

Debugging Accessibility Issues





Orca 101



Screen Readers Present Screen Contents

- To users who are blind or visually impaired
- In synthesized speech and/or refreshable braille
- As they navigate using the application's commands:
 - Focus changes
 - Caret movement
 - Selection changes
- And when “events of interest” occur in non-focused widgets
 - Notifications
 - Incoming chat messages

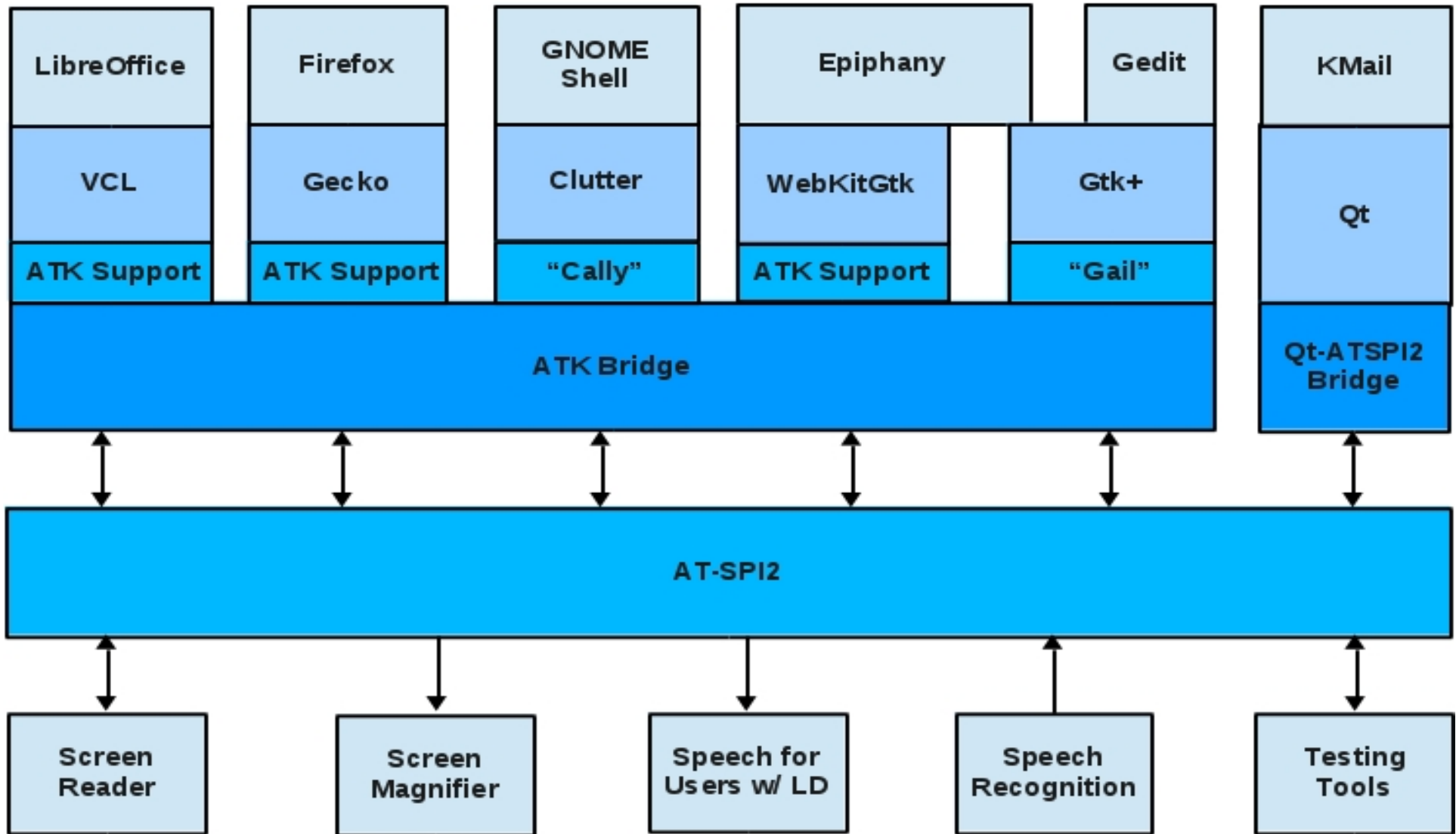
Screen Readers Provide Additional Commands For More Efficient Access

- Review chat messages and notifications
- Navigate by element type
- Obtain a summary of font/formatting information
- Search the screen for an object
- Etc.

