Indywiki: Visually browsing Wikipedia

Markos Gogoulos
mgogoulos@gmail.com
FOSDEM 2008

Why the project was started

“Some times browsing over the plethora of information available through Wikipedia pages becomes a boring activity...”

Indywiki project aims to explore different ways of browsing the Wikipedia universe!

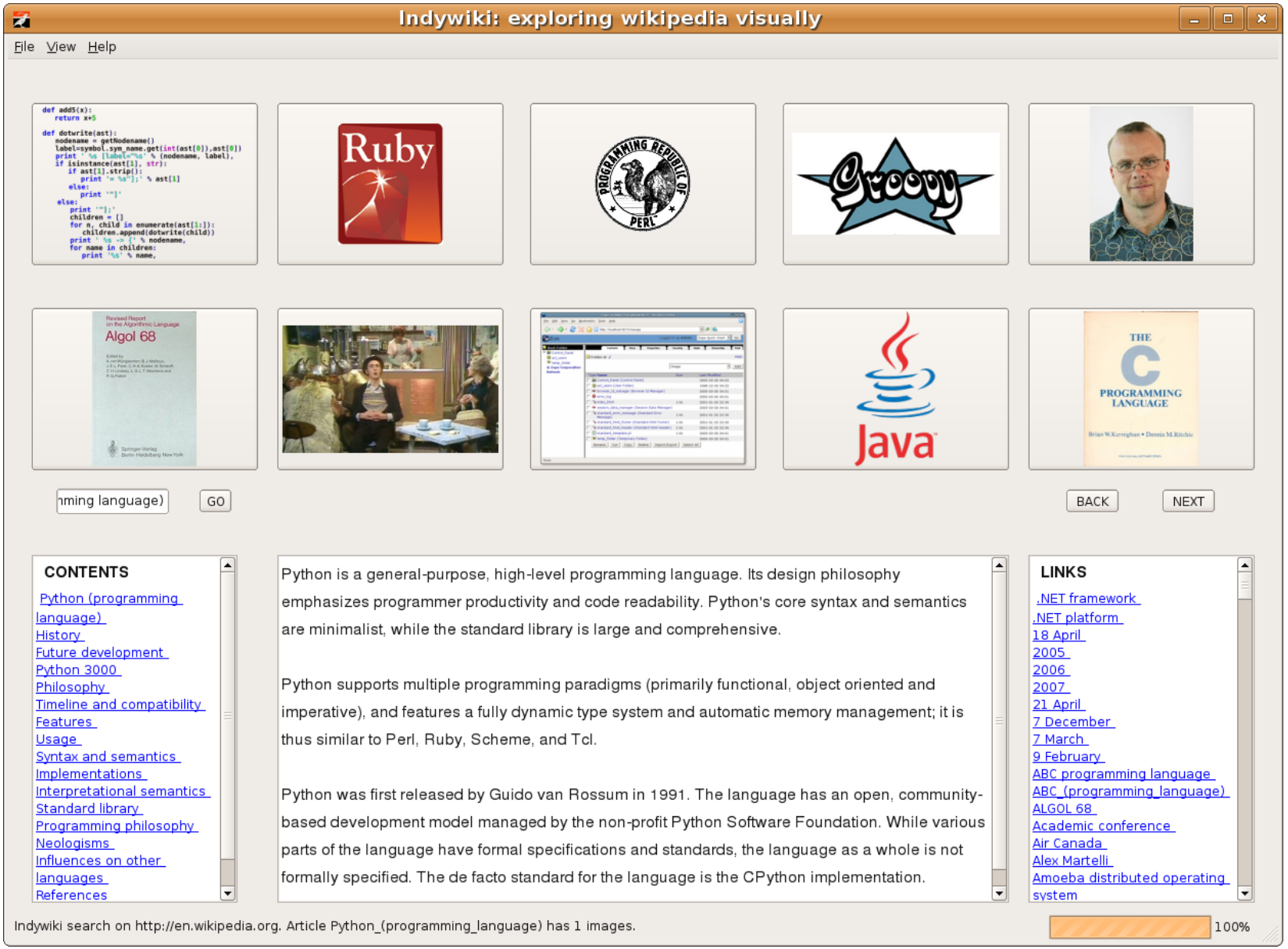
More specifically...

