1 Interface Description

1.1 Model Package

The Model package contains the classes that deal with data representation. It will also contain all connections to the database, and methods of using all of the data types we may need to use in the controller class later.

1.1.1 PersistManager Class

Public class that handles transferring the data currently active in the application to persistent data in a database. It also handles retrieving the data from the database. There are 6 public functions, that are listed below. It uses the Monster class and it works by translating a Monster instance into a set of database friendly fields. It is used by a lot of the classes in the Controller package, so that they can retain information and process it later.

init Function A public function that when called with initialize the class and instantiates the Persistence class to deal with the database. It takes no arguments and returns nothing.

create Function A public function that will create all of the fields in the database for a Monster instance. Takes a Monster instance as an argument.

update Function A public function to update an entry for a monster or a user in the database, takes a Monster instance or a User instance and returns nothing.

search (monster) Function A public function for searching the database to find a list of a user's Monster instances. Takes a User instance and returns an array of Monster instances.

search (user) Function A public function for finding a user in the database takes a nothing and returns a User instance and returns nothing.

delete(monster) Function A public function to delete the representations of the Monster instances from the database. Takes a Monster instance and returns nothing.

delete (user) Function A public function to delete the representations of the User instances from the database. Takes a User instance and returns nothing.

isReal Function A public function to check if the user is stored in the database. Takes two Strings a username and a password. Returns a boolean.

shutdown Function A public function for closing a connection to the database. This function requires nothing and returns nothing.

1.1.2 User Class

A public class to represent a user, it does not inherit from any classes. There are no public variables because everything is dealt with by the PersistManager class.

ChangeBattleMonster Function A public function to change which user's Monster is set for use in a battle situation. Takes a Monster instance to set as the users new battle Monster.

1.1.3 UserFactory Class

A public class designed to create new User instances when called from somewhere. It has one public function, to create the User.

makeIt Function This public function will create a User instance, that will later be stored by the PersistManager. Likely called by the Login Class from the Controller package. It requires the following arguments:

- name: String - The name of the new user.
- password: String - The password of the new user.
- vMoney: int - The amount of money the new user has.
- type: Type - The user's type, ie. regular user or superuser.

This function will return a User instance.

1.1.4 Monster Class

The Monster class only has public getters and setters, it represents each monster and contains everything that is necessary to represent one. The PersistManager uses the Monster class when it converts each monster into a database friendly form.

1.1.5 MonsterFactory Class

A public class to create Monster instances that will later be stored by the PersistManager as part of a User instance. It has one public function.

makeIt Function This public function will create a Monster instance. It may be called from the UserFactory class when a new user is created to make their first Monster. It might also be called from anywhere else that needs to create a new Monster instance, such as the Breed (servlet) class. It needs to be called with the following arguments:

- name: String - The name of the new monster.
- height: int - The new monster's height.
- age: int - The new monster's age (in days I expect).
- strength: int - The new monster's strength.
- health: int - The new monster's health.
- sex: Boolean - The new monster's sex as an isMale boolean.
- aggression: int - The aggression of the new monster.
- dob: int - The new monster's Date of Birth.
- battleMonster: Boolean - States if the new monster is the monster used for battle.

This function returns a Monster instance.

1.2 Controller Package

The Controller package contains all the classes that deal with the logic behind the back end of the application. It contains all the servlets and uses the Model to represent data for it.

1.2.1 CreateMonsterAndUser Class

A public class used by the Login (servlet) Class to handle creating a user and the user's first monster monster. It will rely on the MonsterFactory and UserFactory in the Model. There are two public functions.

CreateMonsterAndUser (Constructor) Function This function will set up the class, by creating a User and with it, it's first Monster. It takes no arguments and returns nothing.

CreateMonster Function A public function to create a Monster instance, it takes no arguments and returns a Monster instance. Presumably used by the constructor.

1.2.2 Login (Servlet) Class

The public class login handles the authentication of the users it is a servlet class and is used over http. There are a number of public functions to deal with users.

checkPass Function A public function for checking a password is correct. Takes a String and returns a Boolean value that reflects whether the password is correct.

checkUserName Function A public function for checking if a user name is correct. It takes a string and returns a boolean value that reflects if the user name is correct.

register Function A public function to register a new user into the system. It needs to be passed two variables; username and password both as strings. it returns nothing.

Get Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

Post Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

1.2.3 Shop (Servlet) Class

This servlet class handles all of the information required to deal with the shop, it allows users to buy and sell Monsters and to find out prices for other listed items.

calcWorth Function A public function that calculates how much a monster is worth so that it can be displayed in the shop. Uses a local variable of Monsters in the shop to find the monsters, and returns an integer that represents the price.

buy Function A public function that allows the user to buy the selected monster from the shop if they can afford it. It takes no arguments and returns nothing.

sell Function Public function that allows the user to sell the selected monster from the user's monster list for the calculated worth of the monster. It takes no arguments and returns nothing.

viewVMoney Function A public function for finding out how much money the user has, because this is needed for display in the shop. There are no arguments and nothing is returned.

addMonster Function A public function that adds a monster to the shop, it takes no arguments and returns nothing.

Get Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

Post Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

1.2.4 FriendsList (Servlet) Class

A public class to deal with the Friends List and all the information required, it also allows users to add, view, and remove from their friends list.

addFriend Function Public function that allows the user to send a friend request and if accepted friend ID will be added to friends array.

removeFriend Function Public function that asks the user to confirm the removal of the specified friend ID, if confirmed, the specified friend ID will be removed from the friend array.

requestFight Function Public function that allows the user to pick a friend from the friend list and to start a battle against the friend's preset battle monster.

getFriend Function A public function that allows the user to pull out a friends profile. Takes no arguments and returns nothing.

findUserMonsters Function A public function to find all of another user's monsters. This function takes no arguments and returns nothing.

updateList Function A public function to update a user's friend list from the server. Takes no arguments and returns nothing.

Get Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

Post Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

1.2.5 TradeMonsters (Servlet) Class

A public (servlet) class to handle monsters being traded.

TradeMonster Function A public constructor function that takes two Monster class instances as an argument, it then swaps them over and returns nothing.

Get Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

Post Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

1.2.6 Breed (Servlet) Class

A public (servlet) class to handle how you might breed monsters in the game. It allows you to perform all relevant checks to the data. There are a few public functions made available. There are no public variables. The only important things are the two parent Monsters, they are passed in as parameters to the createChild function.

isSameSex Function This is a public function to work out if two given Monsters are of the same sex and therefore cannot breed. It takes two Monster instances and returns a boolean that reflects if they are of the same sex or not.

calcCost Function This is a public function to work out what it would cost to buy a particular monster, takes a Monster instance and returns an int that is the price to buy the given monster.

createChild Function A public function that allows the user to breed a monster, takes two Monster instances with different sexes and returns a Monster instance that is a child of the two given monsters.

Get Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

Post Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

1.2.7 SimulateFight (Servlet) Class

A public (servlet) class that represents two monsters fighting in the game. There is only one public function available and no public variables.

calcWinner Function A public function to work out which Monster is going to come off better in a fight. Takes two Monster instances, returns the winning Monster instance.

isPrized Function A public function to work out if a match is prized, takes nothing as an argument, but returns a boolean.

prizeAmount Function A public function to return the amount that is given as a prize, takes nothing as an argument and returns an int that represents the prize money.

Get Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.

Post Function A public function so that this class can be used by Glassfish to deal with web requests. Takes no arguments, returns nothing.