Orca: A Screen Reader for the Free Desktop

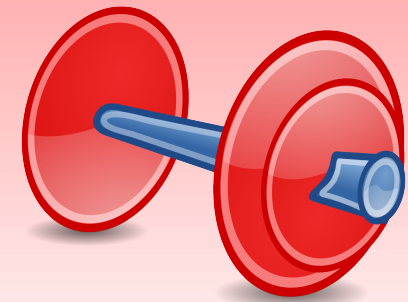
- Written in Python
- Access to graphical desktop environments
- Presentation:
 - Of objects, events, and other functionality via AT-SPI2
 - Speech synthesis via speech-dispatcher
 - Braille translation via liblouis
 - Braille display via brlapi

Where Is the Orca Bug?





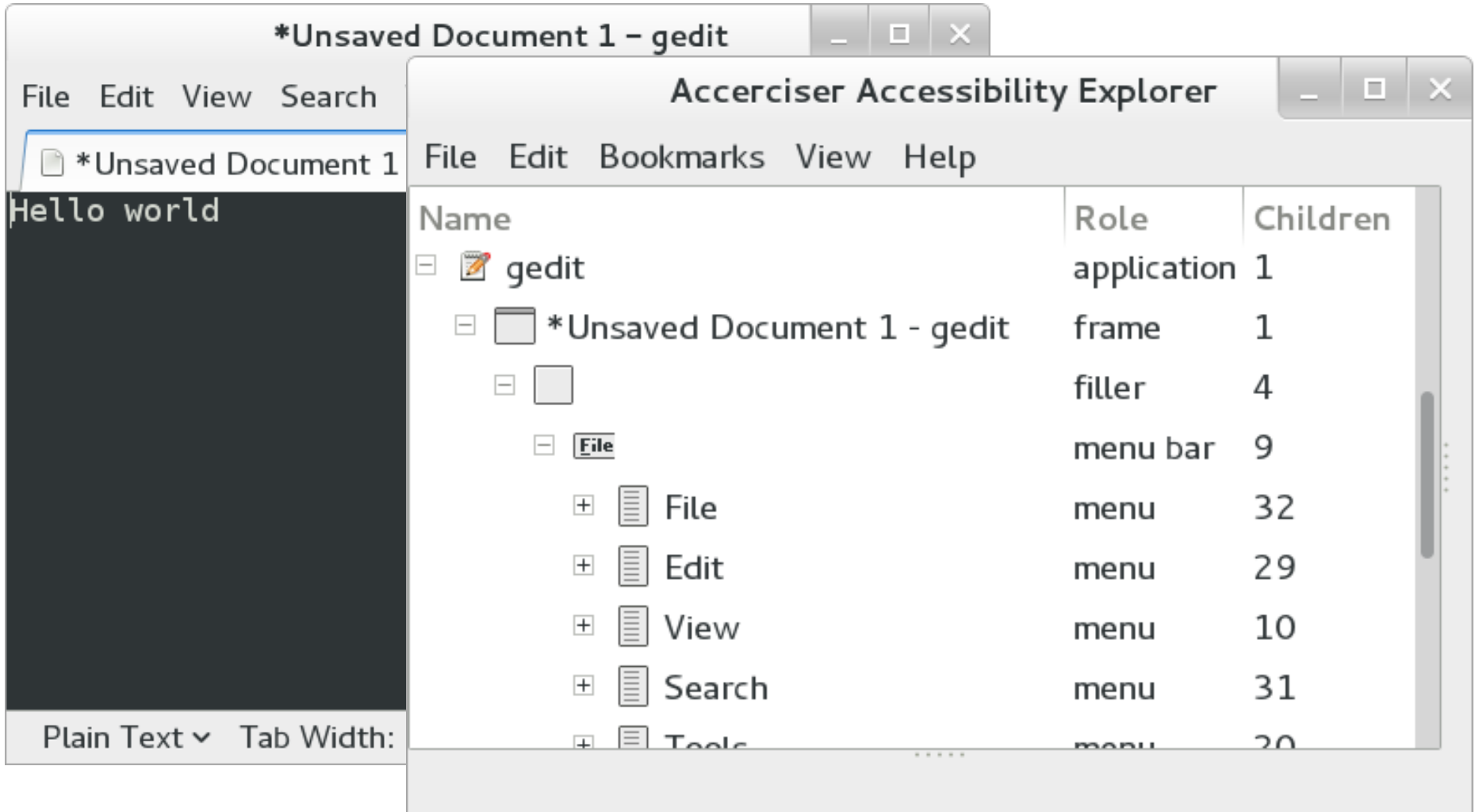
Accerciser 101



Accerciser: The Interactive “Accessibility Explorer”

