Basic Imports

In [1]:
```
import numpy as np
from diffractio import mm, um, degrees
from diffractio.scalar_sources_XY import Scalar_source_XY
from diffractio.scalar_masks_XY import Scalar_mask_XY


# Setting up
length = 1 * mm
num_data = 512
x0 = np.linspace(-length / 2, length / 2, num_data)
y0 = np.linspace(-length / 2, length / 2, num_data)
wavelength = 0.633 * um
```

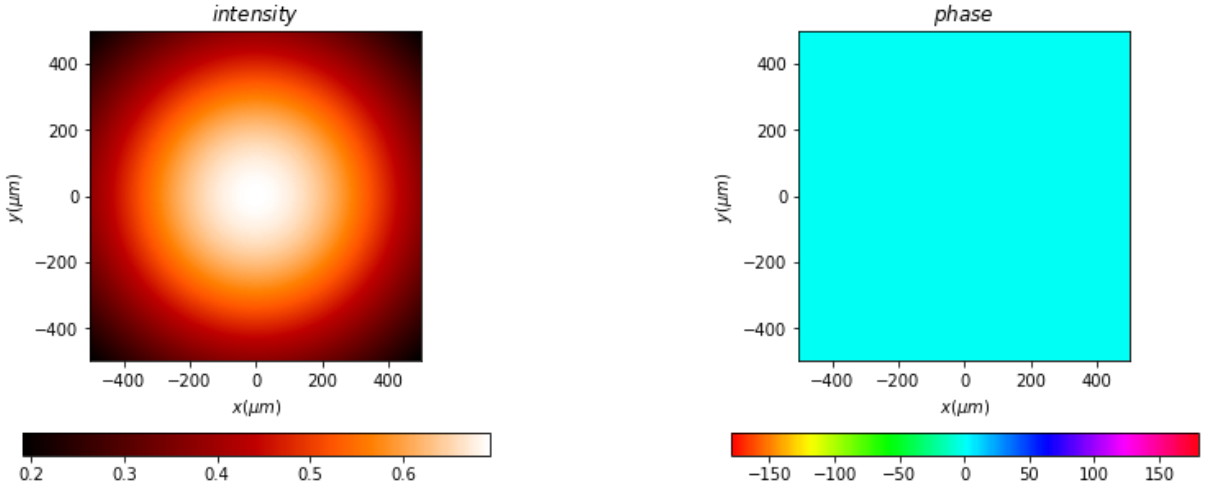
```
number of processors: 12
```

Setting up source

- Gaussian Beam (LASER)

In [2]:
```
# Gaussian Beam Source - like a LASER
u0 = Scalar_source_XY(x=x0, y=y0, wavelength=wavelength)
u0.gauss_beam(r0=(0, 0), w0=(800 * um, 800 * um), z0=0.0)
u0.draw(kind='field', logarithm=True)
```

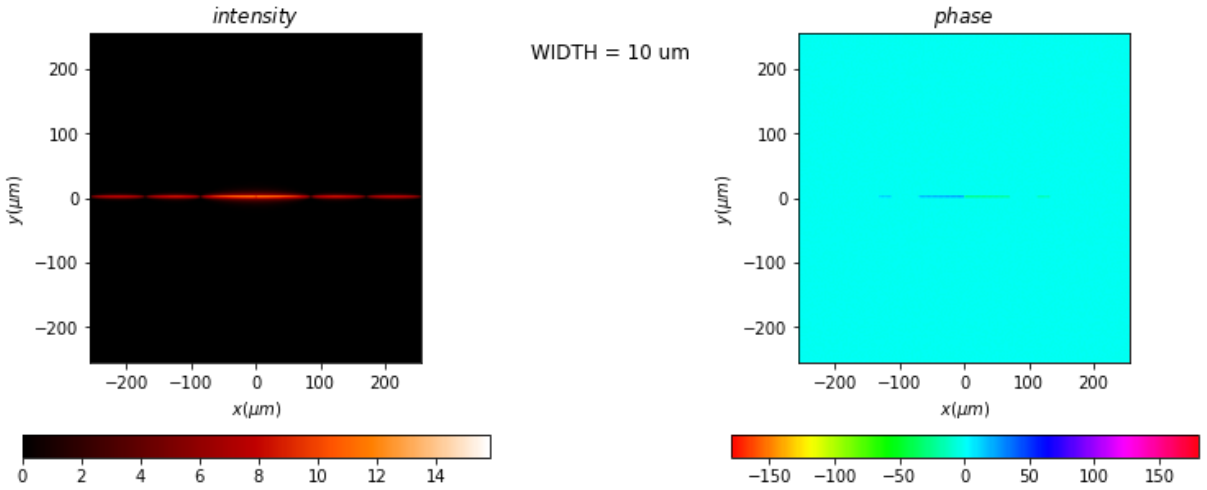
Out[2]:
```
((<matplotlib.image.AxesImage at 0x272b852da60>,
  <matplotlib.image.AxesImage at 0x272b87cc220>),
 None,
 None)
```

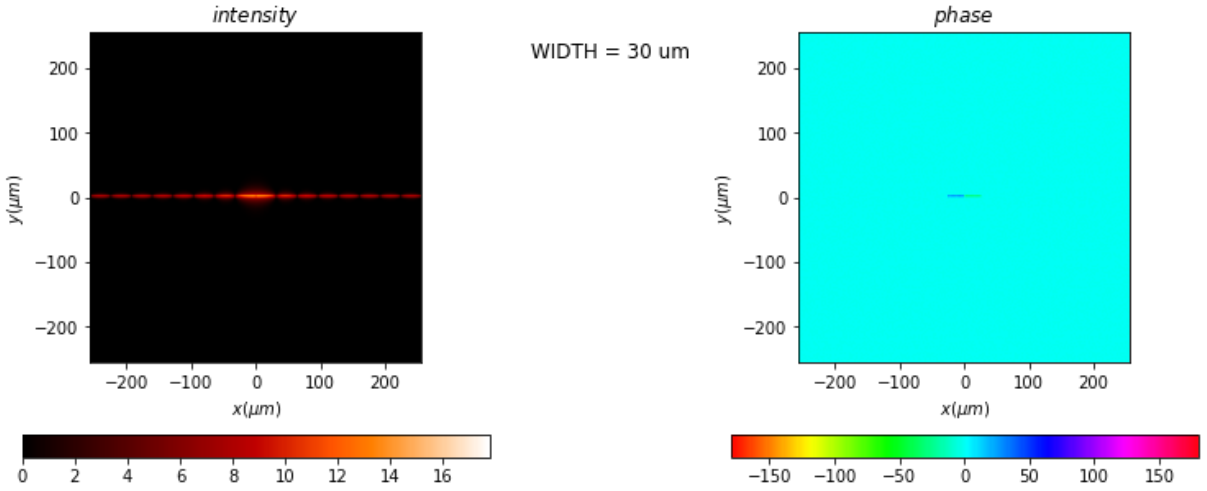


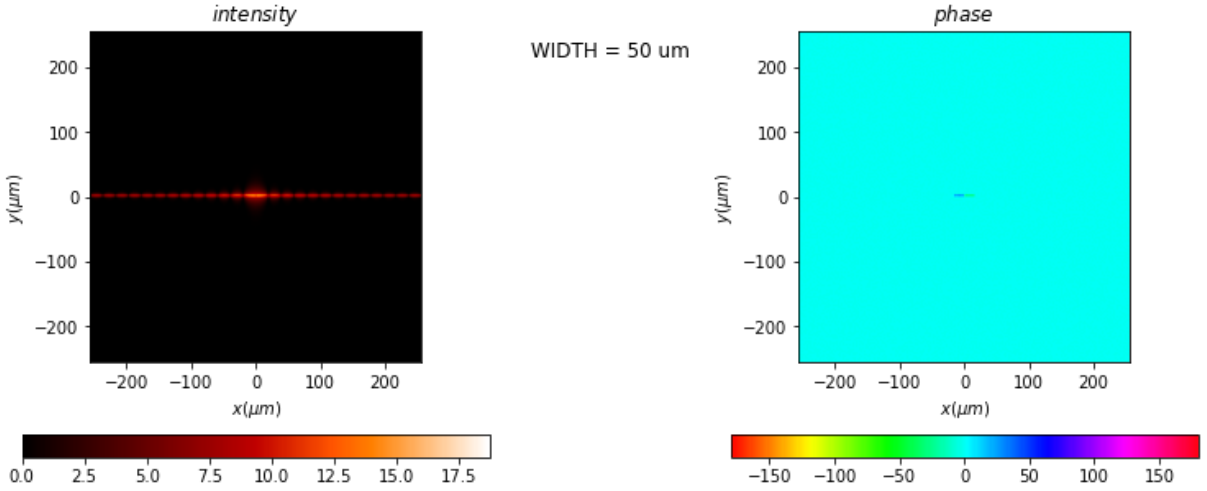
Slits

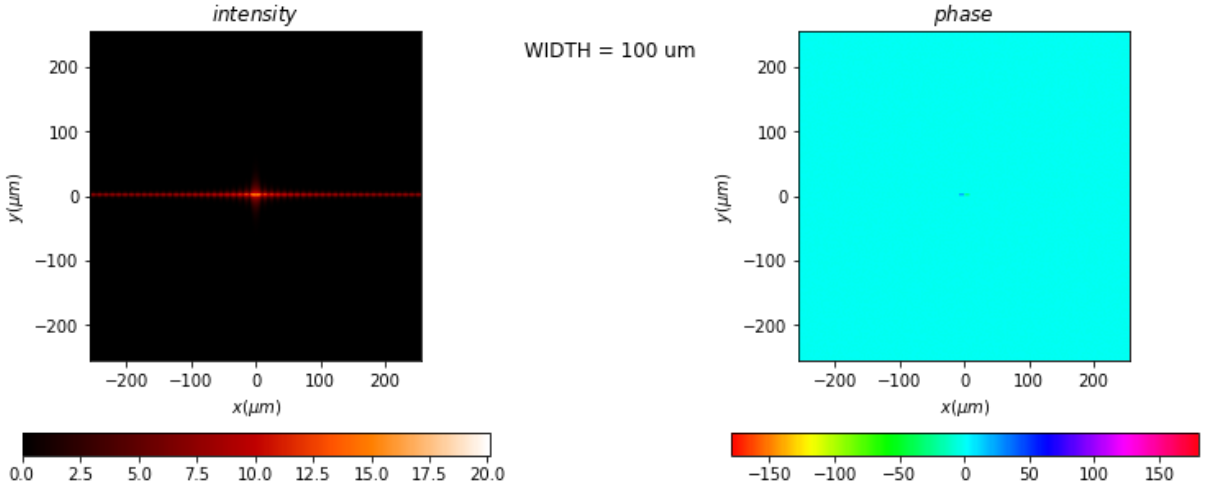
- Fraunhofer Diffraction Patterns for different slit widths

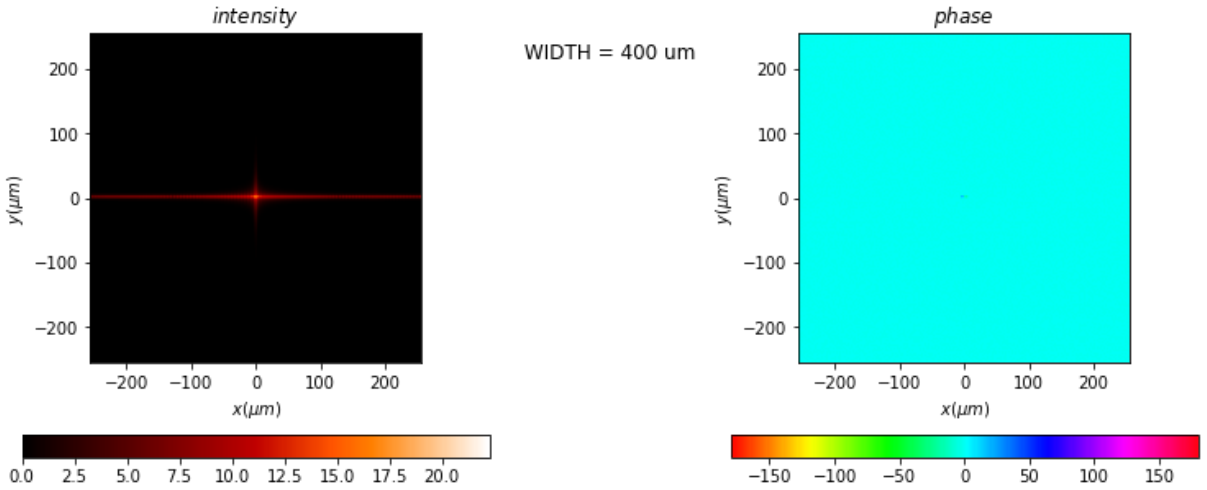
In [3]:
```
widths = [10, 30, 50, 100, 400]  # in um
for width in widths:
    varSlit = Scalar_mask_XY(x0, y0, wavelength)
    varSlit.slit(
        x0=0 * um,
        size=width * um
    )
#     varSlit.draw(kind='field', logarithm=True)
    a_varSlit = (u0 * varSlit).fft(z=1 * mm, new_field=True)
    a_varSlit.draw(title=f" WIDTH = {width} um ", kind='field', logarithm=True)
```


intensity
phase
WIDTH = 10 um
y(μm)
x(μm)


intensity
phase
WIDTH = 30 um
y(μm)
x(μm)


intensity
phase
WIDTH = 50 um
y(μm)
x(μm)


intensity
phase
WIDTH = 100 um
y(μm)
x(μm)



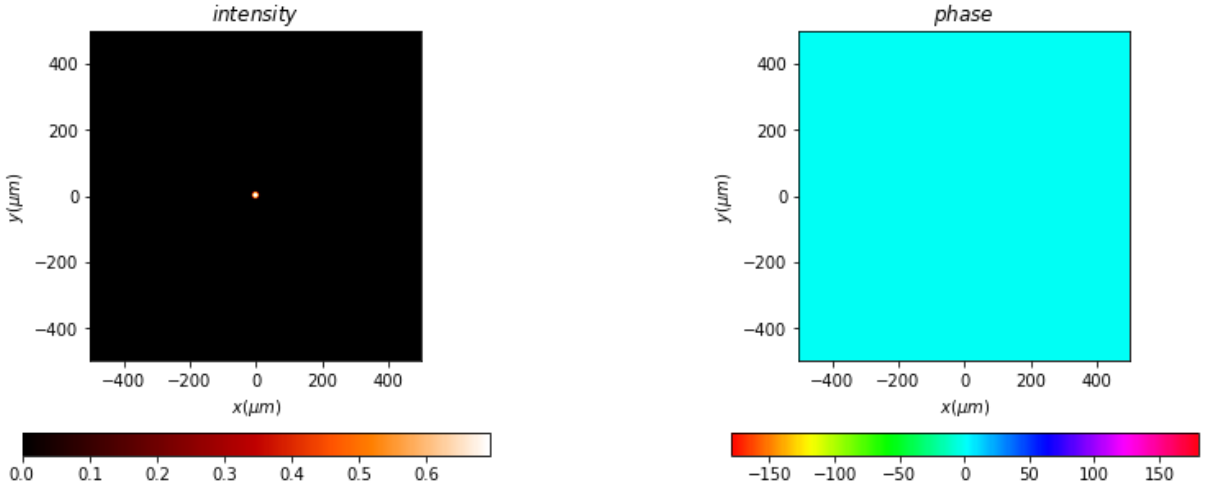
The intensity variation matches the expected profile.

Pinhole

- Fraunhofer Diffraction Patterns for different pinhole radii

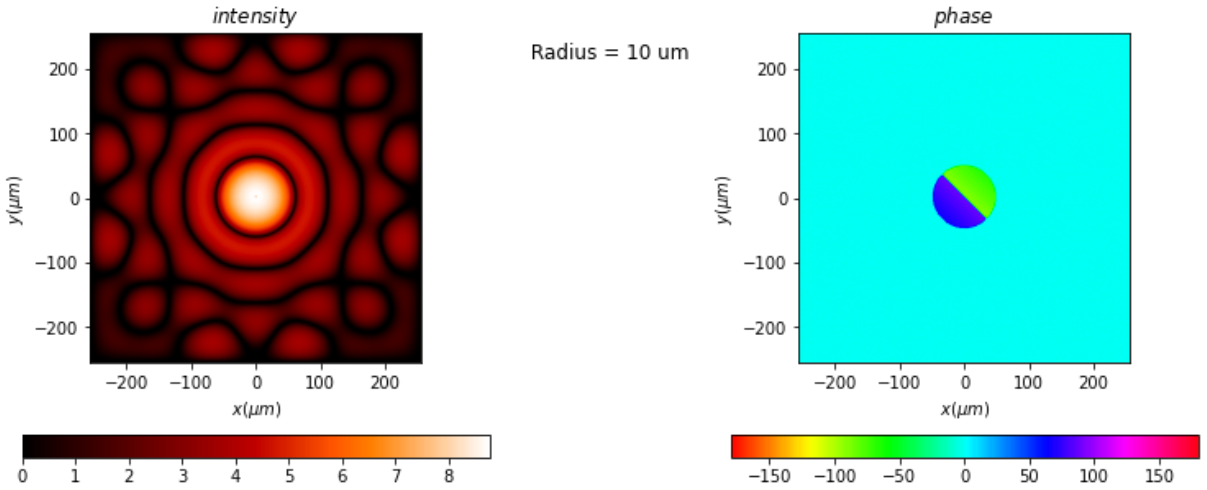
In [4]:
```
# Showcasing a pinhole
circ = Scalar_mask_XY(x0, y0, wavelength)
circ.circle(
    r0=(0 * um, 0 * um),
    radius=(10 * um, 10 * um),
    angle=0
)
circ.draw(kind='field', logarithm=True)
```

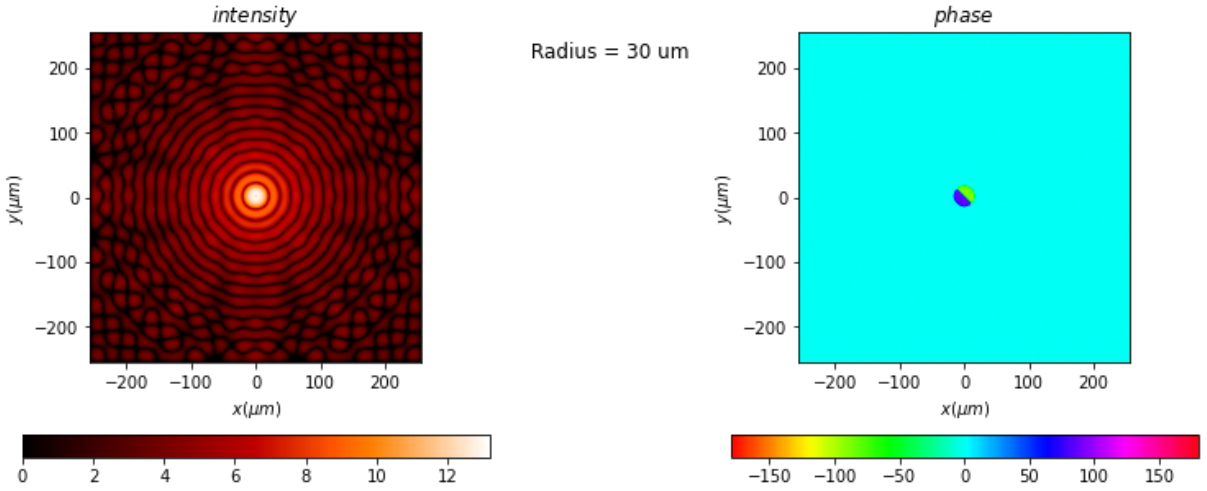
Out[4]:
```
((<matplotlib.image.AxesImage at 0x272bd337eb0>,
  <matplotlib.image.AxesImage at 0x272bd3aa6a0>),
 None,
 None)
```

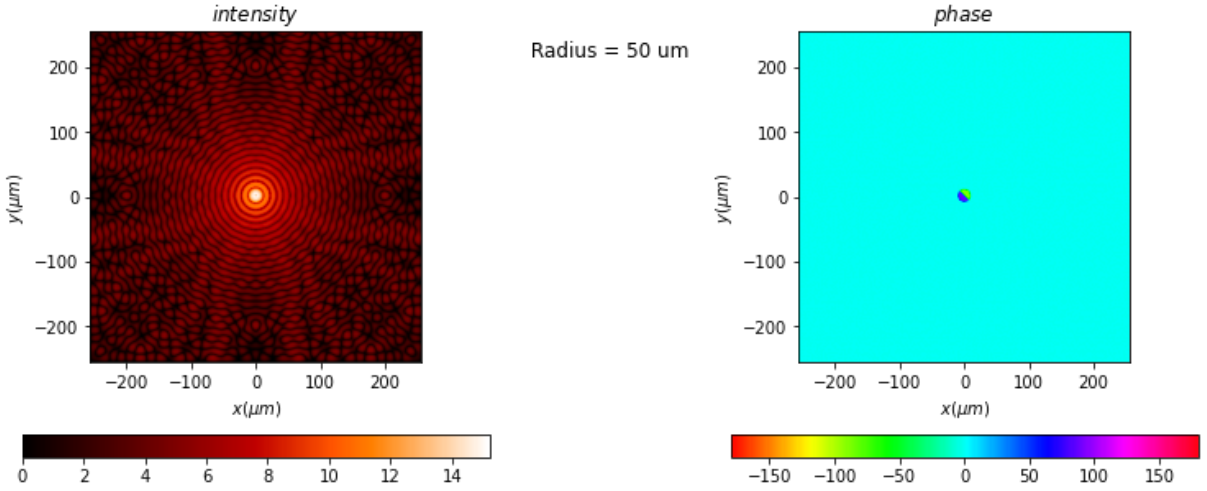


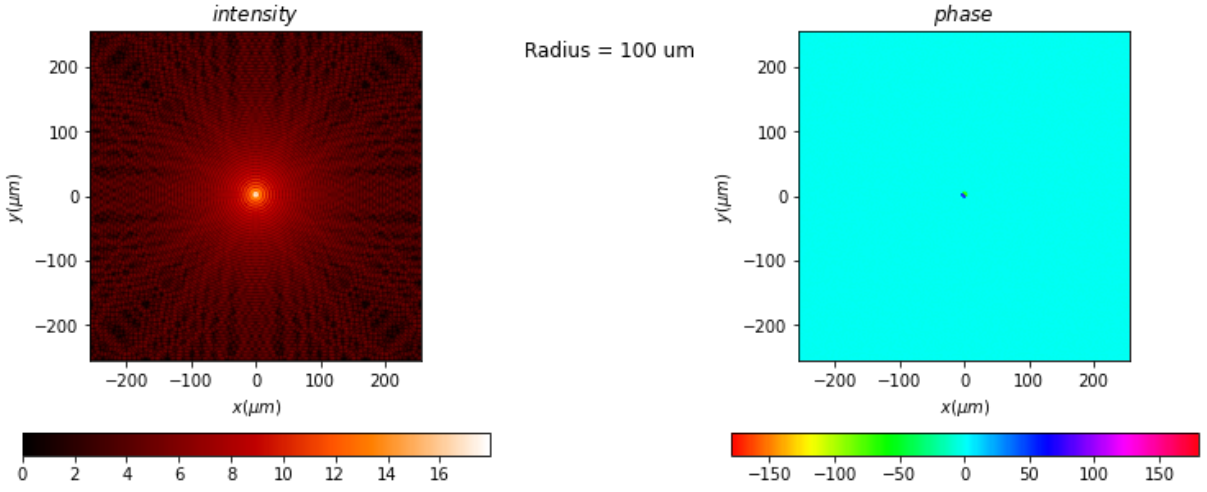
Diffraction pattern for multiple pinholes having different radius

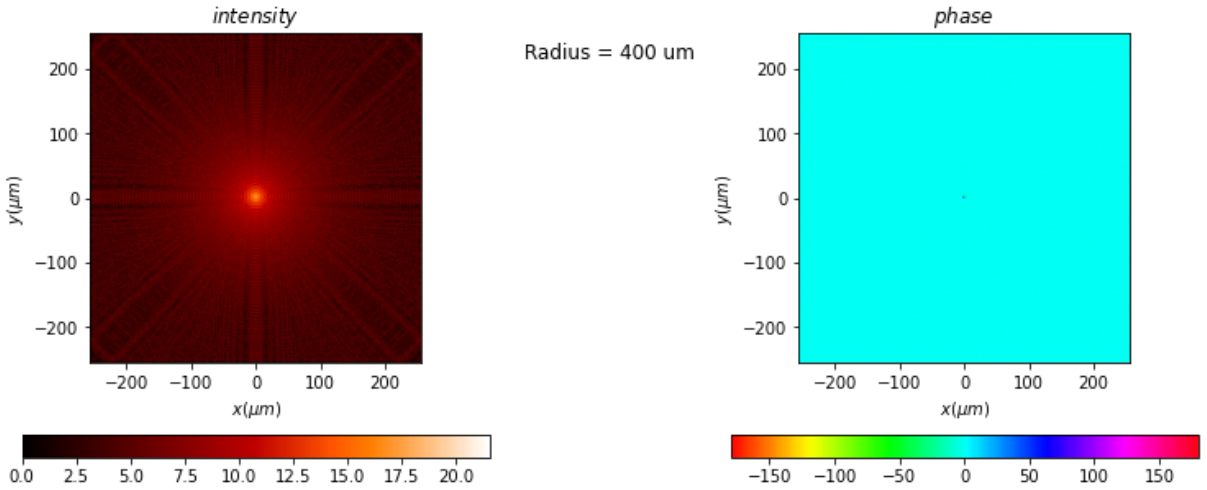
In [5]:
```
radii = [10, 30, 50, 100, 400]  # in um
for rad in radii:
    circ = Scalar_mask_XY(x0, y0, wavelength)
    circ.circle(
        r0=(0 * um, 0 * um),
        radius=(rad * um, rad * um),
        angle=0
    )

    a_circ = (u0 * circ).fft(z=1 * mm, new_field=True)
    a_circ.draw(title=f" Radius = {rad} um ", kind='field', logarithm=True)
```


intensity
phase
Radius = 10 um
x(μm)
y(μm)


intensity
phase
Radius = 30 um
x(μm)
y(μm)


intensity
phase
Radius = 50 um
x(μm)
y(μm)


intensity
phase
Radius = 100 um
x(μm)
y(μm)



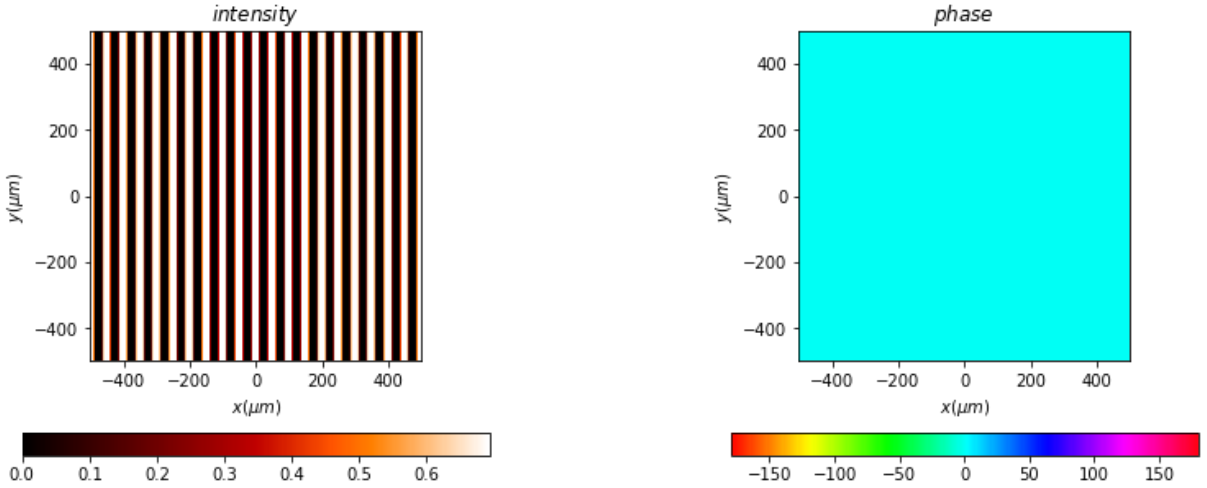
We observe the well-known Airy Disk in all cases, as theoretically expected.

Grating

- Fraunhofer Diffraction Patterns for grating

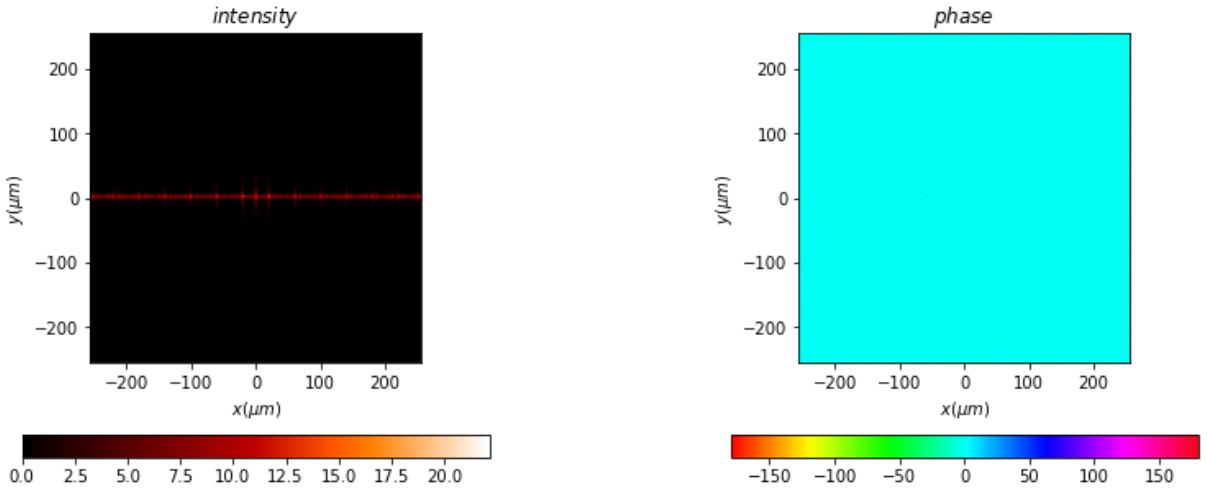
In [6]:
```
grating = Scalar_mask_XY(x0, y0, wavelength)
grating.ronchi_grating(
    period=50 * um,
)

grating.draw(kind='field', logarithm=True)
```

Out[6]:
```
((<matplotlib.image.AxesImage at 0x272bf6b80a0>,
  <matplotlib.image.AxesImage at 0x272bcbed820>),
 None,
 None)
```



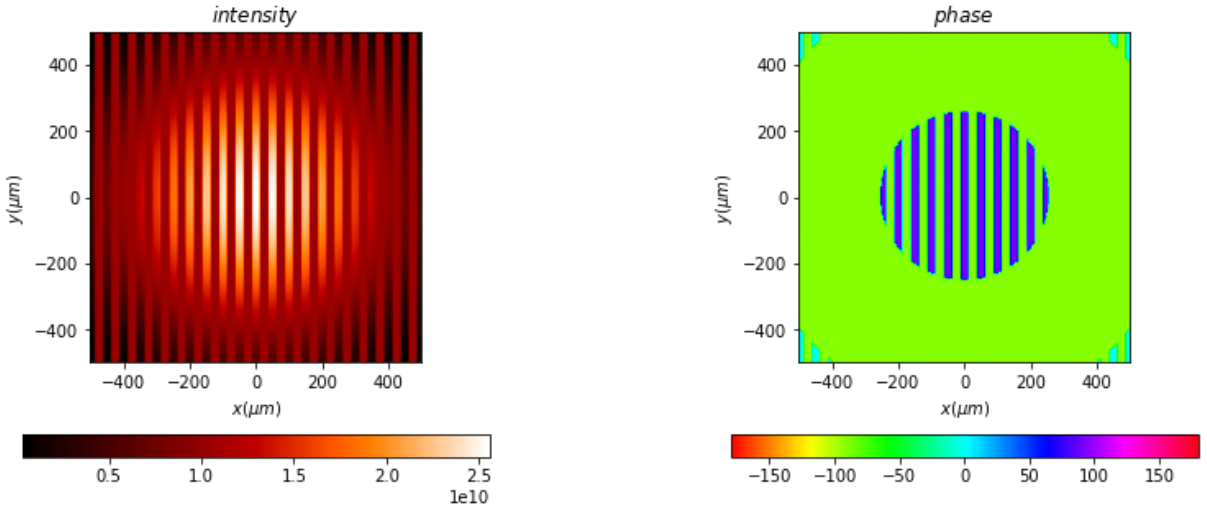
In [7]:
```
# Fourier Plane - No Filter
a_L1 = (u0 * grating).fft(z=1 * mm, new_field=True)
a_L1.draw(kind='field', logarithm=True)
```

Out[7]:
```
((<matplotlib.image.AxesImage at 0x272b95608e0>,
  <matplotlib.image.AxesImage at 0x272b954b0a0>),
 None,
 None)
```



In [8]:
```
a_L2 = a_L1.fft(z=1 * mm, shift=False, remove0=False, new_field=True)
a_L2.draw(kind='field', logarithm=False)
```

Out[8]:
```
((<matplotlib.image.AxesImage at 0x272b95ee6a0>,
  <matplotlib.image.AxesImage at 0x272b928b970>),
 None,
 None)
```



In [ ]:

Mesh

Compare results with figures in Eisenkraft (1977)

In [1]:
```
import numpy as np
from diffractio import mm, um, degrees
from diffractio.scalar_sources_XY import Scalar_source_XY
from diffractio.scalar_masks_XY import Scalar_mask_XY


# Setting up
length = 1 * mm
num_data = 512
x0 = np.linspace(-length / 2, length / 2, num_data)
y0 = np.linspace(-length / 2, length / 2, num_data)
wavelength = 0.633 * um
```

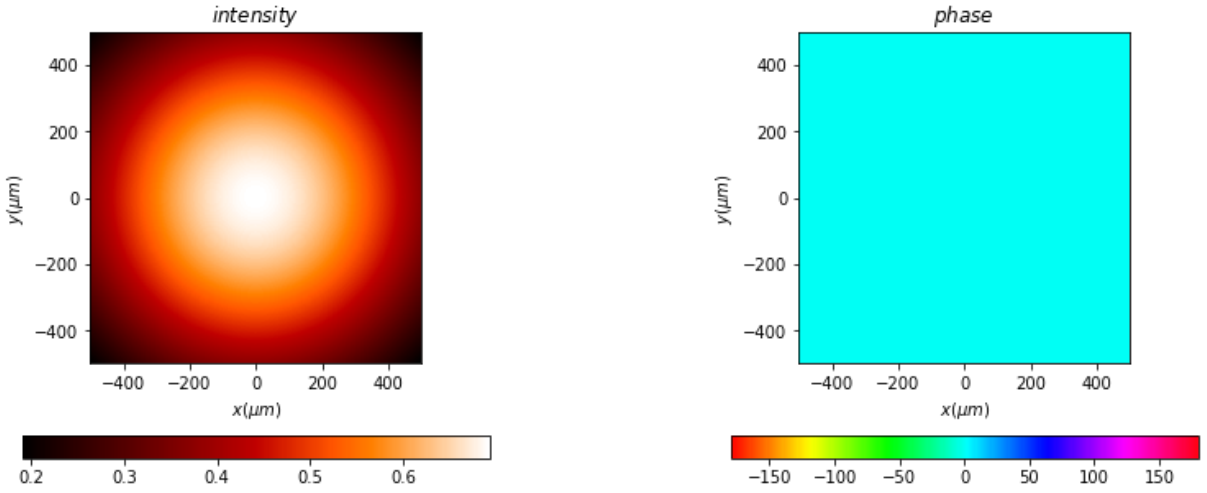
```
number of processors: 12
```

Setting up source(s)

- Gaussian Beam (LASER)

In [2]:
```
# Gaussian Beam Source - like a LASER
u0 = Scalar_source_XY(x=x0, y=y0, wavelength=wavelength)
u0.gauss_beam(r0=(0, 0), w0=(800 * um, 800 * um), z0=0.0)
u0.draw(kind='field', logarithm=True)
```

Out[2]:
```
((<matplotlib.image.AxesImage at 0x249d2c1ea60>,
  <matplotlib.image.AxesImage at 0x249d2ebc220>),
 None,
 None)
```



Mesh

In [3]:
```
mesh = Scalar_mask_XY(x0, y0, wavelength)
mesh.grating_2D(
    period=50 * um,
    fill_factor=0.5,
    angle=0 * degrees
)

mesh.draw(kind='field', logarithm=True)
```

Out[3]:
```
((<matplotlib.image.AxesImage at 0x249d36f11c0>,
  <matplotlib.image.AxesImage at 0x249d3960940>),
 None,
 None)
```

Basic Imports

In [1]:
```
import numpy as np
from diffractio import mm, um, degrees
from diffractio.scalar_sources_XY import Scalar_source_XY
from diffractio.scalar_masks_XY import Scalar_mask_XY


# Setting up
length = 1 * mm
num_data = 512
x0 = np.linspace(-length / 2, length / 2, num_data)
y0 = np.linspace(-length / 2, length / 2, num_data)
wavelength = 0.633 * um
```

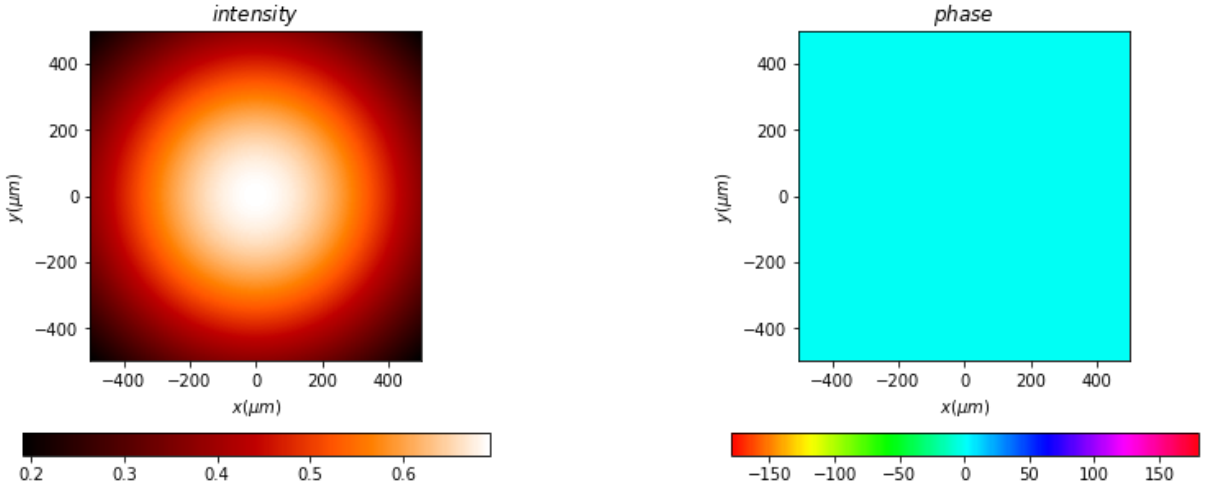
```
number of processors: 12
```

Setting up source

- Gaussian Beam (LASER)

In [2]:
```
# Gaussian Beam Source - like a LASER
u0 = Scalar_source_XY(x=x0, y=y0, wavelength=wavelength)
u0.gauss_beam(r0=(0, 0), w0=(800 * um, 800 * um), z0=0.0)
u0.draw(kind='field', logarithm=True)
```

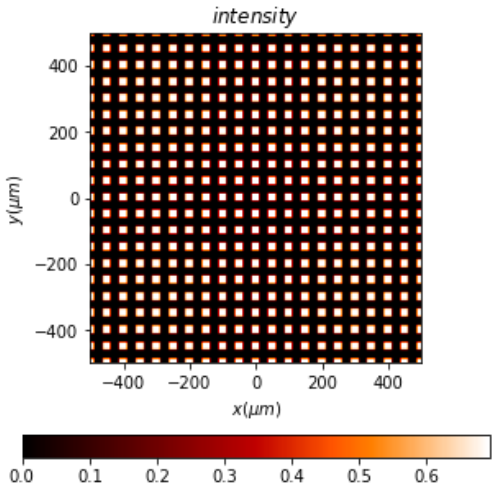
Out[2]:
```
((<matplotlib.image.AxesImage at 0x272b852da60>,
  <matplotlib.image.AxesImage at 0x272b87cc220>),
 None,
 None)
```

