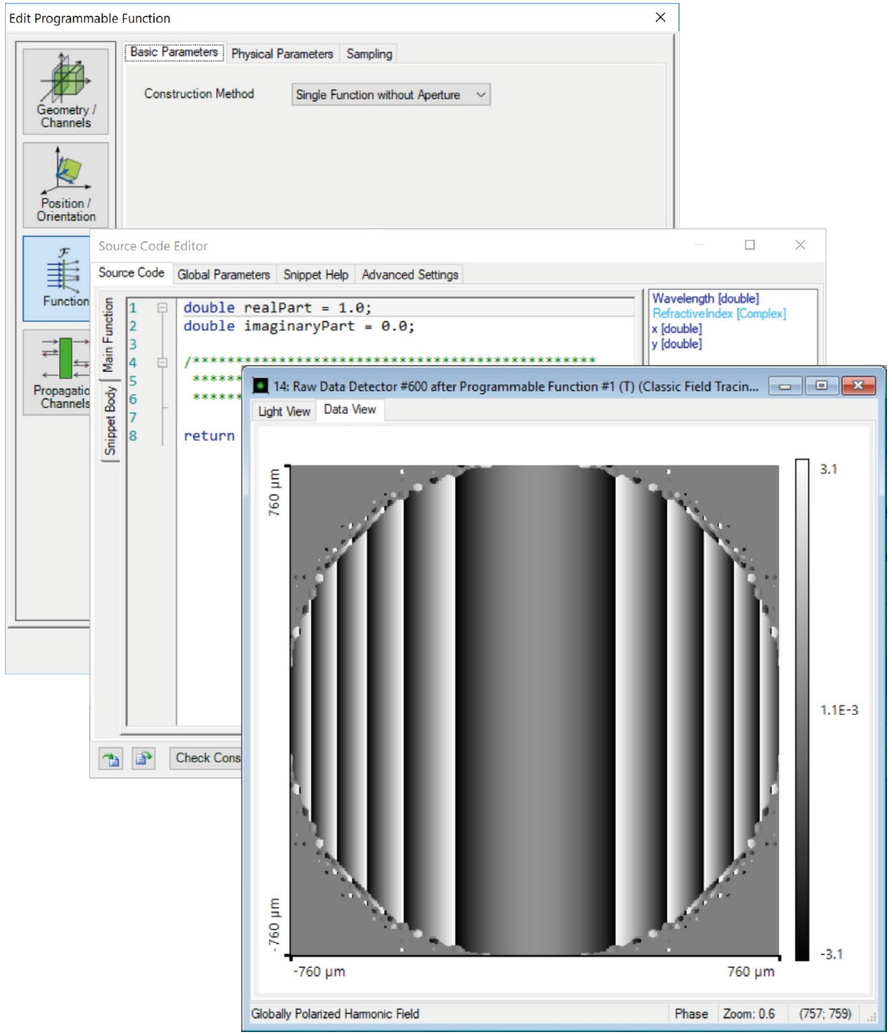
How to Work with the Programmable Function & Example (Cylindrical Lens)

Abstract

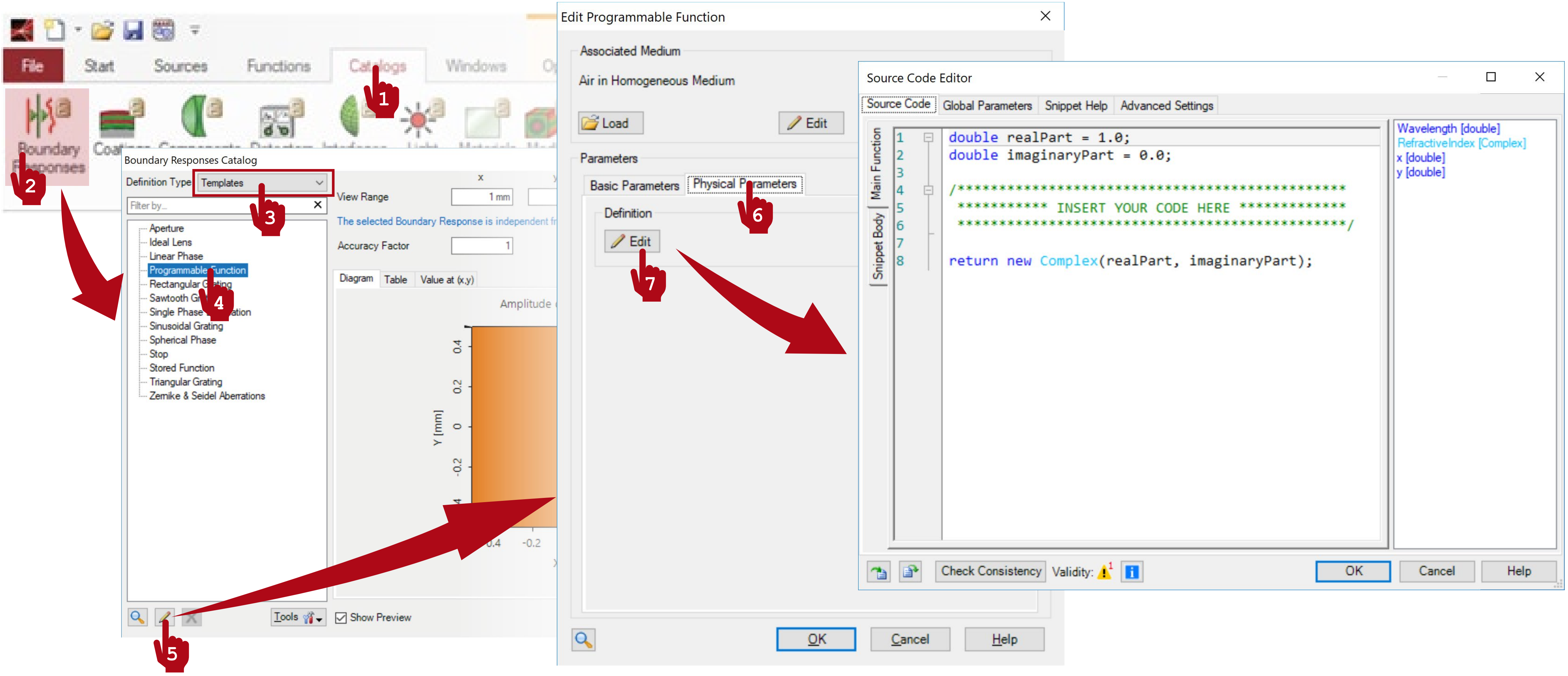


Construction Method: Single Function without Aperture

```
double realPart = 1.0;
double imaginaryPart = 0.0;

/*******************************************************
 ******
 ******

return
```

Providing maximum versatility for your optical simulations is one of our most fundamental objectives. In this tutorial we explain how to work with the Programmable Function, which can also be thought of as an idealised component acting in a single plane: the workflow entails defining a position-dependent, complex-valued function on the *x*, *y* plane, which is then multiplied onto the incoming field. We use the example of an ideal cylindrical lens to go through the whole process in detail.

Where to Find the Programmable Function: Catalog



Definition Type: Templates

View Range: x 1 mm

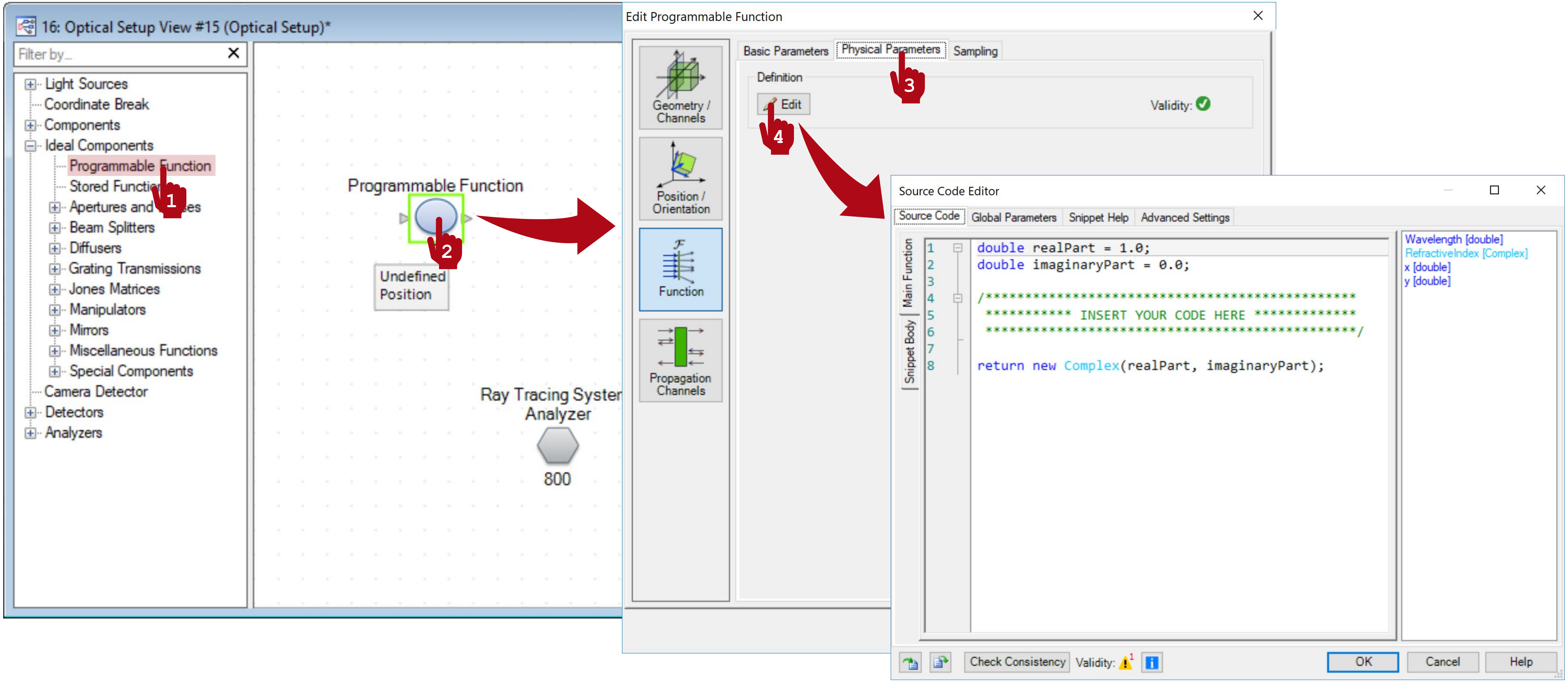
The selected Boundary Response is independent fr