- Written in Python
- View accessible objects
- Examine implemented accessibility interfaces
- Monitor accessibility events
- Interact with accessible objects through their interfaces

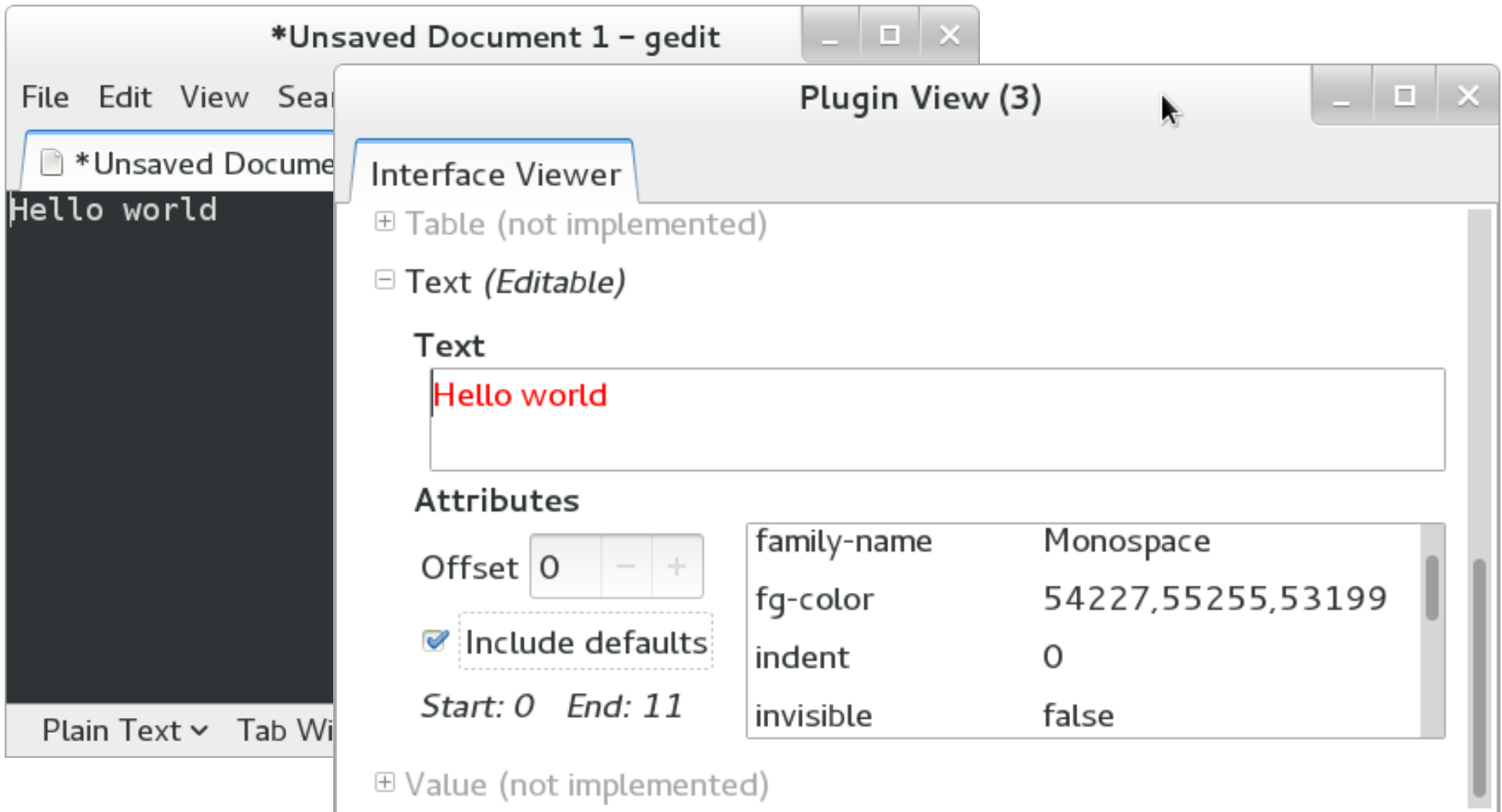
Accerciser: View Accessible Objects



The screenshot shows the Accerciser Accessibility Explorer window overlaid on a gedit window. The gedit window contains the text "Hello world". The Accerciser window displays a tree view of accessible objects with the following structure:

