Python Programming

Module 8 - Date & Time

High performance. Delivered.

accenture

Strategy | Digital | Technology | Operations

Agenda

- Date & Time in Python
- What is Tick?
- TimeTuple
- time module in Python
- calendar module in Python

Copyright © 2016 Accenture All rights reserved.

Module Objectives

At the end of this module, you will be able to understand

- Date & Time in Python
- time module in Python and related functions
- calendar module in Python and related functions

Copyright © 2016 Accenture All rights reserved.

Date & Time in Python

- A Python program can handle date and time in several ways.
- Python's time and calendar modules help manage dates and times easily and efficiently.

Copyright © 2016 Accenture All rights reserved.

What is Tick?

- Time intervals are floating-point numbers in units of seconds.
- Particular instants in time are expressed in seconds since 12:00am, January 1, 1970.
- There is a **time** module available in Python which provides functions for working with times, and for converting between representations.
- The function *time.time()* returns the current system time in ticks since 12:00am, January 1, 1970(epoch)

Note:

***Epoch time** is a system for describing instants in **time**, defined as the number of seconds that have elapsed since 00:00:00 Coordinated Universal **Time** (UTC), Thursday, 1 January 1970, not counting leap seconds. Dates before the epoch cannot be represented in this form. Future Dates also cannot be represented this way - the cutoff point is sometime in 2038 for UNIX and Windows.*

Copyright © 2016 Accenture All rights reserved.

Tick - Example

Example:

```
import time;
ticks = time.time();
print ("Number of ticks since 12:00am, January 1, 1970:", ticks);
```

<u>OUTPUT</u>

```
Number of ticks since 12:00am, January 1, 1970:
1456820428.027663.
```

Copyright © 2016 Accenture All rights reserved.

TimeTuple

Many of Python's time functions handle time as a **tuple of 9 numbers**, as shown below

Index	Field	Values
0	4-digit year	2008
1	Month	1 to 12
2	Day	1 to 31
3	Hour	0 to 23
4	Minute	0 to 59
5	Second	0 to 61 (60 or 61 are leap-seconds)
6	Day of Week	0 to 6 (0 is Monday)
7	Day of year	1 to 366 (Julian day)
8	Daylight savings	-1, 0, 1, -1 means library determines DST

Copyright © 2016 Accenture All rights reserved.

struct_time structure

A tuple is equivalent to **struct_time** structure.

This structure has following attributes −

Index	Attributes	Values
0	tm_year	2008
1	tm_mon	1 to 12
2	tm_mday	1 to 31
3	tm_hour	0 to 23
4	tm_min	0 to 59
5	tm_sec	0 to 61 (60 or 61 are leap-seconds)
6	tm_wday	0 to 6 (0 is Monday)
7	tm_yday	1 to 366 (Julian day)
8	tm_isdst	-1, 0, 1, -1 means library determines DST

Copyright © 2016 Accenture All rights reserved.

Getting current time

To translate a time instant from a *seconds since the epoch* floating-point value into a time-tuple, pass the floating-point value to a function (e.g., localtime) that returns a time-tuple with all nine items valid.

```
import time;
localtime = time.localtime(time.time());
print ("Local current time :", localtime);
```

OUTPUT

```
Local current time : time.struct_time(tm_year=2016, tm_mon=3,
tm_mday=1, tm_hour=14, tm_min=8, tm_sec=5, tm_wday=1,
tm_yday=61, tm_isdst=0)
```

Copyright © 2016 Accenture All rights reserved.

Formatting time - asctime() function

- Time can be formatted as per the requirement,
- One of the methods to get time in readable format is asctime() function

```
import time;
localtime = time.asctime( time.localtime(time.time()) );
print ("Local current time :", localtime);
```

OUTPUT

```
Local current time : Tue Mar  1 14:12:09 2016
```

Copyright © 2016 Accenture All rights reserved.

calendar for a month

- ➢ The calendar module gives a wide range of methods to play with yearly and monthly calendars.

Example:

```
import calendar;
cal = calendar.month(2016, 3);
print("Here is the calendar:");
print(cal);
```

OUTPUT

```
Here is the calendar:
     March 2016
Mo Tu We Th Fr Sa Su
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31
```

Copyright © 2016 Accenture All rights reserved.

The *time* Module - functions

- Following are the functions in python **time** module for working with times and for converting between representations.

