GitHub This repository Search Explore Features Enterprise Pricing Sign up Sign in

TelerikAcademy / **Object-Oriented-Programming** Watch 16 Star 8 Fork 30

Branch: **master** **Object-Oriented-Programming** / 01. Defining Classes - Part 1 / **README.md**

evlogihr on 9 Mar Added homework to 01. Defining Classes - Part 1

2 contributors

56 lines (43 sloc) 2.689 kB Raw Blame History

Homework: Defining Classes - Part 1

Problem 1. Define class

- Define a class that holds information about a mobile phone device: model, manufacturer, price, owner, battery characteristics (model, hours idle and hours talk) and display characteristics (size and number of colors).
- Define 3 separate classes (class `GSM` holding instances of the classes `Battery` and `Display`).

Problem 2. Constructors

- Define several constructors for the defined classes that take different sets of arguments (the full information for the class or part of it).
- Assume that model and manufacturer are mandatory (the others are optional). All unknown data fill with null.

Problem 3. Enumeration

- Add an enumeration `BatteryType` (Li-Ion, NiMH, NiCd, …) and use it as a new field for the batteries.

Problem 4. ToString

- Add a method in the `GSM` class for displaying all information about it.
- Try to override `ToString()` .

Problem 5. Properties

- Use properties to encapsulate the data fields inside the `GSM` , `Battery` and `Display` classes.
- Ensure all fields hold correct data at any given time.

Problem 6. Static field

- Add a static field and a property `IPhone4S` in the GSM class to hold the information about iPhone 4S.

Problem 7. GSM test

- Write a class `GSMTest` to test the `GSM` class:
  - Create an array of few instances of the `GSM` class.
  - Display the information about the GSMs in the array.
  - Display the information about the static property `IPhone4S` .

Problem 8. Calls

- Create a class `Call` to hold a call performed through a GSM.
- It should contain date, time, dialled phone number and duration (in seconds).

Problem 9. Call history

- Add a property `CallHistory` in the `GSM` class to hold a list of the performed calls.
- Try to use the system class `List<Call>` .

Problem 10. Add/Delete calls

- Add methods in the `GSM` class for adding and deleting calls from the calls history.
- Add a method to clear the call history.

Problem 11. Call price

- Add a method that calculates the total price of the calls in the call history.
- Assume the price per minute is fixed and is provided as a parameter.

Problem 12. Call history test

- Write a class `GSMCallHistoryTest` to test the call history functionality of the `GSM` class.
  - Create an instance of the `GSM` class.
  - Add few calls.
  - Display the information about the calls.
  - Assuming that the price per minute is `0.37` calculate and print the total price of the calls in the history.
  - Remove the longest call from the history and calculate the total price again.
  - Finally clear the call history and print it.