*RUBY*



Ruby's supposed to be a little story about a kid who just wants to save her dad. It's not too complicated story wise. You play as her to find loot around the island she lives on to sell to pay for her dad's doctor bills.

Ideally, my game was supposed to be pretty staightforward. The player had to just simply search the three areas until they were sastisfied with the amount of look and then sell it to find out their fate. There were little wildcards that made it so that if you got a diamond you could just go straightaway to the merchant for a quick, funny little win. The randomization variables ensure every play is different.

The player begins in the center of the island, and from there must choose where to go from there by typing in directions. As the player explores the island, they must look around for trasures to sell to the merchant. Each time a treasure is picked up the player heads back to the island center to give them a chance to head into town and sell what they have to the merchant before the day finally ends and the fate of Ruby's dad is revealed based on the amount of money collected. Until the merchant is spoken to, the game gives the player a chance to cycle between locations to pick up treasure as much as they please until they see the merchant and find out if they've won or not.





I tried to do some inheritance but it ended up morphing into something completely useless unfortunately. I did leave the connection there, though.I found that just a generic object and items as well as pricing had a similar enough structure to where it would save a bit of time to just inherit from object as the base class. The encapsulation part may have been my downfall. at some point in the week I ended up just getting so tangled in changing the public-private-static modifiers that I think I ended up absolutely ruining it, which is so incredibly frustrating and disheartening.

EDIT:

After a few days of refactoring and switching to an entirely different IDE, I finally got it working. I managed to untangle everything and removed a lot of redundant code, even if it made the game a little less complex. A lot of what was wrong was improperly placed static and not labeling lists properly and Rider let me know where Visual Studio didn't, which was extremely helpful.