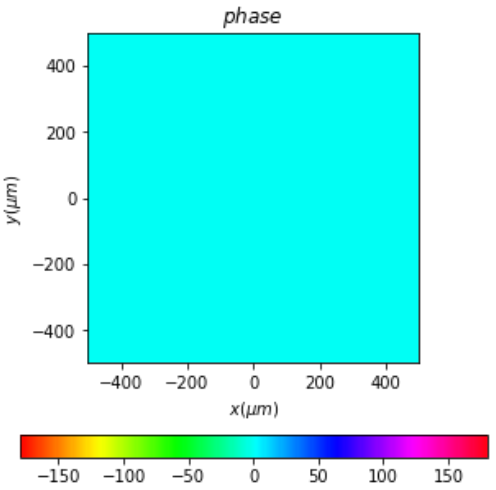


Slits

- Fraunhofer Diffraction Patterns for different slit widths

In [3]:
```
widths = [10, 30, 50, 100, 400]  # in um
for width in widths:
    varSlit = Scalar_mask_XY(x0, y0, wavelength)
    varSlit.slit(
        x0=0 * um,
        size=width * um
    )
#     varSlit.draw(kind='field', logarithm=True)
    a_varSlit = (u0 * varSlit).fft(z=1 * mm, new_field=True)
    a_varSlit.draw(title=f" WIDTH = {width} um ", kind='field', logarithm=True)
```

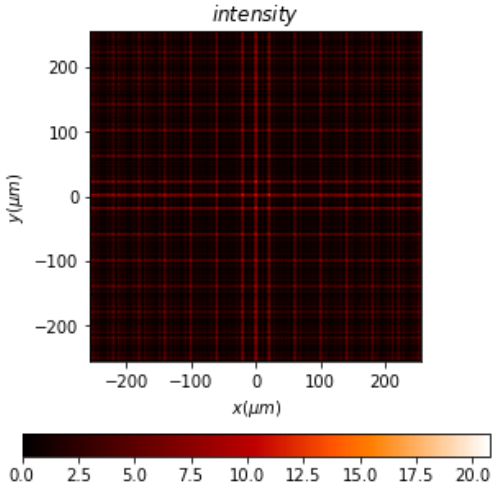


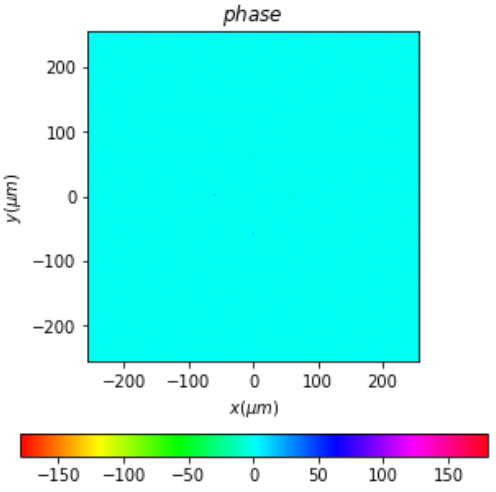


In [4]:
```
# angledMesh = Scalar_mask_XY(x0, y0, wavelength)
# angledMesh.grating_2D(
#     period=50 * um,
#     fill_factor=0.5,
#     angle=45 * degrees
# )

# angledMesh.draw(kind='field', logarithm=True)
```

In [5]:
```
# Fourier Plane - No Filter
a_L1 = (u0 * mesh).fft(z=1 * mm, new_field=True)
a_L1.draw(kind='field', logarithm=True)
```

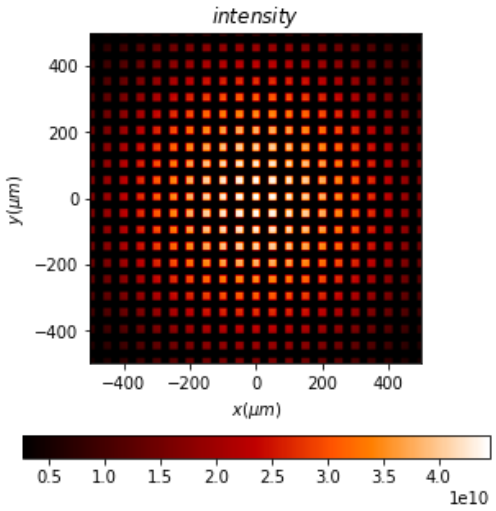
Out[5]:
```
((<matplotlib.image.AxesImage at 0x249d356fe80>,
  <matplotlib.image.AxesImage at 0x249d3607670>),
 None,
 None)
```

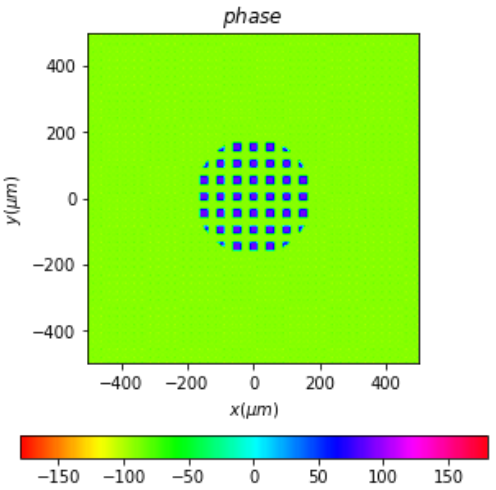




In [6]:
```
# This is how, it'd look without any filter, at the Observation screen
a_L2 = a_L1.fft(z=1 * mm, shift=False, remove0=False, new_field=True)
a_L2.draw(kind='field', logarithm=False)
```

Out[6]:
```
((<matplotlib.image.AxesImage at 0x249d3deaee0>,
  <matplotlib.image.AxesImage at 0x249d6a856d0>),
 None,
 None)
```





Masks / Filters

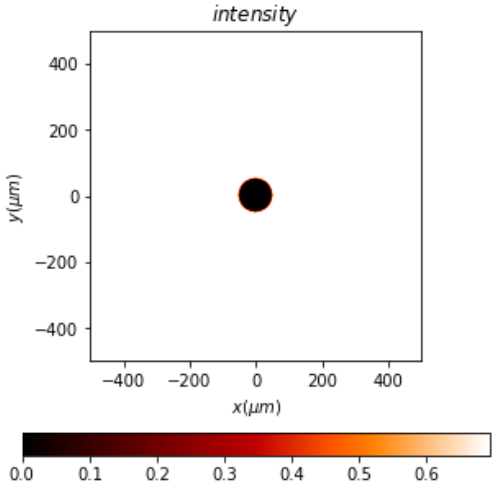
- Center Dot - High Pass
- Square - Low Pass
- Square + Center Dot

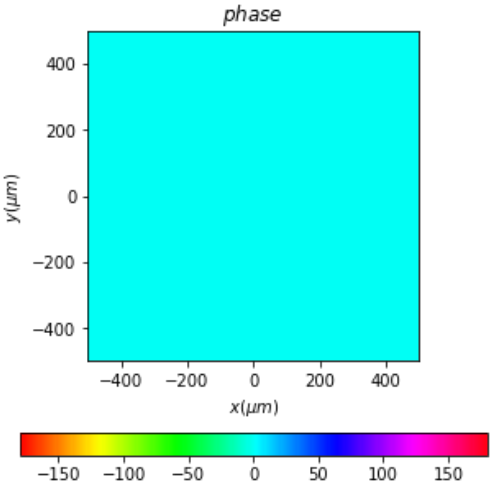
- Vertical Slit

- Horizontal Slit
- Angled Slit

In [7]:
```
cDot = Scalar_mask_XY(x0, y0, wavelength)
cDot.ring(
    r0=(0 * um, 0 * um),
    radius1=(50 * um, 50 * um),
    radius2=(1000 * um, 1000 * um)
)
cDot.draw(kind='field', logarithm=True)
```

Out[7]:
```
((<matplotlib.image.AxesImage at 0x249d6b5e460>,
  <matplotlib.image.AxesImage at 0x249d6bcdbe0>),
 None,
 None)
```

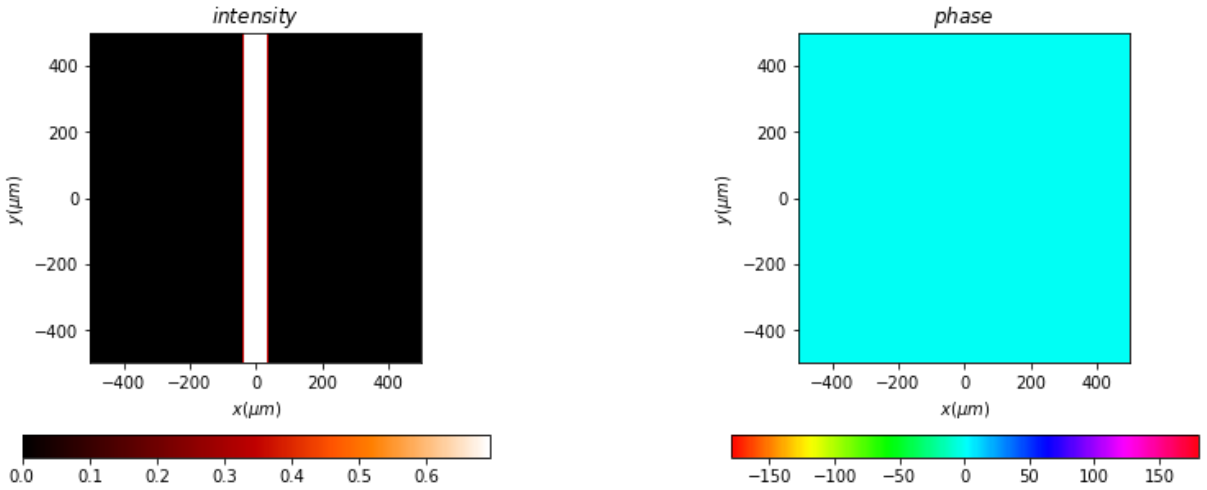




In [8]:
```
vert = Scalar_mask_XY(x0, y0, wavelength)
vert.slit(
    x0=0 * um,
    size=75 * um
)
vert.draw(kind='field', logarithm=True)
```

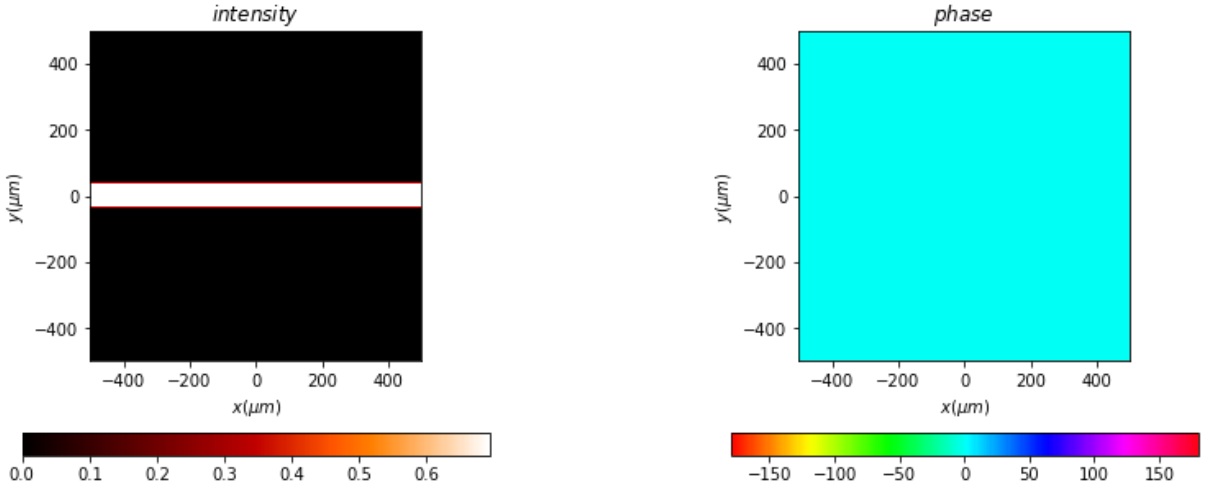
Out[8]:
```
((<matplotlib.image.AxesImage at 0x249d3695460>,
```

```
 <matplotlib.image.AxesImage at 0x249d2e91ac0>),
None,
None)
```



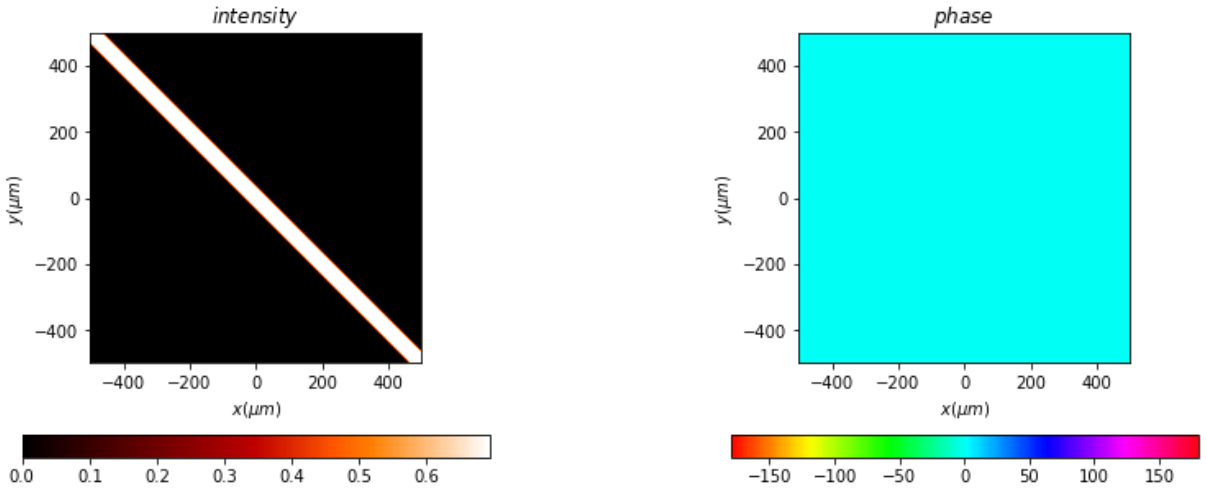
In [9]:
```
horiz = Scalar_mask_XY(x0, y0, wavelength)
horiz.slit(
    x0=0 * um,
    size=75 * um,
    angle=np.pi / 2
)
horiz.draw(kind='field', logarithm=True)
```

Out[9]:
```
((<matplotlib.image.AxesImage at 0x249d8190130>,
  <matplotlib.image.AxesImage at 0x249d81e08e0>),
 None,
 None)
```



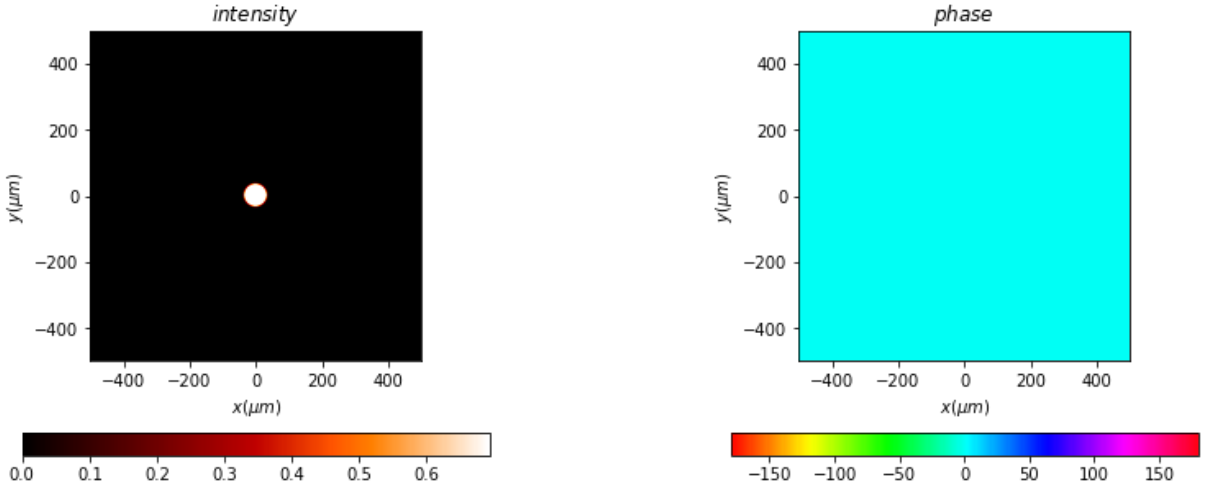
In [10]:
```
angled = Scalar_mask_XY(x0, y0, wavelength)
angled.slit(
    x0=0 * um,
    size=50 * um,
    angle=np.pi / 4
)
angled.draw(kind='field', logarithm=True)
```

Out[10]:
```
((<matplotlib.image.AxesImage at 0x249d8f35a00>,
  <matplotlib.image.AxesImage at 0x249d8f90220>),
 None,
 None)
```



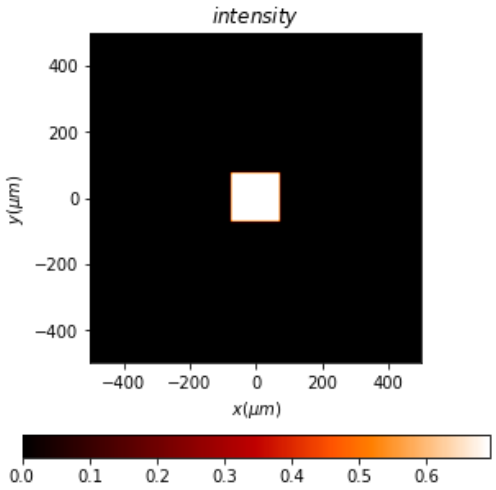
In [11]:
```
circ = Scalar_mask_XY(x0, y0, wavelength)
circ.circle(
    r0=(0 * um, 0 * um),
    radius=(35 * um, 35 * um),
    angle=0
)
circ.draw(kind='field', logarithm=True)
```

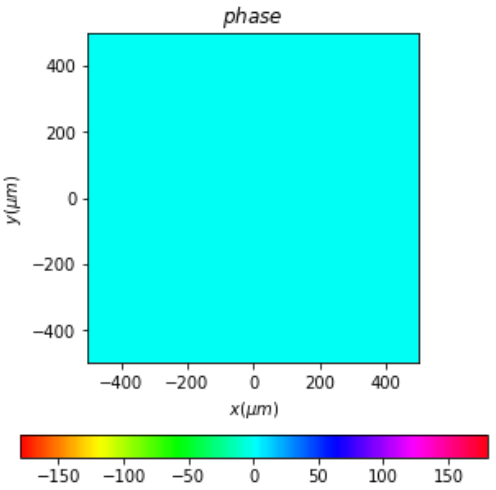
Out[11]:
```
((<matplotlib.image.AxesImage at 0x249d90575e0>,
  <matplotlib.image.AxesImage at 0x249d90d0910>),
 None,
 None)
```



In [12]:
```
sq = Scalar_mask_XY(x0, y0, wavelength)
sq.square(
    r0=(0 * um, 0 * um),
    size=(150 * um, 150 * um),
    angle=0,
)
sq.draw(kind='field', logarithm=True)
```

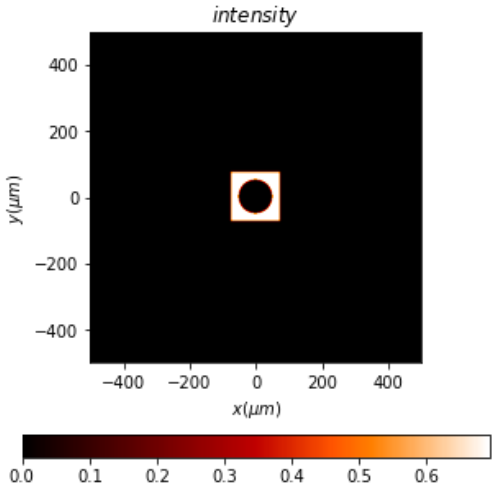
Out[12]:
```
((<matplotlib.image.AxesImage at 0x249da61ddc0>,
  <matplotlib.image.AxesImage at 0x249da698070>),
 None,
 None)
```

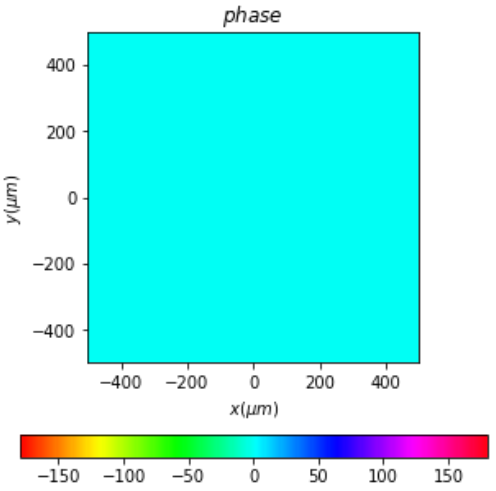




In [13]:
```
# Adding two filters
sqCirc = sq * cDot
sqCirc.draw(kind="field", logarithm=True)
```

Out[13]:
```
((<matplotlib.image.AxesImage at 0x249da6da160>,
  <matplotlib.image.AxesImage at 0x249d8146f70>),
 None,
 None)
```





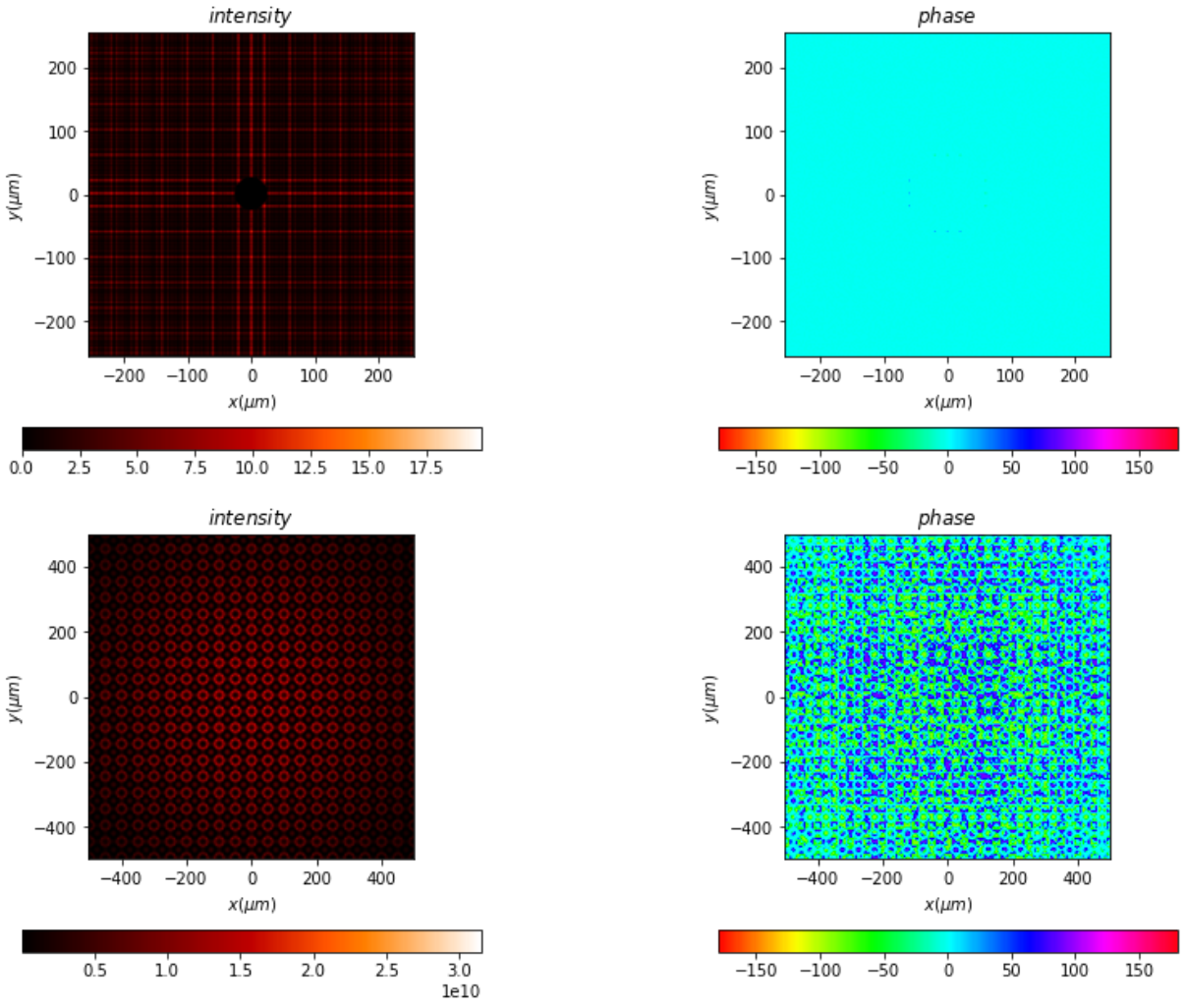
Returns wave after encountering filter and then L2

In [14]:
```
def sim(a_L1, mask):
    """
    a_L1: Wave after passing through Lens 1
    mask: Mask or Filter to apply

    """
    a_L1_Mask = a_L1 * mask
    a_L1_Mask_L2 = a_L1_Mask.fft(z=1 * mm, shift=False, remove0=False, new_field=True)

    return a_L1_Mask, a_L1_Mask_L2
```

Center Dot - High Pass - EDGE DETECTION

In [15]:
```
a_L1_cD, a_L1_cD_L2 = sim(a_L1, cDot)
a_L1_cD.draw(kind='field', logarithm=True)
a_L1_cD_L2.draw(kind='field', logarithm=False)
```

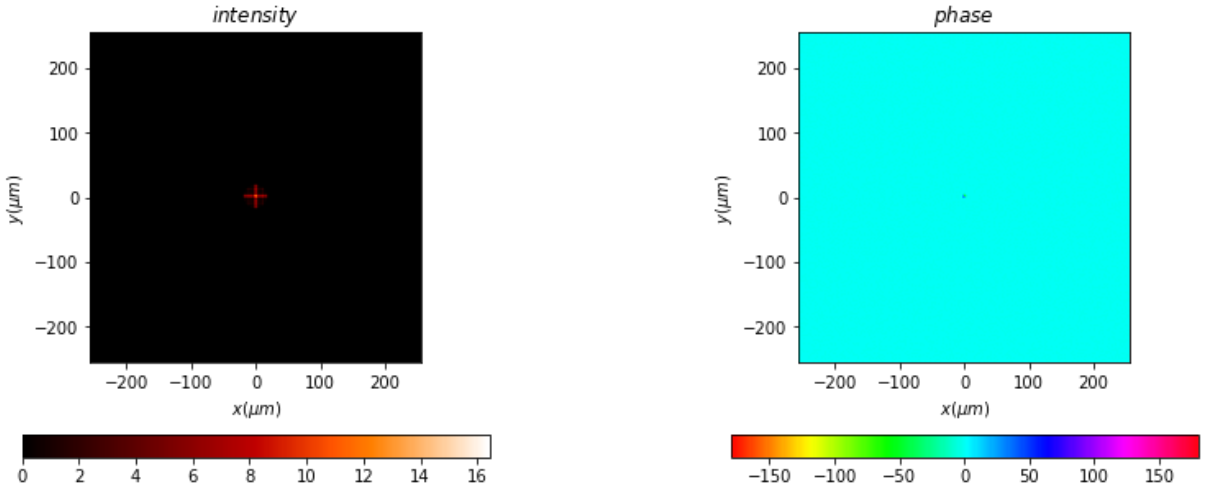
Out[15]:
```
((<matplotlib.image.AxesImage at 0x249db350ee0>,
  <matplotlib.image.AxesImage at 0x249dbe436d0>),
 None,
 None)
```

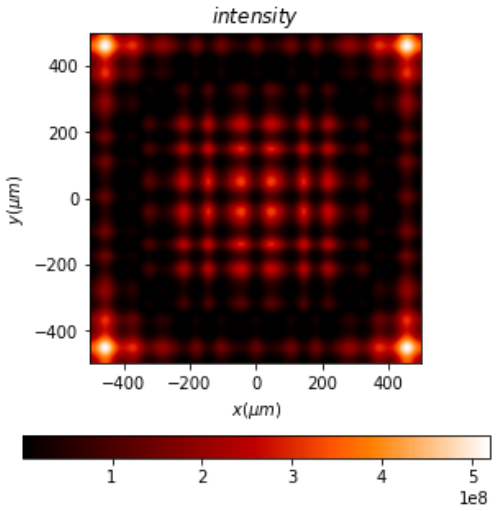


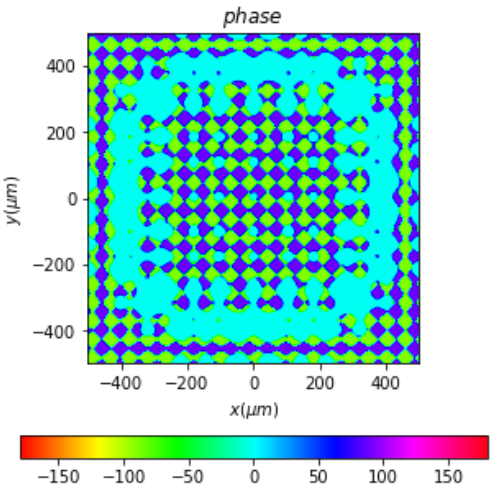
Circle - Low Pass - Blurred Output

In [16]:
```
a_L1_Circ, a_L1_Circ_L2 = sim(a_L1, circ)
a_L1_Circ.draw(kind='field', logarithm=True)
a_L1_Circ_L2.draw(kind='field', logarithm=False)
```

Out[16]:
```
((<matplotlib.image.AxesImage at 0x249dcee2a00>,
  <matplotlib.image.AxesImage at 0x249dcf4dd00>),
 None,
 None)
```



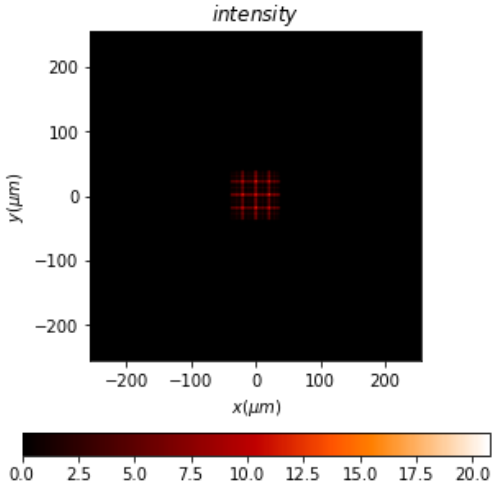


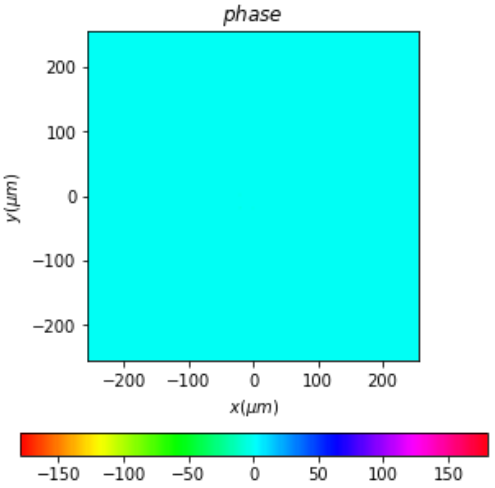


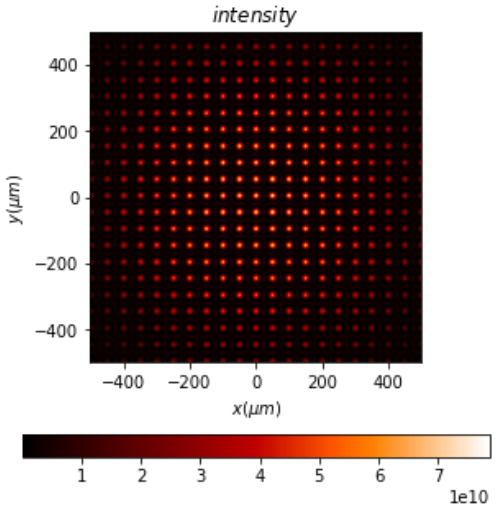
Square

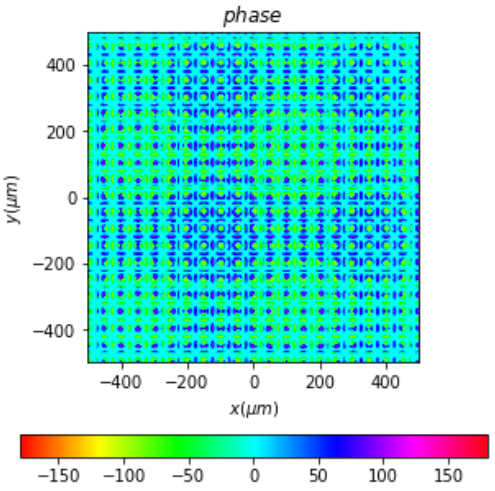
In [17]:
```
a_L1_Sq, a_L1_Sq_L2 = sim(a_L1, sq)
a_L1_Sq.draw(kind='field', logarithm=True)
a_L1_Sq_L2.draw(kind='field', logarithm=False)
```

Out[17]:
```
((<matplotlib.image.AxesImage at 0x249d8f9f0d0>,
  <matplotlib.image.AxesImage at 0x249dcea9220>),
 None,
 None)
```





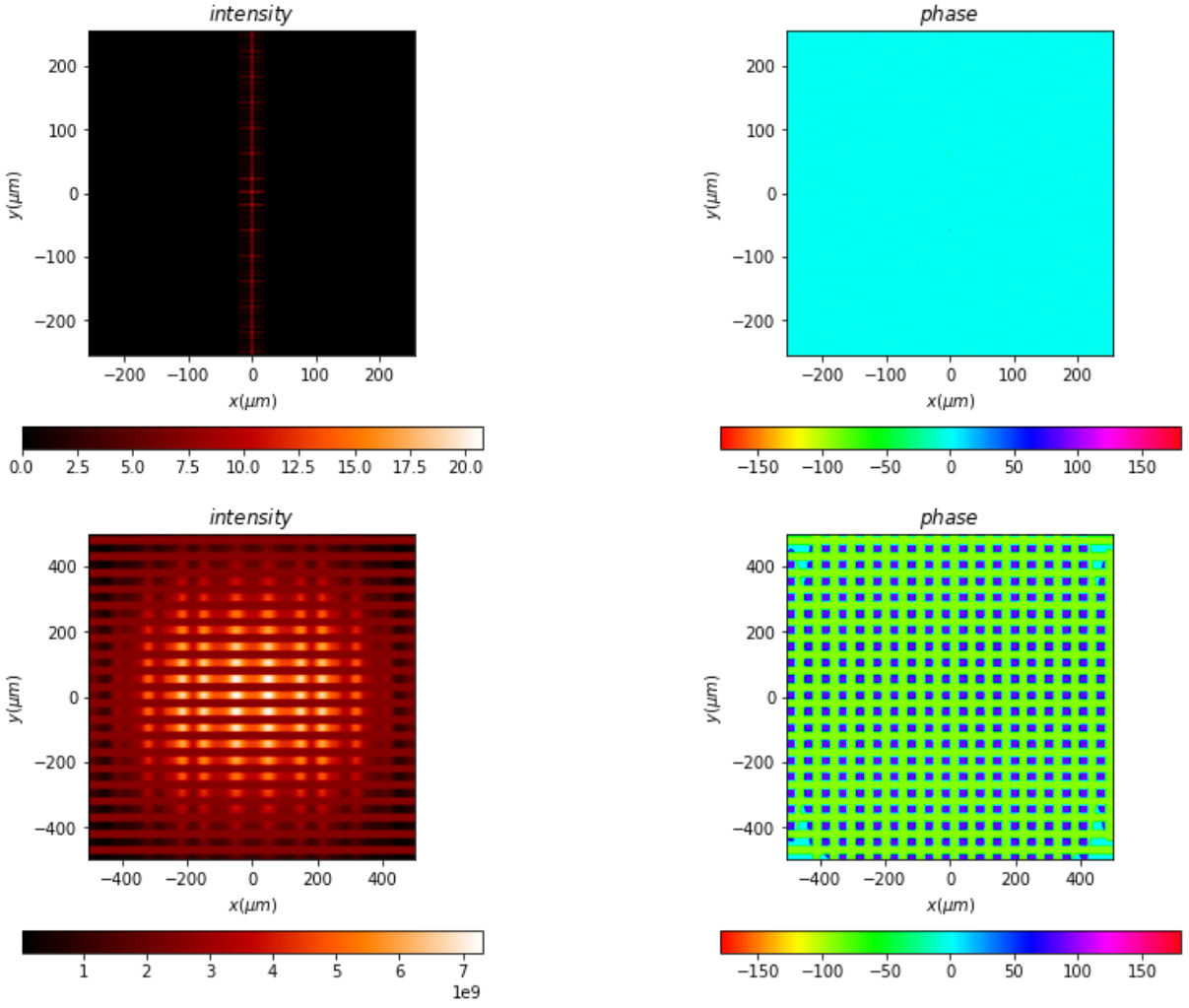




Vertical Slit

In [18]:
```
a_L1_Vert, a_L1_Vert_L2 = sim(a_L1, vert)
a_L1_Vert.draw(kind='field', logarithm=True)
a_L1_Vert_L2.draw(kind='field', logarithm=False)
```

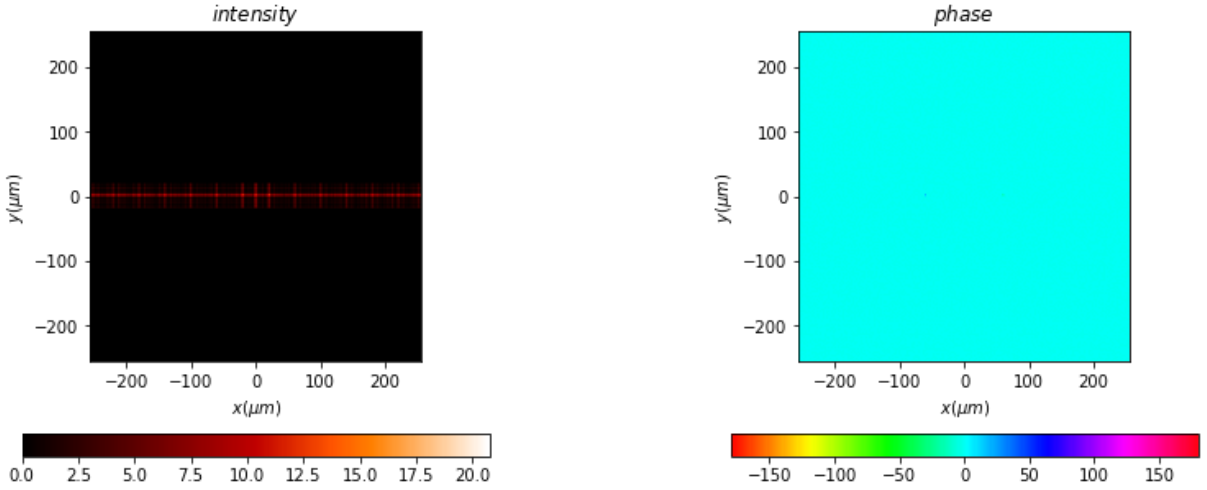
Out[18]:
```
((<matplotlib.image.AxesImage at 0x249de599910>,
  <matplotlib.image.AxesImage at 0x249dbfe10d0>),
 None,
 None)
```

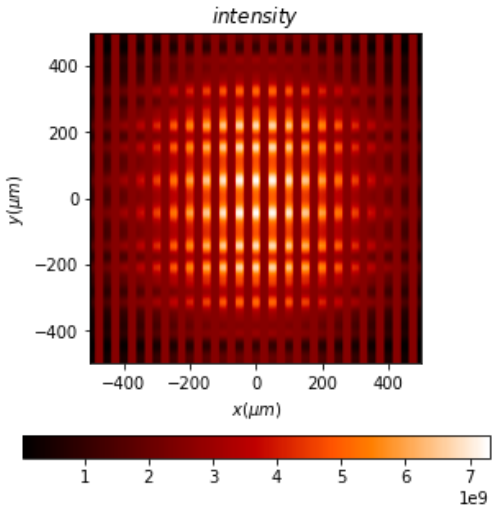


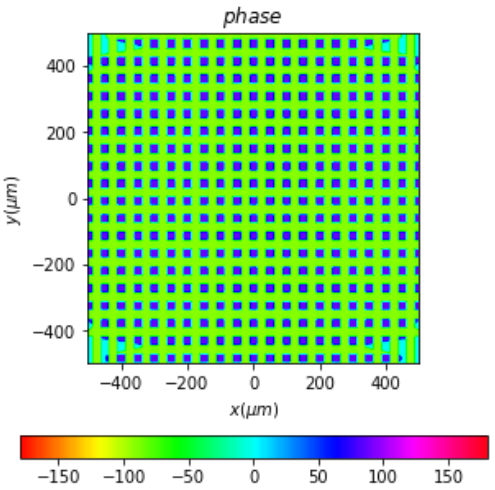
Horizontal Slit

In [19]:
```
a_L1_Horiz, a_L1_Horiz_L2 = sim(a_L1, horiz)
a_L1_Horiz.draw(kind='field', logarithm=True)
a_L1_Horiz_L2.draw(kind='field', logarithm=False)
```

