BitPim

An application in Python

Roger Binns
rogerb@rogerbinns.com

Motivation

- New cell phone, new features
  - Phonebook
  - Calendar
  - Wallpapers
  - Ringtones
  - Messaging
  - Voice and Text notes ...

I am human!

- I will not use a number pad
- Sync products limited
  - 'Tickbox' phone support
  - Phonebook only
  - Windows only
  - Single machine licenses
  - Stupid quirks
- I can do way better ☺

BitPim

File Edit Data View Help

List Images Add Delete

PhoneBook | Wallpaper | Ringers | Calendar | Log

Name	Phone	Phone2
Bartholemew Howes	+1-415-555-3419 (Office)	+1-415-555-4164 (Fax)
Bøørn Hansen	+1 303 555-5555 (Office)	
Evolution Middle Dude (nickname)	bus 903218490809 (Office)	bus fax 908908098908 (Fax)
Exported Desktop	work 3248098098 (Office)	home 90809890809 (Home)
Frank Dobson	+1-919-555-9515 (Office)	+1-919-555-9564 (Fax)
Heidi Laso	301-984-8223 (Home)	301-469-4070 (Office)
Herbert Knack	011-49-6118-945-5466 (Irmgard) (Home)	011-49-166-847-1518 (handy) (Cell)
James Bond	+1-206-555-4544 (Office)	+1-206-555-7329 (Fax)
Julla Kalerrrvo Pela | Jukka Abcde Pela	+358 9 5555 0397 (Office)	+358 9 555 2675 (Home)
Keith Hodgkinson	718-575-1310 (Home)	
Mary Greene	510-649-7872 (Home)	
Meister Berger	+49 3581 123456 (Home)	+49 3581 123456 (Fax)
Moods in Hair	973-762-1070 (Office)	
Mr. Bill P. Bitpim | Bill P. Bitpim	(555) 321-1111 (Home)	(555) 321-2222 (Cell)
My Firstname My Surname	21312312312 (Fax)	123123123123 (Home)
One Another	908908908098098 (Office)	980098098098 (Fax)
Schmidlap Joseph; alias Joe	(646) 366-1234 (Office)	(973) 600-5678 (Cell)
first ; bit last bit | first bit last bit	7987987987 (Office)	987987987 (Fax)

Exported Desktop

Business
Cellphone

Office work 3248098098

Home home 90809890809

Fax fax 09890890809

Home other 90890809809

email@example.com

business
Company
street 1
street 2
street 3
street 4
street 5
city
state
zip
country

This is the note

and it can be across
multiple lines

This record was exported from palm desktop 4.1

the category is busniess

Python File Edit Data View Help (Charged) Wed 12:25 PM Peter Dufault

BitPim

PhoneBook | Wallpaper | Ringers | Calendar | Filesystem | Log | Protocol Log


albert.bmp


dorsay.bmp


isabella.bmp


sigmund.bmp


andy.bmp


fyodor.bmp


mishima.bmp


ben.bmp


goethe.bmp


selby.bmp

goethe.bmp

goethe.bmp

Delete Save ...

There will be a bunch of other buttons here letting you do stuff like convert formats, delete files, resize etc. The actions will apply to all the items selected.

Name	Size	Bytes	Origin
albert.bmp	128 x 128	49206	
andy.bmp	128 x 128	49206	
ben.bmp	98 x 98	29062	
dorsay.bmp	92 x 92	25446	
fyodor.bmp	128 x 128	49206	
goethe.bmp	128 x 128	49206	
isabella.bmp	120 x 131	47214	

Ready

BitPim

File Edit Data View Help

Add Delete

PhoneBook | Wallpaper | Ringers | Calendar | Log


clipboard1.bmp


clipboard8.bmp


garbage.bmp


pic01.jpg


shirley.bmp


clipboard10.bmp


dscn1237.bmp


ollie.bmp


pic02.jpg


clipboard5.bmp


dscn1439.bmp


pic00.jpg


plant.bci

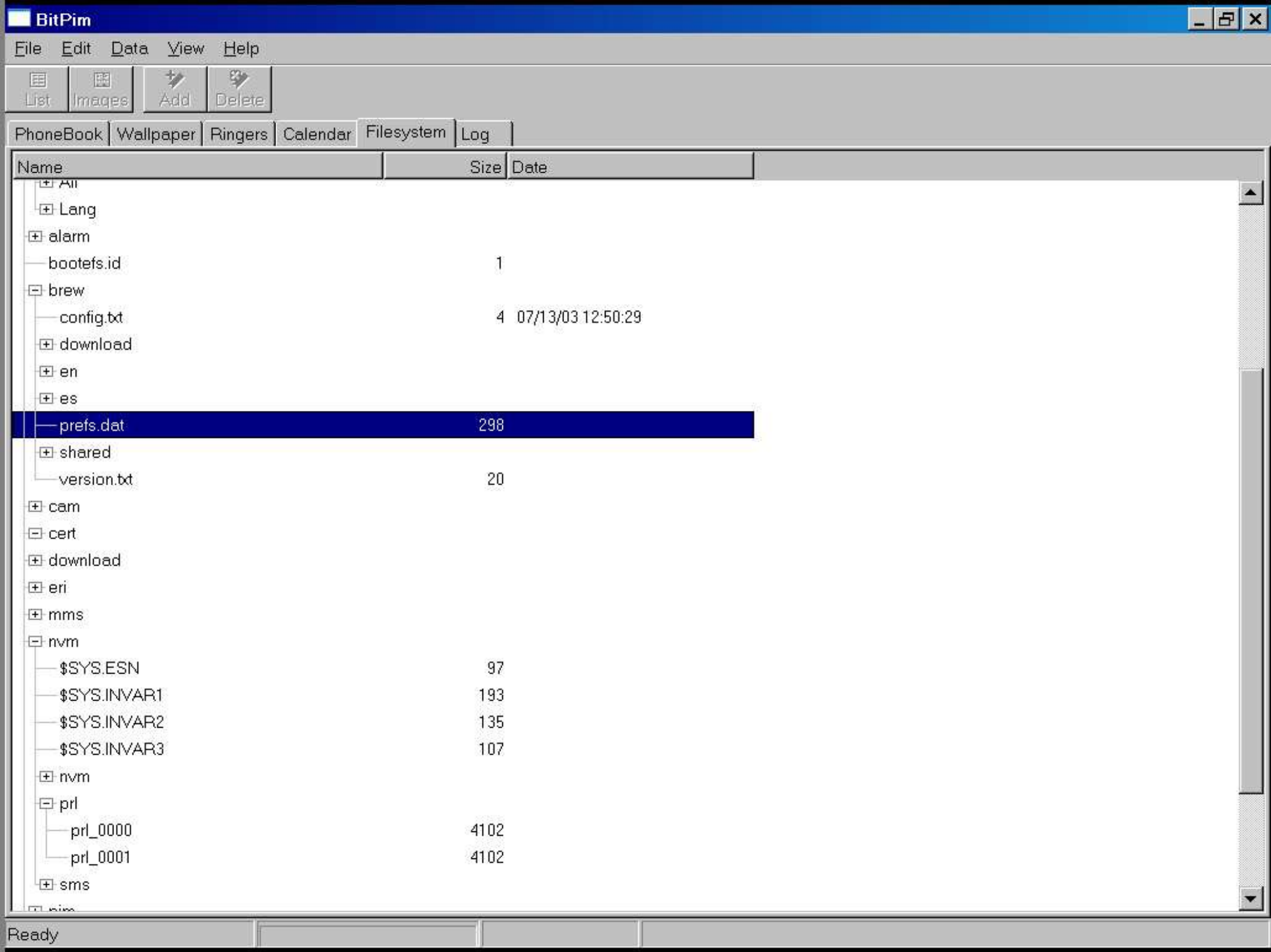
Name	Size	Bytes	Origin
clipboard5.bmp	96 × 96	27702	images
clipboard8.bmp	120 × 98	35334	images
dscn1237.bmp	120 × 98	35334	images
dscn1439.bmp	120 × 98	35334	images
garbage.bmp	120 × 111	40014	images
ollie.bmp	140 × 115	48354	images
pic00.jpg	640 × 480	31080	camera
pic01.jpg	640 × 480	16324	camera

