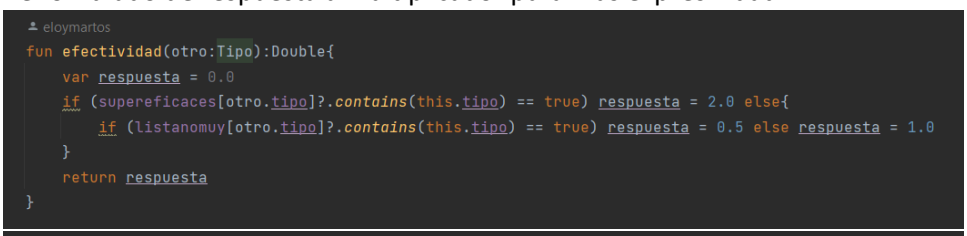
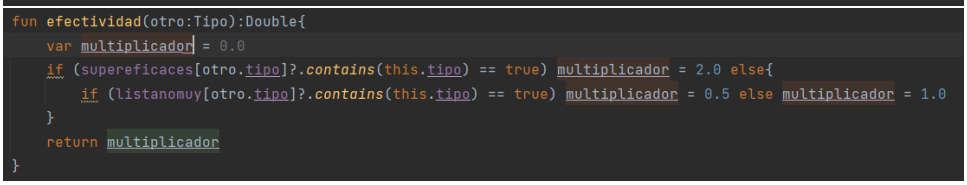
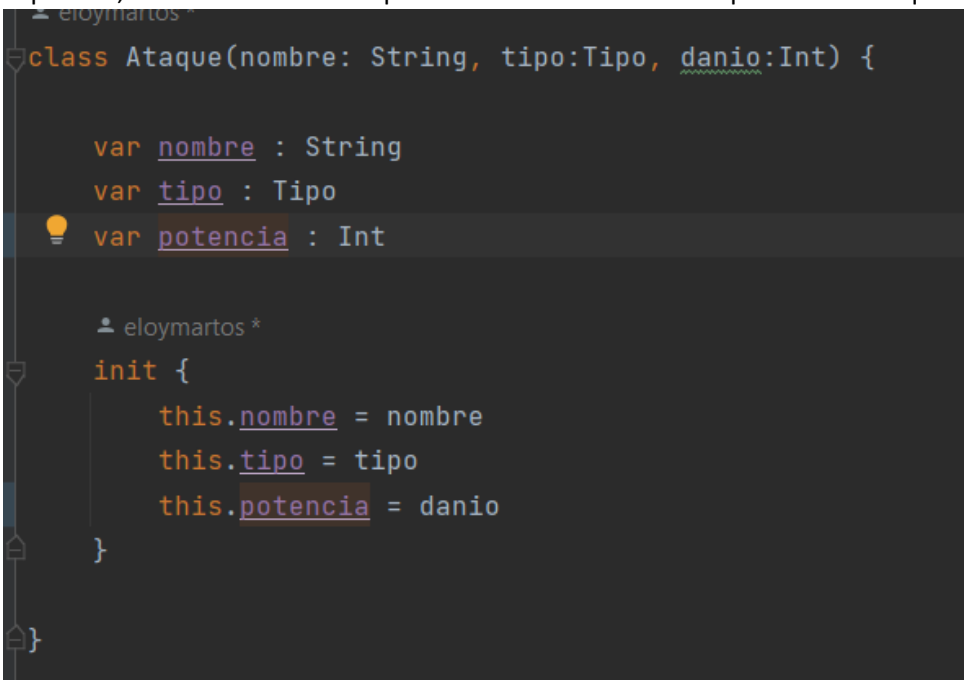
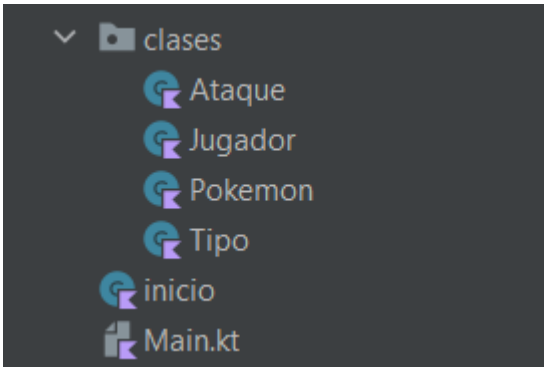
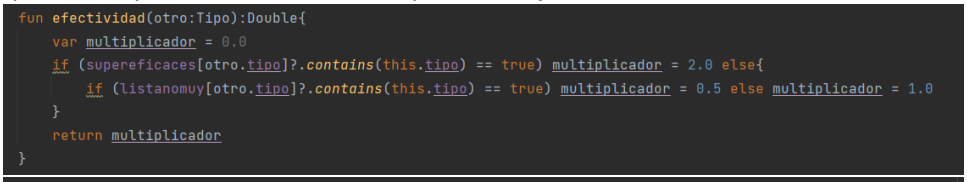