Accuracy Factor: 1

Associated Medium: Air in Homogeneous Medium

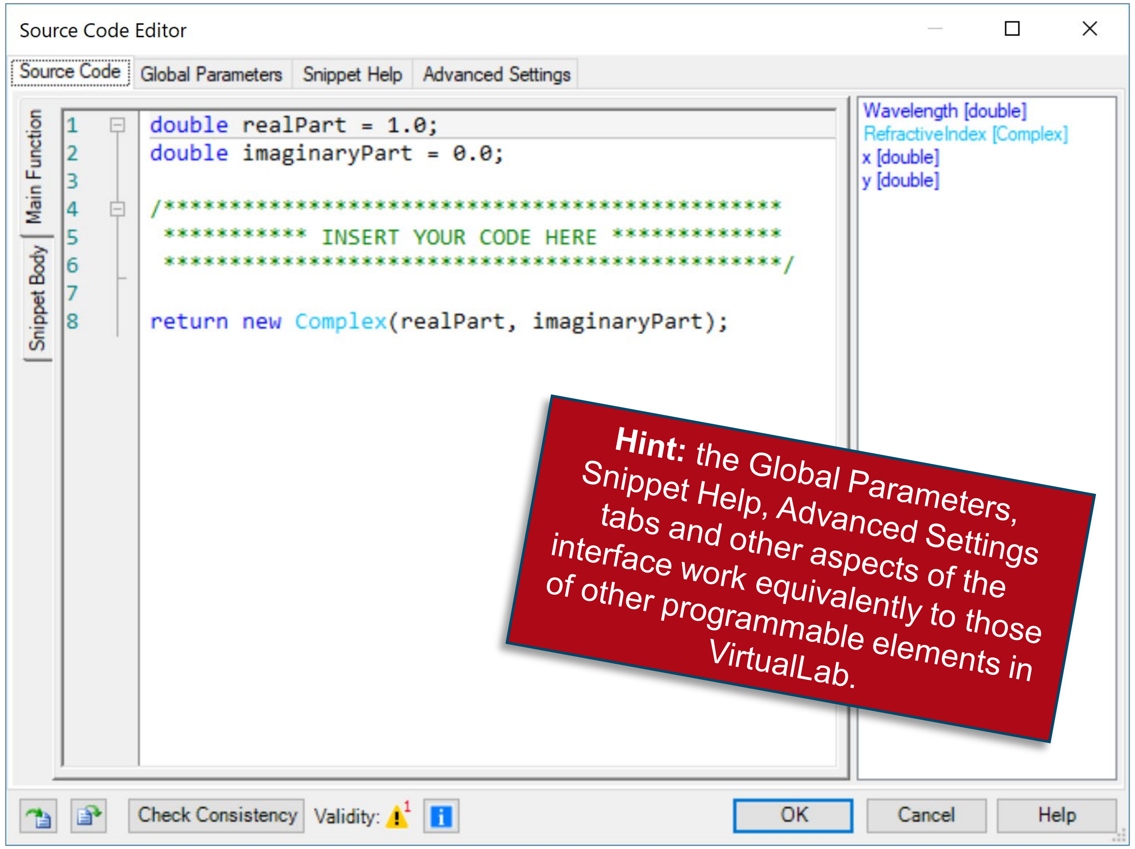
```
double realPart = 1.0;
double imaginaryPart = 0.0;

/*******************************************
 *********** INSERT YOUR CODE HERE *************
 *******************************************/

return new Complex(realPart, imaginaryPart);
```

Where to Find the Programmable Function: Optical Setup



```
double realPart = 1.0;
double imaginaryPart = 0.0;

/*********************************************
 *********** INSERT YOUR CODE HERE ************
 *********************************************/

return new Complex(realPart, imaginaryPart);
```

Writing the Code

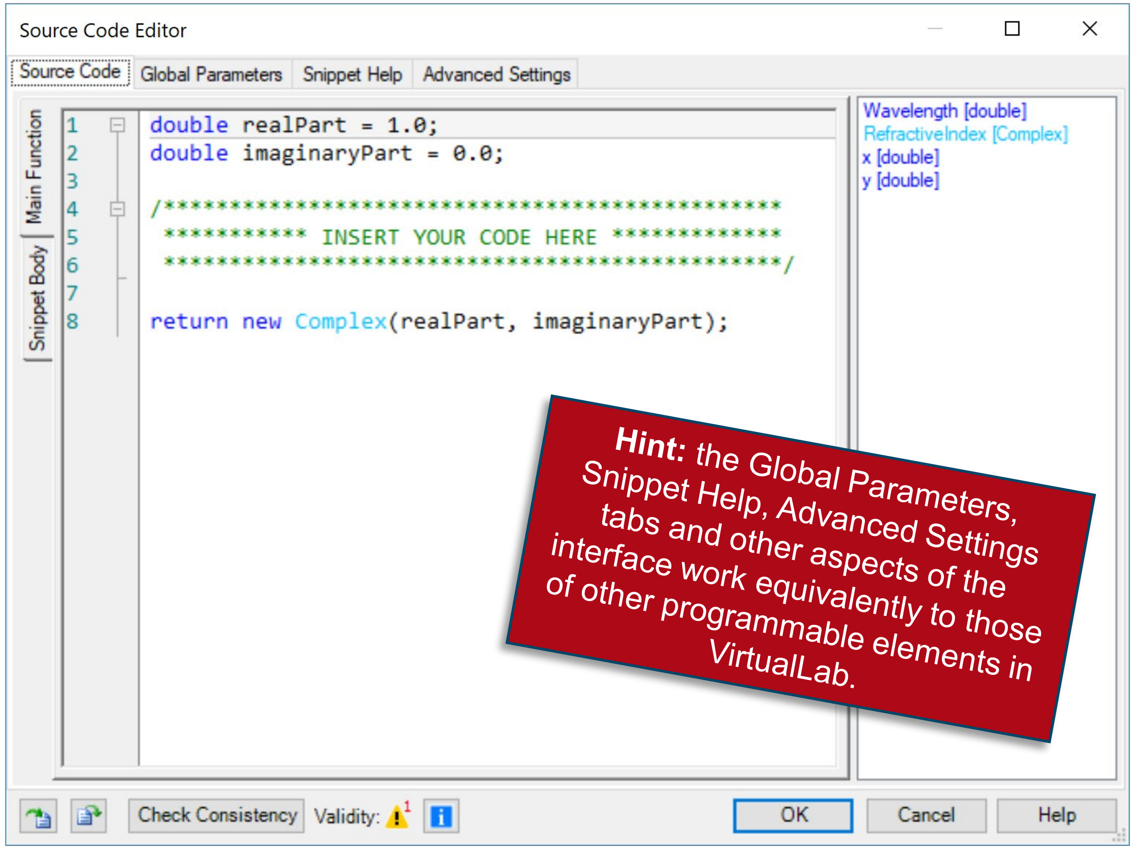


```
double realPart = 1.0;
double imaginaryPart = 0.0;

/*****************************************
 *********** INSERT YOUR CODE HERE *************
 ******************************************/

return new Complex(realPart, imaginaryPart);
```

Hint: the Global Parameters, Snippet Help, Advanced Settings tabs and other aspects of the interface work equivalently to those of other programmable elements in VirtualLab.

- The panel on the right shows a list of available independent parameters.
- `Wavelength` is a default independent parameter that permits the user to implement a dispersive ideal component (function).
- `RefractiveIndex` is another default independent parameter that reads the complex-valued refractive index of the embedding medium.
- Finally, `x` and `y` are the last two default independent parameters. They span the plane on which the ideal component (function) is defined.

