeta[i]*

*Fin*

*Si(oracion.equalsIgnoreCase("Obrero"))*

*constructor = new Workman(map.getMap()[indexRows1][indexCols1])*

*FinSino(oracion.equalsIgnoreCase("Maestro de Obra"))*

*constructor = new paraeman(map.getMap()[indexRows1][indexCols1])*

*FinSino(oracion.equalsIgnoreCase("Arquitecto"))*

*constructor = new Architect(map.getMap()[indexRows1][indexCols1])*

*FinSino(oracion.equalsIgnoreCase("Ingeniero"))*

*constructor = new Engineer(map.getMap()[indexRows1][indexCols1])*

*FinSino*

*Escribir ("Instruccion mal escrita, vuelve a varentar")*

*constructor.getData()*

*map.getMap()[indexRows1][indexCols1] =*

*planetHandler.sellConstructor(map.getMap()[indexRows1][indexCols1], constructor)*

*FinSino*

*Escribir ("No es tu planeta")*

*break*

*case 3:*

*Escribir ("Para vender una nave indica el planeta y el tipo de nave. Ejemplo A1,Naboo N-1")*

*Escribir ("Tipo de Naves: Naboo N-1(40 Galactus), X-Wing(50 Galactus), Millenial Falcon(70 Galactus), StarDestroyer(100 Galactus)")*

*planeta = scanner.nextLine().toCharArray()*

*indexCols1 = planeta[0] - 65*

*oracion = ""*

*para(var i=1 i<planeta.length i++)*

*Si(planeta[i] != ',')*

*oracion += planeta[i]*

*FinSino*

*indexRows1 = vareger.parsevar(oracion)-1*

*indexvar = i+1*

*break*

*Fin*

*Fin*

*Si(map.getMap()[indexRows1][indexCols1].getOwner().equalsIgnoreCase(player))*

*oracion = ""*

*para(var i=indexvar i<planeta.length i++)*

*oracion += planeta[i]*

*Fin*

*map.getMap()[indexRows1][indexCols1] = planetHandler.sellShip(map.getMap()[indexRows1][indexCols1], this.getShipClass(map.getMap()[indexRows1][indexCols1], oracion))*

*FinSino*

*Escribir ("No es tu planeta")*

*break*

*default:*

*Escribir ("Opcion Invalida")*

*Fin*

*Fincatch(Throwable e)*

*Escribir ("Error ejecutando comando, vuelve a varentar")*

*Fin*

*break*

*case 7:*

*this.getProperty()*

*break*

*case 8:*

*state =false*

*break*

*Fin*

```
    Fincatch(Throwable e)
        Escribir ("Algo salio mal, vuelve a varentar")
    Fin
Fin
```

```
    return map
```

```
Fin
```

Muestras las posibles acciones del usuario

### **getShipClass()**

```
private Ship getShipClass(Planets planet,var shipName)
    Ship shipSent=null
    Si(shipName.equalsIgnoreCase("Naboo N-1"))
        shipSent = new Naboo(planet)
    FinSino (shipName.equalsIgnoreCase("X-Wing"))
        shipSent = new XWing(planet)
    FinSino (shipName.equalsIgnoreCase("Millenial Falcon"))
        shipSent = new MillenialFalcon(planet)
    FinSino (shipName.equalsIgnoreCase("StarDestroyer"))
        shipSent = new StarDestroyer(planet)
    FinSino
        Escribir ("Nave no encontrada")
    Fin
    return shipSent
Fin
```

Retorna una clase nave dependiendo del nombre dado

### **getWarriorsClass()**

```
private Warriors getWarriorsClass(Planets planet, var warriorName)
    Warriors warriorsSent=null

    Si(warriorName.equalsIgnoreCase("Mole"))
        warriorsSent = new Mole(planet.getOwner(),planet.getDeathFactor())
    FinSino (warriorName.equalsIgnoreCase("Nemo"))
        warriorsSent = new Nemo(planet.getOwner(),planet.getDeathFactor())
    FinSino (warriorName.equalsIgnoreCase("Magma"))
        warriorsSent = new Magma(planet.getOwner(),planet.getDeathFactor())
    FinSino (warriorName.equalsIgnoreCase("Groot"))
        warriorsSent = new Groot(planet.getOwner(),planet.getDeathFactor())
    FinSino (warriorName.equalsIgnoreCase("Fision Guy"))
        warriorsSent = new FisionGuy(planet.getOwner(),planet.getDeathFactor())
    FinSino
        Escribir ("Guerrero no encontrado")
    Fin
    return warriorsSent
Fin
```

Retorna una clase Guerrero dependiendo del nombre daod

**checkVictory()**

```

public boolean checkVictory(var numberPlanets)
    var counter=0
    para(var x=0  x<map.getMap().length  x++)
        para(var y=0  y<map.getMap()[0].length  y++)
            Si(map.getMap()[x][y] != null)
                try
                    Si(map.getMap()[x][y].getOwner()!=null)
                        Si(map.getMap()[x][y].getOwner().equalsIgnoreCase(player))
                            counter++
                        Fin
                    Fin
                Fincatch(Throwable e)
                    Escribir ("Ha ocurrido un error")
                    return false
                Fin
            Fin
        Fin
    Fin

    Si(counter == numberPlanets)
        Escribir ("El jugador "+player+" Ha ganado")
        return true
    Fin

    return false
Fin

```

Comprueba si hay un ganador**getDistance()**

```

private boolean getDistance()
    Scanner scanner = new Scanner(System.in)
    var[] planeta
    var indexCols1
    var indexRows1
    var indexCols2
    var indexRows2
    WarriorHandler warriorHandler = new WarriorHandler()

    Escribir ("Coloca el Planeta de Origen y luego el planeta al dirigirse, como A1,B2")
    planeta = scanner.nextLine().toCharArray()

    Si(planeta.length!=5)
        Escribir ("Comando mal escrito")
        return false
    Fin
    indexCols1 = planeta[0] - 65
    indexRows1 = planeta[1] - 49

```

```
indexCols2 = planeta[3] -65
indexRows2 = planeta[4] -49
```

```
warriorHandler.getDistance(map.getMap(), map.getMap()[indexRows1]
[indexCols1].getName(), map.getMap()[indexRows2][indexCols2].getName())
return true
Fin
```

Es la accion de obtener distancias

### **getProperty()**

```
private void getProperty()
    para(var x=0 x<map.getMap().length x++)
        para(var y=0 y<map.getMap()[0].length y++)
            Si(map.getMap()[x][y]!= null)
                Si( map.getMap()[x][y].getOwner()!= null)
                    Si(map.getMap()[x][y].getOwner().equalsIgnoreCase(player))
                        map.getMap()[x][y].getData()
                        map.getMap()[x][y].showShip(0)
                        map.getMap()[x][y].showBuilder()
                    Fin
                Fin
            Fin
        Fin
    Fin
```

Fin

Obtiene las propiedades del Usuario

### **getShipsOnGoing()**

```
private void getShipOnGoing()
    para(var x=0 x<map.getMap().length x++)
        para(var y=0 y<map.getMap()[0].length y++)
            Si(map.getMap()[x][y] != null)
                Si(map.getMap()[x][y].getOwner()!= null)
                    Si(map.getMap()[x][y].getOwner().equalsIgnoreCase(player))
                        map.getMap()[x][y].showShip(1)
                    Fin
                Fin
            Fin
        Fin
    Fin
```

Obtiene Datos de las naves que se encuentran en el espacio