Name	Role	Children
gedit	application	1
*Unsaved Document 1 - gedit	frame	1
	filler	4
File	menu bar	9
File	menu	32
Edit	menu	29
View	menu	10
Search	menu	31
Tools	menu	20

Accerciser: Examine Implemented Interfaces

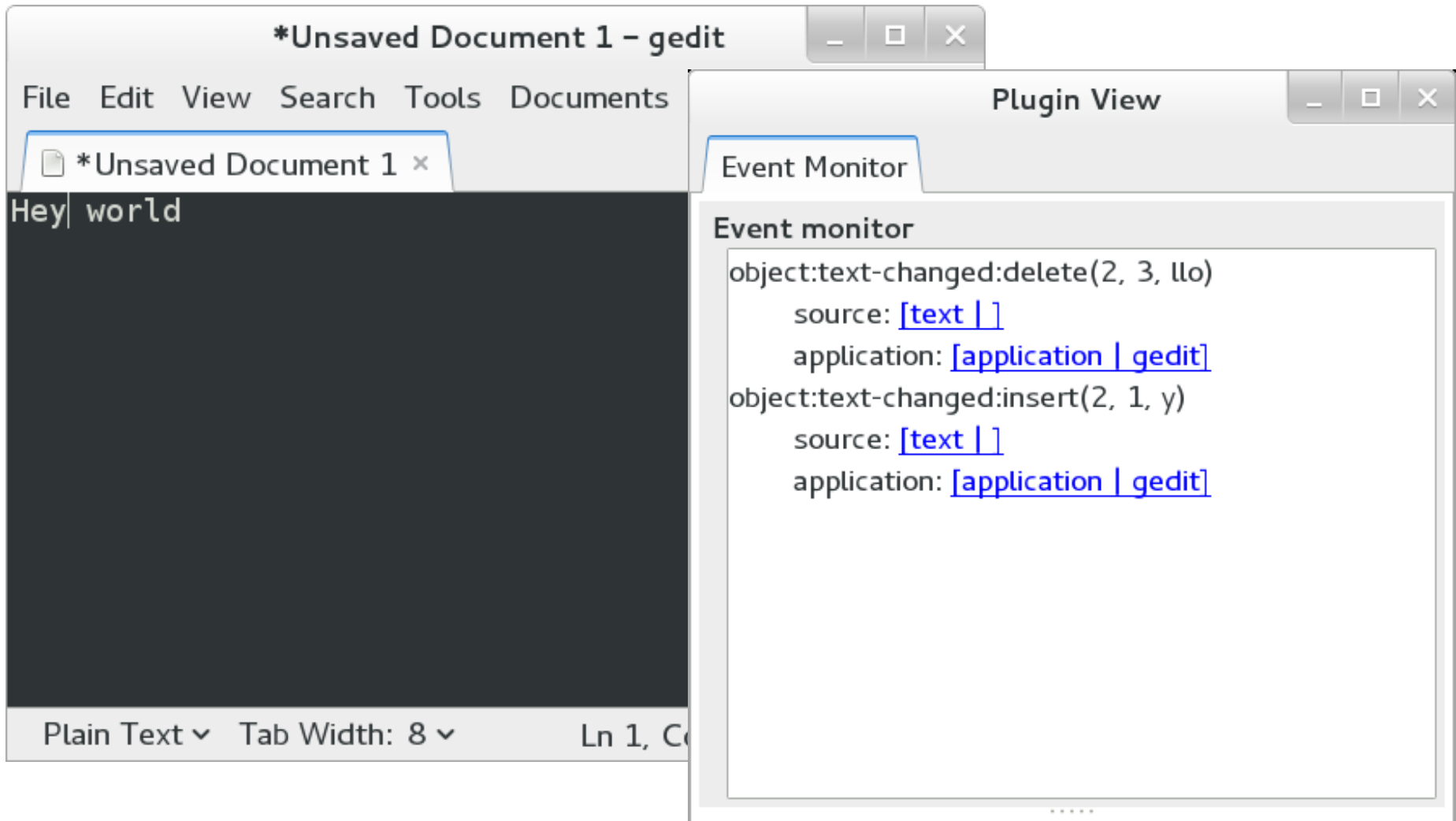


The screenshot shows the Accerciser application window titled "Plugin View (3)". The "Interface Viewer" is active, showing a tree view of the widget hierarchy. The selected widget is