

# Using a Simple Vector Class

Copyright (c) 2019 Tor Olav Kristensen, <http://subcube.com> (<http://subcube.com>)

<https://github.com/t-o-k/scikit-vectors> (<https://github.com/t-o-k/scikit-vectors>)

Use of this source code is governed by a BSD-license that can be found in the LICENSE file.

```
In [1]: 1 from skvectors import create_class_Simple_Vector
```

```
In [2]: 1 # Create a 3-dimensional simple vector class
2
3 # The first argument is a string with the name of the class
4 # to be created.
5
6 # The number of elements in the iterable given as the second
7 # argument determines the number of dimensions for the class.
8
9 SVC = create_class_Simple_Vector('VC', 'IJK')
10
11 # Explicit alternative:
12 # SVC = \
13 #     create_class_Simple_Vector(
14 #         name = 'SVC',
15 #         component_names = [ 'I', 'J', 'K' ],
16 #         brackets = [ '<', '>' ],
17 #         sep = ', '
18 #     )
```

```
In [3]: 1 # Create a vector by applying abs to the I-component of a vector
2 u = SVC(-2, 3, -4)
3 u.c_abs_I()
```

```
Out[3]: VC(I=2, J=3, K=-4)
```

```
In [4]: 1 # Create a vector by applying unary minus to the K-component of a vector
        2 u = SVC(2, 3, 4)
        3 u.c_neg_K()
```

Out[4]: VC(I=2, J=3, K=-4)

```
In [5]: 1 # Create a vector by applying unary minus to all the components of a vector except the K-component
        2 u = SVC(2, 3, 4)
        3 u.c_neg_bar_K()
```

Out[5]: VC(I=-2, J=-3, K=4)

```
In [6]: 1 # Create a vector by applying unary plus to the J-component and the K-component of a vector
        2 u = SVC(2, 3, 4)
        3 u.c_pos_J_K()
```

Out[6]: VC(I=2, J=3, K=4)

```
In [7]: 1 # Create a vector by adding 100 to the K-component of a vector
        2 u = SVC(2, 3, 4)
        3 u.c_add_K(100)
```

Out[7]: VC(I=2, J=3, K=104)

```
In [8]: 1 # In-place addition of 100 to the K-component of a vector
        2 u = SVC(2, 3, 4)
        3 u.c_iadd_K(100)
        4 u
```

Out[8]: VC(I=2, J=3, K=104)

```
In [9]: 1 # Create a vector by subtracting 3 from the J-component of a vector
        2 u = SVC(2, 3, 4)
        3 u.c_sub_J(3)
```

Out[9]: VC(I=2, J=0, K=4)

```
In [10]: 1 # In-place subtraction of 3 from the J-component of a vector
         2 u = SVC(2, 3, 4)
         3 u.c_isub_J(3)
         4 u
```

Out[10]: VC(I=2, J=0, K=4)

```
In [11]: 1 # Create a vector by multiplying all the components of a vector except none by 8
         2 u = SVC(2, 3, 4)
         3 u.c_mul_bar(8)
```

Out[11]: VC(I=16, J=24, K=32)

```
In [12]: 1 # In-place multiplication of all the components of a vector except none by 8
         2 u = SVC(2, 3, 4)
         3 u.c_imul_bar(8)
         4 u
```

Out[12]: VC(I=16, J=24, K=32)

```
In [13]: 1 # Create a vector by raising the I-component of a vector to the power of 10
         2 u = SVC(2, 3, 4)
         3 u.c_pow_I(10)
```

Out[13]: VC(I=1024, J=3, K=4)

```
In [14]: 1 # In-place raising the I-component of a vector to the power of 10
         2 u = SVC(2, 3, 4)
         3 u.c_ipow_I(10)
         4 u
```

Out[14]: VC(I=1024, J=3, K=4)

```
In [15]: 1 # Create a vector by true dividing none of the components of a vector by 0
         2 u = SVC(2, 3, 4)
         3 u.c_truediv(0)
```

Out[15]: VC(I=2, J=3, K=4)

```
In [16]: 1 # In-place true division of all the components of a vector by 10
         2 u = SVC(2, 3, 4)
         3 u.c_itruediv_bar(10)
         4 u
```

Out[16]: VC(I=0.2, J=0.3, K=0.4)

```
In [17]: 1 # Create a vector by floor dividing all the components of a vector by 2
         2 u = SVC(2, 3, 4)
         3 u.c_floordiv_I_J_K(2)
```

