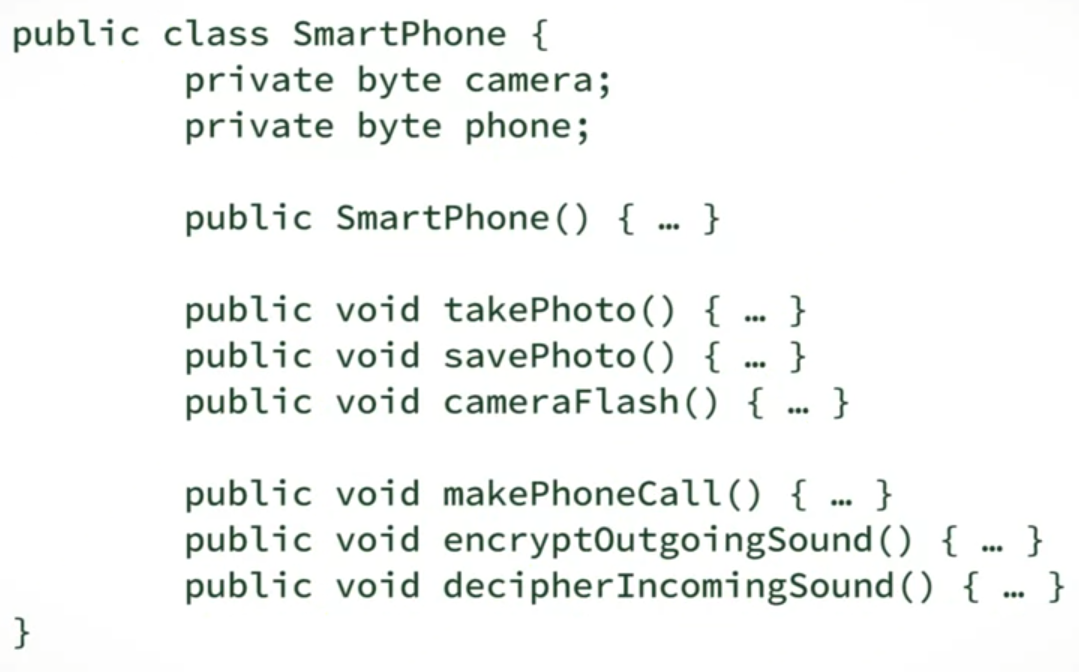


* Removed the concern of how to get food from the Dog
  + Let the DogOwner handle that issue

Another example



* Low cohesion
  + Camera behavior do not need to be encapsulated within the phone in order to do its job
  + We cannot replace the camera without removing the camera COMPLETELY from this class

Our SmartPhone class can be better designed by applying separation of concerns. Pick the two design changes that we can make in order to build a better SmartPhone.



Remove all of the behaviors for the camera and the phone, and put them into a single new class.



Remove all the behaviors of the camera and the phone, and put them into separate new classes.

Correct

Good job! IT's important to separate these responsibilities into different classes.



Remove the attributes from SmartPhone class.