Out[19]:
```
((<matplotlib.image.AxesImage at 0x249df20fd30>,
  <matplotlib.image.AxesImage at 0x249df27bfa0>),
 None,
 None)
```



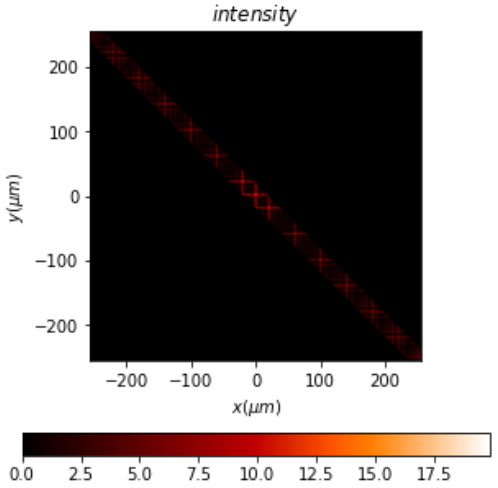


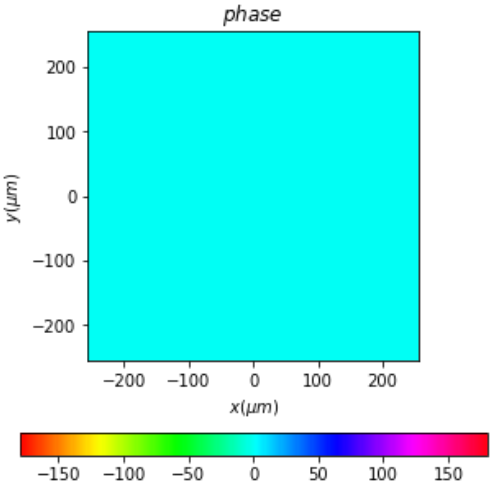


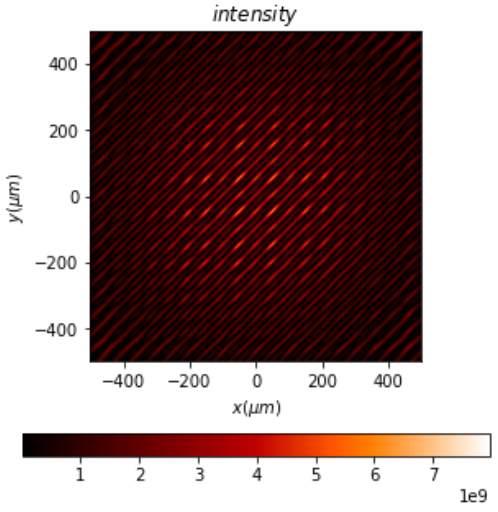
Angled Slit

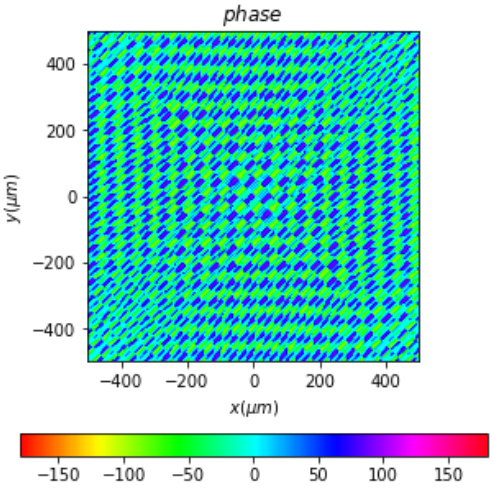
In [20]:
```
a_L1_Angled, a_L1_Angled_L2 = sim(a_L1, angled)
a_L1_Angled.draw(kind='field', logarithm=True)
a_L1_Angled_L2.draw(kind='field', logarithm=False)
```

Out[20]:
```
((<matplotlib.image.AxesImage at 0x249e117e220>,
  <matplotlib.image.AxesImage at 0x249e2a88490>),
 None,
 None)
```





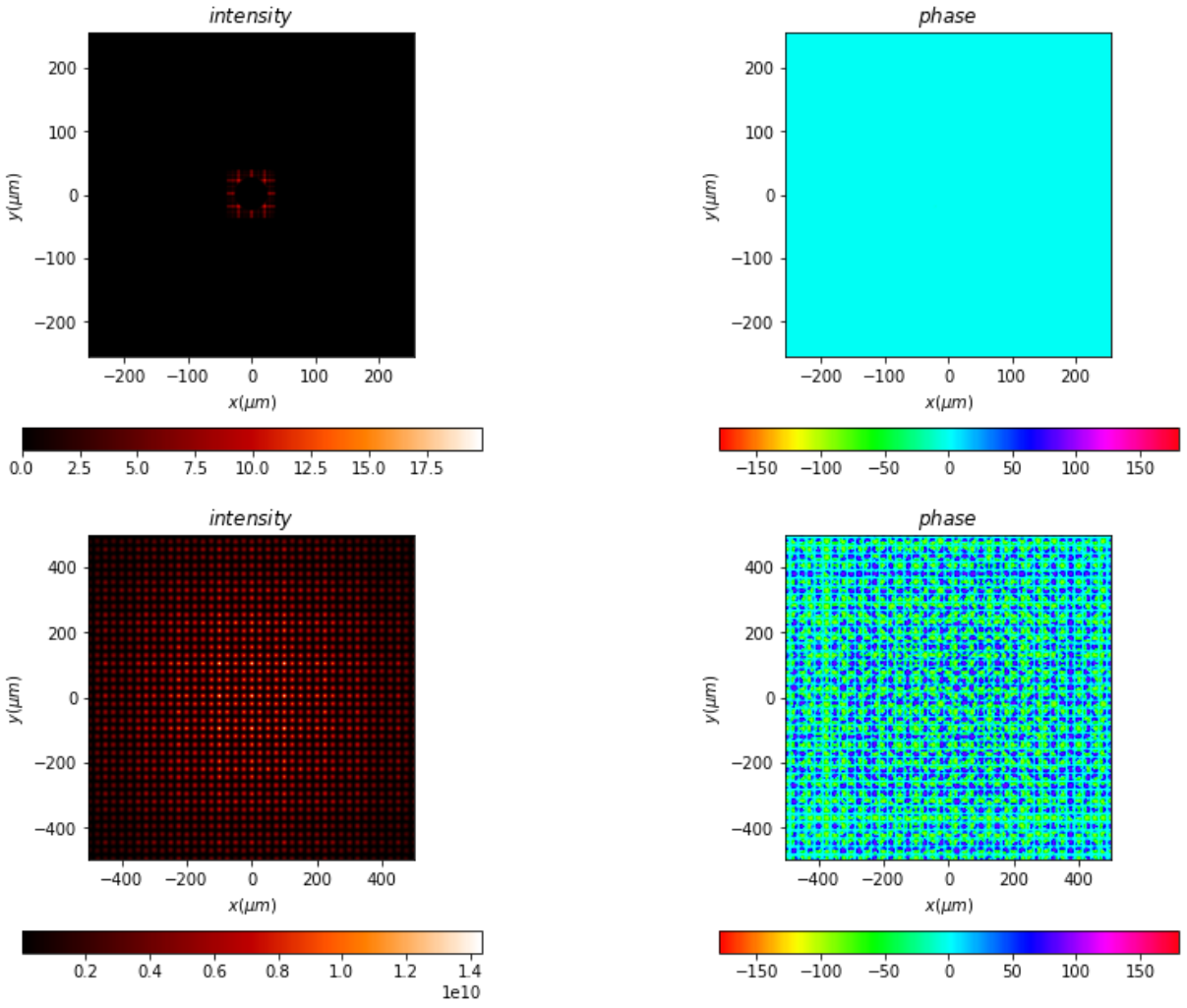




Square + Circle

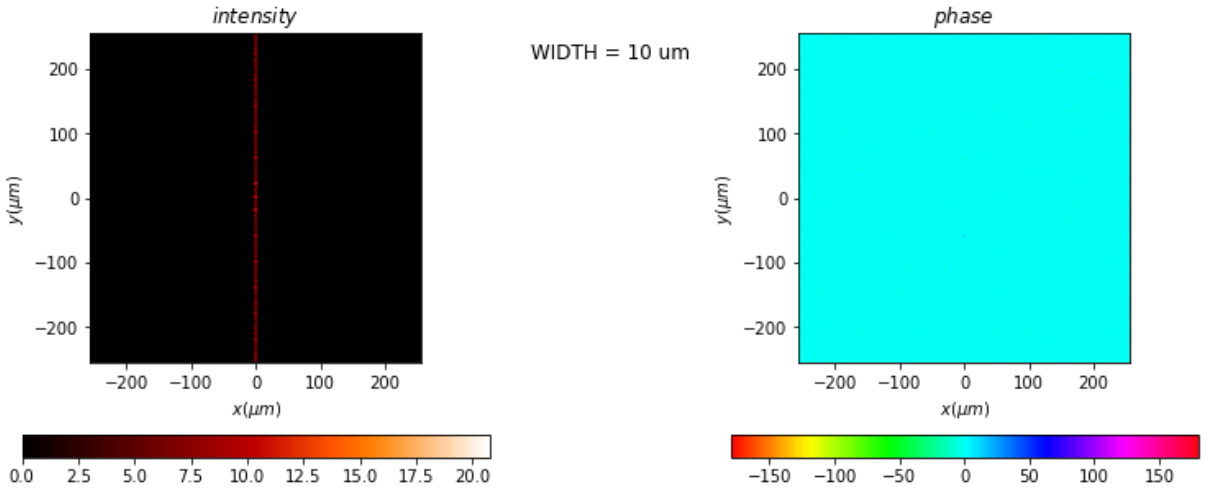
In [21]:
```
a_L1_SqCirc, a_L1_SqCirc_L2 = sim(a_L1, sqCirc)
a_L1_SqCirc.draw(kind='field', logarithm=True)
a_L1_SqCirc_L2.draw(kind='field', logarithm=False)
```

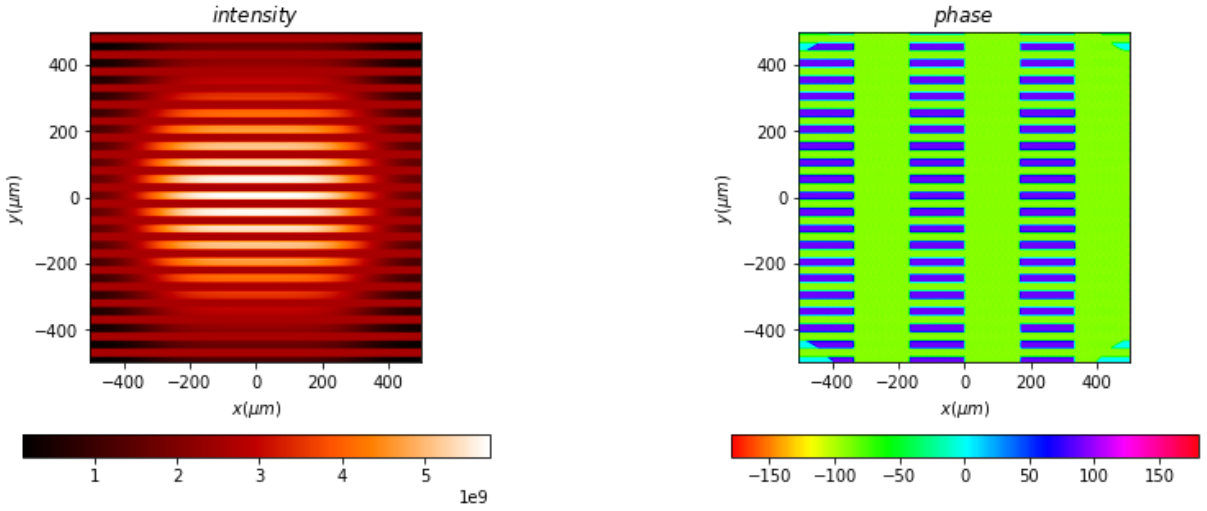
Out[21]:
```
((<matplotlib.image.AxesImage at 0x249dcfbb4f0>,
  <matplotlib.image.AxesImage at 0x249dcfc3580>),
 None,
 None)
```

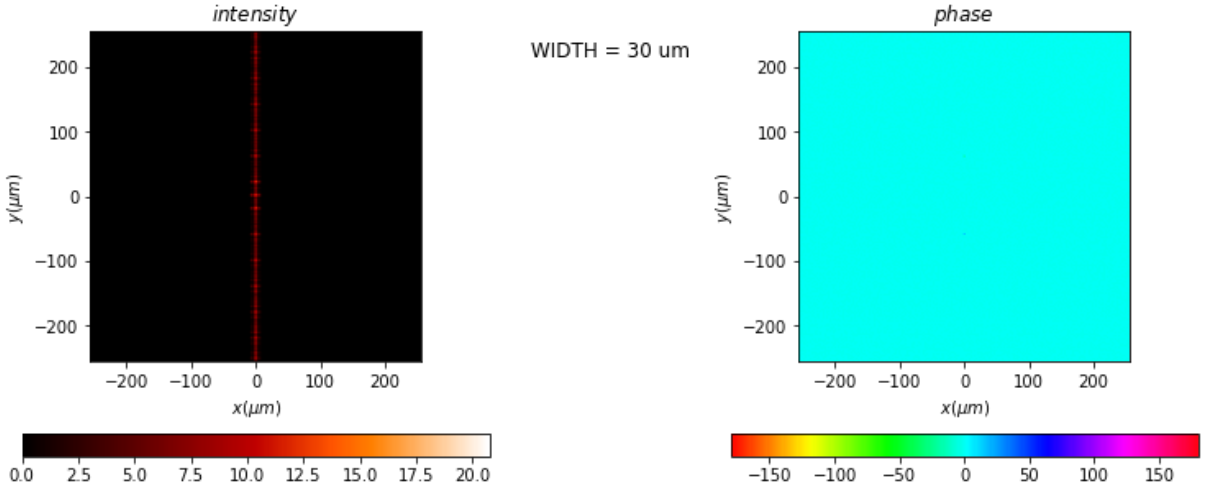


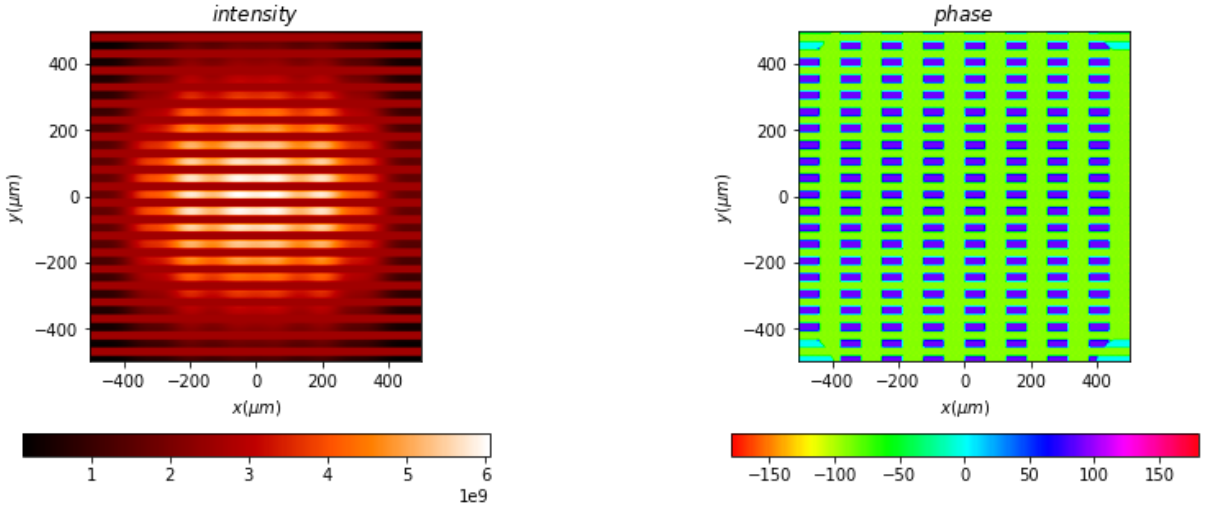
Variable Slit Widths

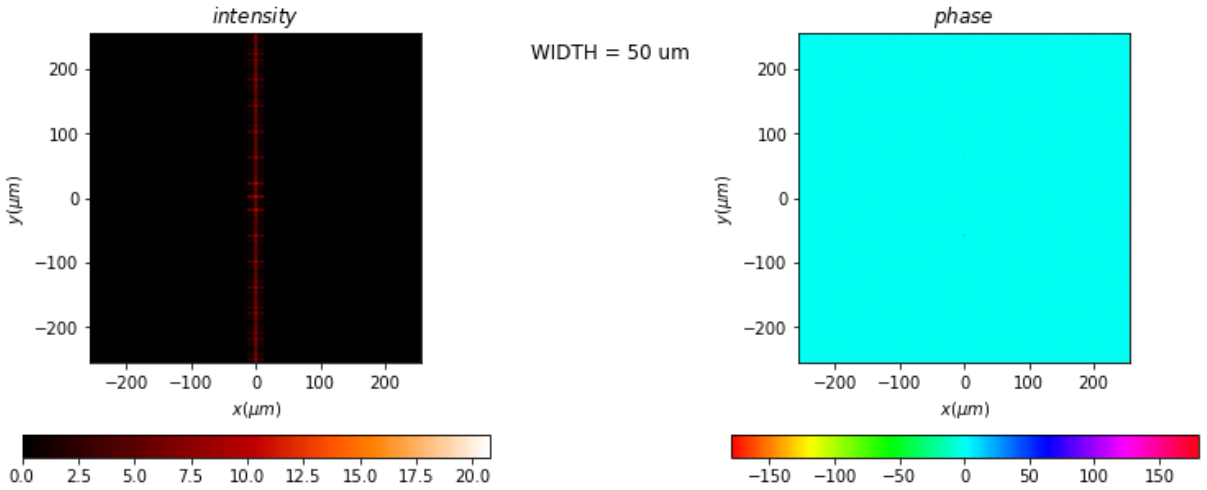
In [22]:
```
widths = [10, 30, 50, 76.3209395, 100, 400]  # in um
for width in widths:
    varSlit = Scalar_mask_XY(x0, y0, wavelength)
    varSlit.slit(
        x0=0 * um,
        size=width * um
    )
#     varSlit.draw(kind='field', logarithm=True)

    a_L1_VarSlit, a_L1_VarSlit_L2 = sim(a_L1, varSlit)
    a_L1_VarSlit.draw(title=f" WIDTH = {width} um ", kind='field', logarithm=True)
    a_L1_VarSlit_L2.draw(kind='field', logarithm=False)
```

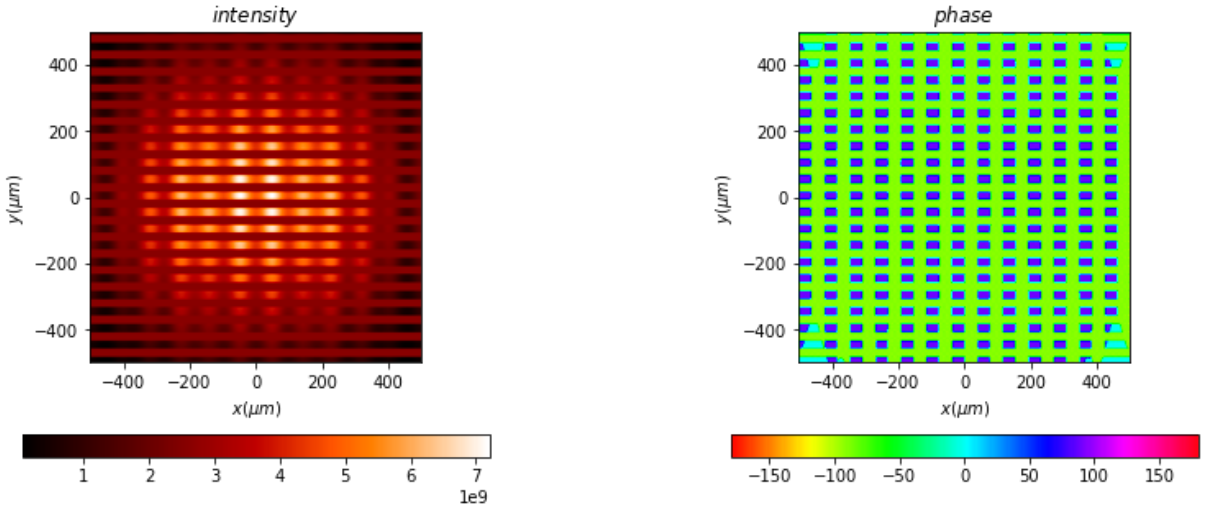

intensity
phase
WIDTH = 10 um
x(μm)
y(μm)
1e9

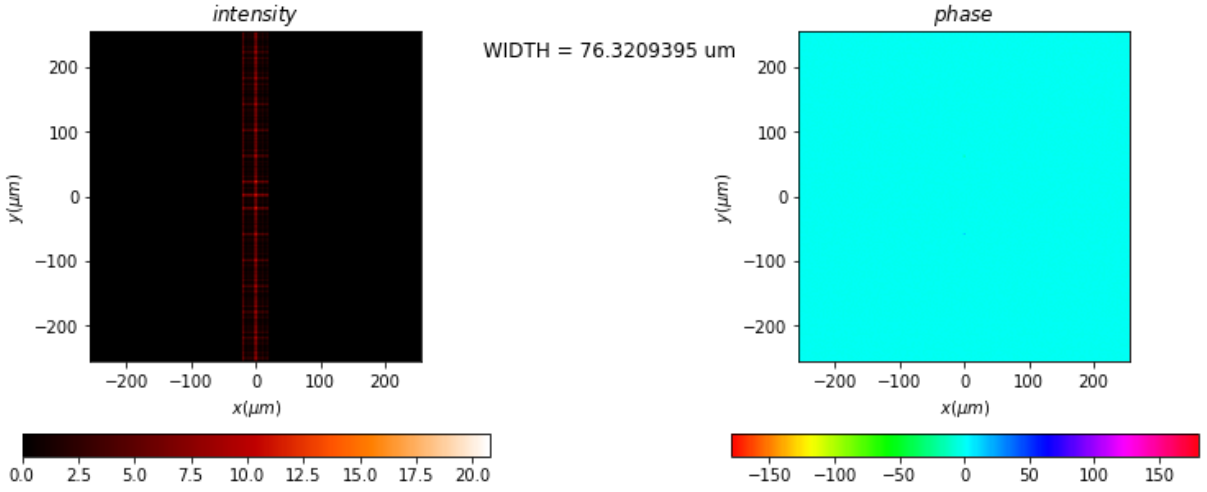

intensity
phase
x(μm)
y(μm)
1e9

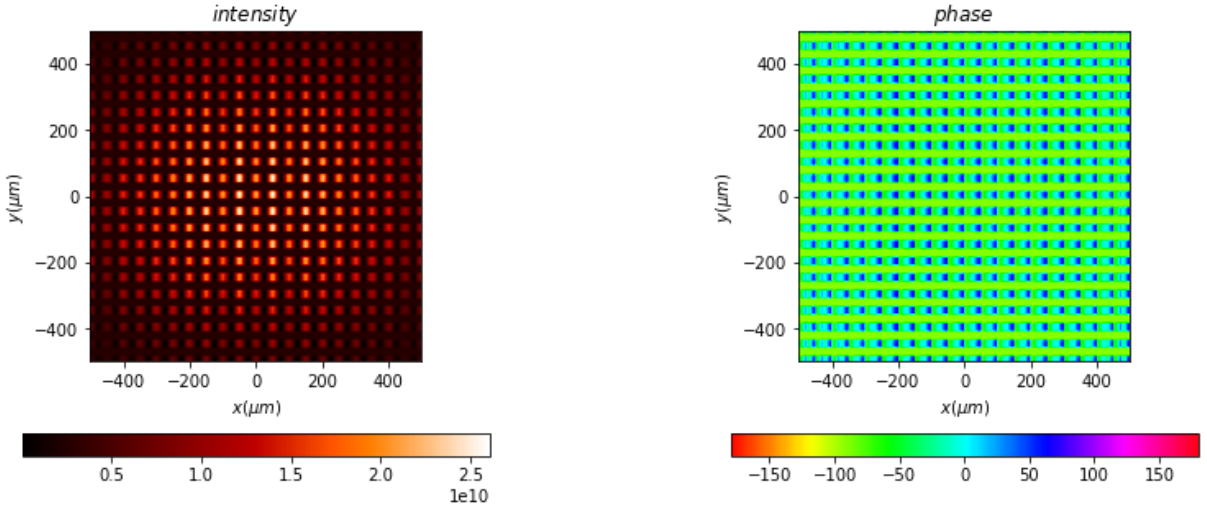

intensity
phase
WIDTH = 30 um
x(μm)
y(μm)
1e9

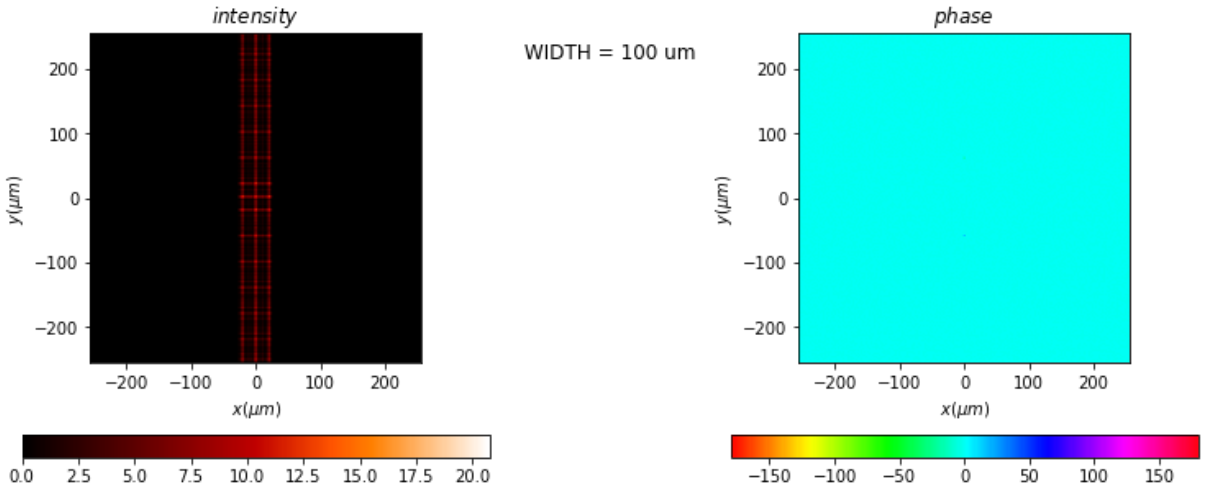

intensity
phase
x(μm)
y(μm)
1e9

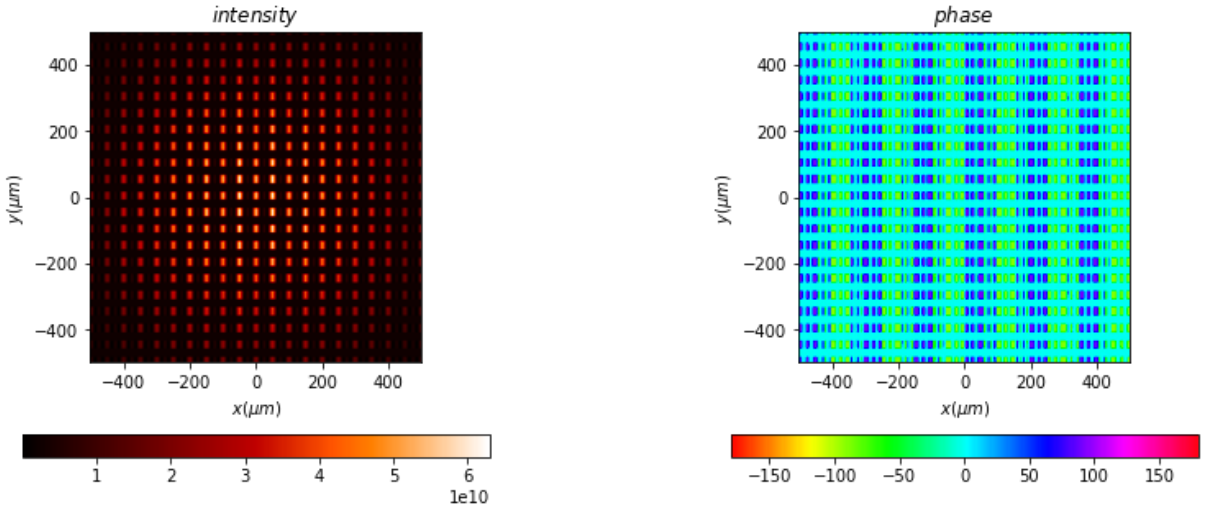

intensity
phase
WIDTH = 50 um
y(μm)
x(μm)

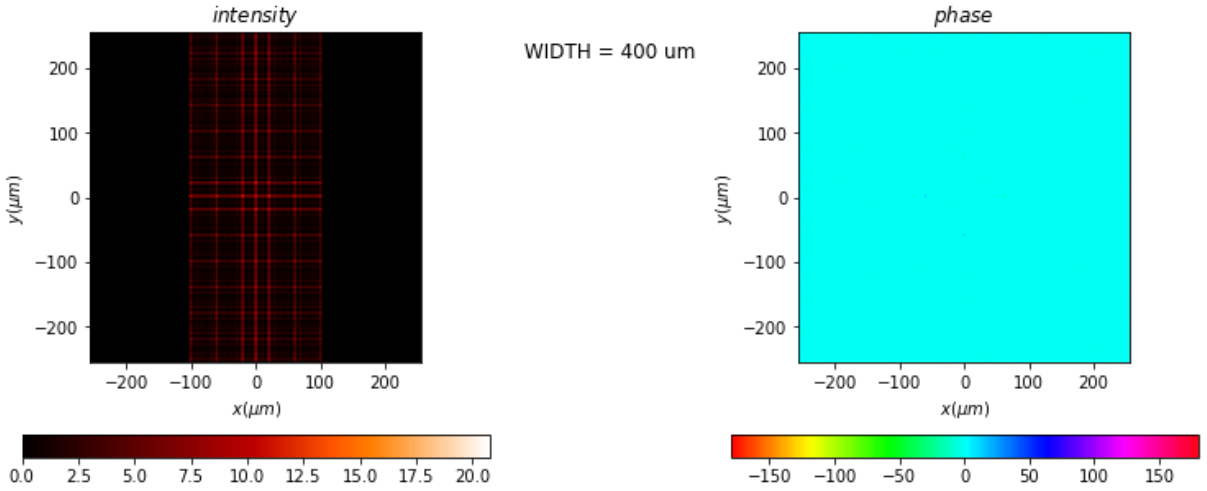

intensity
phase
y(μm)
x(μm)
1e9

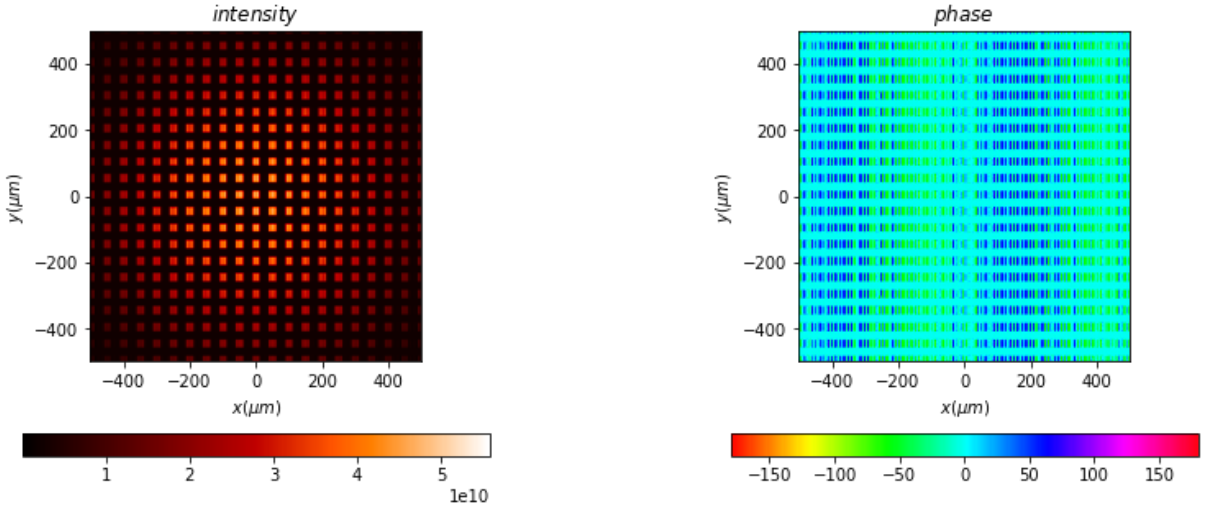

intensity
phase
WIDTH = 76.3209395 um
y(μm)
x(μm)


intensity
phase
y(μm)
x(μm)
1e10


intensity
phase
WIDTH = 100 um
y(μm)
x(μm)
1e10


intensity
phase
y(μm)
x(μm)
1e10


intensity
phase
WIDTH = 400 um
y(μm)
x(μm)
1e10


intensity
phase
y(μm)
x(μm)
1e10

Half-tone Image - Low & High-pass Filtering and Edge Enhancement / Dark-Field Illumination

Compare results with figures in Eisenkraft (1977)

In [1]:
```
import numpy as np
from diffractio import mm, um, degrees
from diffractio.scalar_sources_XY import Scalar_source_XY
from diffractio.scalar_masks_XY import Scalar_mask_XY


# Setting up
length = 1 * mm
num_data = 512
x0 = np.linspace(-length / 2, length / 2, num_data)
y0 = np.linspace(-length / 2, length / 2, num_data)
wavelength = 0.633 * um
```

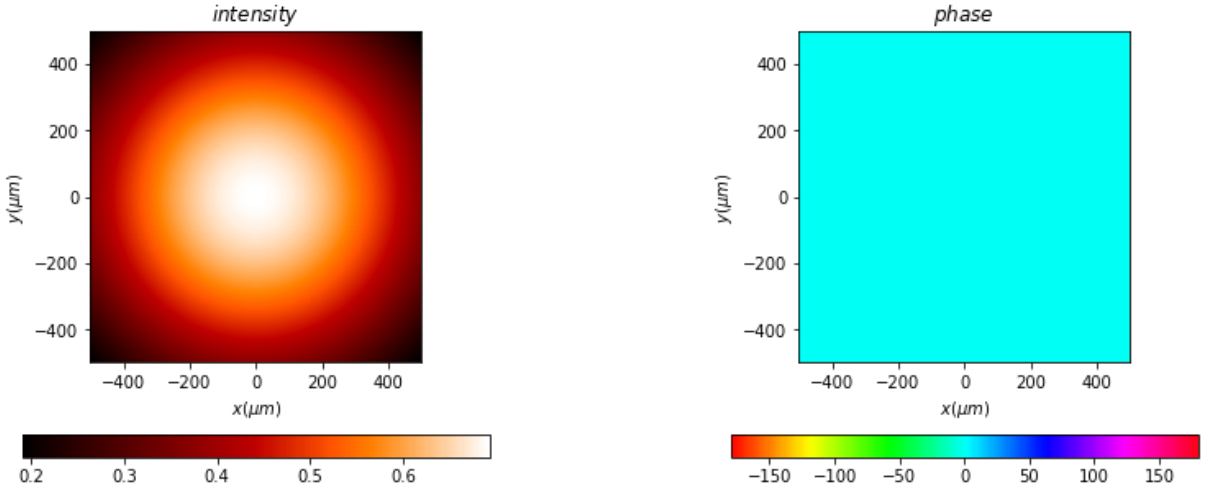
```
number of processors: 12
```

Setting up source(s)

- Gaussian Beam (LASER)
- Experiments (Either only requires modification in filter shapes)
  - Plane Wave
  - Spherical Wave

In [2]:
```
# Gaussian Beam Source - Like a LASER
u0 = Scalar_source_XY(x=x0, y=y0, wavelength=wavelength)

u0.gauss_beam(r0=(0, 0), w0=(800 * um, 800 * um), z0=0.0)
u0.draw(kind='field', logarithm=True)
```

Out[2]:
```
((<matplotlib.image.AxesImage at 0x2c0a2a9daf0>,
  <matplotlib.image.AxesImage at 0x2c0a2d3d2b0>),
 None,
 None)
```



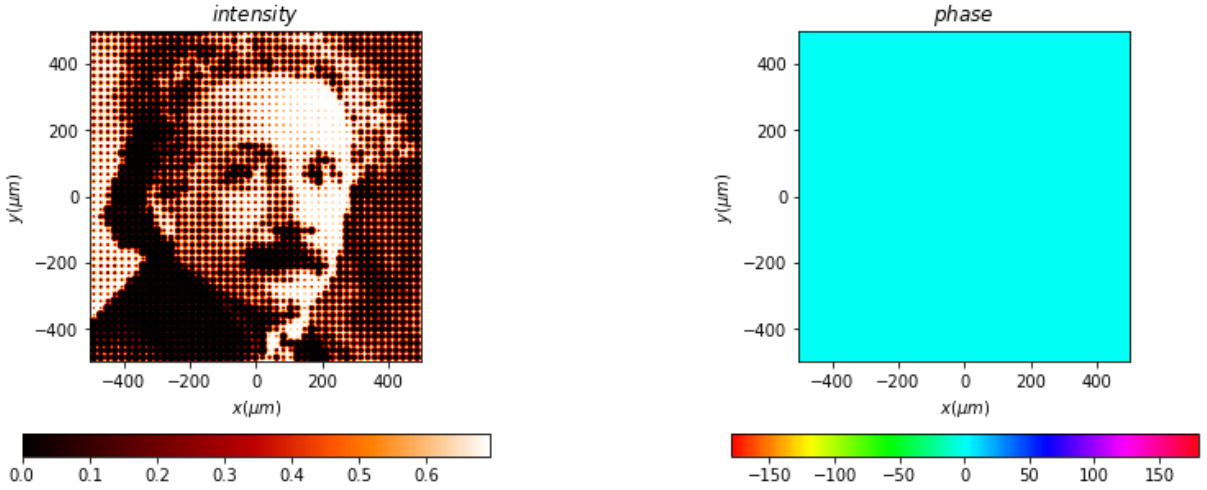
In [3]:
```
# # Plane Wave Source
# u1 = Scalar_source_XY(x=x0, y=y0, wavelength=wavelength)
# u1.plane_wave()
# u1.draw(kind='field', logarithm=True)
```

In [4]:
```
# # Spherical Wave Source
# u2 = Scalar_source_XY(x=x0, y=y0, wavelength=wavelength)
# u2.spherical_wave(z0=1, mask=False)
# u2.draw(kind='field', logarithm=True)
```

Image

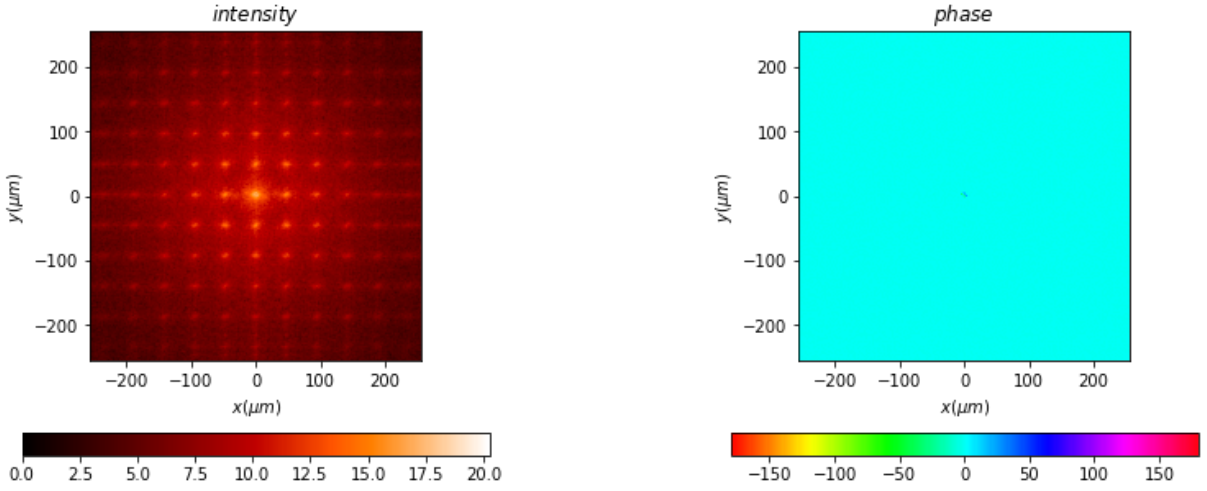
In [5]:
```
y15 = Scalar_mask_XY(x0, y0, wavelength)
y15.image(
    filename="y-15-HT.png",
    normalize=True,
    canal=2,
)

y15.draw(kind='field', logarithm=True)
```

