Innovation Factories C/AV Challenge

Generated by Doxygen 1.8.15

1 Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Car.Car
An ADT that represents a Car **2**

Node.Node
An ADT that represents a Node **4**

2 File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

3 Class Documentation

3.1 Car.Car Class Reference

An ADT that represents a Car.

Public Member Functions

- def __init__ (self, ID, speed)
  this is the initializer method
- def get_ID (self)
  this method is used to get the unique ID of the car
- def get_speed (self)
  this method is used to get the current speed of the car
- def update_speed (self, modifier)
  this method is used to update the current speed of the car
- def __str__ (self)
  this method is return the car object and it's information in the form of a string

Public Attributes

- **speed**

3.1.1 Detailed Description

An ADT that represents a Car.

Parameters

ID	is a unique identifier that is linked to each car
speed	is the speed of the car

3.1.2 Member Function Documentation

3.1.2.1 __str__()

```
def Car.Car.__str__ (
              self )
```

this method is return the car object and it's information in the form of a string

Returns

the car's information in the form of a string

3.1.2.2 get_ID()

```
def Car.Car.get_ID (
              self )
```

this method is used to get the unique ID of the car

Returns

the ID of the car

3.1.2.3 get_speed()

```
def Car.Car.get_speed (
              self )
```

this method is used to get the current speed of the car

Returns

the speed of the car

3.1.2.4 update_speed()

```
def Car.Car.update_speed (
              self,
              modifier )
```

this method is used to update the current speed of the car

Parameters

modifier	is the multiplier that we apply to the current speed to get to the new speed

Returns

the current of the car after the update

The documentation for this class was generated from the following file:

- Car.py

3.2 Node.Node Class Reference

An ADT that represents a Node.

Public Member Functions

- def __init__ (self, humidity, audio, temperature, pressure, vibration, video)

    Node constructor.
- def get_humidity (self)

    Gets the humidity a Node records.
- def get_audio (self)

    Gets the audio a Node records.
- def get_temperature (self)

    Gets the temperature a Node records.
- def get_pressure (self)

    Gets the pressure a Node records.
- def get_vibration (self)

    Gets the vibration a Node records.
- def get_video (self)

    Gets the video a Node records.
- def determine_rain (self, audio)

    Checks conditions to determine when rain will occur.
- def determine_snow (self, video)

    Checks conditions to determine when snow will occur.
- def determine_fog (self)

    Checks conditions to determine when fog will occur.
- def determine_wind (self)

    Checks conditions to determine when wind will occur.
- def determine_day_and_night (self)

    Checks conditions to determine when the time of day is in the morning or night.
- def dynamic_speed (self, car)

    Checks conditions to determine by what magnitude to reduce the overall speed by @car Object of type car that will have its speed modified based on smallest magnitude.

Public Attributes

- **humidity**
- **audio**
- **temperature**
- **pressure**
- **vibration**
- **video**

3.2.1 Detailed Description

An ADT that represents a Node.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 __init__()

```
def Node.Node.__init__ (
              self,
              humidity,
              audio,
              temperature,
              pressure,
              vibration,
              video )
```

Node constructor.

Initializes a Node object with an empty Node

Parameters

humidity	The humidity reported in Percent
audio	The audio is how strong the rain is by using the sound
temperature	The temperature reported in Kelvin
pressure	The pressure reported in Pascals
vibration	The vibration is the vibration of the node in meters per second
video	The video is whether or not it is snowing hard or very little

3.2.3 Member Function Documentation

3.2.3.1 determine_day_and_night()

```
def Node.Node.determine_day_and_night (
              self )
```

Checks conditions to determine when the time of day is in the morning or night.

Returns

returns reduced speed by a fractional portion the time of day is night, and 1 if it is in the morning

3.2.3.2 determine_fog()

```
def Node.Node.determine_fog (
              self )
```

Checks conditions to determine when fog will occur.

Returns

returns reduced speed by a fractional portion if fog exists and 0 otherwise

3.2.3.3 determine_rain()

```
def Node.Node.determine_rain (
              self,
              audio )
```

Checks conditions to determine when rain will occur.

