

CSE 91: Programming Languages Homework

September 28, 2009

To do this homework, you will need to use one of the following web browsers:

- Firefox
- Safari
- Google Chrome

Load the following drawing canvas web page in one of the browsers listed above:
<http://www.cs.ucsd.edu/~prondon/canvas.html>

Tip: This homework will be much easier if you read through it before attempting the exercises.

1 Building Cities With Abstractions

Programming languages *abstract* the details of what the computer does, letting you focus on the task at hand, whether it's writing, drawing, or making music. To further help you concentrate on your goal, a good programming language will allow you to make your own abstractions. In this homework, we'll explore the power of abstraction through an example which draws a city composed of several buildings, each of which is composed of several floors.

With abstraction, you can define a term once and use it many times and in many different ways. This saves you effort in two ways. First, by defining and using a term rather than writing its definition everywhere, you avoid mistakes and unnecessary repetition. Second, by defining and using a term rather than just writing its definition everywhere, it's easy to make a change that applies to all uses of the term just by changing its definition, which would be much harder if you had written it out each time.

Let's experiment with abstraction by drawing a picture of a building. Conveniently, we already have a mental abstraction for a building: a collection of floors. Let's base our picture on this abstraction. To do this, first we'll define what it means to draw a floor. Our floor will be a simple rectangle with two blue windows in it:

```
begin floor
  square
```

```

square color = blue, x = -25, size = 20
square color = blue, x = 25, size = 20
end

```

Now we can add a line to draw the floor:

```

floor

```

Your whole program should now read:

```

begin floor
  square
  square color = blue, x = -25, size = 20
  square color = blue, x = 25, size = 20
end
floor

```

Now click “Run”. You should see the program draw the shape, like in Figure 1. Try changing the “floor” command at the end so that it changes the size and position of the floor.

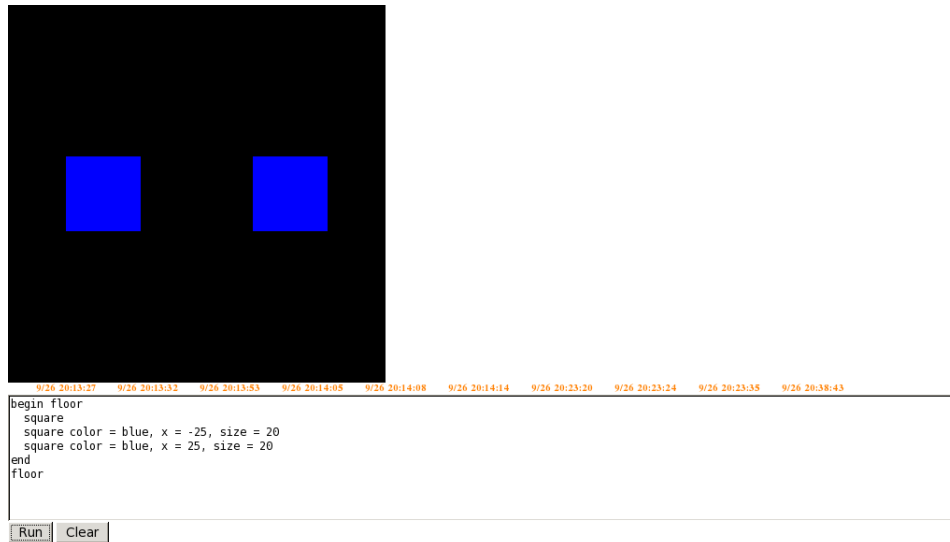


Figure 1: The floor shape

We’re going to add a “building” shape that draws a building by drawing several floors. To do this, erase the final line that says “floor” and instead add:

```

begin building
  [fill in this spot!]
end
building

```

Your program should now read:

```
begin floor
  square
  square color = blue, x = -25, size = 20
  square color = blue, x = 25, size = 20
end
begin building
  [whatever you filled in]
end
building
```

The first line says that we'll be defining the "building" shape. The last line says to draw the shape, "building", that we've just defined. The "[fill in this spot!]" has to be filled in by you:

Exercise 1

Add lines after "begin building" and before "end" which draw several floors on top of each other.