Out[5]:
```
((<matplotlib.image.AxesImage at 0x2c0a313e520>,
  <matplotlib.image.AxesImage at 0x2c0a37bfca0>),
 None,
 None)
```



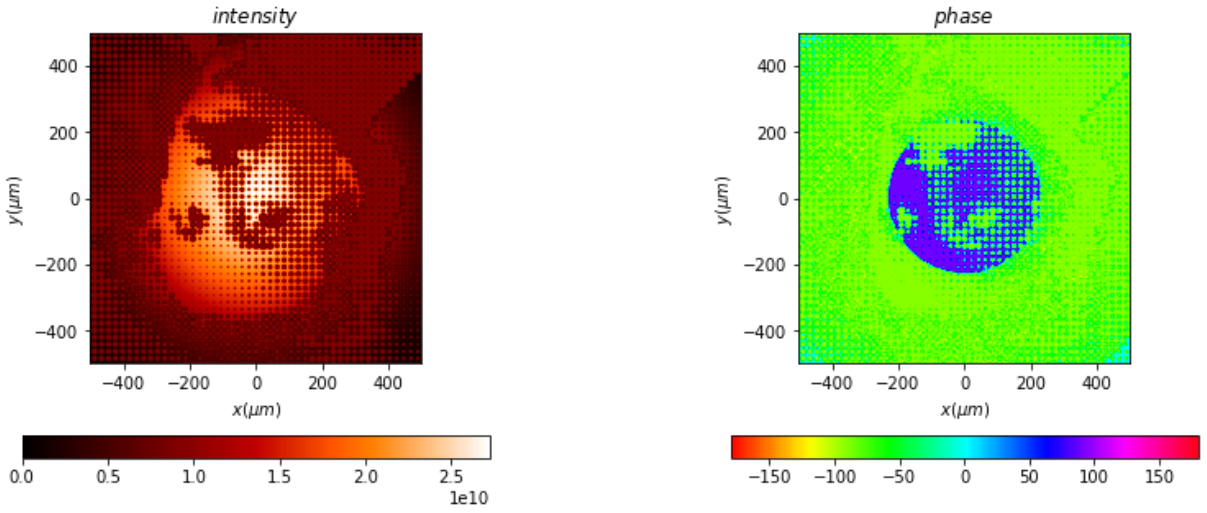
In [6]:
```
# Fourier Plane - No Filter
a_L1 = (u0 * y15).fft(z=1 * mm, new_field=True)
a_L1.draw(kind='field', logarithm=True)
```

Out[6]:
```
((<matplotlib.image.AxesImage at 0x2c0a384fc40>,
  <matplotlib.image.AxesImage at 0x2c0a3922430>),
 None,
 None)
```



In [7]:
```
# This is how, it'd look without any filter, at the Observation screen
a_L2 = a_L1.fft(z=1 * mm, shift=False, remove0=False, new_field=True)
a_L2.draw(kind='field', logarithm=False)
```

Out[7]:
```
((<matplotlib.image.AxesImage at 0x2c0a3e44310>,
  <matplotlib.image.AxesImage at 0x2c0a3e93ac0>),
 None,
 None)
```



Masks / Filters

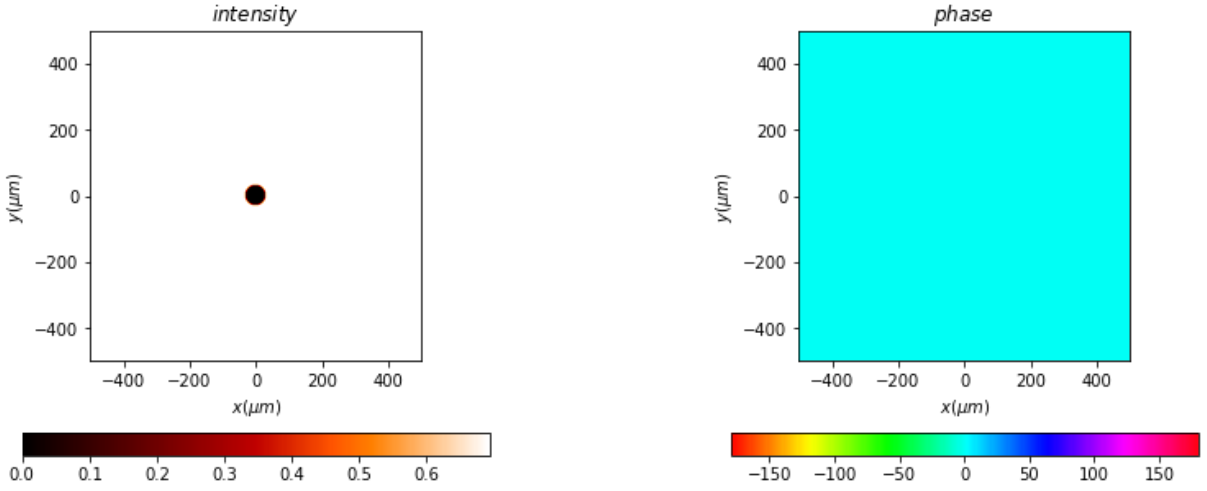
- Center Dot - High Pass
- Square - Low Pass
- Square + Center Dot -

- Vertical Slit

- Horizontal Slit
- Angled Slit

In [8]:
```
cDot = Scalar_mask_XY(x0, y0, wavelength)
cDot.ring(
    r0=(0 * um, 0 * um),
    radius1=(30 * um, 30 * um),
    radius2=(1000 * um, 1000 * um)
)
cDot.draw(kind='field', logarithm=True)
```

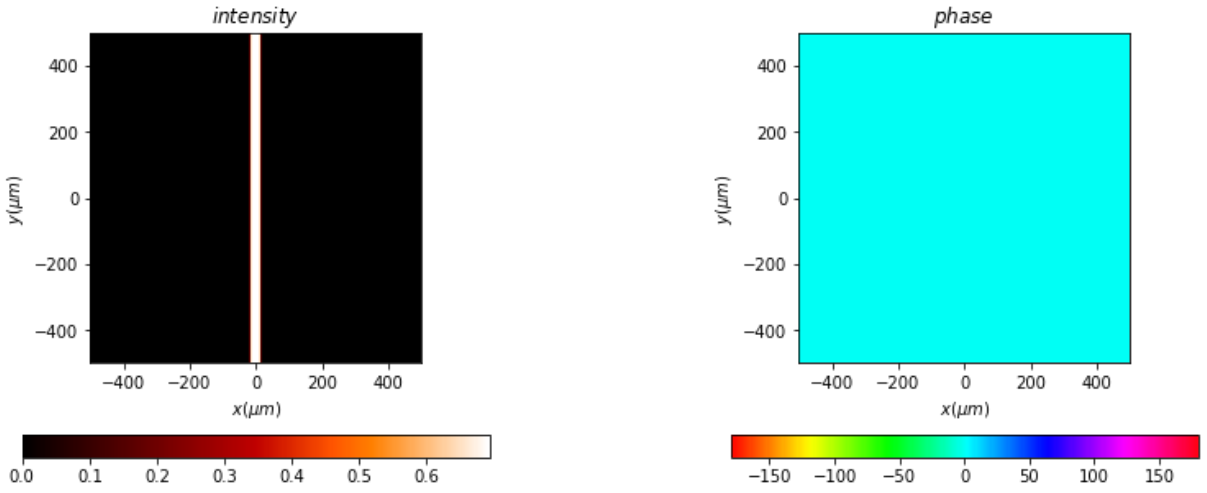
Out[8]:
```
((<matplotlib.image.AxesImage at 0x2c0a7d6dc40>,
  <matplotlib.image.AxesImage at 0x2c0a7e04430>),
 None,
 None)
```



In [9]:
```
vert = Scalar_mask_XY(x0, y0, wavelength)
vert.slit(
    x0=0 * um,
    size=30 * um
)
vert.draw(kind='field', logarithm=True)
```

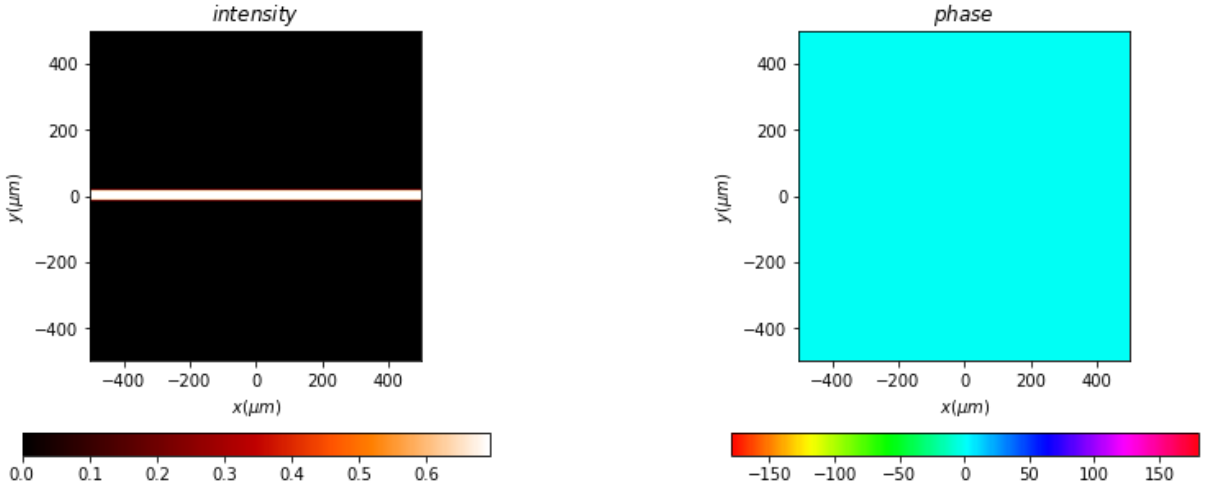
Out[9]:
```
((<matplotlib.image.AxesImage at 0x2c0a7d2fb80>,
```

```
 <matplotlib.image.AxesImage at 0x2c0a30a7970>),
None,
None)
```



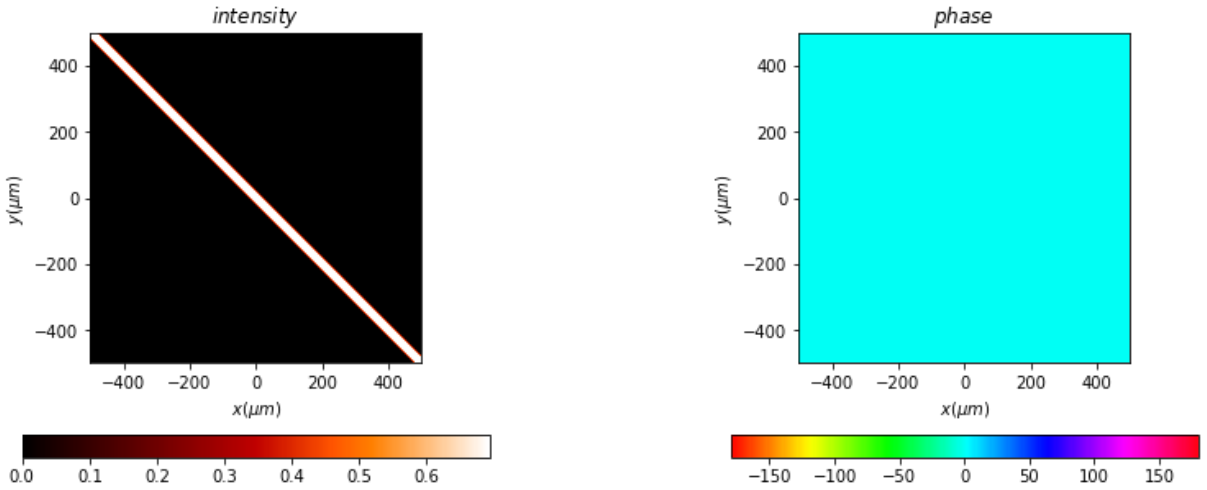
In [10]:
```
horiz = Scalar_mask_XY(x0, y0, wavelength)
horiz.slit(
    x0=0 * um,
    size=30 * um,
    angle=np.pi / 2
)
horiz.draw(kind='field', logarithm=True)
```

Out[10]:
```
((<matplotlib.image.AxesImage at 0x2c0a8de2730>,
  <matplotlib.image.AxesImage at 0x2c0a8e6ceb0>),
None,
None)
```



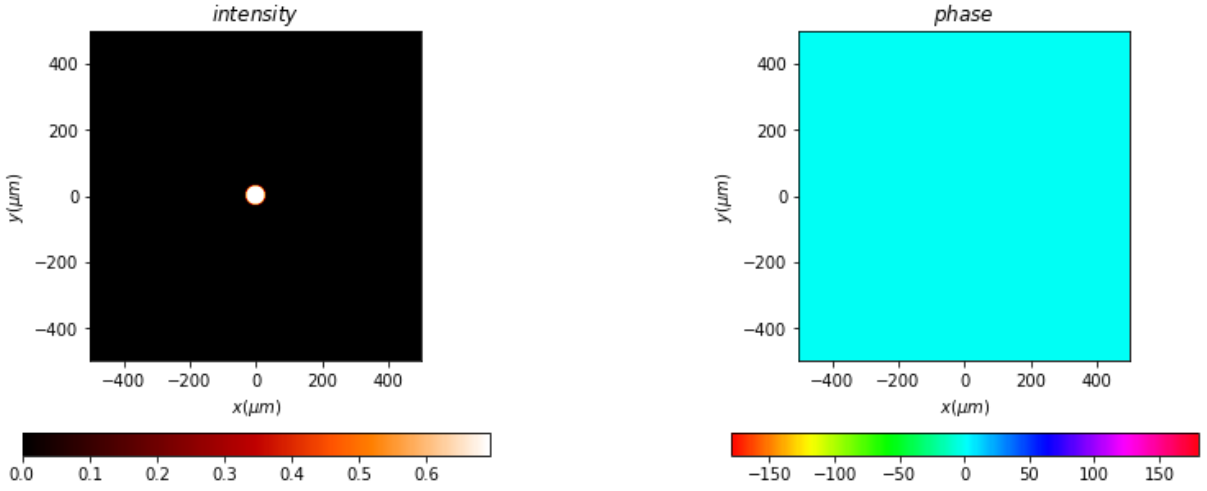
In [11]:
```
angled = Scalar_mask_XY(x0, y0, wavelength)
angled.slit(
    x0=0 * um,
    size=30 * um,
    angle=np.pi / 4
)
angled.draw(kind='field', logarithm=True)
```

Out[11]:
```
((<matplotlib.image.AxesImage at 0x2c0aa1de160>,
  <matplotlib.image.AxesImage at 0x2c0aa22e940>),
None,
None)
```



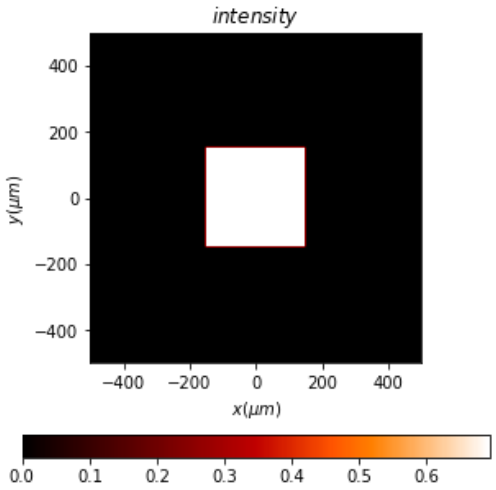
In [12]:
```
circ = Scalar_mask_XY(x0, y0, wavelength)
circ.circle(
    r0=(0 * um, 0 * um),
    radius=(30 * um, 30 * um),
    angle=0
)
circ.draw(kind='field', logarithm=True)
```

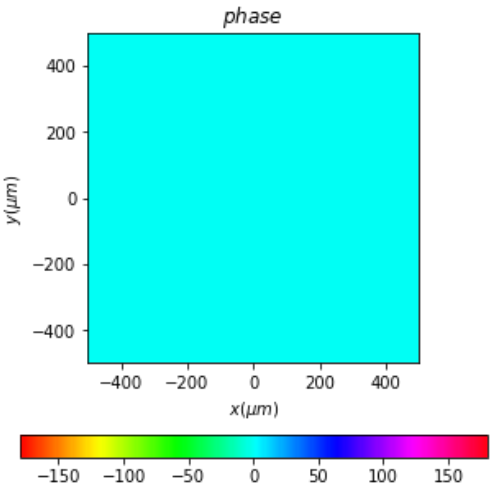
Out[12]:
```
((<matplotlib.image.AxesImage at 0x2c0aa304d00>,
  <matplotlib.image.AxesImage at 0x2c0aa372880>),
 None,
 None)
```



In [13]:
```
sq = Scalar_mask_XY(x0, y0, wavelength)
sq.square(
    r0=(0 * um, 0 * um),
    size=(300 * um, 300 * um),
    angle=0,
)
sq.draw(kind='field', logarithm=True)
```

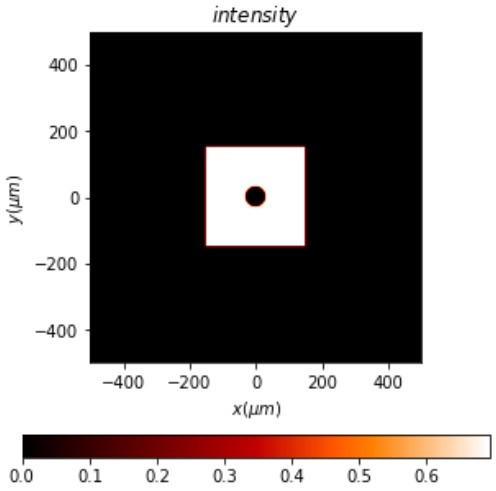
Out[13]:
```
((<matplotlib.image.AxesImage at 0x2c0aae98c10>,
  <matplotlib.image.AxesImage at 0x2c0aaefde80>),
 None,
 None)
```

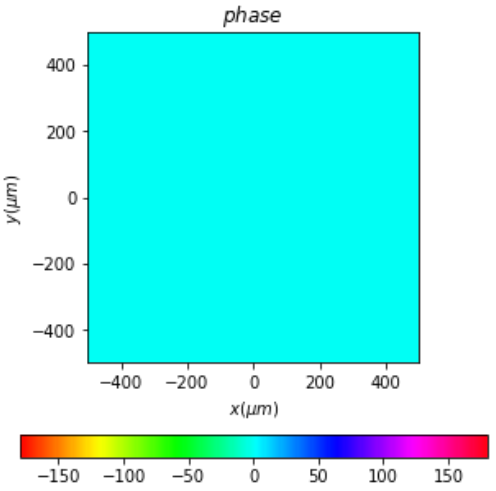




In [14]:
```
# Adding two filters
sqCirc = sq * cDot
sqCirc.draw(kind="field", logarithm=True)
```

Out[14]:
```
((<matplotlib.image.AxesImage at 0x2c0aae94a60>,
  <matplotlib.image.AxesImage at 0x2c0a8dc96a0>),
 None,
 None)
```





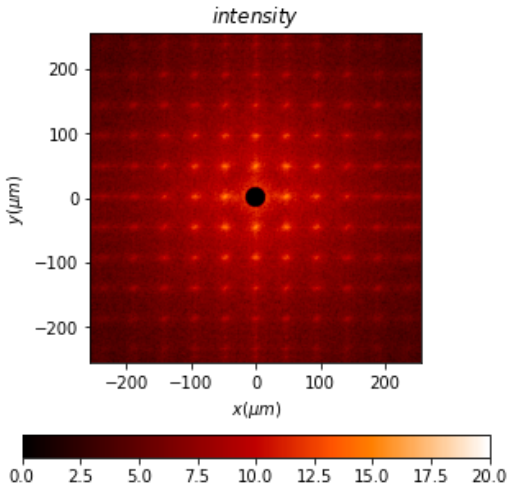
Returns wave after encountering filter and then L2

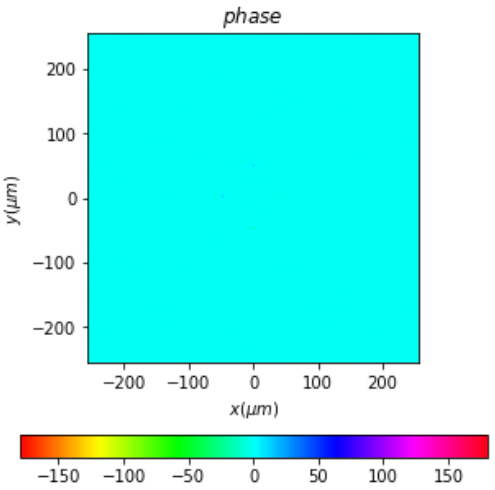
In [15]:
```
def sim(a_L1, mask):
    """
    a_L1: Wave after passing through Lens 1
    mask: Mask or Filter to apply

    """
    a_L1_Mask = a_L1 * mask
    a_L1_Mask_L2 = a_L1_Mask.fft(z=1 * mm, shift=False, remove0=False, new_field=True)

    return a_L1_Mask, a_L1_Mask_L2
```

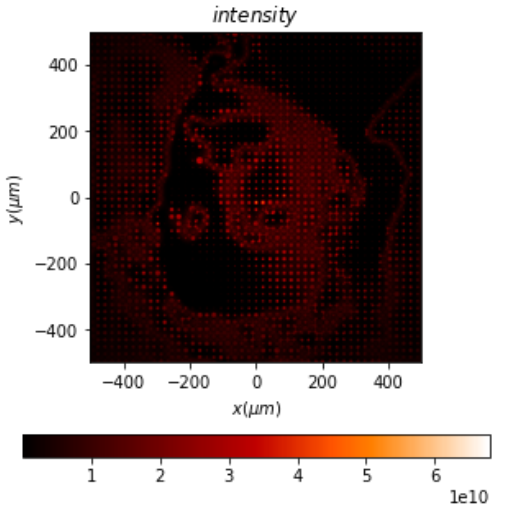
Center Dot - High Pass - HT Structure Visible - EDGE DETECTION

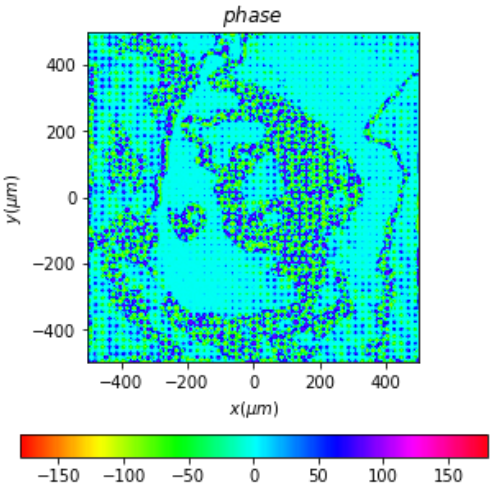
In [16]:
```
a_L1_cD, a_L1_cD_L2 = sim(a_L1, cDot)
a_L1_cD.draw(kind='field', logarithm=True)
a_L1_cD_L2.draw(kind='field', logarithm=False)
```

Out[16]:
```
((<matplotlib.image.AxesImage at 0x2c0ac627f10>,
  <matplotlib.image.AxesImage at 0x2c0ac69d700>),
 None,
 None)
```





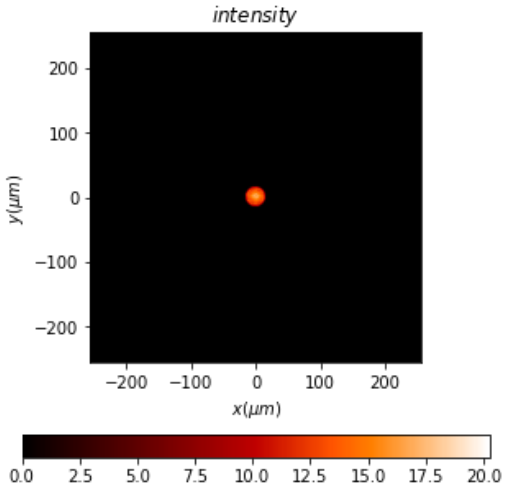


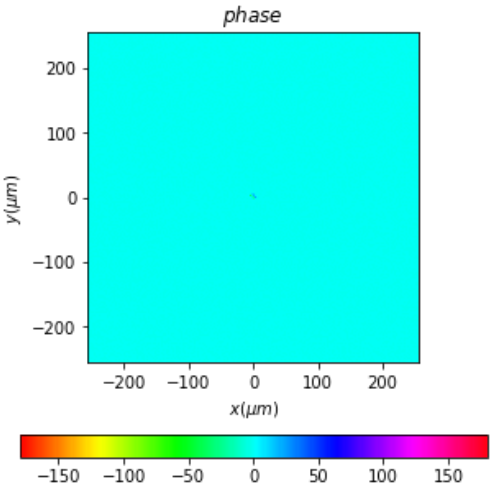


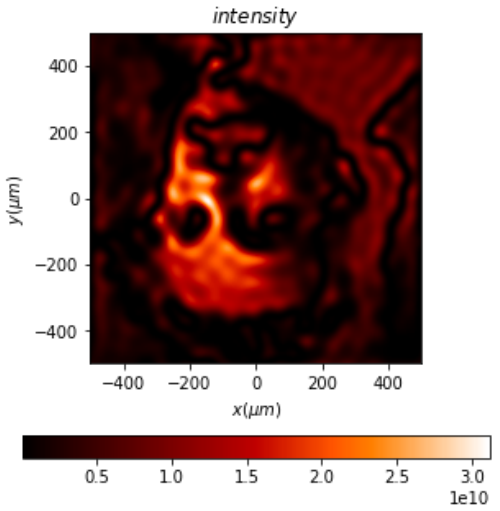
Circle - Low Pass - Blurred Output

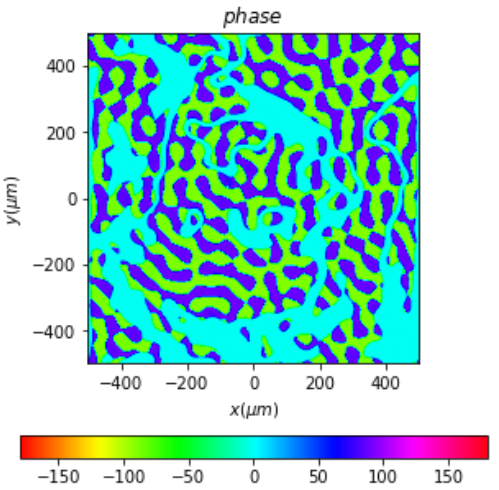
In [17]:
```
a_L1_Circ, a_L1_Circ_L2 = sim(a_L1, circ)
a_L1_Circ.draw(kind='field', logarithm=True)
a_L1_Circ_L2.draw(kind='field', logarithm=False)
```

Out[17]:
```
((<matplotlib.image.AxesImage at 0x2c0ae2eedf0>,
  <matplotlib.image.AxesImage at 0x2c0b1ef8130>),
 None,
 None)
```





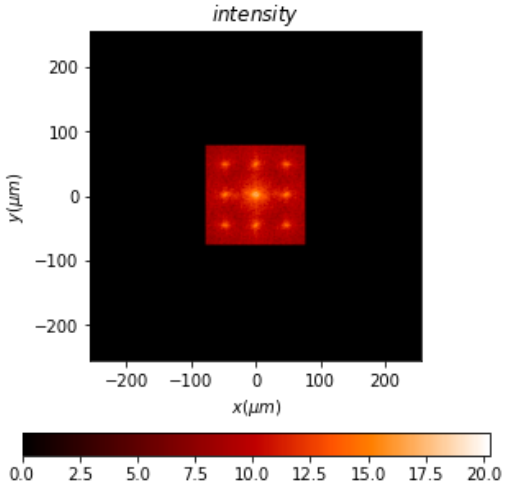


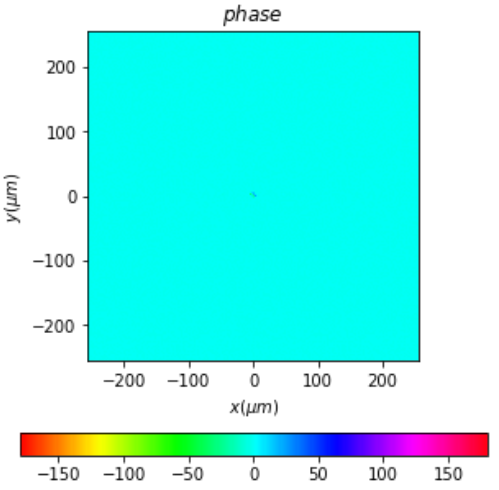


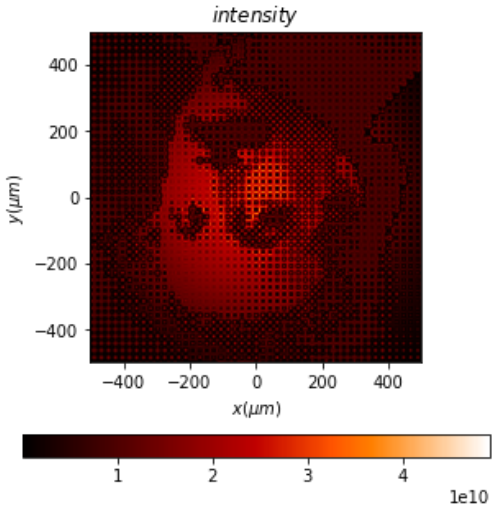
Square

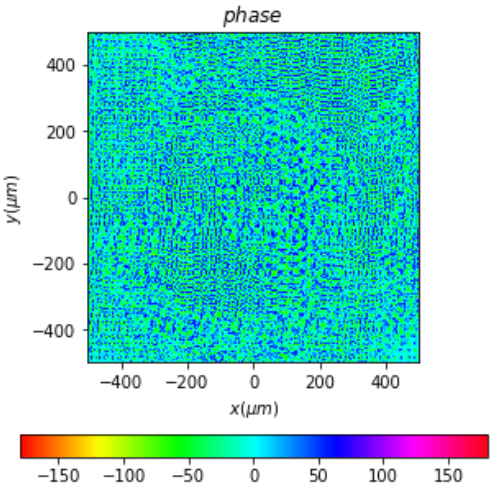
In [18]:
```
a_L1_Sq, a_L1_Sq_L2 = sim(a_L1, sq)
a_L1_Sq.draw(kind='field', logarithm=True)
a_L1_Sq_L2.draw(kind='field', logarithm=False)
```

Out[18]:
```
((<matplotlib.image.AxesImage at 0x2c0b1c24a90>,
  <matplotlib.image.AxesImage at 0x2c0b1a80fd0>),
 None,
 None)
```





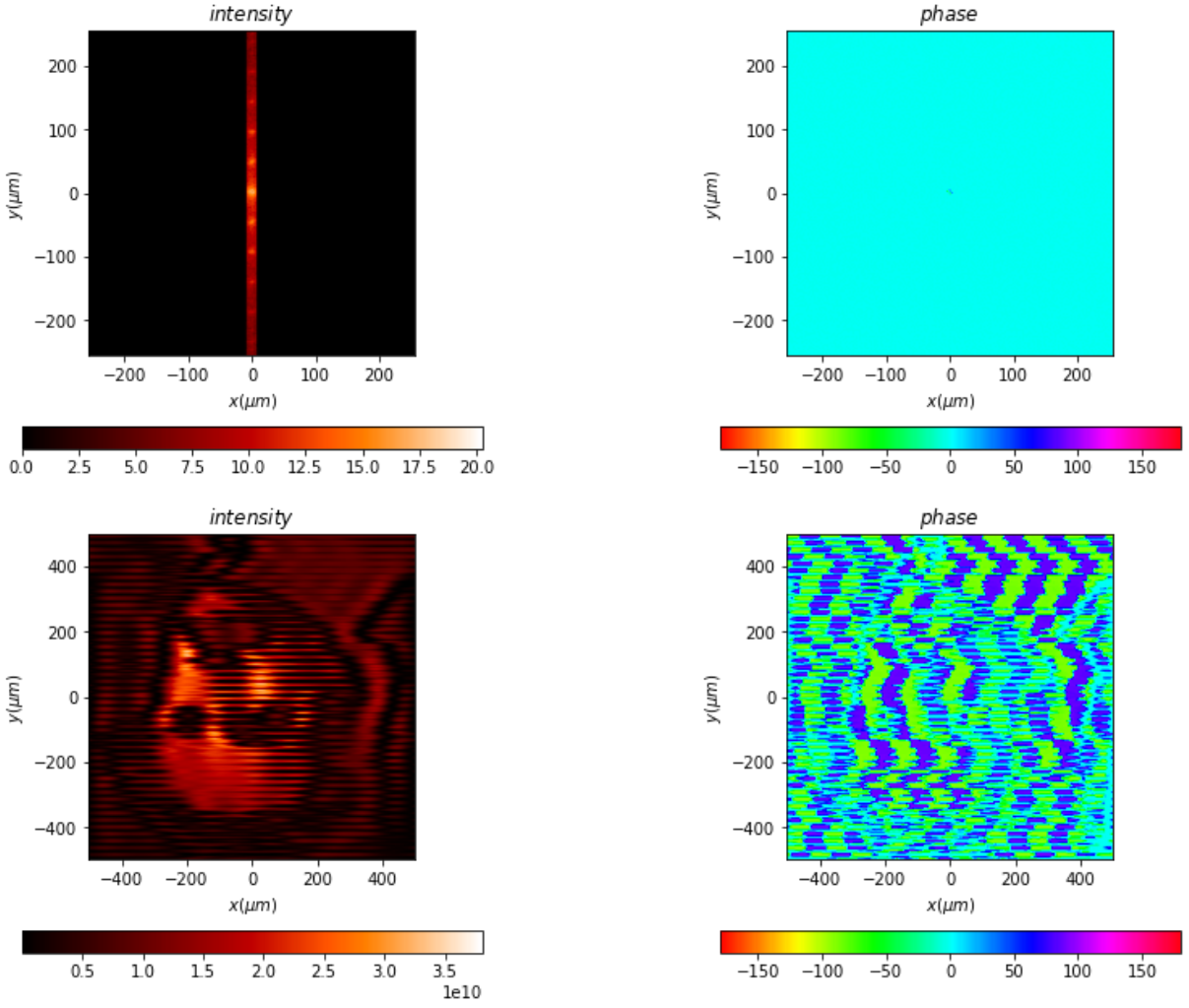




Vertical Slit

In [19]:
```
a_L1_Vert, a_L1_Vert_L2 = sim(a_L1, vert)
a_L1_Vert.draw(kind='field', logarithm=True)
a_L1_Vert_L2.draw(kind='field', logarithm=False)
```

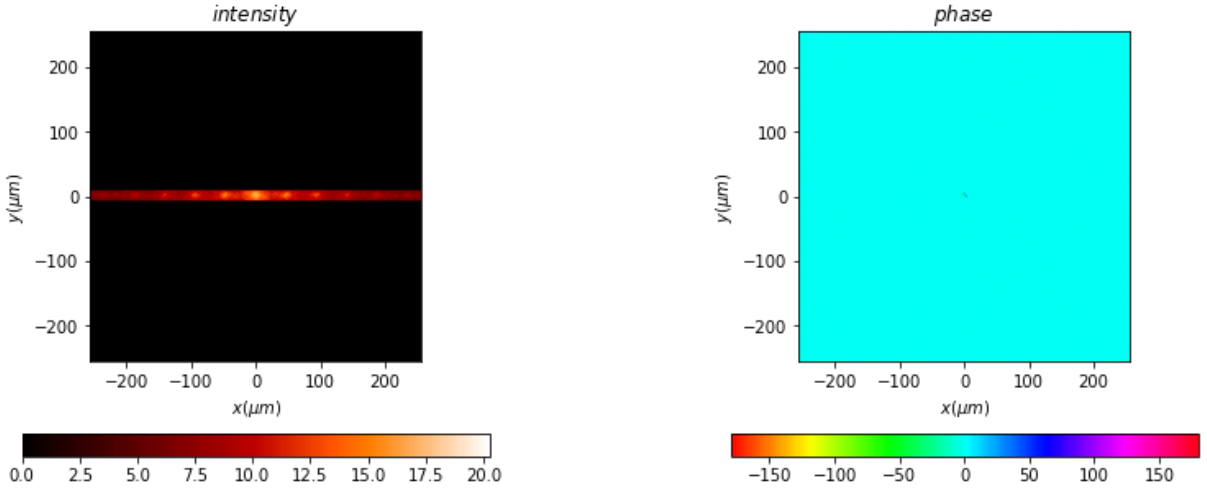
Out[19]:
```
((<matplotlib.image.AxesImage at 0x2c0ac560be0>,
  <matplotlib.image.AxesImage at 0x2c0ae248880>),
 None,
 None)
```

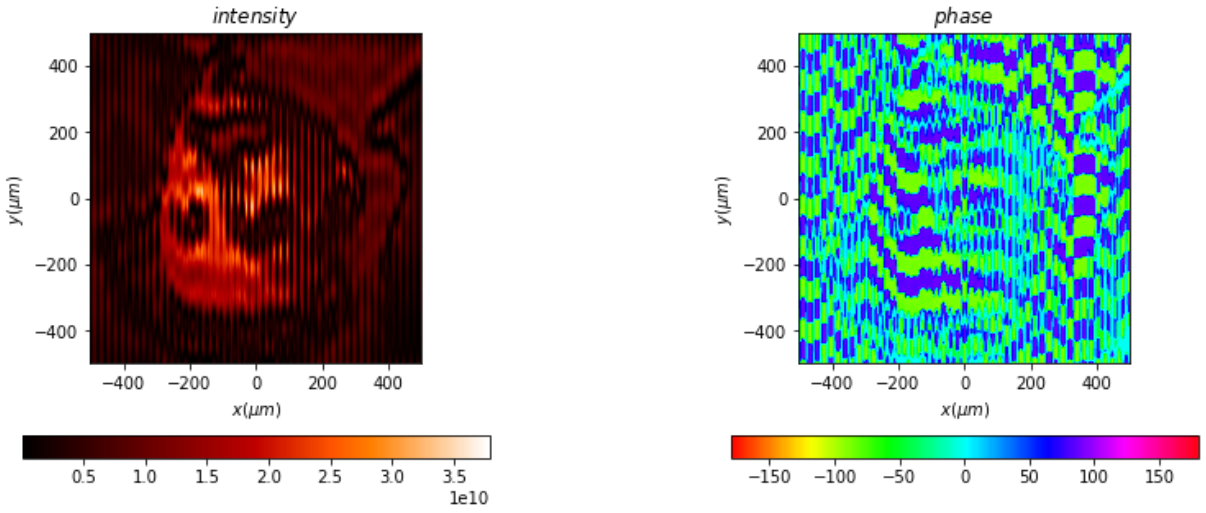


Horizontal Slit

In [20]:
```
a_L1_Horiz, a_L1_Horiz_L2 = sim(a_L1, horiz)
a_L1_Horiz.draw(kind='field', logarithm=True)
a_L1_Horiz_L2.draw(kind='field', logarithm=False)
```

Out[20]:
```
((<matplotlib.image.AxesImage at 0x2c0b48461f0>,
  <matplotlib.image.AxesImage at 0x2c0b48af460>),
 None,
 None)
```

