University of Camerino

Logic and Constraint Programming course

Project Ludicolo

A *Pokémon* look-alike in Java & Drools

Stelluti Francesco Pio, 115544
Zamponi Marco, 115860

A.Y. 2020/2021

A brief introduction to *Pokémon* battles

- During a battle two trainers fight each other using a team of up to 6 Pokémon.
- Each Pokémon knows up to 4 ***moves*** and can throw one of them once in each turn to damage its opponent or defend itself based on its trainer decisions.
- Alternatively, during its turn, a trainer can either choose to ***swap*** its Pokémon with another one of its team or use one of its ***items***.
- The battle goes by **turns**.
  During each turn both trainers make their decisions.
  Such decisions are eventually executed in a specific priority order.
- The battle ends when the team of one of the trainers is completely fainted.

How Pokémon are defined

- The `Pokemon` class takes care of defining the internal state of a Pokémon.

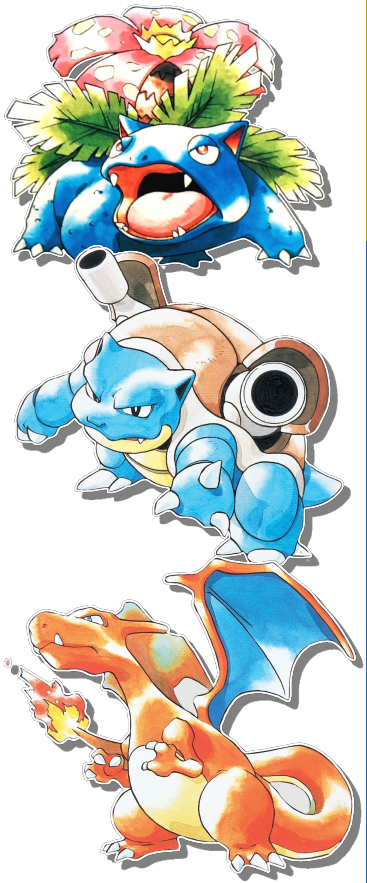
- Each Pokémon has a level value and a series of base stat values, that define how strong and proficient such Pokémon is in many aspects:
  - ***Life***.
  - ***Attack***, used during damage calculation for *Physical* moves.
  - ***Defense***, used during damage calculation for *Physical* moves.
  - ***Special attack***, used during damage calculation for *Special* moves.
  - ***Special defense***, used during damage calculation for *Special* moves.
  - ***Speed***, used during move order definition after they are chosen.

- Based on such base stat values, once a Pokémon enters in battle its full stat values are defined using the formulas:

  life battle stat = ((2 * life base stat * level)/100) + level + 10

  other battle stat = ((2 * other base stat * level)/100) + 10

- When a stat is used for a damage calculation in battle, a number of modifiers may be applied during the calculation.



How Pokémon are defined

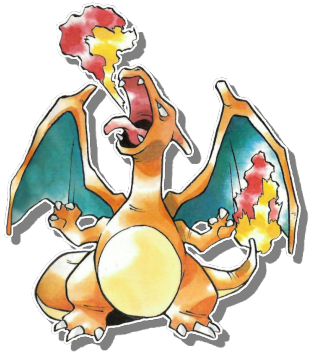
- Each Pokémon can be affected by the following statuses:
  - ***Asleep***, is prevented from making a move.
    It solves after a random number of turns (chosen from 1 to 3).
  - ***Burned***, takes damage equal to 1/16 of its maximum HP each turn at the end of each turn and the damage it deals with physical moves will be halved.
    It solves only with an item.
  - ***Confused***, has a 33% chance to damage themselves instead of the opponent. Confusion damage is calculated as if it were a typeless physical move with a power of 40.
    It solves after a random number of turns (chosen from 1 to 4).
  - ***Frozen***, is unable to make a move for an indeterminate number of turns.
    It solves only with an item.
  - ***Paralyzed***, runs a 25% risk of losing their turn due to full paralysis. In addition, the afflicted Pokémon's Speed is decreased.
    It solves only with an item.
  - ***Poisoned***, takes damage equal to 1/8 of its maximum HP each turn at the end of each turn.
    It solves only with an item.

