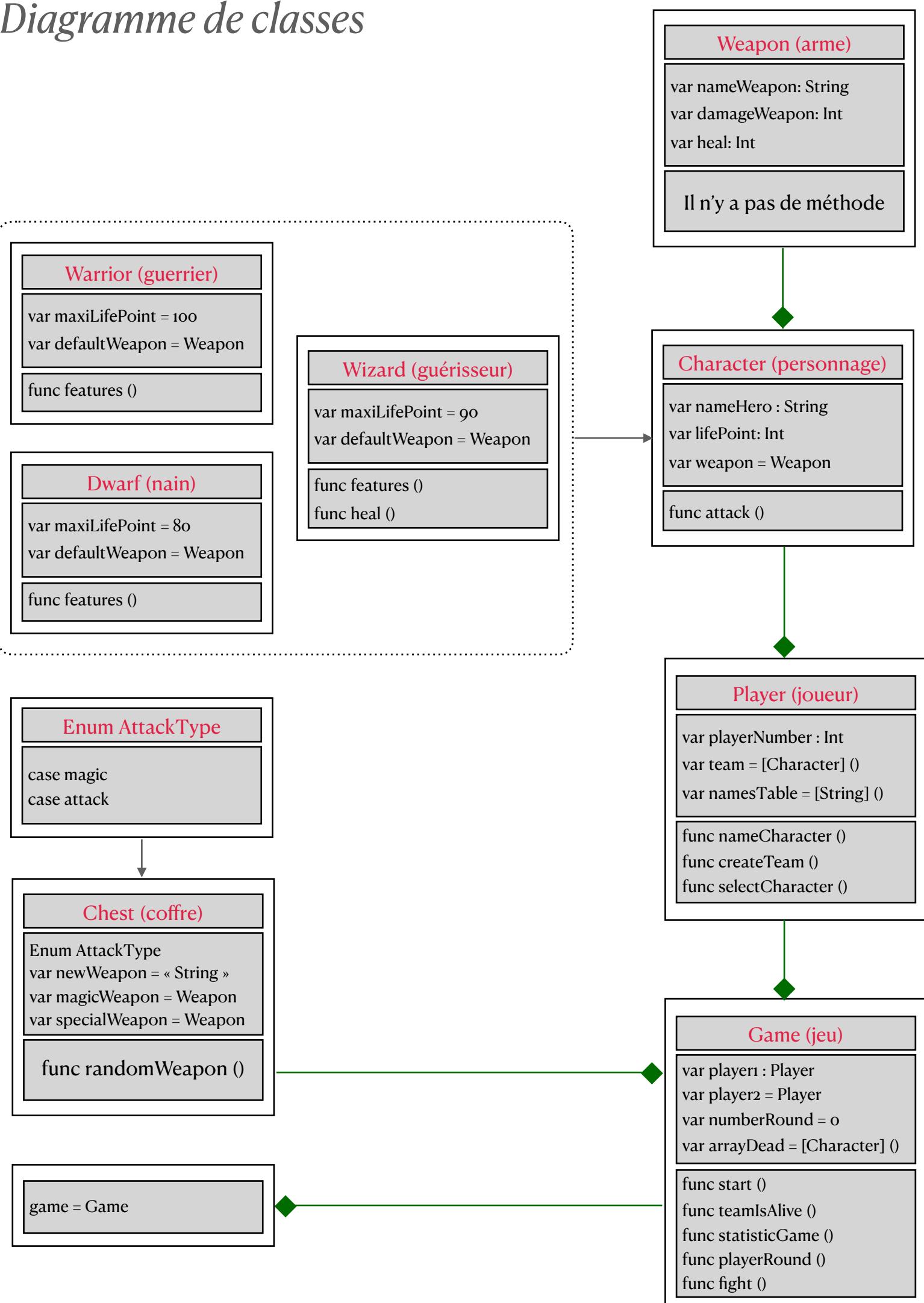


# Diagramme de classes



1

```

11 class Weapon {
12
13
14 //MARK: PROPRIETES
15
16 // On a besoin des variables pour le nom d'arme et le nombre de dégât
17 var nameWeapon: String
18 var damageWeapon: Int
19
20 // On utilise le soin uniquement le personnage Wizard
21 var heal: Int
22
23 // Initialisation des variables de classe
24 init(nameWeapon: String, damageWeapon: Int, heal: Int) {
25     self.nameWeapon = nameWeapon
26     self.damageWeapon = damageWeapon
27     self.heal = heal
28 }
29 }
30 }
```

