sugar labs

A member project of the Software Freedom Conservancy

20/01/10 LCA Wellington Walter Bender
Part I:
1. What is Sugar?
2. Why should you care?
3. What can you do?
1. Sugar is the core component of a worldwide effort to provide every child with an opportunity for a quality education.
2. Sugar is a critical-thinking tool in the context of open-ended exploration and discovery, going beyond the use of the computer as a tool of instruction.
3.1 Try Sugar
3.2 Join our community

sugarlabs

http://sugarlabs.org
3.3 Contribute to Sugar
Part II: The Anatomy of a Sugar Activity
The “number” game
# Copyright (c) 2009, Walter Bender

# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published
# by the Free Software Foundation; either version 3 of the License,
# or (at your option) any later version.

# You should have received a copy of the GNU Lesser General Public
# License along with this library; if not, write to the Free Software
# Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

import pygtk
pygtk.require('2.0')
import gtk
import gobject
import sugar
from sugar.activity import activity

try:
    from sugar.bundle.activitybundle import ActivityBundle
    from sugar.activity.widgets import ActivityToolbarButton
    from sugar.activity.widgets import StopButton
    from sugar.graphics.toolbarbox import ToolbarBox
    from sugar.graphics.toolbarbox import ToolbarButton

What are all these files?

card.py – defines an individual card
gencards.py – svg image generator for cards
sprites.py – draws images on the screen
deck.py – defines a deck of cards
grid.py – defines a playing surface
game.py – game logic
VisualMatchActivity.py – Sugar UI (toolbars, journal, sharing)
What is in the files?

card.py: class Card:
deck.py: class Deck:
game.py: class Game:
game.py: class Combination:
grid.py: class Grid:
sprites.py: class Sprites:
sprites.py: class Sprite:
VisualMatchActivity.py: class ChatTube(ExportedGObject):
def __init__(self, sprites, card_type):
def shuffle(self):
def deal_next_card(self):
def cards_remaining(self):
Let's add Mayan to the Number Game.

![Mayan Numbers](http://en.wikipedia.org/wiki/File:Maya.svg)
Some Python, an icon, ...

def number_mayan(n, stroke):
    x = 42.5
    x1,x2,xc,x3,x4 = x+5,x+15,x+20,x+25,x+35
    y = 60
    y1s,y5s,y10s,y20s = y,y-10,y-20,y-40
    if n == 5:
        svg_string = svg_bar(x,y1s,stroke)
    elif n == 7:
        svg_string = svg_bar(x,y1s,stroke)
        svg_string += svg_circle(x2,y5s,3,stroke,stroke,2)
        svg_string += svg_circle(x3,y5s,3,stroke,stroke,2)
    elif n == 10:
        svg_string = svg_bar(x,y1s,stroke)
        svg_string += svg_bar(x,y5s,stroke)
    elif n == 11:
        svg_string = svg_bar(x,y1s,stroke)
        svg_string += svg_bar(x,y5s,stroke)
        svg_string += svg_circle(x+20,y10s,3,stroke,stroke,2)
    elif n == 14:
        svg_string = svg_bar(x,y1s,stroke)
        svg_string += svg_bar(x,y5s,stroke)
        svg_string += svg_circle(x1,y10s,3,stroke,stroke,2)
        svg_string += svg_circle(x2,y10s,3,stroke,stroke,2)
        svg_string += svg_circle(x3,y10s,3,stroke,stroke,2)
        svg_string += svg_circle(x4,y10s,3,stroke,stroke,2)
    elif n == 15:
        svg_string = svg_bar(x,y1s,stroke)
        svg_string += svg_bar(x,y5s,stroke)
        svg_string += svg_bar(x,y10s,stroke)
    elif n == 21:
        svg_string = svg_circle(xc,y1s,3,stroke,stroke,2)
        svg_string += svg_circle(xc,y20s,3,stroke,stroke,2)
    elif n == 22:
        svg_string = svg_circle(x2,y1s,3,stroke,stroke,2)
        svg_string += svg_circle(x3,y1s,3,stroke,stroke,2)
        svg_string += svg_circle(xc,y20s,3,stroke,stroke,2)
    elif n == 33:
        svg_string = svg_bar(x,y1s,stroke)
        svg_string += svg_bar(x,y5s,stroke)
        svg_string += svg_circle(xc,y10s,3,stroke,stroke,2)
        svg_string += svg_circle(xc,y20s,3,stroke,stroke,2)
    return svg_string
et voila!
walter@sugarlabs.org