ollie.bmp

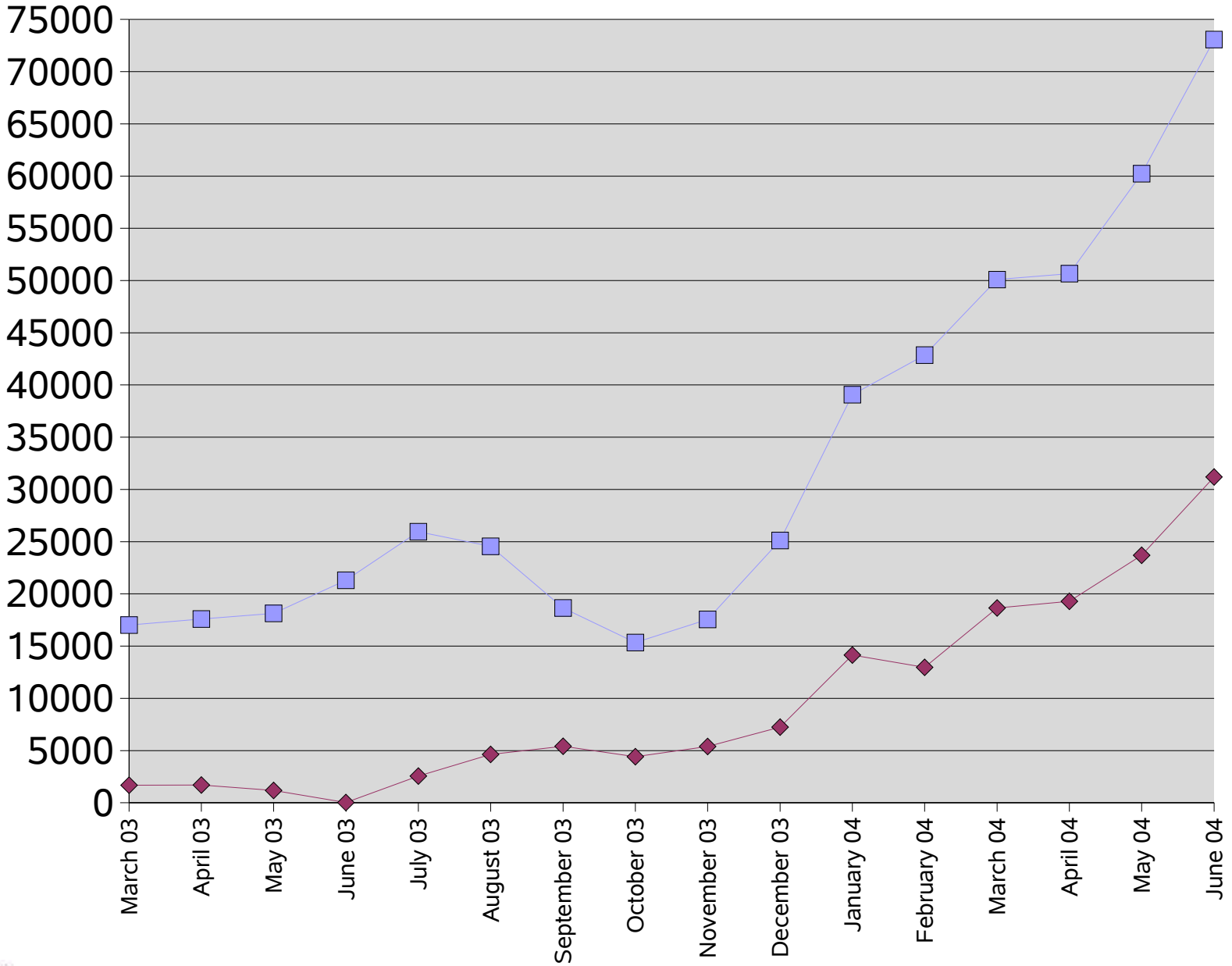
ollie.bmp

Delete Save ...

There will be a bunch of other buttons here letting you do stuff like convert formats, delete files, resize etc. The actions will apply to all the items selected.


