Technical Specification Design (TSD)

Borders

SUMMARY

The game needs to have borders in place to make sure the player is driving only where they should – on the road, within the screen limits.

We will also use the borders to monitor if the player committed traffic violations.

GOALS

We will be able to use the borders to:

* Keep the player from driving beyond the screen limits.
* Monitor if the player is:
  + Switching lanes.
  + Driving over a crosswalk.
  + Entering/exiting a roundabout.
  + Entering/exiting a one-way street.
  + Entering/exiting a parking space.
* Monitor if the player is commiting a traffic violation, such as:
  + Driving past a road border (unto the sidewalk or traffic island).
  + Driving over continuous lane markers.
  + Driving in the wrong direction in a one-way street.
  + Double parking.

Future Goals:

* Count each traffic violation with a specific counter in order to generate a report at the end of the game.
* Enforce traffic signs (stop signs, speed limits).
* Enforce speed bumpers?
* Enforce activating blinkers before switching lanes.

SOLUTION

Creating a Border class with several sub-classes to handle the different kinds of borders we need.

CODE DESIGN

|  |  |
| --- | --- |
| **Border**  **Class** | |
| +init(type, pos) | Initialize the Border object |
|  |  |

|  |  |
| --- | --- |
| **RectBorder**  **Class** | |
| +init(type, pos, width, height) | Initialize the RectBorder object  type = {sidewalk, island, lane} |

|  |  |
| --- | --- |
| **CircleBorder**  **Class** | |
| +init(type, pos, radius) | Initialize the CircleBorder object  type = {sidewalk, island, lane} |

CHANGES IN EXISTING CODE

We would have to change the Player class and add a function to check if the player's car is colliding with any borders, and react accordingly.

For example, the player should not be able to drive beyond the screen limits (hard borders) but would be able to drive over lane markers (soft borders).

TEST PLAN

* Manual tests – make sure:
  + Each border is placed where it should be.
  + The system can recognize when the player is colliding with each of the border.
  + The system counts each collision, and adds it to the corresponding counter.
  + The system enforces each border in the correct way (hard vs. soft).
* See if we can make the code better – more efficient and more lightweight.

WORK

|  |  |
| --- | --- |
| Task | Hours |
| Design and build relevant classes | 1.5 |
| Place all the borders on the screen | 36 |
| Build collision detection | 12 |
| Define appropriate response for each collision type | 2 |
| Test & debug | 8 |
|  |  |
|  |  |