- Sometimes you just need to get a quick idea on an article
- get all images of an article together (in tenths)
- use as testbed for changes/experimentation
- ...and give something back to the amazing community of Wikipedia

Critics

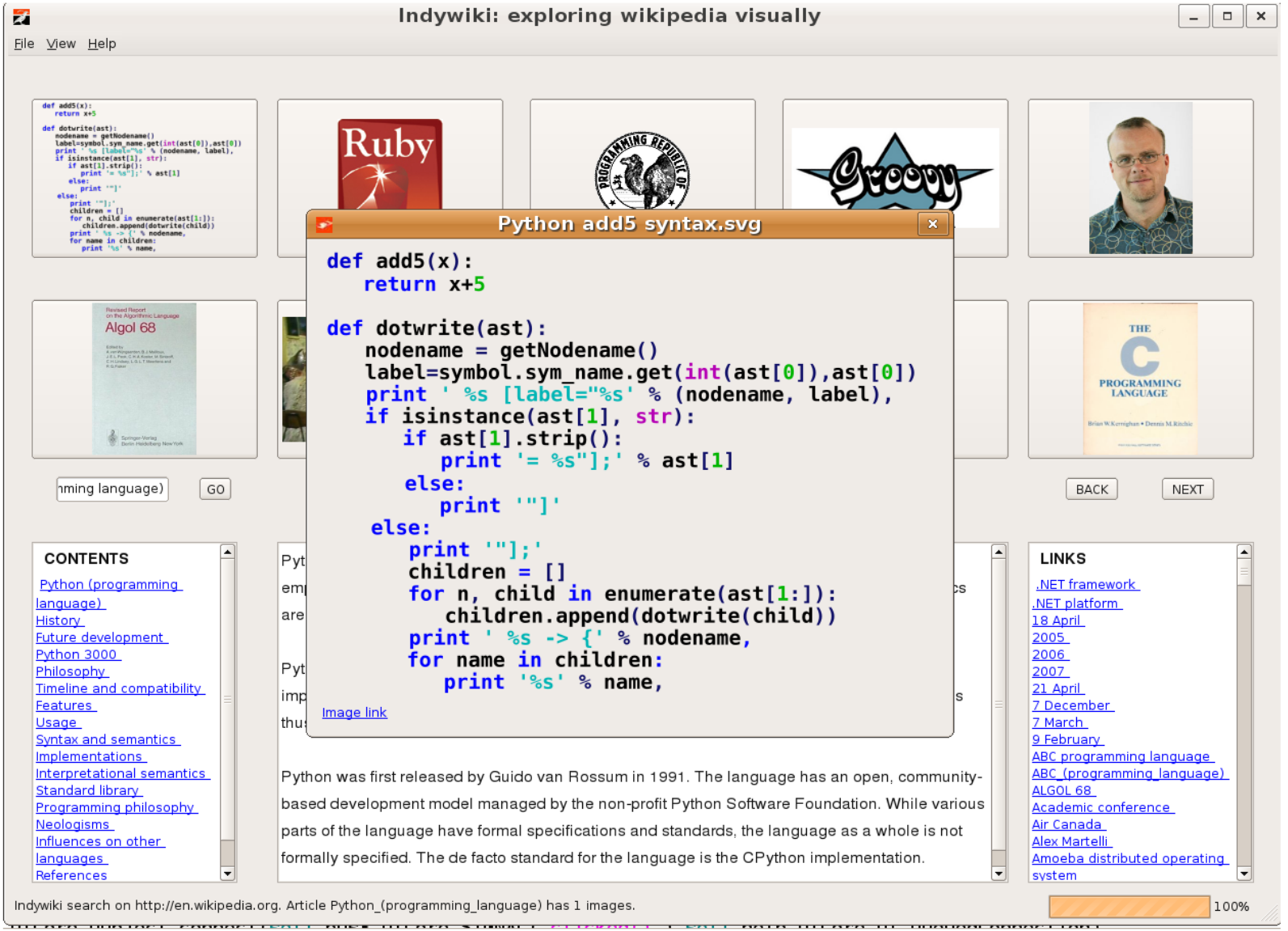
- Some people shout that *'Wikipedia is unreliable'*
- One should first check current research on Wikipedia reliability and then talk.
- I couldn't care less :) This is an open source project/ experiment!