BitPim
File Edit Data View Help
List Images Add Delete
PhoneBook Wallpaper Ringers Calendar Filesystem Log
Name Size Date
Lang
alarm
bootefs.id 1
brew
config.txt 4 07/13/03 12:50:29
download
en
es
prefs.dat 298
shared
version.txt 20
cam
cert
download
eri
mms
nvm
$SYS.ESN 97
$SYS.INVAR1 193
$SYS.INVAR2 135
$SYS.INVAR3 107
nvm
prl
prl_0000 4102
prl_0001 4102
sms
Ready

Popularity



bitpim.org

Scale

- 30,000 lines of Python
- 2,500 lines of description files
  - turns into 41,000 lines of generated code
- 500 lines of C/C++
- 20,000 words of online help

Specifications

- Expose and use all features of my cell phone
  - **No** protocol documentation at all!
- Let me use it on as many machines as I want
  - cross platform
- Interoperate with other data sources
  - No data islands

More specs

- Let others join the party
  - Open source
  - Able to add other phones
- Do not be a PIM
  - Only have UI where no other program does
- Be possible to plug into other programs
  - deal only with cell phone interfacing

Even more specs

- Easy to use
- No installation hurdles
  - No prerequisites
  - No DLL hell
  - No RPM dependencies hell
- Easy to diagnose what has happened on user machines
  - Lack of documentation means learning from the field

The users don't care

- Users **don't** care what language you use
- Users **don't** care how hard it is to write
- Users **don't** care what development methodology you use
- Users **don't** care about you being consistent with a platform they don't use

The users do care

- Users **do** care that your program does what you claim
- .. and they want to use it for as little time as possible
  - Goals, not tasks

I care

- I care about productivity
  - No drudge work thanks!
- I care about refactoring
  - Knowledge will change over time
- I care about ease of talking to other libraries & components
  - Productive!
- Active community

Preaching to the choir

The only solution is

Python!!!

(And some Python libraries)

bitpim.org

Gui options

- Tkinter ✗
  - Lowest common denominator
  - Draws widgets itself
  - Unpleasant prior experience
  - Stagnant
- Qt ✗
  - High common functionality
  - Not open source on Windows
  - Draws widgets itself

Gui options

- GTK ✗
  - Linuxy
  - Draws widgets itself
- wxWidgets/wxPython ✓
  - High common functionality
  - Uses native widgets where possible
  - Very active community
  - Pleasant prior experience

Distribution

- Has to appear as a 'normal' application on each platform
- Two steps
  - Freezing Python code (confusingly called installers)
  - Installer

Freezing

- Gather all Python modules, binary modules and Python interpretter
- And data files you specify
- Launcher stub
  - Loads Python interpretter
  - Set path for binary modules
  - Load Python code from archive (.zip)
- Can be placed anywhere – no need for installation

Freezing - Platform specific

- Windows – py2exe
  - Icons
  - version_info resource
  - COM integration/modules
- Linux – cx-Freeze
  - Strip RPATH from wxPython shared libs
- Mac – BundleBuilder
  - Some manual specification of shared libraries to include

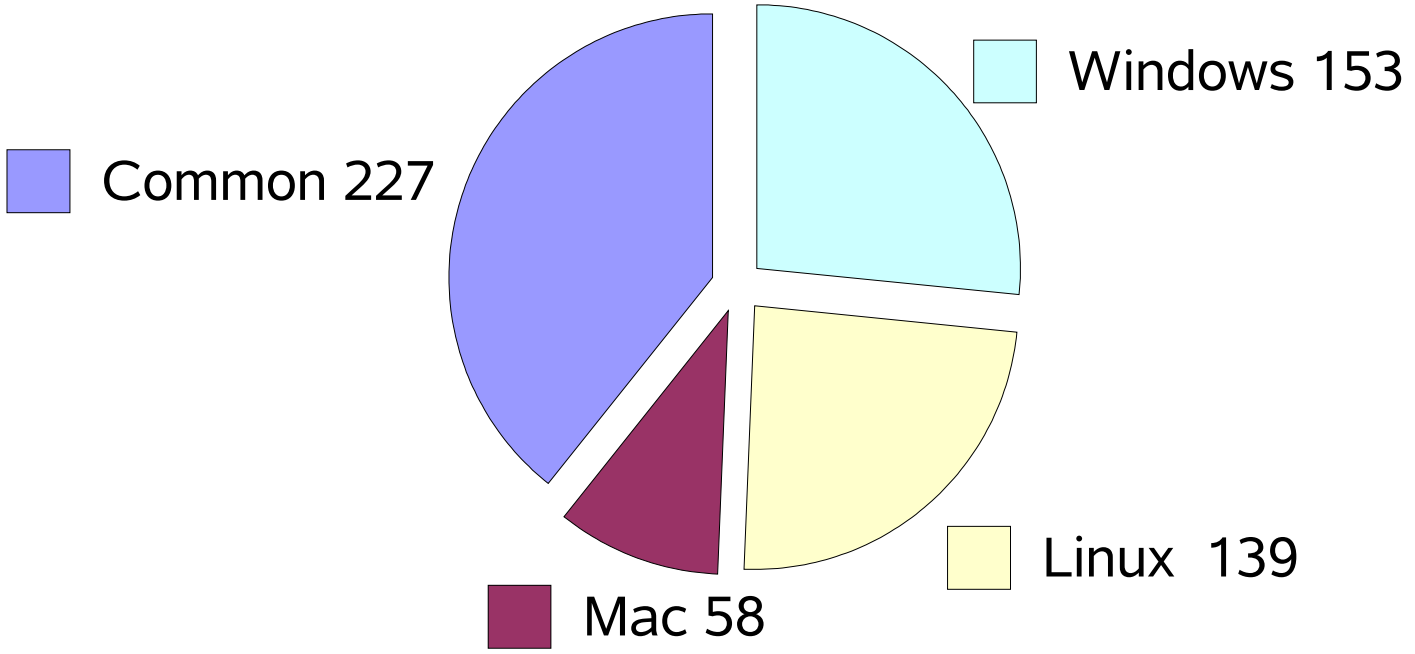
Installation – Platform specific

- Windows – InnoSetup
  - Start Menu
  - Uninstall
  - Control Panel
- Mac – dmg
  - Icon

Installation – Platform specific

- Linux – RPM
  - provides, requires (be careful of rpmbuild being too helpful!)
  - Files out of the way, launcher shell script
  - Menu icon
  - $LD_LIBRARY_PATH (cx-Freeze can do this for you)

