**Triangle Solutions Design Document**

**Tuesday 11/14/17**

First talked to myself about coding the triangle solution thing from scratch in Java.

**Tuesday 11/28/17**

Did some work on the java Triangle bit. I have the data types captured.

**Friday 12/01/17**

I’ve managed to screw up the triangle solution to the point where the sides are now not getting assigned correctly.

I think I’m getting the sides in the correct position internally, but messing up the assignments.

The internal angle assignments aren’t right, but then I’m messing up the outputs, which is making them correct.

**Sunday 12/03/17**

Triangle is now copying to the correct values, but the angles are still in the wrong places. I think the point is that the parameters on the lawOfCosines() thing is messed up.

When I went through it again carefully, and carefully looking at the example solution, it all comes out right.

<http://www.calculator.net/triangle-calculator.html?vc=90&vx=4&vy=3&va=&vz=&vb=&angleunits=d&x=49&y=24>

**Tuesday 01/23/18**

Made some progress on Triangle. I still need to add validCount and an accessor to Sides and pull the for loops out of findSolution(). Done.

**Wednesday 01/24/18**

* See if you can get the Triangle thing to a state where you can send it out.

**Wednesday 03/14/18**

Started more work on Triangle…

* Presently, I have getNext/Prev ID stuff in Triangle rather than as parts of Sides and Angles. It would conceptually make more sense for these to go there. Even better…they should be in the base class of Sides and Angles.
* findSolution() needs for loops to find included sides or included angles.
* What would the base class name be?  
  They are triangle data sets.  
  TriangleData
* What would the common functionality be?
* It would then make more sense to identify angles/sides as a, b, c (or even 0, 1, 2) through the base class
* What maintains the idea of what’s next to what?  
  Uppercase are angles  
  Lowercase are sides  
  (...c) A b C a B c (A...)

**Saturday 04/28/18**

Finally got some work done on Triangle. Wrote the TriangleData base class.

This is where source control would come in handy.

At the level of Triangle.java, there’s nothing to say whether an item ID is for a side or an angle.

**Sunday 04/29/18**

In Triangle.java, I have getNext/PrevSide/AngleID. Since the ID’s are no longer side or angle dependent, they can go in TriangleData.

Maybe it’s not that simple. These functions take a side and return an angle and vs versa.

I don’t know where the output is going during debug.

**Monday 04/30/18**

There’s separate sub-tabs under Output in the NetBeans IDE where the printf stuff is going.

The triangle solution thing appears to have survived the class/sub-class conversion.

Triangle to do…

* findSolution() needs for loops to find included sides or included angles.
* There’s an issue in ASA() and AAS() where the B side is wrong.

**Thursday 05/03/18**

Triangle to do…

* findSolution() needs for loops to find included sides or included angles.
* There’s an issue in ASA() and AAS() where the B side is wrong.  
  It appears that it may have *always* been wrong.  
  done

**Tuesday 05/08/18**

Started complaining about the need for a separate design doc versus burying the details in my job search.

**Wednesday 05/09/18**

Created the Triangle Solutions Design doc.

**Tuesday 05/22/18**

Finally got around to adding data search functions.

It would appear that I still have confusion on traversal direction in getNext/PrevDataID.

The stuff still seems to work.

To do…

* Implement findDataASA()  
  done
* Implement findDataAAS()  
  done
* Implement findDataSAS()  
  done
* Finish SAS() call sequence in findSolution()  
  done
* Call findSolution() instead of hard coded stuff in setFromCmdLine()  
  done with useFindSolution flag.

I’m still confused by interface consistency on ASA and SAS…which should specify the included Side or Angle.

**Wednesday 05/23/18**

Finished the list I made yesterday. Now I need to test it all.

It works on SSS. I need to change the command line data to do the other stuff.

**Thursday 05/24/18**

To do…

* Test findSolution()
* Come up with a way to run a bunch of different data sets instead of just loading stuff from the command line.
* There was a thought to read the data from an XML formatted file, since Java has a built in parser.
* Get started on a GUI

Rearranged the code to move crap out of the command line capture function that is actually unrelated test code.

**Thursday 05/31/18**

Added setAll() and made hacky calls to it inside main().

AAS() is having issues. It doesn’t think there are two angles.

Only one angle and one side provided. No solution possible.