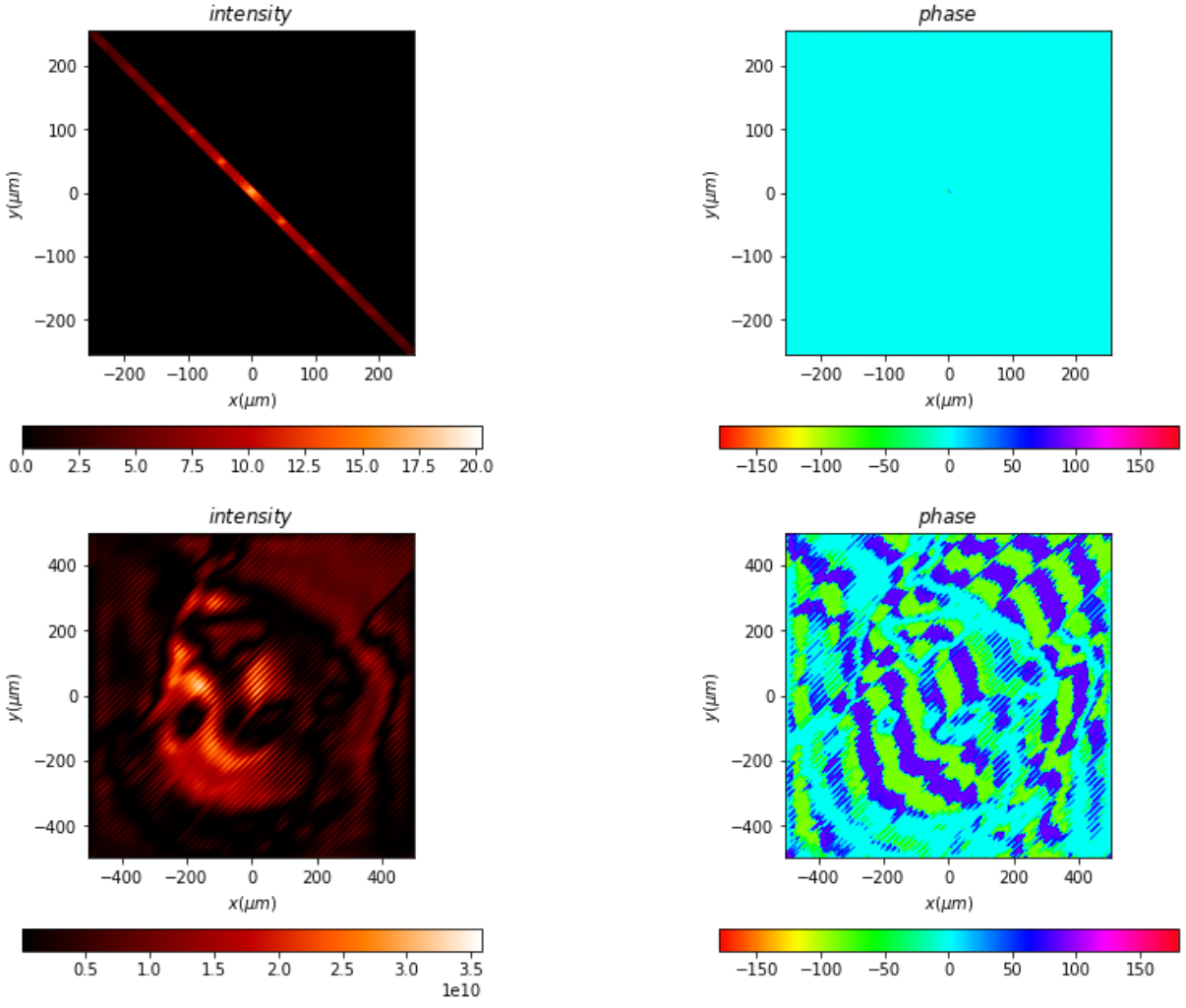




Angled Slit

In [21]:
```
a_L1_Angled, a_L1_Angled_L2 = sim(a_L1, angled)
a_L1_Angled.draw(kind='field', logarithm=True)
a_L1_Angled_L2.draw(kind='field', logarithm=False)
```

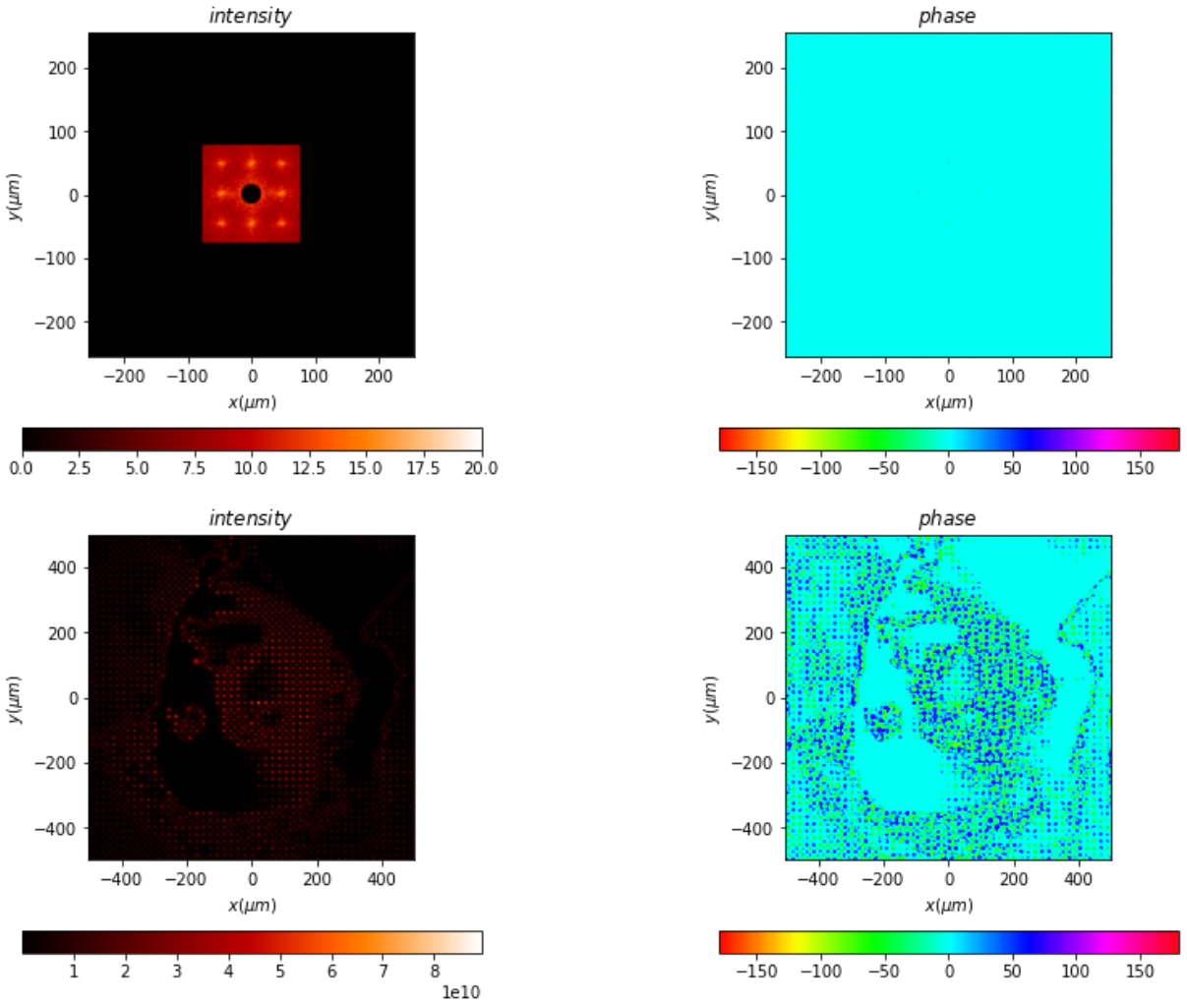
Out[21]:
```
((<matplotlib.image.AxesImage at 0x2c0b4f0e6a0>,
  <matplotlib.image.AxesImage at 0x2c0b7a1a910>),
 None,
 None)
```



Square + Circle - Dark-Field Illumination / Edge Enhancement

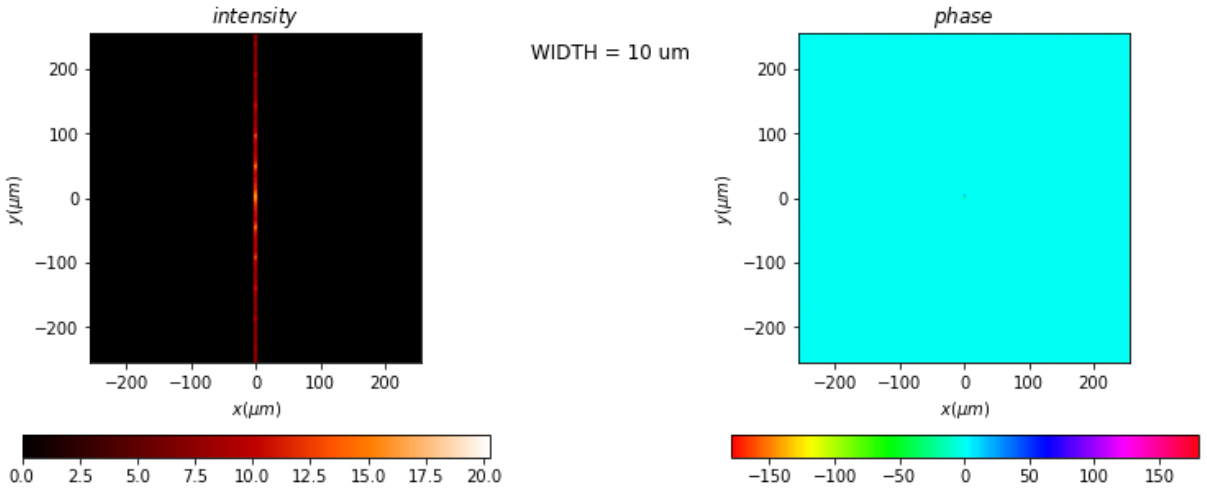
In [22]:
```
a_L1_SqCirc, a_L1_SqCirc_L2 = sim(a_L1, sqCirc)
a_L1_SqCirc.draw(kind='field', logarithm=True)
a_L1_SqCirc_L2.draw(kind='field', logarithm=False)
```

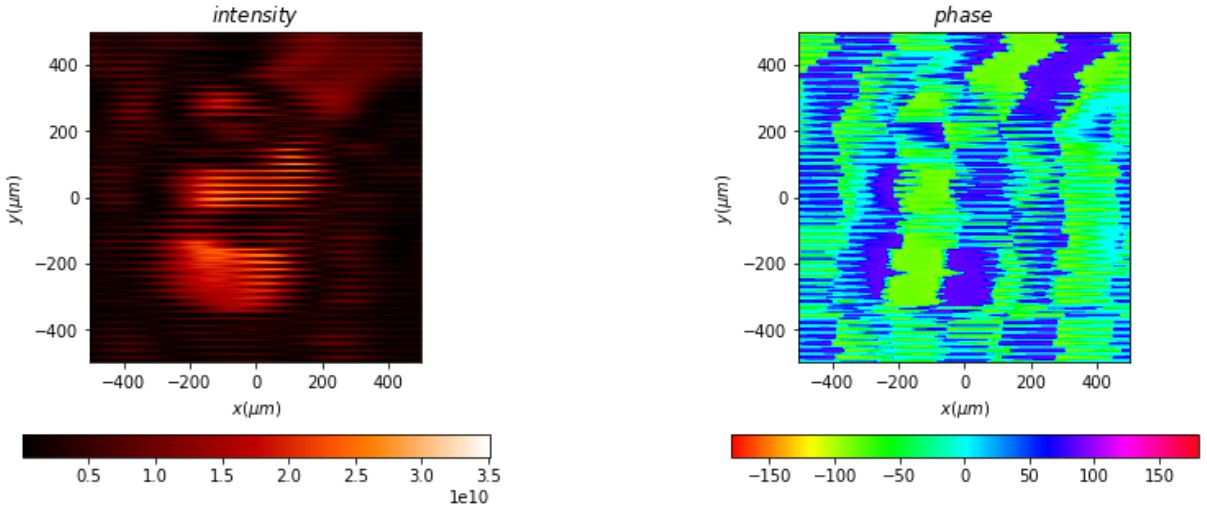
Out[22]:
```
((<matplotlib.image.AxesImage at 0x2c0ac53f4f0>,
  <matplotlib.image.AxesImage at 0x2c0b1a87fa0>),
 None,
 None)
```

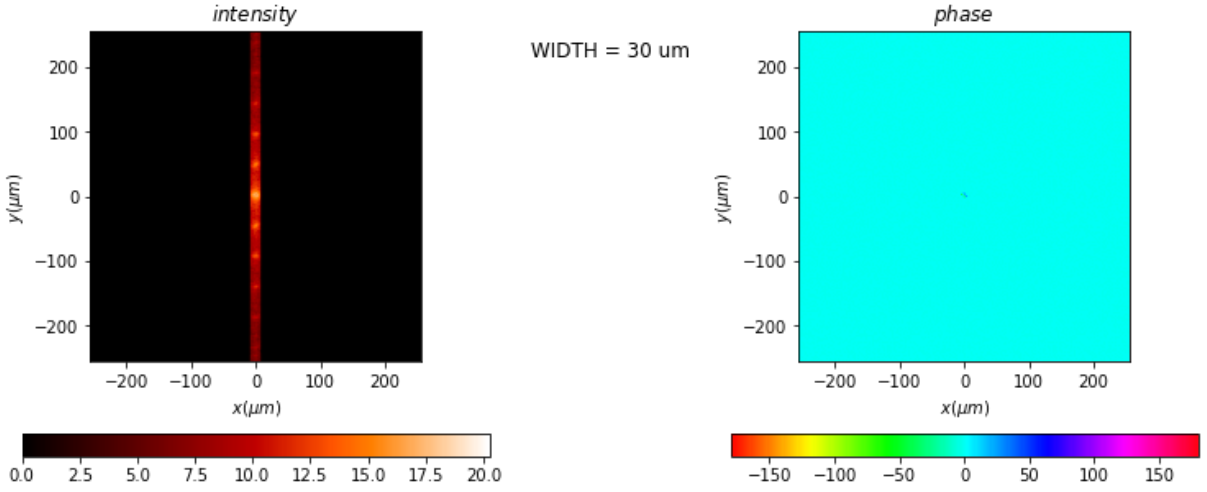


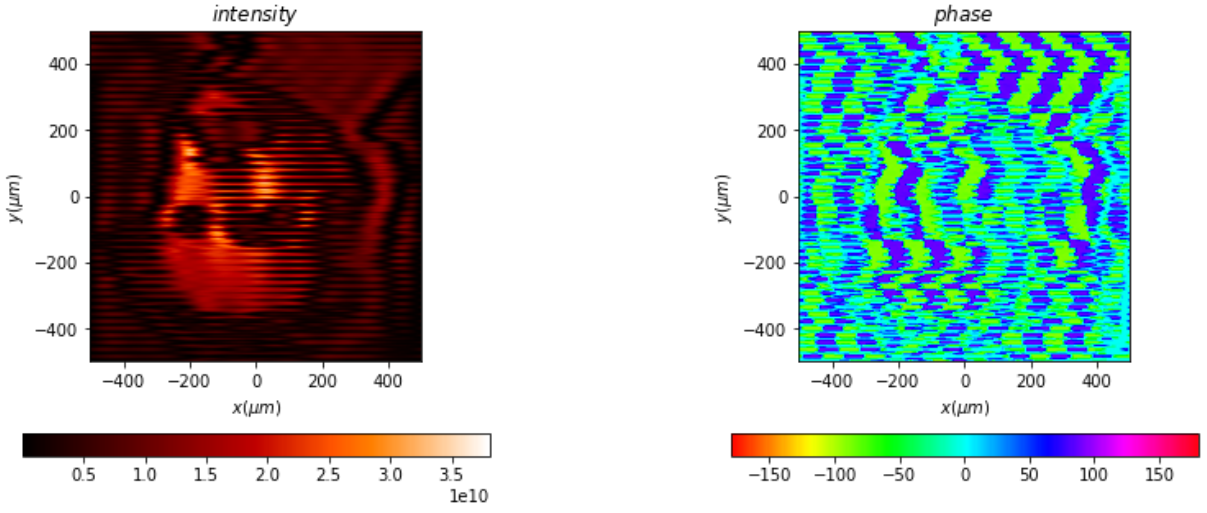
Variable Slit Widths

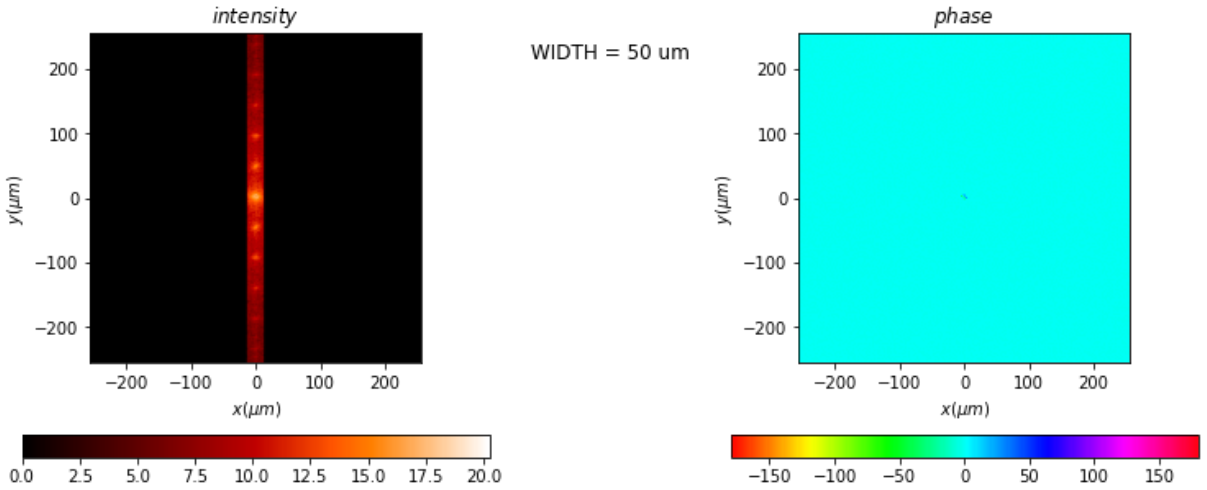
In [23]:
```
widths = [10, 30, 50, 100, 400]  # in um
for width in widths:
    varSlit = Scalar_mask_XY(x0, y0, wavelength)
    varSlit.slit(
        x0=0 * um,
        size=width * um
    )
#     varSlit.draw(kind='field', logarithm=True)

    a_L1_VarSlit, a_L1_VarSlit_L2 = sim(a_L1, varSlit)
    a_L1_VarSlit.draw(title=f" WIDTH = {width} um ", kind='field', logarithm=True)
    a_L1_VarSlit_L2.draw(kind='field', logarithm=False)
```

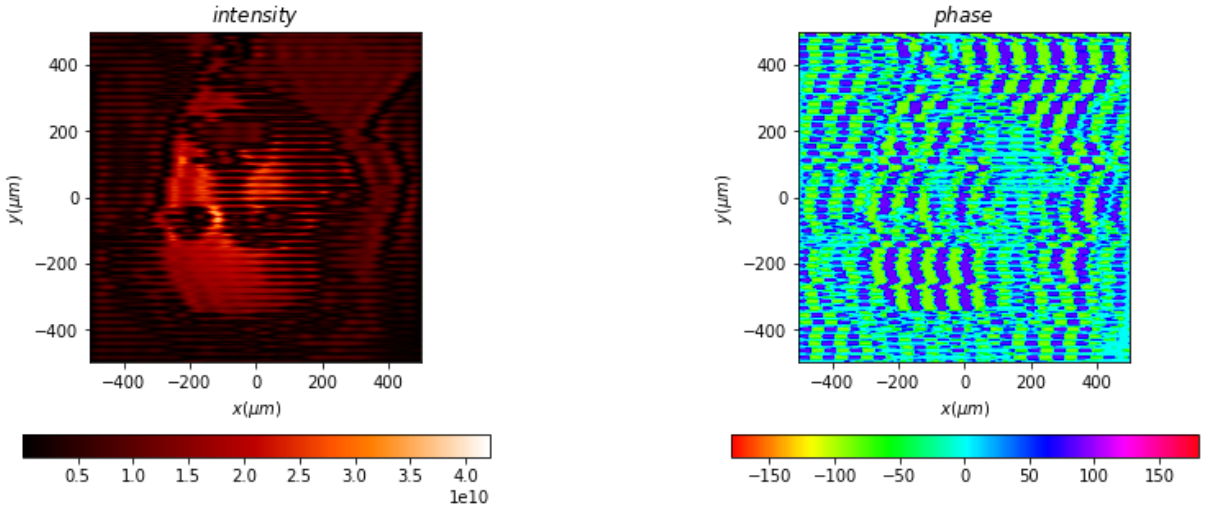

intensity
phase
WIDTH = 10 um
y(μm)
x(μm)
1e10

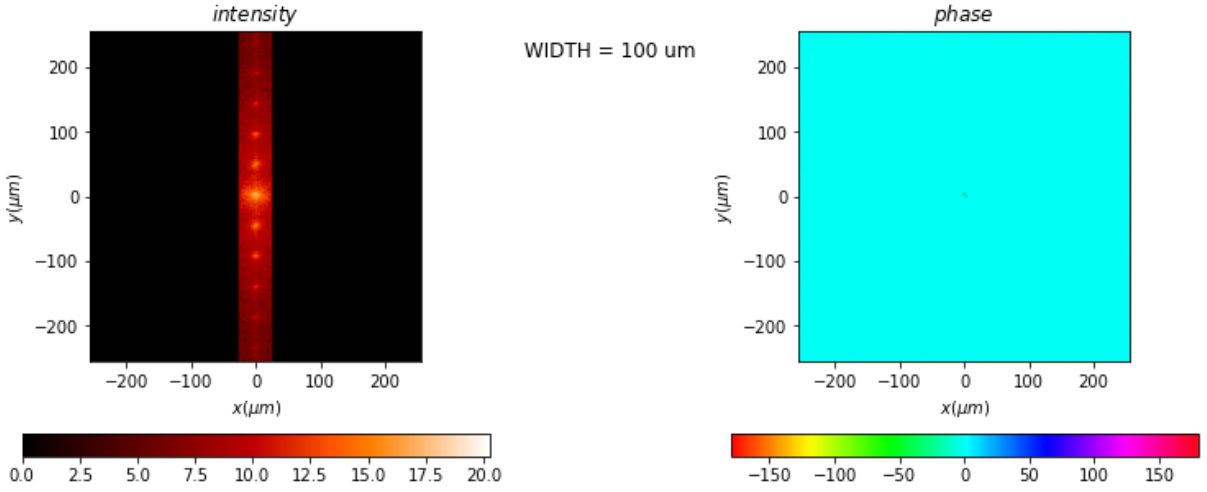

intensity
phase
y(μm)
x(μm)
1e10

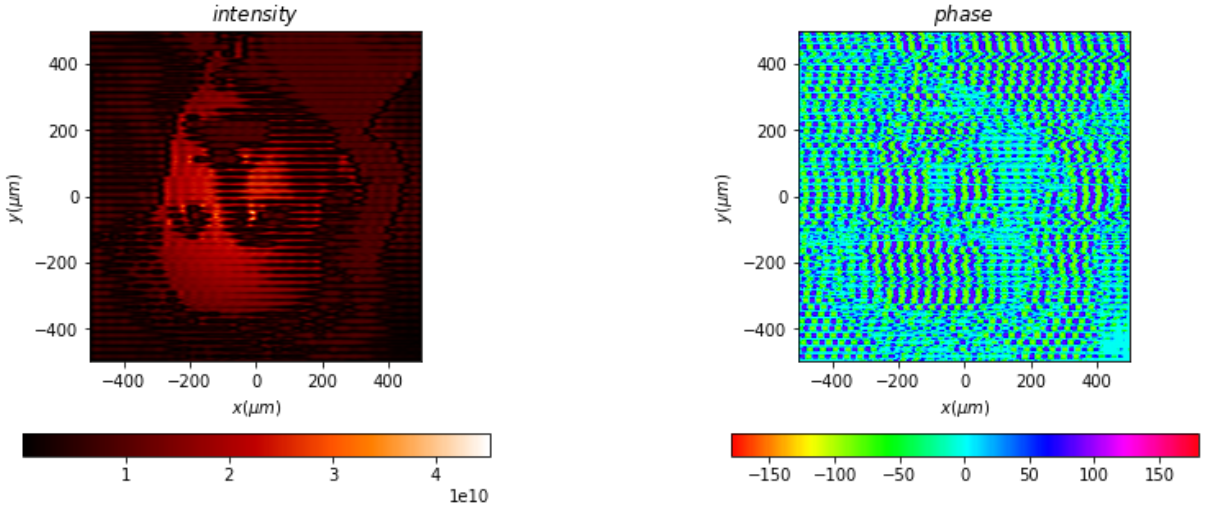

intensity
phase
WIDTH = 30 um
y(μm)
x(μm)
1e10

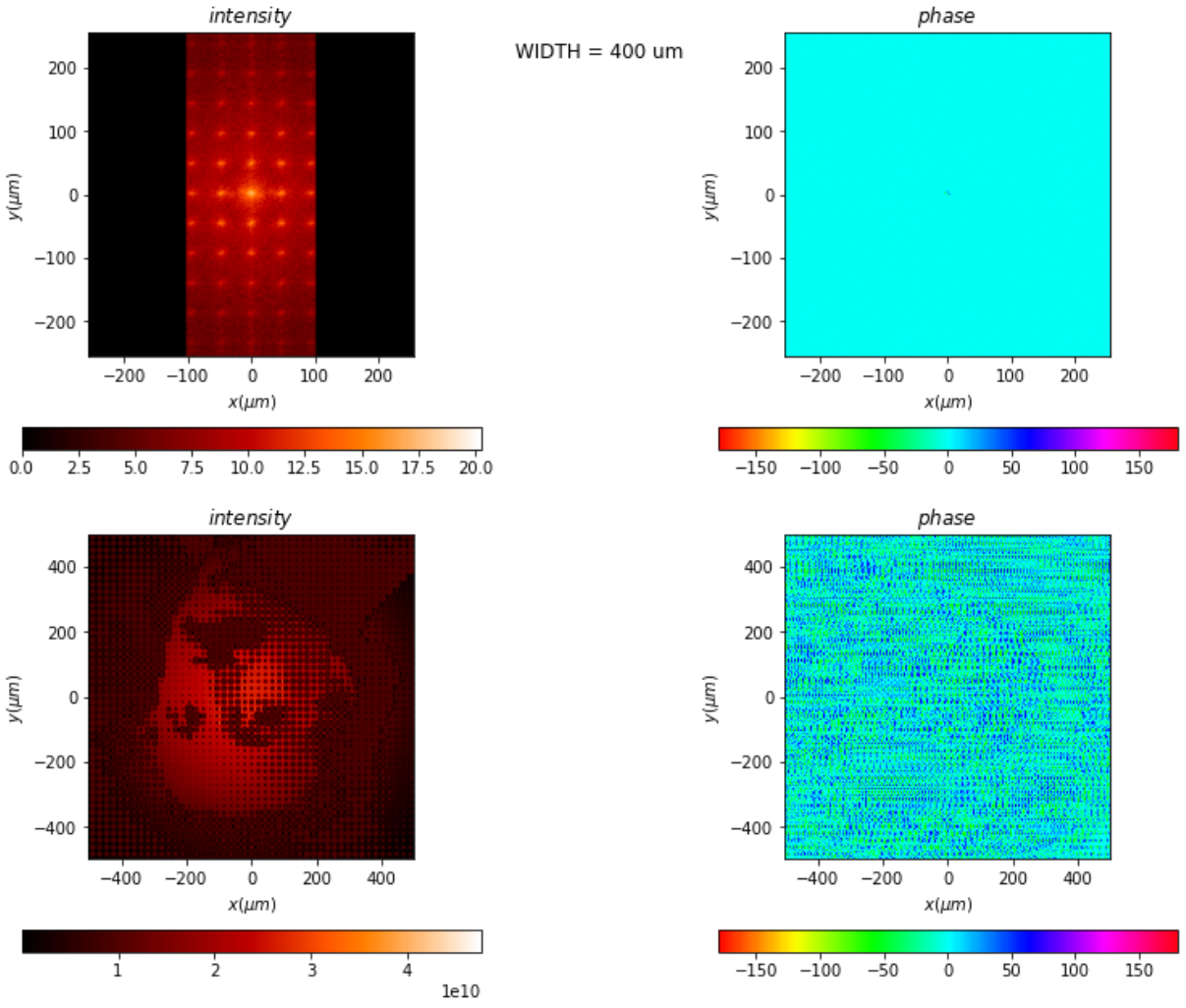

intensity
phase
y(μm)
x(μm)
1e10


intensity
phase
WIDTH = 50 um
y(μm)
x(μm)
1e10


intensity
phase
y(μm)
x(μm)
1e10


intensity
phase
WIDTH = 100 um
y(μm)
x(μm)
1e10


intensity
phase
y(μm)
x(μm)
1e10


intensity
WIDTH = 400 um
phase
200
100
0
−100
−200
y(μm)
−200 −100 0 100 200
x(μm)
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
−150 −100 −50 0 50 100 150
intensity
phase
400
200
0
−200
−400
−400 −200 0 200 400
1 2 3 4
1e10

In [ ]:

Alphabets - Spatial Filtering and Character Recognition

In [1]:
```
import numpy as np
from diffractio import mm, um, degrees
from diffractio.scalar_sources_XY import Scalar_source_XY
from diffractio.scalar_masks_XY import Scalar_mask_XY


# Setting up
length = 1 * mm
num_data = 512
x0 = np.linspace(-length / 2, length / 2, num_data)
y0 = np.linspace(-length / 2, length / 2, num_data)
wavelength = 0.633 * um
```

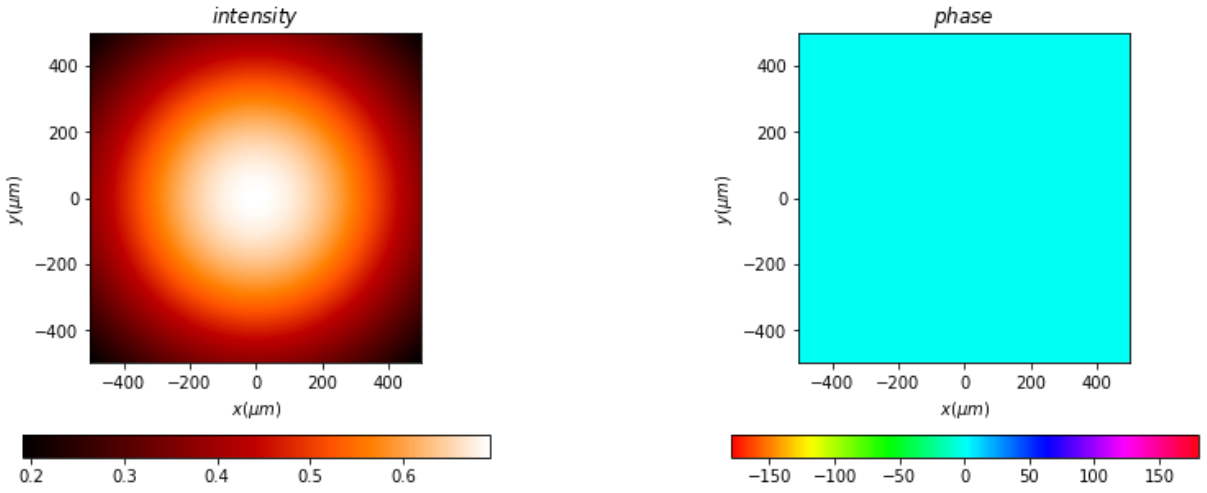
```
number of processors: 12
```

Setting up source

- Gaussian Beam (LASER)

In [2]:
```
# Gaussian Beam Source - Like a LASER
u0 = Scalar_source_XY(x=x0, y=y0, wavelength=wavelength)
u0.gauss_beam(r0=(0, 0), w0=(800 * um, 800 * um), z0=0.0)
u0.draw(kind='field', logarithm=True)
```

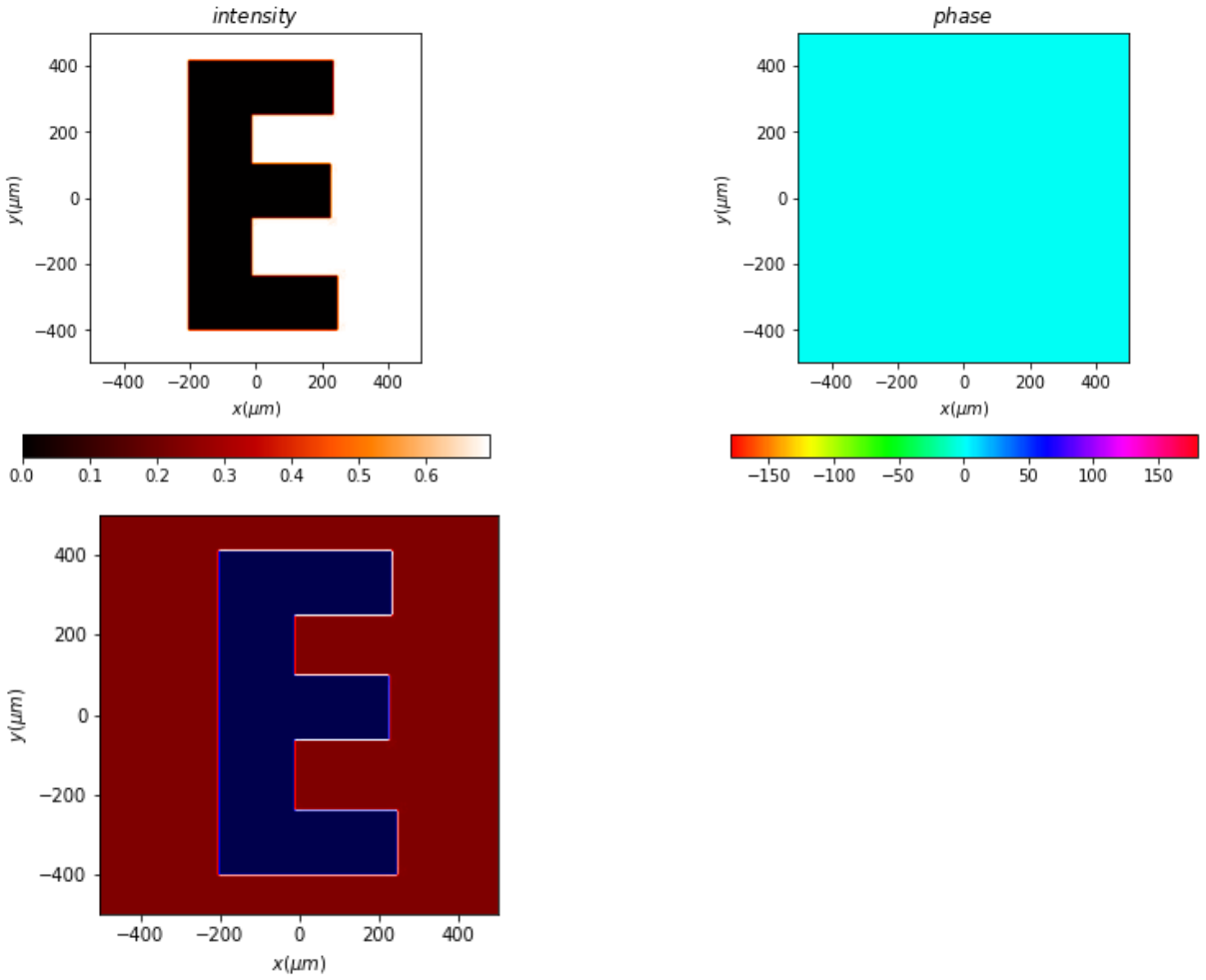
Out[2]:
```
((<matplotlib.image.AxesImage at 0x218cbf0da60>,
  <matplotlib.image.AxesImage at 0x218cc1ad220>),
 None,
 None)
```



E & F

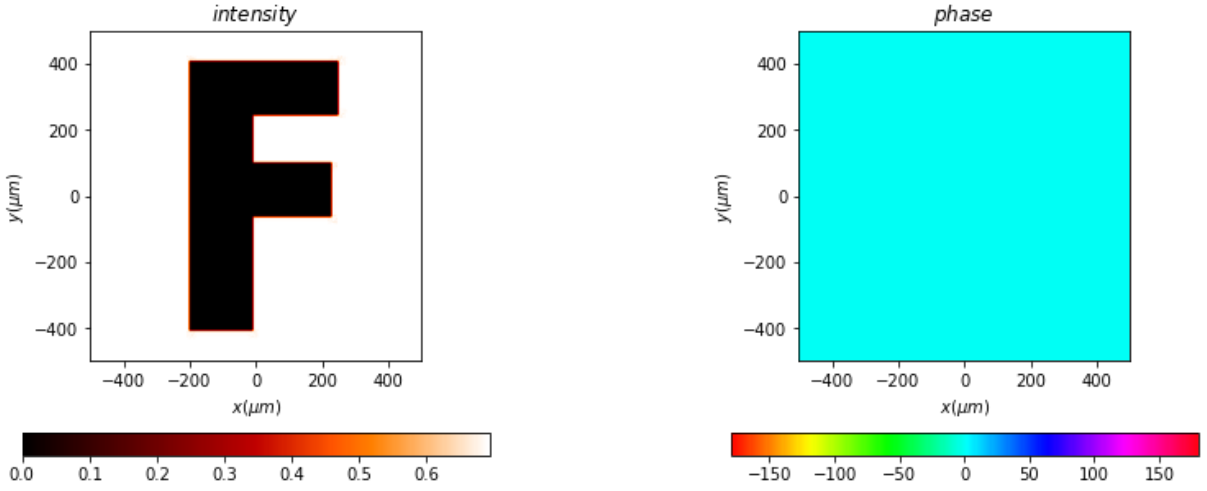
In [3]:
```
E = Scalar_mask_XY(x0, y0, wavelength)
E.image(
    filename="E.png",
    normalize=True,
    canal=0,
)

E.draw(kind='field', logarithm=True)
E.draw(kind='real_field', logarithm=True)
```

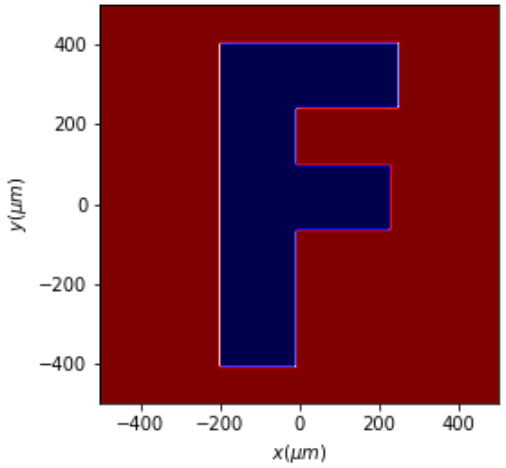
Out[3]:
```
(<Figure size 432x288 with 1 Axes>,
 <AxesSubplot:xlabel='$x  (\\mu m)$', ylabel='$y  (\\mu m)$'>,
 <matplotlib.image.AxesImage at 0x218cca87dc0>)
```



In [4]:
```
F = Scalar_mask_XY(x0, y0, wavelength)
F.image(
    filename="F.png",
    normalize=True,
    canal=0,
)

F.draw(kind='field', logarithm=True)
F.draw(kind='real_field', logarithm=True)
```

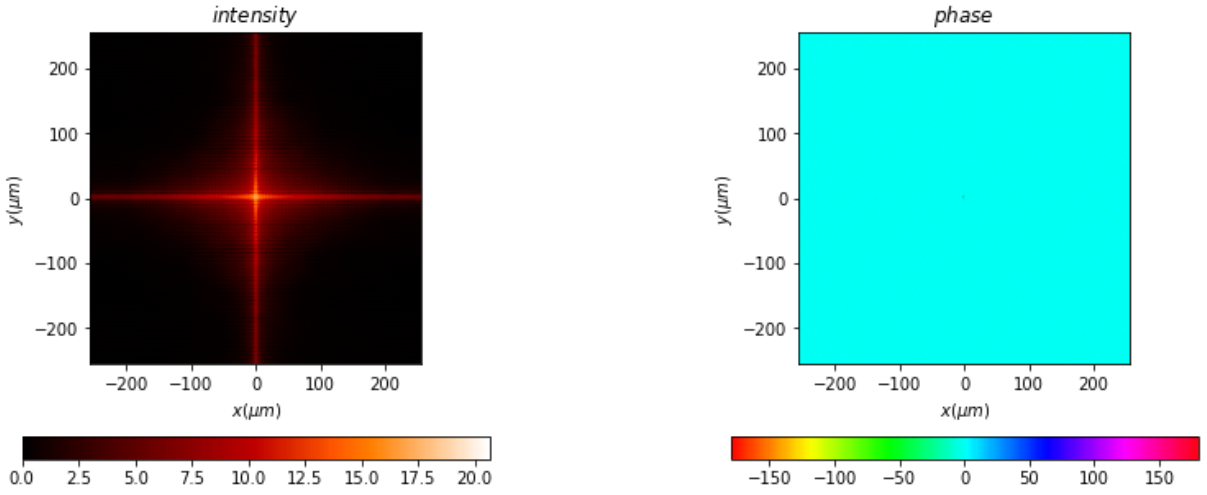
Out[4]:
```
(<Figure size 432x288 with 1 Axes>,
 <AxesSubplot:xlabel='$x  (\\mu m)$', ylabel='$y  (\\mu m)$'>,
 <matplotlib.image.AxesImage at 0x218cd0785b0>)
```





