Oxford International College
Computer Science

Clock DisplayApp

Writing a program to display a working analogue clock

1. Overview

The Clock Display App is a program with a GUI that displays a working clock.

This document describes the design of a Clock Display App to a level of detail that allows the informed reader to write the code for the application.

The clock display is an analogue clock with hour, minute and second hands. The clock is accompanied by a digital time display of hours, minutes and seconds and a separate label for its location e.g. “Oxford”. The program displays the clock in a fixed size window.

The GUI offers options for About and Exit. The About option will display the version number of the App and author credits. The exit option will close the display window and exit the program.

2. Main() routine

The main() routine performs the following functions:

- Create the basic GUI infrastructure (App title, menu structure, routines to service the menu items)
- Define the default configuration data for each clock
- Create a clock object for the clock to be displayed
- Call the GUI main loop

3. Clock Class

The clock display is implemented by a class that defines all of the data and methods required to display a working clock.

Method	Description
__init__()	The constructor method Creates the position tables used to define the positions of the hands around the clock face. Displays the clock face. Makes the first update of the clock so that the hands and digital display appear.
makePosTable()	Creates a table of sin and cos values for 60 positions around the clock face. The hand positions are calculated using the data in this table.
showFace()	Draws the clock face. The face is static and once drawn does not need to be refreshed. The face consists of a circle with suitable marks for the 12 positions around the clock face. The location of the clock is shown by a text label displayed underneath the clock face.
updateHandler()	This routine is called at intervals by the GUI main loop and updates the dynamic parts of the clock display i.e. the position of the hands and the content of the digital time display. The last thing the routine does it to schedule itself to be called again after a delay of typically 1 sec.
firstUpdate()	This routine displays the dynamic parts of the clock display for the first time and then schedules the updateHandler() method to be called after a short interval. Unlike many updates the first update must refresh the display of all three hands which is why it has its own dedicated routine.
updateTimeDisplay()	Updates the digital time display
updateSecHand()	Erases and redraws the second hand at a position appropriate for the current time.
updateMinHand()	Erases and redraws the minute hand at a position appropriate for the current time.
updateHourHand()	Erases and redraws the hour hand at a position appropriate for the current time.
getTimeValues()	Get the correct hours, min and sec values for the location of the clock. This is done by getting the machine's local time, converting that to UTC and then adding on the UTC offset for the displayed clock's location.

4. Configuration Data

The configuration of a clock is defined by a list of four items:

Config item	Description
Diameter	Clock diameter in pixels
CentreX	X coordinate of the clock centre
CentreY	Y coordinate of the clock centre
LocationStr	A text string describing the location of the clock

This information can be stored in the program as a 1D list:

```
# diameter, centreX, centreY, locationStr,
configData=[200,125,150,"Oxford"]
```

5. Implementation hints

5.1 The GUI and clock display

The GUI can be based on the program *GUI-demo.py*.
The code for drawing the clock can be based on the program *CanvasExtended.py*. In particular the clock display is redrawn at short intervals of time using the *root.after()* function from Tkinter.

5.2 Drawing the clock face and hands

The clock App should display a circular clock. Start by looking at a clock and noting the details of the clock face and the hands.

It is easiest to use a co-ordinate system in which the angle is measured from the vertical (in contrast with the usual method of measuring from the horizontal). The position of a point at radius *r* from the clock centre is then given by:

$x = r \sin \theta$ and $y = r \cos \theta$ with $\theta = 0$ at the 12 o'clock position.

Again to make things easier, at the start of the program we can calculate a table of cos and sin values at 60 different positions around the clock face. We then use these values to calculate the position of the marks on the clock face. The same table is used to calculate the position of the hands as the time changes.

The clock face consists of three fixed elements:

- The perimeter (a circle)
- The minute markings
- The number labels 1-12

and three moving elements:

- the second hand
- the minute hand
- the hour hand

Once the hands are drawn in their initial position, they must be updated as time passes.

5.3 Updating the hands
The hour minute and second hands move at different speeds and so need to be updated at different intervals of time.
The second hand completes one revolution in 60 sec.
The minute hand completes one revolution in 1 hour
The hour hand completes one revolution in 12 hours.

If there are 60 positions around the clock face, then:

- The second hand must be updated every second.
- The minute hand must be updated every minute.
- The hour hand must be updated every 12 minutes.

Defining 60 positions around the clock face is a great place to start from, more positions can be used later if you wish to smooth out the motion of the second hand.

To update the position of a hand we must first delete the existing hand and then redraw it in a new position. The easy way to do this is to give the hand a tag name when we draw it. When we update the hand we first delete it by tag name and then draw the hand again in the new position. You can see this technique used in the CanvasExtended.py demonstration software where the rectangle is deleted by tag name and then redrawn in a different colour.

5.3 Time calculations

To keep things simple the time displayed by the clock is simply the local time on the computer running the program.

```
def getTimeValues(self):
    # get local time on this machine in the form of hours, minutes and seconds.
    # localtime is a data structure with members tm_hour   tm_min tm_sec (and more...)
    localtime = time.localtime(time.time())
    return localtime.tm_hour, localtime.tm_min, localtime.tm_sec
```

5.4 Imported libraries

You will need code for the following libraries:
tkinter, math, time, datetime

5.5 Configuration file
The configuration file is a text file that uses tag=value syntax to define the key configuration items for the clock display.

A sample file for a 2 clock display is shown below.
Notice that the clock diameter is the same for all clocks but each clock has its own location, UTC offset and DST offset items.

The file is called ClockApp.cfg.

```
# config file for Clock App
# Display configuration
```

 # clock diameter in pixels, suggested range 150-300
diameter=300
 # Configuration for individual clocks
[clock=1]
location = London
utcOffset = 0
dstOffset = +1
[clock=2]
location = Astana
utcOffset = +6
dstOffset = 0