That implies that it’s in the SAS() section…which means that something in AAS() failed.

**Friday 06/01/18**

The site I’ve been using as a reference seems to have some confusion as to the direction the traverse is made. The result graphic goes the opposite direction from the data entry graphic.

http://www.calculator.net/triangle-calculator.html?vc=&vx=&vy=31&va=37&vz=&vb=54&angleunits=d&x=72&y=18

In findDataAAS() it seems the problem is in getting the ID for the correct side in the if statement. If I have angles A & B, I should look for side B (or A).

I guess what I need is the side opposite the first angle rather than the next from the second angle.

Now it appears that we *don’t* have all three angles. That’s because I assigned the angle result in AAS() incorrectly.

Now it kinda appears to work.

To do…

* There was a thought to read the data from an XML formatted file, since Java has a built in parser.
* Do we want to get the area of the triangle while we’re at it?
* Get started on a GUI

**Sunday 06/10/18**

Got started on some GUI stuff using a tutorial site…

<https://www.codeproject.com/Articles/33536/An-Introduction-to-Java-GUI-Programming>

The code is now in an incomplete and confused state until I figure out what all the panes and panels are supposed to do. I think panes and panels are interchangeable.

**Saturday 06/23/18**

Yeah…where was I?

Now I’m showing a window that doesn’t contain anything.

The example that worked had the JButton added to the (base)JPanel that was added to the Container.

Now I have a calcPanel. Adding it to the basePanel doesn’t help.

Set the layout, location and size of the calcPanel that the button goes on. Now something shows up…in a weird place.

The window is 486 x 493. I had specified 500 x 500. The button is 100 x 30 as specified.

The location is 236, 231 relative to the window. 236 x 200 relative to the client area. So, it seems to be located relative to the client area.

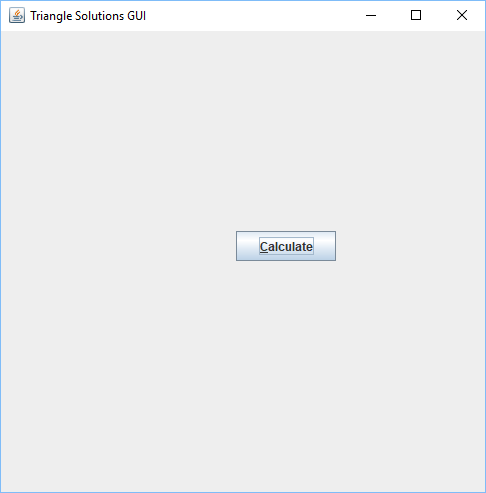
With a 300 x 300 window, the client area is 286 x 263. Total 286 x 292. Suggesting that the title bar is 29 or 30 pixels.

X is under by 14 pixels

Y is under by 7 plus titlebar pixels.

With the fixups the total window size is now 300 x 330, with a client area size of 300 x 300. The button is still in the wrong place. 136 x 101.

It may be that my math is just wrong.



**Sunday 06/24/18**

The math was “right”, but the parameter order was wrong. Now it looks like it should.

To do…

* Find out how to get the screen dimensions
* Find out what gokPixels and tbHeight really represent.
* Find out how to do output fields

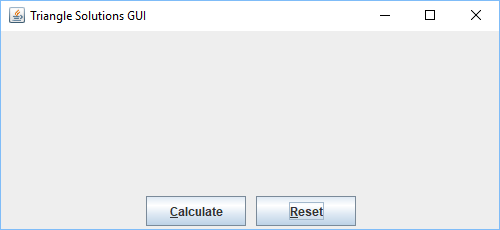
Ctrl-Shift-R turns on rectangular select.

There’s a pile of different things that are piling up in TriangleGUI that probably need to be in separate classes, like…

* calcButton
* dataPanel
* valueField
* valueFieldAngle
* valueFieldSide
* graphicPanel

Did a bunch of fancy stuff to make a button panel and add a reset button. Now I don’t show the right sized anything.

Got it to work…



The example code I’ve been working from doesn’t include input fields.

Got the input field stuff started, but I’m not seeing anything work yet.

**Thursday 06/28/18**

There was a bit where I wanted to return two different objects from a function (JFormattedTextField and JLabel), but Java doesn’t let you pass by reference and there’s no such thing as a pointer. What I need to do instead is create a structure(ish) thing.