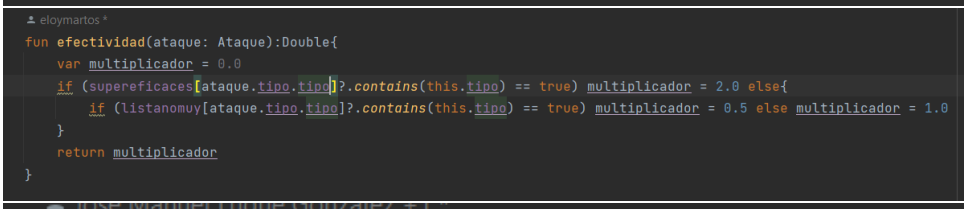
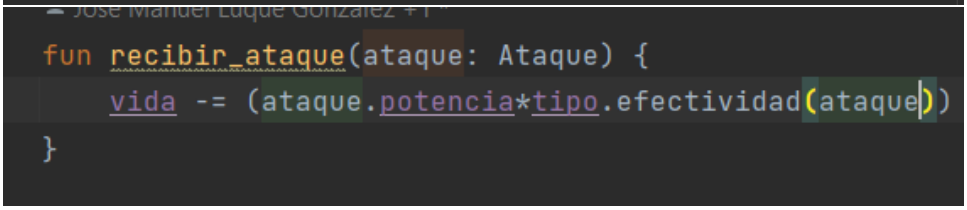
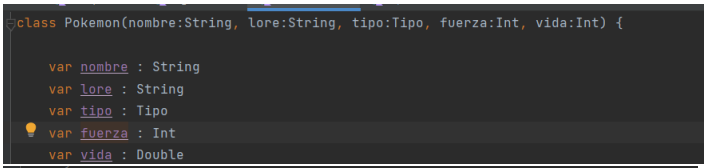
Refactorización del codigo del proyecto Pokemon :

