widget$_{p}eriodictable$Documentation

Release 1.4.0

Giovanni Pizzi and Dou Du

May 26, 2020

INSTALLATION AND USAGE

Version: 1.4.0

A jupyter widget to select chemical elements from the periodic table.

CHAPTER

ONE

QUICKSTART

To get started with widget_periodictable, install with pip:

```
pip install widget_periodictable
```

or with conda:

```
conda install widget_periodictable
```

CHAPTER

TWO

CONTENTS

2.1 Installation

The simplest way to install widget_periodictable is via pip:

```
pip install widget_periodictable
```

or via conda:

```
conda install widget_periodictable
```

If you installed via pip, and notebook version < 5.3, you will also have to install / configure the front-end extension as well. If you are using classic notebook (as opposed to Jupyterlab), run:

```
jupyter nbextension install [--sys-prefix / --user / --system] --py widget_
↪periodictable

jupyter nbextension enable [--sys-prefix / --user / --system] --py widget_
↪periodictable
```

with the appropriate flag. If you are using Jupyterlab, install the extension with:

```
jupyter labextension install widget-periodictable
```

If you are installing using conda, these commands should be unnecessary, but If you need to run them the commands should be the same (just make sure you choose the *–sys-prefix* flag).

2.2 Introduction

This is a Jupyter extension to create a interactive periodic table widget inside the notebooks. Users can select elements by clicking on the elements inside the periodic table. The selected elements can be divided into different states. Besides, elements can be disabled by calling the Python variable **disabled_elements** from the widget class.

2.2.1 Traitlets

- **selected_element** : The dictionary of the selected elements with their states.
- **states** : The number of states for the periodic table.
- **selected_colors** : The colors of the selected elements.
- **selected_disabled** : The list of the disabled elements.

2.2.2 Acknowledges

This work has been done with the support of the EPFL Open Science Fund **OSSCAR**.

2.3 Examples

This section contains several examples generated from Jupyter notebooks. The widgets have been embedded into the page for demonstrative pruposes.

2.3.1 The widget periodic table

```
In [1]: import ipywidgets as widgets
        from widget_periodictable import PTableWidget
```

Visualize the element grid

```
In [2]: # Show the widget
        widget = PTableWidget(states = 5, selected_elements = {"Be":0}, selected_colors = ['red', 'gr
        widget
```

```
PTableWidget(allElements=['H', 'He', 'Li', 'Be', 'B', 'C', 'N', 'O', 'F', 'Ne', 'Na', 'Mg', 'Al', 'Si
```

Set the states of the elements

The periodic table allows users to customize the states of the selected elements. If one do not give the selected element's state, it will set the state as zero.

Init the selected elements by using a dictionary:

```
widget.selected_elements = {"La": 0, "Ce": 1, "Pr": 2}
```

```
In [3]: widget.selected_elements = {"La": 0, "Ce": 1, "Pr": 2}
```

Change or set element state by:

```
widget.set_element_state("Nd",0)
```

```
In [4]: widget.set_element_state("Nd",0)
```

However, you cannot use widget.selected_elements["Nd"] = 1 to set the states of the elements.

Get the elements have the same state:

```
widget.get_elements_by_state(0)
```

```
In [5]: widget.get_elements_by_state(0)
```

```
Out[5]: ['La', 'Nd']
```

Get the selected values in python

Check which elements are currently selected

```
In [6]: output = widgets.Output()

        def on_get_in_python(event):
            output.clear_output()
            with output:
                print(
                    "Currently selected values:",
                    widget.selected_elements)

        button2 = widgets.Button(
            description="Get the currently selected values",
            button_style='success',
            layout={'width': '300px'}
        )
        button2.on_click(on_get_in_python)
        vbox = widgets.VBox([button2, output])
        vbox
```

```
VBox(children=(Button(button_style='success', description='Get the currently selected values', layout
```

Play with enabling/disabling some elements

```
In [7]: toggle_disabled = widgets.Checkbox(
            value="O" in widget.disabled_elements,
            description='Disable oxygen',
            disabled=False
        )

        def on_change_disabled(event):
            if toggle_disabled.value:
                # It's set, meaning we want to disable oxygen
                widget.disabled_elements = ["O"]
            else:
                widget.disabled_elements = []
        toggle_disabled.observe(on_change_disabled, names='value')

        def on_change(event):
            """
            Update the toggle value if manually changing the disabled_elements list.
            """
            toggle_disabled.value = "O" in widget.disabled_elements
        widget.observe(on_change, names='disabled_elements', type='change')

        toggle_disabled
```

```
Checkbox(value=False, description='Disable oxygen')
```

Set the selected values from python

Choose the selected values from python

```
In [8]: def on_set_from_python(event):
            # NOTE! If you put an element which does not exist, it will stay forever in the list, bu
            widget.selected_elements = {"Li":0, "H":0}

        button = widgets.Button(
            description="Select only Li and H (from python)",
            button_style='success',
            layout={'width': '300px'}
        )
        button.on_click(on_set_from_python)
        button
```

```
Button(button_style='success', description='Select only Li and H (from python)', layout=Layout(width=
```

Change the displayed string for some elements

Note that you should pass valid HTML strings, as they will not be escaped. On the other hand this allows to use HTML to change the class, color, ...

```
In [9]: def get_noble_gases_state():
            label_deactivate = "Make noble gases bold"
            label_activate = "Make noble gases not bold"
            def deactivate_noble_gases(event):
                widget.display_names_replacements = {}
            def activate_noble_gases(event):
                widget.display_names_replacements = {
                    elem_name: "<b>{}</b>".format(elem_name)
                    for elem_name in ['He', 'Ne', 'Ar', 'Kr', 'Xe', 'Rn', 'Og']
                }

            if 'He' in widget.display_names_replacements:
                return {
                    'is_active': True,
                    'toggler_function': deactivate_noble_gases,
                    'toggled_label': label_deactivate,
                    'current_label': label_activate
                }
            else:
                return {
                    'is_active': True,
                    'toggler_function': activate_noble_gases,
                    'toggled_label': label_activate,
                    'current_label': label_deactivate
                }

        button_noble = widgets.Button(
            description=get_noble_gases_state()['current_label'],
            button_style='success',
            layout={'width': '300px'}
        )

        def on_toggle_noble_gases(event):
```

```
            """Toggle the state of the button and of the ."""
            state = get_noble_gases_state()
            # Change the table
            state['toggler_function'](event)
            # Change the button description
            button_noble.description = state['toggled_label']

        button_noble.on_click(on_toggle_noble_gases)
        button_noble
```

```
Button(button_style='success', description='Make noble gases bold', layout=Layout(width='300px'), sty
```

This work has been done with the support of the EPFL Open Science Fund OSSCAR.

2.4 Developer install

To install a developer version of widget_periodictable, you will first need to clone the repository:

```
git clone https://github.com/osscar-org/widget-periodictable
cd widget-periodictable
```

Next, install it with a develop install using pip:

```
pip install -e .
```

If you are planning on working on the JS/frontend code, you should also do a link installation of the extension:

```
jupyter nbextension install [--sys-prefix / --user / --system] --symlink --py widget_
↪periodictable

jupyter nbextension enable [--sys-prefix / --user / --system] --py widget_
↪periodictable
```

with the appropriate flag. Or, if you are using Jupyterlab:

```
jupyter labextension install .
```