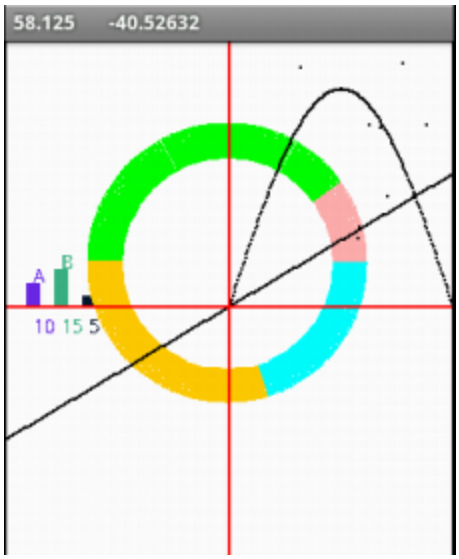
Canvas Graph - Graphing with App Inventor 2 Canvas...an expert tutorial

This tutorial demonstrates how to use the App Inventor 2 Canvas control to graph data. A developer can use the graphing capability of the Canvas to plot data in different ways. The aia code provides the basis for graphing equations with user input of variables and real time graphing. *Canvas Graph* is an Expert level tutorial, it mainly describes what, not how and requires downloading the aia source.

Canvas Graph does the following:

1. Scales World coordinates to Screen coordinates and draws a graph.
2. Creates line graphs, scatter graphs, histograms, pie charts and donut charts.
3. Draws functions.
4. Draws Real Time graphs.



The image above shows on a single display some of the graphing possibilities of *Canvas Graph*. The tutorial shows how to build this stuff.

Features:

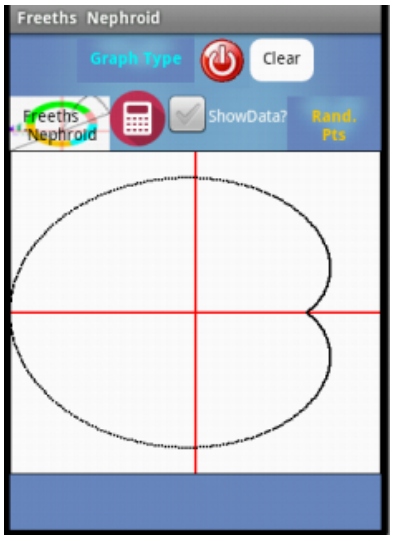
* Histograms are labeled as to item, value. The graphs can be color coded.
* Line graphs are scaled from World coordinates. The AI Canvas can graph individual points or graph functions/equations.
* Real time graphic is possible.

Limitations: Most functions in *Canvas Graph* are hard coded. The tutorial is for demonstration purposes. Developers should use TextBoxes to provide real time manual input of data points and function parameters or csv files to populate graphs to observe trends.

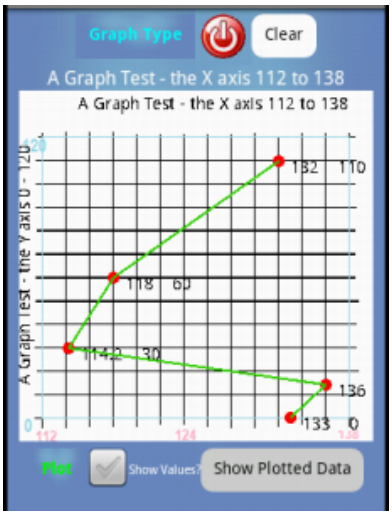
A developer can make simple, or complicated graphs from a data set or graph in real time using the Canvas component. What can you graph? Single points, scatter plots, histograms (with annotations for items and the actual values), line graphs (graph a function or a series of points and connect them). You can make charts. Pie charts, histograms, donut charts and others types of graph are possible.

Some charts and graphs can easily be made without respect to any special preparations regarding the plotting axis. However, when you plot World coordinates on the Canvas as Screen coordinates or to retrieve values from the screen in World units, manipulation of the axis and its values are needed.

