クロスプラットフォーム対応のアプリケーション開発のためのライブラリ比較

箕原 辰夫[*1]
Email: minohara@cuc.ac.jp

*1：千葉商科大学政策情報学部政策情報学科

◎Key Words クロスプラットフォーム，GUI，ライブラリ

1. はじめに

近年、iPad や Android タブレットなどを含むクロスプラットフォームのアプリケーション開発のための GUI ライブラリが無償で提供されている。これらの GUI ライブラリは一長一短があり、どれを採用することがプログラミング教育において教育効果を高められるかについての議論はあまりなされていない。この論文では、これまで Python や Java などのプログラミング言語を用いてゼミナールや演習授業などで教育してきた結果を含めて、いくつかの代表的な GUI ライブラリを採り上げ、その記述方法などを含めて比較・検討する。比較対象としては、Qt, AWT/Swing, Titanium, wxWidgets, Xamarin Studio などの他に、あまり知られていない Kivy, Corona および Tcl/Tk も含む。

2. GUI ライブラリ概説

ここで取り上げた GUI ライブラリは、基本的には学生や教員がフリーで使えるものである。ただし、Titanium に関しては、無料ではあるが各自でライセンスを得る必要がある。各小節のタイトルに、主な利用言語を括弧内に記述した。「Python 等」とあるのは、Python, Ruby などのスクリプト言語で使えるような移植が存在することを示す。サンプルのプログラムは、ウィンドウを表示し、ラベルを出すまでのものとなっているが、Python への移植があるものは、Python で記述を行なった。

2.1 AWT/Swing (Java)

Oracle 社の Java 言語での標準での GUI ライブラリになっているが、Java 自体がランタイムインタープリタを含む実行環境（JRE）さえ用意すればどのプラットフォームでも実行可能であるので、比較対象に含めた。Swing[(1)]は、AWT からの発展形になっているが、Java2D も含めて、AWT のクラスとの混在が問題になっている。これは、長年 Java を教えてきたが、学生の学習時の混乱を齎す要因の一つであった。また、無名クラスをイベントのハンドラとして使った場合、プログラムの構造が崩れてしまうのも教育上の問題として挙げられる。

```
import java.awt.*;
import javax.swing.*;

@SuppressWarnings( "serial")
public class SampleFrame extends JFrame {
   public static void main( String [] arg ) {
      new SampleFrame( );
   }
   public SampleFrame( ) {
      super( "Sample Frame" );
      setSize( 400, 200 );
      setLayout( new FlowLayout( ) );
      add( new JLabel( "Sample " ) );
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      setVisible( true );
   }
}
```

2.2 Qt (C++, Python 等)

Qt[(2)]は Trolltech 社が開発した、Nokia の携帯電話を中心に用いられてきたフリーのライブラリであるが、タブレット機まで含めた幅広いプラットフォームに対応している。現在は、 Digia 社を経て、Qt-Project という団体で一元管理している。Qt4 の Python バインディングである PyQt4（Riverbank Computing 社）をこの 6 年ぐらいゼミナールで使用してきた。Qt4 が Qt5 に移行したことにより、今年から PyQt5[(3)]を使用している。なお、Mac OS X 版の PyQt5 はバイナリ形式で配布されていないので、Qt5 自体をインストールしてから、SIP というライブラリと共に Unix 上で make を用いてソースコードからコンパイルしてインストールする必要がある。これを学生に各自でやらせるのは大変ではないかと思われる。

```
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
import sys

class SampleFrame(QWidget):
   def __init__(self):
      QWidget.__init__(self, None )
      self.resize( 400, 200 )
      self.setWindowTitle( "Sample App" )
      layout = QVBoxLayout( self )
      layout.addWidget( QLabel( "Sample", self ) )

app = QApplication( [ ] )
window = SampleFrame( )
window.show( )
sys.exit( app.exec_( ) )
```

2.3 Xamarin Studio (C#, JavaScript)

Xamarin Studio[(4)]は、Xamarin 社が提供するタブレット用の GUI ライブラリと開発環境で、その基本は Microsoft 社

の.Net フレームワークになっている。統合開発環境として MonoDevelop[5]をリリースしているが、これは、Unity などの 3D 統合開発環境にも組み込まれている。また、MonoDevelop を用いて、Windows, Mac OS X, Linux 用のアプリケーション開発を行なうことも可能である。ただし、MonoDevelop 自体は C#用の良い開発環境なので、C#言語を教材にする場合には利用をお勧めしたい。

```
using System;
using Gtk;

namespace SampleFrame {
  class MainClass {
    public static void Main ( string[ ] args ) {
      Application.Init ( );
      MainWindow window = new MainWindow ( );
      window.SetDefaultSize ( 400, 200 );
      Label label = new Label ( "Sample" );
      window.Add ( label );
      window.ShowAll ( );
      Application.Run () ;
    }
  }
}
```

