Instance

The Instance class inherits all properties and methods of the Element class.

On this page:

Element
Element Class, findChildren, getChild, getParameter, getParameterDefinition, getParameterNormalized, hasParameter, removeFromParent, setName, setParameter, setParameterNormalized

Class Hierarchy

- AudioFile
- Element
  - Bus
  - Effect
  - Instance
  - Layer
    - Program
  - MidiModule
  - ModulationMatrixRow
  - Slot
  - Zone
- Event
- LoadProgress
- ParameterDefinition

Methods

findBusses

findBusses(recursive, nameOrFilterFunction)

Description

Function to find busses in the specified Element object. For example, `this.parent` specifies the parent of the script module as the Element object to be searched in. If recursive is set to `true`, subelements will also be searched. The function returns an array with the Bus objects of the found busses. Particular busses can be searched by name or through a filter function. If searching by name, findBusses accepts only the Bus objects that match the specified name. The filter function uses the Bus object of each bus as argument. Only those Bus objects that return `true` for the search criteria defined in the filter function will be accepted by findBusses. Without a name or filter function the Bus objects of all busses in the searched Element obects will be returned.

Available in: Controller, Processor.

Arguments

recursive	If set to `false`, only the specified Element object will be searched. If set to `true`, subelements will also be searched. The default is `false`.	boolean
nameOrFilterFunction	The name of the busses searched for or a filter function. Only the Bus objects that match the name or return `true` for the search criteria of the filter function will be accepted. Set this to `nil` to deactivate any name filter or search criteria.	string or function, optional

Return Values

Returns an array with the Bus objects of the found busses.

Example

```
-- find all busses and print their names
busses = this.program:findBusses(true)

if busses[1] then
    for i, bus in ipairs(busses) do
        print(bus.name)
    end
else
    print("Could not find any busses!")
end
```

Jump to Top

findEffects

findEffects(recursive, nameOrFilterFunction)

Description

Function to find effects in the specified Element object. For example, `this.parent` specifies the parent of the script module as Element object to be searched in. To specifiy a bus to be searched in, use getBus or findBusses. If recursive is set to `true`, subelements will also be searched. The function returns an array with the Effect objects of the found effects. Particular effects can be searched by name or through a filter function. If searching by name, findEffects accepts only the Effect objects that match the specified name. The filter function uses the Effect object of each effect as argument. Only those Effect objects that return `true` for the search criteria defined in the filter function will be accepted by findEffects. Without a name or filter function the Effect objects of all effects in the searched Element objects will be returned.

Available in: Controller, Processor.

Arguments

recursive	If set to `false`, only the specified Element object will be searched. If set to `true`, subelements will also be searched. The default is `false`.	boolean
nameOrFilterFunction	The name of the effects searched for or a filter function. Only the Effect objects that match the name or return `true` for the search criteria of the filter function will be accepted. Set this to `nil` to deactivate any name filter or search criteria.	string or function, optional

Return Values

Returns an array with the Effect objects of the found effects. Returns an empty table if no effects are found.

Example

```
-- find all effects and print their names
effects = this.program:findEffects(true)

if effects[1] then
    for i, effect in ipairs(effects) do
        print(effect.name)
    end
else
    print("Could not find any effects!")
end
```

Jump to Top

findSlots

findSlots(nameOrFilterFunction)

Description

Function to find the slots of the plug-in instance. Before calling this function you must access the Instance object with `this.program.instance`. The function returns an array with the Slot objects of the found slots. Particular slots can be searched by name or through a filter function. If searching by name, findSlots accepts only the Slot objects that match the specified name. The filter function uses the Slot object of each slot as argument. Only those Slot objects that return `true` for the search criteria defined in the filter function will be accepted by findSlots. Without a name or filter function the Slot objects of all slots in the instance will be returned.

Available in: Controller, Processor.

Arguments

nameOrFilterFunction	The name of the slots searched for or a filter function. Only the Slot objects that match the name or return `true` for the search criteria of the filter function will be accepted. Set this to `nil` to deactivate any name filter or search criteria.	string or function, optional

Return Values

Returns an array with the Slot objects of the found slots. Returns an empty table if no slots are found.

