This diagram represents how the classes inside the game.grounds package and the new game.actors.pokemon classes perceive time and as a result experience changes. The classes in the game.grounds package based on what they perceive from the TimePerception interface, (which determines the time of day using the other two game.time classes) convert themselves into their respective ground type using the FancyGroundFactory class, which creates the new Ground objects using the abstract Ground class. The Ground class is made abstract to avoid the repetition of code (DRY). The Tree class also has a chance of dropping a candy, which it does so using the DropItemAction class. The Pokemon which extend the Actor class also perceive time from the TimePerception interface, which would then apply the statuses of either restoring their hit points or taking damage. It was decided the classes that apply these statuses where not included in this diagram as these behaviours and actions are complex enough to require their own UML diagram.