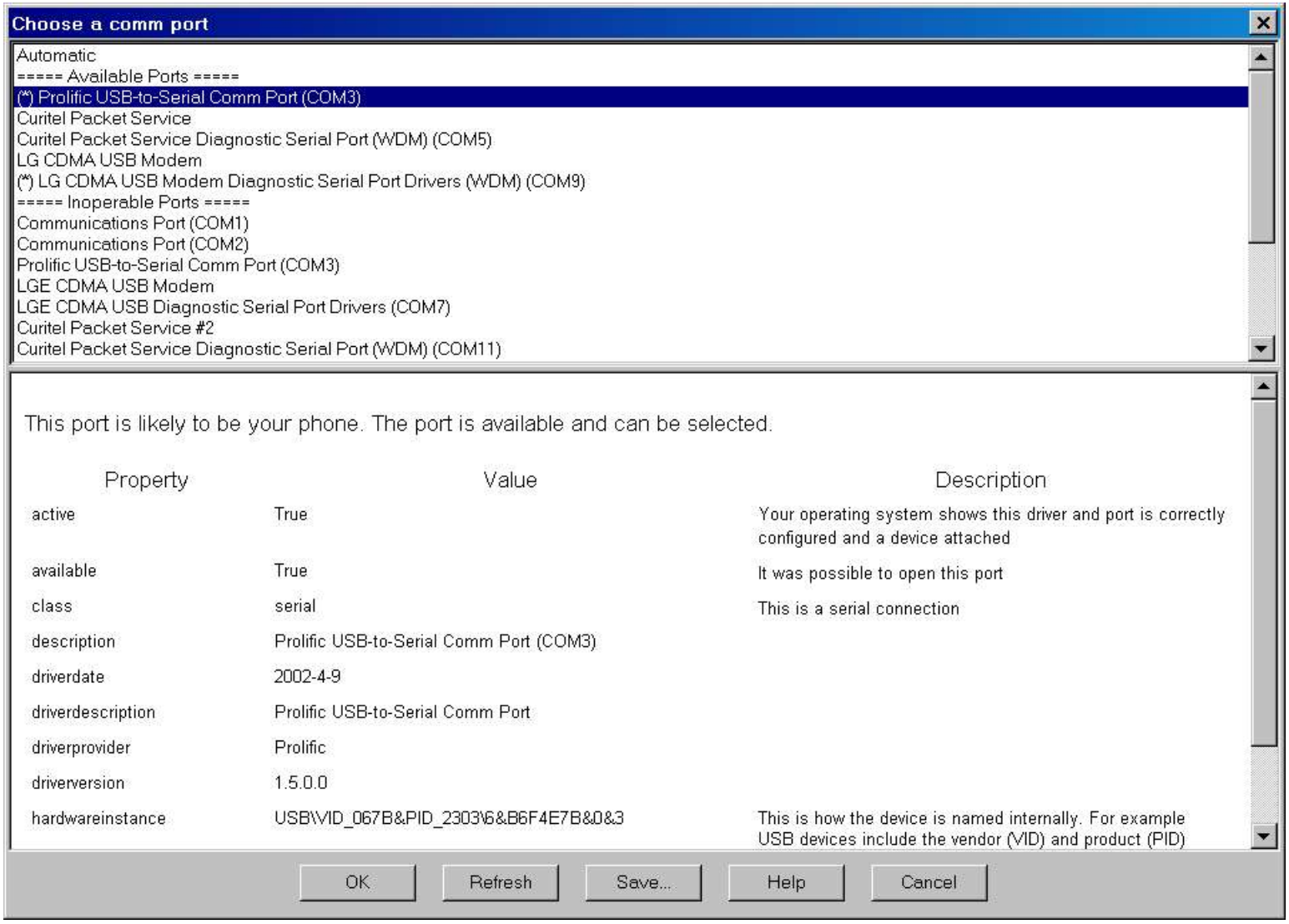
Not that hard!



Serial Ports

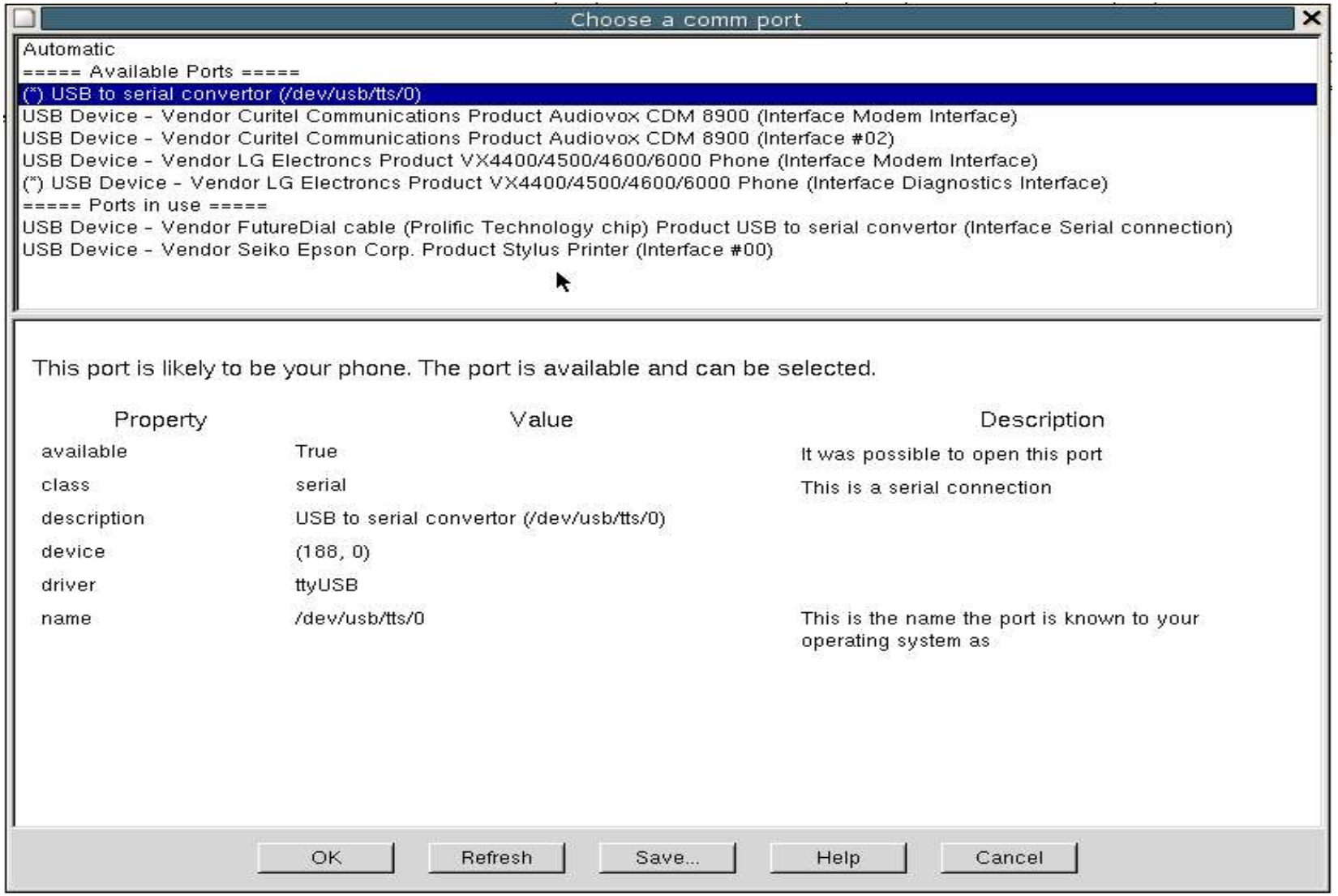
- pyserial library
  - Windows (win32all)
  - Linux & Mac (posix)
  - Interface doesn't know/care about platform
- Full functioned
  - Data rates
  - Flow control
  - DSR/CTS etc

