Can I Remove This Method? How Live Feedback from the Ecosystem Supports Co-Evolution

Manuel Leuenberger

manuel.leuenberger@inf.unibe.ch

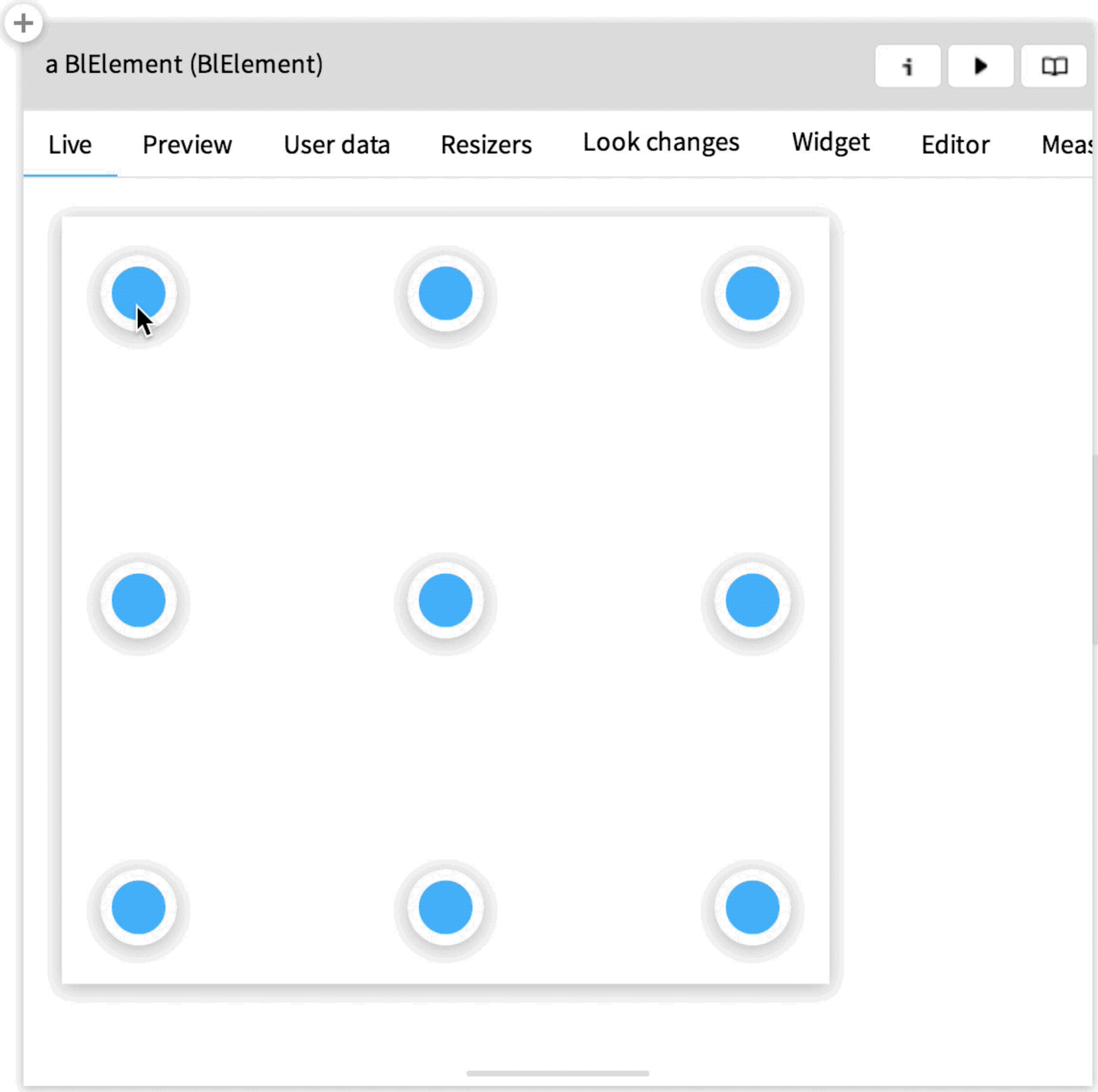
@maenuleu

Software Composition Group
Universität Bern, CH

IWST 2019

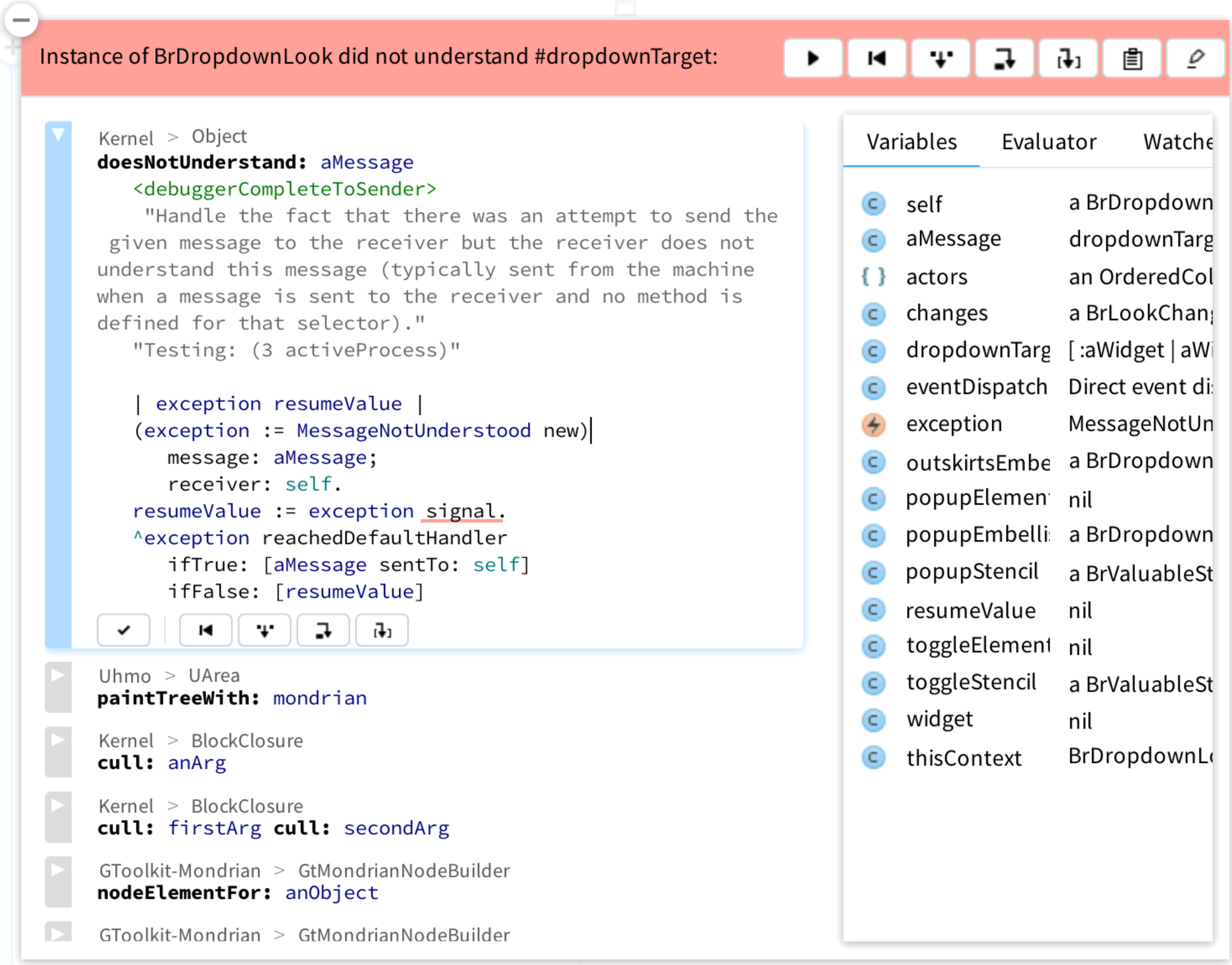
me, a library developer

Focus: Brick Dropdown



A history of co-evolution

- push new dropdown to Brick
- Glamorous Toolkit CI passes
- all good?



Instance of BrDropdownLook did not understand #dropdownTarget:

Kernel > Object

```
doesNotUnderstand: aMessage
   <debuggerCompleteToSender>
    "Handle the fact that there was an attempt to send the
 given message to the receiver but the receiver does not
understand this message (typically sent from the machine
when a message is sent to the receiver and no method is
defined for that selector)."
   "Testing: (3 activeProcess)"

   | exception resumeValue |
   (exception := MessageNotUnderstood new)
      message: aMessage;
      receiver: self.
   resumeValue := exception signal.
   ^exception reachedDefaultHandler
      ifTrue: [aMessage sentTo: self]
      ifFalse: [resumeValue]
```

