|  |
| --- |
| ML inc. |
| Independent Software Project: Pokédex |
| ISP External Documentation |

|  |
| --- |
| Michael Lee  Ms. Barsan  6/16/2014 |

Contents

[Project Proposal 3](#_Toc390616953)

[Description 4](#_Toc390616954)

[Purpose/Usage 4](#_Toc390616955)

[Audience 4](#_Toc390616956)

[Programming Language 4](#_Toc390616957)

[User Interface 4](#_Toc390616958)

[Implementation 5](#_Toc390616959)

[Modular Design 5](#_Toc390616960)

[Class Descriptions 5](#_Toc390616961)

[Testing Data 8](#_Toc390616962)

[Known Errors 8](#_Toc390616963)

[Conclusion 8](#_Toc390616964)

[References 9](#_Toc390616965)

# Project Proposal

# Description

## Purpose/Usage

This pokédex is designed to help Pokémon players by giving them a quick reference resource to information such as a Pokémon type, weaknesses, base stats and more. It also provides the in-game pokédex entries from each version of the game, allowing for the user to read a brief description of a Pokémon.

## Audience

This pokédex is designed for use by Pokémon enthusiasts who would like a quick reference when playing the games, playing competitively online, or are curious about a certain Pokémon’s abilities, description, etc.

## Programming Language

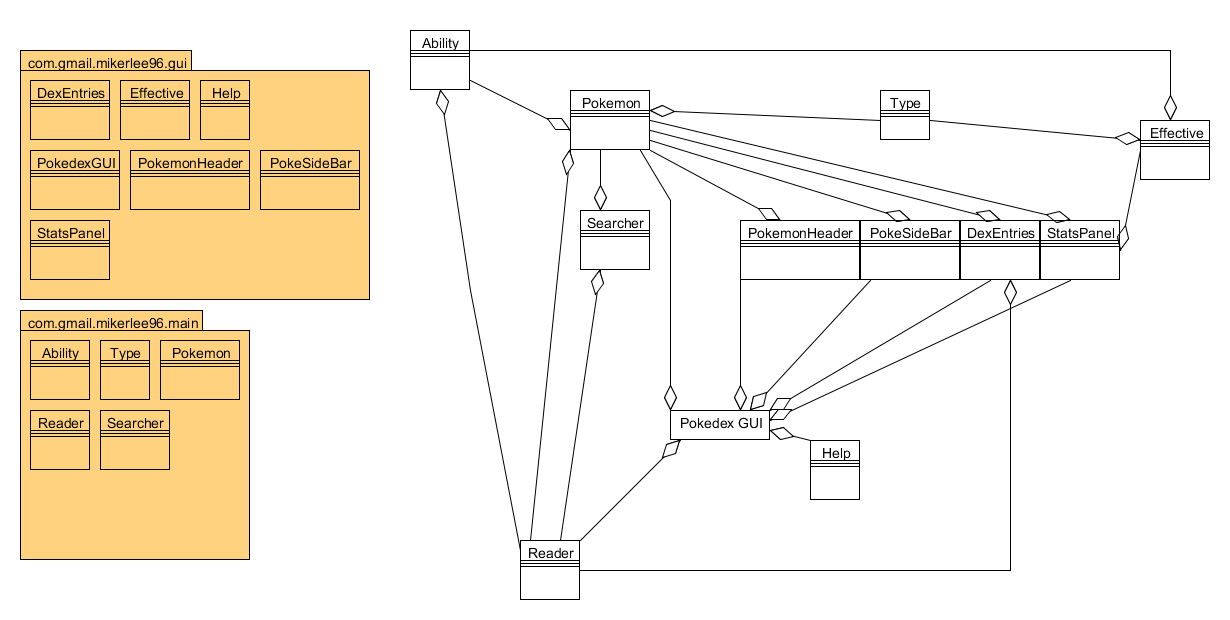
This program was written in Java, allowing for cross-platform use. The program also makes use of Java’s system look and feel option, providing a unique GUI appearance to fix the Operating system the user is running the program on.

# User Interface

The user interface in this program is intuitive and easy to understand. The left hand section of the pokédex is a list of the Pokémon implemented in the pokédex, which the user can scroll through and select a Pokémon by clicking its button. The top section of the program provides a search menu for the user, allowing them to search for a Pokémon by name, number, species, and type using a drop down menu. When a user searches, the list of Pokémon will be replaced by a narrowed list that fits the user’s search query. To return to the default list, the user can search with a blank input, or click the reset button. The majority of the program is taken up by the selected Pokémon (bulbasaur by default). This area shows the Pokémon’s number, name, and species to the top; picture, followed by its type, height, weight, and ability/abilities to the left, and to the right, the user can switch between pokédex entries and stats by selecting the appropriate tab.

# Implementation

## Modular Design



The picture above shows the different classes used in the pokédex. The GUI package contains all GUI components, including any JFrames and JPanels, and what isn’t part of the GUI is in the main package. Other files used include the *Pokémon/* folders, which contain separate files and folder for each Pokémon, the *Pokémon/list.txt* which contains the list of Pokémon, numbered *.png* files which contain the program images, the *Resources/* folders which contain different image sizes and types for each Pokémon, the *Types/* which contain information for Pokémon types, and the *Help/* folder, which contains images for the help window.

