diversity-radar introduction

Erin Yueh @ openmoko

http://projects.openmoko.org/projects/diversity/

Agenda

- Setup & Install python-efl and python-etk
- How to use python setup tools to create a new project
- How to interact with diversity-daemon
- A scanning line animator
- Set a timer to display flashing effects
- Demo

Setup & Install python-efl & python-etk

- use 'get_e.sh' shell script to install EFL http://www.rasterman.com/files/get_e.sh
- use 'build_all.sh' to install python-efl
- git://staff.get-e.org/users/cmarcelo/python-etk.git
- If e object is out of date, use 'cvs update -Pd', then make e objects again

Distributing Python Modules I

* tutorial: http://docs.python.org/dist/dist.html
* use Distutils to make python modules
* a 'setup.py' example

```
from setuptools import setup, find_packages, Extension
dist = setup(name='diversity-radar',
    version='0.04',
    author='Erin Yueh',
    package_dir={'':'src'},
    packages=['radar'],
    scripts=['src/radar_ui.py'],
    data_files=[
    ('diversity-radar/image',
     ['data/image/gtk-add.png',
'data/image/gtk-
preferences.png','data/image/cc_blue.png']),
('diversity-radar', ['data/theme/swallow.edj']),
('applications', ['data/diversity-radar.desktop'])]
)
```

Distributing Python Modules II

```
# python setup.py --help
Common commands: (see '--help-commands' for more)
  setup.py build      will build the package underneath
  'build/'
  setup.py install    will install the package

# python setup.py build
# python setup.py install --prefix=/usr
# python setup.py install_data --install-dir=/usr/share
```

Between Diversity-daemon and Diversity-radar

* use dbus to get the interactions with diversity-daemon
* python-dbus vs. python-edbus
* tutorial:
http://dbus.freedesktop.org/doc/dbus-python/doc/tutoria

```
import dbus
dbus.mainloop.glib.DBusGMainLoop(set_as_default=True)
bus = dbus.SessionBus()

world = dbus.Interface(
bus.get_object('org.openmoko.Diversity',
 '/org/openmoko/Diversity/world'),
 'org.openmoko.Diversity.World')

bard_path = world.GetSelf()

# register a callback for signal GeometryChanged
object_bard.connect_to_signal("GeometryChanged",
GeometryChanged_cb,
dbus_interface="org.openmoko.Diversity.Object")
```