Hint: First, draw a floor with size less than 100 so there's room to draw several. Then notice that all floors should have the same x coordinate but different y coordinates — that is, they sit on top of each other.

The result should look something like Figure 2

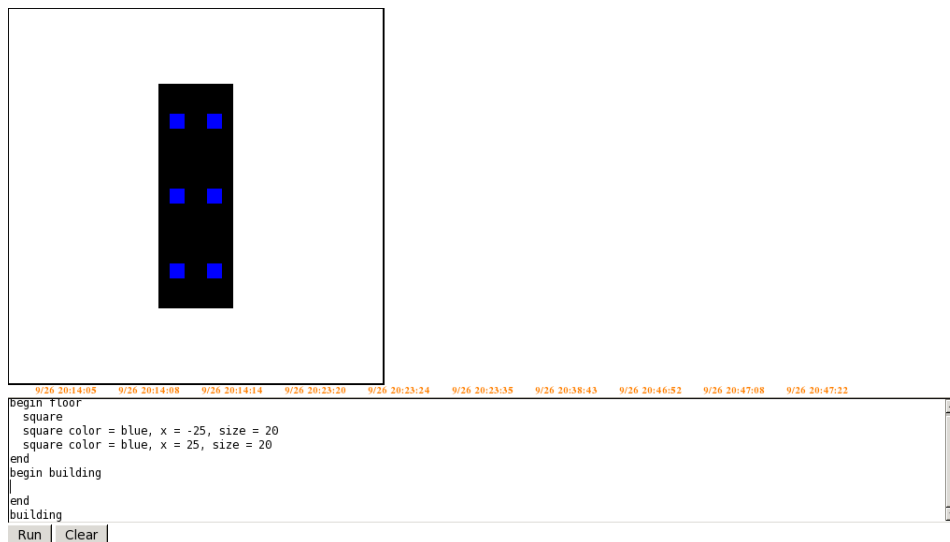


Figure 2: The building shape

Notice what we did. First we made an abstraction for a floor: a floor is just three squares. Then we made an abstraction for a building: a building is just

several floors. Note that we used our first abstraction in defining the second one; we could just as well have said that a building is a whole bunch of squares, but using abstraction was easier, more natural, and, as we'll see, will save us effort in the long run.

Now that we have buildings, let's build a city. What's a city? It's just another abstraction meaning "a bunch of buildings"!

Exercise 2

Add a few more "building" lines to the code you just entered to draw a city made of several different-colored buildings.

Now we have a *beautiful* city made of many colorful buildings. But square windows are so yesterday — circular windows are in today! Let's change them.

Exercise 3

Change all the windows in the city by changing the definition of floor to use circles for the windows instead of squares.

Notice how abstraction allowed you to make a giant change — changing every window in the city — by making very small changes to just two lines in your program. This would not have been possible if we shunned abstraction and just made our building out of tons of squares!

2 Peeking Under the Hood

In class, we saw a few properties of shapes. In particular, we've looked at:

- x
- y
- color
- size

One of the fun things about peeking under the hood of a program is that you can discover it does things nobody told you about.

Exercise 4

Go to the drawing canvas web page. Go to "View Source" in your web browser and find where all the different properties of the shapes are listed. Whoa — there are two new properties I didn't mention! Experiment with these new properties and use one in a drawing.

Once you've got the hang of this "View Source" thing, you can start peeking at the insides of web sites you use every day. While you probably won't understand everything you see on every web site — *I* certainly don't! — you

can still learn a lot about how they work, and maybe discover features they had that you didn't know existed.

Optional Extra Credit

Go to the New York Times web page at <http://www.nytimes.com>. Go to “View Source” in your web browser. Scroll down until you see some lines which begin with `<script>`. The last few lines of this section (before the `</head>`) have to do with choosing a newspaper-related preference (here written “Pref”). What preference does it look like it is? Looking at the top of the leftmost column of the page, what on the page do you think changes this preference?