a "Text (Editable)" widget. The text "Hello world" is displayed in a red font. The "Attributes" section shows the following properties:

Attribute	Value
family-name	Monospace
fg-color	54227,55255,53199
indent	0
invisible	false

Additional details shown include an "Offset" of 0, a checked "Include defaults" checkbox, and a range of "Start: 0 End: 11".

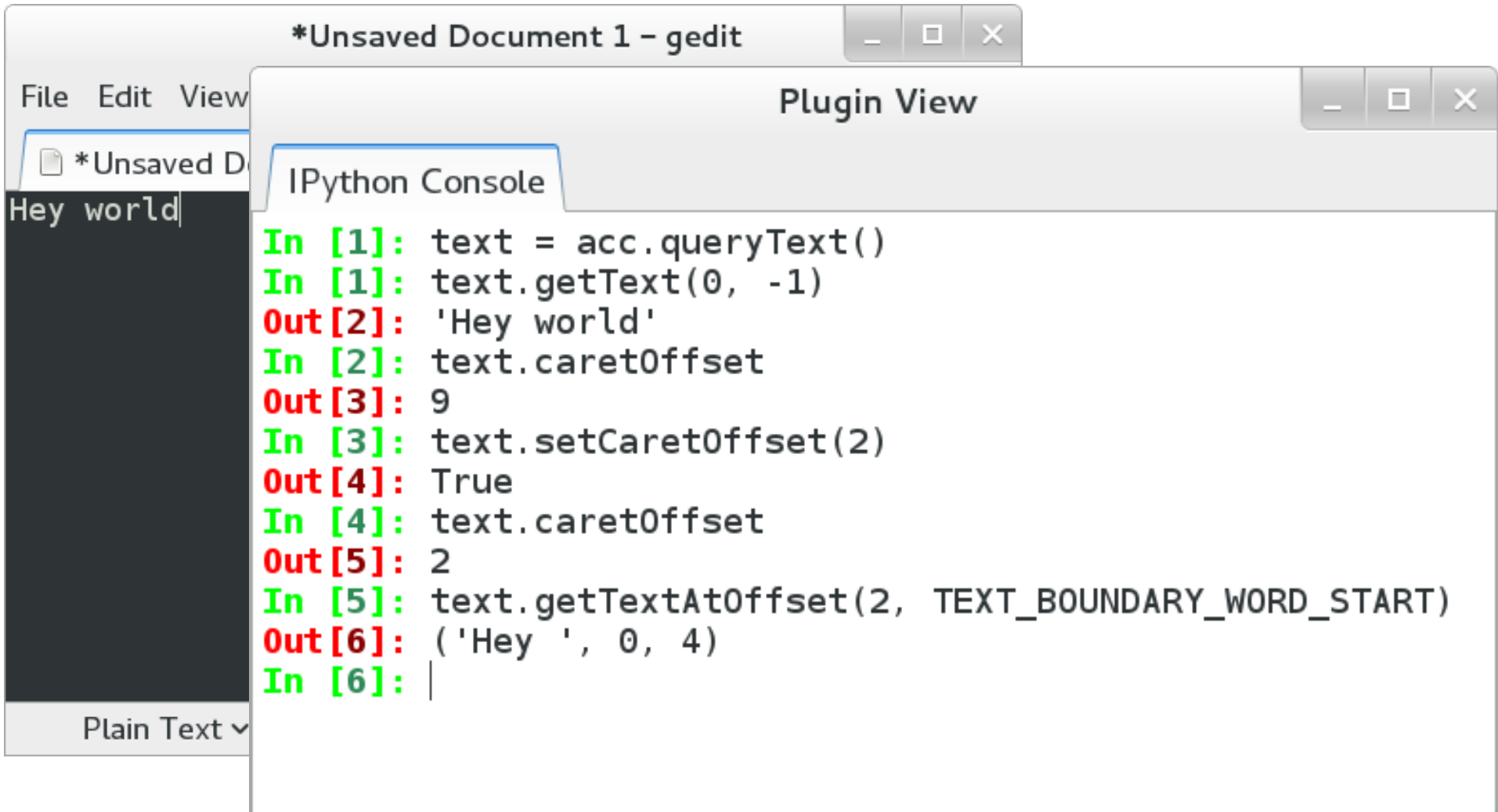
Accerciser: Monitor Accessibility Events