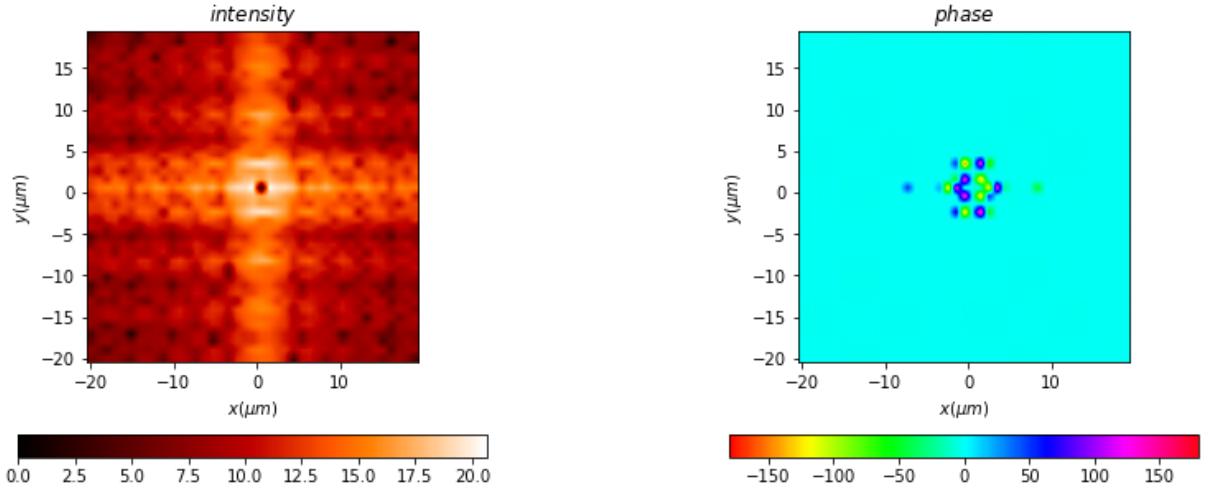
In [5]:
```
# Fourier Plane - No Filter
E_a_L1 = (u0 * E).fft(z=1 * mm, new_field=True)
E_a_L1.draw(kind='field', logarithm=True)
```

Out[5]:
```
((<matplotlib.image.AxesImage at 0x218d0d64190>,
  <matplotlib.image.AxesImage at 0x218d0e0e910>),
 None,
 None)
```



In [6]:
```
# A closer look
E_a_L1_Crop = E_a_L1.cut_resample(x_limits=(-20, 20), y_limits=(-20, 20), new_field=True)
E_a_L1_Crop.draw(kind='field', logarithm=True)
```

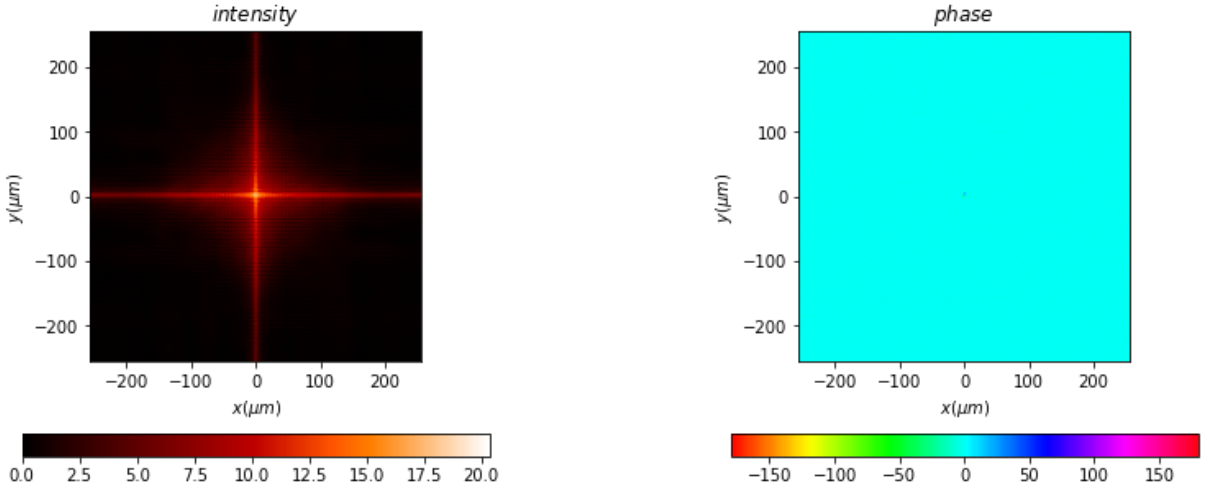
Out[6]:
```
((<matplotlib.image.AxesImage at 0x218d0d9ac10>,
  <matplotlib.image.AxesImage at 0x218cd043400>),
 None,
 None)
```



In [7]:
```
# Fourier Plane - No Filter
```

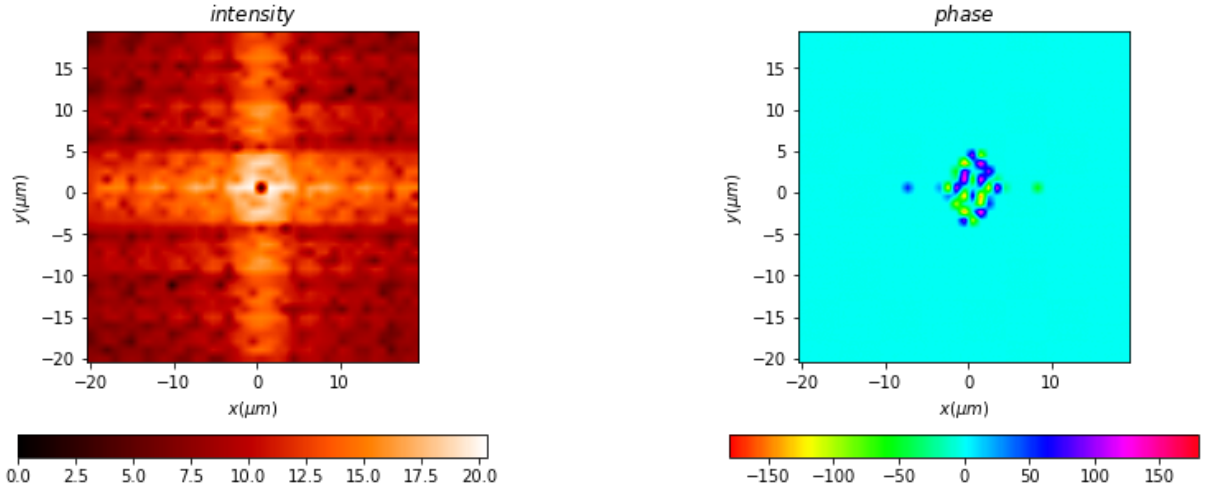
```
F_a_L1 = (u0 * F).fft(z=1 * mm, new_field=True)
F_a_L1.draw(kind='field', logarithm=True)
```

Out[7]:
```
((<matplotlib.image.AxesImage at 0x218d0e9a940>,
  <matplotlib.image.AxesImage at 0x218ccb75520>),
 None,
 None)
```



In [8]:
```
# A closer look
F_a_L1_Crop = F_a_L1.cut_resample(x_limits=(-20, 20), y_limits=(-20, 20), new_field=True)
F_a_L1_Crop.draw(kind='field', logarithm=True)
```

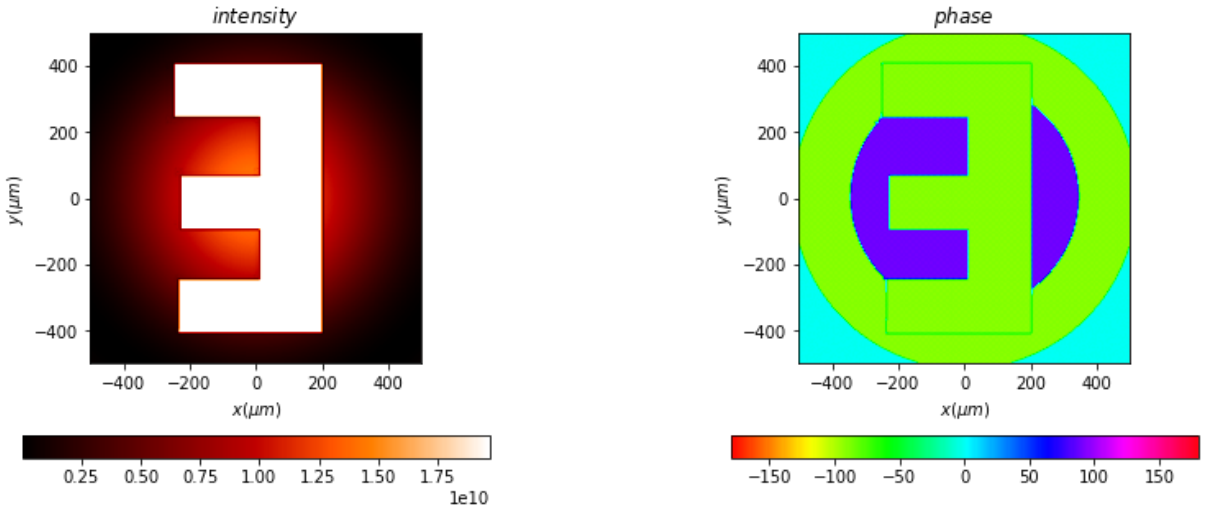
Out[8]:
```
((<matplotlib.image.AxesImage at 0x218ccbc7fa0>,
  <matplotlib.image.AxesImage at 0x218d0f32730>),
 None,
 None)
```



So, the difference in the Fourier encoding for two very similar characters or alphabets lies not so much in the position-space, as it does in the phase-space. This "hidden" phase-space information can be used to enable better machine encodings for Character Recognition software.

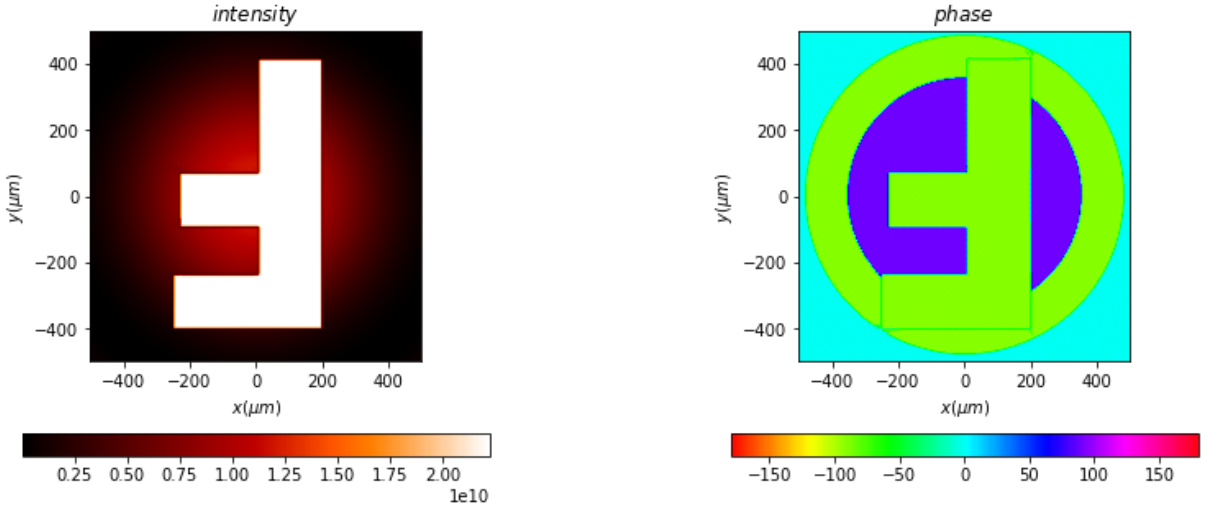
In [9]:
```
# This is how, it'd look without any filter, at the Observation screen
E_a_L2 = E_a_L1.fft(z=1 * mm, shift=False, remove0=False, new_field=True)
E_a_L2.draw(kind='field', logarithm=False)
```

Out[9]:
```
((<matplotlib.image.AxesImage at 0x218d0bf7130>,
  <matplotlib.image.AxesImage at 0x218d0c8c370>),
 None,
 None)
```



In [10]:
```
# This is how, it'd look without any filter, at the Observation screen
F_a_L2 = F_a_L1.fft(z=1 * mm, shift=False, remove0=False, new_field=True)
F_a_L2.draw(kind='field', logarithm=False)
```

Out[10]:
```
((<matplotlib.image.AxesImage at 0x218d15767f0>,
  <matplotlib.image.AxesImage at 0x218d1604a60>),
 None,
 None)
```



Some Masks / Filters

- Center Dot - High Pass
- Square - Low Pass
- Square + Center Dot -

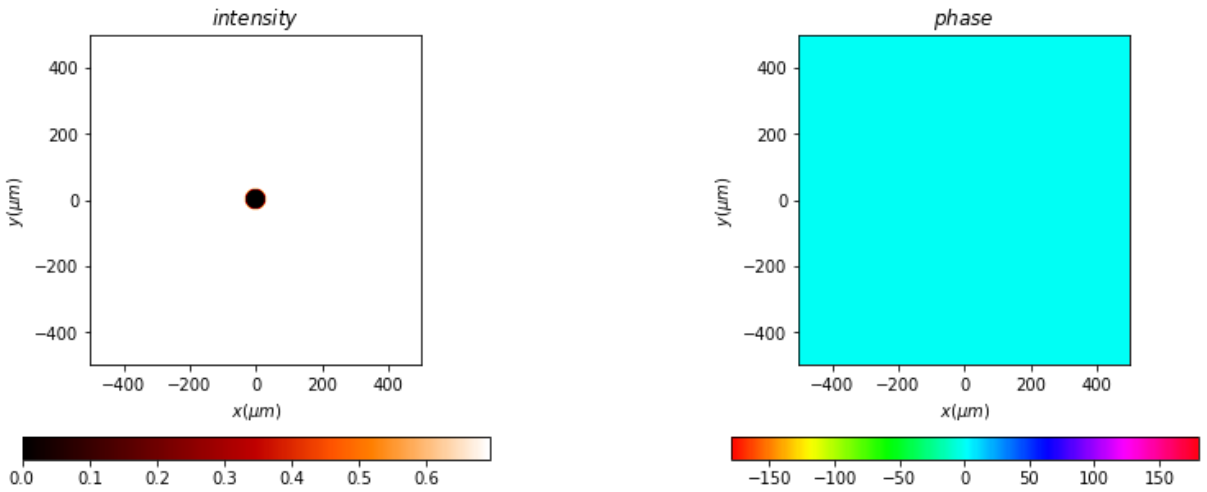
- Vertical Slit

- Horizontal Slit
- Angled Slit
- Variable Slits

- Mesh

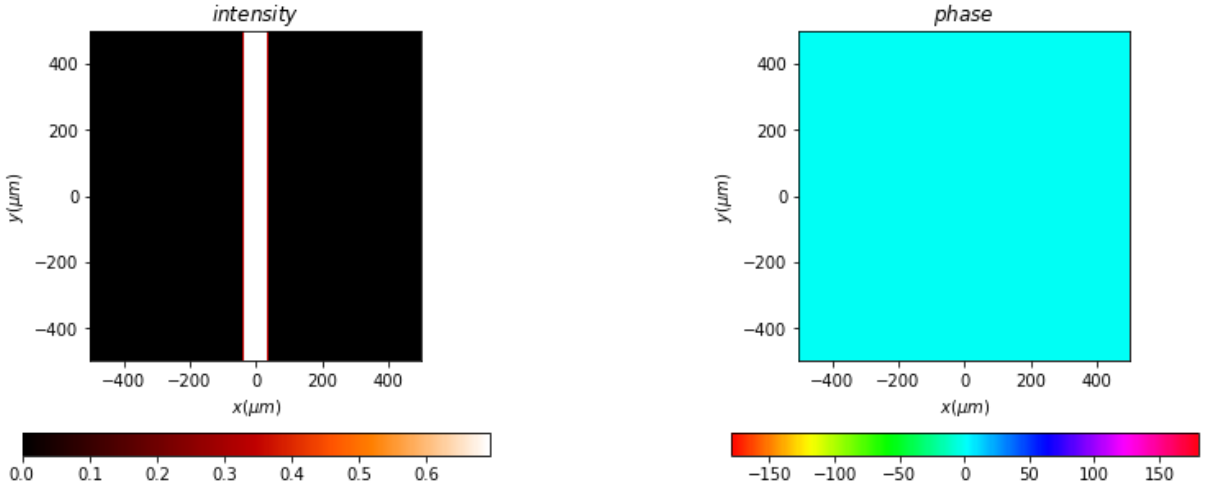
In [11]:
```
cDot = Scalar_mask_XY(x0, y0, wavelength)
cDot.ring(
    r0=(0 * um, 0 * um),
    radius1=(30 * um, 30 * um),
    radius2=(1000 * um, 1000 * um)
)
cDot.draw(kind='field', logarithm=True)
```

Out[11]:
```
((<matplotlib.image.AxesImage at 0x218d15a4190>,
  <matplotlib.image.AxesImage at 0x218d0dc1100>),
 None,
 None)
```



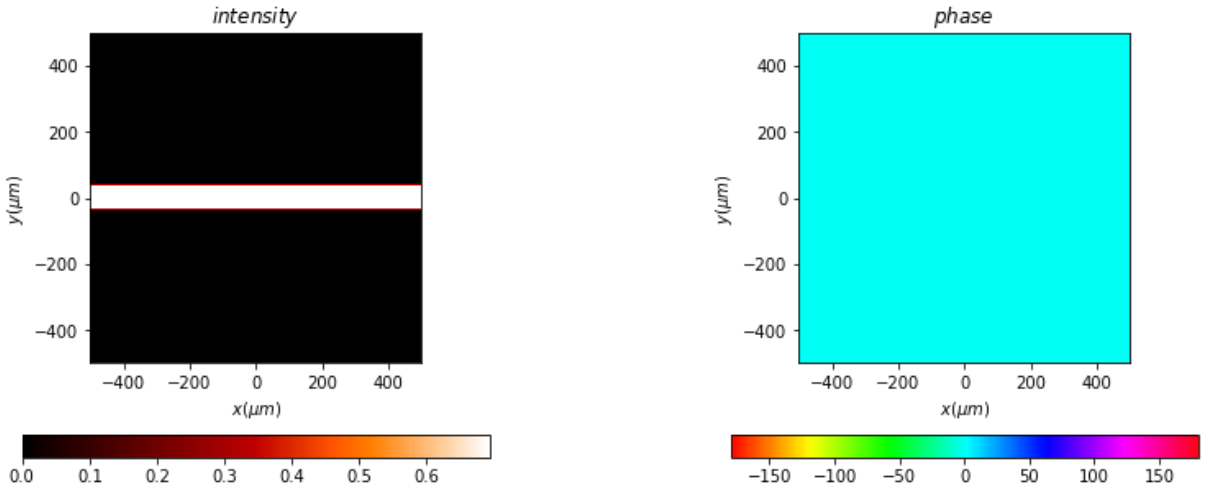
In [12]:
```
vert = Scalar_mask_XY(x0, y0, wavelength)
vert.slit(
    x0=0 * um,
    size=75 * um
)
vert.draw(kind='field', logarithm=True)
```

Out[12]:
```
((<matplotlib.image.AxesImage at 0x218d1fbb910>,
  <matplotlib.image.AxesImage at 0x218d20430d0>),
 None,
 None)
```



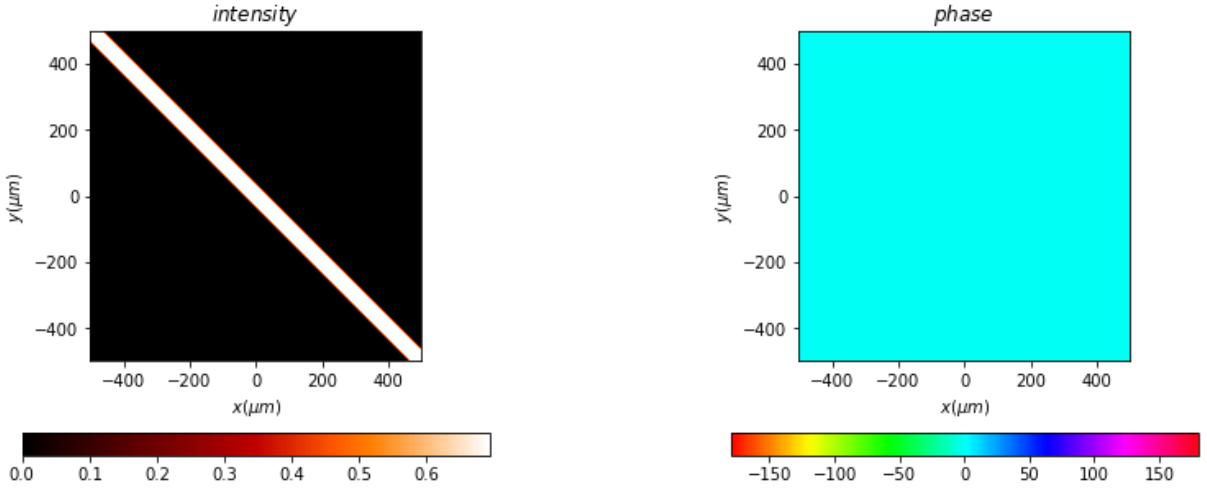
In [13]:
```
horiz = Scalar_mask_XY(x0, y0, wavelength)
horiz.slit(
    x0=0 * um,
    size=75 * um,
    angle=np.pi / 2
)
horiz.draw(kind='field', logarithm=True)
```

Out[13]:
```
((<matplotlib.image.AxesImage at 0x218d2108370>,
  <matplotlib.image.AxesImage at 0x218d2184b50>),
 None,
 None)
```



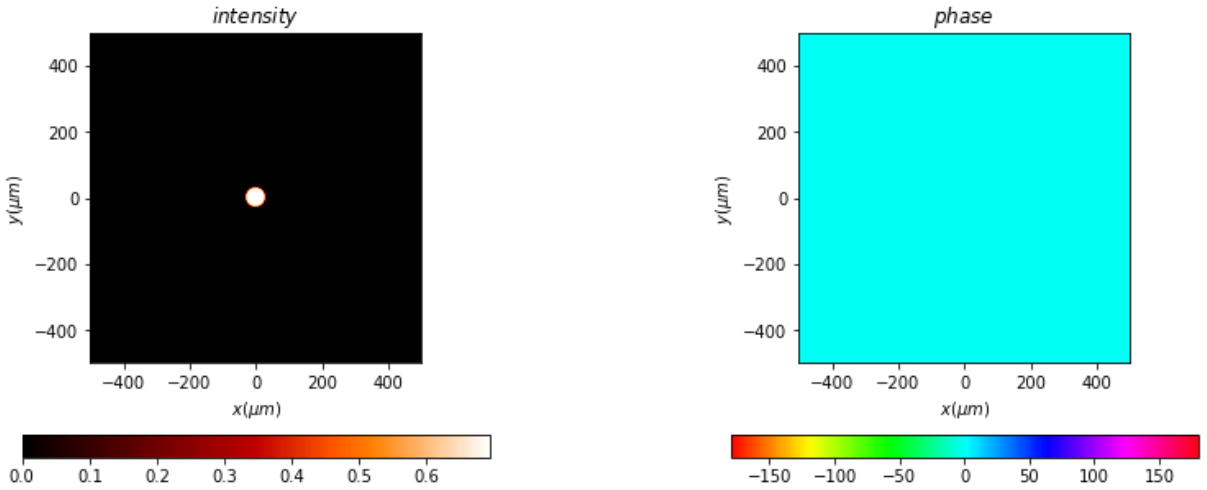
In [14]:
```
angled = Scalar_mask_XY(x0, y0, wavelength)
angled.slit(
    x0=0 * um,
    size=50 * um,
    angle=np.pi / 4
)
angled.draw(kind='field', logarithm=True)
```

Out[14]:
```
((<matplotlib.image.AxesImage at 0x218d30abdf0>,
  <matplotlib.image.AxesImage at 0x218d312d5e0>),
 None,
 None)
```



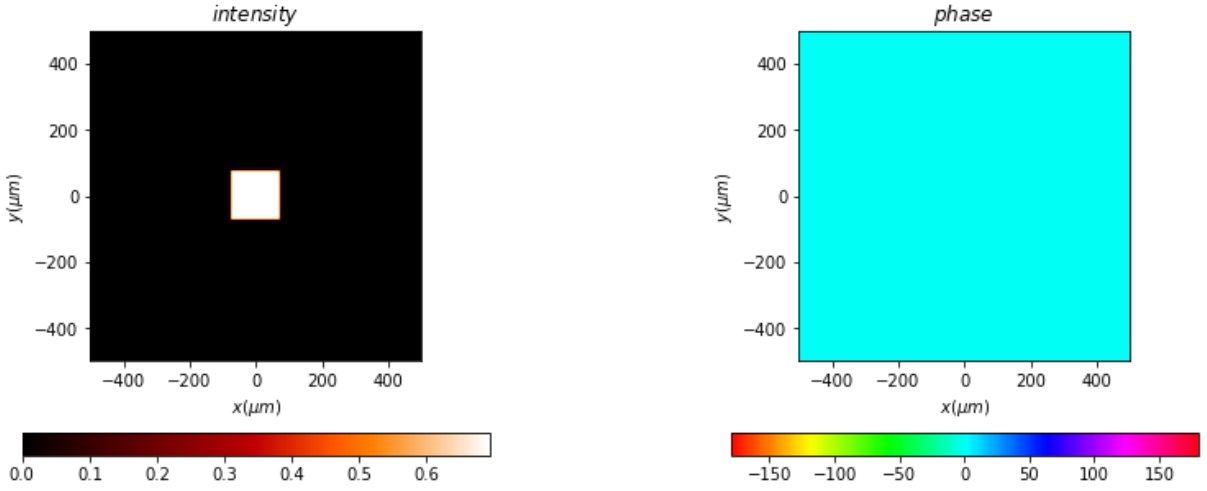
In [15]:
```
circ = Scalar_mask_XY(x0, y0, wavelength)
circ.circle(
    r0=(0 * um, 0 * um),
    radius=(30 * um, 30 * um),
    angle=0
)
circ.draw(kind='field', logarithm=True)
```

Out[15]:
```
((<matplotlib.image.AxesImage at 0x218d31d6ca0>,
  <matplotlib.image.AxesImage at 0x218d4aeef70>),
 None,
 None)
```



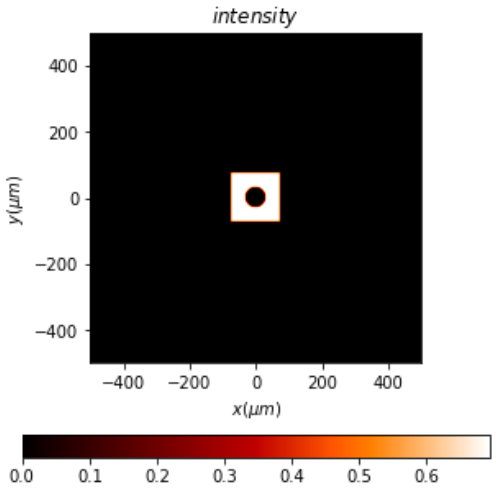
In [16]:
```
sq = Scalar_mask_XY(x0, y0, wavelength)
sq.square(
    r0=(0 * um, 0 * um),
    size=(150 * um, 150 * um),
    angle=0,
)
sq.draw(kind='field', logarithm=True)
```

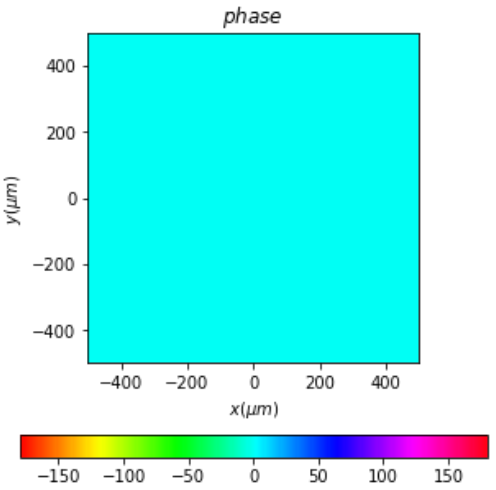
Out[16]:
```
((<matplotlib.image.AxesImage at 0x218d4ba1520>,
  <matplotlib.image.AxesImage at 0x218d4c357c0>),
 None,
 None)
```



In [17]:
```
# Adding two filters
sqCirc = sq * cDot
sqCirc.draw(kind="field", logarithm=True)
```

Out[17]:
```
((<matplotlib.image.AxesImage at 0x218d4bb70d0>,
  <matplotlib.image.AxesImage at 0x218d17567f0>),
 None,
 None)
```





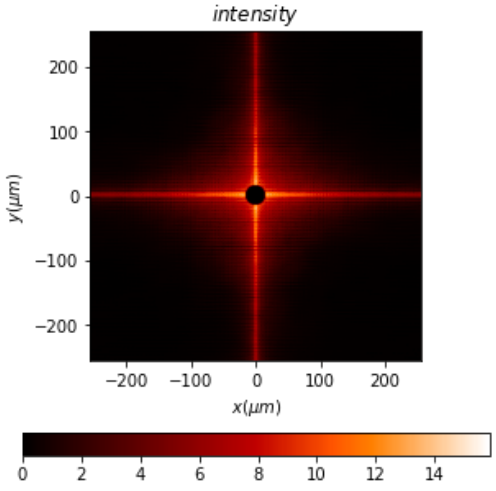
Returns wave after encountering filter and then L2

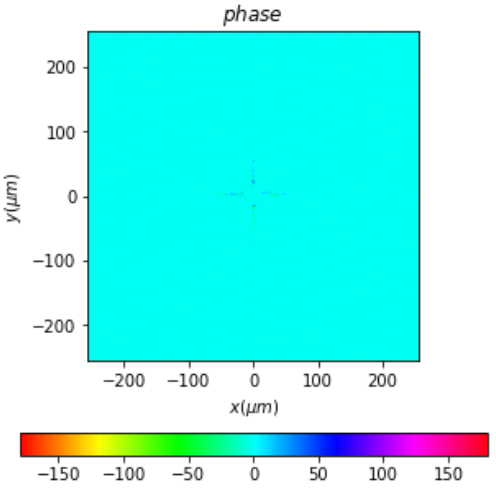
In [18]:
```
def sim(a_L1, mask):
    """
    a_L1: Wave after passing through Lens 1
    mask: Mask or Filter to apply

    """
    a_L1_Mask = a_L1 * mask
    a_L1_Mask_L2 = a_L1_Mask.fft(z=1 * mm, shift=False, remove0=False, new_field=True)

    return a_L1_Mask, a_L1_Mask_L2
```

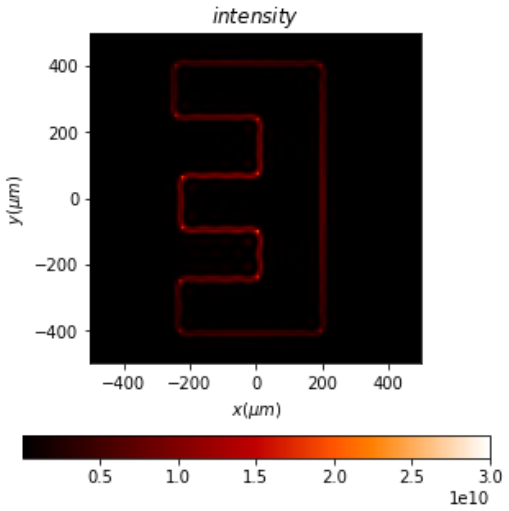
Center Dot - High Pass - HT Structure Visible - Dark-Field Illumination / Edge Enhancement

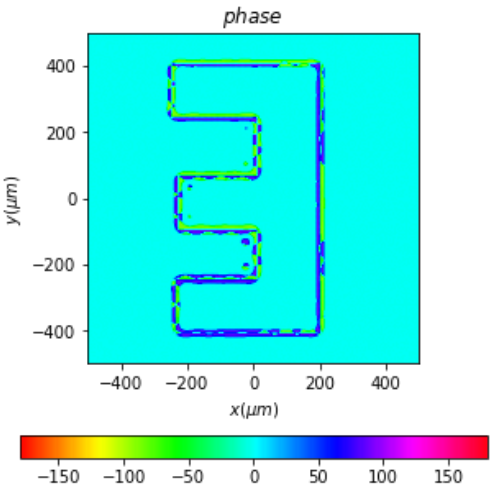
In [19]:
```
E_a_L1_cD, E_a_L1_cD_L2 = sim(E_a_L1, cDot)
E_a_L1_cD.draw(kind='field', logarithm=True)
E_a_L1_cD_L2.draw(kind='field', logarithm=False)
```

Out[19]:
```
((<matplotlib.image.AxesImage at 0x218d5f0c7c0>,
  <matplotlib.image.AxesImage at 0x218d7787f40>),
 None,
 None)
```





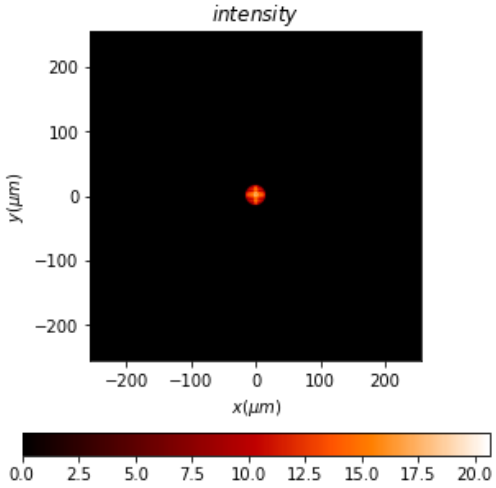


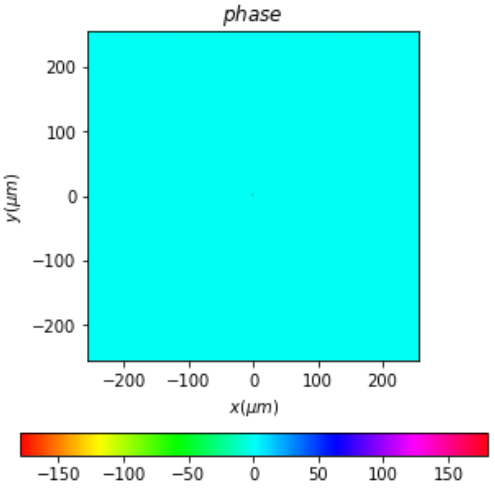


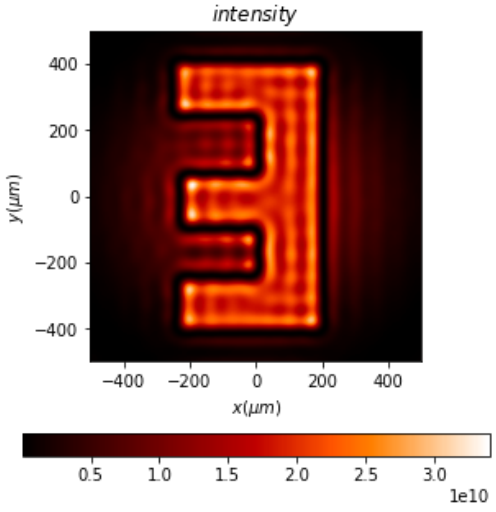
Circle - Low Pass - Blurred Output

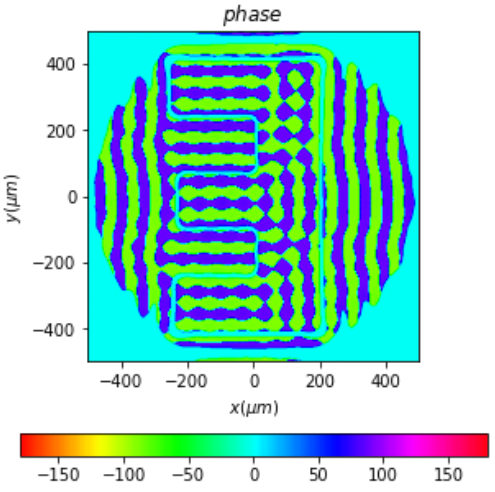
In [20]:
```
E_a_L1_Circ, E_a_L1_Circ_L2 = sim(E_a_L1, circ)
E_a_L1_Circ.draw(kind='field', logarithm=True)
E_a_L1_Circ_L2.draw(kind='field', logarithm=False)
```

Out[20]:
```
((<matplotlib.image.AxesImage at 0x218d8423370>,
  <matplotlib.image.AxesImage at 0x218d848e5e0>),
 None,
 None)
```





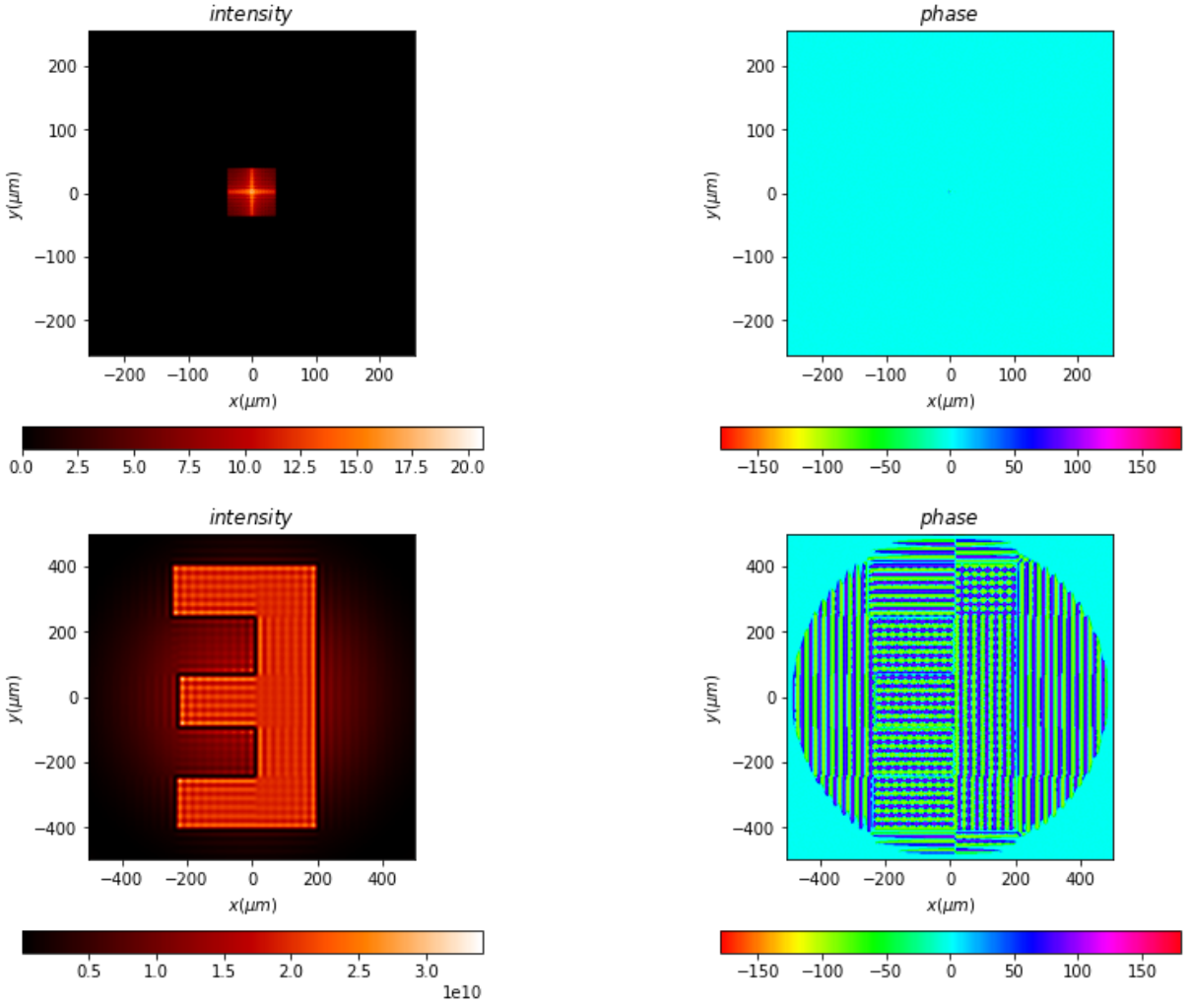




Square

In [21]:
```
E_a_L1_Sq, E_a_L1_Sq_L2 = sim(E_a_L1, sq)
E_a_L1_Sq.draw(kind='field', logarithm=True)
E_a_L1_Sq_L2.draw(kind='field', logarithm=False)
```

