Bytz.Collections.Dispatch

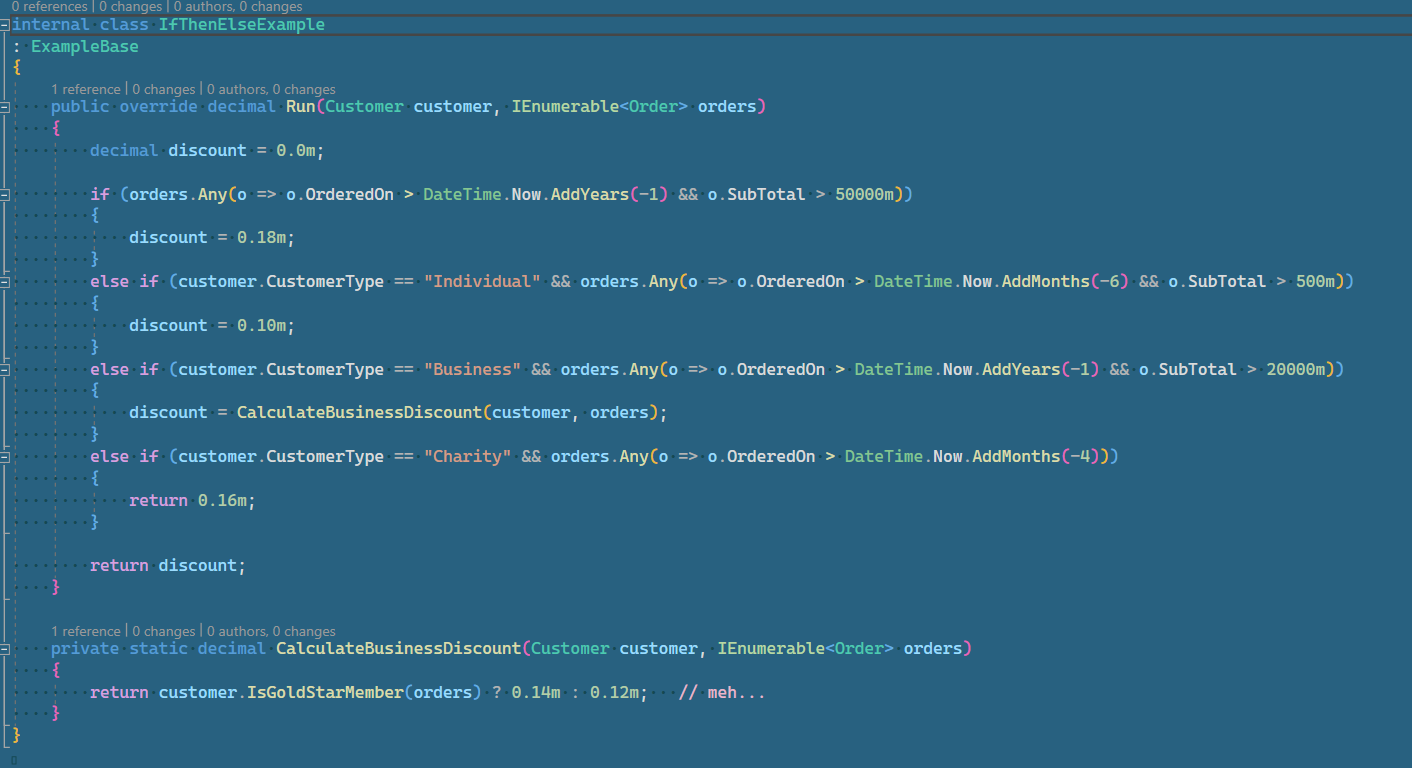
An approach to provide *branchless* coding in .NET.

This has been created out of my personal dislike for long if-then-else statements that I find myself trapped-in from time-to-time.

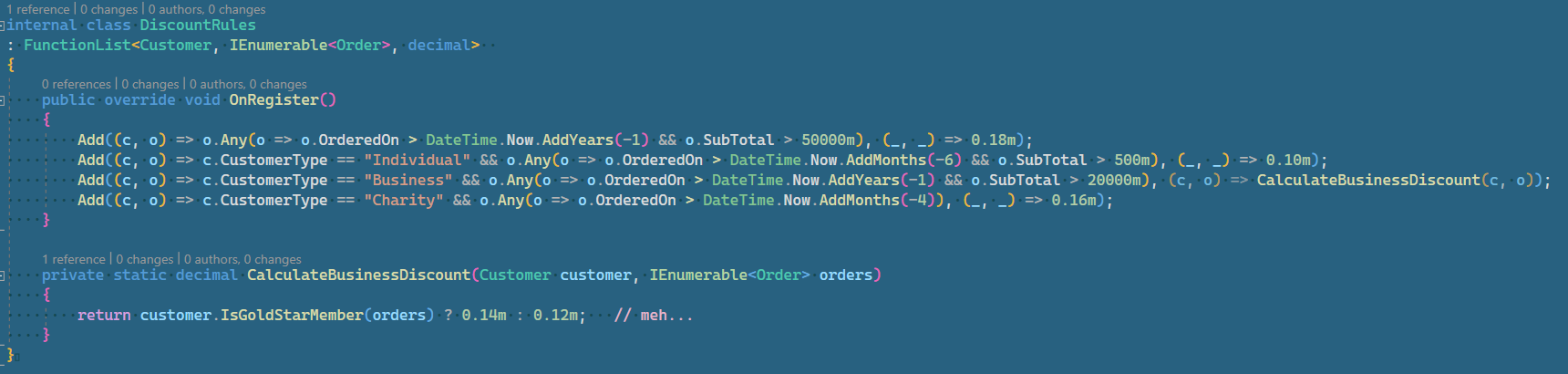
A simple example of *classical* if-then-else:

* If the customer has any orders within the last year that are over $50k, they get an 18% discount.
* If the customer is an Individual and any of their orders within the last year are over $500 they get a 10% discount.
* If the customer is a Business and they have any order within the last year that is over $20K they get a discount calculated off-of their Gold-Star status.
* If the customer is a Charity and they have any order within the last four months, they get a 16% discount.

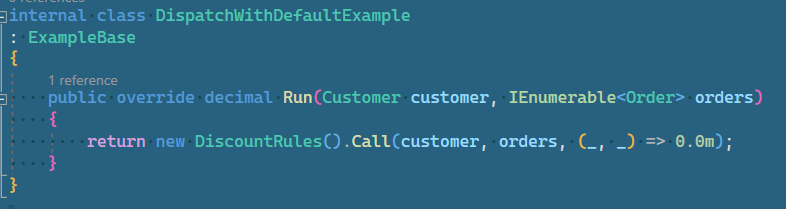
Classical If-Then Implementation:



A function-List Dispatch Implementation and invocation:

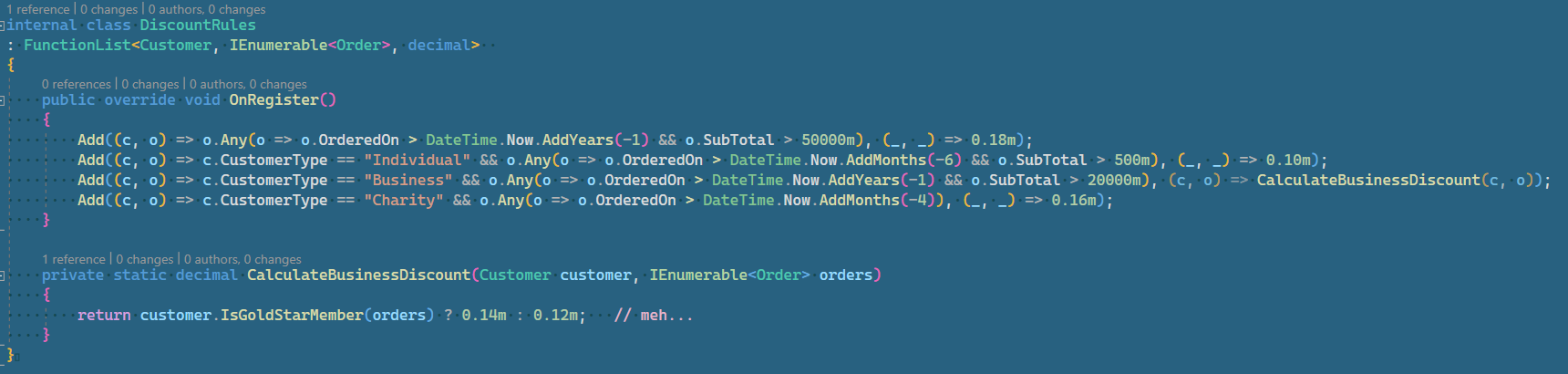
Implementation:

Invocation of calling the dispatch collection with a simple return default of zero



While this may seem a little more succinct in approach, and at the surface may be considered “sugary”, we now have the ability to ***test*** the conditions outside of actually *executing* the code.

Regurgitating the example from before:



With the *OnRegister* override, we can define a list of predicates and associated functions to be called. Since they are created via the *OnRegister* method (or through a collection initializer {more on this later}), the ordinal position is known.

***Note****: When testing rule declarations (*action-list or function-list*), it is an* **expectation***that once configured, that the ordinal positions of the defined conditions will* not *change for any reason once established. This is a* **reasonable****assumption** *in that the ordinal positions of each condition will* never *change, and should not be taken as gospel until vetted by others.*

have defined the conditions within the *OnRegister* action, we can now create unit-tests based on the *ordinal* position of a specific rule defined in this *DiscountRules* function-list.

By using the *IndexOf* method, we can determine the *{zero-based}* ordinal of a matching key within the list of dispatch predicates. If a match is not found, then an *InvalidOperationException* is thrown.

