How to StatSTEM developer

Create new functions

Define new functions in the folder “functions”. Make sure you functions are created using object oriented programming and added to existing classes (in the “@....” folders). Also new classes can be made, make sure that they are a subclass of “StatSTEMfile”.

When creating new functions in existing classes in separate m-files, don’t forget to include them also the main class.

Add functions/panels to graphical interface of StatSTEM

For adding new functions to StatSTEM you should create panels with buttons that execute the function you want to add.

1. Create a new panel in the folder “GUI/LeftPanels/Name_of_the_panel”. In this folder you should create a .m files that holds all the information of the main panel and its buttons. You can use existing .m files for getting an idea how it should look like. The basic structure of the files is as follows:
    a. Define per row buttons. Different buttons are:

        pushbutton – Simple button
        popupmenu – Drop-down menu with different options
        checkbox – Simple checkbox that can be true or false
        radiobutton – Radiobutton that forms with other radiobuttons a group of options
        togglebutton – Button that can be turned on or off
        editfield – Textfield that can be edited by user
        textfield – Non-editable textfield

        Options that each button has are described in the table below. The class-files of the buttons are found in the folder “GUI/ButtonsStatSTEM”, see comments in this class for extra explanation on buttons. Most important is to assign the function name and the input and output this function needs and gives. Be aware that the names of the variables holding the input and output results have predefined names, which can be found in “GUI\generalFunc\listStructNamesClass.m”.
    b. Advanced options in a panel can be made at the end of the script. Here you create for each advanced panel also a text (.m) file in which the buttons are specified. These advanced panel files should be saved in an advanced folder. For example: “GUI\LeftPanels\Fit Model\Advanced”. You can choose to show the advanced option from the start when opening StatSTEM. Check the file: “GUI\LeftPanels\Fit Model\panelFitMain.m” for more details.
2. If you made a new class (subclass of StatSTEMfile), then you should add this to the script: “GUI\generalFunc\listStructNamesClass.m”

Add new images or image options to graphical interface of StatSTEM

When creating new classes or function, one may want to show new images or plot new variables. The images/variables that StatSTEM can show are defined in "GUI/RightPanels/possibleImagesStatSTEM.m". Check this file for the structure and examples. The basic structure is as follows. First define a new image including the function that makes the image and the variable needed, use the function "listImagesStatSTEM('nameImage', 'functionName', 'variableName')". Then add figure options for the variable using the function "addFigOpt(imageOption, "Name of the plotted variable", 'funtionName', 'variableName', showFromStart)".

Use the waitbar

To use the waitbar in your script when running the StatSTEM GUI, use the property value waitbar of the class-variable. Here, you can define the progress that needs to be shown by using:

```
if ~isempty(obj.GUI)
   obj.waitbar.setValue(progress)
end
```

The first part check is the StatSTEM GUI is used. Obj is the class-variable (for example input).

Use the message panel

To display text in the message panel after a function has finished, use the property value message of the class-variable. For example:

```
obj.message = 'some text';
```

Obj is the class-variable (for example input).

List with StatSTEM button options

Button options are given in name-value pairs. The names and descriptions are given below. Check script of existing panels for examples.

Button	Name	Description
general options	width	Width of the button in pixels (per row 162 pixels are available)
	tooltip	Text that appears when mouse is kept on button
	input	Input needed to run a function. Value can be a string (1 input) or a cell holding multiple strings (multiple inputs). The names of the strings should correspond to the variables holding the input (see: "GUI\generalFunc\listStructNamesClass.m"), for example: atomcounting. If a specific class-property is required for a function, gives this. For example, for the pixel size write: input.dx
	optInput	Optional inputs for the function. Works similar as "input" above.
	output	Output generated by the button. . Value can be a string (1 input) or a cell holding multiple strings (multiple inputs). Also give optional output. If a specific class-property is only updated, give the name of this property. For example, input.dx. In this manner, StatSTEM will not asked when pushing the button to delete previous analysis.
	actWhen	Make button active if a specific class-property is true or false. Should be specified as a cell: {true/false,"class-property). For example, {true, "input.fitZeta"}
	fgcolor	Specify the foreground color of a button
	bgcolor	Specify the background color of a button
	margin	Margin of the text in the button

	fitting	Specify whether a function is going to run that takes some time. In this manner, all buttons except the abort buttons will become inactive when pressing the button until the function has finished. To make the abort buttons work in you function, copy this code into your function at different stages:
		