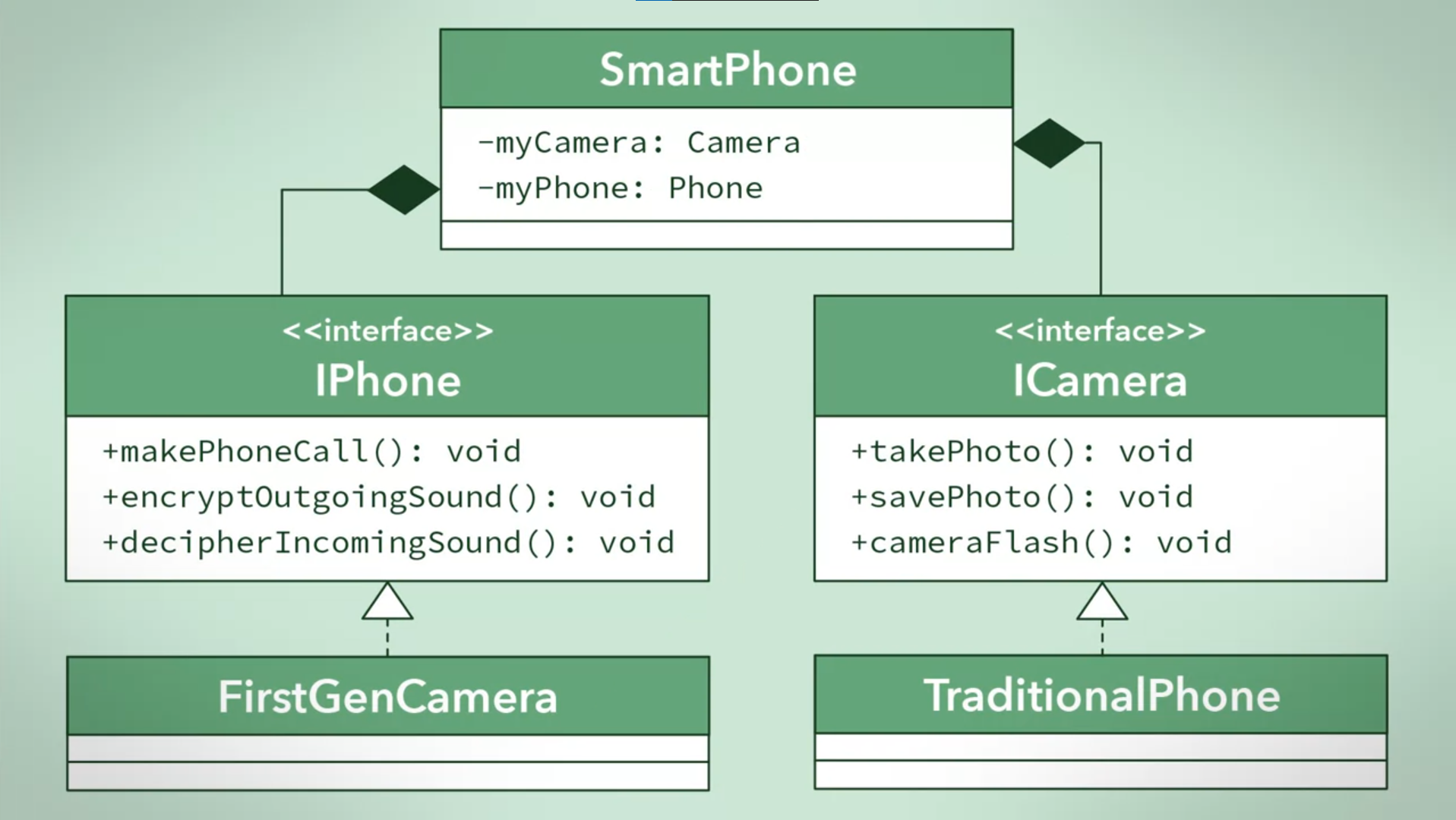


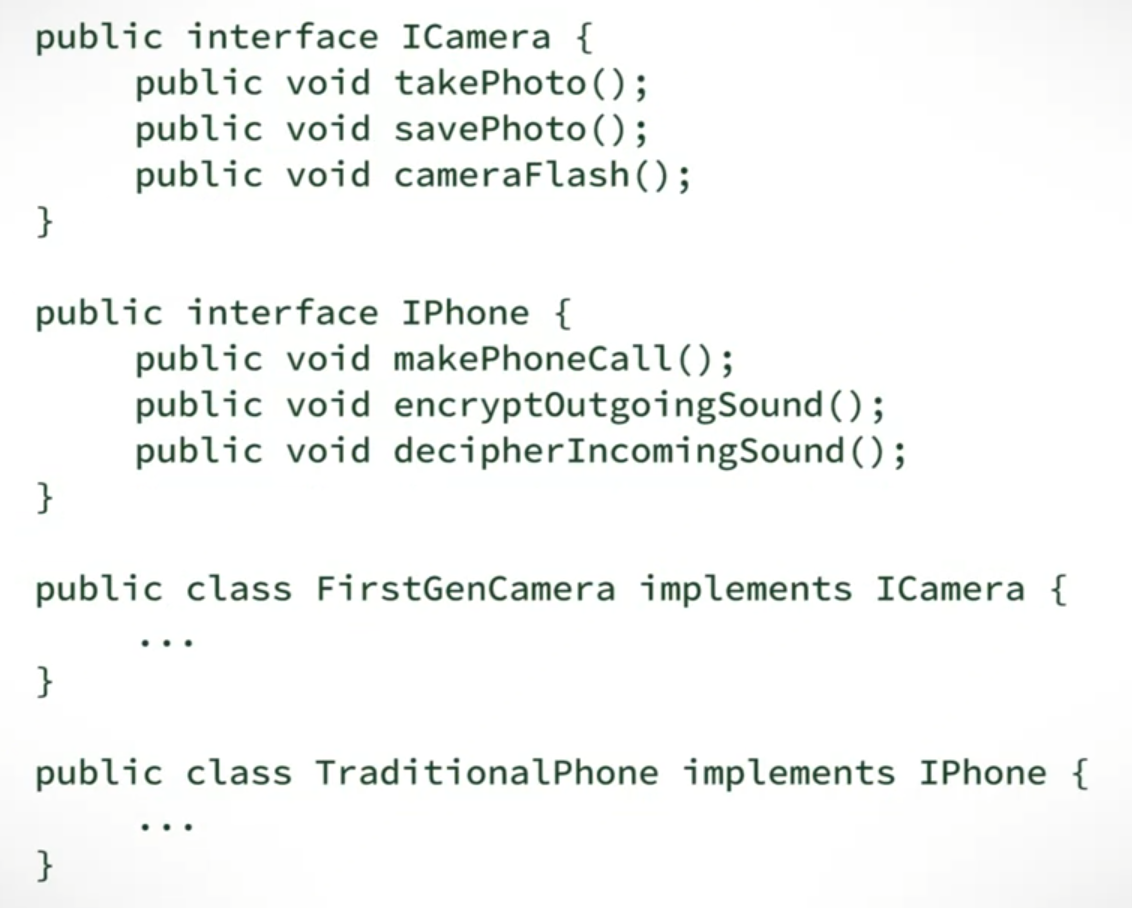
Change the attributes in the SmartPhone class to reference instances of the new classes.