Function	Description
time.altzone	The offset of the local DST timezone, in seconds west of UTC, if one is defined. This is negative if the local DST timezone is east of UTC (as in Western Europe, including the UK). Only use this if daylight is nonzero.
time.asctime([tupletime])	Accepts a time-tuple and returns a readable 24-character string such as 'Tue Dec 11 18:07:14 2008'.
time.clock( )	Returns the current CPU time as a floating-point number of seconds. To measure computational costs of different approaches, the value of time.clock is more useful than that of time.time().
time.ctime([secs])	Like asctime(localtime(secs)) and without arguments is like asctime( )
time.gmtime([secs])	Accepts an instant expressed in seconds since the epoch and returns a time-tuple t with the UTC time. Note : t.tm_isdst is always 0
time.localtime([secs])	Accepts an instant expressed in seconds since the epoch and returns a time-tuple t with the local time (t.tm_isdst is 0 or 1, depending on whether DST applies to instant secs by local rules).
time.mktime(tupletime)	Accepts an instant expressed as a time-tuple in local time and returns a floating-point value with the instant expressed in seconds since the epoch.

Copyright © 2016 Accenture All rights reserved.

The *time* Module – functions (Contd)

Function	Description
time.sleep(secs)	Suspends the calling thread for secs seconds.
time.strftime(fmt[,tupletime])	Accepts an instant expressed as a time-tuple in local time and returns a string representing the instant as specified by string fmt.
time.strptime(str,fmt='%a %b %d %H:%M:%S %Y')	Parses str according to format string fmt and returns the instant in time-tuple format.
time.time( )	Returns the current time instant, a floating-point number of seconds since the epoch.
time.tzset()	Resets the time conversion rules used by the library routines. The environment variable TZ specifies how this is done.

Copyright © 2016 Accenture All rights reserved.

Attributes in time module

Following are the two important attributes available with time module.

Attribute	Description
time.timezone	Attribute time.timezone is the offset in seconds of the local time zone (without DST) from UTC (>0 in the Americas; <=0 in most of Europe, Asia, Africa).
time.tzname	Attribute time.tzname is a pair of locale-dependent strings, which are the names of the local time zone without and with DST, respectively.

Copyright © 2016 Accenture All rights reserved.

The *calendar* Module

- The calendar module supplies calendar-related functions,
- It includes functions to print a text calendar for a given month or year.
- By default, calendar takes Monday as the first day of the week and Sunday as the last one.
- To change this, call calendar.setfirstweekday() function.

Copyright © 2016 Accenture All rights reserved.

calendar Module - Functions

Function	**Description**
calendar.calendar(year,w=2,l=1,c=6)	Returns a multiline string with a calendar for year year formatted into three columns separated by c spaces. w is the width in characters of each date; each line has length 21*w+18+2*c. l is the number of lines for each week.
calendar.firstweekday( )	Returns the current setting for the weekday that starts each week. By default, when calendar is first imported, this is 0, meaning Monday.
calendar.isleap(year)	Returns True if year is a leap year; otherwise, False.
calendar.leapdays(y1,y2)	Returns the total number of leap days in the years within range(y1,y2).
calendar.month(year,month,w=2,l=1)	Returns a multiline string with a calendar for month month of year year, one line per week plus two header lines. w is the width in characters of each date; each line has length 7*w+6. l is the number of lines for each week.
calendar.monthcalendar(year,month)	Returns a list of lists of ints. Each sublist denotes a week. Days outside month month of year year are set to 0; days within the month are set to their day-of-month, 1 and up.

Copyright © 2016 Accenture All rights reserved.

calendar Module – Functions (Contd)

Function	Description
calendar.monthrange(year,month)	Returns two integers. The first one is the code of the weekday for the first day of the month month in year year; the second one is the number of days in the month. Weekday codes are 0 (Monday) to 6 (Sunday); month numbers are 1 to 12.
calendar.prcal(year,w=2,l=1,c=6)	Like print calendar.calendar(year,w,l,c).
calendar.prmonth(year,month,w=2,l=1)	Like print calendar.month(year,month,w,l).
calendar.setfirstweekday(weekday)	Sets the first day of each week to weekday code weekday. Weekday codes are 0 (Monday) to 6 (Sunday).
calendar.timegm(tupletime)	The inverse of time.gmtime: accepts a time instant in time-tuple form and returns the same instant as a floating-point number of seconds since the epoch.
calendar.weekday(year,month,day)	Returns the weekday code for the given date. Weekday codes are 0 (Monday) to 6 (Sunday); month numbers are 1 (January) to 12 (December).

Copyright © 2016 Accenture All rights reserved.

Questions