LoggingKata

An exercise in geolocation, csv parsing, and logging

Kata Overview

Here's what you'll need to do for this Kata:

i. Clone this repo to your machine, then create a branch to accomplish your work (`git checkout -b your-branch-name`)
ii. Complete all the `TODO`s, while adding appropriate log statements along the way. You can find more details below in the Kata Details section:
    i. Start with writing a Unit Test to Test the Parse method
    ii. Implement the Parse Method
    iii. Use the Geolocation to calculate distance between two points
iii. Reduce the logging verbosity and rerun
iv. Push your changes (`git push`), create a pull request, and add request a review from your instructor.

Kata Details

Here's some more details for completing the steps above.

TacoParser

Updating the `Parse` method in your `TacoParser`

This method is used to parse a single row from your CSV file as a string and return an ITrackable:

```
public ITrackable Parse(string line)
{
    // Take your line and use line.Split(',') to split it up into an array of strings, separated
by the char ','
    var cells = line.Split(',');

    // If your array.Length is less than 3, something went wrong
    if (cells.Length < 3)
    {
        // Log that and return null
    }

    // grab the long from your array at index 0
    // grab the lat from your array at index 1
    // grab the name from your array at index 2

    // Your going to need to parse your string as a `double`
    // which is similar to parse a string as an `int`

    // You'll need to create a TacoBell class
    // that conforms to ITrackable

    // Then, you'll need an instance of the TacoBell class
    // With the name and point set correctly

    // Then, return the instance of your TacoBell class
```

```
    // Since it conforms to ITrackable
}
```

Program

You now have your `Parse` method working properly. Now, let's get into our Program file in our `Main` static method.

```
static void Main(string[] args)
{
    // DON'T FORGET TO LOG YOUR STEPS
    // Grab the path from Environment.CurrentDirectory + the name of your file

    // use File.ReadAllLines(path) to grab all the lines from your csv file
    // Log and error if you get 0 lines and a warning if you get 1 line

    // Create a new instance of your TacoParser class
    // Grab an IEnumerable of locations using the Select command: var locations = lines.Select(li
ne => parser.Parse(line));

    // Now, here's the new code

    // Create two `ITrackable` variables with initial values of `null`. These will be used to sto
re your two taco bells that are the furthest from each other.
    // Create a `double` variable to store the distance

    // Include the Geolocation toolbox, so you can compare locations
    // Do a loop for your locations to grab each location as the origin (perhaps: `locA`)
    // Create a new Coordinate with your locA's lat and long

    // Now, do another loop on the locations with the scope of your first loop, so you can grab t
he "destination" location (perhaps: `locB`)
    // Create a new Coordinate with your locB's lat and long
    // Now, compare the two using `GeoCalculator.GetDistance(origin, destination)`, which returns
 a double
    // If the distance is greater than the currently saved distance, update the distance and the
two `ITrackable` variables you set above

    // Once you've looped through everything, you've found the two Taco Bells furthest away from
each other.
}
```