- If a Pokémon throws the move ***Protect*** it will be completely protected from incoming attacks during the turn.
  Protected is a condition that can be present in a Pokémon along with any aforementioned status condition.

Pokémon and moves types

- Both Pokémon and their moves are characterized by elemental types that define weaknesses and strengths during battles.
- Each move has one type whereas each Pokémon has one to two types.
- The types are the following:

  - ***Normal***
  - ***Fire***
  - ***Fighting***
  - ***Water***
  - ***Flying***
  - ***Grass***
  - ***Poison***
  - ***Electric***
  - ***Ground***
  - ***Psychic***
  - ***Rock***
  - ***Ice***
  - ***Bug***
  - ***Dragon***
  - ***Ghost***
  - ***Dark***
  - ***Steel***
  - ***Fairy***



DEFENSE → ATTACK ↓	NOR	FIR	WAT	ELE	GRA	ICE	FIG	POI	GRO	FLY	PSY	BUG	ROC	GHO	DRA	DAR	STE	FAI
NORMAL													½	0			½	
FIRE		½	½		2	2						2	½		½		2	
WATER		2	½		½				2				2		½			
ELECTRIC			2	½	½				0	2					½			
GRASS		½	2		½			½	2	½		½	2		½		½	
ICE		½	½		2	½			2	2					2		½	
FIGHTING	2					2		½		½	½	½	2	0		2	2	½
POISON					2			½	½				½	½			0	2
GROUND		2		2	½			2		0		½	2				2	
FLYING				½	2		2					2	½				½	
PSYCHIC							2	2			½					0	½	
BUG		½			2		½	½		½	2			½		2	½	½
ROCK		2				2	½		½	2		2					½	
GHOST	0										2			2		½		
DRAGON															2		½	0
DARK							½				2			2		½		½
STEEL		½	½	½		2							2				½	2
FAIRY		½					2	½							2	2	½	

The full type chart here displays the strengths and weaknesses of each type.

How moves are defined

- Each Pokémon has a maximum of four **`Move`** instances to choose from during battle.

- Each move has defined the following parameters:
  - ***Type***.
  - ***PP***, an integer value that represents how many times the move can be used without the use of items.
  - ***Power***, an integer value that states “how strong” a move is.
  - ***Accuracy***, a percentage value that is used to determine whether a move is fired successfully or not.
  - ***Priority***, an integer value that is used when defining moves order before they are fired.

- A move can be of one of the three following categories:
  - ***Physical***, which moves deal damage depending on the Attack stat of the attacking Pokémon and the Defense stat of the defending Pokémon.
  - ***Special***, which moves deal damage depending on the Special Attack stat of the attacking Pokémon and the Special Defense stat of the defending Pokémon.
  - ***Status***, the only one category which moves do not inflict damage.

- Each move, including *Physical* and *Special* ones, can inflict the following side effects:
  - ***Stage modification***, a Pokémon's effective stats may be raised or lowered by certain moves.
  - ***Status infliction***, a Pokémon is assigned with a status if it has not already one.