User friendly serial ports



bitpim.org

User friendly serial ports



bitpim.org

Months of fun!

- Windows – scan registry
  - Different on Win9x, Win2k, WinXP
  - Good detail (drivers, dates etc)
- Linux
  - Device nodes in various directories
  - Kernel module associated with major
- Mac
  - Device nodes in one directory /dev
  - No other information

USB

- Aka Python/C integration is easy
- Libusb provides good USB access on Windows, Linux and Mac
- SWIG generates Python code wrapping C/C++
- Language neutral but best at Python

SWIG

- SWIG is **really** good
- Does a good job with raw C/C++ header
- Is actually simpler than it seems
- Can pick up cues from parameter names (encourage C library implementors!)
  - Binary strings

C/C++ problems highlighted

- Who owns memory?
- Global pointers
- Data owned by other data
  - May need C++ reference counter
- Poor API design
  - Provide Pythonic layer

Multi-tasking

- Can't do two things at the same time
  - Waiting for GUI event
  - Talking to device
  - Talking to network
- Two approaches
  - Event driven
  - Threading

Event driven

- Twisted/asyncore
- Scales well
- Harder to write (“inside out”)
- Every library has to work the same way

Threading

- Simpler (initially)
- Things can change underneath you
- Harder to deal with exceptions

And the winner was ...

- Threading
- Decouple threads (and data) as much as possible
- wxPython event loop integrates well
- Use Queue.Queue to send messages
- wx.PostMessage to send to GUI event queue

Sample Code - GUI

```
def OnGetFile(self):
    ...
    self.makecall(getfile, path, self.OnGetFileResult, path)

def OnGetFileResults(self, path, exception, data):
    if exception is not None: raise exception
    ...

def makecall(self, callfunc, args, resultfunc, resargs):
    self.queue.put( (callfunc, args, resultfunc, resargs) )
```

Sample Code – thread loop

```
result=exception=None
callfunc, args, resultfunc, resargs = queue.get()
try:
    result=callfunc(*args)
except:
    exception=sys.exc_info()[1]
wx.PostEvent(RESULT_EVENT,
             (resultfunc, resargs, exception, result))
```

Threading and SWIG

```
// deal with data being returned

int usb_bulk_read_wrapped(usb_dev_handle *dev, int ep, char *bytesoutbuffer,
   int *bytesoutbuffersize, int timeout)

 {

   int res;

   Py_BEGIN_ALLOW_THREADS

   res=usb_bulk_read(dev, ep, bytesoutbuffer, *bytesoutbuffersize, timeout);

   Py_END_ALLOW_THREADS

   if (res<=0)

     *bytesoutbuffersize=0;

   else

     *bytesoutbuffersize=res;

   return res;

 }
```

Threading gotchas

- Python doesn't have thread priorities
  - Nor do the event driven frameworks
- Cannot interrupt a thread
  - Have to poll variable
  - Use setDaemon for blocking calls (eg accept)
- Mismatch in producer/consumer rates can have dire effects
  - Update GUI from idle handler

Threading techniques

- Assertions
  - against `thread.get_ident()`
- Class/instance wrappers
  - Forward requests to correct thread
  - Thread pooling
  - Checking

Outlook

- If you can do it easily from VB, you can do it easily from Python (win32all)
- Dynamic module generation at runtime (cached)
- makepy can generate static module
- Binary distribution favours latter

Outlook - code

```
import outlook_com
app=outlook_com.Application()
mapi=app.GetNamespace("MAPI")
contacts=mapi.GetDefaultFolder(constants.olFolder)
for i in range(len(contacts.Items)):
    item=contacts.Items[1+i]
    print item.FullName
    print item.MobileTelephoneNumber
```

Evolution

- vCards stored in berkely db files (bsddb module)
- ebook api
  - Significant versioning issues
  - Significant dependency issues

vCards

- No mature Python vCard modules
- No data source implements vCards correctly anyway
- 530 lines for vFile parser and vCard data converter
- Correctly deals with every correct and broken vCard I could find

Wisdom

- Use the highest level language you can afford, even if you have to write it yourself
  - Intent not implementation
- Higher productivity
- Less hand written code
- Easier to tune

Protocol description

- Each field is some number of bytes
- Each field has a type
  - Marshalled type (eg lsb integer)
  - Python type (eg string)
- Some fields are conditional on the values of others

Protocol Requirements

```
* RESPONSEHEADER header
  1 UINTlsb blockcounter
  1 BOOL thereismore
  * STRING name
  IF thereismore
      4 UINTlsb filesize
      2 UINTlsb datasize
      * DATA data
```

