



FACULTAD DE INGENIERÍAS  
INGENIERÍA EN SISTEMAS (3743)  
FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS- 01  
Natalia Andrea Marín Hernández – 202041622-3743  
Sergio Escudero Tabares - 2040801-3743

---

**MINI PROYECTO 1: GEEK OUT MASTERS**  
**ANÁLISIS DE CLASES**

DADO

• **Int calculateCara()**

La función calculateCara() elige un número aleatorio entre 1 y 6, para definir la cara del dado que se muestra del dado en el juego y que efectos tiene.

Retorna un entero que es el valor de la cara, siendo reconocidos de la siguiente manera:

```
public int calculateCara() {  
    Random aleatorio = new Random();  
    cara = aleatorio.nextInt( bound: 6 ) + 1;  
    return cara;  
}
```

1. Meeple
2. Dragon
3. Corazón

4. Cohete
5. Superhéroe
6. 42

GUIGRIDBAGLAYOUT

• **void initGUI()**

La función initGUI() Inicializa todos los elementos que se muestran y que el main ejecuta durante todo el programa; No retorna nada.

```
private void initGUI() {  
    //Set up JFrame Container's Layout  
    this.getContentPane().setLayout(new GridBagLayout());  
    GridBagConstraints constraints = new GridBagConstraints();  
  
    //Create Listener Object or Control Object  
    escucha = new Escucha();  
    objetoModelGeekOutMaster = new ModelGeekOutMaster();  
    objetoDado = new Dado();  
    vectorDatosInactivos = new JButton[7];  
    vectorDatosActivos = new JButton[7];  
    vectorPuntajes = new JLabel[10];
```

- **Void reiniciarJuego()**

La función reiniciarJuego() reestablece todos los valores de tanto dados como el estado del juego, la puntuación, entre otros.

No retorna nada.

```
public void reiniciarJuego() {
    lanzar.setEnabled(true);

    imageDado = new ImageIcon(getClass().getResource("name: /resources/neutro.png"));
    puntaje1.setIcon(null);
    puntaje2.setIcon(null);
    puntaje3.setIcon(null);
    puntaje4.setIcon(null);
    puntaje5.setIcon(null);
    puntaje6.setIcon(null);
    puntaje7.setIcon(null);
    puntaje8.setIcon(null);
    puntaje9.setIcon(null);
```

- **void pasarRonda()**

La función pasarRonda() pasa de ronda al asignar los dados con efecto “42” a la sección de puntajes y cambia el estado del juego para permitirle al jugador continuar con la partida.

No retorna nada.

```
public void pasarRonda() {
    lanzar.setEnabled(true);

    imageDado = new ImageIcon(getClass().getResource("name: /resources/neutro.png"));

    for(int i = 0;i<10;i++){
        if(vectorPuntajes[i].getIcon() == null){
            if(i>=contador42Juego){
                i = 11;
            }else{
                vectorPuntajes[i].setIcon(imagen42);
            }
        }
    }
}
```

- **boolean vectorVacio (JButton[] vectorAVerificar)**

La función vectorVacio(JButton[] vectorAVerificar) reconoce cuando un vector está vacío, retornando “true” o “false” en caso de no estarlo.

```
public boolean vectorVacio (JButton[] vectorAVerificar){
    for(int i = 0;i<7;i++){
        if(vectorAVerificar[i]!=null){
            return true;
        }
    }
    return false;
}
```

- **Void efectos()**

La función efectos() define los efectos de cada dado dependiendo del valor de su cara a través de un switch con cada caso especificado.

```

public void efectos(int efecto, JButton dadoConAccion, JButton dadoAAccionar){

    System.out.println(efecto + " - " + dadoConAccion.getName() + " - " + dadoAAccionar.getName());

    int[] caras = objetoModelGeekOutMaster.getCaras();
    System.out.println(caras[0] + " " + caras[1] + " " + caras[2] + " " + caras[3] + " " + caras[4] + " " + caras[5] + " " + caras[6] + " " + caras[7]);

    int nuevoValorDado = 0;

    switch (efecto){

```

En el caso 1: meeple

Saca el dado del panel de dados activos y lo añade al panel de dados utilizados. Activa la función calculateCara() en el dado seleccionado después del meeple.

```

    case 1:
        dadoAccion.setEnabled(false);
        dadosActivos.remove(dadoConAccion);
        dadosUtilizados.add(dadoConAccion);

        for(int i = 0;i<7;i++){
            if(vectorDadosActivos[i]==dadoConAccion){
                vectorDadosActivos[i]=null;
                i = 8;
            }
        }

        nuevoValorDado = objetoDado.calculateCara();
        System.out.println("El nuevo valor del dado es: " + nuevoValorDado);
        imageDado = new ImageIcon(getClass().getResource("resources/" + nuevoValorDado + ".png"));
        dadoAAccionar.setIcon(imageDado);

```

En el caso 2: Dragón

No hace nada.

```

case 2:
    //"Dragon"
    break;

```

En el caso 3: Corazón

Saca el dado del panel de dados activos y lo añade al panel de dados utilizados.