How moves' damages are defined

- Each time a *Physical* or *Special* move is struck the associated damage value needs to be calculated.
- The damage value mainly depends on:
  - The level of the attacking Pokémon.
  - The power of the move.
  - The *(Special) Attack* value of the attacking Pokémon.
  - The *(Special) Defense* value of the defending Pokémon.
  - A series of modifiers:
    - ***Critical***, a probability value that multiplies the damage of a damage-dealing move by 1.5.
    - ***Random*****,** a random factor between 0.85 and 1.00 (inclusive).
    - ***Stab*****,** the same-type attack bonus. This is equal to 1.5 if the move's type matches any of the attacking Pokémon's types, 1 otherwise.
    - ***Type*****,** the type effectiveness. This can be 0 (ineffective); 0.25, 0.5 (not very effective); 1 (normally effective); 2, or 4 (super effective), depending on both the move's and target's types.
    - ***Burn*****,** is 0.5 if the attacker is burned.

damage value = (((((((2 * level)/5) + 2) * power) * (attack/defense)) / 50) + 2) * modifiers

How items are defined

- During each turn a player can decide not to throw a move and to utilize an item instead.

- Each `Item` instance that has been defined defined inflicts one or more of the following effects:
  - ***Healing effect***, restores a portion or a percentage of the trainer's Pokémon currently in battle.
  - ***Restore effect***, restores a portion or a percentage of the PPs associated with the moves known by the trainer's Pokémon currently in battle.
  - ***Status resolve effect***, removes a particular status effect from the trainer's Pokémon currently in battle.

The Drools implementation

- The battle is first defined within the classic Java environment. In order to define a battle, both trainers and the respective teams and backpacks full of items shall be instantiated.
- The Drools rule engine is fired once the battle and its participants have been defined.
- The entire battle starts and ends, returning a winner trainer, thanks to the Drools rule engine.

The flow behind the Drools rules

1. Battle is defined and set up
2. Players are set up and their first Pokémon are sent out
3. The first turn is set up
4. Each player chooses its action to be executed during the turn. An action could be a move, an item or a swap.
    a. Actions are ordered by priority
        i. Preliminary effects such as status effects are executed
        ii. First action and all of its side effects are executed
        iii. Checks for fainted Pokémon and for winning conditions are performed
        iv. Preliminary effects such as status effects are executed
        v. Second action and all of its side effects are executed
    b. Effects that must take place after each turn are executed
    c. Checks for fainted Pokémon and for winning conditions are performed
    d. The next new turn is set up

How actions are managed

- The abstract class **`Action`** takes care of the state of each action that is defined by players during the beginning of their shared turns.

- The states that can affect an **`Action`** instance are:
  - **`DEFINED`**, the action has just been defined and waits for its priority order to be set.
  - **`READY_TO_FIRE`**, the priority order of the action has been defined. The **`Action`** is ready to be executed.
  - **`FIRED`**, the main effect of the **`Action`** has been executed. Now side effects can be executed.
  - **`DONE`**, all of the **`Action`** effects of have been executed and the instance can be safely retracted. Now other **`Action`** instances can be set as **`READY_TO_FIRE`** if present.

- The classes that extend **`Action`** are:
  - **`ItemAction`**, associated with the use of an item and the execution of its effects.
  - **`MoveAction`**, associated with the execution of a Pokémon move.
  - **`SwapAction`**, associated with the execution of a Pokémon swap.