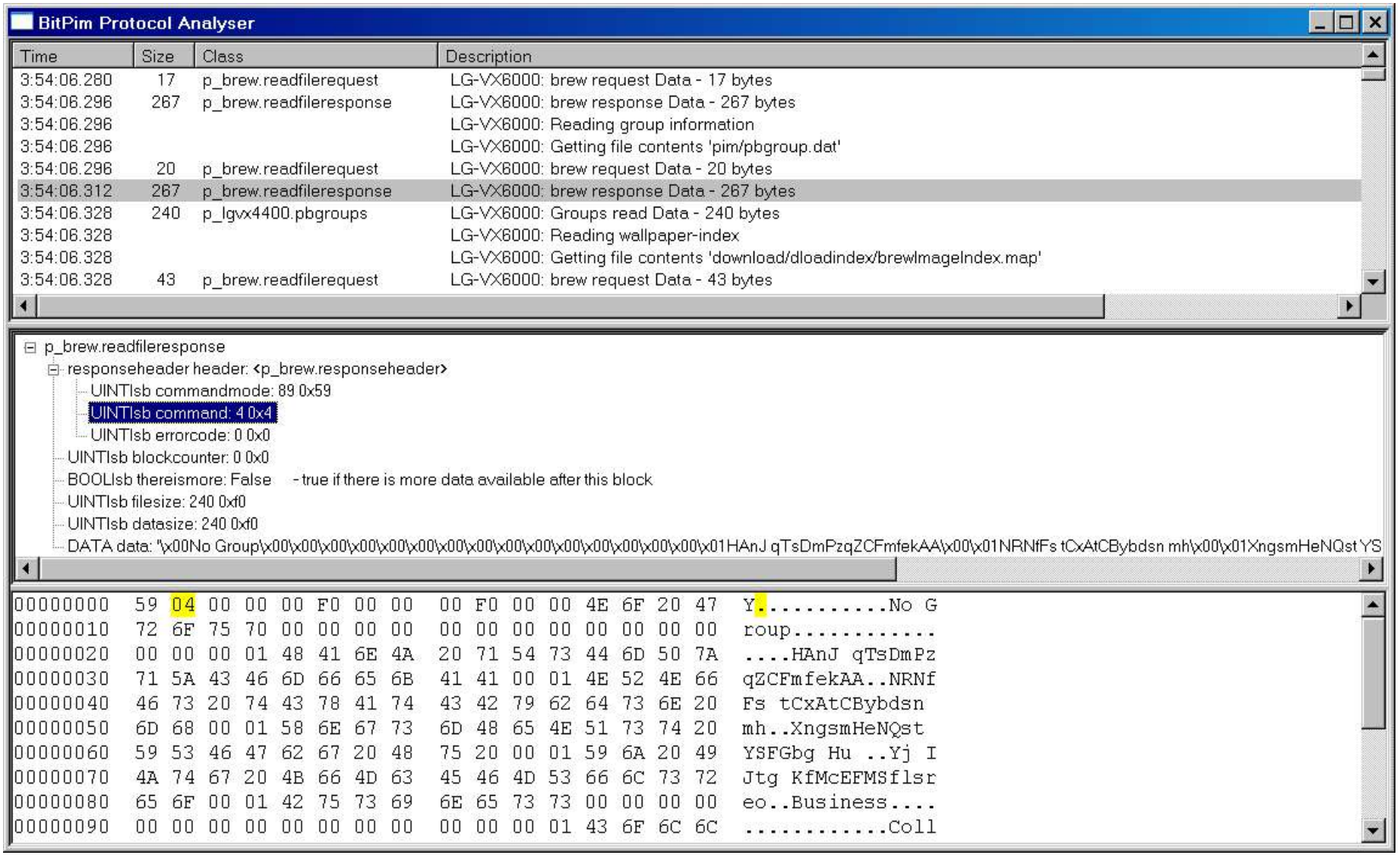
Description language

- Use Python syntax!
- tokenize module to parse
- 650 lines of code to generate Python
- 2,500 lines of protocol description
- 41,000 lines of generated code
  - with comments
  - with checking
  - with introspection

Protocol example

```
PACKET foobarrequest:
  “a comment”
  * requestheader {'command': 0x04} +header
  1 UINT {'constant': 0} +blocknumber
  if self.blocknumber==0:
    * STRING {'terminator': 0, 'pascal': True} filename
    * LIST {'elementclass': speeddial} +speeddials
```

Protocol Visualisation



Internal representation of data

- Dicts
  - Keys
  - Unique ids
- Order
  - Lists of dicts
  - Really hard to add later
- Be database ready
  - SQLite
  - Gadfly

File formats for saving data

- Human readable is great for debugging, testing and interoperability
  - XML
  - Python 'code'
- Concurrent access
- Pretty printed dicts/lists works for me
  - (Except concurrent access)

Data versioning

- Versioning
- Backwards **and** forwards compatible

```
if version==1:
    ... convert to 2 ...  version=2
if version==2:
    ... convert to 3 ...  version=3
if version==3:  ... we are happy ...
if version>=4:
    ... do something user friendly ...
```

bpmedia

Classes

MediaDisplayer

[show private | hide private]

This class does a nice 3 part display of some media type.

Top pane is icon view, bottom pane is list column view, and right pane allows actions. They are implemented using wxListCtrl(LC_ICON), wxGrid (LC_REPORT is broken on Linux and Mac) and wxHtml