Writing the Code



```
double realPart = 1.0;
double imaginaryPart = 0.0;

/*********************************************
 *********** INSERT YOUR CODE HERE *************
 **********************************************/

return new Complex(realPart, imaginaryPart);
```

Hint: the Global Parameters, Snippet Help, Advanced Settings tabs and other aspects of the interface work equivalently to those of other programmable elements in VirtualLab.

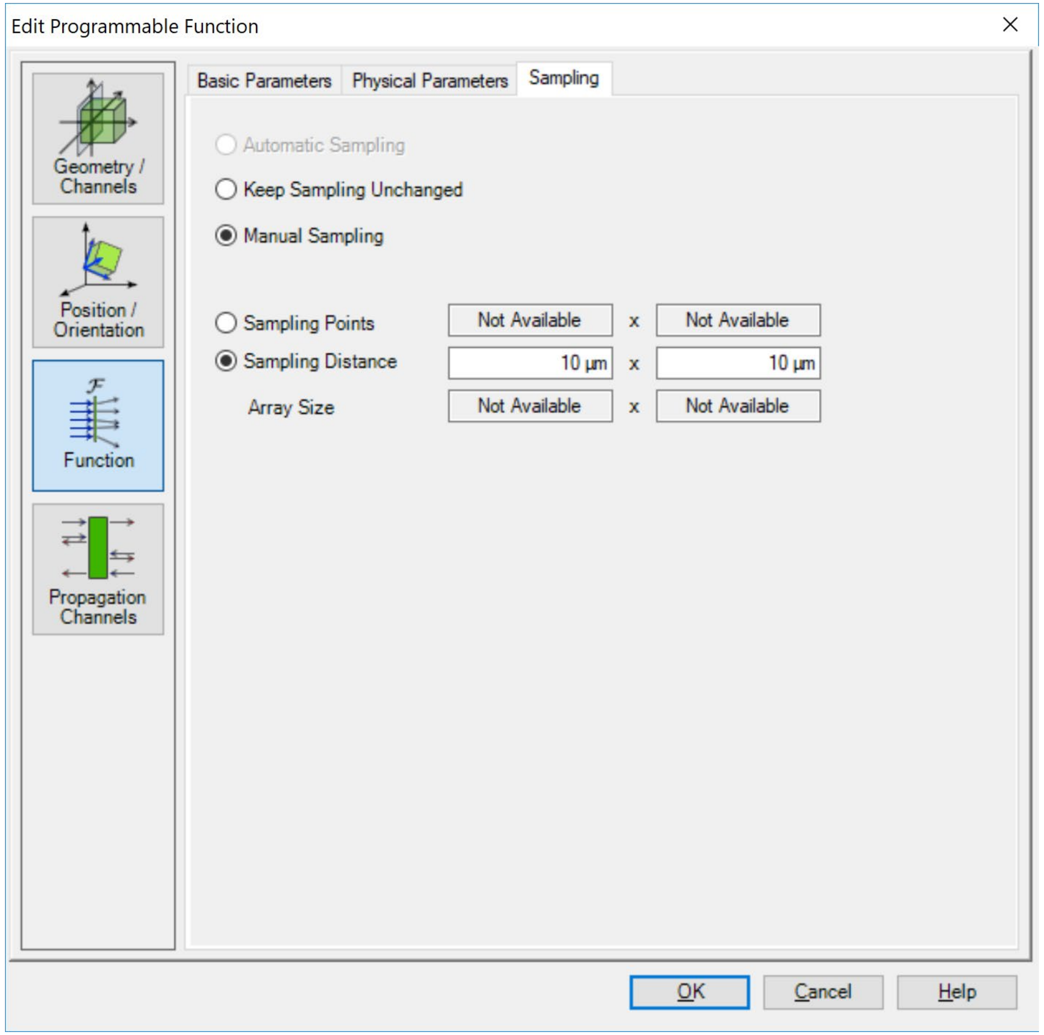
- The Main Function must return a `Complex` value per `x`, `y` (possibly also `Wavelength`) which will then be multiplied onto the incoming field.
- Use the Snippet Body to group parts of the code in support functions.
- Note that it is possible to use an imported reference field and/or stack, and their associated parameters, in the code of the Programmable Function. The reference field and stack can be defined in the Global Parameters tab.

Sampling

- The user must ensure that the sampling (of the field behind the component) is fine enough to resolve the frequencies introduced by the programmable function.

- Use the Sampling tab for this purpose.

- Please note that the sampling may depend on the actual values of the defined global parameters.



○ Automatic Sampling

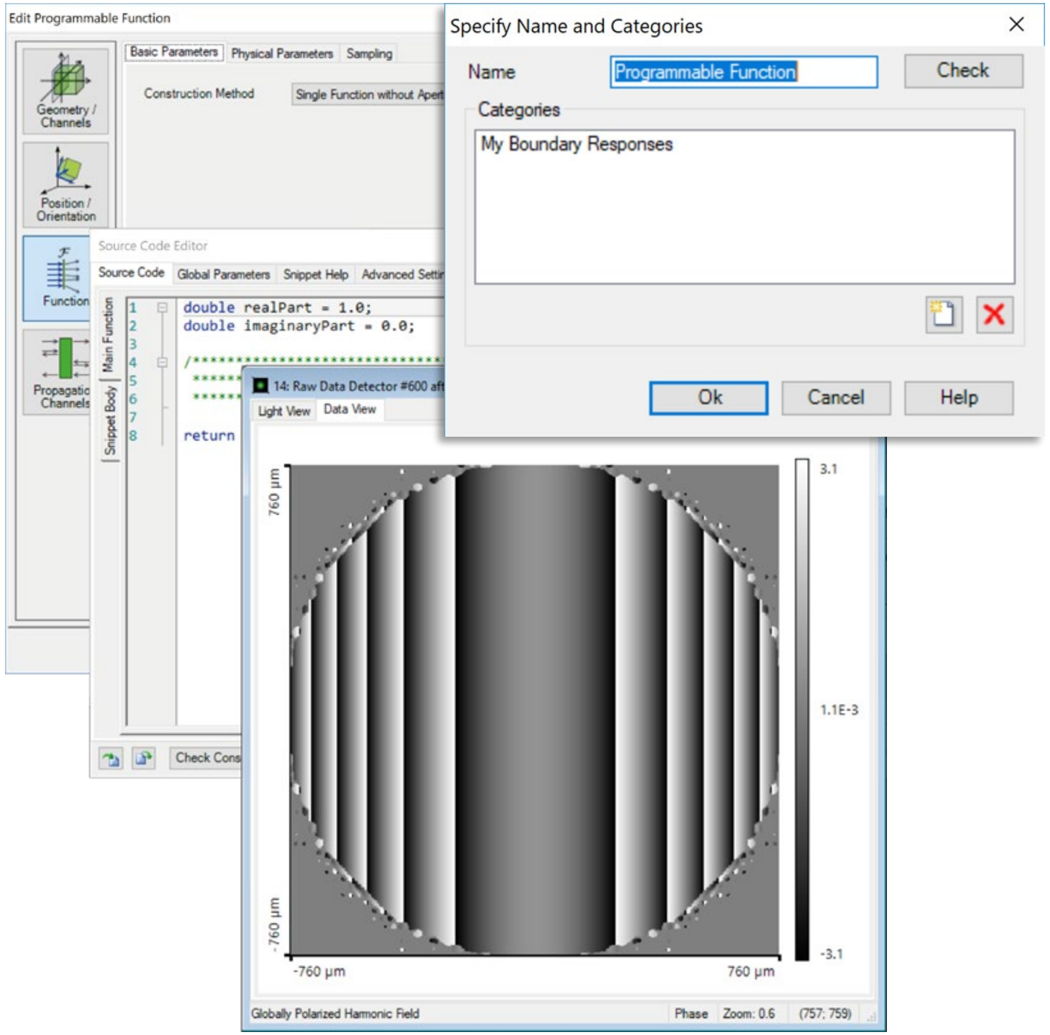
○ Keep Sampling Unchanged

● Manual Sampling

○ Sampling Points	Not Available	x	Not Available
● Sampling Distance	10 µm	x	10 µm
Array Size	Not Available	x	Not Available

Output

- The Programmable Function yields a complex-valued function per wavelength defined on a plane, spanned by *x*, *y*.
- In an Optical Setup, it is multiplied onto the incoming field.
- Hint: a snippet which has been programmed for a function can be employed also in the Programmable Source, and vice versa.
- The function can be saved in the Boundary Responses catalog for later use.