Visually browsing Wikipedia?



"browsing an article "

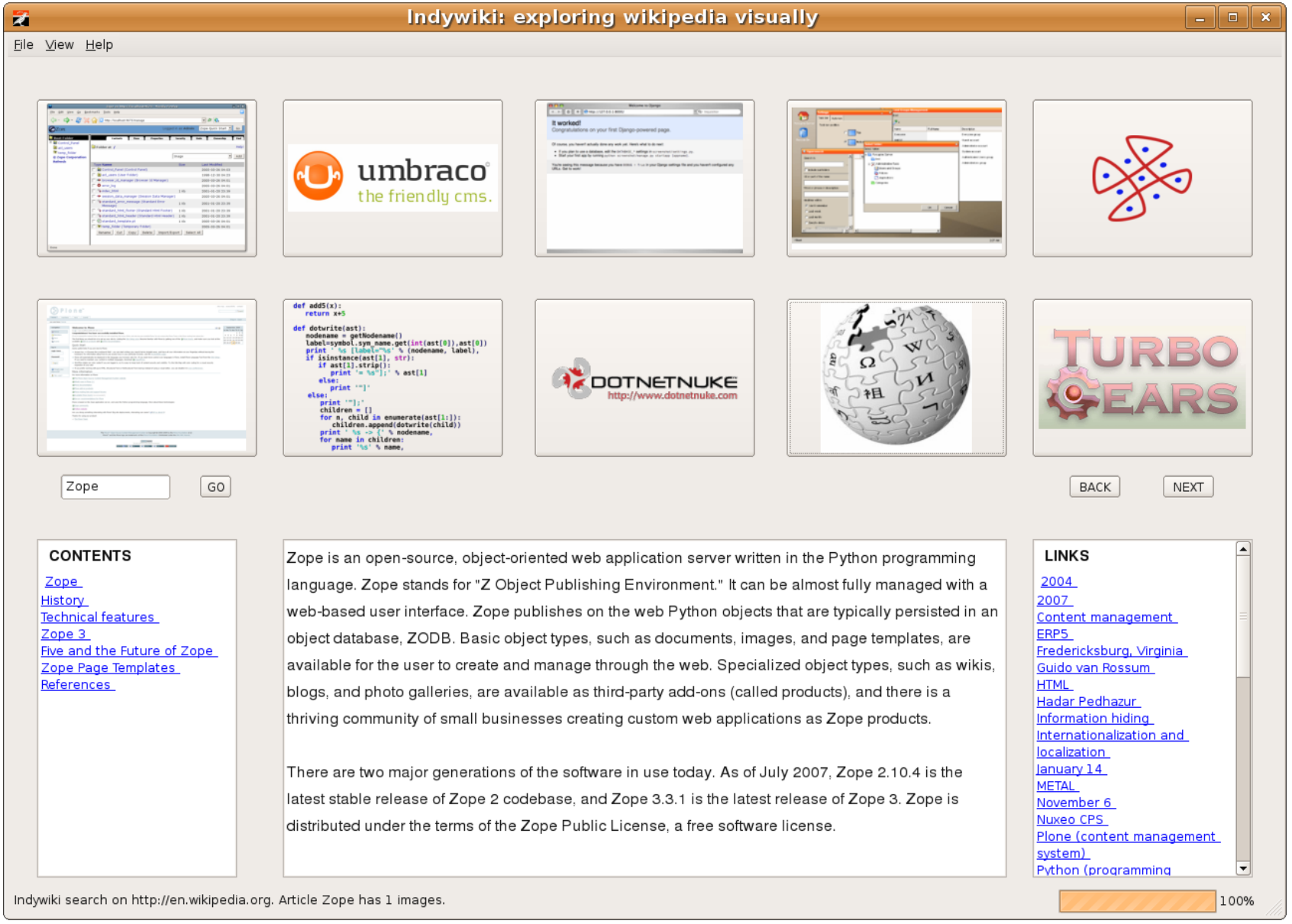
Visually browsing Wikipedia?



"user clicks on a 'native' icon"

...where native is an icon found on the article

Visually browsing Wikipedia?



user clicks on a 'non_native' image

...where non_native is the first image of a linked article!