Example

```
-- print the names of all slots
slots = this.program.instance:findSlots()

for i, slot in ipairs(slots) do
        print(slot.name)
end
```

Jump to Top

getBus

getBus(nameOrPosition)

Description

Function to retrieve the Bus object of a bus in the specified Element object. For example, `this.parent` specifies the parent of the script module as the Element object to be searched in. This function does not search in subelements. A particular bus can be searched by name or position. The position is the number indexing the busses in the specified Element object. If several busses share the same name, only the first match will be returned. If no argument is set, the function returns the first bus it finds.

Available in: Controller, Processor.

Arguments

nameOrPosition	The name or position of the bus. Set this to `nil` to deactivate the search filter.	string or number, optional

Return Values

Returns the Bus object of the found bus. Returns `nil` if no bus is found.

Example

```
-- locate the first bus in the program and print its name
bus = this.program:getBus()

if bus then
    print(bus.name)
else
    print("Could not find a bus!")
end
```

Jump to Top

getSlot

getSlot(nameOrIndex)

Description

Function to retrieve the Slot object of a slot of the plug-in instance. Before calling this function you must access the Instance object with `this.program.instance`. A particular slot can be searched by name or index. The index equals the slot numbering in the **Slot Rack**. If no argument is set, the function returns the first slot it finds.

Available in: Controller, Processor

Arguments

nameOrIndex	The name or index of the slot. Set this to `nil` to deactivate the search filter.	string or number, optional

Return Values

Returns the Slot object of the found slot. Returns `nil` if no slot is found.

Example

```
-- print the name of slot index 3
slot = this.program.instance:getSlot(3)
print(slot.name)
```

Jump to Top

getProgram

getProgram(index)

Description

Function to retrieve the Program object of a program in the **Program Table** of the plug-in instance. Before calling this function you must access the Instance object with `this.program.instance`. The `index` corresponds to the number of the slot in the **Program Table** where the program is set. The function returns the Program object of the program with the specified index.

Available in: Controller.

Arguments

index	The index of the slot in the **Program Table** where the program is set.	number

Return Values

Returns the Program object of the program with the specified index.

Example

```
-- print the name of the program in the third slot of the bank
program = this.program.instance:getProgram(3)
print(program.name)
```

Jump to Top

setProgram

setProgram(programOrNil, index)

Description

Function to set a program in the specified slot of the **Program Table** or the **Slot Rack** of the plug-in instance. Before calling this function, you must access the Instance object with `this.program.instance`. The program is determined by its Program object. To specify the slot in the **Program Table**, you must use the `index` argument. To specify the slot in the **Slot Rack**, you must use a Slot object, for example, via getSlot. The program can be removed from the **Slot Rack** by using `nil` as argument.

An Element object can only have one parent. It cannot be child of multiple parents. Therefore, each program can exist only once in the **Program Table**. Furthermore, an Element object that you retrieved from the running plug-in instance cannot be added twice to the **Program Table**. It must be removed before it can be added again. The Element objects that you retrieve through loadPreset or loadPresetAsync can be added freely to the **Program Table**, because these functions create a copy of the Element objects when reading them.

Available in: Controller.

Arguments

programOrNil	The Program object of the program. Programs can be removed from the **Slot Rack** by using `nil`.	Program or `nil`
index	The index of the slot in the **Program Table** where you want to set the program.	number, optional

Example

To explore the following script:

1. Download Program.vstpreset.
2. Drag the preset on the **MediaBay** to import it to the user folder for VST presets.
3. Create an empty program and add a script module.
4. Paste the script into the text editor of the script module and execute the script.

```
-- set Program.vstpreset in slot 3 of the Program Table and slot 1 of the Slot Rack

-- get the file path for user VST presets
path = getUserPresetPath()

-- load the VST preset
loadedProgram = loadPreset(path.."/Program/Program.vstpreset")

-- set loadedProgram in slot 3 of the Program Table
this.program.instance:setProgram(loadedProgram, 3)

-- set program in slot 1 of the Slot Rack
program = this.program.instance:getProgram(3)
this.program.instance:getSlot(1):setProgram(program)

-- clear slot 2 of the Slot Rack
this.program.instance:getSlot(2):setProgram(nil)
```

Jump to Top