Construction Method: Single Function without Apert

```
double realPart = 1.0;
double imaginaryPart = 0.0;

/*****************************
 ******
 ******

return
```

Name: Programmable Function

Categories: My Boundary Responses

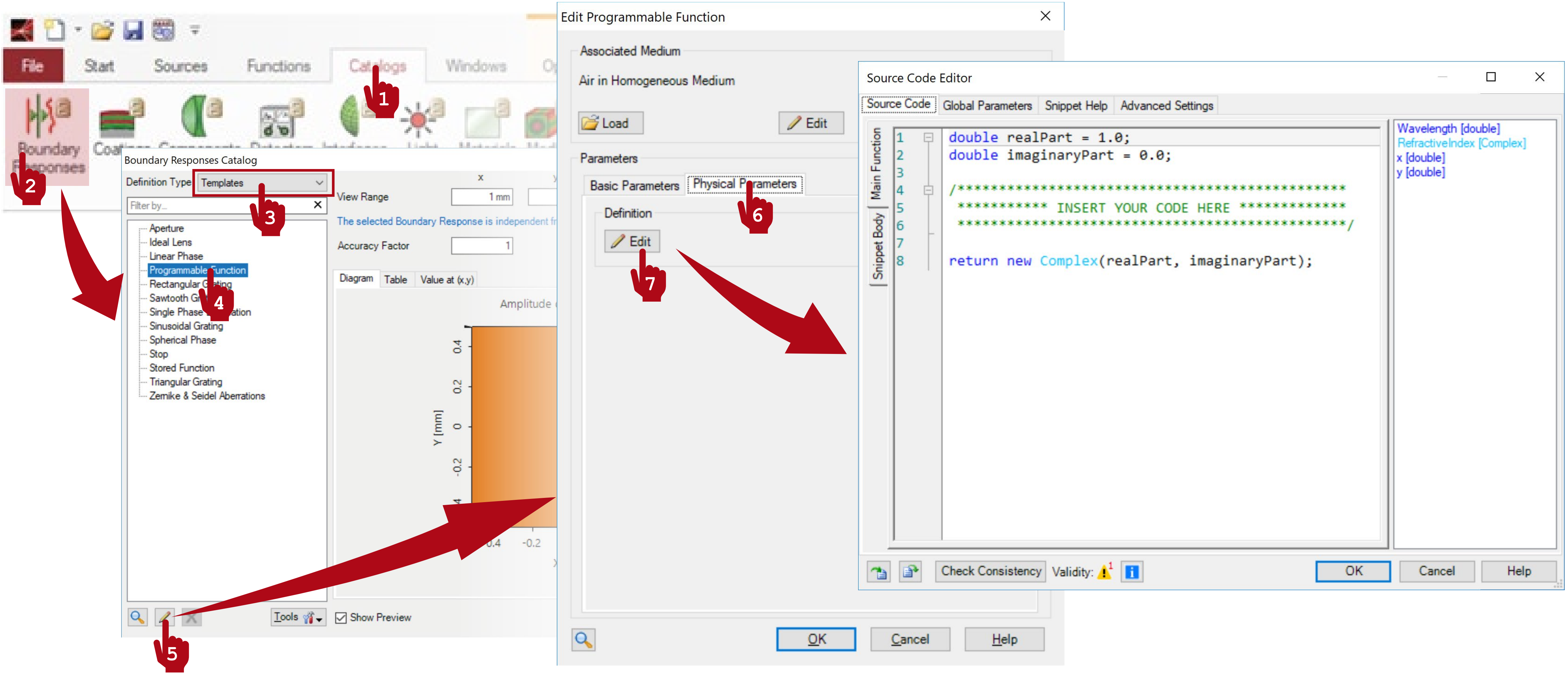
Programming a Cylindrical Lens Function

The Cylindrical Lens

A function that performs as a cylindrical lens is a phase-only modulation of the form:

$$
\begin{aligned}
&\psi^{\text{cyl}}\left(x,y\right)=\operatorname{sign}\left(f\right)k\sqrt{\left(x\cos\alpha+y\sin\alpha\right)^{2}+f^{2}}\\
&f\rightarrow\ \text{Focal length}\\
&k\rightarrow\ \text{Wavenumber}\\
&\alpha\rightarrow\ \text{Angle formed by the optical axis and the focusing direction of the lens}
\end{aligned}
\tag{1}
$$

Where to Find the Programmable Function: Catalog



Definition Type: Templates

View Range: 1 mm

The selected Boundary Response is independent fr

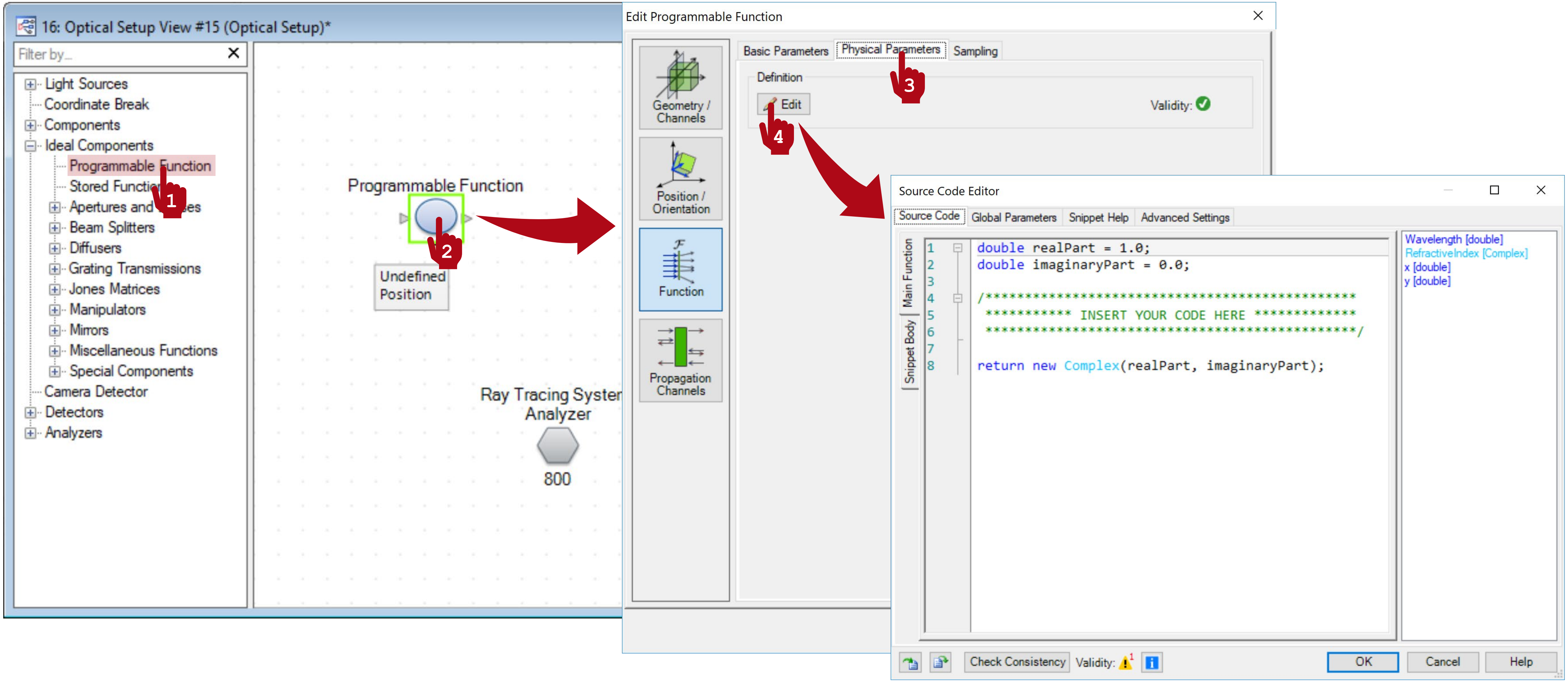
Accuracy Factor: 1

☑ Show Preview

Associated Medium: Air in Homogeneous Medium

```
double realPart = 1.0;
double imaginaryPart = 0.0;

/*********************************************
 *********** INSERT YOUR CODE HERE *************
 *********************************************/

return new Complex(realPart, imaginaryPart);
```