Technical details

- Written on Python and PyQt4

- Current version: 0.9.7 (beta)
- size: < 1500 loc
- uses Nodebox library (for Wikipedia parsing)
- Packaging: py2exe for Windows (bundles python plus all libraries used by indywiki (pyqt, re, xml.dom etc) in one executable. results in ~30 mb size (10compressed)! But it works!

Technical details

- ...
- from PyQt4 import QtCore, QtGui *(PyQt4powers the gui)*
- import urllib,urllib2 *(fetch wikipedia data)*
- import threading *(python threading, for asynchronous images display etc)*
- from web1 import wikipedia *(Nodebox library is used to get most data)*
- from xml.dom import minidom *(parses the xml data, that Mediawiki api gives us)*
- ...

The GUI

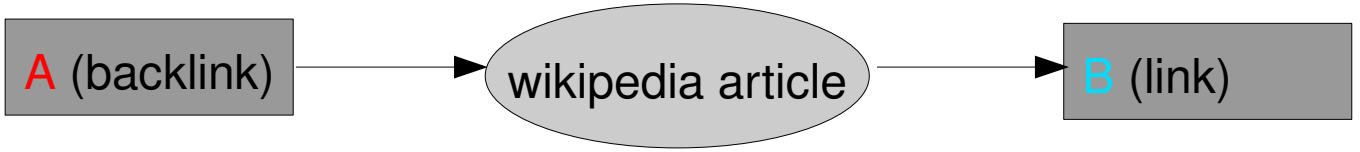
- Several classes and functions
- Asynchronous display of images (achieved through the use of python threads)
- Many emits, because of the xlib first rule ("You do not make X calls from more than one thread" - only from the main)

class Ui_MainWindow

Initiates all widgets, which are:

- centralwidget,
- 10 pushbuttons (thumbnails),
- misc include lineedits, pushbuttons, textbrowsers
- Scheduled toolbar with icons (next, back etc) to substitute all buttons with icons

Links filtering/scoring



C (belongs to category1)
D (belongs to category2)
.
.
X (belongs to category xx)

scores:
A: 2,
B:2,
{C,D...X}:1 ,
A+B:4,
A+B+C:5,
A+D:3

WE GET:
A, {C,D...X}-->Wikipedia api
B-->through Nodebox

the higher the score of a link, the bigger position on the dictionary (will be displayed earlier)

Note:The algorithm can be improved!

Icons or buttons?

(problem is this distracts the eye)

Custom widgets:Thumbnail buttons

```
appwidth = int(QtGui.QApplication.desktop().availableGeometry().width())


class MYB(QtGui.QPushButton):
  ...
    def sizeHint(self):
        if int(appwidth) == 800:
          return QtCore.QSize(140,110)

#while initiating the app:
if int(appwidth) == 800:  self.pushButton_5.setIconSize(QtCore.QSize(130,100))
```

...Continued

```
QtCore.QObject.connect(self.pushButton_5,QtCore.SIGNAL("clicked()"),          \
self.button5_clicked,QtCore.Qt.QueuedConnection)

def button5_clicked(self):
    if type(self.dicta['5']) is list:
        self.display_native_size(...)
    elif type(self.dicta['5']) is tuple:
        self.lineEdit.clear()
        self.lineEdit.insert(QtCore.QString(self.dicta['5'][0]))
        self.centralwidget.first_click()
```

Custom widgets: Text Browser

- Displays article text
- Subclass of QTextBrowser class, so that text titles of a wikipedia page link to their paragraph text

```
#displayed on textedit2
class MyTextBrowser(QtGui.QTextBrowser):
  ...
   def setSource(self, number_of_title):
      ...
       ui.textEdit_2.insertPlainText(ui.aquery.paragraphs[int(number_of_title.toString())])
```

