

PokemonGoBack

stuart.thiel

May 2017

1 Introduction

This project is to create a revolutionary new pokemon app called "Pokemon Go Back". You and your team will implement such advanced features as "a gui" and "one-player mode". Instead of using outdated app technology, you will develop for a stand-alone PC, the wave of the future.

2 Deliverables

I am not picky on format, I only really care about content. Less is more. Be brief and focus on what is important.

I want you to communicate to me about the following for your project, generally considered "The Documents":

- Requirements
- Use Cases
- Design
- Architecture
- GUI Decisions
- Testing Plans
- System Behaviour Analysis
- Scoping/Cost Considerations
- Installation and Commissioning Instructions

You must submit your Test Suite and Source Code as the implementation deliverables. The I & C instructions should be sufficient to help me get it running.

2.1 Evaluation

I cannot directly evaluate your process. You are responsible, as a team, with coming up with a process. This process will be reflected in your Documents and Implementation. Document it as needed.

2.1.1 Documents

In all documents you should answer "What", "Why", and "Why this is important". I want communication about all these areas:

- Requirements
- Use Cases
- Design
- Architecture
- GUI Decisions
- Testing Plans
- System Behaviour Analysis
- Scoping/Cost Considerations
- Installation and Commissioning Instructions

In whatever form you submit deliverables, I would like to know each member's role in each of the deliverables. Do not tell me that Person A wrote Section B, I don't care. I want to know their general role, like "reviewer", "documenter", "developer consulted regard...". How many roles a person has is not part of the evaluation, how well peoples' roles has been documented is.

The most important aspect of the document deliverables (besides relevant content) is that there be excellent traceability between all deliverables (documents, source code, test code, all of it).

You can and should ask questions about what this entails. If you negotiate/clarify something, include that in the documents!

2.1.2 Implementation

Your test code and source code will be reviewed. Make it nice and make it easy for me to navigate, or you will suffer. Coding conventions are not a bad idea.

Generally implementation grading will effectively be a general set of acceptance tests to verify that the underlying game mechanics are working. A mostly exhaustive list follows:

- Game Starts
- Decks Parse
- Cards Dealt
- Mulligans
- Play Initial Pokemon
- Bench Initial Pokemon
- Draw Card at Turn Start
- Play Items¹
- Bench Pokemon
- Evolve Pokemon
- Place Energy
- End Turn
- Use Pokemon
- AI Plays
- Retreat Pokemon
- Knock Out Pokemon
- Collect Prize Card

¹if any are implemented

- Win
- Lose
- Check Both Hand Sizes
- Check Both Deck Sizes
- Check Both Discard Sizes
- Look At Discards
- Check Energy on Active Pokemon
- Check Energy on Benched Pokemon
- Look at Pokemon Ability
- Look at Ability of Card in Hand
- See Current Pokemon Health
- See Max Pokemon Health

Testing will be evaluated on coverage and consistency with your Test Plan.

2.2 Grading

For each submission some grades will be for the documentation (most of them) some will be for the implementation.

For each card that is not implemented/documented, you lose a flat 1%. You can regain this 1% by implementing/documenting in a later submission. If there are abilities on a card that you don't like, replace it with a basic energy and take the 1% hit!

The grades will be:

Documentation 7/8

Implementation 5/5

Implementation grades will be based on how well the game works, can be played (how much of the necessary basic game mechanics are implemented). Implementation grades will also consider the written tests.

Documentation grades measure the quality of the documents as specified in the next section, but focus on traceability.

For example, if your first submission is 5/7 for documentation and 3/5 for implementation, but you chose to skip 20 cards to make life easier, your combined score would be 8/12 with -1.6 for missing cards.

If in your final submission, you scored 7/8 for documentation and 5/5 for implementation, and were down to missing only 10 cards, you would score 12/13 with -1.2 for missing cards. Overall, you would score 20 -3.2 for missing card penalties, and +0.8 for having recovered points in the second submission by implementing more cards. Final grade: 17.6 on 25.

2.3 Due Dates

First Submission: May 24th

Final Submission: June 19th

3 Project Starting Point

We will provide you with a text file (card.txt) detailing available cards and their abilities. Two decks will be provided, by way of a text file (deck1.txt and deck2.txt). Each line in this file is an index onto the card.txt file. An abilities file (ability.txt) will be provided that details all the abilities used on all cards, where not already covered by existing game mechanics.

3.1 Card Validation

Each card follows a specific format to figure out how it can be played. Each card has a name, type, possibly a category and possibly one or more abilities.

Every card has exactly one of the following types:

1. pokemon
2. energy
3. trainer

For each type, there are categories available

- | | |
|-------------------|-------------------|
| 1. type:pokemon | (d) cat:psychic |
| (a) cat:basic | (e) cat:fighting |
| (b) cat:stage-one | |
| 2. type:energy | 3. type:trainer |
| (a) cat:colorless | (a) cat:Stadium |
| (b) cat:water | (b) cat:Supporter |
| (c) cat:lightning | (c) cat:Item |

Pokemon may be affected by the status effects, which affect how they may be used during a player's turn. Status effects are always removed when retreating or evolving. Each status effect may only be applied once per pokemon (you cannot be poisoned twice, taking two damage per turn). Only these effects are considered:

1. status:paralyzed: a paralyzed pokemon may neither attack nor retreat. At the end of a turn, a player removes status:paralyzed from their active pokemon.
2. status:asleep: a sleeping pokemon may neither attack nor retreat. At the end of a turn, a player has a 50% chance that sleep will be removed from their pokemon.
3. status:stuck: a stuck pokemon may not retreat. At the end of a turn, a player removes status:stuck from their active pokemon.