**Friday 07/06/18**

I need to find an input field example that works.

To do…

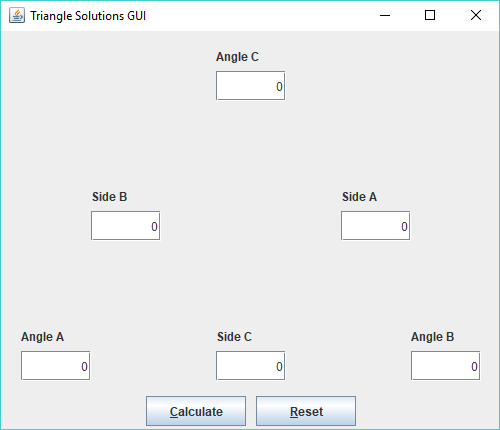
* Find out how to get the screen dimensions
* Find out what gokPixels and tbHeight really represent.
* Find out how to do input and output fields
* Find out how to format fields to only allow doubles
* Find out how to capture field values
* Find out how to populate field values
* Is there a way to handle the fields as either one array of 6 elements, or 2 arrays for sides and angles?

**Saturday 07/07/18**

Discovered that I hadn’t added the field and label to the dataPanel.

Now it’s showing up.

And now I have all 6 fields.



I think I need PropertyChangeListener

**Sunday 07/08/18**

I’m seeing *some* example code, but nothing yet that I think is complete and explicit enough to get all the objects and interactions down.

There are form examples that I probably put in C:\Development\Java, but it doesn’t seem that the forms try to do anything with the data that goes in the fields.

See what’s around 11/30/17. My notes say I installed the NetBeans IDE, but not anything about the examples.

C:\Program Files\NetBeans 8.2

It would appear that it came from the NetBeans IDE.

Got an OpenGL thing that I don’t need today…

C:\Downloads\Development\Java\1246294100841\_netbeans-opengl-pack\_0.5.5.zip

I’m looking at…

<https://netbeans.org/kb/docs/java/gui-functionality.html>

It appears that I might not *need* a listener after all. I might be able to harvest values from the fields at the point that the Calculate button is handled.

I’m still thinking that I need a class for the fields with…

* xPos
* yPos
* Name string

I was thinking that I also needed to hold the listener stuff, but that might not be necessary.

I’ve got something that makes the clear button work.

I’m thinking I need to host the Triangle object in the UI code…maybe. Or maybe I need to pass it to the UI’s constructor.

I think I’ve got something that pulls from the form. Now I need to repopulate.

It seems to properly pull stuff from the fields, get the solution and repopulate the fields. I think the problem now is that I need to format the angle data to ignore tiny floating point drift.

**Monday 07/09/18**

Started poking at Git

C:\Downloads\Development\Git\Git-2.18.0-64-bit.exe

<https://git-scm.com/book/en/v2>

Also git extensions. Should install first.

<https://www.youtube.com/watch?v=zMz9IZjUBFM>

It would appear that you don’t need to install Git itself.

GitExtensions-2.51.04.msi (Norton doesn’t like this.)

GitExtensions-2.51.03.msi (Norton doesn’t mind this one.)

Tried setting up an account on GitHub. You have to pay $7/month for private repos.

TriangleJava

Tried installing and running Git Extensions. It appears that you really *do* need to have Git installed.

Installed Git-2.18.0-64-bit.exe. It would appear that the git extensions version referenced in the install tutorial I found, also included git. The more current version apparently doesn’t.

Apparently KDiff3 wasn’t included either.

Back to triangle…

Added a rounding function roundDouble() which makes the results readable.

It fails to fail if the two shortest sides are shorter than the longest side. In that case lawOfCosines() returns nan. Can you even *test* for that? Double.isNaN()

To do…

* Is there a way to automatically highlight/select text when a field is clicked on or tabbed to?  
  selectAll()  
  WHEN\_FOCUSED  
  registerKeyboardAction
* Create field class to simplify code.
* Find a way to use indexes or iterators on arrays to get through the fields and stuff to simplify the code.
* Find out how to get the screen dimensions
* Find out what gokPixels and tbHeight really represent.
* Find out how to get the window dimensions.
* Implement redraw/resize functions
* Implement drawing of original equilateral triangle and result triangle.
* Move NUM\_DECIMAL\_PLACES and FACTOR calc out of roundDouble() so it is only calculated once.  
  done
* Add more detailed error condition tests and diagnostic messages to SSS()
* It would probably make sense to use NumberFormat and .setMaximumFractionDigits() as a better method than the roundDouble() thing.  
  getNumberInstance

**Tuesday 07/10/18**

Made another commit.