2.4 wxWidgets (C++, Python 等)

wxWidgets[6]はゼミナールで利用した経験はないが、プロジェクトとして開発がされている、フリーライセンス（wxWidgets ライセンス）の GUI ライブラリの一つで、Windows, Mac OS X, Linux などのクロスプラットフォームで統一的に開発が可能となっている。Python の移植版の wxPython[7]で記述してみたがシンプルな記述の仕方になっている。PyQt5 と記述の仕方が似ているので、どちらかを習得していれば移行はすぐに行なえると思われる。

```
import wx

class SampleFrame( wx.Frame ):
  def __init__( self, parent, ID, title ):
    wx.Frame.__init__( self, parent, ID, title, size=(400, 200) )
    label = wx.StaticText( self, -1, "Sample" )
    self.SetAutoLayout( True )
    self.Layout( )

app = wx.App( )
window = SampleFrame( None, -1, "Sample App" )
window.Show( )
app.MainLoop( )
```

2.5 Titanium Mobile (JavaScript)

Titanium[8]は、Appcelerator 社の開発環境およびライブラリであり、JavaScript で Web や iOS および Android のプラットフォームのアプリケーション開発を統一的に行なおうという目的を持って開発されたライブラリである。これまで、卒業研究において JavaScript ベースで利用したことがあるが、Titanium Studio は起動する際にネットワーク接続でライセンスの確認を行なうのが面倒であった。それを除けば、JavaScript ベースで授業を行なう場合は、非常に良い開発環境として推奨することができる。

```
var window = Ti.UI.createWindow({
  exitOnClose: true,
  fullscreen: false,
  title: "Sample App"
});
var label = Ti.UI.createLabel({
  text: "Sample",
  textAlign: Ti.UI.TEXT_ALIGNMENT_CENTER,
  width: Ti.UI.SIZE, height: Ti.UI.SIZE
});

window.add( label );
window.open( );
```

2.6 SDL (C, Java, Python 等)

SDL[9]は、現在はプロジェクトベースでのフリーのライセンスになっている。これもゼミナールでの利用経験はない。現在は SDL2 となっているが、Python の移植版 PySDL[10]は、ソースコードから make を用いてコンパイルする形になるので、学生には大変かも知れない。また、Windows 用では Python の移植版をインストールするためには、Cygwin や Gow などの環境を予めセットアップしておく必要がある。実際試用してみて、GUI のコンポーネントがないことがわかった。すべては、グラフィックスとしての Sprite ベースで管理されている。自分で、ボタンやテキスト入力などのコンポーネントからプログラミングしなければならない。そのため、今回の評価対象から外すことにした。

2.7 Kivy (Python)

Kivy[11]は、Python 用のクロスプラットフォームの GUI ライブラリになっている。卒業研究で 2014 年度使用してみた。感触としては、まだまだ GUI コンポーネントに問題があり、複雑なレイアウトではバグが残っていてそれを回避しなければならなかった。テーブルなどのコンポーネントはなく、自分でリスト表示から作成しなければならない。もちろん、日本語の表示も可能ではあるが、試行錯誤でようやく実現できた。これからを期待したい。

```
import  kivy
from kivy.app import *
from kivy.uix.gridlayout import *
from kivy.uix.label import *

class SampleFrame( GridLayout ):
    def __init__( self, **arg ):
        super( SampleFrame, self ).__init__( **arg )
        self.cols = 1
        self.add_widget( Label( text="Sample") )

class MyApp( App ):
    def build( self ):
        return SampleFrame( )

MyApp( ).run( )
```

2.8 Corona SDK (Lua)

Corona SDK[12]は、Corona Labs 社が開発する Lua 用のクロスプラットフォームの GUI ライブラリである。iOS と

Android に対応している。以前、ゲーム開発を主体としてゼミナールで教えたことがある。そのときの感触では、GUI コンポーネントの整備がまだまだであった。しかし、その後新しい GUI コンポーネントのライブラリが追加された。Lua 言語を教材にしたい場合は、お勧めする。

```
local  text = display.newText( "Sample App",
                100, 100, "Arial", 48 )
 text:setTextColor( 255, 255, 255 )
```

2.9 Tk (Tcl, Python 等)

Tcl/Tk[13]は、よく使われる GUI ライブラリの一つであるが、これもゼミナールでの利用経験はない。Python では標準で Tkinter[14]というモジュールに Tcl/Tk の GUI を使うためのクラスや関数が用意されている。

```
from Tkinter import *

class SampleFrame( Frame ):
    def __init__(self,parent=None ):
        Frame.__init__(self,parent )
        label = Label( self, text="Sample" )
        label.pack({"side": "left"})
        self.pack( )

window = SampleFrame( )
window.master.title( "Sample Frame" )
window.master.minsize( 400, 200 )
window.mainloop()
```