Uhmo > UArea
paintTreeWith: mondrian

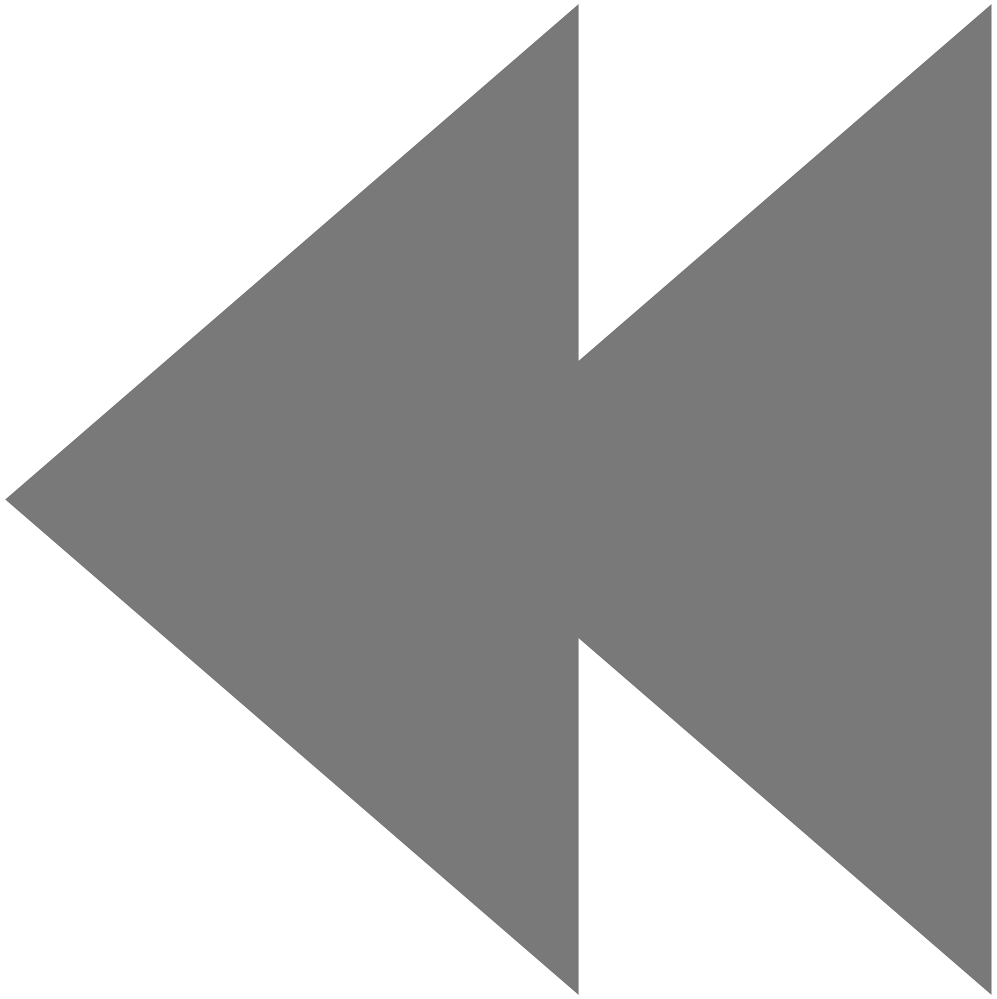
Kernel > BlockClosure
cull: anArg

Kernel > BlockClosure
cull: firstArg **cull:** secondArg

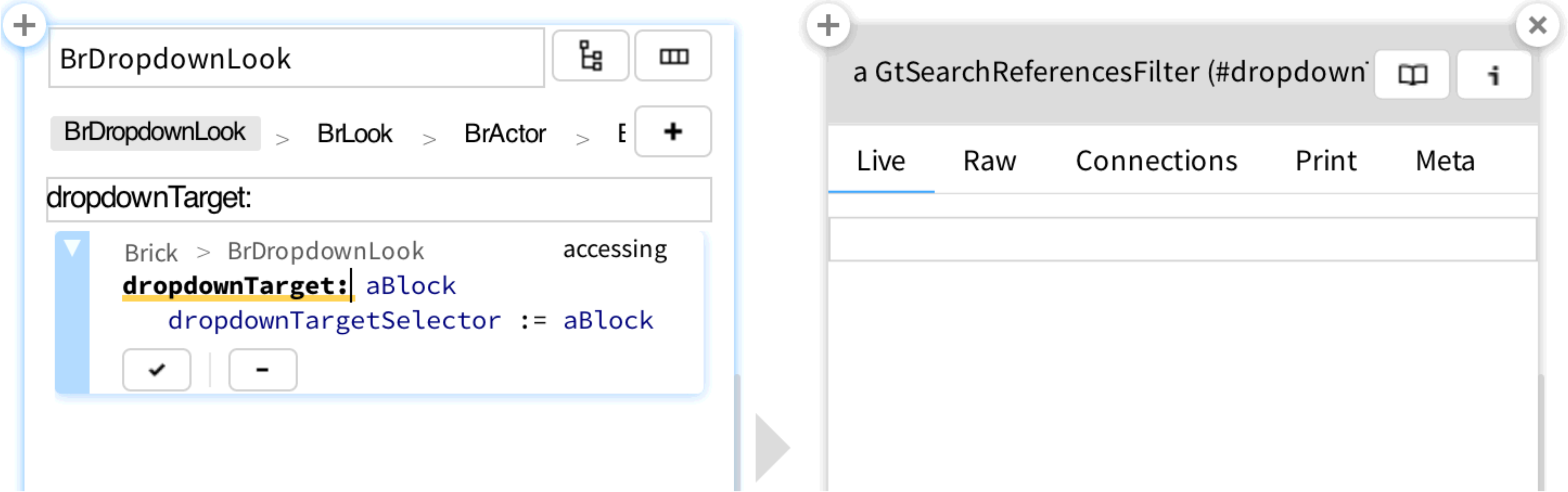
GToolkit-Mondrian > GtMondrianNodeBuilder
nodeElementFor: anObject

GToolkit-Mondrian > GtMondrianNodeBuilder

self	a BrDropdown
aMessage	dropdownTarg
actors	an OrderedCol
changes	a BrLookChan
dropdownTarg	[:aWidget | aWi
eventDispatch	Direct event di
exception	MessageNotUn
outskirtsEmbe	a BrDropdown
popupElemen	nil
popupEmbelli	a BrDropdown
popupStencil	a BrValuableSt
resumeValue	nil
toggleElement	nil
toggleStencil	a BrValuableSt
widget	nil
thisContext	BrDropdownL



Can I remove this method?