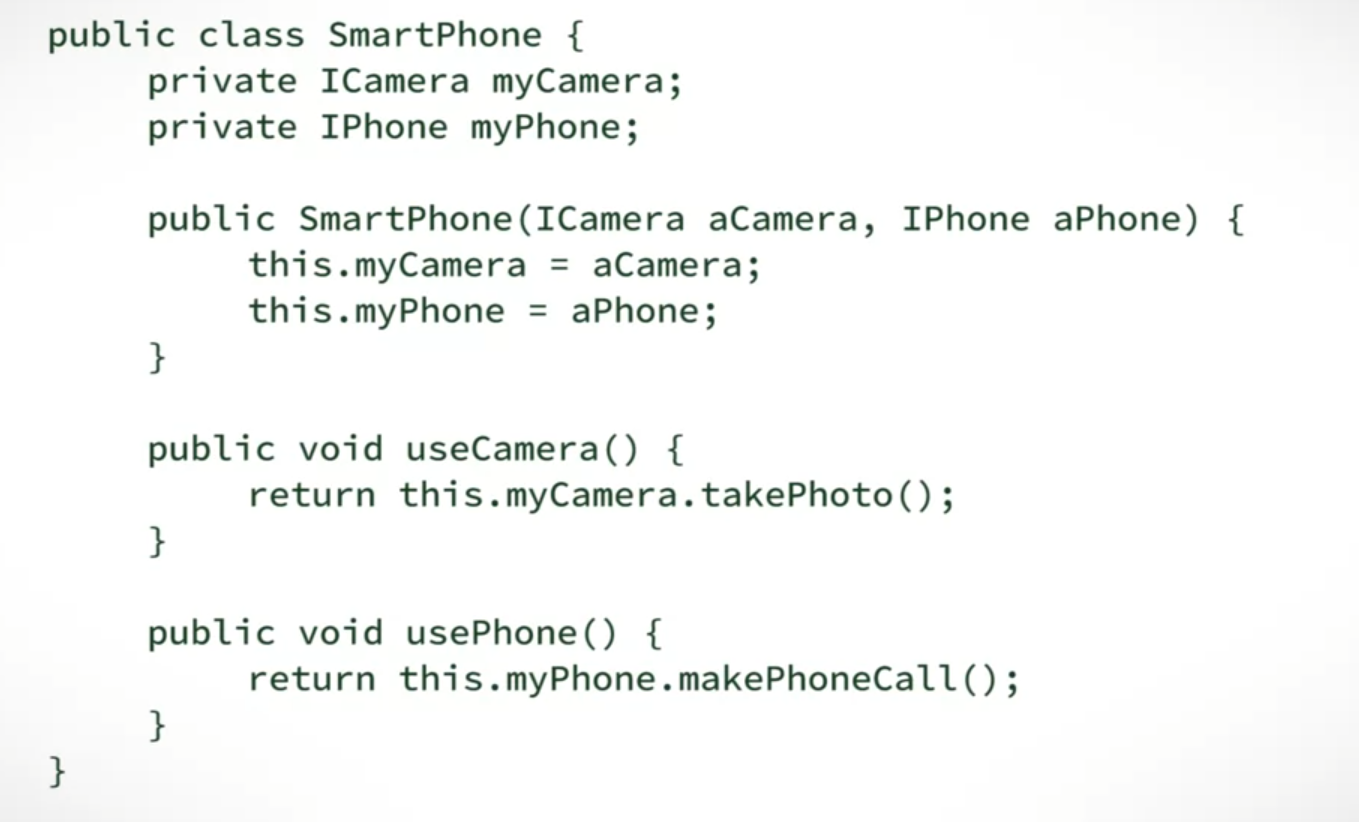
Correct

Yes! The SmartPhone class can be composed with other objects by having them as attributes.









Microsoft in the process of redesigning their SmartPhone in order to increase their market share. As a new hire with years of industry experience, your manager has asked for your opinion of the current design. You noticed that speakers and LED lighting controls are all encapsulated in a single Utility class, and make a suggestion to separate the two into their own classes.

**What are the three reasons that you would give for this design change?**



It is easier to reuse the code because classes are more coupled with each other.



The increase in modularity helps with creating a flexible system where additional classes and functionality can be added with ease or reused in other systems.

Correct

Correct! These cohesive objects are easier to reuse than objects that try to do many different things.



The system becomes easier to maintain because classes are more specialized and contain less code.

Correct

Right! Each class is easier to maintain because its purpose is clear and you don't have to wade through code to fix one specific aspect.



Cohesion of the system is increased by encapsulating related functions into separate, and distinct classes.

Correct

That's right! The new Lighting and Speaker classes will be more cohesive since they encapsulate one responsibility each.