Racer 2D

A 2d racing game application that computes collision between different user defined objects like Cars following a Race track and having arbitrary shapes made up of rectangles, circles or any closed polygon (figure 1).

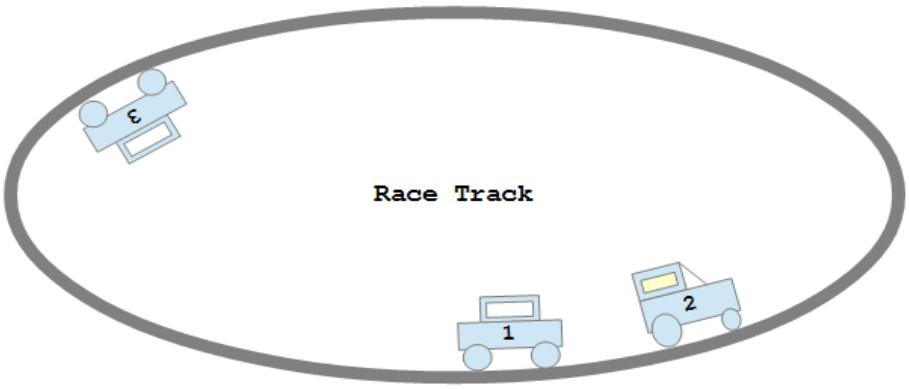


Figure 1

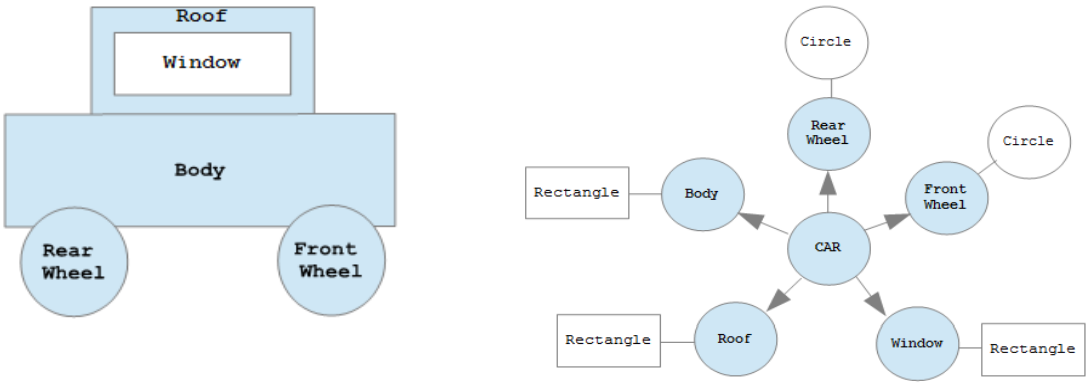
A user can define an object with its attributes like shape, color and speed in a file and give it to the application to load the objects and start the race.

Object file is a text file with extension (.gbj) in a specific format as depicted in (figure 2)

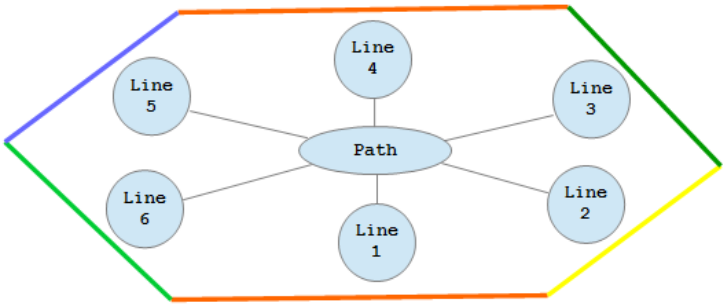
```
Objects
2 .... number of objects
Object
{
    Assembly
    {
        Component
        {
            Part
            {
                ...attributes like shape, color, transformation
            }
            Part
            {
                ...attributes like shape, color, transformation
            }
        }
    }
    Path
    {
        Part
        {
            ...attributes like shape, color, transformation
        }
    }
    ...attributes for object like speed
}
Object
{
    ... Construct the Assembly and Path similarly
}
```

Figure 2

Example for constructing a Car with a race track



Car Assembly



Path

- Specify an Objects tag with number of objects inside it

```
Objects
1           .... number of objects
```

- Define an object with an Assembly, Path and object attributes

```
Object
{
      Assembly
      {
            Name   - tag
            Car    - name given to the assembly
      }
      Path
      {
      }
      Speed - tag
      10.0  - value
}
```

- Create parts of car like body, roof, wheels and window, and place it in components tag inside the Assembly

```
Assembly
{
	Name	- tag
	Car	- name given to the assembly
	Component
	{
		Part
		{
			Name		- tag
			Body		- name given to the part
		}
		Part
		{
			Name		- tag
			Roof		- name given to the part
		}
		Part
		{
			Name		- tag
			Window		- name given to the part
		}
		Part
		{
			Name		- tag
			Front Wheel	- name given to the part
		}
		Part
		{
			Name		- tag
			Rear Wheel	- name given to the part
		}
	}
}
```

- For each part define its shape (in case of body)

```
Part
{
	Name	- tag
	Body	- name given to the part
	Shape		- tag
	{
		Rectangle	- tag for type
		Center		- tag for center of rectangle
		0.0 0.0 0.0	- values (x, y, z)
		Width		- tag for Width
		6.0		- value
		Height		- tag for height
		3.0		- value
	}
}
```

