clojure-babel-tests.org

Greg

2014-01-30 Thur

Contents

1 Introduction

The following sections test Clojure code blocks. The specific behavior being examined is the content of the results block. The results block content will depend on the options set in the header arguments for the code block.

The example code used is as simple as possible. The content of the Clojure code block is a small vector. A Python equivalent is included for comparison.

These tests were conducted on January 30, 2014 using the master branch of the development version.

2 Code Block with no header options

```
#+begin_src clojure
[1 2 3 4]
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 |

#+begin_src python
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 |
```

Conclusion

Clojure and Python yield the same org table result.

3 Header Collection Options (value, output)

```
#+begin_src clojure :results value
[1 2 3 4]
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 |

#+begin_src python :results value
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 |

#+begin_src clojure :results output
[1 2 3 4]
#+end_src

#+RESULTS:
: ""

#+begin_src python :results output
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:

Python error when evaluating the above code block:
  File "<stdin>", line 2
SyntaxError: 'return' outside function
```

Conclusion: Clojure and Python results in org table with option :results value. Python error results when evaluating with option :results output.

4 Header Type Options (table, list, verbatim, file)

```
#+begin_src clojure :results value table
[1 2 3 4]
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 |

#+begin_src python :results value table
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 |

#+begin_src clojure :results value list
[1 2 3 4]
#+end_src

#+RESULTS:
- 1
- 2
- 3
- 4

#+begin_src python :results value list
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
- 1
- 2
- 3
- 4

#+begin_src clojure :results value verbatim
[1 2 3 4]
#+end_src

#+RESULTS:
: [1 2 3 4]

#+begin_src python :results value verbatim
 a = (1, 2, 3, 4)
 return a
#+end_src
```

```
#+RESULTS:
: (1, 2, 3, 4)

#+begin_src clojure :results value file
[1 2 3 4]
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 |

#+begin_src python :results value file
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 |
```

Conclusion

Clojure and Python yield similar results for table, list, and verbatim. The file option yields an org table. It should yield a file link.

5 Header Format Options (raw, org, html, latex, code, pp, drawer)

```
#+begin_src clojure :results value raw
[1 2 3 4]
#+end_src

#+RESULTS:
[1 2 3 4]
[1 2 3 4]

#+begin_src python :results value raw
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
(1, 2, 3, 4)
(1, 2, 3, 4)

#+begin_src clojure :results value org
[1 2 3 4]
#+end_src

#+RESULTS:
#+BEGIN_SRC org
[1 2 3 4]
#+END_SRC

#+begin_src python :results value org
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
#+BEGIN_SRC org
(1, 2, 3, 4)
#+END_SRC

#+begin_src clojure :results value html
[1 2 3 4]
#+end_src

#+RESULTS:
#+BEGIN_HTML
[1 2 3 4]
#+END_HTML

#+begin_src python :results value html
 a = (1, 2, 3, 4)
```

```
 return a
#+end_src

#+RESULTS:
#+BEGIN_HTML
(1, 2, 3, 4)
#+END_HTML

#+begin_src clojure :results value latex
[1 2 3 4]
#+end_src

#+RESULTS:
#+BEGIN_LaTeX
| 1 | 2 | 3 | 4 |
#+END_LaTeX

#+begin_src python :results value latex
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
#+BEGIN_LaTeX
| 1 | 2 | 3 | 4 |
#+END_LaTeX

#+begin_src clojure :results value code
[1 2 3 4]
#+end_src

#+RESULTS:
#+BEGIN_SRC clojure
"[1 2 3 4]\n"
#+END_SRC

#+begin_src python :results value code
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
#+BEGIN_SRC python
(1, 2, 3, 4)
#+END_SRC

#+begin_src clojure :results value pp
[1 2 3 4]
#+end_src
```

```
#+RESULTS:
: "[1 2 3 4]\n"

#+begin_src python :results value pp
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
: (1, 2, 3, 4)

#+begin_src clojure :results value drawer
[1 2 3 4]
#+end_src

#+RESULTS:
:RESULTS:
[1 2 3 4]
:END:

#+begin_src python :results value drawer
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:
:RESULTS:
(1, 2, 3, 4)
:END:
```

Conclusion

Clojure and Python result for the raw option is similar. However, multiple evaluations are causing the result to stack. The default of replace does not appear to be in effect.

Clojure and Python result for org, html, and latex options are similar.

Clojure result for the code option shows the result wrapped in quotations and a line feed appended to the vector. The Python result is only the code.

Clojure result for the pp option has the result with a colon, quotes around the vector and a line feed appended to the end of the vector. Python uses only a colon followed by the code.

Clojure and Python have similar results for the drawer option.

6 Header Handling Options (silent, replace, append, prepend)

```
#+begin_src clojure :results output silent
[1 2 3 4]
#+end_src

#+begin_src python :results output silent
 a = (1, 2, 3, 4)
 return a
#+end_src

#+begin_src clojure :results output replace
[1 2 3 4]
#+end_src

#+RESULTS:
: ""

#+begin_src python :results output replace
 a = (1, 2, 3, 4)
 return a
#+end_src

#+RESULTS:

Python error for above code block:
  File "<stdin>", line 2
SyntaxError: 'return' outside function

#+begin_src clojure :results value append
[1 2 3 4 5]
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 |   |
| 1 | 2 | 3 | 4 | 5 |

#+begin_src python :results value append
 a = (1, 2, 3, 4, 5)
 return a
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 |   |
| 1 | 2 | 3 | 4 | 5 |

#+begin_src clojure :results value prepend
[1 2 3 4 5]
#+end_src
```

```
#+RESULTS:
| 1 | 2 | 3 | 4 | 5 |
| 1 | 2 | 3 | 4 |   |

#+begin_src python :results value prepend
 a = (1, 2, 3, 4, 5)
 return a
#+end_src

#+RESULTS:
| 1 | 2 | 3 | 4 | 5 |
| 1 | 2 | 3 | 4 |   |
```

Conclusion:

Clojure and Python results are consistent for option silent.

Clojure result with replace option is functional. The Python code block generates an error message.

Clojure and Python results with append and prepend are similar and function correctly.