2

```

11 class Chest {
12
13
14 //MARK: ENUMERATION
15
16 enum AttackType {
17     case magic, attack
18 }
19
20
21 //MARK:- PROPRIETES
22
23 private let newWeapon = ""
24 private let magicWeapon = Weapon(nameWeapon: "Magic Weapon", damageWeapon: 0, heal: 10)
25 private let specialWeapon = Weapon(nameWeapon: "Special Weapon", damageWeapon: 25, heal: 0)
26
27
28 //MARK:- METHODES
29
30 // func randomWeapon (arme aléatoire) lorsque vous tombez sur le coffre
31 func randomWeapon(type: AttackType) -> Weapon? {
32     let randomInt = Int.random(in: 1...3)
33
34     if randomInt == 2 {
35         //Cette condition déduit le type du personnage afin de lui attribuer un coffre approprié.
36         if type == .attack {
37             return specialWeapon
38         } else {
39             return magicWeapon
40         }
41     }
42     // retour nul si la chance n'est pas égale à 2
43     return nil
44 }
45 }
46 }
```

3

```

11 class Character {
12
13
14 //MARK: PROPRIETES
15
16 // On a besoin d'un personnage qui a son nom, son point de vie et son arme
17 var nameHero: String
18 var lifePoint: Int
19 var weapon: Weapon
20
21
22 //MARK: INITIALIZER
23
24 // Initialisation des variables de classe
25 init(namenameHero: String, lifePoint: Int, weapon: Weapon) {
26     self.nameHero = namenameHero
27     self.lifePoint = lifePoint
28     self.weapon = weapon
29 }
30
31
32 //MARK:- METHODES
33
34
35 // func attack permet à un personnage d'infliger ses points de dégâts à l'adversaire. Dans le paramètre "target" c'est un cible au personnage pour l'attaque.
36 func attack(target: Character) {
37     lifePoint -= self.weapon.damageWeapon
38
39     //on affiche
40     print("\(self.nameHero) attaque \(target.nameHero) et lui infligé \(self.weapon.damageWeapon) dégâts. \(target.nameHero) a maintenant \(target.lifePoint) pts de vie")
41
42     // Si le personnage n'a plus de vie
43     if target.lifePoint <= 0 {
44         target.lifePoint = 0
45         print("\(nameHero) n'a plus de vie")
46     }
47 }
48
49 }
```

4

```

11 // class Warrior qui hérite de Character
12 class Warrior: Character {
13
14
15 //MARK: PROPRIETES
16
17 // Le point de vie de Warrior est à 100
18 static var maxLifePoint = 100
19
20 // var defaultWeapon initailise Warrior avec les valeurs d'amers par défaut.
21 static var defaultWeapon = Weapon(nameWeapon: "Sword", damageWeapon: 15, heal: 0)
22
23
24 //MARK: INITIALIZER
25
26 //Initialiser la classe Warrior, grâce à Character initialisé
27 init(nameHero: String) {
28     super.init(namenameHero: nameHero, lifePoint: Warrior.maxLifePoint, weapon: Warrior.defaultWeapon)
29 }
30
31
32
33 //MARK:- METHODES
34
35 // func features (caracteristiques) permettent d'ajouter des statistiques du personnage
36 static func features() -> String {
37
38     //On affiche: Warrior a 100 pts de vie, son arme SWORD et a 15 pts de de dégâts
39     return "Warrior a \(Warrior.maxLifePoint) pts de vie, son arme \(Warrior.defaultWeapon.nameWeapon) et a \(Warrior.defaultWeapon.nameWeapon) pts de dégâts"
40 }
41
42

```

5

```

11 // class Dwarf qui hérite de Character
12 class Dwarf: Character {
13
14
15 //MARK: PROPRIETES
16
17 // Le point de vie de Dwarf est à 80
18 static var maxLifePoint = 80
19
20 // var defaultWeapon initailise Dwarf avec les valeurs d'amers par défaut.
21 static var defaultWeapon = Weapon(nameWeapon: "Hatchet", damageWeapon: 20, heal: 0)
22
23
24 //MARK: INITIALIZER
25
26 //Initialiser la classe Dwarf, grâce à Character initialisé
27 init(nameHero: String) {
28     super.init(namenameHero: nameHero, lifePoint: Dwarf.maxLifePoint, weapon: Dwarf.defaultWeapon)
29 }
30
31
32
33 //MARK:- METHODES
34
35 // func features (caracteristiques) permettent d'ajouter des statistiques du personnage
36 static func features() -> String {
37
38     //On affiche: Dwarf a 80 pts de vie, son arme HATCHET et a 20 pts de de dégâts
39     return "Dwarf a \(Dwarf.maxLifePoint) pts de vie, son arme \(Dwarf.defaultWeapon.nameWeapon) et a \(Dwarf.defaultWeapon.nameWeapon) pts de dégâts"
40 }
41
42