C:\Development\SSHKeys\private.ppk

Created an RSA private/public key when the SSH thing didn’t want to work. I’ve now got code up on GitHub, but I still don’t really know how to do a push in git Extensions.

<https://github.com/login>

There was something with refs/for that I needed to understand.

<https://docs.oracle.com/javase/tutorial/displayCode.html?code=https://docs.oracle.com/javase/tutorial/uiswing/examples/components/FormatterFactoryDemoProject/src/components/FormatterFactoryDemo.java>

Note the main() function in the above example. See what’s up with the invokeLater, Runnable, run bit.

The formatter stuff doesn’t work yet.

**Wednesday 07/11/18**

It would appear that the refs/for/ prefix that tells Gerrit that a code review should be created. It’s only part of the tool chain if Gerrit is involved…which it isn’t.

Get the formatter to do what it is supposed to do.

After you tab out of a field, the format gets adjusted correctly. It's messed up (not formatted) after you hit the calc button. It's not just the long fractional stuff. Whole numbers show up as n.0 instead of n.00.

Is there something that needs to happen in field.setText() in actionPerformed()?

I was trying to modify FormatterFactoryDemo.java so the text in the fields would be selected when the field got the focus, but feel into a hole with registerKeyboardAction() which is deprecated.

**Saturday 07/14/18**

Came up with bounding box co-ordinates for the graphics window, but right now they live inside createDataPanel(). I’ll need them outside that function. I could move all that positioning code to a more public place, or I could just have a function to set those values that’s called in createDataPanel().

**Sunday 07/15/18**

It appears that you need to subclass off of Component for a painter object.

It also appears that repaint() doesn’t call paint().

**Monday 07/16/18**

Repaint() isn’t anywhere in the paint() foodchain. It may just be something that says something needs to eventually be repainted.

There’s some Composite stuff that I’m clueless on that I added to my paint(). It doesn’t matter, because paint() still never gets called.

The weather example extends JApplet and implements ChangeListener.

When you slide the slider, there’s a stateChanged() event which sets the temperature and calls repaint.

Right now TriangleGUI is extending JFrame implementing ActionListener. I don’t think I’m currently really doing anything with the listener part. Correction…I’m using ActionListener to call actionPerformed().

**Tuesday 07/19/18**

JavaFX

package **javafx.scene.shape**

<https://www.tutorialspoint.com/javafx/javafx_2d_shapes.htm>

**Friday 07/20/18**

Trying to get paint() to work…

<http://www.oracle.com/technetwork/java/painting-140037.html>

I have mPainter added to dataPanel in createDataPanel(). Is there something else missing?

Am I missing a Container?

JApplet is down the foodchain from Container. Panel (not JPanel) is up the chain from Container in the JApplet heirarchy.

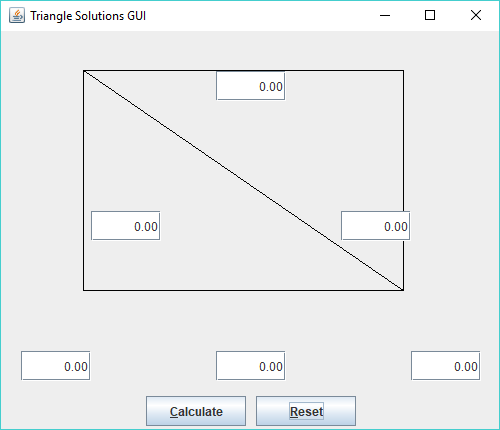
I put in an explicit call to paint(). It gets into paint(), but it blows up when I try to call drawLine().

TriangleGUI->JFrame->Frame->Window->Container->Component

JFrame is also in the Container food chain.

Once I dispensed with the separate painter object and just implemented paint() in TriangleGUI, I managed to get paint() to get called after a repaint().

It also seems to clobber my labels.



The top of the box needs to move down. The height seems right.

Perhaps it’s because it’s relative to the whole app frame rather than the dataPanel.