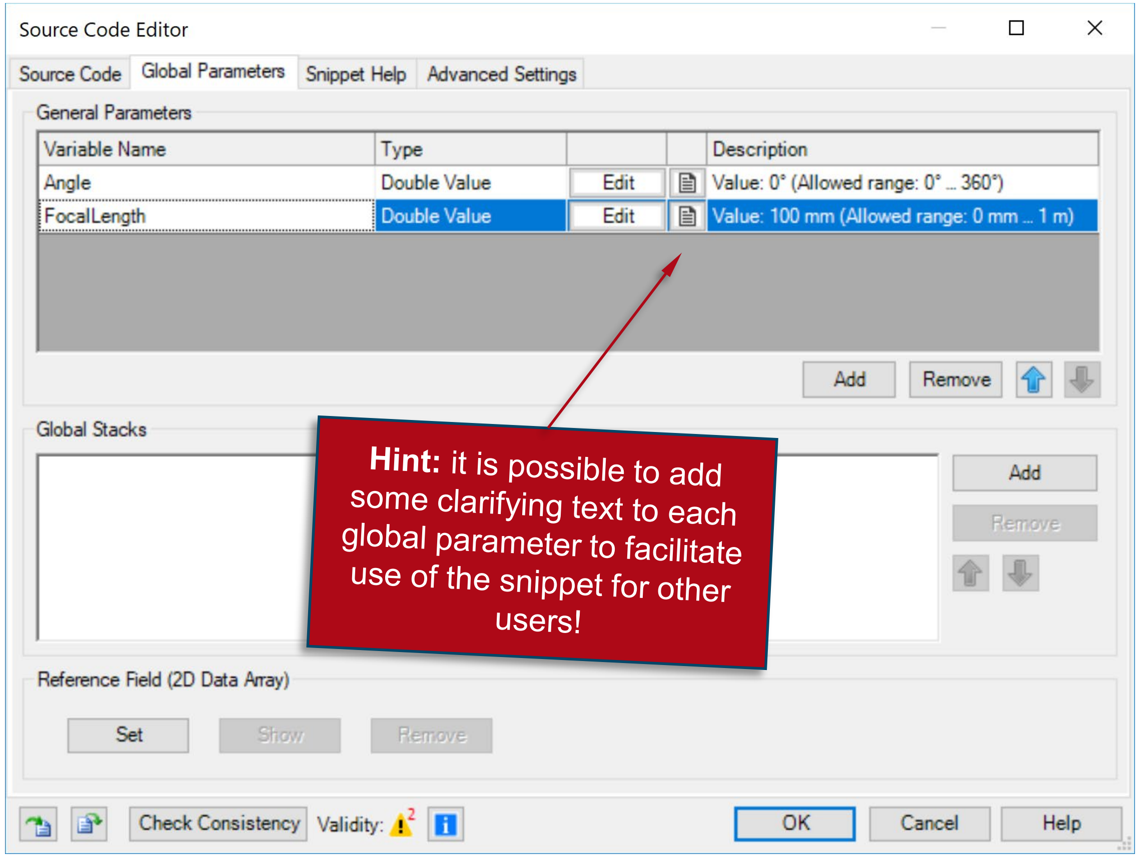
Where to Find the Programmable Function: Optical Setup



```
double realPart = 1.0;
double imaginaryPart = 0.0;

/*********************************************
 *********** INSERT YOUR CODE HERE *************
 *********************************************/

return new Complex(realPart, imaginaryPart);
```

Programmable Interface: Global Parameters

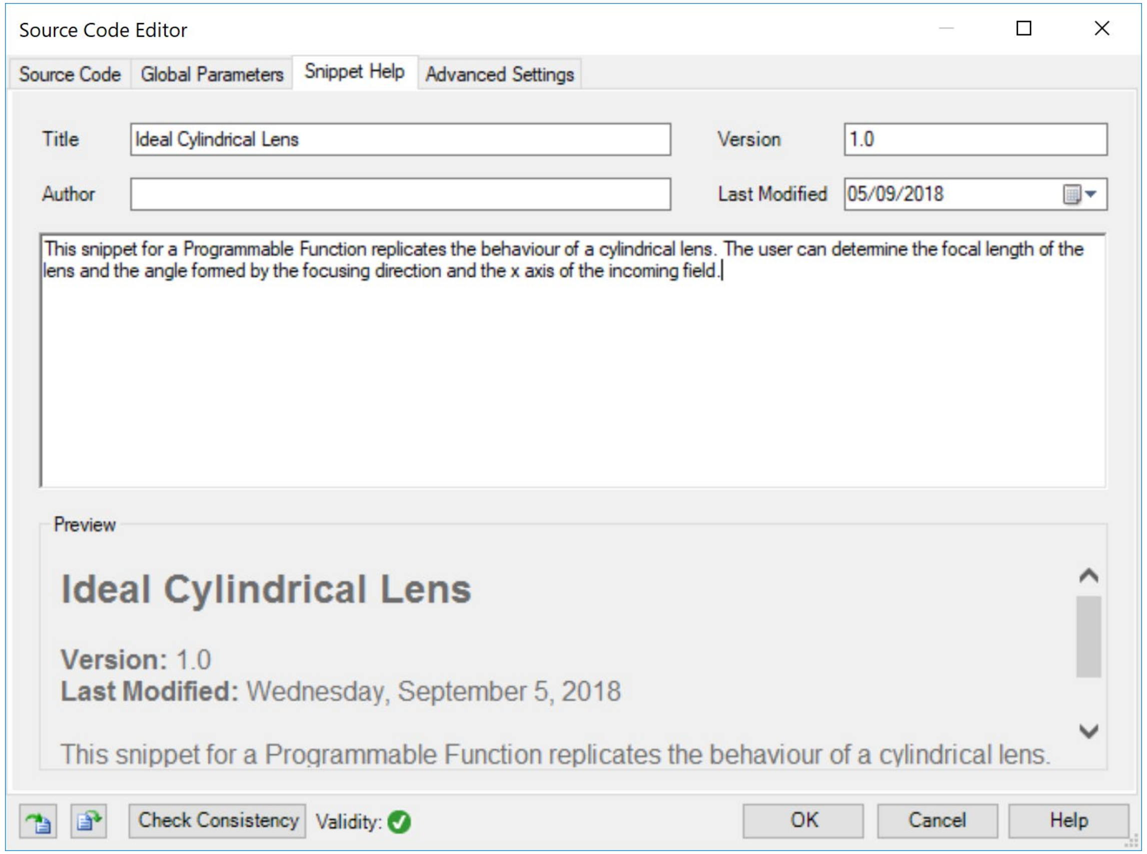
- Once you have triggered open the Edit dialogue, go to the Global Parameters tab.
- There, Add and Edit two global parameters:
  - `double Angle = 0 deg (0 deg, 360 deg)`: represents the angle formed by the optical axis and the focusing direction.
  - `double FocalLength = 100 mm (0 m, 1 m)`: represents the focal length of the lens.
- Use the button with the small "notes" icon to add some explanation to your custom global parameters.



Variable Name	Type			Description
Angle	Double Value	Edit	🗎	Value: 0° (Allowed range: 0° .. 360°)
FocalLength	Double Value	Edit	🗎	Value: 100 mm (Allowed range: 0 mm .. 1 m)

Hint: it is possible to add some clarifying text to each global parameter to facilitate use of the snippet for other users!

Programmable Interface: Snippet Help



Title: Ideal Cylindrical Lens

Version: 1.0

Author:

Last Modified: 05/09/2018

This snippet for a Programmable Function replicates the behaviour of a cylindrical lens. The user can determine the focal length of the lens and the angle formed by the focusing direction and the x axis of the incoming field.

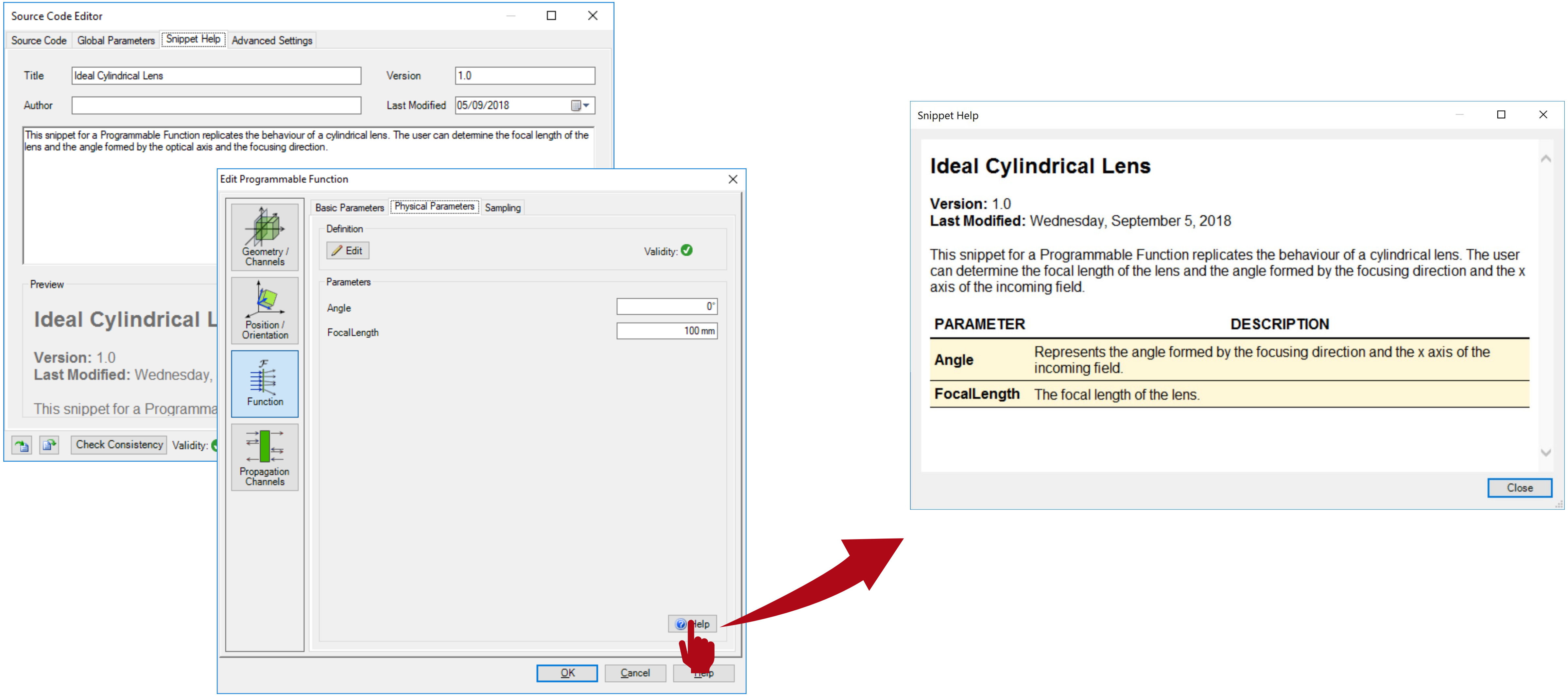
Ideal Cylindrical Lens

Version: 1.0
Last Modified: Wednesday, September 5, 2018

This snippet for a Programmable Function replicates the behaviour of a cylindrical lens.