```

6

```

11 // class Wizard qui hérite de Character
12 class Wizard: Character {
13
14
15 //MARK: PROPRIETES
16
17 // Le point de vie de Wizard est à 90
18 static var maxLifePoint = 90
19
20 // var defaultWeapon initailise Wizard avec les valeurs d'amers par défaut.
21 static var defaultWeapon = Weapon(nameWeapon: "Electrical Attack", damageWeapon: 5, heal: 10)
22
23
24 //MARK: INITIALIZER
25
26 //Initialiser la classe Wizard, grâce à Character initialisé. Paramètre nameHero appelle la valeur stockée dans le tableau.
27 init(nameHero: String) {
28     super.init(namenameHero: nameHero, lifePoint: Wizard.maxLifePoint, weapon: Wizard.defaultWeapon)
29 }
30
31
32
33 //MARK:- METHODES
34
35 // func features (caracteristiques) permettent d'ajouter des statistiques du personnage
36 static func features() -> String {
37
38     //On affiche: Dwarf a 90 pts de vie, son arme ELECTRICAL ATTACK, a 5 pts de de dégâts et a 10 pts de soins
39     return "Dwarf a \(Wizard.maxLifePoint) pts de vie, son arme \(Wizard.defaultWeapon.nameWeapon), a \(Wizard.defaultWeapon.nameWeapon) pts de dégâts et a
40         \(Wizard.defaultWeapon.heal) pts de soins"
41
42
43     func heal(target: Character) {
44         target.lifePoint = target.lifePoint + self.weapon.heal
45     }
46
47

```

```

11 class Player {
12
13
14 //MARK: PROPRIETES
15
16
17 //var playerNumber contiendra le numéro du joueur.
18 var playerNumber: Int
19
20 //var team stocke dans un tableau des personnages choisis par un joueur pour créer une équipe
21 var team = [Character]()
22
23 //var namesTable stocke dans un tableau. Les nom des personnages choisis par le joueur.
24 static private var namesTable = [String]()
25
26
27 //MARK: INITIALIZER
28
29 //Initialiser la variable playerNumber
30 init(playerNumber: Int) {
31     self.playerNumber = playerNumber
32 }
33
34 // MARK: - METHODES
35
36
37
38 //func nameCharacter a mis le nom que le joueur a choisi pour son personnage.
39 //On vérifie que le nom est écrit sans espace de plus 3 caractères et on assure que le nom ne l'est pas déjà.
40 private func nameCharacter() -> String {
41     if let namePlayer = readLine() {
42
43         //Si le nom contient des espaces et ou moins de 3 caractères, un message d'erreur est renvoyé.
44         let choice = namePlayer.trimmingCharacters(in: .whitespaces)
45         if choice.count < 3 || Player.namesTable.contains(choice) {
46             print("Choisissez un nom de plus de 3 caractères et qui n'existe pas encore")
47         } else {
48             Player.namesTable.append(choice)
49
50             return choice
51         }
52     }
53     return nameCharacter()
54 }
55
56
57 //func createTeam permet de créer une équipe.
58 func createTeam() {
59
60     //Si le joueur n'a pas ces 3 personnages, le jeu ne peut pas commencer.
61     while team.count < 3 {
62
63         //On affiche la liste des personnages disponibles et leurs caractéristiques.
64         print("""
65
66             Joueur \(playerNumber) choisissez 3 personnages dans la liste ci-dessous :
67
68             1. \(Warrior.features())
69
70             2. \(Wizard.features())
71
72             3. \(Dwarf.features())
73
74             """)
75
76         //On demande le choix du joueur entre 1 et 3.
77         if let choice = readLine() {
78             switch choice {
79                 case "1" :
80                     print("""
81
82                     Tu as choisi le WARRIOR donne-lui un nom :
83
84                     """)
85
86                     //Demander le nom du personnage
87                     let nameHero = nameCharacter()
88                     let character = Warrior(nameHero: nameHero)
89
90                     //Ajouter le personnage à créer à l'équipe.
91                     team.append(character)
92
93
94                     case "2" :
95                         print("""
96
97                     Tu as choisi le WIZARD donne-lui un nom :
98
99                     """)
100
101                     let nameHero = nameCharacter()
102                     let character = Wizard(nameHero: nameHero)
103
104                     team.append(character)
105
106
107                     case "3" :
108                         print("""
109
110                     Tu as choisi le DWARF donne-lui un nom :
111
112                     """)
113
114                     let nameHero = nameCharacter()
115                     let character = Dwarf(nameHero: nameHero)
116
117                     team.append(character)
118
119                     default:
120                         print("Saisis un chiffre entre 1 et 3")
121                     }
122                 }
123             }
124         }
125         print(Player.namesTable.count)
126         print(Player.namesTable)
127     }
}

```

```

8 import Foundation
9
10 class Game {
11
12
13     // MARK: PROPERTIES
14
15
16     //Les variables qui initialisent le joueur
17     var player1: Player
18     var player2: Player
19
20     //On contient le nombre de tours dans le jeu pour afficher les statistiques à la fin
21     var numberRound = 0
22     var arrayDead = [Character]()
23
24
25     // MARK: INITIALIZER
26
27
28
29     init() {
30         player1 = Player(playerNumber: 1)
31         player1.createTeam()
32
33         player2 = Player(playerNumber: 2)
34         player2.createTeam()
35
36     }
37
38
39     // MARK: - METHODES
40
41
42     //func start affiche les statistiques de combat et de jeu
43     func start() {
44         fight()
45         statisticGame()
46     }
47
48
49     //func teamIsAlive verifie si tous les personnages d'une équipe sont morts
50     private func teamIsAlive(player: Player) -> Bool {
51
52         //Cette boucle "for in" verifie si le personnage 0 pts de vie. Et si c'est le cas, supprimer le tableau
53         for(index, character) in player.team.enumerated() {
54             if character.lifePoint <= 0 {
55                 arrayDead.append(character)
56                 player.team.remove(at: index)
57             }
58         }
59
60         //return false lorsque toute l'équipe était morte
61         if player.team.count == 0 {
62             return false
63         }
64         //return true quand il y au moins un personnage
65         return true
66     }
67
68
69     //func statisticGame affiche le vainqueur et les statistique du jeu
70     private func statisticGame() {
71         print("""
72
73             Désolé toute l'équipe est morte
74
75             """)
76
77         if teamIsAlive(player: player1) {
78             print("""
79
80                 BRAVOOO !!! Le Player1 a gagné!
81
82                 """)
83
84         } else {
85             print("""
86
87                 BRAVOOO !!! Le Player2 a gagné!
88
89                 """)
90         }
91
92         print("""
93
94             Le nombre de tours: \(numberRound)
95
96             """)
97         print("""
98
99             Les Heros morts sont
100
101             """)
102
103         for characterDead in arrayDead {
104             print(" le personnage \(characterDead.nameHero) qui est \(type(of: characterDead)) \(characterDead.lifePoint) ")
105         }
106     }
107 }
```



Voir la suite

```

10
11 //func playerRound gère le tour du joueur. Dans les paramètres, on met attacker et defender
12 private func playerRound(attacker: Player, defender: Player) {
13     //On ajoute chest (le coffre)
14     let chest = Chest()
15
16     //On choisit un personnage dans l'équipe
17     let attackingCharacter = attacker.selectCharacter(team: attacker.team)
18
19     //On vérifie le type pour attribuer un coffre approprié au guerrier ou guérisseur.
20     // Si Wizard (Guérisseur), on attribue une arme de guérison appropriée à Wizard
21     if let wizard = attackingCharacter as? Wizard {
22         if let newWeapon = chest.randomWeapon(type: .magic) {
23             print("""
24
25             COFFRE MAGIQUE: Vous avez trouvé une meilleure arme de guérison
26
27             """
28
29             wizard.weapon = newWeapon
30         }
31
32         //S'il s'agit d'un personnage guerrier (warriorCharacter). On attribue une arme d'attaque
33         let warriorCharacter = attacker.selectCharacter(team: attacker.team)
34         wizard.heal(target: warriorCharacter)
35     } else {
36         if let newWeapon = chest.randomWeapon(type: .attack) {
37             print("""
38
39             COFFRE MAGIQUE: Vous avez trouvé une meilleure arme d'attaque
40
41             """
42
43             attackingCharacter.weapon = newWeapon
44         }
45
46         //Si ce n'est pas Wizard, on attaque
47         let targetCharacter = attacker.selectCharacter(team: defender.team)
48         attackingCharacter.attack(target: targetCharacter)
49     }
50 }
51
52
53 //func attack permet d'attaquer
54 private func fight() {
55
56     //Cette boucle (While) permet au joueur de choisir son attaque et son adversaire
57     while teamIsAlive(player: player1) && teamIsAlive(player: player2) {
58         print("""
59
60             JOUEUR 1 c'est à toi! Tu choisis ton combattant, puis un ennemi
61
62             """
63
64             playerRound(attacker: player1, defender: player2)
65             numberRound += 1
66
67             if teamIsAlive(player: player2) {
68                 print("""
69
70                 JOUEUR 2 c'est à toi! Tu choisis ton combattant, puis un ennemi
71
72                 """
73
74                 playerRound(attacker: player2, defender: player1)
75
76             }
77     }
78 }
79
80 }
81

```

```

1 //
2 // main.swift
3 // Jeu de combat
4 //
5 // Created by PARISATO on 03/05/2020.
6 // Copyright © 2020 PARISATO. All rights reserved.
7 //
8
9 import Foundation
10
11 print("===== W E L C O M E -- G A M E -- R P G =====")
12
13 //Création d'une instance de jeu de classe
14 var game = Game()
15 game.start()
16

```