How priorities are defined

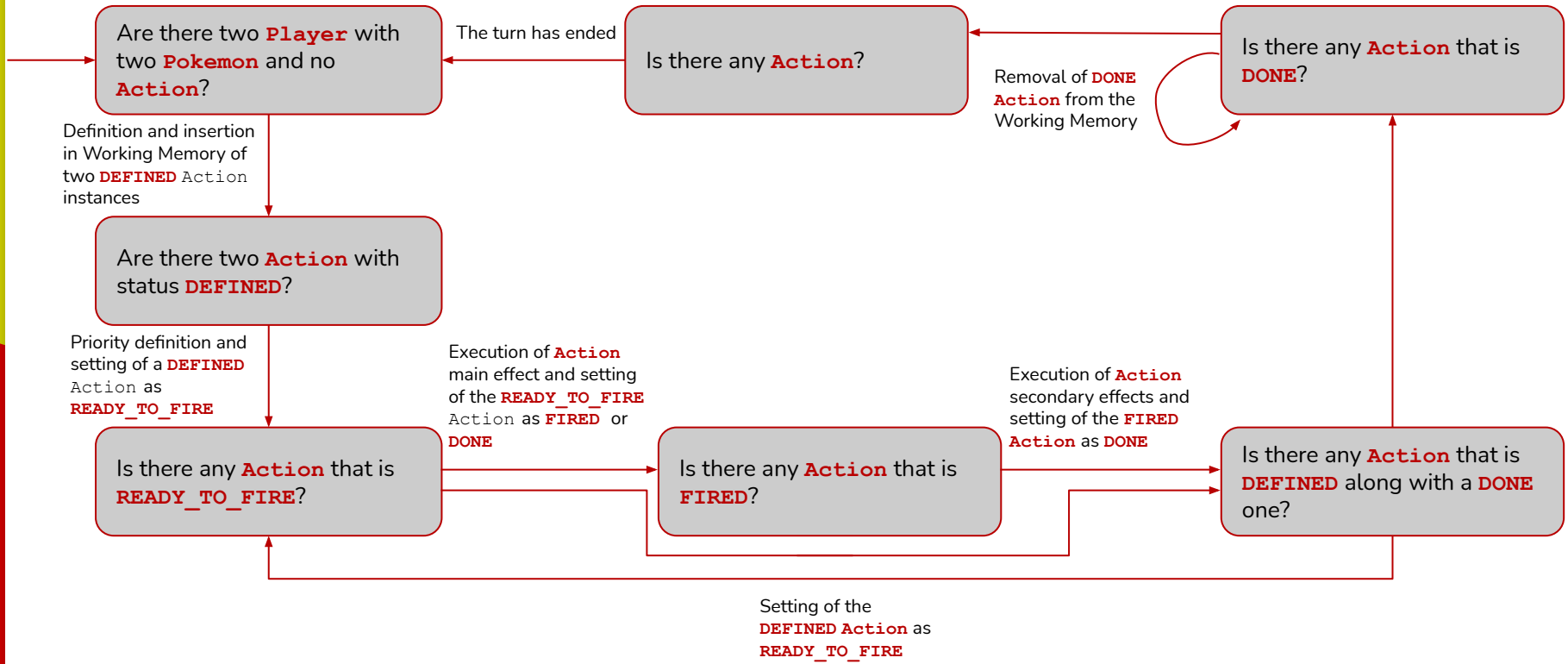
- Each **`Action`** instance is associated with a priority value which is an integer value.
  Such value is important as it is used to define which **`Action`** instance is fired first. The higher the value, the faster is the **`Action`** fired.

- The defined priorities, in a decreasing order, are the following:
  - **`SwapAction`**, associated with the execution of a Pokémon swap.
  - **`ItemAction`**, associated with the use of an item and the execution of its effects.
  - **`MoveAction`**, associated with the execution of a Pokémon move.

- Different **`MoveAction`** instances are associated with different priority values and depend on the priority value defined in the related **`Move`** instance.
  Such values range from -7 to +5 and depend on the specific move.
  - For instance, ***Protect*** has a +4 priority value and is fired before most of all the other moves in order to give the Pokémon that fired it a protection from any other attack during the turn.

