

King Richard's Perspective

As viewed from 1,000 smiles

Mapping the Moon was out of this world...



A Gore reveals an Inconvenient Truth!



Mac-P.C. commercials based on Furno-IT relations

Keegan cools him down

Dick is leaving! Our world is doomed.

Bacon Globular, your highness.

Keegan's recipe for "The Furno"

8 parts coffee

3 parts cream (you can't put in too much)

Two (2) little packets of sugar (removed the paper packets)

Then stir gently and deliver once in the morning at 10 a.m. and once in the afternoon at 4:45 p.m.

Dicktopia
Where all the people—and all the charts—are properly proportioned to each other...

Outtawhacksville
Where people and charts grow without regard to how others are growing! Gasp!

Move over, Oscar.
Dick sets a new standard when he wins the first What on Earth prize.

Who forgot to put the *&@! scale on this thing!

Dick finds soaking in saltwater soothing for the sole.

Dick's world... on a dime!
(dime not to scale)



Ouch!
De facto maps discarded — only Dejure maps allowed, thanks to Dick!

Printed Newspapers

The Washington Post

ISLAND OF MISFIT TECHNOLOGY

Rubyolith

Pen Plotters

Stat Camera

PCs

Bestine, worstine

Remember when line tape literally held the world together?

Many, many Macs

Ouch!

Dick's world... on a dime!



Ouch!

De facto maps discarded — only Dejure maps allowed, thanks to Dick!

Mapping the Earth for the Post was not always a breeze...

r_major = r_maj; r_minor = r_min; false_northing = false_north; false_easting = false_east; temp = r_minor / r_major; es = 1.0 - SQUARE(temp); e = Math.sqrt(es); center_lon = c_lon; center_lat = c_lat; sincos(lat1, sin_po, cos_po); con = sin_po * val; ms1 = msfz / e; sin_po; cos_po; ts1 = tsfz / e; lat1; sin_po; val; sincos(lat2, sin_po, cos_po); ms2 = msfz / e; sin_po; val; ts2 = tsfz / e; lat2; sin_po; val; Math.abs(lat1 - lat2) > EPSLN; ns = Math.log(ms1 / ms2) / Math.log(ts1 / ts2); } else { ns = con; } fo = ms1 / (ns * Math.pow(ts1, ns)); rh = r_major * Math.pow(ts1, ns); } rh = r_major * Math.cos(theta); } rh1 * Math.cos(theta) + false_northing; } else { con = lat * ns; if (con <= 0) { p_error("Point can not be projected", "lamcc-for"); return(44); } rh1 = 0; } theta = ns * adjust_lon(lon - center_lon); prjX = rh - rh1 * Math.sin(theta) + false_easting; prjY = rh - rh1 * Math.cos(theta) + false_northing; } else { fo = Math.pow(ts, ns); } else { con = lat * ns; if (con <= 0) { p_error("Point can not be projected", "lamcc-for"); return(44); } rh1 = 0; } theta = ns * adjust_lon(lon - center_lon); prjX = rh - rh1 * Math.sin(theta) + false_easting; prjY = rh - rh1 * Math.cos(theta) + false_northing; }