Method Summary

	`__init__(self, parent, xyfile, stylefile, topsplit, bottomsplit, rightsplit)`
	`AddItem(self, adict)` Adds one item
	`AddItems(self, items)` Adds mutliple items
	`ClearSelections(self)`
	`EndSelectedFilesContext(self, context, deleteitems)` An operation has completed on the selected files.
	`GetAllItems(self)` Returns a list of dicts of all items
	`GetAllSelectedItems(self)` Returns a list of dicts of all items that are currently selected
	`GetColumnName(self, col)`
	`GetGridValue(self, row, column)` Returns the relevant field for the grid
	`GetGridValueForSorting(self, row, column)` Same as GetGridValue but returns the value that will be used for sort comparisons
	`GetIconSize(self)` return the size of the icons in the icon pane as a tuple
	`GetItem(self, which)` Returns the specific item
	`GetItemCount(self)` Returns how many items there are
	`GetItemSizedBitmap(self, item, width, height)` Return a wxBitmap for the item with specified width and height
	`GetItemValue(self, item, col)` Returns the value for the item and specified column

```
0619                print >>out, indent(i)+f[1]
0620                i+=1
0621            elif f[0]==tokens.CONDITIONALEND:
0622                i-=1
0623        print >>out, indent(2)+"raise StopIteration()\n"
0624        assert i==2
0625
0626
0627        print >>out, "\n\n"
0628
0629    def makefield(self, out, indentamount, field, args="", dictname='dict', isreading=True):
0630        if field[2]!='P' and field[2]>=0:
0631            print >>out, indent(indentamount)+dictname+"={'sizeinbytes': "+`field[2]`+"}"
0632        else:
0633            print >>out, indent(indentamount)+"%s={}" % (dictname,)
0634        if not (isreading and '*' in field[7]):
0635            for xx in 4,5:
0636                if field[xx] is not None:
0637                    print >>out, indent(indentamount)+dictname+".update("+field[xx]+")"
0638        print >>out, indent(indentamount)+"self.__field_%s=%s(%s**%s)" % (field[1], field[3], args,
0639
0640
0641 if __name__=='__main__':
0642     fn=os.path.basename(sys.argv[2])
0643     fn=os.path.splitext(fn)[0]
0644     f=open(sys.argv[1], "rt")
0645     tokens=tokenize.generate_tokens(f.readline)
0646     tt=protogentokenizer(tokens, "_gen_"+fn+"_")
0647     f2=open(sys.argv[2], "wt")
0648     cg=codegen(tt)
0649     f2.write(cg.gencode())
0650     f2.close()
0651
0652
```

Generated by PyXR 0.9.3

User documentation

- HTML is king
- Microsoft CHM
  - Proprietary archive of compiled HTML
- wxWidgets help
  - ZIP archive of HTML
- HelpBlocks
  - Produces both
  - Preprocessor

I get a Brew File Locked exception

Locked files are that way for two reasons. (Note it is the phone that has locked them, not BitPim). One is because their presence is fundamental to the operation of the phone, and if they were deleted would cause the fail to be unable to boot or operate. That covers files containing the ESN and stuff like that.

The other reason seems to be that the file is in use by a program running on the phone. There are usually all sorts of programs running such as SMS listeners, schedulers, contacts etc.

I believe there is a filesystem command to unlock files, but I haven't had the time to look into it.

So consider the whole thing a warning from your phone that they don't want you to change those files by directly overwriting them.

Can't I just offline my phone?

Taking the phone offline just means no unexpected external events will happen. For example if the phone wasn't offline and a phone call came in, the phone would have to look in your contacts to display the caller. If BitPim were busy updating the contacts then the display code may look at incomplete information and worst case crash. So the phone manufacturer just solves that problem by requiring the phone to be offlined while changes are made, and usually require a reboot afterwards since they don't want to deal with the internal programs re-reading the updated data.

Previous: When will you support my phone? **Up:** Frequently Asked Questions **Next:** Why do you do this stupid thing with my phone?

BitPim Online Help

I get a Brew File Locked exception

Locked files are that way for two reasons. (Note it is the phone that has locked them, not BitPim). One is because their presence is fundamental to the operation of the phone, and if they were deleted would cause the fail to be unable to boot or operate. That covers files containing the ESN and stuff like that.

The other reason seems to be that the file is in use by a program running on the phone. There are usually all sorts of programs running such as SMS listeners, schedulers, contacts etc.

I believe there is a filesystem command to unlock files, but I haven't had the time to look into it.

So consider the whole thing a warning from your phone that they don't want you to change those files by directly overwriting them.

Can't I just offline my phone?

Taking the phone offline just means no unexpected external events will happen. For example if the phone wasn't offline and a phone call came in, the phone would have to look in your contacts to display the caller. If BitPim were busy updating the contacts then the display code may look at incomplete information and worst case crash. So the phone manufacturer just solves that problem by requiring the phone to be offlined while changes are made, and usually require a reboot afterwards since they don't want to deal with the internal programs re-reading the updated data.

Previous: When will you support my phone? **Up:** Frequently Asked Questions **Next:** Why do you do this stupid thing with my phone?

Troubleshooting

- Capturing exceptions
  - sys.exc_info() -> type, value, traceback
- Frame variables
  - Python Cookbook
- Can transfer across threads
- Keep original traceback
  - `ex.original_exception=sys.exc_info()`

Exception

```
        __doc__ =  'Main entry point to Bitpim\n\nIt invokes BitPim in gui or commandline mode as a

Frame run in C:\projects\bitpim\gui.py at line 349
          args =  (['C:\\projects\\bitpim\\bp.py'],)
             m =  <gui.MainApp instance; proxy of C++ wxPyApp instance at _84df10_wxPyApp_p>

Frame MainLoop in C:\Python23\Lib\site-packages\wxPython\wx.py at line 1974
          self =  <gui.MainApp instance; proxy of C++ wxPyApp instance at _84df10_wxPyApp_p>

Frame MainLoop in C:\Python23\Lib\site-packages\wxPython\wx.py at line 92
        _kwargs =  {}
          self =  <gui.MainApp instance; proxy of C++ wxPyApp instance at _84df10_wxPyApp_p>
         _args =  ()

Frame OnIdle in C:\projects\bitpim\wallpaper.py at line 92
          self =  <wallpaper.WallpaperView instance; proxy of C++ wxPanel instance at _17cd788_wxP
             _ =  <wxPython.events.wxIdleEventPtr instance; proxy of C++ wxIdleEvent instance at _

Frame populatefs in C:\projects\bitpim\wallpaper.py at line 315
          self =  <wallpaper.WallpaperView instance; proxy of C++ wxPanel instance at _17cd788_wxP
          dict =  {'wallpaper-index': {1: {'origin': 'builtin', 'name': 'Beach Ball'}, 2: {'origin

Frame genericpopulatefs in C:\projects\bitpim\guiwidgets.py at line 1091
       indexkey =  'wallpaper-index'
             d =  {'wallpaper-index': {1: {'origin': 'builtin', 'name': 'Beach Ball'}, 2: {'origin
          self =  <wallpaper.WallpaperView instance; proxy of C++ wxPanel instance at _17cd788_wxP
       version =  2
          dict =  {'wallpaper-index': {1: {'origin': 'builtin', 'name': 'Beach Ball'}, 2: {'origin
           key =  'wallpapers'

Frame writeversionindexfile in C:\projects\bitpim\common.py at line 191
 currentversion =  2
          dict =  {'wallpaper-index': {1: {'origin': 'builtin', 'name': 'Beach Ball'}, 2: {'origin
      filename =  'C:\\Documents and Settings\\rogerb\\My Documents\\bitpim\\wallpaper\\index.idx'
```

