Example plugin

Release 0.1.3.dev7+gcedaca9

Pierre Clisson

Apr 06, 2022

CONTENTS

This is an example plugin that provides a few simple demonstration nodes. Use it as a template to develop your own plugins.

CHAPTER

ONE

INSTALLATION

First, make sure that Timeflux is installed.

You can then install this plugin in the *timeflux* environment:

```
$ conda activate timeflux
$ pip install timeflux_example
```

1.1 API Reference

This page contains auto-generated API reference documentation.

timeflux_example

1.1.1 timeflux_example

timeflux_example.nodes

nodes

timeflux_example.nodes.arithmetic

Simple example nodes

arithmetic

class timeflux_example.nodes.arithmetic.**Add**(*value*)

Bases: timeflux.core.node.Node

Adds `value` to each cell of the input.

This is one of the simplest possible nodes.

Variables

- **i** (*Port*) – Default input, expects DataFrame.
- **o** (*Port*) – Default output, provides DataFrame.

Example

```yaml
graphs:

  - nodes:
    - id: node_1
      module: timeflux.nodes.random
      class: Random
      params:
        columns: 5
        rows_min: 1
        rows_max: 10
        value_min: 0
        value_max: 5
        seed: 1
    - id: node_2
      module: timeflux_example.nodes.arithmetic
      class: Add
      params:
        value: 1
    - id: node_3
      module: timeflux.nodes.debug
      class: Display

    edges:
    - source: node_1
      target: node_2
    - source: node_2
      target: node_3
```

Parameters **value** (*int*) – The value to add to each cell.

update(*self*)

Update the input and output ports.

class timeflux_example.nodes.arithmetic.**MatrixAdd**

Bases: timeflux.core.node.Node

Sum two input matrices together.

This node illustrates multiple named inputs. Note that it is not necessary to declare the ports. They will be created dynamically.

Variables

- **i_m1** (*Port*) – First matrix, expects DataFrame.
- **i_m2** (*Port*) – Second matrix, expects DataFrame.
- **o** (*Port*) – Default output, provides DataFrame.

Example

```yaml
graphs:

  - id: multi
    nodes:
    - id: matrix_1
      module: timeflux.nodes.random
      class: Random
      params:
        columns: 2
        rows_min: 2
        rows_max: 2
        value_min: 1
        value_max: 1
        seed: 1
    - id: matrix_2
      module: timeflux.nodes.random
      class: Random
      params:
        columns: 2
        rows_min: 2
        rows_max: 2
        value_min: 2
        value_max: 2
        seed: 1
    - id: matrix_add
      module: timeflux_example.nodes.arithmetic
      class: MatrixAdd
    - id: display_matrix_1
      module: timeflux.nodes.debug
      class: Display
    - id: display_matrix_2
      module: timeflux.nodes.debug
      class: Display
    - id: display_matrix_add
      module: timeflux.nodes.debug
      class: Display

    edges:
    - source: matrix_1
      target: matrix_add:m1
    - source: matrix_2
      target: matrix_add:m2
    - source: matrix_1
      target: display_matrix_1
    - source: matrix_2
      target: display_matrix_2
    - source: matrix_add
      target: display_matrix_add

    rate: 0.1
```

Instantiate the node.

update(*self*)

Update the input and output ports.

class timeflux_example.nodes.arithmetic.**MatrixDivide**

Bases: timeflux.core.node.Node

Divide one matrix by another.

Variables

- **i_m1** (*Port*) – First matrix, expects DataFrame.
- **i_m2** (*Port*) – Second matrix, expects DataFrame.
- **o** (*Port*) – Default output, provides DataFrame.

Instantiate the node.

update(*self*)

Update the input and output ports.

timeflux_example.nodes.dynamic

Illustrates dynamic inputs and outputs.

dynamic

class timeflux_example.nodes.dynamic.**Outputs**(*prefix=None*, *seed=None*)

Bases: timeflux.core.node.Node

Randomly generate dynamic outputs.

At each update, this node generates a random number of outputs and sets the default output to the number it has created.

Variables

- **o** (*Port*) – Default output, provides DataFrame.
- **o_*** (*Port*) – Dynamic outputs.

Parameters

- **seed** (*int*) – The random number generator seed.
- **prefix** (*string*) – The prefix to add to each dynamic output.