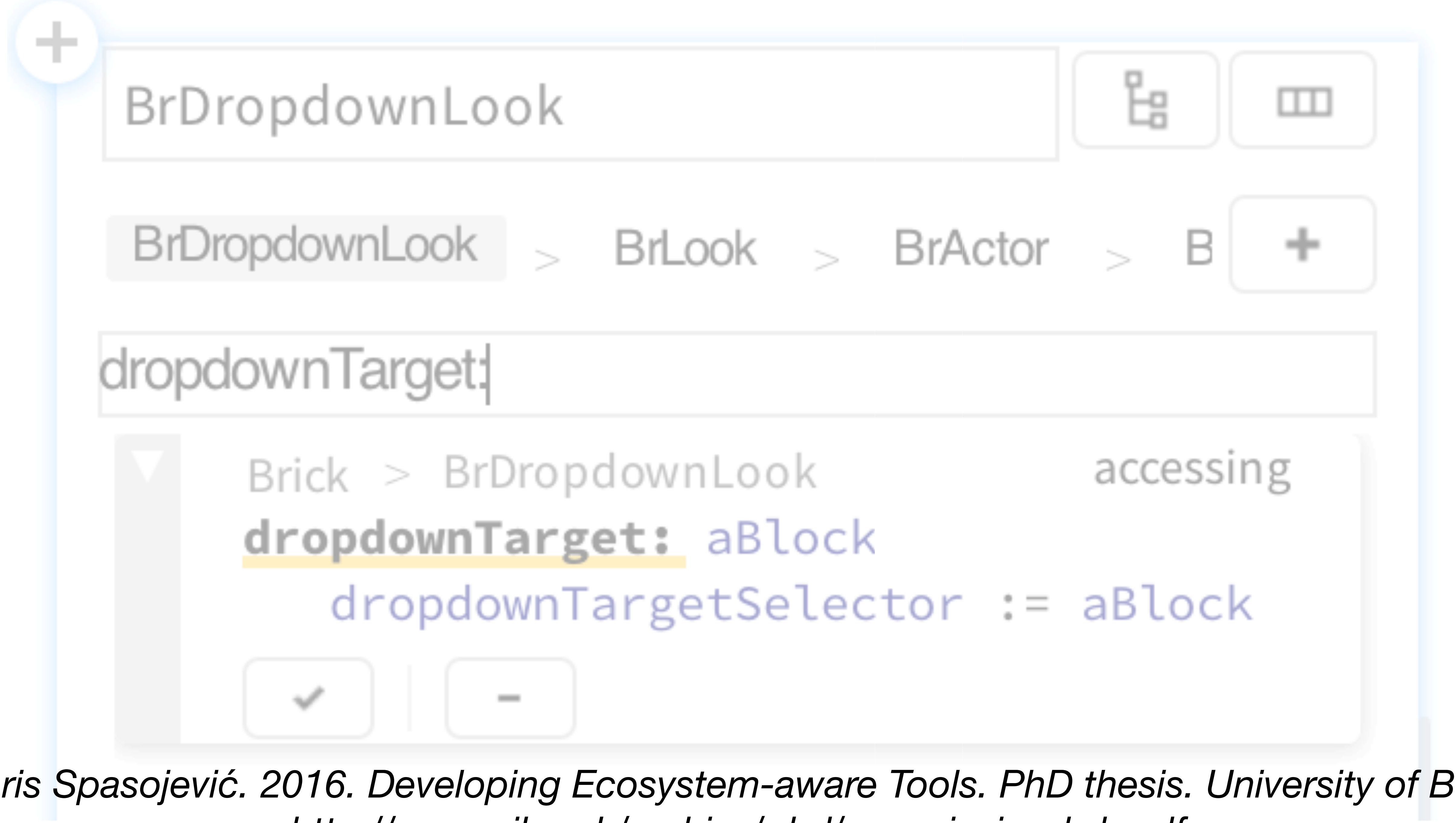


```
dropdownTarget: aBlock
    dropdownTargetSelector := aBlock
```

