Battle est un répertoire qui contient les fonctions liées aux fonctionnements des combats. Nous avons choisi de séparer les effets visuels (stockés dans Graphics) du système du jeu en lui même afin de créer des classes de combats qu’on lance via la méthode start et qui sont indépendantes. Nous avons préféré éviter d’avoir des codes trop long avec des fenêtres graphiques car le mécanisme interne des combats est complexe et cela aurait été une source d’erreur.

Nous avons créé les structures Player et Trainer. Player contient des Pokémon. Ses caractéristiques sont décrites dans le fichier Player du répertoire BackUp, et celles de ses Pokemon dans le fichier PlayerPokemon du même répertoire. Ainsi, si l'on souhaite directement tester des configurations précises ou équilibrer le jeu il n'y a qu'à changer les statistiques dans ces fichiers, les Pokemon sélectionnés ou leurs capacités. Une des fonctions associées est SwapPokemon, qui permet de changer le Pokemon retenu pour le combat.

battle.h contient des morceaux de code définis en macros et les headers de toutes les fonctions graphiques de combat. Les fonctions décidant des actions à réaliser sont elles situées dans BattleWildPokemon. Elles n'interagissent pas directement avec les fonctions graphiques de battle. Par exemple, la fonction swapMenu de battle renvoie un flag à la méthode swapMenu de BattleWildPokemon, qui décide de l'action à effectuer et interagit à nouveau avec la fonction de battle à l'aide de la méthode printSwapMenu.

Ces méthodes print sont de type Void, elles se contentent d'envoyer des paramètres aux fonctions de battle, qui elles-mêmes renvoient l'écran sur lequel elles ont agi. Les méthodes print en tant que tel ne renvoient rien.

A l'inverse, parmi les méthodes ayant un impact sur les actions effectuées, tout commence avec la méthode start, comme on peut le voir sur le graphique11. Le joueur peut ou bien décider de s'enfuir, ce qui provoque l'appel à la méthode runAway, ou bien décider de changer de Pokemon pour combattre à l'aide de la méthode swapMenu (le nouveau menu étant affiché par la fonction swapMenu de battle), ou encore décider de combattre. Il requiert alors l'apparition du menu de combat à l'aide de la méthode fightMenu, méthode qui va enregistrer le choix de l'utilisateur concernant l'attaque à appliquer. Le rendu graphique est déterminé par battle.h