The screenshot shows the Gedit text editor window titled '*Unsaved Document 1 - gedit'. The menu bar includes File, Edit, View, Search, Tools, and Documents. A single document tab is open, showing the text 'Hey| world'. The status bar at the bottom indicates 'Plain Text', 'Tab Width: 8', and 'Ln 1, Co...'. An 'Event Monitor' plugin window is overlaid on the editor, displaying the following event log:

```
Event monitor
object:text-changed:delete(2, 3, llo)
  source: [text | ]
  application: [application | gedit]
object:text-changed:insert(2, 1, y)
  source: [text | ]
  application: [application | gedit]
```

Accerciser: Interact with Accessible Objects



*Unsaved Document 1 - gedit

File Edit View

*Unsaved D

Hey world

Plain Text v

Plugin View

IPython Console

```
In [1]: text = acc.queryText()
In [1]: text.getText(0, -1)
Out [2]: 'Hey world'
In [2]: text.caretOffset
Out [3]: 9
In [3]: text.setCaretOffset(2)
Out [4]: True
In [4]: text.caretOffset
Out [5]: 2
In [5]: text.getTextAtOffset(2, TEXT_BOUNDARY_WORD_START)
Out [6]: ('Hey ', 0, 4)
In [6]: |
```

Is the bug in the application or toolkit?

- Do you see the objects to be presented by Orca in Accerciser?
- Do the objects emit accessible events when you navigate?
- Do the objects have the needed accessible properties?
- Do the objects implement the needed accessible interfaces?

If you CANNOT answer “yes” to all of the above, the bug is likely in the application or toolkit