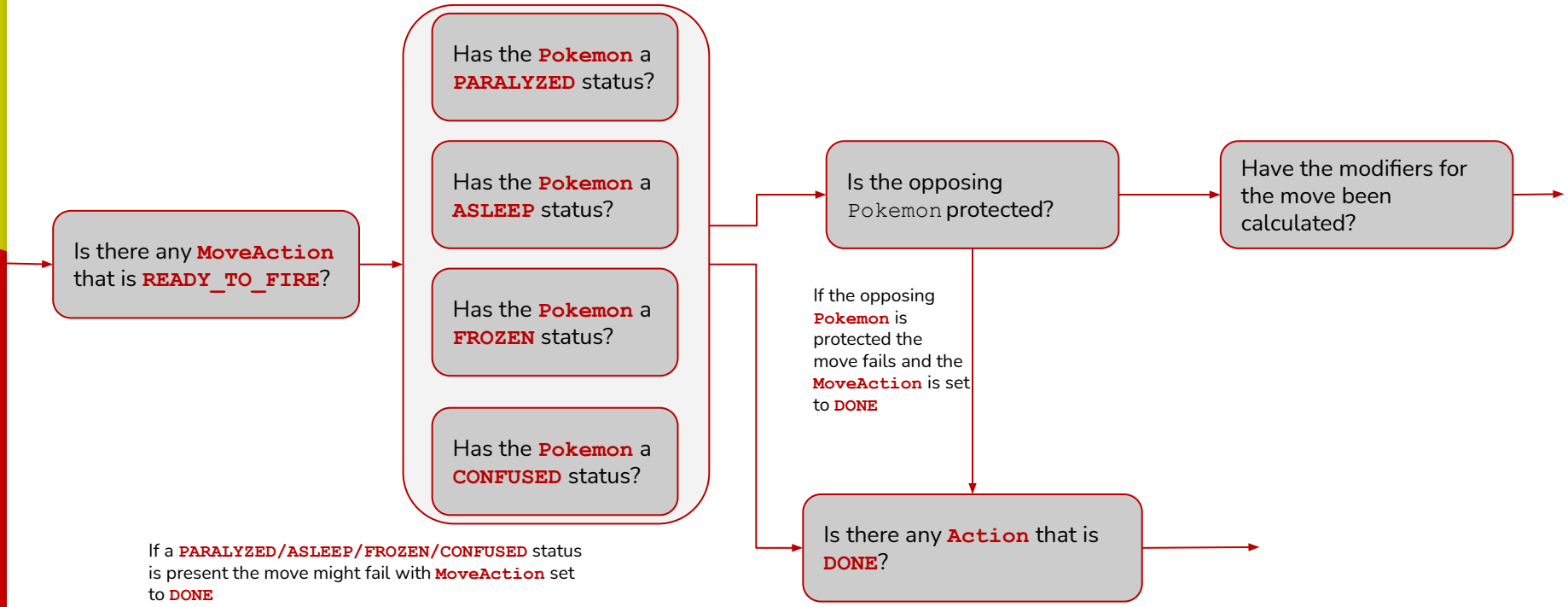
- If two **`Action`** instances have the same integer priority value then the *Speed* values of the associated trainer's **`Pokemon`** are compared in order to determine the **`Action`** instance that is fired first.
  - After all, the faster Pokémon must act first in absence of any form of priority.

How moves are ordered and fired in Drools



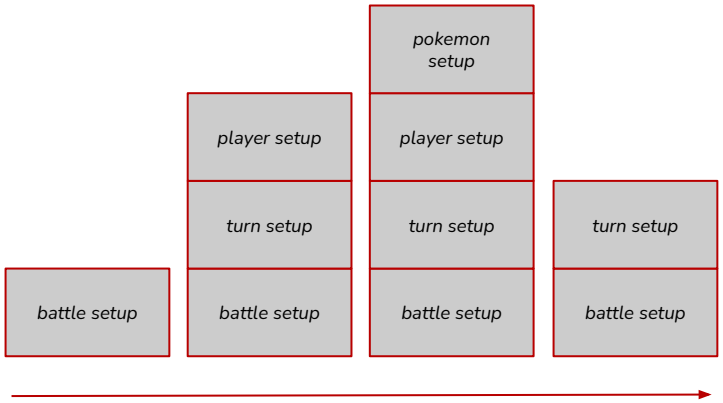
How status/condition effects are fired in Drools

- Several status conditions and the presence of a protection are evaluated before a **`MoveAction`** has its first effect fired.