no senders within the image, nobody seems to use it

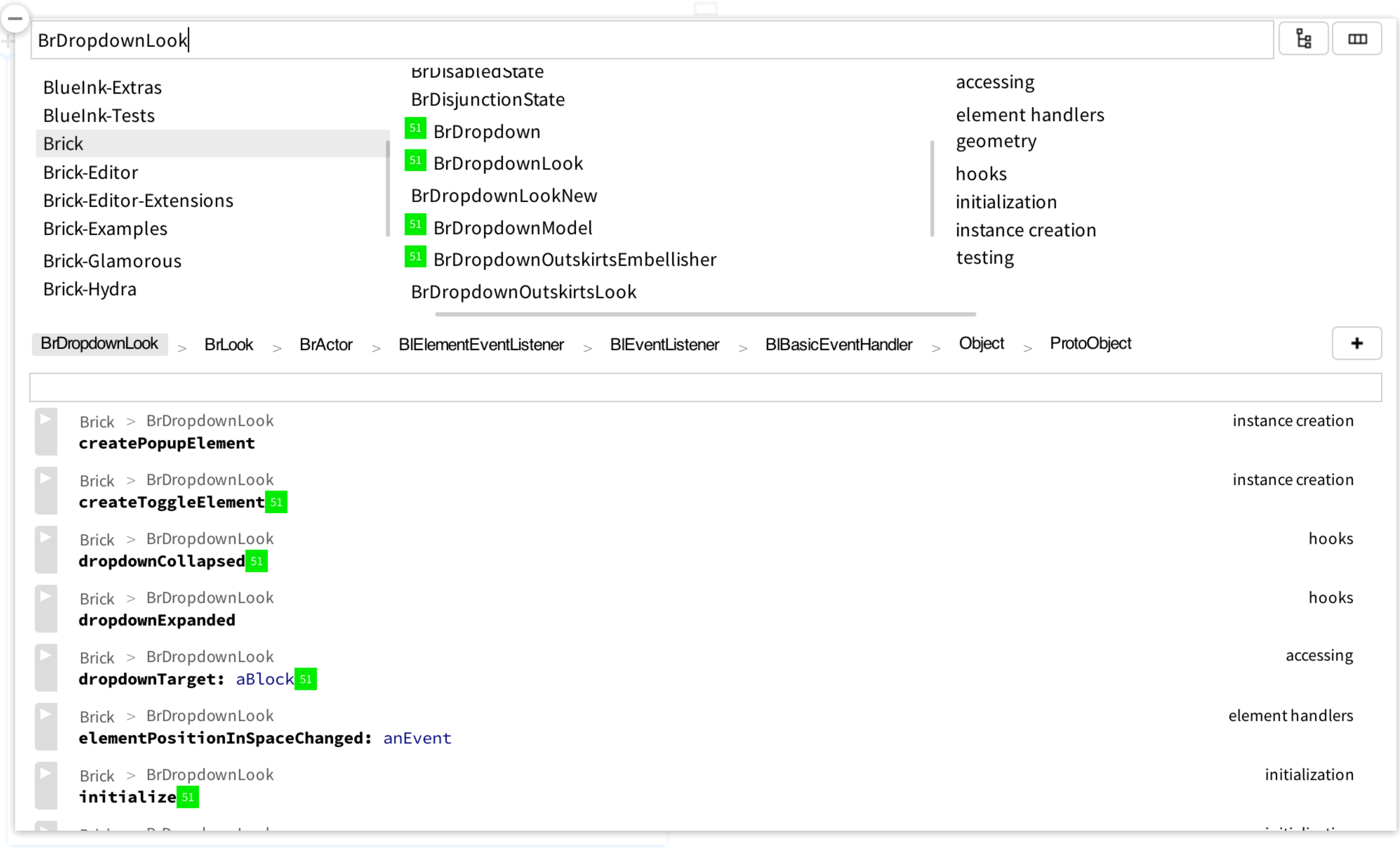
But I am not sure

Who is using a method *in the ecosystem*?



```
dropdownTarget: aBlock
    dropdownTargetSelector := aBlock
```

Boris Spasojević. 2016. Developing Ecosystem-aware Tools. PhD thesis. University of Bern. http://scg.unibe.ch/archive/phd/spasojevic-phd. pdf



BrDropdownLook

BlueInk-Extras
BlueInk-Tests
Brick
Brick-Editor
Brick-Editor-Extensions
Brick-Examples
Brick-Glamorous
Brick-Hydra

BrDisabledState
BrDisjunctionState
51 BrDropdown
51 BrDropdownLook
BrDropdownLookNew
51 BrDropdownModel
51 BrDropdownOutskirtsEmbellisher
BrDropdownOutskirtsLook

accessing
element handlers
geometry
hooks
initialization
instance creation
testing

BrDropdownLook > BrLook > BrActor > BlElementEventListener > BlEventListener > BlBasicEventHandler > Object > ProtoObject