OK Help

Secure remoting

- Express methods and parameters
- Types important
  - int
  - string
  - dict
  - list/tuple
- So are exceptions

Choices

- Home grown
- IDL
  - Corba
  - XDR
- XML based
  - XML-RPC
  - SOAP

XML-RPC

```
<?xml version='1.0'?>
<methodCall>
<methodName>add</methodName>
<params>
    <param>
        <value><int>1</int></value>
    </param>
    <param>
        <value><int>2</int></value>
   </param>
</params>
</methodCall>
```

XML-RPC

- Has exceptions (aka Faults)
- Interoperates with other languages
- No `None`
- Well supported in Python standard library
- HTTP transport in stdlib
  - No security (https, authentication)
  - One request per connection

Attempt 1

- http authentication
  - Every request
- m2crypto (openssl wrapper)
- Very difficult to prevent connection closes
  - See “useless destructors” thread on c.l.p
- X.509 certificates

Attempt 2

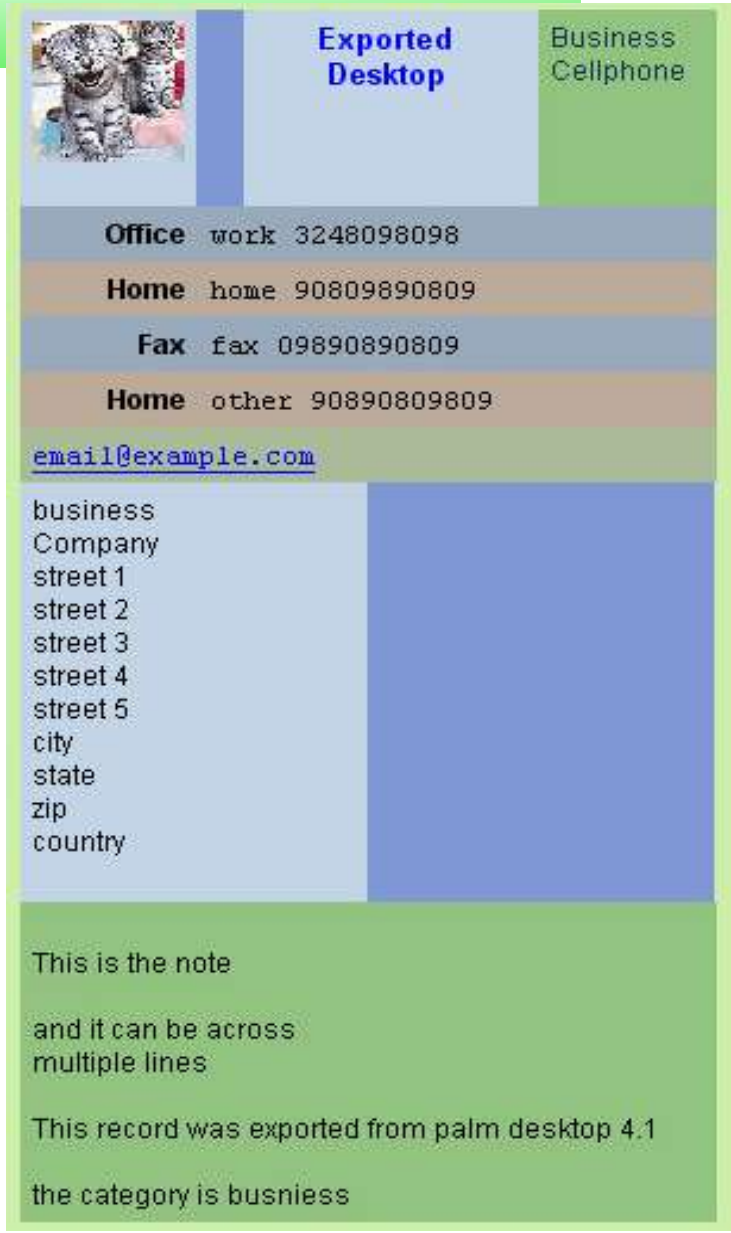
- Paramiko (SSH library)
- SSH has named channels
- Simple certificates
- Authentication on connection establishment

Auto-remoting methods

- `__getattr__` to work out name used
- `__call__` to invoke with args
- See xmlrpclib._Method
- Be careful!
  - A tale of `__str__`

Printing

- Html, the universal panacea
  - Automatic flow
  - Use templates
  - Easy preview
  - wx.Html lacks style sheets
- Can also use in UI



Publish/Subscribe

- Similar to MVC
- Decouples code
- Threading
  - Be careful
- Weakrefs

Debugging

- Print statements
  - They tend to stay a long time
- logging module
- Python debugger

```
import pdb ; pdb.set_trace()
```

- assert
- Do not throw away exceptions!
- pychecker

Do early

- Decide on undo implementation
  - Transaction log
- Decide on forwards compatibility of stored data
- Have a test plan
  - Stress/boundary
  - Normal usage/regression
- Have a position on i18n/l10n
- Use XRC for user interface

Standing on the shoulders of others

Python ⇨ language

wxPython ⇨ gui

pyserial ⇨ com ports

Python-DSV ⇨ csv files

HelpBlocks ⇨ Help authoring

SWIG ⇨ Python wrappers

GCC/MinGW ⇨ C/C++ compilers

SourceForge ⇨ Project hosting

libusb/libusb-win32 ⇨ usb access

paramiko ⇨ ssh

win32all ⇨ Windows APIs

ffmpeg ⇨ multi-media format conversion

InnoSetup ⇨ Windows installer

py2exe ⇨ Windows Python Freezing

cx-Freeze ⇨ Linux freezing

Conclusion

- Happy users & developers
- Python productivity
- Open source
- Cross platform
- Plug
  - Also try Dotamatic