Toma un dado de inactivos, activa la función calculateCara() en dicho dado y lo agrega al panel de dados activos. En caso de no haber dados en inactivos, no hace nada.

```

    case 3:
        //"Corazon"
        dadosActivos.remove(dadoConAccion);
        dadosUtilizados.add(dadoConAccion);
        dadoConAccion.setEnabled(false);

        for(int i = 0;i<7;i++){
            if(vectorDadosActivos[i]==dadoConAccion){
                vectorDadosActivos[i]=null;
                i = 8;
            }
        }

        nuevoValorDado = objetoDado.calculateCara();

        System.out.println("El nuevo valor del dado es: " + nuevoValorDado);
        imageDado = new ImageIcon(getClass().getResource("resources/" + nuevoValorDado + ".png"));
        dadoAAccionar.setIcon(imageDado);

        switch (dadoAAccionar.getName()){
            case "dado1":
                objetoModelGeekOutMaster.campeal = nuevoValorDado;

```

#### En el caso 4: Cohete

Saca el dado del panel de datos activos y lo añade al panel de datos utilizados.

Saca el dado a accionar de datos activos y lo añade a inactivos.

```
case 4:  
    //Cohete  
    dadosActivos.remove(dadoConAccion);  
    dadosUtilizados.add(dadoConAccion);  
    dadoAccion.setEnabled(false);  
  
    for(int i = 0;i<7;i++){  
        if(vectorDadosActivos[i]==dadoConAccion){  
            vectorDadosActivos[i]=null;  
            i = 8;  
        }  
    }  
  
    dadosActivos.remove(dadoAccionar);  
    dadosInactivos.add(dadoAccionar);  
    dadoAccionar.setEnabled(false);  
  
    for(int i = 0;i<7;i++){  
        if(vectorDadosActivos[i] == dadoAccionar){  
            vectorDadosActivos[i] = null;  
            i = 8;  
        }  
    }  
}
```

#### En el caso 5: Superhéroe

Saca el dado del panel de datos activos y lo añade al panel de datos utilizados.

Cambia el valor del dado seleccionado de la siguiente manera:

- 1: Meeple ↔ 4: Cohete
- 2: Dragón ↔ 5: Superhéroe
- 3: Corazón ↔ 6: 42

```
case 5:  
    //Superheroe  
    dadosActivos.remove(dadoConAccion);  
    dadosUtilizados.add(dadoConAccion);  
    dadoAccion.setEnabled(false);  
  
    for(int i = 0;i<7;i++){  
        if(vectorDadosActivos[i]==dadoConAccion){  
            vectorDadosActivos[i]=null;  
            i = 8;  
        }  
    }  
}
```

#### En el caso 6: 42

No hace nada.

```
break;  
case 6:  
    //42  
  
    break;
```

- **void terminarRonda()**

Activa la función pasarRonda() y aumenta el número de la ronda, además de reiniciar el valor de los dados.

No retorna nada.

```
public void terminarRonda(){  
  
    puedeFinalizarRonda = calcularEstadoRonda();  
  
    if(puedeFinalizarRonda == true){  
  
        ronda = ronda + 1;  
        rondaActual.setText("Ronda actual: " + ronda);  
        puntuacionJuego.setText("Puntuación total: " + puntosJuego);  
        pasarRonda();  
  
    }else{  
        JOptionPane.showMessageDialog( parentComponent: null, message: "Aún puedes realizar movimientos.");  
    }  
}
```

- **boolean calcularEstadoRonda()**

Calcula si la ronda puede finalizar o no, retorna true en los casos que pueda pasar de ronda, por ejemplo: Uno o más dados con cara “42” sin haber dragones en dados activos.

```
public boolean calcularEstadoRonda(){

    int valorAyuda = 0;

    for(int i = 0;i<vectorDadosActivos.length;i++){
        if(vectorDadosActivos[i] != null){
            valorAyuda = objetoModelGeekOutMaster.getCarácter(vectorDadosActivos[i]);
            if(valorAyuda==6){
                contador42 = contador42 + 1;
                System.out.println("Se registró un 42");
            }else if(valorAyuda==2){
                contadorDragones = contadorDragones + 1;
                System.out.println("Se registró un dragon");
            }else{
                contadorOtrosDados = contadorOtrosDados + 1;
                System.out.println("Se registró otra cosa");
            }
        }
    }
}
```