		`if isempty(obj.GUI)`
		`  % For aborting function`
		`  drawnow`
		`  if get(obj.GUI,'Userdata')==0`
		`  error('Error: function stopped by user')`
		`  end`
		`end`
		
		obj is the class variable, for example a inputStatSTEM variable. obj.GUI is a reference to the StatSTEM GUI, the Userdata is set to 1 for fitting, otherwise its 0. If you press the abortbutton, it makes this variable 0 such that your function can be stopped.

	figStart	Image that needs to be shown before running a function. For example, the Observation when adding peak position. Value should be a string with the name.
	figOptStart	Figure options that need to be shown before running a function. For example, input coordinates. Values should be a cell

		containing all the options as string values.
	figEnd	Image that needs to be shown after running a function. Value should be a string with the name.
	figOptEnd	Figure options that need to be shown after running a function. For example, input coordinates. Values should be a cell containing all the options as string values.
	reshowFigEnd	Make sure that figure is always remade at the end. For computional purposes, StatSTEM does not remake the figure given in figEnd is this figure is already shown. Still, a particular image can change during processing and must be changed. If so, make the value of this option true.

Button	Name	Description
pushbutton	name	Name shown in button, string value
	func	Name of function coupled to this button
	enable	Make button active or inactive (internal propery for StatSTEM)

Button	Name	Description
checkbox	name	Name shown in button, string value
	func	Name of function coupled to this button
	enable	Make button active or inactive (internal propery for StatSTEM)
	equalTo	Class-property coupled to this button. When pressing this button, the property values will change to true or false. String value of property name or cell with multiple property names. For example, “input.fitZeta”.
	selected	Is button selected (from the Start). True or false
	selWhen	Specify whether button state (true/false) is coupled to a class-property value. In this manner, this manner the button becomes selected or not when loading a new file. Should be given a cell: {true/false, ‘property-name’}. For example, {true, “input.fitZeta”}.

Button	Name	Description
radiobutton	name	Name shown in button, string value
	func	Name of function coupled to this button
	enable	Make button active or inactive (internal propery for StatSTEM)
	equalTo	Class-property coupled to this button. When pressing this button, the property values will change the value given in optName (see next line). String value of property name or cell with multiple property names. For example, “input.widthtype”.
	optN	Number indicating value that should be given to a specific class-property, specified in “equalTo”
	selected	Is button selected (from the Start). True or false
	selWhen	Specify whether button state (true/false) is coupled to a class-property value. In this manner, this manner the button becomes selected or not when loading a new file. Should be given a cell: {optN, ‘property-name’}. For example, {2, “input.widthtype”}.

Button	Name	Description
togglebutton	name	Name shown in button, string value
	func	Name of function coupled to this button. When pressing this button, a function will be executed and the state of the button will be given to this function as a second input.
	enable	Make button active or inactive (internal propery for StatSTEM)
	equalTo	Class-property to see whether button should be turned off or on when starting StatSTEM. If class-property is empty button is turned off, otherwise button is turned on.
	state	Is button selected (from the Start). True or false

Button	Name	Description
popupmenu	name	Names in the popupmenu. Cell variable containing strings with the names
	func	Name of function coupled to this button. When pressing this button, a function will be executed and the name of the selected option will be given to this function as a second input.
	enable	Make button active or inactive (internal propery for StatSTEM)
	equalTo	Make names in the popupmenu equal to a class-property. If keepName (see below) is turned off, names in the popupmenu are replaced. For example, input.types.
	keepName	If the function equalTo is used one may want the keep the original names given name-value pair “name” (see above). Make this option true to keep the names. For example, in the preparation panel one may always want to add or remove atom type. Therefore, these options are specified in the “name” option, and keepName is turned on to always show this options.
	selOptInput	Class-property to see which option should be when starting StatSTEM. For example, input.actType

Button	Name	Description
editfield	name	Name shown in editfield
	func	Function executed after editing the field and pressing Enter
	enable	Make button active or inactive. An inactive editfield is similar to a textfield, only looks different
	equalTo	Class-property coupled to this button. When changing the value/string in this field, the property values will change to this value. String value of property name. For example, “input.dx”.
	alignment	Textalignment in field (leading = leading, l = left, c = center, r = right)

Button	Name	Description
textfield	name	Name shown in textfield
	alignment	Textalignment in field (leading = leading, l = left, c = center, r = right)
	border	Show border around text (RGB value) or string “none”