In [1]:

```
import oommfc as mc
import discretisedfield as df
import micromagneticmodel as mm
```

In [9]:

```
#Part dimension
L = 500e-9 # sample length(m)
w = 200e-9 # sample width(m)
d = 0.6e-9 # sample thickness(m)
Ms = 1.1e6 # saturation magnetization(A/m)
K = 5.1e5 # Uniaxial Anisotropy(j/m***3)
u=(0, 0, 1) # easy axis
D = 3.5e-3 # DM exchange interraction(J/m**2)
A = 1.6e-11 # Exhange energy constant (J/m)

#Mesh definition
p1 = (0, 0, 0)
p2 = (L, w, d)
cell = (25e-9, 10e-9, 0.6e-9)
region = df.Region(p1=p1 , p2=p2)
mesh = df.Mesh(region=region, cell =cell)
# Micromagnetic definition
system = mm.System(name = 'skyrmion_motion')
system.energy = mm.Exchange(A=A) + mm.DMI(D=D) + mm.UniaxialAnisotropy(K=K, u=u)
```

In [11]:

```
def Ms_fun(pos):
    """Function to set magnitude of magnetisation: zero outside circle,
    Ms inside circle.
    circle radius is 50nm.
    """
    x, y, z = pos
    if (x**2 + y**2) < 50e-9:
        return Ms
    else:
        return 0
```

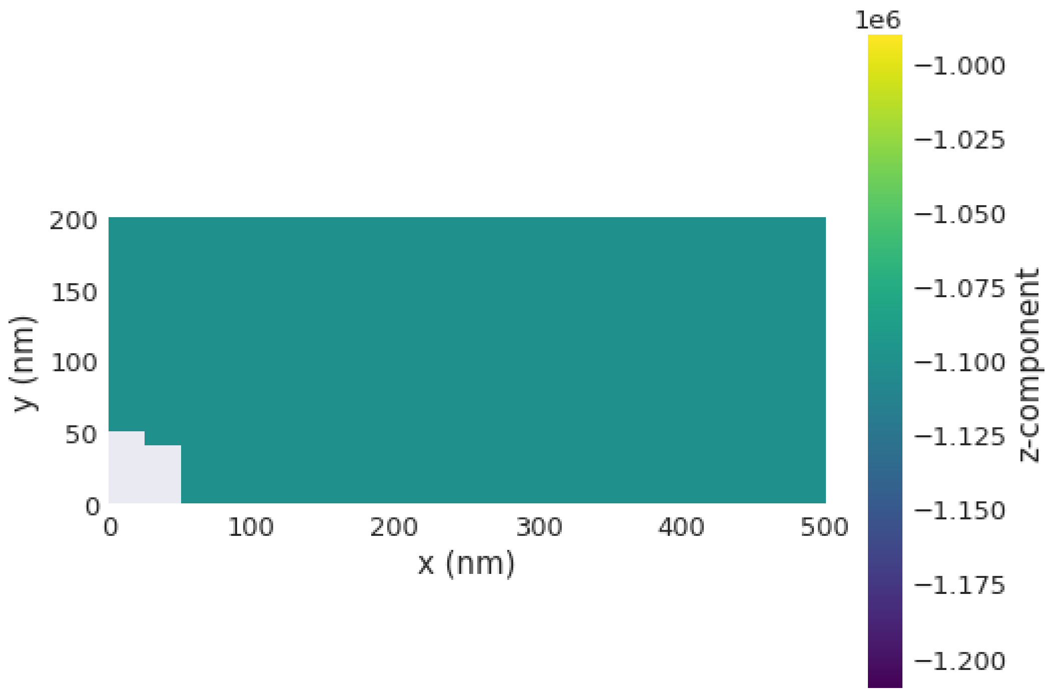
In [12]:

```
def m_init(pos):
    x, y, z = pos
    if (x**2 + y**2)**0.1 < 16e-9**2:
        return (0, 0, 1)
    if (x**2 + y**2)*0.2 > 23e-9**2:
        return (0, 0, -1)
    if 16e-9 < (x**2 + y**2) < 23e-9**2:
        return (1, 0, 0)
system.m = df.Field(mesh, dim=3, value=m_init, norm=Ms_fun)
```

In [13]:

```
system.m.plane('z').mpl()
```

```
/srv/conda/envs/notebook/lib/python3.8/site-packages/matplotlib/quiver.py:71
5: RuntimeWarning: divide by zero encountered in double_scalars
  length = a * (widthu_per_lenu / (self.scale * self.width))
/srv/conda/envs/notebook/lib/python3.8/site-packages/matplotlib/quiver.py:71
5: RuntimeWarning: invalid value encountered in multiply
  length = a * (widthu_per_lenu / (self.scale * self.width))
```


1e6
z-component
y (nm)
x (nm)