Parameters

audio	The audio is used to determine hard rain or light rain

Returns

returns reduced speed by a fractional portion if rain exists and 0 if otherwise

3.2.3.4 determine_snow()

```
def Node.Node.determine_snow (
              self,
              video )
```

Checks conditions to determine when snow will occur.

Parameters

video	The video is used to determine heavy snow, light snow, or medium snow

Returns

returns reduced speed by a fractional portion if snow exists and 0 otherwise

3.2.3.5 determine_wind()

```
def Node.Node.determine_wind (
              self )
```

Checks conditions to determine when wind will occur.

Returns

returns reduced speed by a fractional portion if high wind exists, 1 if windkmh is less than or equal to 25 and 0 otherwise

3.2.3.6 dynamic_speed()

```
def Node.Node.dynamic_speed (
              self,
              car )
```

Checks conditions to determine by what magnitude to reduce the overall speed by @car Object of type car that will have its speed modified based on smallest magnitude.

Returns

returns reduced speed based on the smallest magnitude reduced

3.2.3.7 get_audio()

```
def Node.Node.get_audio (
              self )
```

Gets the audio a Node records.

Returns

returns the audio

3.2.3.8 get_humidity()

```
def Node.Node.get_humidity (
              self )
```

Gets the humidity a Node records.

Returns

returns the humidity

3.2.3.9 get_pressure()

```
def Node.Node.get_pressure (
              self )
```

Gets the pressure a Node records.

Returns

returns the pressure

3.2.3.10 get_temperature()

```
def Node.Node.get_temperature (
              self )
```

Gets the temperature a Node records.

Returns

returns the temperature

3.2.3.11 get_vibration()

```
def Node.Node.get_vibration (
              self )
```

Gets the vibration a Node records.

Returns

returns the vibration

3.2.3.12 get_video()

```
def Node.Node.get_video (
              self )
```

Gets the video a Node records.

Returns

returns the vibration

The documentation for this class was generated from the following file:

- Node.py

4 File Documentation

4.1 Car.py File Reference

Provides the Abstract Data Type for Car.

Classes

- class Car.Car

  An ADT that represents a Car.

4.1.1 Detailed Description

Provides the Abstract Data Type for Car.

Author

Mostafa Mohsen, Chris Vishnu, Seif El Tobgy, Saif Fadhel

Date

26/01/2020

4.2 Node.py File Reference

Provides the Abstract Data Type for Node.

Classes

- class Node.Node

  An ADT that represents a Node.

Variables

- int Node.dewpoint = 273

  A constant that is set to Hamilton on January 26th 2020 for the dewpoint in kelvin.

4.2.1 Detailed Description

Provides the Abstract Data Type for Node.

Author

Mostafa Mohsen, Chris Vishnu, Seif El Tobgy, Saif Fadhel

Date

26/01/2020

4.3 Snow_protocol.py File Reference

Provides function for how to react when apporaching snow patch.

Functions

- def Snow_protocol.Protocol (car, node, patch_length, distance)

  Node constructor.

Variables

- float **Snow_protocol.threshhold** = 4.5
- **Snow_protocol.car1** = Car(10, 80)
- **Snow_protocol.node1** = Node(0.5, "he", -1, 1000000, 0, "light snow")

4.3.1 Detailed Description

Provides function for how to react when apporaching snow patch.

Author

Mostafa Mohsen, Chris Vishnu, Seif El Tobgy, Saif Fadhel

Date

26/01/2020

4.3.2 Function Documentation

4.3.2.1 Protocol()

```
def Snow_protocol.Protocol (
              car,
              node,
              patch_length,
              distance )
```

Node constructor.

Parameters

car	the car is a object that contains that speed and ID of the vehicle
node	the node is a unit that contains various sensors that measure environmental conditions
patch_length	the length of the patch of snow
distance	the distance from the car to the patch of snow

Returns

a string that represents the set of protocols to follow when you approach a patch of snow

Index