Maybe paint() needs to be relative to the mGraphicPanel that I’m not yet using.

**Saturday 07/21/18**

To do…

* Implement drawing of original equilateral triangle and result triangle.
* Create field class to simplify code.
* Find a way to use indexes or iterators on arrays to get through the fields and stuff to simplify the code.
* Implement redraw/resize functions
* Add more detailed error condition tests and diagnostic messages to SSS()
* Find out what gokPixels and tbHeight really represent.
* See if having a separate mGraphicPanel that hosts the graphics makes sense.  
  Would that be by extending JPanel and overriding paint()?  
  done
* Is there a way to automatically highlight/select text when a field is clicked on or tabbed to?  
  selectAll()  
  WHEN\_FOCUSED  
  registerKeyboardAction
* Find out how to get the screen dimensions  
  done
* Find out how to get the window dimensions.  
  getSize()
* Move NUM\_DECIMAL\_PLACES and FACTOR calc out of roundDouble() so it is only calculated once.  
  done
* It would probably make sense to use NumberFormat and .setMaximumFractionDigits() as a better method than the roundDouble() thing.  
  getNumberInstance

Set up the mGraphicPanel. It may be pretty much the way I had the painter before. Now I’m harvesting the width and height from the parent object member data.

The origin seems to be right where I want it. The size has issues. Width = 410, height = 290.

Got the size issues resolved.

**Sunday 07/22/18**

I had some ideas for how to scale the triangle graphic into the available space…but those ideas kept me from getting to sleep.

* Calculate the angle of the GraphicsPanel bounding box diagonal (bbDiagAngle).
* Find the longest side.
* If that side is Side C (the base) then that side will be the width of the bounding box.
  + Verticies at Angle A and B will be at the lower corners of the bounding box
  + Scale the remaining sides as a percentage of that length.
  + Use AAS (implied right triangle) to calculate the position of the vertex at Angle C
* If the longest side is A, the first vertex will be at Angle B at the lower right corner of the bounding box. Otherwise (Side B) first vertex is at Angle A in the lower left corner.
  + The long side will intersect the bounding box…somewhere…which will be the location of Angle C.
  + If the angle is greater than bbDiagAngle, the side will intersect the top of the bounding box. Otherwise, it will intersect the opposite side.
  + Use ASA to get the length of the line that intersects the bounding box.
  + Scale the remaining sides as a percentage of that length.
  + The remaining vertex is the scaled length of Side C (the base) away from the first angle along the bottom of the bounding box.
* You could do a centering last step along the X axis. Centering along the Y axis would tend to interfere with the Side A/B data entry fields.

Started fixing stuff that doesn’t need to be fixed yet just because I understand them now.

It finally kinda looks like it might actually look.

Apparently, there’s not a resize() that’s automatically called when stuff is resized. You need a ComponentListener and a componentResized() function.

**Monday 07/23/18**

To do…

* Implement drawing of original equilateral triangle and result triangle.
* Your use of private/protected/public is pretty scatter shot.
* Create field class to simplify code.
* Find a way to use indexes or iterators on arrays to get through the fields and stuff to simplify the code.
* Implement redraw/resize functions
* Add more detailed error condition tests and diagnostic messages to SSS()
* Find out what gokPixels and tbHeight really represent.
* Do we still need screenWidth/Height in member data?  
  done
* frameX/Y should be mFrame.  
  done

Pushed the code I did yesterday.

I was thinking I could use SAS() and AAS() to get vertex information, but they harvest and modify the triangle data for the triangle we want to represent. You’ll need to use lawOfSines() directly.

**Tuesday 07/24/18**

Putzed with the code to put the default triangle in getVerticies()

The enum setup is a pain in the ass for for loops.

There *is* a way to iterate over the set of enums by using .values() to return an array of the values.

DATA\_INVALID is used to initialize a DataID.

I’m already using the construct in TriangleData.print().

**Wednesday 07/25/18**

To do…

* Change getVerticies() to calcVerticies().  
  done
* Implement drawing of result triangle.
* Create field class to simplify code.
* Implement redraw/resize functions
* Add more detailed error condition tests and diagnostic messages to SSS()
* Find out what gokPixels and tbHeight really represent.
* Find a way to use indexes or iterators on arrays to get through the fields and stuff to simplify the code.  
  May not need to do this.