3. 比較

これらの開発環境の比較については、IEEE Computer 誌に掲載された、Ohrt[15]等の先行研究がある。その研究では、あくまでも開発者としての視点からの評価がされていた。それに対して本論文では、学生の教材として適当かどうかということ、および実際に、これらのライブラリを使って、表計算のアプリケーションを記述してみて、どの程度の記述の違いがあったかを比較する。

3.1 学生の教材としての比較のための評価軸

まず、アプリケーション開発のターゲットをどこに置くかの問題がある。Mac OS X や Windows のような PC で動作させることを前提とするのか、iOS や Android のようなスマートフォンやタブレット上の OS で動かすのかで選択肢が変わってくる。また、開発教材として使う言語の選択にも依存するだろう。なお、Python は、全米の大学での初等プログラミング教育においては、最も用いられているものとして調査結果[16]が出ている。調査は US ニューズ＆ワールド・レポート誌が選んだ、全米で最も優れたコンピューターサイエンス教育を行っている大学ランキングで上位 39 校のデータを集めて検証したもので、調査を実施した 39 校のうち、69%にあたる 27 校が Python を選択し、さらにトップ 10 校に限定すると 8 校が Python の教育を行っているという実態が明らかになっている。そのため、Python の移植があるかどうかも評価対象とすることにした。

3.2 評価結果

評価結果は、以下のような表になった。

表 1 ライブラリの評価

ライブラリ	PC	携帯	言語	Python
AWT / Swing	○	□*1	Java	-
Qt	○	○	C++	□*2
Xamarin	○	○	C#	□*3
wxWidgets	○	×	C++	○
Titanium	×	○	JavaScript	-
Kivy	○	○	Python	○
Corona	×	○	Lua	-
Tk	○	×	Tcl	○

*1 Android SDK のライブラリは、Java で開発することもあって、AWT/Swing のフレームワークと似ている。
*2 PyQt5 for Mac OS X は、ソースコードからコンパイルする必要がある。
*3 MonoDevelop 自体は、Unity4 において Python に近い言語である Boo のスクリプトの編集もサポートしていた。

ここから考えるに、Python ベースで考えれば、携帯端末を対象にしない場合は、wxWidgets が一番であると思われる。また、携帯端末のアプリケーションも含めて、開発を行なう場合は、Qt を使うのが良いだろう。どちらも記述の仕方が似ているため移行しやすいのが特徴である。Kivy を推奨できないのは、GUI のコンポーネントが未だにバグなどがあることと、記述のしやすさから考えれば、Qt や wxWidgets に一歩劣ると思われるからである。Python 以外には、C#を使うのであれば、Xamarin の MonoDevelop を、また JavaScript であれば Titanium の環境が良いだろう。教育用として Java を採用する場合は、標準の AWT/Swing 以外に、Qt を使いたいのであれば、Qt Jambi[17] (Qt for Java) がある。wxWidgets に対しては、wx4j[18] なども利用可能である。

3.3 表計算アプリケーションの記述

いくつかのライブラリで実際に表計算のアプリケーションを作成してみた。その結果について記していく。

AWT / Swing:
Swing 側に JTable というクラスがあるが、表計算ソフトウェアに必要な行番号の提示の機能がない。JTableHeader というクラスを使えば、列番号は提示できる。また、スクロールさせるには、JScrollPane と結合させる必要がある。行番号を表示させるには、左側に別の JTable のオブジェクトを配置して、その 1 列に行番号を表示させ、本体の JTable のオブジェクトと連動させる必要がある。行全体の選択や列全体の選択などは、1 つずつプログラムする必要があり、かなり力量を必要としてしまう。

Qt:
QTableWidget クラスのオブジェクトを作成してしまえば、スクロールや行番号、列番号などの基本的な操作はすべ

てライブラリ側で対応している。また、行の選択、列の選択などもすべて動作する。

wxWidgets:
wxGrid（wxPython 上では、wx.grid モジュールの Grid）クラスがテーブルに対応している。これを、BoxSizer の中に組み込むとスクロールが可能なテーブルができあがる。これも Qt と同様で、行番号、列番号、行選択、列選択などの機能は既に組み込まれている。

Xamarin Mono:
Table クラスが用意されているが、これはテーブル状にウィジェットを配置するレイアウトの役割をもっているだけに過ぎない。実際の表計算ソフトウェアの表として機能させるには、各要素のウィジェットを配置して、行番号表示や、列番号表示、行選択、列選択などは自分でプログラミングしなければならない。また、スクロールさせるには、ScrollWindow クラスと組み合わせる必要がある。