Example

```
graphs:

  - id: DynamicIO

    nodes:
    - id: node_1
      module: timeflux_example.nodes.dynamic
      class: Outputs
```

(continues on next page)

(continued from previous page)

```
      params:
        seed: 1
        prefix: foo
    - id: node_2
      module: timeflux_example.nodes.dynamic
      class: Inputs
      params:
        prefix: bar
    - id: node_3
      module: timeflux.nodes.debug
      class: Display
    - id: node_4
      module: timeflux.nodes.debug
      class: Display

    edges:
    - source: node_1:foo_* # Dynamic inputs can be prefixed
      target: node_2:bar   # The same goes for outputs
    - source: node_1
      target: node_3
    - source: node_2
      target: node_4
```

Instantiate the node.

update(*self*)

Update the input and output ports.

class timeflux_example.nodes.dynamic.**Inputs**(*prefix=None*)

Bases: timeflux.core.node.Node

Count the dynamic outputs.

At each update, this node loops over all dynamic inputs and sets the default output to the number it has found.

Variables

- **i_*** (*Port*) – Dynamic inputs.
- **o** (*Port*) – Default output, provides DataFrame.

Parameters **prefix** (*string*) – The prefix to add to match dynamic inputs.

Example

```
graphs:

  - id: DynamicIO

    nodes:
    - id: node_1
      module: timeflux_example.nodes.dynamic
      class: Outputs
      params:
```

(continues on next page)

(continued from previous page)

```
      seed: 1
  - id: node_2
    module: timeflux_example.nodes.dynamic
    class: Inputs
  - id: node_3
    module: timeflux.nodes.debug
    class: Display
  - id: node_4
    module: timeflux.nodes.debug
    class: Display

  edges:
  - source: node_1:* # The magic happens here
    target: node_2
  - source: node_1
    target: node_3
  - source: node_2
    target: node_4
```

Instantiate the node.

update(*self*)

Update the input and output ports.

timeflux_example.nodes.signal

---

timeflux_example.nodes.signal: generate signals

signal

class timeflux_example.nodes.signal.**Sine**(*frequency=1*, *resolution=200*, *amplitude=1*, *name='sine'*)

Bases: timeflux.core.node.Node

Generate a sinusoidal signal.

Variables **o** (*Port*) – Default output, provides DataFrame.

Parameters

- **frequency** (*float*) – cycles per second. Default: 1.
- **resolution** (*int*) – points per second. Default: 200.
- **amplitude** (*float*) – signal amplitude. Default: 1.
- **name** (*string*) – signal name. Default: sine.

Example

```
graphs:
  - nodes:
      - id: sine
        module: timeflux_example.nodes.signal
        class: Sine
        params:
          frequency: 120
          amplitude: 1
          resolution: 44100
      - id: ui
        module: timeflux_ui.nodes.ui
        class: UI
        params:
          settings:
            monitor:
              millisPerPixel: 0.25
              lineWidth: 1
              interpolation: linear
    edges:
      - source: sine
        target: ui:sine
    rate: 10
```

Instantiate the node.

update(*self*)

Update the input and output ports.

timeflux_example.nodes.sinus

---

timeflux_example.nodes.sinus: generate sinusoidal signal

sinus

class timeflux_example.nodes.sinus.**Sinus**(*amplitude=1*, *rate=1*, *name='sinus'*)

Bases: timeflux.core.node.Node

Return a sinusoidal signal sampled to registry rate.

This node generates a sinusoidal signal of chosen frequency and amplitude. Note that at each update, the node generate one row, so its sampling rate equals the graph parsing rate (given by the Registry).

Variables **o** (*Port*) – Default output, provides DataFrame.

Example

```yaml
graphs:

  - nodes:
      - id: sinus
        module: timeflux_example.nodes.sinus
        class: Sinus
        params:
          rate: 1
          amplitude: 1

      - id: ui
        module: timeflux_ui.nodes.ui
        class: UI

    edges:
      - source: sinus
        target: ui:sinus

    rate: 100
```

Deprecated since version Use: *timeflux_example.nodes.signal.Sine()* instead.

Instantiate the node.

update(*self*)

Update the input and output ports.

PYTHON MODULE INDEX

t

INDEX