- **void sacarInactivo()**

La función sacarInactivo() determina que dado se puede sacar del vector de dados inactivos. No retorna nada.

```
public void sacarInactivo(JButton dadoAccionado){

    int caraAsistente = 0;

    for(int i = 0;i<7;i++){
        if(vectorDadosInactivos[i] != null){
            asistente = i;
        }
    }

    if(vectorDadosInactivos[asistente]==null){
        dadosActivos.remove(dadoAccionado);
        dadosUtilizados.add(dadoAccionado);
        dadoAccionado.setEnabled(false);

        for(int i = 0;i<7;i++){
            if(vectorDadosActivos[i]==dadoAccionado){
                vectorDadosActivos[i]=null;
                i = 8;
            }
        }
        JOptionPane.showMessageDialog(null, "No hay ningún dado en inactivos");
    }
}
```

## ESCUCHA

- **void actionPerformed()**

Selecciona cuál dado accionará a otro.

```
@Override
public void actionPerformed(ActionEvent e) {
    /*
```

## MODEL GEEK OUT MASTERS

- **void calcularDatos()**

Activa la función calculateCara() para cada uno de los dados.

```
public void calcularDatos(){
    caras[0] = dado1.calculateCara();
    caras[1] = dado2.calculateCara();
    caras[2] = dado3.calculateCara();
    caras[3] = dado4.calculateCara();
    caras[4] = dado5.calculateCara();
    caras[5] = dado6.calculateCara();
    caras[6] = dado7.calculateCara();
    caras[7] = dado8.calculateCara();
    caras[8] = dado9.calculateCara();
    caras[9] = dado10.calculateCara();
}
```

- **int getCara()**

La función getCara() obtiene el valor de la cara del dado que se le dé.

```
public int getCara(JButton dadoauxiliar){
    if(dadoauxiliar.getName() == "dado1"){
        return caras[0];
    }else if(dadoauxiliar.getName() == "dado2"){
        return caras[1];
    }else if(dadoauxiliar.getName() == "dado3"){
        return caras[2];
    }else if(dadoauxiliar.getName() == "dado4"){
        return caras[3];
    }else if(dadoauxiliar.getName() == "dado5"){
        return caras[4];
    }else if(dadoauxiliar.getName() == "dado6"){
        return caras[5];
    }else if(dadoauxiliar.getName() == "dado7"){
        return caras[6];
    }else if(dadoauxiliar.getName() == "dado8"){
        return caras[7];
    }else if(dadoauxiliar.getName() == "dado9"){
        return caras[8];
    }else if(dadoauxiliar.getName() == "dado10"){
        return caras[9];
    }
    return 0;
}
```

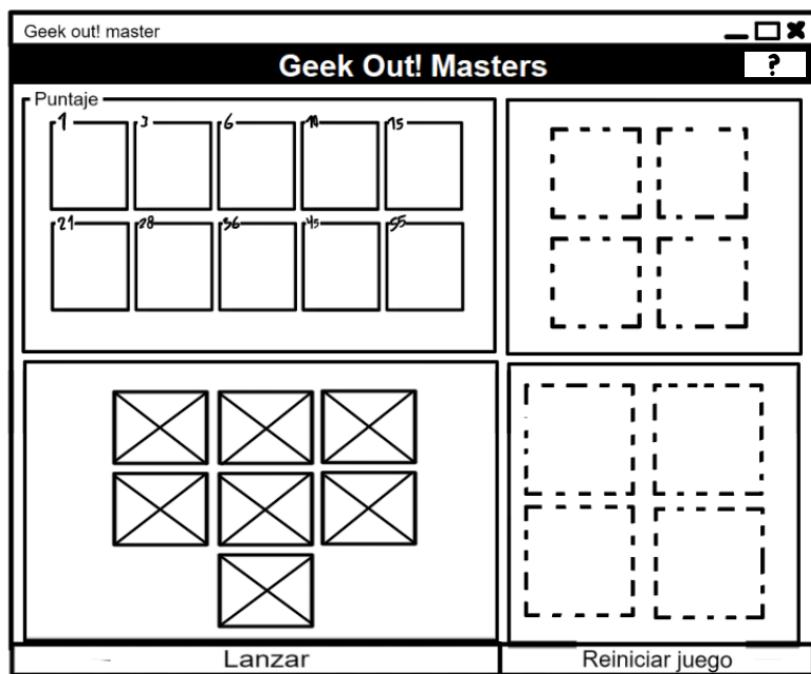
- **int getCaras()**

Retorna la cara del dado.

```
public int[] getCaras() { return caras; }
```

## **BOCETO DE INTERFAZ**

Pantalla principal



Pantalla de ayuda