Titanium:
Titanium.UI.TableView は、テーブル表示ではなく、リストビューになっている。ListView も用意されている根本的な違いはない。そのため、２次元の表を作るためには、ビューで分割しながらの、膨大な量のコードを書く必要がある。２次元のビューあるいは、表計算の表を直接表示できるようなクラスの実装が待たれる。

Kivy:
テーブルの表示の機能がなく、kivy.uix.listview モジュールにある ListView からすべて作らなければならない。これを機能させるのは、膨大な量のコードを書かなければならない。現状では、作るのが大変なので、TableView クラスの実装が待たれる。

Corona SDK:
後から追加された widget のライブラリの中に、TableView のクラスが用意されているが、これはリストビューであり、Titanium の TableView や Kivy の ListView と変わらない。そのため、これを複数配置することになるが、現状では Kivy と同じく作るのが大変なので、複数の列が扱えるような本格的な TableView クラスの実装が待たれる。

以上の結果を踏まえると、Ready-made で表計算コンポーネントとしても動く QTableWidget クラスを持っている Qt が一番プログラミングする量が少ないと考えられる。また、それに続くのは wxGrid を持っている wxWidgets であり、Qt を使うのと同様にプログラムを短くすることができる。Swing の JTable は、時代的に先行したこともあり、未だに中途半端な出来になっている。Mono はテーブル上のレイアウトなので、これも表計算コンポーネントに仕立て上げるのはかなりの苦労を要するし、Titanium や Kivy あるいは Corona に至ってはリストのレイアウトしかない。

4. おわりに

今後、GUI を伴うアプリケーションを作るまでがプログラミング教育の目標とされるのではないかと思われる。学生が Python などの使い慣れたわかりやすい言語で、タブレットやスマートフォンなどのアプリケーションを開発ができるようになれば、プログラマの人口が増えていくのではないかと思われる。２節でプログラムのサンプルを挙げたが、GUI のプログラミングは言語やライブラリの差はあれ、ある程度記述の仕方が共通している。Qt や wxWidgets のようなよく用いられる典型的なライブラリで一度学習してしまえば、別のライブラリでもある程度類推が効く。大学あるいは高校で、学生が自分自身のアプリケーションを開発してリリースした経験があれば、社会に出てどのような環境に行こうとも、ある程度の責任のあるソフトウェア開発に携わることができるのではないだろうか。

参考文献

(1) Oracle,“Java SE 8 API,” http://docs.oracle.com/javase/jp/8/docs/api（2015）.
(2) Qt-Project,“Qt,”http://www.qt.io/（2015）.
(3) Riverbank Computing, “PyQt5”, http://riverbankcomputing.co.uk/（2015）.
(4) Xamarin,“Xamarin Platform,”http://xamarin.com/platform（2015）.
(5) MonoDevelop Project, “MonoDevelop,” http://www.monodevelop.com/（2015）.
(6) wxWidgets,“wxWidgets,”https://www.wxwidgets.org/（2015）.
(7) wxPython, “wxPython,” http://www.wxpython.org/（2015）.
(8) Appcelerator, “Appcelerator Studio and Titanium SDK,” http://www.appcelerator.com/product/（2015）.
(9) SDL Project,“SDL,”http://www.libsdl.org/ （2015）.
(10) Marcus von Appen, “PySDL,” http://pysdl2.readthedocs.org/en/latest/（2015）.
(11) Kivy Organization, “Kivy,” http://kivy.org/#home（2015）.
(12) Corona Labs, “Corona SDK,” https://coronalabs.com/products/corona-sdk/（2015）.
(13) Tcl/Tk, “Tcl/Tk,” https://www.tcl.tk/（2015）.
(14) John W. Shipman, “Tkinter 8.5 reference: a GUI for Python,” http://infohost.nmt.edu/tcc/help/pubs/tkinter/web/index.html（2013）.
(15) Julian Ohrt,Volker Turau, “Cross-Platform Development Tools for Smartphone Applicationss,” IEEE Computer, Issue Date: 2012-09, pp.72-79（2012）.
(16) Philip Guo, “Python is Now the Most Popular Introductory Teaching Language at Top U.S. Universities, ” Blog@CACM, Communications of the ACM, http://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-us-universities/fulltext（2014）.
(17) Qt Jambi Team, “Qt Jambi – Qt for Java,” http://qtjambi.org/ (2015).
(18) wx4j SourceForge project , “wx4j: A Java Binding for wxWidgets,” http://wx4j.sourceforge.net/ (2004).