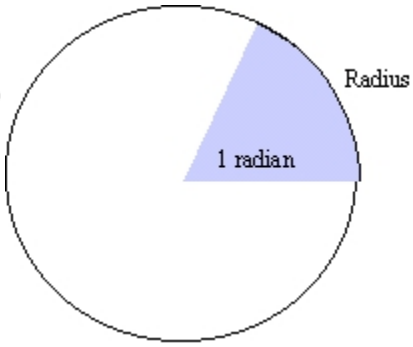
RADIANS AND DEGREES IN JAVASCRIPT

Radians and degrees (or angular measure)

When measuring **angles** we usually measure them in **degrees. There are 360 degrees in a circle.**I.e. a degree simply divides a circle into 360 segments.
An alternative radial measure is known as a radian. A radian is the angle formed by measuring the radius around the **circumference** of a circle as show by the diagram. **1 radian is approximately 57.32 degrees.**



From the formula for calculating the radius of a circle,

circumference = 2 * PI * radius it follows that there are 2 * PI radians in a circle.

> Spreadsheets (such as Excel) and all programming languages will generally use **radians** instead of **degrees** when dealing with the trigonometric ratios of **sine, cosine** and **tangent**.

To convert degrees to radians use the spreadsheet function **radians(degrees)** or **degrees(radians)** to convert back from radians to degrees.
Alternatively since there are 360 degrees in a circle and 2 * PI radians in a circle, it follows that

```
degrees /360 = radians / (2 * PI)  - two equal ratios

rearranging the formula we get
  degrees = radians * 360 / (2 * PI)
  which simplifies to
  degrees = radians * 180 / PI
alternatively
  radians = degrees * ( 2 * PI) / 360
  which simplifies to
  radians = degrees * PI / 180
```

Two functions can be created to assist with the calculations. They make use of the Math's objects built value of PI.

```
function deg2rad(degrees)
{
  //convert degrees to radians
  return Math.PI * degrees/180;
}

function rad2deg(radians)
{
  //convert radians to degrees
  return 180 * radians/Math.PI;
}
```

Conversion between Polar and Rectangular co-ordinates

This example is used to further illustrates the Javascript Math object is based on the conversion between rectangular and polar coordinates. A **vector** can be expressed either in terms of its magnitude (v) and angle (r) in the diagram. These are its **polar coordinates**. Alternatively it can be expressed in terms of its **rectangular coordinates** as a + jb. which is also known as a **complex number**. Don't worry about the j, for the purposes of this example, it simply denotes the vertical axis.

From the diagram it can be seen that

a = V ***** cos **r** and b **=** V ***** sin r

Conversely converting from **rectangular** to **polar** form gives
V = sqrt(a^2 + b^2) and r = $\tan^{-1}$ (b/a)

> **Note:** Most scientific calculators will incorporate these functions for direct conversion between co-ordinate systems. $\tan^{-1}$ is also known as the function atan() or arctangent Likewise $\sin^{-1}$ is asin() and $\cos^{-1}$ is the acos() function

The example also makes use of the functions to convert degrees to radians, and vica versa.

Polar to rectangular coordinates

```
function P2R()
{
   //convert Polar to rectangular coordinates
   tst = document.getElementById("polar").value;
   //find position of comma separator
   var idx = tst.indexOf(",") + 1;
   var r = parseFloat(tst.substring(idx));
   var v = parseFloat(tst);
   var msg = "<p>Polar coordinates " + v + " angle " + r;
   //convert degrees to radians
   //and rouond answer to 2 decimal places
   r = deg2rad(r);
   var a = v * Math.cos(r); a = Math.round(a*100)/100;
   var b = v * Math.sin(r); b = Math.round(b*100)/100;
   msg += " equal rectangular coordinates  " + a + " + j" + b;
   document.getElementById("reply").innerHTML = msg + "</p>";
}
function R2P()
{
  //convert Rectangular to polar coordinates
   tst = document.getElementById("rect").value;
   //find position of j separator
   var idx = tst.indexOf("j") + 1;
   var b = parseFloat(tst.substring(idx));
   var a = parseFloat(tst);
   var msg = "<p>Rectangular coordinates " + a + " + j" + b;
   var r = Math.atan(b/a);
   //convert to degrees
   r = rad2deg(r); r = Math.round(r*100)/100;
   var v = Math.sqrt(Math.pow(a,2) + Math.pow(b,2));
   v = Math.round(v*100)/100;
   msg += " equal polar coordinates " + v
   msg += " angle " + r + " degrees";
   document.getElementById("reply").innerHTML = msg + "</p>";
}
```

Source : http://www.soslug.org/node/1727