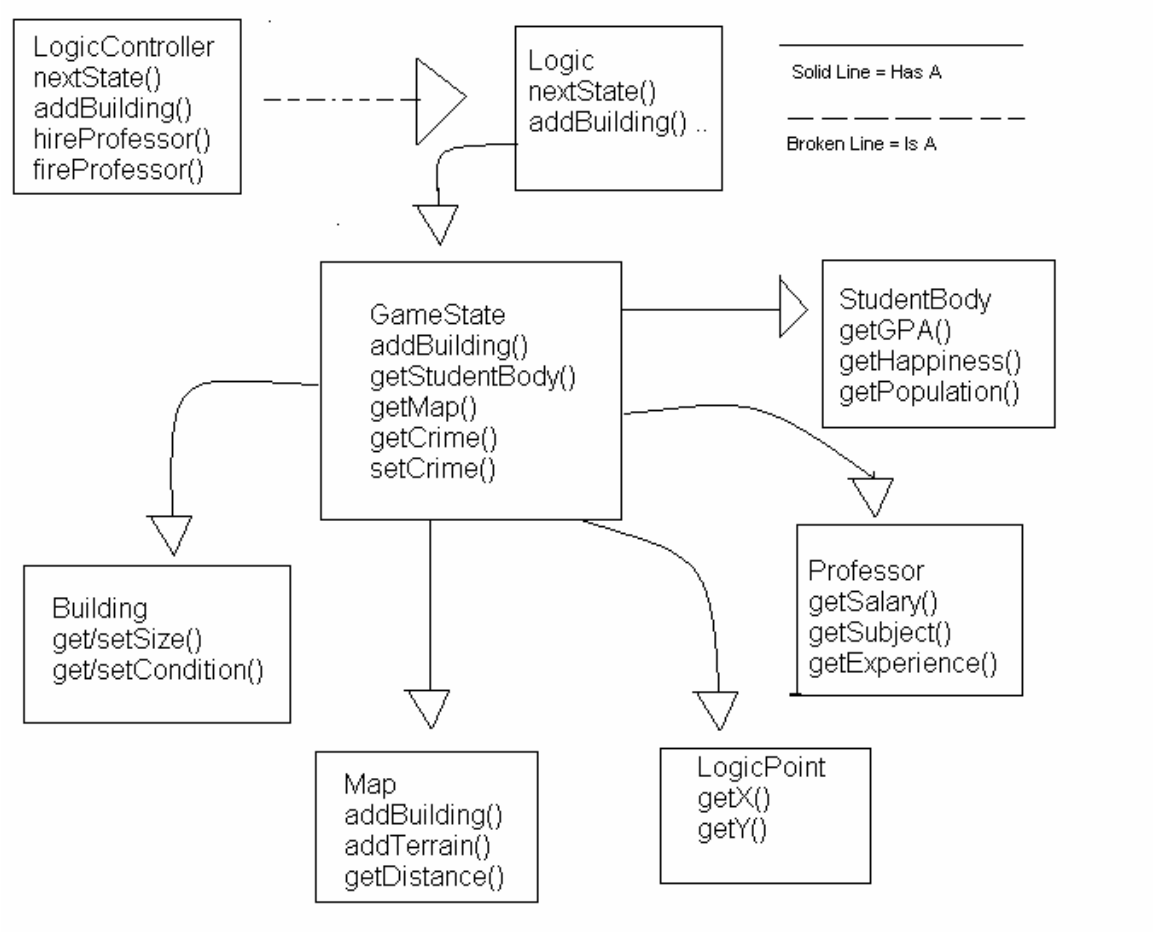

When starting a game, GUI will instantiate Logic, and Logic will instantiate an instance of GameState. During game play, GUI will call Logic's update function periodically, and Logic will compute and update GameState based on the information in GameState. For example, if user wants to add a building, GUI will call addBuilding() in Logic, and Logic will then call addBuilding() in GameState. GameState will check in the map to see if the map is occupied at that specific location. If not, GameState will instantiate new Building passing a reference of Map and ask the building to add itself into the Map. Delete building works more or less the same. GUI will call Logic's deleteBuilding() function, and then Logic will call GameState's deleteBuilding method. GameState will delete the Building object from the list of buildings and the Building will delete itself from the Map.

When doing File IO, Logic will call save/load on File IO passing in an instance of the current GameState(or a new GameState when loading), FileIO will then fill in/extract the corresponding information from GameState as needed.

Important Functions Within Classes

GameState.H

```
# add a new building into Map with location, type of building and name of building
  virtual void addBuilding(LogicPoint p,BuildingType type, std::string name);

# delete a building at location position
  virtual void deleteBuilding(int position);

# return the distance between 2 buildings
  virtual int getDistance(Building *a, Building *b);
```

This class holds the entire state of the game. The most important component is holding each Building object. Each Building has a location, a type, a state of repair, etc. GameState also holds an internal Map, which is used to check to see if a square is occupied, and whether it can be built on. The most important functions are those that allow Logic to get and set the aspects of the game, including Buildings, Professors, the StudentBody and StudentAnimations.

The member variables include vectors of Buildings, Professors, and StudentAnimations, and a Budget, a Map, and a StudentBody. There is also a year, a school ranking, a crime level, and a food quality, represented as ints. GameState is an object that's constantly being updated by Logic after each update() being called and can also be passed around to extract or fill in information.

LogicPoint.H

This is a convenience class, that represents a point on the abstract map. Buildings and StudentAnimations occupy one or more LogicPoints, which are simply a pair of integers between 0 and the size of the (square) map. The X and Y coordinates can be given through get() functions, and are returned as simple ints.

Building.H

```
# constructor takes in location of building, type of building, a pointer to Map and name of
# building as parameters
  Building(LogicPoint p,BuildingType type, Map *map, std::string name);
```

Map.H

```
# add/delete a building into/from map
  virtual void addBuilding(LogicPoint p);
  virtual void deleteBuilding(LogicPoint p);

# get distance between 2 points in map
  virtual int getDistance(LogicPoint p1, LogicPoint p2);
```

```
# check if a tile is occupied
  virtual bool isOccopied(LogicPoint p);
```