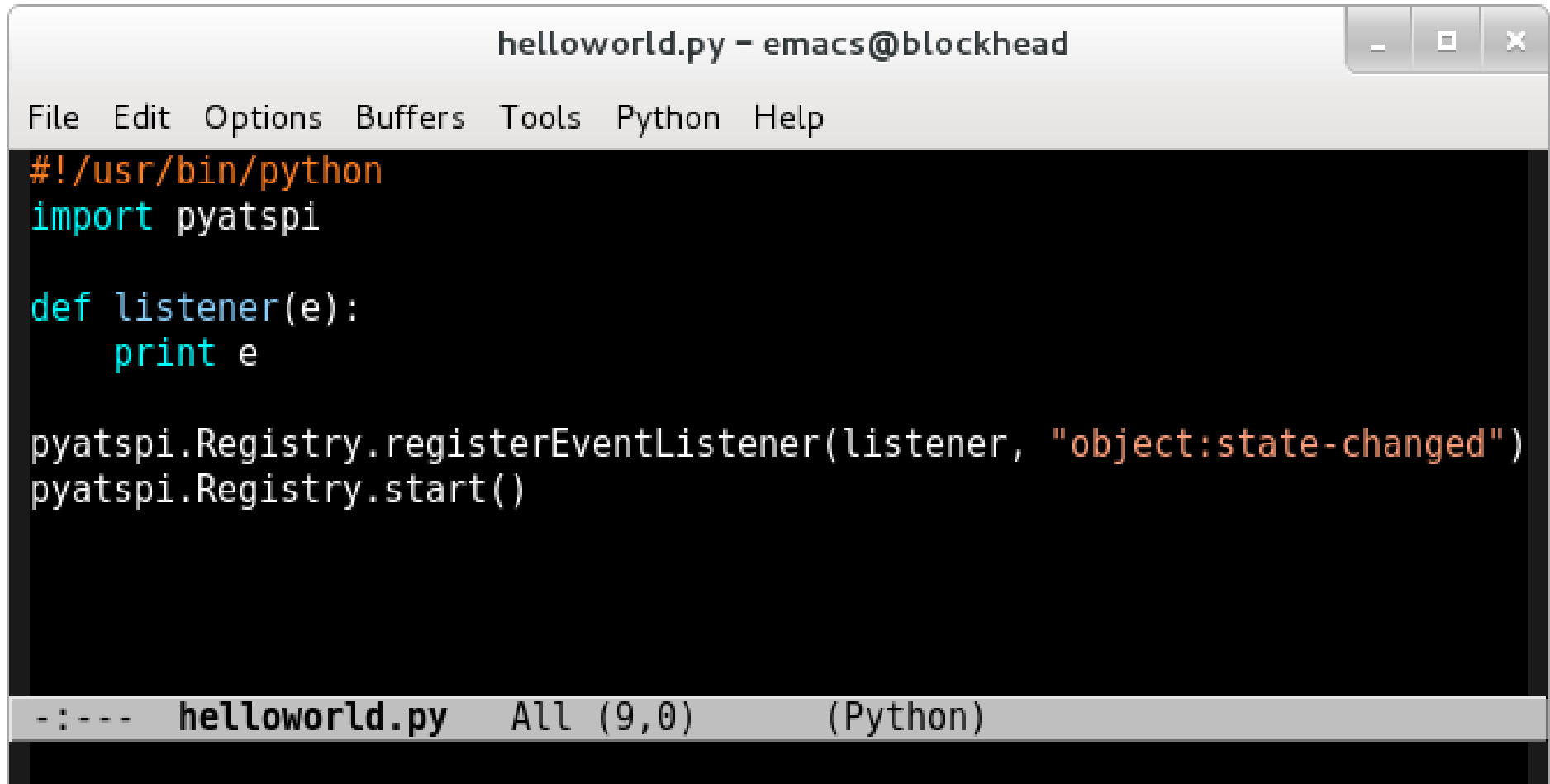


Rolling Your Own Listener

Rolling Your Own Listener

- Pick the event type of interest
- Define the handler's behavior
- Register the listener
- Start the registry

Hello World Listener



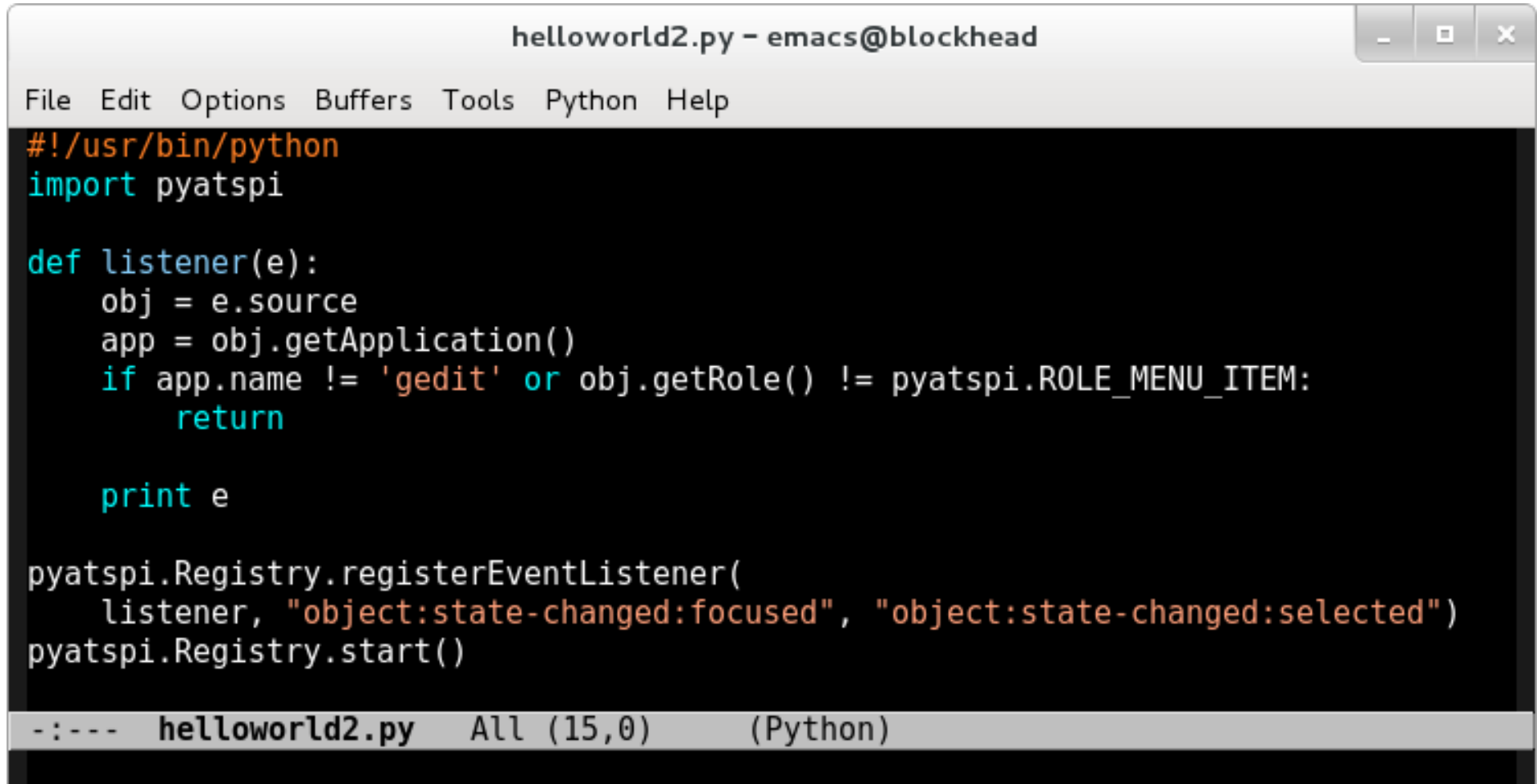
```
helloworld.py - emacs@blockhead
File Edit Options Buffers Tools Python Help
#!/usr/bin/python
import pyatspi

def listener(e):
    print e

pyatspi.Registry.registerEventListener(listener, "object:state-changed")
pyatspi.Registry.start()

-:--- helloworld.py All (9,0) (Python)
```