- **Optional:** you can use the Snippet Help tab to write instructions, clarifications, and some metadata associated to your snippet.
- This option is very helpful to keep track of your progress with a programmable element.
- It is especially useful when the programmable element is later disseminated to be handled by other users!

Programmable Interface: Snippet Help



Title: Ideal Cylindrical Lens

Version: 1.0

Author:

Last Modified: 05/09/2018

This snippet for a Programmable Function replicates the behaviour of a cylindrical lens. The user can determine the focal length of the lens and the angle formed by the optical axis and the focusing direction.

Angle: 0°

FocalLength: 100 mm

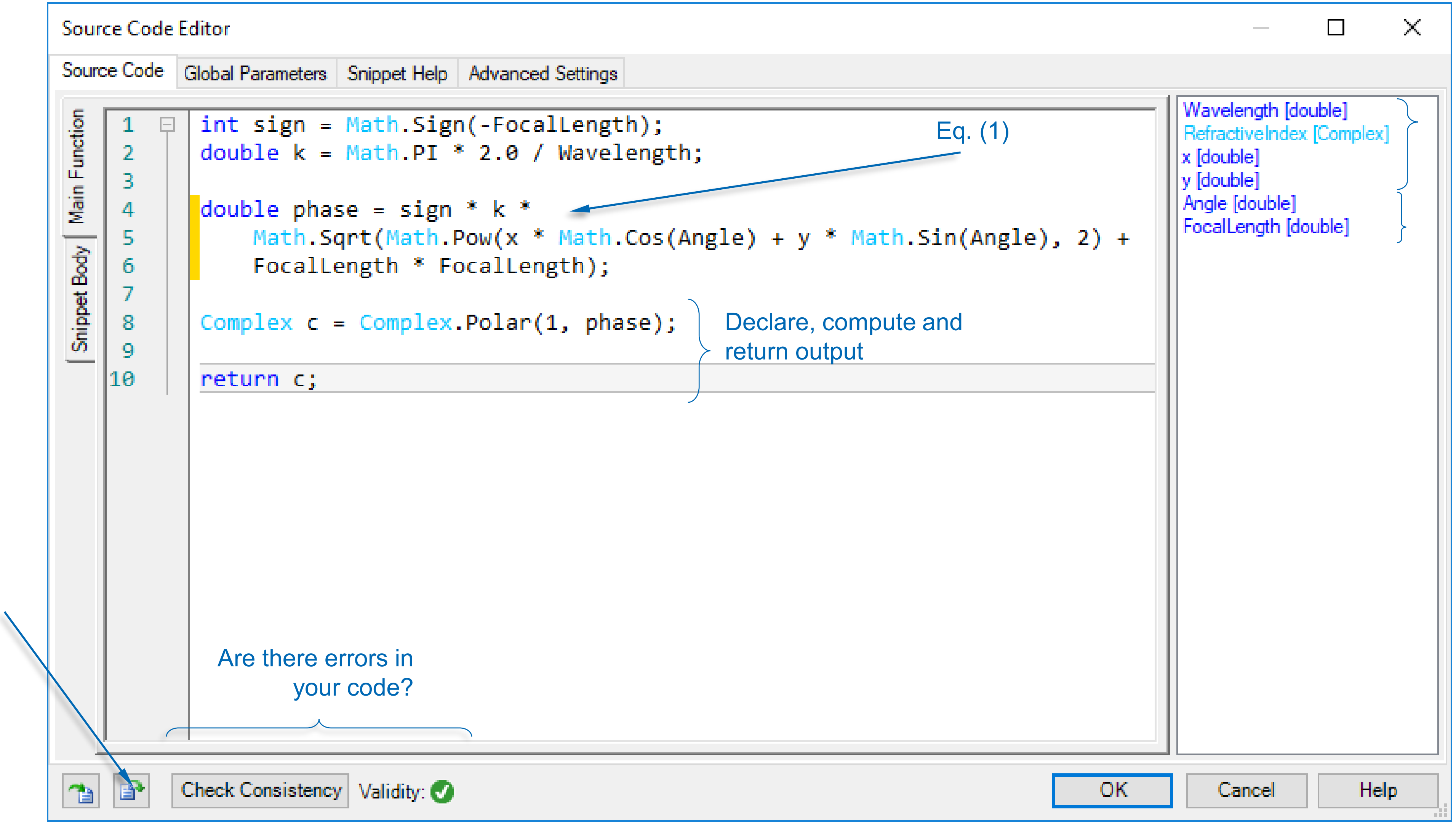
Ideal Cylindrical Lens

Version: 1.0
Last Modified: Wednesday, September 5, 2018

This snippet for a Programmable Function replicates the behaviour of a cylindrical lens. The user can determine the focal length of the lens and the angle formed by the focusing direction and the x axis of the incoming field.

PARAMETER	DESCRIPTION
Angle	Represents the angle formed by the focusing direction and the x axis of the incoming field.
FocalLength	The focal length of the lens.

Programmable Interface: Writing the Code



```
int sign = Math.Sign(-FocalLength);
double k = Math.PI * 2.0 / Wavelength;

double phase = sign * k *
    Math.Sqrt(Math.Pow(x * Math.Cos(Angle) + y * Math.Sin(Angle), 2) +
    FocalLength * FocalLength);

Complex c = Complex.Polar(1, phase);

return c;
```

Wavelength [double]
RefractiveIndex [Complex]
x [double]
y [double]
Angle [double]
FocalLength [double]

Eq. (1)

Declare, compute and return output

Are there errors in your code?

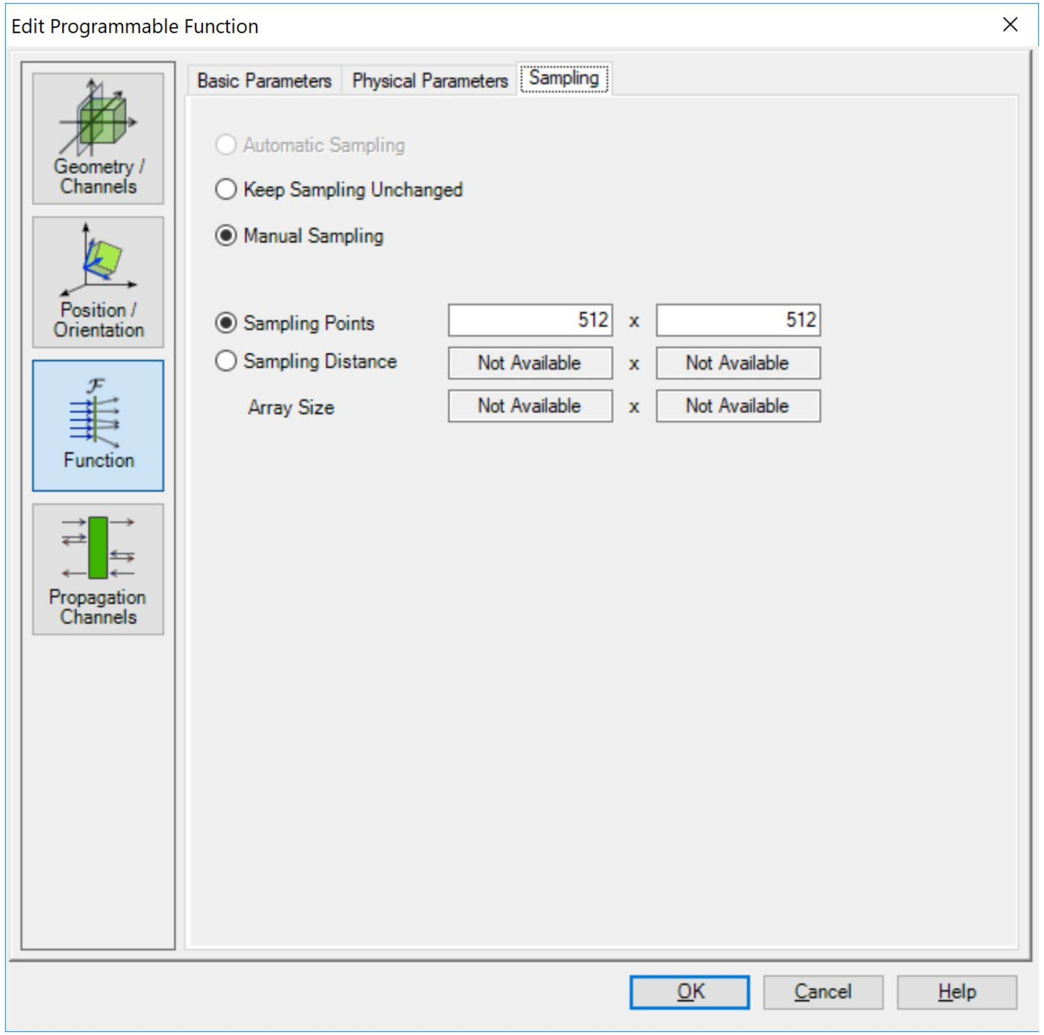
Default global parameters/variables

Global parameter defined by user in Global Parameters tab

Export Snippet to save your work!

