**Displaying Pokemon Stats as a Hexagonal Graph or Bar Graph**

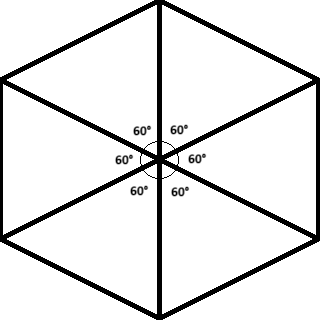
**By Fernando G. Berry**

The basic formula for determining the length of an element or distance of a point from center is:

Where is the base stat of a pokemon, is the maximum possible stat value, and is the maximum available distance of a given element or chart.

For a bar chart, this is all that is really needed to calculate as can just represent an element height or width.

A hexagonal chart is different as the distance represents a vector from the center of the chart to the maximal point that is from a reference angle.



The solution of this problem is to use the trigonometric functions of and to find the and of a given equaling .

&

We can then define the as the coordinate and apply an offset to re-zero the point of origin (PO) to the center of a drawing screen. A - offset will be applied to each angle since angles are being referenced to a cartesian coordinate system.

A function will be invoked to perform the required mathematics and return a pair of coordinates based off of angle, distance, and PO arguments. This function will ignore the and processing of angles. It will instead treat the distance as the coordinate. An inversion of the coordinate will be required for angles greater than . All angles provided as arguments are expected to be after the angle offset is provided.

The function is expected to behave as follows:

**Declare Function** getCoordinate( **number** angle, **number** distance, **array** offset )

**Declare array** coordinate and **assign** an **array** with indices 0 and 1 set to 0

**Check if number** angle > 180 degrees

**true: assign number** distance = negative **number** distance

**false:** continue

**Check if number** angle is 90 **or** 270 degrees

**true: assign array** coordinate index 1 = negative **number** distance

**false:**

**assign array** coordinate index 0 = **sin**(**number** angle) \* **number** distance

**assign array** coordinate index 1 = **cos**(**number** angle) \* **number** distance

**assign** **array** coordinate index 0 = **array** coordinate index 0 - **array** offset index 1

**assign** **array** coordinate index 1 = **array** coordinate index 1 - **array** offset index 1

**return array** coordinate

**End of Function**