Hello World Listener 2



```
helloworld2.py - emacs@blockhead
File Edit Options Buffers Tools Python Help
#!/usr/bin/python
import pyatspi

def listener(e):
    obj = e.source
    app = obj.getApplication()
    if app.name != 'gedit' or obj.getRole() != pyatspi.ROLE_MENU_ITEM:
        return

    print e

pyatspi.Registry.registerEventListener(
    listener, "object:state-changed:focused", "object:state-changed:selected")
pyatspi.Registry.start()

-:--- helloworld2.py All (15,0) (Python)
```

Hello World Listener 2 Results

```
object:state-changed:selected(1, 0, 0)
  source: [menu item | New]
  host_application: [application | gedit]
object:state-changed:selected(0, 0, 0)
  source: [menu item | New]
  host_application: [application | gedit]
object:state-changed:selected(1, 0, 0)
  source: [menu item | Open...]
  host_application: [application | gedit]
object:state-changed:selected(0, 0, 0)
  source: [menu item | Open...]
  host_application: [application | gedit]
object:state-changed:selected(1, 0, 0)
  source: [menu item | Save]
  host_application: [application | gedit]
object:state-changed:selected(0, 0, 0)
  source: [menu item | Save]
  host_application: [application | gedit]
```

Hello World Listener 2 Translation

```
object:state-changed:selected(1, 0, 0)
  source: [menu item | New]
  host_application: [application | gedit]
object:state-changed:selected(0, 0, 0)
  source: [menu item | New]
  host_application: [application | gedit]
object:state-changed:selected(1, 0, 0)
  source: [menu item | Open...]
  host_application: [application | gedit]
object:state-changed:selected(0, 0, 0)
  source: [menu item | Open...]
  host_application: [application | gedit]
object:state-changed:selected(1, 0, 0)
  source: [menu item | Save]
  host_application: [application | gedit]
object:state-changed:selected(0, 0, 0)
  source: [menu item | Save]
  host_application: [application | gedit]
```

To present the selected menu item as a user navigates:

- Listen for:
 - object:state-changed:selected
 - event.detail1 == 1
- Speak and braille:
 - event.source.name

TODO:

Investigate why we are not getting the expected focused events.