## Class Descriptions

* **DexEntries**
  + JPanel class used to show the in-game pokédex entries
  + **organize**: groups identical entries together
  + **isSame**: Checks if two (or more) pokédex entries are the same
  + **createLabels**: Creates the version headings for the pokédex entries
  + **disposeDuplicateLabels**:

Works with createLabels to group headings together

* + **createTextArea**:

Creates a JTextArea for the pokédex entry

* + **addSeparator**: Adds a separator between the pokédex entries based on generations

The next section are methods from the implementation of scrollable (javax.swing)

* + **getPreferredScrollableViewportSize** Returns the preferred size of the viewport for a view component
  + **getScrollableUnitIncrement** Components that display logical rows or columns should compute the scroll increment that will completely expose one new row or column, depending on the value of orientation
  + **getScrollableBlockIncrement** Components that display logical rows or columns should compute the scroll increment that will completely expose one block of rows or columns, depending on the value of orientation
  + **getScrollableTracksViewportWidth** If the width should resize rather than adding a scroll bar
  + **getScrollableTracksViewportHeight** If the height should resize rather than adding a scroll bar
* **Effective**
  + JPanel class to show a Pokémon’s weaknesses, resistances, and immunities
* **Help**
  + JFrame for the help menu
  + **createAndShow** Create and shows the window
* **PokedexGUI**
  + JFrame class used for the GUI
  + **makeHeader**: Deals with the look of the search area
  + **setIcons**: Import and sets the program’s icon
  + **makeButton**: Makes the button for a given Pokémon
  + **getPokemon**: Imports the images of the Pokémon and then calls makeButton
  + **search**: Handles searches using the Search class
  + **actionPerformed**: for when a user interacts with a component. Executes a command if the user clicks a Pokémon button, searches or resets search, or changes the search category
  + **makeDexScroll**: creates and sets the scroll increment the JScrollPanes for the pokédex entries and stats panels
  + **makeInfoTab**: creates and sets initializes the central information panel
  + **main:** runs the program
* **PokemonHeader**
  + JPanel class used to display the Pokémon’s number, name, and species
  + **paintComponent:** Used to change the background colour to a gradient
  + **fixName:** deals with Unicode characters
* **PokeSideBar**
  + JPanel class for the Pokémon’s picture, type(s), height, weight, and ability(ies)
  + **createLabels:** creates the labels and sets text appearance
  + **bottomSpacer**: creates a filler to add to the bottom of the panel
* **StatsPanel**
  + JPanel for showing the Pokémon’s base stats. Also displays Effective as part of the panel
  + **setStatColour:** sets the foreground and background for the base stat
  + **createLayout:** sets the layout of the elements
* **Ability**
  + Sets formatting for displaying a Pokémon’s ability
  + **ability**: returns the formatted ability
* **Pokemon**
  + Contains the information and behavior for a Pokémon
  + **displayName:** returns formatted name
  + **init:** initializes all of the Pokémon’s fields
  + **info:** returns specified information about the Pokémon
  + **hasElement:** returns if a Pokémon has a specified element
  + **weaknesses:** returns a Pokémon’s weaknesses
  + **resistance:** returns a Pokémon’s resistances
  + **immune:** returns a Pokémon’s immunities
  + **getTypeColour:**

returns a colour based on the Pokémon’s primary type

* + **getType2Colour:**

returns a colour based on the Pokémon’s primary type

* + **replaceSymbol:**

formats a Pokémon’s name when the name contains some oddity

* + **gettStats:** returns a Pokémon’s base stats
* **Reader**
  + Gets information from the file
  + **typeInfo:** gets information for each type
  + **pokeInfo:** gets the Pokémon’s basic info
  + **getPokedex:** gets the pokédex entries
  + **theStats:** gets the Pokémon’s base stats
  + **getPokemon:** gets the list of Pokémon
* **Searcher**
  + Searches Pokémon based on user input
  + **Results:** returns search results
  + **getPokemon:** get’s the list of Pokémon
* **Type**
  + Class for the different Pokémon types
  + **whatType**: returns a type’s name based on integer input
  + **pkmnType**: return’s the type specified in the constructor
  + **setColour**: sets the different colour for each type
  + **getTypeColour:**

returns the colour of the Pokémon’s type

* + **colourOf:** returns a specified type’s colour

# Testing Data

The following examples are search queries entered in the search area to the top. The results are buttons showed in the left hand scroll-list.

* Case 1:
  + Search input: saur
  + Search category: name
  + Results:
    - Bulbasaur
    - Ivysaur
    - Venusaur
* Case 2:
  + Search input: saur
  + Search category: number
  + Results:
    - No Results
* Case 3:
  + Search input: dragon
  + Search category: species
  + Results:
    - Horsea
    - Seadra
    - Dratini
    - Dragonair
    - Dragonite

# Known Errors

* Error: Pokédex entries scroll bar is near the bottom when a new Pokémon is selected
* Reason: The way that the JScrollPane is formatted and the way the program changes from Pokémon to Pokémon causes scrollable pane to render the panel incorrectly, causing it to start near the bottom rather than at the top
* Error: Some of the pokemon’s resistances/ weaknesses are cut off.
* Reason: The window size is too small. Simply resize you window and the weaknesses/ resistances will display normally

# Conclusion

Overall, the program is fairly close to the proposal. Things in the proposal that weren’t added are advanced Pokémon properties such as IV yield and over 500 Pokémon that were not added. The program is close to what I had in mind when I wrote it, featuring the basic information for each Pokémon, and would work well as a utility for a Pokémon player who is up to date in the meta-game.

# References

[www.stackoverflow.com](http://www.stackoverflow.com)

[www.bulbapedia.net](http://www.bulbapedia.net)

[www.serebii.net](http://www.serebii.net)

<http://ninjacave.com/jarsplice>

[www.pkparaiso.com](http://www.pkparaiso.com)

<http://docs.oracle.com/>

<http://chatt.hdsb.ca/~barsanl/>

<http://www.java2s.com/>

<http://www.tutorialspoint.com/java/>

<http://wikipedia.org>

<http://www.tutorialspoint.com/>

<http://www.umlet.com/>