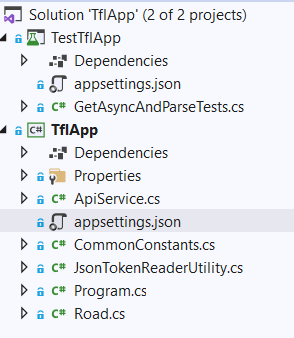
TFL Test App:

1. How to build the code
   1. Download code from the link <https://github.com/keerthijm/TflApp>
      1. Download in Zip format
         1. Extract into local folder
         2. Open TflApp.sln file – Either in Visual Studio 2019 or in Visual Studio Code
         3. Note: require .Net 5.0 framework
         4. Set Command line arguments from Visual Studio-> Solution Explorer-> Right click on solution-> Properties -> Debug -> Application Arguments -> Enter the Value
         5. In Visual Studio -> File->Debug -> Run
      2. Or Open with Visual Studio and repeat steps from the web link (iv, v)
2. Changing App ID and API key
   1. APP ID and API Key are stored under appsettings.json in the application
   2. 
   3. Change below configuration for the file appsettings.json under application TflApp and TestTflApp
      1. "app\_id": "PLEASE ENTER YOUR APP\_ID",
      2. "app\_key": "PLEASE ENTER YOUR APP\_KEY",
3. How to run the output – 2 options
   1. Download TflApp\Exe folder from the link above
      1. You can find exe under the folder structure ...\TflApp\Exe\TflApp\ TflApp.exe
      2. Change the appsettings.json file to add your app id and app key
      3. Run TflApp.exe in command prompt by providing argument input parameter
      4. Eg: C:\> .\RoadStatus.exe A2 or .\RoadStatus.exe A233
   2. Run the application through the visual studio as per the instruction 1
4. How to run any tests that you have written
   1. Use TestTflApp to run tests
   2. Under Visual studio 2019 -> Tests Menu-> click on Run All Tests
5. Any assumptions that you’ve made
   1. I assume users of this has some basic knowledge of running code under visual studio 2019 or visual studio code
6. Anything else you think is relevant
   1. Below is the structure of the project
      1. Main Program: main program where application starts working
      2. ApiService: Service which can GET information from apis, in future if require we can expand the service to support POST
      3. AppSettings – Where we store information related to APP ID and APP Key
      4. Common Constants – String constants used across the project are stored under the class
      5. JsonTokenReaderUtility – used to parse Json token object into result string, in future we can extend it support other Json token formats
      6. Result into road object
         1. Base Abstract class has Road class
         2. Derived into Valid and Invalid Road class
         3. Flow diagram as below:

Invalid Road

Valid Road

Road

Result String

AppSettings

CommonConstants

Main Program