



# Pylatro

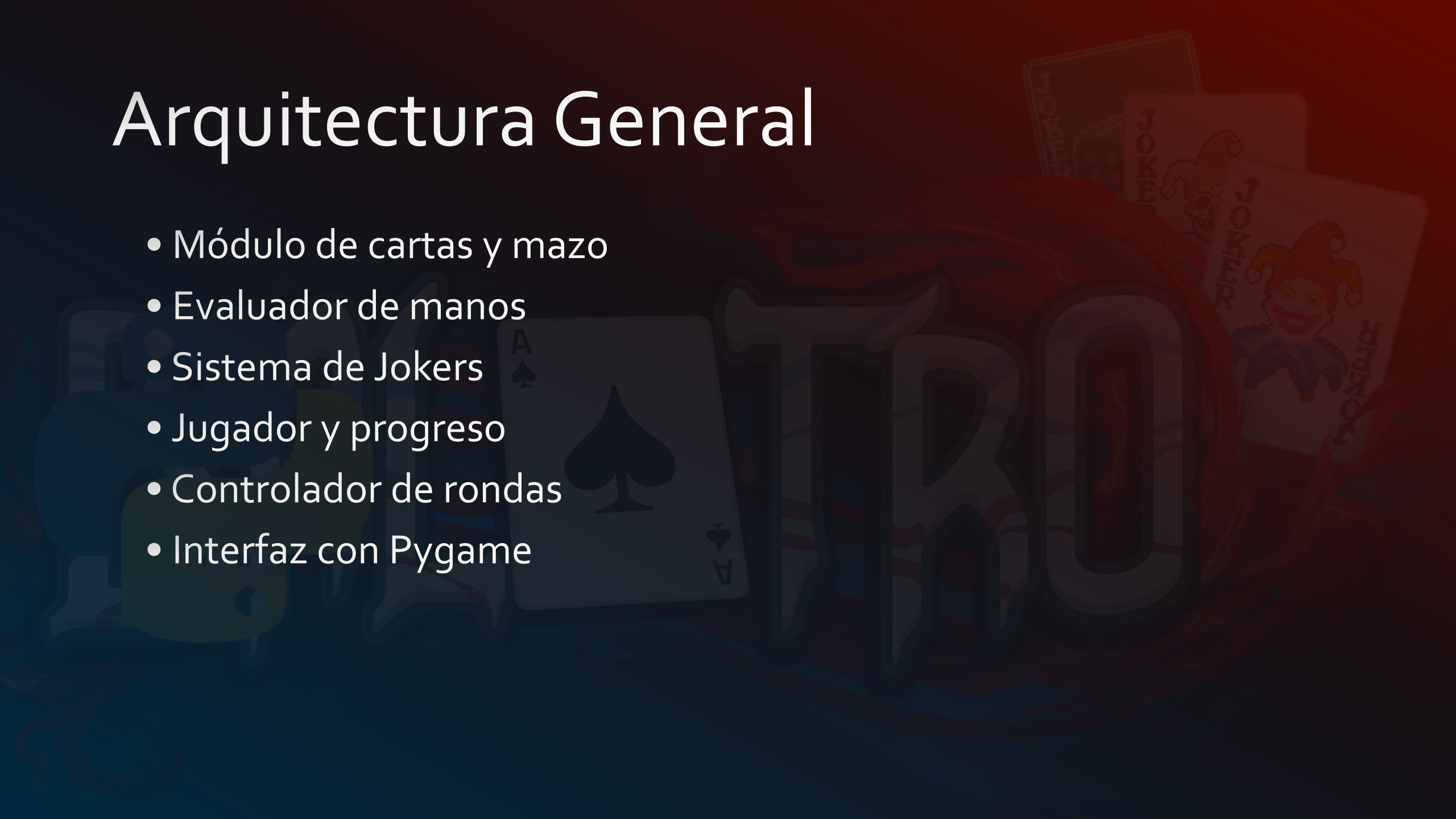
Una recreación de Balatro usando Python y Pygame





# Arquitectura General

- Módulo de cartas y mazo
- Evaluador de manos
- Sistema de Jokers
- Jugador y progreso
- Controlador de rondas
- Interfaz con Pygame