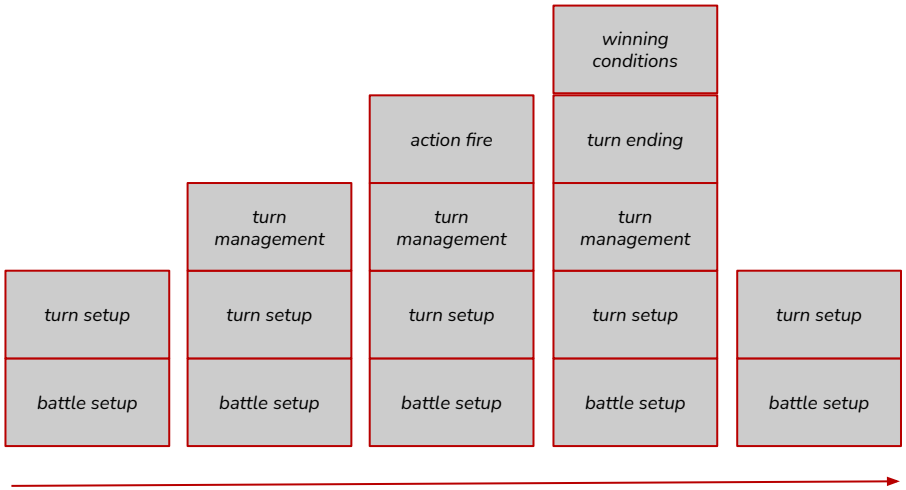
Use of Drools agenda groups

- For a better rules management we have defined a series of agenda groups within Drools.
- The agenda groups let the rules to be evaluated as a group. Such group is placed within a **stack** with other groups in order to be evaluated.
- The agenda groups that have been defined for this project represent phases of the game and are the following



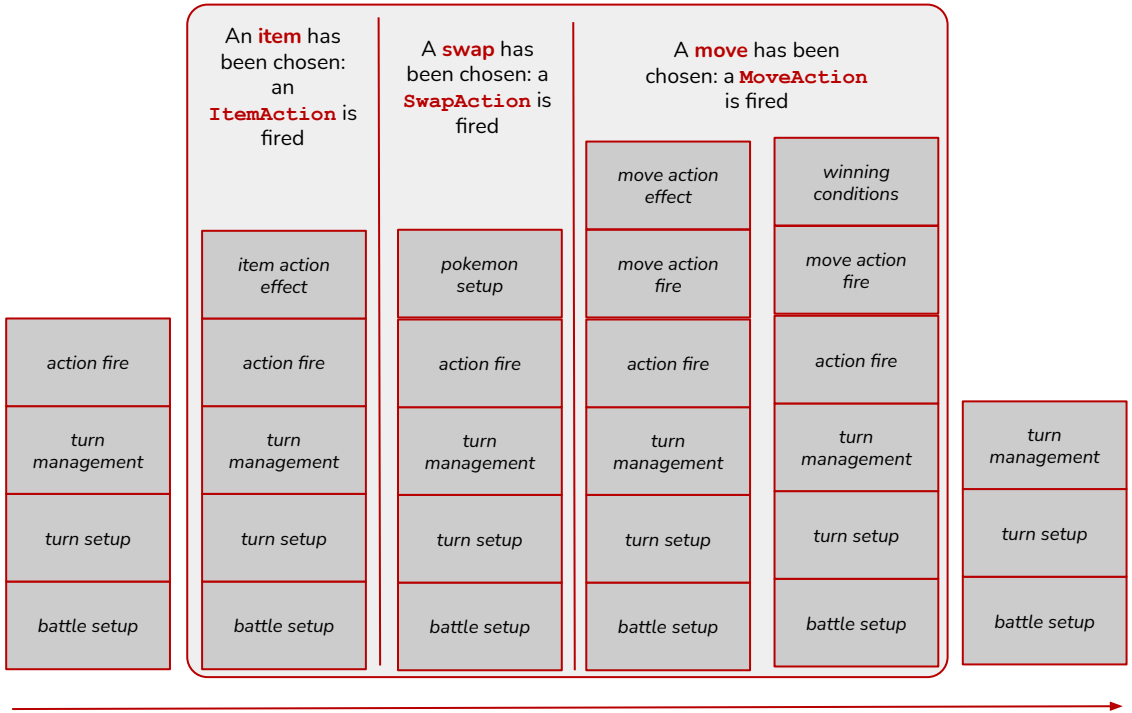
- ***battle setup***, the first agenda group called during the execution. Serves the purpose of setting up a `Battle` instance that is present within the working memory.
- ***player setup***, serves the purpose of setting up each `Player` instance that is present in the working memory, i.e., setting up the life values of each of their `Pokemon` instances and inserting in the working memory their first available Pokémon as the one that is ready to fight.
- ***pokemon setup***, completes the setup of a `Pokemon` instance defining all of its `Stat` values.