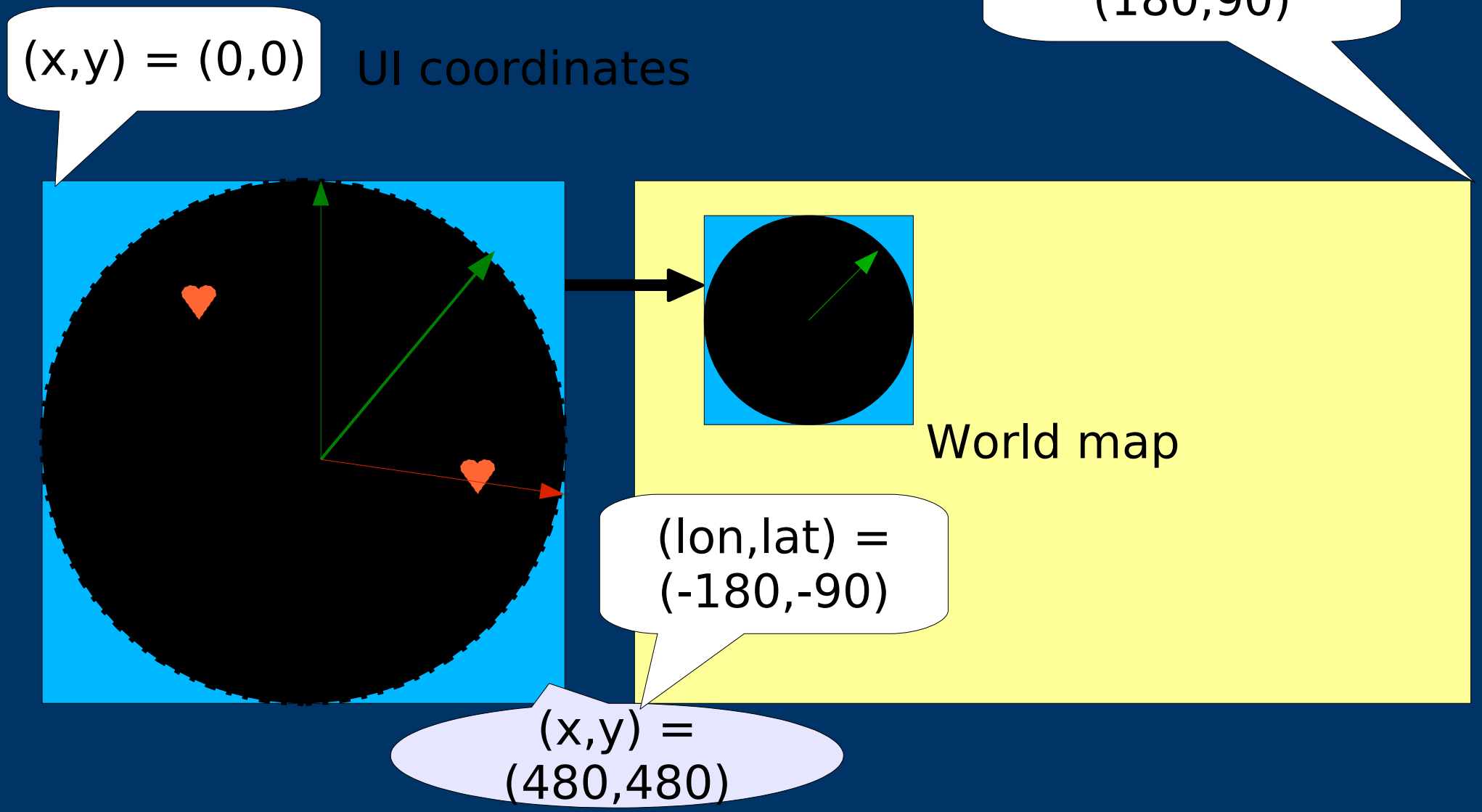
Build the radar game

World:
* GetSelf(): get NEOME self object and then can get its geometry location
* ViewportAdd(x1,y1,x2,y2) eg. (-90.0,-90.0,180.0,180.0)
* TagAdd(x,y,description) eg. TagAdd(-40.1, -30.0, "Jeremy")

Viewport:
* ObjectAdded signal: set a callback to create a target object
* ObjectRemoved signal: set a callback to remove a target object
* Start(): start viewport service and then can listen signals

Object:
* GetType(): get the object type
* GeometrySet(lon,lat,w,h): set target real location in map
* GeometryGet()

Build the radar map



A Scanning Line Animator I

- UI coordinates vs. real world map (radius)
  ca10bjectGeometryToCoord(self,lon,lat,bg,neome_lon, neome_lat,r)
- (lon-neome_lon)/2r = (x-center_x)/w
- x = center_x + (((lon-neome_lon)/(2*r))*w)
- y = center_y - (((lat-neome_lat)/(2*r))*h)
- Create a Line object in canvas
- line_degree = line.data["degree"] +
- (ee.data["speed"] * (ecore.time_get() - line.data["last_time"]))
- x = center_x + w* 0.5 * math.cos(line_degree-(math.pi*0.5))
- y = center_y + h * 0.5 * math.sin(line_degree-(math.pi*0.5))
- line.xy_set (center_x, center_y, int(x), int(y))

A Scanning Line Animator II

- calculate the degree between center and object
- getObjectDegree(self,x0,y0,x,y):
- degree = - (math.atan((x - x0) / (y — y0)))
- obj_degree = target.get_degree(name)
- if(math.fabs(obj_degree - line_degree) < 0.06):
- obj.geometry_set(int(x2),int(y2),int(w1),int(h1))
- # change tag description position
- text = obj.data["text"]
- text.pos_set(x2+5,y2+5)
- my_flash(obj)

A timer to display flashing image I

```
def my_flash(obj):
        if(obj.data["flash_timer"]):
      obj.color_set(255,255,255,255)
          return
        obj.color_set(255,255,255,255) // set it bright
        obj.show()
        text = obj.data["text"]
        text.color_set(200,180,180,255)
        text.show()
        # run flash every 0.05 sec
        flash_timer = ecore.timer_add(0.05, flash,obj)
        obj.data["flash_timer"] = flash_timer
      obj.data["flag"] = 0 // set it to decrease alpha
        return True
```

A timer to display flashing image II

```
def flash(obj):
        (r,g,b,alpha) = obj.color_get()
        text = obj.data["text"]
        if(obj.data["flag"] ==1): # fade in
                alpha+= 10
        elif(obj.data["flag"] ==0): # fade out
                 alpha-= 10
        if(alpha>=255):
                obj.data["flag"] = 0
        elif(alpha<=0):
                obj.hide()
                text.color_set(180,20,100,255)
                obj.data["flash_timer"] = None
                return False
        obj.color_set(alpha,alpha,alpha,alpha)
        return True
```