eGenix.com

mxStack

Fast Stack Data-Type
for Python

Version 3.2

Printed in Germany.

Contents

1. Introduction

Even though stacks can be emulated with Python lists, this type provides a simple interface to the data structure, both in Python and in C.

1.1 Performance Comparison

Because of the function call overhead calling the methods from Python it is only a tad faster than a corresponding list emulation. Called from within an C extension shows a more significant performance increase.

The included `stackbench.py` gives an impression of how the different methods relate with respect to speed:

```
mx/Stack> python stackbench.py 1000 1000 100
list:  0.29
tuples: 0.31
Stack (with push + pop): 0.3
Stack (with push + pop_many): 0.3
Stack (with << + >>): 0.29
Stack (with push_many + pop_many): 0.27
UserStack: 0.79
```

Note that the tuple version has a few disadvantages when used for big stacks: for one it uses lots of memory (20 bytes per entry slot; Stack uses 20 bytes + 4 bytes per entry slot) and deallocation can become a problem -- this is done using recursion with one level per stack element. For small stacks it still is unbeatable, though (it has no function call overhead).

The `UserStack` implementation shown in the above output and included in the mxStack package uses the same technique: the figures shown mainly result from Python method call overhead added by the method interface.

1.2 Memory Management

Because stacks are normally used only temporarily, the Stack implementation only grows the memory buffer used for holding the entry slots. It never shrinks it. This has an advantage of reducing `malloc()` overhead when doing e.g. depth first search, but also the disadvantage of

using more memory in degenerate cases. To compensate for this, simply call the `.resize()` method every now and then. It forces the used buffer to be resized.

2. mx.Stack.Stack Object

The Stack object provides the following interfaces.

2.1 Stack Object Constructors

There are two ways to construct a `Stack` from scratch:

`Stack([initial_size])`

Returns a new empty Stack instance allocating at least the given number of slots for stack elements. If the parameter is not given a reasonable default is chosen.

`StackFromSequence(seq)`

Constructs a Stack instance from the given sequence. The instance is filled with all the elements found in the sequence by pushing the items from index 0 to len(seq)-1 in that order, i.e. popping all elements from the Stack results in a reversed sequence.

2.2 Stack Object Instance Methods

A `Stack` instance has the following methods:

`.as_tuple()`

Returns the stack's content as tuple, without modifying it.

`.as_list()`

Returns the stack's content as list, without modifying it.

`.clear()`

Clears the stack.

`.pop()`

Pops the top element off of the stack.

`.pop_many(n)`

Pops the top `n` elements and returns them in form of a tuple. If less than `n` elements are on the stack, the tuple will contain all stack entries and the stack will then be empty again. The order is top to bottom, i.e. `s.pop_many(2) == (s.pop(),s.pop())`

`.push(x)`

Pushes the object `x` onto the stack.

`.push_many(sequences)`

Pushes the objects in `sequence` from left to right onto the stack. If errors occur during this process, the already pushed elements are discarded from the stack and it returns to its original state.

`.resize([size=len(stack)])`

Resize the stack buffer to hold at least `size` entries.

You can call this method without argument to force the stack to shrink its memory buffer to the minimal limit needed to hold the contained elements.

`.__getitem__(index)`

This is not really a method, but a slot providing access to the items on the Stack without popping them off the Stack.

`index` works just like for Python lists, i.e. negative indices are normalized using the current length of the Stack.

An `IndexError` is raised for invalid indices. This makes the Stack compatible to the `for`-loop statement allowing you to iterate over the Stack contents from bottom to top.

Note that no method for testing emptiness is provided. Use `len()` for that or simply test for trueness, e.g. `while s: print s.pop()` will loop as long as there are elements left on the Stack s. This is much faster than going through the method calling process -- even when the method being called is written in C.

3. mx.Stack Constants

`Error`

Error class used for package specific errors. It is a subclass of `IndexError`.

`EmptyError`

Error class used to signal an empty queue. It is a subclass of `Error`.

4. mx.Stack Python C-API

mxStack exposes a Python C-API which can easily be used by other Python extensions. Please have look at the file `mxStack.h` for details.

Most of the above Python interfaces are also available in the C API.

