# **Multifaceted Dictionary**

Adam Wiemerslage Hunter Wapman Alex Killian

### **Overview**

A dictionary application that allows users to track words that they are interested in.

## Implemented Features

- User can add an Entry.
- User can add a Word Sense to an Entry.
- User can add a Definition to a Word Sense.
- User can add a Word Form to a Word Sense.
- User can add a Part of Speech to a Word Sense.
- User can add entries in batch from a file.
- User can update an Entry word spelling.
- User can update a Word Form spelling.

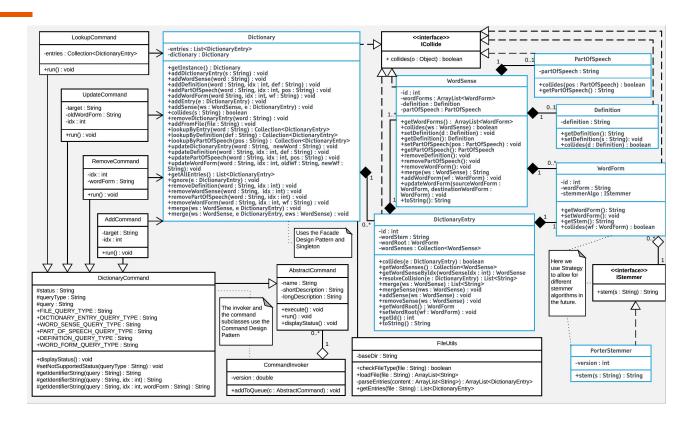
- User can update a Definition.
- User can update a Part of Speech.
- User can lookup Entries by word.
- User can lookup Entries by Definition.
- User can lookup Entries by Part of Speech.
- User can remove an Entry.
- User can remove a Word Sense.
- User can remove a Part of Speech.
- User can remove a Definition.
- User can remove a Word Form.

## Demo

#### Demo ran from our demo driver:

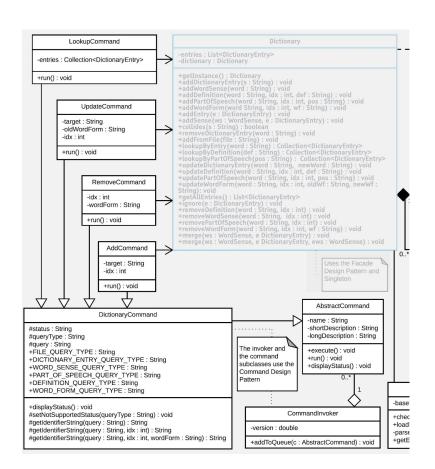
https://github.com/hunter-/dictionary/blob/master/37\_MultifacetedDictionary\_Video.mov

## **Overview of Design**



## **Command Design Pattern**

 Separates view and data access



#### Facade Controller Pattern

- The dictionary class serves as a simplified, unified, interface for the rest of the components of the system.
- This also adds a layer of abstraction over each method of each subclass that is combined in order to use a single feature.

```
Dictionary
  -entries: List<DictionaryEntry>
  -dictionary : Dictionary
  +getInstance(): Dictionary
+addDictionaryEntry(s: String): void
+addVordSense(word: String): void
+addVordSense(word: String): void
+addPartOfSpeech(word: String, idx: int, def: String): void
+addPartOfSpeech(word: String, idx: int, wf: String): void
+addWordForm(word String, idx: int, wf: String): void
+addSense(ws: WordSense, e: DictionaryEntry): void
+addSense(ws: WordSense, e: DictionaryEntry): void
  +collides(s: String): boolean
+removeDictionaryEntry(word: String): void
+removeDictionaryEntry(word:String): void
+addFromFile(file:String): void
+lookupByEntry(word:String): Collection<DictionaryEntry>
+lookupByPartOfSpeech(pos:String): Collection<DictionaryEntry>
+updateDictionaryEntry(word:String): void
+updateDefinition(word:String, idx:int, def:String): void
+updateDefinition(word:String, idx:int, def:String): void
 +updatePartOfSpeech(word: String, idx: int, pos: String): void
+updateWordForm(word: String, idx: int, oldWf: String, newWf:
  String): void
 +getAllEntries() : List<DictionaryEntry>
+ignore(e : DictionaryEntry) : void
  +removeDefinition(word: String, idx: int): void
  +removeWordSense(word: String, idx:int): void
 +removePartOfSpeech(word: String, idx: int): void
+removeWordForm(word: String, idx: int, wf: String): void
  +merge(ws: WordSense, e DictionaryEntry): void
   +merge(ws: WordSense, e DictionaryEntry, ews: WordSense): void
                                                                                                                                                         0..4
```

Uses the Facade
Design Pattern and
Singleton

## Thank You!