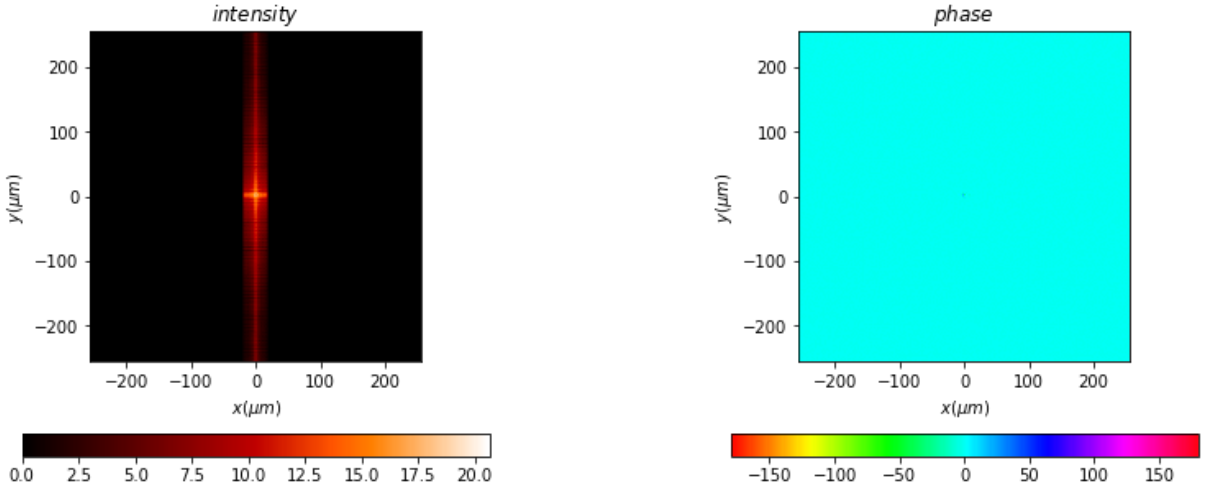
Out[21]:
```
((<matplotlib.image.AxesImage at 0x218d910f580>,
  <matplotlib.image.AxesImage at 0x218d91797f0>),
 None,
 None)
```

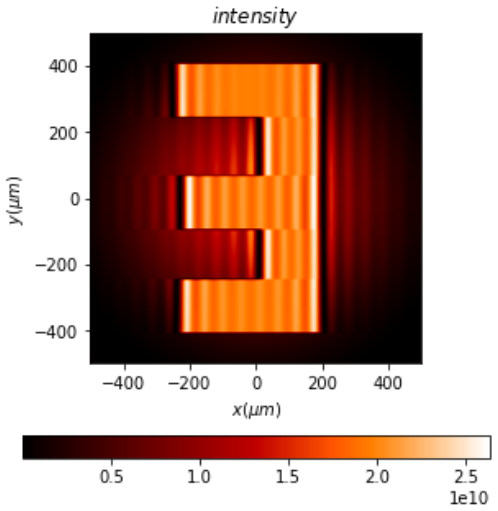


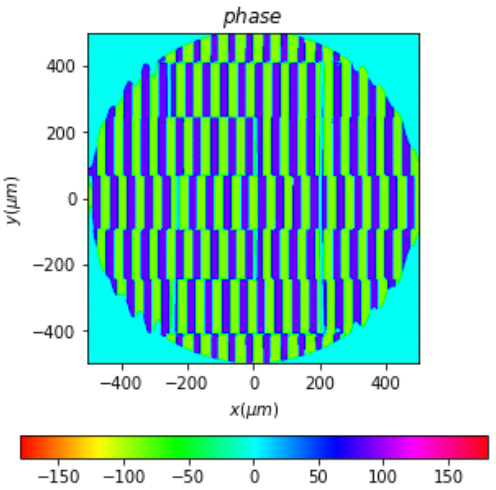
Vertical Slit

In [22]:
```
E_a_L1_Vert, E_a_L1_Vert_L2 = sim(E_a_L1, vert)
E_a_L1_Vert.draw(kind='field', logarithm=True)
E_a_L1_Vert_L2.draw(kind='field', logarithm=False)
```

Out[22]:
```
((<matplotlib.image.AxesImage at 0x218d4c9ba00>,
  <matplotlib.image.AxesImage at 0x218d83cf100>),
 None,
 None)
```



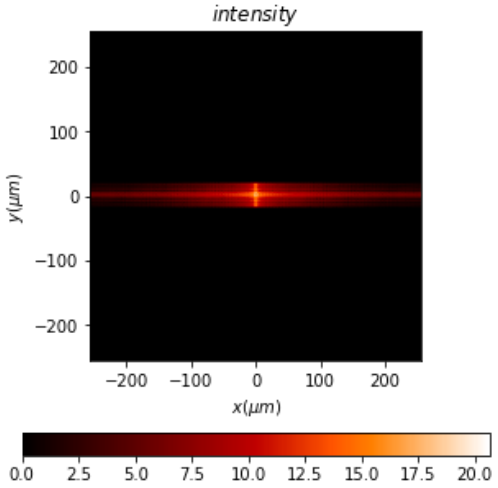


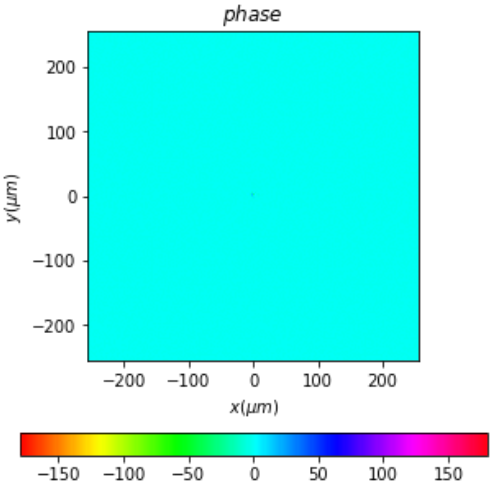


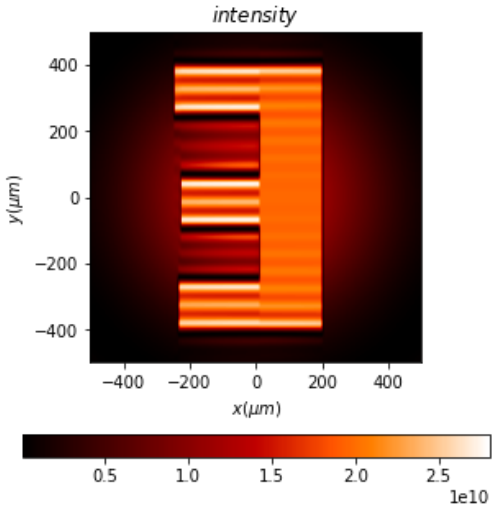
Horizontal Slit

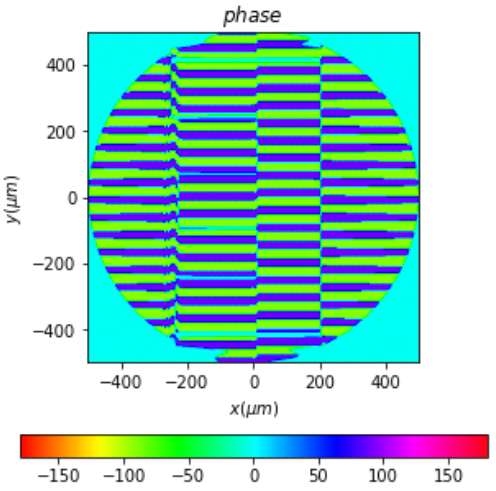
In [23]:
```
E_a_L1_Horiz, E_a_L1_Horiz_L2 = sim(E_a_L1, horiz)
E_a_L1_Horiz.draw(kind='field', logarithm=True)
E_a_L1_Horiz_L2.draw(kind='field', logarithm=False)
```

Out[23]:
```
((<matplotlib.image.AxesImage at 0x218d9d892b0>,
  <matplotlib.image.AxesImage at 0x218d9ded8e0>),
 None,
 None)
```





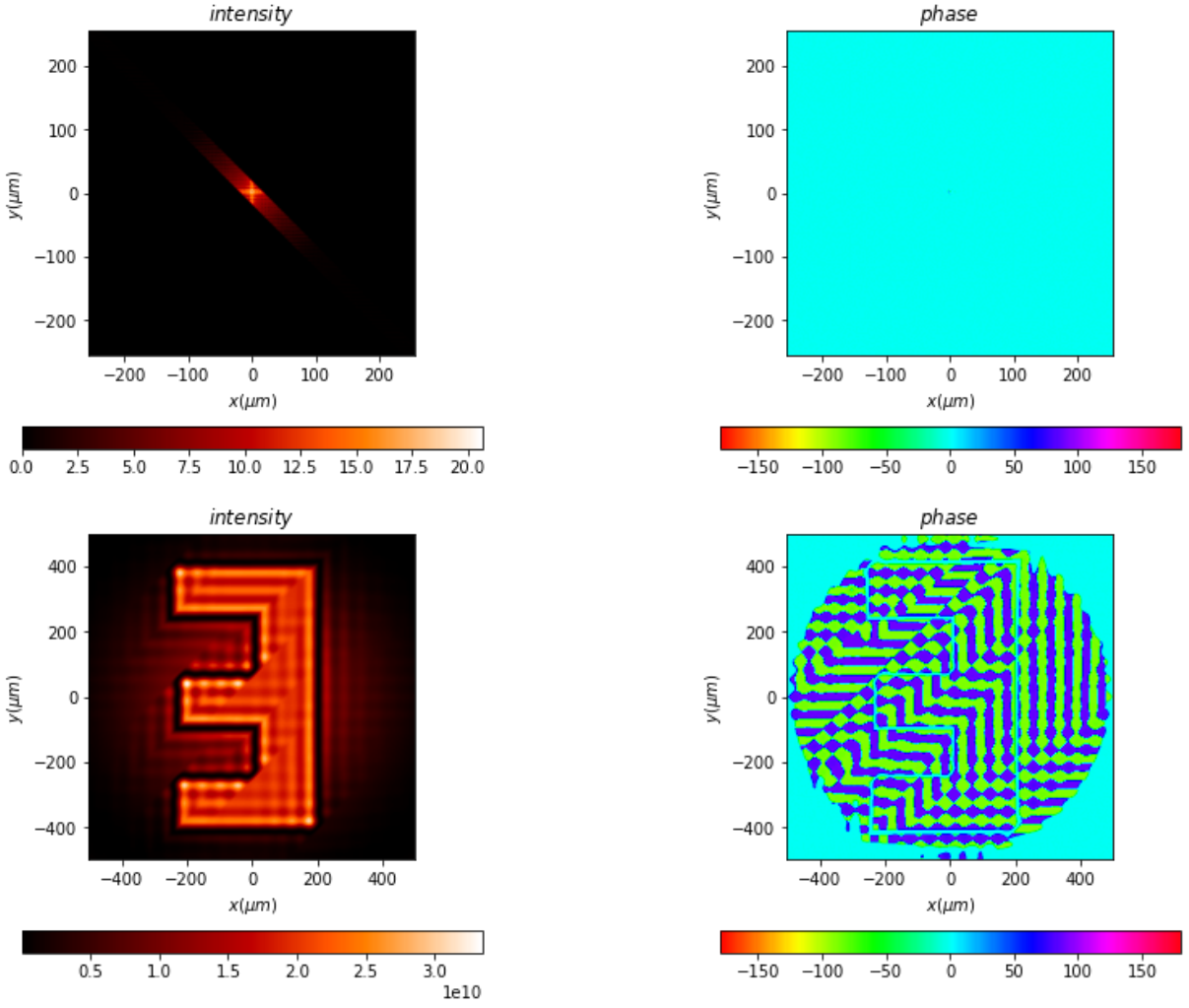




Angled Slit

In [24]:
```
E_a_L1_Angled, E_a_L1_Angled_L2 = sim(E_a_L1, angled)
E_a_L1_Angled.draw(kind='field', logarithm=True)
E_a_L1_Angled_L2.draw(kind='field', logarithm=False)
```

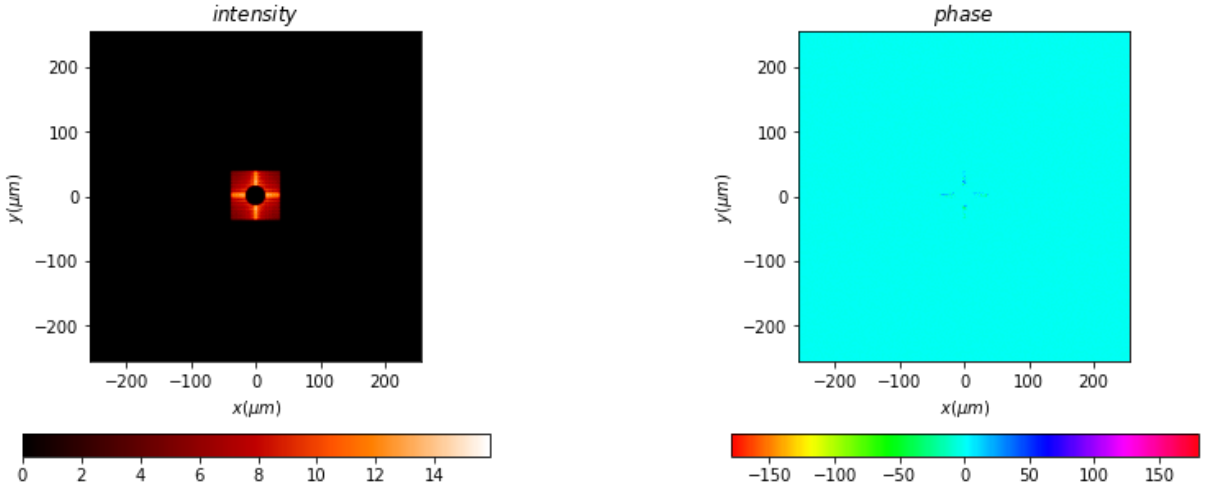
Out[24]:
```
((<matplotlib.image.AxesImage at 0x218da856a30>,
  <matplotlib.image.AxesImage at 0x218dc584ca0>),
 None,
 None)
```

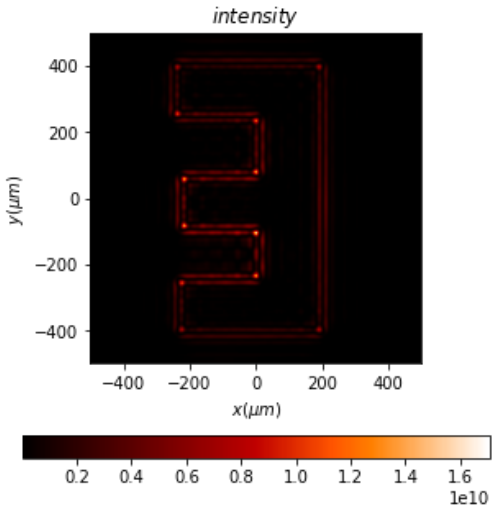


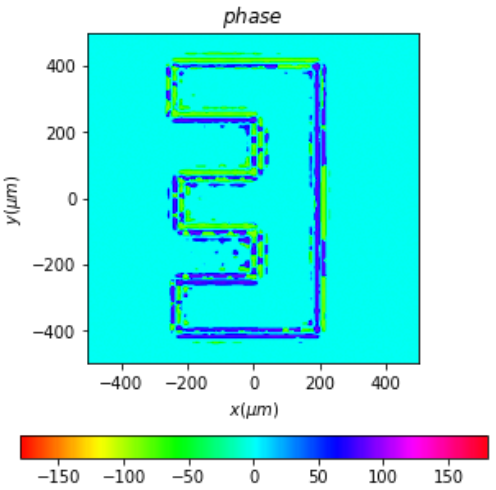
Square + Circle

In [25]:
```
E_a_L1_SqCirc, E_a_L1_SqCirc_L2 = sim(E_a_L1, sqCirc)
E_a_L1_SqCirc.draw(kind='field', logarithm=True)
E_a_L1_SqCirc_L2.draw(kind='field', logarithm=False)
```

Out[25]:
```
((<matplotlib.image.AxesImage at 0x218df855e80>,
  <matplotlib.image.AxesImage at 0x218dc5e2970>),
 None,
 None)
```

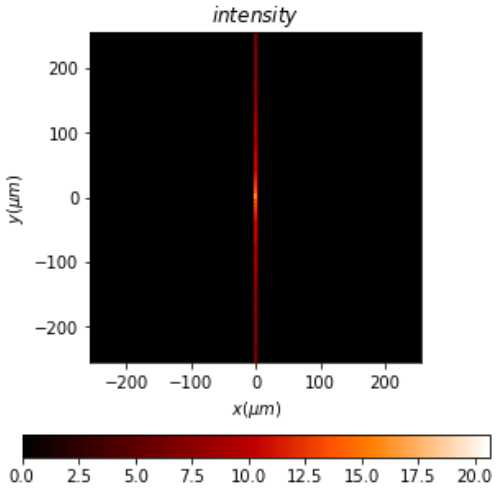




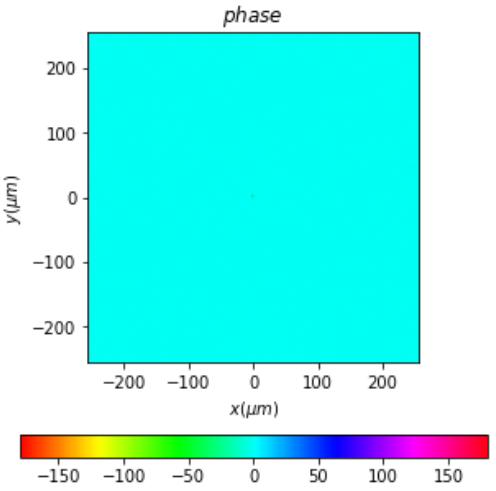


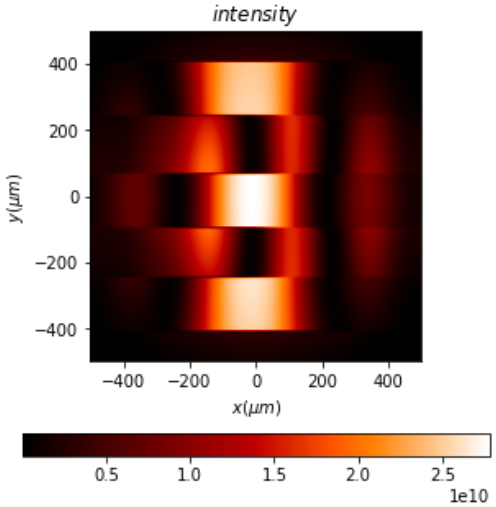
Variable Slit Widths

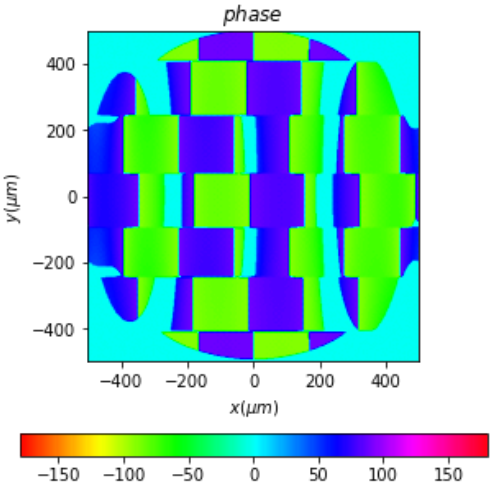
In [26]:
```
widths = [10, 30, 50, 76.3209395, 100, 400]  # in um
for width in widths:
    varSlit = Scalar_mask_XY(x0, y0, wavelength)
    varSlit.slit(
        x0=0 * um,
        size=width * um
    )

    E_a_L1_VarSlit, E_a_L1_VarSlit_L2 = sim(E_a_L1, varSlit)
    E_a_L1_VarSlit.draw(title=f" WIDTH = {width} um ", kind='field', logarithm=True)
    E_a_L1_VarSlit_L2.draw(kind='field', logarithm=False)
```

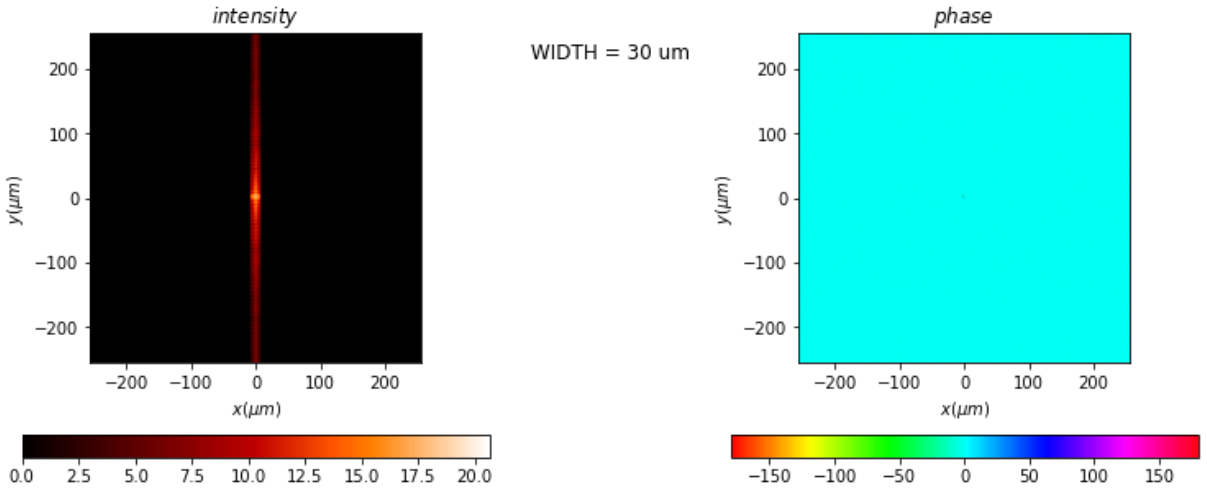


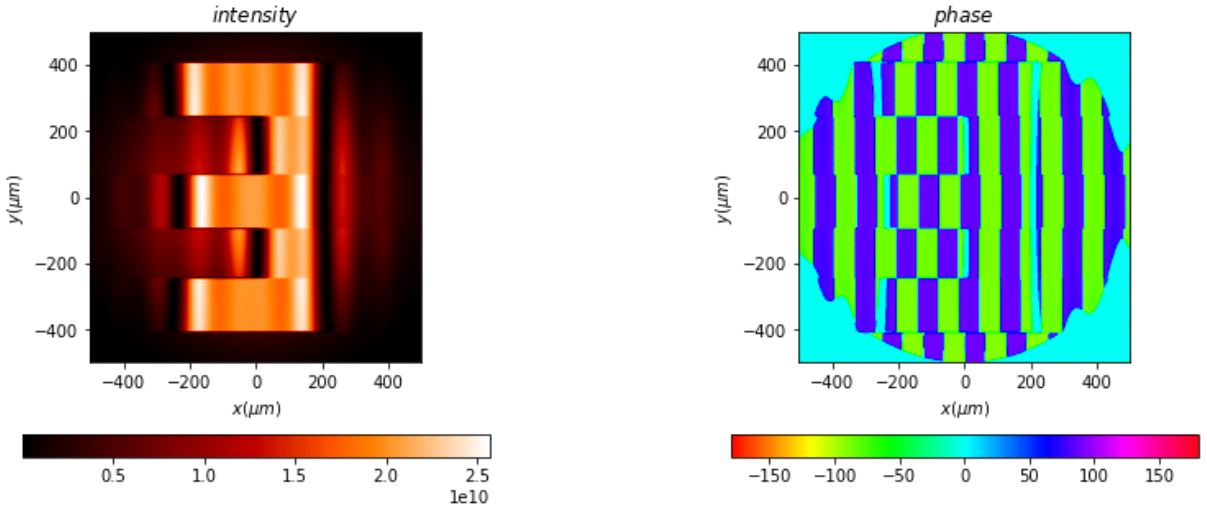
WIDTH = 10 um

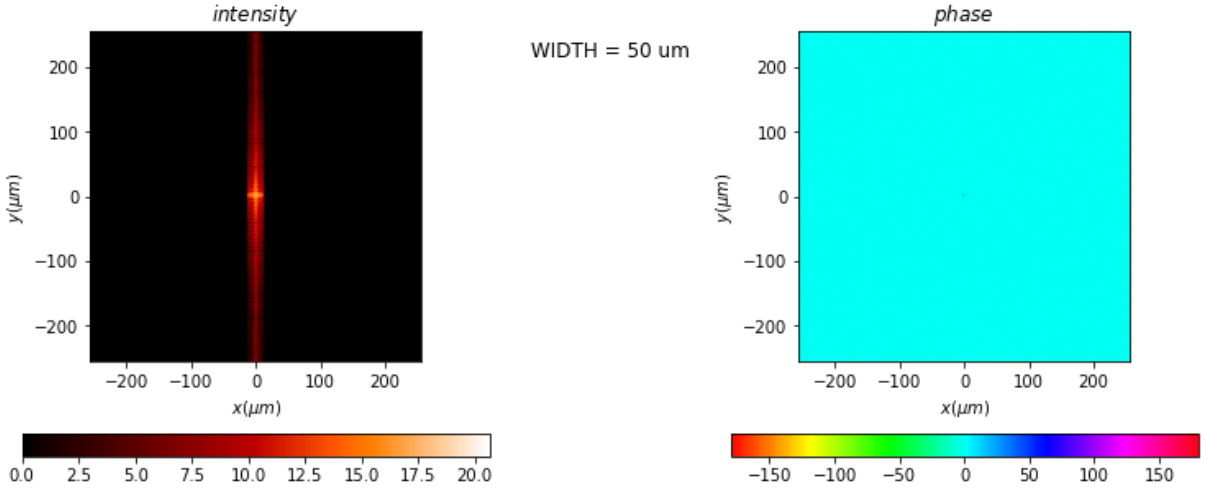


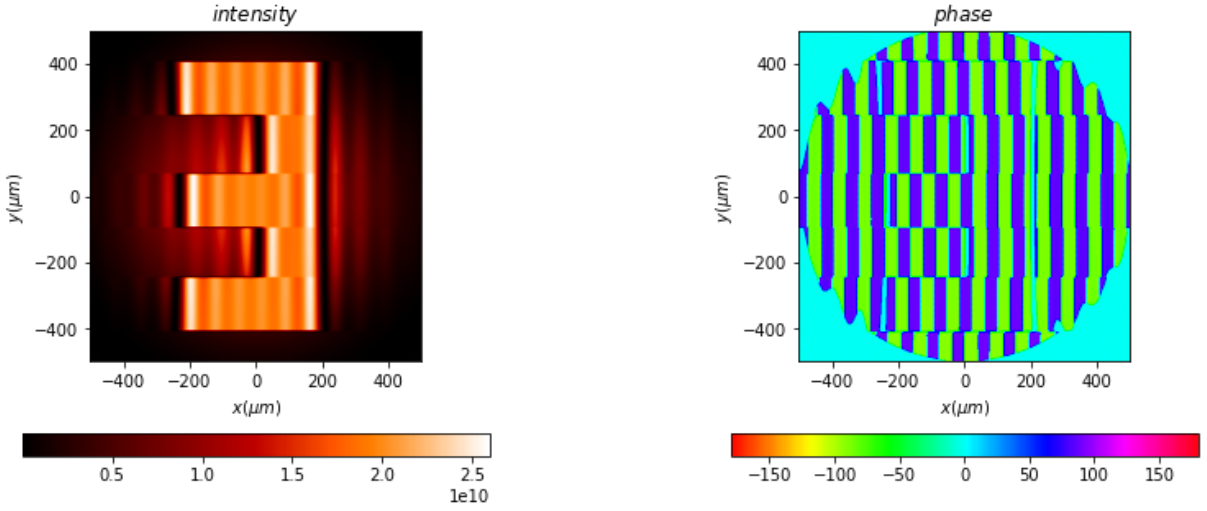


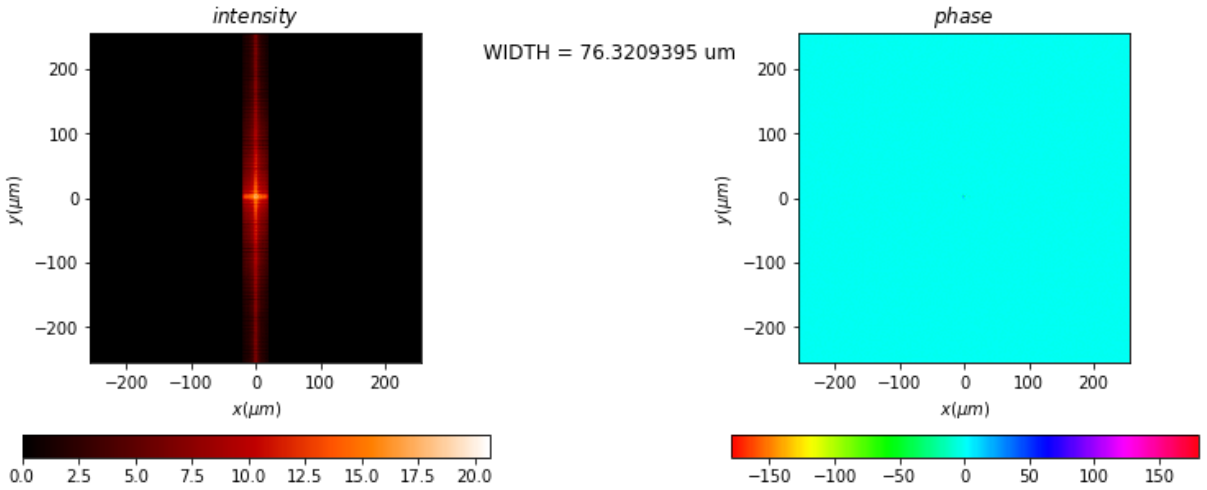


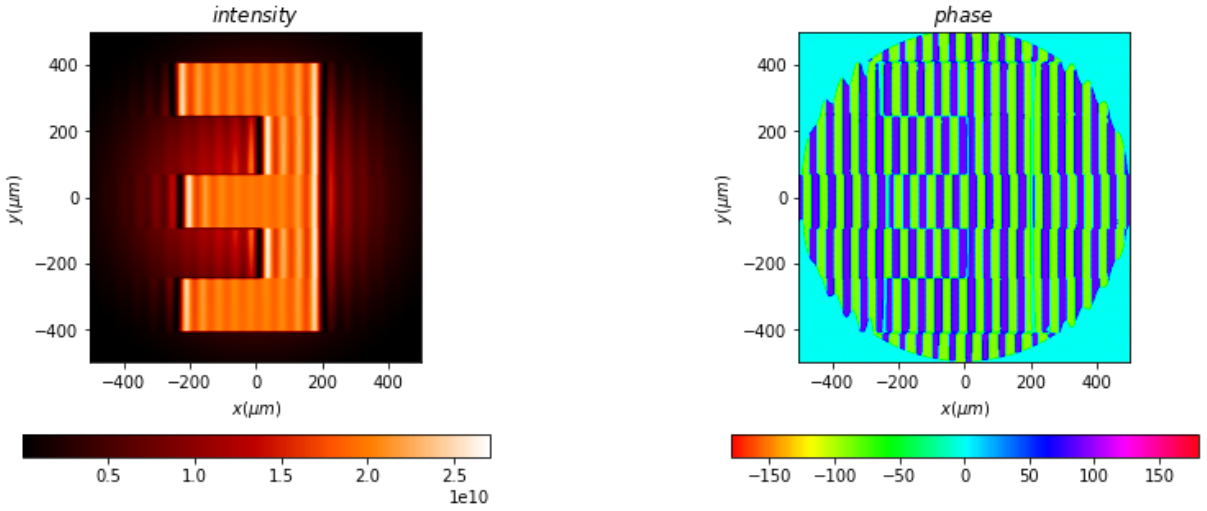

intensity
phase
WIDTH = 30 um
y(μm)
x(μm)
1e10

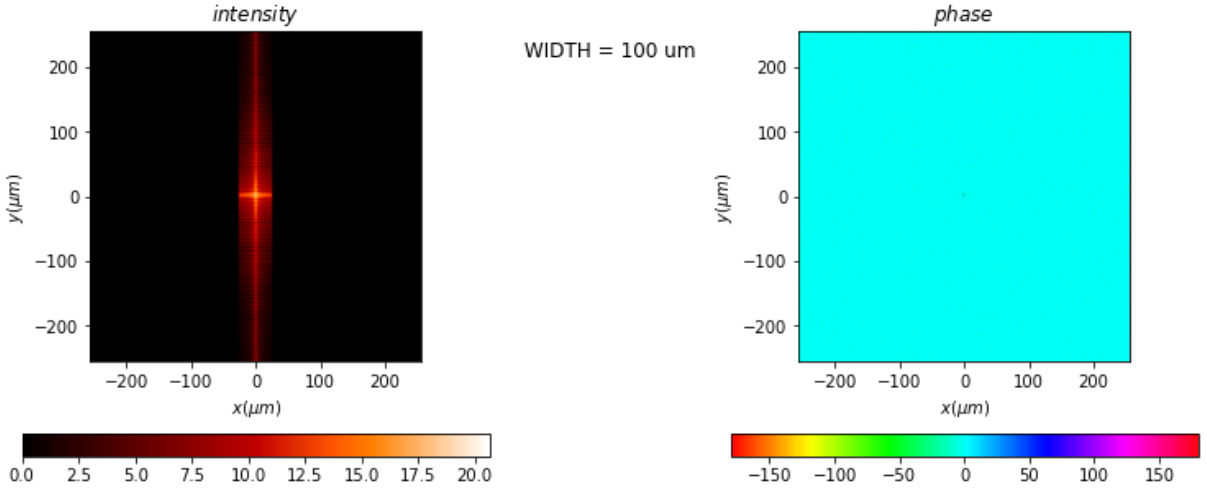

intensity
phase
y(μm)
x(μm)
1e10

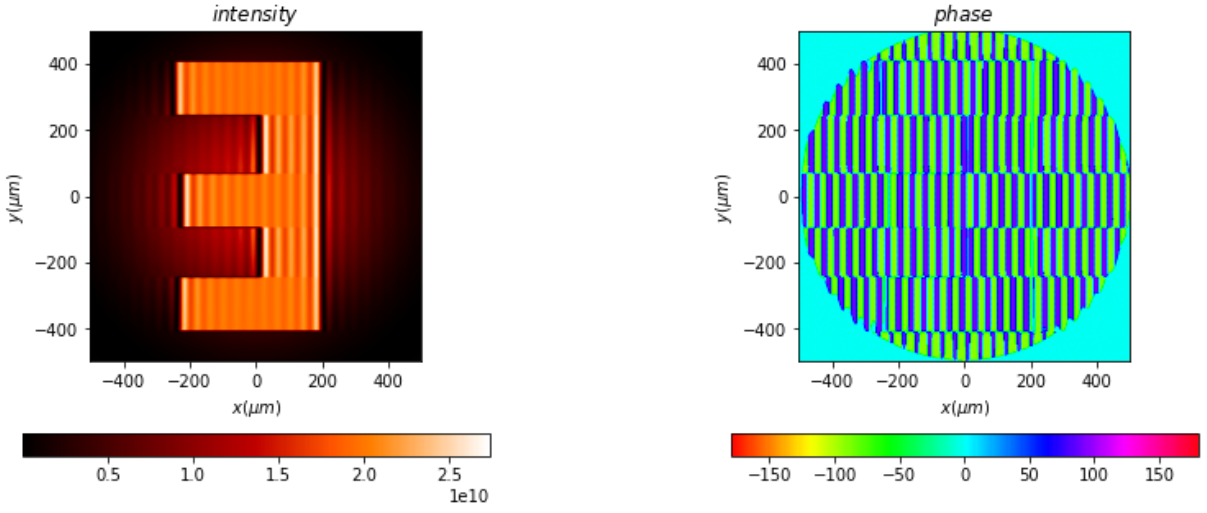

intensity
phase
WIDTH = 50 um
y(μm)
x(μm)
1e10

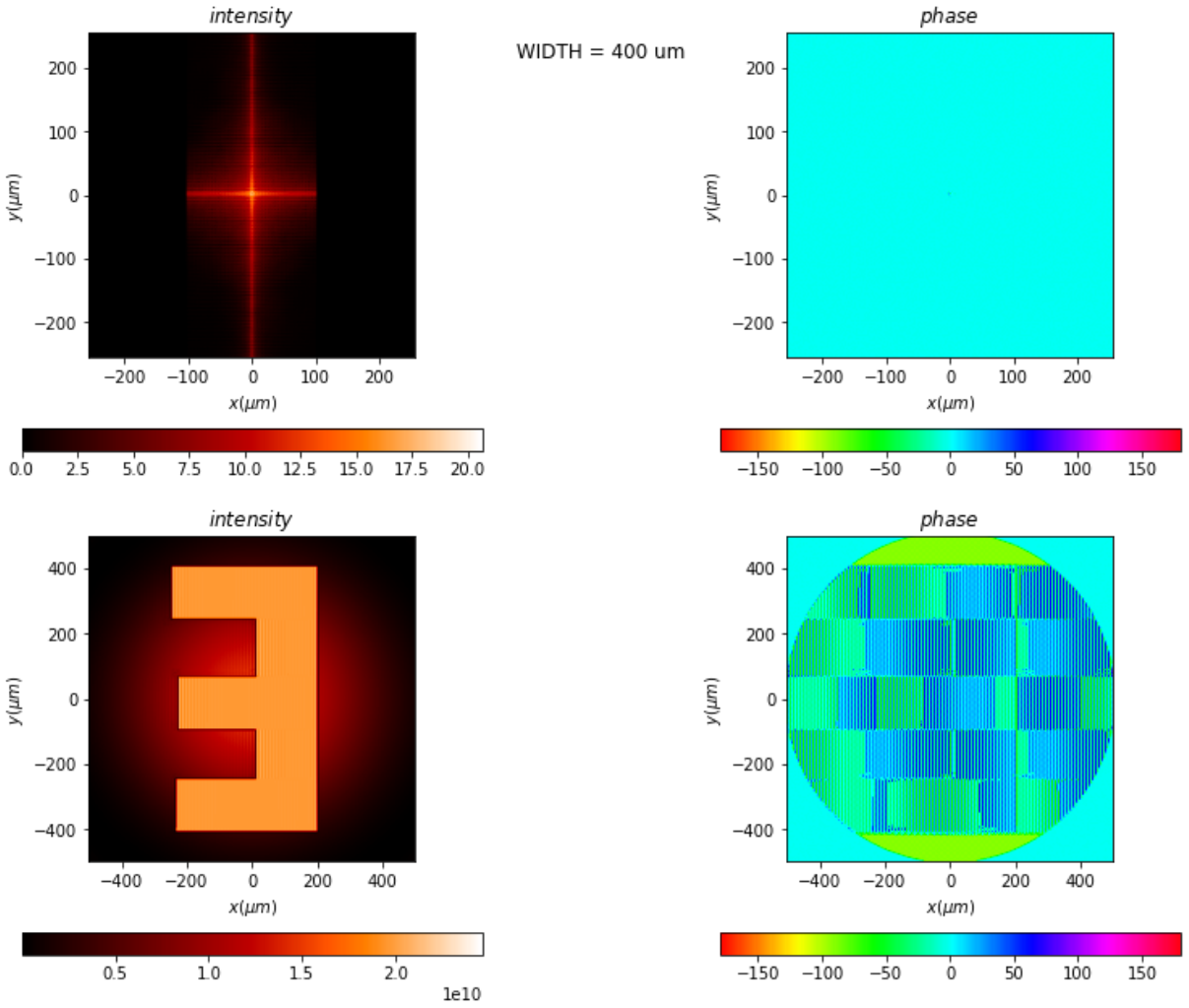

intensity
phase
y(μm)
x(μm)
1e10


intensity
phase
WIDTH = 76.3209395 um
y(μm)
x(μm)
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
−150 −100 −50 0 50 100 150


intensity
phase
y(μm)
x(μm)
0.5 1.0 1.5 2.0 2.5
1e10
−150 −100 −50 0 50 100 150


intensity
phase
WIDTH = 100 um
y(μm)
x(μm)
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
−150 −100 −50 0 50 100 150


intensity
phase
y(μm)
x(μm)
0.5 1.0 1.5 2.0 2.5
1e10
−150 −100 −50 0 50 100 150

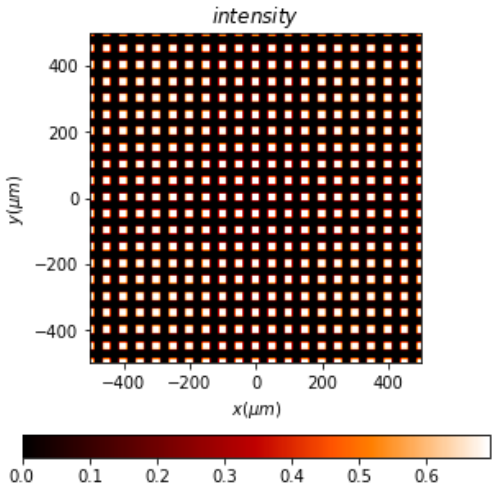


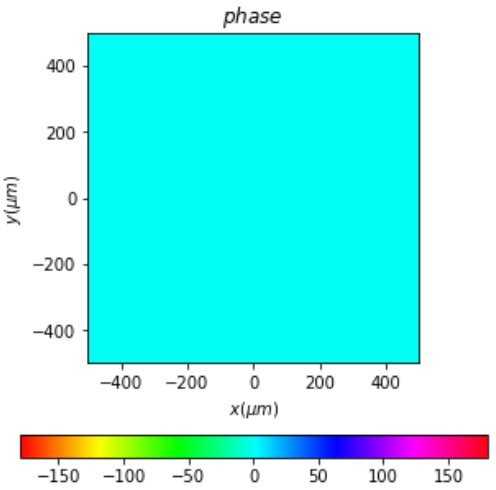
We can observe the "letter being assembled" from various modes, as higher frequencies are allowed to pass through.