**Wednesday 08/01/18**

Cobbled up code to find vertex C. It’s not right yet.

With height and width of 320

sideAPxlLen = 554.25

sideCPxlLen = 640

They should be something like half that.

**Thursday 08/02/18**

sideAPxlLen = 554.25

sideCPxlLen = 184.752

I needed to draw the thing out on paper to get it to work so I could see what’s going on…

sideAPxlLen = 160

sideCPxlLen = 277.128

Finally got the first case to work.

Checked it in.

**Friday 08/03/18**

Got started on the second half of the scaling routine.

**Saturday 08/04/18**

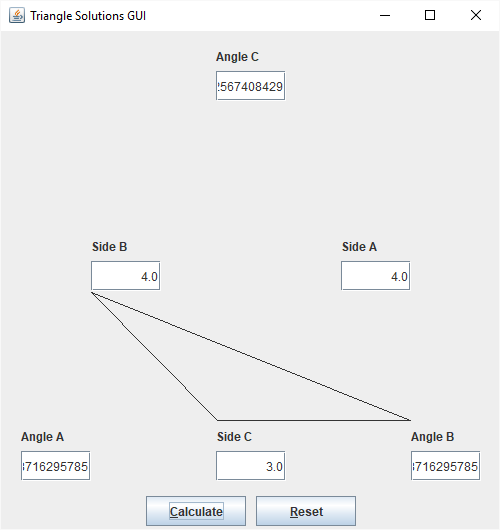
I got to thinking that a lot more general scaling algorithm would involve…

* Rendering the verticies in terms of the real number lengths as they were input and/or calculated.
* From vertex/angle A, build a right triangle to vertex/angle C. The non-hypotenuse sides are the XY vertex co-ordinates.
* Since we assume that the base is always along the X axis, vertex/angle B is just the length of side C along the X axis.
* Determine what the top, bottom, left, right extents are.
* Scale the real co-ordinates to the pixel co-ordinates

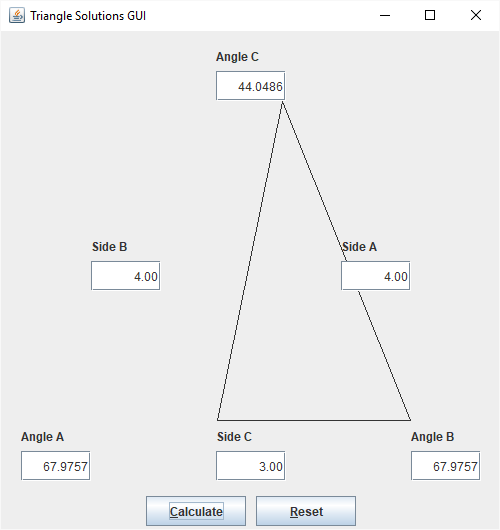
Note that that doesn’t account for a non-square bounding box. If it isn’t square, the triangle’s aspect ratio will be distorted.

**Sunday 08/05/18**

With A=4, B=4, C=3, I’m coming up with numbers that sound plausible, but vertex C is ending up on the left side instead of the top…



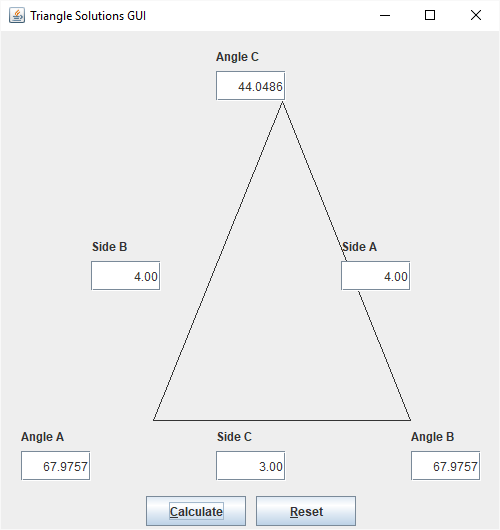
Now it’s better, but still no cigar…



**Monday 08/06/18**

I’m letting this thing scare me again.

I found a problem and fixed it. I think it revealed a problem in the first block of logic though.



A=50, B=26, C=26

Alpha=148

Beta=16

Gamma=16

The pxlSideLen is wrong for starters.

Checked in.

