CRICKET SCORE

IT WORKSHOP 214 PYTHON PROJECT DESCRIPTION

Description/statement:

1. COLLECTED LIST OF LIVE MATCHES.
2. DISPLAYED USING tkinter PACKAGE
3. LEFT WITH NOTIFICATION MECHANISM FOR CERTAIN EVENTS

This application shows the List of live matches

Starting time of matches

Toss details

Team details

Captains of two teams

Score of each player

Total runs

Wickets at which over

Score at each over

Display of single,double,three runs 4,6

Winner of the match

RAMOJI PRASHANTH
2021BCS0092

ICS214 IT WORKSHOP

PYTHON

Course instructor : DR.SELVI C

RAMOJI PRASHANTH

2021BCS0092

CRICKET SCORE PROJECT

```
from tkinter import *
from PIL import ImageTk
from tkinter.ttk import Combobox
import requests
import json
import date_timeฺ
from plyer import notification


# ------ main class  -------- #
class CricketScore:
# ----  gui window  ----  #
    bubble=""
    def __init__(self, root):
        self.root = root
        self.root.title("LIVE CRICKET SCORE")
        width, height = self.root.winfo_screenwidth(),
root.winfo_screenheight()
        self.root.geometry('%dx%d+0+0' % (width,height))
        print(width,height)
        self.bg = ImageTk.PhotoImage(file="design.jpg")
        bg = Label(self.root, image=self.bg, bd=0).place(x=0, y=0)
        # -------   list of live matches <<<<<<< #
        self.label = Label(self.root, text='Live Matches', font=("times new
roman", 50), compound='center').pack(padx=100,    pady=50)
        # ------- adding all live matches combobox  -------#
        self.var = StringVar()
        self.matches_ = self.match_details()
        self.data = [i for i in self.matches_.keys()]
        #print(self.data)
        self.cb = Combobox(self.root, values=self.data, width=50)
```

```
        self.cb.place(x=200,y=200)
        # -------  adding check score button    ------- #
        self.b1 = Button(self.root, text="Check Score", font=("times new
roman", 15),command=self.show_match_details).place(x=550, y=190)


    # -----   command command for check button  ----- #
    def select(self):
        return self.cb.get()

    # ------ scraping from api --------  #
    def scrap(self):
        URL = "https://api.cricapi.com/v1/currentMatches?apikey=aaf4ac23-5d88-
4c58-be3f-edc5849ec307&offset=0"
        page = requests.get(URL)
        #print(json.dumps(page.json(),indent=2))
        return page


    # ------   fetch match details from GUI ------- #
    def match_details(self):
        de = self.scrap().content
        details=de.decode('utf-8')
        #print(details)
        #print(type(details))
        # details=literal_eval(details)
        # print(type(details))
        details=json.loads(details)['data']
        #print(type(details))
        live_match = {}
        for detail in details:
            live_team_details = {}
            summary = self.match_summary(detail)
            start_time = self.match_time(detail)
            teams = self.teams_name(detail)
            score = self.team_score(detail)
            location = self.match_location(detail)
            description = self.match_decription(detail)
            status=self.match_status(detail)
            live_team_details['summary'] = summary
            live_team_details['start_time'] = start_time
            live_team_details['score'] = score
            live_team_details['location'] = location
            live_team_details['status']=status
            live_team_details['description'] = description
            live_match[teams[0] + " VS " + teams[1]] = live_team_details
        return live_match
```

```
    def match_status(self,detail):
        return detail['status']

    # -------- collecting match summary  ------  #
    def match_summary(self, detail):
        #print(detail['name'])
        return detail['name']


    # >>>>> match tiime #
    def match_time(self, detail):
        return detail['dateTimeGMT']


    # -----  getting teams names playing  -------- #
    def teams_name(self, detail):
        teams = detail['teams']
        l = []
        for i in teams:
            l.append(i)
        return l


    # -------  fetching live team score  ------  #
    def team_score(self, detail):
        t_score = detail['score']
        if t_score:
            l=[]
            for i in t_score:
                l.append(i)
            return l
        return 'Match Not Started'


    # ------ fetching match location  ------ #
    def match_location(self, detail):
        return detail['venue']

    # ------ fetching match description  ------- #
    def match_decription(self, detail):
        return detail['matchType']

    def timetostring(self,x):
        x=str(x)
        a,b=x.split("T")
        month,day=date_time.digitalmonth(a)
        return day,month,b

        # >>>>>>  showing details in gui  ------- #
```

```
    def show_match_details(self):
        # >>>>>>  creating match details frame in gui  ------  #
        self.frame1 = Frame(self.root, bg="white")
        self.frame1.place(x=150, y=280, width=700, height=300)
        # >>>>>>  team names ------ #
        Label(self.frame1, text=self.select(), font=("times new roman", 20,
"bold"), bg="cyan", fg="white",bd=0).place(x=160, y=15)
        # >>>>>> showing all details of  each match ------ #
        x = self.matches_[self.select()]
        Label(self.frame1, text="Summary : ", font=("times new roman", 12,
"bold"), bg="white", fg="black",bd=0).place(x=10, y=70)
        Label(self.frame1, text=x['summary'], font=("georgia", 10),
bg="white", fg="black",bd=0).place(x=110, y=70)
        Label(self.frame1, text="Start Time : ", font=("times new roman", 12,
"bold"), bg="white", fg="black",bd=0).place(x=10, y=90)
        d,m,t=self.timetostring(x['start_time'])
        t=t.removesuffix(":00")+","
        Label(self.frame1, text=[t,d,m], font=("serif", 10), bg="white",
fg="black",bd=0).place(x=110, y=90)
        Label(self.frame1, text="Location : ", font=("times new roman", 12,
"bold"), bg="white", fg="black",bd=0).place(x=10, y=110)
        Label(self.frame1, text=x['location'], font=("times new roman", 10),
bg="white", fg="black",bd=0).place(x=110, y=110)
        Label(self.frame1, text="Description : ", font=("times new roman", 12,
"bold"), bg="white", fg="black",bd=0).place(x=10, y=130)
        Label(self.frame1, text=x['description'].upper(), font=("times new
roman", 10), bg="white", fg="black",bd=0).place(x=110, y=130)
        Label(self.frame1, text="Score : ", font=("times new roman", 12,
"bold"), bg="white", fg="black",bd=0).place(x=10, y=170)
        #Label(self.frame1, text=x['score'], font=("times new roman", 15,
"bold"), bg="white", fg="black",bd=0).place(x=20, y=110)
        temp_d=x['score']
        bubble="Match not started yet"
        if x['description'].upper()=="T20" or x['description'].upper()=='ODI':
            if len(temp_d)==2:
                bubble=x['status']
                f=temp_d[0]
                team1=f['inning'].removesuffix("Inning 1")
                team1_runs=f['r']
                team1_wickets=f['w']
                runs_wickets=str(team1_runs)+" / "+str(team1_wickets)
                team1_overs=f['o']
                sample=team1+" : "+runs_wickets+" , "+str(team1_overs)+"
overs"
                bubble+="\nScores:\n"+sample
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=190)
                f=temp_d[1]
```

```
                team2=f['inning'].removesuffix("Inning 1")
                team2_runs=f['r']
                team2_wickets=f['w']
                runs_wickets=str(team2_runs)+" / "+str(team2_wickets)
                team2_overs=f['o']
                sample=team2+" : "+runs_wickets+" , "+str(team2_overs)+"
overs"
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=210)
                Label(self.frame1, text="status :   ".upper()+x['status'],
font=("times new roman", 10,"bold"), bg="white", fg="black",bd=0).place(x=10,
y=240)
                bubble+="\n"+sample
            else:
                f=temp_d[0]
                bubble=x['status']
                team1=f['inning'].removesuffix("Inning 1")
                team1_runs=f['r']
                team1_wickets=f['w']
                runs_wickets=str(team1_runs)+" / "+str(team1_wickets)
                team1_overs=f['o']
                sample=team1+" : "+runs_wickets+" , "+str(team1_overs)+"
overs"
                bubble+="\nSCORES\n"+sample
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=190)

        elif x['description'].upper()=='TEST':
            if len(temp_d)==1:
                f=temp_d[0]
                team1=f['inning'].removesuffix("Inning 1")
                team1_runs=f['r']
                team1_wickets=f['w']
                runs_wickets=str(team1_runs)+" / "+str(team1_wickets)
                team1_overs=f['o']
                sample=team1+" Innings 1 : "+runs_wickets+" ,
"+str(team1_overs)+" overs"
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=190)

                bubble=x['status']+"\nSCORES\n"+sample

            elif len(temp_d)==2:
                f=temp_d[0]
                team1=f['inning'].removesuffix("Inning 1")
                team1_runs=f['r']
                team1_wickets=f['w']
                runs_wickets=str(team1_runs)+" / "+str(team1_wickets)
```

```
                team1_overs=f['o']
                sample=team1+" Innings 1 : "+runs_wickets+" ,
"+str(team1_overs)+" overs"
                bubble=x['status']+"\nScores\n"+sample
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=190)
                f=temp_d[1]
                team2=f['inning'].removesuffix("Inning 1")
                team2_runs=f['r']
                team2_wickets=f['w']
                runs_wickets=str(team2_runs)+" / "+str(team2_wickets)
                team2_overs=f['o']
                sample=team2+"Innings 1 : "+runs_wickets+" ,
"+str(team2_overs)+" overs"
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=210)
                Label(self.frame1, text="status :   ".upper()+x['status'],
font=("times new roman", 10,"bold"), bg="white", fg="black",bd=0).place(x=10,
y=240)
                bubble+="\n"+sample

            elif len(temp_d)==3:
                f=temp_d[0]
                team1=f['inning'].removesuffix("Inning 1")
                team1_runs=f['r']
                team1_wickets=f['w']
                runs_wickets=str(team1_runs)+" / "+str(team1_wickets)
                team1_overs=f['o']
                sample=team1+" Innings 1 : "+runs_wickets+" ,
"+str(team1_overs)+" overs"
                bubble=x['status']+"\nSCORES\n"+sample
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=190)
                f=temp_d[1]
                team2=f['inning'].removesuffix("Inning 1")
                team2_runs=f['r']
                team2_wickets=f['w']
                runs_wickets=str(team2_runs)+" / "+str(team2_wickets)
                team2_overs=f['o']
                sample=team2+"Innings 1 : "+runs_wickets+" ,
"+str(team2_overs)+" overs"
                bubble+="\n"+sample
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=210)
                f=temp_d[2]
                team1=f['inning'].removesuffix("Inning 2")
                team1_runs=f['r']
                team1_wickets=f['w']
```

```
                runs_wickets=str(team1_runs)+" / "+str(team1_wickets)
                team1_overs=f['o']
                sample=team1+"Innings 2 : "+runs_wickets+" ,
"+str(team1_overs)+" overs"
                bubble+="\n"+sample
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=230)
                Label(self.frame1, text="status :   ".upper()+x['status'],
font=("times new roman", 10,"bold"), bg="white", fg="black",bd=0).place(x=10,
y=260)

            elif len(temp_d)==4:
                f=temp_d[0]
                team1=f['inning'].removesuffix("Inning 1")
                team1_runs=f['r']
                team1_wickets=f['w']
                runs_wickets=str(team1_runs)+" / "+str(team1_wickets)
                team1_overs=f['o']
                sample=team1+" Innings 1 : "+runs_wickets+" ,
"+str(team1_overs)+" overs"
                bubble=x['status']+"\nSCORES\n"+sample
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=190)
                f=temp_d[1]
                team2=f['inning'].removesuffix("Inning 1")
                team2_runs=f['r']
                team2_wickets=f['w']
                runs_wickets=str(team2_runs)+" / "+str(team2_wickets)
                team2_overs=f['o']
                sample=team2+"Innings 1 : "+runs_wickets+" ,
"+str(team2_overs)+" overs"
                bubble+="\n"+sample
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=210)
                f=temp_d[2]
                team1=f['inning'].removesuffix("Inning 2")
                team1_runs=f['r']
                team1_wickets=f['w']
                runs_wickets=str(team1_runs)+" / "+str(team1_wickets)
                team1_overs=f['o']
                sample=team1+"Innings 2 : "+runs_wickets+" ,
"+str(team1_overs)+" overs"
                bubble+="\n"+sample
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=230)
                f=temp_d[3]
                team2=f['inning'].removesuffix("Inning 2")
                team2_runs=f['r']
```

```
                team2_wickets=f['w']
                runs_wickets=str(team2_runs)+" / "+str(team2_wickets)
                team2_overs=f['o']
                sample=team1+"Innings 2 : "+runs_wickets+" ,
"+str(team2_overs)+" overs"
                bubble+="\n"+sample
                Label(self.frame1, text=sample, font=("george", 10,),
bg="white", fg="green",bd=0).place(x=10, y=250)
                Label(self.frame1, text="status :   ".upper()+x['status'],
font=("times new roman", 10,"bold"), bg="white", fg="black",bd=0).place(x=10,
y=280)
        #print(bubble)
        self.notifying(str(bubble))

    def notifying(self,s,t="match status",a="python"):
        notification.notify(
            title = t,
            message = s,
            app_icon = None,
            ticker="cricket",
            toast = True,
            )


# >>>>  main function
def main():
    # >>>>>  create window using tkinter
    root = Tk()
    obj = CricketScore(root)
    # >>>>>  start the gui
    root.mainloop()


if __name__ == "__main__":
    main()
```