Mesh

In [27]:
```
mesh = Scalar_mask_XY(x0, y0, wavelength)
mesh.grating_2D(
    period=50 * um,
    fill_factor=0.5,
    angle=0 * degrees
)

mesh.draw(kind='field', logarithm=True)
```

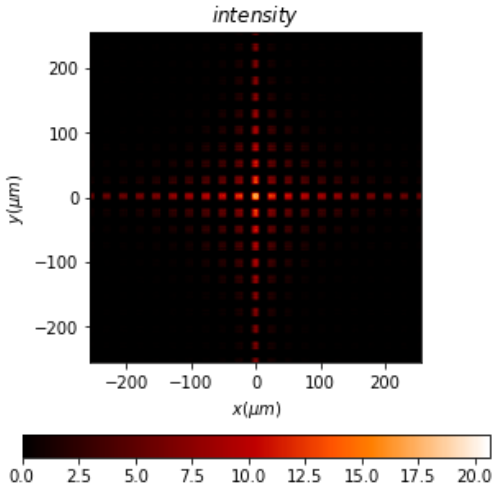
Out[27]:
```
((<matplotlib.image.AxesImage at 0x218e78d8fd0>,
  <matplotlib.image.AxesImage at 0x218e7b4e8b0>),
 None,
 None)
```

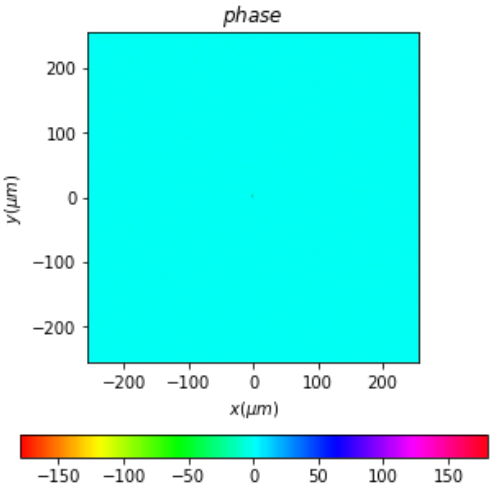


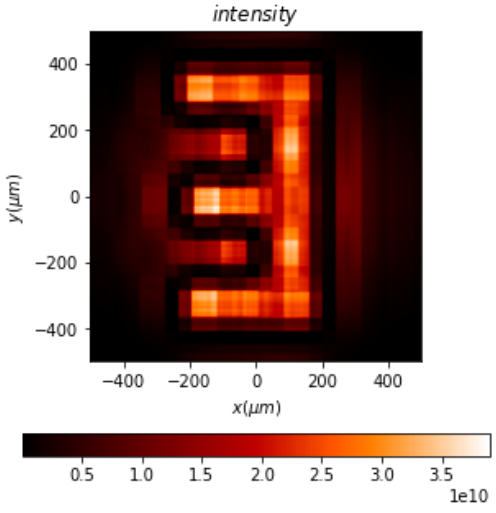


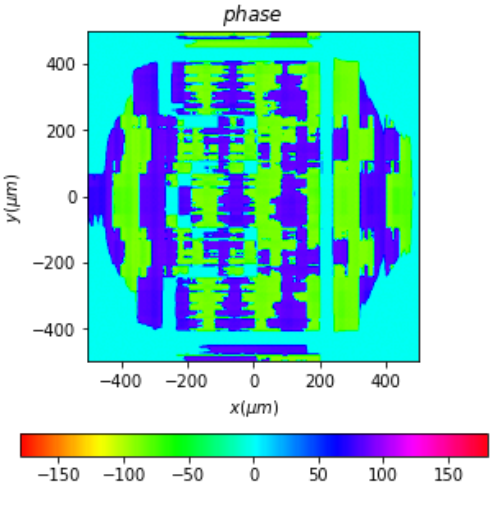
In [28]:
```
E_a_L1_Mesh, E_a_L1_Mesh_L2 = sim(E_a_L1, mesh)
E_a_L1_Mesh.draw(kind='field', logarithm=True)
E_a_L1_Mesh_L2.draw(kind='field', logarithm=False)
```

Out[28]:
```
((<matplotlib.image.AxesImage at 0x218e91c9c70>,
  <matplotlib.image.AxesImage at 0x218e9234ee0>),
 None,
 None)
```









One can obtain similar results with the "F" alphabet-mask.

Phase Constrast Imaging - Using phase information to filter overlapped images

In [1]:
```
import numpy as np
from diffractio import mm, um, degrees
from diffractio.scalar_sources_XY import Scalar_source_XY
from diffractio.scalar_masks_XY import Scalar_mask_XY


# Setting up
length = 1 * mm
num_data = 512
x0 = np.linspace(-length / 2, length / 2, num_data)
y0 = np.linspace(-length / 2, length / 2, num_data)
wavelength = 0.633 * um
```

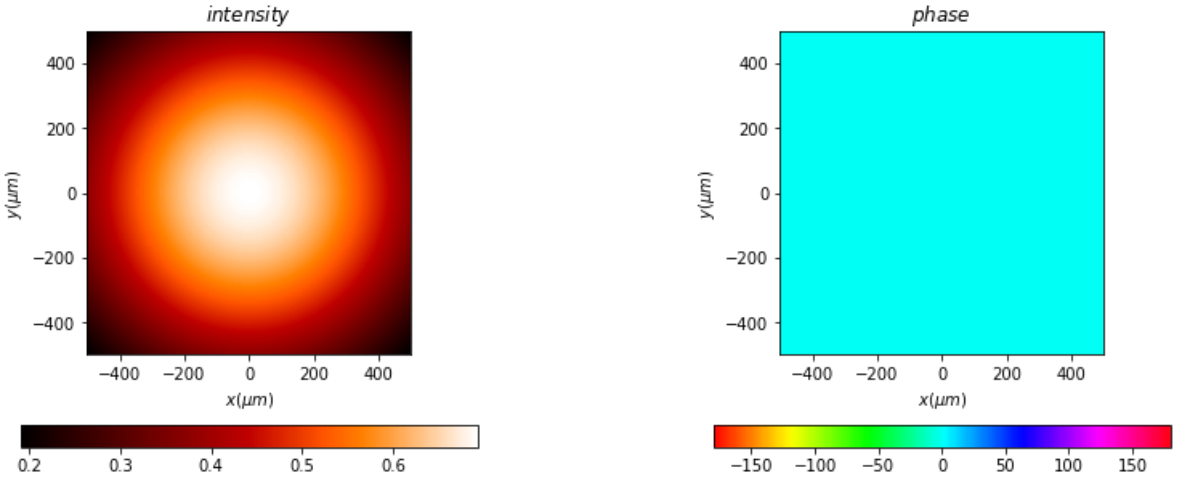
```
number of processors: 12
```

Setting up source

- Gaussian Beam (LASER)

In [2]:
```
# Gaussian Beam Source - like a LASER
u0 = Scalar_source_XY(x=x0, y=y0, wavelength=wavelength)
u0.gauss_beam(r0=(0, 0), w0=(800 * um, 800 * um), z0=0.0)
u0.draw(kind='field', logarithm=True)
```

Out[2]:
```
((<matplotlib.image.AxesImage at 0x14e0a72da60>,
  <matplotlib.image.AxesImage at 0x14e0a9cc220>),
 None,
 None)
```



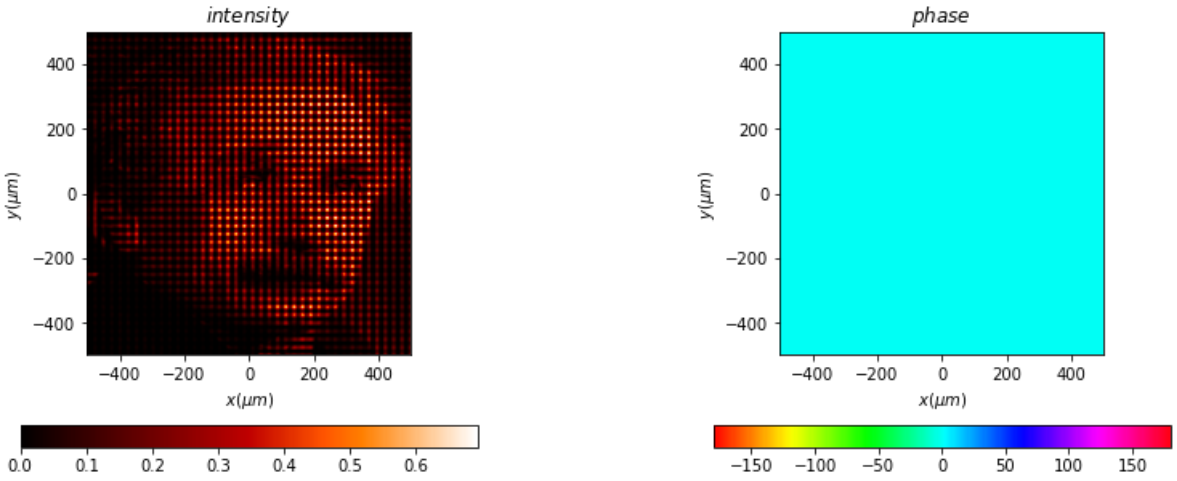
In [3]:
```
# Plane Wave Source - Just an experiment
# u1 = Scalar_source_XY(x=x0, y=y0, wavelength=wavelength)
# u1.plane_wave()
# u1.draw(kind='field', logarithm=True)
```

Image - Phase Contrast

Composition of Horizontal and Vertical Lines

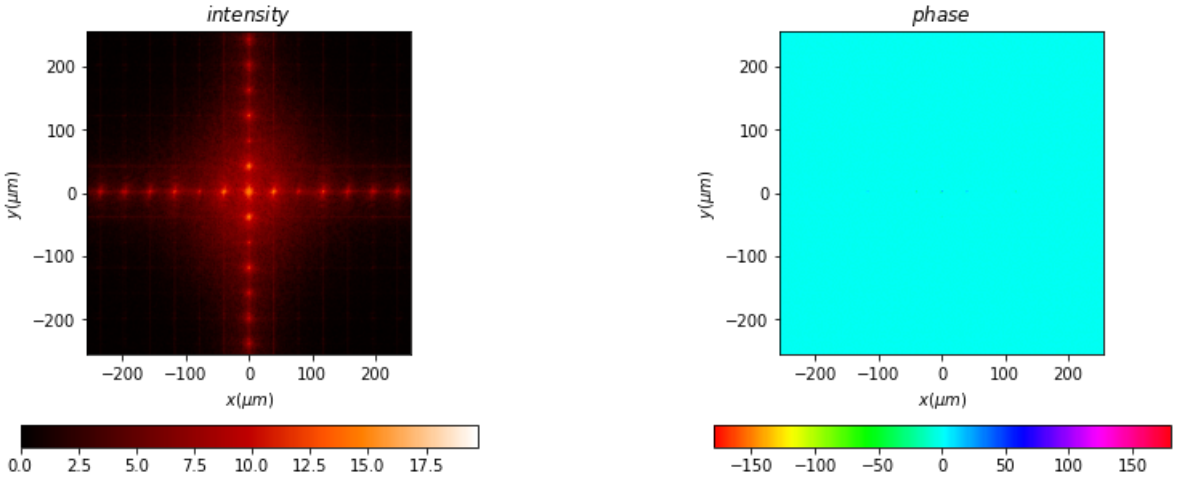
In [4]:
```
compLines = Scalar_mask_XY(x0, y0, wavelength)
compLines.image(
    filename="complines-4.png",
    normalize=True,
    canal=2,
)

compLines.draw(kind='field', logarithm=True)
# compLines.draw(kind='real_field', logarithm=True)
# compLines.draw(kind='amplitude', logarithm=True)
```

Out[4]:
```
((<matplotlib.image.AxesImage at 0x14e0b1f11c0>,
  <matplotlib.image.AxesImage at 0x14e0b452940>),
 None,
 None)
```



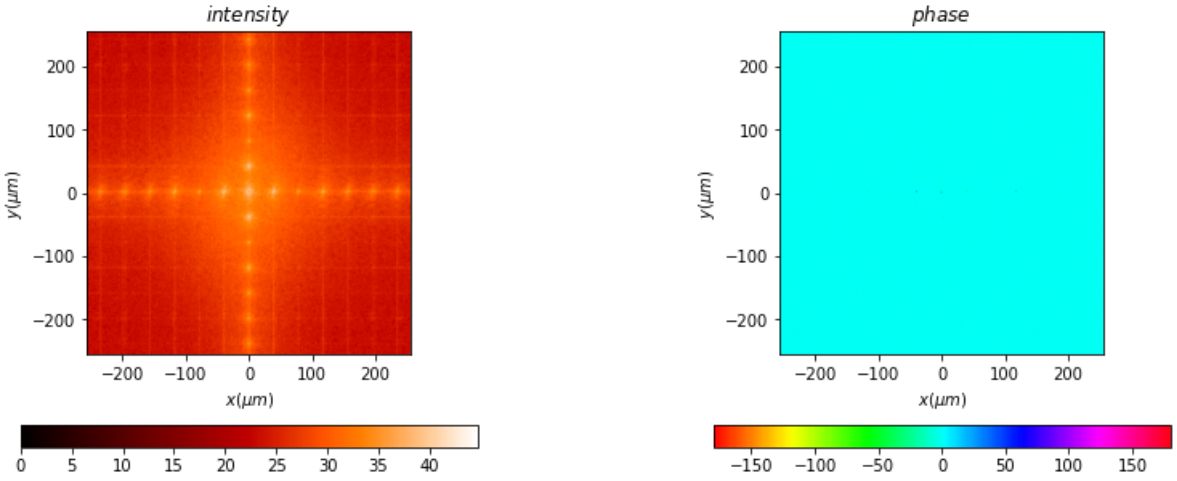
In [5]:
```
a_L1 = (u0 * compLines).fft(z=1 * mm, new_field=True)
a_L1.draw(kind='field', logarithm=True)
# a_L1.draw(kind='amplitude', logarithm=False)
```

Out[5]:
```
((<matplotlib.image.AxesImage at 0x14e0b06c880>,
  <matplotlib.image.AxesImage at 0x14e0b111070>),
 None,
 None)
```



In [6]:
```
a_L2_NF = (a_L1.fft(z=1 * mm, shift=False, remove0=False, new_field=True)).fft(z=1 * mm, shift=False, remove0=
a_L2_NF.draw(kind='field', logarithm=True)
# a_L2_NF.draw(kind='real_field', logarithm=True)
```

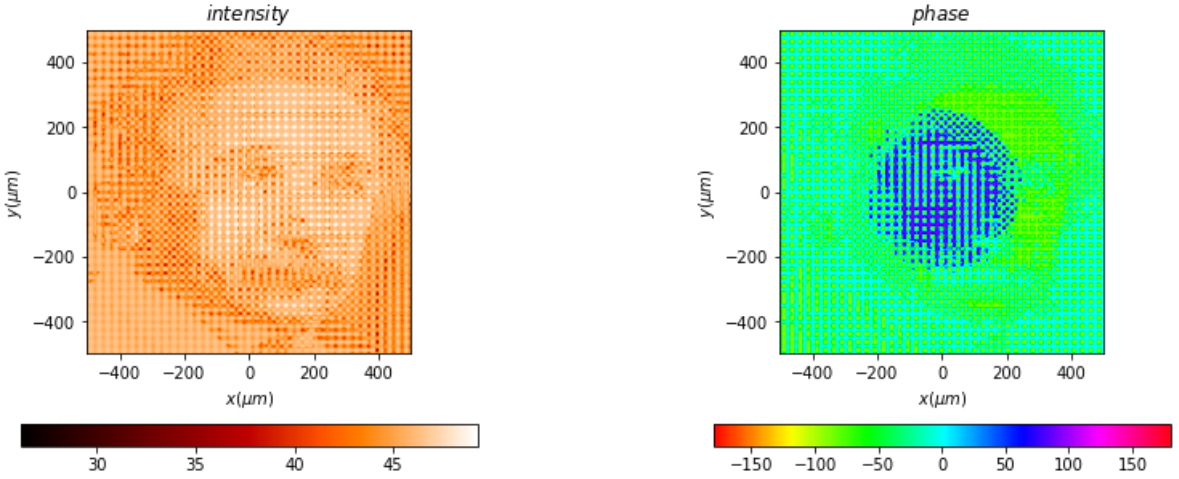
Out[6]:
```
((<matplotlib.image.AxesImage at 0x14e0b8b3c70>,
  <matplotlib.image.AxesImage at 0x14e0fd98460>),
 None,
 None)
```



In [7]:
```
IFFT_NF = a_L2_NF.fft(z=1 * mm, shift=False, remove0=False, new_field=True)
IFFT_NF.draw(kind='field', logarithm=True)
```

Out[7]:
```
((<matplotlib.image.AxesImage at 0x14e0fe49fd0>,
```

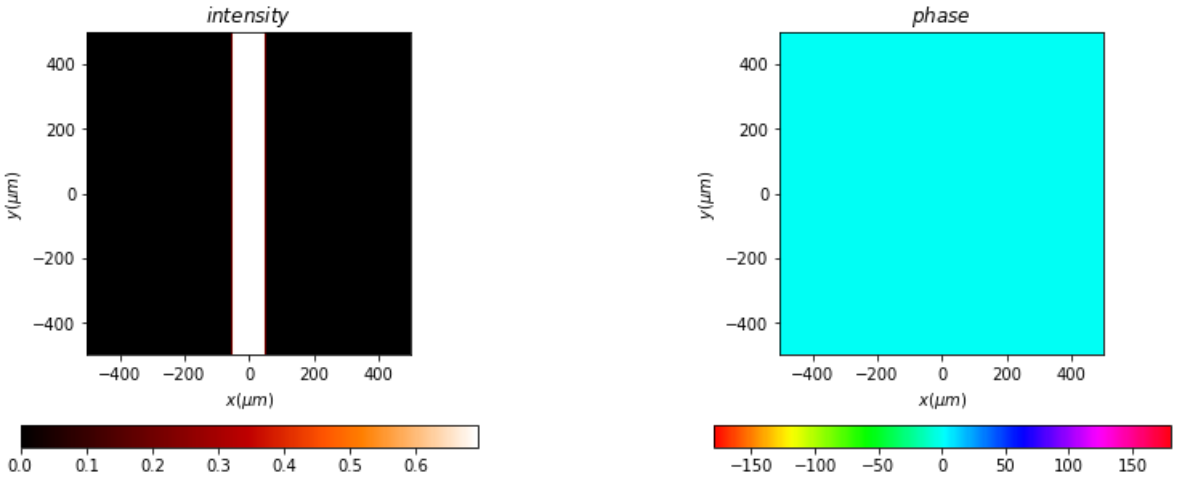
```
 <matplotlib.image.AxesImage at 0x14e10712790>),
None,
None)
```



In [8]:
```
csize = 100  # Size of slit
```

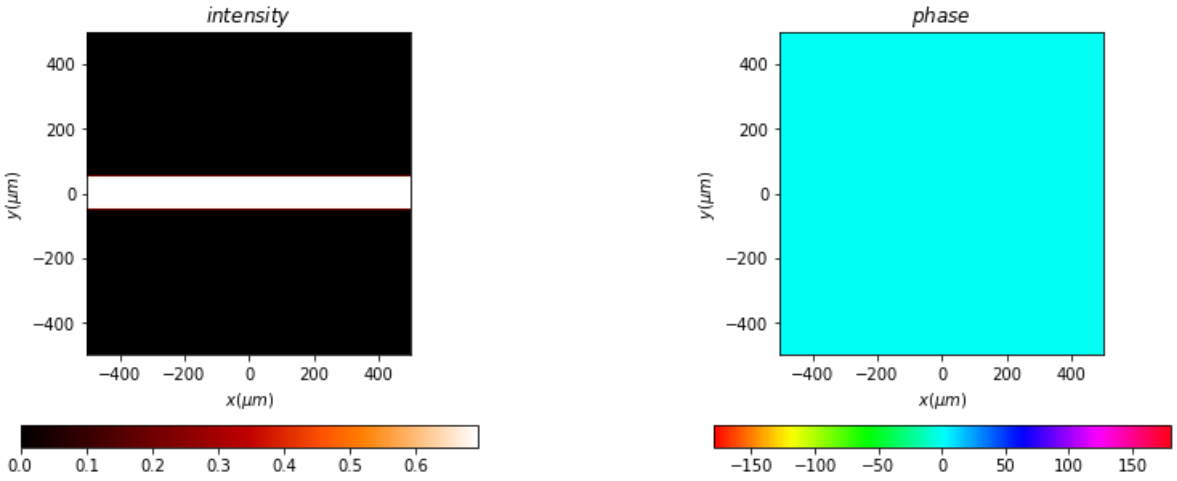
In [9]:
```
vertSlit = Scalar_mask_XY(x0, y0, wavelength)
vertSlit.slit(
    x0=0 * um,
    size=csize * um
)
vertSlit.draw(kind='field', logarithm=True)
```

Out[9]:
```
((<matplotlib.image.AxesImage at 0x14e0b45e340>,
  <matplotlib.image.AxesImage at 0x14e0b0aa970>),
None,
None)
```



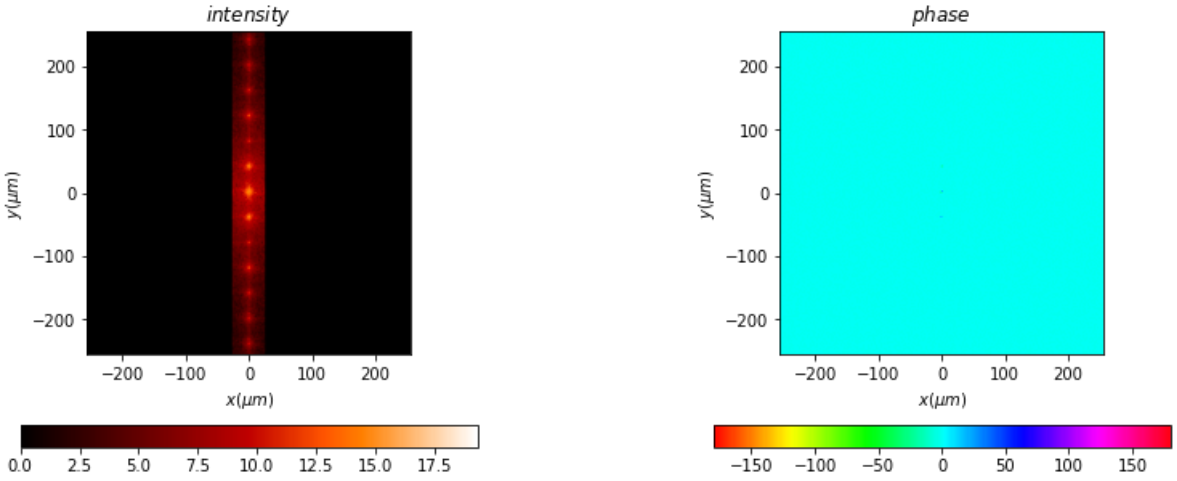
In [10]:
```
horizSlit = Scalar_mask_XY(x0, y0, wavelength)
horizSlit.slit(
    x0=0 * um,
    size=csize * um,
    angle=np.pi / 2
)
horizSlit.draw(kind='field', logarithm=True)
```

Out[10]:
```
((<matplotlib.image.AxesImage at 0x14e112b4eb0>,
  <matplotlib.image.AxesImage at 0x14e1136b670>),
None,
None)
```



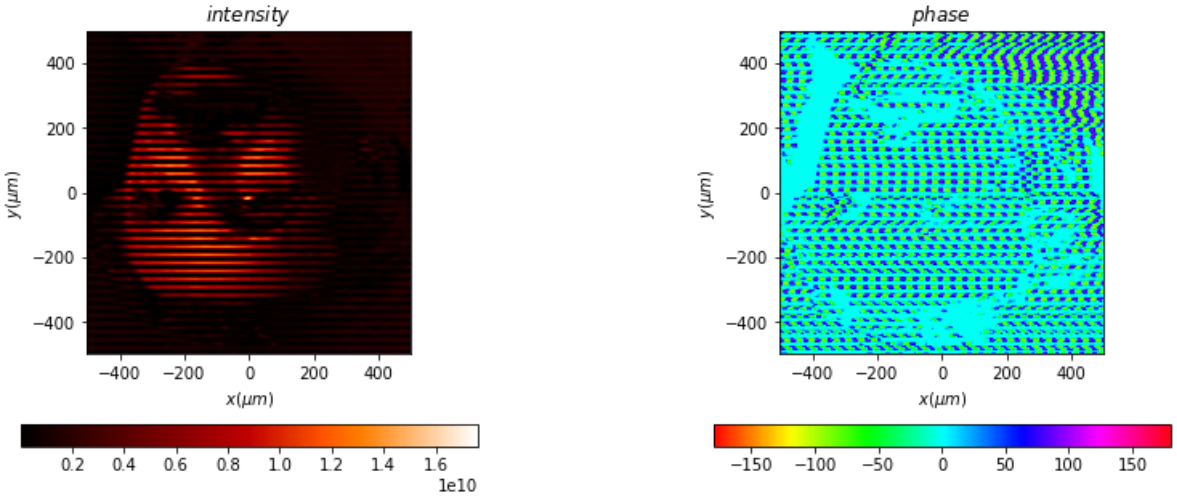
In [11]:
```
b_L2_a_vS = a_L1 * vertSlit
b_L2_a_vS.draw(kind='field', logarithm=True)
# b_L2_a_vS.draw(kind='amplitude', logarithm=True)
```

Out[11]:
```
((<matplotlib.image.AxesImage at 0x14e12921490>,
  <matplotlib.image.AxesImage at 0x14e12983c40>),
 None,
 None)
```



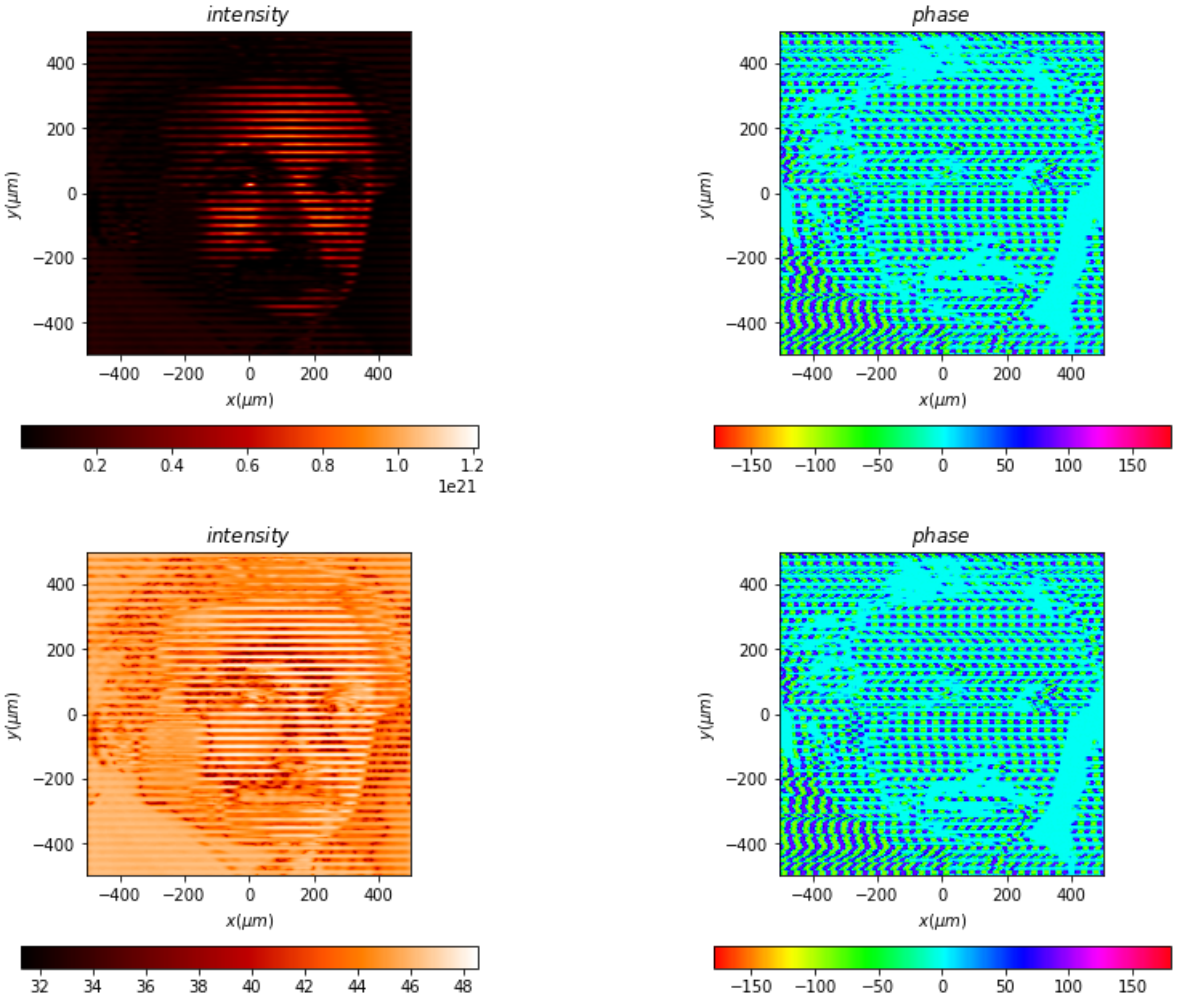
In [12]:
```
a_L2_vS = b_L2_a_vS.fft(z=1 * mm, shift=False, remove0=False, new_field=True)
a_L2_vS.draw(kind='field', logarithm=False)
# a_L2_vS.draw(kind='intensity', logarithm=True)
```

Out[12]:
```
((<matplotlib.image.AxesImage at 0x14e12a45760>,
  <matplotlib.image.AxesImage at 0x14e1331b7c0>),
 None,
 None)
```



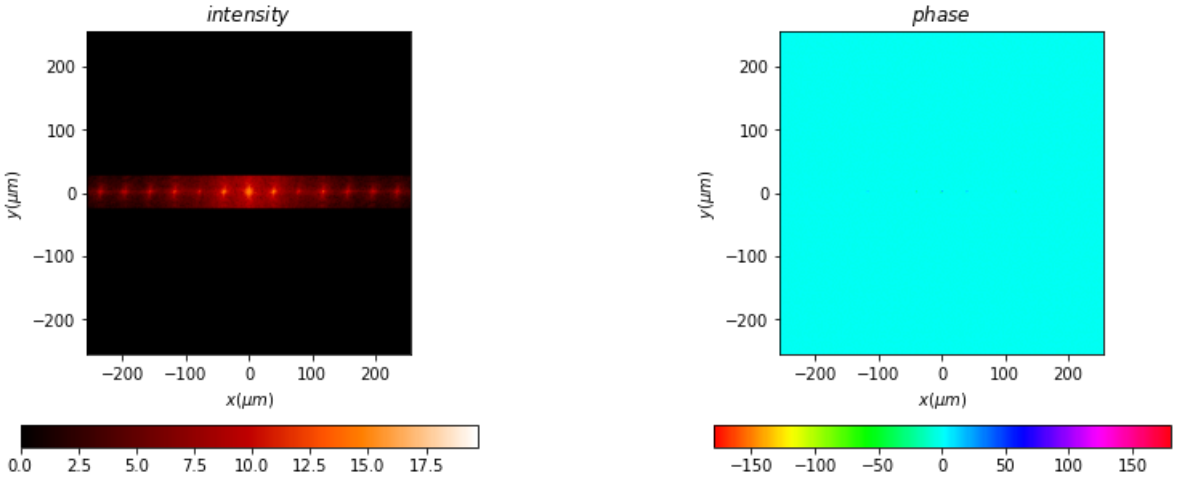
In [13]:
```
IFFT_vS = (a_L2_vS.fft(z=1 * mm, shift=False, remove0=False, new_field=True)).fft(z=1 * mm, shift=False, remov
IFFT_vS.draw(kind='field', logarithm=False)
IFFT_vS.draw(kind='field', logarithm=True)
```

Out[13]:
```
((<matplotlib.image.AxesImage at 0x14e0b0c38e0>,
  <matplotlib.image.AxesImage at 0x14e128b60d0>),
 None,
 None)
```



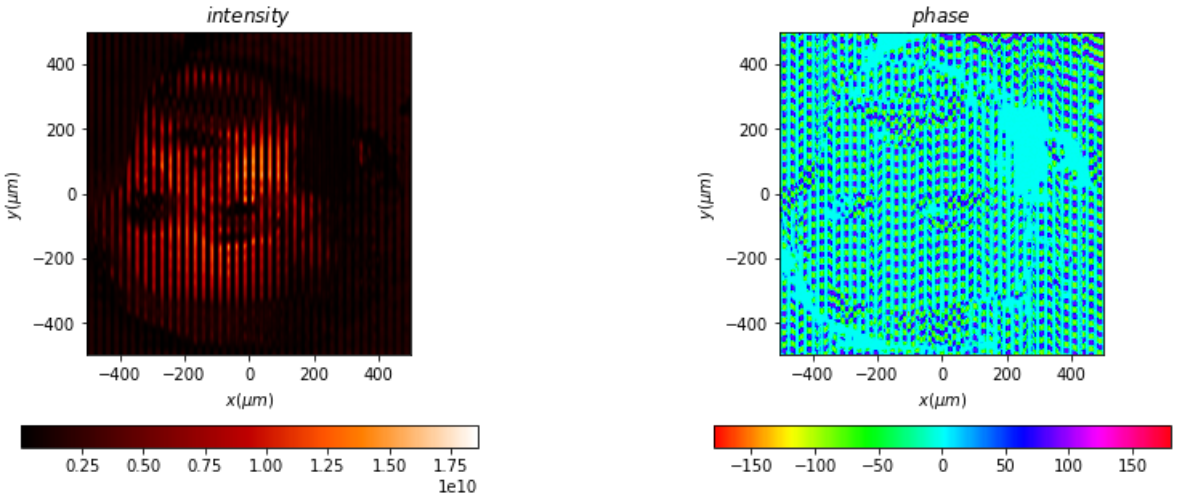
In [14]:
```
b_L2_a_hS = a_L1 * horizSlit
b_L2_a_hS.draw(kind='field', logarithm=True)
# b_L2_a_hS.draw(kind='amplitude', logarithm=True)
```

Out[14]:
```
((<matplotlib.image.AxesImage at 0x14e14874a00>,
  <matplotlib.image.AxesImage at 0x14e148ff220>),
 None,
 None)
```



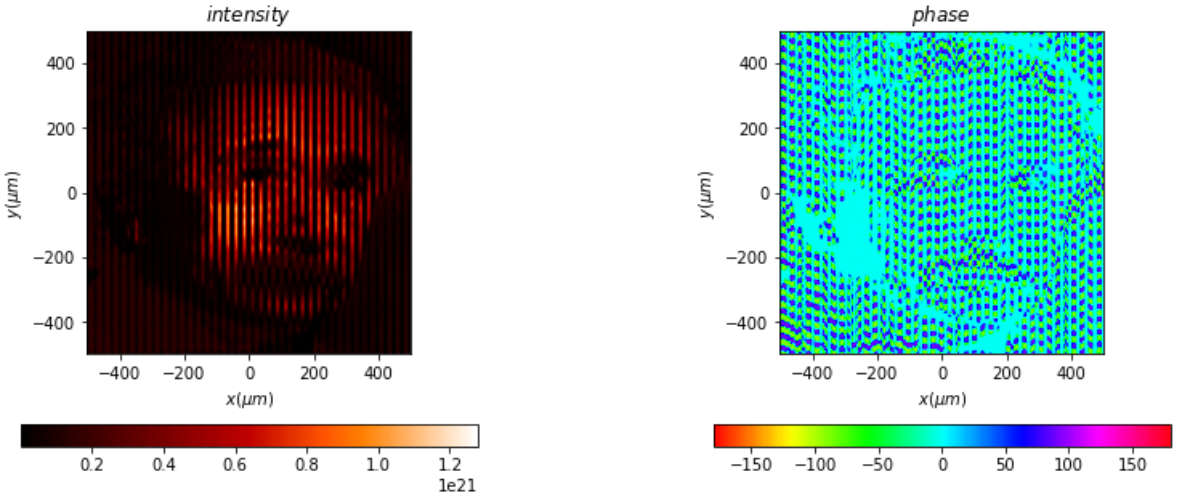
In [15]:
```
a_L2_hS = b_L2_a_hS.fft(z=1 * mm, shift=False, remove0=False, new_field=True)
a_L2_hS.draw(kind='field', logarithm=False)
# a_L2_hS.draw(kind='intensity', logarithm=True)
```

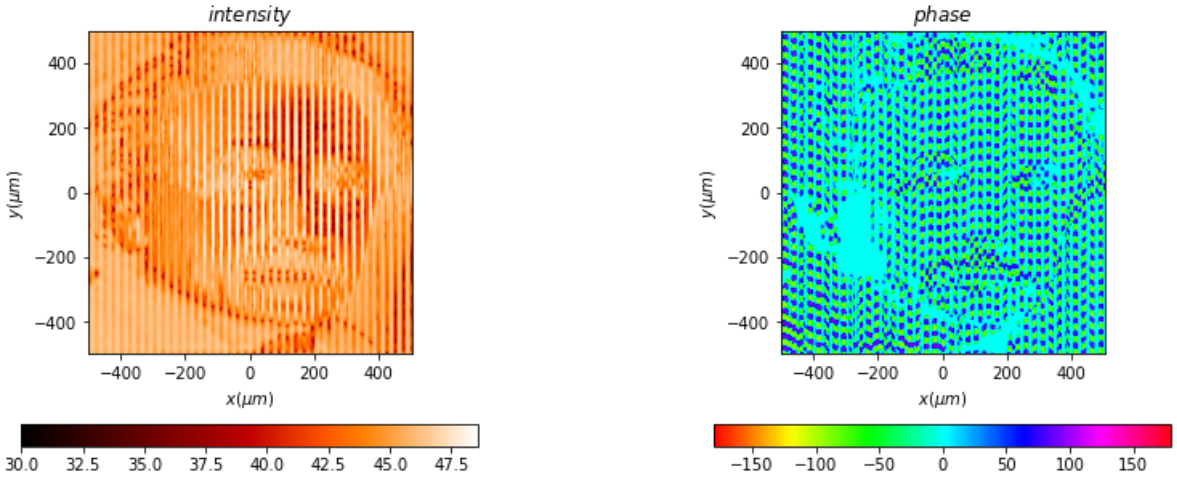
Out[15]:
```
((<matplotlib.image.AxesImage at 0x14e16c12c40>,
  <matplotlib.image.AxesImage at 0x14e16c98400>),
 None,
 None)
```



In [16]:
```
IFFT_hS = (a_L2_hS.fft(z=1 * mm, shift=False, remove0=False, new_field=True)).fft(z=1 * mm, shift=False, remov
IFFT_hS.draw(kind='field', logarithm=False)
IFFT_hS.draw(kind='field', logarithm=True)
```

Out[16]:
```
((<matplotlib.image.AxesImage at 0x14e182461f0>,
  <matplotlib.image.AxesImage at 0x14e182b04f0>),
 None,
 None)
```





In [ ]: