D-Team 33 Milestone 1: Design Document

Shawn Ge, Danica Fliss, Alex Fusco, Cole Thomson, Leah Witt
Course: CS 400 - Prof: Deb Deppeler
Due: 11/16/2018

Class Summary:

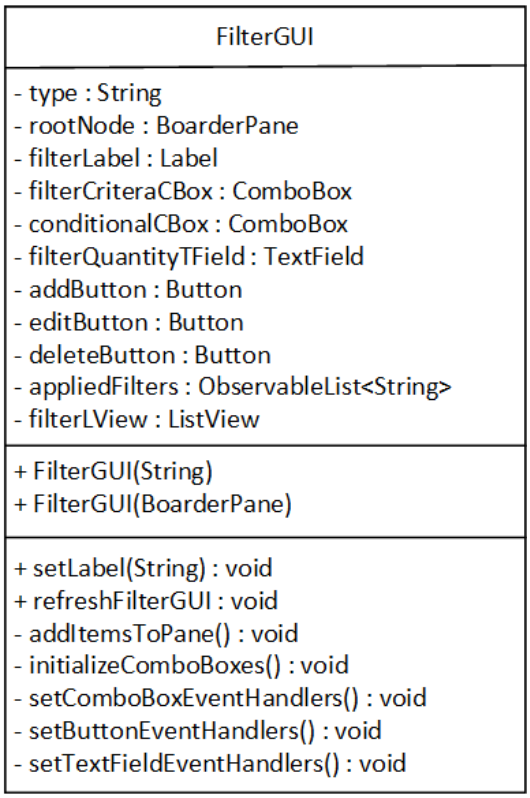
- Overall summary of classes needed for Food Planner Program.

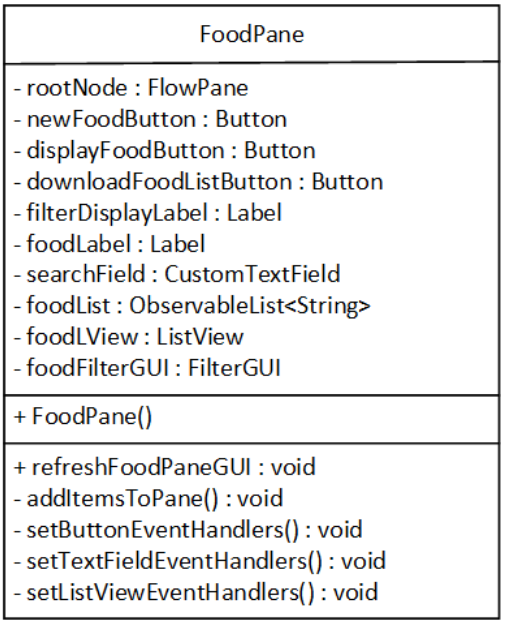
Class, Enum, or Interface	Type Name	Description of Use or Purpose
Class	FilterGUI	Organizing GUI objects for filtering Foods and Meals
Class	FoodFile	Handles the importing of food information from a userFileName.txt into the into the initial food list and the exporting of food information from the current food list back to userFileName.txt
Class	DisplayPane	Shows certain information based on user input at center of GUI
Class	FoodPane	Displays food specific options and a list of current foods. (Left-hand pane in GUI)
Class	MealPane	Displays meal specific options and a list of current meals (Right-hand pane in GUI)
Class	Meal	Handles data related to a meal and methods for storing and manipulating meals.
Class	PopUpFood	Manages the pop-up window used to create a new food or upload a new list of foods
Class	PopUpMeal	Manages the pop-up window used to create a new meal
Class	Main	Runs the application
Interface	BPTreeADT	Defines how our BPTree class must be made.
Interface	FoodDataADT	Defines how our FoodData class must be made.
Class	BPTree	Organizes objects by given comparable keys, and also stores their given values.

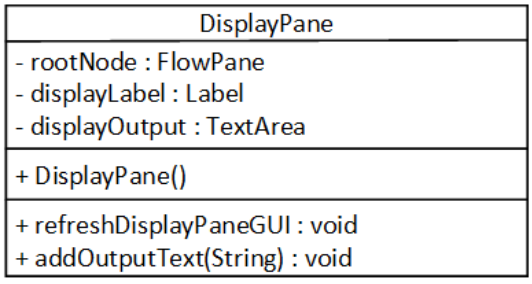
Class	FoodData	List of FoodItem objects, but also organizes them, and can be used to search by certain filters.
Class	FoodItem	A FoodItem object stores information on one type of food.

