

# WIDGETS

CSSE 120—Rose Hulman Institute of Technology

# Update from your SVN repository

- Right-click your NestedLoopPatterns project, choose
  - Team → Update
- Read grader feedback (if any) for your NestedLoopPatterns program
  - The feedback should all begin with CONSIDER

# Viewing Feedback and Tasks

- Turn on Task view in Eclipse
  - Window → Show View Tasks
- Add CONSIDER tag
  - Window → Preferences → Pydev → Task Tags
  - Add CONSIDER: to end of list

# Widgets

- "Widget" is a generic name for a component of a Graphical User interface (such as a button, window, menu, or textbox).
- Our simple Python graphics module only provides one kind of widget.
- We can create our own.
- Create a button widget.

# Button Widget Class

- Button object:
  - ▣ Displayed as text inside a rectangle.
  - ▣ What are its operations (methods)?
  - ▣ What information does a Button object need to know about itself (instance variables)?
  - ▣ Before answering these questions, let's run a demo.

# Constructor for Button class

```
class Button:
```

```
    """ A Button widget similar to the one in Zelle 10.6... """
```

```
    FONT_NAME = 'courier'
```

```
    FONT_SIZE = 20
```

class variables (used as constants here)

```
def __init__(self, win, label, center, height=0, width=0):
```

```
    """ Initialize a Button with the given """
```

```
    self.center = center
```

```
    self.label = label
```

put complex tasks into separate methods, to keep constructor simple

```
    self.__setRectDimensions(height, width)
```

```
    self.__setRect()
```

```
    self.__setText()
```

```
    self.enable()
```

methods (like `__setRect`) whose names begin with two underscores (and do not also end with two underscores) are intended to be used as "private" helper methods, only called from other methods from the same class.

# Private methods used by `__init__()`

```
def __setRect(self) :  
    """Internal method. Called by the constructor.  
    Create the rectangular border of the button. """  
    self.rect = Rectangle(Point(self.minX, self.minY),  
                           Point(self.maxX, self.maxY))  
  
def __setText(self) :  
    """Internal method. Called by the constructor.  
    Create the Text object for the button's label"""  
    self.text = Text(self.center, self.label)  
    self.text.setFace(Button.FONT_NAME)  
    self.text.setSize(Button.FONT_SIZE)
```

Note the use  
of the class  
variables

# Build a Better Button?

## □ The problem

- If width and/or height passed to the constructor are too small, the button's label may extend outside the rectangle that is supposed to bound it.
- User has to rely on trial and error to get a reasonable button width.

## □ Solution

- A better Button constructor that will create a "barely large enough" rectangle if the given width and height are too small.

# Constraints and research

- Fixed-width font (Courier). Why?
- Fixed font size (I chose 20 point).
- Experiments indicated that each character is about 16 pixels wide and about 34 pixels tall.
- Thus these class variables:

```
MIN_HEIGHT = 38 # A button should be at least this tall,  
                # in order to comfortably surround the text.  
FONT_NAME = 'courier'  
FONT_SIZE = 20  
CHAR_WIDTH = 16 # The width of a character (in pixels) of  
                # the chosen font face/size.  
EXTRA_HORZ_SPACE = 6 # Put in this much extra horizontal  
                    # space so that the label doesn't run  
                    # into the end of the rectangle.
```

# Enhance the Button Class

- From your SVN repository, checkout the ButtonWidget project.
- In Button.py, add code to guarantee that a button's rectangle will be large enough to enclose its text.
- Then go to ColorButtons.py.
  - ▣ Study the code that is there (especially the event loop)  
Add code to create the buttonList.
  - ▣ Add the **re-enable all** button and make it work.
- Commit the new version to your repository.

**Look for the TODO tags in the Task view in Eclipse!**



red

blue

tomato

DarkOliveGreen4

SteelBlue2

MistyRose3

violet

orange

gold

cyan

Quit

For info on color names, do a  
Google search for TK COLORS