...Continued

- links found on a wikipedia page become our search keywords

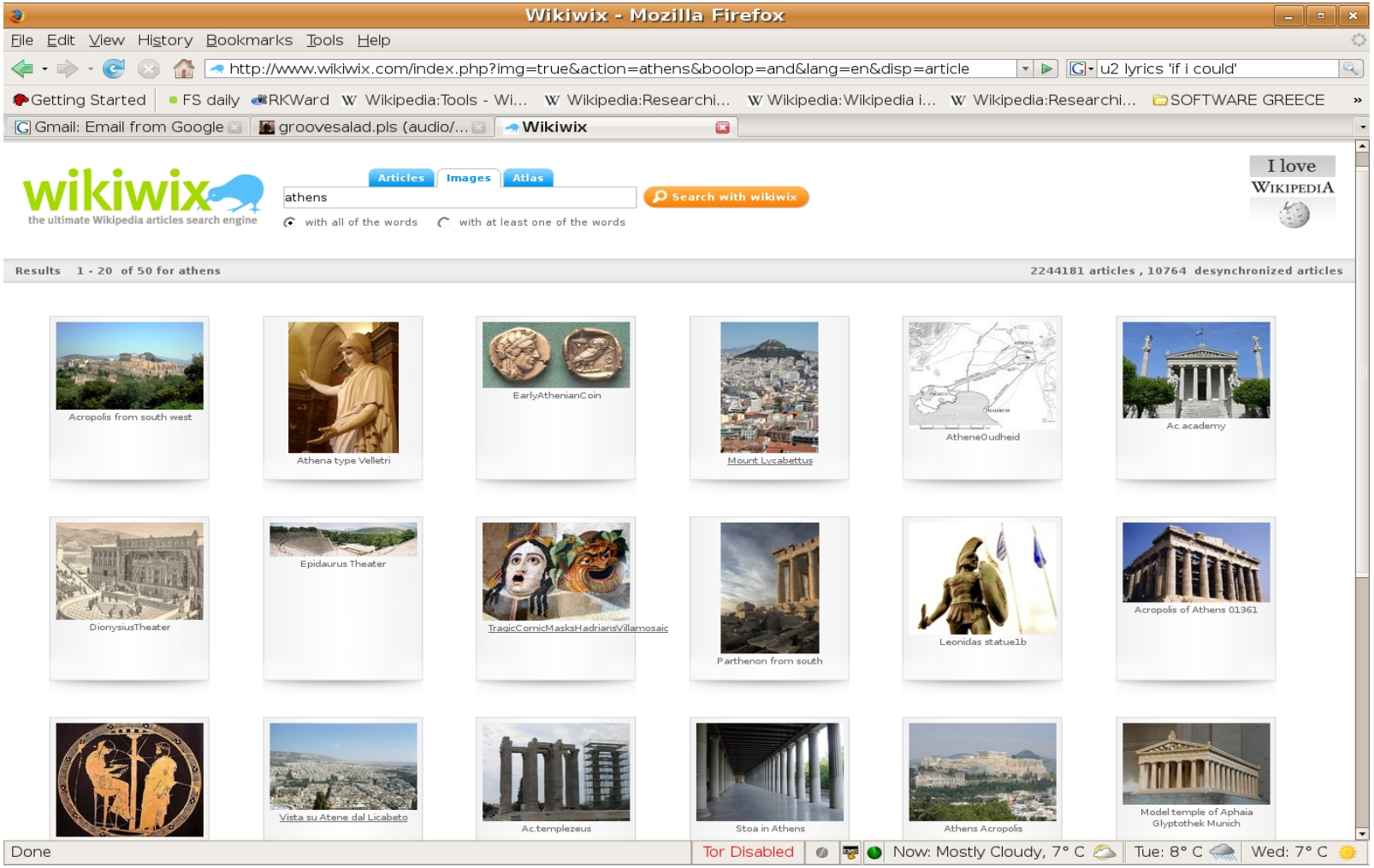
```
class MyTextBrowser2(QtGui.QTextBrowser):
 ...
   def setSource(self, title):
    ...
      ui.lineEdit.insert(QtCore.QString(title.toString()))
      ui.centralwidget.first_click()
```

Nice things to do...

- 'Port' the application for the web
- display HTML (**not as easy as it sounds**)
- improve overall aesthetics
- add caching capabilities
- get responses from multiple wikipedia sites
- add slide show capabilities
- show the relationship between the images (academic)

And for the end...
make the Wikipedia images (similar to google images)!

More on the last bullet



“ Something similar to what wikiwix is doing, but use indywiki's algorithm to score links.”

Community

- Such a nice thing!
- a few thousands downloads
- people mention indywiki on blogs, articles and send suggestions
- indywiki appears on kde-apps.org, sourceforge and freshmeat
- Still no code contribution (that's why I'm at Fosdem) :)

Thank you for your time!

Questions and Demo

http://indywiki.sf.net