Out[17]: VC(I=1, J=1, K=2)

```
In [18]: 1 # In-place floor division of all the components of a vector by 2
         2 u = SVC(2, 3, 4)
         3 u.c_ifloordiv_I_J_K(2)
         4 u
```

Out[18]: VC(I=1, J=1, K=2)

```
In [19]: 1 # Create a vector by applying modulus to all the components of a vector and 2
         2 u = SVC(2, 3, 4)
         3 u.c_mod_I_J_K(2)
```

Out[19]: VC(I=0, J=1, K=0)

```
In [20]: 1 # In-place application of modulus to all the components of a vector and 2
         2 u = SVC(2, 3, 4)
         3 u.c_imod_I_J_K(2)
         4 u
```

Out[20]: VC(I=0, J=1, K=0)

```
In [21]: 1 # Create a vector by multiplying the K-component of a vector by 100
         2 u = SVC(2, 4, 6)
         3 u.c_mul_K(100)
```

Out[21]: VC(I=2, J=4, K=600)

```
In [22]: 1 # In-place multiplication of the K-component of a vector by 100
         2 u = SVC(2, 4, 6)
         3 u.c_imul_K(100)
         4 u
```

Out[22]: VC(I=2, J=4, K=600)

```
In [23]: 1 # Create a vector by applying several operations to the components of vectors
         2 u = SVC(2, 3, 4)
         3 f = u.c_mul_K
         4 f(10).c_add_bar(88).c_mul_I_J(88).c_sub_bar_J_K(100000).c_neg_K()
```

Out[23]: VC(I=-92080, J=8008, K=-128)

```
In [24]: 1 # Create a vector by rounding the components of a vector to 3 decimals
         2 u = SVC(2.22222, 4.44444, 6.66666)
         3 round(u, ndigits=3)
```

Out[24]: VC(I=2.222, J=4.444, K=6.667)

```
In [25]: 1 # Create a vector by rounding the components of a vector to integer value
         2 u = SVC(2.222, 4.444, 6.666)
         3 round(u)
```

Out[25]: VC(I=2.0, J=4.0, K=7.0)

```
In [26]: 1 # Create a vector by rounding the components of a vector
         2 u = SVC(-55555555.5, -33333333.3, 55555555.5)
         3 round(u, -4)
```

Out[26]: VC(I=-55560000.0, J=-33330000.0, K=55560000.0)

```
In [27]: 1 # Create a vector by applying unary minus to a vector
         2 u = SVC(-3, 4, 5)
         3 -u
```

Out[27]: VC(I=3, J=-4, K=-5)

```
In [28]: 1 # Create a vector by applying unary plus to a vector
         2 u = SVC(-3, 4, 5)
         3 +u
```

Out[28]: VC(I=-3, J=4, K=5)

```
In [29]: 1 # Create a vector by adding a vector to another
         2 u = SVC(-3, 4, 5)
         3 v = SVC(1, 1, -1)
         4 u + v
```

Out[29]: VC(I=-2, J=5, K=4)

```
In [30]: 1 # In-place addition of a vector to another
         2 u = SVC(-3, 4, 5)
         3 v = SVC(1, 1, -1)
         4 u += v
         5 u
```

Out[30]: VC(I=-2, J=5, K=4)

```
In [31]: 1 # Create a vector by subtracting a vector from another
         2 u = SVC(-3, 4, 5)
         3 v = SVC(1, 1, -1)
         4 u - v
```

Out[31]: VC(I=-4, J=3, K=6)

```
In [32]: 1 # In-place subtraction of a vector from another
         2 u = SVC(-3, 4, 5)
         3 v = SVC(1, 1, -1)
         4 u -= v
         5 u
```

Out[32]: VC(I=-4, J=3, K=6)

```
In [33]: 1 # Create a vector by multiplying a vector by another
         2 u = SVC(-1, 2, 3)
         3 v = SVC(2, 0, -2)
         4 u * v
```

Out[33]: VC(I=-2, J=0, K=-6)

```
In [34]: 1 # In-place multiplication of a vector by another
        2 u = SVC(-1, 2, 3)
        3 v = SVC(2, 0, -2)
        4 u *= v
        5 u
```