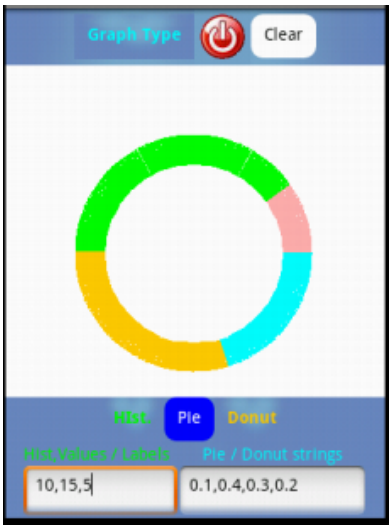
Canvas Graph is not about how to create a universal graphic component from AI2 blocks. A universal graphic app is beyond the scope of this project. The Canvas can be used to graph; handy when your tablet does not have a WIFI connection to access outside spreadsheets or tables.



Graphing a Function



An Annotated Scaled Graph



A Donut Graph

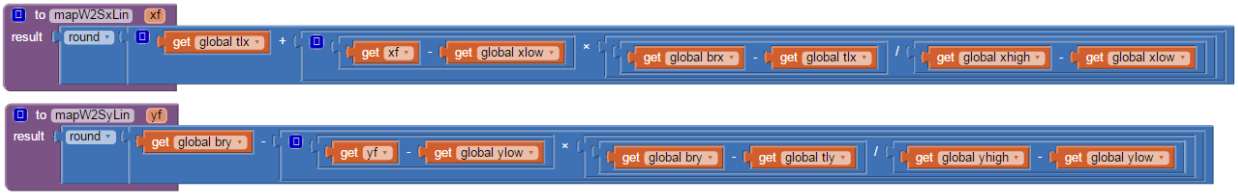
The original code for the Pie graph is by Scott Ferguson; (his algorithm is modified to produce a Donut Graph). Visit Scott's Web page for more ideas as how to use a Canvas https://groups.google.com/forum/#!categories/app-inventor-developers-library or https://groups.google.com/forum/#!forum/app-inventor-developers-library .

Automatic Scaling?

Screen coordinates is the coordinate system used on the Canvas; the top right x,y being 0,0 with x increasing to the right and y increasing downward. World coordinates are whatever

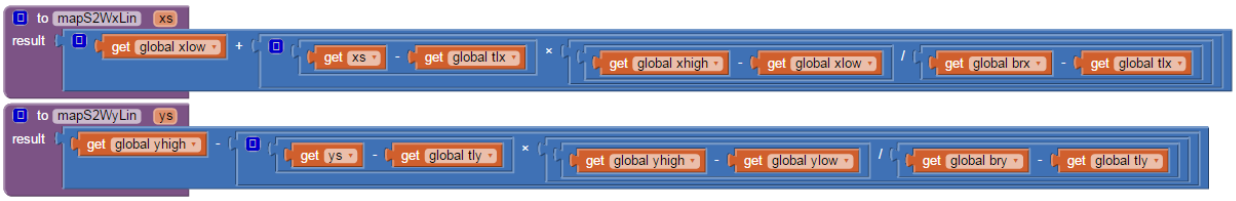
units (and the range you want displayed on the graph). Four procedures in *Canvas Graph* convert between World and screen units. The procedures assist developers scale their graphs and charts on the Canvas control. Two procedures convert World coordinates to the x and y values on the screen. Two other procedures convert the screen's coordinates to World coordinates. The second procedures are useful to extract information from a Canvas graph; touch a position of the screen a find out the value of the graphed point. The screen to World conversion is not precise. When you use the Canvas, you work with a scaling issue. The conversion of screen to World coordinates is a reasonable interpretation of the data usable in many instances where precision is not required. Yes, precision is lost when scaling.

Convert from World to Screen



These two procedures map World coordinates to Screen coordinates. Specialized variants of these routines are also used in the tutorial. The procedures *calibrate* the Canvas coordinates for your data set when used with the global variables show in the Math blocks.

Convert from Screen to World



These two procedures map Screen coordinates to World coordinates. Specialized variants of these routines are also used in the tutorial. Collectively, the procedures are used to get information about the plot directly from the Android screen by touching the plot.

Everything you need to know about graphing is in these four conversion procedures. Well, not quite everything. Knowing how to use these procedures greatly simplifies generic graphing using AI2's Canvas. These routines were derived and ported from *Delphi Object Pascal* code here: http://www.ibrtses.com/delphi/dmcs.html . The Delphi Windows compiler use the same Canvas scheme used on Android devices.

The algorithms in the Button event handlers of *Canvas Graph* can draw a line, draw a function, draw random points, construct an annotated histogram, make a pie chart and make a donut chart.