- Define transformations like translation and rotation of the part to place it inside the assembly

```
Part
{
	Name	- tag
	Body	- name given to the part
```

```
        Shape {...}
        Transformation
        {
                Translate     - tag
                x y z         - values
                Rotate        - tag
                angle         - value in degree about Z axis
        }
}
```

- Define the display attributes for a part

```
Part
{
        Name   - tag
        Body   - name given to the part
        Shape
        {...}
        Transformation
        {...}
        Display                          - tag
        {
                Color                    - tag
                0.0 0.0 0.0              - value (R G B)
                Line Width               - tag
                2.0                      - value (thickness of line)
                Point Size               - tag
                2.0                      - value (size of point)
                Polygon Mode             - tag
                Fill or Line or Point    - value
        }
}
```

- Define path for the object to follow. In case of Polyline path, define each line as a part. ***Currently supports only Polyline***

```
Object
{
        Assembly
        {
        }
        Path
        {
                Part
                {
                        Name   - tag
                        Line 1 - name given to the part
                }
                Part
                {
                        Name   - tag
                        Line 2 - name given to the part
                }
        }
}
```

- Similarly define shape, transformation and display attributes for each part in the path

Supported Attributes

1. Shapes Attributes
   - Point

```
Shape
{
        Point
        0.0 0.0 0.0
}
```

   - Line

```
Shape
{
        Line
        Start
        0.0 0.0 0.0
        End
        100.0 100.0 0.0
}
```

   - Rectangle

```
Shape
{
        Rectangle
        Center
        0.0 0.0 0.0
        Width
        10.0
        Height
        5.0
}
```

   - Circle

```
Shape
{
        Circle
        Center
        0.0 0.0 0.0
        Radius
        10.0
}
```

   - Polygon

```
Shape
{
        Polygon
        3               ...Number of points
        0.0 0.0 0.0     ( pt 1 ... X Y Z)
        10.0 0.0 0.0
        0.0 10.0 0.0
}
```

2. Display Attributes

```
Display
{
        Color
        1.0 0.0 0.0 (R G B)
        Line Width
        2.0
        Point Size
        2.0
        Polygon Mode
        Fill or Line or Point
}
```

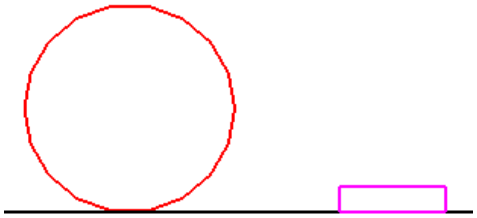
3. Transformation

```
Transformation
{
      Translate     - tag
      x y z         - values
      Rotate        - tag
      angle         - value in degree (supports only Z axis rotation)
}
```

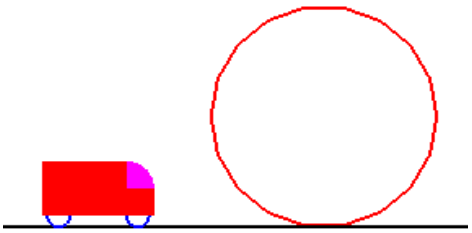
Refer to sample game object files provided in the folder Racer2D\test\unit\object

- objects_1.gbj (Assembly *car* and Path)

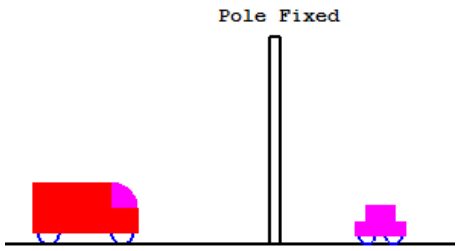
- objects_2.gbj (Part *circle*, Part *rectangle* and Path)



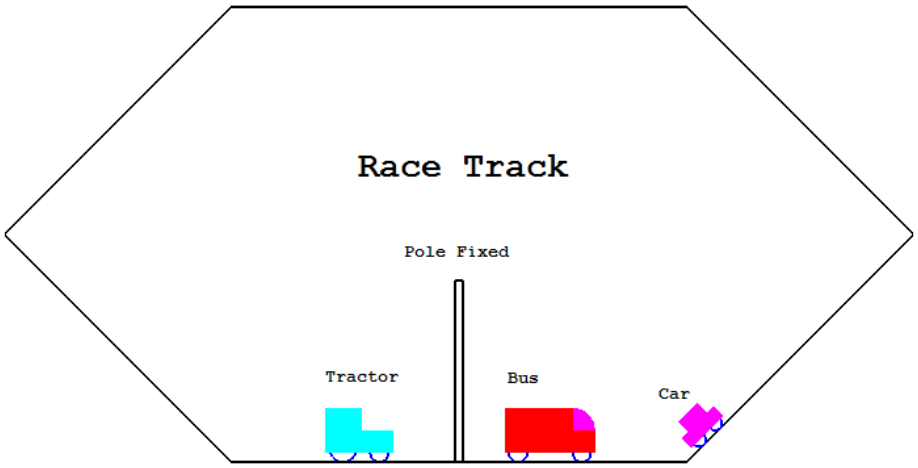
- objects_3.gbj (Assembly *bus*, Part *circle* and Path)



- objects_4.gbj (Assembly *bus*, Assembly *car*, Part *pole fixed* and Path)



- objects_5.gbj (3 Assembly, 1 Part and Path)



Viewer Options

The user can change the view background color, width, height, and location of the window through a file with extension (.viw). A sample file is provided in the folder Racer2D\test\unit\viewer

Example (viewerOpts.viw)

```
Window
{
    Width
    800
    Height
    600
    Top
    50
    Left
    50
    Color
    0.0 0.5 0.0
}
```