To access the module, do the following (note the similarities with Python's way of accessing functions from a module):

```
#include "mxStack.h"

...
    PyObject *v;

    /* Import the mxStack module */
    if (mxStack_ImportModuleAndAPI())
   goto onError;

    /* Access functions from the exported C API through mxStack */
    v = mxStack.Stack(0);
    if (!v)
   goto onError;

    /* Type checking */
    if (mxStack_Check(v))
        printf("Works.\n");

    Py_DECREF(v);
...
```

5. Examples of Use

Well, there's not much to show:

```
from mx.Stack import *
s = Stack()
for i in range(1000):
    s.push(i)
while s:
    print s.pop()
# which could also be done as:
s = StackFromSequence(range(1000))
while s:
    print s.pop()
# or a little different
s = StackFromSequence(range(1000))
print s.as_tuple()
print s.as_list()
```

6. Package Structure

```
[Stack]
   Doc/
   [mxStack]
      test.py
   UserStack.py
   stackbench.py
```

Entries enclosed in brackets are packages (i.e. they are directories that include a `__init__.py` file). Ones without brackets are just simple subdirectories that are not accessible via `import`. These are used for compiling the C extension modules which will get installed in the same place where all your other site specific extensions live (e.g. `/usr/local/lib/python-x.xx/site-packages`).

The package imports all symbols from the Proxy sub module which in turn imports the extension module, so you only need to '`from mx import Stack`' to start working.

7. Support

eGenix.com is providing commercial support for this package. If you are interested in receiving information about this service please see the *eGenix.com Support Conditions*.

8. Copyright & License

EGENIX.COM PUBLIC LICENSE AGREEMENT

Version 1.1.0

This license agreement is based on the Python CNRI License Agreement, a widely accepted open-source license.

1. Introduction

This "License Agreement" is between eGenix.com Software, Skills and Services GmbH ("eGenix.com"), having an office at Pastor-Loeh-Str. 48, D-40764 Langenfeld, Germany, and the Individual or Organization ("Licensee") accessing and otherwise using this software in source or binary form and its associated documentation ("the Software").

2. License

Subject to the terms and conditions of this eGenix.com Public License Agreement, eGenix.com hereby grants Licensee a non-exclusive, royalty-free, world-wide license to reproduce, analyze, test, perform and/or display publicly, prepare derivative works, distribute, and otherwise use the Software alone or in any derivative version, provided, however, that the eGenix.com Public License Agreement is retained in the Software, or in any derivative version of the Software prepared by Licensee.

3. NO WARRANTY

eGenix.com is making the Software available to Licensee on an "AS IS" basis. SUBJECT TO ANY STATUTORY WARRANTIES WHICH CAN NOT BE EXCLUDED, EGENIX.COM MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, EGENIX.COM MAKES NO AND DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF THE SOFTWARE WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

4. LIMITATION OF LIABILITY

EGENIX.COM SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF THE SOFTWARE FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) AS A RESULT OF USING, MODIFYING OR

DISTRIBUTING THE SOFTWARE, OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE EXCLUSION OR LIMITATION MAY NOT APPLY TO LICENSEE.

5. Termination

This License Agreement will automatically terminate upon a material breach of its terms and conditions.

6. Third Party Rights

Any software or documentation in source or binary form provided along with the Software that is associated with a separate license agreement is licensed to Licensee under the terms of that license agreement. This License Agreement does not apply to those portions of the Software. Copies of the third party licenses are included in the Software Distribution.

7. General

Nothing in this License Agreement affects any statutory rights of consumers that cannot be waived or limited by contract.

Nothing in this License Agreement shall be deemed to create any relationship of agency, partnership, or joint venture between eGenix.com and Licensee.

If any provision of this License Agreement shall be unlawful, void, or for any reason unenforceable, such provision shall be modified to the extent necessary to render it enforceable without losing its intent, or, if no such modification is possible, be severed from this License Agreement and shall not affect the validity and enforceability of the remaining provisions of this License Agreement.

This License Agreement shall be governed by and interpreted in all respects by the law of Germany, excluding conflict of law provisions. It shall not be governed by the United Nations Convention on Contracts for International Sale of Goods.

This License Agreement does not grant permission to use eGenix.com trademarks or trade names in a trademark sense to endorse or promote products or services of Licensee, or any third party.

The controlling language of this License Agreement is English. If Licensee has received a translation into another language, it has been provided for Licensee's convenience only.

8. Agreement

By downloading, copying, installing or otherwise using the Software, Licensee agrees to be bound by the terms and conditions of this License Agreement.

For question regarding this License Agreement, please write to:

eGenix.com Software, Skills and Services GmbH

Pastor-Loeh-Str. 48

D-40764 Langenfeld

Germany