Brick > BrDropdownLook
createPopupElement
instance creation

Brick > BrDropdownLook
createToggleElement 51
instance creation

Brick > BrDropdownLook
dropdownCollapsed 51
hooks

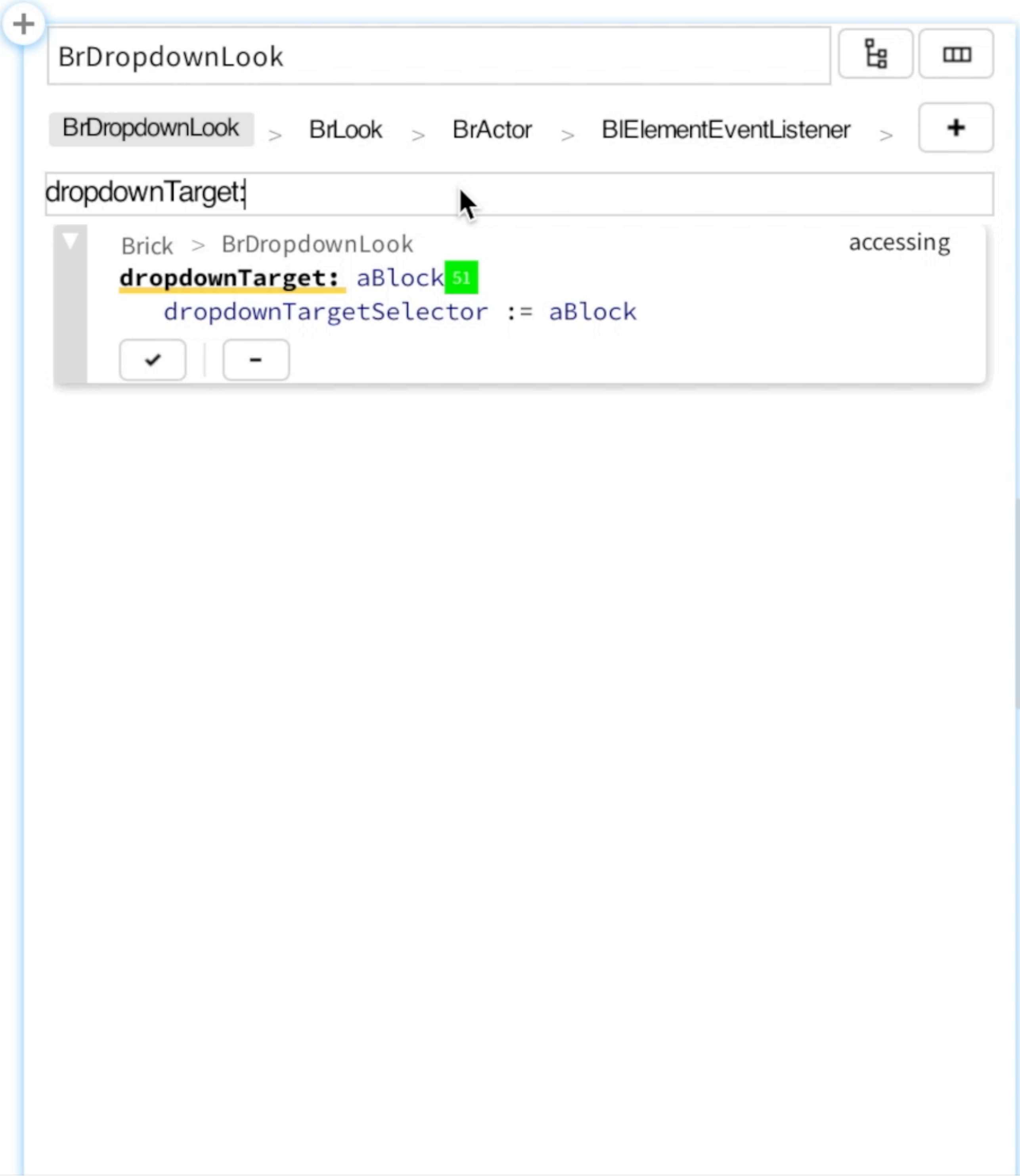
Brick > BrDropdownLook
dropdownExpanded
hooks