Sampling

- Depending on the properties of the incoming field and the custom function, the user must determine the appropriate sampling in the Sampling tab.
- For instance, in the case of our cylindrical lens, and for an on-axis collimated incident beam, the sampling must be finer (higher number of sampling points) for a smaller focal length of the lens.



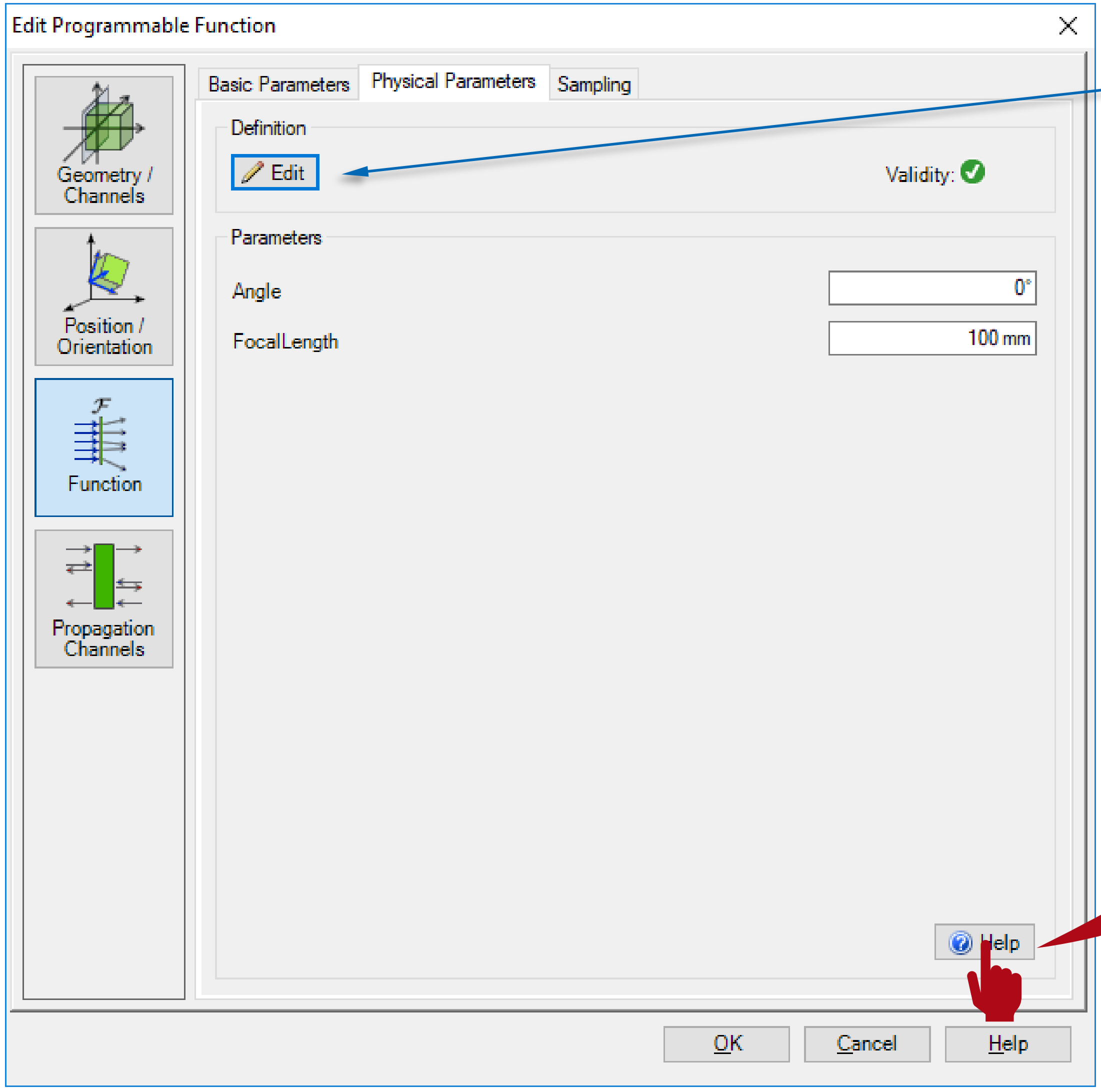
○ Automatic Sampling

○ Keep Sampling Unchanged

● Manual Sampling

● Sampling Points	512	x	512
○ Sampling Distance	Not Available	x	Not Available
Array Size	Not Available	x	Not Available