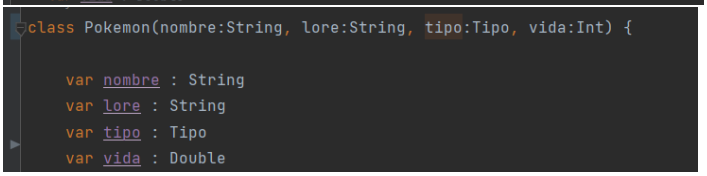
- Renombrado
  - Renombrado de respuesta a multiplicador para mas expresividad
  - 

  - 

  - Renombrado de danio a potencia en ataque porque nos daba toc y porque es mas expresivo, danio suena mas a representar el daño definitivo que hace el ataque
  - 

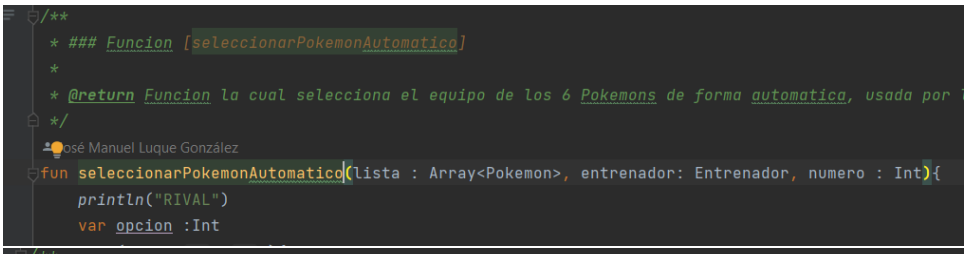
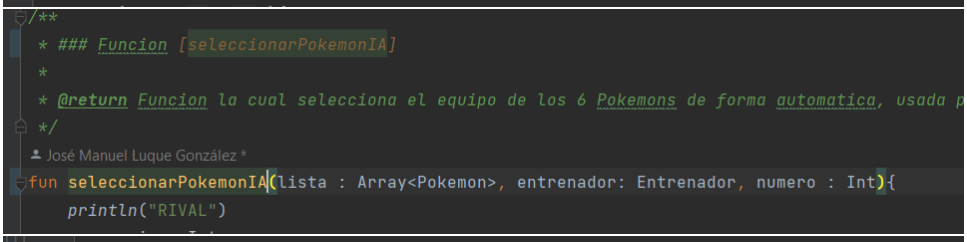
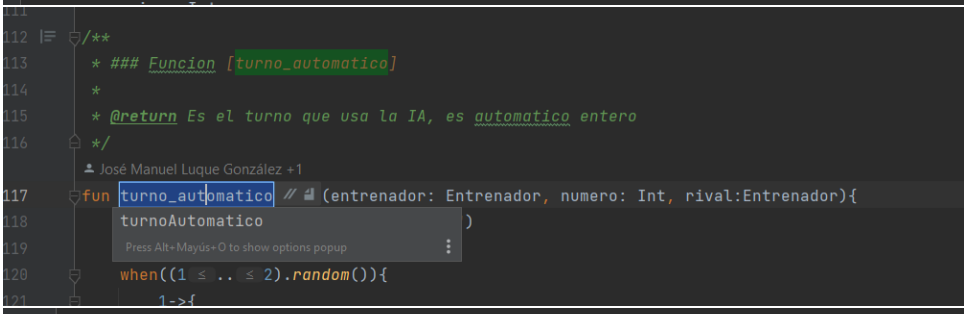
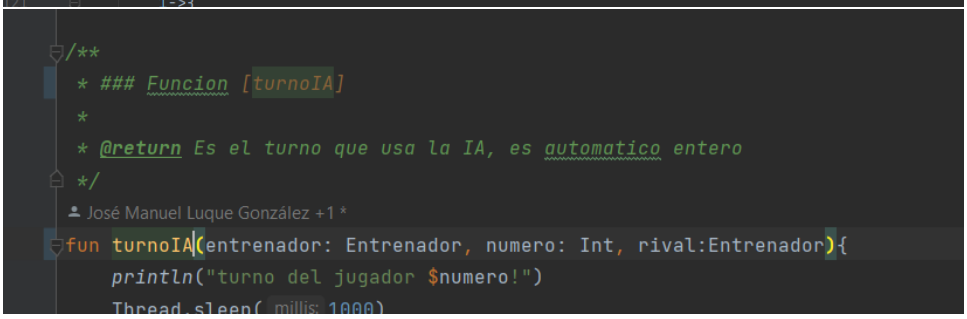
- Sustitucion de bloques
  - -
- Campos encapsulados
  - Ya lo están
- Movimieto de clases
  - 

- Extraer interfaz
  - -
- Cambios de parámetros
  - Efectividad ahora recibe directamente el ataque y saca por su cuenta el tipo, para que la comprension en recibir ataque sea mejor
  - 
```
fun efectividad(otro:Tipo):Double{
    var multiplicador = 0.0
    if (supereficaces[otro.tipo]?.contains(this.tipo) == true) multiplicador = 2.0 else{
        if (listanomuy[otro.tipo]?.contains(this.tipo) == true) multiplicador = 0.5 else multiplicador = 1.0
    }
    return multiplicador
}
```
  - 
```
eloymartos *
fun efectividad(ataque: Ataque):Double{
    var multiplicador = 0.0
    if (supereficaces[ataque.tipo.tipo]?.contains(this.tipo) == true) multiplicador = 2.0 else{
        if (listanomuy[ataque.tipo.tipo]?.contains(this.tipo) == true) multiplicador = 0.5 else multiplicador = 1.0
    }
    return multiplicador
}
```
  - 
```
fun recibir_ataque(ataque: Ataque) {
    vida -= (ataque.potencia*tipo.efectividad(ataque))
}
```
- Mover a otro nivel
  - -
- Borrado seguro
  - Borrado de fuerza de la clase pokemon, no se usa en ningun momento
  - 
```
class Pokemon(nombre:String, lore:String, tipo:Tipo, fuerza:Int, vida:Int) {

    var nombre : String
    var lore : String
    var tipo : Tipo
    var fuerza : Int
    var vida : Double
```
  - 
```
class Pokemon(nombre:String, lore:String, tipo:Tipo, vida:Int) {

    var nombre : String
    var lore : String
    var tipo : Tipo
    var vida : Double
```

Refactorizacion de codigo COMPLETO de pokemon

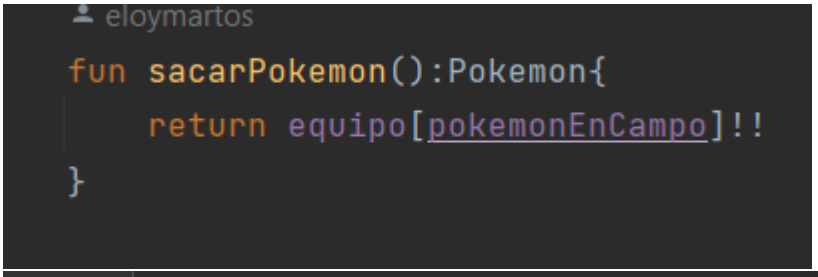
- Renombrado
  - 
```
/**
 * ### Funcion [seleccionarPokemonAutomatico]
 *
 * @return Funcion la cual selecciona el equipo de los 6 Pokemons de forma automatica, usada por
 */
José Manuel Luque González
fun seleccionarPokemonAutomatico(lista : Array<Pokemon>, entrenador: Entrenador, numero : Int){
    println("RIVAL")
    var opcion :Int
```
  - 
```
/**
 * ### Funcion [seleccionarPokemonIA]
 *
 * @return Funcion la cual selecciona el equipo de los 6 Pokemons de forma automatica, usada p
 */
José Manuel Luque González *
fun seleccionarPokemonIA(lista : Array<Pokemon>, entrenador: Entrenador, numero : Int){
    println("RIVAL")
```
  - 
```
/**
 * ### Funcion [turno_automatico]
 *
 * @return Es el turno que usa la IA, es automatico entero
 */
José Manuel Luque González +1
fun turno_automatico (entrenador: Entrenador, numero: Int, rival:Entrenador){
    turnoAutomatico
    Press Alt+Mayús+O to show options popup
    when((1 ≤ .. ≤ 2).random()){
        1->{
```
  - 
```
/**
 * ### Funcion [turnoIA]
 *
 * @return Es el turno que usa la IA, es automatico entero
 */
José Manuel Luque González +1 *
fun turnoIA(entrenador: Entrenador, numero: Int, rival:Entrenador){
    println("turno del jugador $numero!")
    Thread.sleep( millis: 1000)
```

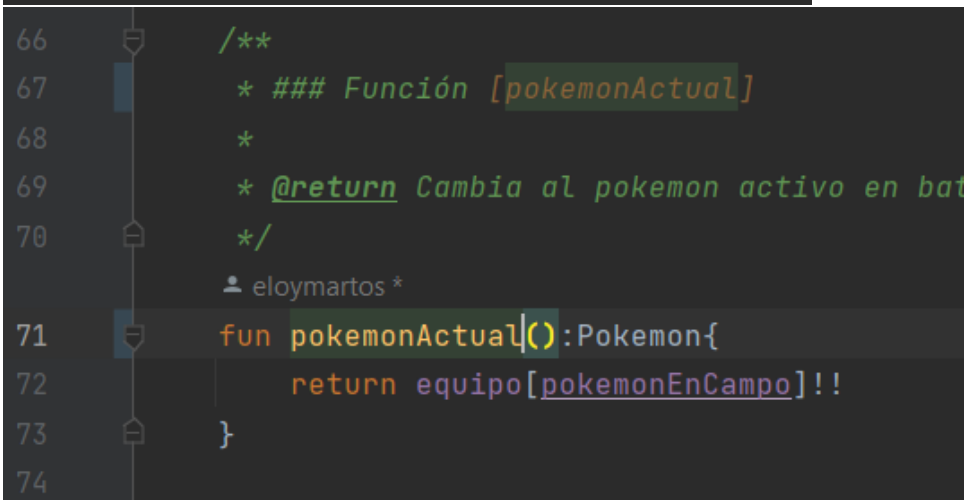
-
```
/**
 * ### Funcion [ataqueAutomatico]
 *
 * @return Funcion que usa la IA para atacar.
 */
eloymartos
fun ataqueAutomatico // (): Ataque{
    return ataques.random()
}
```

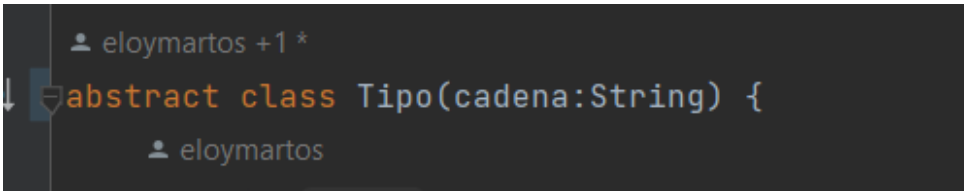
-
```
    /**
     * ### Funcion [ataqueIA]
     *
     * @return Funcion que usa la IA para atacar.
     */
    eloymartos
    fun ataqueIA // (): Ataque{
        return ataques.random()
    }
}
```

-
```
fun asignarAtaques(listamovimientos : Array<Ataque>){
    var contador = 0
    while (true){
        var i = listamovimientos.random()
        if(tipo.efectividad(i) == 1.0 || tipo == i.tipo){
            ataques.add(i)
            contador ++
            if(contador == 4) break
        }
```

-
```
fun asignarAtaques(listamovimientos : Array<Ataque>){
    var contador = 0
    while (true){
        var nuevo = listamovimientos.random()
        if(tipo.efectividad(nuevo) == 1.0 || tipo == nuevo.tipo){
            ataques.add(nuevo)
            contador ++
            if(contador == 4) break
        }
    }
```

- 

- 

- Sustitucion de bloques
  - Funciones del main = combate1vs1, combatevsIA, turno, turnoia, seleccionarPokemon, seleccionarpokemonIA
- Campos Encapsulados
  - -
- MOVIMIENTO DE CLASES
  - Ya hecho en la refactorizacion anterior
- Extraer interfaz
  - -
- CAMBIO DE PARÁMETROS
  - -
- Mover a otro nivel
  - Clase tipo ahora es abstracta y ataque y pokemon heredan de ella
  - 

-
```
eloymartos *
class Ataque(nombre: String, tipo:String, danio:Int):Tipo( cadena: "") {

    var nombre : String
    var potencia : Int

    eloymartos
    init {
        this.nombre = nombre
        this.tipo = tipo
        this.potencia = danio
    }
```
-
```
class Pokemon(nombre:String, lore:String, tipo:String, vida:Int):Tipo( cadena: "") {

    var nombre : String
    var lore : String
    var vida : Double
    var ataques : MutableList<Ataque> = mutableListOf<Ataque>()

    eloymartos
    init {
        this.nombre = nombre
        this.lore = lore
        this.tipo = tipo
        this.vida = vida.toDouble()
    }
```
- Las funciones en las que se accedia a este mediante atributos se han cambiado para su correcto funcionamiento

- BORRADO SEGURO
  - Borrado de jugador, ya no lo usamos
  -