Brick > BrDropdownLook
dropdownTarget: aBlock 51
accessing

Brick > BrDropdownLook
elementPositionInSpaceChanged: anEvent
element handlers

Brick > BrDropdownLook
initialize 51
initialization

Brick > BrDropdownLook
initialization



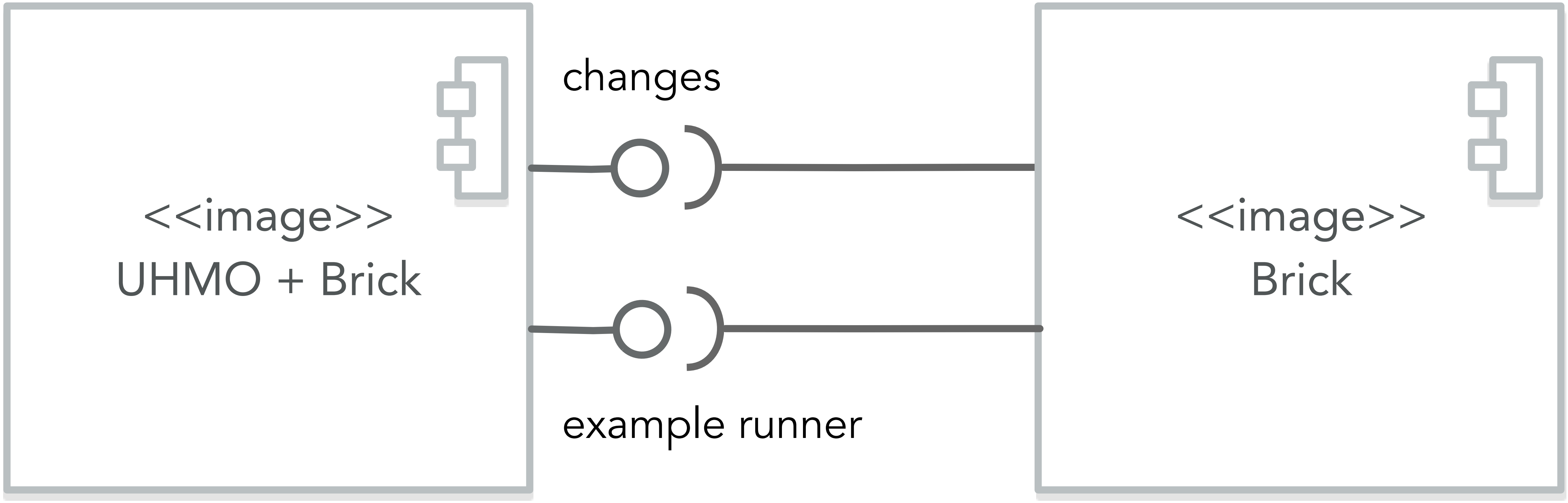
```
dropdownTarget: aBlock
	dropdownTargetSelector := aBlock
```

Pros and Cons

+ integrate information from the ecosystem
+ serves as sanity checks for changes
+ *live* feedback on the impact of library changes on clients

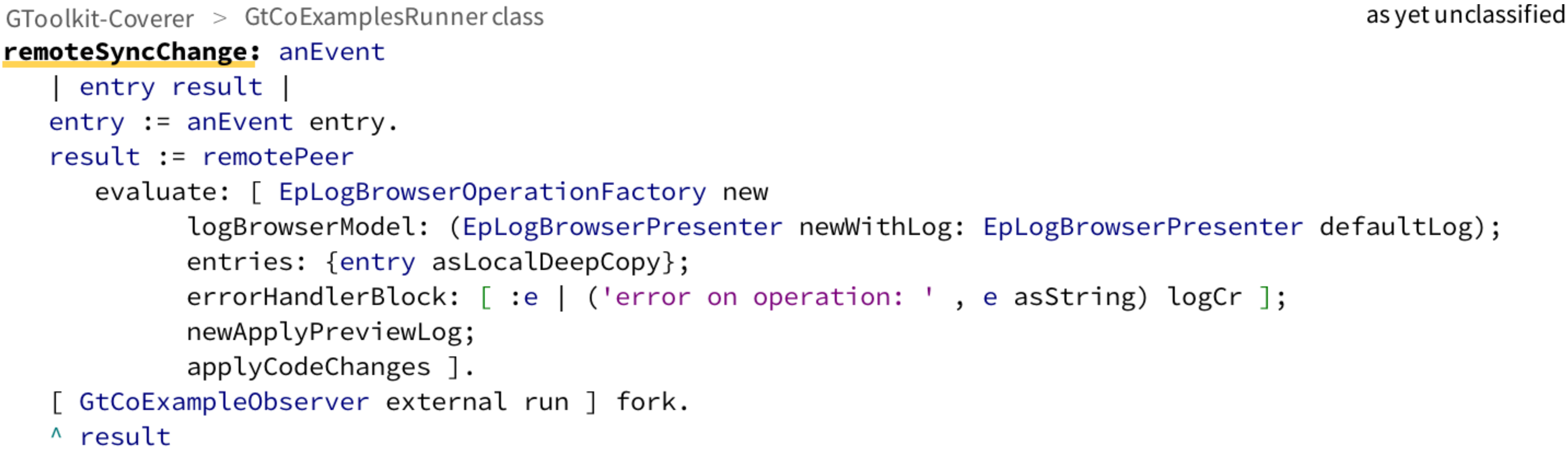
- incomplete: 100% precision, unknown recall
- recall depends on degree of exercising library
- requires running || *{devs} x {endorsed library clients}* || images
- local vs. remote: logging? inspection? debugging?