Programmable Interface: Using Your Snippet

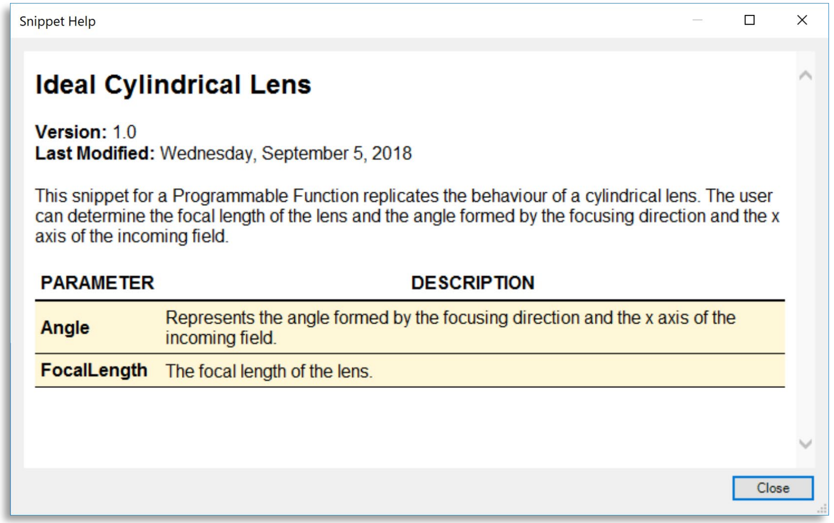


Angle: 0°

FocalLength: 100 mm

Modify your snippet by clicking on Edit

You can modify the value of the global parameters you defined here



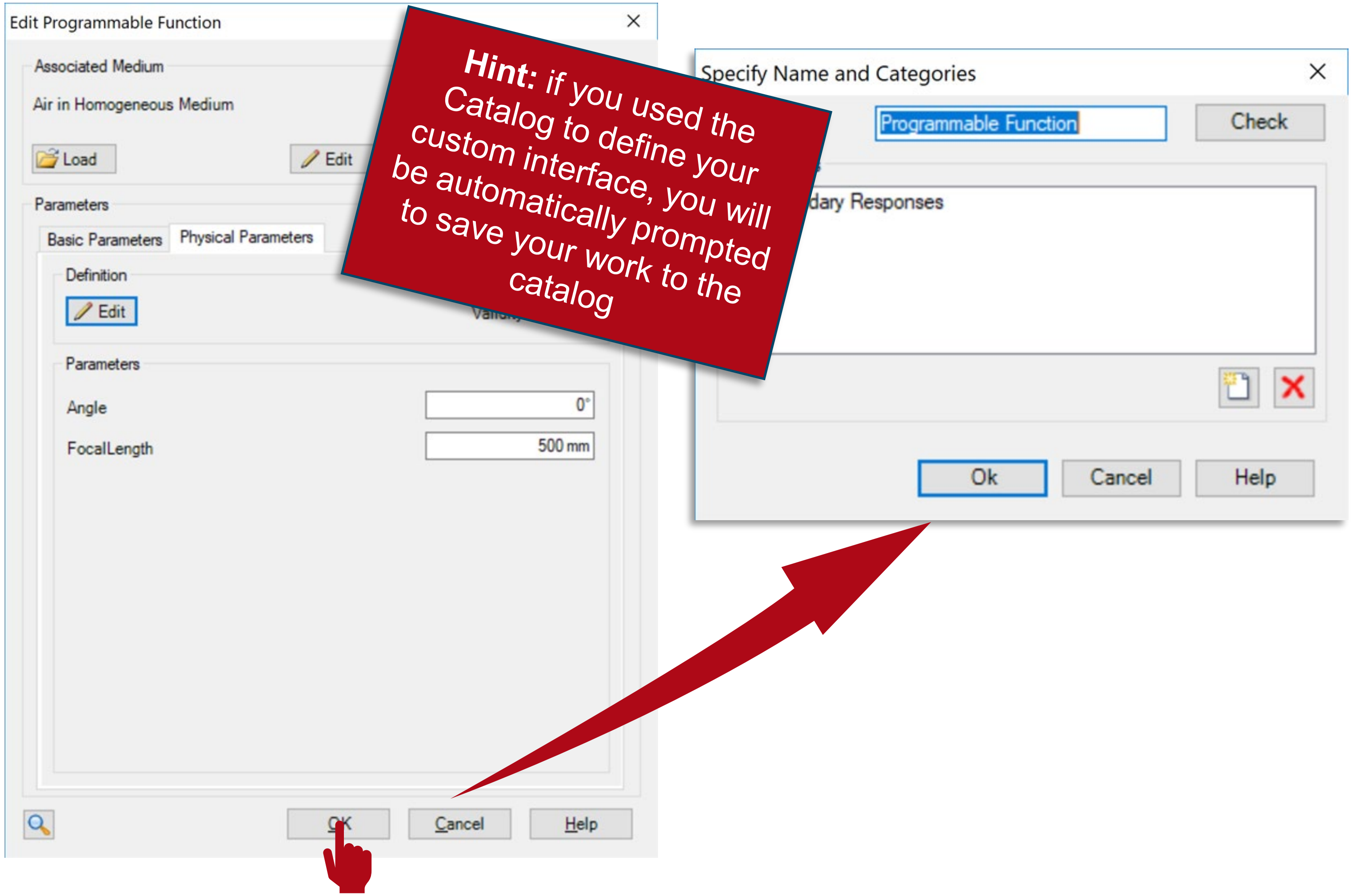
Ideal Cylindrical Lens

Version: 1.0
Last Modified: Wednesday, September 5, 2018

This snippet for a Programmable Function replicates the behaviour of a cylindrical lens. The user can determine the focal length of the lens and the angle formed by the focusing direction and the x axis of the incoming field.

PARAMETER	DESCRIPTION
Angle	Represents the angle formed by the focusing direction and the x axis of the incoming field.
FocalLength	The focal length of the lens.

Saving the Custom Function to the Catalog



Edit Programmable Function

Associated Medium

Air in Homogeneous Medium

Parameters

Definition

Parameters

Angle 0°

FocalLength 500 mm

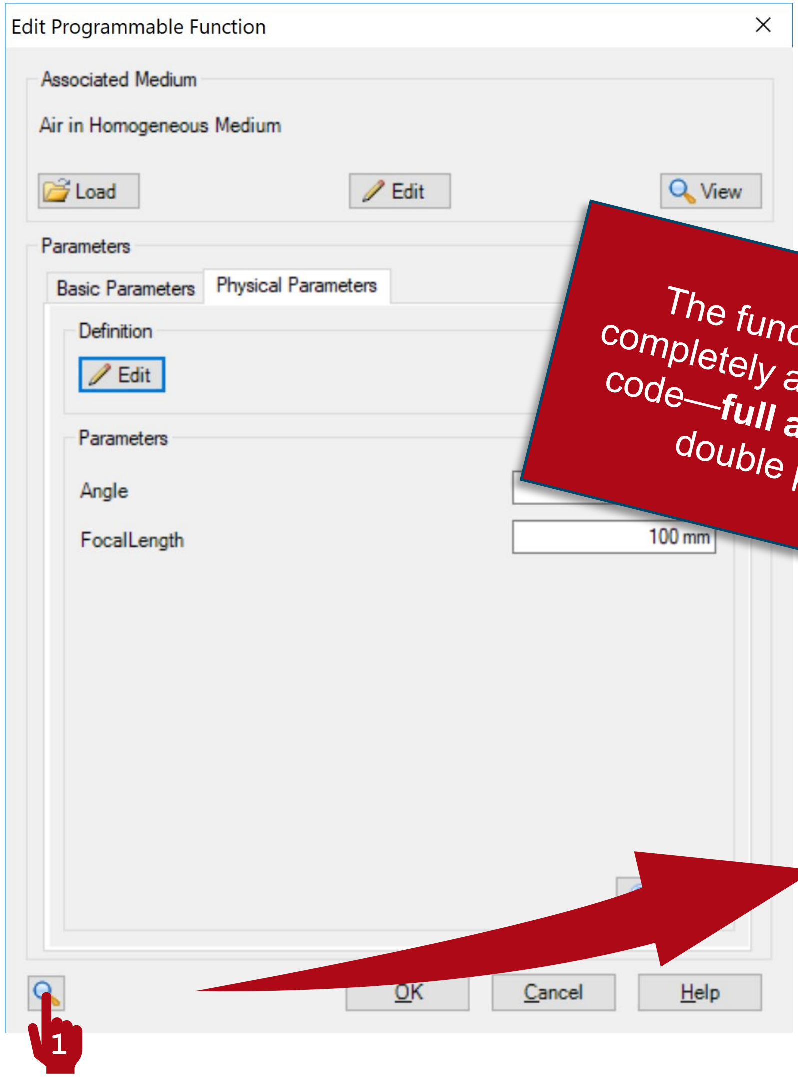
Specify Name and Categories

Programmable Function

dary Responses

Hint: if you used the Catalog to define your custom interface, you will be automatically prompted to save your work to the catalog

Output of Programmable Function



Edit Programmable Function

Associated Medium

Air in Homogeneous Medium

Parameters

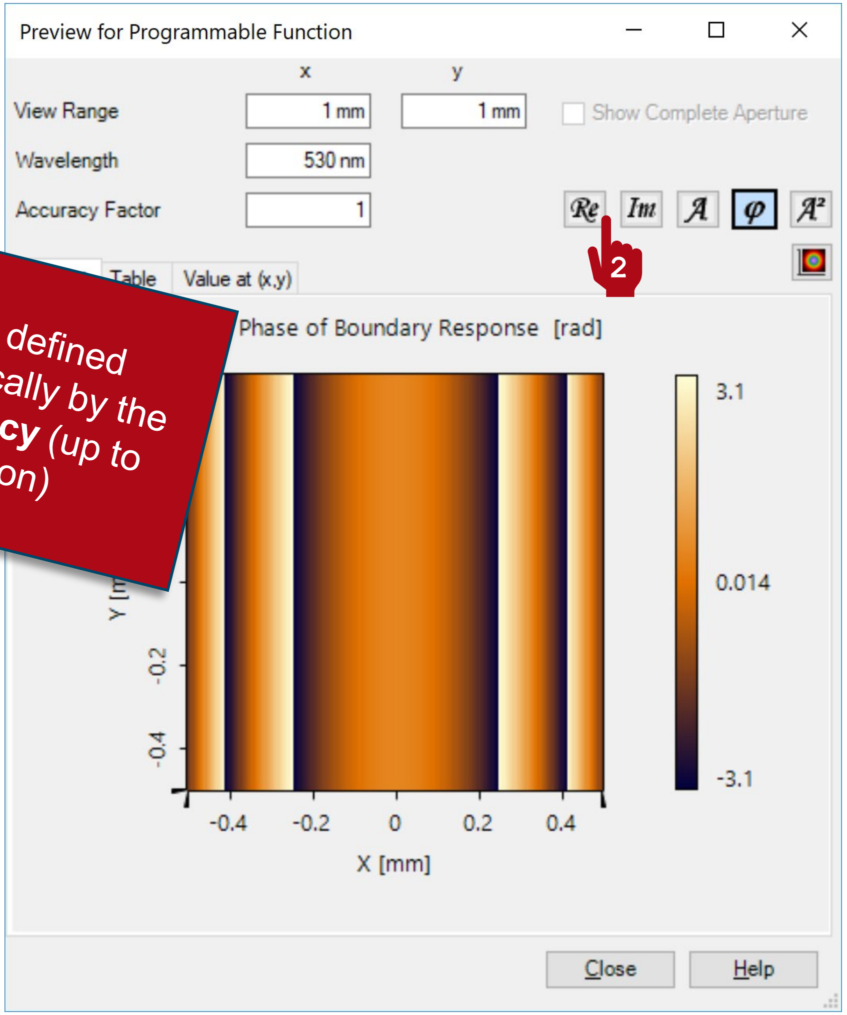
Basic Parameters | Physical Parameters

Definition

Parameters

Angle

FocalLength: 100 mm



Preview for Programmable Function

	x	y
View Range	1 mm	1 mm
Wavelength	530 nm	
Accuracy Factor	1	

☐ Show Complete Aperture

Table | Value at (x,y)

The function is defined completely analytically by the code—**full accuracy** (up to double precision)

Visualization tool only available in Catalog definition mode!

Test the Code!

Main Function

```
int sign = Math.Sign(-FocalLength); // The sign of the focal length
// (convergent or divergent lens).
double k = Math.PI * 2.0 / Wavelength; // The wavenumber.

double phase = sign * k *
    Math.Sqrt(Math.Pow(x * Math.Cos(Angle) + y * Math.Sin(Angle), 2) +
    FocalLength * FocalLength); // Eq. 1

Complex c = Complex.Polar(1, phase); // Generate the complex-valued function,
// with phase-only modulation.

return c;
```

Document Information

title	How to Work with the Programmable Function in VirtualLab Fusion + Example: Cylindrical Lens
document code	CZT.0099
version	1.0
toolbox(es)	Starter Toolbox
VL version used for simulations	7.4.0.49
category	Feature Use Case
further reading	- Customizable Help for Programmable Elements - Programming an Axicon Transmission Function