# Modelo de Cartas

- Cartas representadas como tuplas (valor, palo)
- Mazo generado dinámicamente



```
def crearMazo():  
    palos = ["trebol", "picas", "diamantes", "corazones"]  
    mazo = [(r, palo) for palo in palos for r in range(1, 14)]  
    random.shuffle(mazo)  
    return mazo
```

# Evaluación de Manos

- Se cuenta primero la repetición de valores, y se compara:



```
hayPar = any(x == 2 for x in contarNums.values())
hay2Par = sum(1 for x in contarNums.values() if x == 2)
hayTrio = [x for x,c in contarNums.items() if c == 3]
hayPoker = [x for x,c in contarNums.items() if c == 4]
hayQuintillo = [x for x,c in contarNums.items() if c == 5]
hayColor = any(palos.count(p) == 5 for p in palos)
hayFull = (len(hayTrio) == 1 and hay2Par == 1)
```



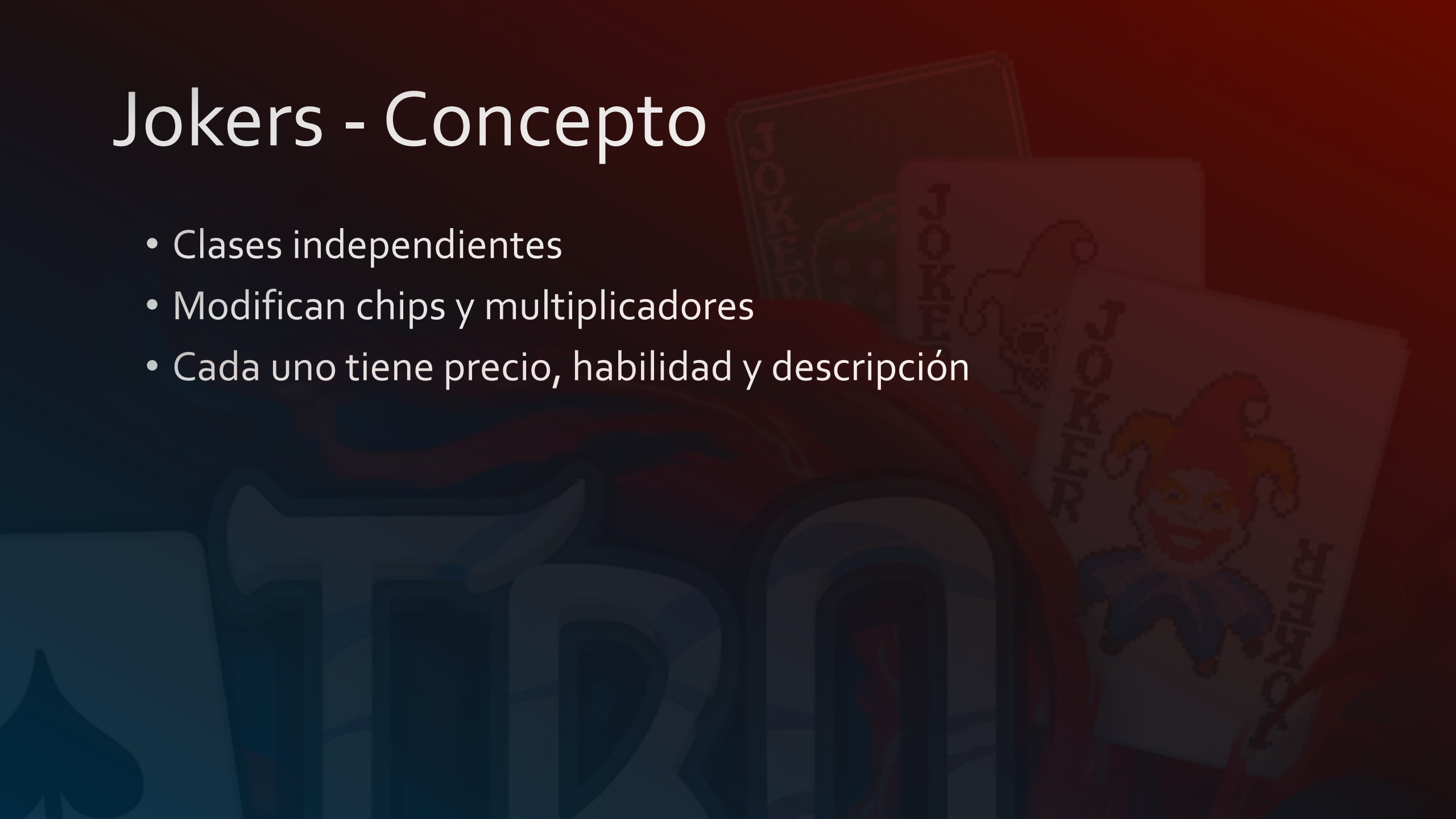






# Jokers - Concepto

- Clases independientes
- Modifican chips y multiplicadores
- Cada uno tiene precio, habilidad y descripción





```
class Joker:
    def __init__(self, nombre, multiplicadorAnadir, chips, precio, multiplicador):
        self.nombre = nombre
        self.multiplicadorAnadir = multiplicadorAnadir
        self.chips = chips
        self.precio = precio
        self.multiplicador = multiplicador
    def habilidad(self, mano=None):
        pass
    def reset(self):
        self.multiplicadorAnadir = 0
        self.chips = 0
        self.multiplicador = 1
    def descripcion(self):
        return "Joker genérico"
    def __repr__(self):
        return f"<Joker {self.nombre}>"
```



```
class JollyJoker(Joker):
    def __init__(self):
        super().__init__("Jolly Joker", 0, 0, 4, 1)
    def habilidad(self, mano):
        # si hay un par -> multiplicadorAnadir = 8
        valores = [v for v,s in mano]
        counts = {}
        for v in valores: counts[v] = counts.get(v,0) + 1
        if any(c==2 for c in counts.values()):
            self.multiplicadorAnadir = 8
    def descripcion(self):
        return "Si hay un par, aumenta en 8 el multiplicador"
```



# Jugador

- Métodos:
  - comprar\_joker()
  - mostrarJokers()
  - obtenerJokers()

```
def comprar_joker(self, joker: Joker):  
    # compra un joker si hay espacio y dinero suficiente  
    if len(self.jokers) >= 5:  
        return False, "Máximo de jokers alcanzado (5)."  
    if self.dinero < joker.precio:  
        return False, "No tenés suficiente dinero."  
    self.jokers.append(joker)  
    self.dinero -= joker.precio  
    return True, f"Compraste {joker.nombre}."  
  
def mostrarJokers(self) -> List[str]:  
    # nombres de los jokers del jugador  
    return [j.nombre for j in self.jokers]  
  
def obtenerJokers(self) -> List[Joker]:  
    # devuelve la lista de jokers  
    return self.jokers
```

# Rondas - Estructura

- Incrementa la dificultad, la cantidad requerida de fichas aumenta 150 por ronda
- Cada 3 rondas el valor se duplica

```
def objetivo_ronda(n: int) -> int:
    # devuelve el objetivo de puntos para la ronda n
    bloques = [
        [300, 450, 600],
        [900, 1200, 1500],
        [2100, 2700, 3000],
    ]
    bloque = (n - 1) // 3
    index = (n - 1) % 3

    if bloque < len(bloques):
        return bloques[bloque][index]

    factor = 2 ** (bloque - len(bloques) + 1)
    base = bloques[-1][index] if index < 3 else 300 * (index + 1)
    return base * factor
```

# Código - Objetivos



```
def objetivo_ronda(n: int) -> int:
    # devuelve el objetivo de puntos para la ronda n
    bloques = [
        [300, 450, 600],
        [900, 1200, 1500],
        [2100, 2700, 3000],
    ]
    bloque = (n - 1) // 3
    index = (n - 1) % 3

    if bloque < len(bloques):
        return bloques[bloque][index]

    factor = 2 ** (bloque - len(bloques) + 1)
    base = bloques[-1][index] if index < 3 else 300 * (index + 1)
    return base * factor
```



# Interfaz Gráfica

- Cartas en abanico
- HUD dinámico
- Botones interactivos



# Tienda de Jokers

- Compra por dinero Ganado
- Cada Joker cambia el cálculo final de puntos

## TIENDA DE JOKERS

Tu dinero: \$13

### Joker

Aumenta en 4 el multiplicador

Precio: \$4

COMPRAR

### Smiley Face

Por cada carta J, Q o K aumenta en 5 el multiplicador

Precio: \$6

COMPRAR