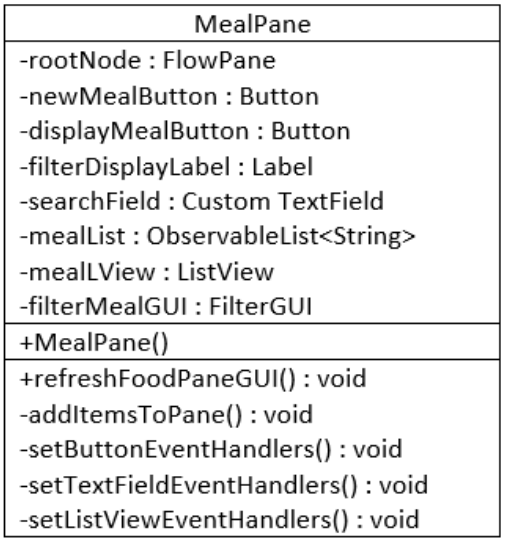
Class Diagrams:

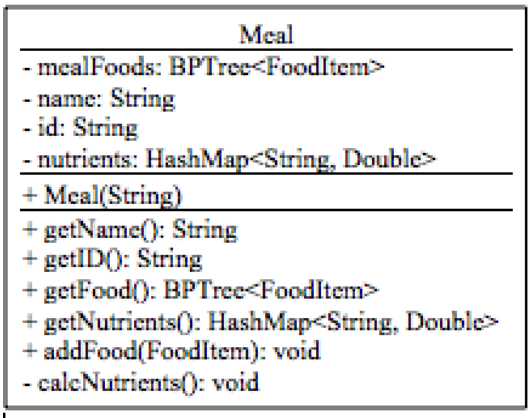
- Classes that will help with building program with the desired functionality. Classes summarized by UML diagrams or tables.

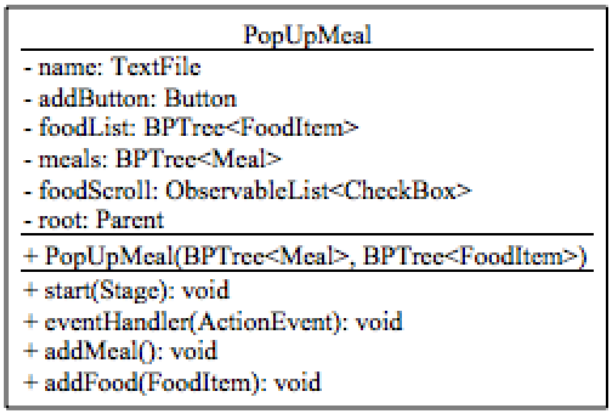


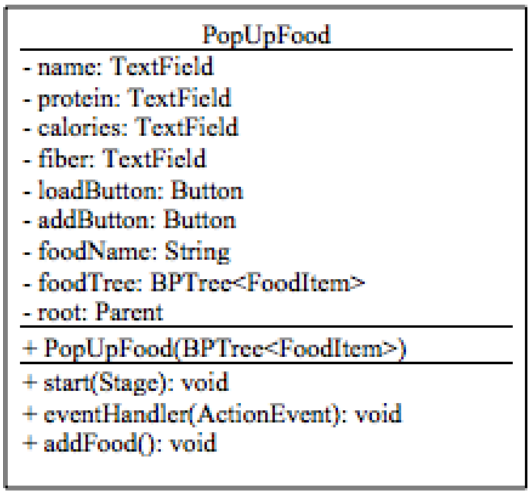


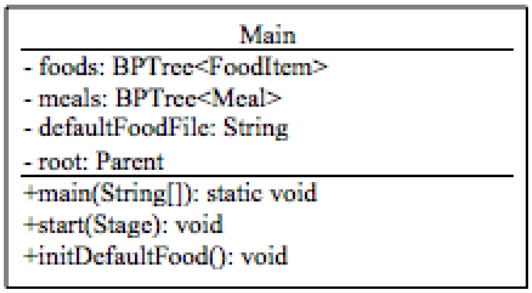












FoodFile			
Type (fields)	Name	Description	
-String	fileName	Stores text file name	
-Button	inputButton	Button instance for adding a food list from fileName.txt	
-Button	outputButton	Button instance for flushing a food list to fileName.txt	
-FileReader	textReader	Parses through user provided corpus	
-PrintWriter	textEditor	Pushes current food information to fileName.txt	
Return Type (methods)		Parameters	Description
+void	readInFile()	String filename	Reads in fileName.txt
+void	pushFile()	String fileName	Writes current food information to fileName.txt
+void	editFileName()	String fileName	Enables user to change text file name

Class BPTree<K extends Comparable<K>, V> implements BPTreeADT<K, V>

Field Type	Field Name		Description
-Node	root		Root of the tree. Everything branches off from this node.
-int	branchingFactor		Number of branches one node is allowed to have.
Return Type	Method Name	Parameter List	Description
+BPTree	BPTree	int branchingFactor	Constructs an instance of BPTree with nodes allowed to have up to as many branches as the branching factor.
+void	insert	K key, V value	Will insert a node with the key and value given as arguments.
+List<V>	rangeSearch	K key, String comparator	Will compare key to other keys in the tree using the comparator, and return a list of nodes that work with the comparator.
+String	toString	void	Returns a String representation of the entire BPTree.

-Abstract Class Node

Field Type	Field Name		Description
-List<K>	keys		List of keys.
Return Type	Method Name	Parameter List	Description
+Node	Node	void	Constructor for instance of Node.
void	insert	K key, V value	Inserts a key and value into appropriate node, and balances tree if required to.
K	getFirstLeafKey	void	Returns key for first leaf of the tree.
Node	split	void	Returns the Node for the new sibling created after splitting the tree.
List<V>	rangeSearch	K key, String comparator	Returns list of keys that hold the relationship with the key argument given by the comparator. Comparator must be either "==", "<=", or ">=".
boolean	isOverflow	void	Returns true if node is overflowed.
String	toString	void	Returns String representation of node.

-Class InternalNode

Field Type	Field Name		Description
List<Node>	children		List of all children nodes.
Return Type	Method Name	Parameter List	Description
InternalNode	InternalNode	void	Constructs and returns instance of InternalNode.
K	getFirstLeafKey	void	Returns key of first leaf in the tree.
boolean	isOverflow	void	Returns true if the node is overflowed.
void	insert	K key, V value	Inserts given key and value into the tree.
Node	split	void	Returns the Node for the new sibling created after splitting the tree.
List<V>	rangeSearch	K key, String comparator	Returns list of values that hold the relationship given by the comparator with the key given as an argument.

-Class LeafNode extends Node

Field Type	Field Name		Description
List<V>	Values		List of stored values.
LeafNode	next		Next LeafNode in the tree.
LeafNode	previous		Previous LeafNode in the tree.
Return Type	Method Name	Parameter List	Description
LeafNode	LeafNode	void	Constructs and returns an instance of the class LeafNode.
K	getFirstLeafKey	void	Returns key of first leaf of the tree.
boolean	isOverflow	void	Returns true if there is overflow for given node.
void	insert	K key, V value	Inserts node with given key and value into tree.
Node	split	void	Splits node and balances tree appropriately.
List<V>	rangeSearch	K key, String comparator	Returns list of values that hold the relationship given by the comparator with the key given as an argument.

Class FoodData implements FoodDataADT<FoodItem>

Field Type	Field Name		Description
-List<FoodItem>	foodItemList		List of all food items.
-HashMap<String, BPTree<Double, FoodItem>>	indexes		Map of food items organized by their nutrients.
Return Type	Field/Method Name	Parameter List	Description
+FoodData	FoodData	void	Constructor to make instance of FoodData.
+void	loadFoodItems	String filePath	Loads data from file given as argument.
+List<FoodItem>	filterByName	String substring	Returns list of food items that contain String given as argument.
+List<FoodItem>	filterByNutrients	List<String> rules	Returns list of food items that list of rules given as argument.
+void	addFoodItem	FoodItem foodItem	Adds a food item to the food items currently in the memory.
+List<FoodItem>	getAllFoodItems	String filename	Saves food items to file given as argument in ascending order.

Class FoodItem

Field Type	Field Name		Description
-String	name		Name of food item.
-String	id		ID of food item.
-HashMap<String, Double>	nutrients		Map of nutrients, and the corresponding value.
Return Type	Method Name	Parameter List	Description
+FoodItem	FoodItem	String id, String name	Constructor to make instance of FoodItem with given id and name.
+String	getName	void	Returns name of food item.
+String	getID	void	Returns ID of food item.
+HashMap	getNutrients	void	Returns map of food's nutrient, and the corresponding values.
+void	addNutrient	String name, double value	Adds nutrient given as argument to the field containing the map of the food's nutrients.
+double	getNutrientValue	String name	Returns value of nutrient by finding value for nutrient given by the String parameter.