Out[34]: VC(I=-2, J=0, K=-6)

```
In [35]: 1 # Create a vector by multiplying a vector and a scalar
        2 u = SVC(-1, 2, 3)
        3 s = 2
        4 s * u, u * s
```

Out[35]: (VC(I=-2, J=4, K=6), VC(I=-2, J=4, K=6))

```
In [36]: 1 # In-place multiplication of a vector by a scalar
        2 u = SVC(-1, 2, 3)
        3 s = 2
        4 u *= s
        5 u
```

Out[36]: VC(I=-2, J=4, K=6)

```
In [37]: 1 # Create a vector by dividing a vector by another
        2 u = SVC(-3, 4, 6)
        3 v = SVC(2, -2, 2)
        4 u / v
```

Out[37]: VC(I=-1.5, J=-2.0, K=3.0)

```
In [38]: 1 # In-place true division of a vector by another
        2 u = SVC(-3, 4, 6)
        3 v = SVC(2, -2, 2)
        4 u /= v
        5 u
```

Out[38]: VC(I=-1.5, J=-2.0, K=3.0)

```
In [39]: 1 # Create a vector by true dividing a vector by a scalar
        2 u = SVC(-3, 4, 6)
        3 s = 6
        4 u / s
```

Out[39]: VC(I=-0.5, J=0.6666666666666666, K=1.0)

```
In [40]: 1 # In-place true division of a vector by a scalar
        2 u = SVC(-3, 4, 6)
        3 s = 2
        4 u /= s
        5 u
```

Out[40]: VC(I=-1.5, J=2.0, K=3.0)

```
In [41]: 1 # Create a vector by raising a vector to the power of another
        2 u = SVC(-3, 4, 6)
        3 v = SVC(2, -2, 2)
        4 u**v
```

Out[41]: VC(I=9, J=0.0625, K=36)

```
In [42]: 1 # In-place raising a vector to the power of vector
        2 u = SVC(-3, 4, 6)
        3 v = SVC(2, -2, 2)
        4 u **= v
        5 u
```

Out[42]: VC(I=9, J=0.0625, K=36)

```
In [43]: 1 # Create a vector by raising a vector to the power of a scalar
        2 u = SVC(-3, 5, 6)
        3 s = 2
        4 u**s
```

Out[43]: VC(I=9, J=25, K=36)

```
In [44]: 1 # In-place raising a vector to the power of a scalar
2 u = SVC(-3, 5, 6)
3 s = 2
4 u **= s
5 u
```

Out[44]: VC(I=9, J=25, K=36)

```
In [45]: 1 # Create a vector by floor dividing a vector by another
2 u = SVC(-3, 5, 6)
3 v = SVC(2, -2, 2)
4 u // v
```

Out[45]: VC(I=-2, J=-3, K=3)

```
In [46]: 1 # In-place floor division of a vector by another
2 u = SVC(-3, 5, 6)
3 v = SVC(2, -2, 2)
4 u //= v
5 u
```

Out[46]: VC(I=-2, J=-3, K=3)

```
In [47]: 1 # Create a vector by floor dividing a vector by a scalar
2 u = SVC(-3, 5, 6)
3 s = 2
4 u // s
```

Out[47]: VC(I=-2, J=2, K=3)

```
In [48]: 1 # In-place floor division of a vector and a scalar
2 u = SVC(-3, 5, 6)
3 s = 2
4 u //= s
5 u
```

Out[48]: VC(I=-2, J=2, K=3)

```
In [49]: 1 # Create a vector by applying modulus to a vector and another
2 u = SVC(-3, 5, 6)
3 v = SVC(2, -2, 2)
4 u % v
```

Out[49]: VC(I=1, J=-1, K=0)

```
In [50]: 1 # In-place application of modulus to a vector and another
2 u = SVC(-3, 5, 6)
3 v = SVC(2, -2, 2)
4 u %= v
5 u
```

Out[50]: VC(I=1, J=-1, K=0)

```
In [51]: 1 # Create a vector by applying modulus to a vector and a scalar
2 u = SVC(-3, 5, 6)
3 s = 2
4 u % s
```

Out[51]: VC(I=1, J=1, K=0)

```
In [52]: 1 # In-place application of modulus to a vector and a scalar
2 u = SVC(-3, 5, 6)
3 s = 2
4 u %= s
5 u
```

Out[52]: VC(I=1, J=1, K=0)

```
In [ ]: 1
```