Messed with the parameters at pxlSideLen and got it to come up with 307.7. It should be more than 320. Now it’s 332.8

Things are starting to work. However…

A=33, B=50, C=33

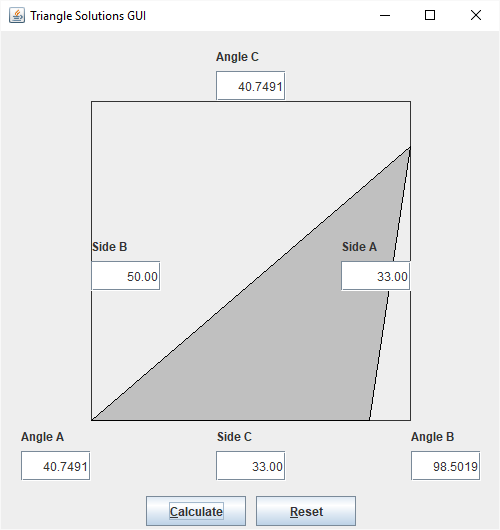
…is off in Oz.

Got that fixed.

SideA = 5, SideC = 4, AngleB = 90 gives a spurious error.

SideB = 5, SideC = 4, AngleA = 90 works.

Added a filled polygon…



To do…

* Create field class to simplify code.
* Implement redraw/resize functions
* Add more detailed error condition tests and diagnostic messages to SSS()
* Find out what gokPixels and tbHeight really represent.
* Come up with a center() function to reposition the vertices to center the triangle within the bounding box.
* Fix the right triangle starting at Angle B problem. It says No angles provided.
* See if there’s a way to collapse some of the code in calcVerticies().
* Try recoding calcVerticies() using the approach described on 08/01/18

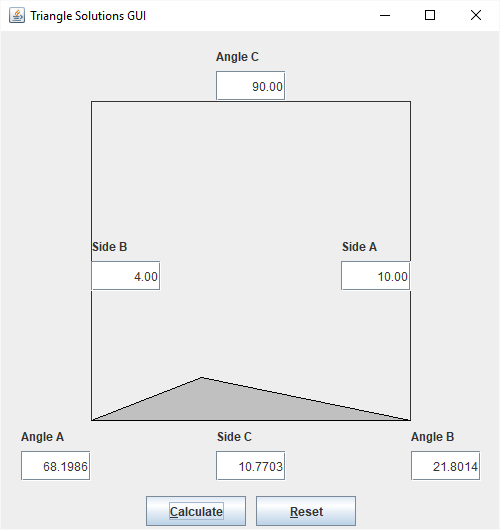
With 90 in Angle B and no other angles, findSolution() is getting 0 for knownAngles. The dataArray is all zeros.

**Tuesday 08/07/18**

It turns out that the code that harvests angles from input fields in actionPerformed() has been wrong for some time now. I was pulling from the AngleA field for all three angles.

Ok…this is kinda effed up…

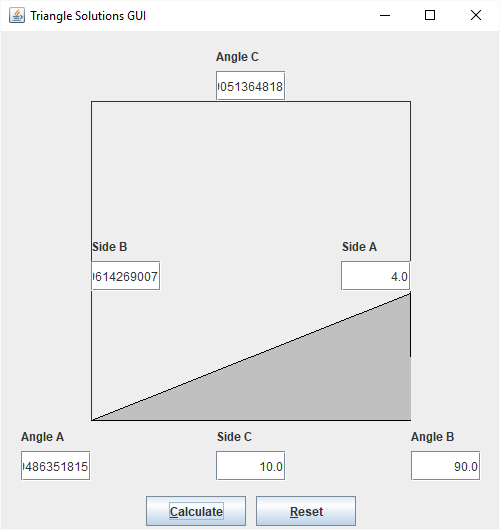
SideB = 4, AngleC = 90, SideA = 10…



That doesn’t look like a right triangle.

The numbers are all right though.

There’s also this bit that reveals that I need to pay attention to the -1 thing…



To do…

* Revisit where vertex C is getting located in the longest side C code.
* Find out where the missing -1 is at vertex B (X).
* Set up input fields as arrays  
  done a/o 06/27/18
* Create field class to simplify code.  
  done a/o 06/27/18
* Implement redraw/resize functions
* Add more detailed error condition tests and diagnostic messages to SSS()
* Come up with a center() function to reposition the vertices to center the triangle within the bounding box.
* See if there’s a way to collapse some of the code in calcVerticies().
* Try recoding calcVerticies() using the approach described on 08/01/18

Where are we now?