Interface FoodDataADT<F extends FoodItem>

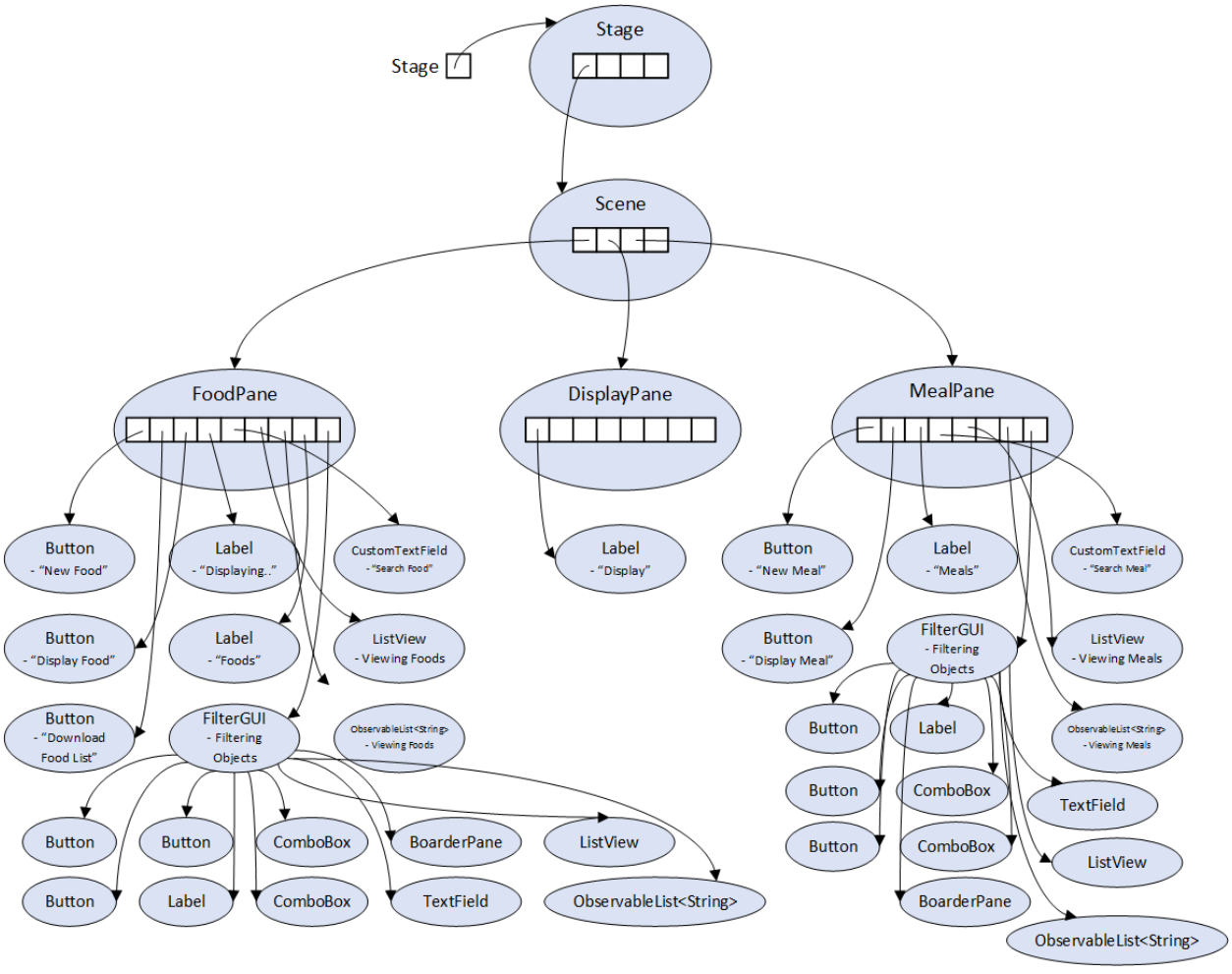
void loadFoodItems(String filePath)	Loads food and data from file given as argument into the memory.
List<F> filterByName(String substring)	Returns list of all food items containing substring given as argument.
List<F> filterByNutrients(String subsgtring)	Returns list of all food items that follow the list of rules given as argument.
addFoodItem(FoodItem foodItem)	Adds the food item given as argument to the food items in the memory.
List<FoodItem> getAllFoodItems()	Returns a list of all food items loaded in memory.
saveFoodItems(String filename)	Writes to file given as argument to store data on all food items.

Interface BPTreeADT<K, V>

Insert(K key, V value)	Inserts the key and value into the tree. Placement in tree is determined by key.
List<V> rangeSearch(K key, String comparator)	Returns list of items that satisfy comparator using key. Comparator can be "<=", ">=", or "==".
String toString()	Returns a String representing the entire tree.

Object Diagram:

- Object instances that exist when program launches. These objects are responsible for GUI display, so back-end data structures are not shown.



GUI Layout Sketch:

- Intended GUI at the start of the Food Planner Program.