Use of Drools agenda groups (turn management)



- ***turn setup***, during which each trainer chooses the **`Action`** to take during their turn.
- ***turn management***, sets up the execution of the **`Action`** instances present in the working memory, ensuring that each is executed following a specific order.
- ***turn ending***, takes care of the final phase of a turn after each trainer's **`Action`** has been successfully executed. ***Burned*** and ***Poisoned*** statuses effects are evaluated as in this phase. ***Protect*** effects are reset.
- ***action fire***, provides execution of an **`Action`** present in the working memory. Many Pokémon status related effects are evaluated as well. ***Confused*** status effects are evaluated in this phase.
- ***winning conditions***, serves the purpose of checking whether a Pokémon is fainted or not after an rule fire that might have inflicted damages. It also checks whether a **`Player`** team is completely fainted or not.

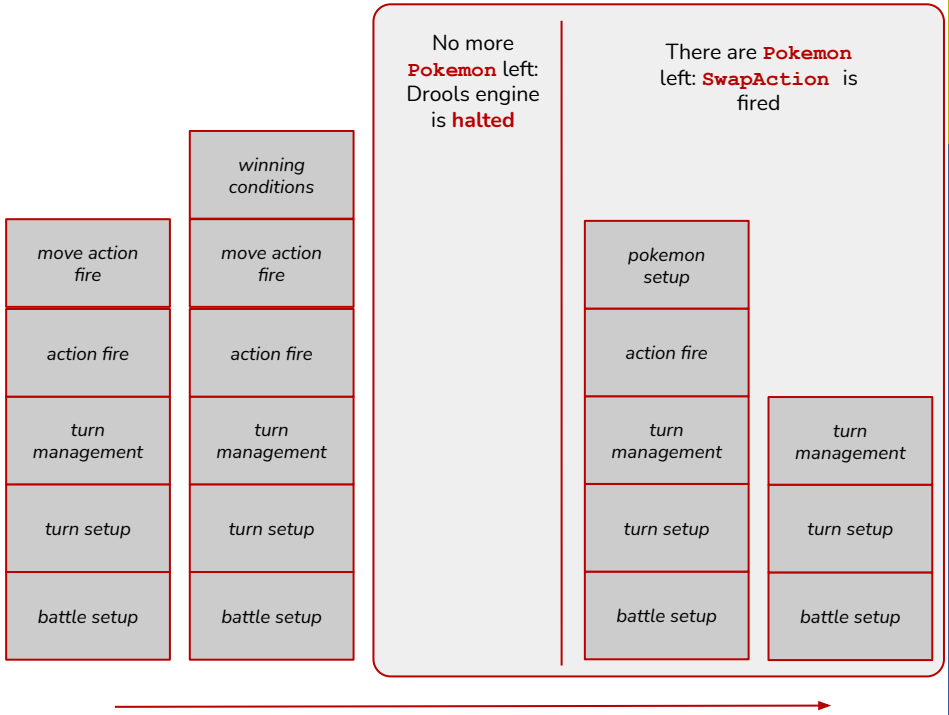
Use of Drools agenda groups (action firing)



