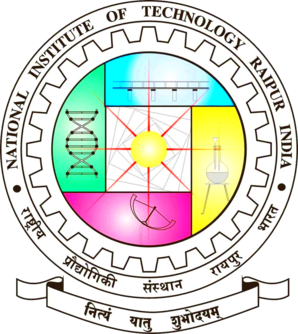
National Institute of Technology, Raipur



Computer Graphics Term Project

Name : Ayush Ganguli

Branch : Information Technology

Semester : 5th

Roll No. : 18118013

Submitted To: Dr. Sanjay Kumar

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Project Name** | **Pages** |
| **1** | **Program to implement digital clock** | **3-6** |
| **2** | **Program to implement a rainbow** | **7-10** |
| **3** | **Program to implement a moving car** | **10-16** |

**INDEX**

**1.Program to Implement a Digital Clock**

* **Tech-Stack Used:** JavaScript, Hyper Text Markup Language(HTML 5), Cascading Style Sheets(CSS 3),setTimeout() inbuilt javaScript Function
* **Aim:** The project aims to implement a digital clock which displays the time in Hours Minutes Seconds XM(AM/PM) 12 hour format along with day of week, day of month, month name , year. Real-time data is shown.
* **Working:**
* The project works on the basis of repeatedly creating a current date time object through setTimeout() function recursively and repeatedly manipulating the Document Object Model(DOM) at set intervals to display current time.
* **setTimout():** It allows us to run a function once after an interval of time.
* Here same function is called recursively after a fixed interval of time.
* **Procedure:**
* A user interface and design is created using HTML and CSS. Styling including font type , font-color, background color, text-alignment is done using CSS.
* As soon as the page loads an onload event showTime is called.
* Show time creates a date time object which contains current date time.
* Time is stored in 24 hours format and it is converted to 12 hr format.
* Day of week name, month name is stored as integer i.e 0 for Sunday 1 for Monday and so on. They are converted into words by mapping every integer index of array to day of week name and month name.
* JavaScript is used for DOM manipulation to change the inner HTML content of time field. Date time is successfully overwritten in the time field.
* After a small interval showTime is called again and the procedure repeats. This process goes on infinitely
* **Code:**
* **HTML Code(index.html)**
* <!DOCTYPE html>
* <html lang="en">
* <head>
* <meta charset="UTF-8">
* <meta name="viewport" content="width=device-width, initial-scale=1.0">
* <title>Digital Clock</title>
* </head>
* <link rel="stylesheet" href="style.css">
* <script src="script.js"></script>
* <body onload="showTime()">
* <div id="date\_time\_container">
* <div class="wrapper">
* <div id="time"></div>
* <div id="date"></div>
* </div>
* </div>
* </body>
* </html>
* **CSS Code(style.css)**
* body
* {
* background-color: #333;
* overflow-x: hidden;
* }
* .wrapper {
* padding:25px;
* max-width:1500px;
* width:100%;
* text-align:center;
* margin:0 auto;
* }
* #time{
* background-color:#333;
* font-family: sans-serif;
* font-size:200px;
* color:#fff;
* }
* #time span {
* color:#888;
* font-size:300px;
* position:relative;
* top:-5px;
* left:10px;
* }
* #date {
* letter-spacing:10px;
* font-size:80px;
* font-family:arial,sans-serif;
* color:#fff;
* }
* **JavaScript Code(script.js)**

**Please Turn Over to next Page**

function addZero(time\_input) {

    if (time\_input < 10) {

        time\_input = "0" + time\_input;

    }

    return time\_input;

}

function showTime() {

    var curr\_time = new Date();

    var hr = curr\_time.getHours();

    var min = curr\_time.getMinutes();

    var sec = curr\_time.getSeconds();

    var suffix\_time;

    if (hr < 12)

    {

        suffix\_time = "<span style='color:yellow;'>AM</span>";

    }

    else

    {

        suffix\_time = "<span style='color:yellow;'>PM</span>";

    }

    if (hr == 0)

    {

        hr = 12;

    }

    if (hr > 12)

    {

        hr -= 12;

    }

    // adding 0 to all time field less than 10

 hr = addZero(hr);

    min = addZero(min);

    sec = addZero(sec);

    document.getElementById("time").innerHTML = hr + ":" + min + ":" + sec + " " + suffix\_time;

    var day\_name\_arr = ['Sun', 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat'];

    var month\_arr = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December'];

    var curWeekDay = day\_name\_arr[curr\_time.getDay()];

    var curDay = curr\_time.getDate();

    var curMonth = month\_arr[curr\_time.getMonth()];

    var curYear = curr\_time.getFullYear();

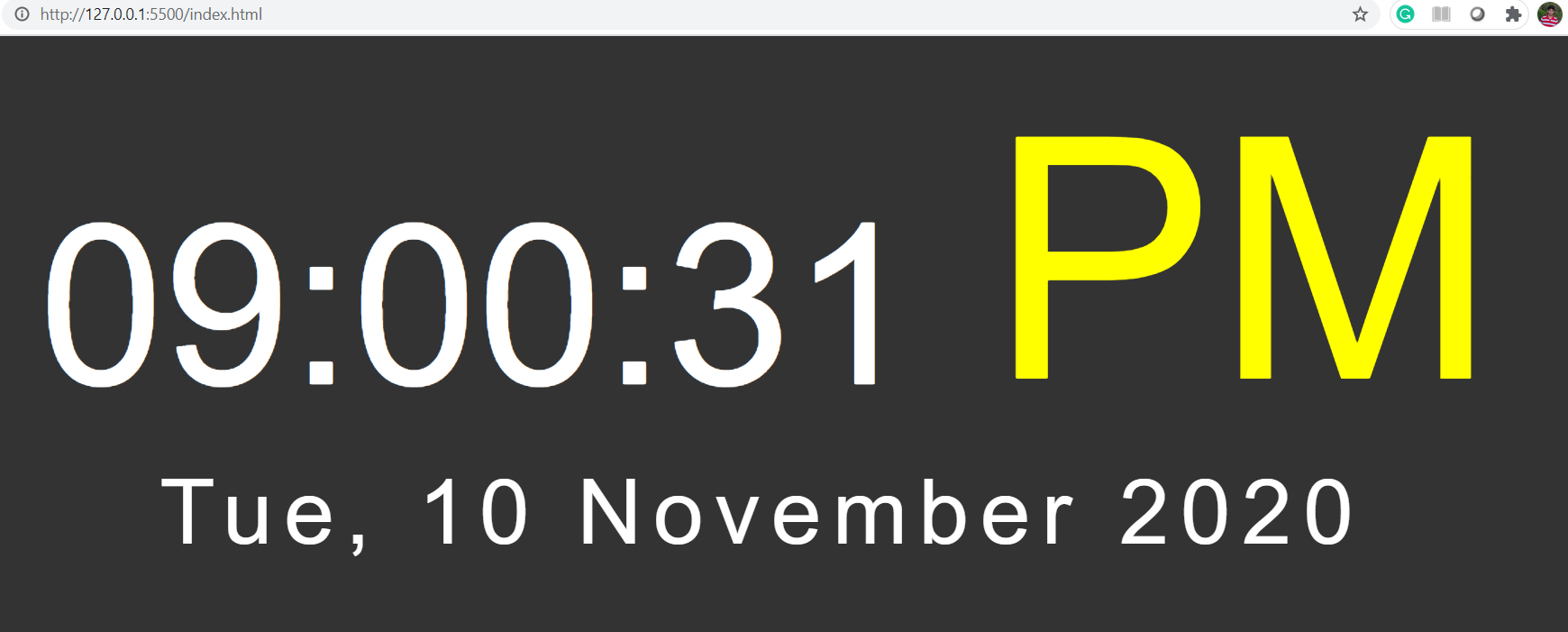
    var date = curWeekDay+", "+curDay+" "+curMonth+" "+curYear;

    document.getElementById("date").innerHTML = date;

    setTimeout(function(){ showTime() }, 500);

}

* **Result:** A digital clock has successfully been created.



**2.Program to implement a rainbow**

* **Tech Stack Used:** Python(3.8.2), Turtle(python standard graphics library)
* **Aim:** The program aims to create rainbow consisting of all 7 colours or a custom rainbow consisting of only specific colours on the basis of user input
* **Working:**

The program uses turtle a python standard graphic library. Turtle is like a drawing board which let us draw all over a screen through a few standard functions.

This program uses turtle to draw concentric circles of different colours. The program gives users a choice to draw a rainbow of all 7 colours or draw a rainbow with specific colours only.

* **How to run the program?**
* Navigate to the directory containing the rainbow\_main.py file using terminal
* Type the following command to start the program(python version above 3.5 required):

python rainbow\_main.py

* Now press 1 to generate a rainbow consisting of all 7 colours.
* Press any key other than 1 to generate a custom rainbow with specific colours with colours as listed by the programme. In the next line enter the colour numbers you want in your rainbow as space separated input.
* A turtle window pops up and a rainbow is generated according to user choice.
* **Code**

**Please turn over for code**

import turtle

import time

def drawSemiCircleFunction(col,rad,val):

    my\_turtle.color(col)

    my\_turtle.circle(rad,-180)

    my\_turtle.up()

    my\_turtle.setpos(val,0)

    my\_turtle.down()

    my\_turtle.right(180)

colours = ['violet','indigo','blue','green','yellow','orange','red']

print("If you want rainbow with all colours press 1 else if you want partial rainbow press any other key")

bool\_val = int(input())

if bool\_val == 1:

    col = colours

else:

    print("You have choosen the option to make a custom rainbow.")

    print("Choose the colours from list below. Enter the colours you want to choose as space seperated input")

    for i in range(len(colours)):

        print("%d -- %s"%(i+1,colours[i]))

    colours\_index = list(map(int,input().strip().split()))

    col = []

    for x in colours\_index:

        col.append(colours[x-1])

my\_turtle = turtle.Turtle()

screen = turtle.Screen()

screen.setup(600,600)

screen.bgcolor('black')

my\_turtle.right(90)

my\_turtle.width(10)

my\_turtle.speed(7)

for i in range(len(col)):

        drawSemiCircleFunction(col[i],10\*(i+8),-10\*(i+1))

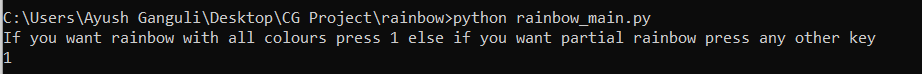
time.sleep(10)

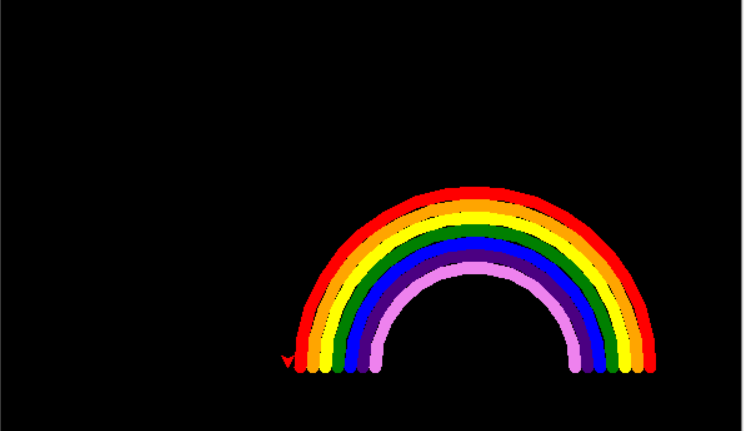
my\_turtle.hideturtle()

**Result:**

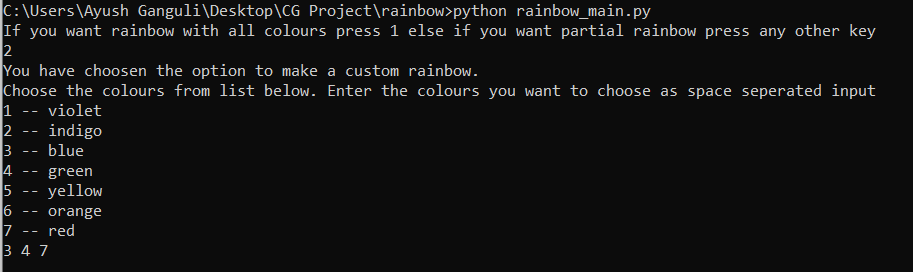
Rainbow with all 7 colours and custom rainbow with specific colours has been created.

**Rainbow with all 7 colours**

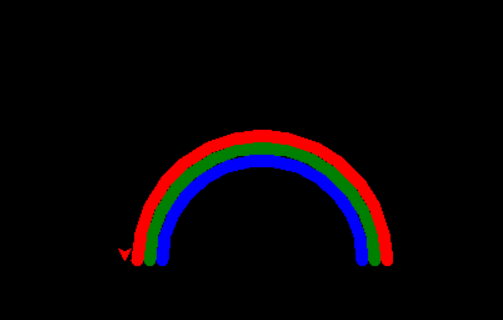




**Blue Green Red Rainbow(Partial Rainbow)**



**Please Turn Over**



**3.Program to Implement Moving Car**

* **Tech Stack Used:** JavaScript, Hyper Text Markup Language(HTML 5), Cascading Style Sheets(CSS 3), CSS animations
* **Aim:** To create a moving whose movement can be controlled with the help of start and stop button
* **Working:**

The car is actually static about it’s position. The road is rotating like a conveyer belt in backward direction and the wheels of the car which are a separate image is rotating in clock-wise direction. This gives the viewer an illusion that the car is actually. Plus engine sound audio controlled through javaScript also contribute towards making this illusion seem real. The car also shakes further adding a better visual effect.

All the animation are done through CSS keyFrames controlled through javaScript. KeyFrames specify animation rule. Animations are created by gradually changing the styling according to keyFrame rules.

* **Procedure:**
* An image of a car was photo shopped. It’s body was separated from it’s 2 wheels.
* In the top part of HTML page scenery along with trees were created.
* In the bottom a road was set as background image.
* The car along with it’s 2 wheels are positioned in the webpage
* The overlapping is done in such a way that car appears on the top layer
* A hidden Boolean text field containing 0 and 1 and a button is added to the page to control the car.
* As soon as the button is clicked onclick event gets activated. fn1() function is called. Since Boolean field is 0, the start car method starts working.
* The track is animated to move backwards and wheels of car is rotated in clockwise direction.
* Car is moved linearly along the y axis to give a shaking effect.
* Engine sound audio starts playing.
* Boolean field is set to 1. As soon as stop button is clicked all the shaking, road moving backward, wheels rotating, audio effect is stopped. Boolean field is again set to 0.
* **Code**

**HTML Code(index.html)**

**Please Turn Over**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>My Moving Car</title>

    <link rel="stylesheet" href="style.css">

    <script src="jquery-3.5.1.min.js"></script>

    <script src="script.js"></script>

</head>

<body>

   <input type="hidden" value="0" id="bool\_check">

    <div class="container">

        <div style="position: absolute ; z-index: 3;"><a href="#" class="button" onclick="fn1(this)" >START CAR</a></div>

        <div id="sky">

            <div id="trees"></div>

            <div id="track"></div>

            <div id="car">

                <div class="wheel wheel1">

                    <img src="car\_wheel\_left.png" id ="wheel1\_img" alt="">

                </div>

                <div class="wheel wheel2">

                    <img src="car\_wheel\_right.png" id ="wheel2\_img" alt="">

                </div>

            </div>

        </div>

    </div>

</body>

</html>

**CSS Code(style.css)**

**Please Turn Over**

\*{

    margin: 0;

    padding: 0;

}

.button {

    background-color: #4CAF50; /\* Green \*/

    border: none;

    color: white;

    padding: 15px 32px;

    text-align: center;

    text-decoration: none;

    display: inline-block;

    font-size: 16px;

  }

body{

    overflow: hidden;

}

#sky{

    height: 100vh;

    width: 100%;

    background-image: url(star\_backgrounjpg.jpg);

    background-repeat: no-repeat;

    position: absolute;

}

#trees{

    height: 100vh;

    width: 100%;

    background-image: url(trees.png);

    background-size: cover;

    position: absolute;

    top: -144px;

}

#track{

    height: 60vh;

    width: 800vw;

    background-image: url(track.png);

    background-repeat: repeat-x;

    position: absolute;

    top: 70vh;

}

#car{

    height: 100px;

    width: 380px;

    background-image: url(car\_body.png);

    background-size: cover;

    background-repeat: no-repeat;

    position: absolute;

    left: 444px;

    bottom:30vh;

}

#wheel1\_img

{

    width: 77px;

    position: relative;

    top: 41.4px;

    left: 41px;

}

#wheel2\_img

{

    width: 77px;

    position: relative;

    top:-38.8px;

    left: 233px;

}

@keyframes rotateWheelFunction

{

    100%{

        transform: rotate(360deg);

    }

}

@keyframes moveCarFunction

{

    100%{

        transform: translateX(-500vw);

    }

}

@keyframes bounceFunction

{

    0%{

        transform: translateY(-5px);

    }

    50%{

        transform: translateY(5px);

    }

    100%{

        transform: translateY(-5px);

    }

}

**JavaScript Code(script.js)**

var engine\_sound;

function fn1(btn)

{

    bool\_check\_div = document.getElementById('bool\_check');

    bool\_check = parseInt(bool\_check\_div.value);

    track = document.getElementById('track');

    car = document.getElementById('car');

    wheel1\_img = document.getElementById('wheel1\_img');

    wheel2\_img = document.getElementById('wheel2\_img');

    if (bool\_check == 0)

    {

        track.style.animation = 'moveCarFunction linear 13s infinite';

        car.style.animation = 'bounceFunction linear .3s infinite';

        wheel1\_img.style.animation = 'rotateWheelFunction linear 0.6s infinite';

        wheel2\_img.style.animation = 'rotateWheelFunction linear 0.6s infinite';

        bool\_check\_div.value = '1';

        btn.innerHTML = 'Stop Car';

        engine\_sound = document.createElement('audio');

        engine\_sound.id = 'engine\_sound\_element';

        engine\_sound.setAttribute('src','engine\_sound.mp3');

        engine\_sound.loop = true;

        engine\_sound.play();

    }

    else

    {

        track.style.animation = 'moveCarFunction linear 0.1s 1s';

        car.style.animation = 'bounceFunction linear .1s 1s';

        wheel1\_img.style.animation = 'rotateWheelFunction linear 0.1s 1s';

        wheel2\_img.style.animation = 'rotateWheelFunction linear 0.1s 1s';

        bool\_check\_div.value = '0';

        btn.innerHTML = 'Start Car';

        engine\_sound.pause();

    }

}

* **Result**

Following are some snaps of the output obtained:

**When car is stationary**



**When car is in motion**



**Please Note:** The animation effects can only be experienced by viewing the project. Please open the index.html file through google chrome of moving\_car\_final\_code folder uploaded to experience the effects.