We produce good numeric results. Some of the graphic results have issues. Some of the implementation is clearly clumsy.

It may be a good time to back away from this…for today anyway.

**Saturday 08/25/18**

What will the field class/container look like? That was in a comment in TriangleData.java…

// &&& Need a class to contain...

// dataField

// label

// labelText

// xPos

// yPos

// angle/side

// ...and set up as arrays for sides and angles

Discovered Javadoc format and block tags…then I let it distract me from getting real work done.

I got the DataField nested class done and the arrays set up. Now I have an issue with assigning the output of the DataField constructor to the field array.

I checked the code in with the error in place.

**Monday 08/27/18**

I’m thinking that the problem I’m having is either a missing copy constructor or assignment operator…*or*…I haven’t defined the size of the array.

Added = new DataField[3] to the mSide/AngleField declaration. It appears that this doesn’t call an object constructor to create an object for each array element. It just creates null references. That’s what I needed to do to get the array assignments in createDataPanel() work.

Now it would appear that I’m not getting stuff from the fields. I’ve specified all the sides and it says there are none.

There’s an error with specifying 30/60/90 and one side that says you only specified one angle.

To do…

* Figure out the correct way to format the results so they display the correct number of decimal places at all times.
* Find out how to highlight the field contents when the field is selected so you don’t have to explicitly highlight to select the text yourself.
* Come up with a center() function to reposition the vertices to center the triangle within the bounding box.
* Implement redraw/resize functions
* Try recoding calcVerticies() using the approach described on 08/01/18
* Revisit where vertex C is getting located in the longest side C code.
* Find out where the missing -1 is at vertex B (X).  
  SidA = 4  
  AngB = 90  
  SidC = 10
* Add more detailed error condition tests and diagnostic messages to SSS()
* See if there’s a way to collapse some of the code in calcVerticies().
* There was at least one triangle rendering error. Where was it?  
  SidB = 4  
  AngC = 90  
  SidA = 10  
  Fixed it.
* Come up with a way to read test data from a file and iterate through the data sets.  
  This may also include expected results.
* Is there a way to get the error and/or status messages from the solution code in Triangle.java to display in the GUI?
* Show a little box if the angle is a right angle.

AngC = 30  
AngB = 60  
AngA = 90  
SidC = 4

That problem was due to a logic error. There are probably many more.

**Tuesday 08/28/18**

I thought I knew what the problem was with…  
SidA = 4  
AngB = 90  
SidC = 10

I was expecting that the width of the longest side (C), needed to be mWidth - 1, but that was already the case. The problem might instead be a rounding error at vertex (C).

Where was I on formatting the input fields? It looks like I gave up on 07/11/18.

Then I ran into the web page that showed how to handle resizing.

<https://www.javacodex.com/More-Examples/2/7>

Look at setSize for each component.

**Saturday 09/01/18**

I’m having trouble with setSize() and resize(). Both of these are already methods in the class that want to take x and y parameters. Resize() also happens to be deprecated.

I had a resize() method in the graphics panel that didn’t take parameters. The problem I’m having is doing something that makes sense and doesn’t break existing functionality. Part of the problem is that I can even really describe what the problem is.

I guess part of the problem is that both of these assume that you know what the size is supposed to be. What *I’m* trying to do is determine what that size *is*. Perhaps I’m looking for calcSize().

**Sunday 09/02/18**

I’m thinkin’ it would make sense to subclass all the panels that need to have calcSize() since they also need to carry around member data for the UI sizes and positions.

* JPanel mBasePanel  
  May not need calcSize() for this since the contained panels take care of their UI size/positions.
* JPanel mButtonPanel
* JPanel mDataPanel

It may make sense to also do a button class like the DataField class.

I keep running into chicken and egg problems.

The first pass of code done with the DataField class and calcLocations() worked well to relocate the data fields. The graphicsPanel seems to be of fixed size and position though. The button panel doesn’t move, but that’s expected since I haven’t touched it yet.

Look at mFrameHeight/Width south of componentResized.

mGraphicsPanel.calcSize() calls it’s own getSize().

Found the problem.