traits-enaml Documentation

Release 0.2.0

Enthought

March 20, 2014

Traits-Enaml is an extension library to facilitate interoperation of *Enaml 0.8.x* with *Traits* and allow a programmer to drive enaml views using traits models, enable/chaco components and mayavi 3D scenes.

CHAPTER 1

Usage

To use traits model classes to drive enaml components it is enough to use the *with traits_enaml.imports():* context manager.

Example python code:

```
#------------------------------------------------------------------------------
#
#  Copyright (c) 2013-14, Enthought, Inc.
#  All rights reserved.
#
#  This software is provided without warranty under the terms of the BSD
#  license included in /LICENSE.txt and may be redistributed only
#  under the conditions described in the aforementioned license.  The license
#  is also available online at http://www.enthought.com/licenses/BSD.txt
#
#  Thanks for using Enthought open source!
#
#------------------------------------------------------------------------------
import enaml
from enaml.qt.qt_application import QtApplication

from traits.api import HasTraits, Str, Range
from traitsui.api import View



class Person(HasTraits):
    """ A simple class representing a person object.

    """
    last_name = Str()

    first_name = Str()

    age = Range(low=0)

    traits_view = View('last_name', 'first_name', 'age', resizable=True)

if __name__ == '__main__':
    with enaml.imports():
        from traits_view import PersonView

    john = Person(first_name='John', last_name='Doe', age=42)
```

```
    app = QtApplication()
    view = PersonView(person=john)
    view.show()

    app.start()
```

And related Enaml definition:

```
#------------------------------------------------------------------------------
#
#  Copyright (c) 2013-14, Enthought, Inc.
#  All rights reserved.
#
#  This software is provided without warranty under the terms of the BSD
#  license included in /LICENSE.txt and may be redistributed only
#  under the conditions described in the aforementioned license.  The license
#  is also available online at http://www.enthought.com/licenses/BSD.txt
#
#  Thanks for using Enthought open source!
#
#------------------------------------------------------------------------------
from enaml.widgets.api import Window, Container
from traits_enaml.widgets.traits_view import TraitsView

enamldef PersonView(Window):
    attr person

    Container:
        TraitsView:
            model := person
```

CHAPTER 2

Widgets

To further simplify the inter-operation with traits-related libraries additional widgets and factories are provided. These components are available in the *widgets* module.

2.1 AutoView

Using the **auto_window** or the **auto_view** factories we can create Enaml components for *HasTraits* classes in an automatic way similar to the default TraitsUI views.

Example python code using the *auto_window* factory:

```
#------------------------------------------------------------------------------
#
#  Copyright (c) 2013-14, Enthought, Inc.
#  All rights reserved.
#
#  This software is provided without warranty under the terms of the BSD
#  license included in /LICENSE.txt and may be redistributed only
#  under the conditions described in the aforementioned license.  The license
#  is also available online at http://www.enthought.com/licenses/BSD.txt
#
#  Thanks for using Enthought open source!
#
#------------------------------------------------------------------------------
import datetime

from enaml.qt.qt_application import QtApplication
from traits.api import (HasTraits, Bool, Button, Date, Enum, Float, Int, List,
                        Range, Str, Time)
import traits_enaml
from traits_enaml.widgets.auto_view import auto_window
with traits_enaml.imports():
    from traits_enaml.widgets.auto_editors import DefaultEditor


class AllTypes(HasTraits):
    """ A simple class with all kinds of traits

    """
    boolean_value = Bool(True, label="Custom Bool Label:")
    button_value = Button("I'm a button!")
    int_value = Int(42, tooltip="You can add a tooltip as well.")
```

```
    float_value = Float(3.141592)
    enum_value = Enum("foo", "bar", "baz", "qux")
    int_range_value = Range(low=0, high=10)
    float_range_value = Range(low=0.0, high=1.0)
    list_value = List([0, 1, 2])
    str_value = Str("Word")
    date_value = Date(datetime.date.today())
    time_value = Time(datetime.time())
    range_value = Range(low=0, high=100,
                        label="Traits Range Editor:",
                        enaml_editor=DefaultEditor)

    _my_float = Float

    def _button_value_fired(self):
        print "Button was pressed"

    def _anytrait_changed(self, name, old, new):
        print name, "changed from", old, "to", new

if __name__ == '__main__':
    all = AllTypes()
    app = QtApplication()
    view = auto_window(all)
    view.show()

    app.start()
```

2.2 TraitsView

Complete TraitsUI Views can be embedded inside an enaml component using the `TraitsView` widget.

2.3 EnableCanvas

Enable and Chaco components can be embedded inside an enaml component using the `EnableCanvas` widget.

2.4 MayaviCanvas

To embed a Mayavi 3D scene inside an Enaml component, one should use the `MayaviCanvas` widget.