Mediator

I am producing a mediator pattern as i identified that there was certain behavior that could be used to determine calls to other objects. I constructed a mediator to handle all the implementation of that.

Class Diagram before the modification:

pattern1\classes\_SINGLETON.png

Location of code:

Interpreter.py

Lines 325-328, 330-406, 415, 418

Identified InputChecker, RegExp, and Enum as potential colleagues.

The reasons why I have applied this pattern:

I have found that the model is handling a lot of the functionality of a director for a set of checkers. The intent of observing this pattern is to make it possible for other data checkers to be mediated in without having to modify how that happens.

The mediator is responsible for coordinating between these checkers and informing other classes that require it of it. Because I was only needing to check input, the actual functionality that is expected of it (read line) is able to be handled directly by the mediator where as the actual ins and outs of it are not known to the other classes. This means that these checkers are able to be related to independently of the framework.

The Colleague classes also operated in the way that colleagues should - ie the mediator is responsible for their behavior as it pertains to implementing a data checker. This makes them independent of the actual task of producing checked input.

Class Diagram after the modification:

pattern2\classes\_MEDIATOR.png