Graphing Examples

Pre-loaded Function Examples

Data

There is no data loader. Data values are presently instantiated with code. Provide your own data import routines using csv files or provide values and equations using TextBoxes to input the required values to make the concepts here work within your own app.

Graph Axes

Axes values for the x and y axes are computed using the data ranges provided by the user for the graphs. Axes are not labeled in the Function demonstration. You can figure out how to do the labeling (not hard) (or use the scheme used in the Annotated Line Graph example), leave the graph without axes labels or use a fixed background image in the Canvas to provide the scale (as demonstrated in the Real Time graphing and Special Histogram examples).

To make a fixed background scale, run the app without a background, note the max/min axis values for both axis. Using a *Paint* program on your PC, make a pretty background display, complete with axes values, a grid and a plot name. Make a scale grid on the image; add the plot name using blocks in a label above the Canvas. If your app always plots the same data range, a background image is very suitable.

What you do with the axes depends on what you want your app to display and how pretty a display is required.

The plotting screen of the *Canvas Graph* where the tutorial demonstrates how to plot functions is divided into four quadrants. This system allows plotting both negative and positive value x and y coordinates needed to graph many mathematical functions. The canvas you design using these techniques can display all four quadrants or just quadrants I and IV (all y values positive). No attempt was made to add flexibility; scaling is a complicated process and best handled specifically for the requirements of your app.

Plotting

Plotting using the Canvas is not lightning fast; not even race car fast on the emulator or live developing on the device using WIFI. The demos that plot functions generally plot over 300 points or more, pixel by pixel and are s l o w. The plotting is partly an issue of the cpu on most phones, the time required to make a calculation, AI's slow graphic routines, the intervention of Companion and a function of the number of items plotted and where the information comes from. However, build the apk and the performance of the plots still is not at lightning speed but the computation and rendering are significantly faster on the device. Smile, Canvas is a useful plotting tool.

Plotting using the Canvas frees the user from the Internet at the expense of possibly slightly slower plotting than using a Web service. Plots are made on an AI2 Canvas.

There are many published AI2 routines that demonstrate how to save the Canvas. Save a Canvas image and eMail it if the graph you plotted is needed elsewhere.

Real-Time Graphing

Is real-time graphing possible? The answer depends on the data refresh rate required. A limitation is AI2's event driven programming. A very simple example shows how Accelerometer data can be graphed. But, what do you do with the data? I don't know but if you are graphing it, you must have a purpose? And, in this instance the real-time graphing works very well. How does this work with Bluetooth data streams for, perhaps an Arduino? Try some blocks.

The following, from MIT's Accelerometer documentation, shows what the example plot records. *Non-visible component that can detect shaking and measure acceleration approximately in three dimensions using SI units (m/s^2). The components are:*

- *xAccel: 0 when the phone is at rest on a flat surface, positive when the phone is tilted to the right (i.e., its left side is raised), and negative when the phone is tilted to the left (i.e., its right size is raised).*
- *yAccel: 0 when the phone is at rest on a flat surface, positive when its bottom is raised, and negative when its top is raised.*
- *zAccel: Equal to -9.8 (earth's gravity in meters per second per second when the device is at rest parallel to the ground with the display facing up, 0 when perpendicular to the ground, and +9.8 when facing down. The value can also be affected by accelerating it with or against gravity.*

Note both the x and y plotted in the example are 0 at rest; however the z value is -9.8. None of the raw values are plotted, the values are scaled to fit on the plot and may be inappropriate for your purposes.

Visit Scott's web page (see above) for more elaborate examples of real time graphing.

When is the Tutorial going to show how to make the Blocks?

The aia source for *Canvas Graph* is available. Expert developers figure out what most of the blocks do by experimenting; this is an Expert tutorial and what you need to do.

Is there a source file? **Download Source Code**

Download the source code to your computer, then open App Inventor 2, click Projects, choose Import project (.aia) from my computer..., and select the source code you just downloaded. There is a single aia file: **CanvasGraph.aia** (the complete project). The link to the Google folder where the aia is stored is on the *App Inventor for Fun* tutorial blog page. The aia loads

slowly; there are lots of layouts and controls. Expect to see a white screen for almost a minute. Everything works in the emulator except the real time graphing example. Do you know why that *Canvas Graph* example does not work? If you do, you truly understand App Inventor programming.

Copyright 2015 SJG