Propagate changes & run examples



Propagating changes

Seamless + Epicea



GToolkit-Coverer > GtCoExamplesRunner class

as yet unclassified

```
remoteSyncChange: anEvent
   | entry result |
  entry := anEvent entry.
  result := remotePeer
     evaluate: [ EpLogBrowserOperationFactory new
           logBrowserModel: (EpLogBrowserPresenter newWithLog: EpLogBrowserPresenter defaultLog);
           entries: {entry asLocalDeepCopy};
           errorHandlerBlock: [ :e | ('error on operation: ' , e asString) logCr ];
           newApplyPreviewLog;
           applyCodeChanges ].
  [ GtCoExampleObserver external run ] fork.
  ^ result
```

Profiling examples

Seamless + MetaLinks

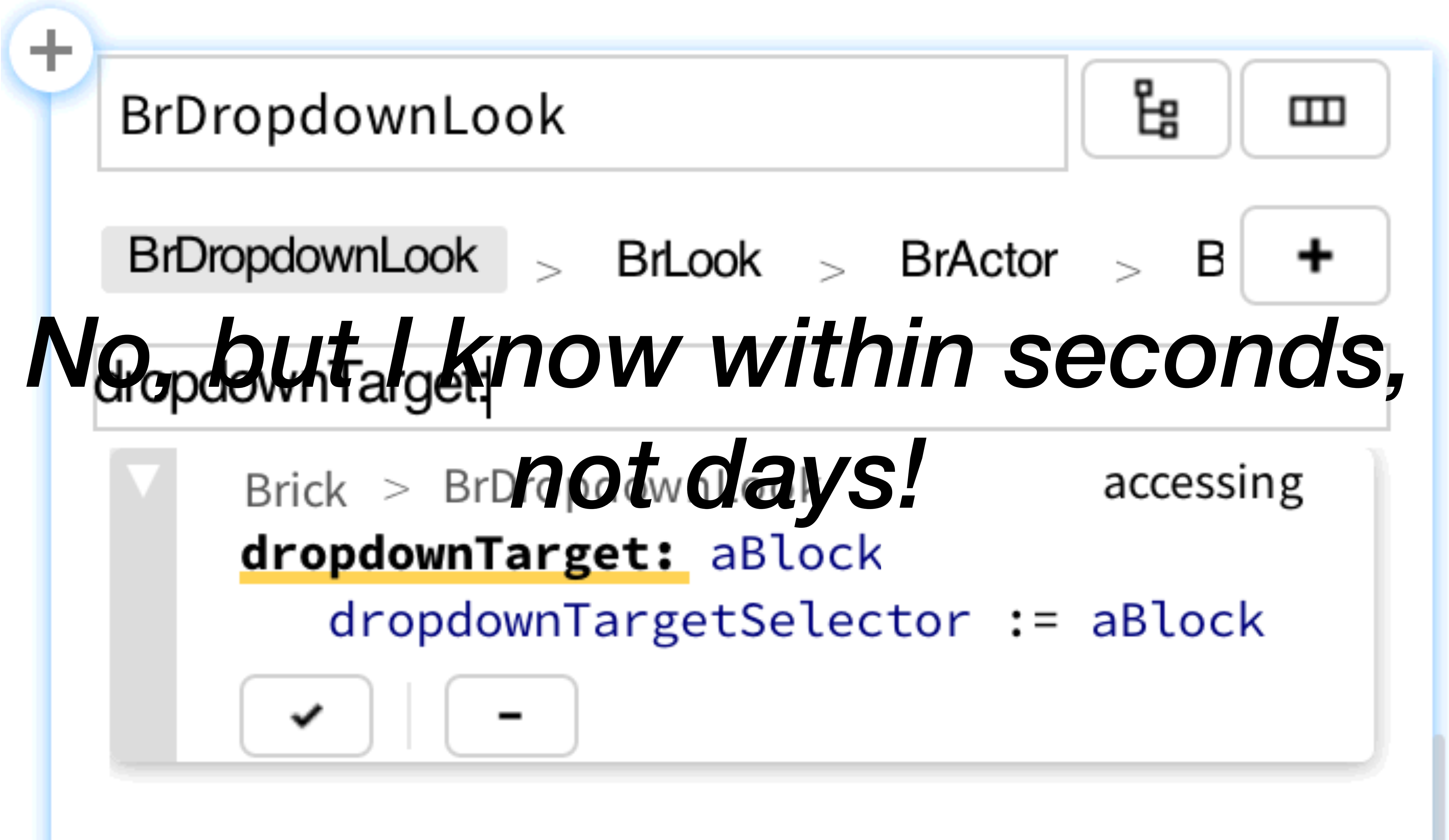
- 191 examples in UHMO
- only interested in examples using dropdown

! run only examples calling dropdown methods

In the UHMO image:

1. instrument 75 methods in dropdown package
2. run all 191 examples, takes 11s
3. identify 51 examples calling dropdown methods
4. remove instrumentation from dropdown methods
5. run 51 UHMO examples in 5s on every change

Can I remove this method?



BrDropdownLook

```
dropdownTarget: aBlock
    dropdownTargetSelector := aBlock
```

No, but I know within seconds, not days!

Next: Better co-evolution

- detecting breaking changes
- estimating change impact
- *automatically migrate library clients*
  - provide migration process alongside library version
  - code transformations, just as database migrations
  - domain-specific tools to guide the process