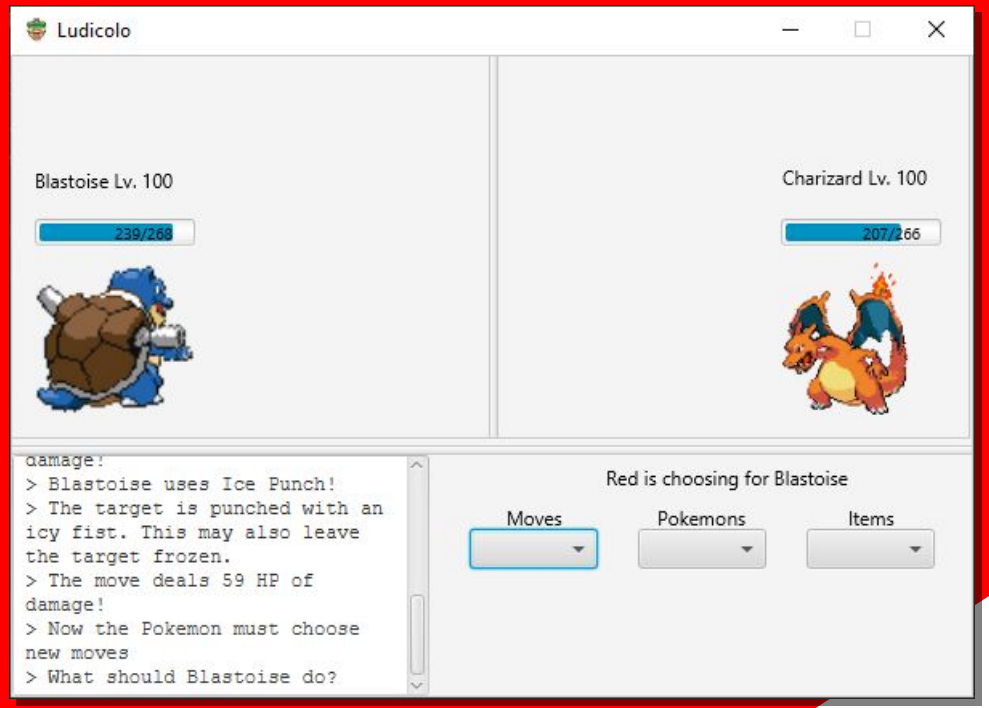
- ***item action effect***, provides execution of all the effects of a fired `ItemAction` present in the working memory.
- ***move action fire***, provides execution of all the effects of a `MoveAction` present in the working memory that is ready to be fired. ***Confused***, ***Frozen***, ***Asleep***, ***Paralyzed*** statuses effects are evaluated in this phase. ***Protect*** effects are evaluated as well.
- ***move action effect***, provides execution of all the effects of a fired `MoveAction` present in the working memory.

When does the game stop?

- The agenda group ***winning conditions*** is called each time a Pokémon might have lost a portion of its health.
  - If there's any **`Pokemon`** that has lost the entirety of its health such **`Pokemon`** such entity is removed from the Working Memory and the game is therefore updated.
- After a **`Pokemon`** is retracted the following scenario is evaluated:
  - The **`Pokemon`** instances of the respective trainer all fainted?
    - In such case the Drools engine is halted, the winner is declared and the game stops.
- If there are still available (not fainted) **`Pokemon`** instances then game carries on. Another **`Pokemon`** instance must be added to the Working memory and therefore sent out in battle.
  - To do so an instance of **`SwapAction`** is inserted in the working memory and the agenda groups ***action fire*** and ***pokemon setup*** are eventually evaluated.



Time for a short demo



Thanks for the attention