4. status:poisoned: a poisoned pokemon takes one damage at the end of their turn.

Abilities on cards act when appropriate. A Stadium card would create a permanent effect, Supporter and Item cards apply an effect when played. Pokemon abilities apply when they attack. Abilities target various elements in the game, they are listed below for clarification.

- | | |
|----------------------------|----------------------------------|
| 1. targeting pokemon | (e) target:choice:opponent-bench |
| (a) target:your-active | (f) target:choice:your-bench |
| (b) target:opponent-active | 2. targeting players |
| (c) target:choice:opponent | (a) target:you |
| (d) target:choice:your | (b) target:them |

The abilities themselves are:

1. null
 - do nothing
2. dam
 - dam:[target]:[amount]
 - damages a pokemon
3. heal
 - heal:[target]:[amount]
 - heals a pokemon
4. deenergize
 - deenergize:[target]:[amount]
 - owner of target removes [amount] energy from that pokemon
5. reenergize
 - reenergize:[target:source]:[target:destination]:[amount]
 - move energy from one pokemon to another, [amount] times
6. redamage
 - redamage:[target:source]:[target:destination]:[amount]
 - move damage from one pokemon to another, [amount] times, allowing multiple destinations at once.

7. swap

- swap:[target:source]:[target:destination]
- swap one pokemon from source with one pokemon from target. swapping an active pokemon to the bench removes all status effects.

8. destat

- destat:[target]
- removes all status effects from the target pokemon

9. applystat

- applystat:[status]:[target]
- the targeted pokemon gets the specified status

10. draw

- draw:[amount]
- draw [amount] cards

11. search

- search[target]:[source]:[filter]:[amount]
- source may be either source:deck or source:discard
- filter is a combination of type and category, e.g. type:energy—cat:water would allow searching for any water energy
- using this ability, the player searches their deck or discard pile and adds the specified [amount] of allowed cards to their hand. ²

12. deck

- deck:[target]:[destination]:[choice]:[amount]
- destination may be either destination:deck or destination:discard, qualified by either top or bottom
- choice specifies whether you choose, whether the target chooses or whether the choice is random
- using this ability, the player target can place cards from their hand into either the deck or discard, as specified.

13. shuffle

- shuffle:[target]
- using this ability, shuffles the target deck.

²cards accepted by the filter are shown in the interface, out of order. A player may choose those cards, and they will be removed from where they were in the deck/discard, but there is no reshuffling

14. cond

- cond:[condition]:[ability](—[ability])*(:else:[ability](—[ability]))*
- If [condition] is met, apply the ability listed, otherwise apply the second ability (or do nothing if there is none listed)
- conditions may be any of the following
 - (a) healed:[target] (applies if target has been healed)
 - (b) flip (applies 50% of the time)
 - (c) ability:[ability] (applies if the player chooses to use that ability)
 - (d) choice (the player playing the ability chooses whether the condition passes)

15. add

- add:[target]:[ability](—[ability])*
- append this ability to all other abilities on the targeted pokemon

Any listed [amount] may be replaced by a count[target](—[filter])* , e.g. count[target:your-bench] would use the amount of pokemon on your bench instead of an amount. Further, amount should support basic arithmetic with +/—/* and the use of parentheses. A more complex example would be count(target:your-active—damage) which would count the damage on your active pokemon.

3.2 Deck Building

Loading the game will first check that deck-building rules are in effect.

1. each deck must have exactly 60 cards
2. you may have any number of cards that are of type:energy
3. you may have at most 4 of any card that is not of type:energy
4. each deck must have at least one type:pokemon card
5. all cards must be "playable"
 - (a) a type:pokemon,cat:stage-one is "playable" if the deck contains its corresponding type:pokemon,cat:basic
 - (b) a type:pokemon is "playable" if your deck contains sufficient energy to use at least one of its abilities

Note that decks are not shuffled at the start of the game. While certain abilities may be used to shuffle the deck, The initial decks should be played in the order of the files provided. Use this to your advantage during testing! I certainly will.

I reserve the right to include minor variations of cards during my testing, so do not hardcode. You **must** parse cards yourself!

Resources <http://www.pokemon.com/us/pokemon-tcg/play-online/>

You can actually play, for free, exactly the system I want you to make. It doesn't have to be this pretty (or even fun), but this will teach you about pokemon. Use this to learn the rules of pokemon.

3.3 The cards

In order to be able to interact with

28 unique cards in deck 1 24 unique cards in deck 2

You can find the import files for Pokemon TCG online called: deck1.ptcgo.txt
deck2.ptcgo.txt

You won't be able to play them, as you don't have all the cards, but you'll be able to see them as if you had the cards.